Cooperative, MultiStep Negotiation Over a Multi-Dimensional Utility Function

XiaoQin Zhang, Rodion Podorozhny, Victor Lesser
Department of Computer Science
University of Massachusetts at Amherst
xqzhang@cs.umass.edu
phone: 413-545-0675

Abstract

We present a cooperative, multistep negotiation mechanism for multiagent systems. This mechanism uses marginal utility gain and marginal utility cost to structure the negotiation process. This enables an agent to understand another agent's situation in order to find a solution that increases their combined utility. These two values summarize the agent's local information and reduce the communication load. We also introduce a multiple attribute utility theory into negotiations. This allows agents to negotiate over multiple attributes of the commitment which makes it more likely for agents to find a solution that increases the global utility by producing more options.

1 Introduction

Negotiation is a process by which two or more parties make a joint decision. The parties first verbalize demands and then move toward an agreement by a process of concession or search for new alternatives[5]. In multi-agent systems(MAS), negotiation is used for task and resource allocation, recognition of conflicts, resolution of goal disparities, determination of the organizational structure, and hence the coherence of the agent society.

The negotiation research in multi-agent systems falls into two main categories, competitive negotiation and cooperative negotiation. Competitive negotiation occurs among self-interested agents [9], each agent trying to maximize its local utility; while in the cooperative negotiation, agents try to reach the maximum global utility that takes into account the worth of all their activities. This form of negotiation is quite different from competitive negotiation and can be viewed as a distributed search process. We will focus on this cooperative negotiation which, as of late, has not received very much attention in the related literature [4]. In fact, we feel there seems to be very little work on cooperative negotiation that explicitly tries to maximize the multi-dimensional global utility function. The closest work to our knowledge is that of Moehlman et al. [6]; however their work involves a much simpler and more structured utility function and does not evaluate their approach in different contexts.

There are different degrees of cooperation in a multi-agent system. The most extreme one is "global cooperation", which occurs when an agent, while making its local decision, always tries to maximize the global utility function that takes into account the activities of all agents in the system. Global cooperation is unachievable in most realistic situations because of the number of agents and bounds on computational power and bandwidth. Thus we focus our research on the "local cooperation" [3] which occurs when two or more agents, while negotiating on one issue, try to find a solution that increases the sum of their local utilities, without taking into account the rest of the agents in the system.

We introduce a multiple attribute utility theory from economic research [7] into the negotiation process. As the theory's name suggests agents are assumed to negotiate over multiple issues rather than over a single dimension. For example, agent A wants agent B to do task T for it by time 10, and requests the minimum quality of 8 for the task to be achieved. Agent B replies that it can do task T by time 10 but only with the quality of 6, however, if agent A can wait until time 15, it can get the quality of 12. Then agent A will consider which choice is better for both. The negotiation is about both the issue of the completion time and the achieved quality of the task, and thus the scope of the search space for the negotiation is increased, improving agents' chance of finding a solution that increases the combined utility.

Our approach puts an emphasis on a multi-step negotiation in which agents engage in a series of proposals and counter offers. This is a search for those schedules of agent's local activities that increase or maximize the combined utility of the agent. We will use measures of marginal gain and marginal cost first used in the TRACONET agents [8] to structure the search. In that work, these measures were used for a single phase evaluation rather than as a basis for cooperative/distributed search among agents to find the best combined local schedules.

The cooperative negotiation process can potentially have many outcomes depending upon the amount of effort that the agents want to expend on the negotiation. One possibility is that they will find a solution that leads to the maximum global utility; another possibility is that they will find a solution that increases the global utility from their current state; a third possibility is that they may find that either there is no solution that increases the global utility or they can not find one by the limited search.

After the negotiation starts, the agent needs to decide when to stop the negotiation because the negotiation cost increases with time. It may stop after it gets the first acceptable solution that increases the utility or it may decide to continue looking for a better one. The agent needs to establish a balance between the negotiation cost and the negotiation benefit.Thus, there are many different variations of a cooperative negotiation protocol depending on the alternatives chosen above. Therefore as part of this paper we will examine experimentally these questions to get insight about how the characteristics of the current situation affect the variant of the protocol to be chosen.

