Improvements to the FIPA Contract Net Protocol for Performance Increase and Cascading Applications

Tore Knabe, Michael Schillo, Klaus Fischer

July 12, 2002

Abstract

The *Contract Net Protocol* (CNP) is a widely used protocol in DAI, as it proved to be a flexible and low communication interaction protocol for task assignment. The situation it is best suited for is that of a single task to be assigned among a number of individual agents. However, it has shortcomings if the setting for task assignment is more complicated. If there are several agents who concurrently start protocols to assign tasks, early commitment of bidder agents in the standard CNP leads to suboptimal outcomes, as solutions that are possible are not found. We propose the *Contract Net With Confirmation Protocol* (CNCP), an extension to the CNP that avoids the problems of early commitment. In scenarios where tasks are assigned in cascades, for example in holonic agents, this extension takes the form of the *Holonic Contract Net With Confirmation Protocol* (HCNCP). Examples of settings where the extensions improve on the standard protocol are given, as well as a discussion of the new protocols.

1 Introduction

The assignment of tasks to agents and the (re-)allocation of tasks in a multiagent system (MAS) is one of the key features of automated negotiation systems [18]. The contract net protocol (CNP), originally proposed in [16], and other more general auction mechanisms can be widely applied to resource and task allocation problems. The contract net has been applied e.g. to online dispatching in the transportation domain [1, 5], meeting scheduling [6, 14] and flexible manufacturing [15, 11, 10]. Our discussion is based on the FIPA interpretation of the contract net [3], which is a minor modification of

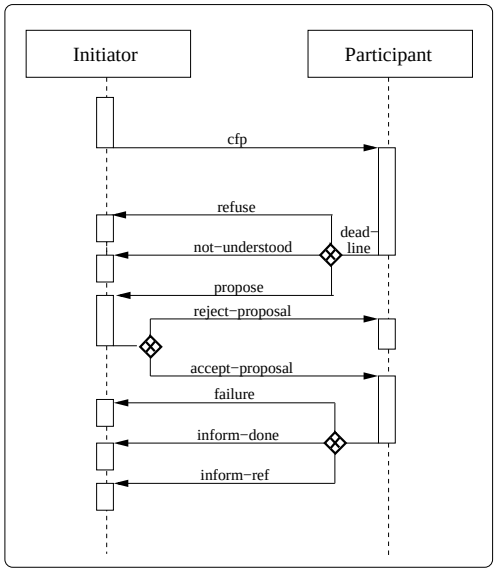

Figure 1: The FIPA Contract Net Protocol.

the original protocol in that it adds rejection and confirmation speech acts. Currently, this interpretation is the standard for a whole range of prominent agent platform implementations [4, 7, 19]. Figure 1 shows an UML interaction diagram for this protocol. In order to comply with the FIPA standards, we call the agent with the task *initiator*, agents that compete for acquiring the task *participants*. In general, the procedure requires the initiator to send a "call for proposals" including a task description to all participants. They can specify their required costs for this task in a proposal (or refuse to do the task at all). The initiator then accepts one of these proposals, and rejects all others. The agent who got his bid accepted is then required to inform the initiator about the result of the task (or its failure).

This protocol was designed for distributing one task among a number of agents. However, if we assume a large number of initiators and bounded resources for each of the participants (as is common in today's multi-agent systems), new problems arise. Although the execution of this protocol is very efficient, it is a hard problem for each agent to decide when to allocate the resources for which task. Imagine that among the agents in a large-size multi-agent system there are n agents with tasks (initiators) and m providers of services (participants). While a participant is in negotiation with a large number of initiators, it may still receive more call for proposals without having received any reject messages as the initiators are still busy evaluating the proposals.

