A DAI Approach to Modeling the Transportation Domain

Klaus Fischer, Norbert Kuhn
Deutsches Forschungszentrum für Künstliche Intelligenz
Stuhlsatzenhausweg 3, D-66123 Saarbrücken 11

DFKI-Research Report RR-93-25

Abstract

A central problem in the study of autonomous cooperating systems is that of how to establish mechanisms for controlling the interactions between different parts (which are called agents) of the system. One way to integrate such mechanisms into a *multi-agent system* is to exploit the technique of cooperation or negotiation protocols. In a protocol we distinguish to essential layers: the communication layer specifying the possible flow of messages between different agents, and the decision layer, which controls the selection of a message (speech-act) that the agent sends in a specific situation.

In this report we first give a short introduction of our agent model InteRRap which provides the basis for the modeling of the different scenarios considered in the AKA-Mod project at the DFKI. The techniques we will discuss in the following are located in the plan based component and in the cooperation component of this model. The domain of application is the MARS scenario (Modeling a Multi-Agent Scenario for Shipping Companies) which implements a group of shipping companies whose goal it is to deliver a set of dynamically given orders, satisfying a set of given time and/or cost constraints. The complexity of the orders may exceed the capacities of a single company. Therefore, cooperation between companies is required in order to achieve the goal in a satisfactory way. This domain is of considerable interest for studies with economical background as well as for research projects.

We give a short summary of results from economical studies that are concerned with the real-world situation in Germany in the transportation domain. They show the need for the development of new techniques from the field of computer science to tackle the problems therein. Then, an overview on related research is presented. Two approaches are discussed in more detail: the first one being based on OR-techniques and a second one being based on the concept of partial intelligent agents attempting to integrate techniques from OR and DAI. Both approaches are concerned with the situation in a single company. However, our purpose to handle the case of distributed shipping companies requires additional mechanisms, e.g. to cope with the problems of task allocation and task decomposition in multi-agent systems.

Mechanisms for distributed task decomposition and task allocation processes in multi-agent systems belong to the core of our studies. Therefore, we will first discuss techniques for these problems in a general setting and then describe their implementations in the MARS system. In this description, particular emphasis is placed on the cooperation within a shipping company. Here, one company agent has to allocate a set of orders its truck agents. The truck agents support the company agents by giving cost estimations based on their route planning facility. Thus, this procedure provides the basis for the decisions of the company agents and is discussed in very detail.

Finally, we present results from a series of benchmark tests. The test sets have also been run with OR-based implementations and thus, give us the opportunity to compare our implementation against these approaches.

1 Introduction

1.1 Themes of DAI

Distributed Artificial Intelligence (DAI) is the subfield of AI concerned with concurrency in AI computations. Bond and Gasser [Bond & Gasser 88] divide the world of DAI in two primary arenas: Research in *Distributed Problem Solving* (DPS) investigates how the work of solving a particular problem can be divided among a number of "nodes" or modules that cooperate at the level of dividing and sharing knowledge about the problem and about its solution. The second arena, called *Multi-agent systems* (MAS), deals with coordinating intelligent behaviour among a collection of (possibly pre-existing) autonomous intelligent agents. Emphasis is placed on how these agents coordinate their knowledge, goals, skills and plans jointly to take action or to solve problems. Like modules in a DPS system, agents in a multi-agent system must share knowledge about problems and solutions. However, apart from these issues, they also have to reason about the *process of inter-agent coordination* itself.

For a long time, the problem of agent coordination was dominated by the metaphor of the cooperating expert society, for which Hewitt, in his early ACTORS work, raised the broad research question of " what should be communication mechanisms and conventions of civilized discourse for effective problem solving by a society of experts?" ([Hewitt 77]). The cooperating expert paradigm dominated the research in DAI for more than a decade. In the field of agent architectures it provided the basis for developments like Lenat's "Beings" ([Lenat 75]), Hewitt's ACTORS (cf. e.g. [Hewitt 73]), and for blackboard systems such as HEARSAY ([Ermann et al. 80]) or the DVMT testbed ([Corkill & Lesser 88]).

Since the late eighties, research in Multi-agent systems has paid more attention to particular concepts that are of relevance for the coordination in dynamic agent societies, such as *cooperative planning* ([Lux et al. 92, Jennings 92, Kreifelts & v. Martial 91]), *conflict resolution* [Klein 90, Sycara 88], and *negotiation*, ([Sycara 89, Durfee & Lesser 89, Zlotkin & Rosenschein 91]). The purpose of particular agent models was now to provide a framework for integrating instances of these concepts required to deal with a particular domain of application.

In addition to this, there are quite a few more good reasons to concentrate on the agent architecture in the first place, and to use one of its instantiations in order to describe the actual process of problem solving:

- The architecture provides a valuable general guideline for the methodology of the design and the implementation of an application.
- The modules of the agent model precisely structure the classes of operational knowledge.
- The execution model which is implicit to the architecture avoids programming from scratch.
- Application-independent, predefined mechanisms such as negotiation protocols (e.g. the Contract Net) are directly available.
- The emergent functionality of the society can be predicted up to a certain level by regarding the basic patterns of interaction of the instantiated agents.

- An agent architecture provides a basis for the investigation of special strategies and of extensions of the modules.

In the following we briefly describe the INTERRAP Agent Model which provides the basis for the modelings of the application we are concerned with in the project AKA-Mod at the DFKI. It turns out that this agent model is particularly suitable in agent domains that show a dynamic behaviour, i.e. where the environment of the individual agent is constantly changing because other agents enter or leave the system or because the "physical" setting of the environment changes.

The INTERRAP Agent Model

The agent model INTERRAPis essentially an extension of the RATMAN model [Bürckert & Müller 91]. INTERRAP was developed in order to meet the requirements of modeling dynamic agent societies. A basic feature is that it provides means to combine reactive behaviour of an agents with explicit planning facilities. For the reactive part, *Patterns of behaviour* allow an agent to react quickly and flexibly to changes in its environment. On the other hand, the ability to devise *plans* is generally regarded necessary to solve more sophisticated tasks.

So far, INTERRAP has been evaluated using three applications: (1) the implementation of a society of cooperating vehicles in a loading-dock [Müller & Pischel 93a], (2) the MARS system, a simulation of cooperating transportation companies [Kuhn et al. 93a], and (3) COSMA, a distributed appointment scheduling manager [Schupeta 92].

The INTERRAP Architecture

While the novel feature of RATMAN - the idea of structuring a knowledge base according to the complexity of the knowledge contained - was commonly accepted, there was one main point of criticism of the model, namely the lacking separation between aspects of the *knowledge* used in the agents and the *functionality* shown by the model: the hierarchically structured levels of knowledge were not only constructed using the concepts of the lower levels, but they were also used to trigger activities at these lower levels.

INTERRAP clearly draws the separation between the pure knowledge base and the functional part, while preserving the hierarchical structure of the model. Thus, the two parts of the INTERRAP model are

- the hierarchical agent knowledge base, and
- the multi-stage control unit.

Figure 1 shows the INTERRAP model in more detail.

1.1.1 The Agent Knowledge Base

The lowest level of the Agent KB contains the world model of the agent. It is organized as a taxonomical knowledge base. This kind of knowledge represents the objects in the world and the relationships which hold among these objects (which corresponds to the standard T-Box/A-Box structures). The second level describes the patterns of behaviour

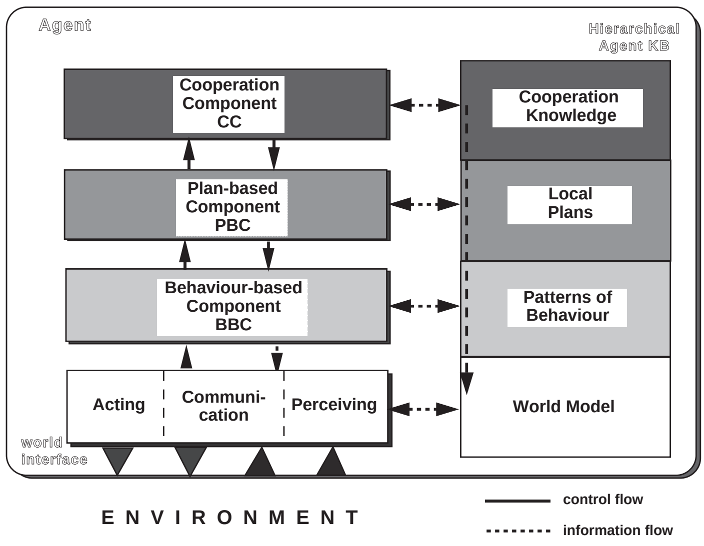


Figure 1: The INTERRAP Agent Model

and the basic actions an agent can perform. A plan library, given as a set of skeletal plans, is modeled at the third level. The plans are defined recursively starting from basic actions, patterns of behaviour, or uninstantiated subplans. Finally, knowledge about cooperation and coordination, such as communication and negotiation protocols, and *joint plans* (which are basically multi-agent plans) is represented at the highest layer of the hierarchy.

1.1.2 The Control Unit

The agent control reflects the hierarchical structure of the knowledge base. It shows the operational flow as discussed in the RATMAN model [Bürckert & Müller 91], from the world interface level, where sensoric data is perceived, up to the behaviour-based level, to the plan-based and cooperation levels, and back again to the interface level, where finally actions in the world are performed. On the other hand, it was built according to the idea of combining the rational, plan-based paradigm with the concept of behaviour-based, reactive systems and situated actions [Brooks 86, Suchman 87, Steels 90]. The four components of the INTERRAP agent control as shown in figure 1 are the world interface, the behaviour-based component (BBC), the plan-based component (PBC), and the cooperation component (CC).

Instead of discussing the single levels one by one (see [Müller & Pischel 93a] for a detailed description), at this stage, the flow of control and information through the different stages will be outlined. The lowest level reflects the input/output interface of the agent, the perception of changes in the world, and the receiving of messages. This information passes a first filter and flows into the world model of the agent. It is the basic information used by the BBC. There, it may either directly trigger a certain pattern of behaviour (e.g. the pattern "avoid collision" which has the agent moving aside)[1].

[1]Note that, since patterns of behaviour may be concurrently active, the BBC needs a hard-wired control mechanism for coordinating these patterns. This must not be confounded with the deliberate

If there is no need for such a fast response, or if the situation is too complex to be coped with by the BBC, control is shifted up to the plan-based component. This component contains the agent's facilities for planning and local decision-making. If the actual situation requires cooperation and coordination with other agents (such as resolving a blocking conflict between two forklift agents in a narrow shelf), the PBC passes control to the CC, where a cooperative solution of the problem is worked out (for example a joint plan for resolving the conflict). In any case, the order for the next working step is passed to the next lower level:

- a joint plan is transformed into a set of single-agent plans together with a set of synchronization commands (representing the constraints among the plans) and passed to the PBC.
- a pattern of behaviour is activated by the PBC.
- the performance of basic actions or the sending of messages via the world interface is activated by the BBC.

Finally, note that each component of the agent control has access to its corresponding layer in the agent knowledge base *and* to all lower layers, but not to higher layers. For example, a pattern of behaviour has never access to the representation of plans.

So far, we have given a brief overview of the INTERRAP agent model which underlies our applications. In the report at hand we pay particular attention to the planning capabilities of the agents that are part of our multi-agent system MARS for modeling the Transportation Domain. The activities of these agents are controlled mainly by the Plan-based Component PBC and the Cooperation Component CC.

2 The MARS Multi-agent Scenario

2.1 The Transportation Domain

In a time of constantly growing world-wide economical transparency and interdependency, logistics and the planning of freight transports get more and more important both for economical and ecological reasons. For Germany, until the year 2000 an increase of transport traffic of about 60% compared to that of 1992 is forecasted.

One reason for this dramatic development of traffic load is the usual annuary growth of economy. But, additionally this development is driven by the political changes in Europe, such as the new Common Market of the countries of the European Community, which brings about a process of deregulation to the transportation companies. Like this has already happened in the USA, deregulation, which means for example the ability of the shipping companies to calculate free shipping rates will bring about new structures for this branch of industry. The idea of the Common Market itself will lead to more orders of small amount and will increase the portion of collective consignment in the global transportation process. A second political event which reinforces the effects mentioned before is the integration of the Eastern European countries into the economical process.

mechanisms for decision-making located at the plan-based and the cooperation layer.

Due to the geographical position this will have a particular impact for Germany, which is now lying in the "middle of Europe".

For the situation of today, we know from statistical studies that currently about 82% of the transportation orders are delivered by trucks (cf. [Blamauer 83]). This means, that if the forecasts for the future development turn out to be true and if there will be no essential change in the frame conditions of the transportation domain this will end up with a complete collapse of traffic.
Rittman ([Rittmann 91]) states that more than one third of the trucks in the streets of Europe are driving without carriage, since they are on their way to pick up goods or on their way back home. This also shows that the actual situation of planning in the transportation domain is far from being satisfactory.

On the one hand this is due to the role of the transportation process in the global industrial and economical chain: producers and consumers both pursue the idea of utilizing the roads (and the trucks) as depots. This strategy has effects like

- decreasing volumes of the orders
- higher frequency of delivery
- higher cost of transport, and in the end
- higher load of road traffic

An essential improvement to this dilemma can be the *exploitation of cooperation* between different members of this domain. In particular (and that is what we are focusing on in our project), this includes the cooperation of different service logistics companies to raise the load of the individual company.

To check these statements and to get some feeling for the real-world cooperation mechanisms between shipping companies we contacted a medium size shipping company (which runs about 700 trucks) located in Saarbrücken. There we learned that in the past, the cooperative approach has already proven to be of great promise, when autonomous companies had founded joint organisations to coordinate their transportation activities. The reference company for our project is also participating in such an organisation, called UNITRANS, which consisted of 39 different shipping companies in the beginning. In the interviews with our partner we figured out to major types of cooperation between different companies which motivated the cooperation mechanisms we have implemented in our MARS system: Namely, the *unbooked leg cooperation* and the *cooperation for coupling long-distance transportation and local distribution.*

Another aspect of the situation within a shipping company is the lack of disposition support by computer systems in most of the companies. A technical reason for this might be the complexity of the disposition task, where even simplified and quite abstracted instances of the problem can be proven to be NP-hard. Examples of this kind can be found e.g. in [Bachem et al. 92] for the depot delivery problem or in section 2.4 of the report at hand for the problem of the routing of only one truck.
Moreover, not only since just-in-time production has come up, planning must be performed under a high degree of uncertainty and incompleteness, and it is highly dynamic. Furthermore, the dispatchers usually have to deal with an open-ended, real-time problem where a large number of constraints is associated with each order. Thus, e.g. Standard

Operations Research approaches (see [Bodin 83, Müller-Merbach 70] for an overview) cannot cope with the dynamics of this domain.

2.2 The (D)AI Aspects

Why is it adequate to use AI techniques and more specifically DAI approaches to tackle the transportation problems described above? One reason is the complexity of the scheduling problem, which makes it very attractive for AI research[2]. However, there are also more pragmatic reasons: *Commonsense knowledge* (e.g. taxonomical, topological, temporal, or expert knowledge) is necessary to solve the scheduling problems effectively. *Local knowledge about the capabilities* of the transportation company as well as knowledge about competitive (and maybe cooperative) companies massively influences the solutions. Moreover, since a global view is impossible (because of the complexity), there is a need to operate from a local point of view and thus to deal with *incomplete knowledge* with all its consequences.

The last aspect leads to the DAI arguments:

1. The domain is inherently distributed. Hence, it is very natural to look at it as a multi-agent system. However, instead of tackling the problem from the point of view of the entities which are to be modeled and then relying on the emergence of the global solution, the classical approach to the problem is an (artificially) centralized one.

2. The task of centrally maintaining and processing the knowledge about the shipping companies, their vehicles, and behaviour is very complex. Moreover, knowledge is often not even centrally available (real-life companies are not willing to share *all* their local information with other companies). Therefore, modeling the companies as independent and autonomous units seems the only acceptable way to proceed.

3. In real business, companies usually solve capacity problems by contacting partners that might be able to perform the problematic tasks. Then, the parties negotiate the contract. However, task allocation, contracting, negotiating and performing joint actions are main topics in DAI research.

2.3 The Scenario

2.3.1 General Description

The MARS scenario (Modeling a Multi-Agent Scenario for Shipping Companies) implements a group of shipping companies whose goal it is to deliver a set of dynamically given orders, satisfying a set of given time and/or cost constraints The complexity of the orders may exceed the capacities of a single company. Therefore, cooperation between companies is required in order to achieve the goal in a satisfactory way. This general setting can be seen in figure 2.

[2]At this year's International Conference on AI and Applications (CAIA'93), seven out of sixty-one papers dealt with scheduling problems!

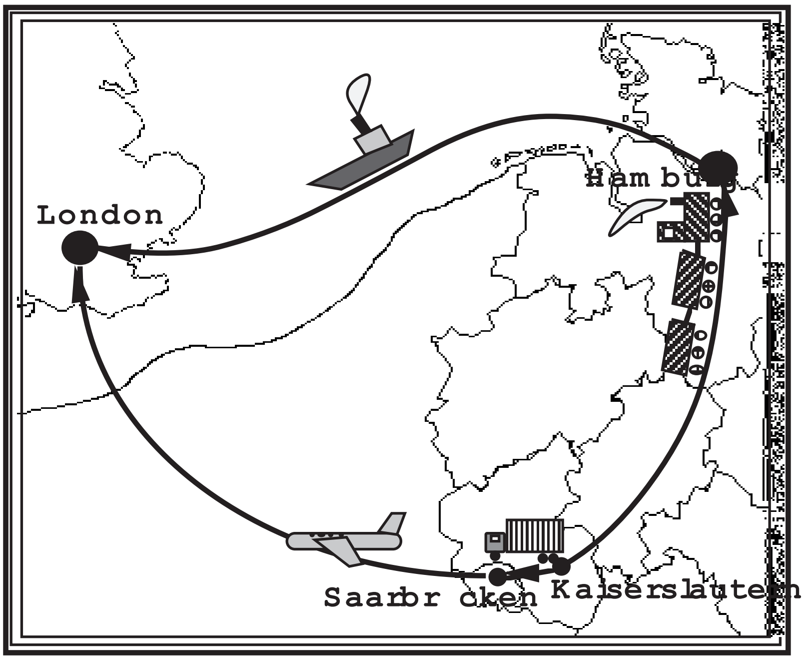


Figure 2: The Domain of Distributed Shipping Companies

The common use of shared resources, e.g. train or ship, requires coordination between the companies. Although each company has a local, primarily self-interested view, cooperation between the shipping companies is necessary in order to achieve reasonable global plans (see section 9).

2.3.2 The Agent Society

Apart from internal *system agents*, which perform tasks such as the representation and visualization of the simulation world, the MARS agent society consists of two sorts of *domain agents*, which correspond to the logical entities in the domain: *shipping companies* and *trucks*. The general architecture used to model a single company is sketched in figure 3.
In contrast to other approaches (e.g. Falk et al., cf. section 3) we decided to look upon trucks as agents. This view allows to delegate problem-solving skills to them (such as route-planning and local plan optimization). Furthermore, we obtain a logical distribution of the system even if we consider only a single company. Communication within this system may only occur between agents who are connected by an arc in figure 3. It is enabled by direct communication channels between them.

What should also become clear from this figure is the hierarchical relationship between the different agents of the scenario: There is a *Master-Slave* relationship between the shipping company agents and their truck agents and a *peer to peer* relationship between different company agents. According to this hierarchical relationships we define different modes of cooperation between the agents in section 4.

From a functional point of view the different types of agents play different roles in the transportation scenario:

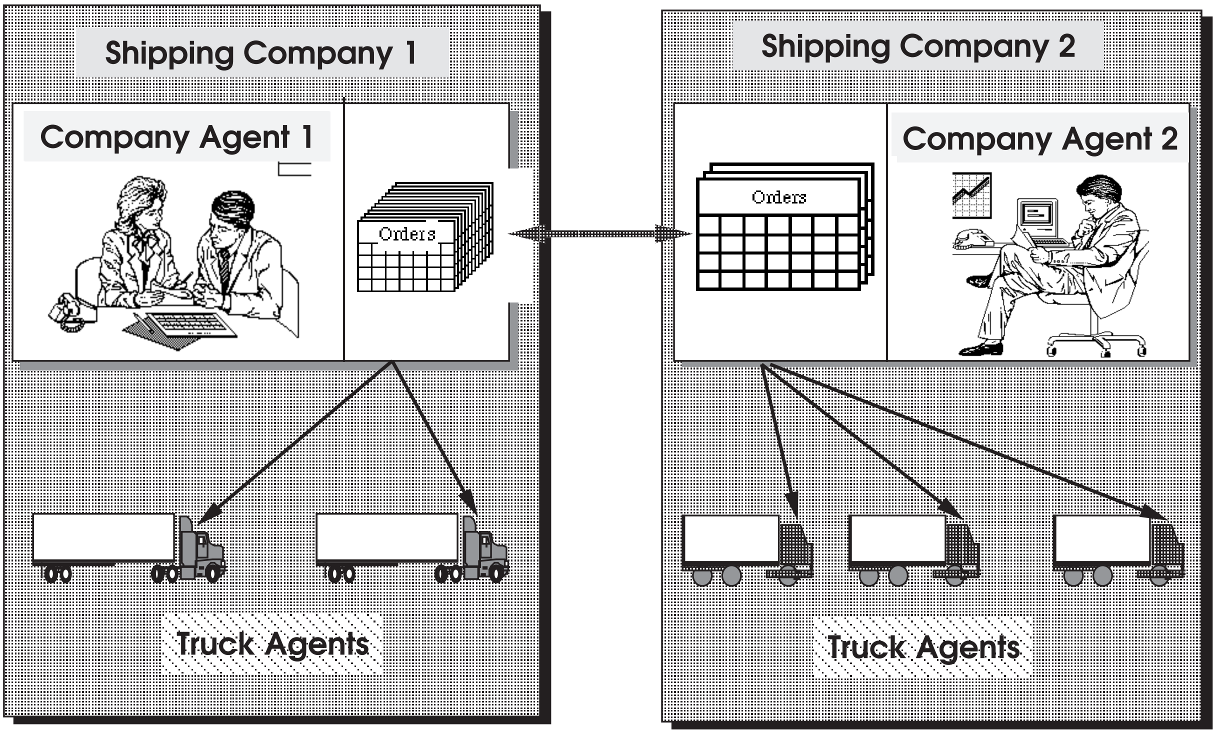


Figure 3: Modeling of Shipping Companies as a Multi-Agent System

The *company* agent is responsible for the disposition of the orders that have been confided to him. Thus, it has to allocate the orders to its trucks, while trying to satisfy the constraints provided by the user as well as local optimality criteria. The shipping companies can be regarded as experts for cooperation and cooperative problem solving. They are equipped with additional global knowledge which is needed for cooperating successfully with other companies.

The *truck* agents represent the means of transport of a transportation company. Each truck agent is associated with a particular shipping company from which it receives orders of the form *"Load a goods g_1 at location l_1 and transport it to location l_2"*. Given such an order, the truck agent does the planning of the route ([Kuhn et al. 93a], see also section 8) according to its geographical knowledge and it will inform the shipping company agent about the delivery of the goods. Furthermore, it is able to support the shipping company during the disposition phase: The truck reports remaining capacities, planned routes and it is able to estimate the effort (and the effects)[3] that are caused by an order.

2.4 Analysis of the Problem

The task of delivering several orders is basically a scheduling problem. What makes it even harder is the two-dimensionality of task decomposition resulting from the special domain. First, the goods to be transported can be distributed to several means of transport, as this could be e.g. trucks, trains, ships, or planes. In the following, we do not consider this

[3]i.e. cost, time, security of transport, ...

terminological distinction. Instead, we map this difference of means of transport to the different capacities of *trucks*. The second dimension consists of finding the route between two cities on a road map which can be splitted up into sub-routes that can be taken at different times using different conveyances.
Routing and Scheduling Prolems similar to that of our group have been considered since a long time ago. In different settings some of them turn out to be (at least) NP-hard, some of them can be shown to be solvable in polynomial time. In this section we show that the routing and the scheduling task described by the informal notion above can be shown to be NP-hard.

Analysis of Complexity

If we do not consider precedence relations between orders or time windows for the problem described informally above we have to deal with a routing problem which can be defined as follows:

Definition *The Routing Decision Problem RDP*

> **INSTANCE**:
> Graph $G = (V, E)$,
> Length $l(e) \in Z_0^+$ for each $e \in E$,
> Set of orders O = { o_i=(s_i,t_i,w_i) | i = 1, ..., m, with $s_i \in E$ being the starting point of o_i, $t_i \in E$ being the target point of o_i, and $w_i \in Z^+$ giving the weight (or volume) of o_i },
> Trucks $T = \{t_1, \ldots, t_m\}$,
> Function *Capacity*: $T \to Z^+$ giving the capapcity of each truck, and
> Bound $B \in Z^+$.
>
> **QUESTION**:
> Is there a disposition function D: $O \to T$ and a routing of the trucks $t_i \in T$, such that all orders are delivered and that the sum of the length of the route of the trucks is at most B?

