Leveled Commitment Contracting among Myopic Individually Rational Agents

Martin R. Andersson and Tuomas W. Sandholm[1]
Washington University, Department of Computer Science
One Brookings Drive
St. Louis, MO 63130-4899
{mra, sandholm}@cs.wustl.edu

Abstract

In automated negotiation systems consisting of self-interested agents, contracts have traditionally been binding, i.e., impossible to breach. Such contracts do not allow the agents to act efficiently upon future events. A leveled commitment protocol allows the agents to decommit from contracts by paying a monetary penalty to the contracting partner. The efficiency of such protocols depends heavily on how the penalties are decided.

In this paper, different leveled commitment protocols and their parameterizations are empirically compared to each other and to several full commitment protocols. Many different aspects of contracting are studied, such as social welfare achieved, CPU-time usage, and amount of contracting and decommitting.

If a global clock is used for increasing the decommitment penalties, infinite decommitment loops are prevented, while a local clock cannot guarantee this. Concerning solution quality, the leveled commitment protocols are significantly better than the full commitment protocols of the same type, but the differences between the different leveled commitment protocols are minor.

1. Introduction

New technology has increased the importance of automated negotiation in modern society. Two examples applications are automated markets for electric power [15, 18] (which are a reality in both the United States and in Europe), and electronic commerce [8] (which has become more viable with the expansion of the Internet and novel methods for electronic payments).

Contracts in automated negotiation systems consisting of *self-interested* agents have traditionally been binding, *i.e.*, impossible to breach. Such contracts do not allow the agents to act efficiently upon future events because contracts might become unfavorable to one or both of the agents after the contracting. If the agents were allowed to breach contracts, they could accommodate changes in the environment more efficiently and the social welfare could improve.

In *contingency contracts* the obligations of the contracts are made conditional on future events, which can increase the expected payoff of both parties, enabling contracts that are impossible via full commitment [10]. However, it is necessary to anticipate, enumerate, and monitor all future events, and the associated complexity can be high.

Recently, a *leveled commitment protocol* has been proposed, which allows self-interested agents to decommit from a contract by simply paying a decommitment penalty to the contract partner [14]. In this protocol, the decommitment penalties are decided at the time of contracting and the penalties do not need to be the same for the two contracting parties. This leveled commitment protocol offers Pareto improvements over full commitment protocols and can make contracts individually rational to both parties even in cases where full commitment contracts cannot.

The concept of breaching contracts in the real world is analyzed by Posner [9]. His main ideas are that the party that breaches must compensate the other party for lost profit and that the penalties for breaching contracts should be set so that the social welfare is maximized. Diamond and Maskin [7] have studied systems in which both agents can decommit from a contract by paying a decommitment penalty to the other party of the contract. Those penalties can be set in different ways: they can be compensatory or privately decided (*i.e.* liquidated; not necessarily decided by the parties of the contract – maybe imposed by a court).

[1]Supported by NSF CAREER award IRI-9703122 and NSF grant IRI-9610122.

In this paper leveled commitment contracting protocols are empirically compared to full commitment protocols. The different types of full commitment contracts used in the comparisons are: *original-* (O) contracts which is the contract type most commonly used in multiagent contracting systems and only allows for one task to move from one agent to another at a time [16][17], *cluster-* (C) contracts which move two or more tasks between two agents [13], *swap-* (S) contracts which let two agents swap tasks (one task is transferred from each agent to the other agent) [13], and *multiagent-* (M) contracts which allow at least three tasks to be transferred between at least three agents [13]. Any time a contract is performed, agents that take on tasks can receive side payments from the agents that give the tasks to cover their expenses from handling those tasks. More elaborate discussions on the different full commitment contract types are presented in [2, 12, 13].

To examine the difference in contracting behavior and reachable optima for different contract types, a multiagent version of the optimization Traveling Salesman Problem (TSP) was constructed.[2] The price of a contract is decided so that the profit of the contract is split equally between the contracting parties.

The multiagent TSP is defined as follows: several salesmen will visit several cities in a world that consists of a square with sides of length one. Each city must be visited by exactly one salesman, and each salesman must return to his starting location after visiting the cities assigned to him. A salesman can visit the cities assigned to him in any order.

Initially the locations of the cities and starting points of the salesmen are chosen randomly as is each salesman's initial assignment of cities to visit. After this initial assignment the salesmen can exchange cities with each other.