Up to now it remains an unanswered question which policy the agent should use for resource allocation, i.e. in what manner it should reserve resources for tasks it made a bid for. If the agent allocates too many resources too early, it may not get its bid accepted and therefore resources will not be available for other tasks. If it allocates too late, it may have committed to more tasks than it has resources. Several approaches have been proposed: leveled commitments ([12]; for an extension see [2]), and statistical methods (as they are being used e.g. in flight booking systems) [13]. The latter depend on data gathered over a long period of time and involve the risk of overbooking (as is the common experience with frequent flyers) while the former requires more complex communication, resulting in higher computational costs for both participant and initiator.

The paper is structured as follows: section two introduces the concept of cascading applications of the CNP and gives a concrete example. The third section describes the problem of using the CNP in multiple concurrent task assignments and offers a solution. Section four modifies this solution to work in cascading applications. Finally, section five summarizes the results of this paper.

2 Example of a Cascading Application: Holons

Cascading applications of task assignments occur in systems where the receiver of a call for proposals starts one or more new CNPs to delegate the task of parts or the task to other agents. One example of such a system is a system of *holonic agents*. A holonic agent consists of several agents, but interacts with agents outside the holon as a single agent. This concept is inspired by the idea of recursive or self-similar structures in biological systems [9] and has found applications in the domain of holonic manufacturing [11, 17]. In this paper, we will use holonic agents to illustrate the application of cascading task assignments.

3 Multiple Concurrent Task Assignments

3.1 Shortcoming of the Contract Net Protocol

Let us consider the case where the agent allocates resources at the time of sending the bid. We call this solution the ad hoc solution, or the conservative approach. This solution makes sure that only correct assignments of tasks to agents are created, i.e. that every agent only commits to the tasks it can perform. However, if several participants send their proposal to the same

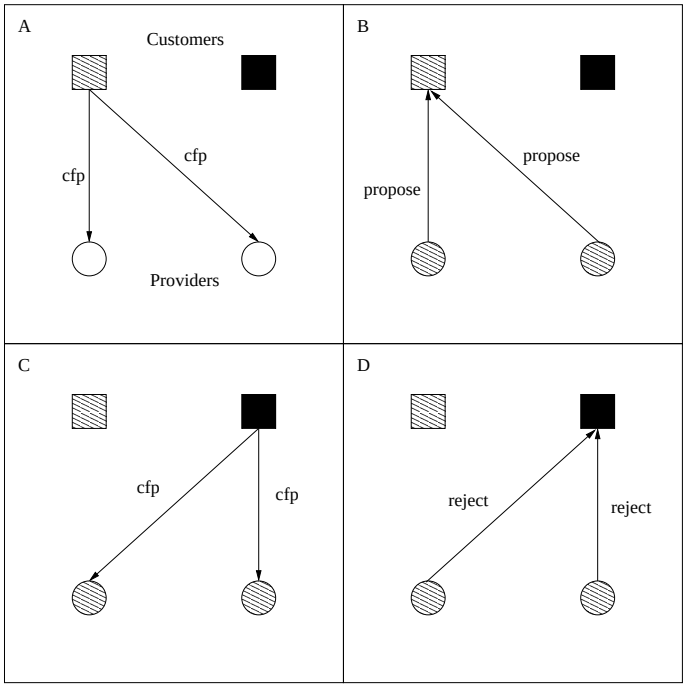

Figure 2: Example of CNP sub-optimality.

initiators, which is not unlikely, the result is that only some of them get a task assigned, while others remain idle. Therefore, this procedure is not complete in that it will not compute solutions that could be found with better approaches.

Figure 2 gives an example of a simple situation with two customers and two providers where the CNP can lead to a suboptimal outcome. In the phase A, the first customer sends a call for proposals (cfp) to each of the provider agents. The provider agents reply with their proposals in phase B, allocating the resources required for the task. If, as shown in phase C, the second customer sends its cfps before the first could finish its auctions, both providers will still have their resources allocated for the first task, and therefore have to send reject messages to the second customer in phase D. Even though the two providers could in principle handle the two tasks, the system did not find this optimal solution.

