Deutsches
Forschungszentrum
für Künstliche
Intelligenz GmbH

Technical
Memo
TM-01-01

The Contract-Net with Confirmation Protocol: An Improved Mechanism for Task Assignment

Michael Schillo, Klaus Fischer, and Tore Knabe

November 2001

Deutsches Forschungszentrum für Künstliche Intelligenz GmbH

Postfach 2080
67608 Kaiserslautern, FRG
Tel: +49 (631) 205-3211
Fax: +49 (631) 205-3210
E-Mail: info@dfki.uni-kl.de

Stuhlsatzenhausweg 3
66123 Saarbrücken, FRG
Tel: +49 (631) 302-5252
Fax: +49 (631) 302-5341
E-Mail: info@dfki.de

WWW: http://www.dfki.de

The Contract-Net with Confirmation Protocol: An Improved Mechanism for Task Assignment

Michael Schillo, Klaus Fischer, and Tore Knabe

DFKI-TM-01-01

This work has been supported by the Deutsche Forschungsgemeinschaft under contract Fi 420/1-1

ISSN 0946-008X

The Contract-Net with Confirmation Protocol: An Improved Mechanism for Task Assignment

Michael Schillo Klaus Fischer Tore Knabe

1 Problem Description and Related Work

Assume we have an agent that has a task it needs to be done but does not have the ability or the resources to do it. The contract-net protocol (Smith 1980, Smith and Davis, 1983) was designed to describe the communication necessary to determine some other agent that can do the task. 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* (FIPA, 2001). 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).

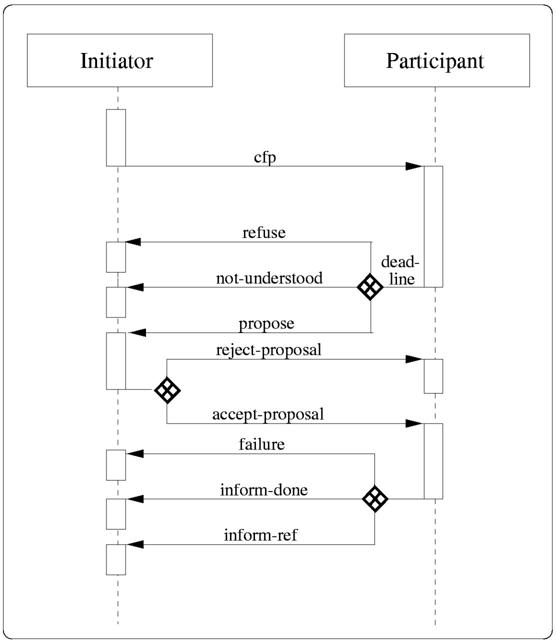

Figure 1 FIPA Contract Net Interaction Protocol.

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 evaluation 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: levelled commitments (Sandholm and Lesser, 1980; for an extension see Excelente-Toledo, Bourne and Jennings, 2001), and statistical methods (as they are being used e.g. in flight booking systems). 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.

2 Importance of the problem

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 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.

For illustration, 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 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 almost 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.

3 Solution

The approach of this protocol 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 award. 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.

3.1 Procedure

The CNCP (cf. Figure 2) 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.*

3.2 Discussion and Analysis

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 neglectible 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.