The cost c_{qr} of traveling between locations q and r (either city or location of salesman) equals the Euclidean distance between the locations. The total cost each salesman incurs, c_i, when visiting his cities, is the sum of the costs along his tour:

$$c_i = \sum_{\substack{q \text{ and } r \text{ that} \\ \text{are consecutive} \\ \text{cities on the tour} \\ \text{of salesman } i}} c_{qr}$$

The objective of agent i is to maximize payments received from others (for handling their tasks or from decommitment penalties received) minus the payments paid to others (for handling this agent's tasks and decommitment penalties paid to others) minus the agent's own total cost of traveling, c_i. The social welfare is $-\sum_i c_i$. Neither side payments nor decommitment penalties affect the social welfare since they merely redistribute wealth among the agents. In the following discussion the general multiagent task allocation terms, "agent" and "task" will be used instead of the TSP specific terms "salesman" and "city".

This paper studies the efficiency of different variants of leveled commitment contracting protocols. The different protocols and their features are described in Section 2 which is followed by a presentation of the results in Section 3. Section 4 concludes the paper and outlines future research directions.

2. Protocols Compared

In this paper, different leveled commitment protocols are compared to each other and to several full commitment protocols. Table 1 presents the notation used in the paper.

T	Set of tasks
t	Task, element of T
A	Set of agents
a	Agent, element of A
ρ	Contract price
,	Set of decommitment penalties
γ	Decommitment penalty, element of ,
$\underline{\gamma} = \underline{\gamma}_0 + \underline{\gamma}_1 \rho$	Linear mechanism for setting the decommitment penalty at $\tau = 0$
$\underline{\gamma}_0$	Fixed constant of the penalty
$\underline{\gamma}_1$	Fraction of contract price used in the penalty
$\gamma = \underline{\gamma}\alpha^{\tau}$	Decommitment penalty by time τ (when increased at the rate α)
α	Rate of increase of the decommitment penalty
τ	Time
τ_g	Time that has elapsed from the start of the entire negotiation
τ_l	Time that has elapsed from the making of a specific contract
r	Ratio bound of the social welfare
c	Cost for the agent to execute a particular contract

Table 1. Symbols used in the paper.

2.1. Leveled Commitment Protocols

All the leveled commitment protocols in this study were based on full commitment O-contracts in the sense that in each contract one task was transferred between the agents. Four different properties of the leveled commitment contracts were varied:

Decommitment Strategies. When offered a new contract, the agent needs to check if it would be profitable to decommit from any of the earlier contracts it has taken on. It might be profitable for an agent to decommit from one contract, or from any combination

[2]This is the same domain as the one used in a comparison of different full commitment protocols [3, 5]. The TSP was used as an example domain because it is combinatorially complex (NP-complete) and the space of task allocations contains many local optima when using hill-climbing-based contracting algorithms [11, 13]. Another advantage of the TSP is its structural simplicity, which provides for repeatability and presentability without unnecessary contextual overhead.

of the contracts it has. Two different methods to determine from which contracts the agent should decommit from were studied. When searching[3] for profitable combinations of contracts to decommit from, either the first combination found profitable, or the combination that maximized the immediate payoff for the agent was selected.

Methods of Setting the Constant Penalty. Two different methods of setting the constant part of the penalty, $\underline{\gamma}$, were tried in the simulations.[4]

In the first method, the penalty was decided by the protocol: $\underline{\gamma} = \underline{\gamma}_0$, *i.e.*, it was the same in all contracts no matter what the contract price was. This penalty is referred to as *fixed*, since it does not vary between the different contracts.

In the second method of choosing the constant part of the penalty, it was made dependent on the contract price ρ: $\underline{\gamma} = \underline{\gamma}_1 \rho$, *i.e.*, the penalty was a fraction, $\underline{\gamma}_1$, of the contract price. The fraction is also referred to as a *percentage*. Throughout one negotiation the same percentage was used.[5]

Methods of Sharing the Profit. Two different ways of splitting the profit from a contract were used. In both methods the profit from the contract was divided equally between the agents. However, in one the profit was deducted with the potential decommitment penalties, in the other not.[6]

Methods of Increasing the Penalty. A breach close to the execution deadline of the contract or late in a negotiation is likely to be more costly to the victim of the breach since it can be hard to find someone to contract with within a short amount of time. In order to prevent such occurrences, the decommitment penalties can be increased over time. Another reason for increasing the decommitment penalties is that the agents otherwise can get stuck in infinite decommit-recommit loops.

