An Implementation of the Contract Net Protocol Based on Marginal Cost Calculations

Tuomas Sandholm

Computer Science Department
University of Massachusetts
Amherst, Massachusetts 01003
sandholm@cs.umass.edu

Abstract

This paper presents a formalization of the bidding and awarding decision process that was left undefined in the original contract net task allocation protocol. This formalization is based on marginal cost calculations based on local agent criteria. In this way, agents having very different local criteria (based on their self-interest) can interact to distribute tasks so that the network as a whole functions more effectively. In this model, both competitive and cooperative agents can interact. In addition, the contract net protocol is extended to allow for clustering of tasks, to deal with the possibility of a large number of announcement and bid messages and to effectively handle situations, in which new bidding and awarding is being done during the period when the results of previous bids are unknown. The protocol is verified by the TRACONET (TRAnsportation COoperation NET) system, where dispatch centers of different companies cooperate automatically in vehicle routing. The implementation is asynchronous and truly distributed, and it provides the agents extensive autonomy. The protocol is discussed in detail and test results with real data are presented.[1]

1 Introduction

The *contract net protocol* (CNP) (Smith 1980; Smith & Davis 1981; Davis & Smith 1988) for decentralized task allocation is one of the important paradigms developed in distributed artificial intelligence (DAI). Its significance lies in that it was the first work to use a negotiation process involving a *mutual selection* by both *managers* and *contractors*. It was initially applied to a simulated distributed acoustic sensor network. In this interpretation application, the agents were totally cooperative, and selection of a contractor was based on suitability, for example adjacency, processing capability, and current agent load. However, there was no formal model discussed in this work for making task *announcing*, *bidding* and *awarding* decisions. This paper presents such a formal model, where agents locally calculate their marginal costs for performing sets of tasks. The choice of a contractor is based solely on these costs. The pricing mechanism generalizes the CNP to work for both cooperative and competitive agents. Another important issue not covered in previous work on the CNP is the risk attitude of an agent toward being committed to activities it may not be able to honor, or the honoring of which may turn out to be unbeneficial. Additionally, in previous CNP implementations, tasks have been negotiated one at a time. This is not sufficient, if the effort of carrying out a task depends on the carrying out of other tasks. The framework is extended to handle *task interactions* by clustering tasks into sets to be negotiated over as atomic bargaining items. Finally, the practical problem of announcement message congestion is solved.

Our case problem, vehicle routing, is structured in terms of a number of geographically dispersed *dispatch centers* of different companies. Each center is responsible for the deliveries initiated by certain factories and has a certain number of vehicles to take care of the deliveries. The geographical main operation areas of the centers overlap considerably. This provides for the potential for multiple centers to be able to handle a delivery. Every delivery has to be included in the route of some vehicle. The local problem of each agent is a heterogeneous fleet multi-depot routing problem, where the vehicle attributes include cost per kilometer, maximum route duration, maximum route length, maximum load weight and maximum load volume (Sandholm 1992a). The objective is to minimize the transportation costs.

In solving the problem, each dispatch center - represented by one intelligent agent[2] - first solves its local

[1]Primary support for this work came from the Technology Development Centre of Finland, during the period which the author was working at the Technical Research Centre of Finland, Laboratory for Information Processing, Lehtisaarentie 2A, SF-00340 Helsinki, Finland. Additional support comes from DARPA contract N00014-92-J-1698. The content of the information does not necessarily reflect the position or the policy of the Government and no official endorsement should be inferred.

[2]Another choice would be that each agent represented one vehicle. This small grain size approach would probably not be as efficient, because such a large number of agents would congest the negotiation network and the method would be too

routing problem. After that, an agent can potentially negotiate with other dispatch agents to take on some of their deliveries or to let them take on some of its deliveries for a dynamically constructed charge. In the negotiations the agents exchange sets of deliveries whenever this is profitable, i.e., whenever a contractor is able to carry out the task set with less costs than the manager agent. The negotiations can be viewed as an iterative way of making the routing solution better by going through only feasible solutions.[3] Here 'feasible' means that each center can take care of all of its deliveries. This is how a solution closer to the global optimum is reached although no global optimization run is performed. The use of contract nets as opposed to centralized problem solving is most fruitful in operative decision making in volatile domains such as ours and the factory domain of (Parunak 1987).

