Deutsches
Forschungszentrum
für Künstliche
Intelligenz GmbH

Research
Report
RR-04-04

Linking Flat Predicate Argument Structures

Feiyu Xu

Oct. 2004

Linking Flat Predicate Argument Structures

Feiyu Xu

DFKI-RR-04-04

The work reported here is funded by the BMBF under the grant no. 01 IW C02 (QUETAL) and by the Human Language Technology Research Institute (HLTRI) in the Computer Science Department of the University of Texas at Dallas.

ISSN 0946-008X

Linking Flat Predicate Argument Structures

Feiyu Xu

DFKI GmbH
Language Technology Lab
Stuhlsatzenhausweg 3
D-66123 Saarbrücken
Germany

Email: feiyu@dfki.de, Telefon: 0681-3025287, Fax: 0681-3025338

Abstract

This report presents an approach to enriching flat and robust predicate argument structures with more fine-grained semantic information, extracted from underspecified semantic representations and encoded in Minimal Recursion Semantics (MRS). Such representations are provided by a hand-built HPSG grammar with a wide linguistic coverage. A specific semantic representation, called *linked predicate argument structure (LPAS)*, has been worked out, which describes the explicit embedding relationships among predicate argument structures. LPAS can be used as a generic interface language for integrating semantic representations with different granularities. Some initial experiments have been conducted to convert MRS expressions into LPASs. A simple constraint solver is developed to resolve the underspecified dominance relations between the predicates and their arguments in MRS expressions. LPASs are useful for high-precision information extraction and question answering tasks because of their fine-grained semantic structures. In addition, I have attempted to extend the lexicon of the HPSG English Resource Grammar (ERG) exploiting WordNet and to disambiguate the readings of HPSG parsing with the help of a probabilistic parser, in order to process texts from application domains. Following the presented approach, the HPSG ERG grammar can be used for annotating some standard treebank, e.g., the Penn Treebank, with its fine-grained semantics. In this vein, I point out opportunities for a fruitful cooperation of the HPSG annotated Redwood Treebank and the Penn PropBank. In my current work, I exploit HPSG as an additional knowledge resource for the automatic learning of LPASs from dependency structures.

Contents

1 Introduction

High-precision information extraction and question answering tasks require more precise and deep semantic understanding of natural language texts. [Surdeanu et al, 2003] suggest a new information extraction architecture, in which predicate argument structures play a central role for the template filling task. They show that mapping predicate argument structures to template structures is more straightforward and efficient than the traditional pattern-based approaches (e.g., [Hobbs et al., 1997]. At the same time, several attempts ([Crysmann et al., 2002], [Frank et al., 2003], [Riezler et al., 2002], [Tsujii, 2000], [Uszkoreit, 2002] and [Xu & Krieger, 2003] etc.), have been made to combine shallow and deep NLP, in order to achieve both robustness and precise semantic understanding of free texts. Most of these composition approaches work at the lexical and/or syntactic level, by, adding named entity recognition results or chunking results into the deep analysis. The starting point of my research is the other way around, namely, I want to make the robust semantics more precise by utilizing deep NLP semantic analysis as an additional knowledge resource. Advantages of my work are that inputs are more compatible with each other and that I can use predicates as my indices for integration operations. I suggest a semantic representation language (named *linked predicate argument structures, LPAS)* as the interface representation. LPAS adds some components of Minimal Recursion Semantics (MRS) [Copestake et al, 1999] to the existing flat predicate argument structure representation, e.g., meta variables for indexing the elementary predicate argument structures. LPAS is a meta-language for describing dominance and embedding relationships among predicate argument structures. LPAS can be regarded as a sub-language of MRS. In my work, I exploit HPSG as a source of additional semantic information for the automatic learning of LPASs from dependency structures. An automatic conversion algorithm has been developed and implemented. It extracts LPASs from MRS expressions. In addition, I attempted to extend the HPSG grammar lexicon by employing WordNet [Fellbaum, 1998] resources. Furthermore, a simple disambiguation method has been developed, which selects HPSG readings by using the parsing results of a probabilistic parser.

The remainder of this report is organized as follows. Section 2 gives a brief introduction to the shallow and deep semantic analysis. In Section 3, the LPAS representation is described. In Section 4, I present some corpora for training and evaluation. In Section 5, I explain how I extend the HPSG lexicon with WordNet. Section 6 shows how to convert MRS expressions to LPASs. Section 7 demonstrates the utilization of probabilistic parsing results as a selection criterion for disambiguating the results of the HPSG analysis. Section 8 identifies some features for the automatic learning of LPASs. Section 9 defines the evaluation task, designed to indicate which IE tasks will benefit from applying LPASs instead of shallow predicate argument structures. In Section 10, I summarize my results and discuss options for future work.

2 Shallow and Robust vs. Deep and Fine-Grained Semantic Analysis

The robust semantic kernel for predicate-argument classification [Moschitti and Bejan, 2004], called SEM, is a further development of the approach described by [Surdeanu et al., 2003]. Given a sentence, the output of SEM is a fat list of predicate argument structures where the predicates are in most cases verbs and the arguments are chunks in the form of surface strings without any internal semantic structures. The dominance or embedding relationships between two predicate argument structures are not specified, even if they are semantically related. Therefore, modality (e.g., belief, possibility, necessity) and scope (e.g., negation) information of a proposition is not identified by the system. However, it is known that modality and scope information modify or restrict the truth conditions of a proposition. Wrong facts will be extracted from the elementary predicate argument structures when their modality contexts are not considered. In addition, SEM cannot deal with phenomena where predicate argument relationships are expressed implicitly in the surface form. Typical examples can be found in some linguistic constructions where passive, infinitive VP, control or unbounded dependencies interact with each other. For example,

```
(1)   After the retirement of Peter Smith, Mary Hopp was asked to take over
      the development sector.
```

This example contains a passive VP and an infinitive VP. The NP "Mary Hopp" is the object of the verb "ask" and is at the same time the subject of the verb "take over". Further, the proposition expressed by the predicate "take over" is embedded in the modality uttered by the verb "ask".

As mentioned above, SEM cannot specify the embedding relationships between the two predicates "take over" and "ask". Therefore, its output is only a flat list of elementary predicate argument structures, see (2):

(2)

```
                  {
                  [PRED: ask,
                   ARG0: __,
                   ARG1: Mary Hopp,
                   ARG2: take over the development sector],

                  [PRED: take_over,
                   ARG0: __
                   ARG1: the development sector]
                  }
```

Although ARG2 of the predicate "ask" gives us a hint that there is a potential embedding relationship between "ask" and "take over", there is no structural link available. Therefore, two important relationships remain unresolved in this output. First, the embedding relationship between the two predicate argument structures, namely, the "take_over" predicate argument structure should be ARG2 of the "ask" structure. Second, ARG0 of the verb "take over" should refer to the same entity as ARG1 of the verb "ask".

Resolving these relationships is important for both information extraction and question answering tasks. For example, consider the following question:

(3) `Who took over the development sector after the retirement of Peter Smith?`

Since ARG0 of the predicate "take over" is not resolved in (2), an exact answer to "who" cannot be derived. Even if "Mary Hopp" is detected as ARG0 of the verb "take over" but the embedding relationship is unresolved, extracting "Mary Hopp" as a direct answer to the question is also wrong because the modality expressed by "ask" weakens the certainty of the fact. If I want to extract the correct management succession information from sentence (1), I therefore cannot simply add "Mary Hopp" as the person who obtains a new position because of the modality context.

Let me provide another simple example from open-domain question answering:

(4)

Question:
`What did the researchers report about asbestos?`
Answer:
`A form of asbestos … has caused a high percentage of cancer deaths …, researchers reported …`

The semantic representation of the question can be:

(5)

```
        [PRED: report,
         ARG0: researchers,
         ARG1: ?/asbestos]
```

"?/asbestos" represents an unknown predicate argument structure "?" and "?" takes "asbestos" as an argument.