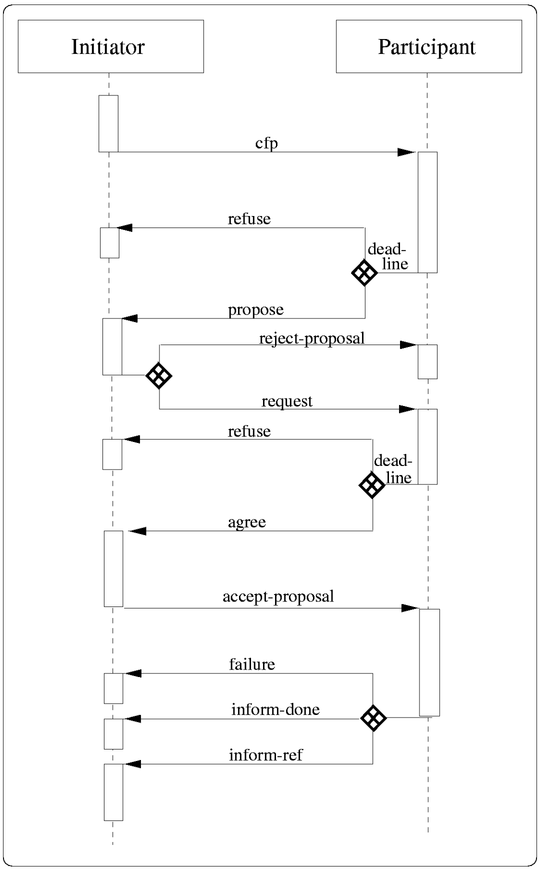

Figure 2. The Contract Net with Confirmation Protocol.

4 The holonic extension

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 behaviour, 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.

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. In the following we present a solution to this inefficiency.

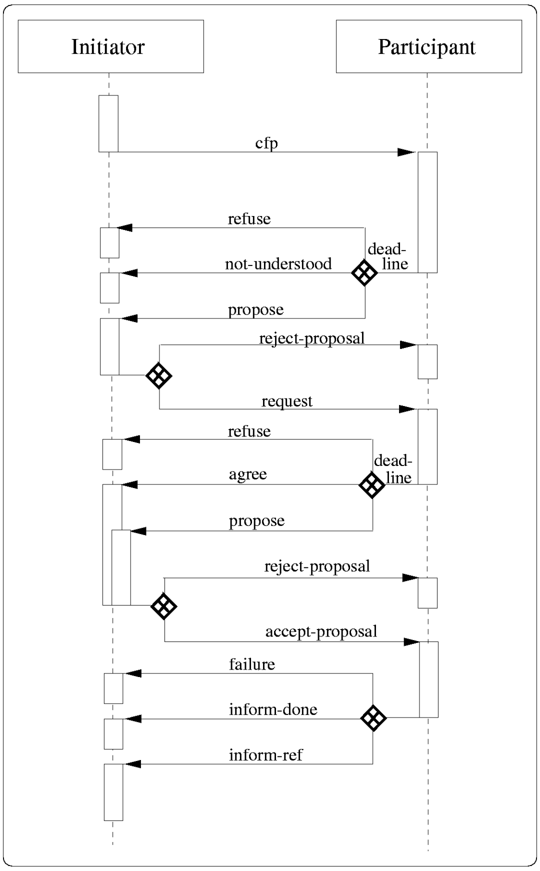

Figure 3 The Holonic Contract Net with Confirmation Protocol.

Assume that the holon head which receives a *call for proposals* (cfp) 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. If the initiator of the original cfp replies with a request for confirmation, this request is forwarded to the body agent with the best bid. In the case that this body agent can no longer allocate the required resources, we would like the holon to be able to check whether the body agent with the second best bid can do the job by sending it a request for confirmation. 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 allowe such a second proposal, the holon head has 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 of adding a second proposal as possible reply to a request. Figure 3 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.

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, 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.

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.

6 References

Excelente-Toledo, C., Bourne, R., and Jennings, N. (2001). Reasoning about Commitments and Penalties for Coordination between Autonomous Agents. *Proceedings of Autonomous Agents'01,* ACM Press, pp.131-138.

FIPA (2001) http://www.fipa.org/repository/ips.html

Gerber, C., Siekmann, J. und Vierke, G. (1999). Flexible autonomy in holonic multi-agent systems. In AAAI Spring Symposium on Agents with Adjustable Autonomy.

Sandholm, T., and Lesser, V. (1995). Issues in Automated Negotiation and Electronic Commerce: Extending the Contract Net Framework. In: *Proceedings of the International Conference on Multi-Agent Systems (ICMAS),* pp. 328-335.

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

Smith, R., and Davis, R. (1983). Negotiation as a metaphor for distributed problem solving. In *Artificial Intelligence*, vol. 20, pp. 63-109.

The Contract-Net with Confirmation Protocol: An Improved Mechanism for Task Assignment

Michael Schillo, Klaus Fischer, and Tore Knabe

TM-01-01

Technical Memo