The negotiation is real-time since after each contract is made the exchange of deliveries is made immediately. Thus, between individual negotiations some delivery orders may have been dispatched, new orders may have arrived, and the available vehicles may have changed. There is no iteration among the agents until an equilibrium is reached unlike the approach of (Wellman 1992), where the bids include a number of the similar items an agent wants to buy and it is assumed that the purchase of one type of items is independent of the purchase of other types of items. In our system, each item (task set) is different and task sets of different announcements are highly interdependent. In the equilibrium approach of (Kuwabara and Ishida 1992), at each iteration, the seller sets the price based on demand and the buyers state the quantity they want to buy.

Section 2 presents the architecture of our implementation. Section 3 discusses the local control strategy of an agent. In sections 4 to 7, the negotiation phases of announcing, bidding, awarding and award taking are detailed respectively. Section 8 presents test results with real data and section 9 concludes.

2 TRACONET Architecture

The vehicle routing application is implemented in a system called TRACONET (TRAnsportation COoperation NET).[4] The asynchronous automatic negotiations in TRACONET resemble a directed government contracting scheme, where each involved party is allowed to make one bid for each announcement it receives, and the bids of the other parties are not revealed to it. The negotiations are directed in the sense that an announcement is not sent to all other agents (Parunak 1987), fig. 1. The agents have no fixed hierarchy among themselves. An agent can act both as a manager and a contractor of delivery sets, but it does not have to take both roles, nor is it required to negotiate with all other agents. Further, each agent can reallocate deliveries received from other agents. When announcing, an agent tries to buy some other agent's transportation services at a price, the maximum of which it specifies in the announcement. When bidding, an agent tries to sell its own services at a price, the minimum of which it specifies in the bid. Awarding means actually buying the services of some other center and award taking means actually selling one's services. Unlike the original CNP, in the awarding phase explicit *loser messages* are sent, fig. 1. These messages free the bidder agents from the commitment of their bids, which affects the pricing of new bids and the evaluation of other agents' bids as will be described. Another option would be to consider a bid a loser if it has not received an award within a time limit, but this does not fit our asynchronous approach, because it forces the manager to award within a strict time limit. The time to analyze bids varies depending on the state of the agent and the number of messages received by it. At this point, we do not know how to realistically set an appropriate upper bound for this time. In our approach, we introduce additional message traffic, which hopefully results in more accurate announcing, bidding and awarding, since the agent will know early on, which of its bids it still may have to honor.

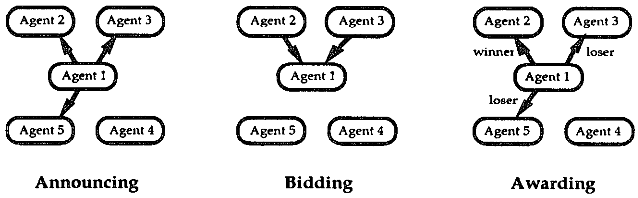

Figure 1. Message passing, when agent 1 gives a set of deliveries to agent 2 to be done.

Each agent has two main parts: the *bargaining system* and the *local optimizer*. The bargaining system is divided into four major components: the *announcer*, the *bidder*, the *awarder* and the *award taker*. The bargaining system is not restricted to any specific local optimization algorithm[5], but the local optimizer has to provide five services. These relate to the counting of marginal costs of a set of deliveries (to remove or to add), to optimizing all deliveries of an agent and to removing and adding sets of deliveries to the agent's routing solution. Agents in the same negotiation network can use different local optimization algorithms tuned to the requirements of each center separately. The local optimizer services could also be given manually by a transportation coordinator in dispatch centers that do not use automatic optimization. Interactive routing is discussed in (Waters 1984) and (Powell & Sheffi 1989).

opportunistic. When the number of vehicles is small, this approach does work, though. An example is given in (McElroy et al. 1989), where automatically guided vehicles transport items inside a factory.

[3]Centralized versions of iterative routing are discussed in (Waters 1987) and (Wong & Beasley 1984).

[4]The system is implemented in an object-oriented fashion using the C++ language and the X11 Window System on a network of HP 9000 workstations. Each agent is implemented as one HP-UX (UNIX) process. The agents negotiate over the file system and share no memory.

[5]A good overview of centralized routing algorithms is given in (Bodin et al. 1983).