In the remainder of the paper, we present our work on cooperative negotiation in the task allocation domain. First, we describe the negotiation framework, followed by the negotiation mechanism. Then we discuss the experiment results obtained by these protocols and the observations from these results. Finally, we summarize our work and discuss the future work.

2 Framework -TÆMS & DTC

The TÆMS framework [1] is used to represent the agent's local tasks and activities. The TÆMS task modeling language is a domain-independent framework used to model the agent's candidate activities. It is a hierarchical task representation language that features the ability to express alternative ways of performing tasks, statistical characterization of methods via discrete probability distributions in three dimensions (quality, cost and duration), and the explicit representation of interactions between tasks.

The cooperative negotiation mechanism makes the following assumption about the agent architecture: the presence of a local scheduling mechanism that can decide what method execution actions should take place and when. The local scheduler attempts to maximize the multi-dimensional utility function. The DTC (Design-To-Criteria) [10] scheduler is used as the agent's local scheduler in our research. It is a domain-independent scheduler that aims to find a feasible schedule that matches the agent's local criteria request. The first input for the DTC scheduler is the TÆMS task structure that describes the agent's local activities and the objective criteria used to evaluate alternative schedules. The second input is a set of existing and proposed commitments, C, that indicates that this agent will produce specific results of certain qualities by certain times. The third input is a set of non-local commitments, NLC, that are commitments made to this agent by other agents. The scheduler uses this information to find the best schedule given the objective criteria, that exploits the given non-local commitments, honors the existing commitments and satisfies the proposal commitment as best as possible.

3 Task Allocation Negotiation Mechanism

In a multiagent system, an agent may need to assign one of its local tasks to another agent because it can not perform this task locally. This task can potentially be part of a larger activity that the agent performs in order to achieve some desired goal. The agent needs to negotiate with another agent about the appropriate time and approach to execute this task, so that the combined utility (the sum of both agent's local utilities) can be increased. By an approach we mean an alternative way for another agent to perform the task differing from other ways in the resources used and the quality of the solution obtained.

An agent will contract out a task to another agent if it does not have the capabilities to perform this task locally or if it is overloaded. We assume that the agent will use the TÆMS task representation of its activities to communicate with the negotiation system about which task it definitely can not do locally and those tasks that it thinks may be advantageous to be done by another agent. As part of the negotiation process, the relative merits of the option of doing the task locally or not doing it at all versus the option of contracting will be taken into account.

3.1 Definitions

- Contractor Agent (contractor): the agent which has a task (non-local task NL) that needs to be assigned to another agent, the contractor gains quality from this task when it is completed (TCR is the contractor's local task structure).
- Contractee Agent (contractee): the agent which performs this task for the contractor, it devotes some processing time and certain cost to this task without gaining quality from this task(TCE is the contractee's local task structure).
- Marginal Quality[NL,C](MQ) - The overall quality increment for the contractor by having task NL performed with completion time and quality specified as in commitment C. It is not simply the quality of NL, since NL may affect other tasks in TCR. For example, the quality of NL is 10 (see figure 2, where NL is task M4), and there is an "enables" relationship between NL and task M5, whose quality is also 10. If task NL is not to be performed, task M5 can not be performed either, so the marginal quality of NL is 20 in this case. The marginal quality is used by the contractee to evaluate how important NL is for the contractor.
- Marginal Utility Gain[NL, C](MUG) - The local utility increment for the contractor by having task NL performed with completion time and quality specified as in commitment C.
- Marginal Utility Cost [NL, C](MUC) - The local utility decrement for the contractee by performing task NL with completion time and quality specified as in commitment C.