This definition reflects exactly the two-dimensionality of the problem as mentioned above: Firstly, the orders have to be allocated to the trucks and secondly, a route for each truck has to be constructed. If we consider the routing problem for a set of distributed shipping companies we have to find in addition an allocation of the orders to the different shipping companies S, i.e. a mapping $D : O \to (S \to T)$. However, as we will see this does not contribute to the general complexity of the task. So, for reasons of simplicity we restrict ourselves in this subsection to complexity considerations for the RDP.

We will prove the *Routing Decision Problem* (RDP) to be NP-complete in the following. Here, NP denotes the class of languages that can be recognized by a nondeterministic Turing Machine in polynomial time (cf. e.g., [Hopcroft & Ullman 79]). The proof is done in two steps. First, a polynomial time reduction of the *Modified Rural Postman Problem* which is known to be NP-complete (cf. [Garey & Johnson 79]) to the RDP is given. This shows that the Routing Optimization Problem is at least NP-hard. Then, a nondeterministic polynomial-time algorithm for RDP is provided, showing that RDP itself belongs to NP.

The *Modified Rural Postman Problem* (MRPP) originates from the *Rural Postman Problem* (RPP), which is defined as follows:

Definition *The Rural Postman Problem*

> **INSTANCE**: Graph $G = (V, E)$, Length $l(e) \in Z_0^+$ for each $e \in E$, Subset $E' \subseteq E$, and Bound $B \in Z^+$.
>
> **QUESTION**: Is there a circuit in G that includes each edge in E' and that has total length of at most B?

The *Rural Postman Problem* remains NP-complete even if $l(e) = 1$ for all $e \in E$. This instance of the problem will be called the *Modified Rural Postman Problem* (MRPP) in the following.

The first lemma states that the MRPP can be mapped in polynomial time to the routing decision problem where only one truck t is available and all orders cover exactly the capacity of t. This shows that MRPP is polynomial-time reducible to RPP.

Lemma 1: MRPP $\propto$ RDP

Proof: Consider an instance i=(G,E',B) of the MRPP, with $G = (V, E)$, $E' \subseteq E$, and $B \in Z^+$.

To i, construct an instance i_1=(G_1,l_1,T_1,$Capacity_1$,O_1,B_1) of the RDP as follows:

$G_1 := G$, i.e. $V_1 := V$, $E_1 := E$
$l_1(e) := 1$ for all $e \in E_1$
$T_1 := t$
$Capacity(t) := c_0$
$O_1 := \{ (e,c_0) \mid e \in E'$, and $c_0 := Capacity(t) \}$
$B_1 := B$

Claim: MRPP(i) iff RDP(i_1)

i) If MRPP(i) holds then there exists a circuit c such that c contains all edges $e \in E'$, and such that the length of c is at most B.
Circuit c is also a suitable route for truck t to deliver all orders $o_i \in O_1$. Since the lengths of the edges in G_1 and G are identical, $l_1(c) \leq B_1$.
Thus, RDP(i_1) holds.

ii) Now, assume that RDP(i_1) holds.
The problem to prove MRPP(i) from RDP(i_1) is that to deliver an order o=(e,c_0) the truck need not necessarily go through edge e.

However, suppose that there exists a route c (described by a sequence of edges) for t which satisfies $l_1(c) \leq B$, and there exists an order o=(v_1,v_2,c_0) with e = (v_1, v_2) $\notin$ c.
Because t delivers all orders, once it will visit v_1 to pick up the goods of o_1. Then, it will run through different vertices v_0' ... v_k' eventually arriving at v_2 to deliver o_2.
Since, c_0 = *Capacity*(t), there did not occur a loading or unloading activity on the route between v_1 and v_2. Thus, we can transform the route c to route c' by substituting the vertex v_2 for the vertices v_0' ... v_k' in c. It follows that the resulting circuit c' is another valid route to deliver all orders in O_1.
Since, $l_1(e) = 1$ for all e $\in E_1$, $l_1(c') \leq l_1(c) \leq B$.
An inductive argument shows that by this procedure we can construct a circuit c in G which eventually contains each edge in E' and which satisfies $l(c) \leq B$.
This proves that MRPP(i) holds.

Obviously, the transformation of i to i_1 can be done in polynomial time. Thus, MRPP is polynomial-time reducible to MSDP. □

What is impicitly assumed in the construction of the proof of Lemma 1 is that the trucks must not use intermediate storages during the delivery of an order. That is, they are not allowed to pick up a specific order, transport it to some depot where it is dropped and another order (or a set thereof) is delivered. After that the truck returns to the depot picks up its origial order and finally, delivers it. This procedure may yield a considerable improvement to routes. However, it is also clear that it brings about a more complicated task of planning a route for a truck. So we neglect the possibility of using intermediate storages for the delivery process. This process is also underlying the algorithm used in the proof of the following lemma.

Lemma 2: RDP belongs to NP

Proof: A nondeterministic algorithm for solving RDP can be specified as follows:

INPUT:
Graph G=(V,E)
Length $l(e) \in Z^+$ for each e $\in$ E
Set of orders O = $\{(s_i,t_i,w_i) \mid 1 \leq i \leq n\}$
Set of Trucks T = $\{t_1, \ldots, t_k\}$
Function Capacity: T $\rightarrow Z^+$
Bound B $\in Z^+$

OUTPUT: RDP (G,O,T,Capacity,B)

begin
Choose - an ordering π of $\{s_i,t_i \mid (s_i,t_i,w_i) \in O\}$ and
- a mapping D: $O \rightarrow \{1 \ldots k\}$
For each truck $t_j \in$ T **do**
begin
$O_j := \{ o \in O \mid D(o) = j\}$

If $\pi(O_j)$ describes a suitable tour for t_j **then** c_j:=tour_length($\pi(O_j)$)
else c_j:= B+1

end
C := $\sum_{j=1}^{k} c_j$
output(C $\leq$ B)

end

The sequence $\pi(O_j)$ describes a suitable route for truck t_j, if the following conditions hold:

1. the pickup of an order must occur before its delivery, i.e. $\pi(s_i,t_i)$

2. the capacity constraints for truck t_i are satisfied, i.e.
$\forall$ i: ($w_i + \sum_{j:\pi(s_i,s_j) and \pi(s_j,t_i)} w_j \leq$ Capacity(t_i)).
In particular, this means that each single order dedicated to a truck must not exceed its capacity. Furthermore, if it transports goods for several orders at the same time all must fit into its loading space.

The evaluation of these conditions can obviously be computed in polynomial time. To determine the tour_length the minimal distances between two successive locations in a circuit π(O) have to be summed up, which can also be done in polynomial time.
Thus, the specification above gives a nondeterministic polynomial time algorithm for the RDP. $\square$

Lemma 1 and Lemma 2 provide the following result:

THEOREM: RDP is NP-complete.

In the proof of Lemma 1 we constructed an instance i_1 of the RPP to an instance i of the MRPP such that MRPP(i) iff RPP(i_1), where in i_1 only one truck was used. Together with Lemma 2 this gives the following statement.

Corollary 1: The RDP remains NP-complete even if $|T| = 1$ and $l(e) = 1$ for all $e \in E$. $\square$

For the more general problem of routing for a set of distributed shipping companies it can be shown analogously that the corresponding decision problem is NP-complete. Restricting this problem to considering one company shows that the reduction of the MRPP can be done in analogy to Lemma 1. Extending the algorithm used in the proof of Lemma 2 to choose in addition a distribution of the orders over the companies gives a nondeterministic algorithm to decide the routing decision problem for distributed shipping companies.

In analogy to the definition of the *Routing Decision Problem RDP* we may define the *Scheduling Decision Problem SDP* where each order is associated with a time window determining temporal constraints for its delivery instead of being associated with a weight as in the RDP, and where each edge in E is labeled by the time needed to travel along this line. The corresponding decision problem is stated by the question whether there exists a disposition function D and a schedule for the allocated orders by each truck such that the orders can be delivered within a certain time bound B.

Obviously the RDP can be easily reduced to the SDP. Furthermore, the algorithm used in the proof of Lemma 2 can be modified to solve the SDP, thus showing that SDP belongs to NP, and hereby, SDP is NP-complete.
The same argument shows that also the combination of both problems, namely the *Routing and Scheduling Decision Problem* is NP-complete. We conclude this section with the following corollary which summarizes these statements.

Corollary: Both, the *Scheduling Decision Problem* (SDP) and the *Routing and Scheduling Decision Problem* (RSDP) are NP-complete. □

3 Related Work

The problem of delivering a set of orders as it has been stated in this report is often regarded as a scheduling, a routing task, or a combination of both. These are the categories into which this kind of problems is usually classified (cf., e.g. [Bodin 83]).

The difference between routing and scheduling tasks is that routing problems have no restriction on delivery time nor are there precedence relationships between stops. Hence, routing problems focus exclusively on the spatial or geometrical aspects of the problem. On the other hand, scheduling focuses exclusively on the time constraints of the problem. Routing and scheduling problems incorporate both spatial and temporal characteristics.

By the problem statement of section 2.1, which is characterized by the fact that orders may enter the system at any time an open scheduling or routing problem is defined where the exact instance is not known in advance. Usually, these problems are called the *Dynamic Vehicle Routing Problem* (DVRP) or the *Dynamic Vehicle Scheduling Problem* (DVSP), respectively.

Compared to the large number of papers dealing with static scheduling or routing problems the dynamic problem instance has found considerably less interest. One site where particular attention has been to it is the "Center for Transportation Studies" at MIT where solutions to this problem have been developed since the mid seventies. An overview of several of these approaches is contained e.g. in [Bodin 83].

Most of the approaches presented there rely on applying OR-based methods. However, it turns out that there are problems with these approaches when the number of constraints to deal with grows or when real-time response of the system is required. This is the case if one wants to support with such a system a dispatcher who has to tell customers an estimated cost of an order at the phone. For this class of problems often knowledge-based approaches are used as e.g., by Bagchi and Nag. ([Bagchi & Nag 91]). They deal with the problem that a vehicle scheduler at a centralized facility receives requests from all over the country for truck capacities on specific dates and times. The scheduler has to assign these loads originating from various parts of the countries to trucks obtained from contract carriers. Bagchi and Nag solve this problem using a heuristic based on the cognitive rules of an experienced scheduler. Although their approach is a centralized one there are some close similarities to the approach that we are pursuing and which will be described in this report. Based on a study of the concepts of a human scheduler Bagchi and Nag have derived a set of rules which are used to build up a plan incrementally and to do some repairing if necessary. To implement these rules and to develop their dynamic load scheduling system EXLOAD they decided to use a rule-based expert system shell. Within their system, global optimisation is reduced to assigning a new shipment to a contract with minimal incremental cost caused by that insertion. This is based on a result of Psaraftis ([Psaraftis 88]) who shows that in a dynamic scheduling environment global minimization over a period of time is best achieved by minimizing the incremental cost of each assignment.

The system EXLOAD has been implemented on IBM 80386 personal computers. This means that it can be used within almost any shipping company as a Disposition Support System. However, Bagchi and Nag's approach covers only the situation for one dispatcher in one shipping company. So, we believe that a multi-agent approach used to combine an inter-company view with their solution to model the intra-company situation could essentially extend this approach.

In the rest of this section we will sketch two other approaches that describe a distributed solution to the vehicle routing problem and which are both based on heuristics using the exchange of orders between agents to achieve an overall good solution.

An OR-based distributed approach to modeling the transportation domain

In [Bachem et al. 92] a parallel improvement heuristic for solving vehicle routing problems with side constraints is presented. In their approach Bachem, Hofstättler and Malich deal with the problem that n customers order different amounts of goods which are located at a central depot. The task of the dispatcher is to cluster the orders and to attach the different clusters to trucks which then in turn determine a tour to deliver the cluster allocated to them. This series of steps is shown by figure 4.

The solution to this problem is constructed using a procedure called *Simulated Trading* procedure. It starts with a set of feasible tours T_1, ..., T_{t_0} which may e.g., be obtained by a conventional heuristic which is applicable to this domain. The tours are represented as an ordered list of costumers that have to be visited. Parallelism is achieved by that the data of each tour T_i can be assigned to a single processor i (the tour manager) of a parallel (Multiple Instruction Multiple Data, MIMD) computer. To guide the improvement of the initial solution, an additional processor, the stock manager is added to the system. The task of the stock manager is to coordinate the exchange of costumers orders between the different processors. To do this, it collects offers for buying and selling orders coming from the processors in the system. This architecture is shown by figure 5. To provide a criteria to the stock manager which exchanges of orders could be the best ones a price system is introduced: If processor p sells an order i (i.e., an order from the depot to customer i), its cost should decrease. This saving of costs is associated as the price Pr to i i.e.,

$$\begin{aligned} \mathrm{Pr} &:= \mathrm{cost}(T_p) - \mathrm{cost}(T_p \ominus \{i\}) \\ T_p &:= T_p \ominus \{i\} \end{aligned}$$

Here, the term $T_p \ominus \{i\}$ denotes the tour that evolves from T_p if customer i (or order i, respectively) is deleted from processor p's tour list. Accordingly, the price Pr for processor p buying a customer i is computed as the difference of costs for the old tour T_p and the costs for the new tour $T_p \oplus \{i\}$, which evolves from the insertion of costumer i in T_p, i.e.

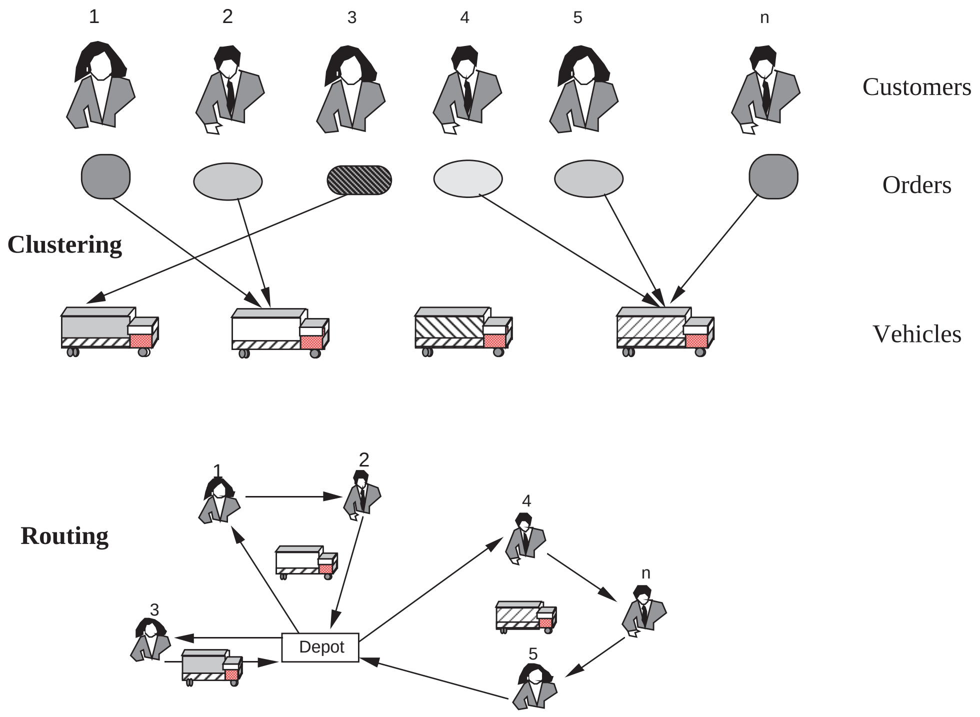


Figure 4: The Standard Vehicle Routing Problem of Bachem et al.

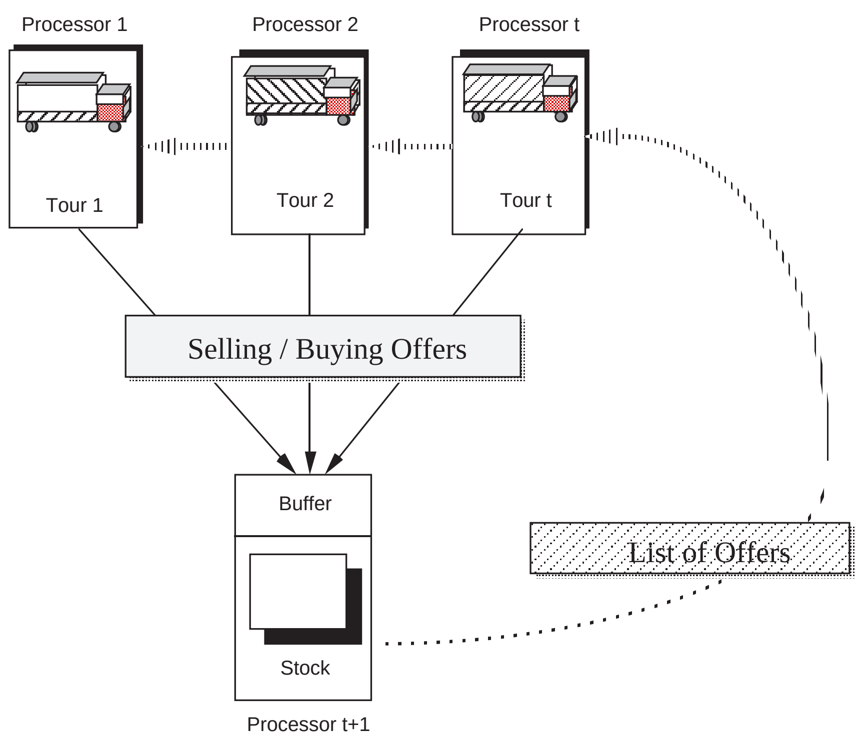


Figure 5: Stock Exchange of Orders in the Simulated Trading Procedure

$\text{Pr} := \text{cost}(\text{T}_p \oplus \{\text{i}\})$ - $\text{cost}(\text{T}_p)$
$\text{T}_p := \text{T}_p \oplus \{\text{i}\}$

The exchange of orders is synchronized by the stock manager according to levels of exchange situations. At each level it asks each processor for a selling or buying order. Having done this, it updates a list of the offers and sends it to all tour managers. Each offer is associated with a quintuple (processor, Level, Selling or Buying, Costumer, Price).

The stock manager maintains a data structure, called *Trading Graph* whose nodes are the selling and buying offers of the processors. Furthermore, there exists an edge between vertices v_i = (processor$_1$, l_i, Selling, c_i, Pr_i) and v_j = (processor$_2$, l_j, Buying, c_i, Pr_j) if processor$_2$ wants to buy customer c_i from processor$_1$. The edge is weighted (or labeled) by the difference of the prices Pr_j - Pr_i, giving the global saving of an exchange of the order between these tours. In this graph the stock manager now looks for a so-called *trading matching* i.e., a subset *M* of the nodes specifying admissible exchanges of orders between tours.

One problem here is, that with offering a selling of an order a processor believes that this order eventually will be bought by another processor, and it will base its future price calculations on its reduced tour. Thus, an admissible exchange must ensure, that with each node $v_i \in M$, all nodes of the processors selling or buying v_i and which have a smaller level than v_i have to be also in *M*.

The *gain* of the matching is obtained by summing up the weights of the edges between nodes in *M*. A *trading matching* is then defined to be an admissible matching whose gain is positive.

Starting from this basic setting, Bachem et al. have implemented some variants of this approach. In one of these variants they tried for example, to reduce the number of message between the tour managers and the stock managers or to reduce the stock manager's role as a bottleneck in this procedure. Furthermore, they have tested their implementations using test sets provided by Solomon ([Desrochers et al. 92]). We have adopted these test sets and we will describe the evaluations of the MARS system for these test sets in section 9.

Apart from this comparison on the run-time level of the two approaches we would like to make some remarks concerning the conceptual differences between this approach and ours. What should have become clear from the above description is, that the solution of Bachem et al. to the Vehicle Routing Problem is primarily tailored to deal with static problems, i.e., the set of orders remains constant during the execution of the simulated trading procedure. Furthermore, because the processors rely on that the orders the offered for selling will eventually be bought by another processor, there will be time periods without the system having a valid plan for the delivery of the orders. In our approach, which is based on negotiation protocols as described in section 4 there exists a valid tour plan at any given time. Thus, we think that this meets better the requirements of interleaving the planning and the execution phase in the vehicle routing problem.

Another difference is due to the different domains we are paying attention to. While Bachem et al. consider route planning and scheduling in one company, where the information in principle may be visible to all members of the scenario, the application domain that we have in mind is a more general one. Between distributed shipping companies there are always competitive relationships implying that the companies try to hide as much information from another as possible. Negotiation protocols, which provide a way of a structured exchange of information between different companies are therefor an adequate means to enhance companies with cooperation mechanisms which at the same time allows them to maintain their autonomy and privacy.

A Combination of OR-based methods and Multi-Agent Systems

OR-based approaches have been exploited successfully to solve static instances of the Vehicle Routing Problem. However, in order to be used in a dynamic environment these methods have to be enhanced with mechanisms providing a real-time behaviour of the corresponding algorithms. Furthermore, usually OR-based methods are difficult to use if the number of constraints is high (cf. [Golden & Assad 83, Psaraftis 88, Bagchi & Nag 91]).

Falk, Spieck, and Mertens (cf. [Falk et al. 93]) pursue an approach based on the integration of knowledge-based mechanisms and OR algorithms. This combination of two methodologies is expressed by the term *Partial Intelligent Agents* (PIAs) they use to denote components of distributed, cooperating systems having a hybrid structure, i.e. modules that include a "conventional" (usually OR-based) and a knowledge-based part.

In the context of the transportation domain they consider tramp agent companies, i.e. shipping companies that are purely concerned with transportation tasks. These companies usually operate from different regional agencies which are autonomous in principle. In the modeling of Falk et al. each agency is represented by a *dispatching PIA* which is responsible for the allocation of the orders of its agency to the trucks belonging to it momentarily. The dispatcher knows the current location of its trucks and it bases its decision on this knowledge. Its objective function considers

- maximizing the utilization of the trucks' capacity
- minimizing the idle time and rides without carriage
- minimizing the length of the route for a single order

Besides that, different restrictions to the solutions, like time constraints formulated by the clients have to be considered. Of course, the goals for the objective function are partially conflicting. Therefore, in a concrete situation it must be possible to specify which goal has to be ranked highest.

The responsibility of a particular dispatching PIA is dedicated to a specific geographical region. When a truck passes from one region to another one the responsibility for the planning of its route carries over to the dispatcher of the current region.

In general, each dispatcher which tries to allocate an order considers only those trucks he is currently responsible for. But, there are situations for which the allocation can be improved essentially by exploiting the cooperation between different PIAs. Such a situation is shown by figure 6.

The process of cooperative planning for a new order is handled as follows: One PIA, namely the one which is responsible for the region where the starting point of the respective order lies in, is chosen to control the allocation to a particular truck, i.e. it is becoming the *Coordinator-PIA* (C-PIA). In the cooperative process all PIAs take part which are responsible for a truck within a certain radius around the starting point or the target point of the new order and are thus becoming *Partner PIAs* (P-PIAs) in this coordination process.

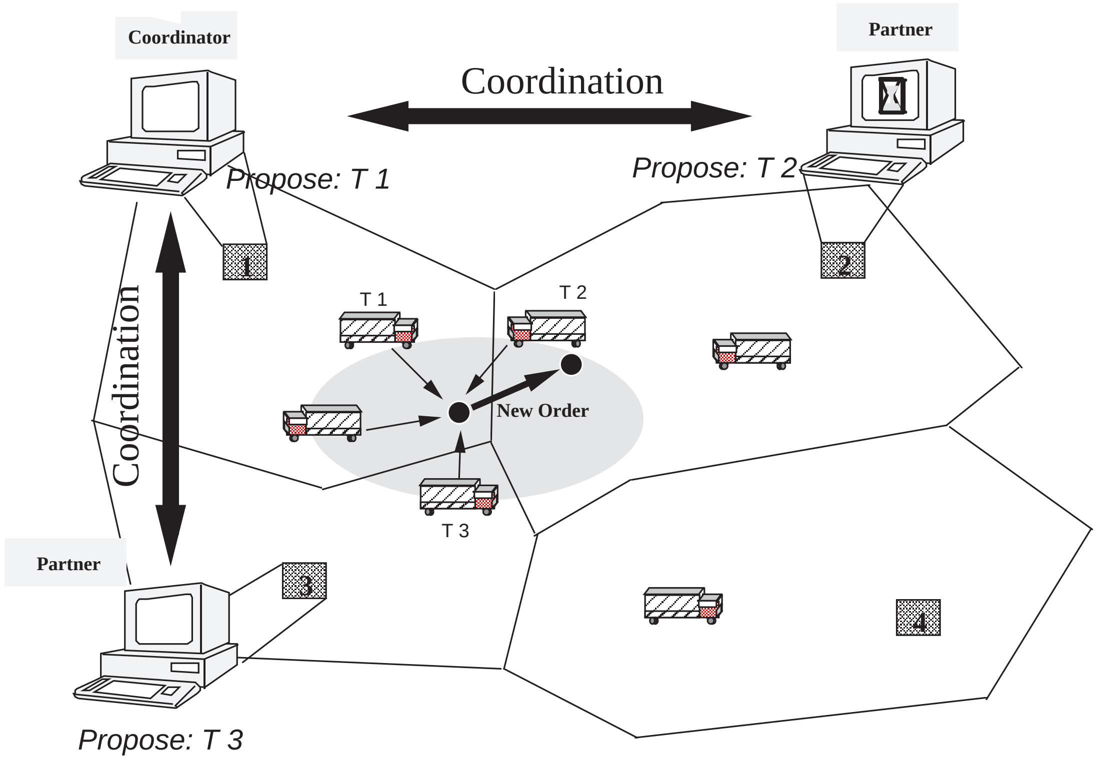


Figure 6: Cooperation in the Tramp Agent Application Domain of Falk et al.

