Semantic Search and Composition in Unstructured Peer-to-Peer Networks

Xiaoqi Cao
Department of Computer Science
Saarland University

A dissertation submitted towards the degree
Doctor of Engineering (Dr.-Ing.) of the
Faculties of Natural Sciences and Technology of
Saarland University

Saarbrücken, Germany 2015

Examination Board:

Dean: Prof. Dr. Markus Bläser
Chairman: Prof. Dr. Markus Bläser
Examiners: PD. Dr. Matthias Klusch
Prof. Dr. Jörg Hoffmann
Prof. Dr. Abraham Bernstein
Academic doctor member: Dr. Luca Mazzola

Date of the defence: 17. Aug. 2015

Abstract

This dissertation focuses on several research questions in the area of semantic search and composition in unstructured peer-to-peer (P2P) networks. Going beyond the state of the art, the proposed semantic-based search strategy S2P2P offers a novel path-suggestion based query routing mechanism, providing a reasonable tradeoff between search performance and network traffic overhead. In addition, the first semantic-based data replication scheme DSDR is proposed. It enables peers to use semantic information to select replica numbers and target peers to address predicted future demands. With DSDR, k-random search can achieve better precision and recall than it can with a near-optimal non-semantic replication strategy. Further, this thesis introduces a functional automatic semantic service composition method, SPSC. Distinctively, it enables peers to jointly compose complex workflows with high cumulative recall but low network traffic overhead, using heuristic-based bidirectional chaining and service memorization mechanisms. Its query branching method helps to handle dead-ends in a pruned search space. SPSC is proved to be sound and a lower bound of its completeness is given. Finally, this thesis presents iRep3D for semantic-index based 3D scene selection in P2P search. Its efficient retrieval scales to answer hybrid queries involving conceptual, functional and geometric aspects. iRep3D outperforms previous representative efforts in terms of search precision and efficiency.

Zusammenfassung

Diese Dissertation bearbeitet Forschungsfragen zur semantischen Suche und Komposition in unstrukturierten Peer-to-Peer-Netzen (P2P). Die semantische Suchstrategie S2P2P verwendet eine neuartige Methode zur Anfrageweiterleitung basierend auf Pfadvorschlägen, welche den Stand der Wissenschaft übertrifft. Sie bietet angemessene Balance zwischen Suchleistung und Kommunikationsbelastung im Netzwerk. Außerdem wird das erste semantische System zur Datenreplikation genannt DSDR vorgestellt, welche semantische Informationen berücksichtigt vorhergesagten zukünftigen Bedarf optimal im P2P zu decken. Hierdurch erzielt k-random-Suche bessere Präzision und Ausbeute als mit nahezu optimaler nicht-semantischer Replikation. SPSC, ein automatisches Verfahren zur funktional korrekten Komposition semantischer Dienste, ermöglicht es Peers, gemeinsam komplexe Ablaufpläne zu komponieren. Mechanismen zur heuristischen bidirektionalen Verkettung und Rückstellung von Diensten ermöglichen hohe Ausbeute bei geringer Belastung des Netzes. Eine Methode zur Anfrageverzweigung vermeidet das Feststecken in Sackgassen im beschnittenen Suchraum. Beweise zur Korrektheit und unteren Schranke der Vollständigkeit von SPSC sind gegeben. iRep3D ist ein neuer semantischer Selektionsmechanismus für 3D-Modelle in P2P. iRep3D beantwortet effizient hybride Anfragen unter Berücksichtigung konzeptioneller, funktionaler und geometrischer Aspekte. Der Ansatz übertrifft vorherige Arbeiten bezüglich Präzision und Effizienz.

Acknowledgements

First and foremost my deep gratitude goes to my supervisor, PD. Dr. Matthias Klusch, for supervising me as a PhD student and for his kind guidance on my research throughout these years. He has never neglected to provide me with help whenever it was needed. Through numerous meetings and discussions, he has shown tremendous patience in guiding me from the day when I was very much a novice to research until now. In particular, I was deeply influenced by his precise scientific approach, which is most invaluable and cannot be learned from books.

My deep gratitude also goes to Prof. Dr. Raimund Seidel, Dr. Michelle Carnell and other officers in the Graduate School of the Computer Science Department, Saarland University for the scholarships during my PhD preparatory phase and the final thesis writeup stage. I would like to give my thanks to PD. Dr. Matthias Klusch and Prof. Dr. Philipp Slusallek for their agreement on funding me via project Collaborate3D during the three years in my dissertation phase.

Special thanks go to Patrick Kapahnke for his constructive suggestions on my research and implementation work, and to all the current and former members in the Intelligent Information Systems (I2S) group and the Agents and Simulated Reality (ASR) department of the German Research Center for Artificial Intelligence (DFKI).

I also extend my thanks to my friends in Germany and China for their motivation and encouragement. Most importantly, I give my greatest gratitude to my family, especially my parents, for their persistent support through these years, particularly in the difficult times. To them and to my lovely daughter Xinruo I dedicate this thesis.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Unstructured peer-to-peer network [323] is a well-known, important principle for resource sharing [346] [318], social network modeling [79] [244], collaborative task solving [16] and robust system construction [309]. Systems and applications depending on this technique (Gnutella [302] [362], FreeNet [65], Morpheus [381], etc.) have developed rapidly over the past few decades. An unstructured P2P network does not impose initial structure on its overlay, but rather is formed by interconnected equi-functional peers. It is highly robust against network dynamics, as all peers behave the same way by design. As a result, systems within this model do not need to incur extra network traffic for index maintenance and appear to have better load balancing than the (semi-) structured network-based variants.

On the other hand, the emergence of the semantic web [27] allows for describing entities in a machine-understandable, more precise way. This complements the major descriptive methods, such as natural language, which are known to be sensitive to keyword ambiguity. With different formalisms, like resource description framework (RDF)[1], web ontology language (OWL)[2], etc., logic-based reasoning can be performed over annotated semantics via modern computing devices. This advantage has been utilized by more and more systems and applications in various domains, such as data comparison [196] [324], decision making [73] [165], system training [266] [257] [284] and privacy protection [50] [304]. In particular, the interdisciplinary field of semantic P2P computing has attracted, in recent years, a large amount of attention from both the P2P computing and semantic web communities [117] [143] [234] [151] [154] [406] [84]. In this regard, the combination of semantic web technology and unstructured P2P networks provides more flexibility in terms of data compatibility, robustness and load balancing, compared to those semantic-based (semi-) structured networks with strict hashing and ID-based matching protocols.

In unstructured P2P networks, a common way to search for desired data is to issue queries. In this regard, query routing is a crucial point for satisfying user requests. Unfortunately, the primary limitations of unstructured networks also arise from this lack of structure. Each peer does not have a

---

[1] `http://www.w3.org/RDF/`

[2] `http://www.w3.org/TR/owl-features/`

global view on network topology and item location. These negative factors create risks of losing relevant items in time-to-live (TTL) restricted flooding-[132][232] or walkers-based routing [95][394] strategies. Flooding [132] [232] [242] or periodically enforced gossiping-based [59] [279] [369] methods tend to cause high network traffic costs due to duplicated messaging or probing. Walker-based variants, like [23] [353] [394] [95] [231], can alleviate this issue with content-based network overlay. However, peers in the solutions of this kind always tend to route a query to neighbor peer(s) where the request is determined to encounter relevant items, even though peers might know much more information about other relevant items on indirect neighbors. This insufficient use of peer knowledge lowers the search performance. Once such knowledge about items on remote peers is expressed in semantic formalisms, it becomes a challenge to determine how peers can make use of observed semantics for direct query routing with a reasonable tradeoff between performance and communication overhead .

In addition, with TTL restricted routing strategies, no matter how intelligent they are, some rare items that are unreachable from a requester peer within the TTL can never be found. This implies the need for an item replication mechanism, which is able to enrich the content overly network with useful information for enhanced query routing. In order to determine replication, syntactic-based approaches, like[337], leverage the keyword/ID matching between peer-observed queries and items. The effect of using semantic-based information in this area is not known. Therefore, leveraging semantic-based information to precisely direct data replication in unstructured P2P networks is a challenge.

Further, an unsatisfied user search often results from the lack of a relevant item in network. In this situation, even intelligent routing and item information propagation methods cannot help at all. In spite of this, it can still be very helpful if a queried P2P-based system is able to answer with relevant items, of which each is jointly composed by peers, once adequate useful components are found. However, the format of the data involved can vary in different domains. No system is able to support a complete solution for composition that covers all data formats. In this regard, semantic service composition in unstructured P2P networks concerns the problem in the functional aspect, which is a challenging first-class research issue that needs to be explored. In this area, there exist quite a few works, but each of them has its limitations. Probabilistic flooding-based solutions [108] [109] [110] tend to yield duplicate messages and large amounts of network traffic overhead. The performance of the efforts [117] [116] [152] [358] [11] is sensitive to the occurrence of dead-ends. The existing walker-based investigations [420] [102] rely heavily on user-designed plan templates. This hinders the automation of composition and lowers the systems' flexibility in answering arbitrary queries. Overall, none of the above works provides proof of its completeness or soundness. How to increase the composition performance (recall, etc.) while also keeping network traffic overhead low should be studied.

Finally, although the use of semantic information to unstructured P2P-based IR makes sense for achieving promising search performance, the benefit of its underlying semantic-based computation (e.g. logical reasoning) is not obtained for free. Depending on the kind of formalisms of semantics, the computational complexity can be high. When combined with domain specific non-semantic data, such as geometric feature data of 3D graphics, the comparison efficiency would become a bottleneck of those P2P systems that work with hybrid complex queries. Particularly, such a query might concern not only geometric, but also conceptual or functional features of 3D scenes. In this regard, the performance of previous non-semantic based efforts, such as [112] [124] [214], is subject to the syntactic keyword ambiguity that would yield mis-categorization or mis-ranking, while the semantic-based approaches, like [15] [176] [147], can still cause misclassification due to their syntactic-based RDF graph pattern matching and strict logical reasoning, which do not appear flexible enough to handle minor conflicts between logical expressions. Therefore, a method for the semantic-based efficient selection of semantically annotated 3D scenes is expected, and investigation of this topic will facilitate the construction of P2P-based intelligent collaboration in 3D virtual worlds.

1.1 Research Questions

Driven by the analysis above, in this section, we present the following targeted research questions in the area of semantic-based search and composition in unstructured P2P networks:

Challenge 1: Expert peer based semantic search in unstructured P2P networks
How can peers in an unstructured P2P network make use of their observed semantic knowledge for conducting collaborative query routing with a reasonable tradeoff between performance and communication overhead?

Challenge 2: Dynamic data replication strategy in unstructured P2P networks
How can peers leverage their observed semantic information on queries and items to collaboratively conduct data replication?

Challenge 3: Functional semantic service composition in unstructured P2P networks
In an unstructured P2P network, how can peers collaboratively perform functional IOPE-level semantic service composition that is able to mitigate the risk of failure caused by dead-ends and gives reasonable completeness?

Challenge 4: Efficient hybrid selection of semantically annotated 3D scenes for P2P search
In P2P-based search, how can 3D scenes with conceptual and functional descriptions be indexed for the purpose of efficient selection?

These research questions will be answered in Chapter 9.

1.2 Contributions

The main contributions of this thesis are as follows:

Expert peer based semantic search in unstructured P2P networks

In this thesis, we present a semantic-driven k walkers- based search scheme, called S2P2P, for data information dissemination and query routing in unstructured P2P networks. In S2P2P, each peer maintains its observation on the semantics of received queries (demands) and data information (supplies), as well as a local view on network topology. On top of this, each peer, in the process of forwarding a query, disseminates its known data information to a selected set of remote peers by taking advantage of query piggybacked data. For routing a query, each peer, instead of merely introducing an immediate neighbor or remote peer, suggests a query routing path containing a sequence of peers with expertise on topics similar to the query. This is achieved by a path suggestion heuristic that iteratively applies the Dijkstra algorithm in a greedy manner. Each iteration manages to detect one more expert peer and augments the current path suggestion with the shortest path from its tail to the detected expert peer. The comparative experimental evaluation shows that S2P2P outperforms a semantic flooding based search strategy in terms of search precision and recall. In addition, our evaluation reveals that S2P2P is at least as robust against the network dynamics as the semantic flooding approach. This work has been in part published in [47] [48] [201].

Dynamic semantic data replication in unstructured P2P networks

For this research question, we present a dynamic semantic data replication scheme called DSDR for classic k-random search in unstructured P2P networks. During its k-random search, each peer periodically updates its local view on the semantic network overlay based on observed queries (demand) and received information about provided items (supply), in particular their semantics. Peers dynamically form potentially overlapping groups for semantically equivalent or similar items they are currently requesting. In addition, each peer predicts the number of needed item replicas in the future based on its local observations in the past. The decision of which item to best replicate to which member is made within each demander group based on the maximal expected utility, traffic costs, and plausibility of such replication. Our experimental evaluation shows that k-random search and the semantic search S2P2P with DSDR-based replication can significantly outperform their combination with a near optimal but non-semantic replication strategy, as well as a peer expertise-based semantic P2P search without replication. This work has in part been published in [43] [195].

Functional semantic service composition in unstructured P2P networks

We attack this research question with a stateless walker-based approach, called SPSC, for functional automatic semantic service composition in unstructured P2P networks. With SPSC, each peer is enabled to observe and record the atomic and composite services from the workflows that answer its observed historic queries. On top of this, workflow answering of a given query is composed collaboratively by peers in a bidirectional chaining fashion. In particular, the work proposes a variable binding method for the atomic composition of services in terms of their inputs, outputs, preconditions and effects under the open world assumption. In addition, the composition at each peer is done with a guiding heuristic and memorization strategy. With with a query branching mechanism, the former enables peers to select more promising services and mitigate the failures caused by dead-ends, while the latter helps peers to alleviate the loss of potentially useful services and therefore improves the correctness of the overall result. Furthermore, by theoretical analysis, a lower bound on completeness and a formal proof of soundness are given for SPSC. Finally, the preliminary experimental evaluation results reveal that SPSC can achieve high cumulative recall with a relatively low network traffic overhead. This work has been in part published in [42].

Efficient hybrid selection of semantically annotated 3D scenes for P2P search

We target this research question in the semantic annotated 3D scene retrieval domain. In this thesis, we present a repository, called iRep3D, for efficient retrieval of semantically annotated 3D scenes in XML3D, X3D or COLLADA. The semantics of a 3D scene can be described by means of its annotations with concepts and services which are defined in appropriate OWL2 ontologies. The iRep3D repository indexes annotated scenes with respect to these annotations and geometric features in three different scene indices. For concept and service-based scene indexing, an approximated logical subsumption-based measure is derived, while the geometric feature-based indexing adheres to the standard specifications of XML-based 3D scene graph models. Each query for 3D scenes is processed by iRep3D in these indices in parallel and answered with the top-k relevant scenes of the final aggregation of the resulting rank lists. In addition, incremental re-indexing is investigated, which enables iRep3D to adapt to updates on annotations of 3D scenes. Furthermore, iRep3D scales to advanced semantic deep search and the reuse of relevant sub-scenes down to the individual object level. Results of experimental performance evaluation show that iRep3D can significantly outperform both semantic-driven multimedia retrieval systems FB3D and RIR, as well as the non-semantic based 3D model repository ADL in terms of precision and with reasonable response time in average. This work has been published in part in [44] [46] [45] [421].

1.3 Thesis Overview

In this thesis, we propose detailed solutions to the above research questions, as well as two application use cases, in which we made practical applications of the contributions to projects. This thesis is organized in 9 chapters.

In Chapter 2, we present background knowledge about the semantic web, P2P computing and their cross-disciplinary field, semantic P2P computing. In the semantic web part, we focus on the approaches and standards of the semantic web. After an introduction on the fundamentals, the basis of semantic service discovery and composition are presented. Each of these contains in addition a categorization and discussion of previous representative work. In the P2P computing part, basic types of P2P networks are introduced. For each type, its general principle, representative systems, strengths and limitations are discussed. On top of this, we introduce the general idea of P2P search and data replication. For each of them, previous representative work is classified and discussed. Chapter 2 ends with an introduction to semantic P2P computing, on which the research work of this thesis focuses. Besides the distinct new features and main challenges in this field, we examine the representative works on P2P-based semantic data (service) search and composition.

In Chapter 3, we present a method, S2P2P, for semantic search in unstructured P2P networks. It includes peer local observation, path suggestion, and item information dissemination processes. The highlight of S2P2P lies in that, for routing a walkers-based query, each peer is able to suggest a path containing a sequence of expert peers with respect to the requested topics, instead of merely an immediate neighbor peer. In this chapter, we also report the results of comparative experimental evaluation. It reveals that S2P2P search strategy outperforms a representative semantic flooding based search method.

In Chapter 4, we present DSDR, a semantic data replication scheme for k-random walkers in unstructured P2P networks. It includes the semantic demander group formation and replication decision processes. Unlike previous efforts, DSDR is the first semantic-based data replication strategy in unstructured P2P networks. It takes advantage of semantic technology in the peer-wise demand-supply relation determination and replication target peer selection. The experimental evaluation results give evidence that with DSDR, the classic k-random search achieves better search performance than the same search with a near-optimal non-semantic data replication strategy.

In Chapter 5, we present SPSC, a method for stateless functional semantic service composition in unstructured P2P networks. In a bidirectional manner, SPSC peers jointly compose services into complex workflows with more concrete parameter bindings. Such a collaborative process is guided by a heuristic that effectively alleviates the risk of failure due to dead-ends. Besides the peer local observation process, a memorization strategy is also introduced. Together, they enable peers to record, in a given query, un-chainable but potentially useful services, which increases the chance of achieving correct solutions. In addition to a path suggestion based query routing strategy, a query branching mechanism is derived for the purpose of investigating the possible solutions with alternative services. SPSC includes the theoretical analysis on its soundness and a lower bound of its completeness. These are unfortunately ignored by those previous efforts targeting the same problem. The experimental evaluation results show that SPSC can achieve a high cumulative recall with low network traffic overhead.

In Chapter 6, we present iRep3D, a method for the efficient deep search of semantically annotated 3D scenes, towards the fast semantic-based peer data selection for P2P search. This approach covers the off-line semantic indexing process for XML-based 3D graphics in terms of conceptual, functional and geometric feature descriptions, as well as a solution for fast on-line hybrid query answering. iRep3D scales to the automatic incremental indexing and the object-level deep search for sub-scenes.

The comparative evaluation results reveal that the search precision of iRep3D outperforms a syntactic based 3D search engine, a resource description framework based 3D scene retrieval method and an ontology based search strategy.

In Chapter 7, we present the application of the iRep3D system in the collaborative virtual product design and engineering domain. In this scenario, iRep3D is used as a 3D scene repository of a virtual production facility designer. The latter is a proof-of-concept demonstration for collaborative 3D engineering in industry and science[3]. In this chapter, we introduce the important parts of the design and implementation of the iRep3D system.

In Chapter 8, we present the application of the semantic search strategy S2P2P and the data replication scheme DSDR in the social media sharing domain. In this scenario, S2P2P and DSDR are employed as the pivotal solutions for the retrieval of media contents and services in a mobile P2P system, called MyMedia[4]. As an additional practical contribution of this thesis, a P2P framework has been designed and implemented. On top of this, we have also implemented S2P2P, DSDR libraries and also a P2P evaluation framework on which the search performance of the MyMedia system has been tested in unstructured P2P networks with various configurations. In this chapter, representative parts of the design and implementation of the above P2P based retrieval and evaluation suite are introduced.

In Chapter 9, we conclude our research on the targeted problems in the semantic P2P computing field. It includes the answers to the research questions proposed in Chapter 1 and discussions of future work.

---

[3] This work is driven by the BMB+F research project Scalable Intelligent Support and Reliability for Collaboration in the Future 3D Internet (Collaborate3D). `http://c3d.dfki.de/`.

[4] This work is driven by the EC FP7 integrated project SocialSensor `http://www.socialsensor.eu/`.

Publications:

The research and practical contributions in this thesis have in part been published in the following articles:

1. Klusch, M.; Cao, X.; Kapahnke, P.; Warwas, S. (2011): Semantics and Agents for Intelligent Simulation and Collaboration in the 3D Internet. Proceedings of the 7th International Conference on Semantics, Knowledge and Grids (SKG), 9–16. IEEE Press.

2. Cao, X.; Klusch, M. (2012): Dynamic Semantic Data Replication for K-Random Search in Peer-to-Peer Networks. Proceedings of the 11th International IEEE Symposium on Network Computing and Applications, 20–27. Cambridge (USA). IEEE Press.

3. Cao, X.; Klusch, M. (2012): Semantic Indexing for Efficient Multimedia Retrieval. Proceedings of the 10th International Workshop on Adaptive Multimedia Retrieval, 165–180. Copenhagen, Denmark. Springer, LNCS.

4. Cao, X.; Klusch, M. (2013): S2P2P: Semantic Search in Unstructured Peer-to-Peer Networks. Proceedings of the 15th IEEE International Conference on High-Performance Computing and Communications (HPCC), 971–978. IEEE Press.

5. Cao, X.; Klusch, M. (2013): Semantic-Driven K-Walker Search in Unstructured Peer-to-Peer Networks. *IEEE Computer Society STC on Social Networking E-Letter*, 1(3), IEEE.

6. Cao, X.; Klusch, M. (2013): iRep3D: Efficient Semantic 3D Scene Retrieval. Proceedings of the 8th International Conference on Computer Vision Theory and Applications, 19–28. Barcelona, Spain. ScitePress.

7. Cao, X.; Klusch, M. (2013): Advanced Semantic Deep Search for 3D Scenes. Proceedings of the 7th IEEE International Conference on Semantic Computing, 236–243. Stanford CA, USA. IEEE Press.

8. Zinnikus, I.; Spieldenner, T.; Cao, X.; Klusch, M.; Krauss, C.; Nonnengart, A.; Slusallek, P. (2013): A Collaborative Virtual Workspace for Factory Configuration and Evaluation. Proceedings of the 9th IEEE International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom), 353–362. Austin TX, USA. IEEE.

9. Klusch, M.; Kapahnke, P.; Cao, X.; Rainer, B.; Timmerer, C.; Mangold, S. (2014): MyMedia:

Mobile Semantic Peer-to-Peer Video Search and Live Streaming. Proceedings of the 11th ACM International Conference on Mobile and Ubiquitous Systems (Mobiquitous), 277–286. London, UK. ACM Press.

10. Cao, X.; Kapahnke, P.; Klusch, M. (2015): SPSC: Efficient Composition of Semantic Services in Unstructured P2P Networks. Proceedings of the 12th Extended Semantic Web Conference. Portoroz, Slovenia. Springer LNCS.

Chapter 2

Background

In this chapter, we present the fundamentals in the area of semantic-based search and composition in unstructured P2P networks. This chapter comprises three parts: semantic web and services (Section 2.1), P2P computing (Section 2.2) and semantic P2P computing (Section 2.3). The first part starts with a brief introduction about the approaches and standards of the semantic web. They are followed by an introduction of basic knowledge about semantic web services as well as the representative works on service discovery and composition. In the P2P computing part, we present the primitives including network topology, data items, user requests, topic distribution and item distribution. In addition, previous works about P2P search and data replication will be summarized. In the semantic P2P computing part, we introduce its features, research problems and representative works on the search and service composition in P2P networks. For those readers who are familiar with the semantic web and P2P computing fields, it is possible to skip the first two parts and directly enter the third part, a cross-disciplinary research field semantic P2P computing, in which the research of this thesis lies.

2.1 Semantic Web and Services

In this section, the basic notions of the semantic web and services are introduced. Starting with an overview on the semantic web, the discussion will cover the concept of semantic services and the representative works on its centralized discovery and composition.

2.1.1 Semantic Web in a Nutshell

Led by the international standards organization World Wide Web Consortium (W3C)[1], the semantic web is a worldwide progressive development of ideas that aims at converting and organizing the current web contents into a web of data[2]. Semantic web provides methods for describing the formations,

---

[1] `http://www.w3.org/`

[2] `http://www.w3.org/2001/sw/`

attributes, functions and relations of real world objects in a machine-understandable way [27].

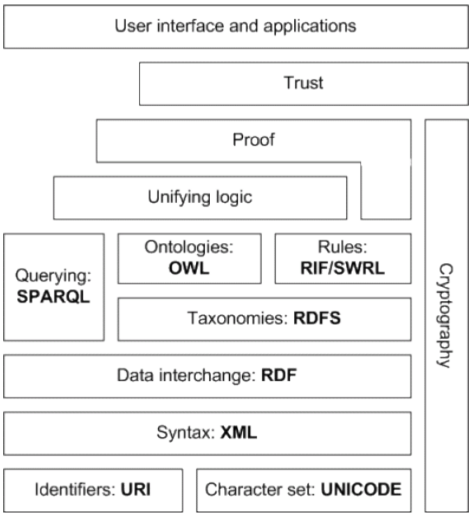


Fig. 2.1 The Semantic Web stack.

Semantic web architecture is conceptually illustrated by the semantic web stack (cf. Figure 2.1). As the most fundamental component, the platform-independent Extensible Markup Language (XML)[3] (and XML Schema[4]) provides the rules for encoding documents based on Uniform Resource Identifiers (URI)[5] and standard character set UNICODE[6]. On top of this, the core layers provide the standards for the formal description of Web data and its interoperability. The Resource Description Framework (RDF)[7] is a standard model for Web data interchange. The standard ontology description languages are concerned with the RDF Schema (RDF-S)[8] and the Web Ontology Language (OWL)[9]. As a semantic extension of RDF, RDF-S provides a data-modeling vocabulary for RDF data. To access RDF data, the standard way is to search with SPARQL[10] queries. OWL is a language to represent rich and complex knowledge about, group of and relations between things.RDF-S and OWL (Lite and DL) are decidable subsets of monotonic First Order Logic (FOL). Envisioned as the "rule layer" of the semantic web, the Rule Interchange Format (RIF)[11] is a standard for the integration and synthesis[12] of the rule sets in different rule based languages, such as SWRL[13], IBM ILog[14], Oracle BR[15].

---

[3] `http://www.w3.org/XML/`

[4] `http://www.w3.org/TR/xmlschema11-2/`

[5] `http://www.w3.org/Addressing/`

[6] `http://www.unicode.org/standard/WhatIsUnicode.html`

[7] `http://www.w3.org/RDF/`

[8] `http://www.w3.org/TR/rdf-schema/`

[9] `http://www.w3.org/TR/2004/REC-owl-features-20040210/`

[10] `http://www.w3.org/TR/rdf-sparql-query/`

[11] `http://www.w3.org/TR/rif-overview/`

[12] `http://www.w3.org/TR/2013/NOTE-rif-primer-20130205/`

[13] `http://www.w3.org/Submission/SWRL/`

[14] `http://www-01.ibm.com/software/info/ilog/`

[15] `http://www.oracle.com/technetwork/middleware/business-rules/overview/index.html`

Apart from the above, the other components are remained un-standardized and stay as research areas. The Proof component is designed for supporting the provenance knowledge, which is obtained from the automated agents that consume the information provided by lower layers [331]. The Cryptography component provides the encryption and decryption facilities. Working with it, the Trust component is supposed to provide the mechanisms and metadata that allow a user to verify whether a resource or statement coming from the lower layers is really issued by the trusted source, or not. Though not yet standardized, the solutions for the Cryptography and Trust components have been widely discussed in the semantic computing field [327]. The User interface and applications component gives a conceptual pre-vision that user applications can be built on top of all stacked techniques.

Concept and Ontology: In philosophy, the notion of concept refers to a fundamental category of existence[16]. It should have a definitional structure derived from a list of features. A feature entailed by the definition of a concept must be both necessary and sufficient for the membership in the class of things covered by this concept [262]. This idea has been applied in information technology where a concept means a set or class of individual objects [18] (*individual* in short). Those objects have the same types of features, called *atomic* (*primitive*) *terms*. Features of an individual should logically satisfy the definition of a concept, if the individual belongs to the type defined by that concept. Relationships can be specified between concepts or individuals. The most representative relationship is the concept subsumption ($\sqsupseteq$). A concept C subsumes a concept D ($C \sqsupseteq D$) (or D is subsumed by C) if any individual in D is an individual of C. In this case, C is called a *parent class* of D; while D is called a *subclass* of C. If any individual of C is also in the meanwhile an individual of D, C and D are equivalent ($C \equiv D$). Besides, the non-hierarchical relationships between categories can be described by roles. A role R describes the value restriction ($\forall R.C$) or the existential restriction ($\exists R.C$) of a relation in the world. It is considered as a binary predicate and the concept C is called as the *range* of C or a *role filler*; while the concept being described is named as the *domain* of the role. Another kind of role restrictions rules the cardinality of the range set, which is called *cardinality restriction*. With their relationships, concepts can be organized in a structure, called *ontology*. It is a formal, explicit specification of a shared conceptualisation [126]. In DL Attributive Language (***AL***) family, concepts can be formally specified according to syntax rules with terminological interpretation. For its detail, we refer the reader to the literature [322].

Ontology description languages: There are several ontology description languages for encoding domains knowledge, e.g. KL-ONE [388], KQML [96], DAML+OIL[17], OWL (OWL2), RDF-S, etc.. We briefly introduce RDF-S and OWL (OWL2) and compare their expressivity and complexity.

RDF-S. RDF-S is a semantic extension of RDF, which provides the abstract syntax for specifying

---

[16] `http://plato.stanford.edu/entries/concepts/`

[17] `http://www.w3.org/Submission/2001/12/`

(non-) hierarchical relations between RDF entities. It allows to specify the groups of resources, each of which is a RDF-S class (*rdfs:Class*). In addition, RDF-S makes it possible to specify the hierarchical relations between classes. One class can have multiple super-class as well as multiple subclasses. RDF-S Resource (*rdfs:Resource*) is the super-class of every class. Non-hierarchical relations are meant with properties. A property describes the relation between a subject resource and an object resource. RDF-S supports the hierarchical definition of properties. The subject of a property in RDF-S is modeled as domain (*rdfs:domain*); while the object of a property is modeled as range (*rdfs:range*). Commonly, a concrete object in the real world can have various properties which range is a number, string, date and other primitive types. RDF-S models this as Literal (*rdfs:Literals*).

OWL. The aim of OWL is to describe the classes and their relations of entities that exist among Web documents and applications. The described ontological information by OWL is named *Web Ontology*. Different with a XML schema, an OWL ontology is the form of knowledge representation, instead of a rule for formatting XML documents. Such knowledge can be used for reasoning, compared to the straightforward XML data processing. There exist several tools, such as Pellet[18], Jana[19], etc., for the automation of logical reasoning over the knowledge formulated in OWL. OWL offers three sub-languages with increasing expressivity: OWL-Lite, OWL-DL and OWL-Full. OWL-Lite supports the definitions for the primitive hierarchical classification and simple constraint. Having the correspondence with Description Logic [18], OWL-DL maximizes the expressiveness without losing computational completeness. OWL-Full is concerned for maximizing expressiveness and syntactic freedom of RDF without computational guarantees. A feature of OWL is that it makes an open world assumption. This indicates that resources' descriptions are not restricted in a single ontology, but inter-extendable in different distributed ontologies. Such kind of the connection between ontologies is achieved via ontology importation.

OWL2. An extension OWL2[20] of OWL has been derived by W3C OWL Working Group[21]. Compared with OWL, OWL2 has increased expressivity power for properties, extended support for datatypes, simple metamodeling capabilities, extended annotation capabilities, and keys. It has three sub-languages (syntactic subsets), OWL2-EL, OWL2-QL and OWL2-RL, called together as OWL2 profiles, which are pruned out of OWL2 for the applications with various expressivity requirements.

Expressivity and Complexity. Rather than introducing the OWL2 profile here, we refer the reader to their specification[22]. Instead, we conclude the expressivity and computational complexity on reasoning problems (ontology consistency (OC), class expression satisfiability (CES), class expression

---

[18] `http://clarkparsia.com/pellet/`

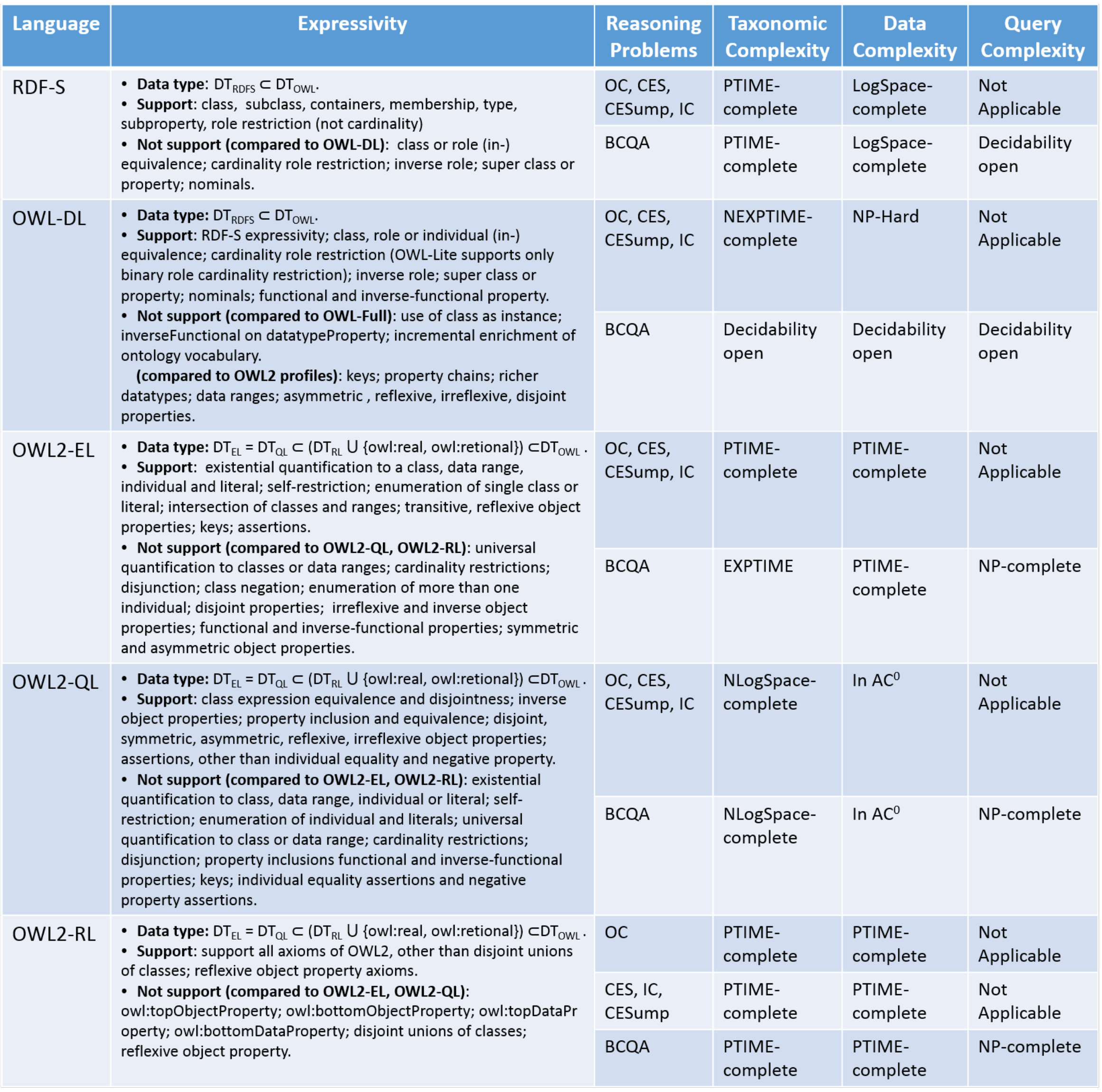
[19] `https://jena.apache.org/`

[20] `http://www.w3.org/TR/2012/REC-owl2-primer-20121211/`

[21] `http://www.w3.org/2007/OWL/wiki/OWL_Working_Group`

[22] `http://www.w3.org/TR/2012/REC-owl2-profiles-20121211/`

subsumption (CESump), instance checking (IC) and binary conjunctive query answering (BCQA)) (cf. Figure 2.2) of ontology description languages: RDF-S, OWL-DL, and OWL2 profiles.



Language	Expressivity	Reasoning Problems	Taxonomic Complexity	Data Complexity	Query Complexity
RDF-S	• **Data type**: $DT_{RDFS} \subset DT_{OWL}$. • **Support**: class, subclass, containers, membership, type, subproperty, role restriction (not cardinality) • **Not support (compared to OWL-DL)**: class or role (in-) equivalence; cardinality role restriction; inverse role; super class or property; nominals.	OC, CES, CESump, IC	PTIME-complete	LogSpace-complete	Not Applicable
		BCQA	PTIME-complete	LogSpace-complete	Decidability open
OWL-DL	• **Data type:** $DT_{RDFS} \subset DT_{OWL}$. • **Support**: RDF-S expressivity; class, role or individual (in-) equivalence; cardinality role restriction (OWL-Lite supports only binary role cardinality restriction); inverse role; super class or property; nominals; functional and inverse-functional property. • **Not support (compared to OWL-Full)**: use of class as instance; inverseFunctional on datatypeProperty; incremental enrichment of ontology vocabulary. **(compared to OWL2 profiles)**: keys; property chains; richer datatypes; data ranges; asymmetric , reflexive, irreflexive, disjoint properties.	OC, CES, CESump, IC	NEXPTIME-complete	NP-Hard	Not Applicable
		BCQA	Decidability open	Decidability open	Decidability open
OWL2-EL	• **Data type:** $DT_{EL} = DT_{QL} \subset (DT_{RL} \cup \{owl:real, owl:retional\}) \subset DT_{OWL}$. • **Support**: existential quantification to a class, data range, individual and literal; self-restriction; enumeration of single class or literal; intersection of classes and ranges; transitive, reflexive object properties; keys; assertions. • **Not support (compared to OWL2-QL, OWL2-RL)**: universal quantification to classes or data ranges; cardinality restrictions; disjunction; class negation; enumeration of more than one individual; disjoint properties; irreflexive and inverse object properties; functional and inverse-functional properties; symmetric and asymmetric object properties.	OC, CES, CESump, IC	PTIME-complete	PTIME-complete	Not Applicable
		BCQA	EXPTIME	PTIME-complete	NP-complete
OWL2-QL	• **Data type:** $DT_{EL} = DT_{QL} \subset (DT_{RL} \cup \{owl:real, owl:retional\}) \subset DT_{OWL}$. • **Support**: class expression equivalence and disjointness; inverse object properties; property inclusion and equivalence; disjoint, symmetric, asymmetric, reflexive, irreflexive object properties; assertions, other than individual equality and negative property. • **Not support (compared to OWL2-EL, OWL2-RL)**: existential quantification to class, data range, individual or literal; self-restriction; enumeration of individual and literals; universal quantification to class or data range; cardinality restrictions; disjunction; property inclusions functional and inverse-functional properties; keys; individual equality assertions and negative property assertions.	OC, CES, CESump, IC	NLogSpace-complete	In AC^0	Not Applicable
		BCQA	NLogSpace-complete	In AC^0	NP-complete
OWL2-RL	• **Data type:** $DT_{EL} = DT_{QL} \subset (DT_{RL} \cup \{owl:real, owl:retional\}) \subset DT_{OWL}$. • **Support**: support all axioms of OWL2, other than disjoint unions of classes; reflexive object property axioms. • **Not support (compared to OWL2-EL, OWL2-QL)**: owl:topObjectProperty; owl:bottomObjectProperty; owl:topDataProperty; owl:bottomDataProperty; disjoint unions of classes; reflexive object property.	OC	PTIME-complete	PTIME-complete	Not Applicable
		CES, IC, CESump	PTIME-complete	PTIME-complete	Not Applicable
		BCQA	PTIME-complete	PTIME-complete	NP-complete

Fig. 2.2 Expressivities and complexities of RDF-S, OWL-DL and OWL2 profiles. The complexities of RDF-S is given in the literature [191]; while the complexities of OWL-DL and OWL2 profiles are collected from OWL2 specification[22].

OWL-DL contains more vocabulary than RDF-S for describing classes, properties, cardinality, equality, richer typing of properties, characteristics of properties and enumerated classes. For instance, RDF-S does not support class or property (in-)equivalence, inverse, transitive, symmetric functional and inverse functional properties, role cardinality restriction, union, complement, or intersection of class expressions.

Compared with OWL-DL, OWL-Lite does not support enumerated classes, data range, equivalence on class expressions, boolean combinations of class expressions (union, complement and intersection), arbitrary cardinality (min and max cardinality) and filler information (hasValue), which are all supported by OWL-DL and OWL-Full. OWL-Full maximizes the expressiveness but without computational guarantees. Although OWL-DL and OWL-Full contain the same constructs, OWL-Full is more liberal about how they can be combined. In particular, it allows an ontology to augment the meaning of its pre-defined (RDF-S or OWL) vocabulary. This means OWL-Full is not fully supported by the current reasoner implementations. In addition, OWL-Full does not strictly follow the type separation restriction required by OWL-DL. For example, in the latter a class (property) cannot be an individual and property (an individual or class) at the same time. OWL-Full imposes fewer restrictions on object and datatype properties than OWL-DL.

OWL2 extends OWL and provides richer constructs. Overall, OWL2 profiles support keys, property chains, richer (customizable) datatypes, data ranges, asymmetric, (ir-)reflexive and disjoint properties, while OWL does not. However, OWL can make use of nearly all XML Schema datatypes. The OWL2 profiles are only compatible with specific subsets of them. OWL2-EL and OWL-QL use exactly the same set of datatypes. OWL2-RL adds more XSD types from XML Schema, but disallows $owl:real$ and $owl:retional$ (cf. $DT_{(\cdot)}$ in Figure 2.2). Each profile has its own expressivity characteristics. Their supported constructs overlap (cf. Figure 2.2). In particular, OWL2-RL supports nearly all OWL2 axioms, other than the disjoint union of classes and the reflexive object property. The former is disallowed in all three OWL2 profiles, but the latter is supported by OWL2-EL and OWL2-QL. The supported sets of constructs of OWL2-EL and OWL2-QL are approximately complement each other. The OWL2-EL compatible constructs, such as existential quantification to class, data range, individual and literal, self-restriction, keys, etc., are disallowed with OWL2-QL, while OWL2-QL offers a group of constructs, like disjoint, (a)symmetric, irreflexive, inverse object properties, which are not allowed in OWL2-EL. Neither OWL2-EL nor OWL2-QL support functional and inverse-functional properties.

Interlinked RDF data:

RDF. RDF[23] is a framework for describing Web resources and their relations. It refines the representation of knowledge, form unstructured natural language to structured *triples*. Each is expressed in the form of subject-predicate-object pattern. There are several data serialization format for defining triples, such as XML/RDF[24], Turtle[25], N-Triples[26], Notation3[27], etc.. Amongst them, the XML/RDF format is the standard representation. Besides, RDF triples can be represented via a directed graph, called *RDF graph*[28]. Each node in a RDF graph is a resource that can be a subject or an object;

---

[23] `http://www.w3.org/TR/rdf11-concepts/`

[24] `http://www.w3.org/TR/rdf-syntax-grammar/`

[25] `http://www.w3.org/TR/2012/WD-turtle-20120710/`

[26] `http://www.w3.org/TR/2014/REC-n-triples-20140225/`

[27] `http://www.w3.org/TeamSubmission/n3/`

[28] `http://www.w3.org/TR/2014/REC-rdf11-concepts-20140225/#section-rdf-graph`

while each directed edge is meant with a predicate that starts from its subject and points to its object resource. There exist a fair number of publicly available RDF data sets, like DBPedia[29], Yago[30], Freebase[31], OpenCyc[32], GeoNames[33], etc.. Among them, DBPedia is a crowd-sourced community that makes effort on extracting structured information from Wikipedia[34] and provides those information to the Web. It provides (version 3.9) the RDF triples about 24.9 million entities (12.6 million uniquely) in total via its localized versions in 119 languages. Yago contains more than 10 million entities that are derived from Wikipedia, WordNet[35] and GeoNames. Freebase is a collaborative knowledge base containing collections of structured data extracted from many sources. In beginning of 2014, it has collected about 2.4 billion facts over 44 million topics.

RDFa. The Resource Description Framework in Attributes (RDFa)[36] is a W3C recommendation that allows to add attribute-level rich metadata extensions to HTML[37], XHTML[38] and other XML based Web documents, such as SVG[39]. RDFa has two versions: (1) the RDFa-Core[40] specifies the full RDFa syntax including the basic and advanced features, which enables users to defined fairly complex structured data, e.g. relations between entities described in a Web document. In comparison with RDFa-Core, (2) the RDFa-Lite[41] has less predefined attributes and it is intended to be used by non-expert users for enhancing their Web documents with simple semantic meta-contents.

Linked Data. Linked Data[42] is about using the Web to connect related data that wasn't previously linked, or using the Web to lower the barriers to linking data currently linked using other methods. By using URIs and RDF, Linked Data is an excellent practice for exposing, sharing, connecting distributed data, information and knowledge on the semantic web. Tim Berners-Lee pointed out four rules[43] for the Linked Data:

- *Use URIs as names for things.* This rule conforms to the base of the semantic web that uses universal URIs for identifying objects and resources.

- *Use HTTP URIs so that people can look up those names.* The use of HTTP URIs makes

---

[29] `http://dbpedia.org/`
[30] `http://www.mpi-inf.mpg.de/yago-naga/yago`
[31] `https://www.freebase.com/`
[32] `http://www.cyc.com/platform/opencyc`
[33] `http://www.geonames.org/`
[34] `http://www.wikipedia.org/`
[35] `http://wordnet.princeton.edu/`
[36] `http://www.w3.org/TR/xhtml-rdfa-primer/`
[37] `http://www.w3.org/TR/html401/`
[38] `http://www.w3.org/TR/xhtml1/`
[39] `http://www.w3.org/TR/SVG/`
[40] `http://www.w3.org/TR/2012/REC-rdfa-core-20120607/`
[41] `http://www.w3.org/TR/rdfa-lite/`
[42] `http://linkeddata.org/`
[43] `http://www.w3.org/DesignIssues/LinkedData.html`

Linked Data to be compatible with the current web and such seamless integration can takes place easily by any users worldwide.

- *When someone looks up a URI, provide useful information, using the standards (RDF*, SPARQL).* This rule indicates that the Linked Data gives a new opportunity for the Web-based search. Besides the text-based search engines, SPARQL[10] can also be used for serving user queries.

- *Include links to other URIs. so that they can discover more things.* This rule implies the trend of web evolution. The result of search of the current Web search is commonly a list of web documents that contains syntactic relevant parts; while the search result of the Web of Data can be a list of objects in RDF format. Each object is organized on-the-fly based on the linked partial results collected from distributed data providers. For this, links-traversal-based search technologies, such as [135] [136] [203] [360], have been investigated and applied to serve queries with more intelligence and accuracy.

The most impressive feature of Linked Data is the links between distributed entities, which are specified in RDF, by Web users, organizations and published in a large number of HTTP accessible datasets crossing a wide range of domains. This enables users and machines to explore information and knowledge through links. Relying on this, the *Web of Data* is formed, in which data objects are related by object identifier URIs with semantics, instead of mere the hyper link-based references. The latest status of the linked data cloud is shown in Figure 2.3. According to the statistics by LODSTATS[44], there are 1048 interlinked datasets, in which 74,635,268,868 triples from 353 large datasets (686,415,104 triples from 276 dumps, 73,948,853,764 from 87 datasets via SPARQL). For the basics of Linked Data, we refer the reader to the journal paper [139] by Heath et. al..

Query languages:

SPARQL. The standard query language for searching RDF data is SPARQL[10]. In SPARQL, query criteria are expressed as graph patterns. A *basic graph pattern* is a set of triple patterns, which a solution (selected object in RDF) must match. A *group graph pattern* is a set of graph patterns, which a solution must all match. Graph pattern matching yields a set of solutions. Each has a set of bindings of variables to RDF terms. In basic and group graph pattern matching, every variable in a pattern must be bound to an RDF term in a solution. In addition, users can define query criteria on specific RDF term by a *filter*. A filter is a condition statement that asserts certain characteristics of the concerned RDF term value. The result of filtering is binary (true or false). However, this strict criteria may be not always satisfied by the underlying data, since the regular, complete structures cannot be assumed in all RDF graphs[45]. Therefore, SPARQL offers the the flexibility for the selection of

---

[44] `http://stats.lod2.eu/`

[45] `http://www.w3.org/TR/rdf-sparql-query/#optionals`

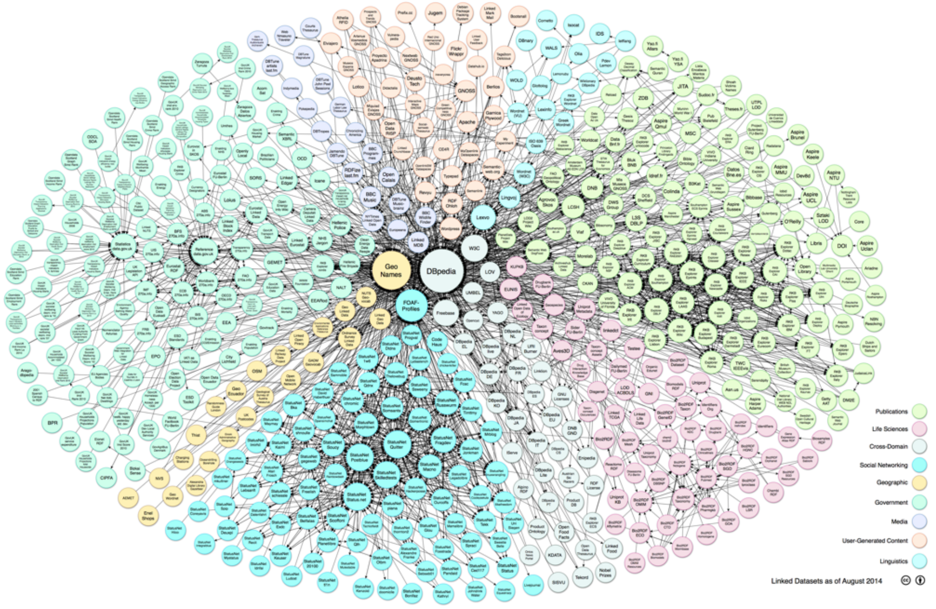


Fig. 2.3 Diagram of the linked datasets as of August 2014.

partially or alternatively matched entities. Similar with SQL, SPARQL supports a series of modifiers, like *Order*, *Projection*, *Distinct*, *Offset*, *Limit*, etc..

SPARQL 1.1[46] adds a number of new features to SPARQL. It supports subquery (nested) definition, path expression, aggregation means (like COUNT), value assignment, etc. It is designed to be compatible of working with several SERVICE extensions. In particular, the entailment regimes extension[47] of SPARQL 1.1 allows a user to specify more elaborated entailment, compared to the standard simple entailment of SPARQL, i.e. the basic graph pattern matching. With this, a SPARQL 1.1 query can trigger more complete materialization, for inferring the underlying semantics between data objects and properties. Another representative SERVICE extension of SPARQL 1.1 is the federated query[48]. With it, a query can explicitly delegate certain subqueries to different SPARQL endpoints. Their partial answers are combined as the final result.

SPARQL-DL. SPARQL is not applicable to OWL-DL ontologies, since the RDF data format mixes the syntax and language with its assertions [329], but OWL-DL does not. A solution proposed by Sirin et. al. called SPARQL-DL [329] allows to query OWL ontologies. SPARQL-DL is a subset of SPARQL, which supports to specify search criteria in TBox, ABox, RBox or their combinations. In practical perspective, SPARQL-DL has been supported by a search engine provided by DERIVO[49], which functionalities are fully built on top of OWL API 3. Besides, this implementation is entirely

---

[46] `http://www.w3.org/TR/2013/REC-sparql11-overview-20130321/`
[47] `http://www.w3.org/TR/sparql11-entailment/`
[48] `http://www.w3.org/TR/sparql11-federated-query/`
[49] `http://www.derivo.de/en/resources/sparql-dl-api/`

compatible with the OWL2 standard.

SQWRL. Another query language for OWL is Semantic Query-enhanced Web Rule Language: SQWRL [270]. Built on top of SWRL rule language, SQWRL has a standard SWRL rule antecedent and effectively treats it as a pattern specification for a query. In particular, SQWRL utilizes SWRL's built-in facilities as an extension point. Similar to SPARQL-DL, it allows the mixed TBox, ABox, RBox or their combined queries on OWL ontology. In addition, it contains a set of operators that can be used to perform closure operations and partially supports the negation, counting and aggregation. An implementation[50] of SQWRL engine is compatible with the Protégé[51] editor.

Semantics and related terminologies: In Merriam-Webster's Collegiate Dictionary [341], the term "semantics" is explained as "the study of meanings" or "the meaning or relationship of meanings of a sign or set of signs; especially connotative meaning". It indicates that a symbol (e.g. a piece of data, signal, keyword, variable, ID, number, etc.) in the world will has semantics, as long as it has meanings or relationships with other symbols. However, we would like to differentiate this general semantics of symbols from the semantics concerned by the semantic computing field, though their boundary is always blurred by some literatures that have imprecisely used the term "semantic". In the scope of semantic computing, the notion of semantics is more restrictive, and particularly refers to those items or processes that are described or formed based on either semantic web technology or lexical terms with their defined relations. To follow, we provide the notion of *semantics* and its related terminologies in the semantic computing field, which are used in the presentation of this thesis.

- *Semantic-based*: When we designate an item or a process using the term "semantic-based", it means that at least one of the two kinds of technologies are involved: semantic web technology or lexical terms with their relations (e.g. WordNet[52]). These technologies are either used for formulating the meaning, context and intention of the item or for item processing. In the former case, we say that the item has a "semantic-based description", while in the latter case, we say that "semantic-based processing" can be applied on that item. Of course, if the item itself is lexical terms with underlying relations, or in a formalism conforming to the standards of the semantic web (e.g. RDF, OWL, etc.), we describe the item itself as "semantic item (data)", and we describe the computation directly on it as "semantic-based".
- *Non-semantic based*: When we designate an item or process using the term "non-semantic based", it means that none of the semantic web technology or lexical terms with relations is used for either item description or processing. Accordingly, if an item is described by symbols (e.g. variable, keyword, ID, number, signal, etc.), compared via exact matching between

---

[50] `http://protege.cim3.net/cgi-bin/wiki.pl?SQWRL`

[51] `http://protege.stanford.edu/`

[52] `http://wordnet.princeton.edu/`

symbols, or processed based on the occurrence of symbols, we say that the item description or the processing is "non-semantic based". Here, we emphasize that the linguistics technology should be "semantic-based", as it cares about not only the occurrence of words but also their relations.

- *Logic-based*: When we designate an item or process using the term "logic-based", it means that (1) the item formulation is compatible with and supports logical reasoning, or (2) the item processing itself includes logical reasoning.
- *Non-logic based*: When we designate an item or process using the term "non-logic based", it means that (1) the item formulation is not compatible with or does not support logical reasoning, and (2) the item processing itself does not include logical reasoning.
- *Hybrid*: When we designate an item or process using the term "hybrid", it means that both logic-based and non-logic based item formulation or processing are involved. The term "hybrid" here is different from the term "hybrid" in the scope of P2P computing (cf. Section 2.2). In the latter, the term "hybrid" is used to describe the organization of item indexes in P2P networks and has nothing to do with computational logics.

2.1.2 Semantic Services

The aim of semantic web service (*semantic service* in short) is to enable the automated, interoperable and meaningful web services coordination [192] [93]. Intuitively, a semantic service is a web service (WSDL[53] and RESTful [308]) with its semantic annotations. Supposed to be performed by intelligent agents, service coordination covers a wide range of sub-tasks, including automatic service discovery, composition, interoperation and invocation. Unfortunately, the classic web service (WSDL and Restful) standards appear not adequate for this purpose. Semantic service offers the description methods to specify input (I), output (O) parameters, as well as the precondition (P) and effect (E). Moreover, semantic services are defined in the logic-based formalisms, on which automatic service coordination can be realized. We introduce the description languages of really deployed prominent semantic services [202]: OWL-S[54], WSMO framework[55], WSDL-S[56] and SAWSDL[57].

- *OWL-S*: Semantic Markup for Web Services, is an upper ontology of services, which is built on top of OWL and grounded in WSDL. OWL-S became a W3C candidate recommendation in 2005. OWL-S consists of three main sub-ontologies: Profile, Process Model and Grounding. Profile ontology defines the metadata for describing the functionalities of semantic services. This metadata covers not only the IOPE signatures essentially required by semantic service,

---

[53] `http://www.w3.org/TR/wsdl`

[54] `http://www.w3.org/Submission/OWL-S/`

[55] `http://www.wsmo.org/TR/d2/v1.4/20061106/`

[56] `http://www.w3.org/Submission/WSDL-S/`

[57] `http://www.w3.org/TR/sawsdl/`

but the service attributes, text description, category, etc.. Process Model sub-ontology specifies the metadata for the description of composite process (composite service). The latter refers to a workflow consisting of possibly multiple atomic services, each of which has different output and effect given the variety of input and precondition. Grounding sub-ontology supports the pragmatic binding between the logic-based description of a semantic service with its execution-oriented definitions, for example, a WSDL profile.

- *WSMO framework*: Web Service Modeling Ontology framework, contains a conceptual model and its derivative Web Service Modeling Language (WSML) for defining semantic services. WSMO contains 4 components, Ontologies, Goals, Web services and Mediators. The Ontologies component provides the terminology used by WSMO components for describing the targeted domains of discourse; Goals refers to the user expectation about the execution of a Web service; Web services component is used to describe the capabilities, interfaces, and even internal working processes of a web service; while the Mediators component provides the metadata for specifying service compatibilities and interoperabilities.

- *WSDL-S*: Web Service Semantics, is an incremental approach aimed at adding semantics to WSDL based web services. Its highlight consists of two folds: With WSDL-S, a user can specify both the semantics and operations for WSDL services in a compatible way. Besides, it allows the domain model semantics to be language-agnostic, which increases the reusability of business models. In more detail, WSDL-S provides the abstract syntaxes for the mapping between WSDL complex types and concepts in an external semantic model. The annotation of precondition and effect is also covered, which can be added to an *operation* element in WSDL. Like OWL-S, it scales to describe the categorization of service as well.

- *SAWSDL*: Semantic Annotations for WSDL and XML Schema, is a technical recommendation of W3C for building semantic WSDL services. Similar to WSDL-S, it allows for descriptions of additional semantics in WSDL components. For this, SAWSDL specification provides a series of extensions on WSDL interfaces and operations, which allows for specifying the mappings between complex WSDL data types with the concepts defined in external ontologies.

Semantic service discovery: "Service discovery is the process of locating existing Web services based on the description of their functional and non-functional semantics" [192]. Intuitively, service discovery is meant with the retrieval of services [380]. In this paradigm, the request asking for services is commonly given as a service signature in a certain description language, while the result of service discovery is a list (set) of ranked (or non-ranked) services with relevancy scores (or only the binary relevancy). In semantic service discovery, two important aspects are often considered: (1) how the request matches the candidate services and (2) what the discovery architecture is. For the first aspect, semantic web services can be formulated via different description languages (cf. Section 2.1.2). Depending on the type of description, different discovery mechanisms can be applied, which might vary in terms of the concerned service signature (IO, PE or IOPE) and matching (selection) means (logic-based, non-logic based or hybrid; adaptive or non-adaptive). The logic-based service selection methods match service candidates with requests relying on logical subsumption of IO parameter concepts and/or logical specification match over PE signatures. The non-logic based variants focus on comparison over a syntactic based description or structural similarity. Adaptive selection means learning how to best aggregate different matching filters [194]. For the second aspect, service discovery architecture is subjectively determined by the distribution and organization of service maintainers (centralized or distributed). In this section, we discuss the recent representative centralized semantic web service discovery approaches. They are categorized in Figure 2.4. For the distributed variants, we refer the reader to Section 2.3.2, in the context of semantic P2P computing. We show a classification of them in Figure 2.11.

Approach	IO Level	IOPE Level	Logic-based	Non-logic based	Hybrid	Adaptive	Service Formalism
OWLS-MX (Klusch+ 2006)	√				√		OWL-S
OWLS-MX3 (Klusch+ 2009)	√				√	√	OWL-S
OWLSM (Jaeger+ 2005)	√				√		OWL-S
DLH (Meditskos+ 2010)	√				√		WSMO
PSemMa (Cassar+ 2012)	√			√		√	OWL-S / WSMO
SAWSDL-iMatcher (Wei+ 2011)	√			√			SAWSDL
HomSemMa (Liu+ 2009)	√			√			WSDL (OWL for IO)
iServe (Pedrinaci + 2010)	√		√				OWL-S / WSMO (in RDF)
UBRE (Plebani+ 2009)	√			√			WSDL / SAWSDL
COV4SWS (Lampe+ 2012)	√			√			WSDL (OWL for IO)
LOG4SWS (Lampe+ 2012)	√		√			√	SAWSDL
OWLS-iMatcher (Kiefer+ 2008)	√				√		OWL-S (in RDF)
ESVMSemMa (Wang+ 2011)		√		√			RGPS
GLUE (DellaValle+ 2005)		√	√				WSMO
WSMO-MX (kaufer+ 2006)		√	√				WSMO
ALS (Keller+ 2005)		√	√				WSMO
iSeM (Klusch+ 2012)		√			√	√	OWL-S
LTSSemMa (Junghans+ 2010)		√	√				OWL-S / SAWSDL
BiSemMa (Bener+ 2009)		√	√				OWL-S
YASA-M (Chabeb+ 2010)		√	√				(SA) WSDL (in RDF)
iRep3D (Cao+ 2013)		√	√				OWL-S
SeMa^2 (Masuch+ 2012)		√			√		OWL-S
SPARQLent (Sbodio+ 2010)		√	√				OWL-S (in RDF)
XSSD (Li 2013)		√			√		OWL-S
Opossum (Toch+ 2007)		√			√		OWL-S

Fig. 2.4 Representative centralized semantic service discovery approaches. These approaches are discussed in Section 2.1.2. As for the distributed variants, we categorize them into the semantic P2P computing field. These related work is studied in Section 2.3.2 and classified in Figure 2.11.

IO-Level semantic service discovery. Klusch et. al. present a centralized hybrid matchmaker OWLS-MX [196] for the selection of OWL-S services. OWLS-MX exploits both logic-based reasoning and content-based information retrieval techniques for the selection of services at IO level. OWLS-MX concludes the matching between services as seven levels, which contains the logic-based matching degrees *EXACT*, *PLUG-IN*, *SUBSUMES*, *SUBSUMED-BY*, *LOGIC-FAIL* as well as the non-logic based matching degrees *NEAREST-NEIGHBOR* and *FAIL*. From *EXACT* to *FAIL*, the semantics of the filters' constraints are gradually relaxed. An revisited version of OWLS-MX, OWLS-MX3 [199] allows for the aggregation of logical, text-based as well as structural similarity via Support Vector Machine (SVM) classifier. Its aggregation is optimal with respect to average classification accuracy achieved by its binary SVM classifier. In brief, the classifier works with a 7-dimensional feature space. The latter contains the matching degrees from the five binary logical, the real-valued text-based and structural aspects. With over 700 labeled queries, the parameters of SVM classifier kernel are trained by means of grid search and 6-folded cross validation.

Instead of return a binary (match or not match) answer, another IO-level OWL-S matchmaker, OWLSM [158], measures the strength of matching by a real-value at each matching degree. It considers the contravariance for the service parameter types and their subtypes in OWL ontology and also provides the matching via service categorization, quality and other constraints that might be ruled by user queries. DLH [254] combines the advantage of the WSMO discovery framework and OWL-S service profile modeling paradigms and proposes a method for the centralized hybrid selection of OWL-S service. Its hybrid similarity measure checks (1) the terminological similarity over service topic category, (2) concept hierarchical similarity against service IO signatures, and (3) non-functional similarity over the values of data- and object-properties. The aggregation is the weighted average of the similarity scores over the 3 dimensions above. Such weights are assumed to be set by external users. Lampe et. al. present an IO-level service matchmaker, called LOG4SWS.KOM [366], for SAWSDL services. Built on top of the traditional subsumption reasoning, it in addition offers the mapping between classic service matching degrees and a continuous numeric space by a path length-based measure. By using the Ordinary Least Squares (OLS) [389] estimator over a training data set, it overcomes the possible arbitrariness of the matching degree-numeric value correspondence. LOG4SWS.KOM achieved the best search performance amongst the SAWSDL matchmakers in the S3 Contest 2012 [193].

Cassar et. al. present a probabilistic matchmaker, PSemMa [51], for IO-level selection of OWL-S and WSMO services. It models a probabilistic distribution, called service transaction matrix, of parameter concepts over services. Each service is mapped onto this matrix. Based on the Latent Dirichlet Allocation [34] learning technique, the origin high-dimension matrix is transformed to a low-dimension matrix. This narrows the search space. Wrapped as a low-dimensional vector, a query is compared with the service vectors in the latter matrix via cosine similarity. The value of this contribution is that it considers the matchmaking of service in a non-traditional way. However, it does not provide this possible improvement and its learning process has the risk of losing semantics.

Another non-logic based IO-level matchmaker, SAWSDL-iMatcher [386] is proposed by Wei et. al., for the selection of SAWSDL services. It proposes IXQuery similarity measure by means of the fairly mature XQuery technique. Besides, it takes advantage of ontology and a Pellet OWL reasoner for checking the similarity between service IO parameter concept similarity. The highlight of this work lies in that it investigates the technical solution targeting the matchmaking of semantic web services, even though it has not too much theoretical or methodological contribution. Plebani et. al. present a non-semantic IO-level WSDL web service selection method, called URBE [290]. The similarity between services is computed by a hybrid measure that combines the structural and textual relevances. In addition, URBE contains a semantic-based variant, which works with semantically annotated WSDL services, SAWSDL. Another non-logical IO-level service selection method, COV4SWS.KOM [366], applies information commonality [226] and information content [306] similarity measures based on the assignment of probabilities of the occurrence of concepts in ontologies. In particular, COV4SWS.KOM leverages the OLS estimator [389] to training its kernel in order to be able to automatically determine the weights of different abstraction levels of services.

OWLS-iMatcher [186] offers hybrid IO-level approximated selection of OWL-S services. It is built on top a approximated RDF matching mechanism, called iSPARQL [187], which is able to perform fast matching over the RDF representations of OWL-S services. As a prerequisite, the concerned OWL-S services and their related syntactic descriptions are transformed into RDF data format. In particular, OWLS-iMatcher features machine learning based matchmaking. That is agnostic to the learning and semantic matching strategies. With a given training set, OWLS-iMatcher learns the similarity relation patterns between queries and relevant services and further organizes the results by an induction model. Given a query, it computes the similarities between the query and all known services, then uses the induction model to predict the combined similarity.

Another IO-level service discovery method, HomSemMa [230], allows to return a list of not only the matched existing services, but also the ones composed on top of the known component services. Each in latter group is composed on-the-fly based on a pre-generated service aggregation graph (SAG). Similar with the idea of graphplan (used in the FF planner [145]), HomSemMa system forms SAG by in prior exploiting the possible chaining of services via clustered service IO parameter concepts. Given a request, it computes the subgraph that traverses, via shortest paths, from all the requested input nodes to all output nodes. The value of HomSemMa is the combination of classic service discovery with graph-based AI planning. Nonetheless, this approach cares only the chaining at IO-level.

IOPE-Level semantic service discovery. Besides the GLUE [76] and WSMO-MX [183] matchmakers, another early contribution ALS [185], is presented by Keller et. al., for the centralized IOPE-level selection of semantic web services. ALS maps service IOPE signatures and request onto a state space, which encapsulates the transitions between known states. Given a query, ALS locates a set of pre-defined goals that is similar to the desired goal (service). The actual service discovery happens for finding services that match the pre-defined goals. Finally, the service contracting sub-process checks the results and prunes out the irrelevant ones to the original request.

Another representative centralized hybrid IOPE-level semantic service matchmaker is iSeM [200]. In iSeM, authors investigate the concept abduction-based similarity measure, which is used to determine the approximated logical subumption relationships between parameter concepts. The PE matching part of iSeM is accomplished by $\theta-$subsumption [240]. Similar with OWLS-MX3, iSeM proposes to use SVM learning for result aggregation over a 10-dimensional vector space. iSeM was the best hybrid OWL-S matchmaker in S3 Contest 2012 [193].

Apart from the θ-subsumption and machine learning methodologies, the graph algorithms, RDF-based similarity and graphplan techniques can be also considered for the matchmaking at PE-level. Bener et. al. present BiSemMa [24] for centralized IOPE-level selection of OWL-S services. Its IO-level part is delegated to the comparison between service parameter concepts by both logic-based reasoning as well as the word synonyms similarity; while its PE part is accomplished by bipartite matching. In brief, it creates a graph that contains the nodes from the preconditions (effect) of a request and also a candidate service. Each predicate in precondition (effect) is modeled as a node with weighted atoms. On top of this, the target of matching then becomes to find a bipartite matching in this graph. A limitation of BiSemMa is that it can only deal with predicates with conceptual parameters, rather than literals of primitive types, like number, string, etc. The SeMa^2 software module [267] implements a hybrid IOPE-level matchmaker. This work combines the syntactic and semantic similarity measures on OWL-S services. In particular, it offers a structural comparison for the PE matching in SWRL by instance-based containment relations.

SPARQLent [320] offers semantic-based IOPE-level selection of semantic web services by leveraging the SPARQL query facilities on their transformed RDF representation. For each service in SPARQLent, the transformation treats the IO variables of services as the variables in a SPARQL query; while the PE formulas are represented as RDF graphs between variables and instances. Given a requested service, SPARQLent transforms it into RDF graph-based representation and performs graph matching. The highlight of this approach lies in that it leverages the SPARQL search engine to facilitate the efficient retrieval of semantic services. However, with mere the simple entailment, i.e. basic graph pattern matching, the SPARQL query is not able to detect the hidden semantics between objects, but only performs the matching based on the occurrence of terms. Another work XSSD [221] put attention on shortening the query response time of service selection. Similar to SPARQLent, the semantic-based hybrid IOPE-level semantic service discovery method XSSD extracts the useful semantics of OWL-S services during service publishing period by means of Semantic Pretreatment Algorithm. Such extracted semantics are organized via so-called service feature matrix (SFM). Likewise, a service request is can be also represented in the SFM form. On top of this, operation on the offer and request matrices can be applied to determine their similarity.

Chabeb et. al. propose YASA-M [54] for semantic-based IOPE-level selection of semantic services. Differing with other methods, a request in YASA-M is represented ordinarily in SPARQL and the candidate services are annotated additionally with RDF descriptions. The discovery process of YASA-M contains two steps: (1) pre-selection and (2) YASA matching. In the first step, a set of services are pre-selected. In the second step, YASA-M uses three measures to check the similarity between the request with the pre-selected services and further aggregates their results. In brief, its Elementary Matching module compares the conceptual description of each pre-selected service with the request; while the Min-Average Matching module mainly performs IO-level selection. In addition, YASA-M hire in addition a measure that checks the graph similarity, by Cupid algorithm [239], between the schema graphs of each pre-selected service and request. Another system, iServe [280], applies the similar SPARQL-based principle for the selection of OWL-S and WSMO services.

A contribution, LTSSemMa [170] employs a graphplan based model for IOPE-level discovery of semantic services in OWL-S and SAWSDL. Its main idea is to create a graph-based transition system that records compatibility of services. Each service is an action edge. The service input and precondition (output and effect) is combined and further modeled as as a state node. The matching of service with this transition system becomes to find an edge with certain end points. A similar approach Opossum [357] targets the problem of efficient IOPE-level selection of OWL-S services or their compositions. In Opossum, authors apply the inverse principle used in HomSemMa [230], which maps service precondition and effect to input and output, respectively. It proposes a structure, called service network, to model the IO-level dependencies between services. In particular, such dependency relations are indexed into concept index and service index. In concept index, service parameter concepts are mapped to their associated services. This is aimed at efficient matching with query concepts; while the service index is a graph-based index that concerns the structural summary of service network for the purpose of answering those queries with several atomic services.

ESVMSemMa [378] uses machine learning methodologies for selection of RGPS-compatible services [377]. The latter meta-model provides abstract syntax for the static (service name, service target, etc.) and behavioral (IOPE, QoS, etc.) semantics of services. It provides an extension of SVM that is particularly designed for verifying and adjusting the existing author/user-centered service categorization as well as enriching the domain ontology. ESVMSemMa approach uses a new keyword frequency- inverse document frequency-domain frequency as the SVM classifier kernel, which bridges over the gap between query keywords and service category domain. Via keyword frequency-inverse repository frequency, the services in the repository and a query can be matched. Strictly speaking, this approach does not leverage logic-based reasoning for the service matchmaking, because it follows the RGPS model.

Semantic service composition: "Semantic service composition (abbr. service composition) is the act of taking several semantically annotated component services, and bundling them together to meet the needs of a given customer. Automating this process is desirable to improve speed and efficiency of customer response, and in the semantic web, supported by the formal grounding of service and data annotation in logics" [192]. In this context, the "needs of a given customer" are meant as the goal (or requested service), which is supposed to be composed by some planning sites. A *planning site* is a computational device that is capable of performing a certain service composition algorithm, based on its knowledge about services. Depending on the type of service composition approaches, one or multiple planning sites can be involved in resolving a given request. The result of composition is commonly called the *plan* (or workflow). It is a sequence of services plus parameter bindings, which can themselves contain parallel sub-processes controlled by splitting and joining operations.

We distinguish the notion of semantic service composition in the context of this thesis with the notion of semantic web service mapping, though in the literature of the latter the term "composition" is also used. The service mapping approaches target discovering concrete semantic services depending on certain criteria in order to fulfill the requirements that are commonly pre-derived (by requesters at design time) abstract workflow templates. Since it is different from the automatic service composition problem in this thesis and the AI planning domain, we do not discuss efforts on this topic and refer the reader to the representative works, such as [128], [403], [238], [88], [393], [351], [260], [134], [106], [407], [49], [131] as well as the ones in the surveys [188], [157] and [326].

For the classification of service composition approaches, we follow the two groups of criteria, dynamic or static and function-level or process-level, in the literature [192], and in addition consider whether a method is centralized or distributed, state-based or stateless.

- *Centralized or distributed service composition*: Depending on whether the requested service is composed in a centralized or distributed manner, service composition methods can be classified into two types: centralized and distributed. Centralized approaches always work with only one planning site. This site has the global view on all services. In contrast, distributed variants can have multiple planning sites. A planning site has either a global view of services and their locations via network index facilities, such as indexes of structured P2P networks, or only a local view of its observed services.

- *Dynamic or static service composition*: Depending on whether the service composition and the plan execution are inherently interleaved, or not, service composition methods can be classified into two types: dynamic and static. In general, the target of service composition is for further execution, which is particularly needed in real-world industrial scenarios. Static service composition does not allow one to execute a partial plan before the composition process gets finished. In contrast, dynamic service composition does. Therefore, dynamic service composition planners are more suitable to work in dynamic environments, where service availability, signature and other such attributes might change at execution time. Furthermore, such experimental execution of a partial plan is even chosen for some scenarios that require the resulting plan to fulfill certain criteria in terms of overall plan quality or execution time.

- *Functional-level or process-level service composition*: Depending on whether the plan generation relies on only the service profile semantics, or additionally the process model semantics (e.g. data, conditional controls, etc.), service composition methods can be classified into two types: functional-level and process-level. Functional-level service composition often results in a sequence of semantic web services, and this is done based solely on the semantics of service profiles (e.g. IOPE signature). In this context, each component service is treated as an atomic service, which has fixed output and effect as long as the provided input values meet the precondition restriction. Process-level service composition allows one to work with composite services, which might internally contain flow control operators over multiple atomic services. Given different preconditions and inputs, a composite service can have different outputs and effects.

- *State-based or stateless service composition*: Depending on whether the plan generation relies on a pre-defined world state space, or not, service composition methods can be classified into two types: state-based and stateless. Generally, state-based service composition approaches treat a component service as an action. It can be applied as long as the current world state meets its pre-condition. After its execution, the world state will possibly be changed, which is governed by its post-condition. A state-based planning task is described by the initial state, goal state and a set of available actions. Commonly, the state-based planner presumes that the world state set would not change during the planning and the resulting plan is therefore executable with those assumed states only. To apply state-based planning for semantic web service composition, one option is to transform the service I/O signature into extra predicates and combine them with P/E, respectively. Through this transformation, a semantic service can be treated as an action. In contrast, stateless service composition does not impose a closed world state set on the planner. A stateless planning task is directly given as a requested service. Without the extra transformation from I/O parameters to P/E predicates, the composition is performed directly based on IOPE signatures. In addition, because of its dependency on the open world assumption, once composed, the resulting plan is expected to be able to accept any input, as long as the data type restriction and precondition are satisfied.

Regarding a composition approach, two additional important theoretical aspects should be considered:

- *Completeness*: A semantic service composition planner is called *complete*, if it can find the solution for a given request, as long as the solution exists; the planner reports failure when no solution exists. It is reasonable to check this aspect of a system in which all component services are visible to the planning processes. However, in some distributed (as in unstructured P2P networks) service composition systems, no individual participant has a global view of all the resources of the entire system. The completeness of the whole system might not be guaranteed, unless a request can be exhaustively spread to all participants with proper coordination mechanisms. Commonly realized by message flooding (cf. Section 2.2.2), such propagation and possible coordinations could cause tremendous network traffic. This is precisely the issue that should be avoided by distributed systems. Therefore, instead of strict completeness, a lower bound of completeness is considered in such systems: Given a request, a distributed service composition planner can find the solution with some probability above a lower bound, if such a solution exists within the topological range that is reachable by the specific request message propagation strategy and possible coordinations; the planner replies failure, if the request is not solvable within the reachable range of the system's request message propagation and coordination.

In this perspective, an issue is the *dead-end*. In state-based composition, a dead-end refers to a state iff it is reachable and no sequence of actions achieves the goal from it [145]. In stateless composition, a dead-end stands for a set of output parameters and a conjunction of effects of the services in a partial plan, from which no sequence of known services in a system can be applied in order to achieve the requested output and effect overall. Dead-ends can happen particularly in systems where planning sites do not have a global view on services and they perform only the chaining of actions (services). Therefore, a mechanism is expected, which should be to be able to predict or measure the prospects for the current results caused by the latest attempt at action (service) chaining, and further redirect the composition to a better status, if the use of that action (service) is determined not to be promising.

- *Correctness* (*Soundness*): Given a set of component services and a requested service with pre- and post-conditions, a service composition approach is *correct* (*sound*), if it can always produce output meeting the post-condition when its overall input satisfies the precondition. This notion is borrowed from the definition of algorithm correctness [99] [144], as the composition process itself is an algorithm. In this context, a pre- or post-condition means either the parameter type restriction, service precondition/effect satisfaction, or both, depending on the formalism of services (e.g. OWL-S supports IOPE and therefore the pre-/post-condition can cover both criteria; SAWSDL supports IO only and hence the pre-/post-condition concerns parameter type restriction). Proving the correctness of a service composition planner is an important theoretical aspect. The basic principle for this is to prove the correctness of each of its statements. Here, a statement also has its own pre- and post-conditions that govern its input and result. However, achieving this appears quite complex and the possible proof would have a very large number of similar sub-proofs for statements. A simpler but effective alternative is induction, according to Floyd-Hoare theory [99][144]. The essence of this is to find certain invariants and prove they hold (i) in the initial state of the algorithm; (ii) after the i-th internal step if they hold before the i-th step; (iii) after the algorithm terminates. A service composition algorithm can be regarded as a recursive process. In each iteration, some actions (services) are chained to the current partial plan in order to reach the requested goal state (form the requested service). Therefore, the task of (ii) above is to prove the selected invariants hold after each iteration, if they hold before that iteration.

  A crucial phase here is the selection of invariants. In general, the invariants should be able to guarantee or imply that the algorithm post-condition holds in the end, given the truth of the overall pre-condition. The underlying principle of doing so is the certifying algorithm

> theory by Mehlhorn et al. [252]. Concretely, the invariants of a service composition approach should give evidence that the composition process eventually leads the partial result to a correct solution to a given request.

Following on this section, representative centralized service composition methods will be introduced, while the distributed variants will be studied in Section 2.3.3. We classify the centralized and distributed approaches in Figures 2.5 and 2.12, respectively.

A representative centralized semantic service composition planner is OWLS-XPlan. It has two variants: OWLS-XPlan1 [198] performs state-based IOPE-level composition for OWL-S services; while OWLS-XPlan2 [197] allows for the advanced dynamic composition in stochastic environments. OWLS-XPlan1 is built on top of the relaxed GraphPlan-based fast forward (FF) planner [145] and the hierarchical task network (HTN) planner [91] of domain-specific complex actions. In order to apply FF planner, OWLS-XPlan1 converts the domain ontology in OWL and service descriptions in OWL-S to equivalent PDDL 2.1 [104] description. In particular, it converts service I/O parameters to additional predicates in P/E, respectively. Combined with the original PE of service, these predicates are treated as parts of pre- and post-condition of an action. On top of this, OWLS-XPlan1 applies the enforced hill-climbing search of FF to select the best useful action for reaching the next state with the minimal heuristic goal distance in the graphplan [36]. Another feature of OWLS-XPlan1 is that it can take advantage of any part of a complex action for the current sub-goal of composition. This is inherited from HTN planner. The use of FF planner overcomes the limitation of HTN planner, which requires task specific decomposition rules and methods developed at design time. OWLS-XPlan1 is complete and sound as it inherits the features of FF planner. OWLS-XPlan2 makes it possible to adapt to real-time world state changes at planning time. For this, A new module is introduced to OWLS-XPlan1, which periodically observes state changing events and finds the approximated optimal re-entry point to restart, when the change affects the current patrial plan.

Besides OWLS-XPlan, another great application of FF planner is FF-WSC [146], which offers the centralized state-based automatic composition of semantic web services in OWL-S or WSMO. In FF-WSC, the semantics of services is modeled by beliefs, each referring to a set of models. Each model means a particular situation during the execution of a composition. On top of this, FF-WSC takes advantage of a graphplan, called approximated composition graph (ACG). With ACG, FF-WSC introduces a heuristic function in order to protect the composition from arbitrary augmentation. Unlike OWLS-XPlan2, FF-WSC is not designed for adapting to the dynamics of services.

In addition to OWLS-XPlan, FF-WSC, SHOP2 and the early works Golog [253], OntoMat [4], WSPlan [281], PLCP [287] as well as the ones in the survey [300], there are several representative centralized semantic service composition planners. Similar to OWLS-XPlan, IAAComp [137] translates OWL-S services into PDDL format and accomplishes the composition via the classic STRIPS [94] AI planner. This work does not follow the classic state-based planning, but treats service parameter concepts as states and considers the semantic similarity (called semantic awareness) between concepts during chaining. The similarity measure offers a relaxation mechanism that enables the

Approach	Static	Dynamic	Process -level	Function -level	State-based	Stateless	Service Formalism
OWLS-XPlan (Klusch+ 2005)	√		√		√		OWL-S
OWLS-XPlan2 (Klusch+ 2006)		√	√		√		OWL-S
EHTN-Plan (Gholam+ 2010)	√		√		√		WSMO
IAAComp (Hatzi+ 2012)	√			√	√		OWL-S
PetriComp (Tan+ 2010)	√			√		√	WSDL (OWL for IO)
Golog (McIlrailh+ 2002)	√			√	√		DAML-S
OntoMat (Agarwal+ 2004)	√			√	√		WSDL / DAML-S
SHOP2 (Sirin+ 2004)		√	√		√		OWL-S
WSPlan (Peer+ 2005)	√			√	√		SESMA
PLCP (Pistore+ 2005)	√			√	√		WSDL / BPEL4WS
LL (Rao+ 2006)	√			√	√		DAML-S
METEOR (Wu+ 2007)	√			√		√	SAWSDL
CLM (Lecue+ 2006)	√			√	√		OWL-S / WSMO
GAComp (Weise+ 2008)	√			√		√	OWL-S
WS-QoS (Yan+ 2009)	√			√		√	WSDL (OWL for IO)
QDComp (Lecue+ 2009)	√			√	√		OWL-S / WSMO / SAWSDL
QFComp (Bartalos+ 2010)	√			√		√	OWL-S
PBWSC (sohrabi+ 2010)	√		√		√		OWL-S
IMAComp (Zhang+ 2003)	√			√		√	DAML-S
AutoComp (Kona+ 2007)	√			√		√	OWL-S / WSDL-S / WSMO
BF* (Oh+ 2005)	√			√		√	WSDL (OWL for IO)
ServiceMap (Tan+ 2011)	√			√		√	WSDL (OWL for IO)
PhysarumComp (Zhou+ 2011)	√			√	√		WSDL (OWL for IO)
SWSDCF (Rodriguez-Mier+ 2015)	√			√		√	OWL-S / SAWSDL / WSMO
RuGCo (Aiello+ 2008)	√			√		√	WSDL (OWL for IO)
TAComp (Nam+ 2008)	√		√		√		WSDL (OWL for IO)
Qsynth (Jiang+ 2012)	√			√		√	WSDL / OWL-S / WSLA
TOPKComp (Deng+ 2014)	√			√		√	WSDL (OWL for IO)
FAPComp (Puttonen+ 2013)	√			√	√		OWL-S
FPPN (Cheng+ 2014)	√			√	√		OWL-S
IONumComp (Wang+ 2014)	√			√		√	WSDL
DCComp (Paganelli+ 2013)	√			√	√		SAWSDL
FF-WSC (Hoffmann+ 2008)	√			√	√		OWL-S / WSMO

Fig. 2.5 Representative centralized semantic service composition approaches. These are discussed in Section 2.1.2. As for the distributed variants, we categorize them into the semantic P2P computing field. They are examined in Section 2.3.3 and classified in Figure 2.12.

chaining of services with non-exact matched IO signatures. Due to the use of STRIPS, IAAComp is complete. However, its semantic distance-based parameter concept similarity measure can put difficulty for the execution, since it is not guaranteed that an input parameter a service can use an output of its predecessor service.

PetriComp [348] also takes advantage of the pre-generated graphplan, which is organized as a Petri Net containing all transitions among IO parameter concepts of WSDL services. Given a request, a composition is derived by searching the parameters and pruning the Petri Net. The searching is delegated to a decomposition algorithm that eliminates any conflicting places and in the meanwhile, the pruning process removes all unrelated parts. Since its exploration of transitions, PetriComp is complete. Likewise, the SWSDCF [311] [310] approach accomplishes the stateless IO-level service composition by creating the request-specific layered service dependency graph on-the-fly. Starting from a dummy service that uses the requested input as output, the construction of the graph is done iteratively, until the requested output is achieved. SWSDCF supports to return the optimal solution in terms of concerned criteria, such as plan length, service quality, etc.. Once the whole graph is explored, SWSDCF simplifies it by graph-based backward pruning and dominant service substitution. The highlight of SWSDCF is that it allows for efficient index based service discovery. Another similar approach RuGCo [7] allows for the stateless IO-level composition of WSDL services by a heuristics-based depth first backward search, which produces a service dependency graph at runtime. Its heuristics estimates the likelihood that a branch will lead to a valid composition by counting the number of chain-able services.

QSynth [164] also leverages a pre-created service dependency graph, but in addition is able to adapt to the change of service QoS properties during composition. A re-composition is triggered, once the change of services is known by the service repository. Unlike OWLS-XPlan2, QSynth does not consider the service change at the execution phase and not choose a suitable entry point to re-start. Deng et. al. propose a method TOPKComp [78], which makes effort on composition process parallelization. The entire task is divided into several sub-tasks. Delegated to an agent, each sub-task requires to find critical paths from all input parameters to one output parameter. Inspired by the MapReduce principle, the completed sub-tasks will be merged in order to generate the final solution. Apart from the above, the graph-based composition family also includes the state-based method FAPComp [293] for industrial process composition with minimum plan length, FPPN [60] that models the graphplan via fuzzy petri net, IONumComp [382] that performs a backward search with a heuristics that prefers the paths with smaller number of input and large number of output parameters, TAComp [264] that performs state-based WSDL service composition, which takes use of a boolean satisfactory solver to find the shortest path between initial and goal states, DCComp [274] that is designed for the IO-level composition of SAWSDL services.

SHOP2 is a domain-independent HTN planning system [330]. The highlight of HTN planner is that it creates plan by task decomposition, i.e. the planning system decomposes tasks smaller and smaller subtasks, until primitive tasks are found that can be performed directly. For each nonprimitive task, the planner chooses an applicable method, instantiates it to decompose the task into subtasks, and instantiates methods to decompose the subtasks even further. If the plan later turns out to be infeasible, the planning system will backtrack and try other methods. A problem of HTN-based planner is that they require task specific decomposition rules and methods developed at design time, and therefore do not guarantee to resolve arbitrary planning requests. The HTN planning principle has also been used for the security-aware composition of WSMO services. EHTN-Plan [119] adds abstract syntax to WSMO ontology to model the security factors for both services and request. The composition regarding to this aspect is captured by an extended service selection, which specifies several security related degrees similar to the model proposed in [196]. The HTN-based planner PBWSC [335] allows for the preference biased state-based IOPE-level OWL-S service composition. It combines HTN with an optimization heuristics that can effectively prune the search space based on service quality-based preference. It uses Optimistic Metric Function to estimate the metric value resulting from the current node in a decomposition. In addition, it employs a Lookahead Metric Function, which determines the metric of the best successor to the current node.

Rao et. al. present a Linear Logic based planner (LL)[299] for the composition of services in DAML-S[58]. It uses LL as the internal representation of semantic services. Existing services and user request are transformed into the form of LL sequence. The concrete service composition is done by LL theorem proving. After this, the resulted composite service is transformed back into DAML-S representation. The SAWSDL service planner METEOR-S by Wu et. al. in [391] extends state by adding a set of semantic data types in order to ensure that the data for the input message of an operation is available before the operation is executed. Besides, it provides a revisited version of graphplan algorithm that adapts to the extended states. Lecue et. al. propose casual link matrix (CLM) planner [212] for IOPE-level semantic web service composition. Its essence is to establish casual link matrix between services, which records their chaining relations. Given a goal, CLM planner first determines the positions of requested input, precondition and output, effect in CLM. It then tries to exhaustively find transitions in CLM, until all requested output and effect states get reached. When no transition reaches the requested state, CLM planner returns failure.

IMAComp [410] performs stateless IO-level web service composition. Similar with the graphplan-based variants, IMAComp works with an interface matching graph, in which a service is represented as a node and each of its input (output) parameters is modeled as an incoming (outgoing) link. Before the actual composition process, such a graph is pre-constructed by exploring all the possible parameter dependencies among services. Given a requested service the composition can be captured via Bellman-Ford shortest-path dynamic programming. Another stateless IOPE-level web service composition method [204] AutoComp also relies on the graph-based dependencies between service IOPE signatures. However, such a graph is created on-the-fly for a given request. Starting from the input and precondition of the request service, AutoComp algorithm exhaustively detects all services that can be chained. This process is iteratively performed and terminates when either the request

---

[58] `http://www.daml.org/services/`

service output and effect are achieved, or no service can be chained. Due to its arbitrary chaining, the AutoComp approach has the risk of failure with dead-ends. A similar approach BF* planner overcomes this limitation by an extra retreating process. Within a layered transition graph that is also created for specific query on-the-fly, BF* planner starts with the requested input at the top layer and performs a heuristic-based search. When a dead-end is detected, it forces the composition to retreat to the next upper layer and re-select a node as the destination. ServiceMap [349] performs IO-level stateless service composition based on adjacent matrix, in which transitions of services are computed based on compatible service IO parameter concepts. In principle, ServiceMap is similar with graphplan, but uses different representation for transitions.

Weise et. al. introduce an generic algorithm (GA)-based approach GAComp [387] for the IO-level composition of OWL-S services. Each iteration of GA process focuses on deriving promising mutations as offsprings from which the next iteration can start. The mutation is either the one that does not contain the first service in partial solution or the one that contains the current partial solution and a new promising service. In each iteration, GAComp randomly chooses one from the both ways above and applies it to the partial plan. The highlight of GAComp is that it consider the GA process for service composition, though such an application does not guarantee the completeness. Another interesting method PhysarumComp [415] borrows the idea of Physarum-based maze solving for the IO-level state-based composition of WSDL services over their IO parameter concepts. Physarum-based algorithm finds the shortest path between initial and goal states.

In [398], the authors present WS-QoS for the QoS-driven web service composition at IO-level. WS-QoS works in three steps: (1) breath-first-based chaining over all services to detect the service transition graph; (2) semantic optimization to cut out redundant services; and (3) QoS-driven optimization to get workflow with high overall QoS. The advantage of WS-QoS is that it scales to optimize a workflow based on QoS factors, in terms of average plan execution response time and service quality; while it has a limitation that its breath-first chaining would lose alternative workflows, as it considers considered services in an arbitrary order. That does not guarantee to explore all possibilities. Another QoS-based effort QDComp [213] takes the execution price and response time into account. In particular, it allows user to specify QoS-based constraint in queries and the QDComp approach derives an optimization module that iteratively betters the current composition in terms of the overall semantic link quality. This is done by adopting the hill climbing algorithm, which numerically models the quality and the difference between compositions as heuristics. Bartalos et. al. preset a QoS-aware approach, QFComp [22], for the functional composition of semantic services with IOPE signatures. QFComp takes use of an aggregated QoS score of the current workflow as heuristic value, by which the composition process determines whether or not to keep the current service, once that service can be chained to the current workflow. However, QFComp emphasizes the chaining at IO-level and treats PE implication as a backup. The latter is considered, if the IO chaining gets failed. Besides, QFComp does not fully explore all possible chaining of services, but merely relying a QoS-based heuristics. It is incomplete and has the risk of producing incorrect result.

2.2 P2P Computing

P2P computing has been investigated for resolving computing problems that often request distributed data processing and resource management, real-time collaboration, ad-hoc networking, parallelization, scalability and fault-tolerance. A *P2P network* commonly comprises a set of networked computation devices. Formally, it contains a set $\mathbf{P}$ of peers (nodes) interconnected via a set $\mathbf{C}$ of connections. Identified with an IP address or a public key, each individual peer $p \in \mathbf{P}$ is able to act as both a provider and a consumer of resources [323], including its processing power, disk storage, memory, network bandwidth and data. A connection $c \in \mathbf{C}$ between two peers indicates that (i) each of them knows the network address of the other; (ii) both of them agree on certain rules for exchanging messages, by means of the underlying network protocol (e.g. the IP layer in the OSI model[59]); and (iii) each of them is a (direct) *neighbor* peer of the other. Maintained by a peer, an item i in a P2P network refers to a piece of data plus its descriptions. All items in the network comprise a multiset $\mathbf{I}$.

The idea of P2P computing was popularized by file sharing systems. A common feature of them is that they "*allow millions of Internet users to connect directly, forming groups and collaborating to become user-created search engines, virtual supercomputers and file systems*" [272]. From its application perspective, the P2P family covers the early-stage systems ARPANET [343] and USENET [236], as well as the systems for content sharing (Gnutella, FreeNet, Morpheus, Napster [317], BitTorrent [68], Kad [379] (Kademlia [251]), the Storm botnet [211], Chord [344], Pastry [314], P-Grid [1]), media streaming (TVU Networks[60], CoolStreaming[61], PeerCast[62], FreeCast[63], Osiris[64]), instant messaging (ICQ[65], Fire[66]), and high-availability distributed data storage (Bigtable [55], GFS [118] [100]).

As a basic need for supporting data sharing, the ability to answer user requests is essential to P2P networks. In the terminology of this field, a user request (for data or services) is wrapped in a *query*, which is supposed to be answered collaboratively by peers in the network. A query will be transitively and repeatedly transmitted from one peer to a selected subset of the other peers, until certain criteria are met. This process is called *query routing* and the selected peers are named as *target peers*. The method for determining the target peers is called the *routing strategy*. On receiving a query, a peer tries to find some items locally, that match the query in terms of the semantics involved. This local process is called (local) *item selection*. Once a matching item is found, the peer records the item description (commonly not including item data) to the query and sends that query according to the routing strategy. The combined process including the query routing as well as the local item selection comprises a P2P based search. When receiving a query, a peer is able

---

[59] `http://www.iso.org/iso/catalogue_detail.htm?csnumber=20269`

[60] `http://www.tvupack.com/index_TVUPack.html`

[61] `http://www.coolstreaming.us/`

[62] `http://peercast.sourceforge.net/`

[63] `http://www.freecast.org/`

[64] `http://www.osiris-sps.org/`

[65] `http://www.icq.com/`

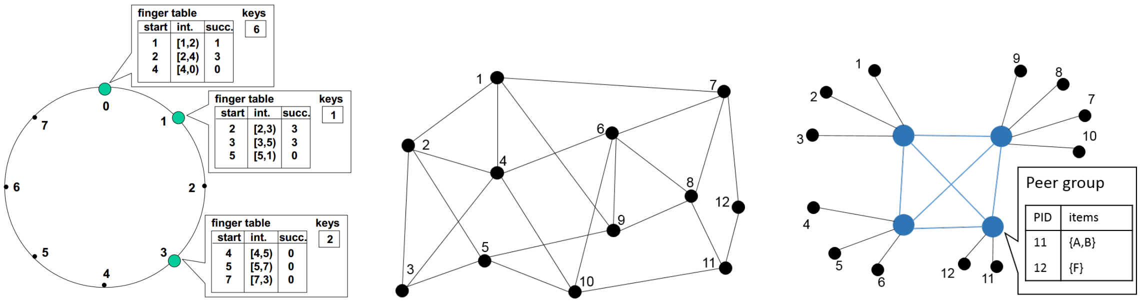
[66] `http://fire.sourceforge.net/`

to observe the content of that query, which includes the wrapped request, selected items and other piggybacked information needed by the customized routing strategy. The observation process is described as *peer local observation*. Many factors and methods can affect the result of a search. One of the most important factors is the network structure. This does not mean the topological features of the network, but the location of items and their indices. Such indices can either be required to be maintained by a certain group of peers, or created by any peer locally, relying on its local observation. In addition, network parameters, such as network connectivity, topology, item distribution and topic distribution, play important roles for the search quality.

In Section 2.2.1 we first introduce the primitives about P2P networks, and this is followed by the discussion of the state-of-the-art works regarding non-semantic based search and data replication in Sections 2.2.2 and 2.2.3, respectively.

2.2.1 Basic Notions

Network structure: For the purpose of sharing files and resources, the locations of data items or their indices are important. Based on the ways how they are organized, P2P networks are classified into three types: structured, unstructured and hybrid. To give an intuition, representative examples from the three kinds of networks are illustrated in Figure 2.6.



(a) Example of structured P2P network: a Chord [344] network with 8 peers. Each peer maintains a hash table that contains the addresses of and the items on a set of remote peers.

(b) Example of unstructured P2P network: a Gnutella [302] network with 12 functionally equivalent peers. Each peer does not have any initial data indexes, but only connects to its direct neighbor peers.

(c) Example of hybrid P2P network: a JXTA [384] network with 4 hub peers. Each hub peer maintains the addresses and the item indexes of its consumer peers. Each hub connects to a set of other hubs. Each consumer only connects to one hub peer.

Fig. 2.6 Examples of P2P network types.

Structured P2P network. Structured P2P networks impose item indices on peers for answering queries. Systems in this type [87] often use a key out of a hash function, for identifying items. Keys are mapped to peers where the corresponding items are maintained. A generic principle, called Distributed Hash Table (DHT), is often considered to distributively store mappings on different peers. In structured P2P networks, query satisfaction (even searching for rare items) can be eventually guaranteed, as long as those items exist in the network. For answering a query, to check the item indices is always the first choice. This is called *lookup*. On top of this, queries can be routed to correct peers and retrieve the desired item. Data indices need to be adjusted for adapting to the network dynamics events, such as peer arrival and departure as well as item addition, deletion and modification. For this, peers need to exchange additional messages for preserving the consistency and completeness of indices. We introduce the following representative structured P2P networks.

In a CAN [303] network, peers and items are mapped to a multi-dimensional virtual space via a uniform hashing function. One execution of the function yields the value on one dimension. The coordinate space is dynamically partitioned among all peers. Each peer maintains its distinct zone within the overall space. An item with a key is maintained by the peer that corresponds to the zone to which that key is mapped. Two neighboring peers have pointers that mutually refers to each other if their zones share a common side. The lookup process is done collaboratively by adjacent peers in the space. That is, each peer routes a query to the closest neighboring zone to the destination point. The arrival of a peer causes in the split of a zone in multi-dimensions. For this, the new peer learns its neighbors from the maintainer peer of the old splitting zone, and also informs the neighbors about its arrival. In contrast, the departure of a peer causes in the merge of zones. The leaving peer informs one of its neighbor to merge the zone. A zone-balancing process is additionally employed in order to reconfigure the zones after a number of arrival and departure events. The advantage of CAN network is that is provides a sub-linear search mechanism, as it enforces strict constraints on network evolution and resource placement. The length of a lookup path is $dlogN^{(1/d)}$, where d and N are the number of dimensions, respectively the size of the identifier space. The main limitation of CAN network is also caused by its strictly enforced constraints. The number of messages for maintaining the network and item indices can be large, if the network dynamics event happens frequently.

The Chord [344] network hires a circular identifying space of size N. Each peer p has a pointer to the first successor/predecessor peer clockwise in the identifying space. Besides, each peer maintains $log(N)$ pointers, called fingers. The finger set F_p of peer p is defined as $F_p = \{successor(p+2^{j+1})\}$, $1 \leq j \leq log(N)$. The pointers in a finger set are chosen for partitioning the space into two halves. One out of the two is further partitioned into two quarters, etc.. If items and peers can be identified via positive integer numbers. An item i is stored at the first successor peer which identifier is not larger than the id of i. Therefore, the searching and insertion of items are reduced to the search for the successor of item identifiers. By means of its network partitioning and finger sets, each hop halves the distance to the desired id. The arrival of a new peer p to a Chord network is managed by the periodical stabilization algorithm: first, p find its successor and predecessor by a lookup process; second, p copies the routing table from its successor and adjusts this table via lookups. The contacted peers adjust their tables to adapt to the arrival of p; finally, p requires the transfer of items which id is not larger than the identifier of p. The leave of a peer p leads to the transfer of its items to the successor peer. Via stabilization algorithm, the finger tables of those pointed peers will be correct. Similar with CAN network, the main advantage of Chord is that its supports a sub-linear search mechanism, i.e. the number of needed hops for lookup is $\boldsymbol{O}(logN)$. The disadvantage of Chord network is also the large number of extra messages that are used to adjust the item indices, when peer joins and departs frequently. In this case, the replication of routing table for peer initialization can consequently leads to a large amount of network traffic.