3.2 Mechanism

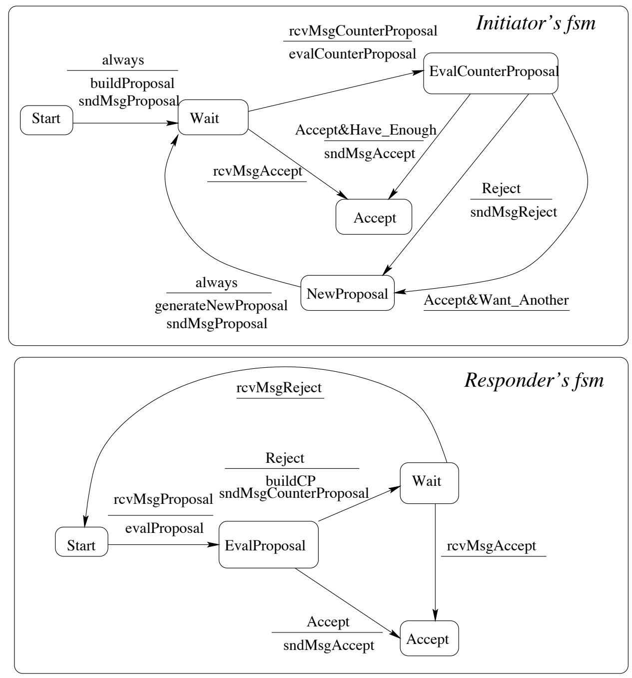

Figure 1: Cooperative Task Allocation Protocol

Figure 1 shows the agents' protocol that implements the task allocation mechanism. The mechanism is based on the assumption that the combined utility will have a maximum in the time range defined by the earliest and latest reasonable times a non-local task NL can be finished. This time range serves as the range for a binary search on the values of NL's finishing times for a maximum combined utility. Essentially, each next proposal from the contractor is the midpoint between its own current proposal and the contractee's counter proposal, while each next counter proposal from the contractee is the midpoint between the contractor's proposal and the contractee's own current counter proposal. The mechanism also allows for the possibility of varying NL's quality throughout the range specified by alternative ways of accomplishment in the contractee.

Let us describe the mechanism in greater detail. This protocol is structured as three functions. One is to generate the initial proposal, the second is to generate a counter proposal by the contractee, and the third is for the contractor to generate a new proposal in response to the counter proposal. When the contractor obtains its task structure and finds there is a non-local task NL needed to be assigned to another agent, it builds a proposal commitment PC based on its local schedule. This commitment specifies the finishing time and the quality request for NL's execution. The contractor knows only a range of values that task NL's quality can take. Currently, the quality estimate sent by the contractor to the contractee in proposal commitment PC is the midpoint of the range.[1] In addition to this information, the marginal quality of NL and the marginal utility gain of this commitment are also provided by the contractor. This commitment and associated information are sent to the contractee. The contractee schedules this commitment in the context of the existing set of potential activities and other commitments as specified in its local task structure. If this commitment can be satisfied with the marginal utility cost below the marginal utility gain, the contractee accepts this commitment; otherwise, the contractee tries to refine this commitment (CounterProposalGeneration function, see section 3.3), and sends counter proposal CC to the contractor. When the contractor receives this counter proposal CC, it evaluates CC by adding it to its local task structure and seeing what local schedule is possible based on this counter proposal. If there is a local schedule whose marginal utility gain exceeds the marginal utility cost of the counter proposal, this counter proposal is accepted. Otherwise, it is rejected. If the counter proposal is rejected or the contractor wants to find a better commitment, the contractor tries to improve the commitment (NewProposalGeneration function, see section 3.3). The improvement is a two dimensional search process based on the values of the finishing time and quality suggested in the previous commitment and the counter proposal from the contractee. The new commitment is sent to the contractee and another negotiation cycle starts. As the negotiation progresses, the contractor keeps track of the number of accepted commitments and stores the accepted commitment with the highest global utility. The negotiation process ends either when the number of negotiation cycles exceeds a predefined limit or the contractor has registered that the desirable number of improvements over the original accepted commitment has been made or the binary search algorithm terminates. If the contractor has registered an accepted commitment by the time any of these events occurred, the contractor notifies the contractee of the commitment that has been finally agreed upon.

3.3 Elaboration of protocol functions

CounterProposalGeneration

This function is used by the contractee to generate a counter proposal in response to an unacceptable proposal. The function works as follows. If there is no previous counter proposal, the contractee builds the first counter proposal by removing both the deadline and the quality request, setting the marginal quality as half of its original value and finding the schedule that performs task NL with the minimum marginal utility cost. If the previous counter proposal exists, the contractee compares the contractor's current proposal with the previous counter proposal. If PC requests a finishing time for NL sooner than CC provided, the contractee sets the finishing time request as the average of the finishing times specified in PC and CC, otherwise, it keeps PC's finishing time request. Then the contractee removes the quality request for task NL and finds the schedule with the minimum marginal utility cost that contains task NL. The new counter proposal is built based on this schedule (i.e. the finishing time for NL and NL's quality corresponding to that finishing time are extracted from the schedule and put into a newly created commitment).