3 Local control

In TRACONET, an agent first calls its own local optimizer to make the routing decisions concerning the deliveries and vehicles that belong to the associated dispatch center. Based on these initial solutions, the agents start the negotiations. During the negotiations, the *local control loop* of an agent repeatedly goes through a sequence of invoking the bidder, awarder, award taker and announcer. The bidder, awarder and award taker handle all the messages that have been received by the time of their calls. In contrast, the announcer sends at most one announcement to agents during one local control loop cycle. It is preferable to first handle all received messages before sending a new announcement, so that the agents do not get congested by announcements, and announcements are constructed according to the most up to date view of the agent's local routing decisions. The messages received during the operation of the bidder, awarder or award taker are handled on the next cycle of the local control loop. This prevents the system from getting stuck at any single phase even if large amounts of messages are coming in.

An agent can enter and exit the negotiation network dynamically. When joining the network the agent first deletes all announcements and loser messages that may have accumulated in the incoming message media. Then the agent is ready for the negotiations. However, exiting the negotiation process is not as simple for two reasons. First, some other agent might be awarding a delivery set to the agent and if the agent has exited the negotiations, it will not receive the award. Secondly, some other agent might be making a bid to the agent and if the agent exits the negotiation, the other agent does not receive even a loser message for the bid and will not be freed from the commitment of its bid. The second problem is solved by sending a loser message to the other agents for all unhandled announcements sent to them previously. The first problem is solved by going through a listening phase before logging out of the network. During this phase no announcements and no bids are made. The phase can be ended, when replies (awards or loser messages) have been received for all unhandled bids that have been sent out. If an agent wants to *reoptimize* its local solution, it must first exit the negotiations, reoptimize and then possibly rejoin the negotiations. If the agent did not exit temporarily, the marginal costs calculated before reoptimization would not be valid after it.

4 Announcing

An agent's announcer chooses a set of deliveries from the deliveries of the center and announces them to other centers in order to get bids from them. In the implementation the announcements focus on deliveries ending in the geographical main operation areas of the potential contractors, because these deliveries are most likely to lead to contracts. The announcing methods differ from each other in the number of tasks (deliveries) to be clustered into each announcement, and in whether a delivery set that has already been announced can be reannounced (Sandholm 1992b). Reannouncing leads to better results, but the negotiations are considerably longer. This, however, is not a serious problem, if we assume that actual deliveries are being done during the negotiations and reannouncing is not done immediately. In algorithm 1, a set of deliveries consists of only one (randomly chosen) delivery, and reannouncing is allowed. The $c'_{rem}(T)$ service provided by the local optimizer gives a heuristic approximation of the marginal cost $c_{rem}(T)$ saved if the delivery set T were removed from the routing solution of the agent. The implemented calculation of $c'_{rem}(T)$ will be described in section 6. If the estimate $c'_{rem}(T)$ is too low, the other center's will not bid even though that might be beneficial. On the other hand, if the estimate is too high, the agent will receive also unbeneficial bids. The actual value of $c'_{rem}(T)$ is not as crucial here as it is in the awarding phase, because announcements are not binding. Therefore, even an incorrect calculation of $c'_{rem}(T)$ will not lead to unbeneficial contracting.

Randomly choose one of the deliveries ending in another center's main operation area.
T = {the chosen delivery}.
Maximum price of the announcement $c_{max} = c'_{rem}(T)$.
For all centers except this center itself
    If the end stop of the delivery is in the center's main operation area
        Then send an announcement to the center.

Algorithm 1. A simple announcer algorithm.

Announcing one delivery at a time is not sufficient in general. This is due to the fact that the deliveries are dependent, i.e., for two disjoint delivery sets T_1 and T_2, for the manager, $c_{rem}(T_1 \cup T_2) \neq c_{rem}(T_1) + c_{rem}(T_2)$. For example, if the removal cost of either of two deliveries alone is small, but the removal cost of both of them together is large, announcing one delivery at a time would probably not lead to a contract, but announcing two at a time probably would. For the tasks to be truly independent, the following would also have to hold for each potential contractor: $c_{add}(T_1 \cup T_2) = c_{add}(T_1) + c_{add}(T_2)$, where $c_{add}(T)$ gives the marginal cost of adding task set T to the agent's routing solution, as will be explained in section 5. The clustering of tasks into (not necessarily disjoint) sets to be bargained over as atomic bargaining items is a complex problem. To solve it, TRACONET's more refined announcer algorithms use domain dependent heuristics. These algorithms and experiments with them in a domain, where all deliveries originate at a common factory have been discussed in (Sandholm 1992b). For example, in one of them, a delivery d_1 was clustered with another delivery d_2, the end stop of which was next to the end stop of d_1 in a route, if $c'_{rem}(\{d_1, d_2\}) > \alpha * c'_{rem}(\{d_1\})$, where α was a constant.