The desired semantic representation of the answer sentence should be:

(6)

```
          [PRED: report,
           ARG0: researchers,
           ARG1: [PRED: cause,
                  ARG0: asbestos,
                  ARG1: a high percentage of cancer deaths]]
```

Matching the question semantics against the answer semantics, ARG1 of the predicate "report" in the question semantics can unify with ARG1 of the predicate "report" in the answer semantics. Thus, the equation looks like (7):

```
(7)
?/asbestos=cause(arg0:asbestos,
                 arg1:a high percentage of cancer deaths)
```

The resolution of ? is then:

```
(8)
?=lambda x. cause(arg0:x,
                  arg1:a high percentage of cancer deaths)
```

Given this semantic representation, I can generate an exact answer like

```
"Researchers reported that asbestos are something, that cause a high
percentage of cancer deaths"
```

However, the output of the current system SEM for the answer sentence is:

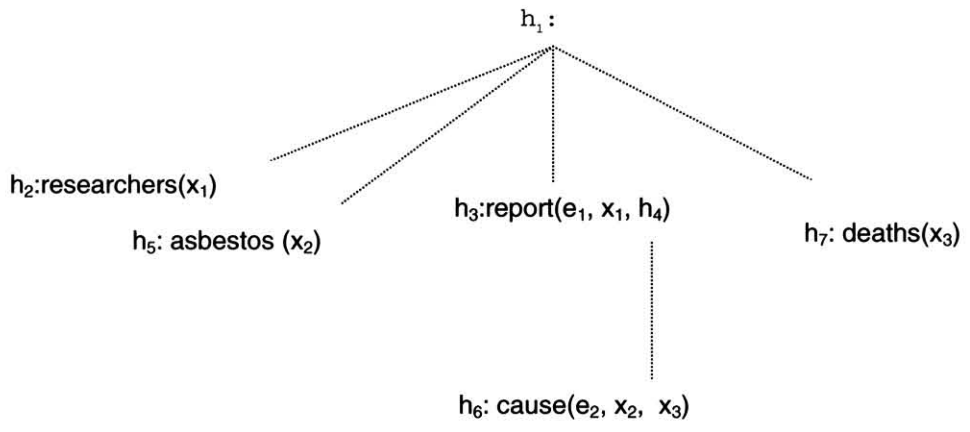
```
(9)
         {
         [pred: report,
          arg0: researchers,
          arg1: a form of asbestos … has caused … death],

         [pred: cause,
          arg0: a form of asbestos,
          arg1: a high percentage of cancer deaths]
         }
```

Of course, I might be able to apply some simple heuristics, based on simple string match, to find out that "asbestos" is embedded in an argument of the predicate "report" and delivers *arg1* of "report" as my answer. But this would not work for other embeddings. Here I attempt to find some generic methods for solving these kinds of problems properly.

In contrast to SEM, MRS establishes some implicit linkage (dominance relation) among the predicate argument structures that expresses the potential embedding relationship. The predicate argument structures are called *elementary predications* (EP) in MRS expressions. The MRS representation of the semantics for the answer sentence can simply be represented as a graph with underspecified dominance relations among the nodes. Each node represents an EP, labelled with an index "h_i", see the following example (10):

(10)



I see that the predicate argument structure of "cause" is labelled by "h_6". One of its arguments is "x_2" and "x_2" is described in "h_5" as "asbestos". At the same time, "h_6" is dominated by "h_4". "h_4" is an argument of the predicate argument structure of "report", labelled by "h_3". This means that h_6 contributes to the semantics of h_4 and therefore also h_3. Given these dominance relations, I can extract from the MRS representation (10) a semantic representation, which is semantically equivalent with (6). But the predicate-argument-structure output (9) provided by SEM does not contain sufficiently precise information for producing (6).

This raises the question whether MRS expressions alone are not sufficient for the high-precision real-world applications when they are fine-grained enough. The first problem I are faced with is the ambiguity problem. The English MRS is constructed by the large-scale English HPSG[1] grammar, called English Resource Grammar (ERG)[2] [Copestake and Flickinger, 2000]. The current ERG grammar keeps linguistic analysis and semantic construction as generic as possible. There is no domain information encoded. The HPSG parser delivers for each sentence all syntactic analyses. Therefore, additional processing and knowledge is needed for choosing the right reading for the application. In addition, the generic semantic representation keeps all the uncertain relationships underspecified. In particular, the spans of the arguments are often underspecified, in order to cover the quantifier scope ambiguities in its representation. However, in real-world applications it is important to know at least the boundary information of each argument. Moreover, HPSG has to overcome the robustness and coverage problem, in order to be able to deal with realistic texts such as newswire reports. In comparison to the HPSG parser, SEM was developed for real-world texts. It is very robust and its output is unambiguous, providing only one reading for each sentence. The argument boundary of a predicate is also specified and unambiguous.

3 A Richer Semantic Representation: Linked Predicate Argument Structure (LPAS)

In this section, I will at first give a very brief introduction to MRS and then describe the representation language LPAS.

3.1 MRS

MRS [Copestake et al., 1999] was developed as a flexible and generic semantic representation language for HPSG. MRS provides a compact semantic representation, which can express scope ambiguities without listing all readings. In particular, it provides a framework allowing construction of semantic representations for computational grammars, which can be expressed in terms of feature structures.

MRS is not a semantic theory. It is a meta language for describing semantic structures for some underlying object language. The object language in the ERG grammar is first-order predicate logic with generalized quantifiers. The units of MRS are elementary predicate argument structures called elementary predications (EPs). An EP is a single relation with its associated arguments:

```
(11) F(x,y, z, ...).
```

ERG uses Neo-Davidsonian representation to express verb semantics:

```
(12) sleep(e1,x), peter(x)
```

Here the sleeping event is introduced as an argument of the sleep relation.

The syntax of an MRS is defined as follows:

```
 An MRS structure is a triple <T, L, C>
T: a top handle (no handle h_i can outscope T)
L: a bag of EPs
C:  a  bag  of  handle  constraints,  using  qeq  (equality  modulo
quantifiers) operator, expressing the embedding and scope information
between EPs
```

Let us give a simple example of the MRS representation of the scope ambiguities:

```
(13) Every dog chases some fat cat.
```

This sentence has two readings corresponding to the two orderings of the quantifiers "every" and "some", with respect to scope see a) and b).

```
a) every (x, dog(x), some(y, fat(y)&cat(y), chase(e,x,y)))
b) some(y, fat(y)&cat(y), every(x,dog(x),chase(e,x,y)))
```

[1] Head-driven Phrase Structure Grammar (HPSG) [Pollard and Sag, 1994]. HPSG is a constraint-based, lexicalist approach to grammatical theory. It models languages as systems of constraints. Linguistic units and their relations are expressed by typed feature structures. In case of unbounded dependencies, there are no transformations needed. The lexicon is organized via multiple inheritance hierarchies. More information about the HPSG can be found under: http://lingo.stanford.edu/erg.html, http://www.coli.uni-sb.de/%7Ehansu/courses/hpsg_arch.html, http://www.coli.uni-sb.de/%7Ehansu/psfiles/hpsg.ps
[2] http://lingo.stanford.edu/erg.html

A quantifier in MRS has three arguments: a variable, the handle index of the restrictor, and the handle index of the body. For example, the MRS representation of the determiner "a" is:

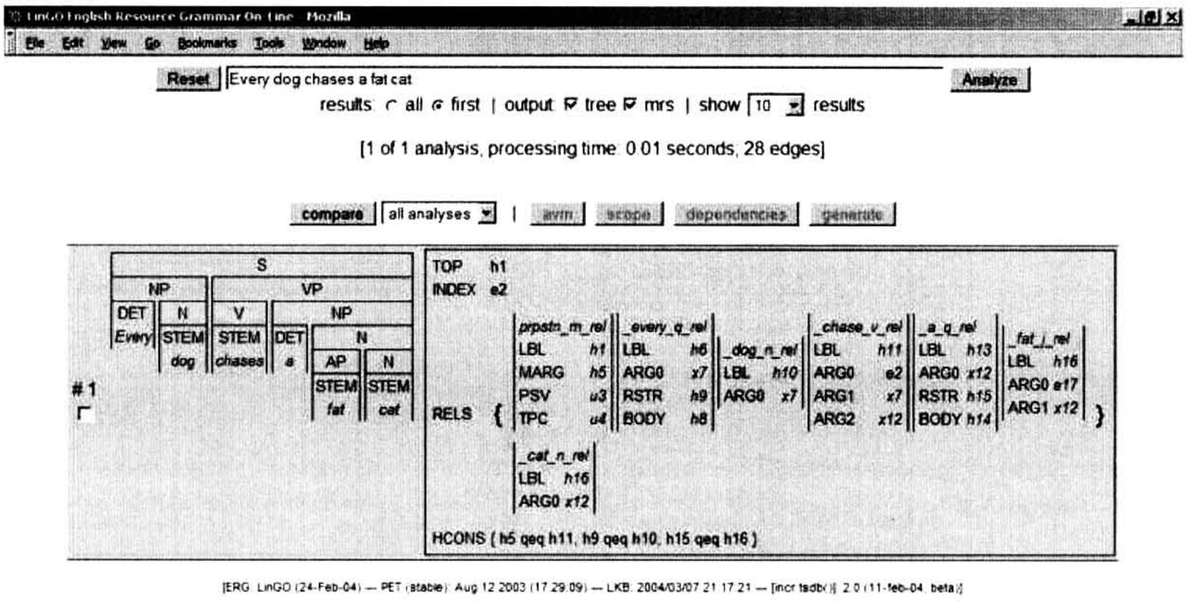
$$\texttt{some}(x,\ h_i,\ h_j)$$

x: quantification variable
h_i: handle index of the restrictor
h_j: handle index of the scope body

The MRS representation of (13) can be simplified as follows:

$<h_0,$
$\{$
  h_8: every(x, h_6, h_7),
  h_9: fat(y),
  h_9: cat(y),
  h_{11}: chase(x,y),
  h_{12}: dog(x),
  h_{13}: a(y, h_1, h_2),
$\}$
$\{$
  h_6 qeq h_{12},
  h_1 qeq h_9
$\}$
$>$

A more detailed representation of the MRS in the typed feature structures can be found in the ERG grammar online demo[3]. The screenshot below shows the phrase structure analysis and the MRS output of my sample sentence.



Every dog chases a fat cat

results: ( ) all (•) first | output: ☑ tree ☑ mrs | show 10 results

[1 of 1 analysis, processing time: 0.01 seconds; 28 edges]

all analyses

```
TOP    h1
INDEX  e2
RELS { [ prpstn_m_rel  LBL h1  MARG h5  PSV u3  TPC u4 ]
       [ _every_q_rel  LBL h6  ARG0 x7  RSTR h9  BODY h8 ]
       [ _dog_n_rel  LBL h10  ARG0 x7 ]
       [ _chase_v_rel  LBL h11  ARG0 e2  ARG1 x7  ARG2 x12 ]
       [ _a_q_rel  LBL h13  ARG0 x12  RSTR h15  BODY h14 ]
       [ _fat_j_rel  LBL h16  ARG0 e17  ARG1 x12 ]
       [ _cat_n_rel  LBL h16  ARG0 x12 ] }
HCONS ( h5 qeq h11, h9 qeq h10, h15 qeq h16 )
```

[ERG: LinGO (24-Feb-04) — PET (stable): Aug 12 2003 (17:29:09) — LKB: 2004/03/07 21:17:21 — [incr tsdb()]: 2.0 (11-feb-04, beta)]

[3] http://lingo.stanford.edu:8000/erg

3.2 Linked Predicate Argument Structure (LPAS)

Just as MRS, LPAS is not a semantic theory but a meta language for describing the relationships between predicates and arguments in the underlying object language. The primary unit of the object language is composed of predicates and arguments. The generalized quantifiers are not considered now. I borrow some components of MRS and add them into the shallow predicate argument structures. Therefore, LPAS can be regarded as a sub-language of MRS.

- `argument variable:  x, y, z denoting individual variables`

  ```
  Mary(x)
  ```

- `handle: labels a predicate argument structure, e.g.,`

  h_1: visit(arg0:h_2, arg1:h_3)
  h_2: Mary(x)
  h_3: Peter(y)

- `linking relations: a handle fills the value of an argument`

  arg_i: h_i

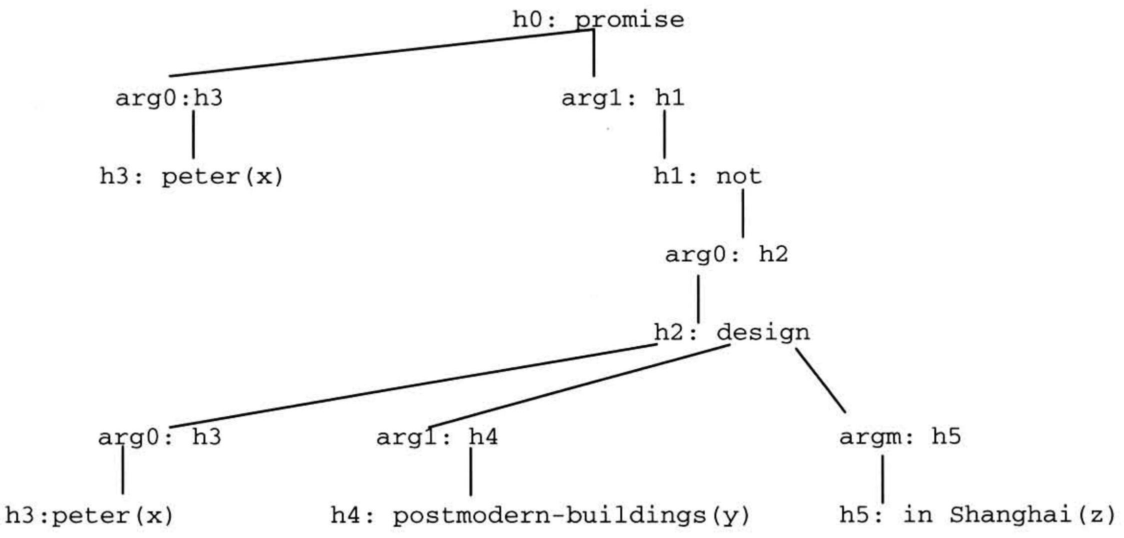
Given these components, I can easily express the linkage among predicates and arguments together with the explicit argument bindings. For the sentence

```
(14) Peter promised not to design postmodern buildings in Shanghai.
```

Its LPAS representation is

```
h0: promise (arg0: h3, arg1:h1)
h1: not(arg0:h2)
h3: Peter(x)
h2: design(arg0:h3, arg1:h4, argm:h5)
h4: postmodern buildings(y)
h5: in Shanghai(z)
```

Its graphical visualization can be depicted as follows:



4 Corpora

I consider different resources as the potential corpora for my experiments and evaluations:

- A treebank with fine-grained semantic annotation: Redwoods treebank4
- A treebank with rather shallow predicate-argument structure annotation: PropBank5
- An annotated real-world information extraction corpus: terrorism domain in Iraq, MUC-6/MUC-7

4.1 Redwoods Treebank

The Redwoods treebank has been developed by the Linguistic Grammars Online (LinGo) Consortium. This treebank currently contains 10,000 annotated trees based on the Verbmobil data. Verbmobil data has been used for the development of a speech-to-speech translation dialog system. In comparison to other treebanks, which use newswire texts, the sentences in the Verbmobil data are relative short and simple. The annotation is in three different formats, viz. (i) as a derivation tree composed of identifiers of lexical items and constructions used to construct the analysis, (ii) as a traditional phrase structure tree labelled with an inventory of some fifty atomic labels (of the type `S', `NP', `VP', etc.), and (iii) as an underspecified MRS meaning representation. I will use the Redwoods treebank to create an LPAS gold standard corpus by converting the MRS expressions in the treebank to LPAS.