The penalty was increased exponentially: $\gamma = \underline{\gamma}\alpha^{\tau}$, where τ is the time and α is the rate of increase. The starting point of the time τ was chosen in two different ways:

- The overall starting point of the negotiation. That means that all the penalties, γ, in the system were equal at any given time.
- The time when the contract in question was made. In this case the penalty, γ, for each contract depended on the time at which the contract was made.

[3]Let T be the set of contracts that the agent has. The order of the search was: $(t_1), (t_2), \ldots, (t_{|T|}, (t_1, t_2), (t_1, t_3), \ldots, (t_1), t_{|T|}), (t_2, t_3), \ldots, (t_{|T|-1}, t_{|T|}), (t_1, t_2, t_3), \ldots$, where $t \in T$.

[4]The constant part refers to the penalty used when there is no increase of penalties with the time of negotiation, which also is the penalty at time zero when the penalties are increased over time.

[5]The experiments also include the case where the penalties were set to zero, *i.e.*, $\underline{\gamma} = 0$, (*i.e.* a fixed penalty equal to zero or a percentage penalty equal to zero).

[6]Note that the agents always acted myopically rational, so the former method would enable less contracts.

Three different protocols were constructed from these two ways of choosing time zero. The first, called *global*, had the time of negotiation as the starting point. The second, *local*, used the time of contracting as the starting point. The third construction combined these two methods by taking the average time of the two, $\tau_c = \frac{\tau_g + \tau_l}{2}$, where t_g refers to the time elapsed from the beginning of the negotiation and t_l refers to the time elapsed from when the contract was made. This way of increasing the penalty is referred to as *combined*.

Parameterizations Studied. Different values of the fixed decommitment penalty $\underline{\gamma}_0$, the percentage $\underline{\gamma}_1$ of the constant price, and the rate α of increase of the decommitment penalty were used. These are referred to as the *parameterization* of the leveled commitment protocol (Table 2).

Fixed decommitment penalty: $\underline{\gamma}_0$
0.0
0.005
0.01
0.1
0.5

Rate of increase: α
1.0
1.001
1.01
1.05

Percentage of the contract price: $\underline{\gamma}_1$ (Note that *e.g.* 5%=0.05)
0.0
0.01
0.1
0.2
1.0

Table 2. Parameterizations tried in the experiments

2.2. Conventions Used in Naming the Protocols

The full commitment contract types are denoted as in the introduction (*i.e.* O, C, S, and M). The notation for the different combinations of features of the leveled commitment protocols is summarized in Table 3. Basically the capital letter L (for leveled commitment) is followed by four letters that characterize features of the protocol, followed by two numbers which constitute the parameterization. One or more properties may be left out if it is clear from the context which properties are referred to. If there is a dash ("-") in place of a property, all possible types of that property are considered.

Simulations with leveled commitment protocols of all combinations of the properties were conducted for several different parameterizations, that is, different values of fixed decommitment penalties $\underline{\gamma}_0$, or fractions of the contract price $\underline{\gamma}_1$, and the rate α at which they were increased.[7]

Notation of a leveled commitment protocol: Labcd-$\underline{\gamma}$-α. L symbolizes that it is a leveled commitment protocol. Four properties are denoted a, b, c, and d, and the parameterization $\underline{\gamma}$ and α.	
a∈{f, m}	Decommitment strategy: contract combination from which to decommit is chosen so that profit is maximized (m), or the first profitable combination is chosen (f).
b∈{f, p}	The constant decommitment penalty: fixed (f), or percentage (p).
c∈{p, w}	The even split of profit from the contract: incorporates decommitment penalties (p), or does not incorporate decommitment penalties (w).
d∈{g, c, l}	Method of increase of the penalties: global (g), combined (c), or local (l).
$\underline{\gamma} \in \Re_0^+$	For the protocols with a fixed penalty, this is the fixed ecommitment penalty, and for the protocols with percentage penalties this is the fraction of the price that is the decommitment penalty.
$\alpha \in \Re_0^+$	The rate of increase of the decommitment penalty.

Table 3. Notation of the leveled commitment protocols.