Each P-PIA proposes a possible allocation of the order to its truck to the C-PIA who evaluates all the proposals and which eventually chooses the best one.

This procedure is basically an implementation of the Contract Net Protocol as proposed in [Davis & Smith 83]. However, the approach of Falk et al. does not only aim at using the Contract Net for the task allocation process in this domain but also to use it for the task decomposition process.
To process a new order each PIA has available different operators to modify its local plans, as there are *insertion, moving,exchange* of orders, *reload* of goods, and *joining* of tours. In a first round, the C-PIA asks for bids where only the insertion operator will be used. For the insertion of an order into their local plan all PIAs use a two stage Branch and Bound algorithm of Wilson (in [Bodin 83]) by which the order is inserted due to a minimal detour. If the bids of the P-PIAs show that no insertion can be done in a satisfactory way, the C-PIA initiates a new bidding round where bids including the operator of moving orders are requested. If this does not result in a satisfactory solution the next operator is chosen, and so on.

The work of Falk et al. was initiated by a Logistics Support Company who needed a new technology to provide planning tools to its customers. Compared to our modeling the approach described above considers an instance of our domain, namely a single company which is geographically distributed. Thus, the dispatching agents are willing to exchange all the information (in this case, the complete route plans) in a cooperation process. A further difference to our modeling is that the trucks are not modeled as agents. This might be motivated by the fact that the PIAs have to exchange their information about routes of the trucks, implying that they have to know them. This fact reduces the advantage of modeling a truck as an autonomous entity.

Another difference in terms of the motivation for the research activities is that the modeling of the MARS system intended to study the applicability of DAI techniques to real world applications. On the other hand, the group at the university of Nürnberg tried to find new mechanisms to solve the Dynamic Vehicle Routing Problem. From this point of view it is worth to mention that the architectures associated with these two approaches converge to the implementation of a multi-agent system providing hopefully good solutions to the DVRP.

The approaches discussed in this section are more or less aiming at the development of a disposition support system that can be used in a real-world company. All authors agree that for this goal it is not suitable to integrate into such a system e.g. algorithms that solve the routing or scheduling *exactly*. Instead, they are looking for heuristics that provide a "good" solution in a reasonable amount of time. This is also the purpose for our application. However, compared to the approaches presented in this section we consider a more general scenario where we try to model shipping companies that are primarily self-interested and only secondary cooperative.

This latter aspect was also one of the reasons why we decided to choose the transportation domain as an application to study the applicability of DAI techniques. In most of the approaches in the DAI field that are dealing with modeling cooperation agents mostly are either cooperative or not, i.e. there is no reflection about the conditions and the motivation for agents participating in a cooperation process. This aspect also raises questions for the decision processes in the planning phase within the agents or questions for an appropriate choice of partners at the beginning of an cooperative act. This twofold view on cooperation was formalized in the model of *Pattern of Interaction* in ([Müller 93]).

However, in the report at hand we concentrate on the more technical aspects of cooperation in the transportation domain. In section 2.4 we formalized the routing and scheduling problem of a set of distributed shipping companies. and gave a lower bound for this problem. In the next section we figure out two main steps in this distribution problems, namely the *task decomposition phase* done by the company agents and their trucks and the *phase of route planning* which is done by the trucks.

4 A DAI based Heuristic approach to Scheduling and Routing for Distributed Shipping Companies

The complexity theoretical results of section 2.4 show the intractability of the scheduling and routing optimization problem within the MARS scenario. In order to cope with this problems despite of this results and to keep them manageable in a computer implementation usually *heuristics* are applied to solve the problem. However, this implies that in general the solution constructed for a problem instance may be far from being optimal.

Nevertheless, we also have chosen the goal to apply heuristic mechanisms to solve the problem stated in section 2.1. We will present the ideas underlying them in the following. An evaluation of the algorithms based on these methods in comparison to other approaches is contained in section 9.

Our approach to develop a heuristic algorithm for the scenario is based on exploiting techniques from the field of DAI. This is motivated from looking at the real-world situation (cf. section 2.1): The description of the scenario reveals the autonomy of the agents as a necessary condition for a modeling that reflects the real-world situation and that can even support the dispatcher in a real shipping company.

In the routing and the scheduling problem of the MARS scenario we distinguish different phases as shown by figure 7: The orders a client enters into the system have to be first allocated to particular companies and then to a set of trucks belonging to the company agents. In this terms, this describes a pure process of *task allocation* in the system. However, if we allow the orders to exceed the capacities of the single companies or trucks this process has to be combined with a process of *task decomposition*, i.e. the order has to be split up into several sub-orders that can be allocated to the transporting units.

Besides these two processes for the *task handling* in this multi-agent system, there is a

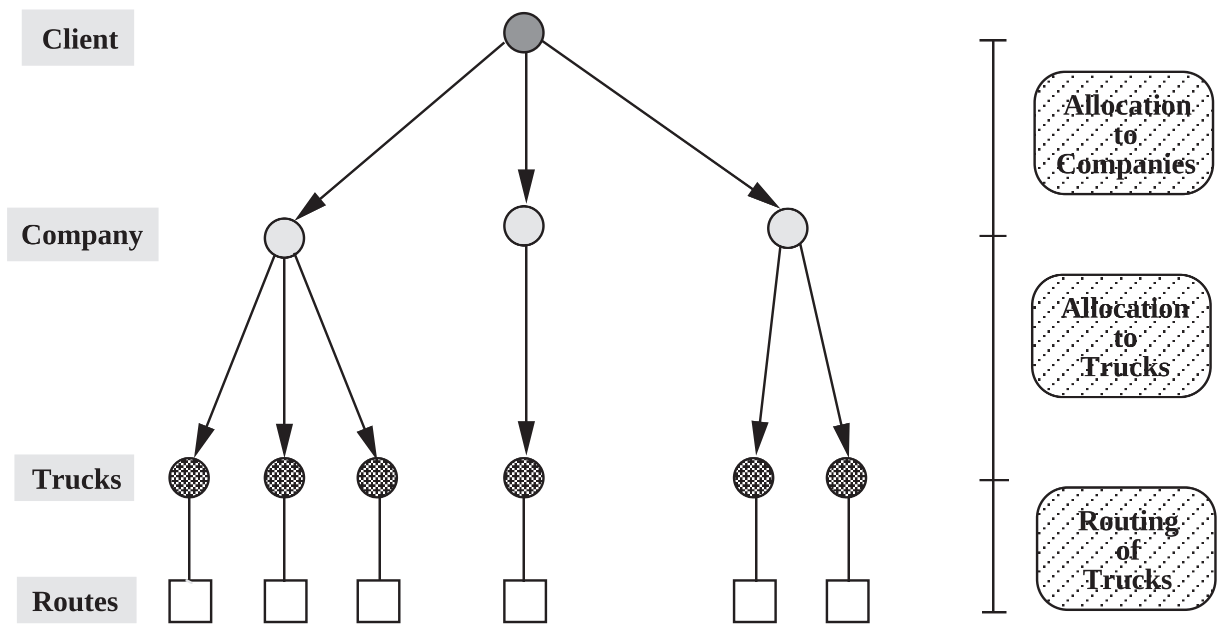


Figure 7: Phases of Task Handling in the MARS Scenario

second process which we want to mention here, namely the process of *routing* within the individual truck.

For the heuristic routing and scheduling approach in the MARS domain we are using heuristic functions in both of the two main phases mentioned above, namely for the task decomposition as well as for the task allocation process, and for the individual routing within a truck. The routing process will be presented in detail in section 8.

In the remainder of this section we first present different models that can be taken into account to model the task handling process in our domain. Then, we discuss how the structured exchange of information by the process of negotiation can contribute to an optimization of an existing task decomposition and task allocation solution.

4.1 The Process of Task Decomposition and Task Allocation

A multi-agent system (MAS) $\mathcal{M}$ is a pair $\mathcal{M} = (\mathcal{T}, \mathcal{A})$, where $\mathcal{A} = \{a_1, ..., a_n\}$ denotes the set of agents that comprise the system $\mathcal{M}$, and $\mathcal{T} = \{t_1, ..., t_l\}$ describes the set of *tasks* that the society of agents is able to perform. These tasks are accomplished by that the agents perform a set of *actions* they are capable of. We call a task t an *atomic task for agent a* if it can be accomplished by the agent alone. Otherwise, we call the task a *complicated task* for agent a.

In general, the process of the *task decomposition* is as follows: Given a task $t \in \mathcal{T}$ as input, t has to be compiled into a set of atomic tasks $T_t = \{t_{t_1}, ..., t_{t_m}\} \subseteq T$ that can be attached to particular agents.
Usually, there may be several alternatives to compile a task into a set of subtasks according to different possible solutions to a problem (or task) or to the set of agents that are actually part of the system. Thus, it is often suitable to implement the task decomposition process as an iterative process that gathers the information necessary in a series of steps.
Furthermore, in general the two processes of task decomposition and task allocation have to be closely interleaved, because a task may be atomic for one agent while it is complicated for another agent. For instance, carrying a table from a location A to a location B might be done either by one strong agent or by two weaker ones. If the task allocation process prefers the latter case, the task decomposition process must proceed, and in order to express the conjunction of the two agents for the accomplishment of the task, it has to add constraints to the description of the new subtasks that must be satisfied when the task is finally accomplished.

Another process in a multi-agent system that can have impacts on the task decomposition and the task allocation processes is the process of *planning*. The multi-agent systems that we are interested in can be characterized by the term *dynamic multi-agent systems*. On one hand,we want to stress with this notion the fact that agents may enter or leave the system at any time which may result in a change of the topology or of the hierarchical structure of the multi-agent system. Important to mention in this context is that we have no longer a system that is given a set of tasks at some starting time t and which will finish after some while having fulfilled all the initial tasks. Instead, the agents have to deal with a continuous stream of incoming tasks, imposing as a consequence that a new incoming task can force the system to modify plans (or decompositions) that have been worked out before, because the former solution suddenly looks less reasonable now or even, does not work any more now. This may even include a rollback of actions that have been already executed[4]. In other words, the input of new tasks may imply the necessity of replanning sequences of actions for some of the agents.

On the agent level the allocation of a new goal to some agents can involve that these agents are no longer able to accomplish each task they have been committed to before. Rather, some of the tasks have to be retracted from the agents, and are thus open for decomposition again.

Therefore, a process for the decomposition of the tasks in a MAS $\mathcal{M}$ should keep track of at least the following parameters:

[4]This might be not the case for actions that consume some limited resources, like fuel, etc.!

1. The "state" of $\mathcal{M}$ from the viewpoint of the agents, yielding e.g. information about the current set of tasks in the system (i.e., which tasks are open for decomposition; which decomposition has been chosen for the other ones), and knowledge about preferable decompositions.

2. agents that are in general suitable for the accomplishment of a particular task

3. agents that are actually available for the accomplishment of a particular task.

The interleaving of the different processes in a multi-agent system is influenced by different paramaters, among them

- the structure of the task
- the topology of the multi-agent system
- the roles of the individual agents

The implementation of the coupling of the different processes in multi-agent systems (as well as in other kinds of distributed systems) often applies the technique of *protocols* by which the agents provide the information necessary for a process to each other.

The most widely known protocol in this field is the *Contract Net Protocol* of Davis and Smith. Its applications cover almost the whole spectrum of multi-agent systems, ranging e.g. from the domain of distributed manufacturing systems (cf. [Parunak 87]) to the domain of air traffic control (cf. [Cammarata et al. 88]). In our application this protocol also is considered. However, the applications have also shown the need for modifications of the contract net protocol and for other protocols supporting cooperation of agents.

In [Kuhn et al. 93b] we discussed different models of task-decomposition and their implementation in form of protocols. The rest of this section gives a brief overview on this discussion.

4.2 The Contract Net Protocol

The Contract Net Protocol (CNP) was introduced by Smith and Davis in a series of publications ([Davis & Smith 83, Smith 80]). The general idea is the following: A certain task is given to a society of agents. One agent, called the manager, receives the task and divides it into a set of subtasks. He announces them (in a sequence of announcements) to a set of eligible agents (chosen on the basis of his knowledge about the others). These agents process the task announcement, i.e., they rank the task relative to others currently under consideration. When being idle at some time, they prepare bids for stored tasks and send the bids to the respective managers. The manager ranks the incoming bids and after an expiring time he chooses the best one. An instantiation of the CNP for the domain of a shipping company is shown by figure 8.

Though we think that the CNP is a very elegant way to coordinate agents in the task allocation process, there is one big bottleneck with the approach. For many interesting applications, there are quite a few good reasons to consider the central role of the manager as being too powerful:

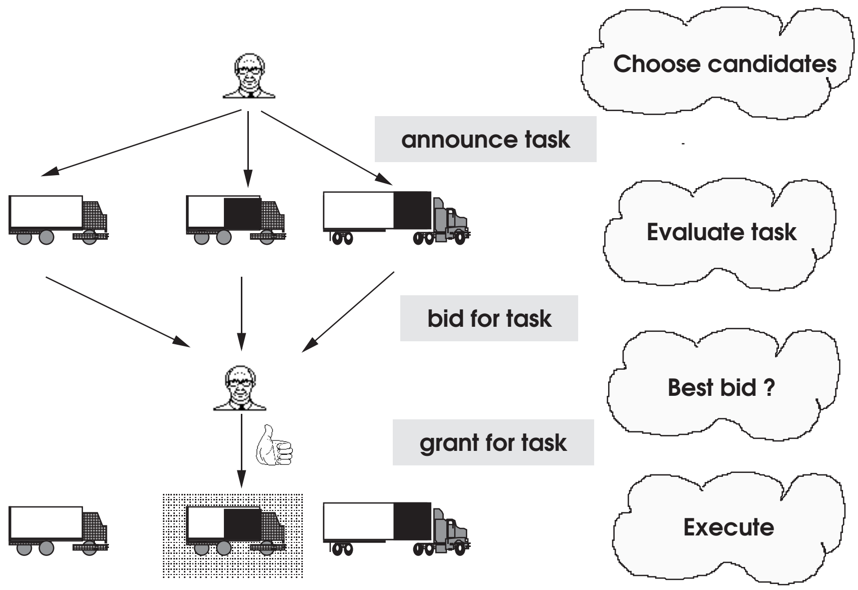


Figure 8: The Contract Net Protocol in Shipping Companies

1. To choose a subset of eligible agents, the manager needs to have a large amount of knowledge about the other agents in the society. This does not correspond to the philosophy of decentralization of data. Even more, since each agent is a potential manager, the knowledge must be available to each agent.

2. The manager has to have several strategies to decompose a task. Choosing the wrong decomposition means that several rounds of announcing and bidding are necessary until a complete subtask allocation is installed.

We eliminate the first problem in giving the task decomposition procedures to the contractors (bidders). This reduces the amount of global knowledge needed by the agents and allocates the different task decomposition procedures to responsible and eligible agents. In other words, each agent will decompose a given task on his own behalf and pick out a maximum executable subtask of his own. The role of the manager now becomes that of a solution synthesizing specialist who actually organizes the cooperative work of the group. The advantage of the described Decentralized Task Decomposition Model (DTDM) is the local expertise of agents.

However, the DTDM has still the problem of reliability due to the central position of the manager. The next step will eliminate this drawback in delegating the mission of the manager to the society. Since it cannot be distributed in the same way as the task decomposition procedures, there must be another mechanism which goes beyond the interpretation of negotiation provided by the CNP. The agents explicitly have to negotiate on the subtasks they like to work on. By collecting piece by piece the subsolutions, a "joint plan" must be built. If there are conflicts, for instance if more than one agent applies to the same subtask, or if subtasks overlay, the agents have to negotiate with the aim of a balanced load.

4.3 The Decentralized Task Decomposition Model

Motivated by the shortcomings of the Contract Net Model of task decomposition for dynamic agent societies described in section 4.2, we would like to approach one stage closer to the paradigm of a decentralized system by a model which we call the Decentralized Task Decomposition Model (DTDM). In this model, the original structure of the contract net is softened by shifting the task decomposition to the society of contractors: the manager receives a task and passes it as a whole to a set of eligible contractors. The contractors work out a bid for a part of the task, and pass it back to the manager. Now, the manager can synthesize a plan for the task from the bids for subtasks received by some of the contractors, while rejecting the bids of other contractors. In section 5.2, we will describe how negotiation between the manager and the contractors can help to find more appropriate task decompositions which lead to better solutions to the overall task.

Compared to the Contract Net Model, the DTDM yields a more flexible behaviour of the system, since

- The manager needs less knowledge about the different contractors. Rather, each contractor may choose a subtask which seems appropriate to him. However, by knowing the subtasks offered by the contractors, the manager can have an important coordinating function.
- Communication costs are reduced, because instead of announcing each subtask, the manager only announces the task as a whole.
- By employing negotiation between the manager and the contractors, task decompositions can be achieved which are both locally and globally acceptable.
- The dynamic nature of the system can be handled more easily. The manager does not have to know the agents that are currently in the system to decompose a task into a subtask. If the CNP is used and some relevant agents are missing or unavailable for the solution of a task it may take several rounds of task announcements until a suitable task decomposition is found.

However, for some domains, even the existence of a manager is not desired or just impossible to assume. For these domains, the DTDM might be regarded not satisfactory. Therefore, in subsection 4.4, we introduce a model which provides a degree of decentralization which is even higher than in the case of the DTDM.

4.4 The Completely Decentralized Model

In the Completely Decentralized Model (CDM), the society of agents has to decompose and to allocate the tasks and to synthesize a plan for carrying out the task without the help of a manager. This decentralized task decomposition and task synthesis can be viewed as a decentralized planning process. Agents may either propose whole plans or partial plans to other agents, or they may construct a joint plan e.g. by using a system of a circular letter which is sent from agent to agent, and which can be modified by each agent, until a complete plan is built which is accepted by all participants. The absence of a central instance causes many new problems to occur: agents may have different and even inconsistent intentions, different degrees of cooperativeness, very diverse amounts and types of knowledge and beliefs, and different skills and abilities. Finding coordinated plans requires such an agent society to communicate, to exchange goals, plans, arguments, and intentions, to cope with conflicts etc.

Negotiation [Durfee & Montgomery 90, Kreifelts & v. Martial 91] establishes a very powerful tool for handling this kind of problem. By negotiation, conflicts between agents can be bridged, an agent can convince another agent of the benefits of his proposal, or the frame conditions for a joint plan and the joint plan itself, i. e. the task decomposition and allocation, can be agreed on. In section 5 we will give an overview on different ways to use negotiation for task decomposition in Multi-Agent Systems. In the case of the CDM, we can say that to use some form of negotiation between agents is not a choice which is up to the agents (or to the designer of the agent society).

5 Task Decomposition by Negotiation

In chapter 4 we introduced several models of task decomposition which differed by their degree of decentralization. There we supposed that agents would send proposals for task decomposition to other agents, and that these might either accept or reject the proposal.

However, if we want to obtain a more realistic view on cooperation, aspects of negotiation should be integrated. By [Bussmann 92], negotiation in a multi-agent context is defined as *the communication process of a group of agents in order to reach a mutually accepted agreement on some matter*. According to this definition, the task decomposition itself may be negotiated on. In this section we would like to outline how task decomposition can be negotiated in the models defined in section 4.

5.1 Negotiation in the Contract Net Model

The model for task handling based on the contract net is characterized by centralized task decomposition and centralized task synthesis. The manager splits a task into several subtasks and announces each subtask to one agent or a group of agents. In the original Contract Net Protocol, each agent may either make a bid for a subtask, or it may show no interest for doing that subtask. Thus, in order to find a suitable task decomposition, the manager needs profound knowledge of other agents' problem solving capabilities and even of their internal representations. Otherwise, there is a considerable risk that for a given subtask no contractor will be found.

A more flexible mechanism for task decomposition in the Contract Net model can be achieved by allowing negotiation on the frame conditions of a subtask between the manager and the potential contractors. By this, satisfactory task decompositions can be reached even when the manager has no complete knowledge (or even wrong beliefs) of the potential contractors. Figure 9, taken gives an example for this which is taken from the MARS domain.

Example 1 *Assume S_1, S_2 are shipping companies. S_1 owns one truck with a loading capacity of* 20 *units, S_2 owns one truck with a loading capacity of* 10 *units. A customer C has a task $T =$*"Transport 6 pallets each of five units from place A to place B!". *Assume that C has no knowledge about the loading capacities of S_1 and S_2, respectively. In this case, it may use a heuristics, namely to decompose the order in two equal parts $T_1 =$*

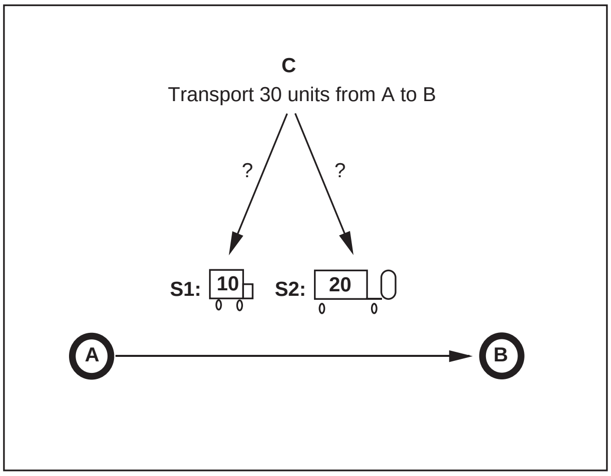


Figure 9: Example 1

"Transport 3 pallets from A to B!" *which it decides to send to* S_1 *and* T_2 = "Transport 3 pallets from A to B!" *which it sends to* S_2.

Using the normal CNP, S_2 *would recognize that it is not able to carry out the task (at least not directly). Therefore,* S_1 *would be granted* T_1 *whereas* T_2 *could not be carried out, at all.*

Negotiation in this context can be integrated if the companies report the manager their free capacities. This may either be on request or they could do it on their own if they recognize that there is some capacity left that could be filled up by a similar order. Although the first one of these alternatives can work in a computer implementation it seems to be unrealistic in a real-world setting of autonomous agents. However, the latter alternative has already some similarity to the unbooked leg cooperation in the MARS scenario (see section 5.4).

If we allow negotiation between the customer and the potential contractors in the example above, the following will happen:

Example 1, contd.: S_2 *might tell* C*: "I cannot transport* 15 *units, but I can transport* 10 *units." Now,* C *can use the knowledge obtained by the negotiation with* S_2 *in order to choose another task decomposition consisting of* T_1' = *"Transport* 2 *pallets from* A *to* B*" which it sends to* S_1*, and* T_2' = *"Transport* 4 *pallets from* A *to* B*" which it sends to* S_2.

Thus, an appropriate task decomposition can be found.

5.2 Negotiation in the Decentralized Task Decomposition Model

In the decentralized task decomposition model of section 4.3, the manager is no longer responsible for task decomposition. Instead, it sends the task as a whole to the potential contractors, each of which may cut a slice of the task for himself, and announce to the manager his interest in that particular part of the task. The manager now synthesizes a plan for the complete task from the proposals of the agents.

In some ways, decentralized task decomposition models suffer from their locality. The contractors have only a local view, and they will choose subtasks without taking into consideration the behaviour of other agents. Therefore, it often happens that either there are conflicts between several contractors (e.g., some contractor wants to do the task as a whole, which is certainly impossible), or that parts of the task are not chosen by any contractor. Negotiation can be considered as a solution to these problems: there can be a negotiation between the manager and potential contractors in order to modify announcements of subtasks, which correspond to the task decomposition proposed by a contractor. Here, the manager can take advantage of his more global view obtained by knowing the offers of several contractors. On the other hand, contractors can negotiate with each other. This is a step into the direction of a completely decentralized system, where no manager is required (cf. subsection 5.3) at all. How ever, we can imagine a hybrid solution where a manager announces the tasks and receives and synthesizes offers for task decomposition, but where negotiation between contractors (e.g., in order to form a group solving a single subtask) is possible.