If no more beneficial contracts of any k tasks at a time can be made between any two agents, the solution is called *k-optimal*, which is a necessary, but not a sufficient condition for optimality. Neither does m-optimality guarantee n-optimality, if $n \neq m$.

5 Bidding

An agent's bidder reads the announcements sent by other agents. If the maximum price mentioned in the announcement is higher than the price that the deliveries would cost if done by this center, a bid is sent with the latter price. Otherwise, no bid is sent for the specified announcement. Denote an arbitrary bid by b and the set of tasks of that bid by T_b. Let B_{uns} be the set of unsettled bids sent by an agent previously. Define B_{pos} to be the set of possible bids that can be awarded to the agent when b is also awarded to the agent, i.e., $B_{pos} = \{x \mid x \in B_{uns}, T_x \cap T_b = \varnothing\}$. Let T_{cur} be the current set of tasks of the agent. Let function f(T) compute the total cost of the local optimal solution with task set T. Let $c_{add}(T)$ be the marginal cost of adding task set T into the local solution. For any bid b, the cost $c_{add}(T_b)$ is bounded below by

$$c^{-}_{add}(T_b) = \min_{B \subseteq B_{pos}} \left[f(T_b \cup T_{cur} \cup \bigcup_{z \in B} T_z) - f(T_{cur} \cup \bigcup_{z \in B} T_z) \right],$$

and above by

$$c^{+}_{add}(T_b) = \max_{B \subseteq B_{pos}} \left[f(T_b \cup T_{cur} \cup \bigcup_{z \in B} T_z) - f(T_{cur} \cup \bigcup_{z \in B} T_z) \right].$$

Setting the bid price to be $c^{-}_{add}(T_b)$ is an opportunistic approach, and setting it to be $c^{+}_{add}(T_b)$ is a safe approach. Assuming that all of the unsettled bids sent by the agent will be awarded to the agent, the bid price can be calculated by

$$c^{all}_{add}(T_b) = f(T_b \cup T_{cur} \cup \bigcup_{z \in B_{pos}} T_z) - f(T_{cur} \cup \bigcup_{z \in B_{pos}} T_z),$$

and assuming that none of the unsettled bids sent by the agent will be awarded to it, the bid price is as follows:

$$c^{non}_{add}(T_b) = f(T_b \cup T_{cur}) - f(T_{cur}).$$

Clearly, $c^{-}_{add}(T_b) \leq c^{all}_{add}(T_b) \leq c^{+}_{add}(T_b)$ and $c^{-}_{add}(T_b) \leq c^{non}_{add}(T_b) \leq c^{+}_{add}(T_b)$, but the partial order of $c^{all}_{add}(T_b)$ and $c^{non}_{add}(T_b)$ varies. This is because in this domain, both economies of scale (implying $c^{all}_{add}(T_b) < c^{non}_{add}(T_b)$) and diseconomies of scale (implying $c^{non}_{add}(T_b) < c^{all}_{add}(T_b)$) are present. In (Wellman 1992), only diseconomies of scale are present.

The cost $c^{non}_{add}(T_b)$ is faster to compute than $c^{all}_{add}(T_b)$, and it gives a better approximation of $c_{add}(T_b)$ when bids are seldom awarded to the agent. This is usually the case, if the network has many agents.

In the original CNP, an agent could have multiple bids concerning different contracts pending concurrently in order to speed up the operation of the system (Smith 1980). We have followed this approach for the same reason, although negotiations over only one contract at a time allow a more precise bid price. If only one bid is allowed to be pending from one agent at a time, $B_{pos} = \varnothing$ and $c^{-}_{add}(T_b) = c^{+}_{add}(T_b) = c^{all}_{add}(T_b) = c^{non}_{add}(T_b)$. Fig. 2 compares results of allowing multiple bids and awards simultaneously to those of allowing only one announcement (implying only one award) and one bid at a time.