3. Results

To compare the different contract types, the mean ratio bounds and the significance of the difference in ratio bounds were computed so the results could be statistically analyzed (in paired t-tests [6]). The description below concerns a fixed number of agents and a fixed number of tasks, but each such experiment was conducted for all combinations of agents (2..8) and tasks (2..8). Let x_j^l denote the social welfare of the jth problem instance, $j \in 1, \ldots, n$ (n=1000), after task reallocation[8] has been performed until a local optimum has been reached using contract type $l \in \{$O, C, S, M, L$_1$,...,L$_{max}$, G$\}$, where L$_i$ denotes one of the leveled commitment contracts and G indicates the global optimum (or equivalently OCSM-contracts). The ratio bound, r_j^l, for the jth problem instance using l-contracts is given by the ratio of the social welfare of the local optimum obtained using l-contracts over the social welfare of the global optimum $r_j^l = \frac{x_j^l}{x_j^G}$ The difference in ratio bounds between two different contract types applied to the same instance j, is given by $r_j^{kl} = r_j^k - r_j^l$. The mean difference, $\overline{r}^{kl}$ between the contract types is $\overline{r}^{kl} = \frac{1}{n}\sum_{j=1}^{n} r_j^{kl}$. The average of the ratio bounds of the different contract types, $l \in \{$O, C, S, M, L$_1$,...,L$_{max}\}$ is $\overline{r}^l = \frac{1}{n}\sum_{j=1}^{n} r_j^l$.

3.1. Protocol Comparison Based on Solution Quality

The mean differences of ratio bounds, $\overline{r}^{kl}$, between leveled commitment contracts and full commitment O-contracts and C-contracts were analyzed. A comparison was conducted among the different leveled commitment protocols as well.

Leveled Commitment vs. Full Commitment. Compared to full commitment O-contracts, the leveled commitment protocols clearly improve the social welfare of the solution (Figures 1 and 2), if the number of agents and tasks is large enough. In earlier research [3, 5], the different full commitment protocols have been compared; Figure 1, bottom graph, summarizes which of O-, C-, S-, or M-contracts were the best for different numbers of tasks and agents.

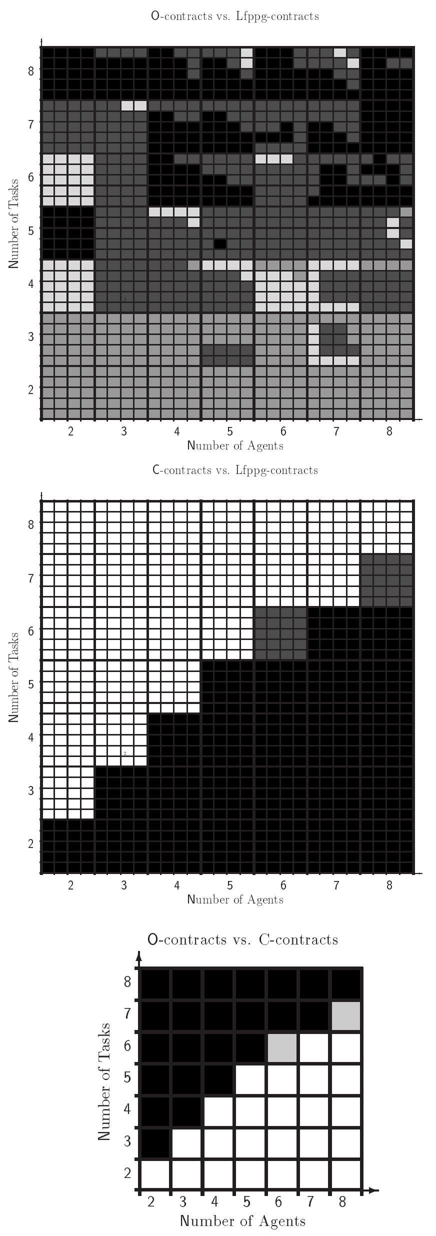
Comparing the graphs in Figure 1 we see that leveled commitment O-contracts and full commitment O-contracts are better than full commitment C-contracts in the same region of the agent-task space: C-contracts outperform the other two when the number of tasks is greater than the number of agents. This tells us that the leveled commitment framework cannot completely play the role of a different full commitment contract type. In certain domains, more can be gained by changing full commitment protocols than by using leveled commitment on top of the old full commitment protocol.