4.2 Proposition Bank (PropBank)

PropBank ([Kingsbury et al, 2002] and [Kingsbury and Palmer, 2002]) is a one million word corpus annotated with predicate-argument structures. The corpus consists of the Penn Treebank 2 Wall Street Journal texts[6]. At the current stage, predicates in PropBank are mostly verbs. For each verb predicate, a survey was made to determine the syntactic frame of the verb and its major senses. I consider PropBank as an additional resource for my training and evaluation tasks. This is an example of the PropBank annotation:

```
(15)
      Analysts have been expecting a GM-Jaguar pact that would give the
      U.S. car maker an eventual 30% stake in the British company.
```

```
(S Arg0 (NP-SBJ Analysts)
   (VP have
      (VP been
         (VP expecting
               Arg1 (NP (NP a GM-Jaguar pact)
                        (SBAR (WHNP-1 that)
                              (S Arg0 (NP-SBJ *T*-1)
                                  (VP would
                                     (VP give
                                        Arg2 (NP the U.S. car maker)
                                        Arg1 (NP (NP an eventual (ADJP 30 %) stake)
                                                 (PP-LOC in (NP the British company))))))))))))
```

The annotated predicate argument information of this sentence is

```
expect(Analysts, GM-J pact)
give(GM-J pact, US car maker, 30% stake)
```

This semantics is compatible with LPAS.

[4] http://redwoods.stanford.edu/
[5] http://www.cis.upenn.edu/~ace/
[6] http://www.cis.upenn.edu/~treebank

4.3 IE Corpus

In addition to the linguistic resources, I plan to choose one or two domains and evaluate the contribution of LPASs to the performance of the IE tasks. Both MUC-6 and MUC-7 data can provide suitable domains and data samples.

5 Extending HPSG Lexical Coverage with WordNet

The ERG lexicon has 11991 entries (about 4896 nouns, 2451 verbs). Still the lexicon coverage is very small for real world applications, in particular, for processing newspaper texts. I have developed a method, which can automatically generate lexical entries for an unknown word based on the information provided in WordNet[7].

My first experiment with ERG grammar shows that the lexical coverage plays a crucial role for sentence parsing. I took 74 questions from the domain of the Iraq mass destruction weapons. Here are some sample questions:

```
-   Does Iraq have biological weapons?
-   How can a biological weapons program be detected?
-   What is the evidence that Iraq has biological weapons?
-   Can UN inspectors find weapons in Iraq?
-   What toxins might be a constituent in biological weapons?
-   What makes biological weapons so lethal?
-   What is a biological weapon?
```

At the beginning, no question can be parsed because of the missing lexical entries. After I have manually added 56 lexical entries to the ERG grammar, 58 sentences can be parsed.

The lexicon entries in HPSG are organized in a multiple inheritance type hierarchy. Each lexicon entry has been defined as an instance of a type. Therefore, given an unknown word, the first task is to find out its corresponding type definition. I propose to use WordNet as a resource for my lexicon extension. This approach is based on the assumption that words with similar or same senses in the same part-of-speech category should own similar or same syntactic features.

WordNet is a lexical database where sets of synonyms, called, *synsets*, are connected via different semantic relations: ISA, part-of, etc. Each synset represents one underlying lexical concept. WordNet contains totally 152,059 words (114648 nouns, 11306 verbs, 21436 adjectives and 4669 adverbs).

In my approach, the detection of the type definitions of an unknown word is done by finding a specific word in the WordNet. The word should already exist in the HPSG lexicon and that is at the same time most similar to the unknown word in its lexical semantics. I call such a word, "known and similar word". A similar word is either a synonym or a hypernym, or the hyponym of a hypernym of the unknown word. If I can find such a "known and similar" word, I will adapt its type definition to the unknown word. For example, given a new word, e.g., "Iraq", I look for the hypernyms of "Iraq". They are "Asian country", "country", "state" and "land" etc. The named entity recognition tool tells us that "Iraq" is a country name. Then I look for the hyponyms of the word "country". WordNet gives us a list of country names: "Australia", "Israel", etc. The type definition of "Australia" in the HPSG lexicon is

```
australia_n1 := n_proper_le &
                  [ STEM < "australia" >,
                  SYNSEM.LKEYS.KEYREL.CARG "australia" ].
```

Then I can use this type definition for all the country names I know. If a word has multiple senses or multiple part-of-speeches, multiple lexical entries will be generated. For example, the word "threaten" is an unknown word for HPSG and has three senses in the WordNet:

```
Sense 1
endanger, jeopardize, jeopardise, menace, threaten, imperil, peril
       => exist, be
Sense 2
```

[7] http://www.cogsci.princeton.edu/~wn/

```
threaten
       => warn
           => inform
               => communicate, intercommunicate
                   => interact
                       => act, move
```
Sense 3
```
threaten
       => bode, portend, auspicate, prognosticate, omen, presage, bet
oken, foreshadow, augur, foretell, prefigure, forecast, predict
           => bespeak, betoken, indicate, point, signal
               => tell
                   => inform
                       => communicate, intercommunicate
                           => interact
                               => act, move
```