[1] The issue of how to assign this initial value is interesting in itself. A lower quality request allows the contractee to have more choices to perform the task and thus find an acceptable but not maximal solution quality while a higher quality request may bring a higher marginal utility gain for the contractor but it may take much longer to find an acceptable solution.

NewProposalGeneration

This function is used by the contractor to build a new commitment based on the contractor's previous proposal and the contractee's current counter proposal. It does a two-dimensional search in the time-quality space. If there is already an acceptable solution, it tries to find a new solution either with a higher MUQ or lower MUC. It there is no acceptable solution, it tries to find a solution by relaxing previous request constraints (in quality and/or in time). A commitment (be it a proposal or a counter proposal) carries two principal values that are negotiated over: the finishing time of the non-local task (NL) and the quality achieved or desired if NL is executed with the said finishing time. Therefore there are four possible outcomes of comparison of these two values between two commitments. The result of the NewProposalGeneration function is essentially a new pair of the finishing time and quality. The NewProposalGeneration function assigns values to this pair differently depending on the outcome of comparison of finishing time and quality carried in the previous contractor's proposal and the current contractee's counter proposal. The function also behaves differently depending on whether it is improving over an already existing acceptable commitment or formulating a new proposal in response to a rejection.

Let us see what actions the contractor takes if there is an existing acceptable commitment (the quality value is not compared):

- The contractee informs the contractor that it cannot do task NL as early as the contractor requested: the contractor now asks for a finishing time to be the average of those of the counter proposal and previous proposal and it decreases the requested quality at a certain rate (by multiplying it by a value between 0 and 1) thus trying to meet the contractee halfway and with a reduced quality.
- The contractee informs the contractor that it can do task NL as early as the contractor requested: the contractor asks for a finishing time to be the average of those of the counter proposal and previous proposal and requests the quality as the contractee offered thus trying to see if this new pair reduces the contractee's cost.

Let us see what actions the contractor takes if there is not an existing acceptable commitment yet:

- The contractee informs the contractor that it cannot do task NL as early as the contractor requested, but it can do it later with a higher quality: the contractor now asks for a finishing time to be the average of those of the counter proposal and previous proposal and it keeps the requested quality the same thus trying to meet the contractee halfway.
- The contractee informs the contractor that it cannot do task NL as early as the contractor requested and the quality requested is not possible: the contractor asks for a finishing time to be the sum of that of the previous proposal and the duration estimate of task NL and it keeps the requested quality the same thus trying to do the task later.
- The contractee informs the contractor that it can do task NL at the requested time or earlier and even with a higher quality than requested: the contractor asks for a finishing time to be the average of those of the counter proposal and previous proposal and request the quality as the contractee offered thus trying to see if this new pair reduces the contractee's cost.

- The contractee informs the contractor that it can do task NL at the requested time or earlier but the quality requested is not possible: the contractor asks for a finishing time to be the sum of that of the previous proposal and the duration estimate of task NL and it keeps the requested quality the same thus trying to do the task later.

3.4 Example

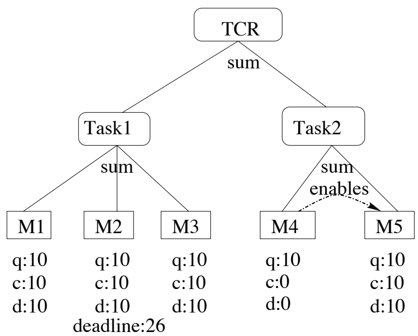

Figure 2: the contractor's task Structure

For example, the contractor is working on task TCR (Figure 2). Subtask M4 is a task that needs to be assigned to another agent (suppose the problem solver makes this decision). The contractee is an agent that could potentially perform task M4. (There could be more than one potential agent, to make the example clearer, we only show one).