Comparison of Different Leveled Commitment Protocols. All of the leveled commitment protocols in the study were compared against each other. It was found that the differences in ratio bounds between many of the different leveled commitment protocols and parameterizations were so small (Figure 2) that one should not conclude that one of the protocols clearly performed better than the others in terms of solution quality (even for our large sample sizes). However, there are significant differences between the different leveled commitment protocols when other criteria than solution quality are considered, as will be dis-

[7]A rate of increase $\alpha = 1.0$ is equivalent to leveled commitment protocols without increment of the penalty over time. If $\underline{\gamma} = \infty$, the leveled commitment contracts become equivalent to full commitment contracts.

[8]A complete description of the contracting system and the sequencing of the contracts can be found in [1, 2].

[9]In the top two graphs each square representing a combination of agents and tasks contains 4×5 rectangles. Each of those rectangles symbolizes a parameterization of the leveled commitment protocol. On the x-axis is the rate of increase of the decommitment penalties (values from 1.0 to 1.05), and on the y-axis is the decommitment penalty (values from 0.0 to 0.5). The parameter values increase to the right and upwards within the rectangle. See Table 2 for the specific values.



□ The former protocols perform better, at the 0.05 significant level in a paired t-test.

░ The former perform better, but not at the 0.05 significance level in a paired t-test.

▒ The two protocols have identical ratio bounds.

▓ The latter perform better, but not at the 0.05 significance level in a paired t-test.

■ The latter protocol perform better, at the 0.05 significant level in a paired t-test.

Figure 1. Pairwise comparison of the differences in ratio bounds $\overline{r}^{kl}$ between O-, C-, and Lfppg-contracts (S- and M-contracts were always worse). The two former contracts are full commitment contracts and the last one is a leveled commitment contract.[9]

cussed later. The solution quality also decreased with higher penalties and rate of increase of penalties (Figure 2). This has also been seen in other research [1, 4].

We can conclude that in real world applications, when considering solution quality, it is more important to use the leveled commitment protocols than it is to choose among the qualitatively different variants of those protocols. This is because leveled commitment protocols perform significantly better than the full commitment protocols (if the parameters are well set) but the differences in performance among the different leveled commitment protocols are minor.[10]

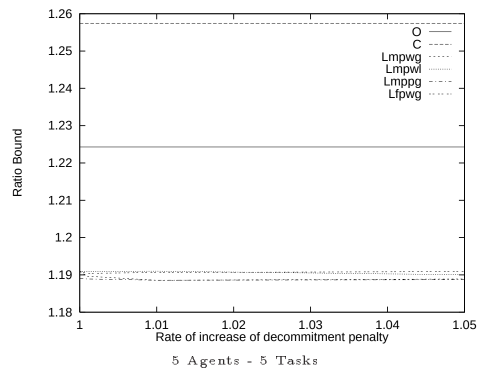


Figure 2. Ratio bounds of the social welfare as a function of the rate of increase of decommitment penalty for selected leveled commitment contracts. As a reference, the ratio bounds for two full commitment contracts (O- and C-contracts) are also shown. The leveled commitment penalty was a percentage of the contract price, with the following parameters: $\underline{\gamma}_0 = 0$ and $\underline{\gamma}_1 = 0.01$.[11]

3.2. Protocol Comparison Based on the Number of Contracts

The average number of contracts performed varied from less than 1, for the environment containing the lowest numbers of agents and tasks, to more than 45, for the problems with eight agents and eight tasks. The number of contracts performed decreased both with increased fixed and percentage decommitment penalties. The number of decommitments was related to the number of contracts performed.

The number of contracts that the agents tried to make during the negotiation includes both accepted contracts and rejected proposals. The number of tried contracts was considerably greater than the number of contracts performed. Approximately 3% of the tried contracts were acceptable to both agents, and were

[10] A more comprehensive discussion can be found in [1, 2].

[11] Note that a lower ratio bound is better since the social welfares are negative.