A Pastry network [314] can be built based on the Plaxton [289] mesh. Similar to Chord, Pastry employs a circular identifying space. Each peer maintains a leaf set containing $\frac{L}{2}$ successors and $\frac{L}{2}$ predecessors. In addition, it maintains a set of peers that are close to p in terms of network delay, for the purpose of keeping locality properties. Further, p maintains a routing table containing $\lceil log_{2^b}(N) \rceil$ rows and $2^b - 1$ columns. N is the identifier space and L, b are positive integer system parameters. Identifier of peer is a digital string of based 2^b. The j-th row in table contains the addresses of the peers which id differs with p's in the j-th digit. An item is maintained by the peer whose id shares the longest prefix with the id of that item. To search for an item by id, p checks within the leaf set. If the item identifier is covered, p routes the request to that covering peer. If the leaf set does not help, p routes the request to a peer in its table, which id is the closest to requested item id. The upper bound of routing path length is $log_{2^b}(N)$. Arrival of peer p leads to a series of processes. Let p' the peer to which p initially contacted: p copies the neighbor set from p'; p finds from the routing table of p' the closest peer p^* to itself; p subsequently retrieves the j-th row from the j-th peer in the path from p' to p^*. These data forms the routing table of p; further, p inherits the leaf set from p^*; finally, p informs all peers in the above sets about its arrival. The departure of p is totally passive. Instead of causing extra communication overhead, peers adjust the absence of p the on the chance of messaging failure. Pastry network can perform fast lookup process, which is sub-linear with the identifier space. However, extra communication overhead is caused for maintaining the network and item indices when network dynamics happens, though it hires a passive policy for peer departure.

Like Pastry, Kad [379] (Kademlia [251]) has the similar partitioning over the identifying space. It is organized via a binary tree. The position of a peer p in this tree refers to the shortest unique prefix of p's identifier. Normally, a peer in Kad network has a 128-bit identifier, which is stored at p even after it departs from the network and re-used after its re-arrival. p maintains pointers to $log_2(N)$ subtrees where N is the identifier space. The j-th subtree is determined by a prefix that is closest to the first j bits of p's identifier. Kad hires a publish and retrieval mechanism to enable its mappings the keywords (in resource file names) to values. Each keyword of a file and the file itself are hashed. Such information about the keyword, file, and the address of its owner peer is stored at those peers in DHT whose identifers are the closest to the hash values. Therefore, the hash value of an item is maintained by a peer p if it is closest to the identifer of p. Furthermore, Kad allows to store a hash value duplicately in up to k peers ($k > 0, k \in \mathbb{N}$), where k is a network parameter. This setting increases the data availability in a Kad network where the network dynamics event happens frequently. On top of this, the routing of a query is performed by successive selection of subtrees according to the hash value of item. To join the network, a new peer p finds its closest peer p' (via a bootstrap node) and asks p' to help to fill its routing table. This is done by querying about nodes in different subtrees. The advantage of Kad (Kademlia) network is that its lookup process spends at most $\boldsymbol{O}(N)$ hops. Besides, the duplication of the same hash value in the network increases the data availability. Like the structured P2P network variants above, the limitation of Kad is that the extra network traffic overhead gets very large, when peers join and leave the network frequently.

BitTorrent [68] is one of the most common protocols for transferring large files in Internet. Differing from other structured P2P protocols, BitTorrent can reduce the server and network impact of distributed large files, as it allows user clients to download and upload certain pieces of data from each other simultaneously. A peer that is sharing a file firstly treats the file as a number of pieces with the same sizes. It subsequently creates a SHA-1 hash value for each piece, which is recorded in torrent file. Torrent files are typically published on Web sites and registered with at least one tracker. The latter maintains lists of the clients that contains pieces of seeded file. BitTorrent protocol does not impose any rule for indexing torrent files. A user search for certain file is delegated to the search on the Web. Once user client peer has downloaded torrent file, it knows the partitioning of the desired file and retrieves the participant client addresses where the corresponding data pieces are maintained. Finally, the requester peer downloads the pieces of data of desired file from multiple sources. The limitation of this architecture lies in that the failure of tracker peer would results in the dissatisfaction of each user request. To cope with this, BitTorrent protocol supports distributed sloppy hash table for storing peer contact information for "trackless" torrents, in which each peer can act as a tracker.

Unstructured P2P network. Peers in structured P2P networks commonly maintain their routing tables as well as the mappings between peers and items. On one hand, this makes the routing of query to be more precise and faster; on the other hand, the network traffic overhead increases largely against the network dynamics events. Besides, the structured P2P systems tend to have hot spots and imbalanced message loads. In contrast, the principle of unstructured P2P network does not impose any content-based overlay to a network by design. In an unstructured network, there does not exist explicit global data indices. Although the network topology might have certain properties, the insertion and deletion of items are not based on any knowledge of the topology, compared to the structured P2P systems. Apart from this, it behaves more robust against network dynamics.

The simple initial setting of unstructured P2P networks can not guarantee the satisfaction of search. For this, the typical way is to ask neighbor peers epidemically. The key question is to how many and which peers should be asked. In this perspective, there are two types of solutions, *flooding*- and *k-walkers* based search strategies. For both of them, time-to-live (TTL) mechanism is often used particularly in those systems with large scale networks. It restricts that how many times a query can be routed, in order to control the network traffic overhead as well as the query response time. In practice, TTL is a positive integer value attached to a query. Once receiving a query, a peer performs item selection and decreases TTL by 1. It forwards this query, if its TTL is still larger than 0. It force a query to backtrack, if its TTL is 0.

The basic principle of flooding enables each peer, under a TTL restriction, to broadcast a query to all of its direct neighbors, except the one from which this query is received. This process is repeated until all reachable peers within TTL have been contacted. The advantage of flooding is that it can definitely find the desired items if they are located at those peers reachable in TTL hops. However, this can not be achieved for free. A major problem introduced by flooding is the large network traffic overhead due to the duplicated messages [237]. This gets worse in highly connected networks. The extreme case happens, for instance, in the network with complete graph topology. Another issue of flooding is the choice of proper initial TTL value. If TTL is too high, the unnecessary message will cause large network traffic overhead; while if is small, relevant items might be missed, thought it exists somewhere in the network. Unfortunately, the choice of this value appears different in systems with various network parameters, like scales, topologies and connectivities.

An alternative of flooding is the random walker query routing strategy. It allows a user to delegate its request to a random walker, which can have relatively larger initial TTL value T ($T > 0, T \in \mathbb{N}$) and traverses a sequence of peers. Particularly, it restricts a peer to forward a query to one random direct neighbor. Although routing a query in this way can reduce the duplicated messages of flooding, the system has the risk of increasing the query response time. To cope with this, the requester peer can alternatively route a query to k ($k > 1, k \in \mathbb{N}$) random neighbors. Each of them is a random walker with the initial TTL value $\lceil \frac{T}{k} \rceil$. This setting can, to certain extent, ensures the walkers to reach similar number of peers with one-walker approach yet in a shorter query response time. In k-random walker routing strategy, the choice of the successor peer of each walker is totally random. It is easy to realize in practice, but an improper selection by chance might lead to the lost of relevant results. To handle this, a common solution is to construct a content-based overlay network, in which peers references each other about their items and interests. We discuss this in Section 2.2.2.

Another important issue of the performance of a walkers-based search lies in the termination criteria of a walker. Besides the exhaustion of TTL, a walker can be forced to backtrack if either a matched item is found or all the neighbor peers has been traversed by the same walker (called *blind path*). Both of them can happen, though the TTL is still larger than 0. The former condition is used in identifier based retrieval systems, e.g. [237] [69]. However, it is worth to consider to discard this criteria in non-identifier based systems, as their query-item non-binary similarities are determined based on descriptions (e.g. text description of query and item). The exact matching cases in those systems appear minor. The blind path criteria has been applied implicitly to nearly all the walkers-based systems, except those variants of Freenet [65], where a peer can be traversed again if that is needed in order to reach a new peer. In the following, we briefly present the representative unstructured P2P networks.

Gnutella [302] is a popular unstructured P2P-based file sharing protocol. In its early version 0.4[67], it applied the TTL restricted pure flooding principle for maintaining the network and search. In this work, a set of descriptors used for exchanging data amongst participants. A message with PING descriptor is concerned for actively discovering hosts in the network. On receiving a PING message, a peer responses to the PING message sender with a PONG message. A PONG message

---

[67] `http://rfc-gnutella.sourceforge.net/developer/stable/`

includes the Gnutella peer network address and other information with respect to the maintained data contents. Gnutella peer attaches a QUERY descriptor to a message, if the latter is aimed at searching the network. Any responder to this query will reply to the requester peer with a QUERYHIT message, as long as a matched item has been found locally. In Gnutella, a peer connects itself to the network by establishing a connection with another on-line peer in the network. Commonly, the address is configured or pre-cached in advance. Once contacted, a TCP connection between them is created. At this moment, the both sides mutually regard the other as on-line. To maintain the local view on network topology, each peer is allowed to send PING messages to probe the network and reply with PONG message, under a traffic limitation policy. Relying on the classic flooding routing strategy, Gnutella v0.4- based systems can expect a relatively high search precision but at a price of heavy network traffic overhead. This situation has been changed in the version v0.6 and v2.0, which introduce super peers into network for more efficient query routing. These variants will be studied in the following hybrid P2P network section.

Freenet [65] is a distributed file sharing, storage and caching system. Each participant peer in Freenet is required to allocate some amount of disk space to store data. An item in this system is split into blocks and they can be maintained by a set of peers. The original publishing peer of an item is free to leave the network, once the distribution of shared item blocks is finished. The merits of Freenet are high availability and anonymity; while its main limitation lies in that chunks of some old file can be deleted from a peer if its allowed storage space gets used up. In Freenet system, the allocated storage space contains files with associated keys, as well as a routing table that maps keys to known files on remote peers. The search for a file becomes the search for a key. In addition, a heuristic walker based search is used. Each peer has location information that is a number between 0 and 1. When a key is requested by a query, the peer will check locally. If the key is not found, the query is routed to the peer whose location is the closest to that key. The routing terminates, if the desired data is found or TTL has been exhausted. Once the data is found, it will be cached on the peers in query path. Initially, the location information of peers are distributed randomly. Due to the location swapping and path folding processes at each peer, peers topologically closer to each other tend to have nearer values of locations. Besides, data files with similar keys tend to be stored on closer peers. This makes the network overlay to be self-organizable, where peers holding similar data items are closer in key space.

Hybrid P2P network. Hybrid P2P network is a combination of unstructured and structured (server-client) P2P models. Commonly, a hybrid P2P network employs a set of servers, that maintain the topology information about the certain region in the network and support peers to find each other [342]. A hybrid P2P network can contain two types of peers: directory peer and leaf peer [233]. Theoretically, a directory peer, also called *hub*, does not maintain item indices but provides services for maintaining and serving those requests that ask for the information about regional network topology. Commonly, multiple directory peers can work collaboratively in hybrid network and each of them responses to manage the topology information for a part of the whole network. A leaf peer that models an individual P2P client is able to issue queries for the desired data or share local resources [400]. However, the functionalities of hub peer in the actual systems often cover both the topology information provider and the coordinator for query routing, e.g. the ultra-peer in Gnutella v0.6. In the following, representative hybrid P2P network based systems are introduced:

JXTA Search [384] network provides "a common distributed query mechanism for devices from Web servers to small computers". The participants in JXTA query and serve via Query Routing Protocol (QRT) that encodes the request and response messages in XML-based language. A JXTA based system consists of three kinds of peers: search information provider peer, consumer peer and hub peer. The search information provider peer provides the information about data for any request formatted in JXTA QRT. Differing with other P2P systems, JXTA allows a Web server to work as a information provider. A consumer peer represents a user that makes requests in QRT. A consumer peer can be a user client or Web site with HTTP client interface to JXTA Search network. A hub peer responses to the efficient query routing between consumers and providers in a JXTA Search network. It works as a media between consumers and providers. A provider peer can register the meta description of itself on hub peers and wait for requests; while a consumer peer queries the network by sending its request to a hub peer and waiting for response. In JXTA, the role of a participant can be multiple. Each peer can, at the same time, work as a provider, consumer, hub or combination of them. In general, the search in JXTA comprises the following stages: (i) a consumer sends its request to JXTA Search network via the nearest hub. (ii) The latter sends the request to those providers that are able to serve the request based on their registered meta-data. (iii) On receiving a query message, a provider finds the matched data locally and responses to the hub. (iv) Finally, the hub replies to the consumer peer. Particularly, JXTA Search is intended to cover two search types: wide and deep. A wide search scales to query multiple hubs, which covers a various of distributed devices, such as PC, smartphone, etc.; while a deep search focuses on finding results from Web servers with large databases.

Conceptually, each peer in JXTA abstracts three layers: core layer, services layer and applications layer. The components in core layer implements the JXTA protocol, which realizes the basic P2P operations, including peer identifier management, boot-strapping for peer arrival, one-way messaging, etc.. Services layer encapsulates more common functionalities that a P2P system might use, e.g. the basic routing skeleton and item selection interface that are necessary for P2P search. In particular, this layer scales to the functionalities of Sun Content Management System (CMS)[68] that provides methods for peers download data content, once it is located. The applications layer supports the run time environment of customized P2P application. Together with the lower two layers, it provides the a package of functions that can be invoked by user application.

The advantage of JXTA is that it allows, by the deep search, to access the rich contents, which have been collected and well-maintained by Web sites. Hiring database technology, the latter com-

---

[68] `http://docs.oracle.com/cd/E19316-01/820-7054/ghpfh/index.html`

monly have great power for indexing data and serving queries. This enlarges the sight of knowledge to the participants in a P2P network. Another merit of JXTA lies in its way for exchanging messages. The employment of XML provides a platform independent format for the structured data. However, this also leads to some side effects. Large network overhead can be caused due to the XML tags, which might be duplicately exchanged. Another limitation of JXTA is that it has its own specification for encoding messages. Despite using XML, it is unfortunately incompatible with the WSDL Web service message. This blocks the inter-accessibility between them. Although its inter-operability and compatibility with Web service have been considered in JXTA 2.0[69], the functionality of actual service invocation is still missed. This problem is targeted by a third party library, called JXTA & Web Services Gateway[70], which aims at making applications in both environments to be able to talk to each other in standard SOAP[71] message.

In comparison with Gnutella v0.4 that is built on pure unstructured P2P network model, Gnutella v0.6[72] made an update: Peers with weak computational power/ bandwidth do not belong to Gnutella overlay any further, even though they can access the overlay. Leaf and ultra peers are distinguished by means of their computation capacities and knowledge to file resources. Each ultra peer works as a proxy for the management of leaf peers. It shields the leaf peers from the query routing. That is, the QUERY and PING messages are transmitted between only the ultra peers. An ultra peer connects to its managed leaf peers as well as a set of other ultra peers. Leaf peers do not accept Gnutella connections, but rather opens commonly 1 connection to its manager ultra peer. This type of overlay in v0.6 is similar to the organization of Internet, where backbone routers sends messages on large bandwidth connections. The query routing in v0.6 is based on query routing table [385] (QRT). An element in QRT is a bit vector composed by slots. Each slot can record the encoded a keyword hash value of a shared file. On top of this, the finding of requested items can be reduced as a comparison between bit vectors. Each leaf peer creates its QRT and sends it to its manager ultra peer. The latter builds a local QRT by aggregating the ones from leaf peers. On receiving a query, an ultra peer looks up in its QRT whether the desired item is offered by its leaf peers. The request will be forwarded to those leaf peers if they are found; ultra peer forwards the query to other ultra peers. Both of the forwarding processes consume query TTL. A query is no longer to be forwarded as soon as its TTL has been exhausted. When a leaf peer receives a query, it checks whether the needed file is maintained locally. It replies to the precedent ultra peer and the latter replies the predecessor as well. The replying processes terminate until the requester peer is informed. Gnutella v 2[73] uses a walker based system, in which an ultra peer sends one request to a known ultra peer at a time, until the resource is found. The advantage of this change lie in that the system is more efficient in terms of network traffic overhead; while a limitation of it is that the query response time can be longer.

---

[69] `https://jxta.kenai.com/Specifications/JXTA_Protocols_2_0.html`

[70] `http://j-x-w-s--gw.sourceforge.net/introduction.htm`

[71] `http://www.w3.org/TR/soap/`

[72] `http://rfc-gnutella.sourceforge.net/src/rfc-0_6-draft.html`

[73] `http://g2.doxu.org/index.php?title=Main_Page`

Small World Bee (SWB) [224] proposes a flooding-based search in super peer-based P2P networks. Each super peer leads a group of peers that have similar content. The similarity is encoded as a QRT relying on the appearance of keywords. When searching for items, the originator peer floods the request to its known super peers. Each super peer compares its QRT with the request. As long as they are similar, the super peer issues a flood to its known leaf peers. Although this method can considerably decrease the network traffic, the finding of super peers works under an assumption that there exists a global cache recording the information of them. Such cache might become a bottleneck to a system if peers arrive and leaf frequently. Moreover, the idea of QRT is limited with respect to the keyword space. The work DiffSearch [376] leverages the power of super peer as well. It gives higher querying priority to peers with high reply capabilities. The super peers in DiffSearch are meant with the content-rich peers, which in addition maintain the data indices of leaf peers. For searching data, requester peer first sends query to its known super peers. If this try can not find matched result, the second round floods the entire network. DiffSearch largely reduces the search space by means of its content-based network overlay formed by super peers but at a price that super peers can become hot spots.

Network parameters: The performance of search in P2P networks is closely related to network parameters, such as averaged connection degree, topology, item distribution and topic distribution. In the following, they will be briefly introduced.

Averaged connection degree. P2P networks can be modeled as undirected graphs. Each node in the latter corresponds to a peer in the network, while an edge in such a graph models the connection between two peers. The connectivity features of P2P networks can be modeled by graph connectivity properties. Let $|\mathbf{P}|$ be the number of peers in a network, $|\mathbf{C}|$ the number of connections. The averaged connection degree dg of a P2P network is defined as:

$$dg = \frac{2|\mathbf{C}|}{|\mathbf{P}|} \tag{2.1}$$

Instead of being set explicitly, this value is often determined by the parameters of the specific underlying graph of a network topology. In the following, we introduce the network topologies concerned.

Network topology. Network topology is the arrangement of links and nodes of a computer network [125]. In the context of P2P networks, it contains two layers of meanings: *P2P underlay* (physical topology) and *P2P overlay* (logical topology) [160]. The former refers to the shape of the cabling layout that is used to interlink physical networking devices. In this layer, the capabilities of the networking devices, data transmission media, telecommunication circuits, etc. are considered; while the latter layer concerns how the signals (data) act on the network media via logical links regardless of their physical interconnections between devices. To a P2P network, its P2P overlay is not necessarily the same as its P2P underlay. A direct connection between two peers in a P2P overlay might be realized by a path of sequentially connected physical devices in the underlay. In this thesis, the network topology stands for the topological shape of the peers and connections in a P2P overlay.

The category of basic network topology [31] includes point-to-point, bus, star, circle, mesh and tree topologies. In the context of this thesis, the underlying network topologies are the more complex hybrid variants. In the following, we briefly introduce the random graph and random power-law graph based network topologies. They are used in the experimental evaluation sections of Chapters 3, 4 and 5.

A random graph is formed by adding connections at random among a set of initially isolated peers. The representative model of random graph was proposed by Edgar Gilbert [121], denoted as $G(|\mathbf{P}|, \rho)$, where every possible connection occurs with probability $\rho \in (0,1)$. In a random power-law graph [90] based network topology, the fraction $P(m)$ of each peer having m ($m > 0, m \in \mathbb{N}$) connections to others is determined by $P(m) = C * m^{-\beta}$, where β ($\beta > 1, \beta \in \mathbb{R}$) is called the skew value of the power-law graph and C is the factor for normalization. This model indicates that the degrees of peers follow a power-law distribution. It is well-known as a good option to model social networks.

Item distribution. Item distribution in our context refers to the initial arrangement of items over peers. It is modeled by the probability $P(i, p)$ of a peer p maintaining an item i at the initial stage of a network. The most prominent distribution is the uniformly random distribution. It designates each peer with equivalent probability $P(i, p) = \frac{1}{|\mathbf{P}|}$ to maintain item i.

Topic distribution. Unlike the item distribution above, topic distribution describes which items are more popular or more frequently requested by peers in a network. The simplest model for this is the uniformly random distribution. It means that each item has an equal chance of being requested by a peer. However, it is common sense that an individual appears more interested in a specific set of topics than others, which is evidenced by the statistics of the cached Web pages in servers [3]. This implies a discrete Zipf distribution of interest over topics. To model this, let us assume that the items in a network can be clustered into a set T of topics. Given a bijection $F : T \leftrightarrow \mathbb{N}^*$ between topics T and a set $\mathbb{N}^*$ of positive continuous integer values starting from 1, each peer requests some item in a topic $t \in T$ with the probability $P(t) = a \cdot F(t)^{-\beta}$, where a ($a > 0$, $a \in \mathbb{R}$) is the normalization factor; β ($\beta > 1, \beta \in \mathbb{R}$) is the skew value.

2.2.2 Search

Routing strategy: Query routing is essential to P2P search. In structured P2P networks, protocols guarantee that data indices are properly maintained by a certain group of peers, which in addition offers mechanisms to tackle the data-index inconsistency problem caused by network dynamics. Despite possible item rarity, a query will be eventually routed to the item maintainer peer, if that item exists in the network. As the query routing strategies of the representative structured P2P networks have been discussed in Section 2.2.1, in the following, we focus on the query routing strategies of unstructured P2P networks.

An unstructured P2P network does not impose any initial indices about data on peers. Each peer determines the location of items in the network by its local observation or prior training. Such a process happens during either the system runtime or initial stage. With the knowledge they have gained, peers form a content-based overlay network (CON). This overlay is an abstract layer often distributively built on top of the P2P overlay. It concerns the locations of domain specific data items and their meanings, such as topics, interest and expertise. Each peer maintains the knowledge about not only itself, but also a set of other peers. We call an unstructured P2P-based search strategy *informed search*, if it enables peers to construct and use CON knowledge for routing queries. For constructing CON, some approaches allow peers to cluster the observed items and queries or their maintainers and requesters, in order to direct query routing. We call this feature as *cluster of queries or items*.

In this section, we discuss the representative non-semantic based routing strategies in unstructured P2P networks. In Figure 2.7, we classify them together with the non-semantic based variants in structured or hybrid P2P networks. As for the semantic-based P2P search strategies, we categorize them into the semantic P2P computing field. We examine and classify them in Section 2.3.1.

Flooding based approaches. In general, the flooding-based unstructured P2P searches often restrict and select the target peers when forwarding a query, in order to achieve better tradeoff between search performance and network traffic. Commonly, such choice about target peers is made relying on the peer local knowledge, which is learned and updated based on the runtime observation or in prior training. Such knowledge and its maintainer peers form a CON that often considerably helps the query routing. Before delving into this paradigm, the basic flooding search variants, like N restricted flooding and expanding rings, will be discussed. These works give the evidence about the importance of CON from the opposite side.

N restricted flooding [174] [98] [81] allows a requester peer to send query to at most N ($N \in \mathbb{N}^+$) random neighbors. On receiving a query, each peer, besides its local item selection, forwards the query to at most N neighbors. These peers do not include the one from which it receives the query. This idea can limit the network traffic overhead of search via a proper choice of the parameter N. However, it in the meanwhile lowers the search performance due to the random choice of targets.

In an unstructured P2P network, the depth of flooding is also an crucial factor for network traffic overhead. The farther a query is propagated, the more peers can be accessed. This holds in both the classic and N restricted flooding. The expanding ring-based efforts [237] [363] [401] investigate this aspect. Overall, they allow requester peer to flood multiple times with increasing TTL values. When the query with smaller TTL fails to find item, another query is issued with larger TTL. This process repeats until the item is found, or the upper bound of TTL is reached. On average, this idea may work well in those unstructured P2P networks with dense items, where the chance of finding

Approach	Structured P2P	Unstructured P2P			Hybrid P2P	Informed search with CON	Cluster of queries or items
		Flooding based	K-walkers based	Flooding & K-walkers hybrid			
CAN (Ratnasamy+ 2001)	√						
Chord (Stoica+ 2001)	√						
Pastry (Rowstron+ 2001)	√						
Kad / Kademlia (Maymounkov+ 2002	√						
BitTorrent (Cohen 2008)	√						
Gnutella v0.4 (GDF 2000)		√					
Morpheus (Wang+ 2007)		√					
N restricted flooding (Kalogeraki+ 2002, Fletcher+ 2005, Dorrigiv+ 2007)		√					
Expanding ring (Lv+ 2002, Tsoumakos+ 2006, Yang+ 2002)		√					
BER (Pu+ 2009)		√					
FastFlood (Barjini+ 2011)		√					
PER (Endo+ 2014)		√					
AntSearch (Kai-Hsiang+ 2003)		√				√	
PF (Margariti+ 2014)		√				√	
DBFS (Yang+ 2002)		√				√	
LSM (Kalogeraki+ 2002)		√				√	
RI (Crespo+ 2002)		√				√	
Freenet (Clarke+ 2002),			√				
2LevelRW (Jawahar+ 2004)			√				
PKW (Tsoumakos+ 2006)			√			√	
PopKW (Bisnik+ 2007)			√			√	
Pwalk (Zhuge+ 2005)			√			√	
GIA (Chawathe+ 2003)			√			√	
APS (Tsoumakos+ 2003a)			√			√	
BuSIS (Hu+ 2009)			√			√	
ES (Hota+ 2013)			√			√	
MPRD (Al-Oqily+ 2014)			√			√	
NVRWK (Ghorbani+ 2013)			√			√	
AQFS (Durr+ 2012)			√			√	√
Agora (Deconinck+ 2009)			√			√	√
F&KW (Dorrigiv+ 2007)				√			
PercolationSearch (Sarshar+ 2004)				√			
CloneRW + DS (Leu+ 2014)				√			
SWB (Liang+ 2011)					√	√	
DiffSearch (Wang+ 2007)					√	√	
Gnutella v0.6 (Klingberg+ 2002)					√		
Gnutella v2 (Stokes+ 2002)					√		
JXTA (Waterhouse+ 2001)					√		

Fig. 2.7 Representative non-semantic based search strategies in P2P networks. These strategies are discussed in Section 2.2.2. As for the semantic-based P2P search variants, we categorize them into the semantic P2P computing field. They are studied in Section 2.3.1 and classified in Figure 2.9.

desired item by flooding in a short range is large. Nonetheless, in the networks with rare items, this approach has the risk of tremendously increasing network traffic, as a query might access the a same group of peers by multiple times. To resolve this problem, Pu et. al introduce the blocking expanding ring search (BER) [291]. Instead of re-flooding over those peers that has been checked, a new flood starts from the peers on the last ring. In [89] Endo et. al. proposes a method (PER) to determine the probability of continuously forwarding query from the peers on the last ring. Barjini et. al. proposes the FastFlood approach [21] that combine the standard flooding with selective flooding (called teeming) for the purpose of control duplicated message. A search in this scheme contains two steps. First, the requester issues a standard flooding with a small initial TTL. Once a optimum threshold is reached, the standard flooding is switched to the selective flooding.

Besides flooding network with TTL restriction, a peer can make decision about whether to forward a query, based on its local knowledge. A probabilistic flooding method PF [242] [243] follows this idea, which enables each peer to estimate a probability regarding whether to forward a query to its neighbors. Computed by means of the local observation on the item duplication rate and network topology, this value indicates to what extent the desired item has not been found on other peers. This knowledge forms a CON, is distributively maintained by peers in the network. This approach works under an assumption that the connectivity degrees of peers are similar. However, this is not true in random power law graph based networks, where most peers have degree 1 but few peers have large number of neighbors. In this situation, the peer local prediction about the item duplication rate on other peers may become imprecise.

A study on flooding-based file sharing system shows that more than half of the peers in an unstructured P2P network usually do not share files [298]. These peers are flooded but without any contribution to answer a query. This costs unnecessary network traffic overhead. AntSearch [172] is derived for attacking this problem. Its basic idea is to send query to a set of neighbors which share files. For this purpose, a peer observes each query it has processed and summarizes a statistic value for each of its direct neighbor peer. Such knowledge comprises a simple CON. When searching for some file, requester peer first issues a probing flood with small TTL, in order to obtain the statistic values on the peers nearby. Subsequently, it computes a suitable TTL for each neighbor and forwards query to it. On receiving a query, each peer behaves similarly. AntSearch can effectively reduce the network traffic overhead but at a price of losing traffic load balance.

DBFS [401] enables each peer to maintain information of neighbors, which comprises the number of answered queries and the network connection latency. This statistical knowledge forms a CON for query routing. Aiming at enhancing Gnutella search, the work LSM [174] follows the same way of building CON. Each peer in LSM keeps a ranked list of the real-time profiles for its neighbors. Query is selectively forwarded to those peers with higher ranks. RI [70] performs selective flooding as well. In comparison with DBFS and LSM, a peer in RI maintains the knowledge about not only its direct neighbors but the peers within a range. The item indices of indirect neighbor is built by three optional aggregation mechanisms: Compound RI, Hop-count RI and Exponential RI. When routing a query, each peer estimates a ranked list of neighbor peers that are helpful to answer this query and subsequently sends it to each of them. However, there can exist inconsistency when network dynamics happens. Extra messages are needed for updating indices.

K-walkers based approaches. In comparison with the flooding-based P2P search, k-walkers based variants in general cost less network traffic overhead. For each walker, the task of routing focuses on the selection of the next target peer. In unstructured P2P networks, each peer accomplishes this work often based on its local knowledge to the network topology, item location and peer clusters. This knowledge and its maintainer peers forms a CON, which helps to conduct query routing for search. In the following, representative k-walkers based methods for the search in unstructured P2P networks will be discussed. This discussion starts with a simple variant of k-random search.

The work two-level k-random walkers (2LevelRW) [161] applies k-random search twice with different initial TTL value. For searching items, a requester peer issues k-random walkers with TTL T_1. If the request is not satisfied, a k-random search with TTL T_2 will be issued by the ending peer of each walker in the former search. The advantage of this work lies in that it costs less network traffic given same total number of walkers, in case that the density of item is high in the network. However, this search routes queries totally by random, which does not incorporate any information of peer knowledge.

Compared to the k-random search and 2LevelRW, GIA approach [57] employs a TTL-restricted bias walker, which tends to forward a query to the immediate neighbor with maximum capacity. The capacity refers to the amount of files that a peer shares. This information is achieved during the peer arrival stage. Indices of maintained items are exchanged between the two peers that are going to establish a connection. GIA leverages a token-based flow control mechanism in order to prevent a peer from being overloaded. A side effect of this control is that it can yields delay on query answering, as a query is buffered locally by requester peer if it has no token left.

Instead of randomly routing a query, each peer with the adaptive probabilistic search [361] (APS) enables each peer to select a neighbor for the routing based on its locally maintained probability table. The latter records a probability value for each neighbor on each known item. For routing a walker, a peer can follow two strategies: optimistic and pessimistic. In pessimistic (optimistic) strategy, if a walker has found (not found) the desired item, the probabilistic value is increased (decreased) by the peer when it receives the backtracking query. A limitation of APS is the cold-start issue. If there is few probabilistic information can be used at system initial stage, APS degrades to the k-random search, as the initial probabilistic values of neighbors are equal. A variant [153] of APS, called BuSIS, presents a probabilistic query routing method based on the knowledge about item popularity, connection quality and network status. This work introduces the notion of budget, which is required to be pre-paid by peers for search, consumed by routing and item selection processes. Better link quality deserves higher charge. The advantage of this work lies in that it models the P2P system in a more realistic way. Nonetheless, it does not provide a mechanism for resolving possible security issues when working with budget.

Another variant of APS, PKW [363], employs a probabilistic table on each peer. It is updated after each query relying on the requester's feedback. A simple statistical value is increased (decreased), if the feedback is positive (negative) or query is (not) satisfied. This information is achieved via extra messages along the invert paths. Inherited from APS, cold start problem remains in PKW. A similar approach PopKW [33] also leverages the user feedbacks and it allows each peer to select proper value of k and initial TTL. For this, each peer maintains a popularity table for its known items with its popularity value for computing k and TTL. These values are updated by means of the analysis on the user feedback to the previous queries. When searching an item with popularity p, a specific random search is issued based on the estimated k and TTL values. Intuitively, PopKW search features to learn the relation between item popularity and the parameters of k-random search. For popular items, smaller parameter values can be chosen; while for rare items, larger values are going to be applied. Besides the cold starting issue, the quality of search is subject to its random routing strategy. The PWalk [419] search makes use of user feedback as well. It introduces a probabilistic model that allows each peer to correlate the user feedback on specific item with the trust estimation of each neighbor. Such information about trust is in addition sorted. As a consequence, highly ranked neighbors are prone to be selected as the routing targets. The main limitation of PWalk is the hot spots. NVRWK [120] also works with probabilistic routing table. Used by each peer, it records a successful rate for each of its neighbor. Such a value is learned on-the-fly based on the "hit"-information of historical queries. On issuing a request, the requester peer delegates it to the top-k promising neighbors, from which the queries in the past get highest successful rates. Each neighbor issues a random walker searching for the results. Although NVRWK enables peers to learn knowledge from the queries in the past, the use of random walker may lead to inaccuracy.

The philosophy of clustering peers based on their interests is often applied in the k-walkers based search strategies. The clustering can be performed over demands (queries) or supplies (items). A cluster is commonly formed in a distributed manner and each peer in the cluster concludes semantic links with other members by analyzing its observed query and item. This process is performed at runtime and yields an evolving CON overall. On top of this, peer is able to direct a walker to a certain sequence of promising peers. ES [149] approach applies an enhanced k-random walkers based search strategy. Each peer, at its arrival stage, selects a set of neighbors as the destination for item information propagation. It then concludes the knowledge about items nearby and further estimates the density of items. These actions yield a CON, which knowledge is used by each peer to determine the values of k and TTL. A limitation of ES is the lack of routing target selection means. Differing with ES, the AQFS [85] suggests different walker-based routing strategies for the search in social networks. Besides the node similarity strategy, it also applies other routing methods that are in particular designed for social networks, such as closeness centrality, betweenness centrality, weak ties, etc.. Each of them relies on a specific kind of knowledge, which are concluded by peers distributively and form CON. By their experimental evaluations, the closeness centrality and betweenness centrality based routing strategies outperform other options in the networks with random graph and random power law graph based topologies.

Al-Oqily et. al. propose a peer clustering method, MPRD [10] for the resource discovery in service specific overlay networks (SSON). It introduces a way of clustering media ports (peers providing services) in SSON and a resource discovery algorithm. MPRD propses a neighborhood based clustering strategy, by which peers exchanges the topological information via flooding with short TTL. On this basis, cluster head peers are elected distributively. For locating resources, MPRD uses enhanced random walk. Each walker, besides transmitting the request, piggybacks item information to neighbor peers. By observation, cluster heads can know each other in terms of topology and item topics. This knowledge is used to direct walker from one head peer to another. In contrast to MPRD, Agora system [74] follows the idea of clustering peers into small worlds and establishing inter- and intra-group semantic links. It considers a series of relations between group members, such as semantic distance, companion, pupil, far and orphan links. On top of this, walkers are directed to the semantic groups with similar interests.

Flooding and K-walkers hybrid approaches. One option of alleviating the heavy network traffic is to combine flooding and k walkers and use the advantages of each. Commonly, search strategies in this kind often divide a search into two or more phases. In each phase, flooding or k-walkers based search is applied.

A simple application of this idea is *F&KW* search [81]. For searching, requester peer issues flooding based query with a small TTL. If the desired item is found, the query succeeds; otherwise, requester peer issues k-random walkers based query with larger TTL. Query fails, if the second attempt fails. Despite its alleviating the duplicated messages, search performance can still be low in particular when searching rare items. The selection of the initial TTL value for the first try becomes crucial. Larger value increases the chance of query satisfaction, but also yields more network traffic overhead. When the flooding phase has a large TTL, the second attempt will become helpless.

CloneRW + DS [219] [218] combines random walker and flooding. Without CON, a request is delegated to a random walker with a large TTL. If the number of its traversed peers reaches a cloning point (e.g. the *i*-th peer), the current peer clones the request and issues a set of random walkers in order to increase the chance of finding rare items. The cloning point repeats for every *n* peers. This process continues until the TTL is exhausted or the desired item is hit. Whenever a walker reaches a dominating peer, the latter will issue an one-hop flood to its neighbors. Such dominating peers are selected by means of Local Randomized Greedy (LRG) algorithm [163] for alleviating the duplicated messages of flooding. However, LRG needs the global view to the network.

PercolationSearch [319] divides the peer life cycle into three phases: content caching, query implantation and bound percolation. The content caching phase is mainly used for knowledge propagation. Each peer, when arrival, issue a random walk with small TTL, in order to propagate the its local item index to other peers. When issuing a request, peer life cycle goes to the query implantation phase. That is to issue a short random walk with the request and implant the request to its visited peers. Once implant, the receiver sends a query message to a neighbor at certain probability. Such value is estimated based on the observation of propagated item indices.

Local item selection: As a key part of search, local item selection stands for the process that a peer compares its locally maintained items with a given query q, in order to find matched items. Once found, the item is recorded in q before it is routed. To avoid the unauthorized disclosure, the recorded information about an item commonly refers to the item description, which includes the possible file identifier (e.g. hash value of a movie), syntactic (e.g synopsis of a movie), meta-properties (e.g. actors of a movie) or semantic descriptions (e.g. conceptual tag of a movie), instead of the actual data (e.g. bit stream file of a movie). In more detail, the selection process compares the query with each item via a similarity measure. It is a function that outputs either a real value: $sim : \mathbf{D} \times \mathbf{D} \rightarrow \mathbb{R}$ or a binary result in $\{0,1\}$: $sim : \mathbf{D} \times \mathbf{D} \rightarrow \{0,1\}$ as the similarity score, where $\mathbf{D}$ is the description space. Similarity measure can vary in different systems with specific data and concerns.

2.2.3 Data Replication

The goal of data replication in distributed systems is to increase search performance and data availability. For this purpose, a replication strategy determines how many copies (replicas) of an item should be replicated to which peers. Such a principle can be realized differently by distributed systems regardless of their network settings (structured, unstructured or hybrid).

Since the structured P2P network variants provide data and index organization mechanisms, the queries to these systems can succeed, as long as the requested items exist. Replication strategies for the networks of this type aim to enhance system robustness and data availability. Hybrid P2P based systems leverage data replication as well to increase not only system robustness but also search performance. The replications in hybrid P2P systems are often controlled by super peers, as they have more information about items and requests. Replication strategies in unstructured P2P networks place relatively more emphasis on improving the search performance, in comparison with the systems of the former two types. Since each peer does not initially have a global knowledge of either network topology or items, the previous efforts attack the replication problem by taking advantage of the peer observation of items (supply), queries (demand), network topology, network traffic, peer capacity, etc. On top of this, each peer determines the needs for replication by itself or other peers and decides on the replication of items. In the context of this thesis, a replica refers to the actual data of an item file plus its description.

In the following, the representative non-semantic based replication approaches in P2P networks will be discussed, and their classification is summarized in Figure 2.8. As for the variants that leverage semantic-web technology, we categorize them into the semantic P2P computing field. We examine and classify them in Section 2.3.1.

Replication in structured P2P networks: Compared with the data replication strategies in unstructured or hybrid P2P networks, the structured P2P-based variants put more attention on optimizing

Approach	Structured P2P	Unstructured P2P		Hybrid P2P	QoS based	Cluster of queries or items
		With CON	Without CON			
CAN (Ratnasamy+ 2001)	√					
Chord (Stoica+ 2001)	√					
Pastry (Rowstron+ 2001)	√					
Kad / Kademlia (Maymounkov+ 2002)	√					
BitTorrent (Cohen 2008)	√					
RRP (Yamada+ 2005)		√				√
P2R2 (Sozio+ 2008)		√				
TopKMFR (Kangasharju+ 2002)		√				
HormoneRep (Sobe+ 2011)		√			√	
ARMAP (Forestiero+ 2008b		√			√	
SquareRoot (Cohen+ 2002)			√			
PtP (Leontiadis+ 2006)			√			
UniformAtRandom (Cohen+ 2002)			√			
Proportional (Cohen+ 2002)			√			
PathRep (Ata+ 2003)			√			
PathRandom (Yamamoto+ 2006)			√			
PathAdaptiveRandom (Yamamoto+ 2006)			√			
Gnutella v0.4 (GDF 2000)			√			
Freenet (Clarke+ 2002)			√	√		
PPR (Rajasekhar+ 2006)				√		
ODR (Rajasekhar+ 2006)				√		
IDXR (Puttaswamy+ 2008)				√		
MRRS (Rong 2008)				√		
Gnutella v0.6 (Klingberg+ 2002)				√		
Gnutella v2 (Stokes+ 2002)				√		
Napster (Saroiu+ 2003)				√		

Fig. 2.8 Representative non-semantic based replication strategies in P2P networks. As for the semantic-based variants, we categorize them into the semantic P2P computing field. They are studied in Section 2.3.1 and classified in Figure 2.10.

load balance, increasing resource availability and enhancing churn tolerance. Since the research of this thesis focuses on the data replication in unstructured P2P networks, in the following, we sketch the data replication strategies in state of the art structured P2P networks. For other approaches in this aspect, we refer the reader to the survey [296].

In Chord [344] networks, the unexpected departure of a participant can cause the ring structure disconnection and the loss of items. This is handled by enabling each peer p to maintain a list of its $log_2(N)$ successor peers and the items of p are replicated to the peers in its successor list. On detecting the failure of a successor peer, p replaces it with the next successor. In CAN [303] network, failure of a peer can be detected by its zone neighbors via periodic update message exchange. Such ungraceful departure can result in the dead zone as well as the loss of item. CAN network addresses this issue in two optional ways: The first is to use a α hash function that maps a single key to a set of α peers in the network, and via this mapping, items on each peer is replicated to α peers. The second approach is to keep multiple mirrors (called *reality*) of coordinate space, where the items on each peer are stored duplicately. Once a peer has left and its items are lost, a request can also retrieve those items from the backup realities. The replication of data in Pastry [314] network is conducted by each peer that replicates each of its items to a set of closest peers in the leaf set. In this way, the search in a Tapestry-based system can be replied more quickly and the loss of a peer and its items does not lower the system availability. Kad (Kademlia) [379] [251] system enables each peer to store a number of copies of an item to a set of peers which IDs are the closest. Besides, Kademlia provides a mechanism, called republishing, which enables each peer to observe the updates on items and other peers. Furthermore, Kademlia network commonly organizes its peers via sub-tree data structures. Each peer keeps the connections to the peers in its sub-tree. This makes Kademlia to be relatively more robust than other structured P2P networks. In BitTorrent [68] networks, each client peer can provide the content data to other peers, when it is downloading the chunks of a shared file, or after its fully replicating the file. However, data content can still be partially (or fully) missed if the data maintainer peers leave the network, regardless of whether those data are indexed or not. To alleviate the risk of losing data availability, one option is to apply Rarest first rule [67] during the data sharing process. With this, a downloader peer tends to replicate the chunks, which have been downloaded by the least times. Another mechanism BTRep [258] adds extra servers to BitTorrent networks for enhancing data availability. Those servers store selected important subsets of files.

Replication in hybrid P2P networks: A natural way of replicating data is that a requester peer downloads its needed data file from another peer, if the latter has successfully answered the query from the requester. This principle has been applied by Napster [317] system as well as Gnutella 2.0.

In [297], authors present two super peer-based replication strategies: periodic push-based replication (PPR) and on-demand replication (ODR). Each super peer maintains the access frequency of any file on each of its managed leaf peers. On top of this, super peer with PPR periodically checks whether such frequency value exceeds a threshold, or not. If it does, the super peer will replicate files from the leaf peer to itself and further propagate this information to other super peers. The super peer with ORD strategy counts the frequency of requests on its known files. To each file, once the frequency exceeds a threshold, this super peer requests other super peers for a copy.

IDXR [292] focuses on replicating data indices in a super peer based P2P system, for the purpose of enhancing the search performance on rare data. This work proposes three replication strategies: full replication, square-root replication and constant replication. The full replication strategy enables each peer to broadcast its data index to all of its neighbors reachable in 2 hops. With the square-root replication variant, a peer sends its data index to all of known super peers. Each in the latter then propagates that index to a set of other super-peers, which are randomly chosen. The cardinality of the set equals to the square root of the amount of known super peers. The first step of the constant replication strategy behaves exactly the same with square-root strategy. Instead of using the square-root value, a super peer replicates index to a fixed number of known super peers.

MRRS [312] combines the use of super peers and context-aware P2P environment. Peers with MRRS determine replications relying on the request rate of resources. This work is performed by super peers. MRRS is able to reduce the network delays and in the meanwhile increases the success rate. However, it might yield hot spots (super peers) in the network.

Replication in unstructured P2P networks: The idea of replication in the early state of the art work SquareRoot [69] is to replicate data to peers in order to minimize the search size in systems with identifier-based similarity measures. It finds that the optimality can be achieved when the number of the replicas of an item is proportional to the square root of the times that item has been requested. The replica in this context can either be the data or the shortcut of its identifer. Unfortunately, it is hard to achieve the ideal optimality in practice, since this requires the knowledge about the query rate on each item. As an approximated solution, the Pull-Then-Push (PtP) replication strategy [217] enables each requester peer to transmit (push phase) its retrieved item copies to its neighbors in order to approach the optimal number of replicas, once its query (pull phase) has found answers. However, in PtP, only satisfied queries triggers the replication. This is not very helpful to conduct the replication that are precisely concerned for unsatisfied queries.

The uniform at random replication policy (UniformAtRandom) is analyzed in [69]. With this policy, replicas are uniformly distributed over peers and number of replicas of items are approximately equivalent. In addition, each peer in the network has the same chance to receive a replica. The work [17] also discusses the random replication policy. It shows that this policy is effective for achieving both smaller search delays and deviations for k-random search. Nonetheless, in order to accurately apply this policy, the global view to network topology is needed. Unfortunately, this is always not possible in open environment, like unstructured P2P networks.

Cohen et. al. have analyzed the proportional replication strategy (Proportional) in [69]. The main idea is that the probability of replicating item to a target peer is proportional to its query frequency and item size. A variant of this is called owner policy [17]. With it, the peer that received a replica becomes the data provider, which data is supposed to be replicated or used to answer queries in the future. An advantage of this approach is that the replica can be propagated transitively and maintained by some peer topologically far from its provider.

Apart from the above, another option is to replicate the matched items of a query to the peers on the query path (PathRep) [17]. This idea has also been applied by Freenet [65]. After a successful Freenet search, each peer on the query path replicates the matched items, if such items are not known before; while after a file update, the file owner issues the messages to those peers holding a replica of that file and the latter update accordingly. Although PathRep is easy to be implemented, in comparison with the random policy, it give less help to increase the success rate of those queries that rarely succeeded before. In addition, it would introduce unnecessary redundancy of items in the network and therefore overconsume the overall storage resources. As an alternative, the PathRandom [397] policy grants each peer on a query path a probability of whether or not to accept a replica. Consequently, it tends to enlarge the data transmission burden of peers with higher connection degrees, as they have to transmit the replicas no matter they will accept, or not. Another option [397] proposed by Yamamoto et. al. is named path adaptive replication policy (PathAdaptiveRandom). With this, each peer determines whether or not to accept a replica by means of its current resource status or a predefined ratio. This policy improves network load-balance.

Yamada et. al. present RRP [395] that extends the candidate target peer set of a replication by considering more historical requests from multiple peers. RRP enables an item maintainer peer to select the final destination peer from a group of peers, from which historical requests were issued and observed. From this perspective, our proposed DSDR (cf. Chapter 4) behaves similarly. However, DSDR looks into the semantic-based similarity between items and only the unsatisfied requests .

In [103], Forestiero et. al. present a QoS-aware replication strategy ARMAP in P2P based grid networks. Inspired by ant colony, this work derives agents to transmit pre-clustered item descriptors among peers for the purpose of reducing system entropy. For this, each agent performs two operations: Pick and Drop, which respectively makes a probabilistic decision of whether or not picking resource from or dropping resource to a peer. When deciding a pick operation from some peer, agent calculates the overall quality of resources it has met in its journey, and then compares this with a resource quality at that peer. For deciding drop operation at some peer, agent compares the overall quality of resources it has met before with the resource quality at that peer. The comparison only happens between resources in the same cluster. This contribution can improve network load balance as well as the overall service quality of grid system. However, the movement of agent introduces extra message overhead. This can be event more obvious in a dynamic environment. Another QoS based replication strategy HormoneRep [334] is inspired by the principle of hormone dissemination and models peer (storage, communication, etc.) abilities as peer service quality. HormoneRep regards each query as a kind of hormone out of its requester peer. Such hormone dissemination causes the replication of items, which will be accommodated by selected peers with highly ranked abilities.

In [177], Kangasharju et. al. introduce Top-K Most Frequently Requested (TopKMFR) replication strategy. Each peer keeps a demand strength value for each known peer on each requested item. Once a request on certain item is received from a known peer, the corresponding demand strength value increases; while that demander peer gets a replica, the corresponding demand strength value decreases. When considering a replication, an item maintainer peer determines the best destination peer from its known peers based on their current demand strength values. This approach has been analyzed to be near-optimal in terms of hit rates. In addition, it scales to the replacement of items at destination peer for the purpose of evicting space to accommodating new replicas. Another near-optimal replication strategy P2R2 [337] attacks the same research problem and provides the best results so far for the data with non-semantic descriptions. The replication problem in P2R2 is reduced to the known multi-knapsack problem. Replicas are regarded as elements that are supposed to be put into bins which are representing those target peers. In particular, this method has been proven to converge to a 2-approximation solution within small networks.

2.3 Semantic P2P Computing

As the interdisciplinary combination of semantic web technology and P2P computing, semantic P2P computing has been widely discussed in the semantic web, computer networking and intelligent information retrieval communities for the past ten years [345]. The merging of the two techniques brings good features to both.

Well-known in the intelligent IR community, the emergence of the semantic web offers opportunities for more accurately describing and searching for things in a machine-understandable way. This benefit has shown its power gradually in the last decade for the retrieval of web-based content, such as web documents, web services, multimedia data, etc.. As a trend of Web 2.0 [273], web users are not merely content consumers, but also at the same time are capable of acting as content providers, even via services. This characteristic is also partly driven by and conforming to the principle of service oriented architecture. Users with Web 2.0 form social and networked groups and they are becoming the major energy powering the engine of the web economy [52]. Towards the intelligent Web 3.0 [123] [141], the integration of classic IR techniques with semantic web techniques requires proper networking models, which adapt to the new interaction phenomenon and facilitates the communication between users. On the other hand, the development of P2P computing brings user collaboration as well as resource retrieval and sharing to a new stage, where the original consumers in the client-server age become data and service providers. As a consequence, the need for more precise and intelligent data/service retrieval and composition increases. To handle the challenges from both sides, the integration of semantic web and P2P computing is a natural choice for the users and enterprises from both communities. Once integrated, the cross-disciplinary field of semantic P2P computing has the following new features that did not belong to either of the two originally:

- *Semantics enhanced peer local computation*: With semantic web techniques, a peer in a P2P network is not only processing data, but also capable of understanding and using the meaning of data. Instead of being entirely directed by a user, a P2P client can be granted more ability for automated computation and decisions on semantic-based data in more complex context. This can become true, once the peer local computation is properly controlled by a certain algorithm. The latter works compatibly with data and peer local knowledge expressed in semantic-based formalisms.

- *Mapping between peers and knowledge expertise*: In conversional P2P computing, peers know each other in terms of addresses and observed data, while in semantic P2P computing, each peer is able to maintain and perform reasoning on knowledge, which is properly organized as relations between individuals and conceptual topics. On top of this, peers, via traditional P2P message exchange, are able to know about each other in terms of expertise, formally expressed as topics of interest. As a result, by knowledge propagation and inter-reference, peers form a content overlay network, in which data and knowledge are formally expressed, grouped and more importantly mapped onto peers.

- *Semantic-based support for collaborative computing*: The collaboration between peers in the traditional structured, unstructured or hybrid P2P networks is driven by data and protocols. With semantic-based automated reasoning and planning, peers are able to decide on complex collaborative actions based on semantics of context and primitive networking operations at a small granularity. From the opposite point of view, P2P-based message exchange and gossiping facilitate the propagation of knowledge. This potentially increases the field of view of a peer in terms of the semantically expressed knowledge on other peers. With semantic-based formal description, that knowledge can be more precisely understood and thus more helpful for collaborative task solving and information retrieval.

In both the research and application perspectives, the semantic P2P computing field consists of two aspects: semantic data search and composition in P2P networks as well as semantic service discovery and composition in P2P networks. To the best of our knowledge, apart from service composition, there is no standard problem definition for semantic data composition in P2P networks. Strictly speaking, no method provides an overall solution to this challenge, since it is hardly possible to cover all data formats out of all domains. As for those P2P based methods for the federated search and integration of RDF data, we identify them in the context of P2P-based semantic data search, rather than the composition. The reason is that, even when integrated, there is still no interaction between the component data pieces inside a resultant composite RDF entity.

This section is further structured as follows: We summarize the representative methods for the P2P-based semantic data search in Section 2.3.1. Subsequently, the P2P-based semantic service discovery and composition approaches are discussed in Sections 2.3.2 and 2.3.3, respectively. Finally, in Section 2.3.4, we point out several existing challenges in the semantic P2P computing field, which are targeted by the research of this thesis.

2.3.1 Semantic Data Search and Replication in P2P Networks

Semantic data search in P2P networks: P2P-based semantic data search, in the context of this thesis, describes those retrieval methods built upon P2P networks which either take advantage of semantic-based facilities to direct query routing (*semantic query routing*) or are aimed to find semantic items (concept or individual in OWL, RDF data, lexical terms with relations, etc.) (*semantic item selection*). Although it is possible that some data search strategies can be adapted to service discovery scenarios, we adhere to their literature and still categorize them in the scope of data search. Approaches of both kinds often work with semantic-based CON. Such an abstract overlay network in the semantic data search paradigm keeps, in particular, the information that involves certain semantic web technologies for the item, topic or location description. In general, this information is distributively maintained by peers and updated over time, and indicates a *semantic mapping* overall between peers and items. Depending on the network type, this mapping can be achieved and maintained in different ways. In a structured P2P network (e.g. CAN, Chord, etc.), the semantic information about each item is often encoded as tag (concept) vectors or simply bit strings, from which keys are computed and further mapped to data locations. As structured P2P networks have their fixed protocols for query routing, the research points in this context include the encoding of semantic information, the selection of hash functions and the index management in DHTs, for the purpose of achieving higher data availability, network traffic load balance, etc. In hybrid P2P networks (e.g. JXTA), such a semantic mapping is organized in a more loosely-coupled manner and can be held by a group of specific peers, which in particular respond to the message (query) propagation. While for the unstructured P2P variants, the semantic mapping can be distributively maintained by all peers. It is often achieved via training before the initial stage, or it evolves distributively based on peer observations. In this section, we study the representative approaches for semantic-based data search in P2P networks. Their classification is shown in Figure 2.9. The non-semantic based variants are discussed in Section 2.2.2. For the classification of them, we refer the reader to Figure 2.7.

Semantic data search in structured P2P networks. A representative method GridVine [2] is proposed by Aberer et. al. for building full-fledged scalable semantic overlay networks. GridVine allows to answer conjunctive and disjunctive triple pattern queries on distributed RDF data. Built on top of a P-Grid network [1], GridVine derives an upper layer, which supports to insert, semantic-link and search RDF data. As for the data-peer mapping, GridVine leverages the hash method of P-Grid system, which allows to distribute RDF data to tree structured network infrastructure and guarantees a logarithmic search size complexity. For dealing with the heterogeneity, GridVine suggests to establish semantic links between RDF schemas, by means of using OWL owl:equivalentProperty statements. In this perspective, GridVine outperforms a former effort PIER [155] that is built on CAN network for the distributed query answering on RDF data with an assumed global RDF schema.

Another state of the art effort RDFPeers [39] offers the efficient search on distributed RDF triples based on an extended Chord network, MAAN [40]. In RDFPeers, each RDF triple is stored on three

Approach	Structured P2P	Unstructured P2P			Hybrid P2P	Informed search with CON	Cluster of queries or items
		Flooding based	K-walkers based	Flooding & K-walkers hybrid			
RDFPeers (Cai+ 2004)	√						
TLP2P (Gu+ 2007)	√						√
SemKad (Kim+ 2011)	√						√
Atlas (Kaoudi+ 2010)	√						
RDFCube (Matono+ 2007)	√						
CQRDFP2P (Liarou+ 2007)	√						
OSMAP2P (Pellegrino+ 2013)	√						
6RDFP2P (Ali+ 2014)	√						
AHRDF (Karnstedt+ 2008)	√						√
PIER (Huebsch+ 2003)	√						
GridVine (Aberer+ 2004)	√						√
DSR (Bianchini+ 2009)		√				√	√
Bibster (Haase+ 2004)		√				√	√
INGA (Loser+ 2007)		√				√	
GES (Zhu+ 2006)		√				√	√
SEIF (Yang+ 2009)		√				√	√
GrouPeer (Kantere+ 2009)		√				√	
REMINDIN (Tempich+ 2004)		√				√	
RTTP2P (Brunner+ 2012)		√				√	√
SOSL (Giunchiglia+ 2010)		√				√	√
S2P2P (Cao+ 2013)			√			√	√
AP2P (Tirado+ 2010)				√		√	√
OntoZilla (Joung+ 2009)				√		√	√
Edutella (Wolfgang+ 2002)					√		
SOSPNet (garbacki+ 2010)					√	√	√
Sem2i (Yang+ 2012)					√	√	√
SPAP2P (Zhou+ 2014)					√	√	
SocSemP2P (Mei+ 2010)					√	√	√
PAIRSE (Benslimane+ 2013)					√	√	

Fig. 2.9 Representative semantic-based search strategies in P2P networks. These strategies are discussed in Section 2.3.1. As for the non-semantic based P2P search variants, we examine them in Section 2.2.2 and show a classification of them in Figure 2.7.

peers. To do this, the first step is to map the subject, predicate and object values of a triple to the m-bit identifying space of MAAN. For string-typed values, SHA-1 function is applied; while for the numeric-literals in object terms, locality preserving hashing is used. The next step is to follow the Chord protocol that stores each data item (RDF triple) to specific peer which ID is the closest successor to the key. An atomic query with simple triple is answered based on its non-variable term. This is done by at most three searches, each at traffic cost $log(N)$, where N is the network size. Moreover, RDFPeers allows for the disjunctive and range queries. Via an extra sorting over the ranges, the logarithmic traffic complexity is also guaranteed. A limitation of RDFPeers is that it would yield unnecessary network traffic overhead for on-line query answering. RDFCube [247] made progress on solving this issue. RDFCube is a three-dimensional hash index. The identifying space is divided into a lot of 3-dimensional 1-bit units. Called a cell, each contains 3 bit flags that correspondingly represents the presence / absence of a term in a triple that was mapped to this cell. By inter-referencing the indexes, peers in a MAAN network is able to decide to which peer a query should be forward.

Another improvement Atlas [180] on RDFPeers is proposed by Kaoudi et. al.. Atlas system hires the same mechanism of RDFPeers to encode and store RDF data in an optimized Pastry network, Bamboo [307]. In particular, Atlas allows for the distributed RDF data search with SPARQL. It is realized by the following: Each peer contains a local translator that responses to transform the SPARQL query to a certain internal representation. When a peer issues a SPARQL query, it locally performs the translation and distributes the partial queries to other Atlas peers. Altas in addition scales to the semantic regimes of RDF data. Two alternatives are proposed: the "forwarding" process infers the hidden triples and stores them in advance; while the "backward" process works on demand for queries in a top-down fashion.

Targeting on the same problem, Gu et. al. propose a two-layered structured P2P network TLP2P [127] with semantic-based clustering. The upper layer of TLP2P models the semantic clusters as a ring structure; while in the lower layer nodes in each cluster is organized as Chord network. Based on a share domain ontology, the upper layer clustering is achieved by checking the meta-type (rdf:type or object/data property) of RDF predicate in each triple. For constructing the lower-layer, standard SHA-1 hashing is applied to the subject and predicates. On receiving a query, a peer first forwards it to the cluster by means of the upper layer knowledge. In the lower layer, query is routed to the correct peer that maintains the same key.

CQRDFP2P [225] allows for the distributed search of RDF data in DHT-based P2P networks. Besides, it is able to answer continuous conjunctive multiple-predicates queries (CCMPQ). A CCMPQ is a kind of subscribed query, like "tell me the basic information about the current worker at some position, once workers switch", to dynamic RDF data sources. CQRDFP2P organizes the sources by a generic DHT-based P2P network. In CQRDFP2P, a CCMPQ is also registered distributively, one triple (a subscription) on a peer. Once an update on data happens, the query maintainer peers re-issue the sub-queries. Following the similar data indexing method of RDFPeers, CQRDFP2P proposes a query chain mechanism, that takes charge of collecting and merging partial results. A query chain consists of the peers that have successfully answered a partial conjunctive query. For merging partial results, a walk is issued. It traverses through the chained peers. Each peer, when receiving the walker, performs the local RDF data merging operation and transmits the partial result to the next peer. A number of works are propose based on the principle of CQRDFP2P. A variant of CQRDFP2P is OSMAP2P [282], which takes use of a CAN network instead and provides a mechanism to optimize the network traffic overhead. 6RDFP2P [13] hires six indexes for the six combinations over the RDF triple terms, which is reported to have higher run time efficiency.

Another search strategy SemKad [189] captures the topic based retrieval with Kademlia network. SemKad assumes that all peers share a common set of topics as well as the semantics of item and query are represented via vector space model. In addition, a high-dimensional topic vector is further summarized as a low-dimensional vector by the latent semantic index process. Each peer holds a centroid semantic vector over its items. By initial setting, the semantic vectors of peers are propagated to each other, such that each peer is able to determine its semantic neighbors. On top of this, each peer locally computes the semantic clusters in terms of topics. Finally, a CON is built upon the DHTs of Kademlia network. On receiving a query, peer determines up to K closest semantic neighbors for query routing.

Karnstedt et. al. propose a method, called AHRDF [182], for distributed SPARQL query answering on DHT-based layered networks. AHRDF is able to organize and query data in a way that supports data integration in an ad-hoc or pay-as-you-go manner. The proposed architecture contains several layers. On top of the P2P layer, RDF data is distributed to peers with DHT-based indices. Based on this, a semantic layer is built, in which RDF triples are grouped and mapped. Compared to RDFPeers and PIER, AHRDF suggests to leverage edit distance-based similarity measure to bridge the gap of schema heterogeneous. This feature can be applied on demand, i.e. AHRDF provides an extension to SPARQL language, which allows to specify the syntactic similarity between those concerned terms possibly defined in different schemas. In this aspect, GridVine can even reached farther by its "semantic gossiping" that provides a logic-based way to match and inter-translate terms (properties, classes) between RDF schemas.

Semantic data search in hybrid P2P networks. Edutella[74] is a hybrid P2P network, built on top of JXTA, aimed at providing an RDF-based metadata infrastructure for P2P based e-learning. Edutella comprises two key components: Edutella service and Edutella peer. Built on the JXTA core layer, the Edutella service leverages the functions provided by JXTA service layer and further exposes a set of facilities, including query, data replication, peer-data schema mapping, mediation and clustering. In particular, this component supports the translation mechanism between Edutella query & result exchange format and the peer local data format. It enlarges the inter-accessibility between the peers with highly heterogeneous architectures. The Edutella peer component encapsulates the Edutella services and supports other tools that are useful in the context of e-learning.

The SOSPNet [114] proposes a CON with super peers. In SOSPNet, weak peers are clustered into the same group if their demand topics are similar. Each group is managed by a super peer, which is able to know their common requests and also the answers. On one hand, super peer takes use of a two-level cache to record the answers and the addresses of their provider peers; on the other hand, weak peer collects statistics about the contents indexed by super peers. Having this information, weak peer makes local decision about which super peer to connect to. In SOSPNet, a requester weak peer asks the provider peer to share its list about super peers, if the request was satisfied. The

---

[74] `http://www2002.org/CDROM/refereed/597/`

requester peer will merge the obtained list with its own and be able to make a promising choice over super peers for its future queries. In this way, the self-organized CON evolves.

Similar with SOSPNet, the SocSemP2P [255] system organizes its super peers via a structured P2P network. Based on vector space model, leaf peers are clustered based on their item topics. Such information is maintained by the data index of super peer by means of using the BloomFilter-based encoding [35]. Another similar approach Sem2i [405] builds a Chord network over super peers. Difference is that the topics used for clustering is organized as a tree hierarchy. Compared to SOSPNet and SocSemP2P, Sem2i gets more flexibility on approximated categorization and query answering. SPAP2P [416] hires 6 indexes for the RDF triple and supports the SPARQL query answering via a two-layered hybrid P2P network. The lower layer consists of storage peers that can be both the RDF data provider and requester; while the upper layer comprises the router peers, which are in addition organized by a Chord ring. SPAP2P takes use of a two-level index. The router peer maintains the indexes of all its known storage peers; while the storage peer maintains the local index for its own data. On receiving a query, the index on the router peer is checked first. Then the decomposed query with basic graph pattern will be sent to the corresponding storage peer. Sub-results are merged in the common router peer of the storage peers that provide the triples. In PAIRSE [25], authors apply the similar principle for the integration and distributed SPARQL search on RDF data within DHT-based hybrid P2P networks. Leaf peers provide RDF data and each super peer manages a virtual organization of leaf peers that share a common domain ontology. For answering a query, selective flooding is applied amongst super peers and query broadcasting is performed inside each involved organization. In particular, PAIRSE allows for specifying privacy constraint in queries. For this, additional check is conducted during data integration.

Semantic data search in unstructured P2P networks. A early work REMINDIN [353] offers the semantic-based search for RDF data in unstructured P2P networks. Being capable of creating, storing and sharing RDF data, each peer in REMINDIN observes the queries it routes over time and concludes the knowledge about RDF resources on remote peers (semantic mapping). In particular, REMINDIN supports a RDF-S based meta-model, called SWAP, for semantic mapping. In SWAP, two classes Swabbi and Peer are defined. Swabbi responses to capture the main content of mapping, i.e. RDF resources and their location; while a Peer object associates the location with a confidence value. It records the probability of a remote Peer (location) to satisfy a query on the topic indicated by rdf resource. It is estimated over time based on peer observation. On top of this model, REMINDIN peers conduct restricted flooding for query answering. On receiving a query, a REMINDIN peer compares it with its local items. If it does not find a match, it relaxes the query topic by removing the constraint on subject, predicate, object, or replacing the old subject class to its superclass.

In Bibster [132], peer local knowledge is achieved based on the advertisement of other peers, i.e. each peer, when joining in the network, issues flooding based advertising messages that carry the semantic topics (called expertise) of the maintained items. On receiving an advertisement, a peer tries to establish a semantic link with the advertiser peer, if the topic of its local items turns out to be semantically similar. When routing a query, a peer forwards query message to at most *N* peers which expertise are similar with the query. INGA [232] follows the same principle with respect to using logic-based CON. In addition, it enables each peer to collect information about the expertise of remote peers at run time, instead of the one-shot advertisement in Bibster. A shortcut of item on a remote peer is created, if a query gets answers from it or detected by flooding-based probing. On routing a query, a peer with INGA probabilistically selects the target peers depending on logic-based semantic similarity. Similar with Bibster, the DSR approach [28] enables peers to actively send and receive probing messages about local services and establish semantic links in a way ruled by Overlay Management Protocol [370]. To prune the set of peers to be investigated, a peer sends queries to those semantically linked peers only. If no semantic link is useful, the target peers are randomly selected.

Giunchiglia et. al propose a semantic link-based flooding search strategy [122] SOSL. Instead of a flat knowledge organization, each peer employs a local ontology that encodes the hierarchical relation between topics. Each topic corresponds to a set of relevant files. Semantic links between topics are pre-defined among peers. On top of this CON, a query with TTL limitation floods transitively along the links about the relevant topics. Similar with the Bibster and DSR, the network traffic overhead of this approach is lowered. However, how to create such semantic links is left unresolved. Moreover, the maintenance of those links becomes a problem when adapting to the network dynamics. Zhu et. al. present in [418] a method, called GES, for the semantic search in unstructured P2P networks. A distributed CON maintenance algorithm is periodically executed, which organizes the network into overlapping groups. In each group, the member peers have items on similar topics. This relation is encoded as semantic link. Differing with the above semantic link-based approaches, each peer in GES also keeps another kind of semantic link that points to peers which items are in irrelevant topics. GES hires a hybrid search strategy which uses random walker over semantic links in both kinds and floods over the peers inside a group, if the topic of that group is similar with the query. The idea of group has also been applied in the P2P based database field. Built on unstructured P2P networks, the GrouPeer system [179] enables each peer to find "similar peer" via schema matching. The grouped peers are helpful to enhance the search performance, availability and scalability of distributed database system. The work SEIF [404] enables each peer to cluster its observed queries by their topics and associates each topic with a set of observed items. For this, peer performs the SVM-based learning and computes the semantic centroid of grouped queries as the core of topic. On top of this, each peer maintains a semantic link table and a random link table. The former records the clustered peers with semantically similar demands; while the latter records a set of randomly sampled neighbor peers. When forwarding a query, a peer aligns the received query to topics it knows about and suggests the corresponding peers as the targets. Peers in the random link table is used if no suggestion can be made.

A contribution AP2P [356] establishes a Chord like ring-based CON on top of an unstructured

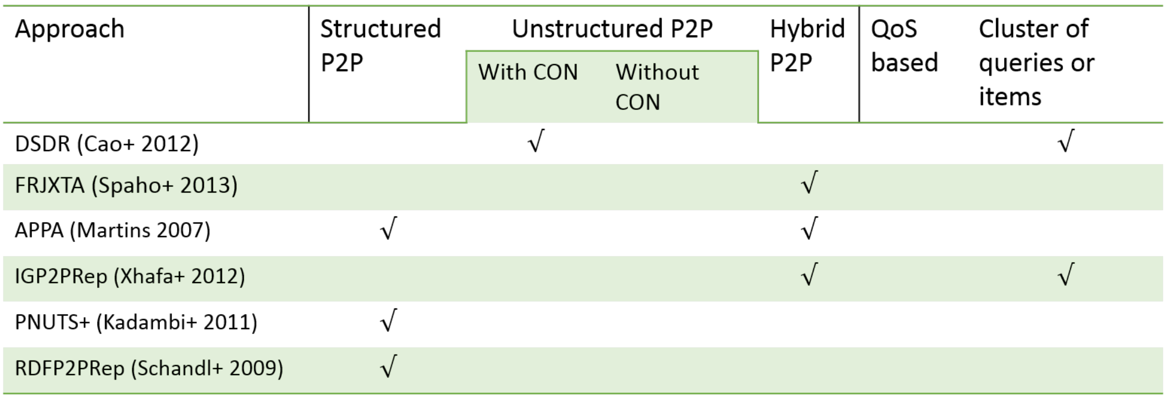
P2P network. Each peer in AP2P classifies its maintained items onto a shared set of topics. In addition, peers are clustered over topics. The chord-like ring records the address of clusters in a prefixed order of topics. Clusters with similar topics are arranged to be near with each other. For creating this, each peer periodically checks its observed query and item topics and adjusts the order of clusters. Each node on the ring structure contains the address of a peer belonging to the corresponding cluster. A AP2P query is processed in two steps: The first is to find the cluster on the relevant topic and send the query to that cluster. The second is to issue a flooding inside the found cluster. By applying organizing the peers in an unstructured P2P network, this method can achieve relatively smaller search size near to the logarithmic scale. However, the clustering of peers and its maintenance needs extra message exchange. Such traffic load can be heavy, when items and peers are added or deleted frequently.

In comparison with AP2P, the OntoZilla system [169] applies more relaxed interlinking between clusters. With a shared ontology, the peers in an unstructured P2P network is clustered based on their items over topic concepts. OntoZilla identifies the inter-cluster relations between peers amongst different clusters via their semantic relation of topics, such as parent / child link, ancestor / descendant link, sibling link, partner link, etc.. In order to maintain these peer-wise relations, extra gossiping messages are exchanged between peers periodically. On top of this, semantic mapping is established. For this, each peer holds a routing table, recording the topics and their associated peers. When routing a query, an OntoZilla peer forwards it to the cluster with the closest semantic topics, if such similarity is adequately large. If no relevant cluster is found locally, it broadcasts an inquiry message to find suitable cluster and send query to it. Within a cluster, a request is flooded along sibling links. Since the topic category (ontology) is shared, the inquiry message can always returns an answer.

Another work RTTP2P [37] proposes a method for semantic-based resource management and retrieval in unstructured P2P networks. It assumes that each peer manages a domain that comprises a set of resources. They are semantically classified to a local hierarchical schema by means of the hill-climbing search Cobweb [97]. A user request in RTTP2P is delegated to a query with TTL limitation. On receiving a query, each peer selects its local resource that are relevant. Then it routes the query to the next peer which domain knowledge is closest to the demand topic. When multiple domains are determined to be relevant, the one yielding the minimum data transfer delay is chosen.

Semantic data replication in P2P networks: Semantic data replication denotes those P2P-based strategies that either take advantage of semantic-based technology for determining replication (replica item, source and destination peers), or replicate semantic items (concept or individual in OWL, RDF data, lexical terms with relations, etc.).

Semantic data replication in structured P2P networks. Replication strategies for structured P2P networks are specified by network protocols (cf. Section 2.2.3). The main target of them is to enhance data availability and network load balance, as summarized in the survey [296]. Since struc-



Approach	Structured P2P	Unstructured P2P		Hybrid P2P	QoS based	Cluster of queries or items
		With CON	Without CON			
DSDR (Cao+ 2012)		√				√
FRJXTA (Spaho+ 2013)				√		
APPA (Martins 2007)	√			√		
IGP2PRep (Xhafa+ 2012)				√		√
PNUTS+ (Kadambi+ 2011)	√					
RDFP2PRep (Schandl+ 2009)	√					

Fig. 2.10 Representative semantic-based replication strategies in P2P networks. As for the non-semantic based variants, we study them in Section 2.2.3, and classify them in Figure 2.8.

tured P2P (or DHT-based) networks impose data-peer mapping via a global hash function, semantic information of item is commonly encoded as a string in the resource identifier space. As a result, there is no need to conduct replication by directly considering the semantics of item. The essence in this context is to keep the availability of data indices (DHTs) and manage the replica consistency.

A early contribution is the APPA [245] system. It is aimed to enhance data availability for XML-based complex documents (e.g. semantic service description, OWL concept or individual, etc.) in DHT-based structured or hybrid P2P networks. It comprise four main components: Key-based Storage and Retrieval (KSR), Persistent Data Management (PDM), Communication Cost Management (CCM), and Replication service. The KSR component provides a hash mechanism. It allows the encoding for not only the whole complex semantic description, but also each of its attributes. In particular, encoded attributes can be stored in different peers and accessed separately. The PDM component is derived to ensure high availability of key-value pairs and in the meanwhile assure mutual consistency among replicas. The CCM component is employed to manage and monitor network run time status. Such information provides evidences for replication determination. The Replication service component concludes the demand and supply information and analyzes which indices and (which parts of) data need to be reproduced.

Kadambi et. al. propose a data replication and management strategy called PNUTS+ [171] for the distributed web databases. This work considers the geographical coherence between data centers with accessor locations. In addition, it distinguishes full data and stub. The latter is concerned with a kind of high-level data summary (overview). PNUTS+ conducts the replication of web item records (e.g. an article on Amazon) or their stubs amongst different data centers by static constraint-based rules, in which the network bandwidth, historical access rate, number of copies and geographical distances are considered. In particular, PNUTS+ combines temporal duration with the management of replica. It allows to set a retention interval to a full replica in some data center in order to keep it for some time in which a large amount of access happens. After that period, the full version copies of a record can be discarded or replaced with a stub, if full content is not interested any more by the users in the same location.

A replication management for RDF graphs RDFP2PRep [321] is proposed by Schandl et. al, for the distributed SPARQL query answering. By using a structured P2P network with star-styled topology, RDFP2PRep ensures the maximal off-line data availability. On the server side (center peer), a replication manager is set. It determines which subset of data is to be replicated to which client, based on a series of factors including: RDF graph structure, ontology structure, statistics on queries, user context and interests. On the client side, a replication engine is settled, which takes charge of reporting to server about the current client status in terms of the above aspects. Once this is done, server side determines the subsets of RDF graphs to be replicated and initializes a transmission procedure that "pushes" the data to client.

Semantic data replication in hybrid P2P networks. In JXTA[75], the content of an item with its JXTA ID is named as *Codat*. The specification also states that the duplication of codat can improve the system robustness and data availability. However, JXTA leaves this function open and as an implementable option for the user applications in the upper layers. The authors present, in [338], a fuzzy rule based replication strategy FRJXTA for JXTA-based P2P networks. The replication of an item in FRJXTA is controlled by a replication factor, which is the ratio of replicas over the total number of copies (including the originals) of that item. This approach establishes fuzzy mappings between the replication factor values and a set of terms: $\{\cdots, Very\ Low, Low, Middle, High, Very\ High, \cdots\}$. The value of replication factor is determined by three parameters: number of documents per peer, replication percentage and scale of replication per peer. Each of these parameters is also mapped to a fuzzy set. On top of this, 27 rules are introduced. Each rule decides the needed replication factor term for an item, given the observed fuzzy member value of the parameters.

The APPA system [245] [8] also provides semantic-based data replication in hybrid P2P networks. In the hybrid P2P-based version of APPA, networked users are grouped into communities by means of their interested topic semantics. Each community has a master (primary) peer (known as super peer), which responses to gather and manage the items indices of its community member peers. A replica is considered to have two kinds of descriptions. The common schema description is defined by community as a shared notion of interests in certain semantic formulism; while the local schema description reflects the local understanding of each peer to common knowledge. Each peer specifies a mapping between these two descriptions about an identical item. APPA follows a simple demand proportion-driven replication strategy. The value of APPA lies in its replica conflict solving and reconciliation. It employs three criteria to manage the updates: timestamp ordering, sequencing of update arrivals and application semantics. On top of this, APPA suggests a distributed semantic reconciliation method. The basic idea is to collect the applied actions out of member peers to their local replicas of the same item, then grouping them and derive a new schedule of actions, which can be applied by each member in order to preserve the data consistency.

---

[75] `https://jxta.kenai.com/Specifications/JXTAProtocols2_0.pdf`

IGP2PRep system [392] proposes strategies for maintaining replica consistency in JXTA-based networks. The replica in IGP2PRep refers to the XML-based structured documents with semantics. IGP2PRep, networked peers are organized into groups. Led by a super peer, each contains a set of peers having common interests. On top of this, super peer of a group builds a data repository that manages the data of member peers. Once an update on some document has been made by a peer, such events is propagated to other peers in the same group by two optional ways: multicast and unicast. In addition, each peer periodically queries the super peer for the newest version of its concerned files and incorporates the newer updates. For supporting this, super peer holds a DHT-styled table that maps document ID to logs, sorted by the version of update.

Semantic data replication in unstructured P2P networks. To the best of our knowledge, except DSDR (cf. Chapter 4), there does not exist semantic-based data replication methods in unstructured P2P networks. Although several literatures entitle their work with the term "semantic", no semantic-based technology has been applied for either replication determination or the description of replicas.

2.3.2 Semantic Service Discovery in P2P Networks

In comparison with centralized semantic service discovery, the discovery methods within P2P-based environments combine service matchmaking means with query routing strategy, and further collect and merge results from multiple inter-connected sources. In this perspective, the query routing strategy is essential. As discussed in Section 2.2.2, the purpose of P2P-based query routing is to locate data for a given request. Specifically, in the context of distributed service discovery, the goal of query routing is to locate a list (set) of service provider peers whose services are relevant to a request. For this, different kinds of networks can give various types of support. Structured or hybrid P2P networks impose an initial distribution of service indices over peers. With a fixed query routing protocol, high search satisfaction can be guaranteed, but at the cost of high network traffic overhead for index maintenance against network dynamics. Therefore, the key point of the work on this aspect is to organize service indices in a proper way in order to lower query response time and traffic cost for maintenance, as well as achieve reasonable system load balance. In contrast, unstructured P2P networks do not impose initial data indexes and peers do not have a global view of data location or network topology. Although this loosely-coupled structure offers higher robustness and lower traffic cost for maintenance, the quality of service discovery is heavily subject to the CON and query routing means. In this regard, the research puts more effort on enhancing query routing and the construction of CON about semantic services.

In this section, we consider the representative methods for P2P-based semantic service discovery. A classification of them is shown in Figure 2.11. For the centralized variants, we refer the reader to Section 2.1.2 and Figure 2.4. The early state of the art contributions in this area include SSLinkNet [228], WSML-P2P[371] and WSPDS [20] for service discovery in structured and respectively un-

Approach	IO Level	IOPE Level	Logic-based	Non-logic based	Hybrid	Adaptive	Service Formalism	Structured P2P	Unstructured P2P	Hybrid P2P
DHAWSD (Jia+ 2010)	√		√				OWL-S	√		
RTSemMa (Skoutas+ 2008)	√			√			OWL-S	√		
SSLinkNet (Liu+ 2005)	√			√			WSDL (with service class)	√		
NIPPERS (Sioutas+ 2009)	√			√			WSDL (with service class)	√		
ERGOT (Pirro+ 2012)	√				√		SAWSDL	√		
Chord4S (He+ 2013)	√			√			OWL-S	√		
IPIS (Najar+ 2012)		√	√				OWL-S	√		
WSML-P2P (Vu+ 2006)		√			√		WSMO	√		
RS2D (Basters+ 2006)	√				√	√	OWL-S		√	
P2P-SDSD (Bianchini+ 2008)	√		√				Ontology-based desc.		√	
Wspds (Banaei + 2004)	√		√				DAML-S		√	
QoSSemMaCC (Lin+ 2013)	√			√			WSDL (with service class)		√	
PAN (del Val+ 2012)	√		√				OWL-S / WSMO / SAWSDL		√	
AntSrvDis (Zhang+ 2014)	√		√				Service class		√	
AntCoMx (Mastroianni+ 2014)	√		√				Service class		√	
QoSSemMa (Zhang+ 2009)	√				√		OWL-S			√
PYRAMID (Pilioura+ 2009)	√				√		PS-WSDL			√
JSOAP2P (Di Modica+ 2011)	√				√		OWL-S / WSMO / SAWSDL			√
MWSP2P (Srirama+ 2010)	√				√		WSDL (OWL for IO)			√
HACSemMa (PE only) (Rasch+ 2011)		√	√				Context-aware services			√

Fig. 2.11 Representative distributed semantic service discovery approaches. As for the centralized variants, we examine them in Section 2.1.2 and they are classified in Figure 2.4.

structured P2P networks. In the following, we focus on the recent representative approaches. We do not independently discuss those P2P search means for generic purposes. For the discussion on this topic, we refer the reader to Section 2.2.2.

Structured P2P-based semantic service discovery: Jia et. al. present a distributed logic-based matchmaker, DHAWSD [162], for IO-level OWL-S service selection in DHT-based structured P2P networks. The designed network contains a group of "backbone" peers that maintain the indexes of services profiles. Given a query, a backbone peer matches the query description with its known services. It adds the matched candidates to query answer set and routes the query to another backbone peer. The local selection process compares the query with each of its known service. The similarity score is computed based on service IO signatures. For each pair of the parameter concepts to be compared, a peer with DHAWSD computes their hierarchical relation in an ontology and derives the similarity score.

Another DHT-based system ERGOT for hybrid IO-level discovery of SAWSDL services is presented by Pirro et. al. in [286]. Built on top of a Chord network, ERGOT combines DHT with an extra CON. It allows peers to dynamically form clusters as CON via continuously detecting semantic links between peers. Two peers are linked if their services are semantically similar. With this facility, queries are routed along the semantic links to relevant peers. The combination in this way overcomes the shortage of exact term match of traditional DHTs. ERGOT exploits an ontology-based similarity measure for matching service offer and request. Two levels of ontologies are employed. First, each hashtable maintainer peer categorizes its known services into a high-level ontology, and second each service are described on the basis of domain ontologies. In addition, ERGOT offers a hybrid service matchmaking means, which estimates in addition the service name signature and structural similarities, combines them with service IO parameter concept relevance by a weight-based aggregation means.

Another distributed system, IPIS [263], realizes its service discovery on top of a pervasive computing system. This work extends OWL-S description language by allowing additionally the intention-oriented conceptual annotation. Besides the matchmaking on service intentions, IPIS also checks the similarity over the verbs in intention descriptions as a backup, if a conceptual comparison gets failed. The RTSemMa approach [332] suggests to build an IO-level OWL-S semantic service discovery framework based on the structured P2P network, called SpatialP2P [178]. In particular, RTSemMa models service IO matching to R-Tree based overlapping of minimum bounding rectangles. The model in this kind can be regarded as certain index of service. With it, the semantic matchmaking of service with query is reduced to the comparison with indexes. As for the the search, the SpatialP2P network of RTSemMa guarantees that any stored area can be searched and reached from any other peer by exploiting local area knowledge. In addition, SpartialP2P supports a top-k search for range queries.

Another work, NIPPERS [328], takes advantage of structured P2P network for distributed IO-level selection of web services. In NIPPERS, the syntactic description of a web service with its IO signatures is mapped to pre-defined conceptual tags, and the latter is further associated with peer addresses. With a tree-styled network structure, NIPPERS is able to achieve the efficient service discovery within a logarithmic search size complexity. Consuming the similar amount of network traffic, the Chord4S system [138] performs IO-level non-logic based service discovery in an extended Chord network. The idea of Chord4S follows its traditional data-peer mapping principle. Namely, each OWL-S service is encoded as a bit string by SHA-1. This process wraps the service input and output parameter concepts defined by a shared parameter type registry. Overall, the similar services are encoded with similar bit strings. On top of this, the services in a Chord ring are in addition cross-referenced by peers, which means that adjacent peers know similar services. This yields a CON, called virtual segments. To enable approximated search, Chord4S allows a query to use wildcards.

Hybrid P2P-based semantic service discovery: In [301], authors propose a method HACSemMa for PE-level distributed semantic service discovery in pervasive environments. HACSemMA models the services and user requests in a multi-dimensional space, called Hyperspace Analogue to Context, in which user specific properties, such as location, time, environment, health status, device status, etc., are mapped to a high-dimensional coordinate system that concerns not only the conceptual description but also numeric values. This allows to compare the concrete predicates (e.g. magnitudes of datatype property ranges) in PE and therefore supports a more accurate selection of services. In addition, HACSemMa is able to adapt to context changes about user preference and services. Handled by the underlying pervasive infrastructure, such a change is reflected to and incorporated by the upper discovery component in realtime.

Another fully-fledged IO-level semantic service discovery approach PYRAMID-S [285] is built on top of a hybrid P2P network. In particular, PYRAMID-S has its own language PS-WSDL for the semantic annotation of WSDL 2.0 services, which resembles SAWSDL but was invented even earlier. PS-WSDL works with extended domain ontologies and offers the abstract syntax for annotating the service capacities, parameter types, domain knowledge, provider information and QoS attributes. Apart from PS-WSDL, PYRAMID-S specifies its own XML-based query language USQL for service discovery. It offers specific operators to define user requests on different types of annotated features in PS-WSDL. With USQL, PYRAMID-S provides a hybrid matchmaking of PS-WSDL services over the syntactic service signature and semantic attributes above. No third party semantic matchmaker is hired in the PYRAMID-S system. Similar to JXTA, the network infrastructure of PYRAMID-S contains two types of peers. A gateway peer works as the leaf peer in JXTA, which is able to publish and query services, but not holding any index information for query routing; while a router peer behaves as the JXTA ultra (rendezvous) peer. It controls the access to its managed gateway peers and responses to maintain service indices. In addition, an extra service index synchronization mechanism is imposed on router peers, which facilitates the search with a fast lookup service.

The authors of [80] propose a JXTA-based approach JOSAP2P for the discovery of services in OWL-S, WSMO and SAWSDL. Similar to PYRAMID-S, JOSAP2P does not barely use the JXTA network, but additionally offers a method for semantically grouping peers. When peer joins in the network, a rendezvous peer is appointed as the group manager, which analyzes and compares the expertise of this new peer with existing groups. Overall, this process results in inter- and intra-group similarity links between peers. Similar to JXTA, the discovery of JOSAP2P is performed in two phases. In the first phase, a requester peer sends its request to its group manager peer, which computes the affinity between the request and overall semantics of the services in this group. Once the returned similarity value is smaller than a threshold, a JXTA walk is issued. It first traverses along inter-group links and attempts to ask those managed peers in its reached groups. It forwards the request to random groups, if no relevant result is found. JOSAP2P claims to use an open similarity measure that is agnostic with the type of service description.

Also relying on the JXTA network, the MWSP2P system [339] offers both the syntactic and semantic-based IO-level discovery of web services. For the syntactic-based selection, a presumed closed k-dimensional space of keywords is hired. Each service or request is encoded as a vector, in which the values at corresponding positions are set based on the occurrence of keywords. On top of it, a TF-IDF-based similarity measure is applied for the selection. As for the semantic-based selection part, MWSP2P combines the functional IO signature similarity with context-based attribute similarity, such as geographical location and distance. The matchmaking is performed by an additional node in the network, called context engine.

Another work, QoSSemMa [411], builds its QoS-aware service discovery model on top of a hybrid P2P network. In its underlying network infrastructure, each super peer records its known services on the leaf peers in its managed group. A query for certain service is delivered to the relevant groups. In QoSSemMa, authors extend OWL-S abstract syntax by adding extra QoS related concepts and properties. Each service in this work is presumed to be annotated with such QoS properties. The selection of service compares a query with each service by checking the relevance of IO parameters as well as QoS-based properties.

Unstructured P2P-based semantic service discovery: In contrast to the discovery within structured P2P networks, the search for services in unstructured P2P networks is harder in general, as unstructured P2P networks do not impose any data (service) index to participant peers. Each peer only has limited knowledge about network topology and data (service). In this situation, a commonly way for enhancing the search is to let peers to gradually form a CON, in which each peer observes and makes use as much knowledge as possible to direct queries.

In this paradigm, a representative approach, RS2D [23] by Basters et. al., leverages machine learning methodologies to enable each peer to learn the services from all of its direct neighbors before system starts. This forms a CON in terms of the services in 1 hop. On top of this, each peer forwards a query to a selected set of neighbors, under the constraint that minimizes the mixed Bayesian risk. RS2D takes advantage of OWLS-MX matchmaker for the local selection of services. Once selected, a matched service is added to the answer set of query. The latter will be routed back to its requester peer when the query TTL gets exhausted.

Bianchini et. al. present an approach, P2P-SDSD [30], for semantic-driven service discovery in P2P networks. In P2P-SDSD, a global ontology is hired to support the construction of a service semantic overlay that is formed by intra- and inter-peer semantic links between services. A semantic link is meant with the IO-level relevance between two services. When a peer receives a service request, intra-peer semantic links are exploited to select services from the local service repository. In addition, query is routed along the inter-peer semantic links. In order to maintain its CON, P2P-SDSD derives a shuffling-based overlay management protocol. The semantic overlay is modeled as a graph, where each edge represents a inter-peer semantic link between two peers. By this protocol, each peer continuously changes the set of its logical neighbors by periodically contacting a random neighbor. Subsequently, they in addition exchange the service information of their neighbors. As a result, P2P-SDSD system is able to quickly adapt to the dynamics of network and services (departure and arrival of peers, addition and deletion of services).

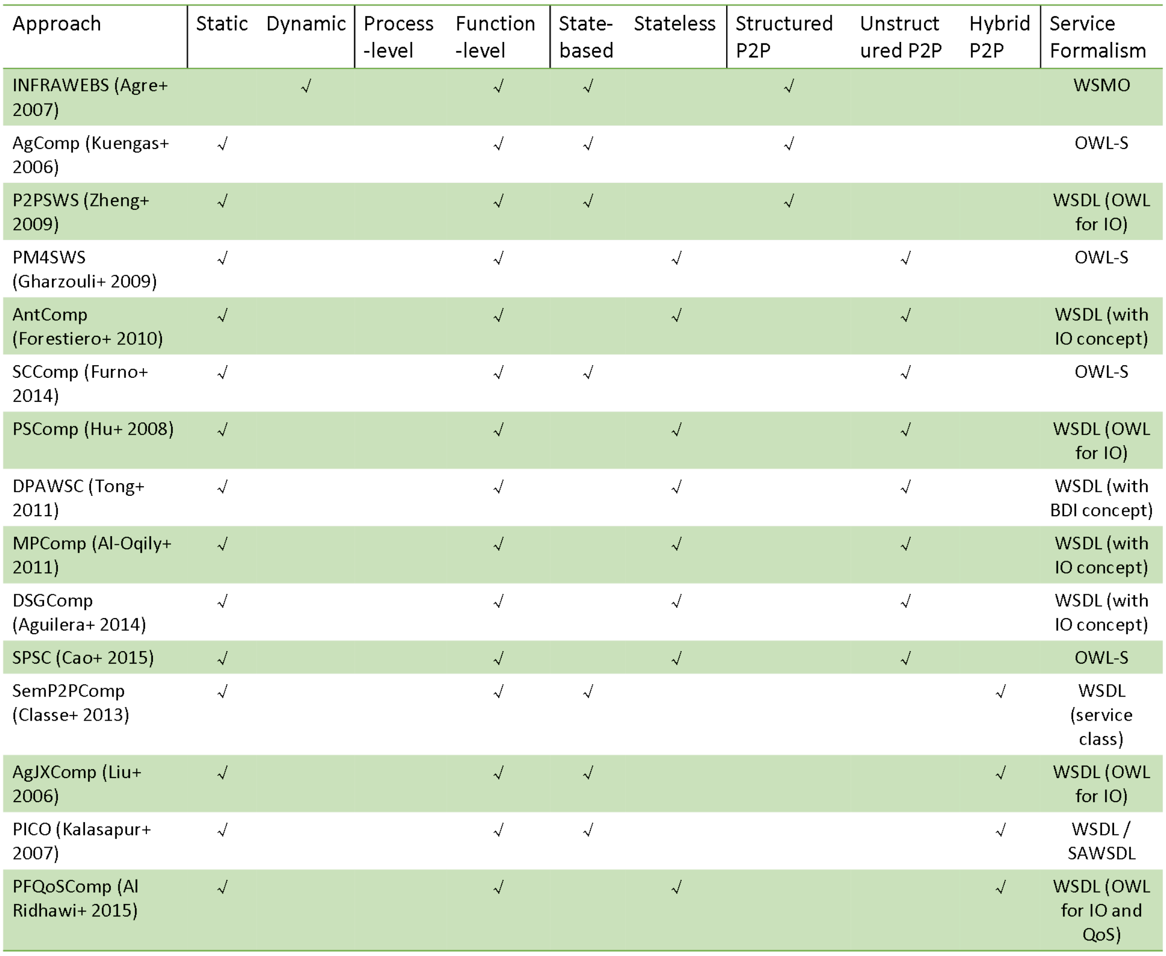
Lin et. al. propose QoSSemMaCC [227] to investigate the non-logic based IO-level QoS-aware WSDL service discovery in an unstructured P2P network-based cloud computing infrastructure. QoSSemMaCC process comprises two phases: service registering and service discovery. In the first phase, each peer that joins into the network propagates its service information via limited flooding. By this process, a CON is formed. Based on this knowledge, when searching for services, probabilistic flooding is hired for query routing, which enables each peer to route a query according to its local knowledge in terms of network traffic, service semantics, quality of service, etc..

The PAN approach [75] is derived for logic-based IO-level semantic service discovery in unstructured P2P-based social networks. In this context, each peer is called an agent. It is presumed to be able to provide some services, which IO signatures are annotated with concepts. Each connection between two peers are modeled as a semantic link, if the agents on both sides offer similar services. This work in addition models the preferential social network as a power-law graph. Some agents, working as hubs, can have some more links to other agents that have the expertise on different service topics. The discovery of services is delegated to a walker-based traveler. It is directed by agents collaboratively according to their local knowledge on services and connectivity degrees. In PAN, the P2P overlay and CON about services are identical. This might be true in social networks, but not necessarily hold in other scenarios.

The authors of the AntSrvDis [408] approach make use of an enhanced ant colony algorithm to conduct query routing for logic-based web service discovery in unstructured P2P networks. In AntSrvDis, each service is associated with a concept. The discovery of services is reduced to the search for similar concepts. Each query is a TTL restricted walker, which is sent from the requester and collaboratively directed by the peers on the path. The satisfaction of queries is treated as a kind of pheromone, indicating that the target peers have certain expertise on some concepts. This information is observed on-the-fly by peers and further organized in their routing tables. After each query, peers on the path will adjust their routing tables based on the query satisfaction information.

Another work AntCoMx [246] applies the similar ant-inspired algorithm for logic-based semantic web service discovery in unstructured P2P networks. As a component of the semi-automatic semantic service composition system AntComp [102] [101] [420] (cf. Section 2.3.3), AntCoMx responses to discover semantic services that conceptually match the sub-goals of a service composition request. In AntCoMx, the overall goal of a service composition request is given as a set of concepts. Each concept is treated as a sub-goal. AntCoMx contains two types of agents: ant agent and query agent. An ant agent is modeled as a walker and it is issued by a peer when arriving in the network. Periodically, it is forwarded to a neighbor peer and executes a pick-drop based algorithm, which enables the agent to pick up the information of new services and transmit it to other peers. Each peer maintains a co-use matrix that records the compositions in the past. When an ant agent arrives, such records are merged with the information piggybacked by the agent. A composition request is delegated to a query agent. The latter is directed collaboratively by the peers based on their co-use matrix and discovers the services relevant to the initialized sub-goal concepts.

2.3.3 Semantic Service Composition in P2P Networks



Approach	Static	Dynamic	Process -level	Function -level	State- based	Stateless	Structured P2P	Unstructured P2P	Hybrid P2P	Service Formalism
INFRAWEBS (Agre+ 2007)		√		√	√		√			WSMO
AgComp (Kuengas+ 2006)	√			√	√		√			OWL-S
P2PSWS (Zheng+ 2009)	√			√	√		√			WSDL (OWL for IO)
PM4SWS (Gharzouli+ 2009)	√			√		√		√		OWL-S
AntComp (Forestiero+ 2010)	√			√		√		√		WSDL (with IO concept)
SCComp (Furno+ 2014)	√			√	√			√		OWL-S
PSComp (Hu+ 2008)	√			√		√		√		WSDL (OWL for IO)
DPAWSC (Tong+ 2011)	√			√		√		√		WSDL (with BDI concept)
MPComp (Al-Oqily+ 2011)	√			√		√		√		WSDL (with IO concept)
DSGComp (Aguilera+ 2014)	√			√		√		√		WSDL (with IO concept)
SPSC (Cao+ 2015)	√			√		√		√		OWL-S
SemP2PComp (Classe+ 2013)	√			√	√				√	WSDL (service class)
AgJXComp (Liu+ 2006)	√			√	√				√	WSDL (OWL for IO)
PICO (Kalasapur+ 2007)	√			√	√				√	WSDL / SAWSDL
PFQoSComp (Al Ridhawi+ 2015)	√			√		√			√	WSDL (OWL for IO and QoS)

Fig. 2.12 Representative distributed semantic service composition approaches. They are discussed in Section 2.3.3. As for the centralized variants, we study them in Section 2.1.2 and show a classification of them in Figure 2.5.

Different environment properties can affect the quality of semantic service composition. In particular, an unstructured P2P network does not impose initial data indexes. Although this feature offers higher robustness and better network traffic load balancing (cf. Section 2.2.1), its loosely-coupled organization would create difficulties for service composition, as each peer has only a local view of the network topology and services. Previous efforts attacking the semantic service composition problem within unstructured P2P networks are unfortunately either not complete or tend to cause heavy network traffic overhead. In addition, no solution in this paradigm has presented any theoretical analysis about completeness and soundness. In Chapter 5, we describe our investigation into this problem and propose SPSC [42] that provides heuristics-based functional IOPE-level service composition with a lower bound of completeness and a formal proof of soundness.

In this section, we conclude the distributed semantic service composition methods in P2P networks. They are classified in Figure 2.12. As for the centralized service composition planners, we discuss them in Section 2.1.2 and show a classification of them in Figure 2.5.

Structured P2P-based semantic service composition: In [5], authors introduce an approach, called INFRAWEBS, for WSMO-based semantic web service composition in P2P networks. By INFRAWEBS, the service composition is guided by the algorithm that is designed for the run-time decomposition of user goal into sub-goals and the discovery of existing services satisfying these sub-goals. Generally, a composite goal can be modeled as a set of sub-goals. To fulfill the overall goal is to search for those services that satisfy all its sub-goals. In INFRAWEBS, this process has been simplified, i.e. a goal is recursively split into two sub-goals, in which the first one is an atomic goal and the second contains the rest. To each sub-goal, INFRAWEBS system tries to discover a service at runtime via its P2P-based search facility. This dynamic composition method can not guarantee the completeness, as it can not ensure the correctness of each division, i.e. it can not predict whether the produced sub-goals will be satisfied, or not.

Küngas et. al. present an approach, called AgComp [207], for agent-based OWL-S service composition in structured P2P networks. AgComp uses Chord as its base infrastructure that supports reliable indexing and efficient location of services. On top of this, a multiagent system is designed in order to discover and mediate keys that are associated to the semantics of services. Each key in AgComp corresponds to a literal that encapsulates the input and output of a service. And its corresponding agent responses to manage the access to it. A good feature of realizing service composition system on structured P2P networks is that the latter can support the indexing and basic discovery of services. AgComp is complete, as its underlying Chord network guarantees that each composition site is able to know all available services in the system and it searches exhaustively through the service space for a given request. However, a limitation of using structured P2P networks is that they have to spend extra network traffic overhead to handle the dynamics of network or services.

Zheng et. al. propose the method P2PSWS [414], for semantic web service composition in CAN networks. P2PSWS tries to integrate and organize semantic web service provider sites by a structured CAN P2P network, which facilitates the basic search and index maintenance of services. An associative relationship is established between each service and its provider site. On top of this, P2PSWS groups the service according to their semantics and area via a mechanism for the division of sites. Depending on the CAN indexing and replication facility, P2PSWS is helpful to overcome the problem "the failure of a single node" on the traditional CAN structure.

Hybrid P2P-based semantic service composition Similar to AgComp, AgJXComp [229] takes use of a hybrid P2P network JXTA as its basic infrastructure for multi agent system, which performs composition for WSDL services with IO parameter concepts in OWL. In AgJXComp, each agent is assumed to be capable of providing certain service, which is then registered as JXTA advertisement. Moreover, each agent leverages the JXTA network facility to search for its predecessor and successor in terms of maintained service IO parameters. Overall, this yields a service graph that records the transitions between services. On top of this, its service discovery and composition task is done by the search alone the successor relations between agents. Although building the composition system on the hybrid JXTA network, the messaging for service (agent) information propagation and synchronization still might cause heavy network traffic overhead. Another work SemP2PComp [66] also takes advantage of JXTA network for the composition of WSDL services. Differing with AgJX-Comp, SemP2PComp maps each service to a concept defined in a shared ontology. The composition task is reduced to the search for a set of similar concepts, which corresponding services are provided by peers in the network.