The contractor schedules local task structure TCR assuming M4 is not to be done and gets the following schedule:
$S1 : M2(0-10)M1(10-20)M3(20-30)$
$Quality(S1) = 30; Utility(S1) = 0.56$
then it schedules TCR assuming that another agent could do M4 for it and gets schedule:
$S2 : M2(0-10)M1(10-30)M3(20-30)M5(30-40)$
(with M4's result available at time 30)
$Quality(S1) = 50; Utility(S1) = 0.84$
Then it builds commitment PC based on S2:
$PC : [M4, completion\_time30, quality\_request10]$
$MQ(M4) = 50-30 = 20, MUG(M4) = 0.84-0.56 = 0.28$

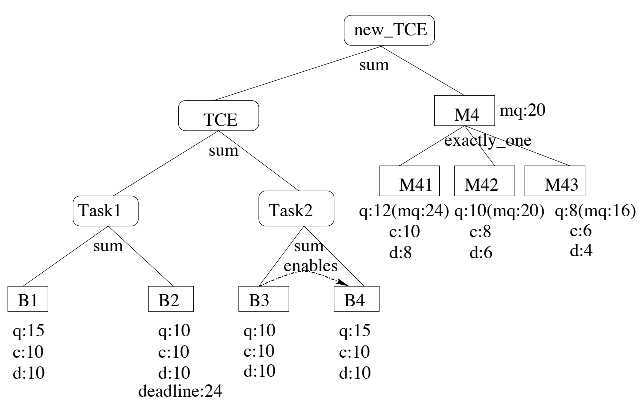

Figure 3: the contractee's task structure

The contractee receives this commitment, puts M4 into its local task structure TCE and gets a new task structure new_TCE (Figure 3). The contractee instantiates M4 and finds three different plans to perform M4: M41, M42 and M43[2]. Each plan has different quality, cost and duration characteristics. These three choices are represented as three subtasks of M4 with "exactly_ one" qaf in TÆMS structure, and their quality is mapped into the marginal quality according to the following formula:
$MQ(M4i) = \frac{MQ(M4)*quality(M4i)}{quality\_request(M4)}; i = 1, 2, 3.$

The contractee schedules new_TCE with PC, finds the schedule:
$S3 : B2(0-10)B1(10-20)M41(20-28)$
$Utility(S3) = 0.478$
Compares to schedule S4 of TCE without PC:
$S4 : B2(0-10)B1(10-20)B3(20-30)B4(30-40)$
$Utility(S4) = 0.76$
$MUC(PC) = 0.76 - 0.478 = 0.282 > MUG(PC) = 0.28$
so PC is rejected. In a single step (see section 4) protocol, the negotiation process ends here with a failure. The contractee then tries to refine this commitment, and finds the schedule S5:
$B2(0-10)B1(10-20)B3(20-30)B4(30-40)M42[40-46]$
$Utility(S5) = 0.74$
Then it constructs counter proposal CC:
$CC[M4, completion\_time46, quality\_achieved10]$
$MUC(CC) = 0.04$. The contractor receives CC and puts it into TCR, gets the schedule:
$S6 : M2(0-10)M1(10-20)M3(20-30)M5(46-56)$
with M4's result available at 46 and achieved quality is 10.
$Utility(S6) = 0.808$
$MUG(CC) = 0.808 - 0.56 = 0.248 > MUC(CC) = 0.04$
so this is an acceptable commitment. In a Multistep-One-Try (see section 4) protocol, the contractor stops here and accepts CC with the marginal utility gain of 0.248. In a Multistep-Multiple-Try (see section 4) protocol, the contractor continues negotiation and tries to find a better commitment.

If the contractor decides to find another solution, it improves the commitment based on its previous commitment and the counter proposal from agent R. It rebuilds commitment: $PC1[M4, completion\_time38, quality\_request10]$ and finds the schedule S6 with this commitment.
$S6 : M2(0-10)M1(10-20)M3(20-30)M5(38-48)$ (with M4's result available at 38 and achieved quality is 10.)
$Utility(S6) = 0.856, MQ(M4) = 20, MUG(PC1) = 0.296$
PC1 is sent to agent R. The contractee finds schedule S7 that satisfies this commitment.
$S7 : B2(0-10)B1(10-20)B3(20-30)M41(30-38)B4(38-48)$
$Utility(S7) = 0.708, MUC(PC1) = 0.76 - 0.708 = 0.052$
M41 is chosen in S7 that produces the quality of 12 that is better than M4 which produces the quality of 10. So PC1 is accepted.