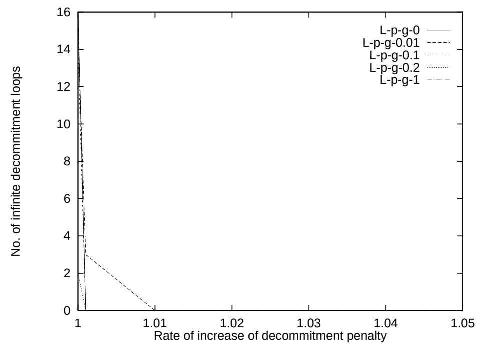
thus performed. The contracting behavior of the leveled commitment contracts converges to the behavior of the full commitment O-contract with greater penalties, $\underline{\gamma}$, as it should, since infinite decommitment penalties will emulate full commitment.[10]

Infinite Decommitting Loops. An infinite decommitment loop can occur between three agents or more. Let us look at an example with three agents: First, agent 1 and 2 commmit to a contrac. Agent 3 then makes an offer to 1. Agent 1 accepts and decommits its contract with 2. Agent 2 now offers 3 a contract, which 3 accepts after decommitting the contract it has with 1. Now agent 1 offers 2 a contract, which is accepted after a decommitment, and so the agents are back where they started form. If the penalties are not increased with the contracting time, this decommit-recommit loop can go on forever.

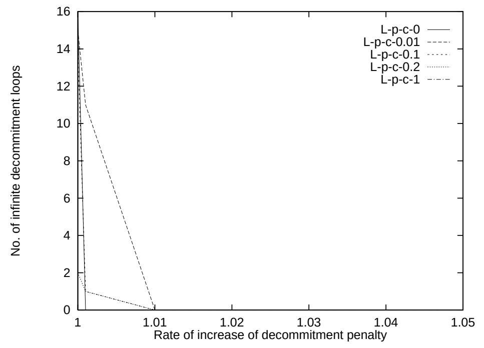
For two of the three methods of increasing the penalties over time (global and combined), decommitment loops are prevented. For the third method (local), the number of them is reduced (compared to no increase at all). With the local method it took more time before the infinite decommitment loops were terminated if at all, than it took with the global and combined methods.

Figure 3 (top graph) shows the number of infinite decommitment loops as a function of the rate of increase, α, of the decommitment penalty when the penalties were a percentage of the contract price and increased using the global method. The number of infinite decommitment loops dropped to zero very quickly, and with $\alpha = 1.01$ no infinite decommitment loops would ever occur, for any $\underline{\gamma}$ in the study. This is because the penalties γ will be so high in the end that no infinite decommitment loops can occur. The same result holds for fixed penalties.[10]

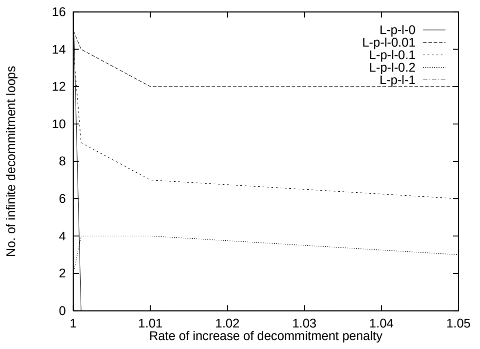
When local time is used to increase the penalties, the number of decommitment loops is often reduced, but not always for cases with small numbers of tasks and agents (Figure 3, bottom graph). The mechanism of increasing the penalties that combined the above methods was more successful than the local method, since it prevented infinite decommitment loops almost as effectively as the global mechanism, which terminated them quicker (Figure 3, middle graph). The local mechanism is more practical in open systems, since it is completely controlled by the agents involved in the contract, and does not depend on the start time of the overall negotiation. However, it does not prevent the infinite decommitment loops completely. With the local method no finite increase of the decommitment penalty can guarantee avoidance of infinite decommit-



Global, 5 Agents - 5 Tasks



Combined, 5 Agents - 5 Tasks



Local, 5 Agents - 5 Tasks

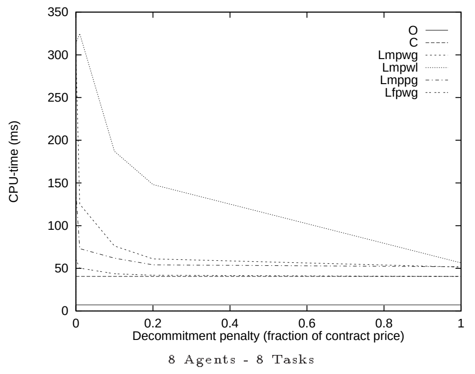
Figure 3. Number of infinite decommitment loops as a function of the rate of increase of the decommitment penalty for leveled commitment contracts with decommitment penalty as a percentage of the contract price. The protocols in the graphs used $\underline{\gamma}_0 = 0$ and $\underline{\gamma}_1 = 0.01$.

recommit loops.