The likelihood of failing to find possible solution is even higher in scenarios with more agents. Consider using the conservative approach in a setting with 100 initiators, each having one task to assign and 100 participants, each capable of performing one task. Further consider that the deadlines are set in a way that the participants cannot reply to the calls sequentially

(otherwise the multi-agent approach would hardly apply). If in this case every participant uses a conservative approach to the problem and just sends one bid, the chance of getting a bid accepted assuming lottery on the side of the initiator is ca. 0.64 (the computation of this probability is out of scope here, but from the problem chosen, it is in any case clear that the probability is below 1). If other agents make more than one bid, the probability is even lower. So in more than one third of all cases, the available resources of the participant will be idle due to the conservative strategy. Correspondingly, the same number of initiators will be left with unassigned tasks, as they did not get any bids for their tasks, although the resources are in the system.

3.2 Solution

Our approach is based on redesigning the protocol to postpone the time of commitment as far as possible. The major inefficiency in the CNP is that in every execution of the protocol all participating agents need to commit themselves to do the job, although only one of them will actually get the task awarded. We now present the contract net with confirmation protocol (CNCP), which precisely addresses this issue and improves the CNP procedure by drastically reducing the number of commitments made.

The CNCP (figure 3) is very similar to the CNP. It starts with a call for proposals and gathers the responses from the participants, until the initiator received messages from all participants or the deadline has passed. As in the contract net protocol, this deadline safeguards that singular message dropouts do not prevent the whole protocol from terminating. In the original contract net, the participant makes its commitment in the bidding stage. In the CNCP this is not the case: the commitment is only made when the initiator requests that the participant should take over the task. For this purpose the initiator arranges all bids in a sorted list and sends requests to participants starting with the best bid to find out if they can actually do the job. The next participant is sent a request message if the previous participant has sent a refuse or a deadline has passed. This iteration stops if one participant sends an agree message. All other agents are sent a reject-proposal message (except those who have already received the request and sent the refuse). The participant only needs to commit at the time of sending the agree message. In order to trigger task execution and to correspond to the CNP it is required that the agent sends an accept-proposal while the participant will reply (as it does in the CNP) with failure, inform-done, or inform-ref.

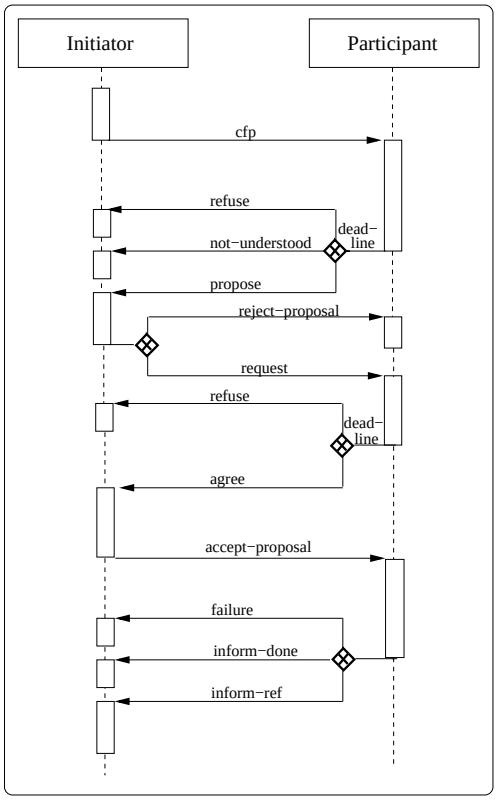

Figure 3: The Contract Net with Confirmation Protocol.

3.3 Discussion

As well as the original contract-net protocol, the proposed procedure needs O(n) messages, where n denotes the number of participants. In the best case, the CNCP requires only two more messages (the request for confirmation and the reply to it) while still solving the resource allocation problem of the initiator. In the worst case, the initiator needs to contact all participants to find out that no one can do the task. Although this results in a plus of $2n$ messages for the CNCP, its great advantage is that it only requires one agent to make a single commitment. This is achieved by using the confirmation stage in the protocol, to postpone the commitment and allow the participants to reply to all incoming call for proposals without need to already allocate the resources at this early stage of interaction or to risk penalties for multiply allocating resources. A minor disadvantage of this approach is that the initiator possibly needs some overhead to repeatedly find the next best bid, while the CNP only requires it once to find the maximum.