Calculation of the local utility function takes time. This has not been taken into account in the CNP or in work in game theory. In our domain, calculating the marginal costs (and therefore the announcing, bidding and awarding) takes computational time. Because the calculation of the truly optimizing function f takes exponential time in our domain, we use a heuristic approximation f', for which $f(T) \leq f'(T)$ for any task set T. In our domain, the calculation of $f'(T \cup T_{cur})$ would be very fast if we knew $f(T_{cur})$, because it could be calculated incrementally by just adding the new tasks T to the solution without altering the original solution. The problem is that we do not know the optimal $f(T_{cur})$, but only a heuristic approximation $f'(T_{cur})$ of it. In the tests presented in this paper, the bid price $c'_{add}(T_b)$ was calculated incrementally like this with respect to the current heuristic solution assuming that none of the agent's unsettled bids are awarded to it. This assumption makes the calculation semi-opportunistic. Therefore an agent using this strategy may make unbeneficial contracts now and then. A safe approach would be to use a heuristic upper bound for $c^{+}_{add}(T_b)$ as the bid price, but its calculation is slower than that of $c'_{add}(T_b)$.

Read in all received announcements and call this set A.
For each announcement $a \in A$
  Call the set of deliveries in a T_a and the maximum price c_{max}.
  If $f'(T_{cur} \cup T_a \cup T_{pos}) < \infty$ (Feasibility check; T_{pos} defined w.r.t. a potential bid b with the deliveries of a.)
    Set $c_{bid} = c'_{add}(T_a)$.
    If $c_{bid} < c_{max}$
      Send a bid with the identifier of the announcement, the name of this center and cost c_{bid}.

Algorithm 2. The bidding algorithm.

Because of binding bids, a feasibility check in algorithm 2 checks that the agent's transportation solution will be feasible even if all of the previous unsettled possible bids and this bid are awarded to the agent. In domains (unlike ours), where the feasibility check often restricts the bidding, the bidder should choose the most profitable combination among the possible combinations of beneficial bids to send.

Using the previously discussed bidding methods, the negotiation network got congested with announcements, i.e., some of the agents were receiving announcements at a faster pace than they could process. The problem occurred only with announcements, because in our domain the number of them far exceeds the number of other messages. The reason the congested agents could not keep in pace was that the time to handle an announcement increased with the number of previously sent unsettled bids – mainly because of the feasibility check. The more announcements an agent had received, the more bids it was able to make, which slowed it down, and during the bidding process even more announcements kept coming in. The congestion problem was solved by making the bidder consider only announcements newer than a certain time limit. This is sensible also, because bids made on older announcements would probably not get to the managers before the negotiations concerning these announcements would be over.

6 Awarding

An agent's awarder reads the bids of other agents. Before handling the bids concerning a certain announcement, it checks that a fixed time has passed since the sending of the announcement, so that many potential contractors have had time to bid. An award or loser message is sent to every agent to whom an announcement concerning the same contract was sent earlier. The award is sent to the agent with the most inexpensive bid.[6] After an award is sent, the awarder removes the set of deliveries from the agent's current deliveries T_{cur} and from its transportation solution. If no bids for an announcement have been received by the time of the mentioned time limit, the awarding is postponed until the first bid for this announcement is received. If this takes longer than a second time limit, the agent simply forgets that it has made such an announcement and sends loser messages to all agents to whom the announcement was sent previously. Bids received later for this announcement are deleted.

In the awarding phase the manager has a chance to check that awarding is still beneficial to itself, i.e., it does not have to accept any bid. In deciding whether the awarding is beneficial, the manager has to also consider the unsettled bids that it has sent. Awarding to bid b is beneficial iff $c_{rem}(T_b) > c_b$, where c_b is the price mentioned in the bid b, and $c_{rem}(T_b)$ is the cost of removing the tasks T_b from the manager's own local solution. Unlike in the bidding phase, $B_{pos} = B_{uns}$. The cost $c_{rem}(T_b)$ is bounded above by

$$c^{+}_{rem}(T_b) = \max_{B \subseteq B_{pos}} \left[f(T_{cur} \cup_{z \in B} T_z) - f((T_{cur} - T_b) \cup_{z \in B} T_z) \right],$$

and below by

$$c^{-}_{rem}(T_b) = \min_{B \subseteq B_{pos}} \left[f(T_{cur} \cup_{z \in B} T_z) - f((T_{cur} - T_b) \cup_{z \in B} T_z) \right].$$

Assuming that all of the agent's unsettled bids will be awarded to it, $c_{rem}(T_b)$ is calculated by

$$c^{all}_{rem}(T_b) = f(T_{cur} \cup_{z \in B_{pos}} T_z) - f((T_{cur} - T_b) \cup_{z \in B_{pos}} T_z),$$