The authors in [173] propose PICO system for WSDL / SAWSDL service composition based on pervasive computing techniques. It derives a service-oriented middleware platform called pervasive information communities organization (PICO) to model and represent resources as services. Further, relations between services are modeled as directed attribute graphs, which are maintained by the participants in a hierarchical structure. The latter is created on top of the devices, aimed at exploiting the resource unevenness. As a consequence, the PICO system is competent to provide a kind of balancing mechanism between different devices with rich or poor resources.

Al Ridhawi et. al. propose a method, PFQoSComp [12], for stateless IO-level QoS-aware service composition in Chord-based multi-level hybrid P2P networks. In PFQoSComp, semantics of services are modeled based on a service ontology, which specifies the concepts about not only the parameter types but also service quality factors. In a forward chaining fashion, services are chained at IO-level, which considers the compatibility of parameter type as well as the service quality. The latter is conducted with a fuzzy set based model. Besides, this approach applies the heuristics principle in AI planning domain, which checks the similarity of the overall output of the current workflow with the requested output to determine solution validness.

Unstructured P2P-based semantic service composition: To the best of our knowledge, quite a few work addresses the service composition problem with unstructured P2P network-based dynamic settings. The PM4SWS system [117] [116] performs IO-level semantic service composition by forward or backward chaining. Each peer maintains a composition lookup table that records its observed historic compositions. Given a query, if no correct solution exists in the table, a peer tries to find a known service that can be chained to the current composition. This work leverages the classic flooding protocol, which causes in immense network traffic overhead.

Furno et.al. present a probabilistic flooding-based method SCComp [109] [110] [108] for OWL-S service composition. In SCComp, each peer routes a query to a set of selected neighbors, which are determined to be helpful to resolve the current sub-goal. Once a sub-goal has been resolved, extra transitive messages are needed for re-constructing the overall solution. Despite its selective routing strategy, the network traffic of this approach can still be heavy, as multiple peers may send duplicate messages to the same peer for the same sub-goal. Moreover, SCComp needs extra message exchange to adapt the network dynamics. Further, no factor about QoS or plan length has been considered by SCComp.

Hu et. al. propose an approach, called PSComp [152], for stateless distributed composition of WSDL services based on their IO parameter concepts. The environment setting of PSComp is a broker network without any initial data overlay or coordination. PSComp models the service requester as a service agent; the service provider as a request agent. Agents in both kinds can connect to arbitrary nodes in the broker network. When connecting to the network, a service agent advertises its services via flooding such that some of the other service agents can get to know. If a user want to request the network for a desired web service, the request agent takes this response and issues a query (called publication) to the network. This query is flooded to those service agents providing compatible (chain-able) services. On receiving a request, each service agent then attempts to contribute to the current partial workflow by exploring the possible forward chaining based on its known services. Once the workflow is augmented, a new query is sent to all the other service agents again. This process is iteratively performed until the solution is found. With the exhaustive forward chaining, the performance of PSComp is subject to the size of search space, since no heuristics is applied for pruning. In addition, the exhaustive and arbitrary chaining may lead to the failures with dead-ends.

Similar with PSComp, the DPAWSC [358] system offers stateless distributed IO-level composition of WSDL services. DPAWSC assumes that the composition is performed by a group of interconnected BDI (Belief, Desire and Intention) agents. Each agent knows a set of web services and their features are summarized as agent belief, which is formulated as a concept. The composition with DPAWSC is performed via flooding. When receiving a partial plan, each agent exhaustively searches its local service repository and tries to chain services to plan, as long as compatible IO parameters are found. This process is recursively conducted until the solution is found or all agents have been detected. Besides the risk of failure with dead-ends, DPAWSC works with a large unpruned search space, which yields a large number of unnecessary computation and network traffics.

MPComp [11] is proposed by Al-Oqily et. al. for stateless composition of web services with their IO parameter concepts. Compared to PSComp and DPAWSC, MPComp derives a selective flooding routing strategy. Such routing decision is captured based on the peer observation on queries, result feedbacks and user customized routing destinations. For this, each peer maintains a list of peers, which are estimated to be promising in terms of completing partial plans. Services are composed with MPComp in a forward chaining fashion. Without a guarding heuristics, each peer attempts to chain all local IO-compatible to the current plan. This would yield arbitrary augmentation of plan and therefore the failure due to dead-ends.

Aguilera et. al. propose a method, DSGComp [6], for stateless composition of WSDL services with IO parameter concepts in ad hoc MANET. In DSGComp, each peer can maintain a part of service dependency graph, which are created when peers or services are add / deleted. When an update happens, peer broadcasts this information to the neighbors reachable in a small TTL. When observing this, each peer updates its local dependency graph. Given a service composition request, a peer explores its local graph and try to find multi-paths connecting the requested input and output. If failed, it sends the partial plan to a set of peers which services has been used in the plan. This approach can not handle dead-ends and tends to conduct arbitrary chaining without heuristics. It does not guarantee the completeness and correctness.

The AntComp method, in [102], [101] and [420], enables the ant-inspired agent-based semantic service discovery and composition. Each peer maintains a co-use matrix that contains the pairs of classified reusable services observed in historical plans. In particular, a composition task is assumed to be pre-configured as a set of key classes in a query design phase. On processing a query, each peer selects the closest service classes and sends the query to a known peer which might be able to satisfy other keys. Besides the lack of a completeness guarantee, AntComp needs the extra design phase. This hinders the full automation of composition. A number of related work, such as [58] [111] [241], etc.., focus on the similar semi-automatic composition in distributed environments. Differing with the automatic composition concerned in this thesis, these approaches allow a user to put in a request with an abstract plan template. The composition target of them is actually the selection of services in order to create a solid plan. For more information about the semi-automatic composition means, we refer the reader to the literatures given in Section 2.1.2.

2.3.4 Selected Challenges in Semantic P2P Computing

Despite the promising new features and the previous research efforts, there still exist research questions in this area. In the following, we discuss several representative challenges, on which the research in this thesis focuses:

- *Collaborative semantic-based search and data replication in unstructured P2P networks*: Unlike structured or hybrid P2P networks, an unstructured P2P network does not impose any initial data indexes on peers. With loosely-coupled organization, peers do not have a global view of all the data in the network. In this situation, search performance is largely determined by the number of peers a query can traverse. However, the resulting heavy network traffic overhead should be avoided. Therefore, studies in this field proposed query routing mechanisms (cf. Sections 2.2.2 and 2.3.1) for search with higher precision but lower traffic cost, and there exist some works that take semantic technology into account. Nonetheless, almost every variant of these makes a peer decide on query routing depending only on the local knowledge of peers and always suggests only the immediate neighbor(s) as the next target(s) for routing. This can cause insufficient use of semantics and also loss of search performance. Thus, a semantic-based search model is needed. With it, peers should be able to collaboratively direct queries for higher overall search performance with reasonable traffic overhead, and at the same time, such collaboration takes advantage of the semantic knowledge to query routing paths that contain not only the immediate neighbor peer(s) but also those indirect peers with relevant items. To attack this problem, we propose the semantic-based search strategy S2P2P in Chapter 3. Another issue for search in unstructured P2P networks is data replication, which is particularly helpful in searching for rare items. In this perspective, the current contributions rely on non-semantic based knowledge formalisms (item ID or keywords) and statistical analysis. How to work with semantic web technology to precisely and collaboratively determine the number of replicas and destination peers is a challenge. To cope with this problem, we propose the DSDR approach in Chapter 4. To the best of our knowledge, it is the first semantic-based data replication strategy in unstructured P2P networks (cf. Section 2.2.3).

- *Efficient semantic service composition in unstructured P2P networks*: Distributed semantic service composition has attracted a lot of attention in the AI and semantic web communities. A challenging research question is distributed semantic service composition with limited knowledge. In the semantic P2P computing field, this problem can be modeled as semantic service composition in unstructured P2P networks, in which each peer has limited knowledge about services and network topology. The main target is to achieve higher recall of composition with lower network traffic overhead. In addition, it is challenging to find heuristics that can be used collaboratively by peers in order to protect the partial workflow from arbitrary augmentation and enable a system to handle dead-ends. Further, how the heuristics-based approach can guarantee what kind of completeness and soundness should be theoretically solved. There exist quite a few solutions targeting unstructured P2P-based semantic service composition (cf. Section 2.3.3). However, none of them targets all these crucial problems above. To this end, we propose SPSC [42] in Chapter 5.

- *Efficient semantic data selection for P2P search*: In spite of the promising features, such as machine-understandable formalism, high matching accuracy, etc., semantic computing can have high computational complexity. It would become an efficiency bottleneck in P2P-based realtime retrieval systems. It is possible that the peer data selection process would involve semantic-based computing over a large number of candidate items. Without a proper index on their semantics, the overall search efficiency is sensitive to the amount of data that a P2P system has. Therefore, a semantic-based indexing and fast comparison method is expected especially for the peer item-query matching process. In this regard, different semantic data have to be considered, e.g. a concept can be expressed by its logical unfolding, while a semantic service can have multiple semantically annotated IO parameter concepts and PE formulas. How to index them and further perform fast selection with an efficient aggregation means become challenges. Thus, we present our investigation in Chapter 6, the iRep3D repository for the efficient hybrid retrieval of semantically annotated 3D scenes, which provides the vision of efficient search for semantic data in P2P networks and collaborative virtual product design in intelligent 3D Internet.

2.4 Summary

In this chapter, we introduced the important notions and basic principles of the semantic web and P2P computing. Representative components in these paradigms and related work have been introduced. In addition, as the cross-disciplinary research field combining both, the main idea of semantic P2P computing is sketched out. It comprises the new features, selected challenges as well as the state-of-the-art works on semantic data (service) search and composition in P2P networks. This knowledge as a whole forms the background of the research results proposed in this thesis. As discussed in Section 2.3, one challenge of unstructured P2P search is how to use semantic-based peer knowledge to collaboratively conduct query routing with a reasonable tradeoff between search performance and network traffic overhead. In the next chapter, we will show our solution to this problem.

Chapter 3

S2P2P: Semantic Search in Unstructured P2P Networks

Unstructured P2P networks like Gnutella, FreeNet and Morpheus are widely used for decentralized file sharing. Flooding and k-walkers [237] are two of the classic ways to search for data items. The performance of their variants in terms of search precision/recall and network traffic is highly dependent on item information dissemination and query routing strategy. Many contributions take advantage of (restricted) flooding [132] [232] [242] or selectivity enforced gossiping mechanisms [59] [279] [369] to propagate query or item information to remote peers. These approaches commonly offer fairly high recall but tend to suffer from a relatively large network traffic cost. Besides, periodically fashioned gossiping of item information dissemination has the risk of propagating duplicated messages for identical items. Those messages do not pay off for the unpopular data that very commonly exists. K-walkers based approaches, such as [23] [353] [394] [95] [231], provide intelligent query routing strategies by means of machine learning techniques, network topology and query analysis. Although these variants are capable of alleviating the network traffic problem above, unfortunately, their routing suggestions usually contain only the next peer (commonly one of the immediate neighbors or a remote peer) that maintains the information helpful to answering the current query. This could cause insufficient use of item expertise information and further the loss of relevant results. Efficiently disseminating item information and making sufficient use of it for collaborative query answering is one of the main challenges in the field of information retrieval in unstructured P2P networks.

To this end, we propose S2P2P, a k-walkers based semantic search strategy for item information dissemination and query routing in unstructured P2P networks. Based on its demand-driven selective item information dissemination strategy, each peer in S2P2P is able to suggest paths for routing a query, which contains multiple expert peers, instead of merely a direct neighbor or remote peer. S2P2P is agnostic to the kinds of semantic descriptions of items and the data similarity measurements used by each peer for item selection. Our experimental evaluation provides evidence of

S2P2P's better performance in terms of search precision/recall and averaged precision regardless of item popularity distribution, in comparison with a selective flooding based search approach [232]. In addition, our experiments also reveal that S2P2P is as least as robust against network topology dynamics as [232].

In Section 3.1, we introduce the background and definitions for understanding our approach, which is detailed in Sections 3.3, 3.4 and 3.5. We show an illustrating example of S2P2P query routing and item information dissemination in Section 3.6. The discussion on robustness and the computational complexities of routing determination and item information dissemination are presented in Section 3.7. In addition, the experimental evaluation results are presented in Section 3.8. We discuss related work in Section 3.9 and conclude this chapter in Section 3.10.

3.1 Preliminaries

In this section, we briefly introduce the definitions and assumptions which are necessary to understand S2P2P approach. The classic way of describing data (e.g. a 3D model, a movie) residing in the Internet is by natural language and meta-properties, such as name, author, etc.. Lots of text similarity measures like strict string/word matching, n-gram, etc., have been derived for the purpose of retrieval. Inspired by the growth of Semantic Web, more and more data is described with semantic annotations in different logic-based formalisms, in terms of their formation, content, property and functionalities. With semantic annotation, it is possible to apply logic-based reasoning during data comparison and selection. It alleviates the risk of mismatching caused by the ambiguity and multiplicity of word meaning.

We assume that the item selection process (for query answering) of each peer is capable of determining the relevance between a query and local items described in one or a combination of those formalisms above. In addition, the semantic similarity function $sim(\cdot,\cdot) \in [0,1]$ of S2P2P is designed to be agnostic to the kind of semantic description. This facilitates the adoption of our approach to difference systems with customized concerns. Further, let us assume that all peers share a minimal vocabulary of primitive concepts, roles and predicates, out of which each peer p can canonically build its local knowledge base KB_p (e.g. a local ontology O_p of p) for specifying query and item in needed formalism.

Definition 1: *Item, item description.*
An *item* i provided and maintained by peer p consists of both data and metadata as defined by the item tuple $i = \langle td, sd, URI, pid, isz, n_s, da \rangle$ where td is the text-based description of i; sd the semantic annotation of i based on the local knowledge base KB_p of peer p; URI the item identifier; pid the id of the owner peer p providing i (including $i.da$); da the item data (e.g. mpeg file of a movie); isz the size of $i.da$; and n_s the number of available copies of i at p. The item tuple without the item data $i.da$ is meant with the item metadata and called *item description* ($i.desc$) of i. ■

With k-walkers based search, a requester peer delegates a user request to a set of K ($K > 0$, $K \in \mathbb{N}$) walkers. With a TTL limitation, each walker is forwarded by a peer with S2P2P query routing strategy to one of its direct neighbor peers according to the routing path suggestion. In case that the TTL value of a walker is exhausted, the walker backtracks along the inverted path targeting to the requester peer. The latter subsequently determines the satisfaction of the request based on the selected items by walkers. If the request is determined to be satisfied (unsatisfied), each of the K walkers is set as *Success* (*Fail*) (cf. Definition 2).

Definition 2: *Query.*
A *query* q is defined by the query tuple $q = \langle td,\ sd,\ req,\ A,\ Pa,\ Pa_{sug},\ t,\ st,\ pbd,\ TTL,\ n_d \rangle$ where td denotes the query keyword; sd the semantic annotation used to describe the semantics of the requested item; req the identifier of the requesting peer; $A = \{(res, its)\}$ the actual answer set for the query at run time. It consists of a set of pairs. Each maps an identifier res of a responder peer to an array its of its answered item descriptions; Pa the path of this query; Pa_{sug} the suggested path, which consists of a sequence of peers for routing q afterwards. It is empty, if no suggestion is available or q is backtracking; t the query issuing time; $st \in \{Issued, Success, Fail\}$ the query status where *Success* (*Fail*) means that the query q is satisfied (unsatisfied) and *Issued* indicates that the satisfaction of q has not been determined by the original requester peer req yet (or else that q is not issued by the current peer); pbd the piggybacked data set of a query; TTL the query time-to-live value; n_d the requested number of copies of the desired item. ■

We assume that each peer can provide and request any data item from known peers under the copyright restriction, which in our context limits the number of replicas of an item an individual peer can supply ($i.n_s$ in Definition 1) or request ($q.n_d$ in Definition 2). The item information dissemination process (cf. Section 3.5) of each peer, in our context, is to propagate item description only ($i.desc$ in Definition 1.), rather than the whole item with actual data file. As S2P2P is designed to be not adhering to specific kind of data description, it is convenient to use the terminology *semantic topic* (abbr. *topic*) tp, instead of those formalism-sensitive terms, such as *concept*, *semantic service*, etc., when describing the semantics of an observed query q or item i. Each peer p maintains a set of topics $TP_d(p)$ ($TP_s(p)$) (cf. Section 3.3) derived based on the semantic descriptions of its observed queries and items. In the following, the topics of observed demand $TP_d(p)$ (supply $TP_s(p)$) and the construction of them are introduced.

Definition 3: *Topic of demand (supply) observed by peer p.*
A *topic* tp_d (tp_s) of demand (supply) observed by peer p is defined by the topic tuple: tp_d (tp_s) = $\langle tsd, t_{lst}, P \rangle$, where tsd denotes the semantic description of this topic; t_{lst} the receiving time of the latest query (item description) that semantically similar to tp_d (tp_s); P is a list of membership entries. Each entry $(p', Q_{p'}, str)$ stands for the membership of a remote peer p' to a topic tp_d (tp_s). $Q_{p'}$ ($I_{p'}$) is the set of queries $Q_{p'}$ (item descriptions $I_{p'}$) issued by (propagated from) p' and sufficiently semanti-

cally similar with topic tp_d (tp_s). str ($str \in \mathbb{R}$) is the overall strength value of the observed demand (supply) of p' on topic tp_d (tp_s). Each remote peer corresponds to at most one entry $(p', Q_{p'}, str) \in P$. ■

3.2 Overview

Before the formal introduction of the S2P2P approach, we present an overview of it in this section. S2P2P is a semantic-based search scheme in unstructured P2P networks. It contains four parts: item selection, peer local observation, query routing and item information dissemination. With S2P2P, each peer p in the network is able to perform all four sub-processes independently.

Local item selection: On receiving a query q, the local item selection process of p adds the item description $i.desc$ of a known item i to the query answer set $q.A$, once i is determined to be similar to q. This record is supposed to be maintained by q during its entire routing, and therefore the requester peer can receive the result of this query.

Peer local observation: Besides the local item selection, peer p observes all the queries it routes as well as the received item information dissemination packages (IDP) (cf. Section 3.5). On top of this, p incrementally classifies selected parts of its observed demands (queries) and all the supplies (items in IDPs), respectively, into semantic-based categories, by applying k nearest neighbor (KNN) classification. In addition, p computes the strength about the demands of remote peers, in order to make item information dissemination decisions. Moreover, it determines the expertise topics of remote peers for future routing queries.

Query routing: On forwarding a query q, each peer p does not merely suggest an immediate neighbor peer to which q should be forwarded, but rather suggests paths of peers with expertise on the demand topic of q. Briefly, the routing strategy is twofold: **(1)** For a non-requester peer p, it computes the routing path suggestion for a walker it routes. Such a suggested path contains expert peers for the demand topic of q and, at the same time, minimizes the total inverse expertise gain per traffic cost, under the TTL restriction. This is done via a nearest expert based approximated heuristic that iteratively applies Dijkstra's algorithm for finding the next expert peer in order to augment the current suggested path. If q contains a non-empty path suggestion $q.Pa_{sug}$, p adjusts $q.Pa_{sug}$ by comparing the total inverse expertise gains per traffic cost of $q.Pa_{sug}$ with a new path Pa'_{sug} computed by p. The latter is formed by considering the relevant expert peers known by p together with the ones suggested by $q.Pa_{sug}$. This results in a new path suggestion for routing q. **(2)** For a requester peer p, it computes K path suggestions for K walkers. These paths traverse expert peers on the demand topic of the query and in the meantime their total inverse expertise gain per traffic cost is minimized, under the TTL limitation. This problem is resolved by another approximated heuristic that iteratively invokes the algorithm for case (1).

Item information dissemination: In addition, each peer p is able to selectively propagate its known item information along the path of a query walker, which is currently being forwarded. For this purpose, a subset $D(p,i)$ of peers on the (suggested) query path are selected as the destinations for the propagation of the description of a known item i. The completion of destination peer selection triggers the copy operation on i to the piggybacked data of q, which transmits the item (supply) information to remote peers. If a peer p receives a query q that is backtracking, p maintains a copy of the item description i from the query piggybacked data set $q.pbd$, if p is the dissemination destination of i.

3.3 Peer Local Observation

The path suggestion based query routing and item description dissemination processes performed by each peer p are depending on p's local observation in terms of queries (demands) and items (supplies). The queries comprise the ones issued or routed by p, while the items mean the item descriptions propagated to p and the ones owned by p. On top of this, the local observation of p can be derived as follows:

p's local view to network topology: $G(p) = (V(p), E(p))$ where $V(p)$ denotes the set of known peers by p (including p itself); $E(p)$ the set of known direct connections between peers in $V(p)$.

A set of topics of demands observed by p: $TP_d(p)$. Each peer p maintains a set $TP_d(p)$ of demand topics based on the continuously observed queries. For maintaining this data structure, a revised version of k-nearest neighbor clustering is applied when a new query q is observed. In addition, a simple sliding time window strategy is employed, which collects the queries received during the last t_0 time units (e.g. in last t_0 mins). Let $Q(t_0)$ the set of queries observed in time window t_0; δ ($\delta \in (0,1]$, $\delta \in \mathbb{R}$) the similarity threshold; $rt(q)$ the receiving time of q:

1. If $TP_d(p) = \emptyset$, p creates a new topic $tp_d = \langle tsd,\ t_{lst},\ P\rangle$, where $tsd = q.sd$, $t_{lst} = rt(q)$ and $P = \{(q.req, \{q\}, str)\}$. The computation of str will be presented later on.

2. If $TP_d(p) \neq \emptyset$, p computes a set $Q_k(q)$ containing at most k queries from $Q(t_0)$, which demands are most semantically similar with $q.sd$ and $sim_{sd}(q.sd, q'.sd) > \delta$ holds for each $q' \in Q_k(q)$. If $Q_k(q) = \emptyset$, create a new topic for q. Let $TP_d(p,q) \in TP_d(p)$ be the subset of involved topics. Each contains at least one query in $Q_k(q)$. The final topic tp_d^* for q is determined by:

$$tp_d^* = maxarg_{tp_d \in TP_d(p,q)} \big\{ \sum_{q' \text{ in } tp_d} sim(q'sd, q.sd) \big\}. \tag{3.1}$$

3. If the nearest topic tp_d^* of q has been determined during step (ii), p updates the triple of tp_d^*: $tsd = q.sd$, $t_{lst} = rt(q)$. If $tp_d^*.P$ does not contain a triple corresponding to $q.req$, p adds a new triple $(q.req, \{q\}, str)$ for the requester peer of q; otherwise updates the existing triple $(q.req, Q_{q.req}, str)$ by adding q to $Q_{q.req}$ and recalculating the demand strength str of peer $q.req$:

$$str(q.req, tp_d^*) = \frac{\sum_{q \in Q_{q.req}} q.n_d}{\sum_{\forall (p', Q_{p'}, str) \in tp_d^*.P} \sum_{q' \in Q_{p'}} q'.n_d}. \tag{3.2}$$

Expertise of peer $p' \in V(p) \backslash \{p\}$ *on topic* tp_s: Each peer iteratively constructs the supply topic set $TP_s(p)$ based on its observed items $I(p)$. On receiving the disseminated item description of i, p executes an iteration to classify i to a topic in $TP_s(p)$, which is similar with the formation of $TP_d(p)$ above. For this, k semantically nearest items $I_k(i)$ are selected from $I(p)$ but without considering the time window constraint. Instead, p computes the availability $ava(i, p)$. It measures (from p's local view) the availability of the disseminated information of i by its temporal (dissemination latency) and spatial (concatenated path length) factors:

$$ava(i, p) = [tcr - tfd(i)]^{-1} \cdot topoDist(i.pid, p)^{-1}. \tag{3.3}$$

where $tfd(i)$ is the time point when the description of i is disseminated by its owner peer (cf. Definition 1); tcr is the current time; while $topoDist(i.pid,\ p)$ is the topological distance from i's owner peer to p. It equals to the length of the concatenated path from $i.pid$ to p. $ava(i, p) = 1$, if p is the owner of i. On top of this, the expertise $exp(p', tp_s)$ of a remote peer p' on each topic $tp_s \in TP_s(p)$ can be estimated:

$$exp(p', tp_s) = \sum_{i \in I_{p'}} \{ava(i, p) \cdot sim(i, tp_s.tsd)\}. \tag{3.4}$$

Expertise gain of routing q *to a remote expert peer* p': For estimating the capacity of a remote peer p' with respect to answering a query, a peer p computes the expertise gain of p' (cf. Definition 4). Further, p concludes a sequence of expert peers and computes a path that traverses them. This path is used as a suggestion for routing q. The computation of the path suggestion will be detailed in Section 3.4.

Definition 4: *Expertise gain of a remote peer* p' *with respect to answering query* q.
Expertise gain $eg(p', q)$ measures the expertise of a peer p' with respect to answering a given query q. It is determined by the peer expertise $exp(p', tp_s)$ on a topic tp_s as well as the semantic similarity between tp_s and $q.sd$.

$$eg(p', q) = max_{tp_s \in TP_s(p')} \{exp(p', tp_s) \cdot sim(q.sd, tp_s.tsd)\}. \blacksquare \tag{3.5}$$

3.4 Query Routing

On forwarding a query q, instead of introducing only the immediate neighbor as the routing target, each peer p tries to compute path suggestions that cover multiple expert peers with respect to answering q, under the TTL limitation. A path suggestion Pa_{sug} traverses a list $EP = (p_1, \ldots, p_n)$ of expert peers $p_1, \ldots, p_n$. In the following, we introduce the path suggestion processes of non-requester peer and requester peer, respectively.

1. *For the non-requester peer*: As S2P2P bases on K-walkers, a non-requester peer computes a path for routing the current walker. The suggested path is supposed to traverse a sequence of expert peers on the query topic. In this context, the goal is to minimize the total inverse expertise gain per cost of that path, under the query TTL limitation.
2. *For the requester peer*: Differing with a non-requester peer, the query requester peer responses to compute up to K path suggestions, where K ($K \in \mathbb{N}^+$) is the number of walkers a query can have. Each path is suggested for one walker. It is supposed to traverse a sequence of expert peers on query topic. In this context, the goal is to minimize the total inverse expertise gain per cost of all the K paths, under the query TTL limitation.

Let p_i and p_{i+1} be the $i-$th and $(i+1)-$th expert peers on a path suggestion Pa_{sug}; $len(sPa(p_i,\ p_{i+1}))$ the length of the shortest path $sPa(p_i,\ p_{i+1})$ from p_i to p_{i+1}, based on the local view of p; We define the inverse expertise gain per traffic cost $w(p'', q, p') \in \mathbb{R}^+$ ($p'' \in \{p\} \cup EP,\ p' \in EP$) as follows:

$$w(p'', q, p') = \frac{len(sPa(p'',\ p'))}{eg(q, p')}. \tag{3.6}$$

For the case **(1)** above, a non-requester peer resolves the following optimization problem:

$$\begin{array}{ll} \text{minimize:} & \sum_{p' \in EP} w(p'', q, p'), \quad \text{where } p'' \in \{p\} \cup EP; \\ \text{subject to:} & \sum_{p_i, p_{i+1}} len(sPa(p_i,\ p_{i+1})) \leq q.TTL. \end{array} \tag{3.7}$$

This problem can be reduced to a relaxed Traveler Salesman Problem (TSP), in which:

- the salesman (walker) does not need to return to the starting point, since a query walker will backtrack along the inverse path anyway when its TTL is exhausted.
- a peer (an expert peer or a peer on the path between two experts) is allowed to be traversed multiple times, if this is needed due to the connectivity reason.

Inspired by the closest neighbor heuristics [167], we derive Alg.1 as an approximated solution. Denote $P_{exp}(q)$ the set of remote candidate peers that have expertise on a topic tp_s, which is sufficiently semantically relevant to $q.sd$.

Algorithm 1 $suggestPath(p,\ P_{exp}(q),\ q)$

Input:

p: the current peer which performs the path suggestion;

$P_{exp}(q)$: the set of expert peers with respect to answering q, from the view of p;

q: the query for which a path suggestion is going to made.

Output:

Pa_{sug}: path suggestion;

w_T: corresponding inverse expertise gain per traffic cost value.

```
 1: $Pa_{sug} \leftarrow \{\}$;
 2: $w_T \leftarrow 0$;
 3: $curMin \leftarrow \infty$;
 4: $curBP \leftarrow null$;
 5: $csp \leftarrow p$;
 6: $Cand \leftarrow P_{exp}(q)$;
 7: for each $p' \in Cand$ do
 8:     compute the shortest path $sPa(csp, p')$ from $csp$ to $p'$;
 9:     if $len(sPa(csp, p')) < curMin$ then
10:         $curMin \leftarrow len(sPa(csp, p'))$;
11:         $curBP \leftarrow p'$;
12:     end if
13: end for
14: if $curBP \neq null$ and $len(Pa_{sug}) + len(sPa(csp, curBP)) \leq q.TTL$ then
15:     concatenate $sPa(csp, curBP)$ to the tail of $Pa_{sug}$;
16:     $w_T \leftarrow w_T + w(csp,\ q,\ curBP)$;
17:     $csp \leftarrow curBP$;
18:     $Cand \leftarrow Cand \setminus csp$;
19:     $curMin \leftarrow \infty$;
20:     $curBP \leftarrow null$;
21:     goto line 7;
22: else
23:     return $Pa_{sug}$ and $w_T$;
24: end if
```

In Alg.1, denote Pa_{sug} the current path suggestion; w_T the total inverse expertise gain per traffic cost of the peers in Pa_{sug}; $curMin$ the length of the current shortest path that will be used to augment Pa_{sug}; $curBP$ the nearest expert peer from the current standing point peer csp (line 5). Alg.1 builds path suggestion iteratively: It first treats p itself as the standing point peer. p computes the nearest expert peer $curBP$ (lines 7 – 13). Subsequently, if the length of path that is being formed is smaller than current $q.TTL$, p concatenates the sub-path $sPa(csp,\ curBP)$ to the tail of Pa_{sug}. p then updates the total inverse expertise gain per traffic cost (lines 14 – 20). This triggers a new iteration (line 21) which regards $curBP$ in the last iteration as the new standing point peer. The algorithm runs until the length of concatenated path is larger than $q.TTL$ or no expert peer is found.

In order to decide the candidate expert peers $P_{exp}(q)$ that are the input of Alg.1, p computes a subset of topics $TP_s(p,q)$ from $TP_s(p)$ by measuring the semantic-based similarity of $q.sd$ with each supply topic $tp_s \in TP_s(p)$. If $sim(tp_s.tsd, q.sd)$ is larger than a threshold θ ($\theta \in (0,1], \theta \in \mathbb{R}$), p adds the peers in $tp_s.P$ (cf. Definition 3) to $P_{exp}(q)$.

In case that q contains a path suggestion Pa'_{sug} associated with a total inverse expertise gain per traffic cost value w^*_T, which were made by some peer p^* before, p updates its local view on network topology based on Pa'_{sug} and recomputes a path suggestion Pa_{sug} and new w_T. If $w^*_T < w_T$, p routes q according to the new path suggestion; otherwise, q will be sent according to the old path suggestion. p randomly forwards q to one of its immediate neighbor, if $q.Pa_{sug}$ = $empty$ and no path suggestion can be made by p based on its current knowledge.

For the case **(2)**, the requester peer suggests K paths for K walkers, one for each. Let EP_j ($j \in \mathbb{N}^+,\ j \leq K$) be the list of expert peers on the suggested path for the j-th walker. The problem of case (2) is reduced to a relaxed multiple TSP (mTSP) problem (formulated in Equation 3.8), in which:

- each salesman (walker) does not need to return to the starting point, since each query walker will backtrack along the inverse path anyway when its TTL is exhausted.
- a peer (an expert peer or a peer on the path between two experts) is allowed to be traversed multiple times by one or more walkers, if this is needed due to the connectivity reason.

$$\begin{aligned}
\text{minimize:}\quad & \textstyle\sum_{j=1}^{K}\sum_{p'\in EP_j} w(p'',q,p'), \quad \text{where } p'' \in \{p\}\cup EP_j;\\
\text{subject to:}\quad & \textstyle\sum_{p_i,\ p_{i+1}} len(sPa(p_i,\ p_{i+1})) \leq q.TTL;\\
& EP_j \cap EP_{j'} = \emptyset, \quad \text{for any } j \text{ and } j' : j \neq j'.
\end{aligned} \tag{3.8}$$

To resolve this, we apply the principle mentioned in [269] to transform the mTSP to TSP, and enable a requester peer to recursively compute approximated solution. The requester peer assumes that there is only one walker available and executes Alg. 1. Once Alg. 1 returns, the walker is forced to restart from the requester peer again and the requester peer executes Alg. 1 again for another path suggestion, until the completion of the K-th iteration. Besides, at the end of each iteration, the requester peer ignores the expertise of those peers that have been already considered in the former suggested paths. We formulate this process in Alg. 2.

In Alg. 2, denote $Pas_{sug}(q)$ the set of suggested paths. Its maximal cardinality is K. ws_T the set of their corresponding inverse expertise gain per traffic cost values. In each iteration (lines 5 – 14), it first picks up the expert peers that have not been used in former suggested paths (line 5). Subsequently, Alg. 1 is executed. It returns the suggested path $Pa_{sug}(q,\ j)$ for the walker j and its corresponding inverse expertise gain per traffic cost value $w_{T,j}$. In case that $j < K$ and there is no available (marked with $UNUSED$) and reachable (in $q.TTL$) expert peer, the requester peer

Algorithm 2 $suggestPaths(p,\ P_{exp}(q),\ q,\ K)$

Input:
p: the current peer which performs the path suggestion;
$P_{exp}(q)$: the set of expert peers with respect to answering q, from the view of p;
q: the query for which a path suggestion is going to made;
K: the number of walkers.
Output:
$Pas_{sug}(q)$: a set of path suggestions for query q;
ws_T: corresponding inverse expertise gain per traffic cost values.

```
1:  $j \leftarrow 1$;
2:  $Pas_{sug}(q) \leftarrow \{\}$;
3:  $ws_T \leftarrow \{\}$;
4:  for $j \ \leq K$ do
5:      $P_{exp,\ UNUSED}(q) \leftarrow$ unused expert peers;
6:      $Pa_{sug}(q,j),\ w_{T,j} \leftarrow suggestPath(p, P_{exp,\ UNUSED}(q), q)$;
7:      if $EP_j \neq \emptyset$ then
8:          $Pas_{sug}(q) \leftarrow Pas_{sug}(q) \cup Pa_{sug}(q,\ j)$;
9:          $ws_T \leftarrow ws_T \cup w_{T,j}$;
10:         label $\forall p' \in EP_j$ as $USED$.
11:     else
12:         break;
13:     end if
14:     $j \leftarrow j+1$;
15: end for
16: if $j \ < \ K$ then
17:     $Nbs \leftarrow$ randomly select $(K-j)$ neighbors;
18:     for each $Nb \ \in \ Nbs$ do
19:         $Pa'_{sug}(q) \leftarrow$ create a path suggestion containing $Nb$ only;
20:         $w'_T \leftarrow \infty$;
21:         $Pas_{sug}(q) \leftarrow Pas_{sug}(q) \cup Pa'_{sug}(q)$;
22:         $ws_T \leftarrow ws_T \cup w'_T$;
23:     end for
24: end if
25: return $Pas_{sug}(q)$ and $ws_T$;
```

terminates the loop and randomly selects $(K-j)$ immediate neighbors for the walkers left (lines 16 – 24).

3.5 Item Information Dissemination

With S2P2P, each peer p is able to perform demand-driven item description dissemination to remote peers. This process is triggered by the completion of the query routing decision (cf. Section 3.4) and finished before the query is forwarded. Inspired by [369], without issuing any extra message, the transmission is done by wrapping the description of an item i into a data structure called item dissemination package i_{dp} (cf. Definition 5) and copying i_{dp} into the piggybacked data of a query being forwarded. When receiving a query q that is backtracking, p checks the piggybacked item descriptions. It keeps a copy of the description of an item $i \in q.pbd$ if p is in the receiver set (cf. Definition 5) of i. Subsequently p updates its local knowledge of remote peer expertise.

Definition 6: *Item dissemination package of item i.*
The item dissemination package i_{dp} of an item i being propagated is defined by a tuple: $i_{dp} = \langle idesc, tfd(i), rcv, Pac\rangle$ where $idesc$ is the item description of i; $tfd(i)$ the time point when i is disseminated by its owner peer (cf. Definition 1); rcv the receiver peer set of this package; and Pac the concatenated path from the item owner peer to the current peer that is initializing this package. ■

Peer selection for dissemination: Each peer p decides to propagate an item i (description) to a set $D(p,i)$ of destination peers. The latter is a subset of peers selected out of the current query path $D_1(p,i)$ and the ones $D_2(p,i)$ in the suggested query path. Denote $D_0(p,i) = D_1(p,i) \cup D_2(p,i)$:

1. For each $p' \in D_0(p,i)$, p estimates the semantic utility $U(p,p',i)$ value of propagating the description of i from p to p':

$$\begin{aligned} U(p,\ p',\ i) &= \frac{\sum_{tp_d \in TP_d(p,\ p')} (str(p',\ tp_d)\cdot sim(tp_d,\ i.sd)\cdot ava(i,\ p))}{\sum_{tp_d \in TP_d(p,\ p')} str(p',\ tp_d)}; \\ TP_d(p,\ p') &= \{tp_d | tp_d \in TP_d(p) \textbf{ and} \\ &\quad \exists (p'',\ Q_{p''},\ str) \in tp_d.P \text{ s.t. } p'' = p'\}. \end{aligned} \tag{3.9}$$

   where $TP_d(p,p')$ is the set of demand topics of p', which have been observed by p; $sim(tp_d, i.sd)$ is the semantic similarity between each topic $tp_d \in TP_d(p,\ p')$ and i; $str(p', tp_d)$ refers to the strength of the observed demand of p'; and $ava(i,p)$ is the availability of i according to the observation of p. The intuition behind Equation 3.9 is that p tends to propagate the description of item i to a peer p', if the observed demand topic of p' is frequently asked and semantically similar with $i.sd$;

2. Select the top m ($m \in \mathbb{N}$, $0 \le m < |D_0(p,i)|$) peers with maximal utility values from $D_0(p,i)$ as the set of receiver peers of i. In case that $m > |D_0(p,i)|$, all the peers in $D_0(p,i)$ will be selected ($D(p,i) = D_0(p,i)$).

Subsequently, for each item i which description will be propagated, p instantiates a dissemination package i_{dp}. It sets $rcv = D(p,i)$ and computes concatenated path Pac from p to $i.pid$. If p is the owner of i, $i_{dp}.Pac$ contains p itself only.

Remote peer expertise maintenance: On receiving a query q that is backtracking, p checks the item dissemination package i_{dp} of item i in $q.pbd$. If $p \notin i_{dp}.rcv$, p skips to react on i; otherwise p adds i to local observed item set $I(p)$ and updates its observed expertise of remote peers (cf. Section 3.3).

3.6 Example

We illustrate the principled working of S2P2P-based semantic search with an unstructured P2P network shown in Figure 3.1a. Let us assume that the peer p_0 issues a query q (initial $TTL = 3$ and $K = 2$) for some desired items. As a part of its local knowledge, we assume that p_0 knows that three items i_2, i_3 and i_1 on relevant topics are maintained by peers p_6, p_3 and p_4, respectively. This knowledge is obtained by its local observation in the past. In addition, peer p_1 knows an item i_5 that is maintained by peer p_5. From the view point of p_0, all expert peers are labeled as $UNUSED$ (in Figure 3.1a, marked with orange boxes).

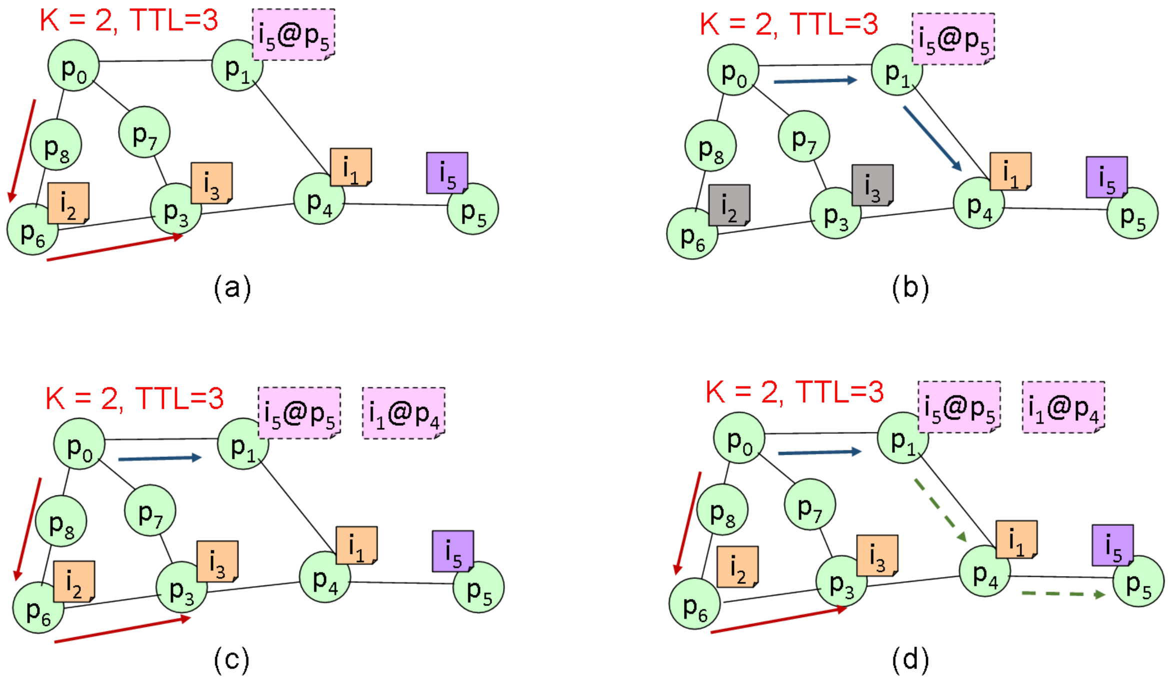


Fig. 3.1 Example of S2P2P query path suggestion.

As the requester peer of q, p_0 suggests one path for each walker of q. The target is to minimize the total inverse expertise gain per traffic cost. According to the routing strategy of S2P2P, p_0 accomplishes this work in a greedy manner: p_0 first determines the inverse expertise gain per traffic cost of each known experts, p_6, p_3 and p_4. For this, p_0 computes $w(p_0,q,p_6)$, $w(p_0,q,p_3)$ and $w(p_0,q,p_4)$. Let us assume that $w(p_0,q,p_6)$ is the minimum. p_0 concludes a partial path suggestion PS_1 for the first walker, which contains the shortest path ($p_0 \to p_8 \to p_6$) from itself to p_6. p_0 compares the length of PS_1 with $q.TTL$. Since $len(PS_1) = 2$ and it is not greater than $q.TTL$, p_0 is able to augment PS_1 by considering other experts. For this purpose, p_0 switches the standing point to p_6 and checks, from p_6, which expert peer yields the minimal inverse expertise gain per traffic cost. To do this, p_0 computes the shortest paths from p_6 to p_3 and p_4, respectively. At this moment, p_0 gets aware of that p_4 is not reachable from p_6 within the remained TTL ($TTL = 1$). Then it concludes the shortest path $p_6 \to p_3$ and concatenates it to the tail of PS_1. It follows that $PS_1 = p_0 \to p_8 \to p_6 \to p_3$ (red arrows in Figure 3.1a).

Since $len(PS_1) = 3$ and it equals to $q.TTL$, p_0 starts to suggest a path PS_2 for the second walker. Before doing this, p_0 marks p_6 and p_3 as $USED$ (gray boxes in Figure 3.1b). It means that these experts will not be considered for creating PS_2. For computing PS_2, p_0 switches the view point from p_6 to itself and selects the best expert peer, which yields the minimal inverse expertise gain per traffic cost. Since there are only one expert peer p_4 left, p_0 computes the shortest path $p_6 \to p_1 \to p_4$. As $len(p_6 \to p_1 \to p_4) < q.TTL$, p_0 suggests this path for the routing of the second walker (blue arrows). Once the two paths are suggested, p_0 sets this information to $q.Pa_{sug}$ of each walker and sends them out. Accordingly, the first walker is sent to p_6 via p_8 and the second is sent to p_1 in order to reach p_4.

Let us focus on the second walker (in blue) and assume that it reaches p_1 now. When p_1 routes this query, it first updates its local knowledge about peer expertise and gets to know that p_4 has the expertise on the requested topic (dashed box ($i_1@p_4$) in Figure 3.1c). Since p_1 additionally knows the expertise of p_5, which is relevant to the requested topic. p_1 recomputes the path suggestion for the walker, which TTL=2. For this, p_1 set PS_2 empty. As both p_4 and p_5 are reachable in 2 hops, p_1 computes $w(p_1,q,p_4)$ and $w(p_1,q,p_5)$. Let $w(p_1,q,p_4) < w(p_1,q,p_5)$. p_1 sets shortest path $p_1 \to p_4$ to PS_2. Then it switches the standing point from itself to p_4 and check which expert should be the next terminal for PS_2. As a result, p_1 concludes the shortest path $p_4 \to p_5$. After this, p_1 does not know any other experts and suggests $PS_2 = p_1 \to p_4 \to p_5$ to the second walker of q (green dashed arrows in Figure 3.1d).

3.7 Theoretical Analysis

In this section, we discuss the robustness and the computational complexity of S2P2P approach. For the robustness, we present how each peer, with S2P2P, behaves against the network dynamics, which is meant with the arrival and departure of peers as well as the addition and deletion of demand/supply topics; while for the computational complexity, we analyze the complexities of two important sub-processes of S2P2P: path suggestion and item information dissemination.

Robustness: S2P2P requires minimal amount of messages to be exchanged to react on dynamic changes such as peers leaving or joining the network. The arrival of a peer in S2P2P enabled P2P network triggers a simple handshake-advertisement: A arriving peer p broadcasts a one-hop advertisement (TTL = 1) to peers in its neighborhood, and waits for acknowledgement-messages. If at least one peer answers, p considers itself to be online and both peers mutually add each other into their local view on the network topology. No action will be triggered by the departure of a peer p from the network. If a peer stops to answer messages, the other part p' of the communication will detect its absence. Subsequently, the local view on network topology of p' is updated.

Furthermore, the maintenance of the demand/supply topic sets on each peer enabled with S2P2P scheme behaves in a lazy manner under the network dynamics in terms of peer arrival/departure. Each allocated membership entry (cf. Definition 3) is associated with a boolean flag indicating its availability. If the absence of p is detected by p', the latter sets the flags of those entries about p with "0", instead of deleting them. It means that they will no longer be taken into account by query routing and item description dissemination, until the flags are set to "1". This would save the cost for memory/disk (re-)allocation, in case that the absence of p is caused by a temporary network disconnection. Likewise, when p' knows the arrival of p, p' searches for the membership entries of p in its demand/supply topic data structure. If there exist some entries about p, p' sets the flags of them to "1". If no entry about p has been found, p' does nothing, as no demand/supply of p is detect at this moment.

Computational complexity: As no extra message is needed for query path suggestion and item information dissemination processes, in this part, we focus on the computational complexity of them. Denote v (e) the total number of peers (edges); $\boldsymbol{O}(s)$ the computational complexity of demand-supply topic similarity measure, which can vary from polynomial to NEXP (nondeterministic exponential) depending on the types of data formalisms [265]; l the initial TTL value of a query; K the number of walkers a query can have. We prove that (i) the computational complexity of query path suggestion is $\boldsymbol{O}(sv+Kl^2(e+vlogv))$; and (ii) the computational complexity of item information dissemination is linear with l.

Lemma 3.1. *The computational complexity of path suggestion is* $\mathbf{O}(sv+Kl^2(e+vlogv))$.
Proof: For routing a query q, a peer, based on its knowledge, first determines the expert peers, which supply topics are relevant to q. In the worst case, the peer knows all the peers in the network and all of them are determined to be experts for answering q. The computational complexity for expert peer selection costs $\boldsymbol{O}(sv)$.

For routing a query, a non-requester peer $p_{non-req}$ computes a path suggestion contains up to $(l-1)$ peers whose expertise is relevant to the requested topic. To do this, it executes Alg.1, which concatenates the shortest paths from one to another under the TTL limitation, such that the total inverse expertise gain per traffic cost of the path is approximatively minimized. In this process, each shortest path is computed via Dijkstra's algorithm that, in the worst case, costs $\boldsymbol{O}(e+vlogv)$ [105]. After each augmentation of the path, the standing point will be switched to the tail of the current path. In the next round, the expert peer with the minimal inverse expertise gain per traffic cost is selected. For this, $p_{non-req}$ will check the shortest paths from the last expert on the current path suggestion to the rest experts. It follows that each augmentation of the suggested path costs $\boldsymbol{O}((l-1)(e+vlogv))$. Since a path suggestion will be augmented by at most l times, the total computational complexity of the path suggestion for routing a walker is $\boldsymbol{O}(sv+l^2(e+vlogv))$.

A requester peer p_{req} computes at most K path suggestions for the K walkers of a query, one for each. This problem is approximatively resolved by Alg.2. p computes the paths one by one. Computed via Alg. 1, each path costs $\boldsymbol{O}(sv + l^2(e + vlogv))$. The total complexity of the path suggestion for K walkers of a query is $\boldsymbol{O}(sv + Kl^2(e + vlogv))$.

Overall, the computational complexity of S2P2P query path suggestion is $\boldsymbol{O}(sv+Kl^2(e+vlogv))$. ■

Lemma 3.2. *The computational complexity of the item information dissemination is linear with l.*
Proof: For disseminating the description of an item, a non-requester peer $p_{non-req}$ computes the semantic utility values for those peers on the current path suggestion. Since each walker can traverse at most l peers under the TTL limitation, the semantic utility values for all of them should be checked. For a candidate destination peer p', $p_{non-req}$ measures the semantic similarity between i and m demand topics (in the worst case). As the computation for demand strength and supplied item availability is incrementally done in advance, the complexity of deciding the information dissemination for i is $\boldsymbol{O}(lm \cdot s)$. For the requester peer case, it decides the item information dissemination for each of its planned path in the similar way. In total, it has to conduct this work for at most K times. Thus, this process in total costs $\boldsymbol{O}(K \cdot lm \cdot s)$. Therefore the computational complexity of the item information dissemination is linear with l. ■

3.8 Experimental Evaluation

In this section, we present and discusses the results of our comparative experimental evaluation. We test the performance of S2P2P search scheme in unstructured P2P networks with different configurations. Besides the experiment results reported in this section, S2P2P has been implemented based on the P2P framework and a real practice-oriented experimental evaluation (cf. Section 8.4) has been conducted for the MyMedia system (cf. Chapter 8).

Settings: For our experiments, we create unstructured P2P networks with 10000 peers and topologies based on random graphs (RG) with averaged connectivity 3.2 and random power law graphs (RLPG). The latter is known to be a realistic model in particular for social networks. Uniform at random and Zipf's law based item popularity distribution models are employed, which are used for many real-world item popularity rankings: The initial value of TTL of each walker in S2P2P is 10 and the number of walkers is 4. Both similarity thresholds δ and θ are 0.5. The time window size t_0 is set to be 600 seconds.

As a test collection, we use a random subset of 20k RDF linked data items (in files: `instance_types_en.nt.bz2` and `mappingbased_properties_en.nt.bz2`) taken from DBpedia[1] with its ontology (`dbpedia_3.7.owl.bz2`) O of 319 defined concepts and 1635 roles. We built peer ontologies through random sampling of 250 concepts and 1450 roles taken from O on average.

---

[1] `http://downloads.dbpedia.org/3.7/en/`

For non-semantic based random search, the relevance between items and queries are determined by the Levenstein edit distance between their topic terms. Since DBpedia does not provide the relevance sets for item queries, we use the following heuristics for relevance judgments: Item i about concept $i.sd = \tau(C)$ is relevant (a true positive) for query item i' about concept $q.sd = \tau(C')$, if any of the logic-based concept relations in $\{C \equiv C', C \sqsubseteq_1 C', C \sqsupseteq_1 C'\}$ holds. For query-item similarity determination, each peer simply checks the data concept subsumption relations based on mere the concept hierarchy but ignores the matching on properties.

We compare S2P2P with INGA[232] in terms of search performance and robustness. The latter introduces a shortcut based restricted semantic flooding search strategy. Particularly, each peer of INGA system creates a semantic network overlay (shortcuts of data) by query analysis, which is the common feature with S2P2P. We implemented two global table data structures for the access of shortcuts in content provider and recommender layers. In addition, both the maximum fanout and initial TTL value of the flooding are set with 3. This setup ensures that a query of INGA can traverse about 40 peers in a random graph based network with averaged connectivity 3.1. It is fair to S2P2P based query, which issues 4 walkers with initial TTL=10 for each.

Metrics: The metrics for unstructured P2P system are slightly different with the ones in the centralized systems. In particular, each query in the former system is not able to ask all the participant peers. With the classic IR metrics, the selected relevant items is supposed to be compared with all the (relevant) items in the entire system. In this case, the traditional metrics, like precision, recall, etc., are highly subject to the network scale, number of reachable peers and item distribution. To avoid this unfairness, we define the distributive search performance test metrics for the retrieval performance evaluation in unstructured P2P networks as follows: Let q be any query in the tested query set Q; T is the initial TTL value of each query; $I_{q,j}$ is the selected items at the j-th $(1 \leq j \leq T,\ j \in \mathbb{N})$ peer in the path of q; $I^*_{q,j}$ is the set of relevant items selected at the j-th peer in query path; $I^*_{tq,j}$ is the set of relevant items located at the j-th peer in query path. Name a query as a *successful query*, if it has collected at least one relevant item.

- *MAP@RE*: Macro-averaged precision (MAP_λ) at 11 recall levels (RE_λ) with equidistant steps of 0.1:

$$\begin{aligned}
MAP_\lambda &= \frac{1}{|Q|}\textstyle\sum_{q\in Q} max\{pre_{q,m} | re_{q,m} \geq RE_\lambda,\ \text{for } \forall\langle pre_{q,m}, re_{q,m}\rangle \in PR_q\};\\
pre_{q,m} &= \frac{\sum_{j=1}^{m} |I^*_{q,j}|}{\sum_{j=1}^{m} |I_{q,j}|};\\
re_{q,m} &= \frac{\sum_{j=1}^{m} |I^*_{q,j}|}{\sum_{j=1}^{m} |I^*_{tq,j}|}.
\end{aligned} \tag{3.10}$$

A set of PR_q of precision-recall $\langle pre_{q,m}, re_{q,m}\rangle$ pairs is computed for q at different number of hops *m*. Nearest-neighbor interpolation can be used for estimation of missed precision values for some queries at some recall levels.

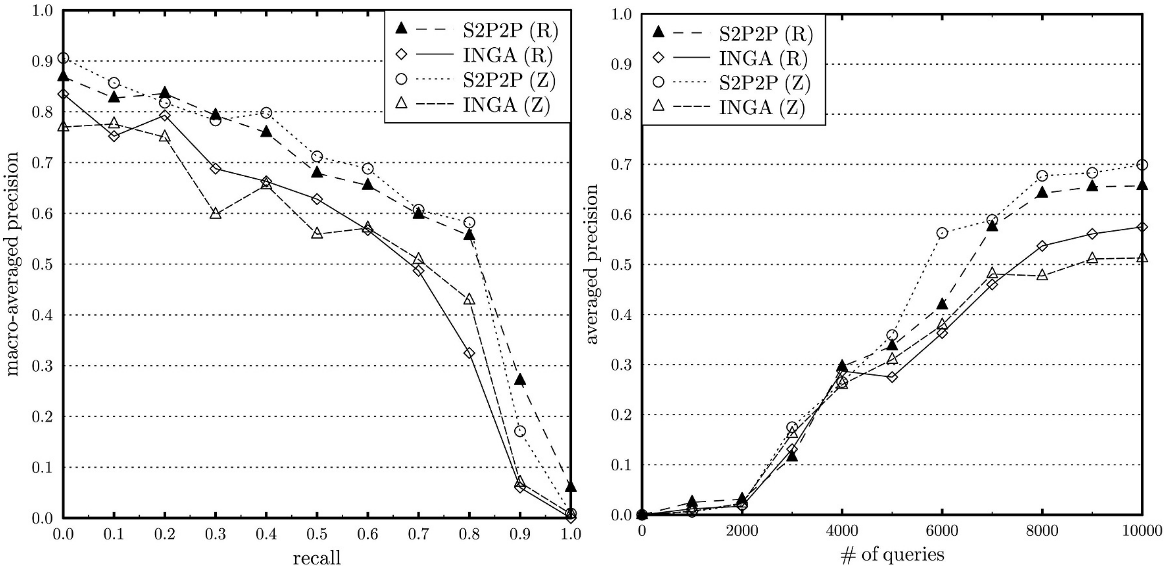
- *AP*: Averaged precision over all queries in *Q*:

$$ap = \frac{1}{|Q|}\sum_{q\in Q}\frac{|I_q^*|}{|I_q|}. \tag{3.11}$$

Search performance:

Search precision and recall. We compare the search performance of S2P2P and INGA in random graph based network. The item popularity distributions are uniform at random (*R*) distribution over all items and Zipf (*Z*) distribution ($\beta = 1.5$) over pre-clustered 79 topics. Our experiments revealed that S2P2P can significantly outperform INGA in terms of macro-averaged precision at recall (cf. Figure 3.2a) and averaged precision (cf. Figure 3.2b) regardless of the kind of item popularity distribution. Particularly, it achieved around 24% more precision at intermediate recall levels (cf. Figure 3.2a.) and around 20% more averaged precision (cf. Figure 3.2b) at 10 000 queries. The reason is that an INGA-enabled peer can not transitively propagate its detected shortcuts information; while a peer in S2P2P system is able to propagate the received item description information to those groups of peers located topologically farther. Because of this, a S2P2P request has a higher chance of meeting more relevant items. This merit of S2P2P effects when the maximal numbers of peers a (S2P2P or INGA) query can access are similar. In addition, both semantic search strategies appear to be relatively not sensitive to the kind of item popularity distribution. This is caused by a feature of search: the item information dissemination is mainly driven by the existence of item than the observed demands. In INGA system, the probing messages for building shortcuts is issued independently from the query.

Robustness. By the second experiment, we analyse the robustness of S2P2P and INGA for networks with random graph-based topology. After the processing of 8k and 18k queries, we randomly delete 25% peers from the network while add them randomly to the network when 15k queries are processed. The results reveals that the departure of peers results in a decrease of precision, since the semantic overlay structure was partially destroyed. The precision of both systems is not sensitive to the arrival of peers. Although the shortcuts of INGA or disseminated item information of S2P2P are diluted by the arrival of peers, the knowledge of the shortcuts and the paths targeting to disseminated items are remained. The averaged precision of both semantic search methods drops at each departure event (cf. Figure 3.3) but both systems were able to recover within almost the same time period.



(a) Macro-averaged precision at recall of S2P2P and INGA. (b) Averaged precision of S2P2P and INGA.

Fig. 3.2 Search performance comparison between S2P2P and INGA

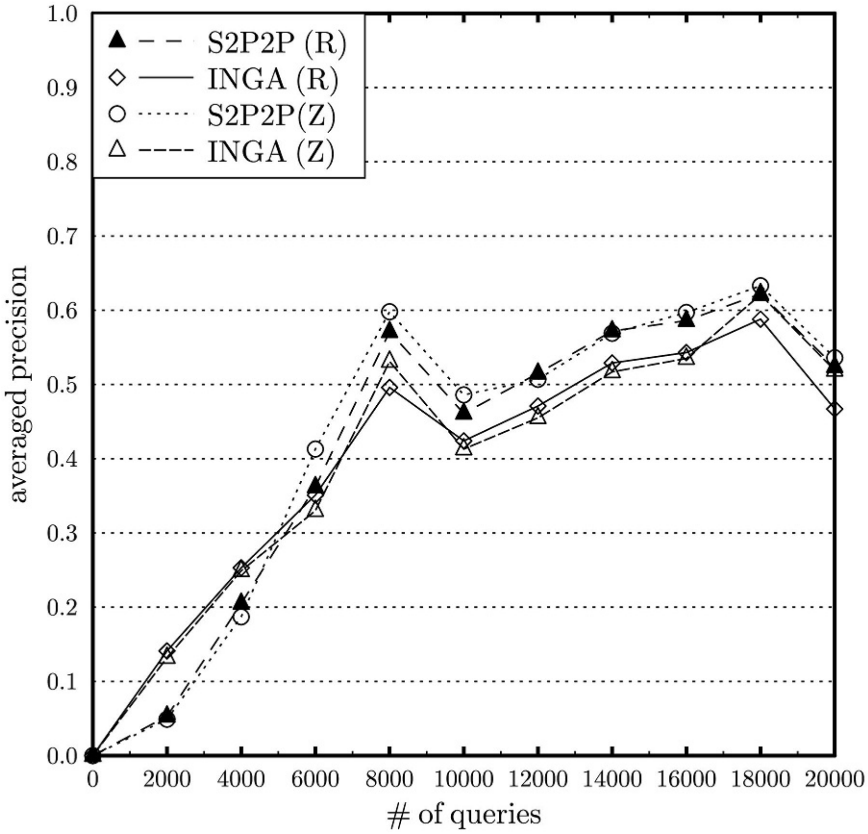


Fig. 3.3 Robustness: Averaged precision of S2P2P and INGA.

3.9 Related Work

The majority of the search strategies in unstructured P2P networks are variants of the classic flooding- or walkers-based search. In this section, we present their representative variants that are closely relevant to S2P2P, on the topic of semantic-based search in unstructured P2P networks. The search strategies in structured, unstructured and hybrid P2P networks are discussed and categorized in Section 2.2.2.

Flooding-based search strategies, such as [132] [232] [231] [242], commonly offer high search recall but suffer from relatively large network traffic cost. In the expertise-based semantic search Bibster [132], each peer advertises the topics of its maintained items via restricted flooding. Semantic links are distributively built among peers. Each peer is therefore enabled to route a query to at most s peers which are expertise on similar topic. Besides the risk of excessive message load in the network, a limitation of this approach comes into that the quality of semantic links is subject to network dynamics, as the links are built in one shot at the stage of peer arrival. In contrast, S2P2P maintains the expertise of peers dynamically in line with the query routing. In P-grid system [1], a virtual distributed search tree is holding by peers. A binary query is routed from a peer to at least one peer whose encoded expertise is determined to be "closer" to the query. Löser et.al. proposes a shortcut-based approach INGA [232], which enables a peer to perform selective flooding relying on the overlay built upon query analysis. A shortcut of the item on a remote peer is created if a query gets answers from it or detected by flooding. This would lead to large amount of comparisons during shortcut selection strategy for forwarding a query. The effort [242] introduces probabilistic flooding strategy aiming at minimizing the cost of excessive message transmissions via the proposed hop distance measure. Whether a peer forwards a query to its neighbors depends on the probability of this query hitting a matched resource. The latter is estimated based on the ratio of the nodes flooded over all in the network. The underlying assumption is that the resource distribution is uniformly random, which is not always true in practice.

K-walkers based search strategies commonly generate much less network traffic than the former flooding based variants. Biased query routing is always a feature of them. For this, machine learning, query analysis, item information dissemination, etc. techniques are used in order to build/maintain a semantic overlay network, by which a peer obtains more information for routing decision. In RS2D [23] and ACS [367], query routing relies on the learned network overlay, which is achieved by training the system with a set of labeled queries or by collaborative graph investigation. Likewise, peer in [249] is enabled to route a query to one of its semantic neighbor peers according to the local view of semantic network overlay. The latter is established by broadcasting peer profile when each peer joins in the network. The search performance of those systems built by means of similar ways is subject to the network dynamics. Liu et.al. proposes a method [231] to build super-peers that form a semantic overlay network by peers' caching the data of in particular the popular items, according to peers' storage capacities. Despite its increment of search precision/recall, hot spots are prone to appear in the network. The effort [394] presents a path-traceable query routing strategy based on the propagation of gains of query hits. For this, each peer analyzes the traversing queries and maintains a dynamic traceable gain matrix, which comes into the base of further query routing decision. In addition to the network load balance issue, the approach works relying on an assumption that a "hit" is a "match", which is not always true in decentralized retrieval system for complex objects, as the query satisfaction should be decided by the requester. Filali et.al. proposes walker-based search strategy [95] based on item information advertising and dynamic TTL heuristics. The latter offers incentive to a walker by decreasing TTL with a probability less than 1 when the walker finds relevant resource. Unfortunately, this would not to large extent increase the chance of find matching results, since the radio of a walker hitting a relevant result can be a constant if the overlay is fixed.

3.10 Summary

In this chapter, we presented a semantic-based search scheme called S2P2P for item information dissemination and query routing in unstructured P2P networks. The main contribution is that S2P2P offers novel query path suggestion heuristics, as well as a strategy for item information dissemination. The experimental evaluation shows that S2P2P outperforms the semantic search INGA in terms of search precision/recall and averaged precision. Besides, our experiment reveals that S2P2P search is at least as robust against network dynamics as INGA.

The time-to-live mechanism of unstructured P2P search is able to alleviate the network traffic overhead and thus guarantee a relatively shorter query response time. However, with this restriction, a query cannot be satisfied anyway, if the relevant items are located at peers unreachable within TTL. This yields an unsatisfactory search performance on rare items. To cope with this, data replication in unstructured P2P networks should be considered. In the context of this thesis, we will focus on and present a semantic-based replication method in Chapter 4. It shows how to leverage the benefits of semantic technology within the key processes of data replication, in order to make the replication decision more accurate. This is evidenced by the comparative experimental evaluations, in which combinations of the proposed replication scheme with different search strategies are tested.

Chapter 4

DSDR: Semantic Replication in Unstructured P2P Networks

Unstructured P2P networks like Gnutella, eMule, Morpheus and FreeNet are widely used for sharing user-generated multimedia content in a completely decentralized way. One classical means for peers to search for relevant data items in such networks is to perform k-walkers random search [237] (KW) in which case the relevance is determined by means of exact matching of query and item topic keywords. It is well known that the performance of KW in unstructured P2P networks can be significantly improved by additional use of data replication strategies with reasonable traffic overhead [355] [215] [69] [337]. On the other hand, semantic search in unstructured P2P systems like in Bibster [132], Remindin [353], SLS [29] and RS2D [23] determines the relevance of items based on the result of logic-based reasoning on formal ontology-based annotations of items. This enables peers to make more informed decisions for query routing and item selection, which may result in higher precision and recall with reasonable traffic and computational overhead [132]. Semantic replication schemes utilize semantic relevance computation for replication decisions, that is to decide how many copies of which semantically relevant items to best replicate to which peers.

However, it is not known under which conditions, and combined with what kind of semantic replication, KW can perform better than (a) its combination with near-optimal but non-semantic based data replication, and (b) semantic search without any replication in such networks. Besides, to the best of our knowledge, there does not exist a semantic replication scheme for unstructured P2P networks yet. In this chapter, we will provide answers to these major research questions. For this purpose, the scheme for dynamic semantic data replication, called DSDR, is presented. It can be used in combination with k-walkers based search in unstructured P2P networks. In addition, we conduct a comparative evaluation with existing but fundamentally different alternatives. In particular, this chapter shows that KW, when combined with DSDR-based semantic replication, can outperform its combination with the near-optimal, non-semantic based replication strategy P2R2 [337]. The main reason is that the DSDR scheme is guaranteed to dynamically replicate items to the most demanding peers with maximal expected utility based on a statistical and semantic analysis of query histories. In addition, the semantic gains of replication options are traded off with their estimated traffic costs.

We additionally show that depending on the item popularity distribution in the network, KW with DSDR can also outperform an approach for semantic peer expertise based search [132] as well as the S2P2P search (cf. Chapter 3) in unstructured P2P networks without replication. Besides, experimental evaluations give evidence for a significant outperformance of KW with P2R2-based replication by its alternative minimal coupling with DSDR in terms of local data lookup tables with semantic synonyms. Moreover, the evaluation reveals that both combinations of DSDR with KW are at least as robust against changes of the network than KW with P2R2-based replication. Finally, the DSDR scheme is agnostic to the kind of semantic description of data items and the selected method for semantic data relevance computation to be used by each peer. Though experimental evaluation has been conducted over a RDF test data collection from DBpedia, DSDR can easily be adjusted for replication of semantic services, or data items with other forms of ontology-based semantic descriptions as well.

The remainder of this chapter is structured as follows: In Section 4.1, we provide the underlying assumptions and definitions of the DSDR approach. The latter is then detailed in Sections 4.3, 4.4 and 4.5. The main results of the comparative experimental evaluations are presented in Section 4.8. We discuss the related works in Section 4.9 and conclude this chapter in Section 4.10.

4.1 Preliminaries

In this section, we briefly present the basic terms and assumptions which are required to understand the DSDR approach. In the context of this chapter, let us assume that (a) each peer can provide, replicate, and request any data item and replica from other peers it knows about, and that (b) all peers share a minimal common vocabulary of concept and role primitives (terms) out of which their users can canonically build the possibly different local ontologies O_p by use of the same ontology language like OWL2.

Definition 1: *Item, item concept.*
An *item* i provided and maintained by peer p consists of both data and metadata as defined by the item tuple $i = \langle l, \tau(C, O_p), URI, pid, sz, n_s, da \rangle$ where denotes l the label (topic term) of i; C the name of the semantic annotation concept and $\tau(C, O_p)$ its self-contained logical definition which describes the semantics of i in the local ontology O_p of peer p; URI the item identifier; pid the identifier of the peer p providing i; da the item data (e.g. movie file); sz the size of the item data $i.da$; and n_s the number of available copies of i at peer p. The semantic annotation concept $i.C$ is called *item concept* of item i, and the item tuple without the item data $i.da$ is called metadata or *item description* ($i.desc$) of i. The item label (topic) $i.l$ may correspond with the item concept $i.C$. ■

Definition 2: *Query, query satisfaction, query concept.*
A *query q* of a peer *req* is defined by the query tuple $q = \langle l,\ C,\ \tau(C,\ O_{req}),\ req,\ \{(res, its)\},\ \{Pa\},\ t,\ st,\ pbd,\ TTL,\ n_d \rangle$ where l denotes the query keyword (or topic of the query item); C the name of the *query concept* used to describe the semantics of the requested item and $\tau(C, O_p)$ its self-contained logical definition in O_{req}; *req* the identifier of the requesting peer; $\{(res, its)\}$ the answer set of the query at runtime. It consists of a set of pairs. Each element maps a peer identifier *res* to an array *its* of item descriptions that are found by *res* and match $q.l$; $\{Pa\}$ the set of query q paths; t the query issuing time; $st \in \{Issued,\ Success,\ Fail\}$ the query status where *Success* (*Fail*) means that q is satisfied (unsatisfied) and *Issued* indicates that the satisfaction of q has not been determined by the original requestor peer *req* yet (or else that q is not issued by the current peer); *pbd* the piggybacked data of a query containing information on provided items of peers along the query path; TTL the query time-to-live value; n_d the requested number of copies of the query item.■

The semantic relevance of an item i to a query q is determined by means of semantic matching $dc(C,C')$ of their semantic annotation concepts C,C'. The semantic relevance computation means $dc : O_p \times O_p \to [0,1]$. It can vary for different kinds of semantic annotations such as the example of the dc-function given in Section 4.8 for logic subsumption-based comparison of item concepts in OWL. The set of selected items in the query answer set $\{(res, its)\}$ is denoted as I_{sq}; the topic terms of items in I_{sq} are syntactically matching with the topic term of the query item. A query q is *satisfied* or successful if $I_{sq} \neq \emptyset$ and $q.n_d \leq \sum_{i \in I_{sq}} i.n_s$ holds.

4.2 Overview

In this section, we first provide a brief overview of the DSDR approach. This is followed by the detailed presentation and an illustrative example in subsequent subsections.

The overall working procedure of each peer p is shown in Alg. 3. During random search, in regular observation intervals, p observes all its received queries along with their piggybacked item descriptions. While the queries indicate the actual demand for items by requesting peers including p itself, the piggybacked data on items, including their semantic descriptions, provides p with knowledge about the actual supply in and the semantic overlay of the network from its local perspective. At the end of each observation period, p then predicts for each semantic concept C the number $n_r(C, p, ot_{next})$ of item copies p will probably be asked for in its next observation interval. It computes the plausibility $pl(C, p)$ of its predicted demand of items based on all demands it has observed in the past. It then forms for each concept C a demander group $dg_p(C)$ with other known peers that have requested items whose topics are semantically similar to the ones requested by p itself.

Once a group $dg_p(C)$ has formed at p, it will make a joint replication decision to determine how many copies of semantically relevant items i supplied by known peers outside the group should be replicated to which peer in this group $dg_p(C)$. For this purpose, p asks any other member peer p' in $dg_p(C)$ to provide their own predicted demands $n_r(C, p', ot_{next})$ of items on C, the expected utility

$EU(i,p')$ of replicating these item i to p', and its plausibility $pl(C,p')$. The replication decision, to which member peer p^* how many item replicas should best be replicated, is then made w.r.t. the maximum expected utility and plausibility, while the number of replicas is the minimum of $i.n_s$ and $n_r(C,p^*,ot_{next})$.

As shown in Alg. 3, applying the semantic replication DSDR does not affect the routing and item selection of KW. At the end of each observation interval, p invokes the group construction and replication decision processes.

4.3 Peer Local Observation

The dynamic local observation is the up-to-date local view of each peer p on the semantic overlay. Based on this, the demander group formation and semantic replication decision can be performed. We introduce dynamic local observation in this section. In addition, we assume that the current interval ot_m is the m-th ($m \geq 0, m \in \mathbb{N}$) interval of p since it joins into the network. When ot_m ends, p computes (updates) its dynamic local observation.

Definition 3: *Local observation of p over all past intervals.*
The observation record of p is a series of values:

- $Q(p)$ ($Q(p,ot_j)$): the set of all queries observed by p (in the j-th ($0 \leq j \leq m$, $j \in \mathbb{N}$) interval);

- $Q_C(p)$ ($Q_C(p,ot_j)$): the set of queries observed in all past intervals (ot_j) about concept C' sufficiently semantically similar to concept C;

- $UQ(p)$ ($UQ(p,ot_j)$): the set of all non-successful queries q ($q.st \in \{Fail, Issued\}$) (observed in ot_j);

- $UQ_C(p)$ ($UQ_C(p,ot_j)$): the set of all non-successful queries (observed in ot_j) about concept C' that are sufficiently semantically similar to concept C;

- $\mathbf{C}_{UQ}(p)$ ($\mathbf{C}_{UQ}(p,ot_j)$): the set of concepts of non-successful queries in $UQ(p)$ ($UQ(p, ot_j)$);

- M: the set of pairs each associating the query concept C of a failed query issued in ot_m by p with a set $Q_{dist,C}(p,ot_m)$ of queries observed by p in ot_m but issued by other peers $p' \neq p$. Any $q \in Q_{dist,C}(p,ot_m)$ is requesting for items on concepts C' that are sufficiently semantically similar with C ($dc(C,C') > \theta, \theta \in [0,1]$), where θ is set individually at each peer. Please note that $Q_{dist,C}(p,ot_m)$ contains exactly one query q for each originator $p' \neq p$ in the interval ot_m. If more than one observed query from p' asks for items on the same topic, the latest counts;

- $pop(C,p)$: the popularity of query concept $C \in \mathbf{C}_{UQ}(p)$ based on the requested number of replicas of observed queries, defined as:

$$\begin{aligned} pop(C,p) &= \textstyle\sum_{j=0}^{m} \frac{pop(C,p,ot_j)}{e\cdot(m-j)!} \in [0,1]; \\ pop(C,p,ot_j) &= \frac{n_C(p,ot_j)}{n_{total}(p,ot_j)} \cdot \frac{|UQ_C(p,ot_j)|}{|Q(p,ot_j)|}; \\ n_C(p,ot_j) &= \textstyle\sum_{q\in UQ_C(p,ot_j)} q.n_d; \\ n_{total}(p,ot_j) &= \textstyle\sum_{q\in Q(p,ot_j)} q.n_d. \end{aligned} \tag{4.1}$$

- $rc(C,p)$: the recentness of a query concept $C \in \mathbf{C}_{UQ}(p)$. It is computed based on the most recent issuing time of a query for items about C overall periods.

  $$rc(C,p) = \frac{q_{mr}.t - ot_{0,start}}{T\cdot m} \in [0,1] \tag{4.2}$$

  where $ot_{0,start}$ is the starting time of the 0-th (first) interval, $q_{mr}.t$ is the issuing time of the most recent query q_{mr} for items about C, T is the length of each interval.■

Based on the local observation, peer p predicts the number $n_r(C,p,ot_{m+1})$ of desired replicas of item about each demand concept $C \in \mathbf{C}_{UQ}(p)$, which is prone to be queried in the next observation interval (the $(m+1)$-th interval) of p.

Definition 4: *Predicted number of desired replicas on demand concept C of peer p in the $(m+1)$-th observation interval.*
$N = \{\langle C, n_r(C,p,ot_{m+1})\rangle\}$. Each element pair associates a demand concept $C \in \mathbf{C}_{UQ}(p)$ with a predicted number $n_{r,C}(p,ot_{m+1})$, which can be computed via time series analysis [133] with double exponential smoothing:

$$\begin{aligned} n_r(C,p,ot_{m+1}) &= s_m + b_m; \\ s_j &= \phi n_C(p,ot_j) + (1-\phi)(s_{j-1}+b_{j-1}); \\ b_j &= \psi(s_j - s_{j-1}) + (1-\psi)b_{j-1}; \\ s_0 &= n_C(p,ot_0); \\ b_0 &= \tfrac{1}{2}(n_C(p,ot_1) - n_C(p,ot_0)).\blacksquare \end{aligned} \tag{4.3}$$

The best values of weights ϕ ($\phi \in [0,1]$) and ψ ($\psi \in [0,1]$) are computed via applying Levenberg-Marquardt Algorithm (LM) [261] that efficiently resolves the following least squares problem:

$$\begin{aligned} MSE_m(\phi,\psi) &= \tfrac{1}{m+1}\textstyle\sum_{j=0}^{m}(s_j - n_C(p,ot_j))^2; \\ \text{minimize}: &\quad MSE_m(\phi,\psi);\ \text{subject to}: \phi \in [0,1], \psi \in [0,1]. \end{aligned} \tag{4.4}$$

where MSE refers to the mean squared error between the smoothed and observed values.

Plausibility $pl(C,p)$ refers to the overall strength of the demand on concept $C \in \mathbf{C}_{UQ}(p)$ observed by p during all its past intervals. Based on the evidence theory in [278], this value will be used as

Algorithm 3 Working procedure of peer *p*.

Input:

p: the peer running this procedure.

Output:

Void.

```
if $p$ receives a query $q$ then
    if $q$ is being forwarded then
        if $p = q.req$ then
            $p$ forwards a copy of $q$ to each of $k$ randomly chosen neighbor peers.
        end if
        $p$ selects a set of local items $I_p$ that match $q$ and adds $(p.id, \{i.desc\})$ to $q.\{(res, its)\}$ where $i \in I_p$;
        $p$ puts the item descriptions of locally maintained items into $q.pbd$;
        $p$ performs the local observation;
        if $q.TTL \geq 1$ then
            $p$ does $q.TTL \leftarrow q.TTL - 1$;
        end if
        if $q.TTL > 0$ then
            $p$ randomly chooses a neighbor peer $p'$ ($p' \notin q.\{Pa\}$) and forward $q$ to $p'$;
        end if
        if $q.TTL = 0$ then
            $p$ makes $q$ backtrack;
        end if
    else
        if $p.id \neq q.req$ then
            $p$ routes $q$ backward;
        end if
        if $p.id = q.req$ then
            $p$ waits for the return of other walkers of $q$ and determines the satisfaction of $q$;
        end if
    end if
end if
if $p$'s current observation interval ends then
    $p$ computes (updates) its local observation and prediction;
    $p$ executes GroupConstruction();
    $p$ executes ReplicationDecision($i.desc$) for each observed item $i$;
end if
```

the support for the expected utility computed in the replication decision process.

Definition 5: *Plausibility of the demand on concept C observed by peer p.*

Let $\mathscr{H} = 2^{\mathbf{C}_{UQ}(p)}$ the power set of $\mathbf{C}_{UQ}(p)$; $v : \mathscr{H} \rightarrow [0,1]$ the mass function subject to the properties:

$v(\emptyset) = 0; \sum_{H \subseteq \mathscr{H}} v(H) = 1$. Plausibility $pl(C,p)$ is computed by:

$$\begin{aligned}
v(H) &= \tfrac{n_H}{n_{\mathscr{H}}};\\
n_H &= \textstyle\sum_{C \in H} n_C(p);\\
n_{\mathscr{H}} &= \textstyle\sum_{H \subseteq \mathscr{H}} n_H;\\
n_C(p) &= \textstyle\sum_{j=\{1,\ldots,m\}} n_C(p, ot_j);\\
Bel(H) &= \textstyle\sum_{h \subseteq H} v(h);\\
pl(C,p) &= 1 - Bel(\mathbf{C}_{UQ}(p) \setminus C)). \blacksquare
\end{aligned} \tag{4.5}$$

4.4 Demander Group Formation

One key idea of DSDR is that each peer attempts to form a demander group (cf. Alg.4) for each of its own unsatisfied requests that were observed in the most recent observation period with peers that actually share semantically similar demands. A demander group is formed in a distributed fashion.

Definition 6: *Demander group.*
A demander group for query concept C at a peer p is:

$$dg_p(C) = \{p' : \exists q \in UQ(p) : q.req = p', dc(q.C', C) \geq \theta\}. \tag{4.6}$$

The same group is represented at each other member peer p as $dg'_p(C')$ and commonly recorded at every member as a tuple $\langle dgid, P_{dg}, C_{dg}\rangle$ where $dgid$ denotes the UUID of the group, P_{dg} the set of member peers p knows, and $C_{dg} = \{C\}$. $\blacksquare$

From the view of peer p' (cf. line 6) whose query $q' \in Q_{dist,C}(p, ot_m)$ with $q.C'$ is observed by p, depending on the status $q'.st$, p' replies to p with different kinds of messages as follows: If $q'.st = Fail$, then p' shares a semantically similar unsatisfied demand with p and therefore acknowledges the invitation to join the demander group on C. In addition, p' represents the initial demander group of p internally as $\langle dgid, \{p', p\}, C'\rangle$. If $q'.st = Success$ or any item i which is relevant for q' has been observed by p' during its current observation interval (which is not necessarily synchronized with the observation interval of p), then p' replies with a gossiping message (Gsp) containing the metadata of i to p. Hence p has a chance to decide on the replication of such item i within the group (cf. Section 4.5). If $q'.st = Issued$ then p' cannot yet determine the satisfaction of q' and replies with a message $Refuse$. The formation of a demander group is not synchronized, and demander groups can overlap. The introduced distributed process can result in multiple versions of representations of the same group at member peers. However, all versions of the same group have the same group identifier. If multiple peers send invitations for group formation to each other at the same time then the invitation from the peer with maximal lexicographical UUID is valid which then also becomes the leading peer of this group.

Algorithm 4 GroupConstruction(p)
Input:
p: the peer executes this group construction process.
Output:
Void.

```
Let $ot_m$ the current observation interval of $p$.
for each pair $\langle C, Q_{dist,C}(p, ot_m)\rangle$ in $M$ do
    Let $q$ the corresponding query of the query concept $C$;
    $p$ locally creates a group tuple $dg_p(C) = \langle dgid, \{p\}, C\rangle$;
    for each query $q'$ in $Q_{\sim C}(p)$ do
        $p$ sends a message $(p.id,\ q.\tau(C, O_p),\ q'.\tau(C', O_p),\ Dgc, dgid)$ to $q'.req$ (denoted as $p'$) and
        receives the reply, where $C'$ is the query concept of $q'$ under $O_p$;
        if $reply = Ack$ then
            $p$ does $P_{dg} \leftarrow P_{dg} \cup p'.id$;
        else if $reply = Gsp$ then
            $p$ executes $ReplicationDecision(i.desc)$ for any gossiped item $i$;
        else if $reply = Refuse$ then
            $p$ does nothing;
        end if
    end for
end for
```

4.5 Replication Decision

The completion of group construction on some concept C at peer p triggers the replication decision for each of its observed item i. Performed by member peers collaboratively, this process (cf. Alg. 5) is to decide how many replicas of what item shall be best replicated to which member peer.

The semantic gains (line 3) $g_r \in [0,1]$ ($g_{nr} \in [0,1]$) indicate the estimated benefit of p for (not) obtaining replicas of items on concept C' from the known item provider *pro* such that its demand for items on C' or semantically similar concepts C could be satisfied. Further, these semantic gains for p are then traded off with their estimated traffic costs, yielding the individual expected utility $EU(X)$ of such replication for p (cf. Definition 7). The predicted number of replicas $n_r(C, p, ot_{m+1})$ is used to compute the network traffic penalty depending on the communication overhead cm_{in} of the messages exchanged between peers during the replication decision process, and the amount of traffic produced by the replication. That is the number $min(n_r(C, p, ot_{m+1}), n_{C'}(pro))$ of replicated items of size $i.sz$ with traffic costs κ per unit of the size. The utility $u(\bar{X})$ of not replicating i bases on the inverse costs cm_{in}.

Definition 7: *Semantic gain, expected utility*
Consider dc, pop, rc as defined in Definition 3. Let i an item provided by peer *pro* on a topic which semantics is defined with concept $C' \in O_{pro}$ and C is a query concept of an unsatisfied query of p. The *semantic gains* g_r, g_{nr} and the *expected utility* EU of replication and non-replication of item i

Algorithm 5 ReplicationDecision(p, $i.desc$)

Input:

p: the peer executes this replication decision process;

$i.desc$: the description of item i, which replication is to be considered.

Output:

Void.

```
1: Let $C'$ the item concept of data item $i$ (maintained by peer $pro$) under $O_p$;
2: for any demander group $dg_p(C)$ $p$ belongs to do
3:     $p$ locally computes:
       (i) semantic gains $g_r(C,C',p,pro)$, $g_{nr}(C,C',p,pro)$,
       (ii) expected utility $EU(i,p)$ of replicating $i$ to $p$,
       (iii) predicted number $n_r(C,p,ot_{m+1})$ of copies of items that would be requested in the next
       interval (the $(m+1)$-th interval), and
       (iv) the plausibility $pl(C,p)$ of the demand on concept $C$;
4:     $p$ sends $i.desc$ to any other peer $p'$ in $dg_p(C)$, requests $p'$ to compute and return the same
       series of values:
       $[g_r(C,C',p',pro)$, $g_{nr}(C,C',p',pro)$, $EU(i,p')$, $n_{r,C}(p',ot_{next})$, $pl(C,p')]$;
5:     $p$ computes a candidate set:
       $P_T = \{p'|g_r(C,C,p',pro) > g_{nr}(C,C',p',pro)$, $p'$ in $dg_p(C)\}$;
6:     $p$ selects the recipient peer:
       $p^*{=}maxarg_{p' \in P_T}\ (EU(i,p') \cdot pl(C,p'))$;
7:     $p$ sends message to item $i$'s providing peer $pro$ for replicating item $i$ to $p^*$, if $p^*$ is $p$; $p$ sends
       message to $p^*$ telling it to replicate item $i$ otherwise.
8: end for
```

from pro to p are defined as follows:

$$
\begin{array}{ll}
g_r(C,C',p,pro) & = dc(C,C') \cdot rc(C,p) \cdot pop(C,p); \\
g_{nr}(C,C',p,pro) & = (1-dc(C,C')) \cdot (1-rc(C,p)) \cdot (1-pop(C,p)); \\
EU(X) & = P(X) \cdot u(X) + P(\bar{X}) \cdot u(\bar{X}); \\
P(X) & = \dfrac{g_r(C,C',p,pro)}{g_r(C,C',p,pro)+g_{nr}(C,C',p,pro)}; \\
P(\bar{X}) & = \dfrac{g_{nr}(C,C',p,pro)}{g_r(C,C',p,pro)+g_{nr}(C,C',p,pro)}; \\
u(X) & = (cm_{in} + \kappa \cdot i.sz \cdot min(n_r(C,p,ot_{m+1}), n_{C'}(pro)))^{-1}; \\
u(\bar{X}) & = -cm_{in}^{-1}.
\end{array}
\tag{4.7}
$$

where X ($\bar{X}$) denotes the event of (not) replicating item i from pro to p; $P(X)$ ($P(\bar{X})$) the probability of (not) replicating based on the computed gains; $u(X)$ the utility of replication in terms of a traffic cost penalty.■

Peer p receives the necessary values (lines 3–4) from all other group members p'. That enables p to identify those qualified candidates for replication (P_T in line 5) and then to determine the most beneficial target peer p^* (lines 6–7) within the group. This process is based on the product of the collected expected utility and supporting plausibility from each peer $p' \in P_T$

$$p^* = maxarg_{p' \in P_T}(EU(i, p') \cdot pl(C, p')). \tag{4.8}$$

where $EU(i, p')$ represents the expected utility $EU(X)$ of replicating item i to some peer p' of the group including p. Finally, only the target peer requests the provider peer pro for the item data on C. Providers may satisfy such requests for item data downloads to the target peers of demander groups on a first come first served basis. After replication, peer p^* (i) discards the local record of the group $dg_{p^*}(C)$; (ii) decreases $q.n_d$ of any $q \in Q(p)$ with demand concept C by the number of actually replicated copies and (iii) updates the local observation and prediction.

4.6 Example

We illustrate the principled working of DSDR-based replication combined with KW by a simple example of an unstructured P2P network $N = (V, E)$ which just consists of three sequentially connected peers with $V = \{p, p_1, p_2\}$, $E = \{(p, p_2), (p_2, p_1)\}$. Each peer has its local ontology defined in OWL and only one item is available which is provided by peer p_1. This item $i_1 = \langle taxi, Taxi, \tau(Taxi, O_{p_1}), uri(i_1), uuid(p_1), 1.5MB, 50, yellowcabs.mpg\rangle$ is labeled with the topic term $i_1.l = cab$ which formal semantics is defined by the concept $i.C = Taxi$ in the local ontology O_{p_1} of p_1 in OWL.

k-random search: Suppose that peer p is searching for one item i on the topic $i.l = taxis$ which formal semantics is defined by the concept $i.C = CAB$ in its local ontology O_p. Since p does not have any items at all, it randomly forwards the issued query $q = \langle cab$, CAB, $\tau(CAB, O_p)$, $uuid(p)$, $\emptyset$, $\emptyset$, $070312 : 1 : 15pm$, $Issued$, $[-]$, 2, 1$\rangle$ to a number k of its neighbor peers, in this case only peer p_2 with $q.TTL = 2$. Since peer p_2 has no items either it forwards the query to one of its neighbor peers, that is p_1. p_1 determines that its only item i_1 is not relevant for q since the item and query topic terms do not syntactically match. Peer p_2 then triggers ($q.TTL = 0$) the backward propagation of the random walker q to its originator p along the same path in reverse direction. On receiving q, p eventually determines that the query q is unsatisfied ($q.st$ = Fail). Meanwhile, peer p_2 issued a query q_1 on the topic "yellowcars" with semantic annotation concept $TaxiVehicles$ defined in the local ontology O_{p_2}.

Observed semantic overlay: Before returning the query q along its path, p_1 adds the semantic descriptions of all its items to the piggybacked dataset of q, in this case only the one for item i_1 $((Taxi, \tau(Taxi, O_{p_1})))$. As a result, upon receipt of the returning random walker q its originator peer p knows the actual semantic domain of items provided by p_1 which becomes part of its observed semantic overlay of the network. Peer p_1 decides when to further exploit some random walker it receives from p to communicate an update of its semantic item domain to p. Besides, peer p observed that its neighbor p_2 is demanding items with a certain semantic description.

Prediction and plausibility: Immediately after p's observation of q, p employs another independent thread to update the local plausibility $pl(CAB, p)$ of demand concept CAB in parallel with its main thread for routing. This yields available result for the replication decision for items on CAB, which is possible to happen in the future. Suppose that the current observation interval (the m-th interval) of p now ends. Based on the observed demands in ot_m, p updates its predicted number $n_r(CAB, p, ot_{m+1})$ of the requested replicas about concept CAB for the next interval. That is to apply the formulas in Definition 4 for computing s_m, b_m and $n_r(CAB, p, ot_{m+1})$ based on the computed volumes of s_{m-1} and b_{m-1} of the $(m-1)$-th interval. Please note that the random search process, however, continues independent from the prediction, plausibility update, the following distributed demander group formation and replication decision making by the peers.

Demander group formation: Since q was unsatisfied in the past period, peer p uses its actual local knowledge about the semantic overlay to form a demander group with those peers from which it received queries for items with semantically similar descriptions. In fact, p invites p_2 to form a demander group $dg_p(CAB) = \{p, p_2\}$ on CAB since both peers demanded semantically equivalent items on this subject in the past observation period: $dc(CAB, TaxiVehicles) = (\tau(TaxiVehicles, O_p2) \equiv \tau(CAB, O_p)) = 1$. Since p_2's query is unsatisfied as well the demander group on concepts CAB, $TaxiVehicles$ is formed and denoted as $dg_{p_2}(TaxiVehicles)$ at p_2 and as $dg_p(CAB)$ at p.

Replication decision within demander groups: In this example, p knows the supply information of the item i on p_1 and, in particular, its formal semantic description with the item concept *Taxi*. For deciding whether to ask for replication of i or not, both peers of the demander group then compute their own semantic gains, expected utility, predicted number of future requested replicas and plausibility of getting a replica of i_1 from the providing peer p_1. These values are sent to p on its request. We assume that peer p has the maximal product of expected utility and plausibility within the group, it requests the provider p_1 for the permission to download the desired number of replicas of item i_1 (data plus item description).

If DSDR is coupled with random search by means of local data lookup tables with semantic synonyms like $(taxi, cab, yellowcars)$ then next time p randomly searches for relevant items on the same or similar topic of $taxi$ it would not again miss but find them already as replica in its local store. Even without lookup table, DSDR can indirectly increase the non-semantic KW performance by replicating items that are also semantically relevant for queries - but which could be missed by peers in their non-semantic based item selection.

4.7 Theoretical Analysis

In this section, we theoretically analyze the robustness and computational complexity of DSDR approach. For the robustness part, we present how each peer, with DSDR scheme, behaves against the network dynamics, which includes the arrival and departure of peers as well as the addition and deletion of demand / supply topics; while for the complexity part, we analyze the computational and traffic complexities of two important sub-processes of DSDR: demander group formation and semantic replication decision.

Robustness: Similar with S2P2P, DSDR leverages the handshake based protocol for the arrival of a new peer p. When arriving, p broadcasts a one-hop message to a set of peers which has been configured as its neighbors or detected by distance restricted transportation means (e.g. bluetooth). Once this is done, p waits for the acknowledgement messages. If it receives a reply from a peer in the set above, p considers itself to be online and each of the both peers adds the other into its local view of the network topology. If the new peer contains an item i, it will consider i as a normal item during its local item selection and item information propagation as well as replication determination processes.

The departure of a peer p would lead to the inconsistent information about item and demander group member. In DSDR, each peer adds its item description to query piggybacked data for the item information propagation. Once an item is chosen for replication, the absence of its maintainer peer will be detected anyway during the request of actual data. The data requester peer subsequently updates its local view on network topologies and attempts to form a new demander group for this demand in the next observation period. If p leaves off during the group formation or replication determination phases, its absence will be detected by another member p' via the exchange of necessary messages. In DSDR, the communication for group formation and replication determination only happens between the group leading peer with a member peer. It follows that if p is not the leading peer, then p' is. In this case, p' simply updates its local view on network topology and delete p from the demander group. In contrast, if p is the leading peer, then p' is not. Once p' knows the departure of p, p' terminates the current group formation or replication determination process, updates its local view on network topology, and attempts to form a new demander group for this demand immediately.

Computational and traffic complexity: In this section, we prove that (i) the traffic complexity of both group construction and replication decision in the worst case is linear to the total number $|\mathbf{P}|$ of peers in the network; and (ii) the computational complexity of each of the both processes is also linear to $|\mathbf{P}|$.

Lemma 4.1. *The traffic complexity of both group construction and replication decision in the worst case is linear with* $|\mathbf{P}|$.

Proof: Let us consider the extreme case, in which the group leader peer p is the center of a network with star topology. If in the concerned time interval, p has observed queries with semantically similar demands from each peer in $\mathbf{P}\backslash p$, p will send message to each of them for group formation as well as ask for necessary values from them for determine a replication. It follows that the traffic complexity of both group construction and replication decision in the worst case is $\boldsymbol{O}(|\mathbf{P}|)$. ■

Lemma 4.2. *The computational complexity of both group construction and replication decision in the worst case is linear with* $|\mathbf{P}|$.

Proof: The computation complexity of them is $\boldsymbol{O}(|\mathbf{P}|\eta)$, where $\boldsymbol{O}(\eta)$ is the complexity of logic subsumption determination. It can vary under different formalisms. The overhead of updating local prediction is strongly alleviated by the iterative computation (cf. Definition 4) that uses the former result as much as possible. In particular, LM algorithm iteratively finds the proper weights ϕ, ψ near to the optimal. It terminates if (i) the difference of target function values $(\nabla MSE_m(\phi, \psi))$ is less than a given threshold ε or (ii) the total number of iteration reaches a given bound. The work [365] proved that LM algorithm terminates in $\boldsymbol{O}(\varepsilon^{-2})$ iterations. For efficiency, it is proper to set $\varepsilon = 1$ since its volume is at the level of $\boldsymbol{O}(mn^2) \gg \boldsymbol{O}(1^{-2})$ where n denotes the number of requested replicas. This guarantees that p can compute good values of weights in one iteration. For computing plausibility $pl(C, p)$ used for replication decision, peer p has to execute $\boldsymbol{O}(2^{|\mathbf{C}_{UQ}(p)|-1})$ addition operations for each observed query in the current interval. However, this can be performed off-line, which is totally parallel with p's routing process. ■

4.8 Experimental Evaluation

Settings: In this section, we present and discuss the comparative experimental evaluation results of the performance of KW with DSDR for different configurations of P2P networks. For this purpose, we simulate unstructured P2P networks with one million peers and topologies based on the model of random graphs (RG) and random power law graphs (RLPG). The latter is known to be a realistic model in particular for social networks. Further, we employed two models of item popularity distribution in these networks, which are used for many real-world item popularity rankings: Uniform at random (R) and Zipf's law (Z) based distribution. The value of k in KW is set with 3 and the initial TTL is 20 for each walker.

As a test collection, we use a random subset of 50k RDF data objects taken from DBpedia[1] with its ontology[2] O of 319 defined concepts and 1635 roles. The local ontologies of the peers are generated via random sampling of around 250 concepts and 1450 roles taken from O on average.

For non-semantic random search the relevance of items for queries is based on the Levenstein edit distance [220] between their topic terms. As mentioned above, query and item concepts C and C' are communicated among peers p_1 and p_2 together with their self-contained formal defini-

---

[1] `http://downloads.dbpedia.org/3.7/en/instance_types_en.nt.bz2; http://download.dbpedia.org/3.7/en/mappingbased_properties_en.nt.bz2`

[2] `http://download.dbpedia.org/3.7/dbpedia_3.7.owl.bz2`

tions $\tau(C, O_{p_1})$ and $\tau(C', O_{p_2})$ in the local ontologies in OWL. The semantic relevance of an item with annotated concept C with a query on concept C' is computed by semantic similarity $dc(C,C')$: $dc(C,C') = [1.0 \quad \text{if } C \equiv C'; 0.9 \quad \text{if } C \sqsubseteq_1 C' \textbf{ or } C \sqsupseteq_1 C'; 0.1 \quad \text{if } C \sqsubseteq_k C' \textbf{ or } C \sqsupseteq_k C', \quad k > 1, k \in \mathbb{N};$ 0 otherwise.] Since DBpedia does not provide the relevance sets for item queries, we use the following heuristics for relevance judgments: Item i semantically annotated with concept C is relevant (a true positive) for query item i' semantically annotated with concept C', if any of the logic-based concept relations in $\{C \equiv C', C \sqsubseteq_1 C', C \sqsupseteq_1 C'\}$ holds. The semantic relevance threshold θ for the demander group construction phase of DSDR is set to 0.5. All experiments are conducted via our semantic P2P simulation framework[3].

The experimental evaluations contain 2 folds: The first part checks the search performance of K-W combined with semantic and non-semantic replication strategies DSDR, respectively P2R2; while the second part concerns the benefit of the both replication strategies with semantic search strategies.

Metrics: For testing the performance of the P2P search strategies with(out) replication, we checks the MAP@RE, AP (definitions cf. Section 3.8) as well as the averaged cumulative recall (CRE), traffic utility (TU) and replica utility (RU). Let q be any query in the tested query set Q; T is the initial TTL value of each query; $I_{q,j}$ is the selected items at the j-th ($1 \leq j \leq T$, $j \in \mathbb{N}$) peer in the path of q; $I^*_{q,j}$ is the set of relevant items selected at the j-th peer in query path; $I^*_{tq,j}$ is the set of relevant items located at the j-th peer in query path. Name a query as a *successful query*, if it has collected at least one relevant item.