Again, we would like to show by an example how task decomposition proposals made by potential contractors can be modified by negotiation in order to reach a better solution of the overall task. The example is illustrated by figure 10.

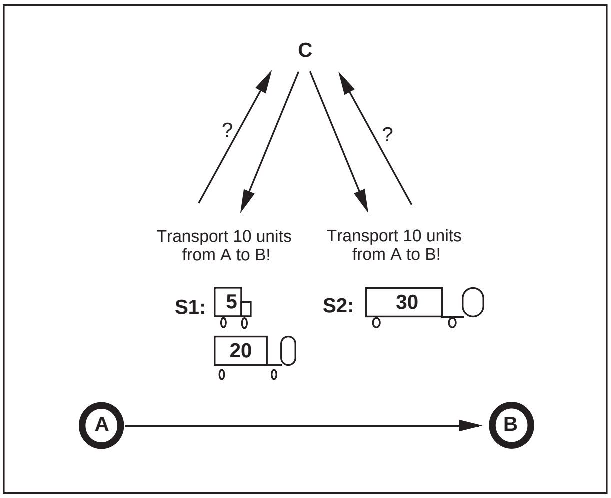


Figure 10: Example 2

Example 2 *Again there are a customer C, two companies S_1 and S_2. S_1 has two trucks with loading-capacities of* 5 *and* 20 *units, respectively, and S_2 owns one truck with a loading capacity of* 30 *units. Now let the task T be* "Transport 10 units from A to B". *Both S_1 and S_2 receive T and check which parts of T they are capable and willing to carry out. Now assume that both S_1 and S_2 use a heuristics which says not to apply for a task if the truck which is to perform it cannot be loaded by more than* 50% *of its loading capacity. In this case, S_1 would apply for transporting* 5 *units with his small truck, and S_2 would not apply for T, at all.*

If we allow negotiation, S_1 could propose to C to transport 10 *units, i. e. to carry out T completely, if C will pay more for it, and they could agree on a higher price for performing T.*

In conclusion, the use of negotiation in decentralized task decomposition models allows higher flexibility and a better performance of the system as a whole.

5.3 Negotiation in the Completely Decentralized Model

As described in section 4.4, by decentralizing the synthesis of tasks we obtain a completely decentralized task model. Here, the manager has become superfluous. Rather, the agent society decomposes the task in a set of subtasks and combines the solution to the subtasks to a plan for the task as a whole.

In completely decentralized models negotiation is not only reasonable, but it is very necessary, since it allows agents to cope with tasks without having complete knowledge about the abilities of others. Therefore, agents maintain models of other agents which contain their beliefs about the capabilities, intentions, and plans of these agents. The partner models are updated by messages received from other agents, and by perceiving the behaviour of these agents. Lacking information needed for making decisions can be acquired from other agents by asking them questions.

As we said before, negotiation between agents is performed via the sending of messages. Agents may create plans for the task or for subtasks and send them to other agents who can accept, reject, refine, or modify these plans (cf. [Durfee & Lesser 89, Kreifelts & v. Martial 91] as examples), thus finally agreeing on a joint plan.

5.4 Negotiation in the MARS Scenario

The MARS scenario as shown by figure 3 comprises two kinds of cooperation which are distinguished according to the hierarchical relationship between the agents involved: At first, there is the cooperation between the company agent and its trucks, which have to support it in the task decomposition and task allocation phase. This form of cooperation is called *Vertical Cooperation* because the responsibility for the decision of the outcome of the cooperation is dedicated to the company agent alone. The bidding process of this cooperation will be discussed in detail in section 8.

A second form of cooperation exists between different shipping companies that negotiate about the exchange of orders to improve the task decomposition and task allocation they have been choosing. This cooperation is called *Horizontal Cooperation* according to the peer-to-peer relationship between the shipping companies. An example for a situation where this form of cooperation provides an essential improvement for the overall task decomposition and task allocation is illustrated by figure 11

The invocation of the decentralized (and cooperative) task handling process will be triggered by the recognition of the cooperation pattern 'avoidance of unbooked legs' or 'coupling of local traffic and long-distance transportation orders' by one of the companies. Figure 11 shows how these types of cooperation can be combined to obtain a solution for the problem of task decomposition and task allocation in a situation with the set of orders $\{o_1, o_2, o_3\}$ for the shipping companies $\{C_1, C_2, C_3\}$. To achieve the solution that is displayed in the right part of the figure the negotiation mechanism could proceed as follows:

- C_1 asks C_2 to take over the local distribution of o_1 and he offers a free truck to C_2

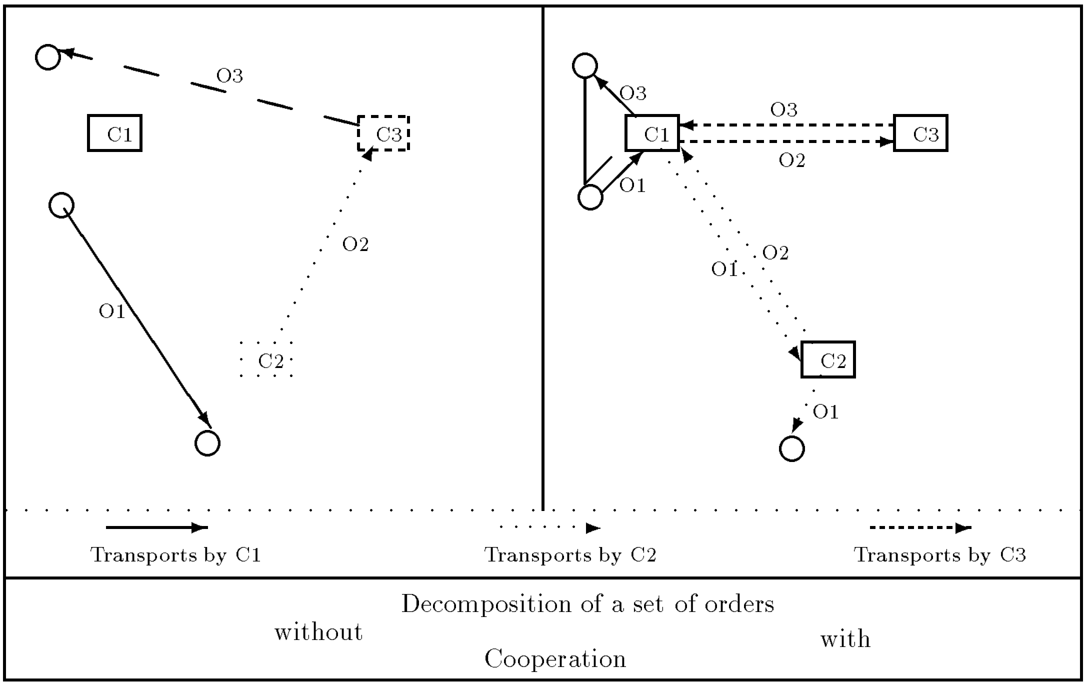


Figure 11: Cooperation Between Shipping Companies in the MARS Scenario

- C_2 offers a free truck to C_3
- C_3 asks C_1 to take over the local distribution of o_3 and he offers a free truck to C_1
- C_1 agrees on doing the local distribution for o_3, if C_3 takes over order o_1
- C_3 disagrees on that because it does not want to go to the location of C_2
- C_1 updates his offer to C_2 concerning o_1 and asks C_2 to do the long-distance part of o_1
- C_2 rejects because it has to deliver order o_2
- C_1 asks C_2 if it would be useful for him to have available the truck of C_3
- C_2 accepts the truck offered and re-plans the route for the order o_2

Negotiation in this example is concerned with both kinds of real-world cooperation that we are integrating in our MARS scenario. Namely, these are negotiation for the *coupling of inter- and intra regional traffic* and for the *avoidance of rides without carriage.* To incorporate these forms of cooperation into our system this requires the integration of both vertical and horizontal cooperation. One way to achieve the avoidance of rides without carriage is the *unbooked leg* cooperation, where the agents try to improve the load of parts of their routes it this turns out to be unsatisfactory. As a consequence from the chosen modeling of the scenario by our multi-agent system it follows that this cooperation may require the interleaving of both, the vertical cooperation, and the horizontal cooperation between the agents. This is due to the roles of the truck and shipping company agents: only the truck agents know the routes, and can therefore give an estimation for the quality of a leg while on the other side only the company agents maintain contacts to different trucks or to other company agents.

Truck ⟶ Company:

ANNOUNCE:	announce an unbooked leg.
REVOKE:	revoke a previously announced unbooked leg.
FINISH:	Indicates the end of the validity of a previous announcement, triggers the confirmation the leg to a suitable applicant.

Company ⟶ Truck:

INSERT:	if a suitable applicant has been elected, the truck shall update its plan.
FREE:	response to a REVOKE, by which the company frees the truck from its obligation as regards a previously announced unbooked leg.

Company ⟶ Company:

OFFER:	A company offers free capacity to another company.
REVOKE:	A company revokes a previously offered unbooked leg.
APPLY:	A company applies to an offer for an unbooked leg.
GRANT:	A company grants the unbooked leg to a suitable applicant.

Figure 12: Protocol Primitives for the Unbooked Leg Cooperation

Figure 12 shows a set of cooperation primitives to model a protocol for the unbooked leg cooperation. The protocol is initiated by a truck who recognizes an unbooked leg in its route planned for the delivery of its orders. It *announces* this leg to its company agent, which decides what it wants to do with this free capacity. One possibility is to offer it to eligible other company agents (e.g., partner companies) which may then apply for it[5]. After an expiration time (e.g., when the truck has to start to deliver the next orders in time) either the company agent chooses the best order among the applications and allocates it to the truck or it allows the truck to leave without an additional order. There may be other cases that make need for a *revoke* message for a previously announced unbooked leg e.g., a new order received from the bulletin-board or that the truck could rearrange its route and does not have the unbooked leg any more. All these cases are synchronized by the company agent.

The description of the protocol for the unbooked leg cooperation based on the speech act primitives of figure 12 expresses the protocol or communication layer of this respective form of cooperation.

In general a cooperation mechanism comprises two different layers: the *protocol layer* which describes the possible sequences of messages that are exchanged in order to provide information to each other that is necessary to establish cooperation, and the *decision layer* that must be present in the agents in order to decide how to react on receiving a message or which message should be sent next. Another decision that has to be taken is e.g., which agent should be asked for participation in a cooperation mechanism. The latter has been discussed in a quite general setting in [Haddadi & Sundermeyer 93].

For the domain of the MARS scenario these two aspects of the decision layer have been considered by [Russ & Vierke 93]. To support the decision of a company agent whether

[5]Another possibility would be to keep it and to wait for a suitable order.

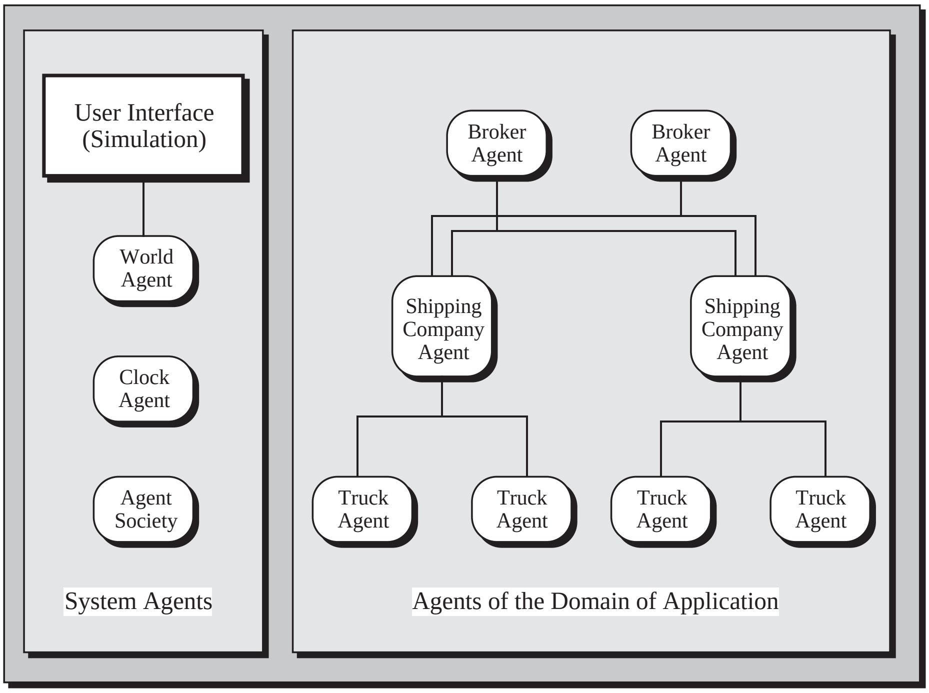


Figure 13: The Agent Society of MARS.

it should pass an announced unbooked leg to one of its trucks or to another company they developed a further negotiation protocol where the price of an order exchange is negotiated on. Therein, each company maintains her own criteria in form of an expected win and bases the decision for an exchange of an order on how this requirement is met. It is worth to mention here, that this protocol allows *multi-lateral* negotiation, i.e., many companies as well as other trucks from the announcing company may compete for the allocation of an unbooked leg.

The second aspect of the decision, the question whom to send a proposal to participate in a cooperation mechanism is solved by associating each company with a certain region in the map. If an announcement of an unbooked leg occurs within a company the company agent contacts other company agents that are located in the region where the staring point or the end point of the unbooked leg is located.

As we will see in section 9 this protocols have no impacts on the results that are discussed there. Thus, we will concentrate in the following on the aspects that are the basics for the implementation of the MARS system and that enabled us to run these test sets. In particular we will focus on the communication mechanisms and on the route planning within the truck agents. The cost functions that may be derived from this data provide the basis to enable the company agent to take its decisions in the task decomposition and in the task allocation process.

6 The Design of the Implementation

6.1 Description of the Agent Society

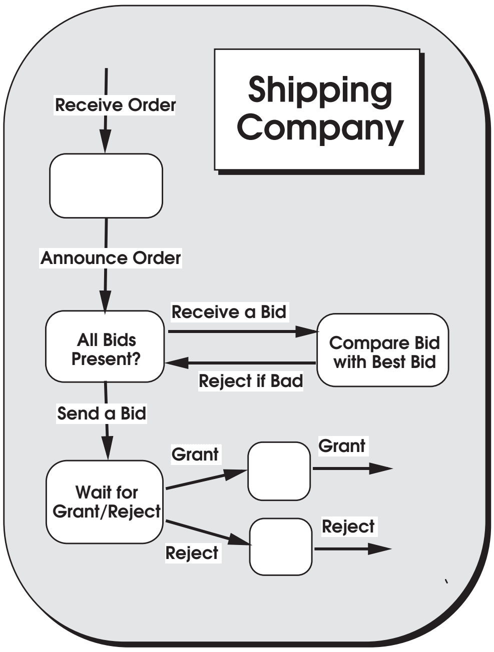
The implementation of the MARS system was done using MAGSY, a rule-based development environment for multi-agent system applications [Fischer 93]. In MAGSY, an agent is a self-contained unit which has its own reasoning capabilities. Figure 13 shows the architecture of the implementation. There are two types of agents in the system: agents for which there is a physical or logical instance in the domain of application, and system agents which were introduced for technical reasons. Agents for which there exists an instance in the domain of the application are introduced at three layers: the layer of the brokers, the layer of the shipping companies, and the layer of the trucks. The agents which represent shipping companies or trucks are explained in more detail in section 7 and 8, respectively. All new orders which are specified by a user are sent to a broker agent. The user is allowed to specify time and cost constraints for an order as well as a set of shipping companies among which the broker agent should select the cheapest one to execute the order. Therefore, the broker agents were introduced mainly to make it more convenient for a user to give orders to the system. Additionally, there were three system agents introduced: the world agent, the clock agent, and an agent for the administration of the agent society. The world agent implements the interface to the user and visualizes to him the actions of the truck agents, e.g. driving from one city to another one. The clock agent maintains the simulation time and offers elementary services such as wath and alarm functions.

6.2 An Extended Version of the Contract Net Protocol

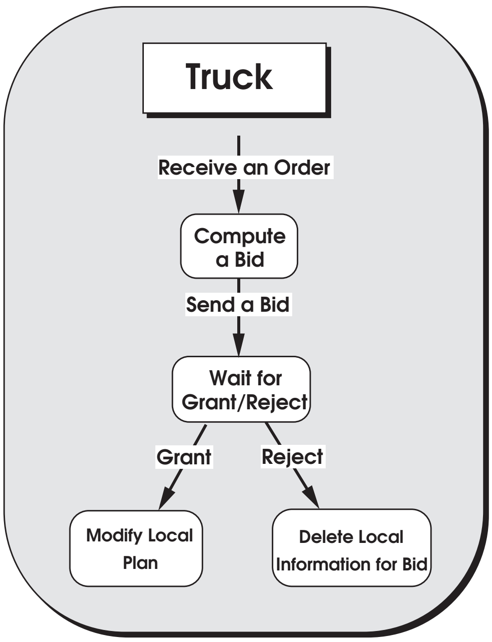
The contract net protocol has already been discussed in section 4.2. Several of its instances were described, starting with the central approach and discussing more sophisticated decentralized instances of the contract net protocol. However, the pure contract net protocol turns out to be not powerful enough when we have to deal with tasks that exceed the capacity of a single truck. This implies that the tasks have to be decomposed, which cannot be done using the pure contract net protocol. Therefore, we use an extended version — called ECN protocol — where the two speech acts *grant* and *reject* are split up into four new speech acts: *temporal grant*, *temporal reject*, *definitive grant*, and *definitive reject*.

We describe communication and cooperation between two agents by specifying patterns of interaction [Müller & Pischel 93b]. In the description of a pattern of interaction we distinguish the protocol layer and the decision layer. We use a flow chart representation to define the protocol layer of a pattern of interaction from the point of view of a single agent. Figure 14 shows the flow chart for the pure contract net protocol (a) from the managers (in this case the shipping company) point of view and (b) from the point of view of a bidder (a truck). Figure 15 shows the flow charts of the protocol layer of the ECN protocol, again, (a) from the manager's point of view and (b) from the point of view of the bidders. The difference to the pure contract net protocol is that the bidders, i.e. the trucks, are allowed to give bids which do not cover the whole amount of an order.

The communication procedure is as follows. The manager, i.e. the shipping company, announces an order to its trucks. It selects the best of the bids it receives for the order



(a)



(b)

Figure 14: Description of the contract net protocol from the point of view of a manager (a) and a bidder (b).

and sends the truck which gave this bid a temporal grant. All other trucks get temporal rejects. If the best bid does not cover the whole amount of an order, the shipping company subtracts the amount of the best bid from the amount of the order and reannounces the reduced order. This procedure is repeated until the shipping company gets a bid which covers the whole amount of the order which was finally reannounced. At this moment the shipping company has a set of bids which cover the amount of the original order which was given to the shipping company. The shipping company uses this set of bids to compute itself a bid for the whole task and gives this bid to the broker agent (see section 6.1). Only when the shipping company itself gets a definitive grant or a definitive reject, the shipping company passes this definitive grant or definitive reject to all trucks which got a temporal grants before.

The trucks on the other hand must be able to cope with the temporal and definitive grant or rejects messages. When a truck gets a temporal grant for the first time, it has to make a backup of its local situation, i.e. the currently valid plan, because it must be able to restore this situation in case it gets a definitive reject. All subsequent temporal grants and temporal rejects are handled as the grants and rejects in the pure contract net protocol. If a truck gets a definitive grant for an order, it removes the copy of the current situation which it created when getting the first temporal grant. On the other hand, if a truck gets a definitive reject, it has to remove all the local information gathered while receiving temporal grants and restore the current situation before it received the first temporal grant.

7 The Shipping Companies

If the pure contract net protocol were used for task allocation, then the decision function to select the bid of best quality would be straightforward: the truck which gave the bid with least costs could be chosen to execute the task. In the ECN protocol which is actually used to select a bid in a shipping company, the trucks are allowed to give bids which do not cover the whole order. In this case, the decision function must take care of both, costs and amount specified in a bid.

Therefore, the bids of the trucks are represented by triples

$$(t, c, a)$$

where t is an identification of the truck giving the bid, c specifies the costs of the bid, and a specifies the amount which could be transported. If the trucks of a shipping company are able to execute an order o, the shipping company will in general get a set

$$\{(t_1, c_1, a_1), \ldots, (t_n, c_n, a_n)\}, n \in I\!N$$

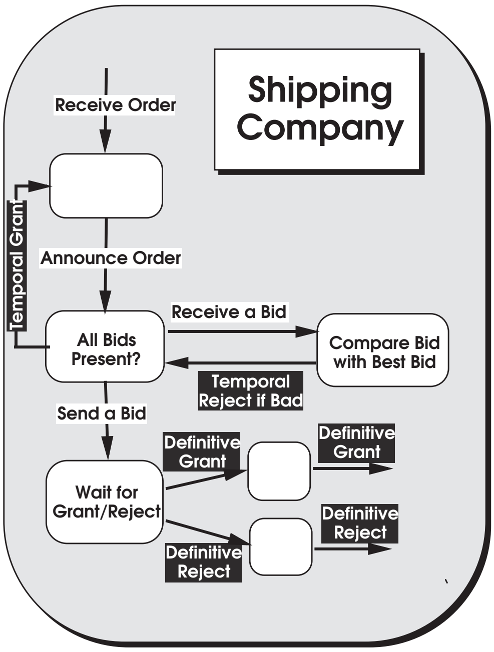
of bids for o where

$$o.amount = \sum_{i=1}^{n} a_i.$$

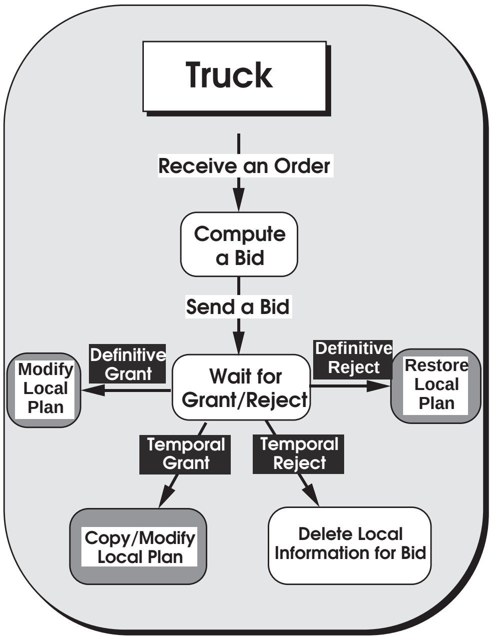
The quality of two bids can be compared by using the function *compare* (see figure 16)[6]. For two bids (t_1, c_1, a_1) and (t_2, c_2, a_2), the first one will be preferred if

$$compare(c_1, a_1, c_2, a_2) = \mathbf{true}$$

[6]The syntax of pseudocode is taken from [Bauer & Wössner 81]



(a)



(b)

Figure 15: Description of the extended contract net protocol from the point of view of a manager (a) and a bidder (b).

```
funct compare ( int new-costs, real new-amount,
                int best-costs, real best-amount ) bool:
    if new-costs / new-amount = best-costs / best-amount
    then best-amount < new-amount
    else new-costs / new-amount < best-costs / best-amount fi
```

Figure 16: Function to compare tow bids with respect to their quality.

holds, i.e. the quality of bid 1 is judged better than the quality of bid 2.

We will now analyse in more detail the decision procedure of the ECN protocol and its impacts on the overall system behaviour. If an order is consecutively announced to a truck and if the truck always gets a temporal grant, it will produce a sequence of bids

$$b^t(o) = ((t, c_1, a_1), \ldots, (t, c_n, a_n)), n \in I\!N_0$$

where

$$\sum_{i=1}^{n} a_i \leq o.amount$$

Note that it may be impossible for a truck to fulfill the whole order in its current situation because of constraints specified with the order. As a special case $n = 0$, may hold, meaning that the truck is not able to do any part of the order. Let $b_i^t(o)$ denote the i-th element of the bid sequence $b^t(o)$, i.e.

$$b_i^t(o) = (t, c_i, a_i)$$

Definition 1 *A sequence of bids $s = ((t_1, c_1, a_1), \ldots, (t_n, c_n, a_n))$ is a* valid bid sequence *for an order o iff*

$$\forall i \in I\!N : 1 \leq i \leq n : b_i^t(o) \in s \Rightarrow b_j^t(o) \in s, 1 \leq j < i.$$

Lemma 1 *The sequence of bids for an order o which is selected by the extended contract net protocol using the decision function* compare *is a valid bid sequence for order o.*

Proof: Due to the definition of the extended contract net protocol a truck t can produce bid $b_i^t(o)$ if it got temporal grants for all bids $b_j^t, 1 \leq j < i$. □

In the ECN protocol, selecting the bid with the minimum costs per unit is a greedy strategy for task allocation. At a first glance, the task allocation problem looks like a *fractional knapsack problem* (for which it is a well-known fact that it can be solved optimally by a greedy strategy [Cormen et al. 92]) because the trucks are able to cut arbitrarily small pieces out of an order. However, from the following example we see that the task decomposition problem does in fact behave like the 0-1 knapsack problem for which is known that a greedy strategy will result in suboptimal solutions.