[6]If some of the deliveries of the announcement have already been awarded out by an award of some other announcement, all messages sent are loser messages.

and assuming that none of the agent's unsettled bids will be awarded to it, $c_{rem}(T_b)$ is calculated as follows:

$$c^{non}_{rem}(T_b) = f(T_{cur}) - f(T_{cur} - T_b).$$

Clearly, $c^{-}_{rem}(T_b) \le c^{all}_{rem}(T_b) \le c^{+}_{rem}(T_b)$ and $c^{-}_{rem}(T_b) \le c^{non}_{rem}(T_b) \le c^{+}_{rem}(T_b)$, but the partial order of $c^{all}_{rem}(T_b)$ and $c^{non}_{rem}(T_b)$ varies. If only one bid is allowed to be pending from an agent at a time, then $[c^{all}_{rem}(T_b) = c^{-}_{rem}(T_b)$ and $c^{non}_{rem}(T_b) = c^{+}_{rem}(T_b)]$ or $[c^{non}_{rem}(T_b) = c^{-}_{rem}(T_b)$ and $c^{all}_{rem}(T_b) = c^{+}_{rem}(T_b)]$.

Similar to our discussion of f', because calculating the truly optimizing f function takes a long time, we use a heuristic approximation f'', for which $f(T) \le f''(T)$ for any task set T. In our domain, the calculation of $f''(T_{cur} - T_b)$ would be fast if we knew $f(T_{cur})$, because it could be calculated decrementally by just removing the tasks T_b from the solution without altering the original solution. The problem is that we do not know the optimal $f(T_{cur})$, but only a heuristic approximation $f''(T_{cur})$ of it. In the tests presented in this paper, the benefit check price $c'_{rem}(T_b)$ was calculated decrementally like this with respect to the current heuristic solution assuming that none of the agent's unsettled bids are awarded to it. The assumption makes this calculation semi-opportunistic, and an agent using this strategy may have to take unbeneficial awards later. A safe approach would be to use a heuristic upper bound for $c^{+}_{rem}(T_b)$ as the benefit check price, but its calculation is slower than that of $c'_{rem}(T_b)$.

In the current implementation, all bids received before the start of the awarding phase are handled in order of receipt before going to any next negotiation phase. If the check for benefit is used, the order of awarding may be important - though this seldom is the case in our domain. The awarding of one task set may disable the beneficial awarding of another. Usually the number of received bids per local control loop cycle is small, so the awarder could try all possible orderings of awarding sets of deliveries and carry out the best ordering.

7 Taking awards

An agent's award taker reads the awards and inserts the deliveries from the awards to the agent's deliveries T_{cur} and its transportation solution. Some contracts may have sneaked in between the bidding for a certain set of deliveries and taking the corresponding award. These contracts have altered the routing solution. If opportunistic pricing is used, taking the award might no longer be profitable for the center. Because bids are binding, the center is committed to take the award anyway. Making bids non-binding would not solve the problem, because the contractor, after receiving an award, would have to inform the manager that it has taken the award or that it will not take it. This would require the manager to keep the delivery set in its routing solution until award taking is confirmed, during which, some changes may have sneaked into its routing solution and the problem rearises.

8 Experimental results

The purpose of the experiments was to validate the distributed problem solving approach in reducing the total transportation costs among autonomous dispatch centers. A detailed presentation of these experiments is given in (Sandholm 1992a). Table 1 provides results of one example experiment. As can be seen, the negotiations led to considerable transportation cost savings in reasonable time even in such a large problem. In the experiment, company A owned the first three centers and company B owned the last two. The centers were located around Finland. The agents had similar local optimization modules and each agent's original local routing solution was acquired heuristically using a parallel insertion algorithm (Sandholm 1992a). Each agent executed on its own HP 9000 s300 workstation. The profit of each contract was divided in half between the agents, i.e., the actual price of a contract was half way between the maximum price mentioned in the announcement and the bid price. A choice closer to a real world competing agent contracting scheme would be to let the contract price equal the bid price. In 30 minutes, each agent goes through its main control loop 100 - 200 times.

Dispatch center	Deliveries	Vehicles	Average delivery length	Cost savings in 15 minutes	Cost savings in 30 minutes
A1	65	10	121 km	5%	6%
A2	200	13	169 km	12%	18%
A3	82	21	44 km	31%	34%
B1	124	18	145 km	11%	23%
B2	300	15	270 km	9%	15%
Total	771	77	187 km	11%	17%

Table 1. Columns 2 - 4 characterize the one week real vehicle and delivery data of the experiments, and the last two columns show results of the negotiations.