By now, the contractor has obtained two acceptable commitments: CC and PC1, PC1 being better than CC because it provides a better combined utility. If the contractor decides to stop the negotiation here, it chooses PC1 and inform the contractee of its decision. Instead of the initial proposal PC (finishing time 30, quality 10), commitment PC1 (finishing time 38, quality 12) is adopted as the final solution.

4 Experiment & Evaluation

The negotiation mechanism described in the previous section serves as a basis for a family of protocol variations differing in the criteria for the negotiation process termination. We examine the following three protocols in our experiment.

- Single Step : The contractor sends a proposal commitment to the contractee, the contractee accepts PC if MUG(PC) > MUC(PC); otherwise it rejects PC. End.

[2]Note that we represent the quality of these tasks not using their local qualities but using their MQ instead as related to the contractor.

- Multistep-One-Try : The contractor sends PC to the contractee, if PC is accepted, end; otherwise continues the "proposal", "counter proposal", "new proposal" ... negotiation process until one acceptable solution is found or the allocated time is up;
- Multistep-Multiple(n) : Try The contractor sends PC to the contractee, if PC is accepted, end; otherwise continues the "proposal", "counter proposal", "new proposal" ... negotiation process until n acceptable solutions are found or the allocated time is up; We use n=2 in our experiments described next.

To examine how different protocols work in different situations and find what major factors affect the negotiation outcomes, we have built two agents, the contractor and the contractee. The utility the agent gains by performing task T using schedule S is a weighted function of the quality achieved, the cost and duration expended when performing task T.

$$
\begin{aligned}
utility(S) &= quality\_gain(S)*quality\_weight + \\
&\quad cost\_gain(S)*cost\_weight + \\
&\quad duration\_gain(S)*duration\_weight \\
quality\_gain(S) &= \frac{quality(S)}{quality\_threshold} \\
cost\_gain(S) &= \frac{cost\_limit - cost(S)}{cost\_limit} \\
duration\_gain(S) &= \frac{duration\_limit - duration(S)}{duration\_limit}
\end{aligned}
$$

quality(S), cost(S) and *duration(S)* are the quality achieved cost spent and time spent by schedule S. *quality_threshold, cost_limit, duration_limit, quality_weight, cost_weight and duration_weight* are defined in the agent's criteria function, the first three values specify the quality the agent wants to achieve from this task, the cost and the time it wants to expend on this task; the other three values specify the relative importantce of the quality, cost and duration attributes.

Each agent sequentially processes 24 different task structures. Each task structure is a variant of the basic task structure shown in Figure 2 and Figure 3. The number of deadline constraints attached to a task varies from 0 to 2, and the number of enables interrelationships among tasks varies from 0 to 3. For example, in Figure 2, there is a deadline constraint attached to the task M2, and there is an enables interrelationship between M4 and M5. Also the outcome characteristics (such as duration/cost) of some methods are changed randomly to produce more variants. There is a total of 576 (24*24) test cases obtained from the combinations of these task structures

We collect the following data for each test case in the experiment:

- Outcome(Success/Fail) : A negotiation session is successful if it ends with a commitment that increases the combined utility. Otherwise it fails.
- Utility Gain: the difference of the MUG(C) and MUC(C), C is the final adopted commitment. If the negotiation session fails, utility Gain is 0.
- Gain Percent : the percent of the utility gain out of the combined utilities without performing the task allocation.
- Marginal quality of NL : The overall quality increment for the contractor by having task NL performed as its first proposal.
- Complexity of task structures - The number of the constraints(deadline and enables relationship) in the task structures is mapped to the complexity.
- Number of Negotiation Step - The length of the negotiation series(Proposal[1] - Counter Proposal[2]- Proposal[3] - Counter Proposal[4] - ...).

	Success	AGP	ANNS
Single Step	150	0.07752	1.0
Multistep-One-Try	531	0.1477	1.74
Multistep-Multiple-Try	531	0.1533	2.80

Table 1: comparison of protocols (AGP: the average of the gain percent over all the successful cases. ANNS: the average number of the negotiation steps over all the cases.)