Example: Assume that an order o of 70 units is announced to a shipping company which has four trucks at hand for doing the job. Assume further that these trucks would produce the following bid sequences:

$b^{t_1}(o)$ $(t_1, 220, 40)$
$b^{t_2}(o)$ $(t_2, 180, 30)$
$b^{t_3}(o)$ $(t_3, 100, 20), (t_3, 100, 20)$
$b^{t_4}(o)$ $(t_4, 100, 20), (t_4, 100, 20)$

In this example the ECN protocol will select the bid sequence:

$$s^{ECN}(o) = (t_3, 100, 20), (t_3, 100, 20), (t_4, 100, 20), (t_4, 100, 10)$$

which has total costs of 400 and costs per unit of $\frac{400}{70} \approx 5.71$. Whereas the bid sequence:

$$s^{opt} = (t_1, 220, 40), (t_2, 180, 30)$$

has total costs of 380 and costs per unit of $\frac{380}{70} \approx 5.43$.

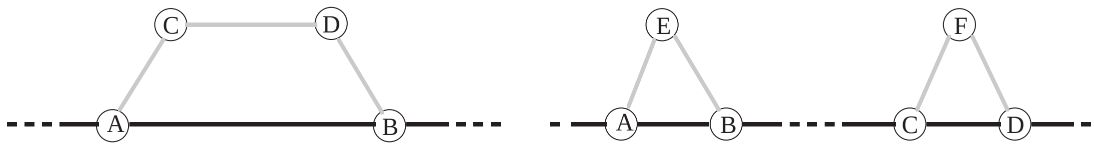
Even though this example shows that the ECN protocol may produce a suboptimal task allocation, one can easily see that such an example cannot be found for bid sequences with length of at most 2 elements. It shows that if at all it is always the last bid which makes the whole bid sequence suboptimal. This means that the whole problem is due to the fact that the shipping company has to collect a bid sequence which covers the whole amount of the order. Knowing that all but the last bid of every bid sequence are optimal choices with respect to the current situation of the trucks gives room for further improvements of the system.

An important thing to note is that the trucks compute their bids with respect to their current situation. The orders are announced to the shipping companies (and thus, to the trucks) one by one. As more and more orders are announced to the shipping companies, the situation in the trucks will change incrementally. Hence, what might look like a good task decomposition in the current situation might turn out to be a bad one on the long run and vice versa. Therefore, it does not make sense to do a time consuming brute force computation to find an optimal solution in a specific situation which might turn out to be a bad solution when time passes by.

8 The Trucks

When a new order is announced to the truck, it computes a bid according to its current situation. The bid states at which costs the truck is able to deliver the order. The current situation of a truck depends on:

1. its current position.
2. its currently valid local plan.



(a)
Allowed cases:
$(A \neq C \wedge B \neq D)\vee$
$(A = C \wedge B \neq D)\vee$
$(A \neq C \wedge B = D)\vee$
$(A = C \wedge B = D)$

(b)
Allowed cases:
Start: $(A \neq E \wedge B \neq E)\vee$
$(A = E \wedge B \neq E)\vee$
End: $(C \neq F \wedge D \neq F)\vee$ $C = F \vee D = F$
D need not be present.

Figure 17: Possible Extensions of a Plan.

To determine the costs, the truck agent computes all possible extensions of its local plan and selects the best one. Figure 17 (a) and figure 17 (b) show the possible extensions of a plan. In our first prototype the cost function considered only the length of the detour of the truck caused by the new order. Here, an order is represented by the following feature structure [Henz et al. 93]:

(order id : symbol
from : symbol
to : symbol
article : symbol
amount : real)

where

id A unique identification of the order.

from The name of the city the order starts from.

to The name of the city the order ends in.

article The type of good.

amount How much of the good has to be transported.

A plan is a list of single plan-steps each of which specifies that the truck has to go from one city (from) to another one (to). Each plan step is represented by the following feature structure:

(plan-step id : **symbol**
next : **symbol**
from : **symbol**
to : **symbol**)

where

id A Unique identification of the plan step.

next Symbolic reference to the next plan step in the list. **nil** is the symbolic reference the last plan step points to.

from Name of the city the plan step starts with.

to Name of the city the plan step ends with.

Let $\mathcal{T}$ denote the set of all trucks, $\mathcal{P}$ the set of all plan steps, $\mathcal{P}^*$ the set of all plans, and $\mathcal{O}$ the set of all orders.

Definition 2 *A plan* $P = p_1 \dots p_n \in \mathcal{P}^*, n \in I\!N$ *is a* valid *plan iff*

$$\forall i \in I\!N : 1 \leq i < n : p_i.next = p_{i+1}.id \wedge \; p_i.to = p_{i+1}.from.$$

Let

$$ext : \begin{cases} \mathcal{T} \times I\!N \times \mathcal{P}^* \times \mathcal{O} \longrightarrow \mathcal{P}^* \times I\!R \\ (t, i, P, o) \mapsto (P', a) \end{cases}$$

be a function which enumerates all extensions of a plan. There are two selector functions $plan(ext(t, i, P, o))$ and $amount(ext(t, i, P, o)$ which select the extended plan and the amount which can be transported, respectively. Note, that it is possible that different extensions of a plan can have different amounts which can be transported because of the capacity constraints which are specified for each truck. A plan is a finite sequence of plan steps, i.e.

$$\forall t \in \mathcal{T} : \forall P \in \mathcal{P}^* : \forall o \in \mathcal{O} : \exists n \in I\!N : \forall i, j \in I\!N : 1 \leq i \leq n \wedge \; j > n :$$

$$plan(ext(t, i, P, o)) \neq \varepsilon \wedge \; plan(ext(t, j, P, o)) = \varepsilon$$

where ε denotes the empty sequence. The costs of a specific extension can be computed with the help of the function:

funct *length* $(\mathcal{P}^* P)$ **int**:
    if $P = \varepsilon$
    then 0
    else $\mathcal{P}\ p \equiv top(P);$
        $\mathcal{P}^* H \equiv rest(P);$
        distance(p.from, p.to) + *length*(H) **fi**

where $distance(a, b)$ looks up the shortest distance between city a and city b in a map. Let

$$\mathcal{E}(P, o) = \{ext(i, P, o) | i \in I\!N\}$$

The bid which is sent to the shipping company is then selected from the set

$$\mathcal{B}(t, P, o) = \{(\hat{P}, \hat{a}) | \; \nexists (\tilde{P}, \tilde{a}) \in \mathcal{E}(t, P, o) : compare(length(\tilde{P}), \tilde{a}, length(\hat{P}), \hat{a})\}$$

If $card(\mathcal{B}) > 1$ then one of these extensions can be chosen freely. if $card(\mathcal{B}) = 0$, then the truck is not able to give a bid for the order and, therefore, gives a *no-bid* telling the shipping company that the truck is not interested in this order.

Although the minimization of the distances is an important criterion for the tour optimization in a truck time constraints are equally important in practice. When time is introduced, in general, an agent has to plan activities which have an earliest start time (EST), a duration (DUR) and a due date (DDA). The agent is not allowed to start the activity before the earliest start time. For the due date two interpretations are possible. In the first one (we will call it *strong interpretation*) the activity has to be finished before the due date. In the second one (we will call it *weak interpretation*) the activity has to be started before the due date. This means that in the weak interpretation an activity is allowed to finish after the due date. In the literature (e.g. [Desrochers et al. 92]), normally the weak interpretation for the time specifications of an order is assumed.

The execution of an order by a truck can be divided into three phases: loading, driving and unloading. All these phases consume time. In addition loading and unloading may have assigned earliest start times and due dates. Therefore, the feature structure of an order must be extended to:

```
( order id        : symbol
        from      : symbol
        to        : symbol
        article   : symbol
        amount    : real
        est_start : integer
        dur_start : integer
        dda_start : integer
        est_end   : integer
        dur_end   : integer
        dda_end   : integer )
```

est_start The earliest time when the loading of the truck can be started.

dur_start The estimated time for the duration of the loading of the truck.

dda_start The due date for loading the truck.

est_end The earliest time when the unloading of the truck can be started.

dur_end The estimated time for the duration of the unloading of the truck.

dda_end The due date for unloading the truck.

Thus, a truck has to plan two different types of activities: loading or unloading goods in a city and driving from one city to another one. Therefore, the feature structure representing plan steps is extended to:

```
( plan-step id    : symbol
            next  : symbol
            from  : symbol
            to    : symbol
            est   : integer
```

duration : **integer**
dda : **integer**
type : { tour city })

where

id Unique identification of the plan step.

from City name of the city the plan step starts with.

to City name of the city the plan step ends with.

est Earliest start time of the plan step. The truck is not allowed to start the execution of this plan step before the point in time specified in this field.

duration The estimated duration of the plan step.

dda Due date of the plan step. Either the truck has to finish the execution of the plan step before the point in time specified (strong interpretation; type = tour) or the truck has to start the execution of the plan step before the point in time specified (weak interpretation; type = city).

type The type of a plan step may have the value city or tour. This distinction became necessary because a truck has to plan activities in cities, which need some amount of time. To be able to represent this intervals in time the plan steps of type city were introduced.

Let $P = (p_1, \dots, p_n)$ be a plan then the following assertions must be valid:

$$p_1.\text{type} = \text{city} \wedge p_n.\text{type} = \text{city} \tag{1}$$
$$\forall i \in I\!N : 1 < i < n : p_i.\text{type} = \text{tour} \Rightarrow p_{i-1}.\text{type} = \text{city} \wedge p_{i+1}.\text{type} = \text{city} \tag{2}$$
$$\forall i \in I\!N : 1 \leq i < n : p_i.\text{type} = \text{city} \Rightarrow p_{i+1}.\text{type} = \text{tour} \tag{3}$$
$$\forall i \in I\!N : 1 \leq i \leq n : p_i.\text{type} = \text{city} \Rightarrow p_i.\text{from} = p_i.\text{to} \tag{4}$$
$$\forall i \in I\!N : 1 \leq i \leq n : p_i.\text{type} = \text{tour} \Rightarrow p_i.\text{from} \neq p_i.\text{to} \tag{5}$$

An important thing to notice is that only for loading and unloading in the cities earliest start times and due dates are specified. Earliest start times and due dates for plan steps of type tour must therefore be derived from the plan steps of type city. This is done by propagating the restrictions for the earliest start times from the beginning of the plan to its end and the restrictions for the due dates from the end of the plan to its start. When this is done the following consistency assertions must hold:

$$\forall i \in I\!N : 1 \leq i \leq n : \begin{cases} p_i.\text{est} + p_i.\text{duration} \leq p_i.\text{lft} & \text{if } p_i.\text{type} = \text{tour} \\ p_i.\text{est} \leq p_i.\text{lft} & \text{if } p_i.\text{city} = \text{city} \end{cases} \tag{6}$$
$$\forall i \in I\!N : 1 \leq i < n : p_i.\text{est} + p_i.\text{duration} \leq p_{i+1}.\text{est} \tag{7}$$
$$\forall i \in I\!N : 1 < i \leq n : \begin{cases} pi - 1.\text{dda} \leq p_i.\text{dda} - p_i.\text{duration} & \text{if } p_i.\text{type} = \text{tour} \\ pi - 1.\text{dda} = p_i.\text{dda} & \text{if } p_i.\text{type} = \text{city} \end{cases} \tag{8}$$

These consistency conditions are exactly the conditions which have to be checked to decide if a specific extension of a currently valid plan fulfills the time constraints.

Definition 3 *A plan P is a* valid plan with respect to its time constraints *if it is a valid plan and additionally satisfies the equations (1), (2), (3), (4), (5), (6), (7), and (8). As a short hand we define:*

$$\text{time-valid}(P) \textit{iff } P \textit{ is valid with respect to its time constraint.}$$

For reasons of efficiency it is important that for a given extension the time constraints can be checked by propagating the earliest start time restrictions specified with the new order from the first plan step modified by the extension to the end of the extended plan. The same is true if the restrictions derived from the due dates are propagated from the last modified step to the beginning of the extended plan. The important point is that one direction is sufficient and that no solutions are lost by treating the time constraints in this manner.

Because only one plan step represents all the loading and unloading activities in a city, to guarantee correctness, in our case, the most constraining restriction must be used for planning. This means that valid solutions may be lost. On the one hand this was done to simplify the planning procedure. On the other hand, one has to notice, that in practice normally only the earliest start times and the due dates for loading and unloading are known exactly because of opening times of the client's office. The duration of an activity is not known exactly and can only be estimated. Therefore, a planning style which is very tight with respect to the time constraints does is not reasonable in practice. Another reason for using just one plan step to represent all the activities within one city is that this single step may represent a whole subschedule for all the activities to be scheduled in this city. This would just result in a more complicated planning procedure which would be somewhat more difficult to implement. For the examples we have tested up to now, there was no need for a detailed scheduling of the activities within a city. This is also true for the benchmark tests we report on in section 9.

When time constraints are specified, it is no longer possible to compute the costs of a plan extension only by the detour attached with it. In doing so, what could happen is that a truck would go quickly to the destination of an order just waiting there for the time when it is allowed to deliver the order. Time spent on waiting is as expensive as time spent on driving! Therefore, we want now derive a selection strategy for the trucks which takes this fact into account in a natural manner.

Definition 4 *Let $P = p_1 \dots p_n, n \in I\!N$, with* time-valid*(P). Function*

$$\text{gap}(p_i) =_{def} p_i.est - p_{i-1}.est - p_{i-1}.duration, 1 < i <= n$$

$$\text{gap}(p_1) = 0$$

specifies the waiting time of plan step p_i.

$$\text{gap}^*(P) = \sum_{i=1}^{n} \text{gap}(p_i)$$

specifies the waiting time of the whole plan P.

Note, that

$$\forall P = p_1 \dots p_n \in \mathcal{P} : \mathit{time\text{-}valid}(P) : \forall i \in I\!N : 1 < i < n : p_i.\text{type} = \mathit{tour} \Rightarrow \mathit{gap}(p_i) = 0$$

because the earliest start time of a plan step of type tour is derived from the earliest start time of an order and therefore of a plan step of type city.

We now assume that the function *ext* is extended to enumerate all plan extensions which are valid with respect to their time constraints. Then $\mathcal{E}(t, P, o)$ is also well-defined for plans with time constraints. We now define

$$\tilde{\mathcal{B}}(t, P, o) =$$

$$\{(\hat{P}, \hat{a}) | \nexists (\tilde{P}, \tilde{a}) \in \mathcal{E}(t, P, o) : \mathit{compare}(\frac{\mathit{length}(\tilde{P})}{t.\text{speed}} + \mathit{gap}^*(\tilde{P}), \tilde{a}, \frac{\mathit{length}(\hat{P})}{t.\text{speed}} + \mathit{gap}^*(\hat{P}), \hat{a}) = \mathit{true}\}$$

If $\mathit{card}(\tilde{\mathcal{B}}(t, P, o)) > 1$, then one of the elements of $\tilde{\mathcal{B}}(t, P, o)$ can be chosen freely as a bid to be sent to the shipping company. This is the strategy for selecting the best plan extension in a truck. It produced the results presented in the next section.

9 Results from a Benchmark Test

In order to evaluate the performance of our implementation, we ran extensive test series with benchmark data developed by [Desrochers et al. 92] at MIT. Up to now, this is the only test data we could get from the outside and which gave us the possibility to compare the performance of our system in an objective manner. Looking at the results, we have to stress that the problem which is given by the test data does not challenge the full power of our system. In the test data, there is only one depot from where a set of clients has to be served. In each example there are 100 orders for 100 clients where no client occurs twice. In the test data, it is assumed that only unloading at the location of the client does need time. There are no time restrictions specified for the process of loading a truck. Moreover, there is only a single transportation company modeled. Finally, it is assumed that there is always a straight line connection between two cities.

The problems which can be solved by our system are more general. Time restrictions may be specified for loading and unloading the order. An order may have any city as source or destination. Last not least our system is designed to solve an *open planning problem* where at any point in time new orders may be given to the system which has to react to them and find a good solution for the currently valid situation.

Optimal solutions can in general only be computed if a problem is treated as a closed planning problem. In this case, when the planning processes is started all input data must be known. Throughout the planning process the input data is not allowed to change. It is clear that for the benchmark given by [Desrochers et al. 92] algorithms exist which perform more efficient than our system for this specific problem, but these algorithms will not be able to solve the more general problem our system is able to. Even though it was very interesting to find out how good our system was able to solve the benchmark problem. Because these solution was found straightforward using the problem solving techniques described in this report, it is very likely that we will be able to find additional cooperation strategies between the truck agents, between truck agents and shipping company agents, and between shipping company agents which will even increase our already good results.

The following table shows the raw data (for 5 out of 27 test sets) we got when we ran the test series. A Column with the entry *yes* in the row *Sorted* represents the result for a test where a preprocessing of the input data was done, i.e. sorting with respect to the earliest start times. If this preprocessing is done the problem is seen as a closed planning problem because all orders have to be known at the time when they have to be sorted. The differences in the test runs if the input is sorted or not gives us an estimation how the quality of the results our system is abel to produce will change if we look at a the closed planning problem or at the open case. One of our overall goals is to find cooperation techniques between the agents of the application domain, which will bring the results of the open planning problem as close to the results we can get looking at the closed planning problem where any (pre-)processing of the whole input data set is allowed to get a solution which is as close as possible to the globally optimal solution.

Test Data	Number of Orders	Number of Trucks	Distance	Time Needed	Waiting	Input Sorted
R102	25	7	614	1309	445	yes
R102	50	12	1250	2247	497	yes
R102	100	20	1961	3714	753	yes
R102	100	23	2392	4181	789	no
R104	25	5	564	1036	222	yes
R104	50	9	1098	1834	236	yes
R104	100	16	1646	3269	613	yes
R104	100	19	2016	3395	379	no
R105	25	7	681	1212	281	yes
R105	50	10	1288	1919	131	yes
R105	100	17	1988	3315	327	yes
R105	100	20	2459	3797	338	no
R106	25	7	720	1359	389	yes
R106	50	10	1288	1919	131	yes
R106	100	18	1959	3476	507	yes
R107	25	5	598	1028	180	yes
R107	50	9	1182	1829	147	yes
R107	100	17	2011	3283	272	no
R108	25	5	651	1041	140	yes
R108	50	9	1081	1784	203	yes
R108	100	14	1530	2872	332	yes
R108	100	17	1905	3190	285	no

To make it possible for a reader to judge the quality of these results, we present the table which was presented in [Desrochers et al. 92] for these 5 examples. If an entry is marked with '-' then until now the globally optimal solution is not known. Unfortunately, we do not know the best suboptimal solution which has ever been found for these examples.

Test Data	Number of Orders	Number of Trucks	Distance
R102	25	7	546.4
R102	50	11	904.6
R102	100	17	1434.0
R104	25	4	416.9
R104	50	-	-
R104	100	-	-
R105	25	6	526.0
R105	50	9	891.7
R105	100	-	-
R106	25	5	457.3
R106	50	8	783.3
R106	100	-	-
R107	25	4	423.0
R107	50	7	703.2
R107	100	-	-
R108	25	4	396.2
R108	50	-	-
R108	100	-	-

When looking more closely at the data, it is very surprising, that the examples which seem to be hard for the algorithm presented in [Desrochers et al. 92] (e.g. R104 and R108 because only the first 25 out of the 100 orders could be solved optimally) seem to be the easy ones for our system. To make this point clear, we do not believe that the solution which was found in these cases is closer to the optimal solution than in the other examples — nor do we believe that the solution is farer away from the optimal solution than in the other cases. What we want to stress is that these cases are better solutions in quality because there are less trucks driving, what will result in a better capacity utilization. Moreover, the overall distance which has to be driven by the trucks is smaller, there is less waiting time, and as a result the time needed for the execution of the whole set of orders is smaller. Therefore, the plans which were derived for these cases can be judged to be better than the plans found for the other examples.

One should be aware, that due to the NP-completeness of the problem for any algorithm which guarantees to find the optimal solution for any instance of these problems, there is an instance of such a problem which will result in an exponential run-time of the algorithm when it tries to solve this problem. It is not clear if such problems will occur in practice. Neither is it clear if the problem set specified by [Desrochers et al. 92] is relevant to judge the practical applicability of an algorithm which tries to tackle these problems in real life. If we look in to the real world domain, we see that plan execution is done with high uncertainties. Sometimes, planning has do be done with incomplete knowledge and on the basis of data which contain errors. All this makes a very tight planning rather doubtful. Our opinion is that for practical applications a system which is able to cope with the dynamics and the uncertainties of the real world environment is needed. We think that the system we have built up is a big step in this direction.

To conclude this section we want to stress the point that we did not built up our system looking at this specific problem and trying to solve this specific problem optimally. What we did was looking at the domain of the application, extracting knowledge about entities in this real world application, and modeling these entities in our system in a natural manner. We are pleased that our system is already now able to produce results of the quality which was shown above. We belive that we will be able to enhance the quality of the results if we put more emphasis on the horizontal cooperation between the shipping company agents on one hand and between the truck agents on the other hand because the results which were presented in this section were produced by a pure hierarchical approach with no horizontal cooperation between shipping companies and trucks.

10 Conclusion

In this report, we discussed different mechanisms that can be used to implement task allocation and task decomposition processes in multi-agent systems and we described how they can be integrated in MARS, a multi-agent system to implement a scenario of distributed shipping companies. The MARS system shows that the multi-agent approach results in a modeling for this domain that allows to reflect to a large extent the real-world situation and thus, that is very natural. To enforce this effect, we have chosen two types of agents, namely the company agents and the truck agents that have to cooperate in different manners. According to the hierarchical relationship between the agents due to the different roles they play in the modeling we distinguished between *vertical cooperation*, denoting the cooperation between the truck agents and their company agents and *horizontal cooperation* which occurs between different company agents.

As an example of a cooperation protocol involving both of these forms of cooperation we presented the protocol for the unbooked leg cooperation. This shows how we can construct new and powerful cooperation mechanisms through the combination of simple cooperation protocols, e.g. two contract net protocols. Furthermore, it motivates the investigation of other simple cooperation protocols and the research for principles of how to choose protocols out of a library of generic protocols in order to obtain complex cooperation mechanisms that can be used to implement a specific form of interaction between agents. This techniques can also improve the flexibility of a multi-agent modeling for a specific application when the agents are able to choose among different protocols the one which seems to be the best for the current interaction.

One major goal within our project is to evaluate the applicability of DAI techniques to real-world applications. One of the domains we consider therefore is the domain of distributed shipping companies discussed in this report. Part of this evaluation task is the comparison of implementations based on DAI methods to other ones using a different paradigm. For the domain of the route planning, which is explicitly contained in the MARS scenario there exists a set of benchmark tests developed by Solomon and which was described in section 9. Although these test sets are designed for the evaluation of systems dealing with the static scheduling problem where the set of orders that have to be scheduled is known in advance and does not change during the scheduling process, we decided to take the opportunity and ran our system with these sets.
Looking at the results of these tests, we see that our system did not succeed in computing the optimal solutions for those examples where Solomon's system did. However, even this implementation could find optimal solutions only for 7 out of 29 instances. Moreover, we can claim that our results are not too far away from Solomon's. Furthermore, we obtained

reasonable solutions for all examples of the test sets.

Two things are important in the comparison of the two approaches: firstly, the benchmark examples start from a single depot. Thus, they test cover only a very special case of the problem class we are able to cope with the MARS system, where orders may occur between arbitrary locations. Secondly, and most important, our system is able to deal with the class of dynamic distributed scheduling problems. When dealing with these problems, also the mechanisms based on the horizontal cooperation can be exploited. The restricted nature of the benchmark test did not allow us to make use of this feature in the benchmark test. Unfortunately, for this class of application, we have no opportunity to run benchmark tests nor is there a system available that we could use for comparative purposes.

References

[Bachem et al. 92] A. **Bachem**, W. **Hochstättler**, and M. **Malich**. *Simulated Trading: A New Approach For Solving Vehicle Routing Problems.* Technical Report 92.125, Mathematisches Institut Universität zu Köln, Dezember 1992.

[Bagchi & Nag 91] P. **Bagchi** and B. **Nag**. *Dynamic Vehicle Scheduling: An Expert System Approach.* Journal of Physical Distribution and Logistics Management, 21(2), 1991.