I generate 26 entries for this word

```
;;;threaten --> exist
      threaten_v1 := v_unacc_le &
       [ STEM < "threaten" >,
         SYNSEM.LKEYS.KEYREL.PRED "_threaten_v_1_rel" ].

      threaten_v2 := v_there_le &
       [ STEM < "threaten" >,
         SYNSEM.LKEYS.KEYREL.PRED "_threaten_v_expl_rel" ].

;;;threaten --> warn
      threaten_v3 := v_np_trans_cp_le &
       [ STEM < "threaten" >,
         SYNSEM.LKEYS.KEYREL.PRED "_threaten_v_rel" ].

;;;threaten --> inform
      threaten_v4 := v_np_trans_cp_le &
       [ STEM < "threaten" >,
         SYNSEM.LKEYS.KEYREL.PRED "_threaten_v_rel" ].

      threaten_v5 := v_np_trans_le &
       [ STEM < "threaten" >,
         SYNSEM.LKEYS.KEYREL.PRED "_threaten_v_rel" ].

      threaten_v6 := v_empty_prep_trans_le &
       [ STEM < "threaten" >,
         SYNSEM.LKEYS [ --OCOMPKEY _of_p_sel_rel,
                        KEYREL.PRED "_threaten_v_rel" ] ].

      threaten_v7 := v_unerg_le &
       [ STEM < "threaten" >,
         SYNSEM.LKEYS.KEYREL.PRED "_threaten_v_rel" ].

      threaten_v8 := v_obj_equi_le &
       [ STEM < "threaten" >,
         SYNSEM.LKEYS.KEYREL.PRED "_threaten_v_rel" ].

;;;threaten --> communicate
      threaten_v9 := v_unerg_le &
       [ STEM < "threaten" >,
         SYNSEM.LKEYS.KEYREL.PRED "_threaten_v_rel" ].

;;;threaten --> act
      threaten_v10 := v_unerg_le &
       [ STEM < "threaten" >,
         SYNSEM.LKEYS.KEYREL.PRED "_threaten_v_rel" ].

;;;threaten --> move
      threaten_v11 := v_unacc_le &
       [ STEM < "threaten" >,
         SYNSEM.LKEYS.KEYREL.PRED "_threaten_v_1_rel" ].

      threaten_v12 := v_np*_trans_double_pp*_le &
```

```
         [ STEM < "threaten" >,
           SYNSEM.LKEYS [ --OCOMPKEY _from_p_sel_rel,
                          KEYREL.PRED "_threaten_v_cause_rel" ] ].

        threaten_v13 := v_np_trans_le &
         [ STEM < "threaten" >,
           SYNSEM.LKEYS.KEYREL.PRED "_threaten_v_cause_rel" ].

;;;threaten --> forecast
        threaten_v14 := v_np_trans_le &
         [ STEM < "threaten" >,
           SYNSEM.LKEYS.KEYREL.PRED "_threaten_v_rel" ].

        threaten_v15 := v_cp_prop_non_trans_le &
         [ STEM < "threaten" >,
           SYNSEM.LKEYS.KEYREL.PRED "_threaten_v_rel" ].

;;;threaten --> predict
        threaten_v16 := v_np_trans_le &
         [ STEM < "threaten" >,
           SYNSEM.LKEYS.KEYREL.PRED "_threaten_v_rel" ].

        threaten_v17 := v_cp_prop_non_trans_le &
         [ STEM < "threaten" >,
           SYNSEM.LKEYS.KEYREL.PRED "_threaten_v_rel" ].

;;;threaten --> point
        threaten_v18 := v_np*_trans_le &
         [ STEM < "threaten" >,
           SYNSEM.LKEYS.KEYREL.PRED "_threaten_v_1_rel" ].

;;;threaten --> signal
        threaten_v19 := v_np*_trans_le &
         [ STEM < "threaten" >,
           SYNSEM.LKEYS.KEYREL.PRED "_threaten_v_rel" ].

        threaten_v20 := v_np*_trans_cp_le &
         [ STEM < "threaten" >,
           SYNSEM.LKEYS.KEYREL.PRED "_threaten_v_rel" ].

        threaten_v21 := v_obj_equi_le &
         [ STEM < "threaten" >,
           SYNSEM.LKEYS.KEYREL.PRED "_threaten_v_rel" ].

;;;threaten --> tell

        threaten_v22 := v_ditrans_only_le &
         [ STEM < "threaten" >,
           SYNSEM.LKEYS.KEYREL.PRED "_threaten_v_1_rel" ].

        threaten_v23 := v_np_trans_cp_fin_or_inf_le &
         [ STEM < "threaten" >,
           SYNSEM.LKEYS.KEYREL.PRED "_threaten_v_1_rel" ].

        threaten_v24 := v_obj_equi_le &
         [ STEM < "threaten" >,
           SYNSEM.LKEYS.KEYREL.PRED "_threaten_v_1_rel" ].

        threaten_v25 := v_np_trans_le &
         [ STEM < "threaten" >,
           SYNSEM.LKEYS.KEYREL.PRED "_threaten_v_1_rel" ].

        threaten_v26 := v_cp_fin_inf_non_trans_le &
         [ STEM < "threaten" >,
           SYNSEM.LKEYS.KEYREL.PRED "_threaten_v_1_rel" ].
```

Of course, some of the above type definitions are redundant and can be further clustered together. In this example, we consider all hypernyms of the unknown word. However, it can lead to overgeneration of lexical variants. Therefore, we use a pre-selection mechanism. We prefer the type definition of the synonyms of the unknown word. If there is no synonym available, the hypernyms or the hyponyms of the hypernyms will be preferred.

After the entries are generated, a manual correction will be followed. There are two typical error sources:

- Wrong sense
    - One of the type definitions of the similar word does not correspond to the shared sense, because the similar word can also have multiple senses and the shared sense is not defined in the HPSG lexicon.
- Wrong subcategorization frames
    - Although verbs share common senses, their arguments can be realized in different syntactic structures, e.g., word choice of prepositions.

6 Normalization of Linguistic Constructions

In addition to the extension of the lexicon entries, I have also normalized the question texts:

- Remove the punctuation: e.g., 'suitcase bomb',

In addition, some linguistic constructions cannot be handled by the current Lingo ERG grammar properly:

- inverted constructions fronted
    - e.g. "..., the researchers said"
- appositions
    - e.g., "Peter Müller, born in Aachen, moved to England last year."
- named entities
    - e.g., "Peter Müller, 60 years old, moved to Germany. "

Methods are to work out, which help to normalize these constructions and make them parsable by the HPSG grammar. Otherwise, If HPSG is used in a connection with a shallow IE system, then I can remove the apposition and proceed in one of the following two ways. (i) present the entire NP with apposition to the shallow system if the shallow system can handle appositions. (ii) add a sentence of the form *<core NP> "is"|"are" <apposition>*, e.g. "Peter Müller is born in Aachen" or "Peter Müller is 60 years old".

7 Converting MRS Expressions to LPASs

As described in Section 2 and 3, the embedding relationships between two predicate argument structures in MRS expressions are in general underspecified. I have developed an algorithm, which converts automatically the underspecified relationships into some explicit embedding relationships. In order to explain my conversion algorithm properly, let me give an example of MRS analysis of the following sentence: The MRS representation of the following sentence shall be served as an example for the explanation of the conversion algorithm:

```
(16)
     Peter persuades Mary to come.
```

In fact, (16) contains two elementary predicate argument structures: one with the predicate "persuade" and another one with the predicate "come". The latter one is an argument of the former one.

The MRS representation of (16) is as follows:

```
[ LTOP: h1                                //top label
  INDEX: e2 [ EVENT                       //top event index
              E.MOOD:  INDICATIVE
              E.TENSE:  PRESENT
              E.ASPECT.PROGR:  -
              E.ASPECT.PERF:  - ]
  RELS: <                                 // a set of EPs
          [ prpstn_m_rel
            LBL: h1
            MARG: h3 ]                    //h3 is dominated by h1
```

```
        [ proper_q_rel  //proper name is treated as a quantifier
          LBL: h4
          ARG0: x5 [ NONCONJ_REF-IND
                       PNG.GEN:  REAL_GENDER
                       PNG.PN:  3SG
                       DIVISIBLE:  -
                       PRONTYPE:  PRONTYPE ]
          RSTR: h7
          BODY: h6 ]
        [ named_rel
          LBL: h8
          ARG0: x5
          CARG: "peter" ]
        ["_persuade_v_rel"
          LBL: h9
          ARG0: e2
          ARG1: x5
          ARG2: x10 [ NONCONJ_REF-IND
                       PNG.GEN:  REAL_GENDER
                       PNG.PN:  3SG
                       DIVISIBLE:  -
                       PRONTYPE:  PRONTYPE ]
          ARG3: h11 ]
        [ proper_q_rel
          LBL: h12
          ARG0: x10
          RSTR: h14
          BODY: h13 ]
        [ named_rel
          LBL: h15
          ARG0: x10
          CARG: "mary" ]
        [ prpstn_m_rel
          LBL: h11
          MARG: h16 ]
        ["_come_v_1_rel"
          LBL: h17
          ARG0: e18 [ EVENT
                       E.TENSE:  NO_TENSE
                       E.MOOD:  INDICATIVE
                       E.ASPECT.PROGR:  -
                       E.ASPECT.PERF:  - ]
          ARG1: x10 ]
      }
HCONS: {
          h3 QEQ h9       //set of handle constraints
          h7 QEQ h8
          h14 QEQ h15
        h16 QEQ h17
    >]
```

Let us consider the two highlighted EPs h_9, h_{17}. In h_9, "persuade" takes in addition to the event argument e_2 three other arguments: x_5, x_{10} and h_{11}. x_5, x_{10} refer to "Peter" and "Mary" respectively. In h_{17}, the predicate "come" takes the argument x_{10}, in addition to the event variable e_{18}. In the set of handle constraints (abbr. *HCONS*), there is a dominance relation between h_{16} and h_{17}. That is h_{17} is an argument of h_{16}. However, there were no explicit dominance relations specified between h_9 and h_{17} or h_{11} and h_{17}. The bridge between h9, h11, h17 can be found in the EP:

```
[ prpstn_m_rel
  LBL: h11
  MARG: h16 ]
```

This tells us that h_{16} is an argument of h_{11}. Since I know from HCONS that h_{17} is an argument of h_{16}, h_{17} is then an argument of h_{11} according to the transitivity property of the embedding relation. Analogically, h_{17} is an argument of h_9, because h_{11} is an argument of h_9.