- *CRE*: Averaged cumulative recall over all queries in Q at the m-th hop:

$$CRE_m = \frac{1}{|Q|} \sum_{q \in Q} \frac{\sum_{j=1}^{m} |I^*_{q,j}|}{|I^*_{tq}|}. \tag{4.9}$$

- *TU*: Traffic utility measures the usage of overall traffic cost on satisfying user queries within an unstructured P2P system.

$$tu = \frac{\text{total \# of successful queries}}{\text{total \# of queries}}. \tag{4.10}$$

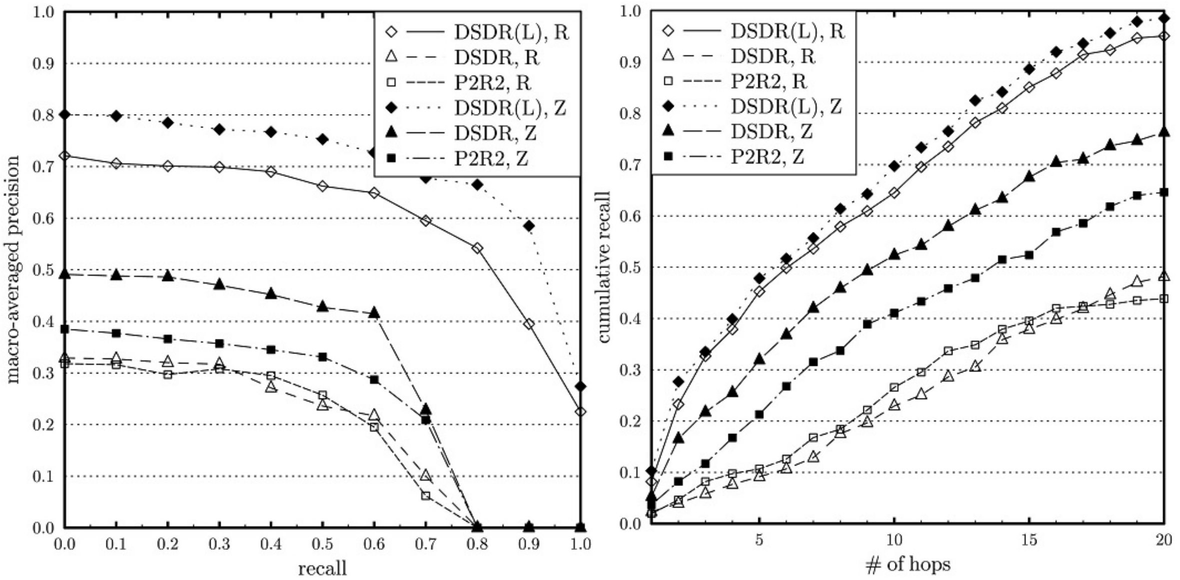
- *RU*: Replica utility measures the usage of replicas in terms of satisfying queries in the system.

$$ru = \frac{\text{total \# of successful queries}}{\text{total \# of replicas}}. \tag{4.11}$$

Search performance:

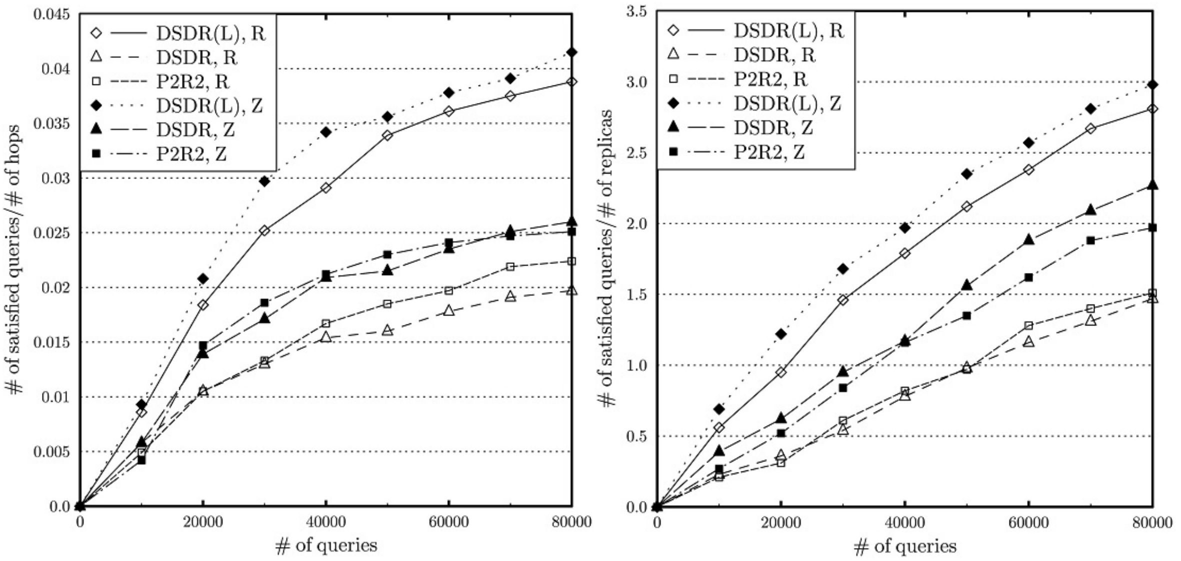
---

[3] `http://sourceforge.net/projects/dsdr/`



(a) Macro-averaged precision at recall of KW combined with DSDR and P2R2.

(b) Cumulative recall of KW with DSDR and P2R2.

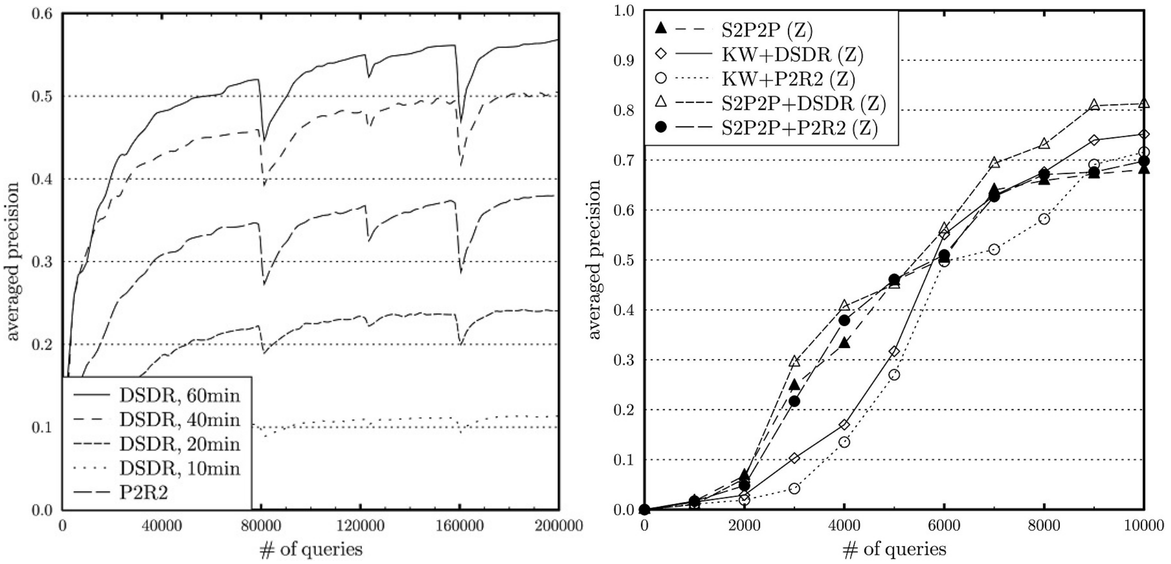


(c) Traffic utility of KW with DSDR and P2R2.

(d) replica utility of KW search with DSDR and P2R2.

Fig. 4.1 Search performance of KW with semantic and non-semantic replication methods.

K-random search with DSDR and P2R2. We compare the search performance of the same KW with different replication schemes: DSDR, DSDR plus lookup table (DSDR(L), and non-semantic P2R2 in a RPLG-based network with one million peers and 50k initial items. The item popularity distributions considered are uniform at random (*R*) distribution over all items, and the Zipf (*Z*) distribution ($\beta = 1.05$) over pre-clustered 127 topics. The latter is a well known model for the common search behavior of human users. The experiments revealed that for Zipf-based popularity



(a) Robustness of KW combined with DSDR and P2R2. (b) Averaged precision of semantic search S2P2P with DSDR and P2R2.

Fig. 4.2 Search performance of KW with semantic and non-semantic replication methods.

distribution, the KW when combined with DSDR-based replication can significantly outperform its combination with the non-semantic replication P2R2 in terms of precision (cf. Figure 4.1a) and cumulative recall (cf. Figure 4.1b). In particular, it achieved 27.3% more precision with similar volumes of traffic utility (cf. Figure 4.1c). When DSDR working without lookup table, syntactic based local item selection at peer p to a query for item i could yield non-semantic false negative given a true positive item i_1 at p. DSDR indirectly bridges this gap by enabling p to replicate another observed item i_2 semantically similar to i, on the observed demand of i. To some extent, the transitive syntactic relevancy between i and i_2 can exist. It follows that the replica i_2 can non-semantically match the future queries for i. Thus, non-semantic true positive can be achieved in future rounds. The experiment result evidences this case. Both precision and recall are indirectly improved over time since they increase monotonically with respect to the probability of p receiving a replica that is a true positive for future queries on the same topic.

DSDR needs some extra network traffic for the group construction and replication decision. For uniform item popularity distribution, KW with DSDR yielded lower traffic utility (cf. Figure 4.1c) than the search with P2R2; while the precision of both were similar in this case. Besides, the replication utility of the KW with DSDR is slightly higher than the one with P2R2 (cf. Figure 4.1d). Finally, the KW with DSDR(L) performs even better than its combination with DSDR or P2R2 regardless of the item popularity distribution, since the relevancy computations of search and replication are minimally coupled with lookup tables. Overall, these results clearly evidence the benefit of employing the semantic-based replication scheme DSDR rather than a non-semantic replication scheme like P2R2 in unstructured P2P networks with KW.

Random search with non-semantic replication by P2R2 has been shown to be highly robust against dynamic changes of the network topology [237]. We analyze the robustness of k-random search with DSDR for networks with RPLG-based topology containing one million peers which randomly issue 200k queries. After the processing of 80k and 160k queries, 200k peers are randomly selected and deleted from the network while the same number of peers are added randomly to the network after 120k queries have been processed. As expected, both types of topology changes resulted in a decrease of precision for some time, since either the replicas were removed or the semantic overlay structure was partially destroyed through peers leaving the network, or the replicas were diluted in case of peers entering the network. The average precision of k-random search with both DSDR and P2R2 replication schemes dropped at each change event (cf. Figure 4.2a) but both systems were able to recover within almost the same time period. In this context, not surprisingly, the leaving of peers had a greater negative impact than the arrival of new ones. For DSDR, the individual time interval T of each peer is crucial. In this experiment, for each run of DSDR, we manually control the expected value $E(T)$ of interval lengths, like 40min, by configuring the interval lengths of peers to uniformly distribute between $0.5E(T)$ and $1.5E(T)$, like 20min and 60min. The evaluation results revealed that shorter intervals (10, 20 min.) are less prone to such topology changes than larger intervals (40, 60 min.) while the recovery behavior is comparable to P2R2 but with significantly higher precision.

Semantic searches with DSDR and P2R2. We show the test of the search performance of (non-)semantic search approaches combined with replication strategies in a network with random power law graph-based topology. The configurations includes the run of S2P2P without replication, and the runs of KW (k=4, TTL=10) or S2P2P combined with the non-semantic replication method P2R2, as well as the same searches with the semantic replication scheme DSDR. The experiment result gives the evidence of the data replication effectiveness, which is able to increase the performance of search in unstructured P2P networks. Particularly, the combination of S2P2P with semantic data replication strategy DSDR yields the best precision after its stable network overlay has been established. In comparison with the combination of S2P2P with P2R2, the incentive by DSDR is larger. The reason is that a P2R2 enabled peer needs to know the query satisfaction of its observed query in order to judge the replication. However, this information is not provided by S2P2P peer as the query satisfaction is determined by the requester peer. In contrast, the group formation and subsequent replication decision of DSDR are conducted by a peer which request was judged to be unsatisfied by itself. This leads to adequate information to perform replication decision. Not surprisingly, k-random search can take advantage of P2R2 because of the syntactic matching happened during each peer's item selection process. It directly provides the information of "hit", by which the peer concludes the demand strength for its data replication based on P2R2.

Moreover, the evaluation shows that the search precision of S2P2P enabled configurations increases relatively more faster than the k-random search based systems. The reason is that S2P2P is capable of forming its own semantic overlay for conducting query routing, but k-random search can not. This is evidenced by the independent run of S2P2P without replication. However, this result also reveals that k-random search can be more robust under the network dynamics because of its capacity of working without semantic overlay. Further, after a sufficiently large number of queries (6000 in Figure 4.2b), the precision of k-random search combined with the both replication methods increases faster than the S2P2P-enabled system without replication. This evidences the merit of replication strategy, which can transitively propagate the item data (not only the item description) to remote peers. This is not achievable by S2P2P since the latter disseminates item description only.

4.9 Related Work

In this section, we discuss the representative works that are closely relevant to DSDR. There exist quite a number of approaches for data replication in P2P networks. For a discussion and classification of those variants working in particular with structured or hybrid P2P networks, we refer the reader to Section 2.2.3.

To the best of our knowledge, DSDR is the first dynamic semantic replication scheme for KW in unstructured P2P networks. It differs from previous efforts, like [69] [337] [113] [215] [237] [292] [354] and the ones in the survey [355], in that it incorporates not only the statistics of observed demand or supply but also the semantic relevance between them. The representative works about data replication in unstructured P2P networks are concluded in Section 2.2.3. In the following, we discuss the previous efforts that are closely relevant to DSDR.

In [69], a closed formula (square-root rule) is derived, which identify the optimal number of replicas to minimize the expected random search size rather than to improve the search performance in terms of cumulative recall and precision@recall like DSDR. The works [113] [292] [215] and [354] propose the proactive item replication strategies by which each item providing peer issues probing or limited flooding messages that either detect the item rareness or advertises its items. In addition to the risk of large network traffic, the actual demands of users in the network are not considered. In comparison with such proactive replication strategies, DSDR scheme appears to be more demand-oriented based on actual local observations.

In P2R2 [337], the replica distribution problem is reduced to the known multi-knapsack problem by regarding replicas as elements that are supposed to be put into bins which are representing target peers. Its replication scheme has been proven to converge to a 2-approximation solution of the problem under the following assumptions: the scheme is executed only in small networks in a sufficiently long steady state since the knapsack algorithm needs to know all the bins in advance. Unlike DSDR, this assumption hinders the applicability of P2R2 to large-scale scenarios.

In another replication strategy [325], each peer is prone to replicate its provided items to the remote peers that have larger query routing traffic, in order to guarantee higher replication utility. In contrast to DSDR, this replication strategy is less robust, highly sensitive to network dynamics.

Other approaches, such as [83] [82] [417], are based on peer grouping or partitioning widely used in P2P systems so far. While the first two systems make the assumption of global knowledge of the network which renders them unsuitable for unstructured P2P scenarios, the latter is restricted to the flooding radius thus, unlike DSDR, being very sensitive to the network topology.

4.10 Summary

In this chapter, we presented a semantic replication scheme, called DSDR, for walker based searches in unstructured P2P networks. The main contribution is that the search performance of k-random search with DSDR can outperform the same search combined with a near-optimal data replication P2R2 with about the same amount of traffic overhead. In addition, the search performance of semantic search with DSDR and P2R2 is experimentally evaluated as well. This provides evidence that the combination of semantic search and replication yields the best performance, in comparison to the non-semantic search KW with replications as well as the semantic routing based search S2P2P without replication.

In Chapters 3 and 4, we have presented the semantic-based search and replication strategies. A remaining issue is that any search with any replication cannot give a satisfactory answer if the desired item does not exist in the network at all. This motivates research on the semantic-based data composition in unstructured P2P networks, which aims at collaboratively composing the desired data on demand, when the desired features are described in a query. As a key aspect of this problem, semantic web service composition focuses on the composition of data in terms of their functionalities. In the next chapter, we describe research results on semantic service composition in unstructured P2P networks.

Chapter 5

SPSC: Semantic Service Composition in Unstructured P2P Networks

Recently, collaborative business processes are getting more and more important for distributed enterprises to fulfill today's needs for flexibility in production and service provisioning. Automating their construction improves efficiency in such distributed settings. Despite being platform independent, WSDL or Restful services appear not adequate for the automation due to the lack of expressivity in terms of standard support on service precondition and effect, which declare the criteria and result of service execution. Built upon semantic web technology, semantic web services [93], e.g. the most dominant OWL-S services [202], offer facility for defining these in a machine-understandable way and supporting the automatic service coordination.

In distributed business communities, each participant can join or leave the network at anytime and act as both service provider and consumer. Compared to client-server model or the specific networks with fixed topologies, P2P networks provide fairly mature protocols meeting these requirements. In the P2P-based semantic service composition paradigm, the structured or hybrid P2P-based approaches, such as [295], [414], [111], etc., benefit from respective data index coordination but at the cost of traffic overhead for maintaining their structured overlay in dynamic environments. Without imposing any initial overlay about data, approaches built on top of unstructured P2P networks can expect lower cost for overlay maintenance but higher robustness.

However, this task is challenging, as each peer in unstructured P2P networks does not have a global view of either network topology or services. Only a few efforts have been made to investigate this problem, and unfortunately each of them has limitations. A simplistic compensation for the lack of initial data indexes is flooding. As an example, the classic flooding employed by PM4SWS [117], [116] can cause heavy network traffic for online query answering. SCComp [108], [109], [110] enables peers to perform probabilistic flooding based on historical satisfied solutions, gossiping of state transition and analysis about query/network status. This may alleviate the heavy traffic overhead, but a peer still runs the risk of receiving duplicated messages with the same sub-goal.

Besides, no factor about quality of service (QoS) is considered for composition. In contrast, the AntAgt approach [101], [420], [102], yields less network traffic load by using a walker-based query routing strategy but suffers from its dependence on the user-specified plan template with slots. The need of each slot is represented by a class. Consequently, composition is reduced to the class-level discovery of semantic services. Obviously, AntAgt cannot fulfill the needs of automation.

To this end, we propose an approach, called SPSC, for automatic functional QoS-aware stateless semantic web service composition in unstructured P2P networks. In essence, the joint generation of complex service workflows by peers with SPSC relies on (a) the local semantic IOPE matching-based combination of OWL-S services with respective signature variable bindings, and (b) two heuristic strategies for query routing and memorization of potentially useful services for compositions. In particular, the local decision on which of the computed service workflow branches will be best to follow is based on their estimated semantic chaining scores and aggregated QoS values. As a result, SPSC peers jointly explore a heuristically pruned search space for best solutions to a given composition request within the given TTL without flooding the network and by mitigating the risk of failure with composition dead-ends at the same time. Since each peer continuously observes and accordingly updates its local knowledge on the semantic overlay, SPSC is robust against dynamic changes of the network topology and service descriptions. Further, we prove that the fully decentralized process of service composition is sound and has a reasonable lower bound of completeness for a given composition request, if it exists in the network being considered. Finally, our experimental evaluation revealed that SPSC can achieve a high cumulative recall with relatively low traffic overhead.

In Section 5.1, we briefly introduce the definitions and assumptions that are necessary to understand the SPSC approach, which is detailed in Section 5.2. In Sections 5.3 and 5.4, we show the theoretic analysis of a lower bound of completeness and prove the soundness of SPSC. We report our experimental evaluation results of SPSC approach in Section 5.6. We discuss the related work in Section 5.7 and summarize this chapter in Section 5.8.

5.1 Preliminaries

In this section, we introduce the stateless semantic service composition problem in unstructured P2P networks. Starting with the centralized version of stateless service composition problem, we present the definitions of the semantic service, composition request, parameter binding and composition problem. On top of this, we define the stateless composition problem in unstructured P2P networks.

Definition 1: *Semantic service S provided by a service provider p.*
A *Semantic service* (abbr. *service*) S is defined by a tuple: $S = \langle URI, I, O, P, E,\ p,\ qos \rangle$, where URI is the identifier of this service; I (O) is the input (output) parameter set of S. Each parameter $?x$ has its data type X defined in OWL-DL ontology O_p. P (E) is the precondition (effect) of S. Each precondition or effect is a logical formula in a conjunctive normal form (CNF) over the predicates

in A_p, classes in O_p and IO parameters of S. Each predicate in A_p is the first order logic (FOL) translation of the concept or property defined in O_p [18]; qos ($qos \in \mathbb{R}$, $qos \in (0,1]$) refers to the quality of this service, which is the overall service availability and computed by p according to its observed numbers of (un-)successful invocations; p is the service provider identifier. ■

For instance, a service is defined as:

S10 $= \langle$`http://.../shapingSrv2`$, I_{10}, O_{10}, P_{10}, E_{10}, p1, 0.95\rangle$.
$I_{10} = (Material\ \ ?m)$;
$O_{10} = (Product\ ?pro)$;
$P_{10} = tempLargerThan\ (?m, 200) \wedge qualityNotBad\ (?m)$;
$E_{10} = shaped(?pro)$.

$S10$ models an industrial production process that accepts a variable of type *Material* and outputs the *Product* $?pro$. To execute $S10$, its precondition P_{10} has to be satisfied in advance. It restricts that the temperature of the material $?m$ should be larger than 200 degrees and the quality of $?m$ is not bad. The effect after the execution of $S10$ is that the product $?pro$ is shaped. Services like $S10$ are commonly stored in a registry for the purpose of retrieval. The latter is aimed at searching for a list (or set) of semantic services that meet certain user specified criteria. We name the criteria as the request for semantic services. It can specify the input, output parameter sets, precondition and effect of some desired service.

Definition 2: *Request R for semantic services.*
A request R for semantic services is defined by a tuple: $R = \langle req, I, O, P, E\rangle$, where req is the requester identifier; I (O) is the input (output) parameter set. Each parameter $?x'$ has its type X' defined in the requester's ontology O_{req} in OWL-DL; P (E) is the precondition (effect). A precondition or effect is a CNF formula over the predicates in A_{req}, classes in O_{req} and IO parameters of R. ■

In order to satisfy a request to a service registry, semantic service matchmakers, like iSeM [200], can be applied, which matches the request with each service in the registry and determines the relevance. However, when there does not exist a service that matches a request, service matchmaker returns empty or null. As a solution, semantic service composition can be used to compose the desired services based on the services in a registry. In our context, we name the answer to a request as workflow (cf. Definition 4). It is an orchestration of services, which data flow between services are ruled by parameter bindings (cf. Definition 3). If an output parameter variable $?x$ of $S.O$ is bound with an input parameter $?y$ of $S'.I$, the data of $?x$ can be transmitted to $?y$ and used by S' as an input. This indicates a substitution $\{?z \mapsto ?x, ?z \mapsto ?y\}$ that means to replace all the occurrence of $?x$ and $?y$ in S and S' with a new variable $?z$. The idea of binding plays an important role for the service composition without facts. We define the parameter binding and workflow as follows.

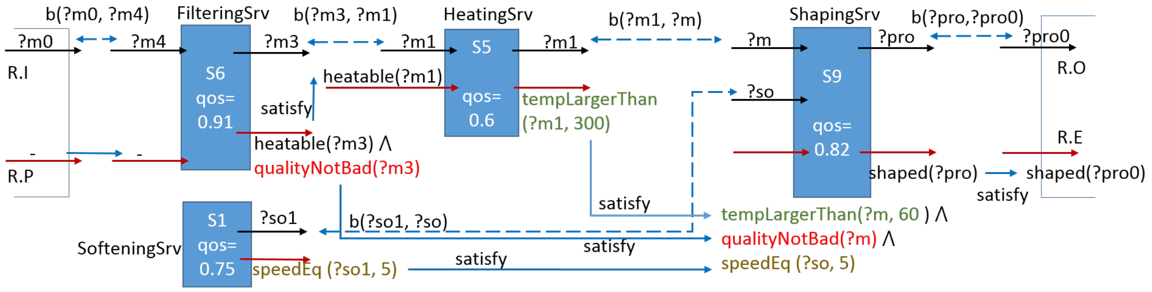


Fig. 5.1 An example workflow wf of services with parameter bindings.

Definition 3: *Binding of service parameter $?x$ and $?y$.*
A binding $b(?x,?y)$ of service parameters $?x$ and $?y$ of services S, respectively S' is defined by a tuple $\langle ?x, ?y, \varphi \rangle$ where φ is a substitution $\{?z \mapsto ?x, ?z \mapsto ?y\}$. It means to replace all the occurrence of $?x$ and $?y$ with a new variable $?z$. ■

Definition 4: *Workflow wf.*
A workflow $wf = \langle \mathscr{S}_{wf}, B \rangle$ is an orchestration of semantic services. It contains a set $\mathscr{S}_{wf}$ of services and a set B of parameter bindings between services in $\mathscr{S}_{wf}$. In wf, we name the side that starts (ends) with $R.I$ and $R.P$ ($R.O$ and $R.E$) as its *left* $L(wf)$ *(right* $R(wf)$*) side*. A workflow wf is *correct* with respect to answering to a request R, iff:

1. each input parameter of a service in wf is bound with an output parameter of another service in wf or an input parameter in $R.I$;
2. wf plugs into the requested service R. In detail, for any input parameter $?x$ of type X in wf, $\exists$ $?x'$ of type X' in $R.I$ or $S.I$ of a service S in wf: $X' \sqsubseteq X$; for any $?y'$ of type Y' in $R.O$, $\exists$ an output parameter $?y$ of type Y in wf: $Y \sqsubseteq Y'$;
3. given the parameter bindings, there is no contradicting literals in $(\bigcup_{S \text{ in } wf} S.E) \cup R.P$;
4. all preconditions of the services in wf are satisfied, given the parameter bindings. Formally, $\forall S \in wf$, $\exists$ a set $\mathbb{S}$ of services in wf with bound parameters: $(\bigwedge_{S' \in \mathbb{S}} S'.E) \wedge R.P \implies S.P$;
5. $R.E$ can be implied based on the truth of all the effects of services in wf and parameter bindings: $\bigwedge_{S \text{ in } wf} S.E \implies R.E$. ■

As an example, there is a correct workflow wf (cf. Figure 5.1) that answers to a request $R = \langle req, I, O, P, E \rangle$, where $R.I = (Material\ ?m0)$; $R.O = (Product\ ?pro0)$; $R.P = -$; $R.E = shaped(?pro0)$. This request is asking for a service that is able to produce some shaped product by using the input material. It accepts a parameter $m0$ of type *Material* as input and outputs a parameter $?pro0$ of type *Product*. wf consists of four semantic services (service definitions are given in Section 5.2.6). Their IO parameters and PE are illustrated with black, respectively red arrows. The parameter bindings are shown with blue dashed arrows. The implication relations are shown with blue solid arrows. Services used in the workflow are defined based on an ontology O_p. The predicates in service precondition and effect are the FOL translations of the concepts and properties in O_p.

To produce shaped product $?pro0$ based on the given material $?m0$, wf first uses a filtering service ($S6$) that chooses the heatable material, which quality is not bad. The binding $b(?m0,\ ?m4)$ indicates that the given material in $R.I$ can be used by the parameter $?m4$ of $S6$. Subsequently, the chosen material is then transmitted to a heating service ($S5$), which heats the material to at least 300 degrees. Here, the material $?m3$ out of $S6$ is bound with the input parameter $?m1$. Further, the heated material will be shaped by a shaping service ($S9$). The precondition of $S9$ specifies that the temperature of its input material $?m$ should be larger than 60 degrees and the material quality is not bad. $S9$ requires in addition some softener as an additive and the speed of adding softener is 5 ($speedEq(?so,\ 5)$). Finally, $S1$ is used for this, which does not require and input but outputs softener with the requested speed, and this softener is used by $S9$ as input based on the binding $b(?so1,\ ?so)$.

wf is correct because of the following reason:

1. each input parameter of any service in wf has been bound with an output parameter of another service in wf. The input parameter $?m4$ of $S6$ is bound with $?m0$ in $R.I$;

2. wf plugs into R, as the type $Material$ of $?m0$ is equal to the type of $?m4$, and the type $Product$ of $?pro$ in $S9$ is equal to the type of $?pro0$;

3. there is no contradicting literals in $(\bigcup_{S \text{ in } wf} S.E) \cup R.P$, given the bindings;

4. all preconditions of services in wf are satisfied with the bindings:
   $S9.P = tempLargerThan(?m, 60) \wedge qualityNotBad\ (?m)\ \wedge speedEquals\ (?so,\ 5)$:
   - $qualityNotBad\ (?m)$ can be implied by the truth of $S6.E$ through two bindings $b(?m3, ?m1)$ and $b(?m1, ?m)$. A part of $S6.E$ (colored in red, cf. Figure 5.1) claims that the output of the filtering service ($S6$) is some material which quality is not bad. Together with the bindings $b(?m3, ?m1)$ and $b(?m1, ?m)$, it indicates that $qualityNotBad(?m) \Rightarrow qualityNotBad\ (?m3)$ holds.
   - $tempLargerThan(?m, 60)$ can be implied by the truth of $S5.E = tempLar\ gerThan\ (?m1),$ 300) and the binding $b(?m, ?m1)$;
   - $speedEquals\ (?so,\ 5)$ can be implied by the truth of $S1.E = speedLargerThan(?so1,\ 1) \wedge speedSmallerThan(\ ?so1,\ 7)$ and the binding $b(?so1,\ ?so)$;

   $S5.P = heatable(?m1)$ can be implied by the truth of $heatable(?m3)$ in $S6.E$ and the binding $b(?m3, ?m1)$.

5. $R.E = shaped(?pro?)$ is implied by $S9.E = shaped(?pro)$ with the binding $b(?pro, ?pro0)$.

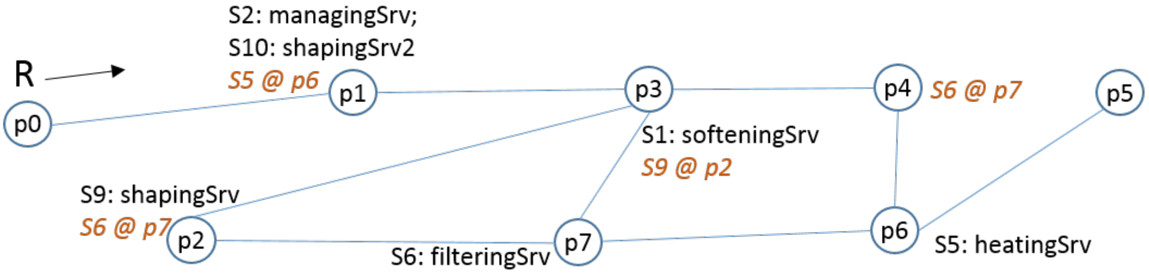


Fig. 5.2 An unstructured P2P network with items (services).

Definition 5: *Centralized stateless semantic service composition problem.*
The centralized stateless semantic service composition problem is defined by a tuple: $\langle \mathscr{S}, R, wf \rangle$. Given a request R for semantic services and a set $\mathscr{S}$ of services, the target is to build a correct workflow (cf. Definition 4) wf that answers to R.

In a distributed ad hoc environment like unstructured P2P network, the services in Definition 5. are distributively provided by the participant peers. Particularly, each peer has no global view to either the network topology or the services on other peers. To model this, we assume that all peers in unstructured P2P network share a vocabulary V consisting of a set of primitive terms. On top of this, each peer p defines its own knowledge base KB_p. It includes a peer local ontology O_p in OWL-DL and a set A_p of predicates. Each predicate is the FOL interpretation of a concept or object property defined in O_p. All peers in the network can create, expose and maintain semantic services in OWL-S. Denote $\mathscr{S}_p$ the set of known services by peer p, which contain the services provided by itself as well as the references to the ones provided by the other peers but known by p via its peer observation process (cf. Section 5.2.3).

For example, in the unstructured P2P network shown in Figure 5.2, peer $p1$ knows three semantic services: $\mathscr{S}_{p1} = \{\ S2, S10, S5\ \}$. $S2$ and $S10$ (in black normal) are provided by $p1$ itself; while $S5@p6$ (in brown italic) is the reference to service $S5$ that is provided by $p6$. $p1$ knows about $S5$ by its peer observation process. The knowledge about a service $S \in \mathscr{S}_p$ refers to its entire semantic description. We define distributed stateless semantic service composition problem as follows:

Definition 6: *Distributed stateless semantic service composition problem.*
The stateless distributed semantic service composition problem is defined by a tuple: $\langle \mathscr{N}, \mathscr{S}, R, wf \rangle$. Given a request R (cf. Definition 2) for semantic services, the target is to build a correct workflow (cf. Definition 4) wf that answer to R. wf is collaboratively composed by the peers in a network $\mathscr{N}$ based on their services $\mathscr{S}$. ■

In principle, such distributed environment $\mathscr{N}$ in Definition 6 can be organized by a client-server model, specific networks with fixed topologies, or P2P networks. In this chapter, we focus on the semantic semantic service composition in unstructured P2P networks. As for the centralized vari-

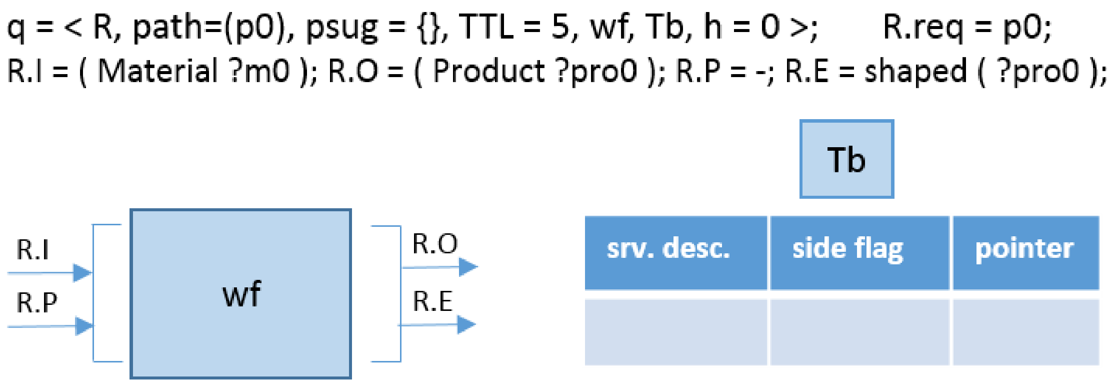


Fig. 5.3 Query q at initialization stage.

ants or the ones with other types of P2P networks, we concluded them in Sections 2.1.2 and 2.3.3, respectively.

With our proposed composition method SPSC, a user of a peer is allowed to issue the request for her desired service to an unstructured P2P network. This request is delegated to a walker-based semantic query. Besides the request R, the query also contains its time-to-live (TTL) value, an empty workflow that is supposed to be accomplished by peers in the network, as well as some auxiliary fields used by the composition and query routing processes. The semantic query is defined as follows:

Definition 7: *Semantic query q.*
A *Semantic query* (abbr. *query*) q is defined by a tuple $q = \langle R,\ req,\ path,\ psug,\ TTL,\ wf,\ Tb,\ h \rangle$, where R is the request for semantic services; $path$ is a sequences of peer identifiers that q has traversed; $psug$ is the path suggestion for this query; TTL is the time-to-live (TTL) value of this query; wf is the workflow (cf. Definition 4) answering to this request; Tb is the memo table, which is used by memorization strategy (cf. Section 5.2.3) to record candidate services for composing wf; h ($h \in \mathbb{R}$, $h \in [0,1]$) is the current heuristic value of wf. ■

We name the services in $L(wf)$ and $R(wf)$ as *useful services* with respect to R; the services in Tb as *candidate services*. For example, a query q issued by $p0$ (cf. Figure 5.2) is formulated as: $q = \langle R, path = (p0), psug = (), TTL = 5,\ wf,\ Tb,\ h \rangle$, where $R.req = p0$; $R.I = (Material\ \ ?m0)$; $R.O = (Product\ ?pro0)$; $R.P = -$; $R.E = shaped(?pro0)$. Its initial status is illustrated in Figure 5.3. At this moment, no useful service is found. Except the requested IOPE signature in R, its workflow wf contains no service internally. The memo table is empty.

5.2 Distributed Semantic Service Composition

In this section, we formally present our SPSC approach for the semantic service composition in unstructured P2P networks. Before its technical details, we show an overview of SPSC, in which we sketch the processes of its main components: peer local observation, heuristics-based composition and query routing. The details of them are presented in Sections 5.2.2, 5.2.3 and 5.2.4, respectively. We discuss the robustness of SPSC in Section 5.2.5 and show an illustrating example in Section 5.2.6.

5.2.1 Overview

Peer local observation: With SPSC, a peer p is allowed to observe the entire content of a received query q, if q is backtracking. Once knowing about (not providing) S, p is called a *service signature maintainer of* S. In addition, p is allowed to override the QoS value of some known service S, if a new version of S that has a higher QoS value is observed. Furthermore, p updates its local view on network topology based on the query path and path suggestion. This local view is used in the query routing process (cf. Section 5.2.4).

Composition with heuristics: A query q in SPSC is an epidemic walker with TTL limitation. In a bidirectional chaining fashion, the building of workflow wf is done collaboratively by peers on the query path. On receiving q, a peer p checks whether its known services can contribute to wf, if q is in its forward journey. p makes q backtrack, once the wf is correct or TTL is exhausted. For each service $S \in \mathscr{S}_p$, p checks whether S can be chained at the left, right or both sides of the current wf with those candidate services in the memo table (cf. Definition 5). The quality of chaining is measured by the chaining score (cf. Section 5.2.2). If it is larger than a chaining threshold (cf. Section 5.2.3), p makes a hypothesis that S would be chained, which leads to a new workflow wf'. In order to prevent the workflow from blind augmentation, a heuristic value h' is subsequently computed based on wf'. If h' is larger than the current heuristic value in q, S is treated as a useful service with respect to wf and subsequently chained to $L(wf)$ $(R(wf))$. Further, the old heuristic value in q will be replaced by h'. In cases that S (a) can be chained but without an increment of heuristic value, or (b) cannot be chained into a workflow at all, p applies a memorization strategy for the purpose of keeping the potentially useful candidate service(s) in the query. In case (a), p adds S into the memo table of q, while in case (b), p adds S into the memo table with a rate r_m. The latter is computed based on the estimation about the statistical (positive or negative) influence about using S in the current wf (cf. Equation 5.6). Intuitively, this strategy enables peers to collaboratively build a partial plan graph. Such a graph is kept in the query, transmitted and enriched by the peers in the query path. The memorized candidates will also be considered in the further composition at the next peers.

Query routing: SPSC enables each peer to suggest a path for routing a query q, after its local composition process. The suggested path contains multiple key peers such that their total inverse importance score per traffic cost is minimized, under the TTL limitation. The key peers are the ones which services are estimated to be helpful for completing the current workflow. Once a path suggestion has been formed, it is piggybacked by q and propagated to other peers on the planned path. When a received query has a non-empty path suggestion, p recomputes a new path suggestion as long as

p has made a contribution to the building of the workflow, i.e. the memo table is updated by p. p routes q according to the old path suggestion of q, otherwise. If no path suggestion can be made and the old path suggestion is empty, p routes q to a random neighbor peer.

5.2.2 Functional IOPE-Level Chaining of Services

The workflow of query is built collaboratively by peers in a bidirectional chaining fashion. When receiving a query, each peer checks whether its known services can contribute to the building of workflow. That is to check whether each service can be chained to the left, right or both sides of the current workflow. As a fundamental process, to chain two services S' after S is to determine (i) to what extent the variables of $S.O$ can be accepted and used by $S'.I$, and (ii) to what extent the truth of $S.E$ can make $S'.P$ to be satisfied. This process means that peer computes the chaining score at IO, respectively PE levels. On top of this, peer computes the overall score of chaining S' after S (cf. Equation 5.1).

$$\begin{aligned} ch(S,S') &= \tfrac{1}{2}(ch_{IO}(S,S') + ch_{PE}(S,S')) \cdot df(ch_{IO}(S,S'), ch_{PE}(S,S')); \\ df(z,z') &= min\{\tfrac{z}{z'}, \tfrac{z'}{z}\}, \quad z,z' \in (0,1]. \end{aligned} \tag{5.1}$$

where $ch_{IO}(S,S')$ and $ch_{PE}(S,S')$ are IO, respectively PE chaining scores. $ch(S,S') \in [0,1], ch(S,S') \in \mathbb{R}$.$ch(S,S') = 0$, if $ch_{IO}(S,S') = 0$ or $ch_{PE}(S,S') = 0$. In Equation 5.1, the average of the IO and PE chaining scores is computed. The reason why they can be added is that the both of them are rates (cf. Equations 5.3 and 5.4) and describes the two aspects of the same matching. This is inspired by the computation of F1 score. The averaged value is further adjusted by $df(\cdot,\cdot)$ ($df(\cdot,\cdot) \in \mathbb{R}$, $df(\cdot,\cdot) > 0$) in order to check the difference between ch_{IO} and ch_{PE}. This is helpful to filter out some low quality chaining, which IO score is large but PE score is small (or in the other way around).

Besides averaging the IO and PE scores, the aggregation can also be done via Support Vector Machine (SVM) based machine learning [200] or a convex combination with trained coefficients [221]. Nonetheless, off-line learning based aggregation can introduce inertia to a system, which in addition is subject to the quality of training set. When needed, the coefficients can be chosen by user for specific concerns.

In the state-based semantic service composition method OWLS-XPlan [198], an OWL-S service is correlated with a PDDL 2.1 [104] action. Relying on the FOL interpretation of OWL-DL, service input and output parameters are converted to additional predicates in the precondition, respectively effect of action. Such combinations ensure the feasibility of applying FF planner [145]. The chaining between services is reduced to the mapping over those grounded facts that are produced by services (actions). The same principle has also been hired by the distributed planning approach in SCComp [109] [108]. In contrast, SPSC works with service variables under OWA and consider the chaining at IO and PE levels separately.

In SPSC, the IO level chaining is processed first. It determines which parameter in $S.O$ can be used by which parameter in $S.I$. This is done by checking the parameter type subsumption relations between $S.O$ and $S'.I$. Besides returning a chaining score at IO level, it yields a set of parameter bindings. Each is meant with a correlation between an output parameter $?x \in S.O$ and an input parameter $?y \in S.I$. Such a binding indicates a predicate $?x =?y$ that is helpful to alleviate the overhead for computing PE chaining score. In detail, before computing the PE chaining score, the process applies a substitution $\{?z \mapsto ?x,\ ?z \mapsto ?y\}$ to $S.E$ and $S'.P$, which replace all the occurrences of $?x$ and $?y$ with a new variable $?z$. If considering the chaining at PE level without substitution, the process has to check the truth of $S.E \implies S'.P$ with all possible parameter bindings. The computation complexity will be large.

IO Chaining Score: For the type concept Y of each variable $?y \in S'.I$, this process tries to find its best subsumee $?x$ in $S.O$. Denote the type of $?x$ as X_Y. If such $?x$ is found, $x?$ and $?y$ are bound.

$$\begin{array}{ll} X_Y & = argmax_{X \in S.O} s_c(X,Y); \\ \text{subject to:} & s_c(X_Y,Y) \geq \beta. \end{array} \tag{5.2}$$

where β ($\beta \in \mathbb{R}$, $\beta \in (0,1]$) is the binding threshold; $s_c(X,Y) \in [0,1]$ is the concept subsumption similarity function, such as [223]. Denote $M \subseteq S'.I$ the set of bound variable types in $S'.I$. Based on the variable bindings, the IO chaining score can be computed:

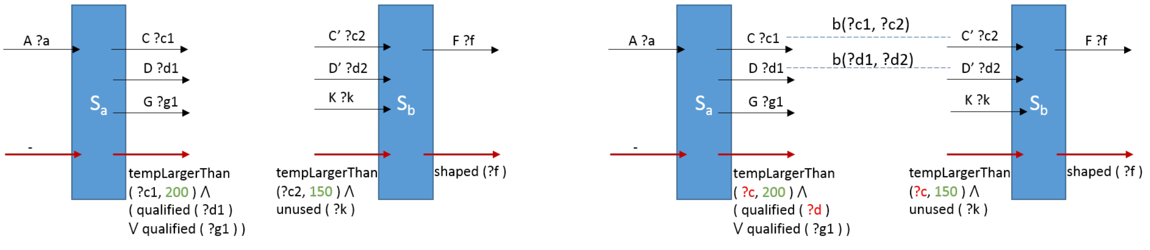
$$ch_{IO}(S,S') = \frac{|M|}{|S'.I|}. \tag{5.3}$$

where $|S'.I|$ is the number of input parameters of service S'. As an alternative, the matching between parameters can be done via bipartite matching, which produces a bijection between the parameters in $S.O$ and $S'.I$. This method returns more precise result but with larger computational complexity. In contrast, SPSC finds the matchings in a greedy manner. Injection can appear in a result, where one output parameter in $S.O$ can match with more than one parameter in $S'.I$. Actually, this is allowed in a service orchestration. Furthermore, the way of matching above can also yield bijection, which simply ignores any matched parameter in $S'.I$ in the next rounds.

PE Chaining Score: The PE chaining score $ch_{PE}(S,S')$ measures that, to what extent, the precondition of S' can be satisfied based on the truth of the $S.E$, given the bindings determined above. For this, the first step is to apply the substitutions out of the bindings to $S.E$ and $S'.P$ formulas. Denote $S'.P'$ a part of $S'.P$, which does not contain unbound parameters. Subsequently, the satisfaction of the implication $S.E \to cl$, for $\forall cl \in S'.P'$, is determined. It is done via θ-subsumption decision [240]. Denote SP the set of satisfied clauses in $S'.P$; $|S'.P|$ the number of clauses in the precondition formula of S':

$$ch_{PE}(S,S') = \frac{|SP|}{|S'.P|}. \tag{5.4}$$

A special case is that there exists contradiction in $(\bigcup_{S \text{ in } wf} S.E) \cup R.P$, given the parameter bindings. When computing the PE chaining score, this will be additionally checked. Once detected, the PE



(a) (b)

Fig. 5.4 Example of chaining score computation.

chaining score is set with 0.

Example: There are two semantic services S_a and S_b. We check the score of chaining S_b after S_a: $ch(S_a, S_b)$ (cf. Figure 5.4a).

$\mathbf{S_a} = \{\texttt{http://.../Sa}, I_a, O_a, P_a, E_a, p1, 0.75\}$.
$I_a = (A\ ?a)$;
$O_a = (C\ ?c1,\ D\ ?d1,\ G\ ?g1)$;
$P_a = -$;
$E_a = tempLargerThan(?c1, 200) \wedge ((qualified(?d1) \vee qualified(?g1))$.

$\mathbf{S_b} = \{\texttt{http://.../Sb}, I_b, O_b, P_b, E_b, p2, 0.8\}$.
$I_b = (C'\ ?c2,\ D'\ ?d2,\ K\ ?k)$;
$O_b = (F\ ?f)$;
$P_b = tempLargerThan(?c2, 150) \wedge unused(?k)$;
$E_b = shaped(?f)$.

The first step is to compute the IO chaining score $ch_{IO}(S_a, S_b)$. This process tries to find the best subsumee from $S_a.O$ for each parameter type in $S_b.I$. We assume that $C \sqsubseteq C'$ and $D \sqsubseteq D'$ hold. It follows that C matches with C' and D matches with D'. On top of this, two bindings $b(?c1,\ ?c2)$ and $b(?d1,\ ?d2)$ can be derived. They indicate two substitutions: $\varphi_c = \{?c1 \mapsto ?c, ?c2 \mapsto ?c\}$ and $\varphi_d = \{?d1 \mapsto ?d, ?d2 \mapsto ?d\}$, where $?c$ and $?d$ are two new variable names. Therefore, $M = \{C', D'\}$. Subsequently, the IO chaining score is computed:
$ch_{IO}(S_a, S_b) = \frac{|M|}{|S_b.I|} = \frac{|\{C',\ D'\}|}{|\{C', D', K\}|} = \frac{2}{3} = 0.667$.

Based on the bindings, PE chaining score can be computed. To do this, the process applies the substitutions φ_c and φ_d to $S_a.E$ and $S_b.P$ (cf. Figure 5.4b). Subsequently, this process ignores the non-decidable clause $unused(?k)$ in $S_b.P$, as it contains unbound variable $?k$. It follows that $S_b.P' = tempLargerThan\ (?c, 150)$ and this can be satisfied by the truth of $S_a.E$. Therefore: $ch_{IO}(S_a, S_b) = \frac{|S_a.P'|}{|S_b.P|} = \frac{1}{2} = 0.5$.

Finally, the chaining score $ch(S_a,S_b)$ can be computed by Equation 5.1:
$ch(S_a,S_b) = \frac{1}{2}(ch_{IO}(S_a,S_b) + ch_{PE}(S_a,S_b)) \cdot df(ch_{IO}(S_a,S_b), ch_{PE}(S_a,S_b)) =$
$\frac{1}{2}(0.667 + 0.5) \cdot min\{\frac{0.667}{0.5}, \frac{0.5}{0.667}\} = 0.583 \cdot 0.75 = 0.44.$

5.2.3 Heuristics-based Semantic Composition of Services

Composition with Heuristics: In general, a service composition process iteratively checks the contribution of chaining services into a workflow in order to achieve the output and effect of the overall requested service. In AI planning domain, a component service can be treated as an action and the composition process is reduced as the search of shortest paths from the current state nodes to the goal state nodes, in graph-based setting. An important principle, called heuristics for planning, is commonly applied. According to the Heuristics for Classical Planning tutorial[1] in the 19th ICAPS (2009) conference, heuristics measure the distance from a node to goal node, in order to protect the partial solution from arbitrary augmentation. When considering each augmentation of the current paths, the planning process applies heuristics that judges to what extent the current chaining can help to reach the goal state nodes. For the service composition in unstructured P2P-based settings, a universal graphplan can not be prepared in advance, since peers do not have global knowledge about services. In addition, stateless IOPE-level service composition does not convert IO signatures to extra predicates in PE, but rather checks the compatibility of IO and PE, separately. Similar to state-based AI planning, service composition also needs heuristics to alleviate those helpless chaining. In this perspective, how to apply the heuristics principle on-the-fly for IOPE-level service composition is challenging. Unfortunately, the previous efforts that attack the same problem were not aware of the importance of heuristics. In contrast, we propose, in this section, a novel heuristics conforming to the heuristics-based planning principle but devoted to the stateless IOPE-level service composition.

In SPSC, each query q is a TTL-restricted epidemic walker starting from the requester peer. In a bidirectional chaining fashion, its workflow is collaboratively built by peers on the query path. On receiving q, each peer p executes *queryProcess*$(p,\ q)$ (cf.Alg.6). The building of workflow happens at p, when q is being forwarded. For each known service $S \in \mathscr{S}_p$, p checks whether or not the possible chaining of S is helpful. That is to estimate, to what extent, the possible chaining of S into the workflow will contribute to achieve the currently requested output and effect.

In detail, when p considers to chain S to $L(wf)$ and $R(wf)$ (line 4), it computes the chaining scores $ch(L(wf),S)$ and $ch(S,R(wf))$ (cf. Section 5.2.2), where the output of $L(wf)$ (input of $R(wf)$) contains the all unbound output (input) parameters of the services in $L(wf)$ ($R(wf)$). For this purpose, p checks the following:

1. If either $ch(L(wf),S)$ or $ch(S,R(wf))$ is larger than a chaining threshold θ ($\theta \in \mathbb{R}, \theta \in (0,1]$), S will be regarded as a candidate service[2]. Subsequently, p makes a hypothesis that S is chained to either the left or right side of wf.

---

[1] `http://icaps09.uom.gr/tutorials/tut1.pdf`

[2] The chaining threshold can vary in different systems with various concerns, as the choice of it is closely relevant

2. If the criteria holds in both directions, p checks whether wf is correct.

In case that (1) is true and (2) is false, the peer makes a hypothesis that S has been chained into wf, which results in a new workflow wf'. Subsequently, p applies the following heuristics: Computing a heuristic value $h' \in [0,\ 1]$ (line 5), which is the chaining score between $L(wf')$ and $R(wf')$.

$$h' = ch(L(wf'),\ R(wf')) \tag{5.5}$$

then comparing h' with the current heuristic value in q (lines 6–7). If $h' > q.h$, S is treated as a useful service for the building of the workflow, and the heuristic value of q is replaced with h'. In case that more than one useful service has been found locally, the service that leads to the highest heuristic value h' counts. If multiple services all give the highest heuristic value, the one with maximal service quality is used. Random selection is applied further, if their service quality values are equivalent.

Our proposed heuristic for stateless IOPE service composition conforms to the general principle of heuristics in AI planning domain. In addition, it is consistent, since the process exhaustively detects all possible chanings and chooses to chain the service that leads to the highest non-zero heuristic value increment. This property is proved in Corollary 5.2.

Memorization Strategy: A query q can encounter some service S that (a) can be chained to one side but without increasing the heuristic value or (b) can not be temporarily chained to none of the both sides at all. Service in this kind would work as a key predecessor/successor of other services. The latter explicitly contribute to the input/precondition of the current $R(wf)$. Unfortunately, service like S can not be considered at next peers as they might not know about it. The deeper reason is that the order of encountering service can not be fixed in advance. This is a crucial difference of our solution with those centralized ones, such as OWLS-XPlan [198] and FF planner [145], which have all needed information at hand.

To cope with this, each peer applies memorization strategy (detailed afterwards) to keep in query the descriptions of those potentially useful services out of the cases (a) and (b). These services will be considered during the local composition processes at next peers. For this purpose, a memo table (Tb in Definition 4.) is used. Each row in Tb corresponds to a candidate service. It contains 3 fields:

1. the description of a candidate service S;
2. a side flag in $\{L, R, null\}$, which indicates whether S can be chained at the left, right or none of the both sides of the workflow;
3. a pointer that references to another service S' in this table, if S can be chained to S' as a direct predecessor or successor. The pointer is $null$, if the service of this row can not be chained to the both sides, or its direct predecessor/successor has not been determined yet.

---

to the services in specific system. In an unstructured P2P network, dynamics on network topology and service data can happen at anytime. Instead of employing a global value for θ, a requester peer can set this value for its queries only; while the other peers on the query path apply this value during their local composition processes.

Algorithm 6 *queryProcess*$(p,\ q)$

Input:

p: the peer that executes this process;

q: the query, which workflow is to be composed by p.

Output:

Void.

```
1:  if $q$ is being forwarded then
2:    $q.TTL \leftarrow q.TTL - 1$;
3:    for each $S \in \mathscr{S}_p$ do
4:      perform the bidirectional chaining process for $S$ w.r.t. the workflow of $q$;
5:      compute the new heuristic value $h'$;
6:      if $h' > q.h$
7:        $q.h \leftarrow h'$;
8:        if the workflow of $q$ is correct, break;   otherwise update $q.Tb$;
9:        goto line 3;
10:     else if $S$ can be chained to one side of the workflow and $h' \leq q.h$
11:       add $S$ into memo table; goto line 3;
12:     otherwise, add $S$ into memo table with the usefulness rate $r_m(S)$;
13:   end for
14:   if the composition of $q$ is correct or $q.TTL = 0$, make $q$ backtrack;
15:   otherwise make the path suggestion for $q$ and route $q$;
16:     for each candidate service $S$ in the workflow of $q$
17:       $\mathscr{S}_{pred} \leftarrow$ alternative predecessor services of $S$ w.r.t. a service $S^*$ in $wf$;
18:       for each $S' \in \mathscr{S}_{pred}$
19:         create a sub-query $q'$ of $q$;
20:         if the workflow of $q'$ is new, queryProcess$(p,\ q')$;
21:       endfor
22:       $\mathscr{S}_{succ} \leftarrow$ alternative successors services of $S$ w.r.t. a service $S^*$ in $wf$;
23:       for each subset $\mathscr{S}'_{succ} \in 2^{(\mathscr{S}_{succ} \cup S)} \setminus \{S\}$
24:         create a sub-query $q'$, which workflow uses all services in $\mathscr{S}'_{succ}$;
25:         if the workflow of $q'$ is new, queryProcess$(p,\ q')$;
26:       endfor
27:     endfor
28: end if
29: if $q$ is backtracking, update the local observation based on $q$;
30:   if $p$ is not the requester peer of $q$, force $q$ to backtrack;
31: endif
```

Memorization strategy is as follows:

- If the case (a) is true, p adds S to memo table; sets the side flag; and initializes the pointers to its direct predecessor/successor. (lines 10–11)
- If the case (b) happens, p adds S to memo table with the usefulness rate $r_m(S)$ of S ($r_m(S) \in (0,1]$, cf. Equation 5.6). (line 12)

If a service $S^* \notin Tb$ can be chained to the both sides and the heuristic value increases, p checks the correctness of the workflow. If it is correct, p makes q backtrack; otherwise, p checks whether there exist unbound input parameters or unsatisfied preconditions in the hypothetically chained services. If it is false for both criteria, p updates memo table by removing the involved services from it.

The rate $r_m(S)$ measures the overall usefulness of S with respect to the building of a workflow. This value is estimated by each peer p locally, according to its observations on S in the workflows of a set Q of queries in the past:

$$r_m(S) = \begin{cases} \omega & \text{, } \mathbf{if}\ a'(S) = 0; \\ \frac{a'(S)}{a(S)+1} \cdot (1 - \frac{b'(S)}{b(S)+1}) & \text{, } \mathbf{otherwise}. \end{cases} \tag{5.6}$$

where $a(S)$ ($a'(S)$) is the number of previous (correct) workflows that use S; while $b(S)$ ($b'(S)$) is the number of previous (correct) workflows that do not use S. The part $\frac{a'(S)}{a(S)+1} \in [0,1)$ measures the statistical positive influence of treating S as a candidate; while the part $\frac{b'(S)}{b(S)+1} \in [0,1)$ is the statistical negative influence of ignoring S as a candidate. ω ($\omega \in [0,1], \omega \in \mathbb{R}$) is the default memorization rate for unchain-able service.

There are two ways for choosing Q. One option is to collect all the queries observed in the past. The other is to take mere the queries that are similar with q into account. By applying the iSeM [200] service matchmaker, the similarity can be measured by a matchmaking score $s_{plug-in}(q.R, q'.R)$ ($q' \in Q$) under the hypothesis that $q.R$ plugs into $q'.R$. Each of the both has its own pros and cons. The first approach can be applied very fast, if particularly the counting is done incrementally. However, the quality of estimating r_m might be low due to those irrelevant queries. The second approach provides better quality of r_m, but needs more computation. Such burden can be large if there are a lot of observed queries. A compromise is to consider at most n ($n \in \mathbb{N}, n > 0$) relevant queries that have been observed in a certain time window.

It is possible that the local composition will produce cyclic dependency (chaining) among services. For instance, a service S_a accepts $?c\ C$ and outputs $?d\ D$, while another service S_b accepts $?d\ D$ and outputs $?c\ C$. By the local composition process, the infinity cyclic chaining $\cdots S_a \rightarrow S_b \rightarrow S_a \rightarrow S_b \cdots$ can appear. To handle this, peer is enabled to cache those intermediate workflows it has investigated. Once a hypothetical chaining is made, it checks whether or not the current workflow has the same output and effect (input and precondition) with any of the cached ones. If it is true, the composition process discards this chaining and removes the corresponding rows in memo table of query. The cache will be released, as long as the local composition process finishes.

Query Branching with Alternative Service: Let S_1 and S_2 be two different semantic services. If both of them can be chained to the same side of another service S with the bindings that contains the same (non-empty) subset of parameters in $S.I$ ($S.O$), we name S_1 and S_2 are *alternative predecessor (successor) services* to each other with respect to S. For example, the *shapingSrv1* and *shapingSrv* are alternative predecessor services to each other with respect to the *coloringSrv* (cf. Figure 5.5),

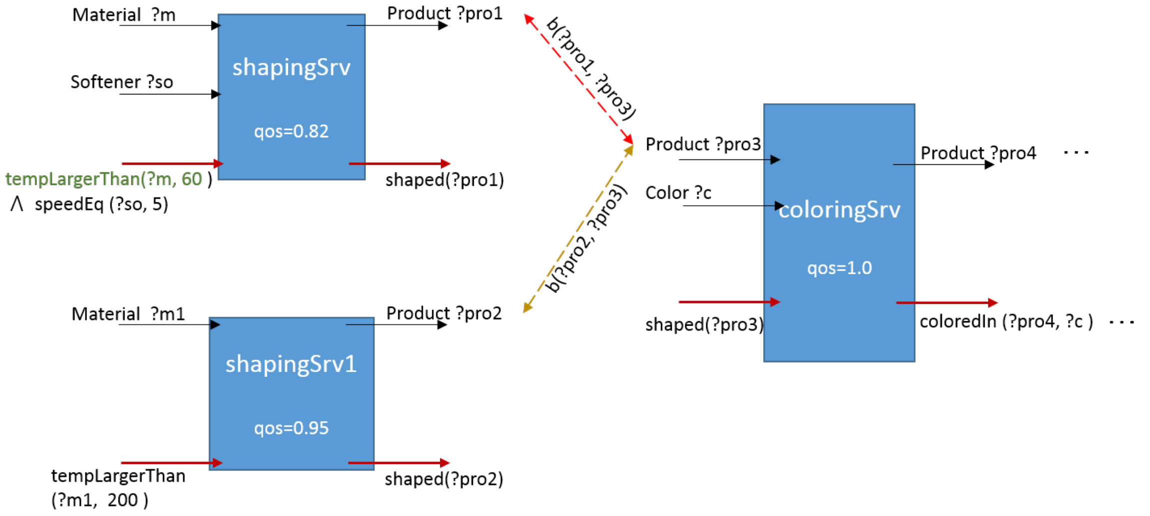
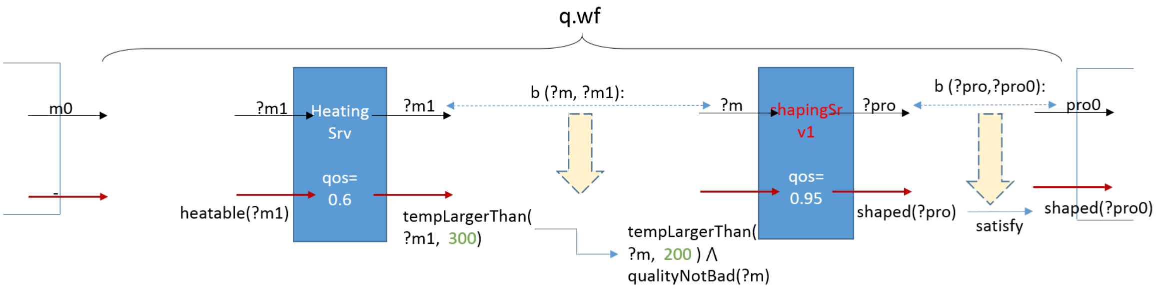


Fig. 5.5 *shapingSrv* and *shapingSrv1* are alternative predecessor services to each other with respect to the *coloringSrv*.

since both of them can be chained to the same side of *coloringSrv* with the bindings that contains the same subset $\{Product\ ?pro3\}$ of variables in *coloringSrv.I*. This indicates that either *shapingSrv* or *shapingSrv1* can work as the predecessor of *coloringSrv*.

If a peer p is not able to find a correct solution locally, it determines the possible sub-queries based on its known services (lines 16–27 in Alg.6). In detail, for each hypothetically chained candidate service S in memo table, the composition process of p checks whether there exists its alternative service S'. If it is the case, the process issues a sub-query q', in order to investigate the possible workflow with S'. p does the same for each alternative service of S. To initialize a sub-query, p firstly initializes q' as a copy of q; subsequently, it replaces S with S' in the memo table of q' and unchains the set of candidate services that depend on S; further, p computes the heuristic value of the workflow in q' by means of Equation 5.1; finally, p executes $queryProcess(q,\ q')$ (cf.Alg.6). In the query processing of q', to avoid the duplication, S will not be considered as the alternative service of S'.

For example, let us assume that peer p receives a query q, which can not be correctly resolved by p (cf. Figure 5.6). To determine the possible sub-queries of q, p locally finds a service *shapingSrv*, which can work as an alternative service of *shapingSrv1* with respect to the right side of $q.wf$. In this situation, besides processing and routing q, p issues a sub-query q' to investigate the possible solution that contains *shapingSrv*, but not *shapingSrv1*. For this, p firstly initializes a sub-query q' that is a copy of q. Subsequently, p unchains the *heatingSrv* from *shapingSrv1* in $q'.wf$, as *heatingSrv* is hypothetically used in $q'.wf$ depending on its chaining with *shapingSrv1*. Further, p removes *shapingSrv1* from $q'.wf$ and takes use of the *shapingSrv* instead. *heatingSrv* is treated as an unchained service in $q'.Tb$ for the composition afterwards. The initial TTL of q' is the current TTL value of q. The sub-query q' after query branching is shown in Figure 5.7.

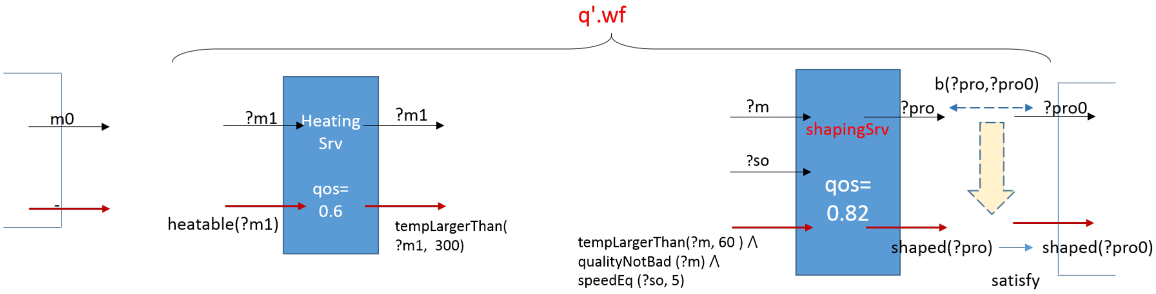


q = < R, path=(p0), psug = {}, TTL = 3, wf, Tb, h = 0 >;
R.req = p0;
R.I = (Material ?m0); R.O = (Product ?pro0);
R.P = -; R.E = shaped (?pro0);

q. Tb

srv. desc.	side flag	pointer
shapingSrv1	R	heatingSrv
heatingSrv	R	null

Fig. 5.6 Current query q before branching.



q' = < R, path=(p0), psug = {}, TTL = 3, wf, Tb, h = 0 >;
R.req = p0;
R.I = (Material ?m0); R.O = (Product ?pro0);
R.P = -; R.E = shaped (?pro0);

q'. Tb

srv. desc.	side flag	pointer
shapingSrv	R	null
heatingSrv	null	null

Fig. 5.7 Sub-query q' of q after query branching.

In case that there exists a non-empty set $\mathscr{S}_{succ}$ of alternative successor services of S with respect to a service S^* in wf, these services in $\mathscr{S}_{succ}$ can be chained to $S^*.O$ at the same time by sharing the data of bound variables of $S^*.O$. It means that, after the execution of S^*, all services in $\mathscr{S}_{succ}$ have chance to be executed in one workflow. In this situation, p will issue one (sub-)query for each non-empty subset of $\mathscr{S}_{succ}$. For example, if there is only one alternative successor service S' of S, p issues the sub-queries for S' and $\{S, S'\}$.

5.2.4 Query Routing

The completion of local composition triggers the query routing process. A query q is forwarded, when its workflow wf is not *correct* and the $q.TTL$ has not been exhausted. Instead of mere a direct neighbor, the query routing process of p computes a path containing a sequence of *key peers* as a suggestion for routing q (line 15 in Alg.6). A key peer is either (1) the provider of a memorized candidate service S in wf, or (2) the signature maintainer peer p_m of S. From p_m, a peer $p^* \in q.path$

got to know S and p^* is the first one (compared with the others in $q.path$) that uses S for the building of wf. Please note that the aim of routing in SPSC differs with the aims of P2P search strategies. It does not intend to find similar services, but the suitable predecessor/successor services that are helpful for building the current workflow.

p computes a path PS containing multiple key peers such that the total inverse importance score per traffic cost of the path is minimized, under the TTL limitation.

$$\begin{array}{ll} \text{minimize:} & \sum_{p' \in P_{key}} w(p'', q, p'), \quad \text{where } p'' \in \{p\} \cup P_{key}; \\ \text{subject to:} & \sum_{p'_i \in PS,\ p'_{i+1} \in PS} len(sPa(p'_i, p'_{i+1})) \leq q.TTL. \end{array} \tag{5.7}$$

where P_{key} is the set of key peers; $w(p'', q, p')$ is the inverse importance score per traffic cost of a peer $p' \in P_{key}$ about S; $len(sPa(p'_i, p'_{i+1}))$ is the length of the shortest path $sPa(p'_i, p'_{i+1})$ between two key peers in PS. $w(p'', q, p')$ ($w(p'', q, p') \in \mathbb{R}^+$) is computed via Equation 5.8, where $sr(p', S)$ is the importance score of S with respect to the building of wf. $sr(p', S)$ is the product of the historical usefulness rate value $r_m(S)$ and the stability factor of the current workflow. The latter is estimated based on the service quality and the dependency relations between S with its predecessors or successors.

$$\begin{aligned} w(p'', q, p') &= \frac{L(p'', p')}{sr(p', S)}; \\ sr(p', S) &= r_m(S) \cdot \frac{qos(S)}{|wf_H| + 1 - dep(S)}. \end{aligned} \tag{5.8}$$

In Equation 5.8, $|wf_H|$ is the total number of hypothetically chained candidate services in wf; the dependency factor $dep(S)$ is the number of the necessary predecessor/successor services of S in Tb.

The object function in Equation 5.7 implies a relaxed variant of Traveler Salesman Problem (TSP), in which the key peers are cities and the shortest path between key peers are the edges between cities. The inverse importance score per traffic cost between p'' and p' is the weight of that edge. This problem can be approximatively solved by adapting the peer expertise-based query routing heuristics in S2P2P [47]. Based on its local view to the network topology, p computes the shortest paths from itself to the key peer that yields the minimum weight, under the TTL limitation; then from the first key peer to the second, and so forth. Finally, p concatenates them and suggests the result for routing q. When p receives q with a non-empty old path suggestion, p recomputes it as long as p has made contribution to the building of workflow, i.e. the memo table of q has been updated by p; otherwise, p routes q according to its old path suggestion. p routes q to a random neighbor, if q's path suggestion is empty and p is not able to suggest a path for q.

5.2.5 Robustness

In this section, we discuss the robustness of SPSC approach. It contains two folds:

Network topology dynamics: Service provider peers can join in or leave from an unstructured P2P network at anytime anywhere. The arrival of a new peer does not break the query answering process; while the departure would lead to some incorrect references to services as well as unprecise path suggestion. With SPSC, peer reacts passively against the departure of a peer p that might provides a service S. It means that p will not send extra message to network for the notification. If a departure event is detected (due to messaging timeout) by another peer p' during query routing, p' will delete the reference of S from $\mathscr{S}_{p'}$; and also delete S plus its successors (predecessors) from the current workflow plus memo table. Subsequently, p' recomputes the path suggestion and attaches this event to the query in order to inform other peers in the path.

Semantic signature dynamics: Semantic-based unstructured P2P network is an open environment. New services can be added/deleted and the semantic description of existing services can be updated by users. In SPSC, the deletion of service is processed by a peer similarly with the departure of a service provider peer. If the addition (with concrete deployment) of a service S is confirmed by the user to her peer p and the latter is in the meanwhile forwarding a query q, p checks the possible contribution of S for answering q. If q is backtracking, p issues a sub-query q from itself in case that S is determined to be useful in the workflow; The update on service description can be treated as a sequence of deletion and addition of the same service.

5.2.6 Example

In this section, we show an example to illustrate the important processes of SPSC approach. In Figure 5.3, a query aiming at composing a production line service is issued by peer $p0$ to a simple unstructured P2P network (cf. Figure 5.2) with services (in black roman) and their references (in brown italic) via previous peer local observation. Let us assume that the chaining threshold $\theta = 0.3$, and the default memorization rate $\omega = 0.5$ (cf. Equation 5.6). The descriptions of services in the network are as follows:

$\mathbf{S1} = \{\texttt{http://.../softeningSrv}, I_1, O_1, P_1, E_1, p3, 0.75\}$.
$I_1 = ()$;
$O_1 = (\ Softener\ \ ?so1)$;
$P_1 = -$;
$E_1 = speedLargerThan(?so1, 1) \wedge speedSmallerThan(\ ?so1,\ 7)$.

$\mathbf{S6} = \{\texttt{http://.../filteringSrv}, I_6, O_6, P_6, E_6, p7, 0.91\}$.
$I_6 = (Material\ ?m)$;
$O_6 = (Material\ \ ?m)$;
$P_6 = -$;
$E_6 = qualityNotBad(?m)\ \wedge heatable(?m)$.

S2 $= \{$`http://.../managingSrv`$, I_2, O_2, P_2, E_2, p1, 0.8\}$.
$I_2 = (ProductionLine\ \ ?pl)$;
$O_2 = (Action\ \ ?act)$;
$P_2 = -$;
$E_2 = running(?pl)$.

S5 $= \{$`http://.../heatingSrv`$, I_5, O_5, P_5, E_5, p6, 0.6\}$.
$I_5 = (Material\ ?m1)$;
$O_5 = (Material\ ?m1)$;
$P_5 = heatable(?m1)$;
$E_5 = tempLargerThan(?m1), 300)$.

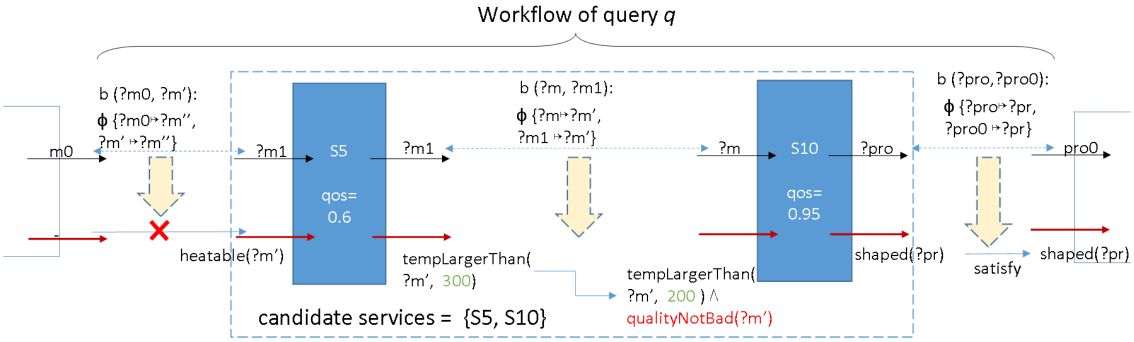
S9 $= \{$`http://.../shapingSrv`$, I_9, O_9, P_9, E_9, p2, 0.82\}$.
$I_9 = (Material\ ?m, Softener?so)$;
$O_9 = (Product\ ?pro)$;
$P_9 = tempLargerThan(?m, 60) \wedge qualityNotBad\ (?m) \wedge speedEquals\ (?so, 5)$;
$E_9 = shaped(?pro)$.

S10 defined in Section 5.1.

• At $p0$, the composition of q is initialized. Its workflow does not contain any service and the memo table is empty. As $p0$ can not contribute to the building of composition, it randomly chooses a neighbor peer $p1$ and send q to it.

• On receiving q, $p1$ considers to contribute to the composition of q, as q is being forwarded. For this, it performs the bidirectional chaining process for each of its known services:

For $S10$, $p1$ checks whether $S10$ can be chained to $L(wf)$ ($R(wf)$). For the left side, it computes:
$ch(L(wf), S10)$
$= \frac{1}{2}(ch_{IO}(L(wf), S10) + ch_{PE}(L(wf), S10)) \cdot df(ch_{IO}(L(wf), S10), ch_{PE}(L(wf), S10))$
$= \frac{1}{2}(1+0) \cdot 0$
$= 0$.
Similarly, for $R(wf)$:
$ch(S10, R(wf))$
$= \frac{1}{2}(ch_{IO}(S10, R(wf)) + ch_{PE}(S10, R(wf))) \cdot df(ch_{IO}(S10, R(wf)), ch_{PE}(S10, R(wf)))$
$= \frac{1}{2}(1+1) \cdot min\{\frac{1}{1}, \frac{1}{1}\}$
$= 1$.



(a)

srv. desc.	side flag	pointer
S10	R	null

(b)

srv. desc.	side flag	pointer
S10	R	S5
S5	R	null

(c)

srv. desc.	side flag	pointer
S10	R	S5
S5	R	null
S1	null	null

(d)

srv. desc.	side flag	pointer
S1	null	null

(e)

srv. desc.	side flag	pointer
S9	R	null
S5	null	null
S1	null	null

(f)

Fig. 5.8 Example of composition process at peer $p1$.

The computation of $ch_{IO}(S10, R(wf))$ leads to the binding $b(?pro, ?pro0) = \langle ?pro, ?pro0, \{?pro \mapsto ?pr, ?pro0 \mapsto ?pr\}\rangle$. It follows that the precondition of $R(wf)$ can be satisfied by the truth of $S10.E$. As $ch(S10, R(wf)) = 1 > \theta$, $p1$ makes a hypothesis that $S10$ is chained to $R(wf)$, which yields a new workflow wf_1. However, the heuristic value $h(wf_1) = ch(L(wf_1), R(wf_1))$ is still 0. Thus, $p1$ treats $S10$ as a candidate service and adds it into memo table (cf. Figure 5.8b).

For $S5$, $p1$ checks whether $S5$ is useful for building the hypothetical composition with the candidate service $S10$. $p1$ finds that $S5$ can not be chained to $R(wf_1)$, but not $L(wf_1)$, since its precondition $heatable(?m1)$ can not be satisfied (cf. Figure 5.8a):

$ch_{IO}(S5, R(wf_1))$

$= \frac{1}{2}(ch_{IO}(S10, R(wf)) + ch_{PE}(S10, R(wf))) \cdot df(ch_{IO}\ (S10, R(wf)),\ ch_{PE}(S10,\ R(wf)))$

$= \frac{1}{2}(1 + 0.5) \cdot min\{\frac{1}{0.5},\ \frac{0.5}{1}\}$

$= 0.375.$

On top of this, $p1$ makes a hypothesis that $S5$ is chained to the right side of wf_1, which leads to a new workflow wf_2. This is followed by the computation of heuristic value $h(wf_2)$: $h(wf_2) = 0$. As a consequence, $S5$ is added into memo table (cf. Figure 5.8c).

$S2$ can not be chained to both sides of wf, wf_1 and wf_2. $p1$ computes $r_m(S2)$ of adding $S2$ into memo table. Let us assume that, based on the historic observation of $p1$, there is no correct workflow that uses $S2$ ($a'(S2) = 0$). Therefore, $r_m(S2) = \omega = 0.5$. We assume that $S2$ has not been added into memo table.

Till now, the composition process at $p1$ finishes. Neither the composition is correct nor the TTL is exhausted ($TTL = 4$). This triggers $p1$ to suggest a routing path for q. For this, $p1$ computes a ranked list of the remote provider peers of memorized candidate services. We assume that $a'(S5) = 1$,

$a(S5) = 4$, $b'(S5) = 1$ and $b(S5) = 2$.
$sr(p6,S5) = r_m(S5) \cdot \frac{qos(S5)}{|wf_{can}|+1-ps(S5)}$
$= \frac{1}{4+1} \cdot (1 - \frac{1}{2+1}) \cdot \frac{0.6}{2+1-1}$
$= 0.1$.
$L(p1, p6) = 3$. It follows that
$w(p1,q,p6) = \frac{L(p1,p6)}{sr(p6,S5)}$
$= \frac{3}{0.1}$
$= 30$.
As there is only one target peer $p6$, $p1$ computes the query routing path suggestion from itself to $p6$: $p1 \rightarrow p3 \rightarrow p4 \rightarrow p6$. q is routed to $p3$.

- At $p3$, $S1$ can not be used in the workflow of q as no parameter binding is found. Let us assume that $a'(S1) = 1$, $a(S1) = 1$, $b'(S1) = 0$ and $b(S1) = 0$. By memorization strategy, $p3$ adds $S1$ to memo table with the rate:
$r_m(S1) = \frac{a'(S1)}{a(S1)+1}(1 - \frac{b'(S1)}{bS1})$
$= \frac{1}{1+1}(1 - \frac{0}{0+1})$
$= 0.5$.
Let us assume that $S1$ is added (cf. Figure 5.8d). In addition, $p3$ finds that $S9$ can be an alternative of $S10$ in terms of $R(wf)$. The reason is that:
$ch(S9,R(lwf))$
$= \frac{1}{2}(ch_{IO}(S9,R(wf)) + ch_{PE}(S9,R(wf))) \cdot df(ch_{IO}\ (S9,R(wf)),\ ch_{PE}(S9,R(wf)))$
$= \frac{1}{2}(1+1) \cdot min(\frac{1}{1}, \frac{1}{1})$
$= 1.0$
$> \theta$.
This triggers $p3$ to issue a new sub-query q'. To do this, $p3$ initializes a copy of q. $p3$ removes $S10$ from the memo table of q' and chains $S9$ to the right side of workflow of q' without increasing the heuristic value. Denote wf_3, in q', the new workflow with the candidate $S9$. $S5$ is unchained (cf. Figure 5.8f). $p3$ sets $q'.TTL = 3$ and $q'.psug = empty$. Subsequently, $p3$ executes $queryProcess(p3,\ q')$. It finds that $S5$ and $S1$ can be chained to $R(wf_3)$, but not increasing the heuristic value. According to the routing strategy, $p3$ computes the path suggestion for q'. Since the remote item owner peer is $p2$, $q'.psug = p3 \rightarrow p2$.

- q is still routed to $p4$. The latter finds that $S6$ can be chained to the both sides of wf_2. This leads to an increment of heuristic value. Accordingly, $p4$ removes $S10$ and $S5$ from memo table (cf. Figure 5.8e) and these two services are not candidate services any more, but useful services. The workflow of q is now correct. $p4$ makes q backtrack along the inverse query path: $p4 \rightarrow p3 \rightarrow p1 \rightarrow p0$.

- q' is routed to $p2$. No services known by $p2$ can be chained to the workflow of q'. This means that

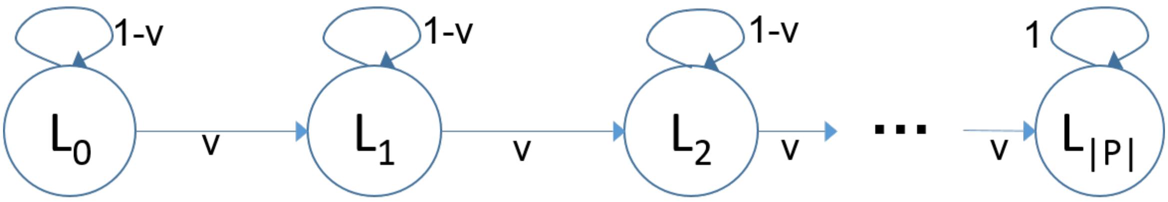


Fig. 5.9 Markov representation of composition process.

$p2$ can not suggest a routing path for q'. Therefore, $p2$ routes q' to a random neighbor $p7$, where the workflow of q' gets correct. The result is shown in Figure 5.1. $p7$ makes q' backtrack along the inverse query path: $p7 \rightarrow p2 \rightarrow p3 \rightarrow p1 \rightarrow p0$.

5.3 A Lower Bound of SPSC Completeness

The use of heuristic value protect a workflow from arbitrary augmentation, which can be caused by mere service chaining. However, the completeness is subject to the knowledge of each peer. The latter does not include a global view to services, constants and topology of the whole network. In contrast, centralized stateful planners, such as [198] [145], leverage a pre-constructed global graph plan that contains the all possible state transmissions. Let F ($F > 0$, $F \in \mathbb{N}$) be the initial TTL value of each query. When solving a request R, SPSC system can be incomplete anyway, if some necessary service is only known by certain peers that can not be traversed within F hops from the requester peer. Therefore, we assume that in a connected unstructured P2P network and all the necessary services $\mathscr{S}$ are known by a set P ($|P| \leq F$, $|P| \geq 1$) of peers that can be traversed in F hops from the requester peer. We name each peer in P as *useful peer*; a peer not in P as *non-useful peer*.

We model the composition process by a finite discrete markov chain, shown in Figure 5.9. Each circle refers to a state during the composition. Please note that the state here is in the context of stochastic process, instead of state-based planning. The state L_i ($i \in \{0, 1, 2, \ldots, |P|\}$) means that a query q has encountered i useful peers. With probability v ($v \in (0,1)$, $u \in \mathbb{R}$), q encounters the next useful peer; while with probability $(1-v)$, q encounters a non-useful peer. On top of this model, starting from the state L_0, a correct workflow can be composed, if a stochastic walk in this markov chain can reach $L_{|P|}$ in F steps. This indicates to compute the last element $\vec{x}^{(F)}_{|P|+1}$ in the vector $\vec{x}^{(F)}$:

$$\vec{x}^{(F)} = \vec{x}^{(0)} D^F \tag{5.9}$$

where $\vec{x}^{(0)} = (1, 0, \cdots, 0)$ is the initial state vector with $(|P|+1)$ elements; D is the transition matrix:

$$D = \begin{array}{c|ccccccc|} & L_0 & L_1 & L_2 & L_3 & \cdots & L_{|P|-1} & L_{|P|} \\ \hline L_0 & 1-v & v & 0 & 0 & \cdots & 0 & 0 \\ L_1 & 0 & 1-v & v & 0 & \cdots & 0 & 0 \\ L_2 & 0 & 0 & 1-v & v & \cdots & 0 & 0 \\ L_3 & 0 & 0 & 0 & 1-v & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ L_{|P|-1} & 0 & 0 & 0 & 0 & \cdots & 1-v & v \\ L_{|P|} & 0 & 0 & 0 & 0 & \cdots & 0 & 1 \end{array} \tag{5.10}$$

If $|P| = 1$, we have the lower bound $\vec{x}_2^{(F)} = v$. If $|P| > 1$, we compute:

$$\vec{x}_{|P|+1}^{(F)} = \begin{cases} 0 & \text{, if } F < |P|; \\ \sum_{j=|P|+1}^{F} \binom{j-1}{|P|-1} \cdot v^{|P|} \cdot (1-v)^{j-|P|} + v^{|P|} & \text{, if } F \geq |P|. \end{cases} \tag{5.11}$$

If $F < |P|$, the lower bound is 0, as q is not possible to reach all useful peers. When $F = |P|$, the lower bound is $v^{|P|}$. It means that the each routing of q should choose the correct target peer. This is the hardest case. When $F > |P|$, the lower bound is largely determined by the volume of v. Denote dg the maximum connectivity degree of a peer; N the total number of peers in the network; n_S the number of peers that know about a useful service $S \in \mathscr{S}$. We have:

$$v = \frac{\frac{n_S \cdot dg}{N}}{dg} = \frac{n_S}{N} \tag{5.12}$$

We assume that in an initial network, only one peer knows about a service $S \in \mathscr{S}$ and the others do not know S. By memorization strategy and peer local observation mechanism, S is supposed to be propagated to other peers by queries, which, without loss of generality, are assumed to happen one by one. i.e. The value of $n_S^{(t)}$ is increasing in line with the system evolution, where $t \in \mathbb{N}^+$ is the number of queries happened in the system. $n_S^{(0)} = 1$. We analyze

1. whether all peers in the network will know S, or not;

2. how fast this epidemic process could be.

We model this process by the disease reproduction [352] [168] in epidemiology theory. The peers in the network are treated as a population, in which one peer is infected (knows S) and the others are susceptible (do not know S). The contact between the infected and the susceptible is done via query memorizing S and peer local observation on S. In this model, we do not consider the birth, mortality or recovery of individuals (peers). We estimate the basic reproduction number n', i.e. the expected number of peers that do not know S before, but get to know S after the t-th query. In epidemiology theory, this number is computed as follows:

$$n' = \tau \cdot \bar{c} \tag{5.13}$$

where τ is the transmissibility (the probability of infection given contact between a susceptible and infected individual); $\bar{c}$ is the average rate of contact between susceptible and infected individuals. In our scenario, $\tau = 1$ due to the peer local observation strategy and $\bar{c}$ is calculated by the following:

$$\begin{aligned} \bar{c} &= \frac{n_S^{(t-1)}}{N} \cdot F \cdot \omega^* \cdot \frac{F \cdot (N - n_S^{(t-1)})}{N} \cdot 1 \\ &= \frac{F^2}{N^2} \omega^* \cdot n_S^{(t-1)} \cdot (N - n_S^{(t-1)}). \end{aligned} \tag{5.14}$$

where ω^* is the average memorization rate of S; $\frac{n_S^{(t-1)}}{N} \cdot F \cdot \omega^*$ is the chance of the t-th query encountering and memorizing S; $\frac{F \cdot (N - n_S^{(t-1)})}{N} \cdot 1$ is the expected number of susceptible peers that get to know S (infected), the "1" at the rightmost means the number of queries considered in this reproduction process. On top of this, we have:

$$n_S^{(t)} = n_S^{(t-1)} + \frac{F^2}{N^2} \omega^* \cdot n_S^{(t-1)} \cdot (N - n_S^{(t-1)}). \tag{5.15}$$

In order to estimate $n_S^{(\infty)}$, we assume that $t \rightarrow \infty$ and solve the function:

$$n_S^{(\infty)} = n_S^{(\infty)} + \frac{F^2}{N^2} \omega^* \cdot n_S^{(\infty)} \cdot (N - n_S^{(\infty)}) \tag{5.16}$$

The result is $n_S^{(\infty)} = N$. It means that the reproduction of the knowledge about S will spread over all the peers in the network. This replies to the issue 1 above and we can observe that $v \rightarrow 1$ in the very end.

In the sequel, we check the speed of this epidemic process. As $n_S^{(\infty)} = N$, we know that the rightmost part $N - n_S^{(t-1)} > 0$ in Equation 5.15, when t is not ∞. We replace the part $N - n_S^{(t-1)}$ by a small positive real number ε and achieve the following equations:

$$\begin{aligned} n_S^{(t)} &= n_S^{(t-1)} + \frac{F^2}{N^2} \omega^* \cdot n_S^{(t-1)} \cdot \varepsilon \\ &= n_S^{(t-1)} \cdot (1 + \frac{F^2 \omega^* \varepsilon}{N^2}). \end{aligned} \tag{5.17}$$

$$n_S^{(t)} = n_S^{(0)} \cdot (1 + \frac{F^2 \omega^* \varepsilon}{N^2})^t. \tag{5.18}$$

By [368], this function is converging to N sublinearly and its convergence rate is $(1 + \frac{F^2 \omega^* \varepsilon}{N^2})$.

In detail, we check the convergence rate of the series $(n_S^{(t)})^{-1}$, i.e. $(1+\frac{F^2\omega^*\varepsilon}{N^2})^{-t}$. To simplify the formula, we assume $x=\frac{F^2\omega^*\varepsilon}{N^2}$ and observe that $(1+\frac{F^2\omega^*\varepsilon}{N^2})^{-t} \to 1$, when $t \to \infty$. We compute the limitation:

$$\begin{aligned} \lim_{t\to\infty} \frac{|\frac{1}{(1+x)^t}-1|}{|\frac{1}{(1+x)^{(t-1)}}-1|} &= \lim_{t\to\infty} \frac{|1-(1+x)^t|}{|1-(1+x)^{(t-1)}|(1+x)} \\ &= \lim_{t\to\infty} \frac{(1+x)^t-1}{((1+x)^{(t-1)}-1)(1+x)}. \end{aligned} \tag{5.19}$$

Let $f(t)=\frac{(1+x)^t-1}{((1+x)^{(t-1)}-1)(1+x)}$ and we define two functions $g_1(t)$ and $g_2(t)$ as follows:

$$\begin{aligned} g_1(t) &= \frac{(1+x)^t-1}{((1+x)^t-1)(1+x)}; \\ g_2(t) &= \frac{(1+x)^t-1}{(1+x)^{(t-1)}-1} \end{aligned} \tag{5.20}$$

We have:

$$\begin{aligned} f(t) &> g_1(t) \\ &= \frac{1}{1+x}. \end{aligned} \tag{5.21}$$

and

$$\begin{aligned} f(t) &< g_2(t) \\ &< \frac{(1+x)^t}{(1+x)^{(t-1)}} \\ &= \frac{1+x}{1} \end{aligned} \tag{5.22}$$

As $\lim_{t\to\infty} g_1(t)=1$ and $\lim_{t\to\infty} g_2(t)=1$, we have $\lim_{t\to\infty} f(t)=1$. It follows that the series $(1+\frac{F^2\omega^*\varepsilon}{N^2})^{-t}$ sub-linearly converges to 1. Therefore, $n_S^{(t)}$ converges to N sub-linearly.

5.4 SPSC Correctness

In this section, we formally prove the correctness of SPSC approach. Intuitively, the basic idea is to prove first that a correct workflow is achieved when its heuristic value equals to 1. This is the correctness of heuristics. We show its proof in Section 5.4.1. Then we prove that the heuristic value of the workflow in each (sub-)query is monotonically increasing, which means that the SPSC system can lead the composition towards the right direction. In other words, the heuristic value of a workflow is getting larger during the distributed composition process and it reaches 1 with a lower bound given in Section 5.3. This is the correctness of the overall distributed composition. We show its proof in Section 5.4.2.

5.4.1 Correctness of Heuristics

Let q be a query for a request R, which initial TTL value is F ($F > 0$, $F \in \mathbb{N}$); wf the workflow of q; h the heuristic value of wf; S any service used in wf.

Lemma 5.1. *if $h = 1$, each input parameter of a service in wf is bound with an output parameter of another service in wf or an input parameter in $R.I$.*
Proof: we prove this by contradiction. Let us assume that there exists an unbound parameter $?x \in S.I$, $S \in wf$, but $h = 1$. During each chaining, the unbound input parameters of all the services in the right side $R(wf)$ (cf. Definition 4.) of wf is considered. By Equation 5.3, the assumption above means that $|M| < |R(wf).I|$, where M is the set of bound parameters in $R(wf).I$ (cf. Section 5.2.2). It follows that $ch_{IO}(L(wf), R(wf)) < 1$. Therefore, by Equation 5.5, $h < 1$ holds. Contradiction. ■

Lemma 5.2. *if $h = 1$, wf plugs into the request R.*
Proof: Without loss of generality, in the composition process, $R.I$ is treated as the output of some assumed dummy service S_L, which output will be used by the workflow wf. We assume that at some peer p, a service S is determined to be chained to $S_L.O$. This process outputs a series of bindings. Each indicates that for the parameters $?x$ ($?x \in S_L.O$) of type X and $?y$ ($?y \in S.I$) of type Y, $X \sqsubseteq Y$ holds. This is guaranteed by the chaining determination process (cf. Section 5.2.2). On the other hand, in the composition process, $R.O$ is treated as the input of some assumed dummy service S_R, which input parameters use the outputs of the services in wf. If a service S' is determined to be chained to $S_R.I$, for each pair of bounded parameters $?x'$ ($?x' \in S_R.I$) of type X' and $?y'$ ($?y \in S'.O$) of type Y', $X' \sqsubseteq Y'$ holds. When h reaches 1, by Lemma 5.1 the input parameters of $R.I$ and all services in wf have been bound. Therefore, wf plugs into R, when $h = 1$. ■

Lemma 5.3. *if $h = 1$, given the parameter bindings, there is no contradiction in $\{S.E\} \cup R.P$: for all S in wf.*
Proof: When computing PE score, peer additionally checks whether the contradiction exists in $(\bigcup_{S \text{ in } wf} S.E) \cup R.P$. If it is the case, the peer will set the PE chaining score for S to 0. It leads the total chaining score to be 0 (cf. Equation 5.1). In this case, S will not be chained at all. Therefore, there is no contradiction in $(\bigcup_{S \text{ in } wf} S.E) \cup R.P$. ■

Lemma 5.4. *if $h = 1$, given the parameter bindings, all preconditions of the services in wf are satisfied.*
Proof: We prove this by contradiction. Let us assume that there exists a service $S \in wf$, which

precondition is not satisfied, but $h = 1$. During each chaining, the precondition of all the services in the right side $R(wf)$ of wf is considered. By Equation 5.4, it follows that $|SP| < |R(wf).P|$, where SP is the set of satisfied clauses in $R(wf).P$. Therefore, $ch_{PE}(L(wf), R(wf)) < 1$. It results in that $h < 1$ (cf. Equation 5.5). Contradiction. ■

Lemma 5.5. *if $h = 1$, given the parameter bindings, R.E can be implied based on the truth of all the effects of services in wf.*

Proof: We prove this by contradiction. In the composition process, the effect of the request $R.E$ is treated as the precondition of an assumed dummy service S_R. Let us assume that $S_R.P$ is not satisfied, but $h = 1$. During each chaining, the precondition of all services in the right side $R(wf)$ is considered. The dissatisfaction of $S_R.P$ indicates that $|SP| < |R(wf).P|$, where SP is the set of satisfied clauses in $R(wf).P$, by Equation 5.4. It follows that $ch_{PE}(L(wf), R(wf)) < 1$. It results in that $h < 1$. Contradiction. ■

Theorem 5.1. *if $h = 1$, wf is correct.*

Proof: A workflow is correct if the five criteria in Definition 4. are all satisfied. Based on the Lemma 5.1. $\sim$ Lemma 5.5., we have that wf is correct, given $h = 1$. ■

Corollary 5.1. *if $h \in [0, 1)$, wf is not correct.*

Proof: If $h < 1$, it means that there exist either unbound input parameter of a service in wf or an unsatisfied precondition of a service in wf. In any of the both cases, the criteria in Definition 4. is not fulfilled. Therefore, wf is not correct, if $h \in [0, 1)$. ■

5.4.2 Correctness of Distributed Composition

Lemma 5.6. *The heuristic value h of a (sub-)query q is monotonically increasing during the entire query processing on all its traversed peers.*

Proof: Based on Alg.6, a new heuristic value h' is used only if $h' > h$. Therefore, no peer can decrease the heuristic value by its local composition process. In SPSC, a query is routed from one peer to another by its routing strategy. Each peer takes the heuristic value of the workflow in the received query as the old value. Hence, the magnitude of h is preserved regardless of query transmission. A sub-query q' of q is derived, when an alternative service S' of a service S in wf is found (cf. Section 5.2.3). In this case, the heuristic value in q' can be smaller than h. However, q' is already an independent query in terms of composition and routing. Its heuristic value is still monotonically increasing during its entire query processing on all its traversed peers. ■

Corollary 5.2. *The heuristic of SPSC local service composition is consistent. I.e. After the local composition at a peer p, $\nexists\, S \in \mathscr{S}_p$, which is chainable to the current workflow yielding a new heuris-*

tic value h' such that h' is larger than the current heuristic value h.

Proof: We prove this by contradiction. Let us assume that there exists a service $S \in \mathscr{S}_p$ that can be chained in the workflow and yields a larger heuristic value. According to Alg. 6, the local composition process will not return, but continuously consider the chaining of S. Contradiction. ■

We prove the correctness of SPSC approach based on the Floyd Hoare theory [99][144] as well as the notes of the lecture Polyhedral Compilation Foundations[3] of the University of California Los Angeles. This theory considers the proof of correctness of a loop algorithm by checking the invariants at the end of iteration. If the invariants are true for all iterations, the algorithm is correct. To do this, the following three aspects should be separately checked:

- *Initialization*: Whether or not the invariants are true before the first iteration of the loop.
- *Maintenance*: If the invariants are true for the $i-$th iteration, whether or not they are still true for the $i+1$-th iteration.
- *Termination*: When the loop terminates, whether or not the invariants are true on the entire input.

To apply this method, we reduce the distributed composition problem of SPSC to a centralized data processing problem. Consider the whole P2P network as a huge computer. Each peer p is regarded as a processing unit and each unit has its local knowledge about services ($\mathscr{S}_p$). A query q is treated as a task. Processed by one unit, q is iteratively delivered to another unit for processing, until its TTL has been exhausted or the task is done (correct workflow composed). On each unit, the composition process (cf. Alg. 6) is executed locally, which is meant with one iteration of the reduced data processing problem.

We specify the following two invariants, which are crucial for achieving $h = 1$:

- *h is monotonically increasing for each (sub-)query q*: The underlying idea of this invariant is that the SPSC composition for each service eventually leads the partial solution (workflow) to the right direction. In addition, we will prove that this feature preserves for each (sub-)query.
- *To q, no alternative service is missed*: This invariant indicates no sub-query will be missed by SPSC approach. In other words, all possible workflow will be investigated.

Theorem 5.2. *SPSC is correct : Given a query q, SPSC returns a correct solution ($h = 1$) with a lower bound given in Section 5.3, if that solution exists within the peers reachable by q in F hops. .*

Proof:

- *Initialization*: The initial value of h is 0. Before any process, the value of h is not changed. Therefore, h is not decreasing. Before any process, no alternative service is found, as the workflow is empty. Hence, no alternative service is missed.

---

[3] `http://www.cs.ucla.edu/~pouchet/lectures/doc/888.11.algo.6.pdf`

N (E)	the total number of peers (edges) in an unstructured P2P network
$\mathbf{m_1}$	the number of primitive terms (concepts, roles and role cardinality restrictions) defined in V
$\mathbf{\|O_p\|}$	the number of concepts defined in peer local ontology
$\mathbf{m_2}$	the number of predicates used in the precondition/effect formula of a service
$\mathbf{m_3}$	the maximal number of input/output variables of a service
$\mathbf{l_{ch}}$	the computational complexity of the service chaining score computation: $l_{ch} = l_{bp} + l_{sat}$, where $l_{bp} \sim \boldsymbol{O}(m_3^2 \cdot m_1^{m_1})$ is the computational complexity for determining service variable bindings; $\boldsymbol{O}(m_1^{m_1})$ is the computational complexity of concept similarity measure. $l_{sat} \sim \boldsymbol{O}(\|O_p\|^{m_3} \cdot m_2)$ is the complexity for determining the satisfaction of service effect formula [240].
n	the number of services a peer can know about
F	the initial value of query TTL
$\mathbf{l_{sp}}$	the computation complexity of any shortest path in the network [105]. $\mathbf{l_{sp}} = \boldsymbol{O}(E + NlogN)$
s	the size of a service description file
L	the number of services in the current workflow
j	a counter ($j \in \mathbb{N}^+$) used by the following complexity analysis

Table 5.1 Useful symbols for the complexity analysis of SPSC approach.

- *Maintenance*: Let us assume that a processing unit p receives a query q, with its current heuristic value h. According to Alg. 6, its new heuristic value h' will not smaller than h. Thus, the output of the iteration at p does not decrease the heuristic value of q. On the other hand, Alg. 6 guarantees that it checks all alternative services for each service in wf. No alternative service is missed at p.

- *Termination*: The entire process terminates, if TTL of q is 0 or $h = 1$. When the entire process terminates, the heuristic value of h is not smaller than its initial value. In addition, no alternative service for each service in wf has been missed, as no one is missed in each iteration.

For each q, its heuristic value is not decreasing and has a lower bound (cf. Section 5.3). When it reaches 1, the workflow of q is correct, by Theorem 5.1. ■

5.5 Complexity

In this section. we analyze the computational and traffic complexities of important processes of SPSC approach. In Table. 5.1, the meaning of useful symbols is listed. We prove that (i) the computational complexity of peer local composition is $\boldsymbol{O}(L \cdot (2n)^n \cdot l_{ch})$; (ii) the computational complexity of SPSC path suggestion is $\boldsymbol{O}(L^2 \cdot F \cdot (E + NlogN))$; and (iii) the traffic complexity of SPSC query routing is $\boldsymbol{O}((L \cdot 2^n + 1)^F)$.