[Bauer & Wössner 81] F.L. **Bauer** and H. **Wössner**. *Algorithmische Sprache und Programmentwicklung.* Berlin, Heidelberg, New York: Springer-Verlag, 1981.

[Blamauer 83] Manfred **Blamauer**. *Just im Stau - Radikales Umdenken notwenig.* Jahrbuch der Logistik, 1983.

[Bodin 83] H. **Bodin**. *Routing and Scheduling of Vehicles and Crews.* Computers and OR Research, 10, 1983. Special Issue.

[Bond & Gasser 88] A. **Bond** and L. **Gasser**. *Readings in Distributed Artificial Intelligence.* Morgan Kaufmann, Los Angeles, CA, 1988.

[Brooks 86] Rodney A. **Brooks**. *A Robust Layered Control System for a Mobile Robot.* In: IEEE Journal of Robotics and Automation, volume RA-2 (1), pp. 14–23, April 1986.

[Bürckert & Müller 91] H.-J. **Bürckert** and H. J. **Müller**. *RATMAN: Rational Agents Testbed for Multi-Agent Networks.* In: Y. Demazeau and J.-P. Müller (eds.), Decentralized A. I., volume 2, pp. 217–230. North-Holland, 1991. Also published in the Proceedings of MAAMAW-90.

[Bussmann 92] S. **Bussmann**. *Simulation Environment for Multi-Agent Worlds.* Document D-92-01, DFKI, Saarbrücken, January 1992.

[Cammarata et al. 88] S. **Cammarata**, F. **Hayes-Roth**, R. **Steeb**, P. **Thorndyke**, and R. **Wesson**. *Architectures for Distributed Air-Traffic Control.* In: A. H. Bond and L. Gasser (eds.), Readings in Distributed Artificial Intelligence, pp. 102–105. San Mateo, CA: Morgan Kaufmann Publishers, Inc., 1988.

[Corkill & Lesser 88] Daniel D. **Corkill** and Victor R. **Lesser**. *The Distributed Vehicle Monitoring Testbed: A Tool for Investigating Distributed Problem Solving Networks.* In: Robert Engelmore and Tony Morgan (eds.), Blackboard Systems. Addison-Wesley Publishing Company, 1988.

[Cormen et al. 92] Thomas H. **Cormen**, Charles E. **Leiserson**, and Ronald L. **Rivest**. *Algorithms.* Cambridge, Massachusets: The MIT Press, 1992.

[Davis & Smith 83] R. **Davis** and R. G. **Smith**. *Negotiation as a metaphor for distributed problem solving.* Artificial Intelligence, 20:63 – 109, 1983.

[Desrochers et al. 92] M. **Desrochers**, J. **Desrosiers**, and M. **Solomon**. *A New Optimization Algorithm for the Vehicle Routing Problem with Time Windows.* Operations Research, 40(2), 1992.

[Durfee & Lesser 89] E. H. **Durfee** and V. R. **Lesser**. *Negotiating Task Decomposition and Allocation Using Partial Global Planning.* In: Distributed Artificial Intelligence, Volume II, pp. 229–244, San Mateo, CA, 1989. Morgan Kaufmann Publishers, Inc.

[Durfee & Montgomery 90] E. H. **Durfee** and T. A. **Montgomery**. *A Hierarchical Protocol for Coordination of Multiagent Behaviour.* In: Proc. of the 8th National Conference on Artificial Intelligence, pp. 86–93. Boston, MA, 1990.

[Ermann et al. 80] L.D. **Ermann**, F. **Hayes-Roth**, V.R. **Lesser**, and **D.R.Reddy**. *The HEARSAY-II speech understanding system: Integrating Knowledge to resolve uncertainty.* In: Computing Surveys 12(2), pp. 213–253, 1980.

[Falk et al. 93] J. **Falk**, S. **Spieck**, and P. **Mertens**. *Unterstützung der Lager- und Transportlogistik durch Teilintelligente Agenten.* Information Management, 2, 1993.

[Fischer 93] K. **Fischer**. *The Rule-based Multi-Agent System MAGSY.* In: Proceedings of the CKBS'92 Workshop. Keele University, 1993.

[Garey & Johnson 79] M. R. **Garey** and D. S. **Johnson**. *Computers and Intractability - A Guide to the Theory of NP-Completeness.* W.H. Freeman, 1979.

[Golden & Assad 83] B. **Golden** and A. **Assad**. *Vehicle Routing: Methods and Studies.* Studies in Management Science and Systems. North Holland, 1983.

[Haddadi & Sundermeyer 93] A. **Haddadi** and K. **Sundermeyer**. *Acquaintance Relations in Autonomous Agents Societies.* In: Proceedings of the International Symposium on Autonomous Decentralized Systems (ISADS-93), Tokyo, Japan, 1993.

[Henz et al. 93] M. **Henz**, G. **Smolka**, and J. **Würtz**. *Oz - A Programming Language for Multi-Agent Systems.* In: Proceedings of the 13th International Joint Conference on Artificial Intelligence, pp. 404–409, Chambery, 1993. Morgan Kaufmann Publishers, Inc.

[Hewitt 73] C. **Hewitt**. *A Universal Modular ACTOR Formalism for AI.* In: Proceedings of IJCAI-73, pp. 235–245. Morgan Kaufmann, 1973.

[Hewitt 77] C. E. **Hewitt**. *Viewing Control Structures as Patterns of Passing Messges.* Artificial Intelligence, 8(3):323–364, 1977.

[Hopcroft & Ullman 79] J. **Hopcroft** and J. **Ullman**. *Introduction to Automata Theory, Languages, and Computation.* Addison-Wesley, 1979.

[Jennings 92] N. R. **Jennings**. *Joint Intentions as a Model of Multi-Agent Cooperation.* PhD thesis, Queen Mary and Westfield College, London, August 1992.

[Klein 90] M. **Klein**. *A Computational Model of Conflict Resolution in Cooperative Design Systems.* In: Proc. of the 10th International Workshop on DAI, Austin, Texas, 1990. MCC.

[Kreifelts & v. Martial 91] T. **Kreifelts** and F. **v. Martial**. *A Negotiation Framework for Autonomous Agents.* In: Y. Demazeau and J.-P. Müller (eds.), Decentralized A.I., volume 2. North-Holland, 1991.

[Kuhn et al. 93a] N. **Kuhn**, H. J. **Müller**, and J. P. **Müller**. *Simulating Cooperative Transportation Companies.* In: Proceedings of the European Simulation Multiconference (ESM-93), Lyon, France, June 1993. Society for Computer Simulation.

[Kuhn et al. 93b] N. **Kuhn**, H. J. **Müller**, and J. P. **Müller**. *Task Decomposition in Dynamic Agent Societies.* In: Proceedings of the International Symposium on Autonomous Decentralized Systems (ISADS-93), Tokyo, Japan, 1993. IEEE Computer Society Press.

[Lenat 75] D. B. **Lenat**. *BEINGS: Knowledge as Interacting Experts.* In: Proceedings of IJCAI-75. Morgan Kaufmann, 1975.

[Lux et al. 92] A. **Lux**, F. **Bomarius**, and D. **Steiner**. *A Model for Supporting Human Computer Cooperation.* In: AAAI Workshop on Cooperation among Heterogeneous Intelligent Systems, July 1992.

[Müller & Pischel 93a] J. P. **Müller** and M. **Pischel**. *Modelling Robot Societies using* InteRRaP. In: IJCAI Workshop on Dynamically Interacting Robots, Chambery, France, August 1993.

[Müller & Pischel 93b] J. P. **Müller** and M. **Pischel**. *The Agent Architecture* InteRRaP*: Concept and Application.* Research Report RR-93-26, German Artificial Intelligence Research Center (DFKI), Saarbrücken, June 1993.

[Müller-Merbach 70] H. **Müller-Merbach**. *Optimale Reihenfolgen (Optimal Sequences).* Springer Verlag, 1970.

[Müller 93] H. J. **Müller** (ed.). *Verteilte Künstliche Intelligenz — Methoden und Anwendungen (Distributed Artificial Intelligence — Methods and Applications).* BI-Verlag, 1993.

[Parunak 87] H.V.D. **Parunak**. *Manufacturing experience with the contract net.* In: M.N. Huhns (ed.), Distributed Artificial Intelligence, pp. 285 – 310. San Mateo, CA: Morgan Kaufmann Publishers, 1987.

[Psaraftis 88] H. **Psaraftis**. *Dynamic Vehicle Routing Problems.* In: Vehicle Routing: Methods and Studies. North-Holland, 1988.

[Rittmann 91] R. **Rittmann**. *Die Macht der Trucks.* Bild der Wissenschaft, 9:112–114, 1991.

[Russ & Vierke 93] C. **Russ** and G. **Vierke**. *Ein Kooperationsmodell für das Speditionsszenario.* Bericht des Fortgeschrittenenpraktikums VKI, Universität Saarbrücken, 1993.

[Schupeta 92] A. **Schupeta**. *COSMA: Ein verteilter Terminplaner als Fallstudie der Verteilten KI.* In: H. J. Müller and D. Steiner (eds.), Workshop Kooperierende Agenten, pp. 50–55. DFKI Document D-92-24, 1992.

[Smith 80] R.G. **Smith**. *The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver.* In: IEEE Transaction on Computers, number 12 in C-29, pp. 1104–1113, 1980.

[Steels 90] L. **Steels**. *Cooperation Between Distributed Agents Through Self-Organization.* In: Y. Demazeau and J.-P. Müller (eds.), Decentralized A.I., pp. 175–196. North-Holland, 1990.

[Suchman 87] L. A. **Suchman**. *Plans and Situated Actions.* Cambridge Universtiy Press, Cambridge, 1987.

[Sycara 88] K. P. **Sycara**. *Resolving Goal Conflicts via Negotiation.* In: Proceedings of the 7th National Conference on Artificial Intelligence, pp. 245–250, St. Paul, Minnesota, August 1988.

[Sycara 89] K. P. **Sycara**. *Multiagent Compromise via Negotiation.* In: L. Gasser and M. N. Huhns (eds.), Distributed Artificial Intelligence, Volume II, pp. 119–137. San Mateo, California: Morgan Kaufmann, 1989.

[Zlotkin & Rosenschein 91] G. **Zlotkin** and J. S. **Rosenschein**. *Negotiation and Goal Relaxation.* In: Y. Demazeau and J.-P. Müller (eds.), Decentralized A.I., volume 2, pp. 273–286. North-Holland, 1991.

11 The Example R102

In this section we present one of the examples out of the benchmark defined by [Desrochers et al. 92]. We chose the example R102 because because only in this case the globally optimal solution is already known for all cases: viz. 25, 50 and 100 orders. We want to point out, that this example is for our system one of the hardest ones in the test series what can be seen in the table shown in section 9. We take these hard problems as a challenge to find out concepts to improve the overall behaviour of our system. This example gives a flavour of how far we did already go.

The example is presented in 5 parts. First of all the set of orders the system has to execute is shown. Each order is represented by a list of 13 symbols:

1. The symbol *order* specifies that the sequence specifying the order does start with this symbol.

2. The name of the city in which the order starts. Due to the specification of the problem this is always the symbol *DEPOT*.

3. The name of the city in which the order ends.

4. The type of the goods which have to be transported. Here is always the symbol *paletten* specified. In fact, the planning procedure does unitil now not really care about this symbol.

5. The amount of goods which has to be transported.

6. The earliest start time of the loading of the truck (always 0).

7. The duration of the loading of the truck (always 0).

8. The due date for loading the truck (always the same as the due date for unloading the truck).

9. The earliest start time for the process of unloading.

10. The duration of the process of unloading (some times also called: service time).

11. The due date for the process of unloading the truck.

12. A numer which specifies the highes price which will be accepted for the order. It is set to a value that does not influence the planning process.

13. The name of the shipping company which should execute the order. Obviously there is only one shipping company needed and this one is called *PFALZEXPRESS*.

After the set of orders the plans which were produced by the system are shown in the sequence 25, 50 and 100 orders in any case the input is assumed to be sorted with respect to the earliest start time of the order. The last example show the plans which were found with the originally unsorted set of 100 orders. In this last case the problem viewed as an open planning problem. At the end of each example a summary is given in a table.

25 Orders (Sorted Input)

PFA14 TRUCK/MOD-SERV/1605 0					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	30	0.0000000000
DEPOT	C_22	0	18	48	69.000000000
C_22		18	10	48	58.000000000
C_22	C_3	28	10	58	58.000000000
C_3		38	10	58	51.000000000
C_3	C_16	48	13	71	51.000000000
C_16		61	10	71	43.000000000
C_16	C_23	71	15	107	43.000000000
C_23		97	10	107	25.000000000
C_23	C_5	107	14	159	25.000000000
C_5		149	10	159	6.0000000000
C_5	C_26	159	10	182	6.0000000000
C_26		172	10	182	0.0000000000
C_26	DEPOT	182	33	230	0.0000000000
order DEPOT C_26 6.0000000000 0 0 182 172 10 182					
order DEPOT C_5 19.000000000 0 0 159 149 10 159					
order DEPOT C_23 18.000000000 0 0 107 97 10 107					
order DEPOT C_16 8.0000000000 0 0 71 61 10 71					
order DEPOT C_22 11.000000000 0 0 201 0 10 201					
order DEPOT C_3 7.0000000000 0 0 202 0 10 202					
Total time needed:			215		
Waiting time:			42		
Maximal gap:			28		
Tatal distance to go:			113		

PFA17 TRUCK/MOD-ELC3/4886 0					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	10	0.0000000000
DEPOT	C_15	0	32	42	75.000000000
C_15		32	10	42	55.000000000
C_15	C_17	42	11	85	55.000000000
C_17		75	10	85	36.000000000
C_17	C_7	85	18	109	36.000000000
C_7		103	10	109	33.000000000
C_7	C_6	113	10	157	33.000000000
C_6		123	10	157	7.0000000000
C_6	C_18	133	10	167	7.0000000000
C_18		157	10	167	5.0000000000
C_18	C_8	167	25	198	5.0000000000
C_8		192	10	198	0.0000000000
C_8	DEPOT	202	21	230	0.0000000000
order DEPOT C_18 2.0000000000 0 0 167 157 10 167					
order DEPOT C_7 3.0000000000 0 0 109 99 10 109					
order DEPOT C_17 19.000000000 0 0 85 75 10 85					
order DEPOT C_15 20.000000000 0 0 42 32 10 42					
order DEPOT C_8 5.0000000000 0 0 198 0 10 198					
order DEPOT C_6 26.000000000 0 0 199 0 10 199					
Total time needed:			223		
Waiting time:			36		
Maximal gap:			22		
Tatal distance to go:			127		

PFA19 TRUCK/MOD-ELC3/4881 0					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	79	0.0000000000
DEPOT	C_9	0	26	105	51.000000000
C_9		95	10	105	42.000000000
C_9	C_2	105	31	180	42.000000000
C_2		136	10	180	32.000000000
C_2	C_4	146	14	194	32.000000000
C_4		160	10	194	19.000000000
C_4	C_13	170	11	205	19.000000000
C_13		181	10	205	0.0000000000
C_13	DEPOT	191	15	230	0.0000000000
order DEPOT C_9 9.0000000000 0 0 105 95 10 105					
order DEPOT C_13 19.000000000 0 0 205 0 10 205					
order DEPOT C_4 13.000000000 0 0 197 0 10 197					
order DEPOT C_2 10.000000000 0 0 204 0 10 204					
Total time needed:			206		
Waiting time:			69		
Maximal gap:			69		
Tatal distance to go:			97		

PFA21 TRUCK/MOD-ELC1/2979 0					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	44	0.0000000000
DEPOT	C_12	0	33	77	68.000000000
C_12		67	10	77	56.000000000
C_12	C_20	77	7	86	56.000000000
C_20		84	10	86	39.000000000
C_20	C_11	94	15	134	39.000000000
C_11		124	10	134	23.000000000
C_11	C_14	134	35	169	23.000000000
C_14		169	10	169	0.0000000000
C_14	DEPOT	179	11	230	0.0000000000
order DEPOT C_14 23.000000000 0 0 169 159 10 169					
order DEPOT C_11 16.000000000 0 0 134 124 10 134					
order DEPOT C_20 17.000000000 0 0 86 76 10 86					
order DEPOT C_12 12.000000000 0 0 77 67 10 77					
Total time needed:			190		
Waiting time:			49		
Maximal gap:			34		
Tatal distance to go:			101		

PFA24 TRUCK/MOD-ELC1/2973 1					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	75	0.0000000000
DEPOT	C_10	0	32	107	16.000000000
C_10		97	10	107	0.0000000000
C_10	DEPOT	107	32	230	0.0000000000
order DEPOT C_10 16.000000000 0 0 107 97 10 107					
Total time needed:			139		
Waiting time:			65		
Maximal gap:			65		
Tatal distance to go:			64		

PFA27 TRUCK/MOD-ELC2/2738 0					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	82	0.0000000000
DEPOT	C_19	0	15	97	21.000000000
C_19		87	10	97	9.0000000000
C_19	C_21	97	35	136	9.0000000000
C_21		132	10	136	0.0000000000
C_21	DEPOT	142	31	230	0.0000000000
order DEPOT C_21 9.0000000000 0 0 136 126 10 136					
order DEPOT C_19 12.000000000 0 0 97 87 10 97					
Total time needed:			173		
Waiting time:			72		
Maximal gap:			72		
Tatal distance to go:			81		

PFA29 TRUCK/MOD-ELC2/2734 0					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	42	0.0000000000
DEPOT	C_24	0	36	78	32.000000000
C_24		68	10	78	3.0000000000
C_24	C_25	78	31	163	3.0000000000
C_25		153	10	163	0.0000000000
C_25	DEPOT	163	30	230	0.0000000000
order DEPOT C_25 3.0000000000 0 0 163 153 10 163					
order DEPOT C_24 29.000000000 0 0 78 68 10 78					
Total time needed:			193		
Waiting time:			76		
Maximal gap:			44		
Tatal distance to go:			97		

Name:	Dist.:	Wait:	Time:	Stops:
PFA29	97	76	193	2
PFA17	127	36	223	6
PFA19	97	69	206	4
PFA27	81	72	173	2
PFA14	113	42	215	6
PFA24	64	65	139	1
PFA21	101	49	190	4
$\sum$	680	409	1339	25

50 Orders (Sorted Input)

PFA12 TRUCK/MOD-SERV/1437 1					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	112	0.0000000000
DEPOT	C_36	0	41	153	8.0000000000
C_36		143	10	153	0.0000000000
C_36	DEPOT	153	41	230	0.0000000000
order DEPOT C_36 8.0000000000 0 0 153 143 10 153					
Total time needed:	194				
Waiting time:	102				
Maximal gap:	102				
Tatal distance to go:	82				

PFA14 TRUCK/MOD-SERV/1431 1					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	51	0.0000000000
DEPOT	C_39	0	42	93	16.000000000
C_39		83	10	93	0.0000000000
C_39	DEPOT	93	42	230	0.0000000000
order DEPOT C_39 16.000000000 0 0 93 83 10 93					
Total time needed:	135				
Waiting time:	41				
Maximal gap:	41				
Tatal distance to go:	84				

PFA17 TRUCK/MOD-ELC3/4852 0					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	10	0.0000000000
DEPOT	C_15	0	32	42	67.000000000
C_15		32	10	42	47.000000000
C_15	C_16	42	15	71	47.000000000
C_16		61	10	71	39.000000000
C_16	C_41	71	22	95	39.000000000
C_41		93	10	95	30.000000000
C_41	C_44	103	27	142	30.000000000
C_44		132	10	142	23.000000000
C_44	C_14	142	23	169	23.000000000
C_14		165	10	169	0.0000000000
C_14	DEPOT	175	11	230	0.0000000000
order DEPOT C_14 23.000000000 0 0 169 159 10 169					
order DEPOT C_44 7.0000000000 0 0 142 132 10 142					
order DEPOT C_41 9.0000000000 0 0 95 85 10 95					
order DEPOT C_16 8.0000000000 0 0 71 61 10 71					
order DEPOT C_15 20.000000000 0 0 42 32 10 42					
Total time needed:	186				
Waiting time:	6				
Maximal gap:	4				
Tatal distance to go:	130				

PFA18 TRUCK/MOD-ELC3/4850 0					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	16	0.0000000000
DEPOT	C_43	0	25	41	144.00000000
C_43		31	10	41	139.00000000
C_43	C_45	41	13	79	139.00000000
C_45		69	10	79	121.00000000
C_45	C_17	79	6	85	121.00000000
C_17		85	10	85	102.00000000
C_17	C_38	95	10	151	102.00000000
C_38		105	10	151	94.000000000
C_38	C_6	115	11	162	94.000000000
C_6		126	10	162	68.000000000
C_6	C_49	136	22	184	68.000000000
C_49		158	10	184	32.000000000
C_49	C_8	168	7	191	32.000000000
C_8		175	10	191	27.000000000
C_8	C_32	185	11	202	27.000000000
C_32		196	10	202	0.0000000000
C_32	DEPOT	206	17	230	0.0000000000
order DEPOT C_17 19.000000000 0 0 85 75 10 85					
order DEPOT C_45 18.000000000 0 0 79 69 10 79					
order DEPOT C_43 5.0000000000 0 0 41 31 10 41					
order DEPOT C_49 36.000000000 0 0 192 0 10 192					
order DEPOT C_38 8.0000000000 0 0 198 0 10 198					
order DEPOT C_32 27.000000000 0 0 202 0 10 202					
order DEPOT C_8 5.0000000000 0 0 198 0 10 198					
order DEPOT C_6 26.000000000 0 0 199 0 10 199					
Total time needed:			223		
Waiting time:			21		
Maximal gap:			15		
Tatal distance to go:			122		

PFA19 TRUCK/MOD-ELC3/4848 1					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	21	0.0000000000
DEPOT	C_40	0	33	54	87.000000000
C_40		44	10	54	56.000000000
C_40	C_24	54	8	78	56.000000000
C_24		68	10	78	27.000000000
C_24	C_23	78	11	107	27.000000000
C_23		97	10	107	9.0000000000
C_23	C_25	107	32	163	9.0000000000
C_25		153	10	163	6.0000000000
C_25	C_26	163	15	182	6.0000000000
C_26		178	10	182	0.0000000000
C_26	DEPOT	188	33	230	0.0000000000
order DEPOT C_26 6.0000000000 0 0 182 172 10 182					
order DEPOT C_25 3.0000000000 0 0 163 153 10 163					
order DEPOT C_23 18.000000000 0 0 107 97 10 107					
order DEPOT C_24 29.000000000 0 0 78 68 10 78					
order DEPOT C_40 31.000000000 0 0 54 44 10 54					
Total time needed:			221		
Waiting time:			39		
Maximal gap:			14		
Tatal distance to go:			132		

PFA20 TRUCK/MOD-ELC2/2689 0					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	42	0.0000000000
DEPOT	C_28	0	5	47	123.00000000
C_28		37	10	47	107.00000000
C_28	C_2	47	10	70	107.00000000
C_2		57	10	70	97.000000000
C_2	C_31	67	11	81	97.000000000
C_31		78	10	81	76.000000000
C_31	C_51	88	14	157	76.000000000
C_51		102	10	157	63.000000000
C_51	C_35	112	19	176	63.000000000
C_35		131	10	176	49.000000000
C_35	C_4	141	14	190	49.000000000
C_4		155	10	190	36.000000000
C_4	C_13	165	11	201	36.000000000
C_13		176	10	201	17.000000000
C_13	C_27	186	7	208	17.000000000
C_27		193	10	208	0.0000000000
C_27	DEPOT	203	11	230	0.0000000000
order DEPOT C_31 21.000000000 0 0 81 71 10 81					
order DEPOT C_28 16.000000000 0 0 47 37 10 47					
order DEPOT C_51 13.000000000 0 0 203 0 10 203					
order DEPOT C_35 14.000000000 0 0 183 0 10 183					
order DEPOT C_27 17.000000000 0 0 208 0 10 208					
order DEPOT C_13 19.000000000 0 0 205 0 10 205					
order DEPOT C_4 13.000000000 0 0 197 0 10 197					
order DEPOT C_2 10.000000000 0 0 204 0 10 204					
Total time needed:			214		
Waiting time:			32		
Maximal gap:			32		
Tatal distance to go:			102		

PFA22 TRUCK/MOD-ELC1/2924 1					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	105	0.0000000000
DEPOT	C_21	0	31	136	9.0000000000
C_21		126	10	136	0.0000000000
C_21	DEPOT	136	31	230	0.0000000000
order DEPOT C_21 9.0000000000 0 0 136 126 10 136					
Total time needed:			167		
Waiting time:			95		
Maximal gap:			95		
Tatal distance to go:			62		