Figure 2 presents example runs with two unsafe bidding schemes. Due to the semi-opportunistic pricing explained before, the local costs of the agents do not decrease monotonically in case 1. An agent is forced to take unbeneficial awards now and then. The unbeneficial contracts are somewhat compensated for by other contracting within the time window shown. The cost of an agent in case 1 decreases faster (in the sense of local control loop cycles required) than in case 2. In case 2, the cost decreased monotonically for every agent. To guarantee monotonic decrease of the cost using opportunistic pricing, one bid at a time should be allowed and awarding should be allowed only when no bid is pending from the agent. This would require even more local control loop cycles than case 2, where awarding can happen while a bid is pending.

In case 1, the agents have to consider more messages on each local control loop cycle. Therefore, the previously mentioned time limits were set to be longer in case 1, and in the same actual time, the agents of case 2 go through more main control loop cycles than in case 1.

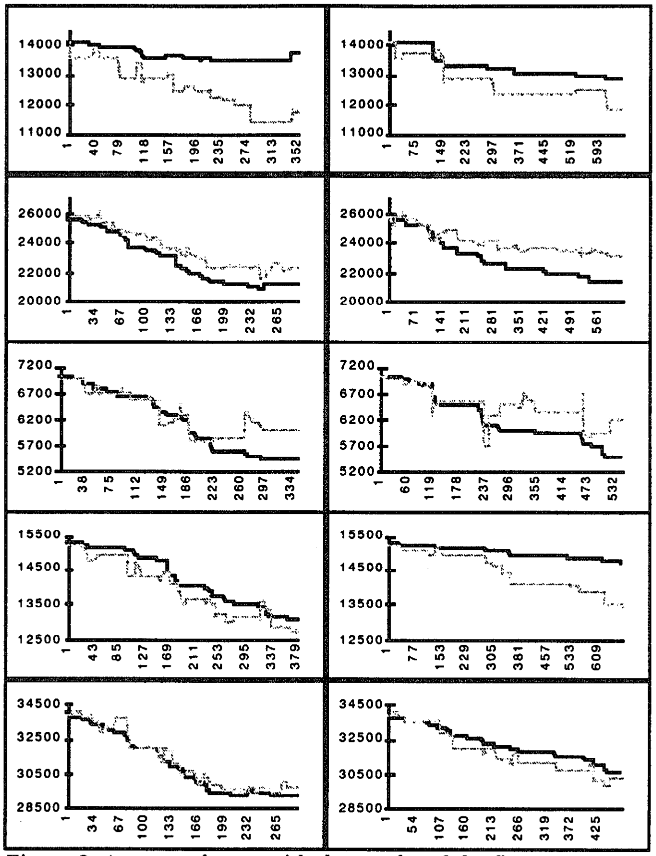

Figure 2. An example run with the results of the five agents one below another. The x-axis show the number of local control loop cycles for each agent. The thin gray line shows the evolution of the total length of the truck routes of an agent in kilometers. The black line shows the evolution of the local cost for each agent, so the black line takes into account the amounts paid by the managers to the contractors for carrying out the transportation tasks. The figures in the left column (case 1) show the normal case, where multiple announcements and bids are allowed simultaneously. The right column (case 2) shows the case, where only one announcement (implying at most one award) and one bid are allowed to be pending from one agent at a time.

9 Conclusions

The role of DAI systems with cooperative and competitive agents is likely to increase in the future. Especially important will be *enterprise cooperation*: allowing autonomous, even competitive, enterprises to cooperate through the on-line, dynamic establishment of contracts among enterprises. The groundwork for computerizing this cooperation is currently being made by building networks of enterprises with electronic data interchange. This paper presents, to our knowledge, the first prototype of an application where different enterprises work together automatically using DAI techniques. Our methodology is presented through a concrete application domain, vehicle routing, but it is applicable to other task allocation problems - assuming that a reasonable local objective function is known for each agent.