Lemma 5.7 *The computational complexity of peer local composition is* $\boldsymbol{O}(L \cdot (2n)^n \cdot l_{ch})$.
Proof: In Alg.6, p computes the chaining scores for all $S \in \mathscr{S}_p$. Its computation complexity is $\boldsymbol{O}(n \cdot l_{ch})$. Once a service is (hypothetically) chained to the workflow, the computation restarts. This yields in total the complexity $\boldsymbol{O}(n^n \cdot l_{ch})$. In addition, the local composition process also checks the alternative services for each service in the current workflow wf. For this, p will additionally process up to $L \cdot 2^n$ sub-queries, and each sub-query will be processed again in $\boldsymbol{O}(n^n \cdot l_{ch})$. It follows that the complexity local composition goes to $\boldsymbol{O}(L \cdot 2^n \cdot n^n \cdot l_{ch} + n^n \cdot l_{ch}) \sim \boldsymbol{O}(L \cdot (2n)^n \cdot l_{ch})$. ■

Lemma 5.8 *The computational complexity of SPSC path suggestion is* $\boldsymbol{O}(L^2 \cdot F \cdot (E + NlogN))$.
Proof: In order to suggest a path for a query q, p computes the inverse importance score per traffic cost for each candidate service S in wf. $r_m(S)$ can be computed incrementally based on peer observation. The check of the workflow size and dependencies can be done in $\boldsymbol{O}(L)$. For each augmentation of the suggested path, p select the key peer with the minimum weight. This costs at most $\boldsymbol{O}(L^2 \cdot (E + NlogN))$, where the part $\boldsymbol{O}(E + NlogN)$ refers to the complexity of shortest path computation. Since the initial TTL is F, there can have at most F augmentations. In total, the computation complexity for path suggestion at p is $\boldsymbol{O}(L^2 \cdot F \cdot (E + NlogN))$. ■

Lemma 5.9 *The traffic complexity of SPSC query routing is* $\boldsymbol{O}((L \cdot 2^n + 1)^F)$.
Proof: We analyzes the number of messages needed for a query in its worst case. As the query forwarding and backtracking journeys spend exactly the same number of messages, we analyze the forwarding process only. During composition process, each peer p at most issues $\boldsymbol{O}(L \cdot 2^n)$ sub-queries. Each sub-query adopts the current query TTL value. Thus, the total messages needed during the query forwarding phase is $\sum_{j=0}^{F-1}((L \cdot 2^n + 1)^j)$. It follows that the traffic complexity for each query is $\boldsymbol{O}((L \cdot 2^n + 1)^F)$. Each message can contain at most nN distinct service profiles. It follows that the total traffic load for one query is $\boldsymbol{O}((L \cdot 2^n + 1)^F \cdot nNs)$. ■

5.6 Experimental Evaluation

In this section, we present and discuss the results of our experimental evaluation of the performance of SPSC.

Settings: For the experiments, we simulated unstructured P2P networks with 1000 peers based on the P2P framework[4]. For the network topology, we choose random graph (RG) and random power law graph (RPLG). The latter is known to be a realistic model for social networks. The average connection degree of each peer in RG and RPLG are 10.295 and 4.457, respectively. The skew value of RPLG is 1.3. For the sake of efficiency, we disable the IP based communication feature of each peer, but enable each to access a global data structure, where the query being processed is stored. In

---

[4] `http://sourceforge.net/projects/mymedia-peer/`

addition, we adapt the query routing path suggestion library of S2P2P [47] implementation for the routing of SPSC query. The initial time-to-live value of each query is 10. The chaining threshold is 0.3[5] and the binding threshold is 1.0.

To the best of our knowledge, there is no test collection publicly available particularly for the stateless composition of IOPE based semantic web services. The international planning competition 2011 (IPC 2011) test collection[6] is well-known to be a good option for testing state-based planners. However, it is not suitable for the stateless case, since each query of the latter needs to be defined as a semantic service with IOPE. Mere the goal states of IPC a query is not adequate to derive the exact output of the targeted workflow. Another known test bed is the test set generator of IEEE web service challenge (WSC)[7], which is able to generate services without preconditions and effects. This is not enough to test our IOPE service composition method. To this end, we created a test collection with 40 IOPE based semantic services which are used by the solutions of 7 queries. The longest optimal solution contains 9 services and the shortest uses 5 services. Besides those correct solutions of the queries, incorrect solutions with dead-ends are also designed. The services in the test collection are created based on an ontology that comprises 36 concepts and 40 roles. Ontology based strict concept subsumption is used in the IO chaining score computation. In the experiments, the reasoning and implication work with the $\boldsymbol{AL}$ sub-language of OWL-DL.

We employ the uniform at random model as the service (item) distribution policy. In each test, one thousand queries are issued. Each query is randomly selected from the query set and its requester peer is randomly chosen as well. The experimental evaluation is conducted on a PC with Intel(R) Core(TM) i7 CPU (2.80GHz) and 8 GB memory.

Evaluation Measures: The aim of our experimental evaluation is to test the composition quality, network traffic overhead and query response time of SPSC in unstructured P2P networks with different settings. We check them by considering the following metrics:

- *Average cumulative recall CRE_m within distance m*: Let Q be the multiset of issued queries. For each query q, $EC_{m,q} \in \{0,1\}$ denotes whether (or not) there exist a set of services on the peers reachable within m ($m > 0$, $m \in \mathbb{N}$) hops from the requester peer. Such a set of services is needed for a correct solution to a query. Please note this value is determined based on the current item distribution that evolutes all the time due to the item information propagation and peer local observation. $C_{m,q} \in \{0,1\}$ denotes whether a correct solution of q has been found within the peers that are reachable in m hops from the requester peer.

$$CRE_m = \frac{1}{|Q|} \sum_{q \in Q} \frac{C_{m,q}}{EC_{m,q}}. \tag{5.23}$$

---

[5] The setting on chaining threshold can vary for different test data.

[6] `http://www.plg.inf.uc3m.es/ipc2011-deterministic/Resources.html`

[7] `http://www.ws-challenge.org/`

The reason we choose this metrics but not average precision lies in that the composition process returns only the correct solutions. This is different with P2P-based search, as the latter can also return those results that are false positive.

- *Average QoS rate QUR of resolved queries*: Each peer in SPSC concludes a path suggestion for routing a query based on the quality of candidate services. This process tends to route a query to the peer which service has higher quality and therefore aims at achieving better workflow quality. To test this feature, besides the services in the test collection with their pre-defined QoS, we deploy another copy of them but with lower quality. The QoS value of each equals to 50% of the pre-defined value. We check to what extent a query at run time can be resolved by the services with the highest (expected) quality. For this, we define the run time quality $rtqu_q$ of a resolved query q as the average quality of all the useful services; expected quality $exqu_q$ of a resolved query q as the average of pre-defined quality of all its useful services.

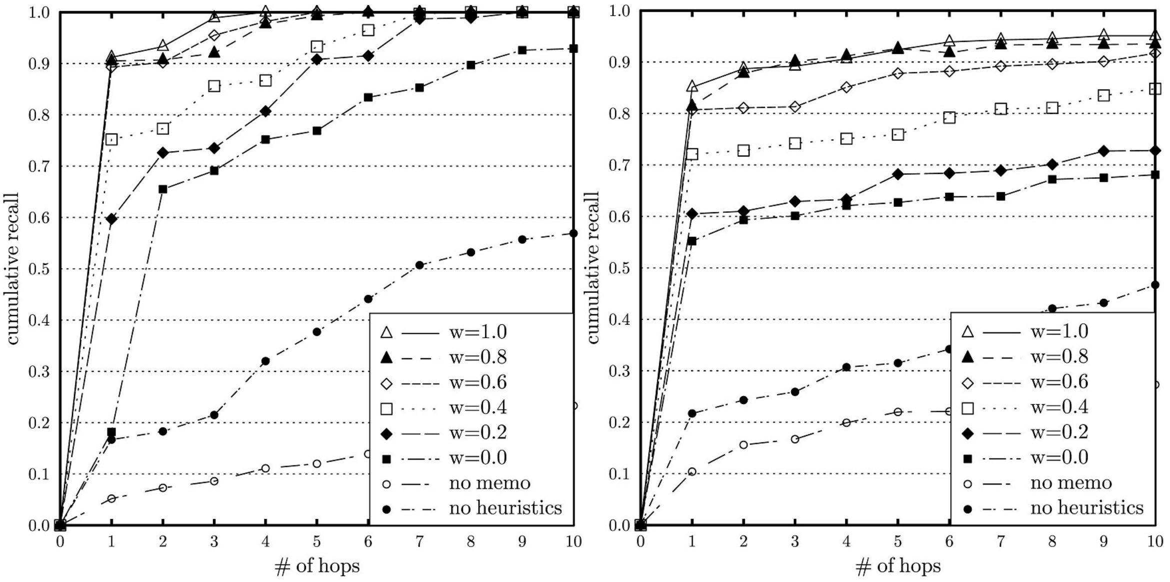
$$QUR = \frac{1}{|Q|} \sum_{q \in Q} \frac{rtqu_q}{exqu_q} \tag{5.24}$$

- *Average number of messages #M per query*: This value covers the forwarding messages only, as they can reflect the number of peers that have been detected for the purpose of composition.

$$\#M = \frac{1}{|Q|} \sum_{q \in Q} M_q. \tag{5.25}$$

- *Average traffic load size TS of query*: In addition to the query branching mechanism, a SPSC query is able to selectively "memorize" the definition of potentially useful services and propagate them to remote peers. We test the averaged total size in KB of the forwarding messages generated by a query.
- *Number of messages each peer has forwarded*: We count the number of forwarding messages for each peer and order them, in order to test the possible hot spots of SPSC in RG and RPLG networks.
- *Average query response time AQRT*: We test how much time is needed for a requester peer to achieve an answer to its query. As we disabled the network communication of peers, the response time of a query consists of mainly the total composition time on its traversed peers.

Cumulative Recall and Quality of Solutions: We compare the cumulative recall of SPSC with different memorization rates ω. The data is collected immediately after 1000 queries. Shown in Figure 5.11, the cumulative recall of SPSC with memorization strategy in both the RG and RPLG based unstructured P2P networks largely outperforms the cases without memorization strategy or heuristic value mechanism. The latter variants are considered as comparative baselines. Overall, SPSC achieves 10% to 20% more cumulative recall with RG than RPLG. The reason is that the generated



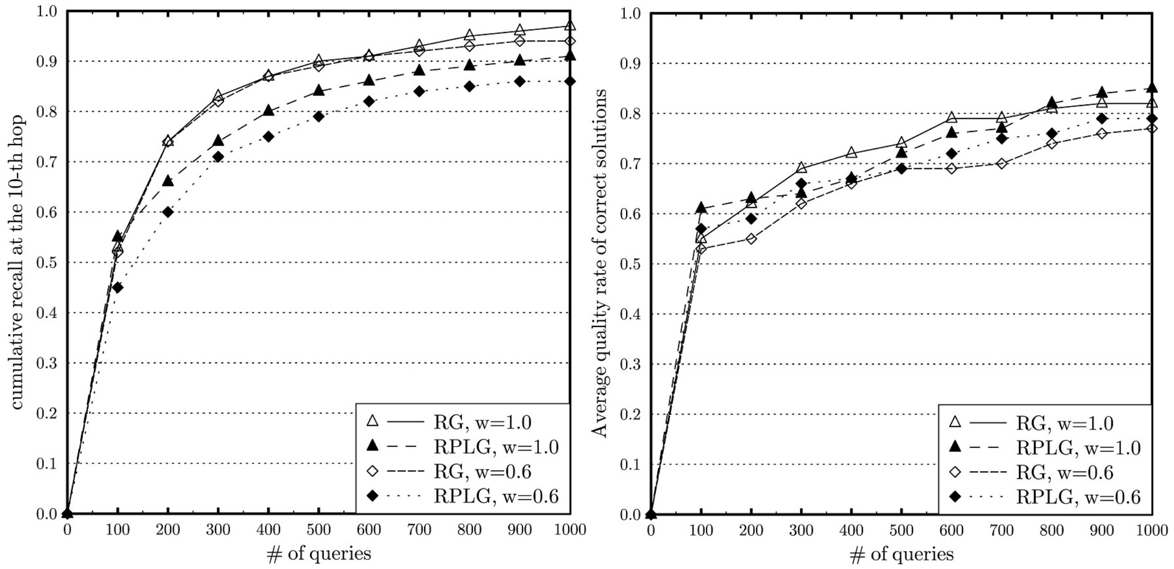
(a) RG

(b) RPLG

Fig. 5.10 Cumulative recall of SPSC in RG and RPLG based networks with 1000 peers.

RPLG contains islands, but RG does not. By memorizing the potential useful services, a query has more chance to be correctly resolved. In RPLG, peers in an island have only the connections to few peers. The latter, with large chance, might not have the necessary services for resolving a query, though each peer is able to observe and reuse the services in its processed queries. In contrast, RG is connected. This results in that, via multiple queries, the important services can be transitively observed by any other peer in the network. This increases the chance of correctly resolving queries.

In each of the both configurations, the test result reveals that the higher ω is, the better cumulative recall can be achieved. In the RG based network with memorization strategy, SPSC successfully resolved more than 90% requests before their query TTLs are exhausted. Particularly, the correct solutions for at least 90% queries are composed at the first hops, when the memorization rate is relatively larger ($\omega \geq 0.6$). This indicates that the necessary services for resolving a request are propagated to a certain group of peers and the queries in the future have been properly routed by the path suggestion policy. However, the result also shows that the increment of composition performance gets slower when $\omega \geq 0.6$. This indicates that the effectiveness of enlarging the memorization rate becomes weaker after certain "sweet point" of ω has been passed, since the peers in the system already have adequate knowledge for the composition. Please note such value can be various with different data. In addition, the differences of the cumulative recall at the first hop and the last hop in the RG based configurations are larger than the cases with RPLG. This is caused by the characteristics of network topology. In RPLG, the most peers have one immediately neighbor and a very small number of peers have large connectivity degrees. Most queries issued by the former peers can reach the latter backbone peers. Such peers have more chance to observe services and resolve requests.



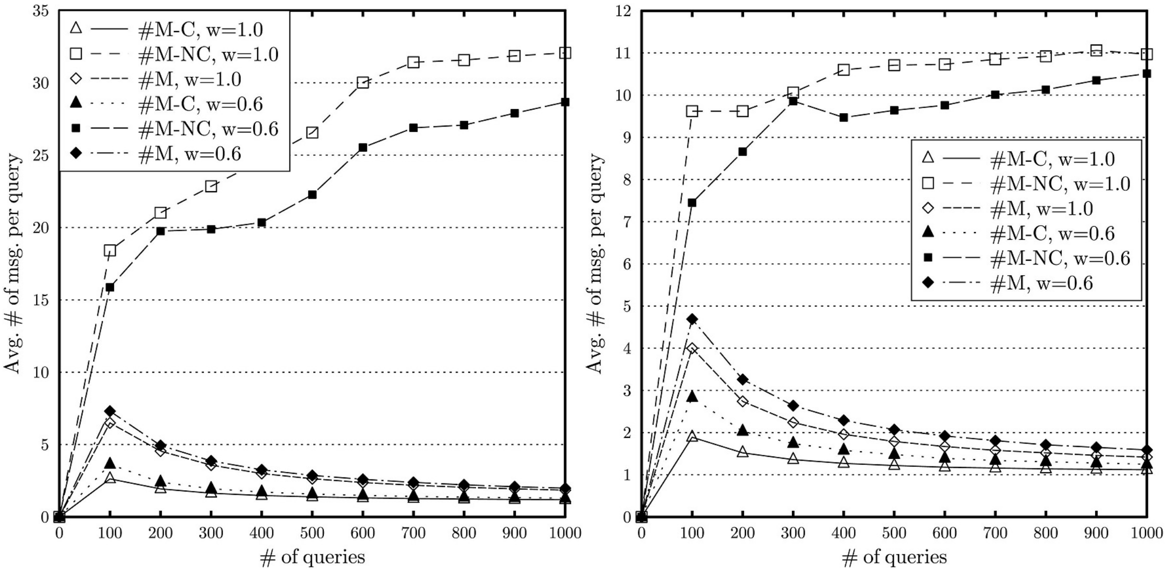
(a) Cumulative recall at the 10-th hop

(b) Quality rate of correct solution

Fig. 5.11 System evolution speed (a) and solution quality (b)

Besides, we check the evolution of the cumulative recall at the 10-th hop over queries (cf. Figure 5.11a). The result indicates that the system can to some extent alleviate the cold start problem since after about 500 queries, more than 80% requests can be resolved correctly regardless of the network topology. The reason is that the peer observation mechanism helps each peer to enrich the local knowledge about services and moreover the memorization strategy decreases the chance of missing potentially useful services. Further, we check how the quality of solution chances over queries. The result (cf. Figure 5.11b) shows that each peer in SPSC can take use of those services with better QoS during its local composition and query routing. If the expected quality of a solution is 1.0, the requests can achieve about 80% of expectation after about 1000 queries.

Network Traffic Overhead: We test the average number of messages per query of SPSC in RG and RPLG based unstructured P2P networks. Shown in Figure 5.12, the overall average number of messages per query (#M) of SPSC with RG (RPLG) is decreasing over the number of queries and it converges to some value less than 3 (2) at the 1000-th query. The average number of messages per correct query (#M_C) is decreasing in both configurations, as the knowledge about services of each peer is enriched over time by means of its local observation process, which directly benefits the composition and also is useful to the routing path suggestion. Consequently, a correct solution can be composed by less number of peers. On the other hand, the average number of investigated peers by unresolved queries (#M_{NC}) is large in each of the both configurations. Relying on the query branching mechanism, a query is able to detect the possible solution with alternative service (cf. Section 5.2.3) by issuing a sub-query which TTL is initialized with current query time-to-live value.



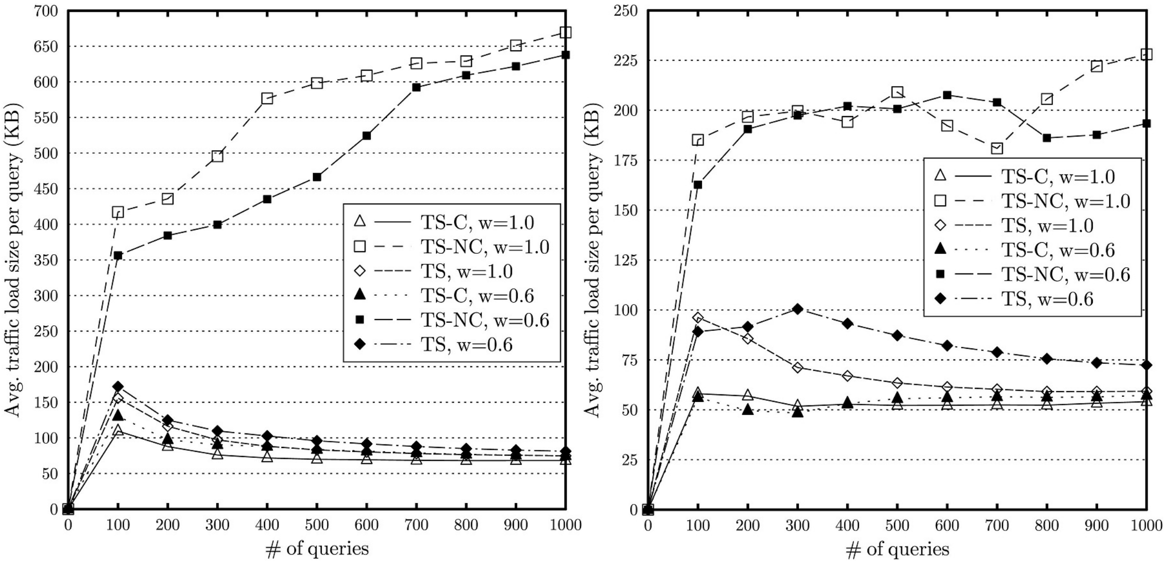
(a) RG

(b) RPLG

Fig. 5.12 Average number of messages per query of SPSC in RG and RPLG based networks with 1000 peers.

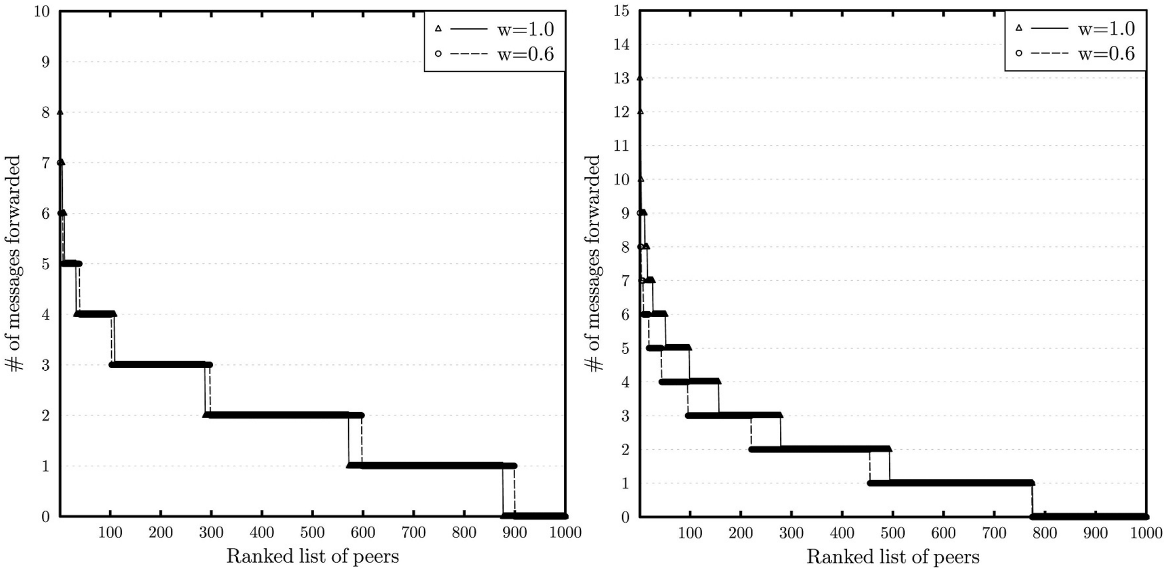
Of course, successfully resolved queries can have branches. However, the chance is much lower than the unresolved ones. The reason lies in that, in SPSC, a peer issues a sub-query only if its received query can not be resolved locally and it has found alternative services. This yields a relatively smaller number of messages overall.

In Figure 5.12, the run of SPSC in RG based network costs more network traffic than the RPLG case, since the peers in RG based networks have similar connectivity degrees but this is not true in RPLG. A query in a network of the latter kind can reach some peers with large degree and those peers have the chance of knowing more services. As a result, a query can be resolved within less hops. In addition, an unresolved query also detects less number of peers in RPLG based network than the configuration with RG, because a large number of peers in RPLG have only one neighbor. A (sub)-query is forced to backtrack as no new neighbor can be used for routing. Furthermore, we compare the number of messages per query of SPSC with different memorization rate values ω (1.0 and 0.6). The result reveals that the larger value of ω yields the smaller network traffic for resolved queries but in the other way around for the unresolved ones. We discuss the reason as follows: a peer tries to resolve a query based on all of its known services. If it can satisfy the request locally, sub-queries will not be issued. Besides, if a peer knows more services, it has larger chance of resolving a request locally, which makes a query to backtrack. The average number of messages per resolved query is therefore smaller in the cases with larger ω value. In contrast, if a peer can not resolve a query locally and it finds alternative services, sub-queries will be issued. In this situation, larger value of ω results in more sub-queries and network traffics.



(a) RG (b) RPLG

Fig. 5.13 Average traffic size per query (in KB) of SPSC in RG and RPLG based networks with 1000 peers.



(a) RG (b) RPLG

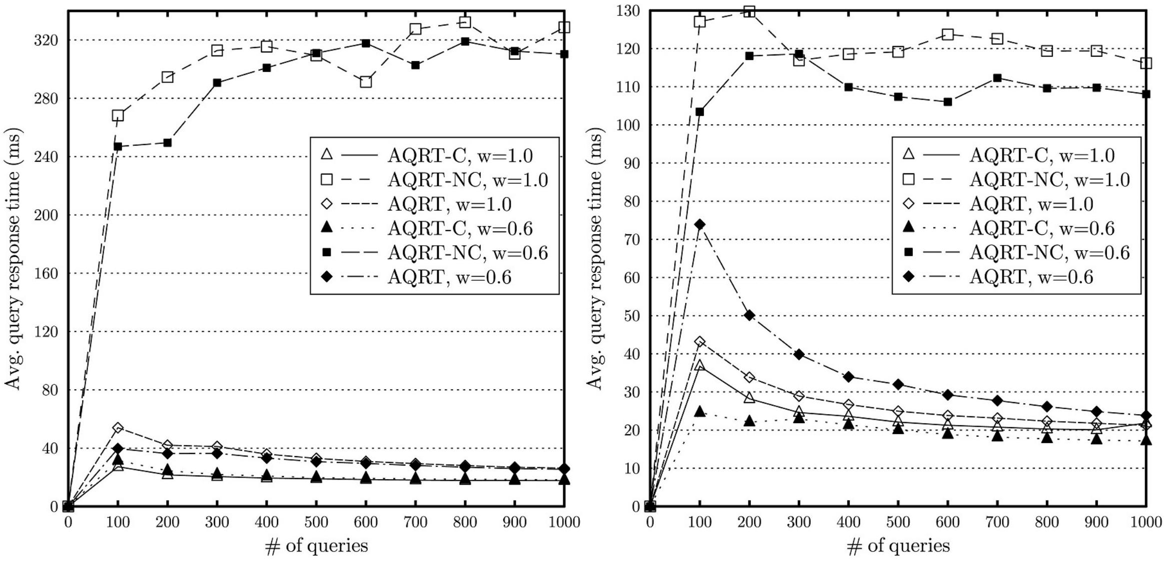
Fig. 5.14 Number of forwarded messages of each peer in RG and RPLG based networks with 1000 peers.

We further test the average traffic load size per query (TS). Shown in Figure 5.13, the traffic load size of resolved (TS_C) query is much smaller than the one which requests are unresolved (TS_{NC}), regardless of the network topology. The reason is that a query does not need to be routed farther, if some peer can correctly answer it. Based on the results of former tests, a large number of queries are resolved after they have been processed by few number of peers. It follows that the generated traffic load gets smaller. In contrast, if a query can not be resolved by many peers, each of those will issue possible sub-queries in order to investigate for alternative services. The more peers a query traverses, the more sub-queries can be issued. Hence, the traffic load will increase dramatically in this situation. Another observation from Figure 5.13 lies in that the traffic load of SPSC with RPLG is smaller than the traffic in RG case. This is caused by the network topology. In RPLG, a query is forced to backtrack if it can not be routed any further due to no untraversed neighbor, though the TTL has not been exhausted. This limits the traffic load size of, in particular, those unresolved queries.

Another interesting feature of distributed system is whether the SPSC system contains hot spots, or not. To test this, we check the numbers of messages forwarded by all peers and rank them in a descending order. Shown in Figure 5.14, the hot spot of SPSC in both RG and RPLG based networks are not obvious, in which a peer in RG (RPLG) forwards at most 8 (13) messages for 1000 queries in total. The reason is that the peers get to know more services when more queries come and such knowledge tremendously helps the peer to resolve a request locally. Despite in RPLG a small number of peers having a large number of connections, the major of queries are resolved by those backbone peers and not further forwarded. This is also evidenced by previous tests. Actually, after most peers know a considerably large number of services, the choice made during routing become less important, since the number of options increases. In addition, the message distribution over peers turns out to be flatter in RG case than in RPLG due to the network topology. We can also notice that, in RPLG, there are more peers that have forward 0 message than in RG. This is a natural consequence led by the existence of isolated peers in RPLG based topology, which in contrast is not the case in RG.

Query Response Time: Besides the cumulative recall and network traffic overhead, we test the average query response time (AQRT) per query of SPSC in RG and RPLG based networks. Similar to the average number of messages per query, the overall AQRT (cf. Figure 5.15) in both configurations decreases over the number of queries. The reason is that peer gets to know more services after observing a larger number of queries and such knowledge helps peers to resolve queries. The average query response time of correctly resolved queries ($AQRT_C$) decreases in both configurations, since the each peer knows more services when it observes more queries and the correct composition to a request in this kind can be achieved within less number of peers.

The AQRT of unresolved queries in RG based network is larger than the one in RPLG based network, since a query in the former configuration is routed to more peers and therefore to be considered with more services in total. Another reason is that the AQRT is mainly determined by the local composition process, instead of the IP based network communication, as we simplify the query



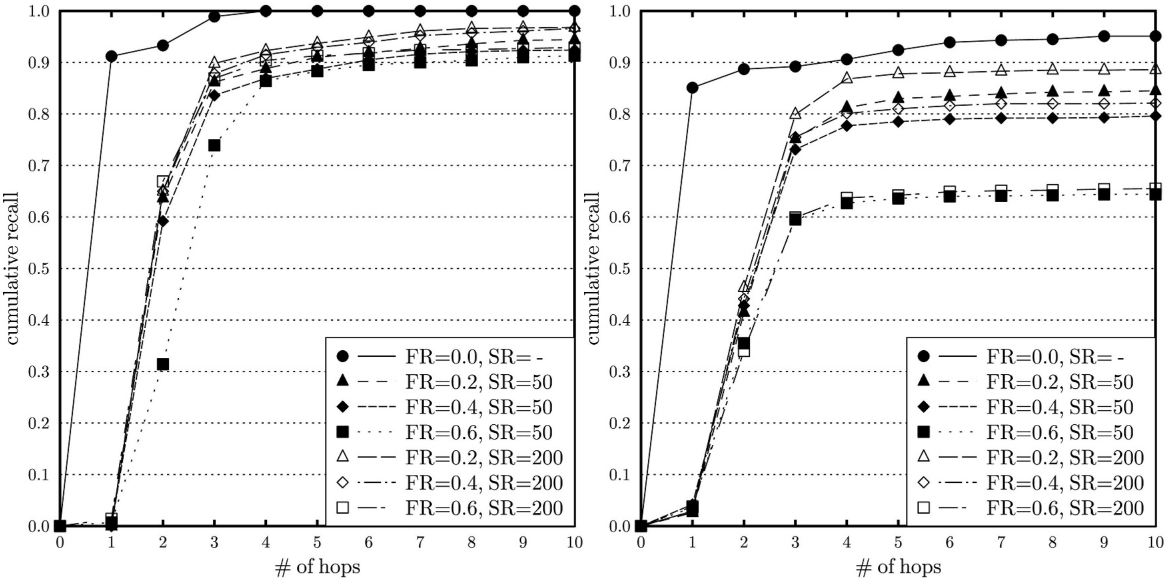
(a) RG (b) RPLG

Fig. 5.15 Average query response time of SPSC in RG and RPLG based networks with 1000 peers.

transmission by using a global data structure to store the current query. The AQRT of resolved queries are similar in the both configurations, as it is mainly determined by the services, rather than the number of traversed peers. This fact also results in that the SPSC with larger ω in RPLG based network needs more time to correctly resolve a query on average than the case with smaller setting on ω.

Robustness: We test the composition performance (average cumulative recall (CR)) of SPSC in RG and RPLG based networks with churn events. For generating network dynamics, we programmatically enforce network churn events (peer arrival and departure). An event happens immediately after every SR ($SR \in \mathbb{N}^+$, $SR < 1000$) queries. We name SR as stable period. A churn event means that a set of randomly selected peers leave and the departed peers in last stable period re-join into the network. We use peer failure rate FR ($FR \in [0,1]$) to represent the percentage of departure peers. In each configuration, two stable periods are concerned: $SR = 50$ or 200; and three peer failure rates are applied: $FR = 0.2$, 0.4 or 0.6. As an example, $FR = 0.2$, $SR = 50$ means that 20% randomly selected peers depart after every 50 queries and the departed peers in the last 50 queries re-join into the network at the same time. For each pair of FR and SR values, 1000 queries are executed and the network starts with all peers online. In both configurations, the default memorization rate ω is 1.

The CR values immediately after 1000 queries of the both configurations are shown in Figure 5.16, in which the results without network churn events ($FR = 0.0$, $SR = -$) are also illustrated as baselines. We observe that the composition performance in each configuration under network dynamics is worse than the case without dynamics. The reason is that network dynamics causes in the



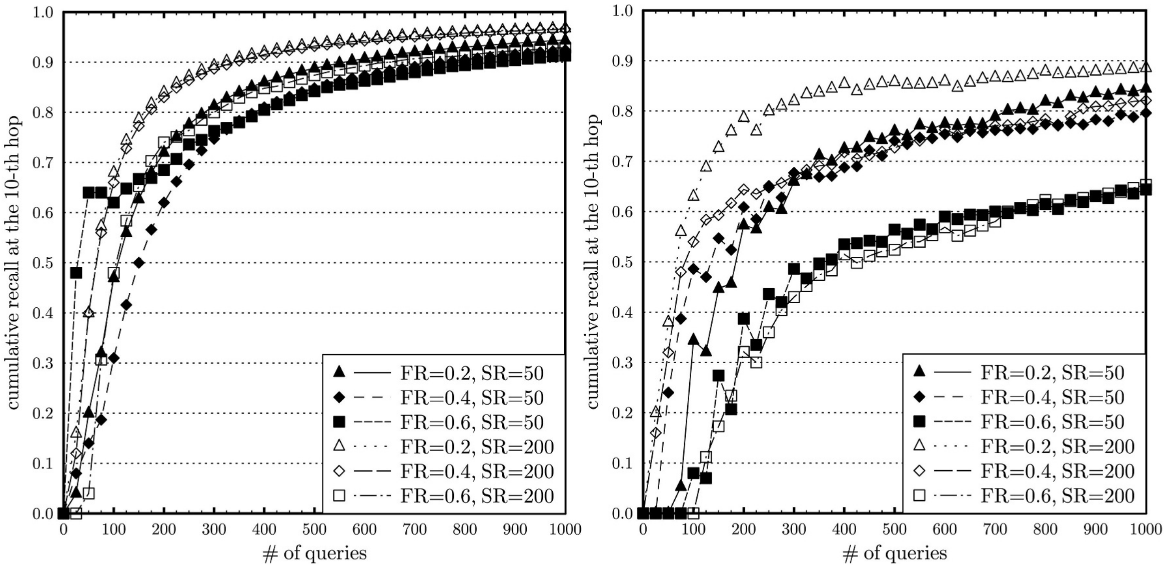
(a) RG (b) RPLG

Fig. 5.16 Cumulative recall of SPSC in RG and RPLG networks with network churn events.

loss of peers and further their referenced services. In addition, the cumulative recall value increases slower under network dynamics than the case without dynamics in each configuration. It reveals that queries might need to traverse more peers in order to be satisfied (correct workflow composed), as the departure of peers decreases the chance of query encountering those helpful peers that know about useful services. Further, the overall performance with RG is better than in RPLG. Since the selection of departing peers are totally random, backbone peers (those peers with large connectivity) in RPLG can leave. As discussed above, backbone peers have larger chance to observe more services and give more help to complete workflows. Once some of those peers left, the performance will be low. However, the situation within RG is different, as peers have almost the same connectivity degrees (10.295). Although some neighbors of a peer leave, the remained keep working.

When the stable period (e.g. $SR = 50$) is fixed, the performance with higher peer failure rate is worse. This phenomenon is more explicit in RPLG-based setting. The reason is that the chance of losing backbone peers gets higher, when the failure rate value is larger. In particular, the performance drops down more greatly when RF changes from 0.4 to 0.6 than from 0.2 to 0.4. This indicates the existence of a certain tolerance limit in terms of keeping relatively stable performance, given the loss backbone peers. This limit is determined by the topological structure of RPLG-based network, i.e. the skew value. In contrast, the RG-based configuration does not have this issue, as all peers have negligible difference on peer connectivity degree. When peer failure rate value is fixed (e.g. $FR = 0.2$), the performance with longer stable period is better. This holds in both configurations. The reason is that the semantic network overlay can get maturer if the stable period can last longer.

We check the recovery speed of SPSC system under network dynamics. For this, we test the

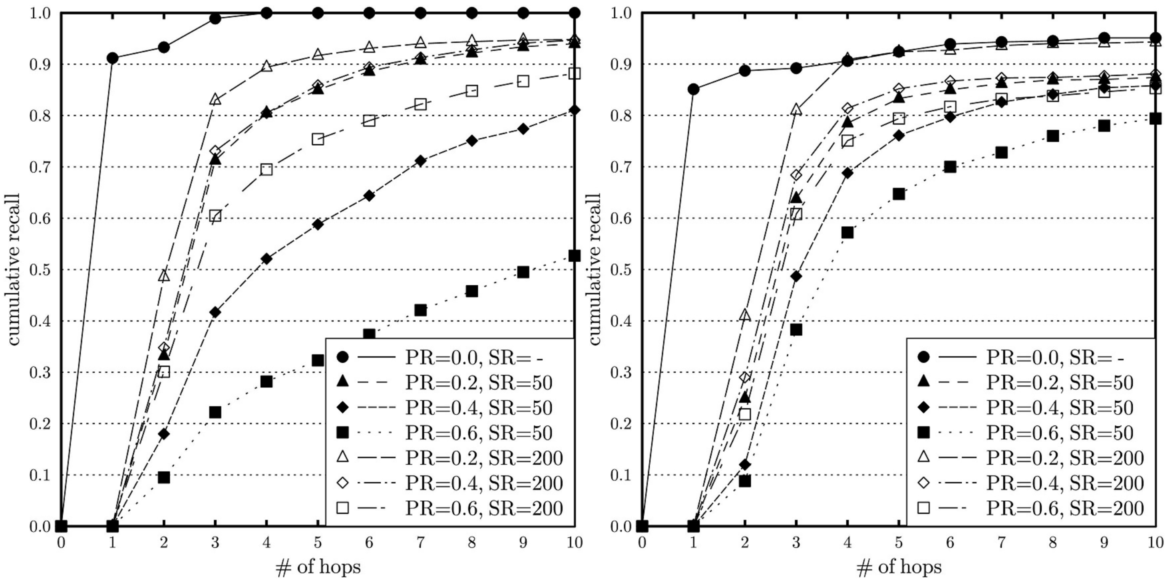


(a) RG (b) RPLG

Fig. 5.17 Cumulative recall at the 10-th hop of SPSC with network churn events.

system in RG- and RPLG-based networks with different combinations of stable period and peer failure rate values. As shown in Figure 5.17a, the average cumulative recall values at the 10-th hop in RG-based configuration are not sensitive to the network dynamics, since for most queries (> 80%), their correct solutions are composed within 3 hops (cf. Figure 5.16a). The reason is explained above. Even though a part of peers leave off, the rest keep connected and work. In RPLG-based configuration, the impact of network dynamics is more obvious. In Figure 5.17b, the cumulative recall value decreases or stops to increase after each churn event. However, SPSC system can also recover in a short period. This is enabled by the peer observation and memorization strategy. The latter greatly helps the service information propagation.

Another kind of dynamics is the addition, deletion and update of services. As we discussed in Section 5.2.5, the addition of service does not decrease the CR of SPSC, as long as its solution exists originally; while the modification of a service can be regarded as a sequence of deletion and addition operations. In the sequel, we focus on the test of SPSC against the deletion of services. For this, we assume that a set of peers periodically forget all of their knowledge about services. The amount of peers is modeled by a peer refreshing rate *PR* ($PR \in [0,1]$), which stands for the percentage of peers in the network that somehow lose all their knowledge about services. These peers are selected totally by random at the end of each period and their knowledge about services are deleted (called refresh in our context). The period length is again represented by number of queries. We call each period as a stable period *SR* ($SR \in \mathbb{N}^+$, $SR < 1000$), in which no dynamics happens and peers keep observing queries and contributing to queries. In the test about service dynamics, the default memorization rate ω is set to 1.



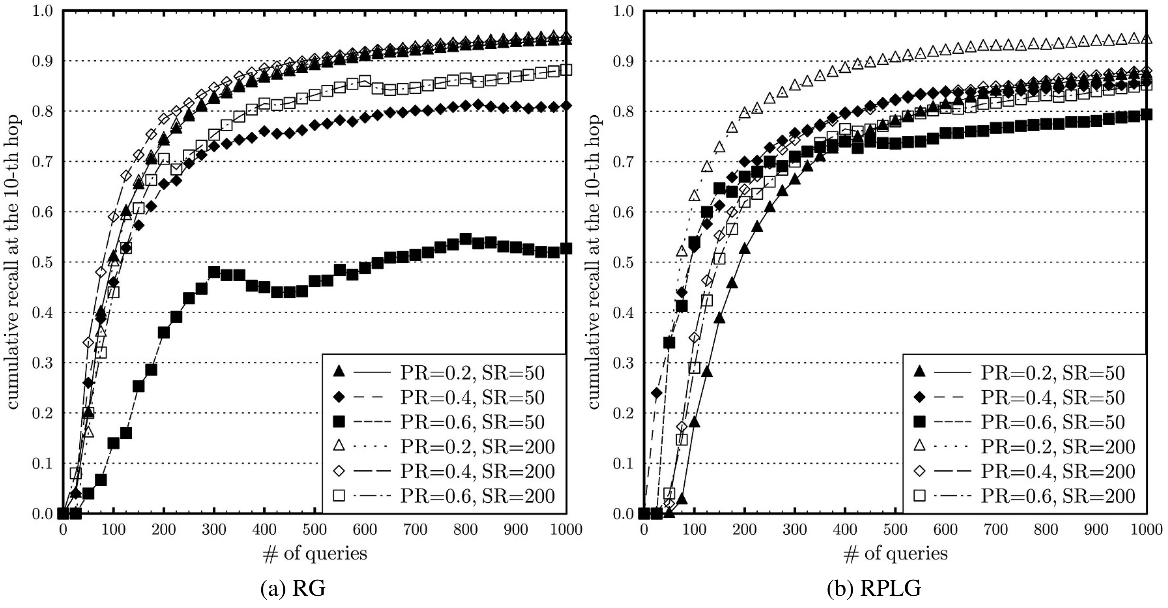
(a) RG

(b) RPLG

Fig. 5.18 Cumulative recall of SPSC with service signature dynamics.

The experiment results of SPSC against service signature dynamics are shown in Figure 5.18. Overall, the impact of service dynamics appears larger in RG-based network than the setting with RPLG. This is caused by network topology. In RPLG, backbone peers are minor but have more connections and chance to observe services, compared to the other peers having a few connections. Since the selection of peer for refreshing is totally by random, the chance of refreshing backbone peers is minor. RG-based network is different, in which connectivity degrees of peers in a RG are similar. It follows that each peer has similar chance of observing queries. This results in a flatter services knowledge distribution. When randomly choosing peers for refreshing, the chance of losing service knowledge overall can approximately be proportional to the refreshing rate. In addition, the run with longer stable period outperforms the ones with shorter stable period, when the peer refreshing rate value is fixed (e.g. $PR = 0.5$). Since the system performance largely depends on the peer knowledge. With longer stable period for observation, a peer is able to gather more knowledge for composition. When the stable period value is fixed, larger peer refreshing rate causes in greater decrement of performance.

The recovery speed of SPSC against service signature dynamics within RG- ane RPLG-based settings are tested. Shown in Figure 5.19, SPSC is able to recover its performance from service dynamics events overall, since SPSC peers are continuously observing and using service knowledge for composition. The cumulative recall at 10-th hop with RG-based network is more sensitive to the service dynamics than the runs with RPLG. This is caused by the network topology property. As explained above, peers in RPLG-based networks have less chance to loss knowledge overall, given a random selection for refreshing. This results in a relatively slighter loss of performance.



(a) RG (b) RPLG

Fig. 5.19 Cumulative recall at the 10-th hop of SPSC with service signature dynamics.

Nonetheless, RG-based networks do not have this property. When peer refreshing rate is fixed, larger stable period promises better performance. This holds in both configurations. It is natural that a more frequently demanded semantic overlay will give less support for composition and therefore yield worst performance.

5.7 Related Work

In this section, we discuss the recent related work on semantic service composition in unstructured P2P networks. There exist quite a number of approaches for centralized or (semi-)structured P2P-based semantic service composition. For a discussion and classification of these, we refer the reader to the Sections 2.1.2 and 2.3.3, respectively.

To the best of our knowledge, quite a few work addresses the service composition problem with unstructured P2P network-based dynamic settings. The PM4SWS system [117] [116] performs IO-level semantic service composition by forward or backward chaining. Each peer maintains a composition lookup table that records its observed historic compositions. Given a query, if no correct solution exists in the table, a peer tries to find a known service that can be chained to the current composition. This work leverages the classic flooding protocol, which causes in immense network traffic overhead. SPSC does not rely on flooding but path suggestion based query routing strategy with query branching mechanism, which allows a peer to issue new sub-queries for only new partial solutions.

Furno et.al. present a probabilistic flooding-based method SCComp [109] [110] [108] for OWL-

S service composition. In SCComp, each peer routes a query to a set of selected neighbors, which are determined to be helpful to resolve the current sub-goal. Once a sub-goal has been resolved, extra transitive messages are needed for re-constructing the overall solution. Despite its selective routing strategy, the network traffic of this approach can still be heavy, as multiple peers may send duplicate messages to the same peer for the same sub-goal. SPSC alleviates this problem by means of a walker-based routing strategy with guided pruning on the search space and memorization mechanism. Moreover, SCComp needs extra message exchange to adapt the network dynamics. In contrast, no message is need by SPSC peer for the same purpose. Further, no factor about QoS or plan length has been considered by SCComp. SPSC takes them into account and supports an approximately optimal path suggestion-based query routing strategy.

Hu et. al. propose an approach, called PSComp [152], for stateless distributed IO-level composition of WSDL services. The environment setting of PSComp is a broker network without any initial data overlay or coordination. PSComp models the service requester as a service agent; the service provider as a request agent. Agents in both kinds can connect to arbitrary nodes in the broker network. When connecting to the network, a service agent advertises its services via flooding such that some of the other service agents can get to know. If a user want to request the network for a desired web service, the request agent takes this response and issues a query (called publication) to the network. This query is flooded to those service agents providing compatible (chain-able) services. On receiving a request, each service agent then attempts to contribute to the current partial workflow by exploring the possible forward chaining based on its known services. Once the workflow is augmented, a new query is sent to all the other service agents again. This process is iteratively performed until the solution is found. With the exhaustive forward chaining, the performance of PSComp is subject to the size of search space, since no heuristics is applied for pruning. In addition, the exhaustive and arbitrary chaining may lead to the failures with dead-ends.

Similar with PSComp, the DPAWSC [358] system offers stateless distributed IO-level composition of WSDL services. DPAWSC assumes that the composition is performed by a group of interconnected BDI (Belief–desire–intention) agents. Each agent knows a set of web services and their features are summarized as agent belief, which is formulated as a concept. The composition with DPAWSC is performed via flooding. When receiving a partial plan, each agent exhaustively searches its local service repository and tries to chain services to plan, as long as compatible IO parameters are found. This process is recursively conducted until the solution is found or all agents have been detected. Besides the risk of failure with dead-ends, DPAWSC works with a large unpruned search space, which yields a large number of unnecessary computation and network traffics.

MPComp [11] is proposed by Al-Oqily et. al. for IO-level stateless composition of web services. Compared to PSComp and DPAWSC, MPComp derives a selective flooding routing strategy. Such routing decision is captured based on the peer observation on queries, result feedbacks and user customized routing destinations. For this, each peer maintains a list of peers, which are estimated to be promising in terms of completing partial plans. Services are composed with MPComp in a forward chaining fashion. Without a guarding heuristics, each peer attempts to chain all local IO-compatible to the current plan. This would yield arbitrary augmentation of plan and therefore the failure due to dead-ends. SPSC overcomes the limitations of DPAWSC, PSComp and MPComp by the query branching mechanism and guarding heuristics.

Aguilera et. al. propose a method, DSGComp [6], for IO-level stateless composition of WSDL services in ad hoc MANET. In DSGComp, each peer can maintain a part of service dependency graph, which are created when peers or services are add / deleted. When an update happens, peer broadcasts this information to the neighbors reachable in a small TTL. When observing this, each peer updates its local dependency graph. Given a service composition request, a peer explores its local graph and try to find multi-paths connecting the requested input and output. If failed, it sends the partial plan to a set of peers which services has been used in the plan. Unlike SPSC, this approach can not handle dead-ends and tends to conduct arbitrary chaining without heuristics. It does not guarantee the completeness and correctness.

The AntComp method, in [102], [101] and [420], enables the ant-inspired agent-based semantic service discovery and composition. Each peer maintains a co-use matrix that contains the pairs of classified reusable services observed in historical plans. In particular, a composition task is assumed to be pre-configured as a set of key classes in a query design phase. On processing a query, each peer selects the closest service classes and sends the query to a known peer which might be able to satisfy other keys. Besides the lack of a completeness guarantee, AntComp needs the extra design phase. This hinders the full automation of composition. A number of related work, such as [58] [111] [241], etc.., focus on the similar semi-automatic composition in distributed environments. Differing with the automatic composition concerned in this thesis, these approaches allow a user to put in a request with an abstract plan template. The composition target of them is actually the selection of services in order to create a solid plan.

5.8 Summary

In this chapter, we propose an approach, called SPSC, for distributed semantic service composition in unstructured P2P networks. The main contribution lies in the bidirectional service composition heuristics with memorization strategy. In addition, we theoretically analyzed the lower bound of SPSC completeness and proved its soundness. The experimental evaluations reveal that SPSC can achieve high cumulative recall with low traffic cost and query response time.

In the next chapter, we will present an approach, called iRep3D, for the semantic indexing and retrieval of 3D scenes, which is concerned with an optional strategy for efficient peer local item selection. The value of iRep3D lies in two aspects: (1) it makes it possible to efficiently retrieve semantically annotated XML-based 3D scenes with hybrid complex query in terms of conceptual, functional and geometric features; and (2) it supports a concept abduction based approximated similarity measure that alleviates the risk of mis-categorization due to strict logic reasoning and at the same time provides high accuracy.

Chapter 6

iRep3D: Fast Hybrid Semantic Selection of 3D Scenes for P2P Search

In recent years, a large number of 3D scenes in XML-based 3D graphics description languages, such as X3D[1], XML3D[2] and COLLADA[3], have widely emerged on the Web. The current 3D scene retrieval methodologies not only leverage the 3D geometric feature matching, but also scale to the comparisons between the syntactic, RDF, logical concept based scene descriptions. In this context, the ISReal platform [181] made a step forward in the integration of 3D graphics with the Semantic Web and a multi-agent system for intelligent 3D simulation of realities on the 3D Internet (3DI), which allows users to formally describe 3D scenes with respect to their interactive and behavioral functions via semantic services (such as OWL-S). The latter is well known to be a fundamental building block of simulated agent based collaboration and interaction in 3DI. When realized on top of P2P networks, such a 3DI-based virtual collaborative system will benefit from their distributed robustness and avoidance of system crashes due to central server failures. The BMB+F project Collaborate3D[4] aims at the research and development of scalable intelligent support for distributed collaboration in the future 3D Internet. In this context, efficient semantic 3D scene selection is crucial for its underlying P2P-based search infrastructure, where each peer is supposed to quickly react to real-time complex queries with conceptual description, scene services and geometric features.

As a key part of P2P-based IR systems, item selection is performed by each peer when the latter is queried. A peer could simply check whether each of its maintained items matches the query or not. Although it might work well in those scenarios where each peer is maintaining only a small number of items, selection in this way still carries the risk of requiring a large query response time if a peer maintains lots of items. Besides, the query response time is also subject to the computational complexity that might be entailed by the item-query comparison. This particularly matters in the

---

[1] `http://www.web3d.org/x3d/`

[2] `http://www.xml3d.org/`

[3] `https://collada.org/`

[4] `http://c3d.dfki.de/`

context of P2P-based semantic 3D scene retrieval, where the possible logic-based reasoning during item selection would result in a large computational overhead and further lead to an unexpected query response time.

To provide sufficient efficiency and preserve high accuracy for the P2P search of annotated 3D scenes, the classic logical relation determination [265] and semantic service matchmakers in [192] are not suitable for use here directly, because the overall computational complexity of local selection tends to closely depend on the number of items. Previous efforts on non-logic-based 3D scene retrieval approaches [112] [124] [214] [150] [396] [205] [294] would lower accuracy, as the syntactic keyword ambiguity can lead to the mis-categorization or mis-ranking of results. RDF [209] [15] and logic-based [176] [147] [148] [288] [399] works alleviate this problem but, unfortunately, still have the risk of misclassification as they apply strict reasoning and SPARQL[5] graph pattern matching. In addition, none of the methods above handles complex queries with conceptual, functional and geometric feature aspects.

To this end, we propose iRep3D for the efficient selection of semantically annotated 3D scenes, to help support P2P-based intelligent collaboration in the future 3DI. The key idea is to perform off-line semantics-preserving indexing of 3D scenes in terms of their conceptual descriptions, semantic services and geometric features. Given a query, three index-based sub-query processing steps are efficiently performed fully in parallel, which yield three ranked lists of 3D scenes relevant to the request with respect to these aspects. Finally, a total ranking is conducted by applying Fagin's threshold algorithm (TA) [92] on the sub-rankings. In particular, iRep3D supports an index maintenance strategy that responds to the 3D scene annotation updates (including insertion, deletion and modification). This reduces the risk of the mis-categorization that would result from strict logical reasoning, and it enables the system to quickly adapt to the dynamics of environments. Besides, for the reuse of 3D scenes, iRep3D scales to semantic deep search and automatic sub-scene extraction, by which a user is able to retrieve relevant parts of a 3D scene. Towards the goal of distributed collaboration on the 3DI, the iRep3D approach has been implemented as a 3D scene repository for the efficient selection of 3D scenes in the Collborate3D system (cf. Chapter 7). As for its retrieval performance, the comparative experimental evaluations revealed that the iRep3D repository outperforms the logic-based functional and behavioral 3D scene retrieval approach FB3D [41] in terms of search precision, recall and averaged query response time. Moreover, iRep3D also achieves better search precision than the RDF-based approach RIR [15] and the open-source in-use 3D repository ADL[6].

This chapter is organized as follows: In Section 6.1, we introduce the underlying notions, definitions and assumptions of the iRep3D approach, an overview of which is presented in Section 6.2, and which is detailed in Sections 6.3 and 6.4. This includes the 3-dimensional indexing of 3D scenes and the efficient on-line hybrid query answering, as well as the incremental indexing and the sub-

---

[5] `http://www.w3.org/TR/rdf-sparql-query/`

[6] `http://3dr.adlnet.gov/Default.aspx`

scene extension of 3D scenes. The discussion of computational complexities of the key components is presented in Section 6.5. The main evaluation results are presented in Section 6.6. We discuss the related work in Section 6.7 and conclude this chapter in Section 6.8.

6.1 Preliminaries

As a common ground of understanding our XML-based semantic 3D scene retrieval iRep3D, we define the basic notions and assumptions in this section.

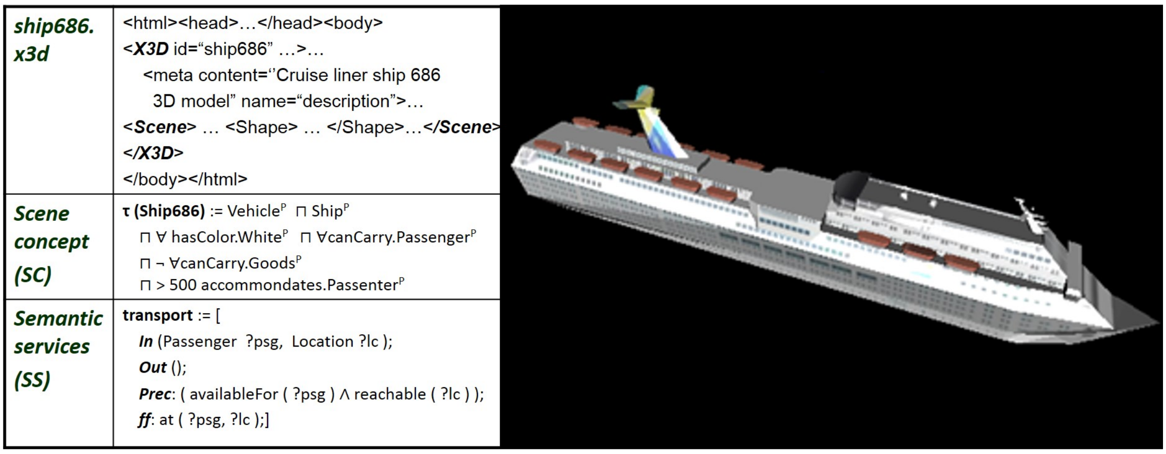
6.1.1 Semantically Annotated 3D Scenes

Semantic annotation of 3D scene is supposed to support the machine-understandable description of their semantics. Logic-based semantic annotation of scene with references to concepts which semantics are defined in a logic-based ontology language like standard OWL-DL or RDFS that are common sense in the semantic web. A successful instance of semantic annotation on XML3D based 3D scene is supported by the ISReal platform [181], which enables the RDFa-based conceptual and semantic service annotation of scenes. In the context of this thesis, the semantically annotated 3D scene is defined as follows:

Definition 1: *Semantically annotated 3D scene x*
A 3D scene x is defined by the scene tuple: $x = [id,\ \tau(C,O),\ SS,\ GF,\ da]$ where ID denotes the UUID of x; $\tau(C,O)$ the logical expression (object abstraction) of the *scene concept* C under the scene concept ontology O in OWL DL. Name this conceptual expression as *scene concept* (SC); SS the set of semantic services (cf. Definition 2) in OWL-S, which are provided by x; GF the set of geometric features (cf. Definition 3) x has; da refers to the data of scene x including the XML-based description file and its referenced resources like images, sounds, etc.. Denote $\mathscr{X}$ the space of x. ■

The complete logical expression of scene concept in the given ontology O is denoted as $\tau(C,O)$ ($\tau(C)$ in short) and contains only logical operators (conjunction $\sqcap$, negation $\neg$) and quantifiers (universal $\forall$, exists $\exists$) over a set of primitive terms ($\cdot^P$). A primitive term can be either a primitive concept indicating the basic type of C, a primitive role term which defines the relation of C with another concept, or a role filler cardinality restriction with primitive concepts C^P. For example, as illustrated in Figure 6.1, the logical expression of scene concept *Ship686* contains the primitive concepts $Vehicle^P$ and $Ship^P$ as well as quantified and cardinality restricted (primitive) roles $\forall hasColor.\ White^P$, $\forall canCarry.\ Passenger^P$, $\neg\forall canCarry.\ Goods^P$ and $> 500\ accommondates.\ Passenger^P$, respectively.

In addition to the static representation of object semantics with formally defined ontological concepts, iRep3D also allows the functional description of object behaviours with appropriate semantic services. Each of these services are supposed to be grounded in executable service programs such as 3D animation scripts for respective functionalities, like the opening or closing of a door. For this purpose, we assume that these semantic services are described by profiles in OWL-S[7] in terms of their input (I) and output (O) parameters as well as preconditions (E) and effects (E). These profiles are referenced by 3D scene description file. The type of any IO parameter is a concept defined in OWL-DL while the precondition and effect are specified in conjunctive normal form over 2-valued predicates over variables and individuals.



ship686. x3d

```
<html><head>…</head><body>
<X3D id="ship686" …>…
   <meta content="Cruise liner ship 686
     3D model" name="description">…
<Scene> … <Shape> … </Shape>…</Scene>
</X3D>
</body></html>
```

Scene concept (SC)

```
τ (Ship686) := Vehicle^P ⊓ Ship^P
   ⊓ ∀ hasColor.White^P  ⊓ ∀canCarry.Passenger^P
   ⊓ ¬ ∀canCarry.Goods^P
   ⊓ > 500 accommondates.Passenter^P
```

Semantic services (SS)

```
transport := [
   In (Passenger ?psg, Location ?lc );
   Out ();
   Prec: ( availableFor ( ?psg ) ∧ reachable ( ?lc ) );
   ff: at ( ?psg, ?lc );]
```

Fig. 6.1 Semantically annotated 3D scene: ship686

Definition 2: *Semantic service of a 3D scene x*

A semantic service *ss* of a semantically annotated scene *x* is defined by the service tuple: $ss = [URI, In, Out, Prec, Eff]$ where *URI* denotes the URI of service description file; *In* the set of input parameters; *Out* the set of output parameters; *Prec* the logic expression of the precondition of *ss*; Eff the logical expression of the effect of *ss*. All concepts in *In* and *Out* (predicates used in *Prec* and Eff) are defined in a service annotation ontology O_{sp} (predicate set A_{sp}) of scene publisher. Denote $ss.In$, $ss.Out$, $ss.Prec$ and $ss.Eff$ as $ss[i]$, $ss[o]$, $ss[p]$ and $ss[e]$, respectively. ■

For example, the functionality of the ship model in Figure 6.1 is partly described in the profile of the semantic service *transport* with the concepts *Passenger*, *Location* of the service (program) input variables $?pg, ?lc$, which semantics are defined in a referenced OWL ontology. In addition, the service precondition requires that the ship should be available for the *Passenger* $?psg$ and the *Location* $?lc$ should be reachable, while the effect $at(?psg, ?lc)$ of executing this transport service means that the *Passenger* will be eventually at the given *Location*. There is a variety of tools for the selection of semantic services for a given service request available [193] like the currently most precise service matchmaker iSeM [200].

Commonly, parameter concepts and Prec/ Eff predicates of different services are defined by various ontologies and predicate sets of service publishers. This could make difficulty when comparing different services. For simplicity, let us assume that the 3D scene repository supports an ontology

---

[7] `http://www.w3.org/Submission/OWL-S/`

O_s that is the semantically merged version of all possible O_{sp}. Moreover, denote with A_s the merged version of the predicate sets of those publishers. Each geometric feature gf of a scene x is an instance of a feature type f defined in X3D, XML3D and COLLADA specifications. Denote all the types as an universal feature type space $\mathscr{F}$.

Definition 3: *Geometric feature of a 3D scene x*
A geometric feature gf of a 3D scene x is defined by the feature tuple: $gf = [name,\ f\ \{(k,v)\}]$ where *name* denotes the individual name of gf in the context of x; $f \in \mathscr{F}$ the feature type of gf; $\{(k,v)\}$ the set of pairs where each pair associates an attribute signature k to its corresponding value v whose data structure is defined by X3D, XML3D or COLLADA specifications. Denote K_f the set of attributes defined in the feature type f; $v(k)$ the instantiated value of attribute k in x. ■

In our context, we assume that both the 3D scene requesting and providing sides share the same vocabulary V of primitive terms, which works as the common sense in the distributed semantic scenarios. On top on this, the requesters and 3D scene provider can define their individual ontologies for semantically describing their desired and offered scenes. iRep3D allows user to ask for 3D scenes with semantic-based hybrid queries. Besides specifying the geometric features, a query can contain the conceptual as well as the functional descriptions that the desired 3D scenes are supposed to meet.

Definition 4: *3D scene query*
A query q for 3D scenes is defined by the query tuple: $q = [\tau(C, O_{req}),\ SS,\ GF,\ m]$ where $\tau(C, O_{req})$ is the logical expression of the requested scene concept C based on the ontology on the requester client *req*; *SS* the set of semantic services that the desired scene should provide; *GF* the set of geometric feature instances that the desired scene should have; m ($m > 0, m \in \mathbb{N}$) the number of the most relevant scenes needed by *req*. ■

6.1.2 Approximated Semantic Subsumption Measure

A semantic query for 3D scenes is efficiently processed, which is done by the parallel sub-query processing in each dimension. To achieve this, off-line indexing of each 3D scene x is performed in advance. With it, x is indexed into 3 sub-indices separately in 3 dimensions: scene concept, semantic service and geometric feature. In each dimension, x is indexed into a certain ranking R with a score. The latter indicates that, to what extent, x can be categorized to the sub-class corresponding to R. The semantic 3D scene indexing and query processing will be detailed in Section 6.3. In this section, we present the approximated semantic subsumption measure, which is essential to the indexing and query answering in the concept and service dimensions. Let C and C' be two concepts. In comparison with the strict logic subsumption determination, instead of giving binary answers, the similarity measure $s_{sb,\sqsubseteq}(C, C') \in [0,1]$ checks, to what extent, the concept C can be subsumed by C' under the concept abduction of C with respect to C'.

For this purpose, the first step is to find, by concept contraction [268], the incompatible G, missed

M and compatible K parts , of C ($C = G \sqcap K$) with respect to C'. Subsequently, G, M and K are used by concept abduction that computes an approximated concept C_{app} fully subsumed by C'. Denote C^P a primitive term in C, which conflicts with a primitive term $\bar{C}^P$ (named as the counter-part of C^P) in C'; $|C|$ the number of conjunctive primitive terms in C; $PC(C)$ the set of primitive concepts of C; $PR(C)$ the set of primitive roles of C; $PRE(C)$ the set of primitive numeric restrictions of C. The score value $s_{ab,\sqsubseteq}(C,C')$ can be calculated as follows:

$$\begin{aligned}
s_{ab,\sqsubseteq}(C,C') &= \frac{|K|}{|C'|} \cdot (1 - s_{acf}(C,C'));\\
s_{acf}(C,C') &= \frac{1}{|C|} \sum\nolimits_{C^P \text{ in } G \sqcap M} \left(s_{cf}(C^P,C') \cdot w(C^P,C') \right);\\
s_{cf}(C^P,C') &= 1, \quad \text{if } C^P \text{ in } M \text{ or } PC(C) \cup PR(C);\\
s_{cf}(C^P,C') &= \frac{rg(C^P) \backslash rg(\bar{C}^P)}{rg(C^P)}, \text{if } C^P \in PRE(C);\\
w(C^P,C') &= \frac{impt(\bar{C}^P,C')}{\sum_{C'^P \text{ in } G \sqcap M} impt(\bar{C}^P,C')}.
\end{aligned} \tag{6.1}$$

where $\frac{|K|}{|C'|}$ is the fraction of compatible part of C with respect to C'; s_{acf} the average strength of conflicts between C and C'; $s_{cf}(C^P,C')$ the strength of atomic conflict about C^P in C with respect to C', which can be further computed as follows: When C^P is in M, it will surely appear in the approximated concept C_{app} of C; while in case that C^P is a primitive concept or role in G, any conflict on C^P raises entire rewriting of C^P during concept abduction; if C^P is a primitive numeric restriction in G, the conflict strength is the fraction of uncovered range of C^P with respect to its counter-part $\bar{C}^P$. The function $rg(C^P)$ computes the restricted numeric range of $C^P \in PRE(C)$. Each atomic conflict strength $s_{cf}(C^P,C')$ is weighted by $w(C^P,C')$ ($\sum_{C^P \text{ in } G \sqcap M} w(C^P,C') = 1$, $w(C^P,C') > 0$). It is determined by estimating the importance of the conflict on C^P with respect to C'. Denote C'_{lp} the least parent concept of C' in its ontology; C'' the mutated concept of C', which is generated by replacing $\bar{C}^P$ with C^P, if C^P is in G, or removing C^P from C', if C^P is in M. The binary function $impt(\bar{C}^P,C') \in \{a,b\}$ ($0 < a < b \leq 1$) determines the importance of $\bar{C}^P$ in terms of keeping the hierarchy of C' in its ontology. It returns b, if $C'' \sqsubseteq C'_{lp}$ is false; a, otherwise. Intuitively, if the replacement of $\bar{C}^P$ (in C') with C^P or the removal of C^P makes C'' to be no longer a subclass of C'_{lp}, the conflict on C^P then has greater negative impact of C being subsumed by C'.

For instance, let $\tau(CPS,O) := Vehicle^P \sqcap Ship^P \sqcap \forall\ canCarry.\ Passenger^P \sqcap \forall\ canCarry.\ Goods^P \sqcap > 300\ accommondates.\ Passenger^P \sqcap > 2000\ accommondates.\ Goods^P \sqcap \forall\ hasEquip.\ AirConditioner^P$. Obviously, $Ship686$ is not strictly subsumed by CPS due to the incompatible part $G = \neg\forall\ canCarry.\ Goods^P \sqcap > 2000\ accommondates.\ Goods^P \sqcap \forall\ hasEquip.\ AirConditioner^P$, despite the compatible part $K = Vehicle^P \sqcap Ship^P \sqcap \forall\ canCarry.\ Passenger^P$ with respect to CPS. For the sake of clearer presentation, denote M ($M \subseteq G$) the missed part of $Ship686$ with respect to CPS: $M => 2000\ accommondates.\ Goods^P \sqcap \forall\ hasEquip.\ AirConditioner^P$.

With strict logical subsumption, the concept $Ship686$ can not be categorized to the class CPS. However, they still have a considerable compatible part, and the minor conflict of them might be acceptable to users. Therefore, it is worth to check, to what extent, the ship in the example can be categorized to CPS, though not perfectly. For this purpose, the approximated similarity measure works as follows: It first computes an abduced (rewritten) concept $Ship686_{app}$ that is fully subsumed by CPS and subsequently calculates a score value indicating to what extent $Ship686$ can be subsumed by CPS by means of the (in-)compatible parts of $Ship686$ with respect to CPS. Let $Ship$ the direct parent concept of CPS with $\tau(Ship) = Vehicle^P \sqcap Ship^P$. The determined (in-)compatible parts of $Ship686$ with respect to CPS leads to the respective conflict strengths: $s_{cf}(\neg\forall carCarry.\ Goods^P, CPS) = 1$, $s_{cf}(> 2000\ accommondates.\ Goods^P, CPS) = 1$ and $s_{cf}(\forall hasEquip.\ AirConditioner^P, CPS) = 1$.

Let $impt(\cdot,\cdot) \in \{0.1, 0.9\}$. If replacing $\forall carCarry.\ Goods^P$ in $\tau(CPS)$ with $\neg\ \forall\ carCarry.\ Goods^P$, then the mutated concept CPS' is still subsumed by $Ship$. This implies that the importance of this conflict is minor: $impt(\forall carCarry.\ Goods^P, CPS) = 0.1$. The same holds for the other two missed components in M. Therefore: $w(\forall carCarry.\ Goods^P, CPS) = w(> 2000 accommondates.\ Goods^P, CPS) = w(\forall hasEquip.\ AirConditioner^P, CPS)\ \frac{0.1}{0.1+0.1+0.1} = 0.33$. Subsequently, the averaged conflict strength is $s_{acf} = \frac{1}{|Ship686|} \cdot \sum_{C^P \text{ in } G} (s_{cf}(C^P, CPS) \cdot w(C^P, CPS)) = \frac{1}{6}(1 \cdot 0.33 + 1 \cdot 0.33 + 1 \cdot 0.33) = 0.17$. Finally, $s_{ab,\sqsubseteq}(Ship686, CPS) = \frac{|K(Ship686, CPS)|}{|CPS|} \cdot (1 - s_{acf}(Ship686, CPS)) = \frac{3}{7} \cdot (1 - 0.17) = 0.358$.

6.2 Overview

iRep3D approach for efficient XML-based semantic 3D scene selection contains two folds: off-line index construction and on-line query processing. In the first step, every 3D scene x annotated with scene concept C and a set SS of semantic services is indexed in scene concept index I_{SC}, semantic service index I_{SS} and geometric feature index I_{GF}, separately.

- Scene concept index I_{SC} is formed by a set of ranked lists. Each list $R(C')$ ($C' \in O$) corresponds to a concept $C' \in O$ (cf. Definition 1). It contains a list of pairs $\{(x.id, d(x, C'))\}$ ($d(x, C') \in [0, 1]$) for all $x \in \mathscr{X}$ in the descending order of $d(x, C')$. The value of $d(x, C')$ is computed by means of applying concept abduction based approximated subsumption $s_{ab,\sqsubseteq}(C, C')$.

- Scene index I_{SS} in terms of semantic service consists of two subindexes: the IO concept subindex I_{IO} and the PE subindex I_{PE}. They are built based on the sets of IO parameter concepts in O_s, respectively, PE predicates defined in A_s.

  Differing with the scene concept index, the IO subindex contains two ranked lists for each concept $C_s \in O_s$: $R(C_s)[i] \in I_{IO}$ and $R(C_s)[o] \in I_{IO}$. They are additionally labeled with suffix $[i]$, respectively, $[o]$. Scene x is ranked in $R(C_s)[i]$ ($R(C_s)[o]$), if a service input (output) parameter C'_s of $ss \in x.SS$ is sufficiently semantically similar with the concept $C_s \in O_s$: $s_{ab,\sqsubseteq}(C'_s, C_s) \geq \theta$, where θ ($\theta \in \mathbb{R}, \theta \in [0,1]$) is a threshold value. The rank of x in $R(C_s)[i]$ is determined by the score $d_s(x,C_s)[i]$ ($d(x,C_s)[o]$), which is estimated based on the approximated subsumption similarity $s_{ab,\sqsubseteq}(C'_s,C_s)$ as well as the occurrence frequency of C'_s in the services $x.SS$.

Likewise, a predicate $\alpha \in A_s$ corresponds to two ranked lists $R(\alpha)[p]$ ($R(\alpha)[e]$) in PE subindex I_{PE}. Scene x is ranked in $R(\alpha)[p]$ ($R(\alpha)[e]$), if the non-negative form of α appears in the precondition (effect) of a service $ss \in x.SS$. The score $d_a(x,\alpha)[p]$ ($d_a(x,\alpha)[e]$) of x in $R(\alpha)[p]$ ($R(\alpha)[e]$) is the plausibility value of α in the precondition (effect) of a service $ss \in x.SS$.

- The geometric feature index I_{GF} comprises a set of B+trees. Each B+tree $bt(f,k)$ records those scenes that instantiate the attribute k of feature type $f \in \mathscr{F}$. To construct $bt(f,k)$, all scenes x containing value on k are firstly sorted in a ranked list $R(f,k)$ in the descending order of $v(k)$. Based on $R(f,k)$, $bt(f,k)$ is then generated according to the pre-defined maximum fanout and the size limitation of nodes. Each leaf node of $bt(f,k)$ points to a set of rank lists of $R(f,k)$ and no lists contain the same entry pair.

Based on the indices constructed off-line, a query q for desired scenes can be processed efficiently. To do this, three subqueries in terms of scene concept, semantic service and geometric feature are processed in parallel based on the corresponding index above. This yields three sub-rankings of scenes $R_{sc}(q)$, $R_{ss}(q)$ and $R_{gf}(q)$, respectively. The final ranking of scene relevant to q is generated by executing TA process on the sub-rankings.

6.3 Efficient Semantic 3D Scene Retrieval

6.3.1 Semantic Indexing

The first step of efficient semantic 3D scene selection in our context is to construct indices of scenes in terms of scene concept, semantic service and geometric feature. This process is conducted totally off-line.

Scene concept index construction: I_{SC} comprises a set of ranked lists. A list $R(C') \in I_{SC}$ about a concept $C' \in O$ (cf. Definition 1) contains a list of pairs $\{(x.id, d(x,C'))\}$ in the descending order of the similarity score $d(x,C')$ between the scene concept C of each x and C'. This value is computed by concept abduction based approximated subsumption above that checks to what extent that x can be categorized to C': $d(x,C') = s_{ab,\sqsubseteq}(x.C,C')$. To index a scene x is to compute the relevancy score $d(x,C')$ between x and all $C' \in O$ and insert the pair $\{x.id, d(x,C')\}$ into the corresponding list $R(C')$.

Semantic service index construction: Each semantic service $ss \in x.SS$ provided by a scene x can have IO parameter sets, Prec and Eff. According to iRep3D, x will be indexed in IO and PE subindexes, respectively.

IO subindex I_{IO} construction. It is possible that a concept $C_s \in O_s$ is semantically similar with an input or output parameter C'_s of $ss \in x.SS$. To distinguish the parameter directions, a concept $C_s \in O_s$ corresponds to two ranked lists $R(C_s)[i]$ and $R(C_s)[o]$. The suffix $[i]$ ($[o]$) indicates that the ranked 3D scene have a service input (output) parameter concept C'_s that is sufficiently semantically similar with concept $C_s \in O_s$: $s_{ab,\sqsubseteq}(C'_s, C_s) \geq \theta$ ($\theta \in [0,1], \theta \in \mathbb{R}$). Please note that the suffix only declares the parameter direction, rather than affects the concept definition. Let $l \in \{i, o\}$, similar with the scene concept case, $R(C_s)[l]$ is a list of pairs $\{(x.id, d_s(x, C_s)[l])\}$ in the descending order of $d_s(x, C_s)[l]$. Each entry associates a scene x with a relevance score $d_s(x, C_s)[i]$ between x and $C_s \in O_s$. It is computed as follows:

$$
\begin{array}{ll}
d_s(x,C_s)[l] & = max_{\substack{C'_s \in ss[l] \\ ss \in x.SS}} d_c(C'_s,C_s)[l]; \\
d_c(C'_s,C_s)[l] & = fr(C'_s)[l] \cdot s_{ab,\sqsubseteq}(C'_s,C_s); \\
fr(C'_s)[l] & = \dfrac{|x.SS_{C'_s}[l]|}{|x.SS|} \cdot max_{ss \in x.SS_{C'_s}} \dfrac{n(C'_s, ss[l])}{|ss|}; \\
\text{subject to:} & s_{ab,\sqsubseteq}(C'_s,C_s) \geq \theta, (\theta \in [0,1], \theta \in \mathbb{R}).
\end{array}
\tag{6.2}
$$

where $x.SS_{C'_s}[l] \subseteq x.SS$ is a subset of $x.SS$. Each uses C'_s as an input ($l = i$) or output ($l = o$) parameter type; $n(C'_s, ss[l])$ the appearance frequency of C'_s in the input ($l = i$) or output ($l = o$) parameter set; $|ss|$ the total number of parameters of ss. $d_c(C'_s, C_s)[l]$ refers to the concept-level similarity between C'_s and $C_s \in O_s$. It is further estimated by the appearance frequency ($fr(C'_s)[l]$) of C'_s in $x.SS$ and the semantic similarity $s_{ab,\sqsubseteq}(C'_s, C_s)$.

When building I_{IO}, the relevancy score $d_s(x.C_s)[l]$ will be computed for each scene $x \in \mathscr{X}$, by considering each parameter concept C'_s of each $ss \in x.SS$ with any concept $C_s \in O_s$. As a result, x can be indexed by multiple ranked lists based on the parameter direction.

PE subindex I_{PE} construction. Each defined predicate $\alpha \in A_s$ corresponds to two lists $R(\alpha)[p]$ and $R(\alpha)[e]$ in I_{PE}. Let $l' \in \{p, e\}$: a scene x is indexed in $R(\alpha)[l']$ if the non-negative form of a predicate appears in $ss.Prec$ ($l' = p$) or $ss.Eff$ ($l' = e$) of any service $ss \in x.SS$. Each $R(\alpha)[l']$ consists of a list of pairs $\{(x.id, d_a(x, \alpha[l']))\}$ in the descending order of $d_a(x, \alpha)[l']$. The value of $d_a(x, \alpha)[l']$ is the plausibility $pl(\alpha, x)[l']$ of the predicate α over the preconditions ($l' = p$) or effects ($l' = e$) of all services of x. Denote $A_s(x)[l']$ the set of non-negative predicates that appear in the preconditions or effects of the services provided by x; and $\mathscr{H} = 2^{A_s(x)[l']}$ is the power set of $A_s(x)[l']$:

$$
\begin{aligned}
pl(\alpha,x)[l'] &= 1 - Bel_{A_s(x)[l'] \backslash \alpha}(x), \\
Bel_H(x)[l'] &= \textstyle\sum_{h \subseteq H} v(h), \\
v_H(x)[l'] &= \frac{n_H(x)[l']}{n_{\mathscr{H}}(x)[l']}, \text{ subject to:} \\
&\quad v(\emptyset) = 0, \textstyle\sum_{H \subseteq \mathscr{H}} v(H) = 1, \\
n_{\mathscr{H}}(x)[l'] &= \textstyle\sum_{H \subseteq \mathscr{H}} n_H(x)[l'], \\
n_H(x)[l'] &= \textstyle\sum_{\alpha \in H} n_\alpha(x)[l'], \\
n_\alpha(x)[l'] &= \textstyle\sum_{ss \in x.SS} P_\alpha(x.ss[l'] | \alpha).
\end{aligned}
\tag{6.3}
$$

where $P_\alpha(x.ss[l']|\alpha)$ is the probability of $x.ss[l']$ being true given that α is true. This value can be computed via the truth table of the formula $x.ss[l']$. Intuitively, if the truth of a predicate $\alpha[l']$ has a larger probability of making $x.ss[l']$ true, then object x is more suitable to be indexed (with greater score value) in $R(\alpha)[l']$ and increases the chance of x to be selected as relevant for a given query q, if $\alpha[l']$ appears in the requested service precondition or effect.

Geometric feature index construction: Differing from the scene concept and services, each geometric feature gf of a scene x can contain a set K_f ($f \in \mathscr{F}$) of attributes. Each attribute $k \in K_f$ can have numeric or text value(s). For constructing geometric feature index I_{GF}, iRep3D approach leverages a B+tree $bt(f,k)$ for each attribute $f.k$. Besides, it maintains any scene x in $bt(k,t)$ in the descending order of the attribute value $v(f.k)$, if x has value on $f.k$. To control the size of $bt(f,k)$, a pointer in a leaf node records the address of a ranked list $R(f,k)$ that contains a list of entry pairs $\{(x.id, x.v(f.k))\}$ in the descending order of $x.v(f.k)$.

Let M_l be the maximum number of scenes that a ranking can accommodate; M_n is the maximum fanout (the number of child nodes) of each node; $\mathscr{X}_f \subseteq \mathscr{X}$ is the subset of scenes containing a value of $k \in K_f$: The construction process for $bt(f,k)$ is done by the following steps:

1. sort the scenes in $\mathscr{X}_f$ in the descending order of $v(f.k)$;
2. compute the number n_l of needed ranked lists: $n_l = \lceil \frac{|\mathscr{X}_f|}{M_l} \rceil$;
3. create $\lceil \frac{n_l}{M_n} \rceil$ leaf-nodes;
4. initialize the pointers from leaf-node to rankings and label each pointer with the attribute value of the first entry in the corresponding ranking;
5. compute the number of needed non-leaf-nodes in each level from bottom to top and create their pointers and labels.

If x contains multiple instances of the same feature type attribute, x will have multiple entries in $bt(f,k)$. Each entry will be additionally labeled with the specific name ($gf.name$ in Definition 3.)

of the feature instance gf in the context of x. All the B+trees of all the possible feature attributes compose the geometric feature index.

6.3.2 Hybrid Query Processing

Once the semantic indices are created, an on-line query q can be answered efficiently. Key idea is to process three sub-queries of q over in parallel the scene concept, semantic service and geometric feature indexes. Those processes yield three ranked lists $R_{sc}(q)$, $R_{ss}(q)$ and $R_{gf}(q)$ of scenes relevant to q in terms of scene concept, semantic service and geometric feature, respectively. Final ranking is computed by applying Fagin's TA [92]. In this section, we detail the sub-query processing in each aspect and the final aggregation.

Scene concept subquery processing: If the logical expression $\tau(C, O_{req})$ of the requested SC C of a query q is not empty, the subquery for the SC will be processed. For this purpose, the process classifies $\tau(C, O_{req})$ as a concept $C' \in O$ into the scene ontology O and then returns the corresponding rank list $R_{SC}(q)$ of candidate scenes that are (partially) relevant to q in terms of the scene concept $q.C$.

Semantic service subquery processing: If $q.SS$ of a query q is not empty, the subquery for the semantic service aspect will be processed: First, for each $ss \in q.SS$, a rank list $R(ss)$ of scenes that are relevant to ss is computed. For this purpose, the indices I_{IO} and I_{PE} are searched in parallel. The resulted ranked lists $R(ss)[io]$ and $R(ss)[pe]$ are further merged into the list $R(ss)$ of scenes relevant to ss. Finally, all the lists $R(ss)$ of $ss \in q.SS$ are merged, which leads to the ranked list $R_{SS}(q)$ of scenes. Each scene in $R_{SS}(q)$ partially matches q in terms of the requested semantic services.

Searching index I_{IO} for ss. For each $ss \in q.SS$, this subprocess first retrieves in parallel a set of ranked lists $\{R(C'_s)[l]\}$ $(l \in \{i, o\})$. Each list corresponds to a distinct parameter concept $C'_s[l]$ in $ss[l]$. For this purpose, the logical expression of each distinct concept C'_s in $ss[i]$ ($ss[o]$) is classified to a concept $C_s \in O_s$ and its corresponding ranked list with suffix $[i]$ ($[o]$) is retrieved. Subsequently, TA [92] is performed on $\{R(C'_s)[l]\}$ to compose a ranked list $R(ss)[io]$ of scenes relevant to q with respect to the IO parameters of the requested service ss.

Let m the cardinality of $\{R(C'_s)[l]\}$ for ss. TA performs a sorted scan of all its input list in $\{R(C'_s)[l]\}$ from top to bottom in parallel. The i-th scan fetches the score values at the i-th positions of all lists in $\{R(C'_s)[l]\}$. Besides, it employs a m-ary function t for computing the aggregated relevancy score and threshold. The general form of t is given in Fagin's work [92] which leaves space to applications for further customization. In this context, we define t as the weighted average of the vector of scores $\vec{s}$ fetched from each rank list in $\{R(C'_s)[l]\}$ per scan. The weight v_j of the j-th list in

$\{R(C'_s)[l]\}$ refers to the number of appearance of its corresponding concept C'_s in either $ss[i]$ or $ss[o]$:

$$t(\vec{s}) = \frac{\sum_{j=1}^{m} v_j \cdot s_j}{\sum_{j=1}^{m} v_j} \tag{6.4}$$

Each scan performed by the TA may find a new scene x_n that does not exist in the current $R(ss)[io]$. To insert x_n into $R(ss)[io]$, it is necessary to compute the aggregated relevance score $s(x_n, ss)[io]$ of x_n with respect to the IO of $ss \in q.SS$: From each ranked list in $\{R(C'_s)[l]\}$, TA collects (possibly by random access) the so far missed $d_s(x_n.id, C'_s)$ of x_n; and further applies the t function on all $d_s(x_n.id, C'_s)$ in order to compute $s(x_n, ss)[io]$. TA maintains a threshold value T for determining its termination, which is updated with the t function value over the latest scanned values. This update happens after each scan. TA terminates, if $T \leq s(x, ss)[io]$ for all the ranked objects x in $R(ss)[io]$.

Searching I_{PE} for ss. For each $ss \in q.SS$, the searching of I_{PE} for ss results in two sets of ranked lists $\{R(\alpha)[l']\}$ ($l' \in \{p, e\}$) for every non-negative predicate α in $ss[l']$. In addition, it merges the ranked lists in each set into a list $R(ss)[l']$ of scenes that are relevant to ss in terms of $ss[l']$. For this purpose, multiple pairs of the same object x in different lists are merged. Pairs in different lists are merged if they share the same scene id. The score value $s(x, ss[l'])$ of x in $R(ss)[l']$ of each result pair is computed by applying the Gödel minimum t-norm and maximum t-conorm functions according to the conjunctive, respectively disjunctive relations between the predicates in $ss[l']$:

$$\begin{aligned} s(x, ss[l']) &= min_{cla \in ss[l']}(s(x, cla[l'])), \\ s(x, cla[l']) &= max_{\alpha \in cla}(d_a(x, \alpha)[l']). \end{aligned} \tag{6.5}$$

where $cla[l']$ denotes a clause of disjunctive predicates. Finally, the search process merges $R(ss)[p]$ and $R(ss)[e]$ in order to compute $R(ss)[pe]$ of scenes which are relevant to ss in terms of the precondition and effect. The completion of the parallel computations of $R(ss)[io]$ and $R(ss)[pe]$ triggers their merging and yields the ranked list $R(ss)$ of scenes relevant to q in terms of $ss \in q.SS$. The relevancy score $s(x, ss)$ of x in $R(ss)$ is the convex combination of the corresponding scores in $R(ss)[io]$ and $R(ss)[pe]$:

$$s(x, ss) = \phi s(x, ss[io]) + \psi s(x, ss[pe]), \tag{6.6}$$

where the real positive values ϕ and ψ ($\phi + \psi = 1$) are the weights of IO and PE matching respectively. They can vary in specific systems with different concerns.

Merging $R(ss)$ **for all** $ss \in q.SS$. The subprocess on I_{SS} merges the resulted ranked lists $R(ss)$ of all $ss \in q.SS$. The entries in different lists are merged if they share the same id. The relevancy score $s(x, q.SS)$ for x with respect to $q.SS$ is the average of the scores $s(x, ss)$ of x in $R(ss)$ for each service

ss:

$$s(x, q.SS) \quad = \tfrac{1}{|q.SS|} \sum_{ss \in q.SS} s(x, ss). \tag{6.7}$$

Finally, the merged list are resorted in descending order of $s(x, q.SS)$ yielding the ranked list $R_{SS}(q)$ of scenes partially relevant to q with respect to $q.SS$.

Geometric feature subquery processing: The subquery processing in geometric feature aspect is done by the following steps:

1. For each $gf \in q.GF$, it first applies parallel searches in the B+trees $bt(gf.f, k)$. Each travel retrieves a ranked list $R(gf.f.k)$ of scenes relevant to q in terms of $gf.f.k$. Please note that $R(gf.f.k)$ does not have similarity scores but the attribute values.
2. For each entry $(x.id, x.v(f.k)) \in R(gf.f.k)$, the process computes a geometric feature attribute similarity score $s_k(v_q(f.k), x.v(f.k))$ between the requested volume $v_q(f.k)$ and $x.v(f.k)$. It results in a new ranking $R(q, gf.f.k)$ of scenes that are relevant to q in terms of the requested volume on $gf.f.k$.
3. All lists $R(q, gf.f.k)$ of the attributes belonging to the same feature type $gf.f$ are further merged (by scene id) into a ranking $R(q, gf)$ of scenes relevant to q in terms of gf.
4. TA algorithm is executed on these the feature-level rankings $R(q, gf)$, which computes the total ranking of scenes relevant to q in terms of $q.GF$.

The geometric data types specified by X3D, XML3D and COLLADA specifications includes the following primitive data types:

- single number, string or boolean (e.g. `SFDouble, SFString`);
- 2-, 3- or 4-ary tuple of numbers or strings (e.g. `SFVec2d, SFVec3f, float4_type`);
- vector of values in the types above (e.g. `MFDouble, MFVec3d`). Denote $tp(k)$ the primitive data type of feature attribute k. The geometric feature attribute similarity score can be computed as follows:

  $s_k(v_1, v_2) =$

  • $and(v_1, v_2)$, if $tp(k)$ is single boolean;

  • $EDS(v_1, v_2) = 1 - \dfrac{ED(v_1, v_2)}{max(|v_1|, |v_2|)}$, if $tp(k)$ is a single string, where $|v_1|$ denotes the length of v_1;

  • $min(\dfrac{v_1}{v_2}, \dfrac{v_2}{v_1})$, if $tp(k)$ is single number;

  • $\dfrac{1}{|v_1|} \sum_{i=1}^{|v_1|} xor(v_{1i}, v_{2i})$, if $tp(k)$ is a boolean vector, where $|v_1|$ denotes cardinality of v_1;

- $cos\_sim(v_1, v_2)$, if $tp(k)$ is a pair, triple or a vector of numbers;
- $VEDS(v_1, v_2) = \frac{1}{|v_1|}\sum_{i=1}^{|v_1|} EDS(v_{1i}, v_{2i})$, if $tp(k)$ is a pair, triple or a vector of strings;
- $\frac{1}{|v_1|}\sum_{i=1}^{|v_1|} cos\_sim(v_{1i}, v_{2i})$, if $tp(k)$ is a vector of pairs or triples of numbers;
- $\frac{1}{|v_1|}\sum_{i=1}^{|v_1|} VEDS(v_{1i}, v_{2i}$, if $tp(k)$ is a vector of pairs or triples of strings;

where $and(v_1, v_2)$ is the conjunction of v_1 and v_2; $EDS(v_1, v_2)$ the Levenstein edit distance of v_1 and v_2; $cos\_sim(v_1, v_2)$ the cosine distance of v_1 and v_2. In the context of iRep3D, we skip the types `SFImage, MFImage, SFTime` and `MFTime` in X3D specification since they are not geometric data type.

The first step in geometric feature subquery processing retrieves a ranking $R(gf.f.k)$ of entries containing the identifiers of scenes and their values on attribute $v(f.k)$. Instead of directly retrieving a ranked list pointed by a leaf node, $R(gf.f.k)$ is then computed by applying tolerance window strategy. It retrieves at most N entries from the both sides of the entry $(x.id, x.v(gf.f.k))$ whose feature attribute value has minimum distance to $v_q(gf.f.k)$. Name N the half-window width value.

Final aggregation: Three sub-rankings of the partially matched 3D scenes with respect to q are constructed by parallel subquery processing. The final ranking is executed immediately after their completion. That is to apply TA on those ranked lists: $R_{sc}(q)$, $R_{ss}(q)$ and $R_{gf}(q)$. In case that the score of a scene x is missing in some rank list, the lowest score in that list is used. TA terminates if the threshold is not larger than the least score of the m-th (cf. Definition 4) entry in the total ranking, or all three lists above have been entirely scanned.

6.4 Extensions

iRep3D features the efficient 3D scene selection for the P2P search with complex queries. One crucial part is the maintenance of indices on the scene annotation change. In this perspective, the large computational complexity of semantic comparison can be a bottleneck of system efficiency, if the re-ranking of an updated 3D scene in all of the predefined topics is considered. The challenge in this context then lies in that how to prune out a part of topics, which can be ignored during the index update process. Moreover, another challenge in the 3D scene sharing and retrieval aspect is the reusability of them. Its major phenomenon are that (i) the useful sub-scenes can be found, only if the entire scene has been retrieved; and (ii) the observation and division for useful sub-scenes are often performed manually. The main difficulty for improving the 3D scene reusability is the extraction of 3D sub-scenes and the indexing of them. In this section, we introduce two extensions to the iRep3D approach. The first is the sub-scene extraction component. It allows to automatically decompose a new XML-based 3D scene and therefore benefits the deep search; while the second component is the semantic-based incremental indexing of 3D scene. With this, the system is able to quickly adapt to the change of semantic annotations on its data.

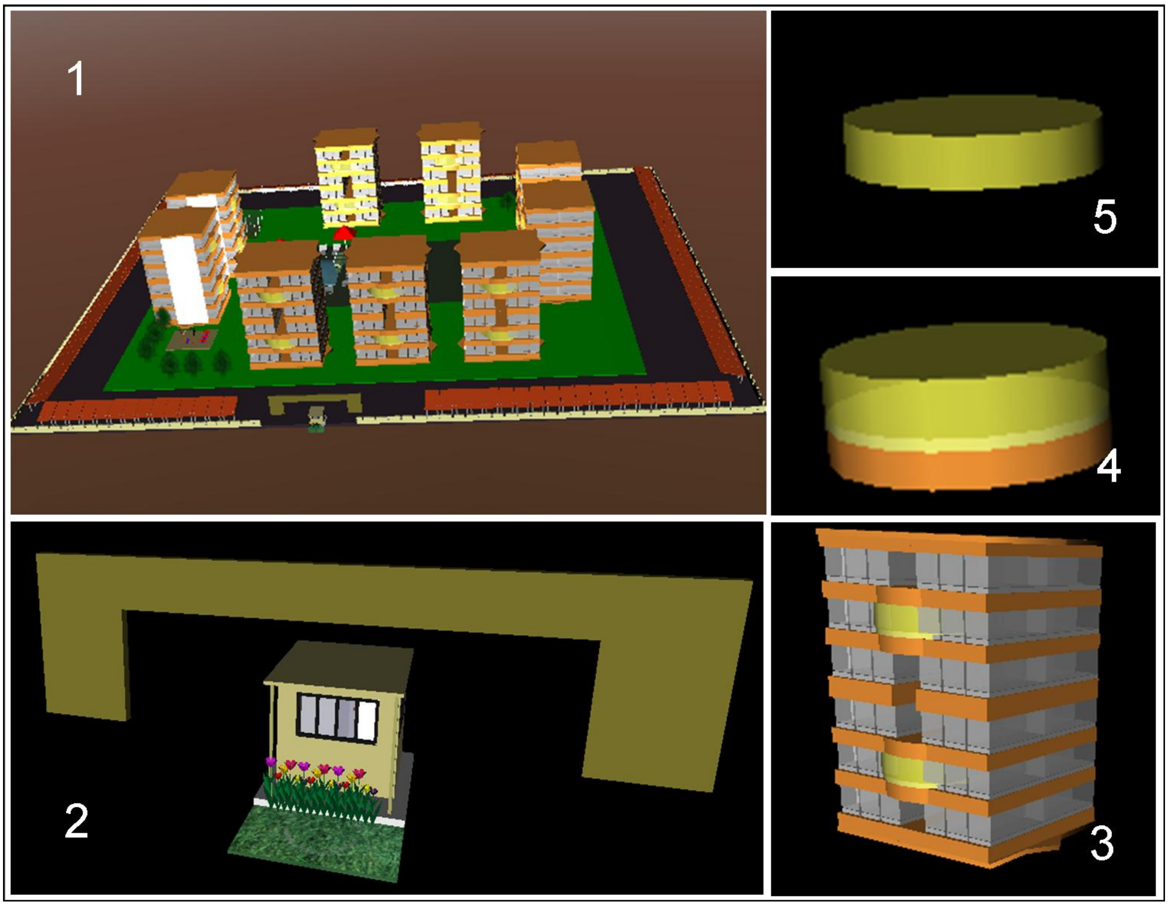


Fig. 6.2 XML-based 3D scene hierarchy

6.4.1 Sub-scene Extraction

Complex XML-based 3D scene is commonly composed by either or both of two methods: (a) main scene file plus a set of referenced sub-scene files with proper transformations and replications; (b) one big file for the whole scene, which internally contains the definitions of all sub-scenes from scratch. For example, a X3D scene shown in Figure 6.2(1) illustrates a residential block containing several buildings (cf. Figure 6.2(3)) and a guard room (cf. Figure 6.2(2)). The latter sub-scenes are defined in separate .x3d files that are referenced by the main scene file[8]; while the sub-scene in Figure 6.2(3) is defined by grouping and transforming the very basic shape components, such as `Cylinder`, `Box`, etc.. By analyzing these (groups of) components of the sub-scene "building", a sub-scene "round window" (cf. Figure 6.2(5)) (with its beam (cf. Figure 6.2(4))) can be extracted. It makes scene if not only the sub-scene files but also those components or in particular a group of

---

[8] `https://savage.nps.edu/Savage/Buildings/ZenCondominium/`

some of them can be extracted, annotated and further indexed for the purpose of scene reuse in 3D modeling. For this, iRep3D is enabled to offer scene hierarchy detection that exhaustively extracts hierarchical sub-scenes of an input scene *x*. The recursive process *SubSceneExtraction(x, sh)* is shown in Alg.7.

Algorithm 7 $SubSceneExtraction(x,\ sh)$

Input:
x: the 3D scene which sub-scenes is to be extracted;
sh: the scene hierarchy of *x*.
Output:
sh: the scene hierarchy of *x*.

```
Record $x$ as a direct sub-scene of $x.p$ in $sh$, if $x.p \neq null$;
Find direct sub-scenes $SUB(x)$ of $x$;
for Each $x' \in SUB(x)$ do
    Record $x'$ as a direct sub-scene of $x$ in $sh$;
    $x'.p \leftarrow x$;
    Extract the definition of $x'$ into a file, if $x'$ is internally defined in $x$;
    if $x'$ is not a basic component then
        $SubSceneExtraction(x', sh)$;
    end if
end for
return $sh$;
```

6.4.2 Incremental Semantic Re-indexing

With an incremental index maintenance strategy, iRep3D is enabled to adapt to the update of the semantic annotations of 3D scenes. To achieve this, semantic comparisons and file/disk accesses are necessary. The goal is to lower the costs of those operations as much as possible. An update of the semantic annotations of a 3D scene x stands for either insertion $ins(x)$, deletion $del(x)$ or modification $mod(x)$ operation on in particular the conceptual and behavioral features of x. As the $mod(x)$ operation equals to an ordered 2-step sequence: $del(x) \rightarrow ins(x)$, the research focuses on the incremental index maintenance of scene concept and semantic service indices on $ins(x)$ and $del(x)$ operations.

$ins(x)$ of x with its conceptual description C causes the update of I_{SC}. The prior task is to find a subset $\mathscr{C}'$ of concepts in O, into each of which x could be indexed. This is achieved via the heuristics as follows: First, the system performs breadth-first-search (BFS) over the concept hierarchy in O and prunes out the sub-hierarchy, if its root concept C' meets both the criteria: *(i)* $s_{ab,\sqsubseteq}(C,C') < \theta_1$ ($\theta_1 \in (0,1]$) and *(ii) there exists conflict between the primitive concepts in C and* C'. By Equation 6.1, the computation of $s_{ab,\sqsubseteq}(C,C')$ can be in addition simplified by considering the value of $\frac{|K|}{|C'|}$ only, as the remained part is a positive real value less or equal to 1. This process constructs the subset $\mathscr{C}'$, which comprises the concepts having not been pruned out. The completion of the prune triggers the computation of $s_{ab,\sqsubseteq}(C,C')$ for each $C' \in \mathscr{C}'$.

For example, given $\theta_1 = 0.6$, consider to insert the 3D scene `Ship686` in Figure 6.1. Let us assume that a simple ontology O^* is pre-defined for scene concepts. In addition, a concept $Vehicle \in O^*$ and two of its direct subclasses Car and $Ship$ are defined as follows: $\tau(Vehicle)$ = $Vehicle^P \sqcap \forall has.Engine^P$; $\tau(Car)$ = $Vehicle^P \sqcap Car^P \sqcap \forall has.Engine^P \sqcap \forall usedIn.Land$; $\tau(Ship)$ = $Vehicle^P \sqcap Ship^P \sqcap \forall has.Engine^P \sqcap \forall usedIn.Water$; Starting BFS process from $Vehicle$, the sub-hierarchy with the root Car is pruned out due to (i) the conflicting primitive terms Car^P and $Ship^P$ in $\tau(Car)$, respectively $\tau(Ship686)$ as well as (ii) $\frac{|K(Ship686,Car)|}{|Car|} = \frac{1}{4} < \theta_1$.

$ins(x)$ of x with a set of semantic services $x.SS$ causes the update of I_{SS}. For each $ss \in x.SS$, the update process comes into two parallel sub-processes for the IO parameter concepts and predicates in PE, respectively. The former works based on the heuristic above under the parameter direction restriction (cf. Section 6.3); while the latter computes the plausibility score value $pl(\alpha,x)[l']$ of a predicate $\alpha \in ss[l']$ $(l' \in \{p,e\})$ and insert a pair $(x, pl(\alpha,x)[l'])$ into the corresponding ranked list $R(\alpha)[l']$.

$del(x)$ triggers the system to remove any entries about x from all the indices where it is recorded. The challenge is to locate those entries. Instead of traversing through all existing indices to find them, iRep3D takes advantage of a lookup table T_{LK}. Each 3D scene x corresponds to one entry $(x.id, lst(x.id))$ in T_{LK}, which consists of the identifier of x and a pointer to the head address of a linked list $lst(x.id)$. Each node in $lst(x.id)$ records the address of a ranked list where an entry of x is indexed. Each insertion of a new scene x' leads to the insertion of $(x'.id, list(x'.id))$ in T_{LK}; while each insertion of a rank about x' in a ranked list R causes an appending operation that adds the address of R to the tail of $lst(x'id)$. On top of this, a B+ tree is applied over the identifiers of all scenes. This shortens the retrieval time of the entry corresponding to $x.id$. For maintaining T_{LK}, $del(x)$ causes system to merely set the pointer of $x.id$ empty, than delete $(x.id, lst(x.id))$ from T_{LK}. This mechanism can leave empty slot in tree node, which would be used by a new scene in the future.