However, with careful implementation this additional computational effort is by several orders of magnitude lower than the effort spent for sending the messages, and is in the general use of MAS a negligible additional cost.

In order to guarantee termination even in the case of faulty participants the second deadline of the protocol is necessary. It makes sure that the next best participant can be sent a request message and has a chance to receive the task.

4 Cascading Applications of the CNCP

In Holonic multi-agent systems (HMAS) each agent can be a holon consisting of several other agents (subholons). Usually this holon consists of one agent in charge of communicating to the outside, called head and a number of other agents responsible for some kind of problem solving behavior, called body agents (for a more detailed discussion see Gerber, Vierke, and Siekmann, 1999). To other agents the holon looks and acts like a single agent.

4.1 Shortcoming of the CNCP

Both the CNP and the CNCP work in conventional as well as in holonic multi-agent systems (HMAS). HMAS require due to their recursive structure the recursion of negotiation protocols, and both protocols can be used in cascades, i.e. each participant which is a holon head can initiate another instance of the same protocol to subcontract the task to other agents (generally agents in the same holon). In the case of the CNP this leads to rapidly increasing allocations of resources as all participants must allocate their resources (see the discussion above). The CNCP avoids this inefficiency. However, in some cases a new inefficiency arises when applying a cascade of CNCPs, namely when some of the agents in the lower part of the cascade refuse to do the job.

Figure 4 shows a scenario of two customers and a holon consisting of two body agents and a head agent who is in charge of communicating to the outside. The cost of the first body agent for completing the task of either customer is 5, that of the second 6, so the second agent is less efficient than the first. Phase A shows what happens after both customers have sent their cfps to the holon head. The head reacts by starting another CNCP among the body agents its holon is composed of. It decides on the basis of its body agents' best bid how to reply to the initial cfp. As all resources are still free, the head chooses the cheapest agent in both cases and sends a proposal of 5 to both customers. After receiving the first request in phase B, the head forwards this request to its cheapest body agent, who in turn allocates the

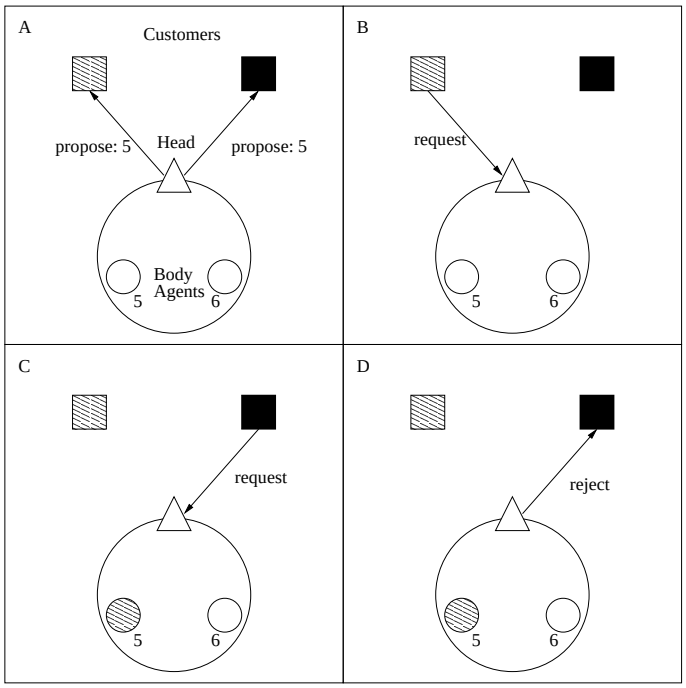

Figure 4: Example of CNCP sub-optimality in cascading applications.

resources for the task, which are now no longer available. The request from the second customer in phase C has therefore to be rejected (phase D), as the holon can no longer complete the task for the proposed price of 5. The CNCP fails to find the solution of assigning the first task to the agent with cost 5 and the second to the agent with cost 6, which would be the optimal outcome in this scenario.