Thus, my algorithm extracts on the one hand the predicate argument structures from the MRS representations and on the other hand resolves the embedding relationships among them. The output of my algorithm contains three parts:

- Embedding relations
- Elementary predicate argument structures with verbs as predicates.
- Inverted index of arguments

This is the preliminary output:

1. h_{17} is-argument-of h_{11}, h_{17} is-argument-of h_9
2.

```
   {
      ["_persuade_v_rel"
        LBL: h9
        ARG0: e2
        ARG1: x5
        ARG2: x10
        ARG3: h11 ],
      ["_come_v_1_rel"
        LBL: h17
        ARG0: e18
        ARG1: x10 ]
 }
```

3.

```
x10: {
   <h17, ARG1, [pred="_come_v_1_rel",
                ARG0=e18]>,
   <h15, ARG0, [pred=named_rel,
                CARG="mary"]>,
   <h12, ARG0, [pred=proper_q_rel,
                RSTR=h14,
                BODY=h13]>,
   <h9, ARG2,  [pred="_persuade_v_rel",
                 ARG3=h11,
                 ARG2=x10,
                 ARG1=x5,
                 ARG0=e2]>
   }
```

...

In the inverted index of arguments, I have collected for each variable all handles where the variable occurs. This inverted index helps us to identify the argument boundary. I can make use of this information and the argument string provided by the SEM for the determination of the right argument boundary.

8 Disambiguation of HPSG Parsing Results with a Statistical Parser

Because of lexical and structural ambiguity the HPSG parser often arrives at several analyses. In order to identify the right MRS interpretation for further processing, I decide to utilize the parsing result provided by SEM. The parser used in SEM is based on the probabilistic parser reported in [Collins, 1997].

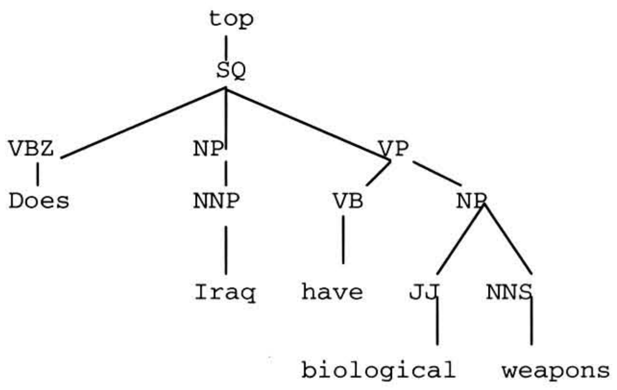
However, the formats of the parsing results by the probabilistic parser and the HPSG parser are very different. The HPSG yields derivation trees with binary and unary branching where the non-terminal nodes are labelled with the names of the HPSG rule that were applied to construct them and the terminal-nodes with the respective surface strings of the input sentence. The result of the probabilistic parser is an n-ary branching tree whose non-terminal nodes are labelled with syntactic categories and the terminal-nodes are labelled with the word surface forms.

My task is to recognize which parse among the HPSG analyses corresponds to the result of the probabilistic parser. If such a corresponding HPSG parse can be found , I will choose this analysis and ignore the alternative results. Consider the following concrete example.

```
(17) Does Iraq have biological weapons?
```

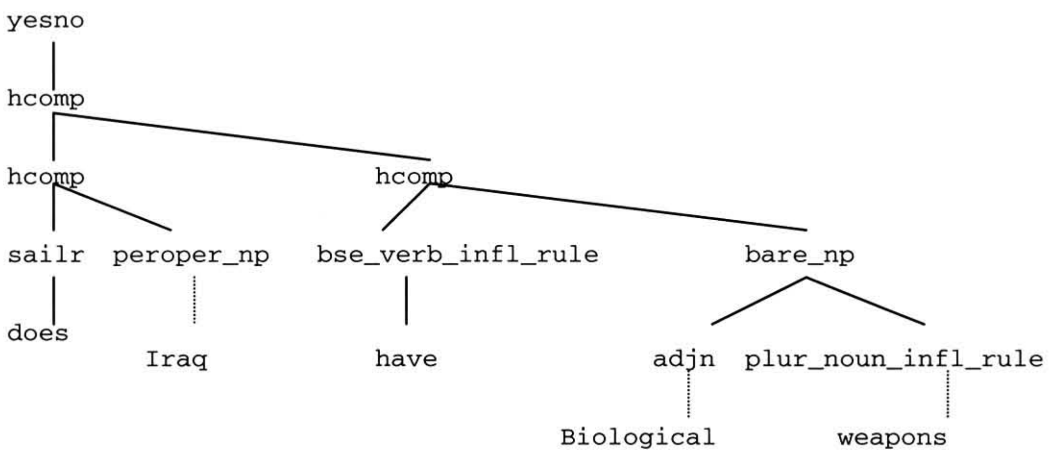
The probabilistic parser delivers the following parsing tree:

```
(18)
(TOP (SQ (VBZ Does) (NP (NNP Iraq)) (VP (VB have) (NP (JJ biological) (NNS
weapons) (. ?)))))
```



The HPSG derivation tree has the following structure. Each line represents information about a node in the derivation tree. The terminal nodes are labeled with an index number, a confidence value, the start and end position of the surface string, and a comment.

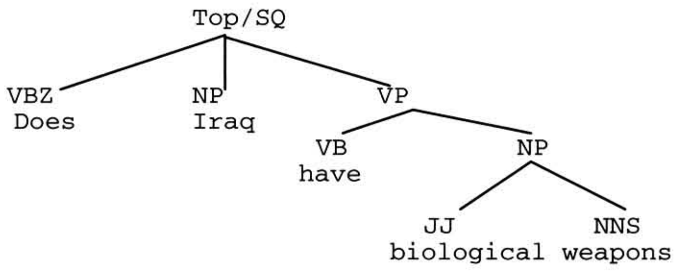
```
(19)
(152 yesno 24.92 0 5 [root_strict]
  (151 hcomp 21.73 0 5
    (61 hcomp 9.81 0 2
      (45 sailr 0.94 0 1
        (1 does1_pos/va_does_le 0.30 0 1 []
          ("does" 0.00 0 1)))
      (59 proper_np 9.05 1 2
        (58 sing_noun_infl_rule 1.86 1 2
          (11 iraq_n1/n_proper_le 0.00 1 2 []
            ("iraq" 0.00 1 2)))))
    (150 hcomp 10.24 2 5
      (30 bse_verb_infl_rule 2.89 2 3
        (18 have-poss/v_poss_le 3.67 2 3 []
            ("have" 0.00 2 3)))
      (140 bare_np -2.47 3 5
(138 adjn_i -3.34 3 5
  (71 pos_adj_infl_rule 0.62 3 4
      (23 biological_isect/adj_intrans_le 0.00 3 4 []
            ("biological" 0.00 3 4)))
  (137 plur_noun_infl_rule -3.96 4 5
           (24 weapon_n1/n_intr_le 0.00 4 5 [plur_noun_infl_rule]
            ("weapons" 0.00  4 5))))))))
```



To permit a comparison of these two different tree formats with completely different labeling, I undertook a form of tree normalization, i.e., I first collapse sequences of unary nodes.