Increasing the decommitment penalties is not always beneficial, since a decrease in decommitments may lead to a decrease in social welfare (Figure 2). The number of decommitments decreases when the rate of increase (α) increases, since fewer contracts can be decommitted profitably with higher penalties. When the number of decommitments decreases, the advantages of the leveled commitment protocols diminish, since these protocols become similar to full commitment protocols. Therefore the social welfare may be lower with increasing penalties.

3.3. Protocol Comparison Based on CPU-time Usage

As expected, the CPU-time usage of leveled commitment protocols was higher than that of full commitment O-contracts, Figure 4.



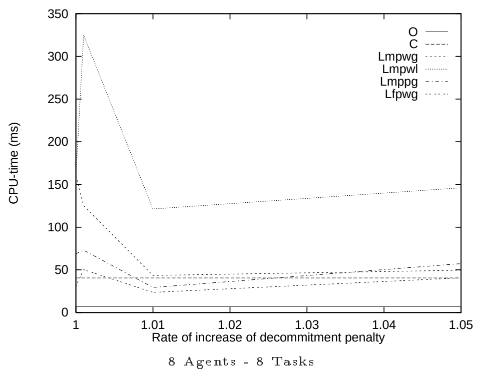


Figure 4. CPU-time usage as a function of the decommitment penalty (left) and the rate of increase of the decommitment penalty (right) for selected leveled commitment contracts percentage penalties. As a reference, the CPU-time usage of two full commitment protocols (O- and C-protocols) is also shown. The protocols in the top graphs used $\alpha = 1.001$ and in the bottom $\underline{\gamma}_0 = 0$ and $\underline{\gamma}_1 = 0.01$.

For domains with a large number of agents and tasks, the CPU-time usage for the leveled commitment protocols was considerably higher than for domains with few agents or tasks.[12]

As the decommitment penalty was increased (Figure 4) the CPU-time usage decreased, which is partly due to the fact that fewer decommitment loops occur when the penalties are high. Another reason is that there will be fewer decommitments in the system over all. Even when the penalties were set to infinity to simulate full commitment, the CPU-time usage was greater than for the full commitment O-contracts. That is a result of the overhead of the leveled commitment protocol and the fact that each agent still tries to find profitable decommitments.

The CPU-time usage decreased quickly for increasing values of the fixed decommitment penalty, as well as for higher percentages when decommitment penalties were based on a percentage of the price. For high penalties, the CPU-time usage remains fairly constant, and the usage does not vary extensively between the different leveled commitment protocols. For all of them the CPU-time usage is 6-10 times greater than that of the full commitment O-contracts.

If the rate of increase, α, of the decommitment penalty is very low, CPU-time usage is higher than if the penalties were not increased at all. This can be explained by the deliberation the agents must conduct in order to decide from which contracts to decommit, the extra overhead of increasing the penalties, and the extra computation to increase them. As the rate of increase, α, was increased more, CPU-time usage decreased, forming a local minimum before it increased again for high values of α (Figure 4).[13] This means that it is possible to find an optimal parameterization (considering CPU-time usage) for interior values of the increment rate. In the case of eight tasks and eight agents, comparing CPU-time usage (Figure 4) and the ratio bound (Figure 2), it can be seen that the ratio bound also had an interior local minimum. These two minima (for the CPU-time usage and the ratio bounds) actually coincide. For considerably lower numbers of agents and tasks, this behavior could not be observed.

The leveled commitment contracts that increase their penalties according to the local method proved to have the highest CPU-time usage of the contracts studied. This can be explained by the fact that they do not reduce the number of infinite decommitment

[12]The CPU-time usage in the extreme case (*i.e.* the worst case considering CPU-time usage) was approximately 40 times greater than that of the full commitment O-contracts. Compared to the other full commitment protocols, this is still a very moderate amount: the M-contracts, for example, used several orders of magnitude more CPU-time [3].

[13]This increase was surprising to us, and we are currently exploring the reasons why it happens.

loops as effectively as the global and combined increment methods do.

4. Conclusions

The efficiency of leveled commitment protocols depends on how the decommitment penalties are decided. In this work we have investigated several different methods for setting them, *e.g.*, the penalties can vary with the contract price or be the same for all contract prices, or they can increase over time or remain the same throughout the contracting process.