4.2 Solution

To avoid this problem in scenarios with holonic agents, we use a version of the CNCP that includes the possibility of a second proposal after the request has arrived. If the head finds that its cheapest body agent can no longer do the job, it can send a request to the second best bidder. If the second best bidder agrees, the holon head can send a second proposal to the initiator (which is possibly higher than the first), who can compare it to the bids it received from the other participants. Since CNCP does not allow such a second proposal, the holon head would have to refuse the job even if its second best bidder could do it for a better price than anyone outside the holon. The modification to the CNCP for systems with holonic agents therefore consists

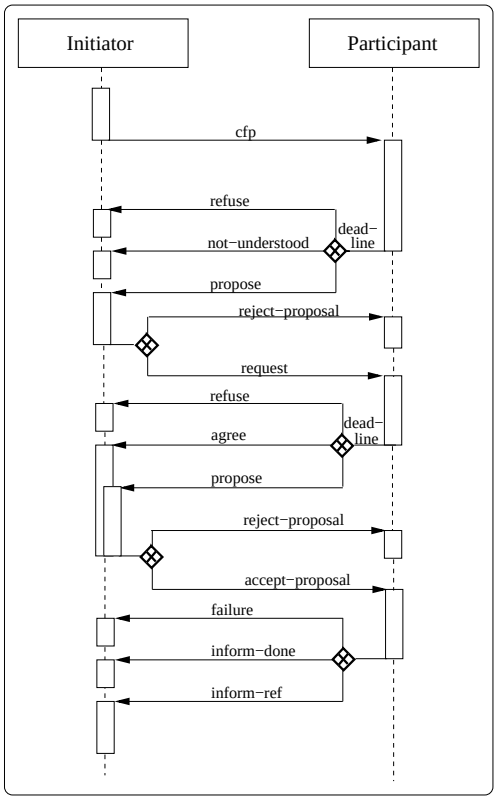

Figure 5: The Holonic Contract Net with Confirmation Protocol.

of adding a second proposal as possible reply to a request. Figure 5 shows the resulting Holonic Contract Net with Confirmation Protocol (HCNCP).

The participant allocates the resources for the task when either agreeing to the request or making a second proposal. The second proposal requires a commitment to ensure the termination of the protocol. A noteworthy difference from the CNCP is that the initiator can send a reject even after the participant has committed. To see why this modification is necessary, recall the scenario mentioned above, where the holon makes a second proposal. It does so only after its second best bidder has committed. However, it is possible that this second proposal is rejected because it is no longer the best bid. In this case, the holon has to forward the reject to its committed subunit. In summary, the HCNCP is a recursively applicable protocol that reduces the number of unnecessary commitments by introducing a confirmation stage and that increases the flexibility of holons by allowing a second proposal to reach better solutions than the cascading CNCP.

4.3 Discussion

Assume that all agents in a cascading HCNCP are nodes in a tree where the problem solving body agents are represented by the leaf nodes. By using the second proposal the worst case occurs if in any holon with leaf node agents, the agent with smallest bid refuses to do the task and a leaf node agent in this holon commits. In this case the number of commitments increases to the number of parents of leaf nodes, but the protocol still reaches the same (optimal) solution as the CNCP for the non-holonic case would.

5 Conclusion

We presented a task allocation mechanism for multi-agent systems that is based on the widely used contract-net protocol. As well as the original contract-net protocol (CNP), the CNCP procedure needs $O(n)$ messages, where n denotes the number of participating agents. In the best case, the CNCP requires only two more messages (the request for confirmation and the reply to it) while still solving the resource allocation problem of the initiator. In the worst case, the initiator needs to contact all participants to find out that no one can do the task. In the average case however this means that the communication requires only $O(n)$ message while allowing highly parallel task allocation with only one commitment by one agent.