6.5 Complexity

In this section, we analyze the computational complexity for both the off-line indexing and on-line query answering processes. The meaning of useful symbols for the complexity analysis is listed in Table 6.1. We prove that (i) the computational complexity for the off-line indexing of an annotated 3D scene x is $\boldsymbol{O}\ (N_1 \cdot n_n + m \cdot (u \cdot N_2 \cdot n^n + p \cdot l_{pl}) + t \cdot s \cdot f \cdot M)$; and (ii) the total computation complexity for the hybrid on-line query answering is $\boldsymbol{O}\ (N_1 \cdot n^n + m(N_2 \cdot n^n \cdot u + p \cdot M) + (m + t \cdot s) l_{TA} + t \cdot s \cdot (f \cdot logM + f \cdot M + MlogM))$.

Lemma 6.1. *The computational complexity for the off-line indexing of an annotated 3D scene x is* $\boldsymbol{O}(N_1 \cdot n_n + m \cdot (u \cdot N_2 \cdot n^n + p \cdot l_{pl}) + t \cdot s \cdot f \cdot M)$.

Proof: The off-line indexing process categorizes a scene x into the scene concept index I_{SC}, service

n	the number of primitive terms in the shared vocabulary V(cf. Section 6.1.1).
N_1	the number of scene concepts defined in scene ontology O.
N_2	the number of concepts defined in the service parameter type ontology O_s.
m	the number of services that a scene (query) can have.
u	the number of input (output) parameters that a service.
p	the number of predicates that a service precondition (effect) can have.
t	the number of geometric features a scene (query) can have.
s	the number of attributes a geometric feature can have.
f	the computational complexity of geometric feature attribute value comparison.
M	the number of 3D scenes in iRep3D repository.
l_{pl}	the computational complexity for computing the plausibility value of a predicate in the precondition (effect) of a service, $l_{pl} = \boldsymbol{O}(2^{(p-1)}) \sim \boldsymbol{O}(2^p)$ times addition operations have to be executed.
l_{TA}	the computational complexity of the aggregation method (threshold algorithm). $\boldsymbol{O}(l_{TA}) = \boldsymbol{O}(M \cdot max(m,u))$.

Table 6.1 Useful symbols for the complexity analysis of iRep3D approach.

index I_{SS} and geometric feature index I_{GF}. The process of indexing x in I_{SC} computes the similarity scores of the scene description concept C with each $C' \in O$. It in the worst case costs $\boldsymbol{O}(N_1 \cdot n^n)$. To index x in I_{SS}, iRep3D tries to categorize x in the IO (I_{IO}) and PE (I_{PE}) sub-indices, respectively. For the IO part, iRep3D checks the similarity scores of each parameter concept C_s and $C_s' \in O_s$. In total, to index x into I_{IO} spends at most $\boldsymbol{O}(2 \cdot u \cdot N_2 \cdot n^n \cdot m) \sim \boldsymbol{O}(u \cdot N_2 \cdot n^n \cdot m)$. For the PE part, iRep3D indexes x in the ranked list a predicate α if the non-negative form of α appears in a service of x. It follows that the PE indexing spends at most $\boldsymbol{O}(2 \cdot p \cdot l_{pl} \cdot m) \sim \boldsymbol{O}(p \cdot l_{pl} \cdot m)$. Therefore, the indexing of x in I_{SS} takes $\boldsymbol{O}(m \cdot (u \cdot N_2 \cdot n^n + p \cdot l_{pl}))$. The process of indexing x at geometric feature level inserts x into the B+tree of each attribute in any specified feature. The computational complexity for indexing x in I_{GF} is $\boldsymbol{O}(t \cdot s \cdot M \cdot f)$. On top of this, the overall computational complexity for the off-line indexing of x is $\boldsymbol{O}(N_1 \cdot n_n + m \cdot (u \cdot N_2 \cdot n^n + p \cdot l_{pl}) + t \cdot s \cdot f \cdot M)$. ■

Lemma 6.2. *The total computation complexity for the hybrid on-line query answering is* $\boldsymbol{O}(N_1 \cdot n^n + m(N_2 \cdot n^n \cdot u + p \cdot M) + (m + t \cdot s)l_{TA} + t \cdot s \cdot (f \cdot logM + f \cdot M + MlogM))$.

Proof: To answer a hybrid query with scene conceptual, functional and geometric descriptions, iRep3D processes three sub-queries in parallel and aggregates their sub-results into a final ranked list by TA. For the search in scene concept index, iRep3D finds the a ranked list by classifying C into O, which spends $\boldsymbol{O}(N_1 \cdot n_n)$.

In the service dimension, the sub-query is processes at IO, respectively PE levels. The search at IO level involves multiple concept classification operations and one aggregation operation. It costs $\boldsymbol{O}((N_2 \cdot n^n \cdot 2 \cdot u) + l_{TA})$, where $\boldsymbol{O}(N_2 \cdot n^n \cdot 2 \cdot u)$ refers to the cost of classifying the service parameter concepts of a service in the request to the service ontology. This process yields a set of ranked lists of scenes, which each contains the scenes that match to the request in terms of one service parameter. In order to obtain the scenes that match the request in term of a service, TA algorithm is executed for aggregation. This additionally costs $\boldsymbol{O}\ (l_{TA})$. As for the search at PE level, iRep3D finds the ranked lists about the predicates used by each service in the request. For one service, this process costs $\boldsymbol{O}\ (2 \cdot p)$ with a hash map based data structure that associates predicate names to the addresses of their corresponding ranked lists. Subsequently, a merge operation with Gödel t-(co)norm based aggregation function is performed. This produces the ranked list of scenes that match one service at PE level. It costs $\boldsymbol{O}\ (2 \cdot p \cdot M)$. To merge the sub-results out of searches in IO and PE levels, at most $\boldsymbol{O}\ (M)$ convex combinations (cf. Equation 6.6) are needed. In total, the searching for one service spends $\boldsymbol{O}((N_2 \cdot n^n \cdot 2 \cdot u) + l_{TA}) + (2 \cdot p + 2 \cdot p \cdot M) + M)$. Since a request can at most contains m services, the above IO and PE search will be performed by $\boldsymbol{O}\ (m)$ times. On top of this, we conclude that the computational complexity for the search at functional level is $\boldsymbol{O}\ (m((N_2 \cdot n^n \cdot 2 \cdot u) + l_{TA}) + (2 \cdot p + 2 \cdot p \cdot M) + M)) \sim \boldsymbol{O}\ (m(((N_2 \cdot n^n \cdot u) + l_{TA}) + p \cdot M))$.

The query processing at geometric feature level finds the ranked list from B+ trees of all requested attributes. Each list is retrieved at the cost $\boldsymbol{O}\ (f \cdot logM)$. For each attribute, similarity scores between the request and the candidate scenes are computed. Scenes will be ranked in a list by the scores. This at most spends $\boldsymbol{O}\ (f \cdot M + MlogM)$ where $\boldsymbol{O}\ (MlogM)$ is spent for the insertion and sorting. In order to achieve the relevant scenes on one requested geometric feature, a merge operation will be conducted over the ranked lists of all its attributes. This costs $\boldsymbol{O}\ (s \cdot M)$. Once the above computation has been done for all geometric features, the results is aggregated by TA. Therefore, the computational complexity at geometric feature level is $\boldsymbol{O}\ (t \cdot s \cdot (f \cdot logM + f \cdot M + MlogM) + (t \cdot s \cdot M)) + l_{TA}) \sim \boldsymbol{O}\ (t \cdot s \cdot (f \cdot logM + f \cdot M + MlogM) + l_{TA})$.

The completion of sub-query process triggers an execution of TA for the final aggregation. The total computation complexity for the hybrid query answering spends $\boldsymbol{O}\ (N_1 \cdot n^n + (m(((N_2 \cdot n^n \cdot u) + l_{TA}) + p \cdot M)) + (t \cdot s \cdot (f \cdot logM + f \cdot M + MlogM) + l_{TA}) + l_{TA}) \sim \boldsymbol{O}\ (N_1 \cdot n^n + m(N_2 \cdot n^n \cdot u + p \cdot M) + (m + t \cdot s) l_{TA} + t \cdot s \cdot (f \cdot logM + f \cdot M + MlogM))$. ■

6.6 Experimental Evaluation

In this section, we present and discusses the results of our comparative experimental evaluation of the performance of iRep3D.

Settings: To the best of our knowledge, there does not exist a test collection with labeled query-answer sets in which the 3D scenes are annotated with both conceptual description and semantic services. Therefore, for the performance experiments, we manually annotated 616 media objects, in particular 591 3D scene graphs in X3D[9] and 25 3D scene graphs in XML3D[10]. The scene concepts are defined in an ontology O in OWL-DL, which comprises 260 concepts, 48 roles and 7 role restric-

---

[9] `http://www.web3d.org/x3d/content/examples`

[10] `http://www.xml3d.org/`

tions. These 3D scene graphs describe real-world objects like a car or computer which functionalities are described by references to in total 33 semantic services in OWL-S. The annotations are embedded into the XHTML files of the scene graphs with standard RDFa. For the labeled query answer set, we created a preliminary set Q of 20 queries. Each is labeled with 10 relevant scene graphs that are scored from 1.0 to 0.1 ($rel \in \{1.0, 0.9, \ldots, 0.1\}$). Others are assumed to be not relevant and scored with 0. Set $A = 10$ for all $q \in Q$; $\theta = 0.5$; $\phi = \psi = 0.5$; $a = 0.1$, $b = 0.9$ for importance function; and $N = 10$ for half tolerance window width.

Competitor approaches: For the comparative experimental evaluation of our approach, we choose:

- FB3D: the functional and behavioral ontology based semantic approach proposed by [41], since this solution, as we known, is the nearest one to our concerned problem in 3D simulated intelligent agent system;
- RIR: the RDF triple index-based approach [15] as a RDF-based competitor;
- ADL: the syntactic based open-source 3D model repository as an in-used competing system.

For FB3D, 12 concepts and 4 roles are additionally derived based on the annotated semantic services and added into the scene ontology such that FB3D is capable to perform reasoning on the functional descriptions of scenes. In addition, we make FB3D approach to pre-load the scene concept for the behavior and functional descriptions before its query processing. This eliminates the loading and parsing time of 3D scenes. We create the RDF triples of the 3D models for the RIR approach by applying the Jena OWL analyzer[11] to the conceptual descriptions as well as service parameters of models. The precondition and effect of services are encoded in RDF plain literals. In addition, the indexing facilities of MySQL database is employed to maintain the generated RDF triples in terms of their subject, predicate and object, respectively. For querying the RDF triples, one SPARQL query is generated for each labeled query. For the open-source ADL system, we store the syntactic descriptions including the meta-properties of our tested 3D scenes into MySQL database, in which the ADL required schemas are pre-created.

Metrics: We test the retrieval performance of iRep3D in terms of the following metrics:

- Macro-averaged precision (MAP_λ) at 11 recall levels (RE_λ) ($MAP@RE$) with equidistant steps of 0.1:

$$MAP_\lambda = \frac{1}{|Q|} \sum_{q \in Q} max\{pre|re \geq RE_\lambda, \text{ for } \forall \langle pre, \ re \rangle \in PR_q\}. \tag{6.8}$$

  A set PR_q of precision-recall $\langle pre, re \rangle$ pairs is observed for each query q when scanning the ranked 3D scenes in the returned answer set of q stepwise for true positive. Nearest-neighbor

---

[11] `http://jena.apache.org/documentation/ontology/`

interpolation is used for estimation of missed precision values for some queries at some recall levels.

- Averaged precision (AP):

$$AP = \frac{1}{|Q|}\sum_{q \in Q} \frac{\text{\# of relevant objects}}{10}. \tag{6.9}$$

- Averaged discounted cumulative gain at rank position 10 (DCG_{10}) :

$$DCG_{10} = \frac{1}{|Q|}\sum_{q \in Q}(rel_1(q) + \sum_{i=2}^{10} \frac{rel_i(q)}{log_2 i}). \tag{6.10}$$

where $rel_i(q)$ is the labeled relevancy score of the scene ranked at i in the result list for a query q.

- Averaged query response time length ($AQRT$) measured in seconds:

$$AL = \frac{1}{|Q|}T(Q). \tag{6.11}$$

where $T(Q)$ is the total processing time of all queries in Q.

Evaluation results: In the experimental comparison of the search performance of iRep3D, FB3D, RIR and ADL, all configurations run though the same set of 20 labeled queries. The experiment result shows that iRep3D outperforms FB3D, RIR and ADL in terms of MAP@RE, AP and DCG_{10} (cf. Figure 6.3 and Table.6.2). iRep3D achieves up to around 34%, 13% and 55% more MAP than FB3D, RIR and ADL, respectively. The reason is that FB3D takes use of the strict logic subsumption determination that has the risk of losing relevant answers. Minor conflicts between two compared concepts can cause logical fail during reasoning. Moreover, the layered query-processing of iRep3D in respect of services offers more tolerance to parameter mismatching than the one-shot functional concept matching in FB3D. RIR approach supports the indexing of RDF triple description of 3D scenes, which alleviates the syntactic mismatching of the text-based search ADL. In comparison with the proposed approximated similarity measure of iRep3D, the exact SPARQL query pattern matching of RIR makes it appear less accurate during the indexing of subject, predicate or object in RDF triples. Given keywords, ADL directly queries its underlying database by wildcard SQL. Besides the keywords ambiguity problem, the search performance is also limited by its ignorance of text segmentation.

On the other hand, iRep3D achieves one-tenth of AQRT in comparison with the logic-based approach FB3D. However, it needs more response time per query than the RDF index-based retrieval approach RIR and the syntactic based ADL system. The reason is that iRep3D classifies the requested scene concept and service parameter concepts based on scene (service concept) ontology. This logical reasoning process needs more computation; whereas the latter approaches directly perform syntactic based matching over the input keywords or textual RDF patterns.

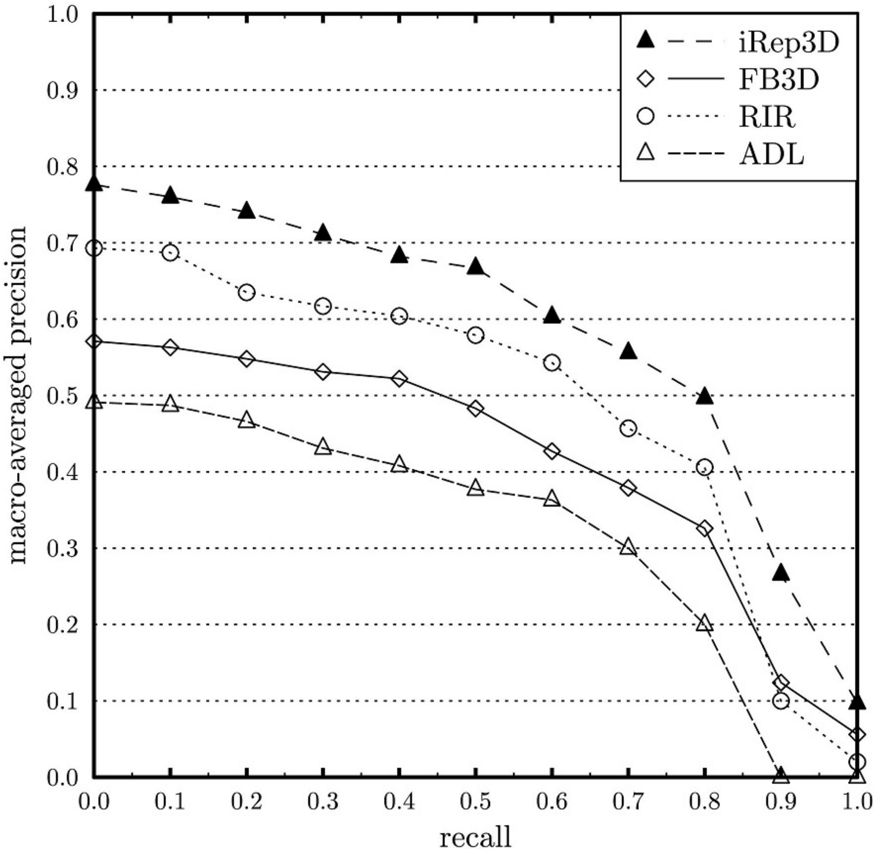


Fig. 6.3 MAP@recall of iRep3D, FB3D, RIR and ADL.

	iRep3D	FB3D	RIR	ADL
AP	**0.721**	0.490	0.633	0.408
DCG_{10}	**2.133**	0.952	1.370	0.767
AQRT (sec)	0.166	1.887	0.059	**0.042**

Table 6.2 AP, DCG_{10} and AQRT of iRep3D, FB3D, RIR and ADL.

6.7 Related Work

In this section, we present the representative contributions about semantic for 3D scene description and retrieval. As for the dominant online 3D repositories, we discuss them in Section 7.4.

Semantic-based 3D scene description: As the prerequisite of semantic-based 3D indexing and retrieval, the semantic description of 3D scenes refers to the mapping between a 3D scene and its identification, formation, category, meta-property and function. It bridges the gap between the 3D geometric data with the knowledge in the world. Commonly represented in different formalisms, such as ontology-based concepts and semantic services, the semantic descriptions can be used to simplify the 3D content indexing, searching and analysing. In this section, We discuss the related work about the semantic-based description of 3D scenes and introduce the representative approaches of semantic-based 3D scene indexing and retrieval afterwards.

A large number of previous works focus on the ontology based semantic description of 3D scenes in terms of their conceptual properties, like identification, formation and structure. The basic principle of them is to describe a 3D scene by means of both the conceptual categories and their relations. Ontologies are often used to keep upper layer knowledge, which provides the categorization of individual 3D scenes. The early work [235] proposes a method for mapping conceptual objects onto collections of graphical primitives. It is used to construct the knowledge-based representation of a virtual environment and further enable the conceptual processing of objects. The Flex-VR approach [374] [372] [373] introduces a method that leverages the parameterized 3D content objects to characterize 3D contents in terms of their geometric, structural spatial, behavioral features. In particular, such knowledge can be encoded as scripts in VRML, X3D or MPEG-4 files. In contrast, [32] enables the annotation of 3D scene characterizations and their linked 3D object to be stored in separate documents, which can be referenced by a XML based MPEG-7 descriptors of 3D animations. The authors introduce, in [130] [115] [129], an ontology based approach to describe virtual human by mapping the concepts of body parts with the geometric and structural features. In particular, this work scales to model the emotional movement of human body. De Paiva. et. al. introduce a content schema [72] for describing historic buildings during 3D based reconstruction. Concepts, for example the surface, repetition and transition, are defined and they can be encoded in the nodes of Maya Embedded Language[12].

Another group of works make a lot efforts on the semantic-based description of the behavioral features of 3D models, which are known to be imperative to the construction of 3D based animation and the virtual simulation of process. The ISReal framework [181] enables the description of 3D objects in virtual world in terms of not only the ontology based conceptual formation but also the functionalities particularly via semantic services. It enables user to immersively interact with the 3D virtual world by Web browser, where the scenes in XML3D format are rendered. This work bases on an extension of XML3D language in RDFa, which bridges the geometric representation of geometric data with semantics. The highlight of this work on describing the behavioral feature of 3D objects lies in the use of semantic service. This provides the opportunity of function-oriented 3D object parametrization, selection and composition, which enhances the deep integration of semantic Web and multimedia technologies. The work [64] takes use of ontology based conceptual description on

---

[12] `http://download.autodesk.com/global/docs/maya2013/en_us/Commands/index.html`

behaviors of 3D objects, like events and actions. Those concepts correspond to their counterparts specified in the well-known 3D scene definition languages, e.g. X3D. Besides, the ability of modelling behavioral features allows to encode the action via code bases. This idea is very similar with the application of Javascript for the interacting HTML components in dynamic Web pages.

Pellens. et. al propose a method [283] to describe the functionalities of 3D objects via diagrams. It utilizes a graphical notation means as well as a series of patterns for behavior parametrization and composition. Complex behavior and interactions of 3D objects can be specified in an abstract level, instead of coding. This idea follows the principle of model-driven development. Likewise, the RAIVE system [53] introduces the high-level conceptual description of events and actions for modeling avatar in 3D virtual world. Corresponding to the basic physical movements, like moving and turning around, such events and actions are further mapped to a set of constructors in natural language. Rather than a diagram based representation, finite-state transition network is used to model the mappings. The work [62] [63] proposes the Multimedia Interaction Model for user to specify the meta-data of the 3D objects' behaviors. In this model, interactions are decomposed into conditional events and actions, which are built on top of their mathematical expressions. This model uses a XML schema as the practical implementation. This work in addition provides a SQL-like language for the search of 3D scenes in terms of functions. iRep3D system achieves this via the semantic indexing of OWL-S services of 3D objects. Instead of depending on specific languages, it accepts the formally defined semantic service as the expression of requested behavior.

Semantic-based 3D scene retrieval: The content and geometric feature based 3D model retrieval approaches, such as [350] [38] [184] [277] [156] has been proposed. However, the incompatible geometric feature definitions in various solutions and possible mis-correspondence among different formalisms limit their usage. More and more concentration has been put to the facilities of classification, tagging and semantics. In the following, we discuss the representative contributions of semantic-based 3D scene retrieval:

Non-logic based approaches employ classification in terms of geometric or non-geometric descriptive properties. The work [112] suggests a probabilistic classification solution for 3D objects based on Gaussian process. By convex combination, it aggregates the geometric and classification based similarities for each pair of request and offer. Leifman et. al. [214] refines the geometric-topological feature matching by the non-supervised off-line learning and subsequent on-line supervised feature extraction. This approach works under the hypothesis that the feedback of relevancy from user is available. Another works [271] [396], via the similar idea, reduce feature dimension by sequentially applying unsupervised dimension reduction (*UDR*) and supervised dimension reduction (*SDR*) independently at 6 resolution levels for each 3D object. The approaches presented by the works [124] and [150] perform the off-line learning about non-geometric features of 3D objects by support vector machine (*SVM*) and label each ground object with the category in a predefined universe. Akguel et. al. propose the similar work [9] using SVM, which extracts from each 3D object a fixed number of geometric features over a defined feature space (off-line) and estimates the probabilistic similarity between a (on-line) query and candidate objects in terms of each feature. A synthetic ranking of matched objects is returned by applying the learned weights to the probabilistic similarity score at each feature. Comparing with iRep3D, these machine learning based approaches needs to train the kernel in advance by using a set with labeled queries and the quality of classification is highly sensitive to the quality of those queries.

In case that 3D scenes are annotated with RDF, quite a few mature RDF stores (indexes) with efficient SPARQL query processing can be used for purpose of retrieval. Alvez et. al. proposed an efficient image retrieval approach [15] based on the indexing of RDF annotations. However, the query on RDF data in this method is based on a boolean query model which is designed for exact matches. Besides, without a largely redundant indexing by means of RDF(S) entailment regime, SPARQL is still prone to ignore the logical concept hierarchy during query processing. The work [209] focuses on the indexing and retrieval for distributed multimedia contents. In addition to the limitations mentioned above, the distributed RDF query answering in [209] closely relies on the prior knowledge of RDF sources and therefore appears sensitive to unshared RDF schema namings. In the contrast, iRep3D takes advantage of the logic-based annotation on multimedia objects, which supports more accurate comparison and is less prone to be affected by syntactic mismatching and strict SPARQL pattern matching.

Logic-based annotations of 3D scenes enables semantic comparison and scene selection by means of logical reasoning facilities. According to our research context, we restrict the scope to the works particularly for retrieval purpose. Hois et. al. presents an approach [148] for 3D image recognition using a domain ontology, which provides a fundamental data set of hierarchical object categories by means of their functions. Such functional bias obtains particular importance for the object recognition during planning of robot actions. The work [41] presents a knowledge based system supporting the functionality driven annotation and retrieval of 3D models. The system supports the semantic annotation of 3D models and maintains the form, functionality and behavior ontology, written in OWL DL, in order to bridge the gap between 3D shapes and their functionalities. However, in comparison with iRep3D, the logical reasoning approach used by these efforts is highly sensitive to minor conflicts between compared concepts, which is prone to yield strict logic false negatives. Yang et.al propose, in [399], an approach to establish high-level content signatures of multimedia contents based on their logic-based representations. In addition, for handling imprecise queries, a hierarchical metadata of objects is constructed on the basis of signatures and linguistic relationships. Unfortunately, the high-level content signature may have lost the original semantics of the indexed multimedia objects. Besides, the linguistic extension can introduce more ambiguity to the remaining semantics. In contrast, iRep3D approach preserves as much semantics as possible during the indexing.

6.8 Summary

In this section, we presented and comparatively evaluated the approach iRep3D for effective XML-based semantic 3D scene retrieval. The main contribution includes the fact that iRep3D outperforms the functional ontology based logical retrieval approach FB3D in both search precision and query response time. In addition, iRep3D achieves better search precision and recall than the RDF based multimedia retrieval approach RIR and the keyword query based 3D repository ADL. Besides, it scales to the semantic incremental indexing strategy as well as the sub-scene extraction for the semantic hybrid deep search of XML-based 3D scenes.

So far, we have presented the solutions for the four targeted research questions in the area of semantic-based search and composition in unstructured P2P networks. Parts of the implementation have been enhanced for use in two specific application scenarios in projects. These will be discussed in Chapters 7 and 8.

Chapter 7

Application to Virtual Product Design and Engineering

In recent years, more and more complex business processes are required and are being created for industrial design and production. Interconnected organizations and companies often work together for this purpose, based on their means for distributed data storage and exchange. 3D visualization and animation techniques are also relevant to these needs because of their capabilities for vivid production process illustration. Although the 3D Internet [14] [71] helps a user to immersively browse and interact with the 3D virtual or mixed reality world, support for user collaboration in 3D-based complex industrial design and production is very limited. This requires at least (i) a medium in which the 3D-based collaborative interaction between distributed users and their business processes can be performed; (ii) a method for reflecting the underlying business data flow with its real-time interactive 3D illustration; (iii) a means for fast and precise retrieval of 3D models in industrial processes, which are often annotated with advanced conceptual or functional descriptions besides their geometric features; and (iv) the framework for the integration of all of these.

To cope with the problems above, the work [421] has investigated the possible componential techniques and presented an integrated proof-of-concept demonstrative system that aims at building a workspace of an intelligent 3D virtual factory, towards P2P-based collaborative efficient 3D object retrieval and modeling. It is supported by the German BMB+F[1] research project Collaborate3D[2]. As one of its subsystems, the iRep3D repository (cf. Chapter 6) has been enhanced in particular for the near-real-time selection of semantically annotated 3D scenes. It includes a semantic 3D scene indexer that is able to automatically categorize annotated XML-based 3D scenes separately to the concept, service and geometric feature indices. On top of this, an on-line semantic query processor has been realized, which is able to answer the hybrid request for 3D scenes in near real-time. This is achieved by the parallel processing of the sub-queries in each aspect above and the fast sub-result

---

[1] BMB+F: Bundesministerium für Bildung und Forschung (Federal Ministry of Education and Research)

[2] `http://c3d.dfki.de/`

aggregation component. Further, the iRep3D sub-system contains a Web-based user interface, which allows users to semantically annotate XML-based 3D scenes in X3D, XML3D and COLLADA with conceptual descriptions and semantic services. It also provides an interface to answer hybrid queries for 3D scenes.

The remainder of this chapter is organized as follows: In Section 7.1, the background of the Collaborate3D project is introduced, and this is followed by a brief discussion of its collaborative 3D engineering system. The design and implementation of the enhanced iRep3D system is presented in Section 7.2. In addition, we discuss the integration of our semantic search strategy S2P2P (cf. Chapter 3) and iRep3D with collaborative virtual designer in Section 7.3. The related work on collaborative virtual production design will be discussed in Section 7.4, and this chapter is concluded in Section 7.5.

7.1 Collaborate3D Project Overview

The project Collaborate3D aims at research on and development of a coherent framework for scalable intelligent support and reliability for collaboration in the future 3D Internet. In detail, it includes the following targets:

- to provide scalable methods for ad-hoc and distributed collaborative 3D engineering of products via (agent-based) P2P search of XML3D models and their highly realistic 3D simulation in a shared 3D space.
- to enable nearly real-time built-in verifications of peer agents in order to help users to check the technical reliability of the collaboratively modeled production systems, regarding whether the virtual production in question fulfills the given technical requirements.
- to support the collaborative P2P-based planning of interactive services of simulated 3D objects in order to automatically form the needed workflow, which is always given by a user via a black box with properties, such as inputs, outputs, preconditions and effects.
- to design and develop a medium where such 3D based collaboration for industrial design and production can be performed. This consists of the method for real-time shared, interactive simulation of 3D models with their underlying business data exchange as well as the method for precise and efficient retrieval of 3D models in terms of their enriched conceptual and functional descriptions.

In the course of the Collaborate3D project, significant progress has been made [333], including:

- a collaborative Web-based editor for 3D scenes in XML3D, which supports configurable 3D animation;

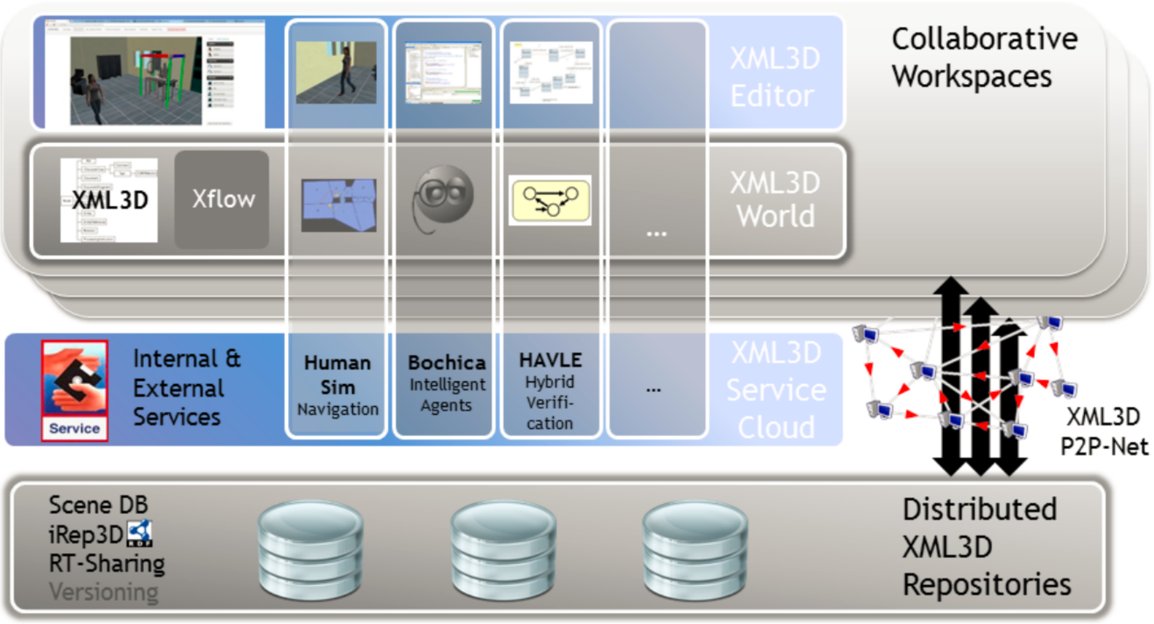


Fig. 7.1 Collaborate3D architecture [421].

- a client-server based 3D rendering system, which in particular provides mobile users with consistency and security guarantees;
- an approach for the collaborative search of semantically annotated 3D models;
- the adaptation of the HALVE [142] verification tool to the dynamic settings where participants configure 3D scenes in a collaborative way;
- an extension to the ISReal [181] agent component, which makes it possible to quickly add and configure agent-based avatars that navigate and orient themselves intelligently in virtual environments.

The achievements above are integrated as a proof-of-concept demonstrative system, called Collaborate3D. Working as the medium between end users and the product co-design system, its 3D virtual collaborative workspace (cf. Figure 7.1) allows designers to interact with and compose 3D models of machines' accessories. Besides, it supports Web browser based simulation of reality of 3D models from distributed provider organizations, which bridges the gaps created by system heterogeneity. Furthermore, it integrates different component systems, such as a 3D model editor, virtual process demonstrator, industrial process verification tool, 3D model repository, etc., which work together as a whole to support the collaborative virtual design of products. A crucial technique for achieving this is the XML3D 3D model language[3], which is based on the open declarative XML format and extensions of HTML Web page elements [336]. Nodes in the XML based description of 3D

---

[3] `http://www.xml3d.org/xml3d/specification/current/`

graphics are in DOM[4] tree representation. The latter is well-known as being accessible and changeable via JavaScript for advanced Web event processing and automation. In addition, the XML3D scenes are combined with dataflows, which is captured by Xflow [190]. Consequently, the manipulation of 3D models in terms of their formation can be reflected on the the underlying business data, and this also holds in the other direction. On top of these, 3D models in XML3D format can be created in the web-based interactive 3D editor. Each model is built starting from a static base geometry and users can select, add and move necessary 3D assets depending on their business. A virtual design of a small factory is demonstrated in Figure 7.2, where the editor contains the interactive 3D scene (left side); while the sidebar (right side) contains the options for the selection of 3D assets.

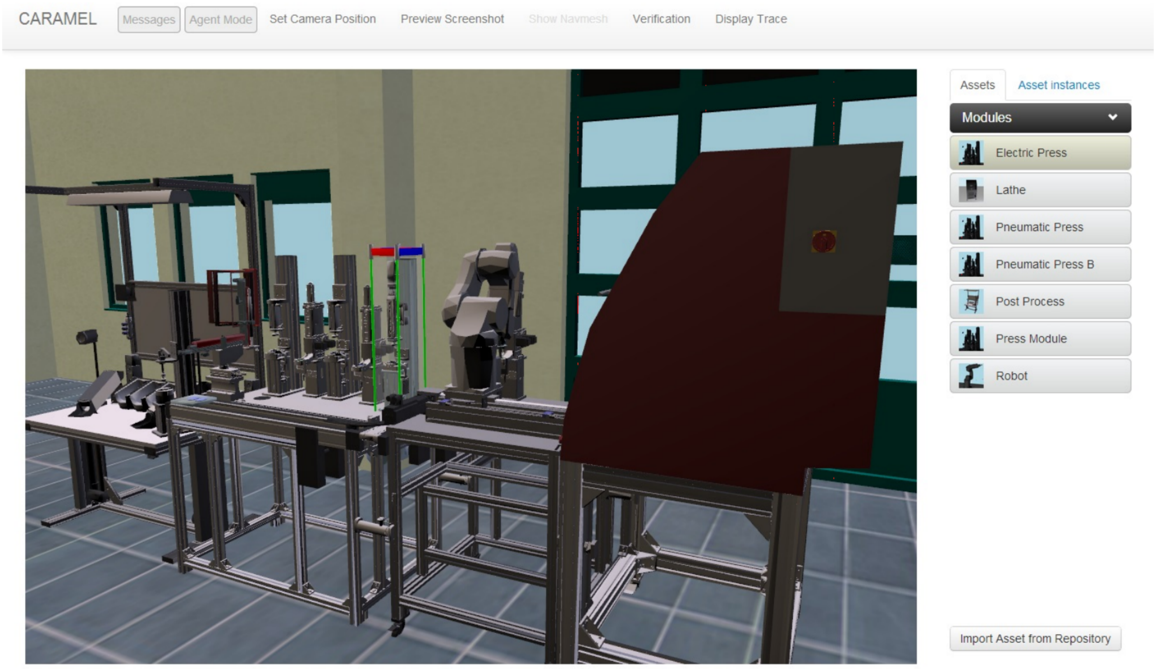


Fig. 7.2 Collaborate 3D model editor [421].

Another important factor of the collaborative 3D model designer is the storage and retrieval of 3D assets. This is captured by integrating the iRep3D repository, which provides the semantic annotation and near-real-time hybrid selection of 3D scenes in X3D, XML3D and COLLADA. Its integration with the 3D virtual editor is achieved by the scene query interface, which, for each 3D scene in a result list, sets up a link for exportation (cf. Figure 7.10).

In addition, our proposed semantic search strategy S2P2P can also be integrated into Collaborate3D system, towards the vision of P2P-based collaborative efficient 3D object retrieval and modeling. Working with iRep3D at each peer, S2P2P is expected to achieve higher search efficiency than its combination with non-index based item selection means. As a result, the collaborative 3D scene retrieval and modeling can be realized with larger community of designers in a more flexible way.

---

[4] `http://www.w3.org/DOM/`

The pre-visioned system will not only benefit from the fast hybrid selection of 3D scenes, but also behave highly robust against the dynamics of distributed 3D scene designer peers. We illustrate the architecture of S2P2P and iRep3D based collaborative 3D virtual editing system in Figure 7.3 and discuss the crucial parts of integration in Section 7.3. Fundamental building blocks, such as iRep3D, S2P2P, P2P framework (cf. Chapter 8), of it have been implemented. The practical integration of them has been planned as further work.

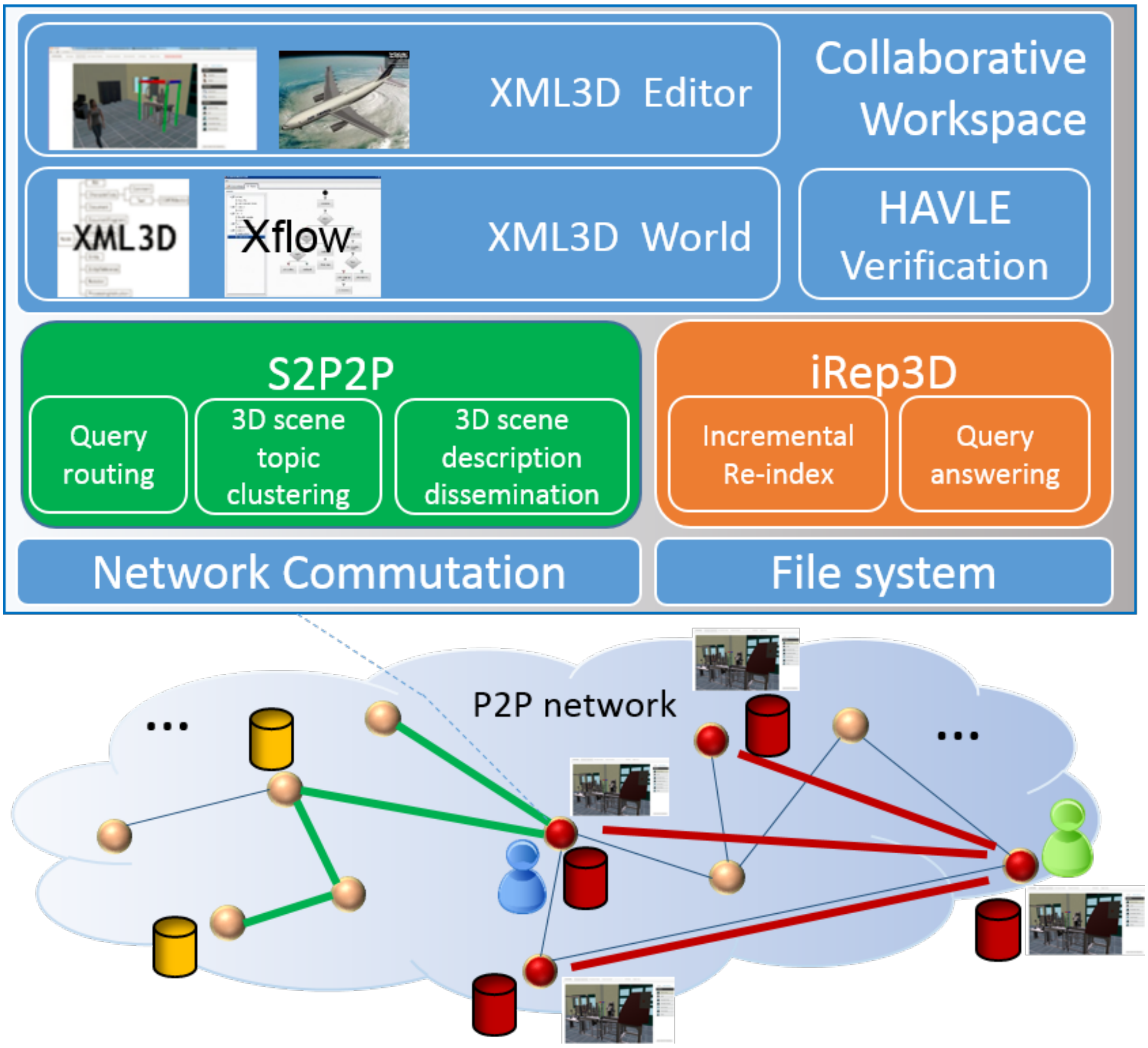


Fig. 7.3 Integration of S2P2P and iRep3D to Collaborate3D. Peers in red color form a working group via direct TCP connections. Amongst them, a XML3D editor frame is shared. The designer peer in the middle of the network topology issues a S2P2P query searching for necessary component 3D assets. Illustrated in green color, that query traverses non-member peers by the collaborative semantic query routing of S2P2P.

Depicted in Figure 7.3, the distributed collaborative virtual design environment allows for the extended search and joint design of 3D scenes. A designer working group can be formed in an ad-hoc fashion containing the participants who possibly come from different affiliations but aim at completing the same task and connect to an unstructured P2P network, where each peer shares 3D scenes and provides the fast hybrid selection that is realized by its local instance of iRep3D repository. Such an instance is competent to index the local 3D scenes and incrementally adapt to the change of semantic scene annotations, caused by observing or creating new scenes, deleting or updating scene descriptions. During the collaborative design for a task, designer peers in the working group share a common editor frame, which is the workspace of the task originator peer. For this, direct long term TCP connections are established, which result in a star-styled network overlay specifically for the current task. By networking facilities, any change of 3D scenes by members are reflected and synchronized inside working group. In addition, each participant is enabled to communicate with others through this interface. Moreover, each member is competent to issue S2P2P queries with hybrid scene annotation to the network and extensively share the selected results with other members. Such a query can traverse, by S2P2P routing strategy, those peers not necessarily within the current working group, as long as their shared 3D scenes are determined to be helpful for answering the query about the current design. On receiving a query, efficient local search is conducted with iRep3D, which adds the annotations of those matched 3D scenes into query answer set.

7.2 iRep3D for Collaborate3D

In this section, we present the design and important features of iRep3D that has been fully implemented and integrated as the 3D scene storage and management component in the collaborative virtual product design environment. The prototype of the iRep3D repository has been implemented in Java based on the Spring framework[5], Ehcache[6] and JPA 2.0[7] with Hibernate 3.0[8]. It also utilizes the XQuery[9]/XPath[10] facilities of the native XML database BaseX[11] for the (sub-)scene storage and extraction. iRep3D supports functionalities for hybrid semantic scene retrieval, new scene insertion and semantic annotation of 3D scenes via socket and servlet programmatic interfaces. Paging and result list pre-caching (as an option) are also provided for upper layer applications. On top of these, a Web-based user interface has been implemented based on HTML, Javascript and Ajax techniques. In the following, we introduce the architecture of iRep3D system as well as its Web based semantic annotation toolkit and user query interface.

---

[5] `http://www.springsource.org/spring-framework`
[6] `http://ehcache.org/`
[7] `http://jcp.org/en/jsr/detail?id=338`
[8] `http://hibernate.org/`
[9] `http://www.w3.org/TR/xquery/`
[10] `http://www.w3.org/TR/xpath20/`
[11] `http://basex.org/`

7.2.1 Architecture

Depicted in Figure 7.4, the architecture of iRep3D prototype consists of three layers. Application Layer supports the functions for end-user access. The current prototype offers Web- and socket-based query-answering and reality simulation. Interaction between end-user (or upper application) and repository is further enhanced by Paging Management Service that enables iRep3D prototype to return the sublist of retrieved 3D scenes on demand.

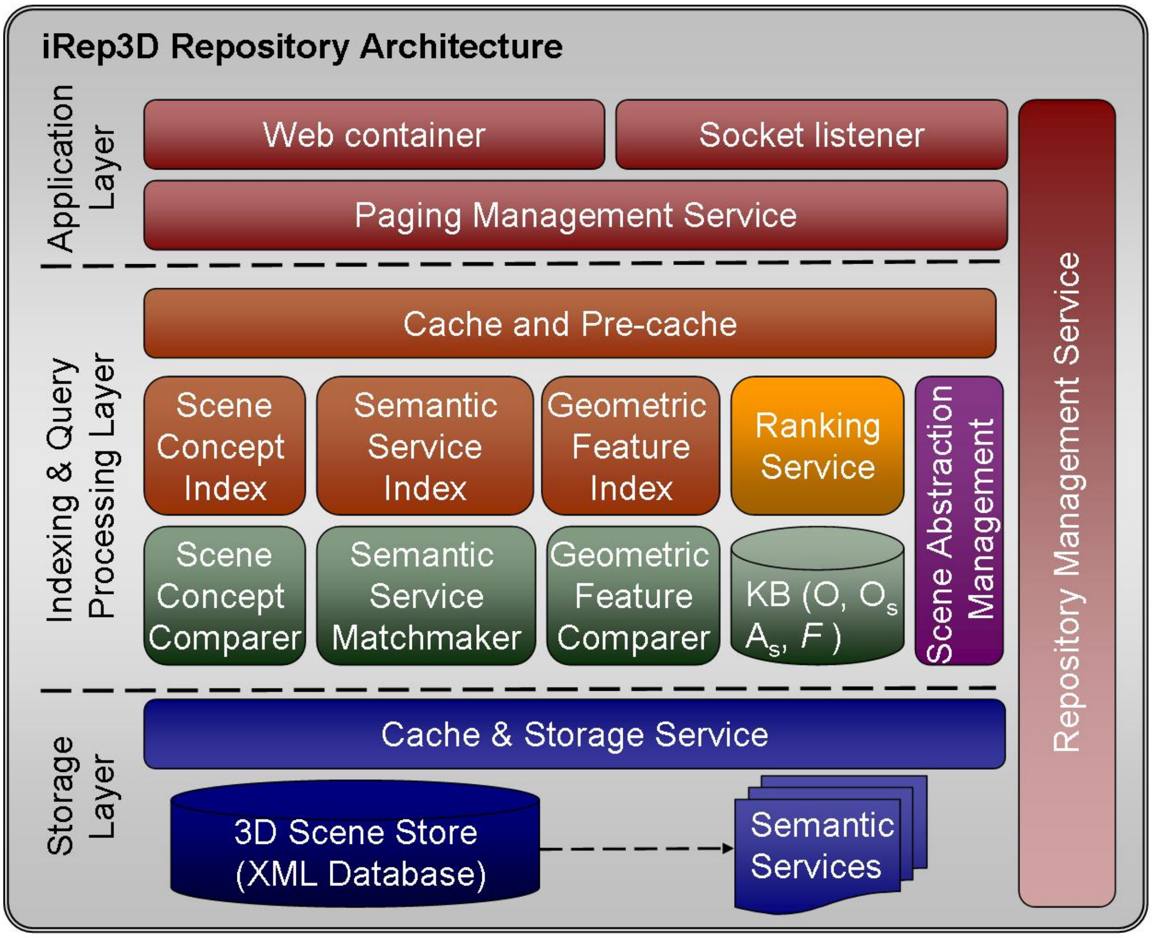


Fig. 7.4 iRep3D repository architecture.

A user query through Application Layer is answered by Indexing & Query Processing Layer. The latter performs three sub-query processing in parallel based on the scene concept index, semantic service index and geometric feature index. The partially relevant scenes in these aspects are further merged and re-ranked via Ranking Service, which supports the merging and Fagin's TA [92] operations. For answering a query, it is not sufficient to return the raw data of the final rank that merely contains the scene identifiers and relevancy scores. As a solution in the prototype, this layer is equipped with the Scene Abstraction Management component to maintain the abstract information of 3D scenes. The abstraction of a 3D scene consists of the syntactic description and brief textual information in terms of the concerned three matching aspects. Those information is extracted in line with the off-line indexing processes and supposed to be the answer content that is returned to the end-user according to the result ranking and scores. In addition, the fetched scene abstractions for received queries are cached under least recent used (LRU) policy under maximal idle time restriction. Further, by the paging information of queries provided by the upper Paging component, this layer provides the optional facility to pre-caching the list of scene abstractions, which might be requested for the next page of some query.

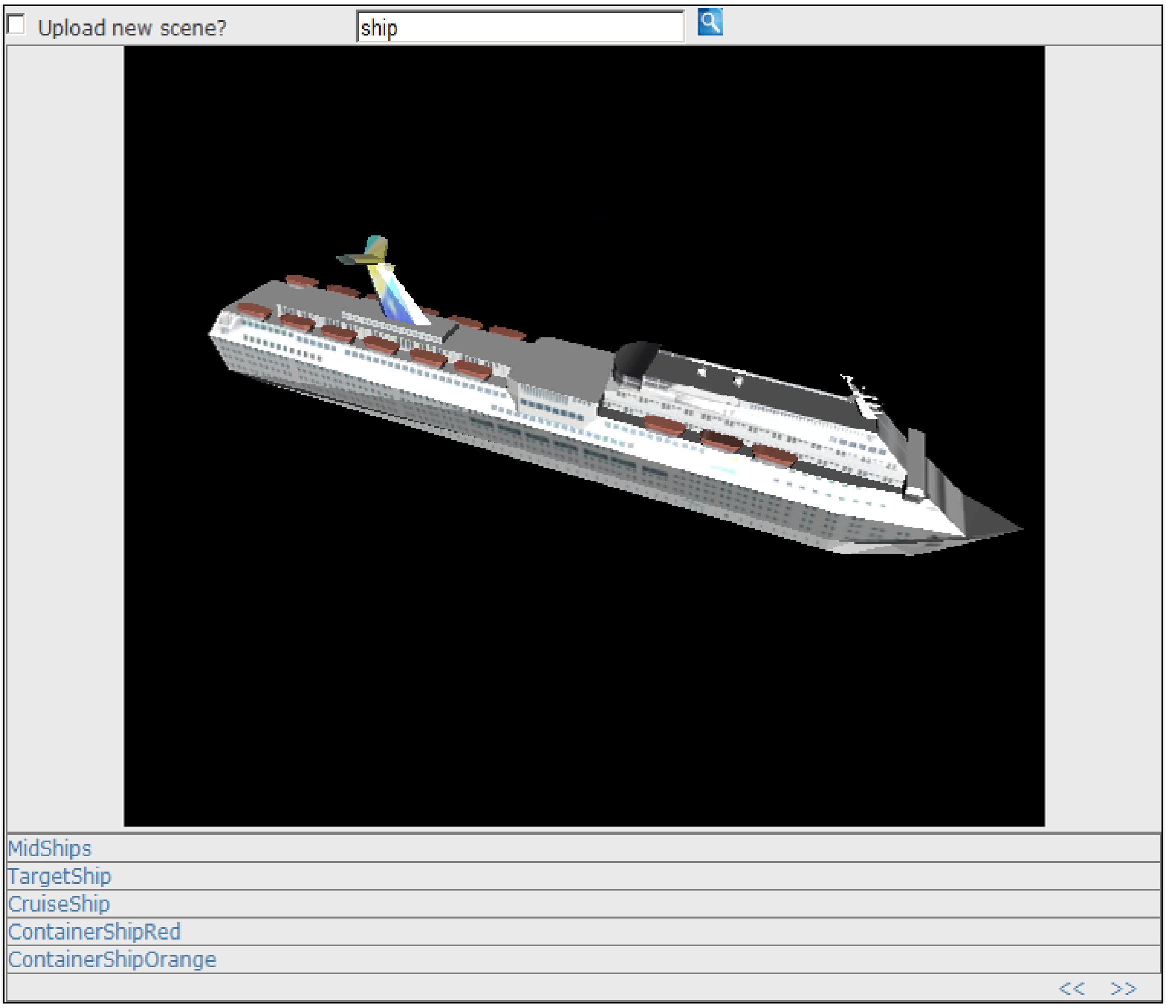
The indexes are pre-constructed off-line by means of independently invoking the functionalities of Scene Concept Comparer, Semantic Service Analyzer and Geometric Feature Comparer over the maintained 3D scenes. Logical reasoning conducted during the comparison relies on the repository knowledge base and realises the proposed approximated concept similarity measure based on the scene concept and service ontology. Besides the predicate sets used by the indexing of service preconditions and effects, geometric feature types extracted from XML-based description language schemas are also maintained by the knowledge base. The index in each aspect consists of a set of ranked lists, and each of them maintains the ordered and scored scenes that are determined to be relevant with a concerned feature (e.g. A conceptual scene category, a service input parameter type or a geometric attribute). Each list of scenes is persisted as a file on disk, which is further used for query processing by a lazy-loading mechanism. That is to load the top T entries (T is configured with 50 in the prototype) into memory during system initialization and reads the next T entries when they are needed. Moreover, the extracted features of 3D scenes are organized as scene abstracts that are further stored in a MySQL[12] database for the purpose of showing 3D scene details (not scene selection) on demand. By means of pooling the connections to database, the proposed solution above performs quicker than the XML-based on-line query directly on 3D scene files.

As the most fundamental component, the Storage Layer supports XPath and XQuery access to the XML-based scene description files for the purpose of indexing, user browsing or simulated reality. This functions are wrapped by a Storage Service that exposes a unified interface to upper layers, which hides the complexity of underlying database and the diversity of various XML schemas of 3D scenes. For those semantic services whose description profiles and groundings reside in remote servers, this layer also supports the functionalities for service profile synchronization, maintenance and analysis. Similar with Indexing & Query Processing Layer, LRU-based caching is also enabled in this layer. Offering fast XML file access, the implementation of caching mechanism is embedded in the Storage Service. This facility unifies the diverse configuration of the possible caching components of different XML databases.

Three layers of iRep3D repository are implemented in a loose-coupled fashion. They can be deployed in separate servers. Each of them is able to communicate with the others via IP sockets. To increase their throughput, each layer maintains a first-in-first-out (FIFO) request cache. Pooled acceptor threads put their received requests from upper layer to the cache and the processor threads

---

[12] `http://www.mysql.com/`



☐ Upload new scene?

ship

MidShips

TargetShip

CruiseShip

ContainerShipRed

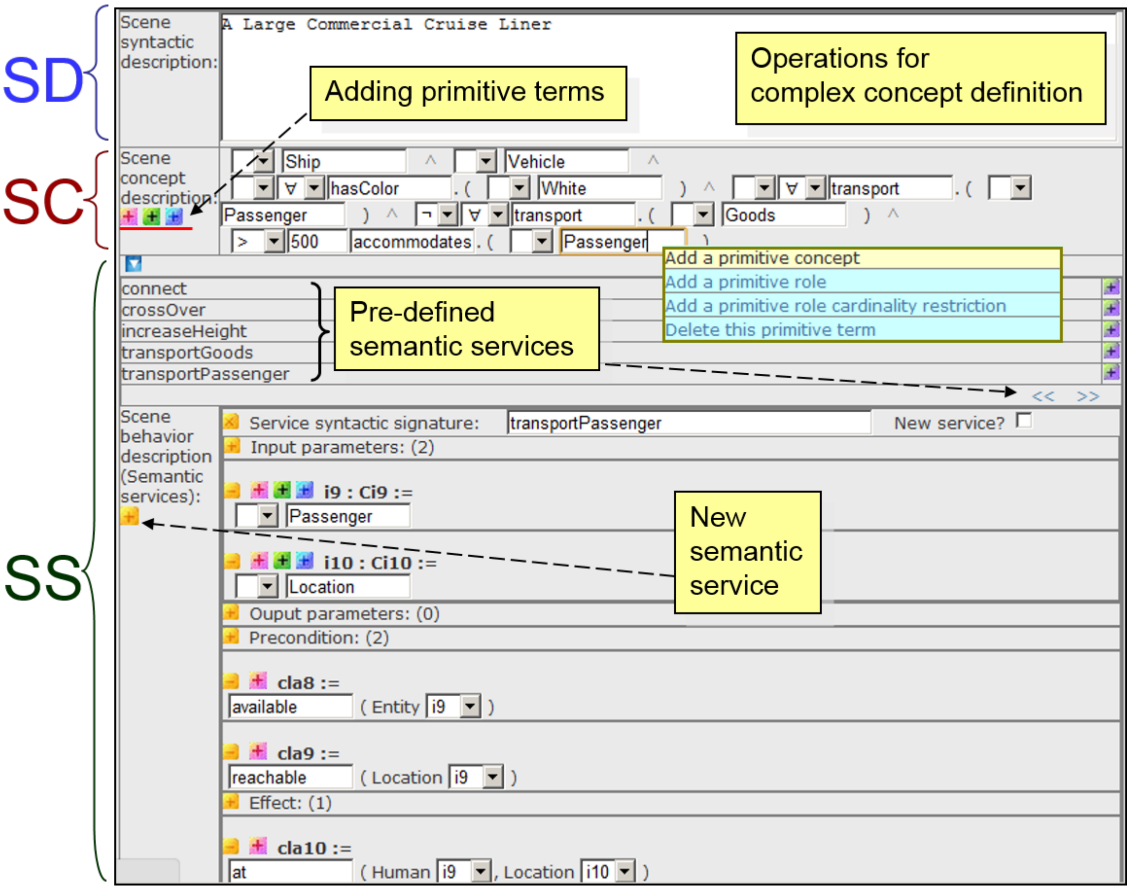
ContainerShipOrange

Fig. 7.5 The 3D scene browser of iRep3D system.

serve the requests out of the cache. A query is processed by three joined threads in parallel. Each of them responses to the searching of scenes in conceptual, semantic service or geometric feature aspect. In addition, a result caching heuristics has been implemented, it enables the system to store the selected scene files and abstracts that would be used on demand. The list of selected 3D scene identifiers is also cached in Query Processing layer. When a page is displayed, the contents for the next one are preloaded. For speeding up the simulation of reality, 3D scene files are (pre-) cached if they were (will be) requested (by the next page).

7.2.2 Semantic Annotation Toolkit for 3D Scenes

Besides the geometric features, XML-based 3D scenes on the Web are commonly annotated with mere text descriptions and meta-properties such as author, version, publish time, etc.. Scenes with logic-based annotations like conceptual description and semantic services are rare, possibly because their creation requires the expertise in the area of logics and the Semantic Web. Nonetheless, those annotations make sense in terms of more precise retrieval and automated 3D scene composition. For bridging this gap, a scene annotation toolkit has been developed for iRep3D system. It contains a simple 3D scene browser (cf. Figure 7.5), which allows a user to annotate new 3D scenes or update the annotation of a scene that is already contained in the repository. The latter case is supported by means of a wildcard search over scene file names. In particular, the toolkit supports scene previews for XML3D and X3D scenes[13].



Scene syntactic description: A Large Commercial Cruise Liner

Scene concept description:
Ship ^ Vehicle ^
∀ hasColor . (White) ^ ∀ transport . (
Passenger) ^ ¬ ∀ transport . (Goods) ^
> 500 accommodates . (Passenger)

connect
crossOver
increaseHeight
transportGoods
transportPassenger

Scene behavior description (Semantic services):
Service syntactic signature: transportPassenger New service? ☐
Input parameters: (2)
i9 : Ci9 :=
Passenger
i10 : Ci10 :=
Location
Ouput parameters: (0)
Precondition: (2)
cla8 :=
available (Entity i9)
cla9 :=
reachable (Location i9)
Effect: (1)
cla10 :=
at (Human i9 , Location i10)

Fig. 7.6 The 3D scene annotation toolkit of iRep3D system.

Once a 3D scene is selected, annotations in terms of syntactic description, semantic scene concept and semantic services can be added via the annotation panel (cf. Figure 7.6). It offers plain text annotation (labeled *SD* in Figure 7.6) for syntactic descriptions. For scene concept (labeled *SC* in Figure 7.6) annotation, it provides the facility for concept definition in negation normal form based on a set of pre-defined primitive terms. For this purpose, typing suggestion is enabled when a user inputs the signature of a primitive concept, role or role cardinality restriction. In addition, it provides the flexibility for defining arbitrarily complex concept, which might have nested primitive terms in a

---

[13] The preview for 3D scene in COLLADA has not been enabled in Web-based toolkit due to the lack of browser plugin for display.

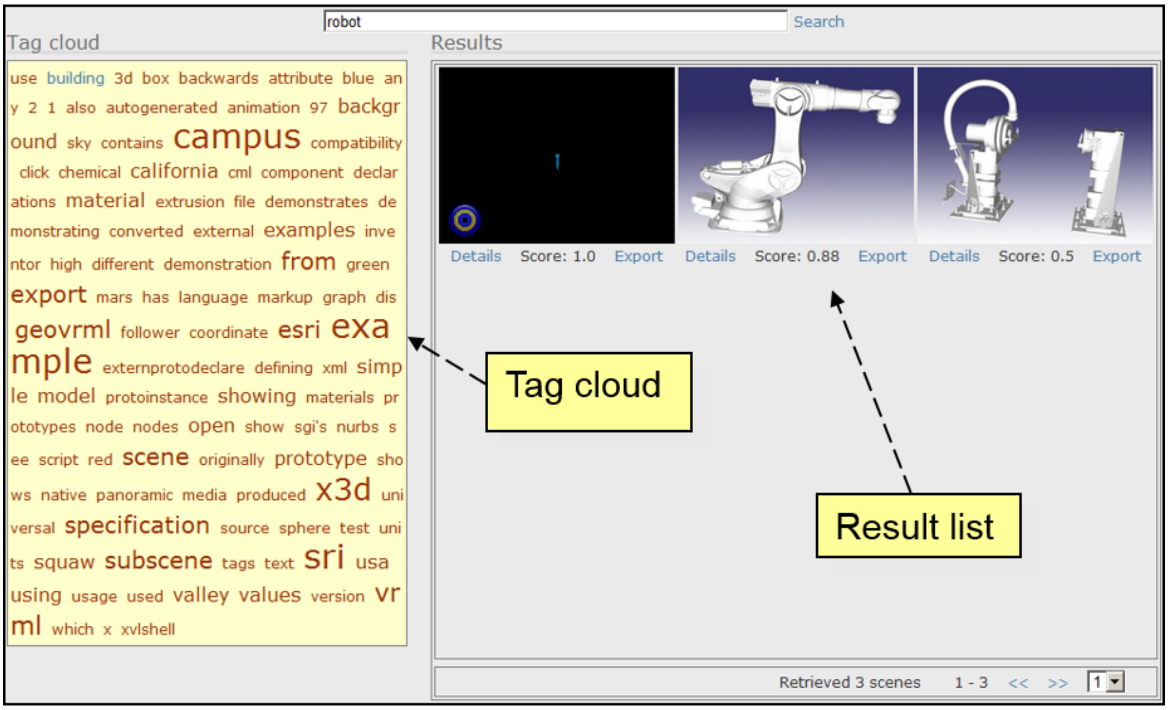


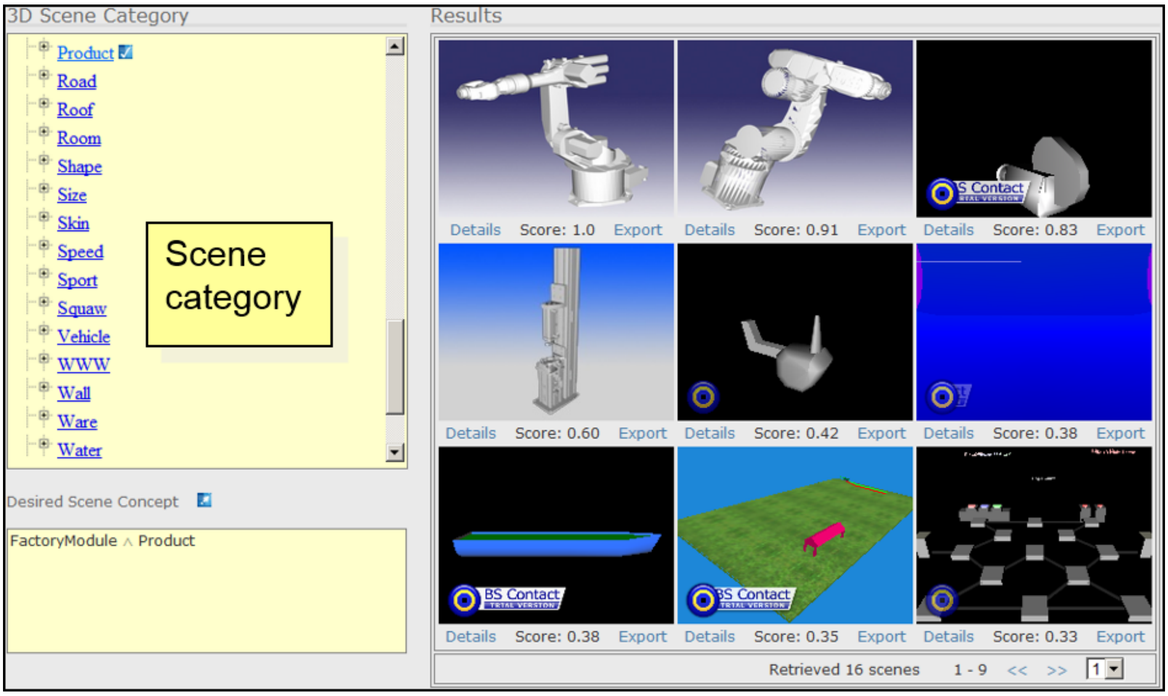
Fig. 7.7 Web based interface of iRep3D query processing (1).

role range field. This can be done by popping up the right click menu inside the concerned role range and choosing the desired option. For describing the behavioral features of 3D scenes (labeled *SS* in Figure 7.6), a user can choose pre-defined services and/or specify new semantic services in terms of service name and signature in terms of input, output parameters precondition and effects. Besides, new services can be stored for reuse in the future. For specifying the service parameter type, the same facility of scene concept specification is offered; while the precondition/effect in conjunctive normal form can be also built up via choosing predicates (typing suggestion) inside any disjunctive clause together with its variables, which have been automatically generated during the service parameter configuration.

7.2.3 User Query Interface

Hybrid semantic search of 3D scenes in terms of their syntactic description, scene conceptual description, semantic services and geometric features is fully supported by the Web-based user interface. In particular, a user is able to build complex queries containing any subset of the four covered search dimensions in an augmenting fashion. The configuration of one dimension of a query will be considered together with the dimensions that might have been set before. Consistent result ranked list are returned regardless of the order in which a user builds her request over those dimensions.

As shown in Figure 7.7, iRep3D accepts syntactic queries based on text indices of 3D scenes, which additionally provides a user with the suggestion via a Tag cloud. The latter is mined based



FactoryModule ∧ Product

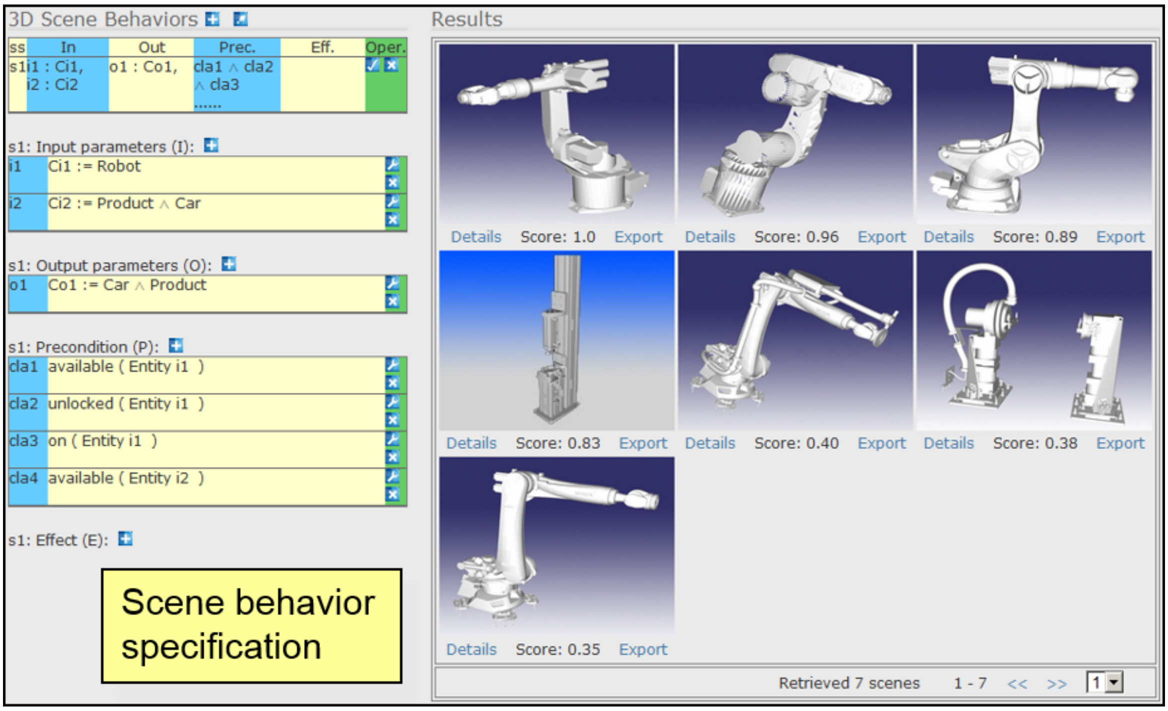
Details Score: 1.0 Export | Details Score: 0.91 Export | Details Score: 0.83 Export

Details Score: 0.60 Export | Details Score: 0.42 Export | Details Score: 0.38 Export

Details Score: 0.38 Export | Details Score: 0.35 Export | Details Score: 0.33 Export

Retrieved 16 scenes 1 - 9 << >> 1

Fig. 7.8 Web based interface of iRep3D query processing (2).



ss	In	Out	Prec.	Eff.	Oper.
s1	i1 : Ci1, i2 : Ci2	o1 : Co1,	cla1 ∧ cla2 ∧ cla3		✓ ✗

s1: Input parameters (I):

i1	Ci1 := Robot
i2	Ci2 := Product ∧ Car

s1: Output parameters (O):

o1	Co1 := Car ∧ Product

s1: Precondition (P):

cla1	available (Entity i1)
cla2	unlocked (Entity i1)
cla3	on (Entity i1)
cla4	available (Entity i2)

s1: Effect (E):

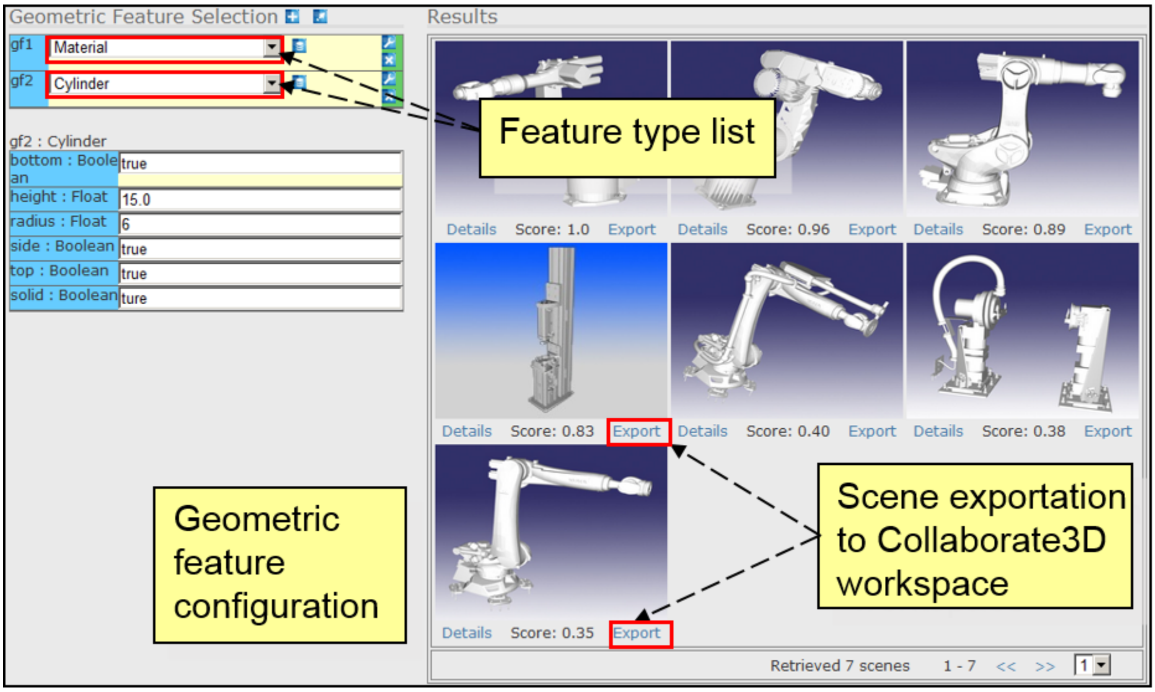
Details Score: 1.0 Export | Details Score: 0.96 Export | Details Score: 0.89 Export

Details Score: 0.83 Export | Details Score: 0.40 Export | Details Score: 0.38 Export

Details Score: 0.35 Export

Retrieved 7 scenes 1 - 7 << >> 1

Fig. 7.9 Web based interface of iRep3D query processing (3).



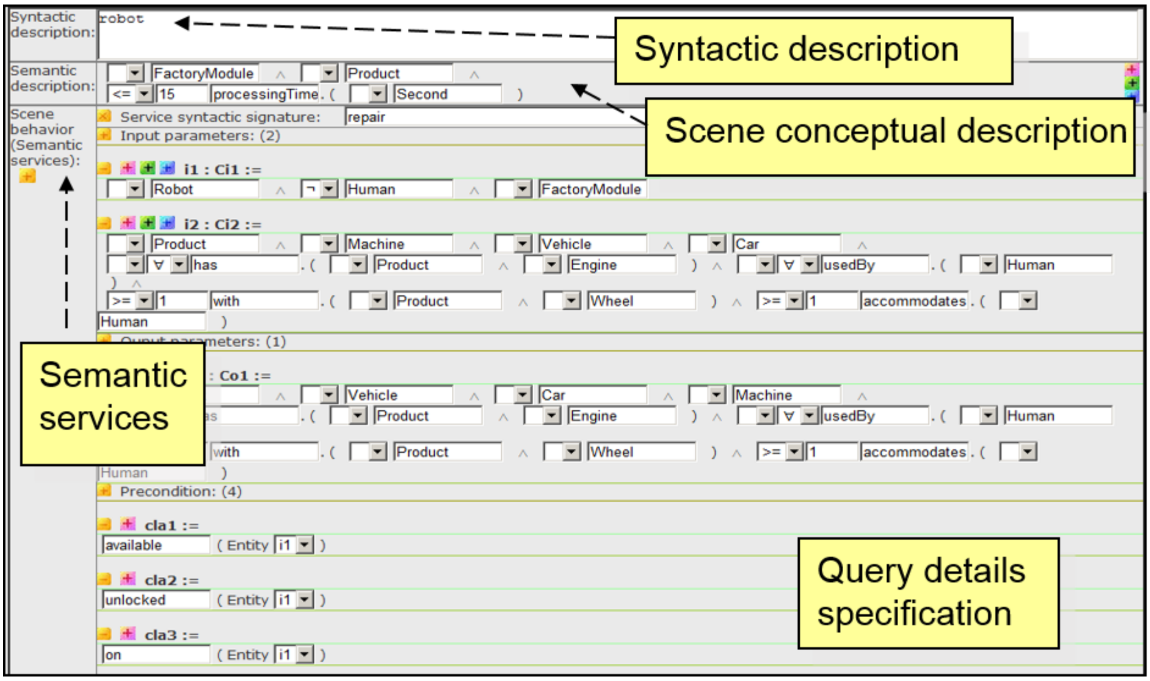
gf1 Material
gf2 Cylinder

gf2 : Cylinder
bottom : Boolean true
height : Float 15.0
radius : Float 6
side : Boolean true
top : Boolean true
solid : Boolean ture

Score: 1.0
Score: 0.96
Score: 0.89
Score: 0.83
Score: 0.40
Score: 0.38
Score: 0.35

Retrieved 7 scenes 1 - 7 1

Fig. 7.10 Web based interface of iRep3D query processing (4).



Syntactic description: robot

Semantic description: FactoryModule ∧ Product ∧
<= 15 processingTime . (Second)

Scene behavior (Semantic services):
Service syntactic signature: repair
Input parameters: (2)

i1 : Ci1 :=
Robot ∧ ¬ Human ∧ FactoryModule

i2 : Ci2 :=
Product ∧ Machine ∧ Vehicle ∧ Car ∧
∀ has . (Product ∧ Engine) ∧ ∀ usedBy . (Human) ∧
>= 1 with . (Product ∧ Wheel) ∧ >= 1 accommodates . (
Human)

Output parameters: (1)

: Co1 :=
∧ Vehicle ∧ Car ∧ Machine ∧
s . (Product ∧ Engine) ∧ ∀ usedBy . (Human
with . (Product ∧ Wheel) ∧ >= 1 accommodates . (
Human)

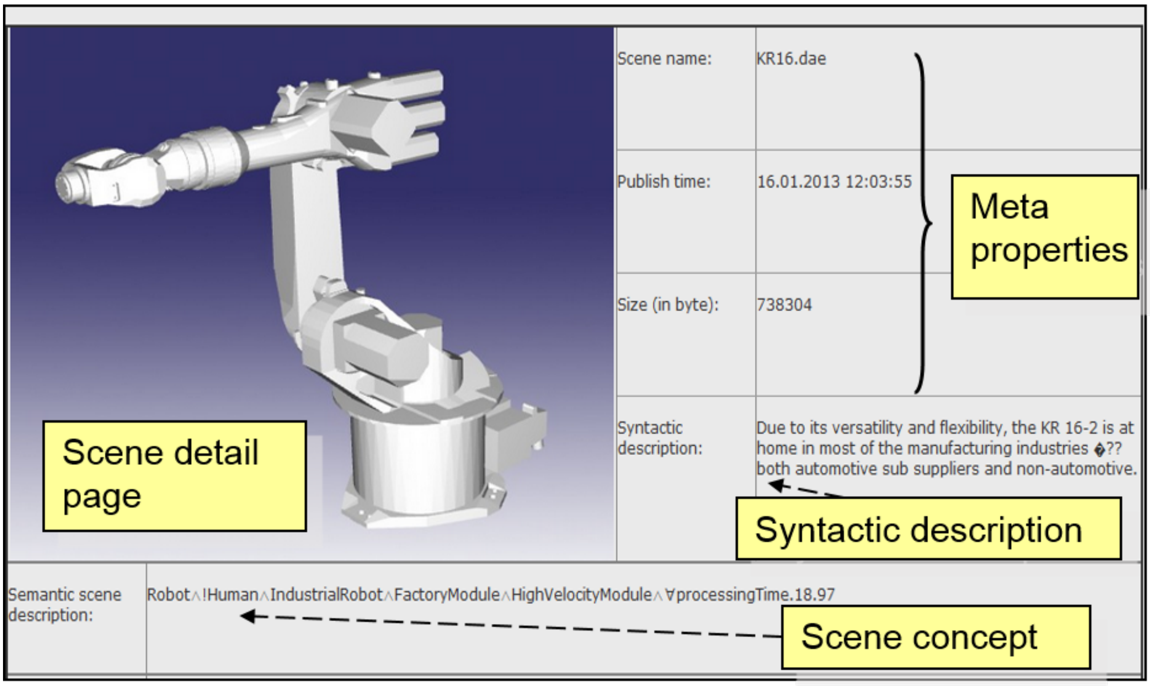
Precondition: (4)

cla1 :=
available (Entity i1)

cla2 :=
unlocked (Entity i1)

cla3 :=
on (Entity i1)

Fig. 7.11 Web based interface of iRep3D query processing (5).



Scene name: KR16.dae

Publish time: 16.01.2013 12:03:55

Size (in byte): 738304

Syntactic description: Due to its versatility and flexibility, the KR 16-2 is at home in most of the manufacturing industries �?? both automotive sub suppliers and non-automotive.

Semantic scene description: Robot∧!Human∧IndustrialRobot∧FactoryModule∧HighVelocityModule∧∀processingTime.18.97

Fig. 7.12 Web based interface of iRep3D query processing (6).

on the term frequency of text descriptions of 3D scenes. The result list offers the interactive preview of matched XML3D and X3D 3D scenes. Besides, the display of each selected scene includes the aggregated score and two links; one for scene details and one for export (download). For the scene conceptual description aspect of a query(cf. Figure 7.8), it offers category-based search. Based on the scene concept ontology, a tree consisting of the pre-defined scene category hierarchy is created. A user is able to select 3D scenes by not only one category but also a conjunction of multiple categories. The selection in terms of 3D scene interactive behavior is captured by the service specification panel in Figure 7.9. The user is able to specify any number of services in terms of input/ouput parameters and precondition/effects, which her desired scene should provide. The service parameter concepts and predicates for condition-based behavior selection are predefined in service ontology, respective, predicate set. The setting of geometric features of a query is covered by the geometric feature configuration panel (cf. Figure 7.10). Multiple geometric features can be configured at feature attribute value level. For this purpose, geometric features defined in XML3D, X3D and COLLADA specifications are extracted in advance. These feature types with their attributes including data type restrictions are pulled from the server once the user switches to this panel. By activating a feature instance, the corresponding list of attributes with data type constraints are shown. Further, iRep3D provides detailed query specification facilities (cf. Figure 7.11) for advanced users who have the knowledge of computational logic and Semantic Web. More precise configuration in terms of scene concept and behavior can be achieved by inserting, updating and deleting primitive terms, in particular, roles and role cardinality restrictions. Finally, the details of a selected scene can be shown by the "scene detail page" (cf. Figure 7.12).

7.3 iRep3D and S2P2P for Collaborate3D

In this section, we discuss the integration of iRep3D with S2P2P search for the P2P-based collaborative 3D scene virtual product design. As depicted in Figure 7.3, the potential combined use of them can be built on top of an unstructured P2P network and provide efficient hybrid search for semantically annotated 3D scenes. For this purpose, each peer runs an instance of iRep3D repository for the fast selection of its local 3D scenes. Besides the geometric features, each scene is annotated additionally with conceptual and functional descriptions, as introduced in Chapter 6. In order to apply S2P2P for the search of 3D graphics in this kind, several necessary adaptation should be made on S2P2P search strategy. We discuss them in the sequel of this section and leave the practical implementation as future work.

Preliminaries and assumptions: S2P2P search strategy is designed to be agnostic to the type of data. As for the annotated 3D scenes introduced in Chapter 6, we use the semantic description field *sd* (cf. Definitions 1 and 2 in Chapter 3) in an item or query to record the features in the concerned three dimensions. The definitions of 3D scene item and query are reformulated as follows:

Definition 1: *Semantically annotated 3D scene in P2P-based collaborate3D.*
An *3D scene item i* provided and maintained by peer p is defined by the tuple $i = \langle td, sd, URI, pid, isz, n_s, da \rangle$ where td is the abbreviated name of i (e.g. "robot A") used in the communication of collaborative design; sd the semantic annotation of i based on the local knowledge base KB_p of peer p. $sd = \langle C, SS, GF \rangle$, which conceptual description C semantic services SS and geometric feature set GF conform to the definitions in Chapter 6; URI the scene identifier; pid the id of the owner peer p providing i (including $i.da$); da the scene data, including the XML-based description files and also those referenced resources, like pictures, sounds, etc.; isz the size of $i.da$; and n_s the number of available copies of i at p. The scene tuple without da is meant with the scene metadata and called *scene description* ($i.desc$) of i. ■

We assume that peers in the P2P-based collaborate3D system share a vocabulary of primitive terms. On top of it, each peer canonically builds its local knowledge base, which consists of a local scene ontology O_p, a service ontology O_{sp} used to define scene service parameter types, and a set of predicates. In addition, let us assume that all peers use the same standard 3D geometric features defined in XML3D, X3D and COLLADA specifications.

Definition 2: *Semantic query for 3D scenes in P2P-based collaborate3D.*
A *3D scene query q* is defined by the query tuple $q = \langle td, sd, req, A, Pa, Pa_{sug}, t, st, pbd, TTL, n_d \rangle$ where td denotes the text-based label used during the collaborative 3D model design; sd the seman-

tic annotation of requested 3D scene. $sd = \langle C, SS, GF \rangle$, which conceptual description *C* semantic services *SS* and geometric feature set *GF* conform to the definitions in Chapter 6; *req* the identifier of the requester peer; $A = \{(res, its)\}$ the actual answer set for the query at run time. It consists of a set of pairs. Each maps an identifier *res* of a responder peer to an array *its* of its answered scene descriptions; *Pa* the path of this query; Pa_{sug} the suggested path, which consists of a sequence of peers for routing q afterwards. It is empty, if no suggestion is available or q is backtracking; t the query issuing time; $st \in \{Issued, Success, Fail\}$ the query status where *Success* (*Fail*) means that the query q is satisfied (unsatisfied) and *Issued* indicates that the satisfaction of q has not been determined by the original requester peer *req* yet (or else that q is not issued by the current peer); *pbd* the piggybacked data set; TTL the query time-to-live value; n_d the requested number of copies of the desired 3D scene. ■

Peer local observation on 3D scenes: The peer observation of new 3D scenes (by scene URI) from a query triggers the local incremental indexing as well as the adjustment of the clusters on supply and demand topics. The former is conducted by iRep3D for scene selection (cf. Section 6.4.2); while the latter is performed according to S2P2P protocol for identifying peer expertise (cf. Section 3.3) or demands, which clusters the new observation into supply (demand) topics by considering the conceptual description services and geometric features of 3D scenes.

S2P2P provides a generic similarity function for each peer to compare the topic of items or queries in order to cluster them as the local knowledge. The topic in the context of P2P-based collaborate3D concerns all three dimensions. The topic similarity is measured by the convex combination of the similarity scores in three dimensions, formulated in Equation 7.1.

$$\begin{aligned} sim(sd,\ sd') = \quad & a \cdot sim_c(sd.C,\ sd'.C) + \\ & b \cdot sim_s(sd.SS,\ sd'.SS) + \\ & c \cdot sim_g(sd.GF,\ sd'.GF). \end{aligned} \tag{7.1}$$

where *sd* is description of a 3D scene (request) that has been already clustered; sd' is the new description observed (requested); Assigned by users or learned, a, b and c are positive real values ($a+b+c=1$) for differentiating the weights of the dimensions (We will discuss the way of choosing them later on.); $sim_c(sd.C,\ sd'.C)$ is the maximum concept abduction-based similarity score between two scene concepts (cf. Section 6.1.2).

$$sim_c(sd.C,\ sd'.C) = max(s_{ab,\sqsubseteq}(sd.C,\ sd'.C), s_{ab,\sqsubseteq}(sd'.C,\ sd.C)). \tag{7.2}$$

$sim_s(sd.SS, sd'.SS)$ is the similarity score of two sets of services. It is the averaged score of the best bipartite matching between the two service sets. To compute this, let us denote SS^* the one out of the two service sets, which has the minimum number of services. Each service $S \in SS^*$ is matched with a service S' in other set, which yields a weight value $w(S,\ S')$ of the edge between them. $w(S,\ S')$ is their maximum service matchmaking score under the hypothesis that S plugs-into S' or in the other way around. For computing matchmaking score, we adapt the iSeM matchmaker [200] by using the approximated concept similarity score in Section 6.1.2.

$$\begin{aligned} sim_s(sd.SS,\ sd'.SS) &= \frac{1}{|SS^*|}\Sigma_{S\in SS^*}\, w(S,\ S'); \\ w(S,\ S') &= max(sim_{plug-in}(S,\ S'),\ sim_{plug-in}(S',\ S)). \end{aligned} \tag{7.3}$$

$sim_g(sd.GF,\ sd'.GF)$ at geometric feature level is the averaged similarity value of matched features. Let us denote GF^* the one out of the two geometric feature sets, which has the minimum number of features. Each geometric feature $gf \in GF^*$ matches a feature gf' in the other set. Since geometric feature types follow the shared standards, the matching between two sets of geometric features can be computed by using the primitive attribute similarity measure $s_k(v_1, v_2)$ proposed in Section 6.3.1, where v_1 (v_2) is the value of an attribute in gf (gf').

$$\begin{aligned} sim_g(sd.GF,\ sd'.GF) &= \frac{1}{max(|sd.GF|,\ |sd'.GF|)}\Sigma_{gf\in GF^*}(sim_f(gf,\ gf')); \\ sim_f(gf,\ gf') &= \frac{1}{|K_{gf}|}s_k(v_1,\ v_2). \end{aligned} \tag{7.4}$$

If a geometric feature $gf \in GF^*$ does not match any one in the other set, $s_f(gf,\ \cdot) = 0$.

Based on the similarity measure, each designer peer with S2P2P is able to incrementally classify the observed 3D scenes or queries as local knowledge about the expertise or demands of other peers. Relying on this, the path suggestion-based query routing strategy can be proceeded.

Query routing: During the collaborative designing of 3D model, each member of the working group can issue queries to search desired scenes. According to S2P2P, the core part of query routing is the path suggestion. Instead of mere a neighbor peer, a path suggestion contains multiple expert peers, which yield the minimum total inverse expertise gain per traffic cost under the TTL limit. To do this, a necessity is to compute the expertise gain of routing a query q to a peer p' whose expert topic ts' has been observed (cf. Equation 3.5). The similarity between q and ts' is estimated by the measure proposed above. For routing q, the requester peer executes Alg. 2 (cf. Section 3.4) to compute up to K path suggestions, one for each walker; while a non-requester peer runs Alg. 1 (cf. Section 3.4) to suggest one path for the current walker. Please note that selected expert peer is not necessary in the current working group for designing a 3D model. Any peer in the network can be chosen, as long as its expertise is determined to be relevant to query.

3D scene description dissemination: Besides the query routing, P2P-based collaborate3D system can also benefit from the S2P2P-based 3D scene description dissemination. Without exchanging extra messages, each peer p, before sending out a query q, determines the destination peers on the (suggested) path, to which the description of a known 3D scene should be propagated. For this purpose,

p estimates the utility values (cf. Equation 3.9) based on the similarity between its observed demand topic of the candidate destination and the supply topic of its known 3D scenes. Likewise, this can be estimated by measure above.

Discussion: An issue of integrating S2P2P and iRep3D for collaborate3D virtual 3D model design is the large network traffic load that is caused by the transmission of 3D geometric features (e.g. complex mesh). Alternative solutions include (1) to discard the geometric feature aspect in search, or (2) to limit the geometric features that can be transmitted, depending on the network bandwidth. For case (1), the query originator does not include the geometric features of desired scene in her query. Once relevant scenes in terms of conceptual and functional aspects are found, the originator can contact the scene provider to obtain the geometric feature data. For case (2), it is possible to restrict that only the geometric features of the query can be transmitted; while each responder peer attaches the geometric feature data size and geometric similarity score of relevant scenes to query answer set. Once determined to be used, the geometric feature data can be requested afterwards by the requester.

Another issue is the selection of the weight values a, b and c in Equation 7.1. Solutions include (1) to let user assign or (2) to learn at runtime or with labeled queries before system initialization. For (1), we distinguish the cases based on that for what purpose the similarity function is used. If the comparison is used by the topic clustering during peer observation or the scene description dissemination processes, the weights can be assigned by the peer owner herself; while if the function is used by query routing or the aggradation of iRep3D search on each peer, the query originator can assign them for specific concerns. For (2), a straightforward way is to train a SVM classifier before the system starts. To do this, a set of labeled queries is needed. Another option is to let query originator to assign the preferred weights and each peer learns the values for topic clustering and scene description dissemination at runtime. For this, each peer adjusts its current local weight value by regression analysis that considers the requester assigned values in the latest N ($N \in \mathbb{N}^+$) observed queries.

Example: In Figure 7.13, we show an illustrating example to give the intuition about the working process of the visioned distributed collaborative 3D product design with S2P2P and iRep3D. In Figure 7.13a, user 1 connects to the unstructured P2P infrastructure of a 3D virtual product design community where his project partners (users 2, 3 and 4) have joined. Each participant in this network is presumed to have installed the S2P2P and iRep3D-based collaborate3D software suite. When user 1 starts his design, he contacts with our partners and shares his XML3D model editing frame with them (cf. Figure 7.13b) via direct long term TCP connections (lines in red). Their connections forms a star-styled network overlay exclusively for completing the current task. During their design, each user leverages its local iRep3D repository for the storage and retrieval of 3D scenes. For accomplishing the task, user 3 finds out that a 3D object is needed but she and other working group members do not have. She then issues a S2P2P query q ($K = 2$, $TTL = 3$) to search in the network (cf. Figure 7.13c). Let us assume that the peer of user 3 knows about some relevant 3D model at peer 7. Accordingly, S2P2P strategy at peer 3 suggests two paths for routing q. One walker is directed to peer 7 and the other is routed to peer 5 by random. Further, we assume that the peer 5 knows about the expertise of peer 6, which match the queried topic by peer 3. Peer 5 re-computes the path suggestion and directs the walker to peer 6 via the shortest path $peer5 \rightarrow peer6$ (lines in green). As a result, peer 3 obtains her desired 3D model[14]. Subsequently, she properly inserts the new object to the current design project locally. Further, peer 3 contacts with the design originator user 1 by sending the data of the current 3D scene in progress (red arrow in Figure 7.13d). Once receiving it, user 1 then notifies and synchronizes his design frame with the other two designers (brown arrows).

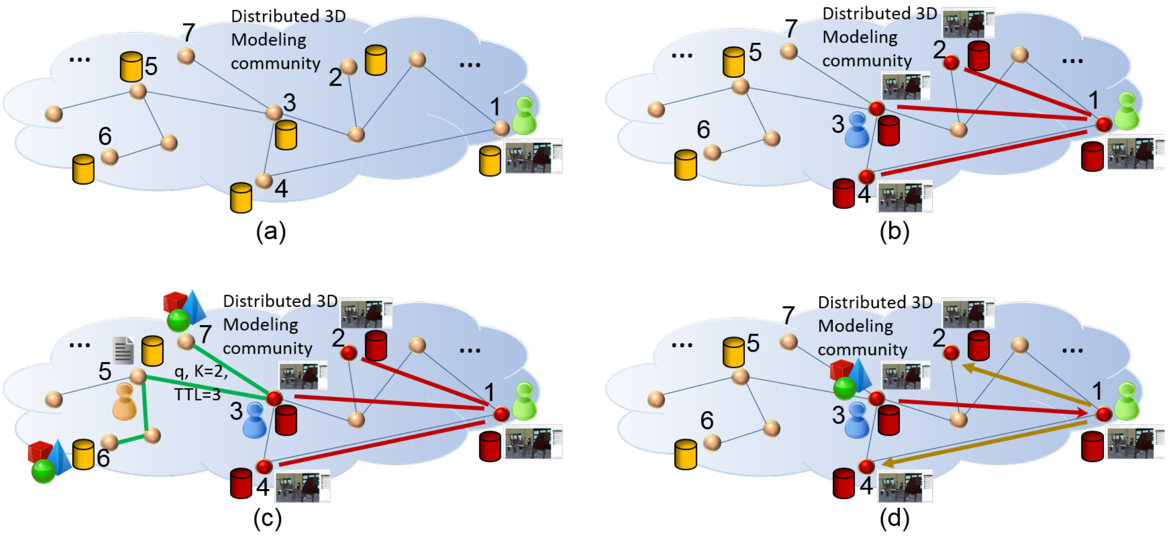


Fig. 7.13 Example of the working process of S2P2P and iRep3D in Collaborate3D.

7.4 Related Work

In this section, we discuss the previous contributions that support toolkits for manual 3D modeling, the approaches for the automatic 3D model generation and the dominant publicly available 3D scene repositories.

3D modeling systems: In [375], Wan et. al. present, WebMWorks, a cloud based platform for collaborative design, physical system modeling, simulation and knowledge sharing. The object-oriented modeling language Modelica[15] is used to express model as equation, which is the generalized representation of services. The process containing interacting models can be therefore abstracted as

---

[14] For a more detailed example of S2P2P search, we refer the reader to Section 3.6

[15] `https://www.modelica.org/`

schematic diagrams. On top of this, Web-based visualization is leveraged for the simulating the creation as well as the interaction process of models. Unfortunately, WebMWorks does not provide 3D visualization.

Menck et al. show, in [256], a virtual system for the immersive collaborative factory planning. With this system, a user is able to construct a virtual factory by creating and moving 3D objects. The latter can be additionally annotated with documentations for business issues. Besides, this platform provides the facilities for holding virtual meetings that can happen in line with the factory design in the virtual world.

Sacco et al. present a virtual reality based factory design tool, called DiFac [315]. This contribution scales to the integration of services for communication and project coordination. Simulated reality is hired in this work to evaluate the factory setup of user specified production process, which additionally includes the visualized services. However, the services used in DiFac are fixed and can not be customized. This differs with the collaborative virtual production design prototype, where the modular design is applied and each module provides a series of API for meeting the various requirements.

In contrast to DiFac, a 3D Web based collaboration platform for collaborative process and product design evaluation is presented in [276], DiCoDEv. With this platform, multiple users at different sites are able to work together on the same design. Virtual reality technique is employed for the displaying the working environment. The latter provides the real time immersive navigation and interaction facilities for users' collaboration. In particular, it provides the real time validation through the entire product design process. This work has been applied in a virtual reality application that is built on top of the commercial PTC Division MockUp platform[16]. On top of DiCoDev, Mavrikios et al. introduce a collaborative prototype designer [248] that scales to the collaborative review of design. With it, users can inspect the creation of models via the free navigation in 3D spaces, and exchange their opinions with others.

The authors in [61] present a mutliagent based collaborative, adaptive & realistic environment for training (MASCARET), which provides a tool for the selection, and composition of 3D models. Differing with Collaborate3D, this system allows the model-driven UML[17]-based 3D scene creation, component relation definition and transformation. In particular, it proposes an abstract syntax of 3D modeling notions based on the upper layer UML schema. Each concept in this syntax corresponds to a class of 3D component which properties are parameterized. The use of each component for creating a 3D scene instantiates a class with designer customized property values. Once the targeted 3D scene has been composed, which XML-based representation can be automatically transformed into 3D scene in domain specific languages. The merit of MASCARET system lies in that the modeling of 3D scene is agnostic to the kind of specific 3D modeling language. Besides, it leverages the UML abstract syntax as the foundation of its proposed syntax for 3D modeling, which facilities

---

[16] `http://www.ptc.com/product/division`

[17] `http://www.uml.org/`

the further 3D based modeling of process, as UML provides the well-known notions for process definition.

Tutenel et. al. present in [364] a system for the creation of 3D buildings. This work puts emphasis on the validation of building model completeness (e.g. the existence of the internal and external elements facade, furniture, etc.) and congruency (e.g. improper features, spatial overlapping, etc.). It hires a predefined ontology to specify and rule the elements that a building should have. Moreover, in order to guarantee the consistency of a model, this system provides a moderator for checking the relation constraints that those elements need to comply with. In [305], Reffat introduces a semantic-based virtual system for the design of 3D buildings. In particular, this work realizes the incorporation of semantics out of different aspects, such as functions and spatial relations, during the 3D content creation. The semantics in terms of function stands for the overall propose of design and allows designers to specify constraints of components in advance, which enables a kind of implicit validation. The spatial relation semantics is concerned to rule the contiguity of 3D components. In addition, this system provides the advanced features that allows the designer to check the esthetics, like symmetry and modality, of 3D buildings. Furthermore, lighting, sound insulation, thermal and air ventilation factors of 3D building models can also be specified and checked via this system.

In the perspective of commercial computer aided design (CAD), quite a few software products support the features for the collaborative design. CATIA by Dassault Systèmes[18] is a well-known tool for 3D CAD design, which allows users to share their workspace interactively with co-workers. This is done by using client-server or P2P based networking means. Real time chat and personal messaging are provided for asynchronous collaboration. Its feature for the concurrent product design is built on top of the Instant Collaborative Design 1[19]. Another representative commercial CAD software is AutoCAD 360[20]. It provides a collaborative 2D workspace with the support of cloud based storage and sharing. This software has different versions for PCs, mobile devices with Web-based frontend, which improves the accessability of data. However, AutoCAD 360 does not provide 3D based visualization of design.

Automatic 3D model generation: Most contributions, such as [175] [56] [159] [413] [107] [206], in the field of automatic 3D model composition focus on the generation of 3D models by swapping the components of existing models, in which geometric compatibilities between parts are often considered. In [175] [56], the authors present a method for model synthesis. It is able to generate new 3D models based on a set of initial models which segmented parts are categorized in advance and the parts in a generated 3D model are compatible in terms of style. This work proposes a mixed Bayesian probabilistic model consisting of both the observable and the latent feature variables. Training is performed in order to make system to learn the parameters of the distribution for each variable. This

---

[18] `http://www.3ds.com/products/catia/`

[19] `http://www.3ds.com/products/catia/portfolio/catia-v5/all-products/domain/Infrastructure/product/CD1/`

[20] `https://www.autocad360.com/`

is done by applying the expectation-maximization algorithm [77]. On top of the trained Bayesian network, structure based search that returns newly generated 3D model with compatible parts. A limitation of these two approaches is that the functionalities of 3D components and the possible data flow relations between the interacting components have not been considered.

Jain et. al. introduces a method [159] for the automatic generation of 3D models. Likewise, this approach performs the recombination of 3D components based on shape analysis in terms of shape segmentation, contact symmetric relation detection. Such analysis yields for each 3D model a hierarchy about its structure and formation. Given an input model, this method can be used to rapidly instantiate new models that have similar symmetry and adjacency structure yet in different appearances. As the first part of the shape analysis, polygonal 3D models are decomposed into smaller polygonal mesh. Each resultant part is generated by region growing. On top of the segmentation, the system is able to detect the geometric contact relations between parts, which are the knowledge about adjacency. In addition, the dominant global symmetry detection is also included in this approach by hiring the RANSAC approach [26]. Based on the analysis, a hierarchy for each 3D model is created in a top-to-down manner, which continuously decomposes the parent node and generates the children nodes of different types with concerned knowledge, e.g. contact relation. Those learned knowledge speed up the process of new model generation, in which compatible children nodes in different hierarchies are swapped.

In [413], the authors propose a method, called symmetric functional arrangements, for the automatic generation of plausible 3D shapes. This method contains two stages: off-line analysis and on-line reshuffling. Given a pre-segmented 3D model, the off-line analysis extracts the mutual relations among object parts and represents them as spatial relation graphs. Symmetric functional relations with attributes in predefined types are subsequently identified. By swapping the compatible parts of 3D models, the on-line reshuffling is able to generate new spatial relation graphs with compatible components from different shape families. Finally, such graphs are used to composing plausible 3D models. This approach investigates the function-oriented relation between 3D components, like Support, Placement, Coaxial, etc., which, nonetheless, are describing geometric relations.

The proposed methods in [107] and [206] are able to create new 3D shapes with user specified parts out of different input models. Instead of the component categorization, adjacency or symmetry relation detection, these works focus on finding better solutions for mesh segmentation and suturation.