PFA24 TRUCK/MOD-ELC1/2920 0					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	43	0.0000000000
DEPOT	C_29	0	6	49	68.000000000
C_29		39	10	49	52.000000000
C_29	C_19	49	21	97	52.000000000
C_19		87	10	97	40.000000000
C_19	C_7	97	11	109	40.000000000
C_7		108	10	109	37.000000000
C_7	C_5	118	31	159	37.000000000
C_5		149	10	159	18.000000000
C_5	C_22	159	10	192	18.000000000
C_22		169	10	192	7.0000000000
C_22	C_3	179	10	202	7.0000000000
C_3		189	10	202	0.0000000000
C_3	DEPOT	199	18	230	0.0000000000
order DEPOT C_5 19.000000000 0 0 159 149 10 159					
order DEPOT C_7 3.0000000000 0 0 109 99 10 109					
order DEPOT C_19 12.000000000 0 0 97 87 10 97					
order DEPOT C_29 16.000000000 0 0 49 39 10 49					
order DEPOT C_22 11.000000000 0 0 201 0 10 201					
order DEPOT C_3 7.0000000000 0 0 202 0 10 202					
Total time needed:	217				
Waiting time:	50				
Maximal gap:	33				
Tatal distance to go:	107				

PFA25 TRUCK/MOD-ELC1/2917 1					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	10	0.0000000000
DEPOT	C_37	0	41	51	70.000000000
C_37		41	10	51	65.000000000
C_37	C_12	51	18	77	65.000000000
C_12		69	10	77	53.000000000
C_12	C_50	79	14	118	53.000000000
C_50		108	10	118	23.000000000
C_50	C_33	118	29	151	23.000000000
C_33		147	10	151	0.0000000000
C_33	DEPOT	157	34	230	0.0000000000
order DEPOT C_33 23.000000000 0 0 151 141 10 151					
order DEPOT C_50 30.000000000 0 0 118 108 10 118					
order DEPOT C_12 12.000000000 0 0 77 67 10 77					
order DEPOT C_37 5.0000000000 0 0 51 41 10 51					
Total time needed:	191				
Waiting time:	15				
Maximal gap:	15				
Tatal distance to go:	136				

PFA26 TRUCK/MOD-ELC2/2699 0					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	23	0.0000000000
DEPOT	C_34	0	24	47	52.000000000
C_34		37	10	47	41.000000000
C_34	C_30	47	14	73	41.000000000
C_30		63	10	73	32.000000000
C_30	C_10	73	20	107	32.000000000
C_10		97	10	107	16.000000000
C_10	C_11	107	25	134	16.000000000
C_11		132	10	134	0.0000000000
C_11	DEPOT	142	25	230	0.0000000000
order DEPOT C_11 16.000000000 0 0 134 124 10 134					
order DEPOT C_10 16.000000000 0 0 107 97 10 107					
order DEPOT C_30 9.0000000000 0 0 73 63 10 73					
order DEPOT C_34 11.000000000 0 0 47 37 10 47					
Total time needed:			167		
Waiting time:			19		
Maximal gap:			13		
Tatal distance to go:			108		

PFA27 TRUCK/MOD-ELC2/2697 0					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	13	0.0000000000
DEPOT	C_46	0	29	42	72.000000000
C_46		32	10	42	56.000000000
C_46	C_48	42	18	61	56.000000000
C_48		60	10	61	29.000000000
C_48	C_20	70	8	86	29.000000000
C_20		78	10	86	12.000000000
C_20	C_9	88	17	105	12.000000000
C_9		105	10	105	3.0000000000
C_9	C_47	115	9	127	3.0000000000
C_47		124	10	127	2.0000000000
C_47	C_18	134	18	167	2.0000000000
C_18		157	10	167	0.0000000000
C_18	DEPOT	167	30	230	0.0000000000
order DEPOT C_18 2.0000000000 0 0 167 157 10 167					
order DEPOT C_47 1.0000000000 0 0 127 117 10 127					
order DEPOT C_9 9.0000000000 0 0 105 95 10 105					
order DEPOT C_20 17.000000000 0 0 86 76 10 86					
order DEPOT C_48 27.000000000 0 0 61 51 10 61					
order DEPOT C_46 16.000000000 0 0 42 32 10 42					
Total time needed:			197		
Waiting time:			8		
Maximal gap:			5		
Tatal distance to go:			129		

PFA28 TRUCK/MOD-ELC2/2694 1					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	79	0.0000000000
DEPOT	C_42	0	28	107	5.0000000000
C_42		97	10	107	0.0000000000
C_42	DEPOT	107	28	230	0.0000000000
order DEPOT C_42 5.0000000000 0 0 107 97 10 107					
Total time needed:			135		
Waiting time:			69		
Maximal gap:			69		
Tatal distance to go:			56		

Name:	Dist.:	Wait:	Time:	Stops:
PFA17	130	6	186	5
PFA19	132	39	221	5
PFA18	122	21	223	8
PFA28	56	69	135	1
PFA20	102	32	214	8
PFA27	129	8	197	6
PFA14	84	41	135	1
PFA26	108	19	167	4
PFA12	82	102	194	1
PFA25	136	15	191	4
PFA22	62	95	167	1
PFA24	107	50	217	6
$\sum$	1250	497	2247	50

100 Orders (Sorted Input)

PFA10 TRUCK/MOD-ELC3/4617 0					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	10	0.0000000000
DEPOT	C_38	0	21	31	160.00000000
C_38		21	10	31	152.00000000
C_38	C_15	31	11	42	152.00000000
C_15		42	10	42	132.00000000
C_15	C_45	52	5	79	132.00000000
C_45		69	10	79	114.00000000
C_45	C_17	79	6	85	114.00000000
C_17		85	10	85	95.000000000
C_17	C_87	95	6	104	95.000000000
C_87		101	10	104	60.000000000
C_87	C_44	111	24	142	60.000000000
C_44		135	10	142	53.000000000
C_44	C_73	145	27	194	53.000000000
C_73		172	10	194	28.000000000
C_73	C_22	182	4	198	28.000000000
C_22		186	10	198	17.000000000
C_22	C_27	196	10	208	17.000000000
C_27		206	10	208	0.0000000000
C_27	DEPOT	216	11	230	0.0000000000
order DEPOT C_73 25.000000000 0 0 197 0 10 197					
order DEPOT C_38 8.0000000000 0 0 198 0 10 198					
order DEPOT C_27 17.000000000 0 0 208 0 10 208					
order DEPOT C_22 11.000000000 0 0 201 0 10 201					
order DEPOT C_44 7.0000000000 0 0 142 132 10 142					
order DEPOT C_87 35.000000000 0 0 104 94 10 104					
order DEPOT C_17 19.000000000 0 0 85 75 10 85					
order DEPOT C_45 18.000000000 0 0 79 69 10 79					
order DEPOT C_15 20.000000000 0 0 42 32 10 42					
Total time needed:			227		
Waiting time:			12		
Maximal gap:			12		
Tatal distance to go:			125		

PFA12 TRUCK/MOD-SERV/4326 0					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	12	0.0000000000
DEPOT	C_51	0	16	28	138.00000000
C_51		16	10	28	125.00000000
C_51	C_66	26	33	61	125.00000000
C_66		59	10	61	105.00000000
C_66	C_72	69	10	87	105.00000000
C_72		79	10	87	90.000000000
C_72	C_82	89	14	104	90.000000000
C_82		103	10	104	64.000000000
C_82	C_21	113	14	136	64.000000000
C_21		127	10	136	55.000000000
C_21	C_33	137	10	151	55.000000000
C_33		147	10	151	32.000000000
C_33	C_71	157	13	192	32.000000000
C_71		182	10	192	27.000000000
C_71	C_32	192	7	202	27.000000000
C_32		199	10	202	0.0000000000
C_32	DEPOT	209	17	230	0.0000000000
order DEPOT C_51 13.000000000 0 0 203 0 10 203					
order DEPOT C_32 27.000000000 0 0 202 0 10 202					
order DEPOT C_71 5.0000000000 0 0 192 182 10 192					
order DEPOT C_33 23.000000000 0 0 151 141 10 151					
order DEPOT C_21 9.0000000000 0 0 136 126 10 136					
order DEPOT C_82 26.000000000 0 0 104 94 10 104					
order DEPOT C_72 15.000000000 0 0 87 77 10 87					
order DEPOT C_66 20.000000000 0 0 61 51 10 61					
Total time needed:			226		
Waiting time:			12		
Maximal gap:			12		
Tatal distance to go:			134		

PFA15 TRUCK/MOD-SERV/4316 1					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	65	0.0000000000
DEPOT	C_89	0	19	84	84.000000000
C_89		74	10	84	75.000000000
C_89	C_19	84	13	97	75.000000000
C_19		97	10	97	63.000000000
C_19	C_84	107	6	177	63.000000000
C_84		113	10	177	52.000000000
C_84	C_83	123	10	187	52.000000000
C_83		133	10	187	36.000000000
C_83	C_49	143	5	192	36.000000000
C_49		148	10	192	0.0000000000
C_49	DEPOT	158	27	230	0.0000000000
order DEPOT C_84 11.000000000 0 0 198 0 10 198					
order DEPOT C_83 16.000000000 0 0 196 0 10 196					
order DEPOT C_49 36.000000000 0 0 192 0 10 192					
order DEPOT C_19 12.000000000 0 0 97 87 10 97					
order DEPOT C_89 9.0000000000 0 0 84 74 10 84					
Total time needed:			185		
Waiting time:			55		
Maximal gap:			55		
Tatal distance to go:			80		

PFA16 TRUCK/MOD-ELC3/4626 0					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	14	0.0000000000
DEPOT	C_13	0	15	29	136.00000000
C_13		15	10	29	117.00000000
C_13	C_40	25	25	54	117.00000000
C_40		50	10	54	86.000000000
C_40	C_24	60	8	78	86.000000000
C_24		68	10	78	57.000000000
C_24	C_68	78	12	93	57.000000000
C_68		90	10	93	32.000000000
C_68	C_56	100	18	146	32.000000000
C_56		136	10	146	30.000000000
C_56	C_25	146	12	163	30.000000000
C_25		158	10	163	27.000000000
C_25	C_78	168	14	189	27.000000000
C_78		182	10	189	13.000000000
C_78	C_4	192	2	197	13.000000000
C_4		194	10	197	0.0000000000
C_4	DEPOT	204	22	230	0.0000000000
order DEPOT C_13 19.000000000 0 0 205 0 10 205					
order DEPOT C_4 13.000000000 0 0 197 0 10 197					
order DEPOT C_78 14.000000000 0 0 189 179 10 189					
order DEPOT C_25 3.0000000000 0 0 163 153 10 163					
order DEPOT C_56 2.0000000000 0 0 146 136 10 146					
order DEPOT C_68 25.000000000 0 0 93 83 10 93					
order DEPOT C_24 29.000000000 0 0 78 68 10 78					
order DEPOT C_40 31.000000000 0 0 54 44 10 54					
Total time needed:			226		
Waiting time:			18		
Maximal gap:			18		
Tatal distance to go:			128		

PFA17 TRUCK/MOD-ELC3/4624 1					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	75	0.0000000000
DEPOT	C_79	0	31	106	3.0000000000
C_79		96	10	106	0.0000000000
C_79	DEPOT	106	31	230	0.0000000000
order DEPOT C_79 3.0000000000 0 0 106 96 10 106					
Total time needed:			137		
Waiting time:			65		
Maximal gap:			65		
Tatal distance to go:			62		

PFA18 TRUCK/MOD-ELC3/4622 0					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	10	0.0000000000
DEPOT	C_64	0	34	44	100.00000000
C_64		34	10	44	90.000000000
C_64	C_63	44	11	68	90.000000000
C_63		58	10	68	71.000000000
C_63	C_12	68	8	77	71.000000000
C_12		76	10	77	59.000000000
C_12	C_91	86	11	105	59.000000000
C_91		97	10	105	56.000000000
C_91	C_11	107	7	134	56.000000000
C_11		124	10	134	40.000000000
C_11	C_14	134	35	169	40.000000000
C_14		169	10	169	17.000000000
C_14	C_101	179	13	195	17.000000000
C_101		192	10	195	0.0000000000
C_101	DEPOT	202	24	230	0.0000000000
order DEPOT C_101 17.000000000 0 0 195 185 10 195					
order DEPOT C_14 23.000000000 0 0 169 159 10 169					
order DEPOT C_11 16.000000000 0 0 134 124 10 134					
order DEPOT C_91 3.0000000000 0 0 105 95 10 105					
order DEPOT C_12 12.000000000 0 0 77 67 10 77					
order DEPOT C_63 19.000000000 0 0 68 58 10 68					
order DEPOT C_64 10.000000000 0 0 44 34 10 44					
Total time needed:			226		
Waiting time:			13		
Maximal gap:			10		
Tatal distance to go:			143		

PFA19 TRUCK/MOD-ELC3/4620 0					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	28	0.0000000000
DEPOT	C_53	0	11	39	141.00000000
C_53		11	10	39	132.00000000
C_53	C_28	21	8	47	132.00000000
C_28		37	10	47	116.00000000
C_28	C_70	47	7	60	116.00000000
C_70		54	10	60	110.00000000
C_70	C_31	64	13	81	110.00000000
C_31		77	10	81	89.000000000
C_31	C_10	87	15	107	89.000000000
C_10		102	10	107	73.000000000
C_10	C_55	112	31	150	73.000000000
C_55		143	10	150	55.000000000
C_55	C_95	153	31	197	55.000000000
C_95		184	10	197	28.000000000
C_95	C_60	194	5	202	28.000000000
C_60		199	10	202	0.0000000000
C_60	DEPOT	209	17	230	0.0000000000
order DEPOT C_95 27.000000000 0 0 207 0 10 207					
order DEPOT C_60 28.000000000 0 0 202 0 10 202					
order DEPOT C_53 9.0000000000 0 0 208 0 10 208					
order DEPOT C_55 18.000000000 0 0 150 140 10 150					
order DEPOT C_10 16.000000000 0 0 107 97 10 107					
order DEPOT C_31 21.000000000 0 0 81 71 10 81					
order DEPOT C_70 6.0000000000 0 0 60 50 10 60					
order DEPOT C_28 16.000000000 0 0 47 37 10 47					
Total time needed:			226		
Waiting time:			8		
Maximal gap:			8		
Tatal distance to go:			138		

PFA21 TRUCK/MOD-ELC1/2644 1					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	87	0.0000000000
DEPOT	C_85	0	24	111	7.0000000000
C_85		101	10	111	0.0000000000
C_85	DEPOT	111	24	230	0.0000000000
order DEPOT C_85 7.0000000000 0 0 111 101 10 111					
Total time needed:			135		
Waiting time:			77		
Maximal gap:			77		
Tatal distance to go:			48		

PFA22 TRUCK/MOD-ELC1/2642 1					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	88	0.0000000000
DEPOT	C_58	0	23	111	7.0000000000
C_58		101	10	111	0.0000000000
C_58	DEPOT	111	23	230	0.0000000000
order DEPOT C_58 7.0000000000 0 0 111 101 10 111					
Total time needed:			134		
Waiting time:			78		
Maximal gap:			78		
Tatal distance to go:			46		

PFA24 TRUCK/MOD-ELC1/2638 1					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	81	0.0000000000
DEPOT	C_23	0	26	107	18.000000000
C_23		97	10	107	0.0000000000
C_23	DEPOT	107	26	230	0.0000000000
order DEPOT C_23 18.000000000 0 0 107 97 10 107					
Total time needed:			133		
Waiting time:			71		
Maximal gap:			71		
Tatal distance to go:			52		

PFA26 TRUCK/MOD-ELC2/2475 0					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	10	0.0000000000
DEPOT	C_37	0	41	51	109.00000000
C_37		41	10	51	104.00000000
C_37	C_48	51	7	61	104.00000000
C_48		58	10	61	77.000000000
C_48	C_20	68	8	86	77.000000000
C_20		76	10	86	60.000000000
C_20	C_9	86	17	105	60.000000000
C_9		103	10	105	51.000000000
C_9	C_47	113	9	127	51.000000000
C_47		122	10	127	50.000000000
C_47	C_18	132	18	167	50.000000000
C_18		157	10	167	48.000000000
C_18	C_6	167	10	192	48.000000000
C_6		177	10	192	22.000000000
C_6	C_94	187	6	198	22.000000000
C_94		193	10	198	0.0000000000
C_94	DEPOT	203	20	230	0.0000000000
order DEPOT C_6 26.000000000 0 0 199 0 10 199					
order DEPOT C_94 22.000000000 0 0 198 188 10 198					
order DEPOT C_18 2.0000000000 0 0 167 157 10 167					
order DEPOT C_47 1.0000000000 0 0 127 117 10 127					
order DEPOT C_9 9.0000000000 0 0 105 95 10 105					
order DEPOT C_20 17.000000000 0 0 86 76 10 86					
order DEPOT C_48 27.000000000 0 0 61 51 10 61					
order DEPOT C_37 5.0000000000 0 0 51 41 10 51					
Total time needed:			223		
Waiting time:			7		
Maximal gap:			7		
Tatal distance to go:			136		

PFA27 TRUCK/MOD-ELC2/2473 1					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	68	0.0000000000
DEPOT	C_74	0	20	88	49.000000000
C_74		78	10	88	40.000000000
C_74	C_54	88	15	105	40.000000000
C_54		103	10	105	26.000000000
C_54	C_75	113	20	159	26.000000000
C_75		149	10	159	18.000000000
C_75	C_76	159	3	192	18.000000000
C_76		162	10	192	0.0000000000
C_76	DEPOT	172	27	230	0.0000000000
order DEPOT C_76 18.000000000 0 0 192 0 10 192					
order DEPOT C_75 8.0000000000 0 0 159 149 10 159					
order DEPOT C_54 14.000000000 0 0 105 95 10 105					
order DEPOT C_74 9.0000000000 0 0 88 78 10 88					
Total time needed:			199		
Waiting time:			74		
Maximal gap:			58		
Tatal distance to go:			85		

PFA28 TRUCK/MOD-ELC2/2471 1					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	49	0.0000000000
DEPOT	C_97	0	15	64	79.000000000
C_97		15	10	64	68.000000000
C_97	C_99	25	6	70	68.000000000
C_99		31	10	70	58.000000000
C_99	C_86	41	3	73	58.000000000
C_86		44	10	73	17.000000000
C_86	C_92	54	3	76	17.000000000
C_92		57	10	76	16.000000000
C_92	C_39	67	17	93	16.000000000
C_39		84	10	93	0.0000000000
C_39	DEPOT	94	42	230	0.0000000000
order DEPOT C_99 10.000000000 0 0 198 0 10 198					
order DEPOT C_97 11.000000000 0 0 204 0 10 204					
order DEPOT C_92 1.0000000000 0 0 194 0 10 194					
order DEPOT C_86 41.000000000 0 0 196 0 10 196					
order DEPOT C_39 16.000000000 0 0 93 83 10 93					
Total time needed:			136		
Waiting time:			0		
Maximal gap:			0		
Tatal distance to go:			86		

PFA29 TRUCK/MOD-ELC2/2469 0					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	20	0.0000000000
DEPOT	C_2	0	15	35	79.000000000
C_2		15	10	35	69.000000000
C_2	C_34	25	12	47	69.000000000
C_34		37	10	47	58.000000000
C_34	C_30	47	14	73	58.000000000
C_30		63	10	73	49.000000000
C_30	C_52	73	21	98	49.000000000
C_52		94	10	98	39.000000000
C_52	C_67	104	15	137	39.000000000
C_67		127	10	137	14.000000000
C_67	C_36	137	16	153	14.000000000
C_36		153	10	153	6.0000000000
C_36	C_81	163	28	192	6.0000000000
C_81		191	10	192	0.0000000000
C_81	DEPOT	201	21	230	0.0000000000
order DEPOT C_2 10.000000000 0 0 204 0 10 204					
order DEPOT C_81 6.0000000000 0 0 192 182 10 192					
order DEPOT C_36 8.0000000000 0 0 153 143 10 153					
order DEPOT C_67 25.000000000 0 0 137 127 10 137					
order DEPOT C_52 10.000000000 0 0 98 88 10 98					
order DEPOT C_30 9.0000000000 0 0 73 63 10 73					
order DEPOT C_34 11.000000000 0 0 47 37 10 47					
Total time needed:			222		
Waiting time:			10		
Maximal gap:			8		
Tatal distance to go:			142		

PFA30 TRUCK/MOD-ELC6/2090 0					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	10	0.0000000000
DEPOT	C_93	0	18	28	62.000000000
C_93		18	10	28	60.000000000
C_93	C_43	28	10	41	60.000000000
C_43		38	10	41	55.000000000
C_43	C_16	48	9	71	55.000000000
C_16		61	10	71	47.000000000
C_16	C_41	71	22	95	47.000000000
C_41		93	10	95	38.000000000
C_41	C_3	103	9	122	38.000000000
C_3		112	10	122	31.000000000
C_3	C_57	122	18	140	31.000000000
C_57		140	10	140	25.000000000
C_57	C_5	150	8	159	25.000000000
C_5		158	10	159	6.0000000000
C_5	C_26	168	10	182	6.0000000000
C_26		178	10	182	0.0000000000
C_26	DEPOT	188	33	230	0.0000000000
order DEPOT C_3 7.0000000000 0 0 202 0 10 202					
order DEPOT C_26 6.0000000000 0 0 182 172 10 182					
order DEPOT C_5 19.000000000 0 0 159 149 10 159					
order DEPOT C_57 6.0000000000 0 0 140 130 10 140					
order DEPOT C_41 9.0000000000 0 0 95 85 10 95					
order DEPOT C_16 8.0000000000 0 0 71 61 10 71					
order DEPOT C_43 5.0000000000 0 0 41 31 10 41					
order DEPOT C_93 2.0000000000 0 0 28 18 10 28					
Total time needed:			221		
Waiting time:			4		
Maximal gap:			4		
Tatal distance to go:			137		

PFA31 TRUCK/MOD-ELC5/1525 1					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	79	0.0000000000
DEPOT	C_42	0	28	107	5.0000000000
C_42		97	10	107	0.0000000000
C_42	DEPOT	107	28	230	0.0000000000
order DEPOT C_42 5.0000000000 0 0 107 97 10 107					
Total time needed:			135		
Waiting time:			69		
Maximal gap:			69		
Tatal distance to go:			56		

PFA32 TRUCK/MOD-ELC5/1523 1					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	61	0.0000000000
DEPOT	C_62	0	25	86	51.000000000
C_62		76	10	86	38.000000000
C_62	C_88	86	17	103	38.000000000
C_88		103	10	103	12.000000000
C_88	C_98	113	4	143	12.000000000
C_98		133	10	143	0.0000000000
C_98	DEPOT	143	17	230	0.0000000000
order DEPOT C_98 12.000000000 0 0 143 133 10 143					
order DEPOT C_88 26.000000000 0 0 103 93 10 103					
order DEPOT C_62 13.000000000 0 0 86 76 10 86					
Total time needed:			160		
Waiting time:			67		
Maximal gap:			51		
Tatal distance to go:			63		

PFA34 TRUCK/MOD-ELC5/1519 1					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	76	0.0000000000
DEPOT	C_100	0	17	93	12.000000000
C_100		83	10	93	3.0000000000
C_100	C_7	93	6	109	3.0000000000
C_7		99	10	109	0.0000000000
C_7	DEPOT	109	11	230	0.0000000000
order DEPOT C_7 3.0000000000 0 0 109 99 10 109					
order DEPOT C_100 9.0000000000 0 0 93 83 10 93					
Total time needed:			120		
Waiting time:			66		
Maximal gap:			66		
Tatal distance to go:			34		

PFA36 TRUCK/MOD-ELC6/2099 0					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	15	0.0000000000
DEPOT	C_96	0	14	29	122.00000000
C_96		14	10	29	102.00000000
C_96	C_29	24	20	49	102.00000000
C_29		44	10	49	86.000000000
C_29	C_77	54	9	83	86.000000000
C_77		73	10	83	73.000000000
C_77	C_80	83	10	102	73.000000000
C_80		93	10	102	50.000000000
C_80	C_69	103	9	152	50.000000000
C_69		142	10	152	14.000000000
C_69	C_35	152	18	183	14.000000000
C_35		170	10	183	0.0000000000
C_35	DEPOT	180	36	230	0.0000000000
order DEPOT C_96 20.000000000 0 0 205 0 10 205					
order DEPOT C_35 14.000000000 0 0 183 0 10 183					
order DEPOT C_69 36.000000000 0 0 152 142 10 152					
order DEPOT C_80 23.000000000 0 0 102 92 10 102					
order DEPOT C_77 13.000000000 0 0 83 73 10 83					
order DEPOT C_29 16.000000000 0 0 49 39 10 49					
Total time needed:			216		
Waiting time:			40		
Maximal gap:			30		
Tatal distance to go:			116		