Table 1 show comparison of these three protocols. Out of these 576 test cases, Single-Step protocol succeeds in 150 cases, both Multistep-One-Try and Multistep-Multiple-Try protocols succeed in 531 cases. Among these 531 cases, there are 153 cases in which the Multistep-Multiple-Try protocol finds a better solution than the Multistep-One-Try protocol. There are 45 cases in which all the protocols fail. In these cases, the MQ(NL) is relatively very low, and the contractee can not find a solution with MUC less than MUG given its local constrained task structure.

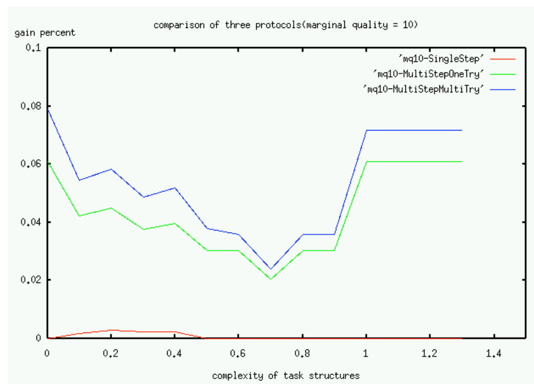

Figure 4: comparison of three protocols (marginal quality = 10)

The 576 cases can be divided into three groups according to the marginal quality of NL. Figure 4,5, and 6 shows the comparison of these three protocols in three different situations: NL has no inter-relationship with another method; NL enables one other method; NL enables two other methods. A method that is enabled by NL can not be executed until NL is performed. Each method has quality 10. Accordingly, marginal quality(NL) is 10, 20 and 30. Marginal quality is used to evaluate how important task NL is to the contractor. The more important NL is, the larger the marginal gain the commitment can have and the higher the likelihood of the contractee finding a solution. We use the number of constraints (deadline constraint and interrelationship among tasks) to estimate the complexity of the task structure. The more complex the agents' local task structures are, the more difficult it is to find a solution by negotiation.

When marginal quality is 10 (Figure 4), the multi stage protocols are obviously better than the Single-Step protocol. The single shot protocol only succeeds when the task structures have very few constraints. The Multistep-Multiple-Try protocol performs better on these cases than the Multistep-One-Try protocol.

When marginal quality is 20 (Figure 5), the Single-Step protocol works better than in the previous situation because the MUG of the first proposal is higher, so the contractee has a better chance of finding a schedule where the MUC is less than

MUG thus the first proposal is accepted. It fails only in the very highly constrained situation, but the gain percent is much less than that of the multi-step protocols because it does not do any further search. The Multistep-Multiple-Try protocol performs better than the Multistep-One-Try protocol in the medium constrained situation. When there are fewer constraints or very many constraints, extra search does not bring a better solution. Because when there are fewer constraints, it is very likely the previous search has found a very good solution. When there are many constraints, it is hard to find a better solution as result of extra search.

When the marginal quality is 30 (Figure 4), it is significant given that the contractee's local maximum quality is 40. Because task NL is so important, in most cases, the first proposal is accepted, thus the Multistep-Multiple-Try protocol does not do further search; it only does further search if the first acceptable solution is a counter proposal. In other cases, the second solution is not better than the first one, because the first one is already very good (about 20% gain), so the Multistep-Multiple-Try protocol has the same gain as the Multistep-One-Try protocol.

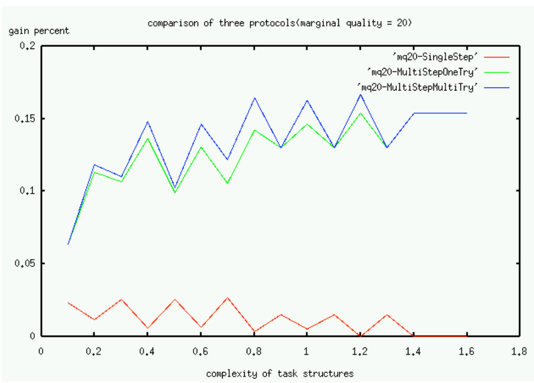

Figure 5: comparison of three protocols (marginal quality = 20)

Based on the above mentioned results, we arrive at the following observations:

1. If the task being negotiated is very important for the agent, the Multistep-One-Try protocol works well, it produces a much better solution than the Single-Step protocol without too much of an extra effort.