A breach close to the execution deadline of the contract or late in a negotiation is likely to be more costly to the victim of the breach, since it could be hard to find someone to contract with within a short amount of time. In order to prevent such occurrences, the decommitment penalties can be increased over time. Another reason for increasing the decommitment penalties is that the agents otherwise can get stuck in infinite decommit-recommit loops. We have shown that increasing the penalties over time is an effective way of preventing infinite decommitment loops.

The CPU-usage can be reduced by choosing the right mechanism of increasing the penalties. There is a trade-off between solution quality and CPU-time usage: both decrease with higher penalties. There is also a trade-off in the search for contracts to decommitment from. If the first profitable decommitment combination was chosen the CPU-time usage was lower, but so was the solution quality. If the most profitable combination was chosen, social welfare was higher, but more CPU-time was used.

Concerning the solution quality, one cannot conclude that one of the leveled commitment protocols is better than others because the differences were small. Comparing the leveled commitment protocols (with an O-contract as a base) to the full commitment O-protocol, the solution quality for the leveled commitment protocols was significantly better. The domains in which the leveled commitment O-protocols and full commitment O-protocols are significantly better than the full commitment C-protocols coincide.

Because C-protocols had a clear region of superiority in the agent-task space, in the future we will study leveled commitment protocols that lie on top of full commitment cluster, swap, and multiagent contracts as well.

References

[1] M. R. Andersson. Performance of leveled commitment protocols for automated negotiation: An empirical study. Master's thesis, Royal Institute of Technology, Stockholm, Sweden, 1998.

[2] M. R. Andersson and T. W. Sandholm. Leveled commitment contracting among myopic individually rational agents. Technical Report WUCS-97-47, Washington University, Department of Computer Science, 1997.

[3] M. R. Andersson and T. W. Sandholm. Contract types for satisficing task allocation: II experimental results. In *AAAI Spring Symposium Series: Satisficing Models*, pages 1–7, Stanford University, CA, Mar. 1998.

[4] M. R. Andersson and T. W. Sandholm. Leveled commitment contracts with myopic and strategic agents. In *Proceedings of the National Conference on Artificial Intelligence*, Madison, WI, July 1998.

[5] M. R. Andersson and T. W. Sandholm. Sequencing of contract types for anytime task reallocation. In *First Workshop on Agent Mediated Electronic Trading (AMET)*, Minneapolis, MN, May 1998.

[6] P. R. Cohen. *Empirical Methods for Artificial Intelligence*. MIT Press, 1995.

[7] P. A. Diamond and E. Maskin. An equilibrium analysis of search and breach of contract, I: Steady states. *Bell Journal of Economics*, 10:282–316, 1979.

[8] R. Kalakota and A. B. Whinston. *Frontiers of Electronic Commerce*. Addison-Wesley Publishing Company, Inc, 1996.

[9] R. A. Posner. *Economic Analysis of Law*. Little, Brown and Company, 2nd edition, 1977.

[10] H. Raiffa. *The Art and Science of Negotiation*. Harvard Univ. Press, Cambridge, Mass., 1982.

[11] T. W. Sandholm. An implementation of the contract net protocol based on marginal cost calculations. In *Proceedings of the National Conference on Artificial Intelligence*, pages 256–262, Washington, D.C., July 1993.

[12] T. W. Sandholm. *Negotiation among Self-Interested Computationally Limited Agents*. PhD thesis, University of Massachusetts, Amherst, 1996. Available at http://www.cs.wustl.edu/~sandholm/dissertation.ps.

[13] T. W. Sandholm. Contract types for satisficing task allocation: I theoretical results. In *AAAI Spring Symposium Series: Satisficing Models*, pages 68–75, Stanford University, CA, Mar. 1998.

[14] T. W. Sandholm and V. R. Lesser. Advantages of a leveled commitment contracting protocol. In *Proceedings of the National Conference on Artificial Intelligence*, pages 126–133, Portland, OR, Aug. 1996.

[15] T. W. Sandholm and F. Ygge. On the gains and losses of speculation in equilibrium markets. In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence*, pages 632–638, Nagoya, Japan, Aug. 1997.

[16] S. Sen. *Tradeoffs in Contract-Based Distributed Scheduling*. PhD thesis, Univ. of Michigan, 1993.

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

[18] F. Ygge and J. M. Akkermans. Power load management as a computational market. In *Proceedings of the Second International Conference on Multi-Agent Systems (ICMAS)*, pages 393–400, Keihanna Plaza, Kyoto, Japan, Dec. 1996.