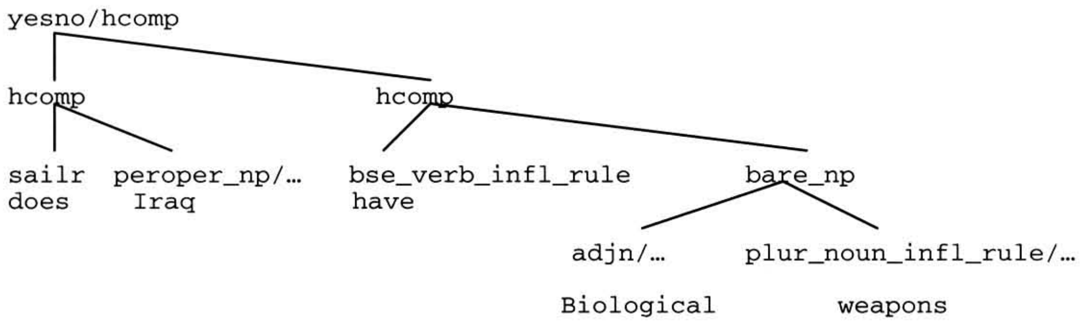
After the normalization, (18) and (19) are simplified to (20) and (21) respectively:

(20)



(does, Iraq, (have, (biological, weapons)))

(21)



((does, Iraq), (have, (biological, weapons)))

Although the underlying syntactic analysis of (20) and (21) is the same, it is still very difficult to match them directly. Therefore, in a further step I will convert one format into the other.

My idea for the automatic matching is:

- utilize the Redwoods treebank as my corpus and apply the statistical parser to it
- learn mapping rules, which can convert one format to another
  - identify features for the learning: category names, lexical information, etc.

9 Statistical Approaches to Learning Predicate Argument Structures

[Surdeanu et al., 2003] shows an inductive learning method, which labels the constituents in the dependency trees with predicate argument structures automatically. Two different feature sets and their combinations have been considered and employed. In this way they were able to obtain an F-score 17% higher than the method suggested in [Gildea and Palmer, 2002]. [Moschitti and Bejan, 2004] apply an SVM to classify even more argument types. However, as mentioned above, all these learned predicate argument structures are still very flat.

In the previous section, I have shown how to convert MRS expressions to LPASs. In order to learn rules for formulating dominance or embedding information between two predicate argument structures and the appropriate variable bindings, I consider the following information resources as potential features:

- *String position*:
  - If the surface form of a predicate argument structure is a sub-string of the surface form of an argument of another predicate, there is a dominance relation between them.
- *Dominance relation*:
  - Dominance constraints encoded in MRS expressions
- *Variable binding*:
  - Variable binding information encoded in MRS expressions
- *Argument span*
  - String information of SEM arguments
  - Inverted index of arguments encoded in the LPASs, which are converted from MRS expressions
- *Linguistic phenomena*:
  - Raising
  - Passive
  - Control/subject
  - Control/object
  - Long-distance dependency
  - Coordination VP

10 Information Extraction with Linked Predicate Argument Structures

[Surdeanu et al., 2003] report a 17% F-measure improvement in the performance of their information extraction task using their methods in comparison to [Gildea and Palmer, 2002]. It is interesting for us to evaluate how much improvement I can achieve when using LPASs instead of the flat predicate argument structures. Furthermore, I want to identify IE tasks that can be improved by more precise information obtained through my approach.

11 Related Work

Several approaches ([Crysmann et al., 2002], [Frank et al., 2003], [Riezler et al., 2002], [Tsujii, 2000], [Uszkoreit, 2002] and [Xu & Krieger, 2003], etc.) have suggested methods for combining deep and shallow NLP. Some of these aim at adding the robustness of shallow methods to deep processing, others try to add the higher accuracy of deep processing to shallow NLP applications.

In [Copestake, 2003], a new semantic formalism called "Robust MRS" is developed, which allows the mapping of linguistic analysis at different processing levels to a uniform representation. This extreme "underspecification"-oriented semantic formalism can be regarded as an interface language, which eases the integration of various linguistic components with different natural language understanding depths. However, the biggest challenge for this ambitious RMRS approach is to merge RMRS expressions provided by the different components into a well-formed, consistent and reasonable semantic representation. I know that most linguistic components to be integrated were developed independently from each other. The problems for integration can start with the different token boundaries, and extend to incompatible parse tree structures. One of the hard problems is to deal with ambiguity at each level and to choose the right reading for the integration. Moreover, it is an open question whether this approach can reach its original goal after the integration, namely, to obtain a more robust and precise semantic representation. My concern is that the integration of too many sources of uncertainties and underspecified information will increase the degree of uncertainty and underspecification in the results whereas shallow language technology applications owe their success to the brute force reduction of such uncertainty.

In comparison to the above approaches, my work focuses only on the integration of semantic analyses. Therefore it is less ambitious than the RMRS approach. Nevertheless, I consider the chances for obtaining more useful, consistent and precise information as rather good, because predicates are reliably indicators for the integration.

My ideas have been strongly influenced by research in the area of semantic role labelling ([Carreras and Màrquez, 2004]). Several groups are working on statistical approaches to labelling flat predicate argument structures to the constituents in the dependency tree structures or even chunks ([Gildea and Jurafsy, 2002], [Gildea and Palmer, 2002], [Surdeanu et al., 2003] and [Moschitti and Bejan, 2004]). However, all these approaches do not determine and exploit the information about the embedding relationships between two related predicate argument structures and cannot specify the variable bindings when the relationships are implicitly expressed in the surface form. My work can also be viewed as a further development in the semantic role labelling research.

[Fuchss et al., 2004] have attempted to translate MRS descriptions into normal dominance constraints. In addition to the theoretical translation, a practical system is developed, for translation, validation and evaluation. The main result of the evaluation shows that 83% of the Redwoods sentences are *nets*, and 17% aren't. The non-net MRS expressions predict more readings than the sentence actually has. All linguistically correct MRS expressions are indeed nets.

In comparison to the other disambiguation methods for the unification-based grammars (UBGs), which build their probability models directly on top of UBGs, [Kiefer et al., 2002] apply a context-free approximation for a given unification-grammar and use a standard probability model for the context-free grammar.

12 Conclusion

In this report, I have shown some ideas on the automatic generation of linked predicate argument structures (LPASs) by integrating the underspecified semantic expressions of MRS provided by a HPSG parser and ERG grammar with the robust predicate argument structures provided by the SEM system. The linked predicate argument structures describe the embedding relationships between two predicate argument structures and the variable binding, which are missing in the robust predicate argument structures. I developed a component, which automatically extracts LPASs from MRS expressions. In addition, some initial ideas have been proposed to make HPSG grammars more robust in order to deal with real world texts. They encompass lexicon extension with WordNet and disambiguation of parsing results with the help of a statistical parser. As a future perspective, HPSG grammars such as ERG may become useful for providing a more fine-grained semantic annotation to some standard treebanks, e.g., the Penn Treebank. Therefore, a potential further cooperation between the PropBank and the Redwoods Treebank might be promising.

Above all, LPAS can be used as a general semantic interface for integrating different predicate-argument structure-based semantic analyses with various granularities.

Acknowledgement

I am grateful to Sanda Harabagiu for encouraging me to pursue this approach and inviting me to work in her group. I am especially indebted to Alessandro Moschitti for the useful discussions and comments. In particular, I am grateful to Alessandro and Cosmin-Adrian Bejan for the annotated predicate argument structures and richer annotations. Most of the examples in this report are provided by them. Many thanks are due to Hans Uszkoreit for the suggestion of writing down the ideas as a research report and in particular also for providing very useful comments on this report. Special thanks to Valia Kordoni for her fruitful discussions and review of this report.

References

[Collins, 1997] Michael Collins. 1997. Three Generative, Lexicalized Models for Statistical Parsing. In Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics (ACL 1997): 16-23, Madrid, Spain.

[Copestake and Flickinger, 2000] Ann Copestake and Dan Flickinger. 2000. An open-source grammar development environment and broad-coverage English grammar using HPSG. In *Proceedings of the Second conference on Language Resources and Evaluation (LREC-2000)*, Athens, Greece.

[Copestake et al., 1999] Ann Copestake, Dan Flickinger, Ivan A. Sag and Carl J. Pollard. 1999. Minimal Recursion Semantics: An Introduction, 1999.

[Copestake et al., 2004] Ann Copestake, Fabre Lambeau, Benjamin Waldron, Francis Bond, Dan Flickinger and Stephan Oepen. 2004. A Lexicon Module for a Grammar Development Environment. In Proceedings of LREC 2004.