TRACONET uses task negotiation. Another solution technique for the same problem is to negotiate over resources. If there are many tasks per resource (eg. many deliveries in one truck route), a higher resolution of cooperation is achieved by exchanging tasks. All possible solutions reached by resource exchange can be reached by task exchange, but not vice versa, so the best possible solution when negotiating tasks is at least as good as the best possible solution when negotiating resources. This does not necessarily imply that after a certain number of iterations, the solution using task negotiation is as good or better than the solution using resource negotiations. Also, if we use a limit on the maximum number of tasks per announcement, it may happen that the best global solution of task negotiations can not be reached at all. If fast computation is crucial, the coarser grain size negotiations - resource negotiations in this case - may be preferred. In domains with many resources per task, the above arguments should be reversed.

We have extended the CNP with a formal model for making announcing, bidding and awarding decisions based on local marginal cost calculations. Additionally, announcing, bidding and awarding are allowed while the results of previous bids are still unknown. Safe and opportunistic pricing policies are discussed: opportunism speeds up the negotiations, but safe policies guarantee monotonic decrease of the local cost. Task interaction is handled by heuristically clustering tasks into announcements negotiated over atomically. The implementation is asynchronous and truly distributed and solves the message congestion problems.

At this stage, the announcing, bidding and awarding decisions do not *anticipate future contracts*. Future research also includes *estimating the marginal costs* when a local solution does not exist, so that the agents could negotiate before they solve the local routing problem, and even if a feasible solution to the local problem does not exist at the moment. In the future we wish to extend the protocol for *contracts involving multiple agents.* In TRACONET, the bidder can only bid for the announced task sets, but allowing *counterproposals* with different content may speed up the negotiations. Currently there is just one focus in the contract space and it is committal. Moving *non-committal foci* in the contract space would enable jumping over local minima, because multiple contracts would be made before the agents have to commit. Finally, other than per centual *profit division* mechanisms, and intelligent *local reoptimization activation* should be implemented.

Acknowledgements

I would like to thank professor Victor Lesser from the University of Massachusetts at Amherst, Computer Science Department, and research professor Seppo Linnainmaa from the Technical Research Centre of Finland, Laboratory for Information Processing, for their support.

References

Bodin, L. et al. 1983. Routing and scheduling of vehicles and crews: The state of the art. *Computers and Operations Research* 10(2):63-211.

Davis, R., and Smith, R.G. 1988. Negotiation as a Metaphor for Distributed Problem Solving. In: Bond, A., and Gasser, L. eds. *Readings in Distributed Artificial Intelligence*, 333-356. San Mateo, Calif.: Morgan Kaufmann.

Kuwabara, K., and Ishida, T. 1992. Symbiotic Approach to Distributed Resource Allocation: Toward Coordinated Balancing. In Proceedings of the European Workshop on Modeling Autonomous Agents and Multi-Agent Worlds '92.

McElroy, J. et al. 1989. Communication and cooperation in a distributed automatic guided vehicle system. In Proceedings of the IEEE Southeastcon '89, 999-1003.

Parunak, H.V.D. 1987. Manufacturing Experience with the Contract Net. In: Huhns, M. ed. *Distributed Artificial Intelligence*, 285-310. Los Altos, Calif.: Morgan Kaufmann.

Powell, W., and Sheffi, Y. 1989. Design and implementation of an interactive optimization system for network design in the motor carrier industry. *Operations Research* 37(1):12-29.

Sandholm, T. 1992a. Automatic Cooperation of Dispatch Centers in Vehicle Routing. M.Sc. Thesis. Research Report No. J-9, Laboratory for Information Processing, Technical Research Centre of Finland.

Sandholm, T. 1992b. Automatic Cooperation of Area-Distributed Dispatch Centers in Vehicle Routing. In Preprints of the International Conference on Artificial Intelligence Applications in Transportation Engineering, 449-467. Institute of Transportation Studies, Univ. of Calif., Irvine.

Smith, R.G. 1980. The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver. *IEEE Trans. on Computers* C-29(12):1104-1113.

Smith, R.G., and Davis, R. 1981. Frameworks for Cooperation in Distributed Problem Solving. *IEEE Trans. on Systems, Man, and Cybernetics* 11(1):61-70.

Waters, C.D. 1984. Interactive vehicle routing. *Journal of the Operational Research Society* 35(9):821-826.

Waters, C.D. 1987. A solution procedure for the vehicle-scheduling problem based on iterative route improvement. *Journal of the Operational Research Society* 38(9):833-839.

Wellman, M. 1992. A General-Equilibrium Approach to Distributed Transportation Planning. Proceedings of the AAAI -92, 282-289.

Wong, K., and Beasley, J. 1984. Vehicle routing using fixed delivery areas. *Omega International Journal of Management Science* 12(6):591-600.