In systems that use cascading applications of protocols for delegating task assignments, our modification to the CNP takes the form of the holonic contract net with confirmation protocol (HCNCP). The best and worst case analysis is the same as with the CNCP, but it applies to each level of the cascade. The advantage of the HCNCP is that it can find solutions to the task assignment problem in agent systems that have a delegation structure which makes the CNCP or CNP fail.

References

[1] H. Bürkert, K. Fischer, and G. Vierke. Holonic transport scheduling with TELETRUCK. *Applied Artificial Intelligence*, 14(7):697–726, 2000.

[2] C. Excelente-Toledo, C. Bourne, and N. Jennings. Reasoning about commitments and penalties for coordination between autonomous agents. In *Proceedings of the Fifth International Conference on Autonomus Agents*, pages 131–138. ACM, 2001.

[3] FIPA. (Foundation for Intelligent Agents), http://www.fipa.org/repository/ips.html.

[4] FIPA-OS. http://fipa-os.sourceforge.net/.

[5] K. Fischer, J. P. Müller, and M. Pischel. A model for cooperative transportation scheduling. In *Proceedings of the 1st International Conference on Multiagent Systems (ICMAS'95)*, pages 109–116, San Francisco, June 1995.

[6] L. Garrido and K. Sycara. Multiagent meeting scheduling: Preliminary experimental results. In *Proceedings of the 2nd International Conference on Multiagent Systems (ICMAS'96)*, 1996.

[7] JADE. http://www.sharon.cselt.it/projects/jade/.

[8] N. R. Jennings. Commitments and conventions: The foundation of coordination in multi-agent systems. *The Knowledge Engineering Review*, 8(3):223–250, 1993.

[9] A. Koestler The Ghost in the Machine Hutchinson & Co, London, 1967

[10] V. Parunak. Manufacturing experience with the contract net. In M. Huhns, editor, *Distributed Artificial Intelligence*, pages 285–310. Pitman, London, 1995.

[11] R. Rabelo, L. Camarinha-Matos, and H. Afsarmanesh. Multi-agent perspectives to agile scheduling. In R. Rabelo, L. Camarinha-Matos, and H. Afsarmanesh, editors, *Intelligent Systems for Manufacturing*, pages 51–66. Kluwer Academic Publishers, 1998.

[12] T. Sandholm and V. Lesser. Advantages of a leveled commitment contracting protocol. In *Proceedings of the 13th National Conference on Artificial Intelligence (AAAI-96)*, pages 126–133, 1996.

[13] Schillo, M. and Kray, C. and Fischer, K. The Eager Bidder Problem:A Fundamental Problem of DAI and Selected Solutions In *Proceedings of the First International Conference on Autonomous Agents and Multiagent Systems*, to appear

[14] S. Sen and E. Durfee. On the design of an adaptive meeting scheduler. In *Proceedings of the 10th IEEE Conference on AI Applications*, 1994.

[15] W. Shen and D. Norrie. An agent-based approach for dynamic manufacturing scheduling. In *Proceedings of the 3rd International Conference on the Practical Applications of Agents and Multiagent Systems*, 1998.

[16] R. G. Smith. The contract net protocol: High level communication and control in a distributed problem solver. *IEEE Transactions on Computers*, Series C-29(12):1104–1113, 1980.

[17] Ulieru, M. and Walker, S. and Brennan, B. Holonic Enterprise as a Collaborative Information Ecosystem *Proceedings of the Workshop on Holons: Autonomous and Cooperating Agents for Industry, Autonomous Agents*, 2001.

[18] G. Weiß, editor. *Multiagent Systems. A Modern Approach to Distributed Artificial Intelligence.* The MIT Press, Cambridge, MA, 1999.

[19] ZEUS. http://www.btexact.com/projects/agents/.