2. If the task is not significant and the task structures have medium constraints, the Multistep-Multiple-Try protocol performs better than others. But it also expends more effort on negotiation. The agent could decide if it is worthwhile to spend any extra effort. If the task structures have very few or very tight constraints, the Multistep-One-Try protocol is sufficient.

5 Conclusion & Future Work

In this paper, we present a cooperative negotiation mechanism over the multi-dimensional utility function. We show the application of this mechanism in the task allocation domain. We have studied how different protocols work in different situations, to see what factors affect the agent striking a good balance between the negotiation gain and the negotiation cost.

We will continue our work in two directions. One is to extend this mechanism to multi-linked negotiation where multiple related negotiation issues occur simultaneously. Another one is to obtain a better understanding of the negotiation problem

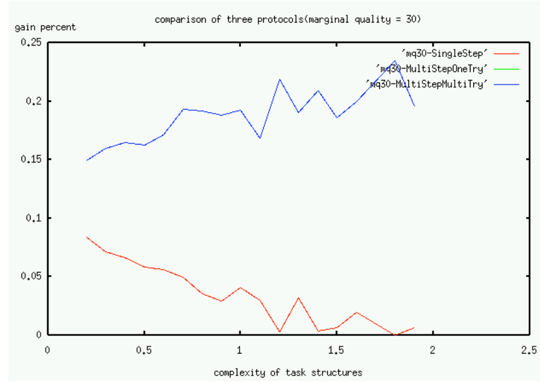

Figure 6: comparison of three protocols (marginal quality = 30)

characteristics. These characteristics should help us rate negotiation problems by their difficulty, estimate the probability of finding a good solution before the negotiation is even started, and ultimately, help the agent make a more reasonable decision about the probable cost and duration of a negotiation.

References

[1] Decker, K., Lesser, V. R. Quantative Modeling of Complex Environments. In International Journal of Intelligent Systems in Accounting, Finance and Management. Special Issue on Mathematical and Computational Models and Characteristics of Agent Behaviour., Volume 2, pp. 215-234, 1993.

[2] Edmund H. Durfee and Victor R. Lesser Negotiating Task Decomposition and allocation Using Partial Global Planning. Distributed Aritificial Intelligence, Volume 2. 1987.

[3] Lander, S. and Lesser, V. Understanding the Role of Negotiation in Distributed Search Among Heterogeneous Agents. In Proceedings of the International Joint Conference on Artificial Intelligence, 1993.

[4] Laasri, B., Laasri, H., Lander, S. and Lesser, V. A Generic Model for Intelligent Negotiating Agents. In International Journal on Intelligent Cooperative Information Systems, Volume 1, Number 2, pp. 291-317, 1992.

[5] O'Hare and N.R.Jennings eds. 1996. Chapter 7, Negotition Priciples. Foundations of Distributed Artificial Intelligence:John Wiley & Sons, Inc.

[6] Moehlman, T., Lesser, V., and Buteau, B. Decentralized Negotiation: An Approach to the Distributed Planning Problem. In Group Decision and Negotiation, Volume 1, Number 2, pp. 161-192, 1992, K. Sycara (ed.), Norwell, MA: Kluwer Academic Publishers.

[7] Robert H. Guttman and Pattie Maes. Cooperative vs. Competitive Multi-Agent Negotiations in Retail Electronic Commerce Proceedings of the Second International Workshop on Cooperative Information Agents (CIA'98). 1998. Paris, France.

[8] Sandholm, T. 1993. An Implementation of the Contract Net Protocol Based on Marginal Cost Calculations. Eleventh National Conference on Artificial Intelligence (AAAI-93), Washington DC, pp. 256-262.

[9] Gerhard Weiss eds. 1999. Chapter 5, Distributed Rational Decision Making. Multiagent System:MIT press

[10] Thomas Wagner, Alan Garvey, and Victor Lesser. Criteria-Directed Heuristic Task Scheduling. Intl. Journal of Approximate Reasoning, Special Issue on Scheduling, 19(1-2):91–118, 1998. A version also available as UMASS CS TR-97-59.