[Crysmann et al., 2002] Berthold Crysmann, Anette Frank, Bernd Kiefer, St. Müller, Günter Neumann, Jakub Piskorski, Ulrich Schäfer, Melanie Siegel, Hans Uszkoreit, Feiyu Xu, Markus Becker and Hans-Ulrich Krieger. 2002. An Integrated Architecture for Deep and Shallow Processing. In *Proceedings of ACL'02*.

[Fellbaum, 1998] C. Fellbaum. 1998. WordNet: An Electronical Lexical Database. MIT Press, Cambridge, MA.

[Fleischman et al., 2003] M. Fleischman, Namhee Kwon and E. Hovy. 2003. Maximum Entropy Models for FrameNet Classification. Empirical Methods in Natural Language Processing, Sapporo, Japan.

[Flickinger, 2002] Dan Flickinger. 2002. On building a more efficient grammar by exploiting types. In Stephan Oepen, Dan Flickinger, Jun'ichi Tsujii and Hans Uszkoreit (eds.) Collaborative Language Engineering, Stanford: CSLI Publications, pp. 1–17.

[Frank et al., 2003]. Anette Frank, Markus Becker, Berthold Crysmann, Bernd Kiefer, Ulrich Schäfer. 2003. "Integrated Shallow and Deep Parsing: TopP meets HPSG". In *Proceedings of the ACL 2003*, Sapporo, Japan.

[Fuchss et al., 2004] Ruth Fuchss, Alexander Koller, Joachim Niehren, and Stefan Thater. 2004. Minimal Recursion Semantics as Dominance Constraints: Translation, Evaluation, and Analysis. In Proceedings of the 42nd ACL, Barcelona.

[Gildea and Palmer, 2002] Daniel Gildea and Martha Palmer. 2002. The necessity of Parsing for Predicate Argument Recognition. In Proceedings of the 40th Meeting of the Association for Computational Linguistics (ACL-2002): 239-246, Philadelphia, PA.

[Gildea and Jurafsy, 2002]. Daniel Gildea and Daniel Jurafsky. 2002. Automatic Labeling of Semantic Roles. In *Computational Linguistics*, 28(3): 245–288.

[Hobbs et al., 1997] Hobbs, Jerry R., Douglas E. Appelt, John Bear, David Israel, Megumi Kameyama, Mark Stickel, and Mabry Tyson. 1997. ``FASTUS: A Cascaded Finite-State Transducer for Extracting Information from Natural-Language Text", in E. Roche and Y. Schabes, eds., Finite State Devices for Natural Language Processing, MIT Press, Cambridge, Massachusetts, pp. 383–406.

[Kiefer et al., 2002] Bernd Kiefer, Hans-Ulrich Krieger und Detelef Prescher. 2002. A novel disambiguation method for unification-based grammars using probabilistic context-free approximations. In Proceedings of the 19th International Conference on Computational Linguistics.

[Kingsbury, Palmer and Marcus, 2002] Paul Kingsbury, Martha Palmer, and Mitch Marcus. 2002. Adding Semantic Annotation to the Penn TreeBank. In Proceedings of the Human Language Technology Conference, San Diego, California, 2002.

[Kingsbury and Palmer, 2002] Paul Kingsbury and Martha Palmer. From Treebank to PropBank. 2002. In Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC-2002), Las Palmas, Spain, 2002.

[Moschitti et al., 2003] Alessandro Moschitti, Paul Morarescu, Sanda M. Harabagiu. 2003. Open Domain Information Extraction via Automatic Semantic Labeling. FLAIRS Conference 2003: 397-401.

[Moschitti and Bejan, 2004] Alessandro Moschitti and Cosmin Adrian Bejan. 2004. A Semantic Kernel for Predicate Argument Classification. In Proceedings of HLT-NAACL 2004 Workshop: Eighth Conference on Computational Natural Language Learning (CoNLL-2004), Boston, Massachusetts, USA, pp. 17–24.

[Oepen et al., 2002] Stephan Oepen, Kristina Toutanova, Stuart Shieber, Christopher Manning, Dan Flickinger and Thorsten Brants. 2002. The LinGO Redwoods Treebank: Motivation and Preliminary Applications. In

Proceedings of the 19th International Conference on Computational Linguistics (COLING 2002), Taipei, Taiwan, pp. 1253–1257, 2002.

[Pollard and Sag, 1994] Pollard, Carl J. and Ivan A. Sag. 1994. *Head-Driven Phrase Structure Grammar*. Chicago: University of Chicago Press, 1994

[Riezler et al., 2002] S. Riezler, T.H. King, R.M. Kaplan, R. Crouch, J.T. Maxwell III, and M. Johnson. 2002. Parsing the Wall Street Journal using a Lexical-Functional Grammar and Discriminative Estimation Techniques. Proceedings of ACL, pp. 271–278, 2002.

[Surdeanu et al., 2003] Mihai Surdeanu, Sanda M. Harabagiu, John Williams and Paul Aarseth. 2003. Using Predicate-Argument Structures for Information Extraction. 2003. In Proceedings of ACL 2003. pp. 8–15.

[Tsujii, 2000] J. Tsujii. 2000. Generic NLP Technologies: Language, Knowledge and Information Extraction. In Proceedings of ACL, pp. 11–18.

[Uszkoreit, 2002] H. Uszkoreit. 2002. New Chances for Deep Linguistic Processing. In Proceedings of COLING 2002, Taipei, 2002.

[Wahlster, 2000] Wahlster, W. (ed.). 2000. Verbmobil: Foundations of Speech-to-Speech Translation. Berlin: Springer, 2000.

[Carreras and Màrquez, 2004] Xavier Carreras and Lluís Màrquez. 2004. Introduction to the CoNLL-2004 Shared Task: Semantic Role Labeling. In Proceedings of Eighth Conference on Computational Natural Language Learning (CoNLL-2004), Boston, USA.

[Xu and Krieger, 2003] Feiyu Xu and Hans-Urlich Krieger. 2003. Integrating Shallow and Deep NLP for Information Extraction. In Proceedings of RANLP 2003, September 2003, Borovets, Bulgaria.

APPENDIX: Comparison between MRSs and flat predicate argument structures, and ERG full sentence parsing performance and statistical parsing performance

Features	**ERG, MRSs** **full sentence parser**	**SEM PredArgs** **Statistical parser**
dominance relation between elementary predictions (arguments and predicate-arguments)	- underspecified - explicit - precise - resolvable	implicit
variable binding	- explicit - precise - specified	not specified
modifiers (adverbs, prepositional modifiers)	- modifier takes a predicate argument structure as its argument, - not as an argument in a relation	as an argument in a relation
robustness	- not robust - no partial results - no output by spelling errors - no output by unknown linguistic constructions	very robust
boundary of the argument	underspecified	specified
modification	underspecified	specified
parsing result	all structural analyses	one
coverage of special linguistic phenomena	general linguistic theory, a big coverage of well-formed linguistic phenomena	
punctuation	does not work well	no problem
matrix clause and subordinate clause relation	explicit	implicit
VP coordination	yes	in case of simple constructions
long sentence	not robust	robust
negation	yes	not treated systematically
passive	yes, but a special treatment	not always correct
control (subject, object)	variable binding is well-done	not well-done

raising	variable binding is well-done	not well-done
relative clause	variable binding is well-done	not well-done
long-distance dependency	variable binding is well-done	not well-done
wh-movement	variable-binding is well-done	not well-done
apposition	cannot treat, e.g., Peter, 61 years old, will come.	yes, recognized, but no internal structure is built up
named entity recognition	only very partially	well-done
PP attachment	not always correct	not always correct, depends on corpus
auxiliary verbs	correct	not always correct, often treated as a predicate e.g. has caused …"has" as a predicate

Linking Flat Predicate Argument Structures

Feiyu Xu

RR-04-04

Research Report