3D scene repositories: There are several representative 3D scene repositories publicly available. In the sequel, we introduce the pros and cons of them, compared to iRep3D approach. The Google 3D Warehause[21] focuses on the keyword and tag based retrieval of 3D scenes in SKP[22] or Google Earth KMZ[23]. In comparison with other 3D repositories, Google 3D Warehouse has the largest number of

---

[21] `https://3dwarehouse.sketchup.com/advancedsearch.html`

[22] `http://fileinfo.com/extension/skp`

[23] `http://fileinfo.com/extension/kmz`

3D scenes indexed for retrieval. These data are contributed by the in particular Google SketchUp users in the world. Besides the keyword, Google 3D Warehouse supports search with meta-data (e.g. author information) and scene complexity (number of polygons). Unlike iRep3D, Google 3D Warehouse relies on keyword similarity. Therefore its search accuracy is subject to the word ambiguity (e.g. search with the keyword "green cone" will return cars and machines with high ranks). Another syntactic based search engine for 3D scenes is the ADL 3D Repository[24]. It allows for the meta-data based search and advanced federated search. Compared to iRep3D, it uses the syntactic based matching within SQL wildcard queries (e.g. search with the keyword "car" can return a cruiseship 3D model syntactically described as "Carrival line Voyager Class"). Without considering geometric feature aspect, ADL supports the retrieval of 3D scenes in multiple formats, including COLLADA(DAE[25]), Autodesk(FBX[26] and 3DS[27]), Wavefront(OBJ[28]), etc.. 3D repository AUTODESK 123D[29] supports the keyword-based search of 3D scenes within Autodesk compatible file formats. In particular, it provides a suite of desktop softwares and mobile apps, which offer the 3D model creation from 2D pictures shotten via cameras.

Inspired by the development of 3D printing technique, several 3D repositories are established for (commercially) sharing 3D models in dominant file formats for print. GRABCAD[30] is a community with over 1,870,000 3D model engineers and 640,000 CAD[31] files. It provides the keyword and tag based search of CAD models. My Mini Factory[32] is a platform for sharing 3D printable models. The exhibition of 3D models is not by virtual view, but the physically printed 3D objects. Keyword-based matching is used for the search. Makershop[33] offers keyword and category combined retrieval of printable 3D models. For more printable 3D model repositories, we refer the reader to 3DPrinting web site[34].

7.5 Summary

In this chapter, we have introduced the features of the collaborative virtual product designer, Collaborate3D, as well as its integrated iRep3D repository. The Collaborate3D provides a Web based development environment for virtual prototyping. It supports distributed, service-oriented and extendable collaboration in interactive design between multiple users. As the storage component of Collaborate3D, the iRep3D repository has been enhanced with parallel query processing, caching

---

[24] `http://3dr.adlnet.gov/`

[25] `http://fileinfo.com/extension/dae`

[26] `http://fileinfo.com/extension/fbx`

[27] `http://fileinfo.com/extension/3ds`

[28] `http://fileinfo.com/extension/obj`

[29] `http://www.123dapp.com/`

[30] `https://grabcad.com/`

[31] `http://fileinfo.com/extension/cad`

[32] `http://www.myminifactory.com/`

[33] `https://www.makershop.co/`

[34] `http://3dprintingforbeginners.com/3d-model-repositories/`

mechanisms and a Web based GUI for the purpose of semantic-based annotation and near realtime hybrid retrieval of 3D scenes. The architecture and prototype implementation of the iRep3D repository were introduced. Moreover, we have discussed the integration of S2P2P search strategy and iRep3D with the collaborative virtual product designer, which demonstrates the prospective of using efficient index-based semantic item retrieval and P2P-based intelligent search in collaborative virtual systems.

In the next chapter, we will present another application of the research results in this thesis, which lies in the social media sharing field. This application is supported by the design and implementations of the semantic P2P search strategy S2P2P (cf. Chapter 3), the semantic replication scheme DSDR (cf. Chapter 4), a P2P framework and a P2P evaluation framework.

Chapter 8

Application to Social Media Sharing

A social network is a structure consisting of a group of social participants (such as individuals or organizations) and the connections (relations) among them, in which the participants share and discuss their knowledge of interest in specific domains, like music, movies, scientific research, etc. [383]. P2P network is a new and popular type of infrastructure for social networks. They are able to alleviate the risks of unauthorized disclosure of personal data and failure of central data servers, which are well-known limitations of the client-server based social Web sites, like Facebook[1], MySpace[2], etc. In P2P-based social networks, a participant is able to access and share social content with others via her computing desktop and mobile devices, including PCs, laptops, smartphones and tablets. By taking advantage of the built-in photographic hardware and mobility of a computing device, a user can create and share content, like videos, photos, etc., anytime and anywhere.

A trendy and interesting domain in this context is media content sharing between the participants in mobile social networks. Created and shared by participants, such content is often incrementally annotated by different users with various kinds of semantic descriptions, like text and conceptual tags. One of the main challenges in this area is the search for desired multimedia content via a software suite of middleware components, which copes with the semantic-based data search and data-provider service coordination in wired or mobile P2P networks. For this purpose, the research results in this thesis (S2P2P in Chapter 3 and DSDR in Chapter 4) can be applied as building blocks for semantic search in mobile social networks. However, their performance in practice regarding social multimedia sharing in a real unstructured P2P network setting (not mere simulation) is unknown. This additionally requires an unstructured P2P network environment (framework) and an integrable P2P network evaluation framework. The former implements the networking and handles the basic messaging issues, while the latter can monitor the tests as well as compute and display the real time results of the concerned metrics.

To this end, we have designed and implemented the P2P network environment and evaluation framework in Java. On top of these, the semantic search strategy S2P2P and semantic replication

---

[1] `https://www.facebook.com/`

[2] `https://myspace.com/`

scheme DSDR have also been implemented. These efforts were used and tested for the construction of the MyMedia system[3], which allows users to share and search multimedia content with semantic-based annotations in unstructured wired and mobile P2P networks. The MyMedia system has been successfully demonstrated at the international IEEE consumer electronics show CES 2014[4] and at Disney Research Zurich in 2014[5]. Its core technologies will be contributed to a planned European H2020 innovation action[6] in the media domain. This work is driven by the European FP7[7]/ICT[8] research project SocialSensor[9].

The reminder of this chapter is organized as follows: In Section 8.1, we briefly introduce the background of the SocialSensor project. It is followed by a presentation of the architecture of the MyMedia application. The design and implementation of the P2P networking environment, the semantic search strategy S2P2P and the semantic replication scheme DSDR are discussed in Section 8.2. In addition, we introduce the design and implementation of the P2P evaluation framework in Section 8.3. Further, the practical evaluation results on the search performance of the MyMedia system are presented in Section 8.4. We discuss the related work on mobile applications for social networks in Section 8.5 and conclude this chapter in Section 8.6.

8.1 SocialSensor Project Overview

The European FP7/ICT research project SocialSensor is aimed at researching and developing a new framework and algorithms to enable the indexing, search, analysis and verification of user generated information from multiple sources in social networks. Instead of the classic text-based centralized retrieval model, this project studies the mining, indexing and search methods on distributed content with enriched semantic annotations. In addition, the distributed analysis of content from the dynamic and massive user base is relevant, as the basis for determining user preferences and experiences to improve the content recommendation and search. Such processes indicate the need for a semantic middleware that allows ad hoc network users to seamlessly discover, compose and share semantically-relevant multimedia data and services. The areas addressed by the project are extremely crucial for both scientific research and business. For this, a wide range of scientific research and technical investigations are planned, such as crawling and searching for social and web content, media mining and tracking in social streams, information personalization and visualization, semantic middleware for collaborative search and sharing of social media content, multimedia streaming, etc. For the semantic-based collaborative social media search and sharing part, the following representative scientific research questions will be studied [275]:

---

[3] `http://sourceforge.net/projects/mymedia-peer/`

[4] `http://www.ieee-region6.org/2014/ces-2014-consumer-electronics-show-ieee-booth/`

[5] `http://zurich.disneyresearch.com/~smangold/`

[6] `http://ec.europa.eu/programmes/horizon2020/`

[7] FP7: Seventh Framework Programme

[8] ICT: Information and Communication Technologies

[9] `http://www.socialsensor.eu/`

- a semantic-based high-precision search strategy for the retrieval of multimedia content and services in unstructured P2P networks.
- a semantic-based data/service replication scheme that helps to enhance the performance and data availability of multimedia content search in unstructured P2P networks.
- an integrated semantic middleware suite that features a high-performance semantic P2P search combined with a dynamic adaptive live streaming of annotated MPEG-DASH videos from mobile to mobile over HTTP in wireless networks with an unstructured and semantic P2P overlay.

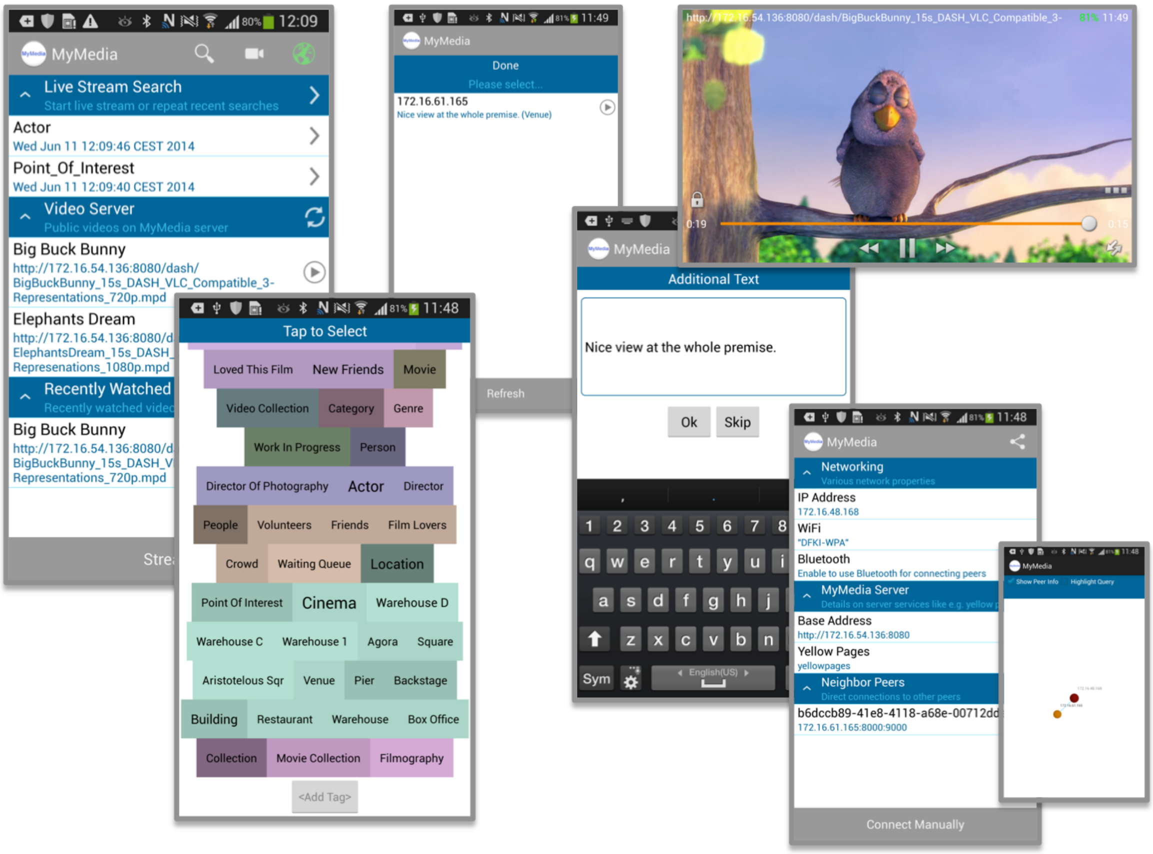
For this, the proposed semantic search strategy S2P2P and replication scheme DSDR has been applied to the social network media content sharing system MyMedia, which in addition is built on top of the P2P framework and tested based on the P2P evaluation framework. These frameworks are also the practical contribution of this thesis. In the following, we introduce the MyMedia app in terms of its functionality and architecture.

Functionality of MyMedia system: MyMedia is a mobile application (cf. Figure 8.1) for the users to share media information (e.g. trendy movies in cinema) in social network. It is designed for the 54th Thessaloniki International Film Festival (TIFF) and built in Java for the mobile devices running Android operating systems. It allows a user to create or join into an unstructured P2P mobile network in an ad hoc manner, in which she can provide or search for media contents and also relevant information of them, like movie trailers, video recordings, etc.. With MyMedia, users are able to annotate their media contents with tags from an ontology-based evolving tag cloud. Each tag is associated with a semantic description in OWL2. With MyMedia app, users can make joint decisions on watching some movie in cinema. Such a decision can be achieved based on some centrally stored movie trailers that were streamed to the mobile devices of a peer user group.

Figure 8.2 illustrates the goals of the MyMedia system in the context of the TIFF film festival. Suppose that Sheila, Pete and Carl are befriended visitors of the TIFF each of whom being at different locations of the festival premise. They launch their MyMedia-empowered festival app on their mobiles and join the wireless P2P network for TIFF visitors.

Pete uses his MyMedia peer to quickly search the mobile P2P network for latest information and media related to festival films in his prefered category, the location of cinemas where these are playing and live recordings of film parties on the festival premise. The peer returns a list of top-ranked media offered by other peers including videos of his favorite film title and a current live recording of some film party in the close proximity of the cinema where this film is playing.

Next, Pete wants to convince Sheila and Carl to jointly watch this film with him at the cinema and go to the film party afterwards. For this purpose, Pete's MyMedia peer can download both videos from the identified peers and then initiate a mobile P2P live streaming session for this content with Sheila and Carl in a peer group. Alternatively, the peer can request the identified peers to



Actor
Wed Jun 11 12:09:46 CEST 2014

Point_Of_Interest
Wed Jun 11 12:09:40 CEST 2014

Big Buck Bunny
http://172.16.54.136:8080/dash/BigBuckBunny_15s_DASH_VLC_Compatible_3-Representations_720p.mpd

Elephants Dream
http://172.16.54.136:8080/d ElephantsDream_15s_DASH_ Represenations_1080p.mpd

Big Buck Bunny
http://172.16.54.136:8080/d BigBuckBunny_15s_DASH_V Representations_720p.mpd

172.16.61.165
Nice view at the whole premise. (Venue)

http://172.16.54.136:8080/dash/BigBuckBunny_15s_DASH_VLC_Compatible_3-

Nice view at the whole premise.

IP Address
172.16.48.168

WiFi
"DFKI-WPA"

Bluetooth
Enable to use Bluetooth for connecting peers

Base Address
http://172.16.54.136:8080

Yellow Pages
yellowpages

b6dccb89-41e8-4118-a68e-00712dd
172.16.61.165:8000:9000

Fig. 8.1 MyMedia system snapshots [201].

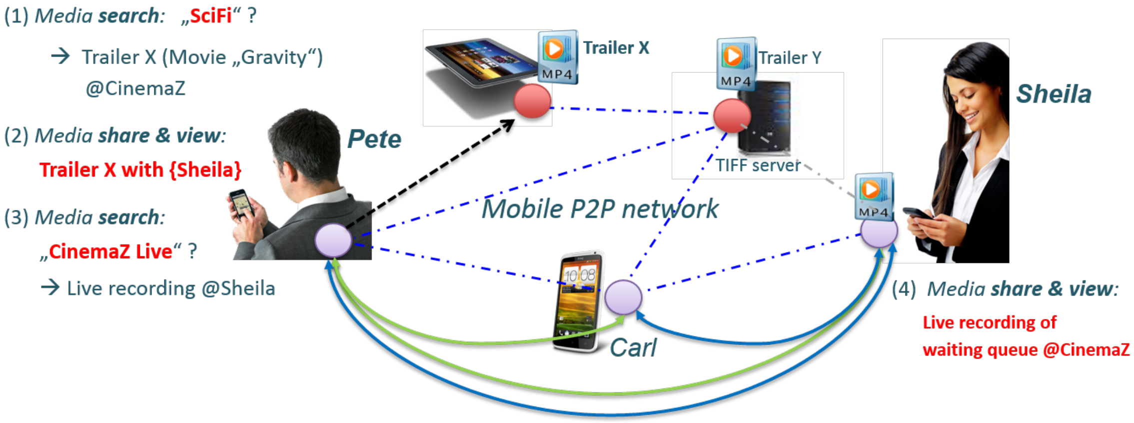


Fig. 8.2 Example of P2P media search and live streaming by MyMedia.

temporarily join the group for their performing of a peer-assisted streaming of their videos to Pete, Carl and Sheila. The latter is done by default in case of a P2P streaming of live recordings from the recording mobile peer.

After they have jointly watched the videos at the same time with acceptable quality and latency time, Sheila agrees with Pete's suggested plan, but Carl is hesitating. He would rather prefer a cinema which is not overly crowded and also has some restaurant in its surrounding with a nice veranda for having a social dinner before joining the film party afterwards.

Fortunately, Sheila happens to be nearby the considered cinema and uses her MyMedia peer to perform and share a live recording of the waiting qeue at this cinema and a restaurant in its surrounding with Carl. This eventually leads to a joint action plan of the group for the rest of the day on the festival premise. While walking to the cinema, Pete is also consent with sharing some of his short live recordings of POIs he is passing by with other visitors and festival staff members on request for different purposes such as navigation, information, and situation-aware coordination of assistance by the festival organisers.

MyMedia system architecture: MyMedia features a high-precision semantic search of annotated media and dynamic network-adaptive live streaming of videos over standard HTTP from mobile to mobile in an unstructured P2P network. Each MyMedia peer can search, download, upload and playback media content; in its current implementation MyMedia 1.0, the media type is restricted to MPEG videos.

The component-based system architecture of a MyMedia peer is shown in Figure 8.3. The semantics of MPEG videos are described in terms of semantic video services in OWL-S which are automatically generated by the peer based on the user-generated video annotation with tags from a local tag cloud, which are mapped to concepts in the peer ontology in OWL2. The peer interactively supports the user through its user interface and with its tag cloud handler in the process of semantic tagging and querying of MPEG videos. Semantic video services, metadata and video content are persistently stored and maintained on the mobile device. The peer components S2P2P [47], iSeM [200], OWLS-Xplan 1.1 [197], and DSDR [43] are implementing a high-precision semantic P2P search, selection, composition, and replication of semantic video services.

Each MPEG video is automatically DASH-encoded/decoded by the peer for different network bandwidths with its DASH-encoder component, and displayed with its embedded VLC player component for Android. The mobile P2P live streaming of MPEG-DASH videos or live recordings within a given peer group is initiated by one of its members and then performed by all members using their pDASH components. The adaptive streaming from mobile to mobile devices over standard HTTP using pDASH aims at providing an optimal quality of experience with respect to available network capacities and peer resources. There is neither a central streaming server nor centralized tracking nor an augmentation of distribution capacities with additional content boosters or desktop peers, nor a computational outsourcing to public or private clouds for this purpose.

The P2P interaction between MyMedia peers is concerned with IP-based wireless network communication and message routing based on the unstructured and semantic P2P overlay of the network. The network communication of peers relies on standard TCP and UDP on top of the IPv4 stack while

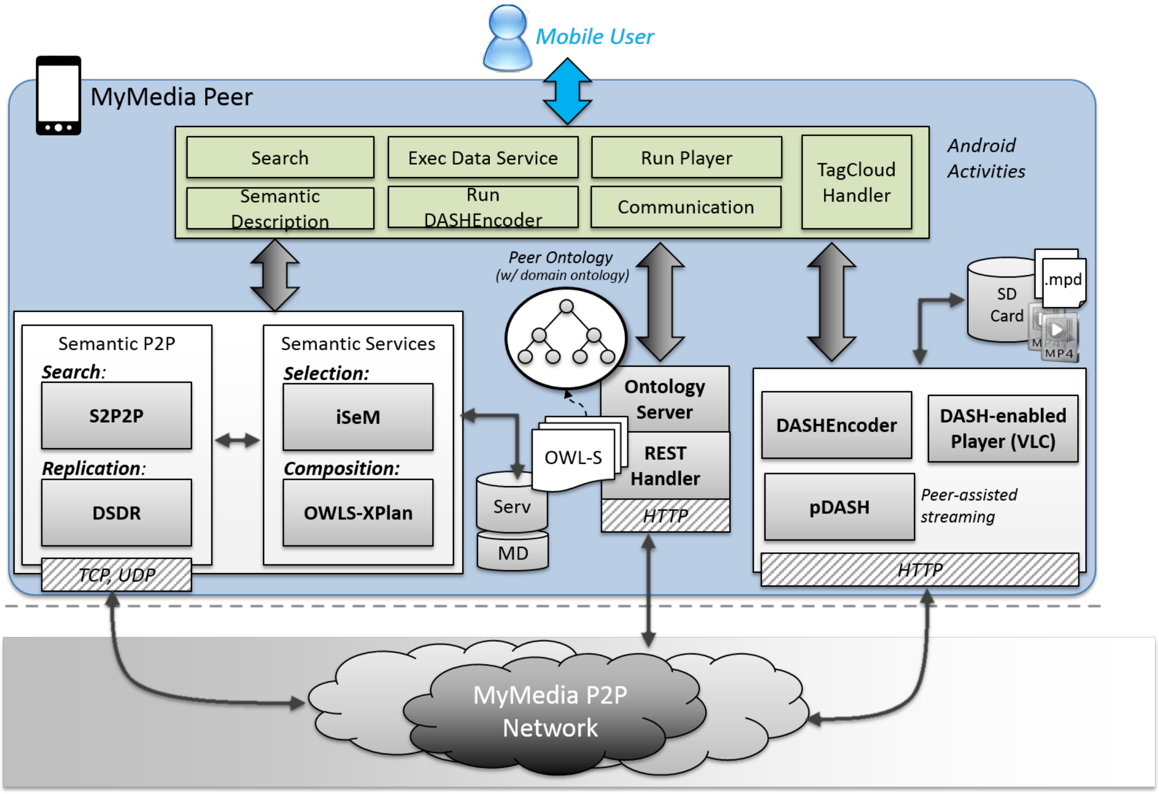


Fig. 8.3 MyMedia peer architecture.

the wireless transmission technology utilized by Android comprises WiFi (IEEE 802.11 standards), and 3G/4G communication.[10]

Direct, bidirectional communication between MyMedia peers refers to their direct connection in the unstructured P2P overlay, and the (semantic) query routing bases on this conceptual view of the network and semantic overlay knowledge. On the network level, messages from a peer to its neighbor peers in the P2P overlay can also be routed by TCP/IP via WiFi access points depending on the underlying network infrastructure. If required, a MyMedia peer can utilize a WiFi-Direct adapter separately from its WiFi adapter in Android, on top of the used IP stack [259].

With MyMedia peer, the peer discovery which is required for an initial handshake of a peer to join the network as a new node is performed either through a central server REST API, or ad hoc using Bluetooth. That is, the peer is informed about the IP addresses and TCP/UDP ports of either all direct neighbor peers by a look-up server, or just the one peer it connected to for this purpose only via Bluetooth. The initial handshake then follows a simple request/response pattern for exchanging small UDP notification messages which include IP addresses and receiving ports of the new and discovered peers. This enables the respective peers to update their local knowledge on the unstructured P2P overlay.

---

[10] NAT traversal is not supported by MyMedia 1.0 yet.

The MyMedia peer system has been implemented in Java for Android 3.2 with .apk size of 4 MB, and validated for different Android-based smartphones and tablets (Samsung Galaxy S2, HTC One, Samsung Galaxy Note 10.1, Sony S1, Google Nexus). In the following, we focus on the description of the search and live streaming features of MyMedia.

8.2 P2P Networking and Retrieval Components of MyMedia System

In this section, we introduce the P2P networking and P2P-based media data retrieval components of MyMedia system. Working as the base of MyMedia system for its P2P message exchange and intelligent search, these components support each MyMedia peer to be able to work as an independent process, which can be settled on a PC, a laptop or a mobile device with Android operating system. Due to their high-level design and implementation, these components, instead of mere the simulation, can be used to build any unstructured P2P-based application. Generally, P2P framework supports the TCP/UDP based peer messaging and the maintenance of peer local view on network topology. It in addition exposes a serious interfaces as well as callback mechanisms for the peer discovering, query routing and item matching, on which user's customized search and item similarity measurement can be implemented. P2P framework manages to work as the foundation of S2P2P and DSDR. S2P2P component is the implementation of the intelligent query routing strategy in Chapter 3; while DSDR component realizes the dynamic data replication scheme in Chapter 4. In Section 8.3, the P2P evaluation framework will be discussed. It provides the straightforward evaluation environment for the actual implementation of a P2P application that is built upon the P2P framework. This component covers a wide range of experimental configurations, real-time intermediate test result visualization and result exportation.

8.2.1 P2P Framework

As the foundation of MyMedia system, the P2P framework facilitates the construction of intelligent semantic search strategy S2P2P and the dynamic data replication scheme DSDR. In this section, we present its features, design and important parts of implementations. Its features include:

- *Message exchange*: Each peer is able to communicate with any other of its known peers via TCP/UDP protocol. It also supports the upper layer user to customize whether her peer should maintain or ignore specific messages for the purpose of controlling the memory consumption or performing data analysis.
- *Maintenance of peer local view on network topology*: Each peer can extract information about real time connections between peers based on its observed historical queries. This information is leveraged to conclude and maintain the peer local view on network topology.
- *Asynchronous and synchronous query models*: P2P framework offers two models for user query. Asynchronous query model supports the non-blocking query and a result notification

mechanism. It is efficient particularly in those scenarios where complex queries and large initial value of TTL are needed; while the synchronous query model is designed and realized for simple queries. User application can avoid to realize the necessary notification handling operation but gets the result in a short term.

- *Interfaces and callback mechanisms*: P2P framework provides a series of interfaces and callback mechanisms for user implementing her query routing, data selection and replication strategies. As the default query routing strategies, the classic flooding and k-random walkers has been implemented. Besides, P2P framework allows a user to implement and apply multiple similarity measurements in peer local data selection process. This facilitates the construction of P2P systems that concern multiple data types and measurements.

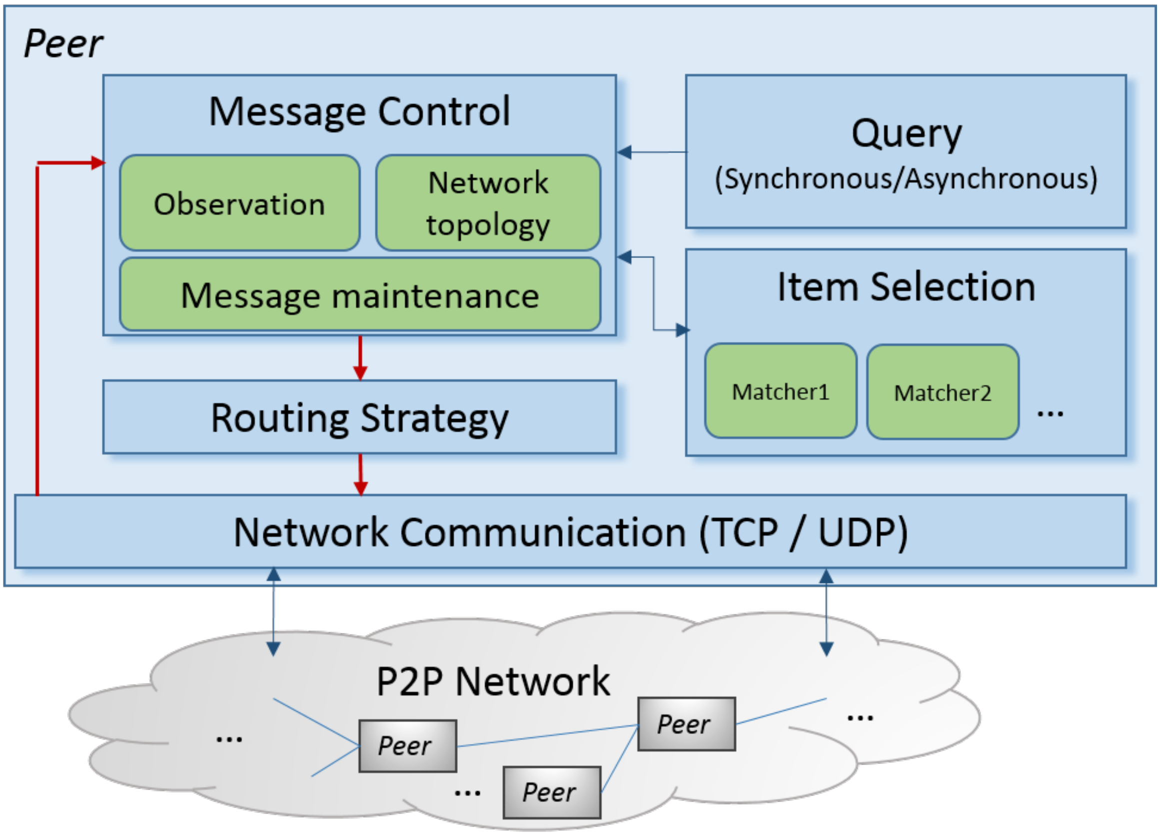


Fig. 8.4 Architecture of a peer in P2P Framework.

As the core of P2P framework, peer wraps all of its attributes and functional components, such as network communication, query routing, item selection, message control, etc.. Its architecture is depicted in Figure 8.4. The network communication component responses to send and receive messages (including query message) via standard TCP / UDP protocol. It manages a set of customizable communicating entities for listening on multiple TCP or UDP ports. Each of them hires an internal long term thread, which writes/loads data object into/from the socket. The content of a message can be a query or any other data, like PING message, which can be eventually customized by a user as needed.

In particular, an asynchronous message process method has been enabled in the current version of P2P framework, which is helpful of alleviating the risk of losing a message. For this, the network communication component maintains a FIFO message queue. On receiving a message from the socket, such thread puts it into the queue and immediately re-listen on its network port, instead of processing it. The processing of that message is done by the Message Control component. The latter leverages a thread to detect and fetch any new message from the other sides of the queue. A fetched message is maintained and processed by the Observation and Network Topology components. The Observation component allows a user to customize the observation process for her application in order to handle messages with different meanings; while the Network Topology component offers the automatic maintenance of the peer local view on its known peers and connections. It work as the base of the query routing path computation that might be needed and realized by user application. Besides, the Message Maintenance component manages to decide whether keeping or removing specific historical messages. In case that the content of message is a query, the Message Control component invokes the local item selection process that in general matches the descriptions of query with local items. The descriptions of matched items are copied to the answer set data structure of query. In this perspective, P2P framework allows a peer to employ multiple matchers for answering complex query with hybrid descriptions in various formalisms, such as text and concept. Once a message has been processed, peer is able to transmit it to a group of intended peers by its Routing Strategy component, if such propagation is necessary. In more detail, the Routing Strategy component chooses the target peer(s) for the current message and send it via calling the TCP/UDP communication components.

P2P framework enables the user of a peer to issue a query in either asynchronous or synchronous fashion. In the asynchronous query model, the querying process returns immediately after the requester peer routes the query. In addition, peer takes use of an event-based notification mechanism to monitor the query return and trigger the process for handling the results. For this, a monitoring thread is enabled to check the received message data, when it is enqueued. In the synchronous query model, the querying process of the requester peer blocks until that query has returned. In contrast to the simple client-server synchronous communication model where the request is processed by the server only before it is returned, a walker based query in synchronous model needs to be transitively processed by multiple peers on the query path. In the current design of P2P framework, this synchronous process is achieved by postponing the return of querying operation until the return data has been found in the message queue. A timeout parameter is additionally employed, in order to prevent the querying process from long term waiting.

P2P framework supports a serious of interfaces, methods and callback mechanisms for the possible user customizations in terms of four aspects: network communication, query routing, item selection and message handling. Shown in the following class diagram (cf. Figure 8.5), these functionalities is modeled by interfaces with generic implementation classes, which are all referenced and used by the Peer class. The latter is the basic implementation of peer functionalities and work as the core of the P2P framework. In the following, we briefly introduce the flexibility of P2P framework in the four aspects above:

- *Network communication*: The network communication part of P2P framework is modeled by interfaces *ICommunicator* and *ICommunicationPolicy*. The former offers the low level peer messaging operations (like *send()*) as well as manages the applied communication policies; while an implementation of the latter interface is supposed to realize the basic communication according to intended networking protocol, such as TCP, UDP, etc.. Once a messaging operation is invoked, the implementation class of *ICommunicator* should call the function of a managed proper communication policy. By early configurations, peer does not need to cope with the complexities and heterogeneities of networking protocols. As a default, TCP and UDP communication policies have been implemented. In addition, the *DefaultCommunicator* maintains one TCP and one UDP policy. It performs the message sending operation via TCP, the ping and broadcast via UDP.
- *Query routing*: P2P framework offers flexibilities for the extension of query routing strategies. Each peer maintains an instance of *IRouter* interface and simply calls its *route()* method as long as there is a query message that should be sent. The abstract class *AbstractRouter* of *IRouter* implements the skeleton of the *route()* method by hiring a series of callback methods, such as *getRoutingTargets()*, *onReceivingMessage()*, *onReturn()*, etc.. (cf. Figure 8.6) They are the key sub-processes for query routing. Concrete processes of them can be realized in user customized routing strategy, if needed. In particular, the method *getRoutingTargets()* returns a list of decided neighboring peers (references) as the routing targets, to which the current query message should be sent. By default, A k-random walker routing strategy has been implemented. As a component of MyMedia system, the S2P2P component includes a customization on these methods, which performs the path suggestion based intelligent routing. They will be discussed in Section 8.2.2.
- *Item selection*: Based on P2P framework, a peer is able to perform multi-dimensional matching of items with a given query. This work is delegated to the instance of *IItemSelector* interface, which instance is owned by a peer at run time. A user can customize the intended peer local item selection strategy by providing the implementation of *IMatcher* at each dimension. During item selection, the order of applying those matchers and the aggregation of results out of multiple dimensions can be fully customized. Besides, as an abstract implementation of *I-ItemSelector* interface, *AbstractItemSelector* allows user to customize the needed actions (e.g. local logging) before and after the local item selection. By default, P2P framework supports implementations of the strict string matcher, Levenstein distance [220] based string matcher and the abduction based conceptual similarity measurement [46].

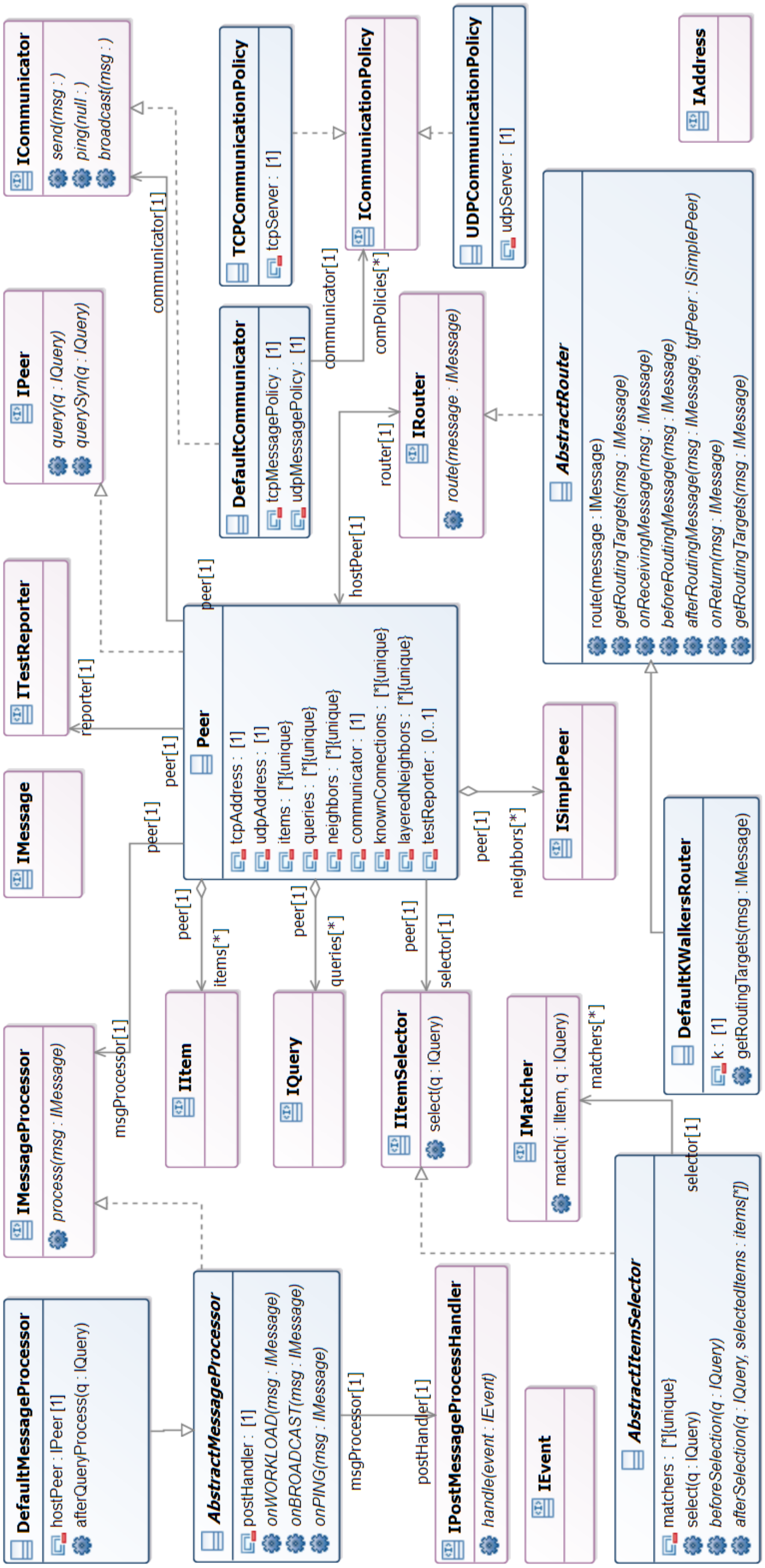


Fig. 8.5 Class diagram of important components in P2P framework.

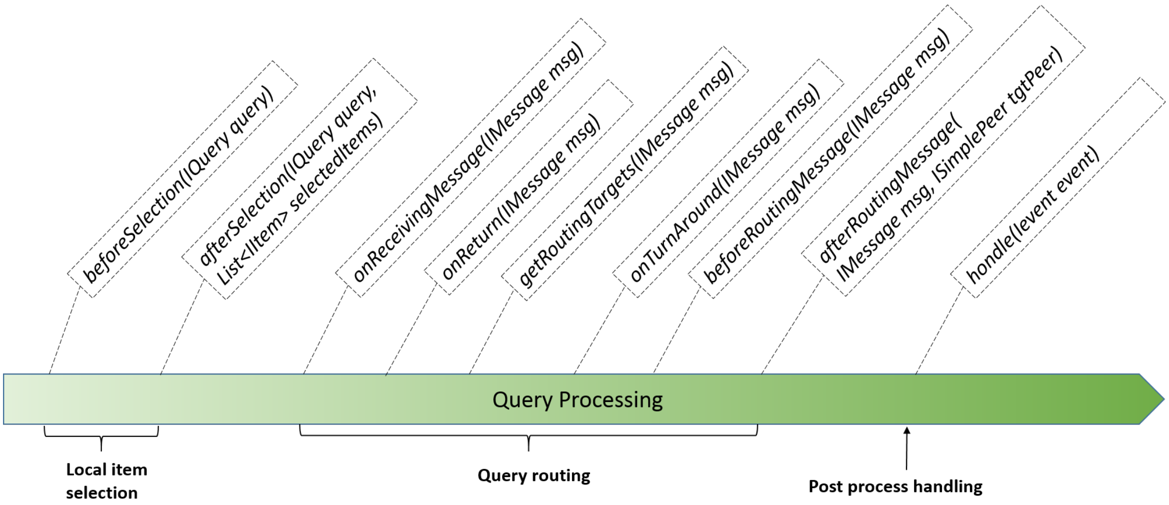


Fig. 8.6 Invocation order of the P2P framework callback methods.

- *Message handling*: The aim of message handling of P2P framework comes into two folds: (i) to support the basic message handling and the feasibility of the extensions for customized message analysis; (ii) to enable the asynchronous notification about message processing progress to external component, like GUI. For these, peer hires the instance of *IMessageProcessor*, which method *process()* means to process a given message in a very general level. Such message is processed by different methods according to its message type. This is done in the abstract class *AbstractMessageProcessor*, which leaves the space to the further implementations that customizes the specific processes on messages with different types. The use of asynchronous notification of message process is optional. Concrete implementation of message processor can realize the callback method *handle()* in *IPostMessageProcessHandler* interface for the external components (cf. Figure 8.6). A default message processor has been implemented, which realizes the message processing needed by k-random search, and prints the progress to a console by a default implementation of *IPostMessageProcessHandler*.

8.2.2 S2P2P and DSDR

On top of the P2P framework, the semantic based search strategy S2P2P (cf. Chapter 3) and dynamic data replication scheme (cf. Chapter 4) have been implemented. They are used by the MyMedia system for its search on semantically annotated movie trailers and services. In this section, we discuss their design and important parts of implementation.

S2P2P: S2P2P is a k-walkers based semantic search strategy in unstructured P2P networks. Each peer concludes demand and supply topics of remote peers based on its observations on their queries and selected items. On top of it, for a given query, a peer is able to suggest a query routing path respecting TTL limitation. Instead of mere an immediate neighbor, The suggested path contains multiple expert peers on the demanded topic, such that the total inverse expertise gain per traffic cost is minimized. In addition, each peer selectively disseminates its known item information to a group of peers on the query path, relying on its observed demand topics from them. These features have been realized as S2P2P Peer. Its architecture is shown in Figure 8.7.

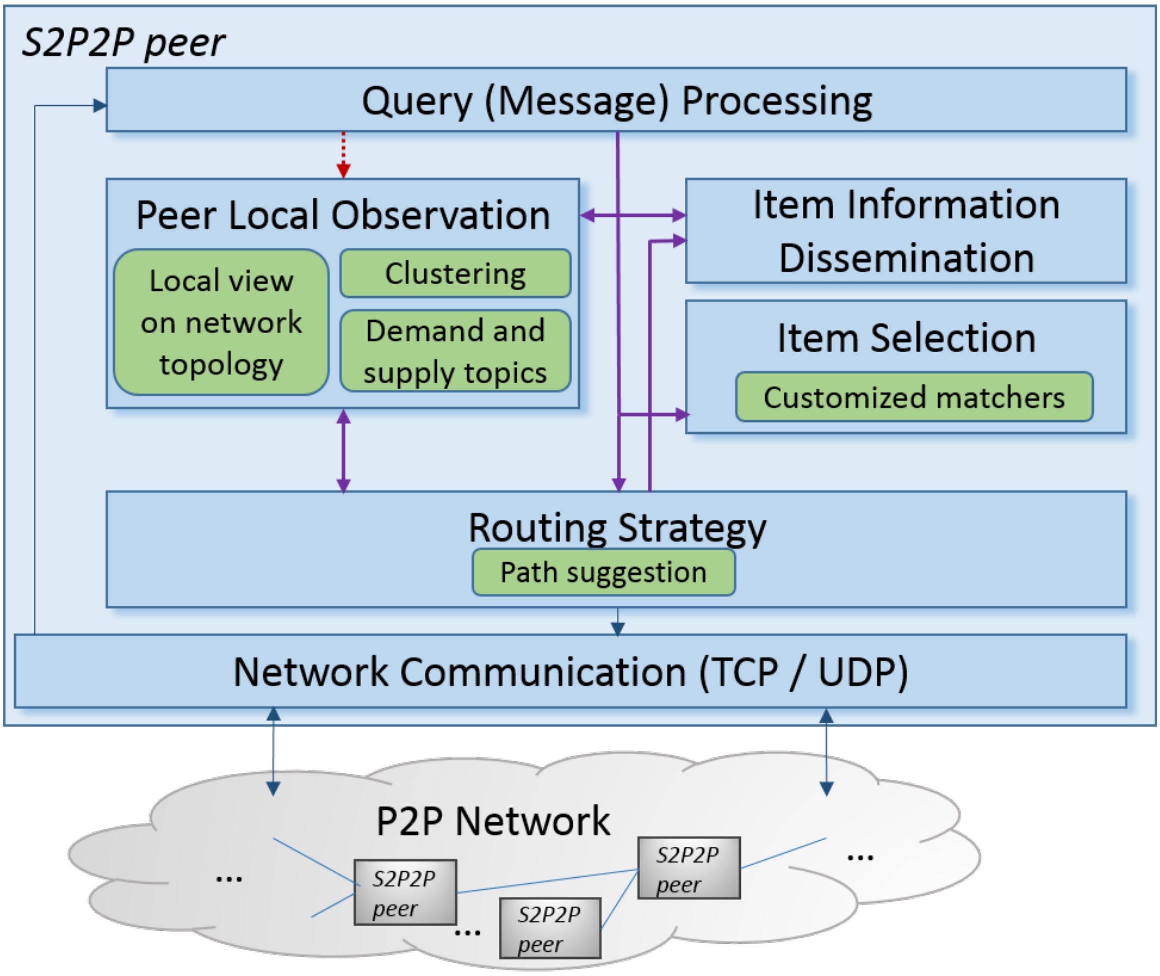


Fig. 8.7 Architecture of S2P2P peer.

A S2P2P Peer is an extension of Peer (cf. Figure 8.4) in P2P framework. It leverages the Network Communication component to send and receive query messages. For a given query, the Query (Message) Processing component performs: (i) the update of local observation on demand and supply topics (red dot arrow); (ii) local item selection, item information dissemination, and query routing (solid purple arrows).

The work of (i) is to maintain the up-to-date observation on the demand and supply topics of remote peers. This information is used by path suggestion and item information dissemination. Given a query, this process is accomplished by the Peer Local Observation component. In brief, by analysing the query path, S2P2P peer updates its local view on network topology. Besides, this component performs k nearest neighbor clustering on queried demand and selected items, respectively.

The work of (ii) realizes the core idea of S2P2P query processing. The local item selection is done by the Item Selection component. It takes advantage of the user customized matchers, as S2P2P is designed to be agnostic to the kinds of item descriptions. Once this process is finished, the query is delivered to the Routing Strategy component. As an extension of the default k random search of the P2P framework, the Routing Strategy component is able to compute a routing path suggestion, which contains multiple expert peers on the demand topic. For this, the Routing Strategy component invokes the functions of the Peer Local Observation component. The latter provides the information of remote peer expertise on relevant topics as well as the local view on network topology. The completion of path suggestion triggers the item information dissemination process. It determines to propagate some item descriptions to a group of peers on the query path (including the suggested path). This is done by the Item Information Dissemination component, which checks whether the observed demands of remote peers match the known items of this peer. After copying the descriptions of chosen item into the piggy-backed data set of query, the Routing Strategy component passes the query to the Network Communication component. The latter sends it to the first peer on the suggested path.

The S2P2P peer components have been implemented based on the P2P framework. Its class diagram is shown in Figure 8.9. Two main extensions for S2P2P peer are the classes *S2P2PPeer* and *S2P2PRouting*, which realize the features about peer local observation, path suggestion based query routing and item information dissemination:

- *Peer local observation*: S2P2P peer takes use of the *DefaultMessageProcessor* class as the entry point for query processing. It firstly triggers the user customized local item selection strategy to search for matched items. Once this is done, *DefaultMessageProcessor* immediately invokes the *route()* method of its referenced peer implementation class, which in S2P2P is the *S2P2PPeer* class. The latter further delegates the task to its router, i.e. *S2P2PRouting* class. The implementation of *route()* method in *AbstractRouter* is a template, which invokes different callback methods in specific situations. Amongst them, the *onReceivingMessage()* method is called at the very beginning in *route()* process, which has been implemented for the peer local observation. In detail, *onReceivingMessage()* method utilizes the functionalities provided by *S2P2PPeer*: it calls the *observe()* method to categorize the observed query (demand), respectively its selected items (supply) to the corresponding topics; in addition, the local view on network topology is updated in the meanwhile by the *updateLocalTopologyView()* method of *S2P2PPeer* class.

- *Path suggestion based query routing*: In S2P2P, the task of finding the target peer for routing a walker is to find the first peer in the suggested query path. For this, the implementation of *S2P2PRouting* invokes the S2P2P peer function *computePathSuggestion()* first, which com-

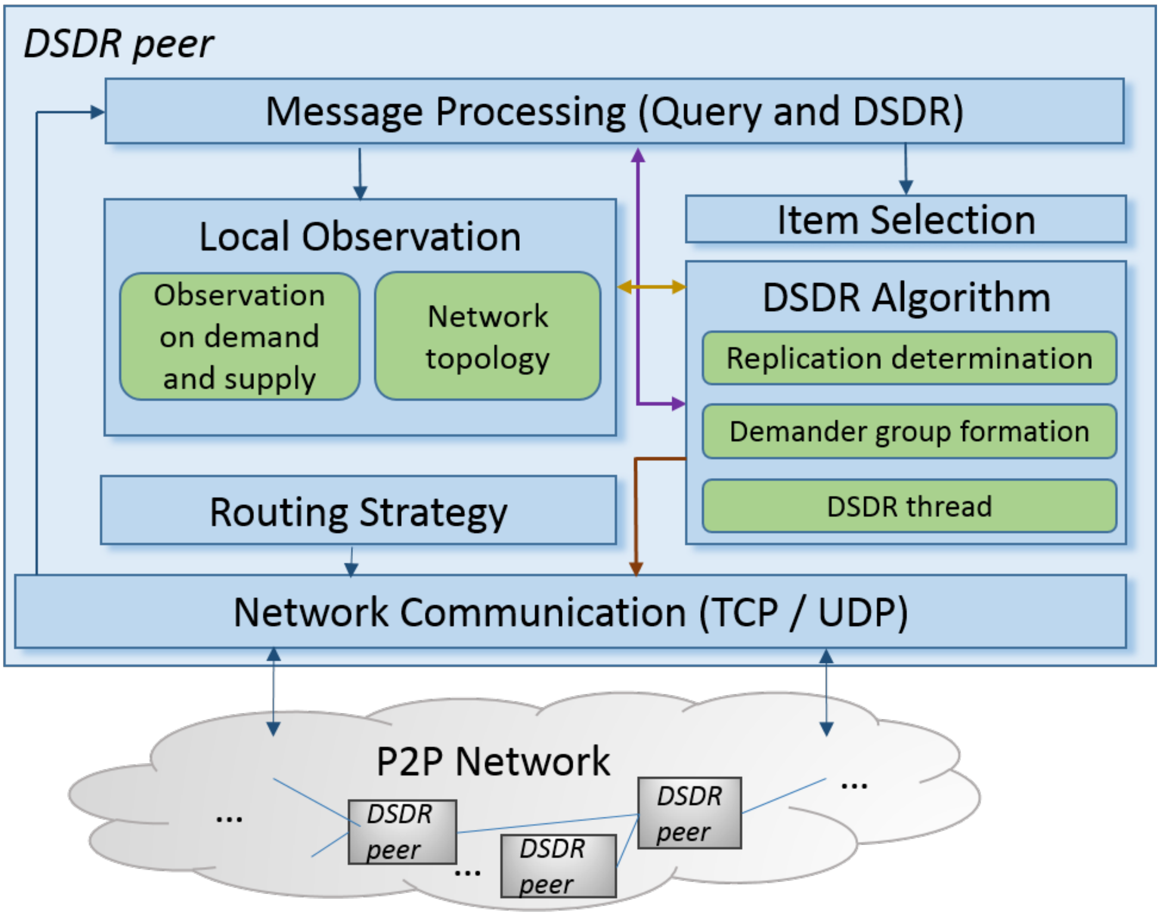


Fig. 8.8 Architecture of DSDR peer.

putes the path suggestion for a query and stores this result in the *pathSug* field of *S2P2PQuery*. The latter is the realization of *IQuery* in the context of S2P2P. Such updated query is then passed to the *getRoutingTargets()* method that manages to return the first peer in the path suggestion, if the suggestion has been made. In case that the path suggestion can not be made due to the lack of relevant supply topics, a random neighbor peer is returned.

- *Item information dissemination*: Each peer performs item information dissemination process, which aims at propagating its locally known item descriptions, to certain group of peers in the current query path or suggested path, if their relevant demands have been observed before. This feature is captured by implementing the callback method *beforeRoutingMessage()*, which is invoked immediately before the query is sent to its routing target peer. A disseminated item description is wrapped in an instance of type *S2P2PItemDissPackage* class. Such instance is piggybacked by the query and propagated to the peers in the query path and the suggested path as well. This information is observed by its corresponding receiver peers by their local observation process.

DSDR: DSDR is a dynamic data replication scheme for k-walker based search in unstructured P2P networks. It enables a peer to determine the replication of its known item(s) to a target peer. For this purpose, each peer, based on its local observation, periodically performs demand group formation process. For each local demand in the latest period, this process finds a set of candidate peers that have semantically similar demand topic. For forming a demander group, this leading peer sends a group formation request message to each of the candidate peers. On receiving an acknowledgement message, this peer adds the replier into the group on the demand topic. On top of this, peers in a demander group jointly determine the replication of each known item from its owner peer to one of the member peers. For this, the leading peer requests a series of factors from each group member and computes the final target for the replication. On receiving a replication notification message, the target peer requests the item (including data) from its owner peer. These features have been realized as DSDR Peer. The architecture is shown in Figure 8.8.

A DSDR peer is an extension of Peer (cf. Figure 8.4) in P2P framework. Its core component DSDR Algorithm realizes the demander group formation and data replication determination processes. In addition, In the following, we introduce the both processes and the interoperations between DSDR Algorithm component with the other customized components in the DSDR peer.

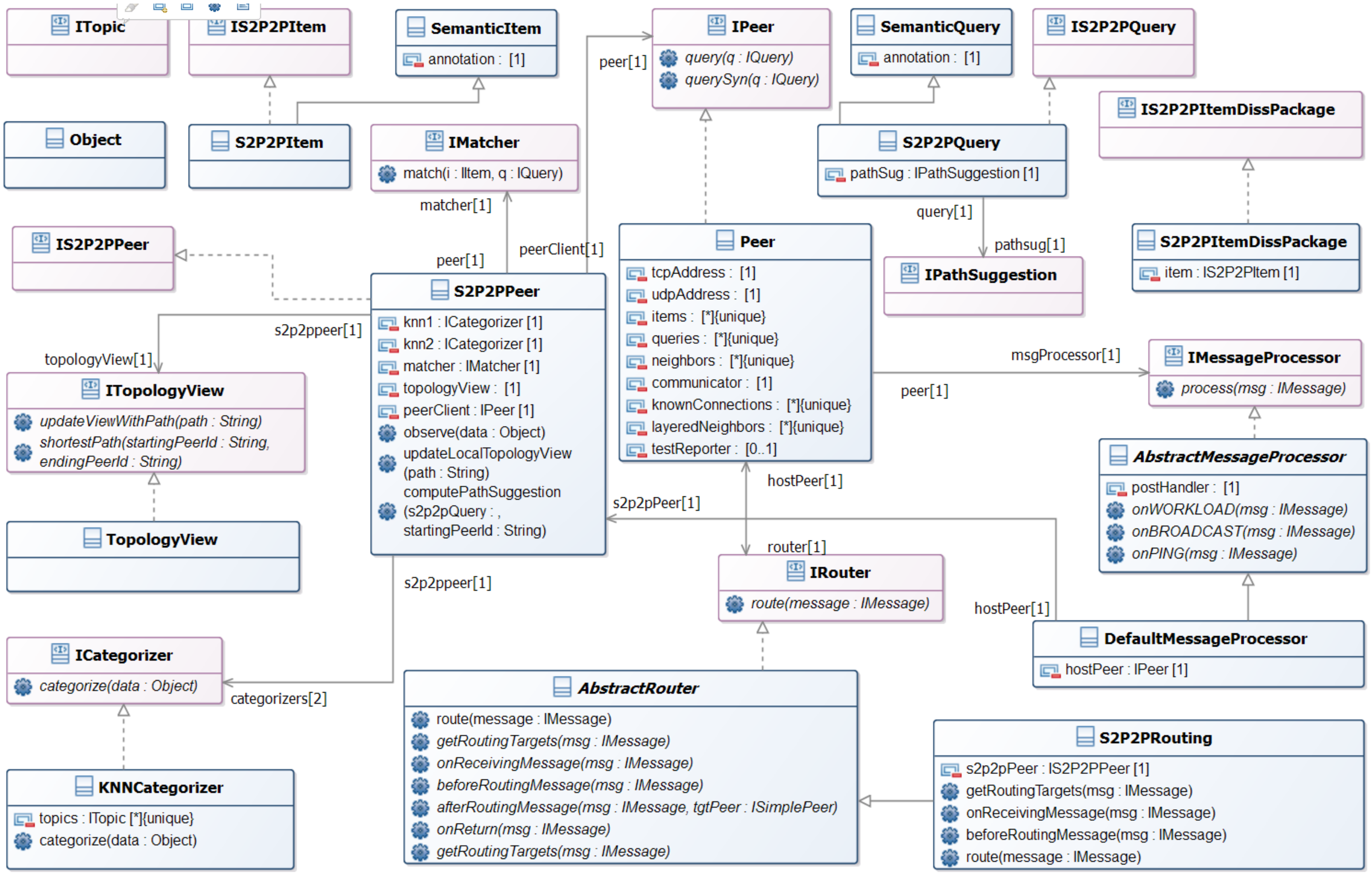


Fig. 8.9 Class diagram of the important components of S2P2P.

The demand group formation process of a DSDR peer is triggered periodically by its DSDR thread. This process utilizes the Local Observation component to find the queries issued by the peer itself in the period. If such a query is not satisfied, DSDR peer tries to build a demander group and to replicate item afterwards. For this, It works as a leading peer and sends the demander group formation request messages to a set of peers via Network Communication component, if the DSDR peer has observed semantically similar demands from them in the same period. The receiving of the replies is handled by Message Processing component, which then passes the DSDR specific message to the Demander group formation component for process. On receiving a group formation request message, the Demander group formation component of a DSDR peer is also triggered by its Message Processing component for the purpose of answering to that request. Once a reply is computed, DSDR peer sends it via Network Communication component.

The data replication determination starts immediately after the group formation process. For determining the target peer for a replication about a known item, the Replication determination component of the leading peer computes a series of values in order to measure the benefit of replicating the item to itself. This peer also requests the same factors from the member peers in the group by sending them the replication decision request message via the Network Communication component. Likewise, the replies are filtered by the Message Processing component and then processed by DSDR Algorithm sequentially. Once the target peer is determined, the leading peer sends the notification message to the target peer via Network Communication component. On receiving a data replication decision request message, the Replication determination component is called by the receiver peer and computes the requested values. Those values are then sent to the leading peer.

DSDR peer components have been implemented based on the P2P framework. Its class diagram is shown in Figure 8.10. Differing with the customization made for S2P2P, the implementation of DSDR does not use the *DefaultMessageProcessor*, but a *DSDRMessageProcessor* that copes with the DSDR specific messages. Each exchanged message is treated as an event, which is labeled with a performative defined in *IDSDREvent* interface. Besides, the implementation uses the default k-random search routing strategy and in addition extends it as a thread class *DSDRThread* for the periodic runs of DSDR algorithm. Main functionalities of the latter are implemented in the *DSDRPeer* classes.

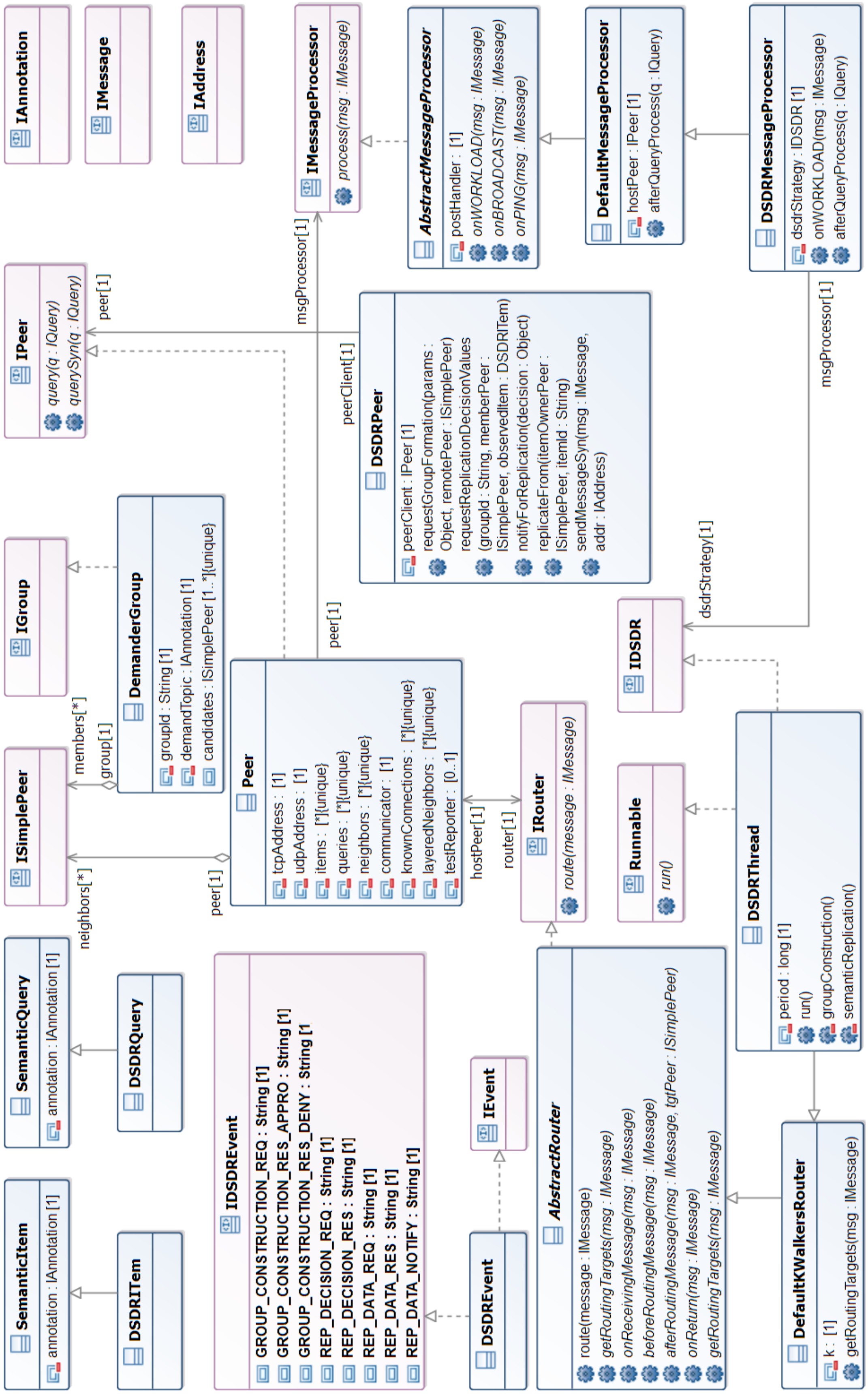


Fig. 8.10 Class diagram of the important components of DSDR.

- *Demander group formation*: *DSDRThread* invokes its *groupFormation()* method at the end of each *period*. The *groupFormation()* process drives its referenced *DSDRPeer* class to check the locally issued queries. If such a local demand was not satisfied, *DSDRPeer* attempts to check whether there exists any observed queries that has the semantically similar demand but issued by another peers. In case that it is found, *DSDRPeer* calls its method *requestGroupFormation()* to request that peer to form a group. This method sends the request and waits for the reply in a synchronous manner. If the reply is labeled with *GROUP_CONSTRUCTION_RES_APPRO*, the leading peer adds that peer to the group.

- *Data replication determination*: The completion of the demander group formation triggers the data replication determination, which is implemented in the method *semanticReplication()*. This method drives the referenced *DSDRPeer* to check, for each of its known item, whether that item can be replicated to a member peer of the group. For this purpose, *DSDRPeer* calls its method *requestReplicationDecisionValues()* to ask for a series of values from each member peer. This is done by sending a DSDR specific message with the label *REP_DECISION_REQ* and subsequently waiting for the reply. Once the replies from all member peers are collected, *DSDRPeer* determines the final target peer for the replication of the concerned item and sends a *REP_DATA_NOTIFY* message to that peer by its method *notifyForReplication()*.

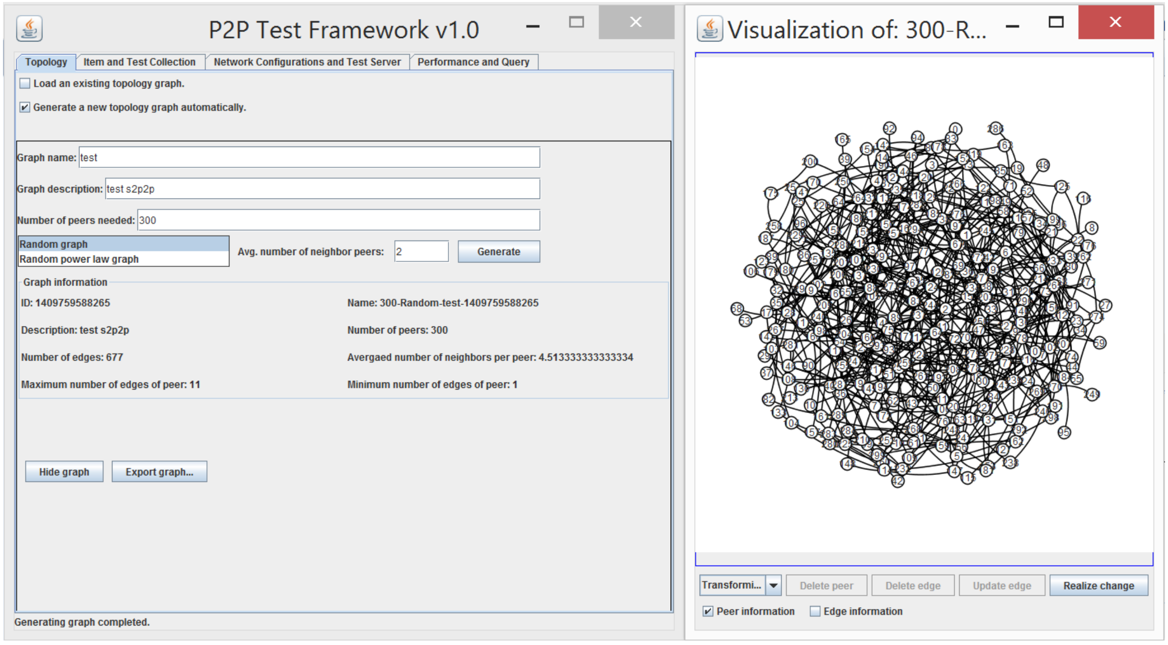
8.3 P2P Evaluation Framework

P2P evaluation framework is designed and implemented in Java for the straightforward evaluation of P2P systems and applications built upon the P2P framework. Not mere the simulation, a peer in the tests with our P2P evaluation framework is able to stand alone as a real P2P client, which communicates with other participants via IP protocol. Without further adaptation, this directly enables the evaluation of MyMedia peer.

With a GUI, P2P evaluation framework provides a set of features for user customized tests. Such features can be classified into two categories: (i) pre-test settings and (ii) run time settings. The pre-test settings incorporate various possibilities for targeted P2P network instance with a set of representative parameters for topology generation, test collection customization, query and item topic distribution, item distribution and multiple hardware test; while the run time settings enable the on-the-fly monitoring and interaction during the running of the configured P2P system. It consists of the single and batch query execution, real time performance visualization, test result exportation as well as run time network property change. In the following, we briefly introduce these features:

Pre-test settings:

- *Topology generation*: P2P evaluation framework allows a user to generate random graph or random power law graph based P2P network topologies with arbitrary number of peers. Shown



☐ Load an existing topology graph.

☑ Generate a new topology graph automatically.

Graph name: test

Graph description: test s2p2p

Number of peers needed: 300

Avg. number of neighbor peers: 2

Graph information

ID: 1409759588265

Name: 300-Random-test-1409759588265

Description: test s2p2p

Number of peers: 300

Number of edges: 677

Avergaed number of neighbors per peer: 4.513333333333334

Maximum number of edges of peer: 11

Minimum number of edges of peer: 1

Generating graph completed.

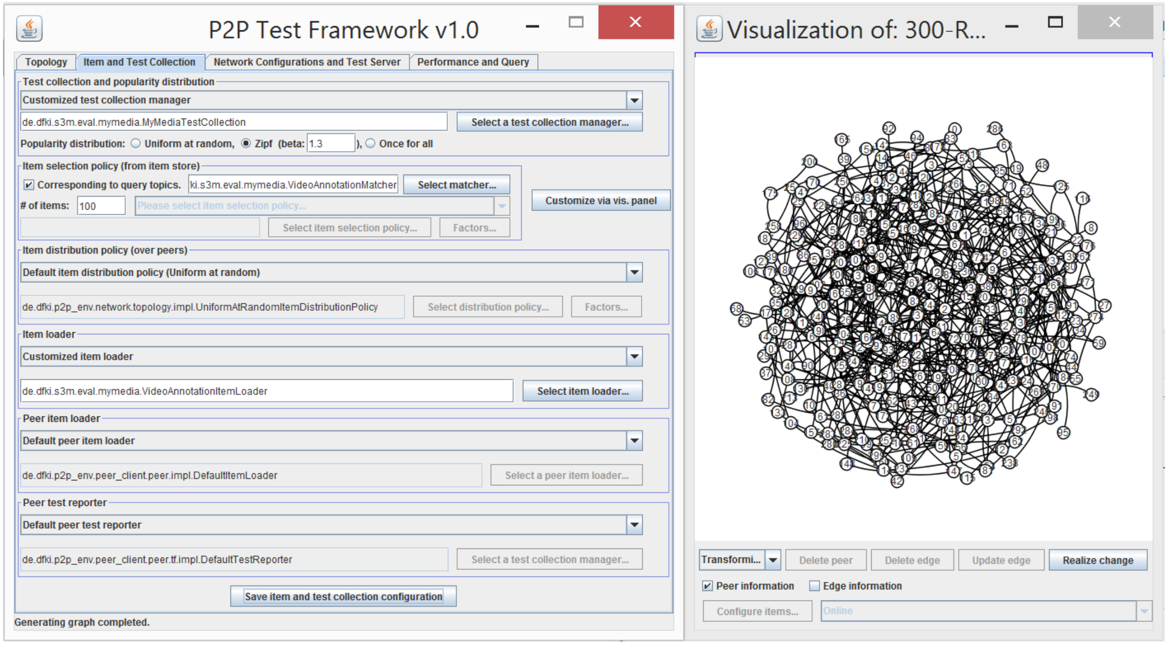
☑ Peer information ☐ Edge information

Fig. 8.11 Topology construction of P2P network.

in Figure 8.11, the generation of random graph based topology allows to setup the average peer degree to control the network connectivity. As for the random power law graph based topology, its peer degrees follow a power law distribution $p(x) = Cx^{-\beta}$. Its skew value β ($\beta > 1$, $\beta \in \mathbb{R}$) can be customized, which specifies the exponent and the steepness of the function, in order to model the social interactions. Besides, P2P evaluation framework enhances the customization on network topologies, which allows a user to manually add/delete peers/connections via a network visualization panel.

- *Test collection customization*: P2P evaluation framework provides flexibility for user to customize the test collection for the intended test. Shown in Figure 8.12, the framework exposes an interface that rules how the labeled queries and items are read from (external) data sources, such as database, files on disk, etc., as well as how their relevance assessments are retrieved. On top of this, the targeted metrics, like the real time precision and recall, can be computed during the run of system.

- *Query and item topic distribution*: P2P evaluation framework scales to mimic the peer preference on queries and its maintained items. It provides the k nearest neighbor based classification for a user to category the queries and items to topics. A peer in the targeted P2P system is able to choose queries and items on those topics under a uniform at random or Zipf distribution . The latter is often used for modeling the observations in social networks (e.g. user preferences on movie types). In addition, the evaluation framework alternatively allows a user to select items from a possibly large item store, which are semantically relevant to the configured query



Test collection and popularity distribution
Customized test collection manager
de.dfki.s3m.eval.mymedia.MyMediaTestCollection
Popularity distribution: ○ Uniform at random, ● Zipf (beta: 1.3), ○ Once for all

Item selection policy (from item store)
☑ Corresponding to query topics. ki.s3m.eval.mymedia.VideoAnnotationMatcher
# of items: 100 Please select item selection policy...

Item distribution policy (over peers)
Default item distribution policy (Uniform at random)
de.dfki.p2p_env.network.topology.impl.UniformAtRandomItemDistributionPolicy

Item loader
Customized item loader
de.dfki.s3m.eval.mymedia.VideoAnnotationItemLoader

Peer item loader
Default peer item loader
de.dfki.p2p_env.peer_client.peer.impl.DefaultItemLoader

Peer test reporter
Default peer test reporter
de.dfki.p2p_env.peer_client.peer.tf.impl.DefaultTestReporter

Generating graph completed.

☑ Peer information ☐ Edge information

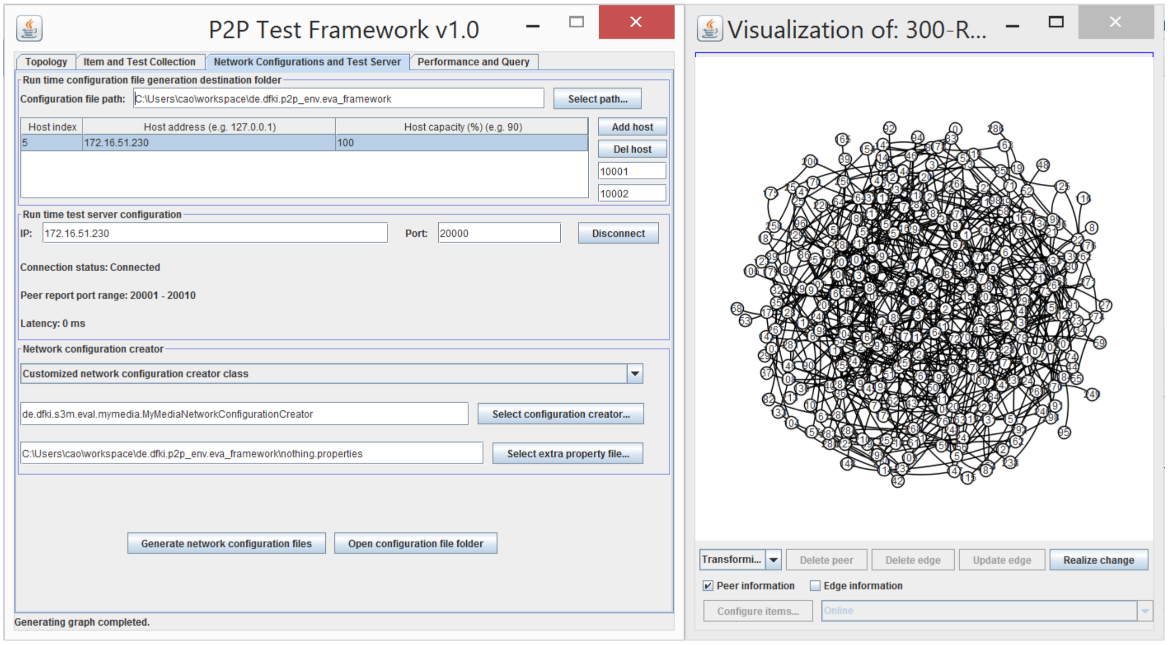
Fig. 8.12 Customization of test collection, query and item topic distribution and item distribution.

topics and follows the same preference.

- *Item distribution over peers*: This feature determines the total number of initial items and their distribution over peers in the targeted P2P system. The distribution defaults to uniform at random and can be customized with parameterized implementations. Besides, the evaluation framework provides interfaces for item data loading at configuration or run time. This makes the user tests to be able to run with different kinds of data and formats.

- *Multiple hardware test*: P2P evaluation framework provides the alternative to run the test of a targeted P2P system in a distributed environment with multiple computational devices (e.g. PC). It scales to arbitrarily divide the peers into multiple groups. The peers of each group can be settled on one computational hardware and all groups form the overall network. At run time, each group can be started separately and its peers establish their configured connections, regardless of the physical locations of the target peers. The communication between any two peers is done via TCP/UDP protocol, no matter whether they are settled on the same or different machines. In addition, P2P evaluation framework facilitates this configuration by its GUI (cf. Figure 8.13), where a user is able to setup the addresses of available machines as well as the pattern of peer communication ports. Such information is used to generate the network configuration files for each machine via a user customizable network configuration creator interface. Once the configurations are settled properly, the targeted system can be started.

Run time settings:



Run time configuration file generation destination folder

Configuration file path: C:\Users\cao\workspace\de.dfki.p2p_env.eva_framework

Host index	Host address (e.g. 127.0.0.1)	Host capacity (%) (e.g. 90)
5	172.16.51.230	100

10001

10002

Run time test server configuration

IP: 172.16.51.230 Port: 20000

Connection status: Connected

Peer report port range: 20001 - 20010

Latency: 0 ms

Network configuration creator

Customized network configuration creator class

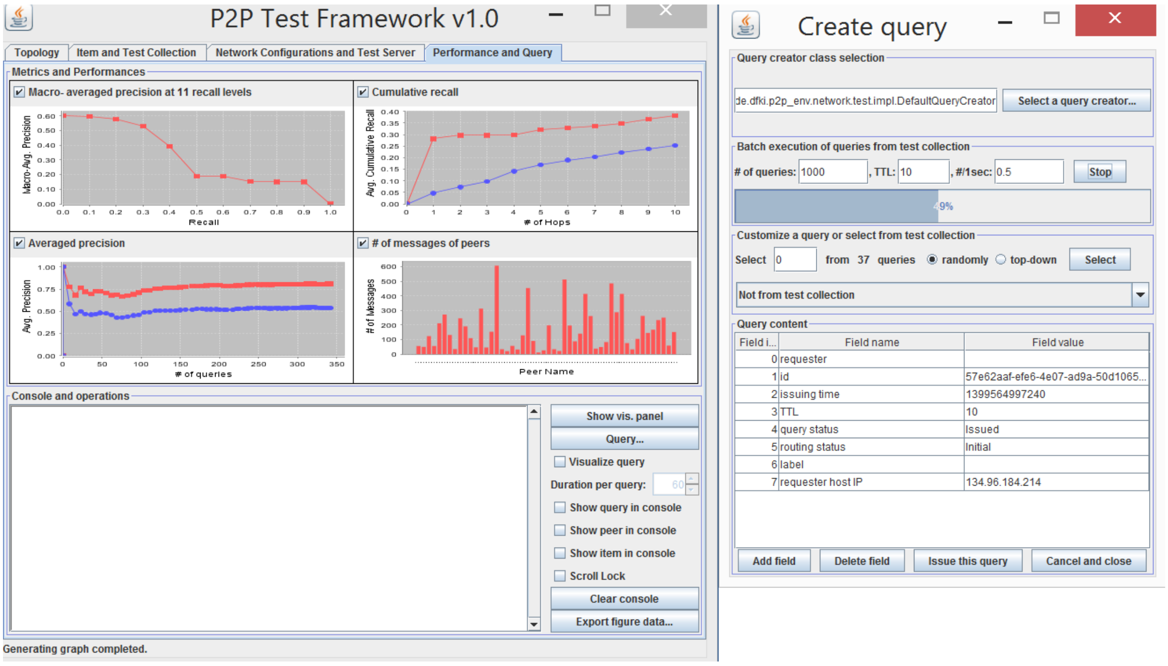
de.dfki.s3m.eval.mymedia.MyMediaNetworkConfigurationCreator

C:\Users\cao\workspace\de.dfki.p2p_env.eva_framework\nothing.properties

Generating graph completed.

☑ Peer information ☐ Edge information

Fig. 8.13 Customization for running tests on multiple hardwares.



☑ Macro- averaged precision at 11 recall levels

☑ Cumulative recall

☑ Averaged precision

☑ # of messages of peers

☐ Visualize query

Duration per query: 60

☐ Show query in console

☐ Show peer in console

☐ Show item in console

☐ Scroll Lock

Generating graph completed.

Query creator class selection

de.dfki.p2p_env.network.test.impl.DefaultQueryCreator

Batch execution of queries from test collection

# of queries: 1000, TTL: 10, #/1sec: 0.5

49%

Customize a query or select from test collection

Select 0 from 37 queries ● randomly ○ top-down

Not from test collection

Query content

Field i...	Field name	Field value
0	requester	
1	id	57e62aaf-efe6-4e07-ad9a-50d1065...
2	issuing time	1399564997240
3	TTL	10
4	query status	Issued
5	routing status	Initial
6	label	
7	requester host IP	134.96.184.214

Fig. 8.14 Run time system performance visualization and query generation

- *Real time performance visualization*: The GUI of the P2P evaluation framework provides the visualization of system real time performance in terms of several representative metrics: precision at recall, cumulative recall, averaged precision (cf. Section **??**) and number of messages of peers (cf. Figure 8.14). This information can be selectively updated periodically. A user is able to check the evolution of the system performance over time while the test is running, which is helpful for optimizing the parameters of a system before delivering its final version to users.

- *Single and batch query execution*: A single or a batch of queries can be issued at any time once the system initialization has been done. For issuing single query, the P2P evaluation framework allows to detail the value on each field of the query via a query creator panel (cf. Figure 8.14). Alternatively, it allows to trigger the execution of a batch of random queries out of the test collection at a given frequency. The issuer peers of them are randomly assigned. For both approaches, P2P evaluation framework provides a query creator interface for a user to specify the customized query implementation with concerned extra fields. This is particularly helpful to single query execution, in which the intended values of those customized fields in a query can be configured via the editable query content table in the query creator panel.

- *Test result exportation*: The P2P evaluation framework allows to export the test result data in CSV format for the possible further analysis. A user can export the real time result data at any time during the run of the system. In addition, these data is also automatically persisted periodically in order to guarantee that no data is lost. Further, the detailed log of each query, including particularly the values of fields: path, selected items, issuing time, ttl, etc., is stored in a database for detailed analysis.

- *Run time network property change*: The P2P evaluation framework additionally allows a user to manually create special cases on network topology dynamics, i.e. the arrival and departure of a peer or the addition and deletion of an item. Via its network visualization panel, this feature facilitates the test of system robustness or demonstration. Once a user manipulates the network topology via GUI, the updates will be synchronized to the network which is running. This feature also work in the other direction. When a new peer has successfully joined or left the network, the change of this will also be reflected to the network visualization panel as well.

The features discussed above have been fully implemented in Java. The current version contains three parts: a GUI based user interface for configuring and managing the evaluation, a test server with a database for collecting and computing all test results, as well as a run time environment that enables to run the targeted P2P system on multiple hardwares. Each part is able to run in a dedicated process and they can be deployed on different machines. The architecture of the P2P evaluation framework is shown in Figure 8.15.

With solid blue colored arrows, it shows the experiment configurations and deployments before the actual run of test; while with red colored dot arrows, it shows the live performance monitoring,

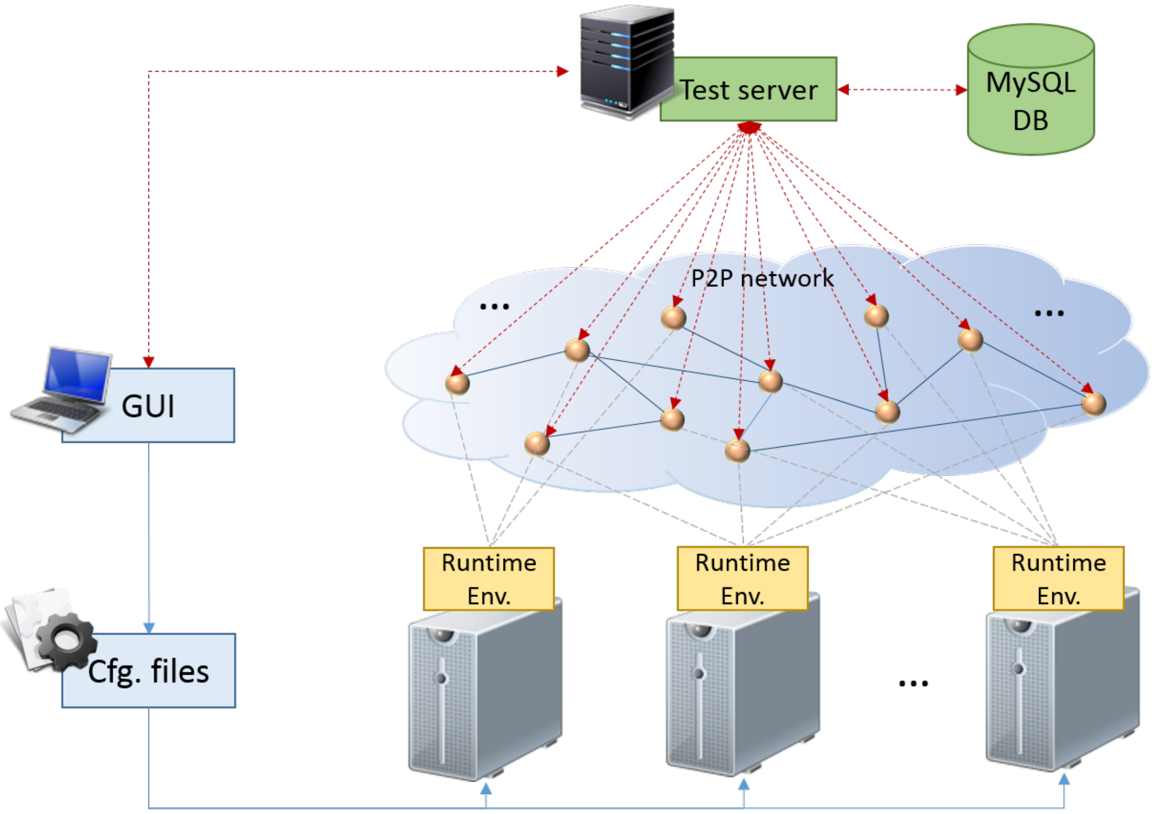


Fig. 8.15 P2P evaluation framework architecture.

peer test result reporting as well as the result persistence processes. To proceed a test, the first step is to use GUI to generate configuration files for the targeted P2P system. This includes the pre-test settings such as the network topology generation, test collection, item selection, query and item topic distribution as well as the distributed run time environment configuration. On top of this, the GUI generates configuration files of the targeted P2P system, which will be deployed to each machine with pre-installed run time environment.

By starting the test on each machine (e.g. via remote control), peers will be created and their connections will be established automatically. Once the initialization is finished, the system is ready for evaluation. Tests can be launched via GUI by issuing queries as described above. Such command of issuing a query will be sent to the run time environment on the specific host machine where the planned requester peer is accommodated. The requester peer issues a query, after it receives the command. Besides, that peer will also reports the status of the query to the test server, when the query is returned. For this, a test reporter component is nested in each peer when it gets initialized. Such a reporter is only required for the purpose of evaluation, which is not needed for any peer client in a deliverable application. Once a test report has been received by the test server, the latter logs the query details into a MySQL database. Based on these query details, the real time system performance measures can be computed. This process is driven by the demand from the GUI (cf.

Figure 8.14), where the visualization of the concerned metrics are supposed to be displayed.

In the sequel, we briefly introduce the important parts of the implementation. As discussed above, the aim of pre-test settings is to support the user to customize the targeted P2P system in terms of its features, such as item distribution, test collection, etc., in order to generate the system configuration files needed by its initialization. Depicted in the class diagram (cf. Figure 8.16), the frame (*TestFrameworkMain*) of GUI is associated with its underlying data model *TestModel*, where the data of configuration for all the features is stored. When a user specifies the class paths or values for features, the data in *TestModel* is correspondingly updated. These data will be additionally persistent into the configuration files that are used by the run time environment at each host machine for the initialization of the targeted system. Once the data of configuration has passed the validation, the generation of the configuration files is delegated to the implementation of an abstract class *NetworkConfigurationCreator*. By customizing its callback methods *generateVertexStr()* and *generateItemStr()*, a user is allowed to generate the encoding of peers and items for the specific application. Such encodings will be loaded during the system initialization. Moreover, *TestModel* maintains an instance of type *TestClient* class, which works as a nested client for retrieving the test result and issuing the command via GUI during the run of the system.

After copying the generated configuration files to the run time environment at each host machine, the system can be launched manually or via remote control means. Each run time environment instantiates an implementation class of the *AbstractNetworkLauncher*, which maintains a set of *AbstractPeerProcess* instances. Each *AbstractPeerProcess* wraps an *IPeer* and instantiates it on demand. *AbstractPeerProcess* is responsible for loading and instantiating the configured properties, such as item loader, communicator, etc., for specific instance of *Peer*. Once this is done, a callback method *afterInitialization()* is invoked, which allows a user to associate the *Peer* instance to a customized wrapper, like *S2P2PPeer*, *MyMediaPeer*. Subsequently, the run time environment calls the *launch()* method of each peer, which initializes the internal components of a peer and starts its live cycle.

Another work of the *AbstractNetworkLauncher* class is to accept the commands (e.g. batch query) from GUI and assign them to the correct peers for execution. For this purpose, it exposes a series callback methods for processing them. A user can alternatively implement these callback methods for specific actions or leverage the default implementations in the class *NetworkLauncher*. In detail, a command coming from the GUI is wrapped in an instance of type *ICommand*. Such command is accepted by an independent long term nested thread *CommandAcceptorThread* in *AbstractNetworkLauncher*. As long as a command is received, *CommandAcceptorThread* put it into a FIFO queue *commandQueue* and immediately re-listen on the port. The execution of a command is managed by another thread, called *CommandExecutorThread*, which analysis the command and invokes the corresponding callback method.

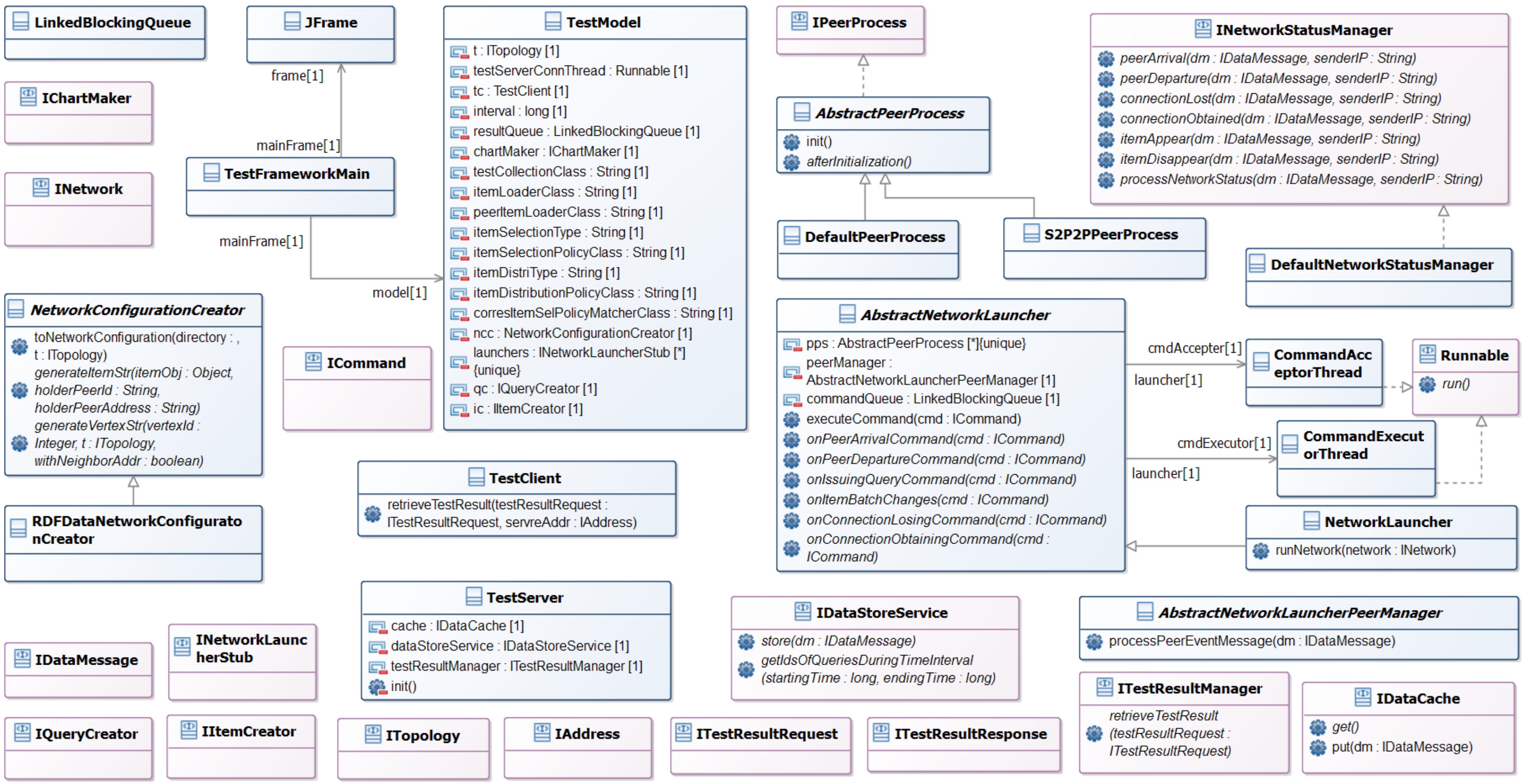


Fig. 8.16 Class diagram of the important components of P2P evaluation framework.

Besides processing the command from GUI, *AbstractNetworkLauncher* also manages to report the change event (e.g. peer arrival) of P2P network to GUI at run time. This work is accomplished by the method *processPeerEventMessage()* in *AbstractNetworkLauncherPeerManager* class. Likewise, *AbstractNetworkLauncherPeerManager* class also maintains a FIFO queue as well as two threads for accepting and processing an event. Such event is wrapped in an instance of type *IDataMessage*.

P2P evaluation framework allows user to check the real time performance of a P2P system during its run. For this, each peer is additionally keeping a nested test result reporter component, which reports the details of each query to the test server (*TestServer* class), when that query is returned. Please note that the test server is different with the run time environment and it can be separately deployed to a machine. As the current implementation, *TestServer* class takes use of a database (e.g. MySQL) to record the query details. In details, on receiving each report from a peer, *TestServer* put it into its FIFO cache *IDataCache*. That report will be further handled by another thread that inserts the report data into database by using the *IDataStoreService*. The reported data is used to compute real time system performance measures. This computation is driven by the periodic demands from GUI. The latter retrieves the new performance data and updates the visualization charts in the performance panel. For this purpose, the test client instance of type *TestClient* in the *TestModel* sends a *ITestResultRequest* message to the test server at the ending time of each *interval*. In a synchronous fashion, it updates the visualization once the reply *ITestResultResponse* is received.

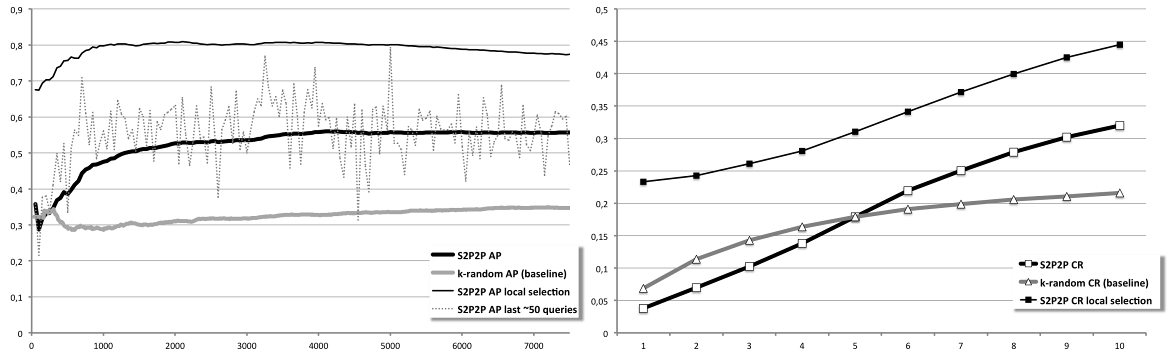
8.4 Practical Evaluation

Settings: The practical evaluation of the MyMedia system search performance has been conducted with a random power-law graph based unstructured P2P network (The skew value is 1.5, $TTL = 10$, k = 2) containing 300 peers[11]. The query distribution is uniform at random and their topic distribution follows the Zipf's law ($\beta = 1.05$). In order to simulate a real practical scenario, MyMedia peers has been distributively deployed to clustered 10 machines, where each runs a P2P evaluation framework run time environment that hosts 30 peers. Each machine was equipped with 6 core Intel Xeon CPUs running at 2.67 GHz and 12 GB of RAM running Linux OS version 3.2.0-58. For these small-scale experiments, the MyMedia peers were configured to use S2P2P, in particular the service selector iSeM, but not their DSDR component for semantic replication. The evaluation hires a user judged test collection, which is able to generate queries with a set of relevant items.

Technical results: The experimental results showed that the average precisions for peers using S2P2P were up to 1.6 times better (AP = .56) than those using traditional k-random search (AP = .35) and stabilized already after 2k queries. In fact, the MyMedia peers achieved a significantly higher search precision with S2P2P at any time (cf. Figure 8.17a, S2P2P AP for the latest 50 queries). Though the

---

[11] This limitation is based on the estimation by TIFF-54 organisation members of the average number of festival on-line users who would most likely use the MyMedia at the same time.



(a) Average precisions of S2P2P and k-random search.

(b) Cumulative recall of S2P2P and k-random search.

Fig. 8.17 Search performance of MyMedia system.

local search performance of each peer using its iSeM component was very high on average (AP = .79), the lower average precision of the whole P2P system with S2P2P is mainly due to the general problem of searching unstructured P2P networks with sparse item distributions. However, in contrast to k-random search, S2P2P was able to mitigate this problem to a large extent.

In addition, S2P2P was able to outperform the baseline in terms of cumulative recall with $CR_{10} = .32$ (versus $CR_{10} = .22$ for k-random). In the latter respect, its much more linear behavior (cf. Figure 8.17b) shows its ability to achieve an increase in recall even at very late steps of query routing up to the given TTL. This is mainly due to the advanced routing scheme of S2P2P where peers are making use of their local observations of semantic expertises of other peers, in particular of those which would not be reached by random walks within the TTL or due to blind paths and therefore can contribute more to a recall in late hops on average. Such knowledge is propagated by MyMedia peers on the path in the piggy-backed data of traversing queries.

The messaging overhead of S2P2P compared to the baseline appears very small: S2P2P just produced 1.22 times more messages. Due to their semantic expertise-driven query routing mechanism, the S2P2P peers are more likely to find and communicate with more relevant expert peers within the TTL than those performing a k-random search. The latter type of search often reached dead-ends very early, within 10 hops in the given network overlay. On average, a peer using S2P2P (baseline) processed 531 (435) queries.

The evaluation also covers the energy consumption of peer mobile device when it searches for live videos. For this, we set up a simple wireless mesh network with three MyMedia peers on Android devices (one Samsung S3, and two Google Nexus 7 Tablets) of which one (Google Nexus) was recording a live stream and one (on Samsung S3) was searching for it using S2P2P The amount of energy the latter peer consumed for its semantic P2P search related activities on the S3 was negligible ($0.5J$ for CPU, $0.9J$ for WiFi) compared to the one consumed for MPEG-DASH en/decoding and playback of the found live stream (cf. Figure 8.18); the WiFi module was put into the *high* energy state only for a very short time period.

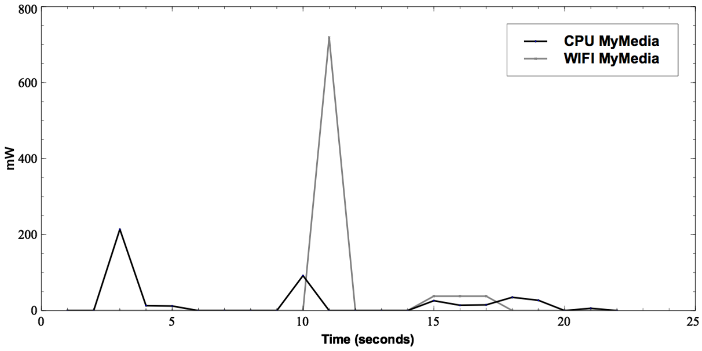


Fig. 8.18 Energy consumption of the MyMedia system during P2P search.

User acceptance in TIFF-54 festival: The user acceptance of the MyMedia features of the TIFF EventLive app was evaluated based on the collected responses of 18 test users (staff and visitors) from the TIFF-ELE group to a brief user questionnaire on the subject and additional personal interviews. Ovverall, the majority of them considered the MyMedia features very useful for enriching their personal festival experience (65% agreed, 30% neutral) and easy to use (75% agreed, 25% neutral).

In particular, the integrated search and joint live experiencing of events on the premise, POIs, waiting queues and partys shared through live recordings of other visitors or buddies was considered most important. The tagging functionality of the MyMedia peer of the festival app was accepted in general by the users (68% agreed, 33% neutral) though some of them (22%/5%) felt uncomfortable with using a tag cloud for semantic annotation/querying of relevant videos.

Most of the users (95% agreed) were positive about and satisfied with the experienced quality of the search results and displayed P2P live recordings. The measured average latency of about 1 second for P2P search and 4 seconds for P2P live streaming in peer groups of two to ten mobile devices on the TIFF-54 festival premise with good wireless network conditions were considered reasonable and acceptable by all users. Few users raised concerns about the lack of integrated mobile app security technologies in the festival app for user data privacy management. A follow-up user evaluation of the MyMedia-empowered festival app is currently in preparation for the TIFF-55 festival.

8.5 Related Work

In general, the MyMedia system is most related to work on mobile P2P live streaming and mobile semantic P2P search. However, to the best of our knowledge, there is currently no system which provides both features and exploits the MPEG-DASH standard for dynamic adaptive P2P live streaming over HTTP.

For example, Tribler Mobile [166] is an open-source mobile app for Android devices for video streaming between users in a network with DHT-based overlay. The underlying Tribler system is augmenting the distribution capacities of the network with user-given content boosters and high-performance desktop computers. A video is not directly published by a peer on a mobile device to its consuming peers on mobile devices. Instead, the publisher uploads the video to all of its boosters in the network based on subscription services and then exits the app to save energy. The implemented publish-subscription mechanism relies on the use of the Twitter service for tweeting respective video links (PPSP URLs) by the publisher to Twitter subscribed consumers of a given peer group. Tribler uses decentralized DHT-based peer tracking for reverse peer discovery by which new or existing peers can periodically request DHT nodes for IP addresses and UDP ports of other peers.

However, Tribler Mobile differs from MyMedia in several aspects: the Tribler system features neither a semantic P2P search for relevant videos and running live recordings, nor a P2P live streaming of live recordings from the mobile device of the recording peer to those of its consumers, nor a network capacity-based adaptive streaming based on the MPEG-DASH standard, and utilizes a structured DHT-based rather than an unstructured P2P overlay.

The Peer2View system for P2P live streaming over HTTP [313] uses a centralized tracking service for peer discovery in networks with unstructured (random) overlay, and employs a gossip-based protocol to collect QoS information such as maximum/average throughput, connectivity information, playback quality, buffering point, and network location from other peers. During P2P streaming each peer selects neighbor peers for the delivery of the stream based on this collected information, in particular their heuristically estimated upload capacities. A similar approach is taken by RapidStream [86]. However, both systems do not feature any semantic P2P video search, and MyMedia does not depend on centralized tracking for peer discovery and peer-assisted streaming of MPEG-DASH videos and live recordings.