PFA37 TRUCK/MOD-ELC6/2097 0					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	13	0.0000000000
DEPOT	C_46	0	29	42	96.000000000
C_46		32	10	42	80.000000000
C_46	C_65	42	40	83	80.000000000
C_65		82	10	83	71.000000000
C_65	C_50	92	12	118	71.000000000
C_50		108	10	118	41.000000000
C_50	C_8	118	22	156	41.000000000
C_8		140	10	156	36.000000000
C_8	C_61	150	16	172	36.000000000
C_61		166	10	172	33.000000000
C_61	C_90	176	9	186	33.000000000
C_90		185	10	186	18.000000000
C_90	C_59	195	13	210	18.000000000
C_59		208	10	210	0.0000000000
C_59	DEPOT	218	9	230	0.0000000000
order DEPOT C_8 5.0000000000 0 0 198 0 10 198					
order DEPOT C_59 18.000000000 0 0 210 200 10 210					
order DEPOT C_90 15.000000000 0 0 186 176 10 186					
order DEPOT C_61 3.0000000000 0 0 172 162 10 172					
order DEPOT C_50 30.000000000 0 0 118 108 10 118					
order DEPOT C_65 9.0000000000 0 0 83 73 10 83					
order DEPOT C_46 16.000000000 0 0 42 32 10 42					
Total time needed:			227		
Waiting time:			7		
Maximal gap:			4		
Tatal distance to go:			150		

Name:	Dist.:	Wait:	Time:	Stops:
PFA10	125	12	227	9
PFA30	137	4	221	8
PFA15	80	55	185	5
PFA19	138	8	226	8
PFA32	63	67	160	3
PFA17	62	65	137	1
PFA37	150	7	227	7
PFA27	85	74	199	4
PFA18	143	13	226	7
PFA28	86	0	136	5
PFA34	34	66	120	2
PFA29	142	10	222	7
PFA31	56	69	135	1
PFA26	136	7	223	8
PFA36	116	40	216	6
PFA22	46	78	134	1
PFA21	48	77	135	1
PFA12	134	12	226	8
PFA16	128	18	226	8
PFA24	52	71	133	1
$\sum$	1961	753	3714	100

100 Orders (Original Input)

PFA10 TRUCK/MOD-ELC3/4656 2					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	77	0.0000000000
DEPOT	C_82	0	27	104	26.000000000
C_82		94	10	104	0.0000000000
C_82	DEPOT	104	27	230	0.0000000000
order DEPOT C_82 26.000000000 0 0 104 94 10 104					
Total time needed:			131		
Waiting time:			67		
Maximal gap:			67		
Tatal distance to go:			54		

PFA11 TRUCK/MOD-SERV/4449 1					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	68	0.0000000000
DEPOT	C_74	0	20	88	9.0000000000
C_74		78	10	88	0.0000000000
C_74	DEPOT	88	20	230	0.0000000000
order DEPOT C_74 9.0000000000 0 0 88 78 10 88					
Total time needed:			108		
Waiting time:			58		
Maximal gap:			58		
Tatal distance to go:			40		

PFA13 TRUCK/MOD-SERV/4440 0					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	21	0.0000000000
DEPOT	C_40	0	33	54	118.00000000
C_40		44	10	54	87.000000000
C_40	C_24	54	8	78	87.000000000
C_24		68	10	78	58.000000000
C_24	C_76	78	8	99	58.000000000
C_76		86	10	99	40.000000000
C_76	C_42	96	8	107	40.000000000
C_42		104	10	107	35.000000000
C_42	C_55	114	26	150	35.000000000
C_55		140	10	150	17.000000000
C_55	C_25	150	10	163	17.000000000
C_25		160	10	163	14.000000000
C_25	C_78	170	14	189	14.000000000
C_78		184	10	189	0.0000000000
C_78	DEPOT	194	19	230	0.0000000000
order DEPOT C_78 14.000000000 0 0 189 179 10 189					
order DEPOT C_76 18.000000000 0 0 192 0 10 192					
order DEPOT C_55 18.000000000 0 0 150 140 10 150					
order DEPOT C_42 5.0000000000 0 0 107 97 10 107					
order DEPOT C_40 31.000000000 0 0 54 44 10 54					
order DEPOT C_25 3.0000000000 0 0 163 153 10 163					
order DEPOT C_24 29.000000000 0 0 78 68 10 78					
Total time needed:			213		
Waiting time:			17		
Maximal gap:			11		
Tatal distance to go:			126		

PFA14 TRUCK/MOD-SERV/4436 0					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	10	0.0000000000
DEPOT	C_37	0	41	51	89.000000000
C_37		41	10	51	84.000000000
C_37	C_48	51	7	61	84.000000000
C_48		58	10	61	57.000000000
C_48	C_20	68	8	86	57.000000000
C_20		76	10	86	40.000000000
C_20	C_91	86	17	105	40.000000000
C_91		103	10	105	37.000000000
C_91	C_21	113	14	136	37.000000000
C_21		127	10	136	28.000000000
C_21	C_33	137	10	151	28.000000000
C_33		147	10	151	5.0000000000
C_33	C_71	157	13	192	5.0000000000
C_71		182	10	192	0.0000000000
C_71	DEPOT	192	21	230	0.0000000000
order DEPOT C_91 3.0000000000 0 0 105 95 10 105					
order DEPOT C_71 5.0000000000 0 0 192 182 10 192					
order DEPOT C_48 27.000000000 0 0 61 51 10 61					
order DEPOT C_37 5.0000000000 0 0 51 41 10 51					
order DEPOT C_33 23.000000000 0 0 151 141 10 151					
order DEPOT C_21 9.0000000000 0 0 136 126 10 136					
order DEPOT C_20 17.000000000 0 0 86 76 10 86					
Total time needed:			213		
Waiting time:			12		
Maximal gap:			12		
Tatal distance to go:			131		

PFA15 TRUCK/MOD-SERV/4433 0					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	42	0.0000000000
DEPOT	C_2	0	15	57	98.000000000
C_2		15	10	57	88.000000000
C_2	C_4	25	14	71	88.000000000
C_4		39	10	71	75.000000000
C_4	C_13	49	11	82	75.000000000
C_13		60	10	82	56.000000000
C_13	C_27	70	7	89	56.000000000
C_27		77	10	89	39.000000000
C_27	C_7	87	20	109	39.000000000
C_7		107	10	109	36.000000000
C_7	C_22	117	22	149	36.000000000
C_22		139	10	149	25.000000000
C_22	C_5	149	10	159	25.000000000
C_5		159	10	159	6.0000000000
C_5	C_26	169	10	182	6.0000000000
C_26		179	10	182	0.0000000000
C_26	DEPOT	189	33	230	0.0000000000
order DEPOT C_27 17.000000000 0 0 208 0 10 208					
order DEPOT C_26 6.0000000000 0 0 182 172 10 182					
order DEPOT C_22 11.000000000 0 0 201 0 10 201					
order DEPOT C_13 19.000000000 0 0 205 0 10 205					
order DEPOT C_7 3.0000000000 0 0 109 99 10 109					
order DEPOT C_5 19.000000000 0 0 159 149 10 159					
order DEPOT C_4 13.000000000 0 0 197 0 10 197					
order DEPOT C_2 10.000000000 0 0 204 0 10 204					
Total time needed:			222		
Waiting time:			0		
Maximal gap:			0		
Tatal distance to go:			142		

PFA16 TRUCK/MOD-ELC3/4665 0					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	23	0.0000000000
DEPOT	C_34	0	24	47	58.000000000
C_34		37	10	47	47.000000000
C_34	C_30	47	14	73	47.000000000
C_30		63	10	73	38.000000000
C_30	C_52	73	21	98	38.000000000
C_52		94	10	98	28.000000000
C_52	C_35	104	16	143	28.000000000
C_35		120	10	143	14.000000000
C_35	C_36	130	10	153	14.000000000
C_36		143	10	153	6.0000000000
C_36	C_81	153	28	192	6.0000000000
C_81		182	10	192	0.0000000000
C_81	DEPOT	192	21	230	0.0000000000
order DEPOT C_81 6.0000000000 0 0 192 182 10 192					
order DEPOT C_52 10.000000000 0 0 98 88 10 98					
order DEPOT C_36 8.0000000000 0 0 153 143 10 153					
order DEPOT C_35 14.000000000 0 0 183 0 10 183					
order DEPOT C_34 11.000000000 0 0 47 37 10 47					
order DEPOT C_30 9.0000000000 0 0 73 63 10 73					
Total time needed:			213		
Waiting time:			19		
Maximal gap:			13		
Tatal distance to go:			134		

PFA17 TRUCK/MOD-ELC3/4663 1					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	50	0.0000000000
DEPOT	C_68	0	43	93	33.000000000
C_68		83	10	93	8.0000000000
C_68	C_75	93	22	159	8.0000000000
C_75		149	10	159	0.0000000000
C_75	DEPOT	159	24	230	0.0000000000
order DEPOT C_75 8.0000000000 0 0 159 149 10 159					
order DEPOT C_68 25.000000000 0 0 93 83 10 93					
Total time needed:			183		
Waiting time:			74		
Maximal gap:			40		
Tatal distance to go:			89		

PFA18 TRUCK/MOD-ELC3/4661 1					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	61	0.0000000000
DEPOT	C_62	0	25	86	32.000000000
C_62		76	10	86	19.000000000
C_62	C_85	86	7	111	19.000000000
C_85		101	10	111	12.000000000
C_85	C_98	111	17	143	12.000000000
C_98		133	10	143	0.0000000000
C_98	DEPOT	143	17	230	0.0000000000
order DEPOT C_98 12.000000000 0 0 143 133 10 143					
order DEPOT C_85 7.0000000000 0 0 111 101 10 111					
order DEPOT C_62 13.000000000 0 0 86 76 10 86					
Total time needed:			160		
Waiting time:			64		
Maximal gap:			51		
Tatal distance to go:			66		

PFA19 TRUCK/MOD-ELC3/4659 0					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	10	0.0000000000
DEPOT	C_60	0	17	27	123.00000000
C_60		17	10	27	95.000000000
C_60	C_15	27	15	42	95.000000000
C_15		42	10	42	75.000000000
C_15	C_16	52	15	71	75.000000000
C_16		67	10	71	67.000000000
C_16	C_23	77	15	107	67.000000000
C_23		97	10	107	49.000000000
C_23	C_73	107	6	123	49.000000000
C_73		113	10	123	24.000000000
C_73	C_18	123	44	167	24.000000000
C_18		167	10	167	22.000000000
C_18	C_94	177	14	198	22.000000000
C_94		191	10	198	0.0000000000
C_94	DEPOT	201	20	230	0.0000000000
order DEPOT C_94 22.000000000 0 0 198 188 10 198					
order DEPOT C_73 25.000000000 0 0 197 0 10 197					
order DEPOT C_60 28.000000000 0 0 202 0 10 202					
order DEPOT C_23 18.000000000 0 0 107 97 10 107					
order DEPOT C_18 2.0000000000 0 0 167 157 10 167					
order DEPOT C_16 8.0000000000 0 0 71 61 10 71					
order DEPOT C_15 20.000000000 0 0 42 32 10 42					
Total time needed:			221		
Waiting time:			5		
Maximal gap:			5		
Tatal distance to go:			146		

PFA20 TRUCK/MOD-ELC2/2503 2					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	69	0.0000000000
DEPOT	C_87	0	35	104	35.000000000
C_87		94	10	104	0.0000000000
C_87	DEPOT	104	35	230	0.0000000000
order DEPOT C_87 35.000000000 0 0 104 94 10 104					
Total time needed:		139			
Waiting time:		59			
Maximal gap:		59			
Tatal distance to go:		70			

PFA21 TRUCK/MOD-ELC1/2681 1					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	48	0.0000000000
DEPOT	C_70	0	12	60	42.000000000
C_70		50	10	60	36.000000000
C_70	C_77	60	13	83	36.000000000
C_77		73	10	83	23.000000000
C_77	C_80	83	10	102	23.000000000
C_80		93	10	102	0.0000000000
C_80	DEPOT	103	25	230	0.0000000000
order DEPOT C_80 23.000000000 0 0 102 92 10 102					
order DEPOT C_77 13.000000000 0 0 83 73 10 83					
order DEPOT C_70 6.0000000000 0 0 60 50 10 60					
Total time needed:		128			
Waiting time:		38			
Maximal gap:		38			
Tatal distance to go:		60			

PFA23 TRUCK/MOD-ELC1/2677 0					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	13	0.0000000000
DEPOT	C_46	0	29	42	78.000000000
C_46		32	10	42	62.000000000
C_46	C_63	42	26	68	62.000000000
C_63		68	10	68	43.000000000
C_63	C_19	78	18	97	43.000000000
C_19		96	10	97	31.000000000
C_19	C_47	106	19	127	31.000000000
C_47		125	10	127	30.000000000
C_47	C_83	135	13	159	30.000000000
C_83		148	10	159	14.000000000
C_83	C_61	158	13	172	14.000000000
C_61		171	10	172	11.000000000
C_61	C_84	181	4	198	11.000000000
C_84		185	10	198	0.0000000000
C_84	DEPOT	195	21	230	0.0000000000
order DEPOT C_84 11.000000000 0 0 198 0 10 198					
order DEPOT C_83 16.000000000 0 0 196 0 10 196					
order DEPOT C_63 19.000000000 0 0 68 58 10 68					
order DEPOT C_61 3.0000000000 0 0 172 162 10 172					
order DEPOT C_47 1.0000000000 0 0 127 117 10 127					
order DEPOT C_46 16.000000000 0 0 42 32 10 42					
order DEPOT C_19 12.000000000 0 0 97 87 10 97					
Total time needed:			216		
Waiting time:			3		
Maximal gap:			3		
Tatal distance to go:			143		

PFA25 TRUCK/MOD-ELC1/2672 1					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	51	0.0000000000
DEPOT	C_39	0	42	93	16.000000000
C_39		83	10	93	0.0000000000
C_39	DEPOT	93	42	230	0.0000000000
order DEPOT C_39 16.000000000 0 0 93 83 10 93					
Total time needed:			135		
Waiting time:			41		
Maximal gap:			41		
Tatal distance to go:			84		

PFA28 TRUCK/MOD-ELC2/2508 1					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	85	0.0000000000
DEPOT	C_88	0	18	103	33.000000000
C_88		93	10	103	7.0000000000
C_88	C_58	103	7	111	7.0000000000
C_58		110	10	111	0.0000000000
C_58	DEPOT	120	23	230	0.0000000000
order DEPOT C_88 26.000000000 0 0 103 93 10 103					
order DEPOT C_58 7.0000000000 0 0 111 101 10 111					
Total time needed:			143		
Waiting time:			75		
Maximal gap:			75		
Tatal distance to go:			48		

PFA29 TRUCK/MOD-ELC2/2506 2					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	76	0.0000000000
DEPOT	C_100	0	17	93	9.0000000000
C_100		83	10	93	0.0000000000
C_100	DEPOT	93	17	230	0.0000000000
order DEPOT C_100 9.0000000000 0 0 93 83 10 93					
Total time needed:			110		
Waiting time:			66		
Maximal gap:			66		
Tatal distance to go:			34		

PFA30 TRUCK/MOD-ELC6/2127 0					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	11	0.0000000000
DEPOT	C_3	0	18	29	140.00000000
C_3		18	10	29	133.00000000
C_3	C_29	28	20	49	133.00000000
C_29		48	10	49	117.00000000
C_29	C_51	58	11	92	117.00000000
C_51		69	10	92	104.00000000
C_51	C_10	79	15	107	104.00000000
C_10		97	10	107	88.000000000
C_10	C_11	107	25	134	88.000000000
C_11		132	10	134	72.000000000
C_11	C_32	142	8	174	72.000000000
C_32		150	10	174	45.000000000
C_32	C_49	160	18	192	45.000000000
C_49		178	10	192	9.0000000000
C_49	C_53	188	16	208	9.0000000000
C_53		204	10	208	0.0000000000
C_53	DEPOT	214	11	230	0.0000000000
order DEPOT C_53 9.0000000000 0 0 208 0 10 208					
order DEPOT C_51 13.000000000 0 0 203 0 10 203					
order DEPOT C_49 36.000000000 0 0 192 0 10 192					
order DEPOT C_32 27.000000000 0 0 202 0 10 202					
order DEPOT C_29 16.000000000 0 0 49 39 10 49					
order DEPOT C_11 16.000000000 0 0 134 124 10 134					
order DEPOT C_10 16.000000000 0 0 107 97 10 107					
order DEPOT C_3 7.0000000000 0 0 202 0 10 202					
Total time needed:	225				
Waiting time:	3				
Maximal gap:	3				
Tatal distance to go:	142				

PFA31 TRUCK/MOD-ELC5/1562 1					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	12	0.0000000000
DEPOT	C_66	0	49	61	91.000000000
C_66		51	10	61	71.000000000
C_66	C_72	61	10	87	71.000000000
C_72		77	10	87	56.000000000
C_72	C_79	87	16	106	56.000000000
C_79		103	10	106	53.000000000
C_79	C_69	113	13	152	53.000000000
C_69		142	10	152	17.000000000
C_69	C_101	152	43	195	17.000000000
C_101		195	10	195	0.0000000000
C_101	DEPOT	205	24	230	0.0000000000
order DEPOT C_101 17.000000000 0 0 195 185 10 195					
order DEPOT C_79 3.0000000000 0 0 106 96 10 106					
order DEPOT C_72 15.000000000 0 0 87 77 10 87					
order DEPOT C_69 36.000000000 0 0 152 142 10 152					
order DEPOT C_66 20.000000000 0 0 61 51 10 61					
Total time needed:			229		
Waiting time:			24		
Maximal gap:			16		
Tatal distance to go:			155		

PFA32 TRUCK/MOD-ELC5/1560 1					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	10	0.0000000000
DEPOT	C_64	0	34	44	44.000000000
C_64		34	10	44	34.000000000
C_64	C_65	44	14	83	34.000000000
C_65		73	10	83	25.000000000
C_65	C_67	83	34	137	25.000000000
C_67		127	10	137	0.0000000000
C_67	DEPOT	137	40	230	0.0000000000
order DEPOT C_67 25.000000000 0 0 137 127 10 137					
order DEPOT C_65 9.0000000000 0 0 83 73 10 83					
order DEPOT C_64 10.000000000 0 0 44 34 10 44					
Total time needed:			177		
Waiting time:			25		
Maximal gap:			15		
Tatal distance to go:			122		

PFA33 TRUCK/MOD-ELC5/1558 0					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	10	0.0000000000
DEPOT	C_93	0	18	28	135.00000000
C_93		18	10	28	133.00000000
C_93	C_43	28	10	41	133.00000000
C_43		38	10	41	128.00000000
C_43	C_45	48	13	79	128.00000000
C_45		69	10	79	110.00000000
C_45	C_17	79	6	85	110.00000000
C_17		85	10	85	91.000000000
C_17	C_86	95	6	122	91.000000000
C_86		101	10	122	50.000000000
C_86	C_92	111	3	125	50.000000000
C_92		114	10	125	49.000000000
C_92	C_44	124	17	142	49.000000000
C_44		141	10	142	42.000000000
C_44	C_90	151	32	186	42.000000000
C_90		183	10	186	27.000000000
C_90	C_95	193	8	207	27.000000000
C_95		201	10	207	0.0000000000
C_95	DEPOT	211	12	230	0.0000000000
order DEPOT C_95 27.000000000 0 0 207 0 10 207					
order DEPOT C_93 2.0000000000 0 0 28 18 10 28					
order DEPOT C_92 1.0000000000 0 0 194 0 10 194					
order DEPOT C_90 15.000000000 0 0 186 176 10 186					
order DEPOT C_86 41.000000000 0 0 196 0 10 196					
order DEPOT C_45 18.000000000 0 0 79 69 10 79					
order DEPOT C_44 7.0000000000 0 0 142 132 10 142					
order DEPOT C_43 5.0000000000 0 0 41 31 10 41					
order DEPOT C_17 19.000000000 0 0 85 75 10 85					
Total time needed:	223				
Waiting time:	8				
Maximal gap:	8				
Tatal distance to go:	125				

PFA34 TRUCK/MOD-ELC5/1556 1					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	65	0.0000000000
DEPOT	C_89	0	19	84	15.000000000
C_89		74	10	84	6.0000000000
C_89	C_57	84	48	140	6.0000000000
C_57		132	10	140	0.0000000000
C_57	DEPOT	142	29	230	0.0000000000
order DEPOT C_89 9.0000000000 0 0 84 74 10 84					
order DEPOT C_57 6.0000000000 0 0 140 130 10 140					
Total time needed:	171				
Waiting time:	55				
Maximal gap:	55				
Tatal distance to go:	96				

PFA35 TRUCK/MOD-ELC5/1553 1					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	84	0.0000000000
DEPOT	C_41	0	11	95	25.000000000
C_41		85	10	95	16.000000000
C_41	C_54	95	6	105	16.000000000
C_54		101	10	105	2.0000000000
C_54	C_56	111	27	146	2.0000000000
C_56		138	10	146	0.0000000000
C_56	DEPOT	148	30	230	0.0000000000
order DEPOT C_56 2.0000000000 0 0 146 136 10 146					
order DEPOT C_54 14.000000000 0 0 105 95 10 105					
order DEPOT C_41 9.0000000000 0 0 95 85 10 95					
Total time needed:			178		
Waiting time:			74		
Maximal gap:			74		
Tatal distance to go:			74		

PFA36 TRUCK/MOD-ELC6/2136 0					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	15	0.0000000000
DEPOT	C_96	0	14	29	108.00000000
C_96		14	10	29	88.000000000
C_96	C_28	24	18	47	88.000000000
C_28		42	10	47	72.000000000
C_28	C_31	52	20	81	72.000000000
C_31		72	10	81	51.000000000
C_31	C_50	82	34	118	51.000000000
C_50		116	10	118	21.000000000
C_50	C_97	126	44	192	21.000000000
C_97		170	10	192	10.000000000
C_97	C_99	180	6	198	10.000000000
C_99		186	10	198	0.0000000000
C_99	DEPOT	196	21	230	0.0000000000
order DEPOT C_99 10.000000000 0 0 198 0 10 198					
order DEPOT C_97 11.000000000 0 0 204 0 10 204					
order DEPOT C_96 20.000000000 0 0 205 0 10 205					
order DEPOT C_50 30.000000000 0 0 118 108 10 118					
order DEPOT C_31 21.000000000 0 0 81 71 10 81					
order DEPOT C_28 16.000000000 0 0 47 37 10 47					
Total time needed:			217		
Waiting time:			0		
Maximal gap:			0		
Tatal distance to go:			157		

PFA39 TRUCK/MOD-ELC6/2130 0					
From:	To:	EST:	DUR:	LFT:	Amount:
DEPOT		0	0	22	0.0000000000
DEPOT	C_6	0	20	42	101.00000000
C_6		20	10	42	75.000000000
C_6	C_12	30	35	77	75.000000000
C_12		67	10	77	63.000000000
C_12	C_9	77	24	105	63.000000000
C_9		101	10	105	54.000000000
C_9	C_8	111	12	143	54.000000000
C_8		123	10	143	49.000000000
C_8	C_14	133	26	169	49.000000000
C_14		159	10	169	26.000000000
C_14	C_38	169	11	193	26.000000000
C_38		180	10	193	18.000000000
C_38	C_59	190	17	210	18.000000000
C_59		207	10	210	0.0000000000
C_59	DEPOT	217	9	230	0.0000000000
order DEPOT C_59 18.000000000 0 0 210 200 10 210					
order DEPOT C_38 8.0000000000 0 0 198 0 10 198					
order DEPOT C_14 23.000000000 0 0 169 159 10 169					
order DEPOT C_12 12.000000000 0 0 77 67 10 77					
order DEPOT C_9 9.0000000000 0 0 105 95 10 105					
order DEPOT C_8 5.0000000000 0 0 198 0 10 198					
order DEPOT C_6 26.000000000 0 0 199 0 10 199					
Total time needed:			226		
Waiting time:			2		
Maximal gap:			2		
Tatal distance to go:			154		

Name:	Dist.:	Wait:	Time:	Stops:
PFA10	54	67	131	1
PFA34	96	55	171	2
PFA20	70	59	139	1
PFA28	48	75	143	2
PFA31	155	24	229	5
PFA13	126	17	213	7
PFA32	122	25	177	3
PFA23	143	3	216	7
PFA29	34	66	110	1
PFA39	154	2	226	7
PFA21	60	38	128	3
PFA36	157	0	217	6
PFA14	131	12	213	7
PFA11	40	58	108	1
PFA17	89	74	183	2
PFA33	125	8	223	9
PFA15	142	0	222	8
PFA25	84	41	135	1
PFA30	142	3	225	8
PFA35	74	74	178	3
PFA18	66	64	160	3
PFA19	146	5	221	7
PFA16	134	19	213	6
$\sum$	2392	789	4181	100