On the other hand, as mentioned above, semantic P2P systems [345] exploit techniques of semantic reasoning on annotated media items in order to significantly improve the search precision over non-semantic approaches in P2P networks. Only a few are featuring semantic P2P search in mobile environments.

For example, the SMSN system [222] performs a semantic-based search for relevant resources and users in mobile ad hoc networks. It employs a semantic distance vector routing protocol in which peers distribute profiles of users and resources semantically described in OWL2 within a given range (k-hop limit) of their neighborhood in the network with an unstructured P2P overlay. Each peer maintains its local view on the semantic overlay in terms of which peers it knows of can provide what ontology concepts and issues semantic query walkers to them depending on the degree of semantic similarity with the given query. The open-source system has been deployed and evaluated on a set of twelve SonyEricsson K750 cellular phones and uses J2ME Bluetooth API (JSR 82) for mobile ad hoc networking.

The MP2PSW system [208] combines search for semantically relevant items from heterogeneous resources: Web servers, P2P networks in the Web and mobile ad hoc networks. Search in the (non-mobile) P2P network is performed using Chord ring-based query routing, utilizing a DHT for item identification and location. In contrast, search in the mobile network is performed using IPv6 anycast, thus effectively treating the whole network as one single black box with exactly one address and unknown physical receiver peer. While this proves to be very efficient with respect to query response time, it is not actually a search in (parts of) the network, but an efficient scheme to distribute workload on coequal peers that optimally provide synchronized content.

This is considerably different from what the MyMedia system achieves, which indeed allows for semantic search among mobile peers that provide different content. In fact, MP2PSW does not aim at providing an integrated solution for semantic P2P search in mobile networks, but rather at computing search results from different sources efficiently and in parallel. Besides, MP2PSW assumes a query agent running on non-mobile hardware in the Internet. The user may use her mobile device to issue queries and interpret results, but the actual query processing and result aggregation is performed by this central agent.

8.6 Summary

In this chapter, we have presented the MyMedia system as well as the design and implementation of its supporting components: the P2P framework, P2P evaluation framework, S2P2P and DSDR. MyMedia features a high-performance semantic P2P search and a dynamic adaptive live streaming of annotated MPEG-DASH videos from mobile to mobile devices over HTTP in wireless networks with an unstructured and semantic P2P overlay. It has been built on top of the Java based P2P framework, which provides the basic peer communication, query routing, item selection and a series of extendable sub-components. As the core of the MyMedia system for media content search, the semantic search strategy S2P2P and the semantic replication scheme DSDR were designed and implemented based on the P2P framework. Moreover, the P2P evaluation framework with many customizable properties was proposed for the actual evaluation of an unstructured P2P system.

The main theoretical and practical contributions of this thesis were presented in Chapters 3 through 8. The theoretical part comprises a semantic-based search strategy S2P2P, a dynamic data replication scheme (DSDR), a semantic service composition approach (SPSC) and an efficient semantic 3D scene selection approach (iRep3D), while the practical contribution consists of the design and implementation of the iRep3D system in the Collaborate3D project, as well as these important components in the SocialSensor project: the P2P framework, S2P2P, DSDR and the P2P evaluation framework of the MyMedia system. In the next chapter, we will conclude this thesis, providing brief answers to the research questions proposed in Chapter 1, as well as discussing future work.

Chapter 9

Conclusions

This thesis has presented investigations on several representative research questions regarding semantic-based search and composition in unstructured P2P networks. It has also introduced practical implementations of the proposed approaches for applications in the fields of collaborative virtual product design as well as social media content sharing. In Section 9.1, we answer the targeted research questions proposed in Chapter 1. This is followed by a discussion on interesting research questions that will be studied in the future.

9.1 Answers to Research Questions

In this section, we provide answers to the research questions proposed in Section 1.1 based on the corresponding scientific contributions and their experimental evaluation results.

Challenge 1: Expert peer based semantic search in unstructured P2P networks
How can peers in an unstructured P2P network make use of their observed semantic knowledge for conducting collaborative query routing with a reasonable tradeoff between performance and communication overhead?

Each peer, with the S2P2P strategy, performs k-nearest-neighbor-based clustering over observed queries (demands) and items (supplies). This process results in two sets of clustered remote peers over topics, in which each peer is regarded as a member on some demand/supply topic with numeric strength values. Based on this, a peer is able to suggest a routing path for a query by means of the derived knowledge above as well as its local view on network topology. Instead of merely containing immediate neighbor(s), such a path can lead a query to traverse a sequence of expert peers on the requested topic and at the same time achieve minimal total inverse expertise gain per traffic cost under the query TTL constraint. As an approximate solution, the path suggestion process always concatenates the shortest path to the next expert peer, such that the total inverse expertise gain per traffic cost reaches the minimum. In addition, this process is done collaboratively by peers on the query path. Each peer contributes to the final results, relying on its local knowledge.

With a path suggestion, each peer in S2P2P is able to know in general which peers will be traversed by the current walker. Via piggybacking the description of an selected item, a suitable destination peer has a chance to receive the item information. The observed demands and supplies are used to choose the item and destination peers for the item information dissemination.

Challenge 2: Dynamic semantic data replication strategy in unstructured P2P networks
How can peers leverage their observed semantic information on queries and items to collaboratively conduct data replication?

The semantic information is used to annotate the item and query. They represent the supply and demand, respectively. The essence of data replication is to find the demand-supply relations between peers. With DSDR, each peer is able to periodically observe the semantic demand of any traversing query, as well as the semantic supply of piggybacked items. Unlike P2P search strategies, for data replication, the mapping between the semantic demands and peers are as important as the mapping of supplies. Together with the observed items, each peer is able to conclude the demand-supply relations from its view. Overall, these relations form the semantic network overlay. Based on this, the replication is carried out.

At the end of each observation period, each peer computes a demander group of remote peers, from which at least one unsatisfied query is observed during the current period, and that query is semantically similar to an unsatisfied query issued by the local peer in the same period. Via message exchange, such peers form a demander group on certain topic. Once this is done, for each known item on other remote peers, the peers in a demander group collaboratively determine the replication destination peer that is optimal in that demander group in terms of semantic gain.

Challenge 3: Functional semantic service composition in unstructured P2P networks
In an unstructured P2P network, how can peers collaboratively perform functional IOPE-level semantic service composition that is able to mitigate the risk of failure caused by dead-ends and gives reasonable completeness?

In SPSC, the collaborative composition is accomplished by two components: peer local composition with heuristics and query routing with service information propagation. The peer local composition process of SPSC enables each peer to exhaustively try to chain its known services to the current workflow, under the control of a heuristic. After each chaining of service, the guarding heuristic estimates to what extent such a chaining can contribute to achieving the overall requested output and effect. Services effectively increasing the heuristic value are marked as useful, while those that do not cause an increase are treated as candidates. In addition, any alternative service is considered, and the sub-queries are issued. This guarantees that the composition process will investigate any possible solution and therefore succeed in alleviating the risk of failure with dead-endds. In addition, SPSC uses a memorization strategy to let queries piggyback potentially useful services. They might work as key predecessors/successors in the workflow, though those services cannot be temporarily chained to the current workflow. Determined by means of the statistics on the relevant queries in the past, such memorization facilitates service information dissemination and increases the chance of forming correct solutions. The collaboration of peers during the composition is directed by the query routing strategy. For this, SPSC makes it possible to suggest a path for routing a (sub-)query, which contains a sequence of key peers for which the total inverse importance score per traffic cost is minimized.

In an unstructured P2P network, each peer only has a limited view of the network topology and services. Without flooding over the network, a composition system with TTL restricted queries cannot guarantee its full completeness, but only a lower bound. It is a probability value, meaning the lowest chance of finding a correct solution for a query, if the solution can be formed by the services on certain peers reachable by a query within TTL limitation. By theoretical analysis, a lower bound of completeness for a generic composition process in unstructured P2P networks has been derived. Its magnitude is determined by the density of useful services as well as the accuracy of the routing strategy. In line with the evolution of the semantic network overlay, the probability of finding a solution by SPSC increases sub-linearly and finally converges to 1. Apart from the completeness, the soundness of the SPSC heuristic has also been proved, in two respects. One is the the soundness of the heuristic itself, which guarantees the correctness of a solution to some query, if its corresponding heuristic value reaches a bound. The other is the soundness of collaboration, which guarantees the truth of a set of key invariants throughout the whole composition process for a query.

Challenge 4: Efficient hybrid semantic selection of annotated 3D scenes for P2P search

In P2P-based search, how can 3D scenes with conceptual and functional descriptions be indexed for the purpose of efficient selection?

In iRep3D, the conceptual, functional and geometric features of 3D scenes are indexed separately. Given a hybrid query that asks for 3D scenes with criteria in these three aspects, three sub-processes can be executed in parallel, and their sub-results are aggregated as the final answer list. As for the indexing of conceptual description, a scene concept ontology is employed from the iRep3D repository, and each of its concepts corresponds to a ranked list of 3D scenes that to some extent can be categorized as an individual representative of that concept. In addition, a vocabulary of primitive terms is presumed to be shared by iRep3D and each scene publisher (creator). The latter describes her 3D scene with a local concept defined using the shared vocabulary. During the conceptual indexing, the scene concept of a 3D scene is categorized (to multiple concepts in iRep3D scene ontology) by means of a concept abduction-based similarity measure. The ranking score is simply the similarity value. Meanwhile, for the indexing of functional descriptions, a service parameter ontology and a set of predicates signatures are maintained in iRep3D. Based on the same shared vocabulary, a publisher specifies the functionality of a 3D scene by OWL-S semantic services with IOPE. Each service is concerned with a simulated function of the 3D scene. The functional indexing of a scene has two aspects: IO-level and PE-level indexing. The IO-level indexing leverages the same similarity measure and categorizes a scene to some service parameter concept. The difference is that each service parameter concept in iRep3D corresponds to two ranked lists for distinguishing the parameter direction (I or O). The ranking score is the similarity value that additionally adjusted based on its statistical weights over the services and parameters. The PE-level indexing checks whether each pre-defined predicate is used by a 3D scene individual. If it does, that scene is indexed into the corresponding ranked list of the predicate. The ranking score is the plausibility value of that predicate over the whole precondition or effect formula. Similar to the IO-level, each pre-defined predicate is associated with two ranked lists in order to differentiate the directions (i.e. P or E). For the indexing of geometric features, iRep3D maintains a set of standard features defined by XML3D, COLLADA and X3D specifications. Each feature contains a set of attributes and each attribute corresponds to a B+ tree. According to the magnitude of scene attributes, a 3D scene is indexed into those B+ trees once its attribute is instantiated by the scene. Based on the indexes, a hybrid query can be processed efficiently. As mentioned above, three sub-queries are processed in parallel. Each yields a ranked list of 3D scenes that is relevant to the query in terms of the aspect concerned. Fagin's threshold algorithm is employed for the final aggregation.

The essence of categorizing is to determine the subsumption relations between scene concepts. For this, strict logical reasoning returns failure, when any conflict on a primitive term occurs. iRep3D overcomes this and allows approximated concept subsumption by a concept abduction-based approximated similarity measure. It is derived based on the abduction theory over concepts in negation normal form. The essence of iRep3D's solution contains two aspects: determining conflict strengths over primitive terms and performing ontology-based weighting over conflicting primitive terms. The novelty of this measure lies in the weighting aspect. By checking how a conflicting part affects the hierarchical relations in the scene concept ontology, the weighting process computes importance scores for each conflicting primitive term. This is helpful to more precisely estimate the approximated concept subsumption score.

9.2 Future Work

For future work, in this section, we discuss the following three interesting research questions:

Dynamic semantic-based replica management in unstructured P2P networks: The goal of the proposed dynamic semantic data replication scheme DSDR is to investigate and use the semantic-based information to replicate data items to proper peers in order to increase the performance of the k-walkers-based search in unstructured P2P networks. Relying on the semantic technology, a peer is now able to replicate data according to the predicted topics on demand. However, the storage space on computational devices is limited. Sometimes, a user device may not be able to afford adequate space for storing certain data, even though such a replica is determined to be quite helpful for resolving future requests. This indicates that a peer might need to remove some data from its local storage or transmit it to other peers. Hence, the selection of the data to be removed or transmitted is crucial to the performance of the search during the evolution of the P2P system. In this paradigm, there exist some efforts targeting this problem, such as [412] [216] [390]. However, the semantic-based data completeness and integrity on topics has been neglected. This issue is particularly relevant for unstructured P2P search on RDF data. Instead of returning a piece of triples, it aims at answering user queries with entire data objects that are formed on demand by the RDF triples from multiple loosely-coupled sources. Furthermore, new data and users can (dis)appear in an unstructured P2P network (e.g. a mobile social network). How a peer with limited storage space can quickly adapt to a change and also guarantee high search performance and data availability is an interesting research question.

Function oriented recognition of 3D objects: iRep3D supports the semantic indexing and efficient retrieval of 3D scenes. It is able to answer complex hybrid user requests in terms of conceptual, functional and geometric aspects. In this context, 3D scenes, besides the geometric features, are assumed to have been annotated with conceptual and functional descriptions. However, the manual annotation requires a user to have the prerequisite knowledge of the logic and semantic services. With respect to automatic annotation, the current research, like [340] [140] [316] [19] [210] [409] [347] [402] [359], etc., only provides the conceptual recognition of 3D objects. Unfortunately, to the best of our knowledge, no approach responds to automatic functionality-oriented annotation. However, this would be an essential part of real intelligence, as similar kinds of tasks (figure out the usages of entities in the world based on their features, e.g. geometric features and textures) have been performed by human beings for more than 10 thousand years. This has played a vital role in human growth, learning and development. In the artificial intelligence field, it will be a significant improvement if machines are capable of behaving similarly, helping to make robots (or software agents) act truly intelligently, detecting the world, recognizing potential tools, and making use of them for a given task.

Semantic web service composition with conditional controls: A big difference between manual and automatic service composition (orchestration) lies in the quality and executability of their results. A workflow resulting from manual service composition (e.g. model-driven service orchestration [250]) can contain not only the splitting and joining operators but also conditional controls, such as if-else, loops, exception handlers, etc., which direct the actual execution by means of the values of certain variables. To a great extent, these conditional controls are essential to the robustness of a workflow, and therefore they determine its executability. To the best of our knowledge, no stateless service composition method working under an open world assumption can produce such workflows automatically, where those key conditional controls over concrete variable values are involved. From this point of view, automatic service composition is still at an abstract level, i.e. producing a conceptual sequence of services, not robust enough for realistic execution. Enhancing the result quality and executablity of automatic service composition is a challenge. In this regard, a sub-problem is precise automatic parameter binding. State-based service composition approaches, like OWLS-XPlan [198], resolve this in a state-driven manner. Namely, if two services can be chained when one of them can produce a state that is consumed by the other, such a state is then generalized to a variable, in order to obtain an overall services orchestration from a plan. As discussed in Chapter 5, the workflow resulting from this may not be flexible and robust enough for real execution, since arbitrary input values of variables can lead to unexpected states. Stateless service composition planners, for instance SPSC, resolve the binding by variable type matching. This actually causes inaccuracy. Another sub-problem of the challenge above is how to introduce extra variables (e.g. a global variable) into a workflow once they are predicted to be necessary for execution. This issue becomes crucial if one wishes to use a recursive loop depending on a concrete value.

References

[1] Aberer, K., Cudré-Mauroux, P., Datta, A., Despotovic, Z., Hauswirth, M., Punceva, M., and Schmidt, R. (2003). P-grid: a self-organizing structured p2p system. *ACM SIGMOD Record*, 32(3):29–33.

[2] Aberer, K., Cudré-Mauroux, P., Hauswirth, M., and Van Pelt, T. (2004). Gridvine: Building internet-scale semantic overlay networks. In *The Semantic Web–ISWC 2004*, pages 107–121. Springer.

[3] Adamic, L. A. and Huberman, B. A. (2002). Zipf's law and the internet. *Glottometrics*, 3(1):143–150.

[4] Agarwal, S., Handschuh, S., and Staab, S. (2004). Annotation, composition and invocation of semantic web services. *Journal of Web Semantics*, 2(1):31–48.

[5] Agre, G. and Marinova, Z. (2007). An infrawebs approach to dynamic composition of semantic web services. *Cybernetics and Information Technologies*, 7(1):45–61.

[6] Aguilera, U. and López-de Ipiña, D. (2014). An architecture for automatic service composition in manet using a distributed service graph. *Future Generation Computer Systems*, 34:176–189.

[7] Aiello, M., van Benthem, N., and El Khoury, E. (2008). Visualizing compositions of services from large repositories. In *Proceedings of the 10th IEEE Conference on E-Commerce Technology and the 5th IEEE Conference on Enterprise Computing, E-Commerce and E-Services*, pages 359–362. IEEE.

[8] Akbarinia, R., Martins, V., Pacitti, E., and Valduriez, P. (2006). Design and implementation of atlas p2p architecture. *Global Data Management*, 8:98.

[9] Akguel, C., Sankur, B., Yemez, Y., and Schmitt, F. (2010). Similarity learning for 3d object retrieval using relevance feedback and risk minimization. *International Journal of Computer Vision*, 89(2):392–407.

[10] Al-Oqily, I., Al-Aubidy, K. M., Alshtnawi, A., and Al-Sarayreh, K. T. (2014). An efficient media ports resource discovery for service networks. *International Journal of Business Information Systems*, 15(2):222–243.

[11] Al-Oqily, I. and Karmouch, A. (2011). A decentralized self-organizing service composition for autonomic entities. *ACM Transactions on Autonomous and Adaptive Systems*, 6(1):1–18.

[12] Al Ridhawi, Y. and Karmouch, A. (2015). Decentralized plan-free semantic-based service composition in mobile networks. *IEEE Transactions on Services Computing*, 8(1):17–31.

[13] Ali, L., Janson, T., and Schindelhauer, C. (2014). Towards load balancing and parallelizing of rdf query processing in p2p based distributed rdf data stores. In *Proceedings of the 22nd Euromicro International Conference on Parallel, Distributed and Network-Based Processing*, pages 307–311. IEEE.

[14] Alpcan, T., Bauckhage, C., and Kotsovinos, E. (2007). Towards 3d internet: Why, what, and how? In *Proceedings of the 2007 International Conference on Cyberworlds*, pages 95–99. IEEE.

[15] Alvez, C. and Vecchietti, A. (2011). Efficiency analysis in content based image retrieval using rdf annotations. *Advances in Soft Computing. 10th Mexican International Conference on Artificial Intelligence, Puebla, Mexico*, pages 285–296.

[16] Armetta, F., Haddad, M., Hassasz, S., and Kheddouci, H. (2012). A self-organized system improving inner topology for data sharing efficiency. In *Proceedings of the 2012 IEEE Conference on Evolving and Adaptive Intelligent Systems*, pages 145–150. IEEE.

[17] Ata, S., Gotoh, Y., and Murata, M. (2003). Replication strategies in peer-to-peer services over power-law overlay networks. In *Proceedings of the 7th Asia-Pacific Network Operations and Management Symposium*. IEEE.

[18] Baader, F. (2003). *The description logic handbook: theory, implementation, and applications*. Cambridge University Press.

[19] Ballan, L., Bertini, M., Del Bimbo, A., Seidenari, L., and Serra, G. (2011). Event detection and recognition for semantic annotation of video. *Multimedia Tools and Applications*, 51(1):279–302.

[20] Banaei-Kashani, F., Chen, C.-C., and Shahabi, C. (2004). Wspds: Web services peer-to-peer discovery service. In *Proceedings of the 5th International Conference on Internet Computing*, pages 733–743. Citeseer.

[21] Barjini, H., Othman, M., and Ibrahim, H. (2011). Quickflood: an efficient search algorithm for unstructured peer-to-peer networks. In *Networked Digital Technologies*, pages 82–92. Springer.

[22] Bartalos, P. and Bieliková, M. (2010). Qos aware semantic web service composition approach considering pre/postconditions. In *Proceedings of the 17th IEEE International Conference on Web Services*, pages 345–352. IEEE.

[23] Basters, U. and Klusch, M. (2006). Rs2d: fast adaptive search for semantic web services in unstructured p2p networks. In *Proceedings of the 5th International Semantic Web Conference*, pages 87–100. Springer.

[24] Bener, A. B., Ozadali, V., and Ilhan, E. S. (2009). Semantic matchmaker with precondition and effect matching using swrl. *Expert Systems with Applications*, 36(5):9371–9377.

[25] Benslimane, D., Barhamgi, M., Cuppens, F., Morvan, F., Defude, B., and Nageba, E. (2013). Pairse: a privacy-preserving service-oriented data integration system. *ACM SIGMOD Record*, 42(3):42–47.

[26] Berner, A., Bokeloh, M., Wand, M., Schilling, A., and Seidel, H.-P. (2008). A graph-based approach to symmetry detection. In *Proceedings of the 5th Eurographics/IEEE VGTC conference on Point-Based Graphics*, pages 1–8. Eurographics Association.

[27] Berners-Lee, T., Hendler, J., Lassila, O., et al. (2001). The semantic web. *Scientific American*, 284(5):28–37.

[28] Bianchini, D., De Antonellis, V., and Melchiori, M. (2009). Service-based semantic search in p2p systems. In *Proceedings of the 7th IEEE European Conference on Web Services*, pages 7–16. IEEE.

[29] Bianchini, D., De Antonellis, V., Melchiori, M., and Salvi, D. (2008a). On-the-fly collaboration in distributed systems through service semantic overlay. In *Proceedings of the 10th International Conference on Information Integration and Web-based Applications & Services*, pages 407–410. ACM.

[30] Bianchini, D., De Antonellis, V., Melchiori, M., and Salvi, D. (2008b). A semantic overlay for service discovery across web information systems. In *Web Information Systems Engineering-WISE 2008*, pages 292–306. Springer.

[31] Bicsi, B. (2002). *Network Design Basics for Cabling Professionals*. City: McGraw-Hill Professional.

[32] Bilasco, I. M., Gensel, J., Villanova-Oliver, M., and Martin, H. (2006). An mpeg-7 framework enhancing the reuse of 3d models. In *Proceedings of the 7th International Conference on 3D Web Technology*, pages 65–74. ACM.

[33] Bisnik, N. and Abouzeid, A. A. (2007). Optimizing random walk search algorithms in p2p networks. *Computer Networks*, 51(6):1499–1514.

[34] Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

[35] Bloom, B. H. (1970). Space/time trade-offs in hash coding with allowable errors. *Communications of the ACM*, 13(7):422–426.

[36] Blum, A. L. and Furst, M. L. (1997). Fast planning through planning graph analysis. *Artificial Intelligence*, 90(1):281–300.

[37] Brunner, R., Caminero, A. C., Rana, O. F., Freitag, F., and Navarro, L. (2012). Network-aware summarisation for resource discovery in p2p-content networks. *Future Generation Computer Systems*, 28(3):563–572.

[38] Bustos, B., Keim, D., Saupe, D., and Schreck, T. (2007). Content-based 3d object retrieval. *IEEE Computer Graphics and Applications*, 27(4):22–27.

[39] Cai, M. and Frank, M. (2004). Rdfpeers: a scalable distributed rdf repository based on a structured peer-to-peer network. In *Proceedings of the 13th international conference on World Wide Web*, pages 650–657. ACM.

[40] Cai, M., Frank, M., Chen, J., and Szekely, P. (2004). Maan: A multi-attribute addressable network for grid information services. *Journal of Grid Computing*, 2(1):3–14.

[41] Camossi, E., Giannini, F., and Monti, M. (2007). Deriving functionality from 3d shapes: Ontology driven annotation and retrieval. *Computer-Aided Design & Applications*, 4(6):773–782.

[42] Cao, X., Kapahnke, P., and Klusch, M. (2015). Spsc: Efficient composition of semantic services in unstructured p2p networks. In *Proceedings of the 12th Extended Semantic Web Conference*, ESWC'15. Springer.

[43] Cao, X. and Klusch, M. (2012a). Dynamic semantic data replication for k-random search in peer-to-peer networks. In *Proceedings of the 11th IEEE International Symposium on Network Computing and Applications*, pages 20–27. IEEE.

[44] Cao, X. and Klusch, M. (2012b). Semantic indexing for efficient multimedia retrieval. In *Proceedings of 10th International Workshop on Adaptive Multimedia Retrieval. Copenhagen, Denmark*, pages 165–180. Springer.

[45] Cao, X. and Klusch, M. (2013a). Advanced semantic deep search for 3d scenes. In *Proceedings of 7th International Conference on Semantic Computing. Irvine, CA, USA*, pages 236–243. IEEE.

[46] Cao, X. and Klusch, M. (2013b). irep3d: Efficient semantic 3d scene retrieval. In *Proceedings of the 8th International Conference on Computer Vision Theory and Applications. Barcelona, Spain*, pages 19–28. ScitePress.

[47] Cao, X. and Klusch, M. (2013c). S2p2p: Semantic search in unstructured peer-to-peer networks. In *Proceedings of the 15th International Conference on HighPerformance Computing and Communications*, pages 971–978. IEEE.

[48] Cao, X. and Klusch, M. (2013d). Semantic-driven k-walker search in unstructured peer-to-peer networks. *IEEE Computer Society Special Technical Community on Social Networking E-Letter (STCSN)*, 1(3).

[49] Cardinale, Y., El Haddad, J., Manouvrier, M., and Rukoz, M. (2010). Web service selection for transactional composition. *Procedia Computer Science*, 1(1):2689–2698.

[50] Carminati, B., Ferrari, E., Heatherly, R., Kantarcioglu, M., and Thuraisingham, B. (2011). Semantic web-based social network access control. *Computers & security*, 30(2):108–115.

[51] Cassar, G., Barnaghi, P., Wang, W., and Moessner, K. (2012). A hybrid semantic matchmaker for iot services. In *Proceedings of the 2012 IEEE International Conference on Green Computing and Communications*, pages 210–216. IEEE.

[52] Castells, M. (2011). *The rise of the network society: The information age: Economy, society, and culture*. John Wiley & Sons. Inc.

[53] Cavazza, M. and Palmer, I. (2000). High-level interpretation in virtual environments. *Applied Artificial Intelligence*, 14(1):125–144.

[54] Chabeb, Y., Tata, S., and Ozanne, A. (2010). Yasa-m: A semantic web service matchmaker. In *Proceedings of the 24th IEEE International Conference on Advanced Information Networking and Applications*, pages 966–973. IEEE.

[55] Chang, F., Dean, J., Ghemawat, S., Hsieh, W. C., Wallach, D. A., Burrows, M., Chandra, T., Fikes, A., and Gruber, R. E. (2008). Bigtable: A distributed storage system for structured data. *ACM Transactions on Computer Systems*, 26(2):4:1–4:26.

[56] Chaudhuri, S., Kalogerakis, E., Guibas, L., and Koltun, V. (2011). Probabilistic reasoning for assembly-based 3d modeling. *ACM Transactions on Graphics*, 30(4):35:1–35:10.

[57] Chawathe, Y., Ratnasamy, S., Breslau, L., Lanham, N., and Shenker, S. (2003). Making gnutella-like p2p systems scalable. In *Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, pages 407–418. ACM.

[58] Chen, N. and Clarke, S. (2014). A dynamic service composition model for adaptive systems in mobile computing environments. In *Service-Oriented Computing*, pages 93–107. Springer.

[59] Cheng, A.-H. and Joung, Y.-J. (2006). Probabilistic file indexing and searching in unstructured peer-to-peer networks. *Computer Networks*, 50(1):106–127.

[60] Cheng, J., Liu, C., Zhou, M., Zeng, Q., and Yla-Jaaski, A. (2014). Automatic composition of semantic web services based on fuzzy predicate petri nets. *IEEE Transactions on Automation Science and Engineering*, 99(1):1–10.

[61] Chevaillier, P., Trinh, T.-H., Barange, M., De Loor, P., Devillers, F., Soler, J., and Querrec, R. (2012). Semantic modeling of virtual environments using mascaret. In *Proceedings of the 5th Workshop on Software Engineering and Architectures for Realtime Interactive Systems*, pages 1–8. IEEE.

[62] Chmielewski, J. (2012). Describing interactivity of 3d content. In *Interactive 3D Multimedia Content*, pages 195–221. Springer.

[63] Chmielewski, J. (2014). Finding interactive 3d objects by their interaction properties. *Multimedia Tools and Applications*, 69(3):773–798.

[64] Chu, Y.-L. and Li, T.-Y. (2008). Using pluggable procedures and ontology to realize semantic virtual environments 2.0. In *Proceedings of The 7th ACM SIGGRAPH International Conference on Virtual-Reality Continuum and Its Applications in Industry*, VRCAI '08, pages 27:1–27:6, New York, NY, USA. ACM.

[65] Clarke, I., Miller, S. G., Hong, T. W., Sandberg, O., and Wiley, B. (2002). Protecting free expression online with freenet. *IEEE Journal on Internet Computing*, 6(1):40–49.

[66] Classe, T., Braga, R., Campos, F., and David, J. M. N. (2013). A semantic peer-to-peer network for service composition in scientific domains. *Procedia Technology*, 9(1):215–225.

[67] Cohen, B. (2003). Incentives build robustness in bittorrent. In *Workshop on Economics of Peer-to-Peer Systems*, volume 6, pages 68–72.

[68] Cohen, B. (2008). The bittorrent protocol specification.

[69] Cohen, E. and Shenker, S. (2002). Replication strategies in unstructured peer-to-peer networks. In *Proceedings of the 2002 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, SIGCOMM '02, pages 177–190, New York, NY, USA. ACM.

[70] Crespo, A. and Garcia-Molina, H. (2002). Routing indices for peer-to-peer systems. In *Proceedings of the 22nd International Conference on Distributed Computing Systems*, pages 23–32. IEEE.

[71] Daras, P. and Alvarez, F. (2009). A future perspective on the 3d media internet. *Towards the Future Internet*, pages 303–312.

[72] De Luca, L., Véron, P., and Florenzano, M. (2007). A generic formalism for the semantic modeling and representation of architectural elements. *The Visual Computer*, 23(3):181–205.

[73] De Lucia, M., Tzovara, A., Bernasconi, F., Spierer, L., and Murray, M. M. (2012). Auditory perceptual decision-making based on semantic categorization of environmental sounds. *NeuroImage*, 60(3):1704–1715.

[74] Deconinck, G. and Vanthournout, K. (2009). Agora: A semantic overlay network. *International Journal of Critical Infrastructures*, 5(1):175–195.

[75] del Val, E., Rebollo, M., and Botti, V. (2012). Decentralized semantic service discovery in preferential attachment networks. In *Agent-based Technologies and applications for enterprise interOPerability*, pages 130–150. Springer.

[76] Della Valle, E., Cerizza, D., and Celino, I. (2005). The mediators centric approach to automatic web service discovery of glue. *MEDIATE2005*, 168:35–50.

[77] Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B*, 39(1):1–38.

[78] Deng, S., Huang, L., Tan, W., and Wu, Z. (2014). Top-k automatic service composition: A parallel method for large-scale service sets. *IEEE Transaction on Automation Science and Engineering*, 11(3):891–905.

[79] Despotovic, Z. and Aberer, K. (2006). P2p reputation management: Probabilistic estimation vs. social networks. *Computer Networks*, 50(4):485–500.

[80] Di Modica, G., Tomarchio, O., and Vita, L. (2011). Resource and service discovery in soas: a p2p oriented semantic approach. *International Journal of Applied Mathematics and Computer Science*, 21(2):285–294.

[81] Dorrigiv, R., Lopez-Ortiz, A., and Pralat, P. (2007). Search algorithms for unstructured peer-to-peer networks. In *Proceedings of the 32nd IEEE Conference on Local Computer Networks*, pages 343–352. IEEE.

[82] Doulamis, N. D., Karamolegkos, P. N., Doulamis, A. D., and Nikolakopoulos, I. G. (2007). Optimal decomposition of p2p networks based on file exchange patterns for multimedia content search & replication. In *Proceedings of the 9th ACM SIGMM International Workshop on Multimedia Information Retrieval*, pages 125–134. ACM.

[83] Doulkeridis, C., Norvag, K., and Vazirgiannis, M. (2007). Desent: Decentralized and distributed semantic overlay generation in p2p networks. *IEEE Journal on Selected Areas in Communications*, 25(1):25–34.

[84] Draidi, F., Pacitti, E., and Kemme, B. (2011). P2prec: a p2p recommendation system for large-scale data sharing. In *Transactions on Large-Scale Data-and Knowledge-Centered Systems III*, volume 6790, pages 87–116. Springer.

[85] Durr, M., Maier, M., and Wiesner, K. (2012). An analysis of query forwarding strategies for secure and privacy-preserving social networks. In *Proceedings of the 2012 International Conference on Advances in Social Networks Analysis and Mining*, pages 535–542. IEEE.

[86] Eittenberger, P. M., Herbst, M., and Krieger, U. R. (2012). Rapidstream: P2p streaming on android. In *Proceedings of the 19th International Packet Video Workshop*, pages 125–130. IEEE.

[87] El-Ansary, S. and Haridi, S. (2005). An overview of structured p2p overlay networks. *Handbook on theoretical and algorithmic aspects of sensor, ad hoc wireless, and peer-to-peer networks. Auerbach Publications*, pages 665–684.

[88] El Hadad, J., Manouvrier, M., and Rukoz, M. (2010). Tqos: Transactional and qos-aware selection algorithm for automatic web service composition. *IEEE Transactions on Services Computing*, 3(1):73–85.

[89] Endo, K., Imaoka, A., Okano, D., and Amano, K. (2014). A search method using temporary links for unstructured p2p networks. *Journal of Networks*, 9(7):1665–1673.

[90] Erdős, P. and Rényi, A. (1976). On the evolution of random graphs. *Selected Papers of Alfréd Rényi, vol*, 2:482–525.

[91] Erol, K., Hendler, J. A., and Nau, D. S. (1994). Umcp: A sound and complete procedure for hierarchical task-network planning. In *Proceedings of the 1994 Conference on Artificial Intelligence Planning Systems*, pages 249–254. AAAI.

[92] Fagin, R. (2002). Combining fuzzy information: an overview. *ACM SIGMOD Record*, 31(2):109–118.

[93] Fensel, D., Facca, F. M., Simperl, E., and Toma, I. (2011). *Semantic web services*. Springer Science & Business Media.

[94] Fikes, R. E. and Nilsson, N. J. (1972). Strips: A new approach to the application of theorem proving to problem solving. *Artificial intelligence*, 2(3):189–208.

[95] Filali, I. and Huet, F. (2010). Dynamic ttl-based search in unstructured peer-to-peer networks. In *Proceedings of the 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, pages 438–447. IEEE.

[96] Finin, T., Fritzson, R., McKay, D., and McEntire, R. (1994). Kqml—a language and protocol for knowledge and information exchange. In *Proceedings of the 13th International Distributed Artificial Intelligence Workshop*, pages 127–136.

[97] Fisher, D. H. (1987). Knowledge acquisition via incremental conceptual clustering. *Machine learning*, 2(2):139–172.

[98] Fletcher, G. H., Sheth, H. A., and Börner, K. (2005). Unstructured peer-to-peer networks: Topological properties and search performance. In *Agents and Peer-to-Peer Computing*, pages 14–27. Springer.

[99] Floyd, R. W. (1967). Assigning meanings to programs. *Program Verification: Fundamental Issues in Computer Science. Part I*, pages 65–81.

[100] Ford, D., Labelle, F., Popovici, F. I., Stokely, M., Truong, V.-A., Barroso, L., Grimes, C., and Quinlan, S. (2010). Availability in globally distributed storage systems. In *OSDI*, pages 61–74.

[101] Forestiero, A., Mastroianni, C., Papadakis, H., Fragopoulou, P., Troisi, A., and Zimeo, E. (2008a). A scalable architecture for discovery and planning in p2p service networks. In *Grid Computing*, pages 97–108. Springer.

[102] Forestiero, A., Mastroianni, C., Papuzzo, G., and Spezzano, G. (2010). A proximity-based self-organizing framework for service composition and discovery. In *Proceedings of the 10th IEEE/ACM International Conference onCluster, Cloud and Grid Computing*, pages 428–437. IEEE.

[103] Forestiero, A., Mastroianni, C., and Spezzano, G. (2008b). Qos-based dissemination of content in grids. *Future Generation Computer Systems*, 24(3):235–244.

[104] Fox, M. and Long, D. (2003). Pddl2. 1: An extension to pddl for expressing temporal planning domains. *Journal of Artificial Intelligence Research*, 20(1):61–124.

[105] Fredman, M. L. and Tarjan, R. E. (1987). Fibonacci heaps and their uses in improved network optimization algorithms. *Journal of the ACM*, 34(3):596–615.

[106] Fujii, K. and Suda, T. (2009). Semantics-based context-aware dynamic service composition. *ACM Transactions on Autonomous and Adaptive Systems*, 4(2):12:1–12:31.

[107] Funkhouser, T., Kazhdan, M., Shilane, P., Min, P., Kiefer, W., Tal, A., Rusinkiewicz, S., and Dobkin, D. (2004). Modeling by example. volume 23, pages 652–663, New York, NY, USA. ACM.

[108] Furno, A. and Zimeo, E. (2013). Efficient cooperative discovery of service compositions in unstructured p2p networks. In *Proceedings of the 21st Euromicro International Conference on Parallel, Distributed and Network-Based Processing*, pages 58–67. IEEE.

[109] Furno, A. and Zimeo, E. (2014a). Gossip strategies for service composition. In *Proceedings of the 22nd Euromicro International Conference on Parallel, Distributed and Network-Based Processing*, pages 27–35. IEEE.

[110] Furno, A. and Zimeo, E. (2014b). Self-scaling cooperative discovery of service compositions in unstructured p2p networks. *Journal of Parallel and Distributed Computing*, 74(10):2994–3025.

[111] Galatopoullos, D. G., Kalofonos, D. N., and Manolakos, E. S. (2008). A p2p soa enabling group collaboration through service composition. In *Proceedings of the 5th International Conference on Pervasive Services*, ICPS '08, pages 111–120, New York, NY, USA. ACM.

[112] Gao, B., Zhang, S., and Pan, X. (2011). Semantic-oriented 3d model classification and retrieval using gaussian processes. *Journal of Computational Information Systems*, 7(4):1029–1037.

[113] Gao, G., Li, R., Wen, K., and Gu, X. (2012). Proactive replication for rare objects in unstructured peer-to-peer networks. *Journal of Network and Computer Applications*, 35(1):85–96.

[114] Garbacki, P., Epema, D. H., and Van Steen, M. (2010). The design and evaluation of a self-organizing superpeer network. *IEEE Transactions on Computers*, 59(3):317–331.

[115] Garcia-Rojas, A., Vexo, F., Thalmann, D., Raouzaiou, A., Karpouzis, K., and Kollias, S. (2006). Emotional body expression parameters in virtual human ontology. In *Proceedings of the 1st International Workshop on Shapes and Semantics*, pages 63–70, Matsushima, Japan.

[116] Gharzouli, M. and Boufaida, M. (2009). A generic p2p collaborative strategy for discovering and composing semantic web services. In *Proceedings of the 4th International Conference on Internet and Web Applications and Services*, pages 449–454. IEEE.

[117] Gharzouli, M. and Boufaida, M. (2011). Pm4sws: A p2p model for semantic web services discovery and composition. *Journal of Advances in Information Technology*, 2(1):15–26.

[118] Ghemawat, S., Gobioff, H., and Leung, S.-T. (2003). The google file system. In *ACM SIGOPS Operating Systems Review*, volume 37, pages 29–43. ACM.

[119] Gholam Hassan Tabatabaei, S., Vahid Dastjerdi, A., Wan Kadir, W. M., Ibrahim, S., and Sarafian, E. (2010). Security conscious ai-planning-based composition of semantic web services. *International Journal of Web Information Systems*, 6(3):203–229.

[120] Ghorbani, M., Meybodi, M. R., and Saghiri, A. M. (2013). A new version of k-random walks algorithm in peer-to-peer networks utilizing learning automata. In *Proceedings of the 5th Conference on Information and Knowledge Technology*, pages 1–6. IEEE.

[121] Gilbert, E. N. (1959). Random graphs. *The Annals of Mathematical Statistics*, pages 1141–1144.

[122] Giunchiglia, F., Kharkevich, U., Hume, A., and Chatvorawit, P. (2010). Semantic flooding: search over semantic links. In *Proceedings of the 26th IEEE International Conference on Data Engineering Workshops*, pages 191–196. IEEE.

[123] Gomez-Perez, A., Fernández-López, M., and Corcho-Garcia, O. (2004). Ontological engineering. *Computing Reviews*, 45(8):478–479.

[124] Gong, B., Liu, J., Wang, X., and Tang, X. (2011). 3d object retrieval with semantic attributes. In *Proceedings of the 19th ACM International Conference on Multimedia*, pages 757–758. ACM.

[125] Groth, D. and Skandier, T. (2006). *Network+ Study Guide*. John Wiley & Sons.

[126] Gruber, T. R. (1993). A translation approach to portable ontology specifications. *Knowledge acquisition*, 5(2):199–220.

[127] Gu, T., Zhang, D., and Pung, H. K. (2007). A two-tier semantic overlay network for p2p search. In *Proceedings of the 13th International Conference on Parallel and Distributed Systems*, pages 1–8. IEEE.

[128] Gu, X. and Nahrstedt, K. (2006). Distributed multimedia service composition with statistical qos assurances. *IEEE Transactions on Multimedia*, 8(1):141–151.

[129] Gutiérrez, M., García-Rojas, A., Thalmann, D., Vexo, F., Moccozet, L., Magnenat-Thalmann, N., Mortara, M., and Spagnuolo, M. (2007). An ontology of virtual humans. *The Visual Computer*, 23(3):207–218.

[130] Gutiérrez Alonso, M. A. (2005). Semantic virtual environments. phd thesis. école polytechnique fédérale de lausanne.

[131] Haak, S. and Grimm, S. (2011). Towards custom cloud services: Using semantic technology to optimize resource configuration. In *Proceedings of the 8th Extended Semantic Web Conference on The Semantic Web: Research and Applications - Volume Part II*, ESWC'11, pages 345–359, Berlin, Heidelberg. Springer-Verlag.

[132] Haase, P., Broekstra, J., Ehrig, M., Menken, M., Mika, P., Olko, M., Plechawski, M., Pyszlak, P., Schnizler, B., Siebes, R., et al. (2004). Bibster–a semantics-based bibliographic peer-to-peer system. In *Proceedings of the 3rd International Conference on Semantic Web*, pages 122–136. Springer.

[133] Hamilton, J. D. (1994). *Time Series Analysis*, volume 2. Princeton University Press.

[134] Hang, C.-W. and Singh, M. P. (2011). Trustworthy service selection and composition. *ACM Transactions on Autonomous and Adaptive Systems*, 6(1):5:1–5:17.

[135] Harth, A., Hose, K., Karnstedt, M., Polleres, A., Sattler, K.-U., and Umbrich, J. (2010). Data summaries for on-demand queries over linked data. In *Proceedings of the 19th International Conference on World Wide web*, pages 411–420. ACM.

[136] Hartig, O., Bizer, C., and Freytag, J.-C. (2009). Executing sparql queries over the web of linked data. In *Proceedings of the 8th International Semantic Web Conference*, pages 293–309. Springer.

[137] Hatzi, O., Vrakas, D., Nikolaidou, M., Bassiliades, N., Anagnostopoulos, D., and Vlahavas, L. (2012). An integrated approach to automated semantic web service composition through planning. *IEEE Transactions on Services Computing*, 5(3):319–332.

[138] He, Q., Yan, J., Yang, Y., Kowalczyk, R., and Jin, H. (2013). A decentralized service discovery approach on peer-to-peer networks. *IEEE Transactions on Services Computing*, 6(1):64–75.

[139] Heath, T. and Bizer, C. (2011). Linked data: Evolving the web into a global data space. *Synthesis Lectures on the Semantic Web: Theory and Technology*, 1(1):1–136.

[140] Hejrati, M. and Ramanan, D. (2014). Analysis by synthesis: 3d object recognition by object reconstruction. In *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2449–2456. IEEE.

[141] Hendler, J. (2009). Web 3.0 emerging. *Computer*, 42(1):111–113.

[142] Herget, W., Krauß, C., Nonnengart, A., Spieldenner, T., Warwas, S., and Zinnikus, I. (2013). A virtual environment for collaborative engineering with formal verification. In Bil, C., Mo, J., and Stjepandic, J., editors, *20th ISPE International Conference on Concurrent Engineering, Melbourne, Australia, September 2-6, 2013*, pages 225–234. IOS Press.

[143] Himali, D., Navathe, S. B., and Prasad, S. K. (2012). Osqr: A framework for ontology-based semantic query routing in unstructured p2p networks. In *Proceedings of the 19th International Conference on High Performance Computing*, pages 1–10. IEEE.

[144] Hoare, C. A. R. (1969). An axiomatic basis for computer programming. *Communications of the ACM*, 12(10):576–580.

[145] Hoffmann, J. (2001). Ff: The fast-forward planning system. *AI magazine*, 22(3):57–62.

[146] Hoffmann, J., Weber, I., Scicluna, J., Kaczmarek, T., and Ankolekar, A. (2008). Combining scalability and expressivity in the automatic composition of semantic web services. In *Proceedings of the 8th International Conference on Web Engineering*, pages 98–107. IEEE.

[147] Hois, J., Schill, K., and Bateman, J. (2007a). Integrating uncertain knowledge in a domain ontology for room concept classifications. *Research and Development in Intelligent Systems XXIII. Proceedings of the AI-2006, the 26th SGAI International Conference on Innovative Techniques and Applications of Artificial Intelligence*, pages 245–258.

[148] Hois, J., Wünstel, M., Bateman, J. A., and Röfer, T. (2007b). Dialog-based 3d-image recognition using a domain ontology. In *Proceedings of the 2006 International Conference on Spatial Cognition V: Reasoning, Action, Interaction*, SC'06, pages 107–126, Berlin, Heidelberg. Springer-Verlag.

[149] Hota, C., Nunia, V., Di Francesco, M., Nurminen, J. K., and Ylä-Jääski, A. (2013). Enhanced search in unstructured peer-to-peer overlay networks. In *Grid and Pervasive Computing*, volume 7861, pages 270–279. Springer.

[150] Hou, S., Lou, K., and Ramani, K. (2005). Svm-based semantic clustering and retrieval of a 3d model database. *Computer Aided Design and Application*, 2(2):155–164.

[151] Hsiao, H.-C. and Su, H.-W. (2012). On optimizing overlay topologies for search in unstructured peer-to-peer networks. *IEEE Transactions on Parallel and Distributed Systems*, 23(5):924–935.

[152] Hu, S., Muthusamy, V., Li, G., and Jacobsen, H. (2008). Distributed automatic service composition in large-scale systems. In *Proceedings of the 2nd International Conference on Distributed Event-Based Systems. Rome, Italy*, pages 233–244.

[153] Hu, Y., Feng, M., Bhuyan, L. N., and Kalogeraki, V. (2009). Budget-based self-optimized incentive search in unstructured p2p networks. In *IEEE INFOCOM 2009*, pages 352–360. IEEE.

[154] Huang, J.-C., Li, X.-Q., and Wu, J. (2011). A semantic searching scheme in heterogeneous unstructured p2p networks. *Journal of Computer Science and Technology*, 26(6):925–941.

[155] Huebsch, R., Hellerstein, J. M., Lanham, N., Loo, B. T., Shenker, S., and Stoica, I. (2003). Querying the internet with pier. In *Proceedings of the 29th international conference on Very large data bases-Volume 29*, pages 321–332. VLDB Endowment.

[156] Ip, C., Lapadat, D., Sieger, L., and Regli, W. (2002). Using shape distributions to compare solid models. In *Proceedings of the 7th ACM Symposium on Solid Modeling and Applications*, pages 273–280. ACM.

[157] Issarny, V., Georgantas, N., Hachem, S., Zarras, A., Vassiliadist, P., Autili, M., Gerosa, M. A., and Hamida, A. B. (2011). Service-oriented middleware for the future internet: state of the art and research directions. *Journal of Internet Services and Applications*, 2(1):23–45.

[158] Jaeger, M. C., Rojec-Goldmann, G., Liebetruth, C., Mühl, G., and Geihs, K. (2005). Ranked matching for service descriptions using owl-s. In *Kommunikation in Verteilten Systemen (KiVS)*, pages 91–102. Springer.

[159] Jain, A., Thormählen, T., Ritschel, T., and Seidel, H.-P. (2012). Exploring shape variations by 3d-model decomposition and part-based recombination. *Computer Graphics Forum*, 31(2pt3):631–640.

[160] Jarboe, G., Thomases, H., Smith, M., Treadaway, C., and Evans, D. (2002). *Networking complete*. Sybex.

[161] Jawahar, I. and Wu, J. (2004). A two level random search protocol for peer-to peer networks. In *Proceedings of the 8th World Multi-Conference on Systemics, Cybernetics and Informatics*, pages 1–5. IIIS.

[162] Jia, J., Meng, C., and Zhou, H. (2010). Hierarchical architecture for semantic peer-to-peer web service discovery. In *Proceedings of the 2010 International Conference on Web Information Systems and Mining*, pages 166–170. IEEE.

[163] Jia, L., Rajaraman, R., and Suel, T. (2002). An efficient distributed algorithm for constructing small dominating sets. *Distributed Computing*, 15(4):193–205.

[164] Jiang, W., Hu, S., Lee, D., Gong, S., and Liu, Z. (2012). Continuous query for qos-aware automatic service composition. In *Proceedings of the 19th IEEE International Conference on Web Services*, pages 50–57. IEEE.

[165] Jiang, Y., Tang, Y., and Chen, Q. (2011). An adjustable approach to intuitionistic fuzzy soft sets based decision making. *Applied Mathematical Modelling*, 35(2):824–836.

[166] Jimenez, R.; Bakker, A. K. B. P. J. and Haridi, S. (2013). Tribler mobile: P2p video streaming from and to mobile devices. *Distributed Peer Discovery in Large-Scale P2P Streaming Systems. PhD Thesis*, 11.

[167] Johnson, D. S. and McGeoch, L. A. (1997). The traveling salesman problem: A case study in local optimization. *Local Search in Combinatorial Optimization*, 1:215–310.

[168] Jones, J. H. (2007). Notes on r0. technical report. *Department of Anthropological Sciences Stanford University*.

[169] Joung, Y.-J. and Chuang, F.-Y. (2009). Ontozilla: An ontology-based, semi-structured, and evolutionary peer-to-peer network for information systems and services. *Future Generation Computer Systems*, 25(1):53–63.

[170] Junghans, M., Agarwal, S., and Studer, R. (2010). Towards practical semantic web service discovery. In *The Semantic Web: Research and Applications*, pages 15–29. Springer.

[171] Kadambi, S., 0001, J. C., Cooper, B. F., Lomax, D., Ramakrishnan, R., Silberstein, A., Tam, E., and Garcia-Molina, H. (2011). Where in the world is my data? *PVLDB*, 4(11):1040–1050.

[172] Kai-Hsiang, Y. and Chi-Jen, W. (2006). Antsearch: An ant search algorithm in unstructured peer-to-peer networks. *IEICE Transactions on Communications*, 89(9):2300–2308.

[173] Kalasapur, S., Kumar, M., and Shirazi, B. (2007). Dynamic service composition in pervasive computing. *IEEE Transactions on Parallel and Distributed Systems*, 18(7):907–918.

[174] Kalogeraki, V., Gunopulos, D., and Zeinalipour-Yazti, D. (2002). A local search mechanism for peer-to-peer networks. In *Proceedings of the 11th International Conference on Information and Knowledge Management*, pages 300–307. ACM.

[175] Kalogerakis, E., Chaudhuri, S., Koller, D., and Koltun, V. (2012). A probabilistic model for component-based shape synthesis. *ACM Transactions on Graphics*, 31(4):55:1–55:11.

[176] Kalogerakis, E., Moumoutzis, N., and Christodoulakis, S. (2006). Coupling ontologies with graphics content for knowledge driven visualization. In *Proceedings of the 2006 IEEE Conference on Virtual Reality*, pages 43–50. IEEE.

[177] Kangasharju, J., Ross, K. W., Antipolls, S., and Turner, D. A. (2002). Optimal content replication in p2p communities. manuscript.

[178] Kantere, V., Skiadopoulos, S., and Sellis, T. (2009a). Storing and indexing spatial data in p2p systems. *IEEE Transactions on Knowledge and Data Engineering*, 21(2):287–300.

[179] Kantere, V., Tsoumakos, D., Sellis, T., and Roussopoulos, N. (2009b). Groupeer: Dynamic clustering of p2p databases. *Information Systems*, 34(1):62–86.

[180] Kaoudi, Z., Koubarakis, M., Kyzirakos, K., Miliaraki, I., Magiridou, M., and Papadakis-Pesaresi, A. (2010). Atlas: Storing, updating and querying rdf (s) data on top of dhts. *Web Semantics: Science, Services and Agents on the World Wide Web*, 8(4):271–277.

[181] Kapahnke, P., Liedtke, P., Nesbigall, S., Warwas, S., and Klusch, M. (2010). Isreal: an open platform for semantic-based 3d simulations in the 3d internet. pages 161–176. Springer.

[182] Karnstedt, M., Sattler, K.-U., Hauswirth, M., and Schmidt, R. (2008). A dht-based infrastructure for ad-hoc integration and querying of semantic data. In *Proceedings of the 2008 international symposium on Database engineering & applications*, pages 19–28. ACM.

[183] Kaufer, F. and Klusch, M. (2006). Wsmo-mx: A logic programming based hybrid service matchmaker. In *Proceedings of the 4th European Conference on Web Services*, pages 161–170. IEEE.

[184] Keim, D. A. (1999). Efficient geometry-based similarity search of 3d spatial databases. *SIGMOD Rec.*, 28(2):419–430.

[185] Keller, U., Lara, R., Lausen, H., Polleres, A., and Fensel, D. (2005). Automatic location of services. In *The Semantic Web: Research and Applications*, pages 1–16. Springer.

[186] Kiefer, C. and Bernstein, A. (2008). The creation and evaluation of isparql strategies for matchmaking. In Bechhofer, S., Hauswirth, M., Hoffmann, J., and Koubarakis, M., editors, *Proceedings of the 5th European Semantic Web Conference*, volume 5021 of *Lecture Notes in Computer Science*, pages 463–477. Springer.

[187] Kiefer, C., Bernstein, A., and Stocker, M. (2007). The fundamentals of isparql - a virtual triple approach for similarity-based semantic web tasks. In Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L. J. B., Golbeck, J., Mika, P., Maynard, D., Schreiber, G., and Cudré-Mauroux, P., editors, *Proceedings of the 6th International Semantic Web Conference and 2nd Asian Semantic Web Conference (ISWC/ASWC2007), Busan, South Korea*, volume 4825 of *LNCS*, pages 295–308, Berlin, Heidelberg. Springer Verlag.

[188] Kim, J., Han, S. W., Yi, D.-H., Kim, N., and Kuo, C.-C. J. (2010). Media-oriented service composition with service overlay networks: Challenges, approaches and future trends. *Journal of Communications*, 5(5):374–389.

[189] Kim, S., Strassner, J., and Hong, J.-K. (2011). Semantic overlay network for peer-to-peer hybrid information search and retrieval. In *Proceedings of the 12th International Symposium on Integrated Network Management*, pages 430–437. IEEE.

[190] Klein, F., Sons, K., John, S., Rubinstein, D., Slusallek, P., and Byelozyorov, S. (2012). Xflow: declarative data processing for the web. In *Proceedings of the 17th International Conference on 3D Web Technology*, pages 37–45. ACM.

[191] Klusch, M. (2008a). *On Agent-Based Semantic Service Coordination. Habilitation Thesis.* Universität des Saarlandes.

[192] Klusch, M. (2008b). Semantic web service coordination. In *CASCOM: Intelligent Service Coordination in the Semantic Web*, pages 59–104. Springer.

[193] Klusch, M. (2012). Overview of the s3 contest: Performance evaluation of semantic service matchmakers. In *Semantic Web Services*, pages 17–34. Springer.

[194] Klusch, M. (2014). Service discovery. *Encyclopedia of Social Networks and Mining (ESNAM), R. Alhajj and J. Rokne, Eds. Springer*.

[195] Klusch, M., Cao, X., Kapahnke, P., and Warwas, S. (2011). Semantics and agents for intelligent simulation and collaboration in the 3d internet. In *Proceedings of the 7th International Conference on Semantics Knowledge and Grid*, pages 9–16. IEEE.

[196] Klusch, M., Fries, B., and Sycara, K. (2006). Automated semantic web service discovery with owls-mx. In *Proceedings of the 5th International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 915–922. ACM.

[197] Klusch, M. and Gerber, A. (2006). Fast composition planning of owl-s services and application. In *Proceedings of the 4th IEEE European Conference on Web Services*, pages 181–190. IEEE.

[198] Klusch, M., Gerber, A., and Schmidt, M. (2005). Semantic web service composition planning with owls-xplan. In *AAAI Fall Symposium on Semantic Web and Agents*, pages 55–62. AAAI Press.

[199] Klusch, M. and Kapahnke, P. (2012a). Adaptive signature-based semantic selection of services with owls-mx3. *Multiagent Grid Syst.*, 8(1):69–82.

[200] Klusch, M. and Kapahnke, P. (2012b). The isem matchmaker: A flexible approach for adaptive hybrid semantic service selection. *Web Semantics: Science, Services and Agents on the World Wide Web*, 15:1–14.

[201] Klusch, M., Kapahnke, P., Cao, X., Rainer, B., Timmerer, C., and Mangold, S. (2014). My-media: Mobile semantic peer-to-peer video search and live streaming. In *Proceedings of the 11th ACM International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, pages 277–286. ACM.

[202] Klusch, M. and Zhing, X. (2008). Deployed semantic services for the common user of the web: A reality check. In *Proceedings of the 2nd IEEE International Conference on Semantic Computing*, volume 8, pages 347–353. IEEE.

[203] Kolda, T. G. and Bader, B. (2006). The tophits model for higher-order web link analysis. In *Proceedings of the 4th Workshop on Link Analysis, Counterterrorism and Security*, volume 7, pages 26–29.

[204] Kona, S., Bansal, A., and Gupta, G. (2007). Automatic composition of semanticweb services. In *Proceedings of the 5th IEEE International Conference on Web Services*, pages 150–158. IEEE.

[205] Koutsoudis, A., Stavroglou, K., Pavlidis, G., and Chamzas, C. (2011). 3dsse–a 3d scene search engine: Exploring 3d scenes using keywords. *Journal of Cultural Heritage*, 13(2):187–194.

[206] Kraevoy, V., Julius, D., and Sheffer, A. (2007). Shuffler: Modeling with interchangeable parts. In *The Visual Computer*. Springer.

[207] Küngas, P. and Matskin, M. (2006). Semantic web service composition through a p2p-based multi-agent environment. In *Agents and Peer-to-Peer Computing*, pages 106–119. Springer.

[208] Kuo, J.-L., Shih, C.-H., and Chen, Y.-C. (2013). An efficient query scheme for semantic web on mobile p2p network. In *Proceedings of the 5th International Conference on Evolving Internet*, pages 4–9.

[209] Laborie, S., Manzat, A., and Sedes, F. (2009). Managing and querying efficiently distributed semantic multimedia metadata collections. *IEEE Multimedia*, 99(1):1–9.

[210] Laclavik, M., Šeleng, M., Ciglan, M., and Hluchỳ, L. (2012). Ontea: Platform for pattern based automated semantic annotation. *Computing and Informatics*, 28(4):555–579.

[211] Larkin, E. (2007). Storm worm's virulence may change tactics. *Network World*, 2.

[212] Lécué, F. and Léger, A. (2006). Semantic web service composition through a matchmaking of domain. In *Proceedings of the 4th IEEE European Conference on Web Services*, pages 171–180. IEEE.

[213] Lecue, F. and Mehandjiev, N. (2009). Towards scalability of quality driven semantic web service composition. In *Proceedings of the 16th IEEE International Conference on Web Services*, pages 469–476. IEEE.

[214] Leifman, G., Meir, R., and Tal, A. (2005). Semantic-oriented 3d shape retrieval using relevance feedback. *The Visual Computer*, 21(8):865–875.

[215] Leng, C., Terpstra, W. W., Kemme, B., Stannat, W., and Buchmann, A. P. (2008a). Maintaining replicas in unstructured p2p systems. In *Proceedings of the 2008 ACM CoNEXT Conference*, page 19. ACM.

[216] Leng, C., Terpstra, W. W., Kemme, B., Stannat, W., and Buchmann, A. P. (2008b). Maintaining replicas in unstructured p2p systems. In *Proceedings of the 2008 ACM CoNEXT Conference*, pages 19:1–19:12. ACM.

[217] Leontiadis, E., Dimakopoulos, V. V., and Pitoura, E. (2006). Creating and maintaining replicas in unstructured peer-to-peer systems. In *Proceedings of the 12th International Euro-Par Conference on Parallel Processing*, pages 1015–1025. Springer.

[218] Leu, J.-S., Lin, W.-H., and Chen, J.-H. (2014). P2p resource searching with cloning random walker assisted by weakly connected dominating set. *Journal of Supercomputing*, 68(1):443–458.

[219] Leu, J.-S., Tsai, C.-W., and Lin, W.-H. (2011). Resource searching in an unstructured p2p network based on cloning random walker assisted by dominating set. *Computer Networks*, 55(3):722–733.

[220] Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions and reversals. In *Soviet Physics Doklady*, volume 10, page 707.

[221] Li, J. (2013). A fast semantic web services matchmaker for owl-s services. *Journal of Networks*, 8(5):1104–1111.

[222] Li, J., Wang, H., and Khan, S. U. (2012). A semantics-based approach to large-scale mobile social networking. *Mobile Networks and Applications*, 17(2):192–205.

[223] Li, Y., Bandar, Z. A., and McLean, D. (2003). An approach for measuring semantic similarity between words using multiple information sources. *IEEE Transactions on Knowledge and Data Engineering*, 15(4):871–882.

[224] Liang, L. C. Y. and Ming, L. T. (2011). Small world bee: Reduce messages flooding and improve recall rate for unstructured p2p system. *International Journal of Computer Science and Network Security*, 11(5):88–99.

[225] Liarou, E., Idreos, S., and Koubarakis, M. (2007). Publish/subscribe with rdf data over large structured overlay networks. In *Databases, Information Systems, and Peer-to-Peer Computing*, pages 135–146. Springer.

[226] Lin, D. (1998). An information-theoretic definition of similarity. In *Proceedings of the 15th International Conference on Machine Learning*, pages 296–304. Morgan Kaufmann Publishers Inc.

[227] Lin, W., Dou, W., Xu, Z., and Chen, J. (2013). A qos-aware service discovery method for elastic cloud computing in an unstructured peer-to-peer network. *Concurrency and Computation: Practice and Experience*, 25(13):1843–1860.

[228] Liu, J. and Zhuge, H. (2005). A semantic-link-based infrastructure for web service discovery in p2p networks. In *Special Interest Tracks and Posters of the 14th International Conference on World Wide Web*, pages 940–941. ACM.

[229] Liu, S., Küngas, P., and Matskin, M. (2006). Agent-based web service composition with jade and jxta. In *Proceedings of the 2nd International Conference on Semantic Web and Web Services*, pages 110–116.

[230] Liu, X., Huang, G., and Mei, H. (2009). Discovering homogeneous web service community in the user-centric web environment. *IEEE Transactions on Services Computing*, 2(2):167–181.

[231] Liu, Y., Wu, S., Xiong, N., Park, J. H., and Zhang, M. (2012). A cache-based search algorithm in unstructured p2p networks. *Journal of Intelligent Manufacturing*, 23(6):2101–2107.

[232] Loser, A., Staab, S., and Tempich, C. (2007). Semantic social overlay networks. *IEEE Journal on Selected Areas in Communications*, 25(1):5–14.

[233] Lu, J. and Callan, J. (2003). Content-based retrieval in hybrid peer-to-peer networks. In *Proceedings of the 12th International Conference on Information and Knowledge Management*, pages 199–206. ACM.

[234] Lu, Q. and Wang, Z. (2012). A semantic based p2p personal knowledge sharing model. *International Journal of Advancements in Computing Technology*, 4(1):33.

[235] Luck, M. and Aylett, R. (2000). Applying artificial intelligence to virtual reality: Intelligent virtual environments. *Applied Artificial Intelligence*, 14(1):3–32.

[236] Lueg, C. and Fisher, D. (2003). *From Usenet to CoWebs: Interacting with social information spaces*. Springer.

[237] Lv, Q., Cao, P., Cohen, E., Li, K., and Shenker, S. (2002). Search and replication in unstructured peer-to-peer networks. In *Proceedings of the 16th international conference on supercomputing*, pages 84–95. ACM.

[238] Mabrouk, N. B., Beauche, S., Kuznetsova, E., Georgantas, N., and Issarny, V. (2009). Qos-aware service composition in dynamic service oriented environments. In *Middleware*, pages 123–142. Springer.

[239] Madhavan, J., Bernstein, P. A., and Rahm, E. (2001). Generic schema matching with cupid. In *Proceedings of the 27th International Conference on Very Large Data Bases*, pages 49–58. Morgan Kaufmann.

[240] Maloberti, J. and Sebag, M. (2004). Fast theta-subsumption with constraint satisfaction algorithms. *Machine Learning*, 55(2):137–174.

[241] Mandreoli, F., Perdichizzi, A. M., and Penzo, W. (2007). A p2p-based architecture for semanticweb service automatic composition. In *Proceedings of the 18th International Workshop on Database and Expert Systems Applications*, pages 429–433. IEEE.

[242] Margariti, S. V. and Dimakopoulos, V. V. (2011). A novel probabilistic flooding strategy for unstructured peer-to-peer networks. In *Proceedings of the 15th Panhellenic Conference on Informatics*, pages 149–153. IEEE.

[243] Margariti, S. V. and Dimakopoulos, V. V. (2014). On probabilistic flooding search over unstructured peer-to-peer networks. *Peer-to-Peer Networking and Applications*, pages 1–12.

[244] Marti, S., Ganesan, P., and Garcia-Molina, H. (2005). Sprout: P2p routing with social networks. In *Current Trends in Database Technology-EDBT Workshops*, pages 425–435. Springer.

[245] Martins, V. (2007). Data replication in p2p systems. phd. thesis. université de nantes.

[246] Mastroianni, C. and Papuzzo, G. (2014). A self-organizing p2p framework for collective service discovery. *Journal of Network and Computer Applications*, 39:214–222.

[247] Matono, A., Pahlevi, S. M., and Kojima, I. (2007). Rdfcube: A p2p-based three-dimensional index for structural joins on distributed triple stores. In *Databases, Information Systems, and Peer-to-Peer Computing*, pages 323–330. Springer.

[248] Mavrikios, D.; Alexopoulos, K. X. V. P. M. S. K. C. G. (2011). A web-based platform for collaborative product design, review and evaluation. In *Digital Factory for Human-oriented Production Systems*, pages 35–56. Springer.

[249] Mawlood-Yunis, A.-R., Weiss, M., and Santoro, N. (2010). From p2p to reliable semantic p2p systems. *Peer-to-peer networking and applications*, 3(4):363–381.

[250] Mayer, P., Schroeder, A., and Koch, N. (2008). A model-driven approach to service orchestration. In *Proceedings of the 2008 IEEE International Conference on Services Computing - Volume 2*, pages 533–536. IEEE.

[251] Maymounkov, P. and Mazieres, D. (2002). Kademlia: A peer-to-peer information system based on the xor metric. In *Peer-to-Peer Systems*, pages 53–65. Springer.

[252] McConnell, R. M., Mehlhorn, K., Näher, S., and Schweitzer, P. (2011). Certifying algorithms. *Computer Science Review*, 5(2):119–161.

[253] McIlraith, S. A. and Son, T. C. (2002). Adapting golog for composition of semantic web services. In *Proceedings of the 8th International Conference on Principles of Knowledge Representation and Reasoning*, Toulouse, France.

[254] Meditskos, G. and Bassiliades, N. (2010). Structural and role-oriented web service discovery with taxonomies in owl-s. *IEEE Transactions on Knowledge and Data Engineering*, 22(2):278–290.

[255] Mei, Y., Wang, J., Wang, Q., Liu, X., Zhao, Z., and Zhang, Y. (2010). Semantic-based query routing in p2p social networks. In *Proceedings of the 2010 IEEE International Conference on Wireless Communications, Networking and Information Security*, pages 674–678. IEEE.

[256] Menck, N., Yang, X., Weidig, C., Winkes, P., Lauer, C., Hagen, H., Hamann, B., and Aurich, J. (2012). Collaborative factory planning in virtual reality. *Procedia CIRP*, 3:317–322.

[257] Mesnil, G., Bordes, A., Weston, J., Chechik, G., and Bengio, Y. (2014). Learning semantic representations of objects and their parts. *Machine Learning*, 94(2):281–301.

[258] Meulpolder, M., Epema, D. H., and Sips, H. J. (2008). Replication in bandwidth-symmetric bittorrent networks. In *Proceedings of the IEEE International Symposium on Parallel and Distributed Processing*, pages 1–8. IEEE.

[259] Michel Lombera, I., Moser, L., Melliar-Smith, P., and Chuang, Y.-T. (2013). Mobile ad-hoc search and retrieval in the itrust over wi-fi direct network. In *Proceedings of the 9th International Conference on Wireless and Mobile Communications*, pages 251–258.

[260] Michlmayr, A., Rosenberg, F., Leitner, P., and Dustdar, S. (2010). End-to-end support for qos-aware service selection, binding, and mediation in vresco. *IEEE Transactions on Services Computing*, 3(3):193–205.

[261] Moré, J. J. (1978). The levenberg-marquardt algorithm: implementation and theory. In *Numerical analysis*, pages 105–116. Springer.

[262] Murphy, G. L. (2002). *The big book of concepts*. MIT Press.

[263] Najar, S., Pinheiro, M. K., Souveyet, C., and Steffenel, L. A. (2012). Service discovery mechanism for an intentional pervasive information system. In *Proceedings of the 19th IEEE International Conference on Web Services*, pages 520–527. IEEE.

[264] Nam, W., Kil, H., and Lee, D. (2008). Type-aware web service composition using boolean satisfiability solver. In *Proceedings of the 10th IEEE Conference on E-Commerce Technology and the 5th IEEE Conference on Enterprise Computing, E-Commerce and E-Services*, pages 331–334. IEEE.

[265] Nebel, B. (1990). *Reasoning and revision in hybrid representation systems*. Springer.

[266] Nickel, M., Tresp, V., and Kriegel, H.-P. (2012). Factorizing yago: scalable machine learning for linked data. In *Proceedings of the 21st International Conference on World Wide Web*, pages 271–280. ACM.

[267] Nils Masuch, Benjamin Hirsch, M. B. A. H. and Albayrak, S. (2012). *Semantic Services: Advancement through Evaluation*, chapter SeMa^2 – a Hybrid Semantic Service Matching Approach, pages 35–48. Springer-Verlag, Berlin / Heidelberg.

[268] Noia, T. D., Sciascio, E. D., and Donini, F. M. (2009). A tableaux-based calculus for abduction in expressive description logics: Preliminary results. In Grau, B. C., Horrocks, I., Motik, B., and Sattler, U., editors, *Description Logics*, volume 477 of *CEUR Workshop Proceedings*. CEUR-WS.org.

[269] Oberlin, P., Rathinam, S., and Darbha, S. (2009). A transformation for a multiple depot, multiple traveling salesman problem. In *American Control Conference*, pages 2636–2641. IEEE.

[270] O'Connor, M. J. and Das, A. K. (2008). Sqwrl: A query language for owl. In Hoekstra, R. and Patel-Schneider, P. F., editors, *Proceedings of the 6st Workshop on OWL: Experiences and Directions*, volume 529 of *CEUR Workshop Proceedings*. CEUR-WS.org.

[271] Ohbuchi, R., Yamamoto, A., and Kobayashi, J. (2007). Learning semantic categories for 3d model retrieval. In *Proceedings of the 9th ACM SIGMM International Workshop on Multimedia Information Retrieval*, pages 31–40. ACM.

[272] Oram, A. (2001). *Peer-to-peer: harnessing the benefits of a disruptive technology*. O'Reilly Media, Inc.

[273] O'reilly, T. (2009). *What is web 2.0*. O'Reilly Media, Inc.

[274] Paganelli, F. and Parlanti, D. (2013). A dynamic composition and stubless invocation approach for information-providing services. *IEEE Transactions on Network and Service Management*, 10(2):218–230.

[275] Papadopoulos, S., Diplaris, S., Kompatsiaris, Y., Kunegis, J., Burnside, G., Aiello, L. M., Jaho, E., Giefer, A., Klusch, M., Mass, Y., Herzig, J., Rainer, B., Timmerer, C., Li, Z., Dancausa, C. M., Goker, A., and Clark, M. (2015). Socialsensor final evaluation report. socialsensor fp7-287975.

[276] Pappas, M., Karabatsou, V., Mavrikios, D., and Chryssolouris, G. (2006). Development of a web-based collaboration platform for manufacturing product and process design evaluation using virtual reality techniques. *International Journal of Computer Integrated Manufacturing*, 19(8):805–814.

[277] Paquet, E., Rioux, M., Murching, A., Naveen, T., and Tabatabai, A. (2000). Description of shape information for 2-d and 3-d objects. *Signal Processing: Image Communication*, 16(1):103–122.

[278] Parsons, S. (2001). *Qualitative methods for reasoning under uncertainty*, volume 13. Mit Press.

[279] Patel, J. A., Gupta, I., and Contractor, N. (2006). Jetstream: Achieving predictable gossip dissemination by leveraging social network principles. In *Proceedings of the 5th IEEE International Symposium on Network Computing and Applications*, pages 32–39. IEEE.

[280] Pedrinaci, C., Liu, D., Maleshkova, M., Lambert, D., Kopecky, J., and Domingue, J. (2010). iserve: a linked services publishing platform. In *CEUR Workshop Proceedings*, volume 596. CEUR-WS.org.

[281] Peer, J. (2005). A pop-based replanning agent for automatic web service composition. In Gómez-Pérez, A. and Euzenat, J., editors, *Proceedings of the 2nd European Semantic Web Conference*, volume 3532 of *Lecture Notes in Computer Science*, pages 47–61. Springer.

[282] Pellegrino, L., Huet, F., Baude, F., and Alshabani, A. (2013). A distributed publish/subscribe system for rdf data. In *Data Management in Cloud, Grid and P2P Systems*, pages 39–50. Springer.

[283] Pellens, B., Kleinermann, F., and De Troyer, O. (2009). A development environment using behavior patterns to facilitate building 3d/vr applications. In *Proceedings of the 6th Australasian Conference on Interactive Entertainment*, pages 1–8. ACM.

[284] Pereira, F., Botvinick, M., and Detre, G. (2013). Using wikipedia to learn semantic feature representations of concrete concepts in neuroimaging experiments. *Artificial intelligence*, 194:240–252.

[285] Pilioura, T. and Tsalgatidou, A. (2009). Unified publication and discovery of semantic web services. *ACM Transactions on the Web (TWEB)*, 3(3):1–44.

[286] Pirrò, G., Talia, D., and Trunfio, P. (2012). A dht-based semantic overlay network for service discovery. *Future Generation Computer Systems*, 28(4):689–707.

[287] Pistore, M., Traverso, P., Bertoli, P., and Marconi, A. (2005). Automated synthesis of composite bpel4ws web services. In *Proceedings of the 12th IEEE International Conference on Web Services*, pages 293–301. IEEE.

[288] Pittarello, F. and De Faveri, A. (2006). Semantic description of 3d environments: a proposal based on web standards. In *Proceedings of the eleventh international conference on 3D web technology*, pages 85–95. ACM.

[289] Plaxton, C. G., Rajaraman, R., and Richa, A. W. (1999). Accessing nearby copies of replicated objects in a distributed environment. *Theory of Computing Systems*, 32(3):241–280.

[290] Plebani, P. and Pernici, B. (2009). Urbe: Web service retrieval based on similarity evaluation. *IEEE Transactions on Knowledge and Data Engineering*, 21(11):1629–1642.

[291] Pu, I. M. and Shen, Y. (2009). Enhanced blocking expanding ring search in mobile ad hoc networks. In *Proceedings of the 3rd International Conference on New Technologies, Mobility and Security*, pages 1–5. IEEE.

[292] Puttaswamy, K. P., Sala, A., and Zhao, B. Y. (2008). Searching for rare objects using index replication. In *Proceedings of the 27th IEEE Conference on Computer Communications*. IEEE.

[293] Puttonen, J., Lobov, A., and Lastra, J. L. M. (2013). Semantics-based composition of factory automation processes encapsulated by web services. *IEEE Transaction on Industrial Informatics*, 9(4):2349–2359.

[294] Qi, G., Aggarwal, C., Tian, Q., Ji, H., and Huang, T. (2011). Exploring context and content links in social media: A latent space method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(5):850–862.

[295] QIN, P.-t. and LIU, R.-h. (2010). Search and combination of semantic web services based on chord. *Computer Knowledge and Technology*, 6(28):7936–7938.

[296] Rahmani, M. and Benchaïba, M. (2014). A comparative study of replication schemes for structured p2p networks. In *Proceedings of the 9th International Conference on Internet and Web Applications and Services*, pages 147–158. IARIA.

[297] Rajasekhar, S., Rong, B., Lai, K. Y., Khalil, I., and Tari, Z. (2006). Load sharing in peer-to-peer networks using dynamic replication. In *Proceedings of the 20th International Conference on Advanced Information Networking and Applications*, pages 1011–1016. IEEE.

[298] Ramaswamy, L. and Liu, L. (2003). Free riding: A new challenge to peer-to-peer file sharing systems. In *Proceedings of the 36th Annual Hawaii International Conference on System Sciences*, pages 10–19. IEEE.

[299] Rao, J., Küngas, P., and Matskin, M. (2006). Composition of semantic web services using linear logic theorem proving. *Information Systems*, 31(4):340–360.

[300] Rao, J. and Su, X. (2004). A survey of automated web service composition methods. In Cardoso, J. and Sheth, A. P., editors, *SWSWPC*, volume 3387 of *Lecture Notes in Computer Science*, pages 43–54. Springer.

[301] Rasch, K., Li, F., Sehic, S., Ayani, R., and Dustdar, S. (2011). Context-driven personalized service discovery in pervasive environments. *World Wide Web*, 14(4):295–319.

[302] Rasti, A. H., Stutzbach, D., and Rejaie, R. (2006). On the long-term evolution of the two-tier gnutella overlay. In *INFOCOM*, pages 1–6. IEEE.

[303] Ratnasamy, S., Francis, P., Handley, M., Karp, R., and Shenker, S. (2001). *A scalable content-addressable network*, volume 31. ACM.

[304] Razzaq, A., Latif, K., Ahmad, H. F., Hur, A., Anwar, Z., and Bloodsworth, P. C. (2014). Semantic security against web application attacks. *Information Sciences*, 254:19–38.

[305] Reffat, R. (2003). Semantic-based virtual design environments for architecture. In *Proceedings of the 21st Education and Research in Computer Aided Architectural Design in Europe*, pages 133–140. eCAADe.

[306] Resnik, P. (1995). Using information content to evaluate semantic similarity in a taxonomy. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 1*, IJCAI'95, pages 448–453, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

[307] Rhea, S., Geels, D., Roscoe, T., and Kubiatowicz, J. (2004). Handling churn in a dht. In *Proceedings of the USENIX Annual Technical Conference*, pages 127–140. Boston, MA, USA.

[308] Richardson, L. and Ruby, S. (2008). *RESTful web services*. O'Reilly Media, Inc.

[309] Risson, J. and Moors, T. (2006). Survey of research towards robust peer-to-peer networks: Search methods. *Computer networks*, 50(17):3485–3521.

[310] Rodriguez-Mier, P., Mucientes, M., and Lama, M. (2011). Automatic web service composition with a heuristic-based search algorithm. In *Proceedings of the 9th IEEE International Conference on Web Services*, pages 81–88. IEEE.

[311] Rodriguez-Mier, P., Pedrinaci, C., Lama, M., and Mucientes, M. (2015). An integrated semantic web service discovery and composition framework (to appear). *IEEE Transactions on Service Computing*.

[312] Rong, L. (2008). Multimedia resource replication strategy for a pervasive peer-to-peer environment. *Journal of Computers*, 3(4):9–15.

[313] Roverso, R., El-Ansary, S., and Haridi, S. (2012). Peer2view: A peer-to-peer http-live streaming platform. In *Proceedings of the 12th IEEE International Conference on Peer-to-Peer Computing*, pages 65–66. IEEE.

[314] Rowstron, A. and Druschel, P. (2001). Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. In *Middleware 2001*, pages 329–350. Springer.

[315] Sacco, M., Redaelli, C., Constantinescu, C., Lawson, G., D'Cruz, M., and Pappas, M. (2007). Difac: digital factory for human oriented production system. In *Human-Computer Interaction. HCI Applications and Services*, pages 1140–1149. Springer.

[316] Sánchez, D., Isern, D., and Millan, M. (2011). Content annotation for the semantic web: an automatic web-based approach. *Knowledge and Information Systems*, 27(3):393–418.

[317] Saroiu, S., Gummadi, K. P., and Gribble, S. D. (2003). Measuring and analyzing the characteristics of napster and gnutella hosts. *Multimedia systems*, 9(2):170–184.

[318] Saroiu, S., Gummadi, P. K., and Gribble, S. D. (2001). Measurement study of peer-to-peer file sharing systems. In *Electronic Imaging*, pages 156–170. International Society for Optics and Photonics.

[319] Sarshar, N., Boykin, P. O., and Roychowdhury, V. P. (2004). Percolation search in power law networks: Making unstructured peer-to-peer networks scalable. In *Proceedings of the 4th IEEE International Conference on Peer-to-Peer Computing*, pages 2–9. IEEE.

[320] Sbodio, M. L., Martin, D., and Moulin, C. (2010). Discovering semantic web services using sparql and intelligent agents. *Web Semantics: Science, Services and Agents on the World Wide Web*, 8(4):310–328.

[321] Schandl, B. and Zander, S. (2009). A framework for adaptive rdf graph replication for mobile semantic web applications. In *Joint Workshop on Advanced Technologies and Techniques for Enterprise Information Systems (Session on Managing Data with Mobile Devices), Milan, Italy*, pages 154–163. INSTICC Press.

[322] Schmidt-Schauß, M. and Smolka, G. (1991). Attributive concept descriptions with complements. *Artificial intelligence*, 48(1):1–26.

[323] Schollmeier, R. (2001). A definition of peer-to-peer networking for the classification of peer-to-peer architectures and applications. In *Proceedings of the 1st International Conference on Peer-to-Peer Computing*, pages 101–102. IEEE.

[324] Schwering, A. (2008). Approaches to semantic similarity measurement for geo-spatial data: A survey. *Transactions in GIS*, 12(1):5–29.

[325] Shen, H. (2010). An efficient and adaptive decentralized file replication algorithm in p2p file sharing systems. *IEEE Transactions on Parallel and Distributed Systems*, 21(6):827–840.

[326] Sheng, Q. Z., Qiao, X., Vasilakos, A. V., Szabo, C., Bourne, S., and Xu, X. (2014). Web services composition: A decade's overview. *Information Sciences*, 280:218–238.

[327] Sherchan, W., Nepal, S., and Paris, C. (2013). A survey of trust in social networks. *ACM Computing Surveys*, 45(4):47.

[328] Sioutas, S., Sakkopoulos, E., Makris, C., Vassiliadis, B., Tsakalidis, A., and Triantafillou, P. (2009). Dynamic web service discovery architecture based on a novel peer based overlay network. *Journal of Systems and Software*, 82(5):809–824.

[329] Sirin, E. and Parsia, B. (2007). Sparql-dl: Sparql query for owl-dl. In Golbreich, C., Kalyanpur, A., and Parsia, B., editors, *Proceedings of the 3rd Workshop on OWL: Experiences and Directions, Innsbruck, Austria*, volume 258 of *CEUR Workshop Proceedings*. CEUR-WS.org.

[330] Sirin, E., Parsia, B., Wu, D., Hendler, J., and Nau, D. (2004). Htn planning for web service composition using shop2. *Web Semantics: Science, Services and Agents on the World Wide Web*, 1(4):377–396.

[331] Sizov, S. (2007). What makes you think that? the semantic web's proof layer. *IEEE Transactions on Intelligent Transportation Systems*, 22(6):94–99.

[332] Skoutas, D., Sacharidis, D., Kantere, V., and Sellis, T. (2008). Efficient semantic web service discovery in centralized and p2p environments. In *Proceedings of the 7th International Conference on The Semantic Web*, pages 583–598. Springer.

[333] Slusallek, P. (2015). Collaborate3d - scalable intelligent support and reliability for collaboration in the future 3d internet. final project report. dfki saarbrücken, germany.

[334] Sobe, A., Elmenreich, W., and Boszormenyi, L. (2011). Replication for bio-inspired delivery in unstructured peer-to-peer networks. In *Proceedings of the 9th Workshop on Intelligent Solutions in Embedded Systems*, pages 109–116. IEEE.

[335] Sohrabi, S. and McIlraith, S. A. (2010). Preference-based web service composition: a middle ground between execution and search. In *Proceedings of the 9th International Semantic Web Conference*, pages 713–729. Springer-Verlag.

[336] Sons, K., Klein, F., Rubinstein, D., Byelozyorov, S., and Slusallek, P. (2010). Xml3d: interactive 3d graphics for the web. In *Proceedings of the 15th International Conference on Web 3D Technology*, pages 175–184. ACM.

[337] Sozio, M., Neumann, T., and Weikum, G. (2008). Near-optimal dynamic replication in unstructured peer-to-peer networks. In *Proceedings of the 27th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, pages 281–290. ACM.

[338] Spaho, E., Barolli, L., Xhafa, F., Biberaj, A., and Shurdi, O. (2013). P2p data replication and trustworthiness for a jxta-overlay p2p system using fuzzy logic. *Applied Soft Computing*, 13(1):321 – 328.

[339] Srirama, S. N. (2010). Publishing and discovery of mobile web services in peer to peer networks. *arXiv preprint arXiv:1007.2980.*

[340] Stadler, A. and Kolbe, T. H. (2007). Spatio-semantic coherence in the integration of 3d city models. In *Proceedings of the 5th International Symposium on Spatial Data Quality*. ISPRS.

[341] Staff, M.-W. E. (2004). *Merriam-Webster's collegiate dictionary*. Merriam-Webster Inc.

[342] Steinmetz, R. and Wehrle, K. (2005a). *Peer-to-Peer Systems and Applications*. Springer Science & Business Media.

[343] Steinmetz, R. and Wehrle, K. (2005b). *What Is This "Peer-to-Peer" About?* Springer.

[344] Stoica, I., Morris, R., Karger, D., Kaashoek, M. F., and Balakrishnan, H. (2001). Chord: A scalable peer-to-peer lookup service for internet applications. In *ACM SIGCOMM Computer Communication Review*, volume 31, pages 149–160. ACM.

[345] Stuckenschmidt, H. and Staab, S. (2006). *Semantic Web and Peer-to-peer: Decentralized Management and Exchange of Knowledge and Information*. Springer.

[346] Stutzbach, D., Rejaie, R., and Sen, S. (2008). Characterizing unstructured overlay topologies in modern p2p file-sharing systems. *IEEE/ACM Transactions on Networking*, 16(2):267–280.

[347] Su, J.-H., Chou, C.-L., Lin, C.-Y., and Tseng, V. S. (2011). Effective semantic annotation by image-to-concept distribution model. *IEEE Transactions on Multimedia*, 13(3):530–538.

[348] Tan, W., Fan, Y., Zhou, M., and Tian, Z. (2010). Data-driven service composition in enterprise soa solutions: a petri net approach. *IEEE Transactions on Automation Science and Engineering*, 7(3):686–694.

[349] Tan, W., Zhang, J., Madduri, R., Foster, I., and De Roure, D. (2011). Providing map and gps assistance to service composition in bioinformatics. In *IEEE International Conference on Services Computing*, pages 608–615. IEEE.

[350] Tangelder, J. and Veltkamp, R. (2004). A survey of content based 3d shape retrieval methods. In *Proceedings of the 2004 International Conference on Shape Modeling and Applications*, pages 145–156. IEEE.

[351] Tao, F., LaiLi, Y., Xu, L., and Zhang, L. (2013). Fc-paco-rm: a parallel method for service composition optimal-selection in cloud manufacturing system. *IEEE Transactions on Industrial Informatics*, 9(4):2023–2033.

[352] Tassier, T. (2013). Simple epidemics and sis models. In *The Economics of Epidemiology*, pages 9–16. Springer.

[353] Tempich, C. and Staab, S. (2006). Semantic query routing in unstructured networks using social metaphors. In *Semantic Web and Peer-to-Peer*, pages 107–123. Springer.

[354] Terpstra, W. W., Kangasharju, J., Leng, C., and Buchmann, A. P. (2007). Bubblestorm: resilient, probabilistic, and exhaustive peer-to-peer search. In *ACM SIGCOMM Computer Communication Review*, volume 37, pages 49–60. ACM.

[355] Thampi, S. M. et al. (2010). Survey of search and replication schemes in unstructured p2p networks. *arXiv preprint arXiv:1008.1629*.

[356] Tirado, J. M., Higuero, D., Isaila, F., Carretero, J., and Iamnitchi, A. (2010). Affinity p2p: A self-organizing content-based locality-aware collaborative peer-to-peer network. *Computer Networks*, 54(12):2056–2070.

[357] Toch, E., Gal, A., Reinhartz-Berger, I., and Dori, D. (2007). A semantic approach to approximate service retrieval. *ACM Transactions on Internet Technology (TOIT)*, 8(1):2:1–2:30.

[358] Tong, H., Cao, J., Zhang, S., and Li, M. (2011). A distributed algorithm for web service composition based on service agent model. *IEEE Transactions on Parallel and Distributed Systems*, 22(12):2008–2021.

[359] Tousch, A.-M., Herbin, S., and Audibert, J.-Y. (2012). Semantic hierarchies for image annotation: A survey. *Pattern Recognition*, 45(1):333–345.

[360] Tran, T., Zhang, L., and Studer, R. (2010). Summary models for routing keywords to linked data sources. In *Proceedings of the 9th International Semantic Web Conference*, pages 781–797. Springer.

[361] Tsoumakos, D. and Roussopoulos, N. (2003a). Adaptive probabilistic search for peer-to-peer networks. In *Proceedings of the 3rd IEEE International Conference on Peer-to-Peer Computing*, pages 102–109. IEEE.

[362] Tsoumakos, D. and Roussopoulos, N. (2003b). A comparison of peer-to-peer search methods. In *WebDB*, pages 61–66. Citeseer.

[363] Tsoumakos, D. and Roussopoulos, N. (2006). Analysis and comparison of p2p search methods. In *Proceedings of the 1st International Conference on Scalable Information Systems*, InfoScale '06, New York, NY, USA. ACM.

[364] Tutenel, T., Smelik, R. M., Lopes, R., de Kraker, K. J., and Bidarra, R. (2011). Generating consistent buildings: a semantic approach for integrating procedural techniques. *IEEE Transactions on Computational Intelligence and AI in Games*, 3(3):274–288.

[365] Ueda, K. and Yamashita, N. (2012). Global complexity bound analysis of the levenberg–marquardt method for nonsmooth equations and its application to the nonlinear complementarity problem. *Journal of Optimization Theory and Applications*, 152(2):450–467.

[366] Ulrich Lampe, S. S. (2012). *Semantic Services: Advancement through Evaluation*, chapter Self-Adaptive Semantic Matchmaking using COV4SWS.KOM and LOG4SWS.KOM, pages 141–157. Springer-Verlag, Berlin / Heidelberg.

[367] Valdez, G. C. (2010). A self-adaptive ant colony system for semantic query routing problem in p2p networks. *Computación y Sistemas*, 13(4):433–448.

[368] Van Tuyl, A. H. (1994). Acceleration of convergence of a family of logarithmically convergent sequences. *Mathematics of Computation*, 63(207):229–246.

[369] Vigfusson, Y., Birman, K., Huang, Q., and Nataraj, D. P. (2009). Go: Platform support for gossip applications. In *Proceedings of the IEEE 9th International Conference on Peer-to-Peer Computing*, pages 222–231. IEEE.

[370] Voulgaris, S., Gavidia, D., and Van Steen, M. (2005). Cyclon: Inexpensive membership management for unstructured p2p overlays. *Journal of Network and Systems Management*, 13(2):197–217.

[371] Vu, L.-H., Hauswirth, M., Porto, F., and Aberer, K. (2006). A search engine for qos-enabled discovery of semantic web services. *International Journal of Business Process Integration and Management*, 1(4):244–255.

[372] Walczak, K. (2012a). Building configurable 3d web applications with flex-vr. In *Interactive 3D Multimedia Content*, pages 103–136. Springer.

[373] Walczak, K. (2012b). Dynamic database modeling of 3d multimedia content. In *Interactive 3D Multimedia Content*, pages 55–102. Springer.

[374] Walczak, K. and Cellary, W. (2003). X-vrml for advanced virtual reality applications. *Computer*, 36(3):89–92.

[375] Wan, L., Wang, C., Xiong, T., Liua, Q., et al. (2013). A modelica-based modeling, simulation and knowledge sharing web platform. In *Proceedings of the 20th International Conference on Concurrent Engineering*, pages 517–525. IOS Press.

[376] Wang, C. and Xiao, L. (2007). An effective p2p search scheme to exploit file sharing heterogeneity. *IEEE Transactions on Parallel and Distributed Systems*, 18(2):145–157.

[377] Wang, J., He, K., Gong, P., Wang, C., Peng, R., and Li, B. (2008a). Rgps: A unified requirements meta-modeling frame for networked software. In *Proceedings of the 3rd International Workshop on Applications and Advances of Problem Frames*, pages 29–35. ACM.

[378] Wang, J., Zhang, J., Hung, P. C., Li, Z., Liu, J., and He, K. (2011). Leveraging fragmental semantic data to enhance services discovery. In *Proceedings of the 13th IEEE International Conference on High Performance Computing and Communications*, pages 687–694. IEEE.

[379] Wang, P., Tyra, J., Chan-Tin, E., Malchow, T., Kune, D. F., Hopper, N., and Kim, Y. (2008b). Attacking the kad network. In *Proceedings of the 4th International Conference on Security and Privacy in Communication Netowrks*, SecureComm '08, pages 23:1–23:10, New York, NY, USA. ACM.

[380] Wang, X. and Halang, W. A. (2012). *Discovery and Selection of Semantic Web Services*. Springer.

[381] Wang, Y., Avramopoulos, I., and Rexford, J. (2007). Morpheus: Making routing programmable. In *SIGCOMM workshop on Internet network management*, pages 285–286. ACM.

[382] Wang, Y. and Huang, J. S. (2014). Method of semantic web service discovery and automatic composition. *Journal of Emerging Technologies in Web Intelligence*, 6(3):298–304.

[383] Wasserman, S. and Faust, K. (1995). *Social Network Analysis. Methods and Applications*. Cambridge University Press.

[384] Waterhouse, S. (2001). Jxta search: Distributed search for distributed networks, technical report.

[385] Watts, D. J. and Strogatz, S. H. (1998). Collective dynamics of 'small-world'networks. *Nature*, 393(6684):440–442.

[386] Wei, D., Wang, T., Wang, J., and Bernstein, A. (2011). Sawsdl-imatcher: A customizable and effective semantic web service matchmaker. *Web Semantics: Science, Services and Agents on the World Wide Web*, 9(4):402–417.

[387] Weise, T., Bleul, S., Comes, D., and Geihs, K. (2008). Different approaches to semantic web service composition. In *Proceedings of the 3rd International Conference on Internet and Web Applications and Services*, pages 90–96. IEEE.

[388] Woods, W. A. and Schmolze, J. G. (1992). The kl-one family. *Computers & Mathematics with Applications*, 23(2):133–177.

[389] Wooldridge, J. (2012). *Introductory econometrics: A modern approach.* Cengage Learning.

[390] Wu, W. and Lui, J. (2012). Exploring the optimal replication strategy in p2p-vod systems: Characterization and evaluation. *IEEE Transactions on Parallel and Distributed Systems*, 23(8):1492–1503.

[391] Wu, Z., Gomadam, K., Ranabahu, A., Sheth, A. P., and Miller, J. A. (2007). Automatic composition of semantic web services using process mediation. In *Proceedings of the 9th International Conference on Enterprise Information Systems*, pages 453–462. INSTICC.

[392] Xhafa, F., Potlog, A.-D., Spaho, E., Pop, F., Cristea, V., and Barolli, L. (2012). Evaluation of intra-group optimistic data replication in p2p groupware systems. *Concurrency and Computation: Practice and Experience*, 27(4):870—-881.

[393] Xiao, H., Zou, Y., Tang, R., Ng, J., and Nigul, L. (2009). An automatic approach for ontology-driven service composition. In *Proceedings of the 2nd IEEE International Conference on Service-Oriented Computing and Applications*, pages 1–8. IEEE.

[394] Xu, M., Zhou, S., Guan, J., and Hu, X. (2010). A path-traceable query routing mechanism for search in unstructured peer-to-peer networks. *Journal of Network and Computer Applications*, 33(2):115–127.

[395] Yamada, T., Aihara, K., Takasu, A., and Adachi, J. (2005). Adaptive replication method based on peer behavior pattern in unstructured peer-to-peer systems. In *Proceedings of the 21th International Conference on Data Engineering Workshops*, pages 12–39. IEEE.

[396] Yamamoto, A., Tezuka, M., Shimizu, T., and Ohbuchi, R. (2008). Shrec¡¯08 entry: Semi-supervised learning for semantic 3d model retrieval. In *Proceedings of the 2008 IEEE International Conference on Shape Modeling and Applications*, pages 241–243. IEEE.

[397] Yamamoto, H. and Maruta, D. (2006). Replication methods for load balancing on distributed storages in p2p networks. *IEICE Transactions on Information and Systems*, 89(1):171–180.

[398] Yan, Y., Xu, B., Gu, Z., and Luo, S. (2009). A qos-driven approach for semantic service composition. In *Proceedings of the 2009 IEEE Conference on Commerce and Enterprise Computing*, pages 523–526. IEEE.

[399] Yang, B. (2010). Dsi: A model for distributed multimedia semantic indexing and content integration. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 6(1):3:1–3:21.

[400] Yang, B. and Garcia-Molina, H. (2001). Comparing hybrid peer-to-peer systems. In *Proceedings of the 27th International Conference on Very Large Data Bases*, pages 561–570.

[401] Yang, B. and Garcia-Molina, H. (2002). Improving search in peer-to-peer networks. In *Proceedings of the 22nd International Conference on Distributed Computing Systems*, pages 5–14. IEEE.

[402] Yang, D. and Guo, P. (2011). Image modeling with combined optimization techniques for image semantic annotation. *Neural Computing and Applications*, 20(7):1001–1015.

[403] Yang, K., Galis, A., and Chen, H.-H. (2010). Qos-aware service selection algorithms for pervasive service composition in mobile wireless environments. *Mobile Networks and Applications*, 15(4):488–501.

[404] Yang, X. and Hu, Y. (2009). Seif: Search enhanced by intelligent feedback in unstructured p2p networks. In *Proceedings of the 38th IEEE International Conference on Parallel Processing*, pages 494–501. IEEE.

[405] Yang, Y. (2012). Semi-structured semantic overlay for information retrieval in self-organizing networks. In *Proceedings of the 21st international conference companion on World Wide Web*, pages 203–208. ACM.

[406] Ye, M., Shou, D., Lee, W.-C., Yin, P., and Janowicz, K. (2011). On the semantic annotation of places in location-based social networks. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 520–528. ACM.

[407] Yin, J., Zhang, W., Li, Y., and Chen, H. (2009). A peer-to-peer-based multi-agent framework for decentralized grid workflow management in collaborative design. *The International Journal of Advanced Manufacturing Technology*, 41(3):407–420.

[408] Zhang, C., Mu, G., Chen, H., Sun, T., and Pang, L. (2014). Distributed service discovery algorithm based on ant colony algorithm. *Journal of Software*, 9(1):70–75.

[409] Zhang, D., Islam, M. M., and Lu, G. (2012). A review on automatic image annotation techniques. *Pattern Recognition*, 45(1):346–362.

[410] Zhang, R., Arpinar, I. B., and Aleman-Meza, B. (2003). Automatic composition of semantic web services. In Zhang, L.-J., editor, *ICWS*, pages 38–41. CSREA Press.

[411] Zhang, Y., Huang, H., Yang, D., Zhang, H., Chao, H.-C., and Huang, Y.-M. (2009). Bring qos to p2p-based semantic service discovery for the universal network. *Personal and Ubiquitous Computing*, 13(7):471–477.

[412] Zhang, Y., Zhao, J., and Cao, G. (2010). Roadcast: a popularity aware content sharing scheme in vanets. *ACM SIGMOBILE Mobile Computing and Communications Review*, 13(4):1–14.

[413] Zheng, Y., Cohen-Or, D., and Mitra, N. J. (2013). Smart variations: Functional substructures for part compatibility. *Computer Graphics Forum*, 32(2):195–204.

[414] Zhengdong, Z., Yahong, H., Ronggui, L., Weiguo, W., and Zengzhi, L. (2009). A p2p-based semantic web services composition architecture. In *Proceedings of the 2009 IEEE International Conference on e-Business Engineering*, pages 403–408. IEEE.

[415] Zhou, H., Zhang, Z., Wu, Y., and Qian, T. (2011). Bio-inspired dynamic composition and reconfiguration of service-oriented internetware systems. In *Advances in Swarm Intelligence*, pages 364–373. Springer.

[416] Zhou, J., Bochmann, G. v., and Shi, Z. (2014). Supporting decentralized sparql queries in an ad-hoc semantic web data sharing system. *International Journal of Networking and Computing*, 4(1):88–110.

[417] Zhou, J., Zhang, X., Bhuyan, L., and Liu, B. (2007). Clustered k-center: Effective replica placement in peer-to-peer systems. In *Proceedings of the 50th IEEE Global Telecommunications Conference*, pages 2008–2013. IEEE.

[418] Zhu, Y. and Hu, Y. (2006). Enhancing search performance on gnutella-like p2p systems. *IEEE Transactions on Parallel and Distributed Systems*, 17(12):1482–1495.

[419] Zhuge, H., Chen, X., and Sun, X. (2005). Preferential walk: towards efficient and scalable search in unstructured peer-to-peer networks. In *Special Interest Tracks and Posters of the 14th International Conference on World Wide Web*, pages 882–883. ACM.

[420] Zimeo, E., Troisi, A., Papadakis, H., Fragopoulou, P., Forestiero, A., and Mastroianni, C. (2008). Cooperative self-composition and discovery of grid services in p2p networks. *Parallel Processing Letters*, 18(3):329–346.

[421] Zinnikus, I., Cao, X., Klusch, M., Krauss, C., Nonnengart, A., Spieldenner, T., and Slusallek, P. (2013). A collaborative virtual workspace for factory configuration and evaluation. In *Proceedings of the 9th International Conference on Collaborative Computing: Networking, Applications and Worksharing*, pages 353–362. IEEE.