PROBABILISTIC AWARD STRATEGY FOR CONTRACT NET PROTOCOL IN MASSIVELY MULTI-AGENT SYSTEMS

Toshiharu Sugawara*
Computer Science and Engineering, Waseda University, Tokyo 169-8555, Japan

Toshio Hirotsu
Computer and Information Sciences, Hosei University, Tokyo 184-8584, Japan

Kensuke Fukuda
National Institute of Informatics, Tokyo 101-8430, Japan

Keywords: Task and resource allocation, Load-balancing, Massively multiagent systems, Contract net protocol.

Abstract: We propose a probabilistic award selection strategy for a contract net protocol (CNP) in massively multi-agent systems (MMASs) for effective task allocations. Recent Internet and sensor network applications require sophisticated multi-agent system technologies to enable the large amounts of software and computing resources to be effectively used. Improving the overall performance of MMASs in which thousands of agents work concurrently requires a new negotiation strategy for appropriately allocating tasks to agents. Our proposed method probabilistically selects the awardee in CNP based on the statistical difference between bid values for subtasks that have different costs. We explain how our proposed method can significantly improve the overall performance of MMASs.

1 INTRODUCTION

Recent advances in many domains, such as the Internet, sensor networks, and grid computing, (Foster, 2002), have increased the need for technologies for massively multi-agent systems (MMASs), in which thousands of agents interact with one another. In particular, a technology is needed for allocating tasks generated in real time to appropriate agents in accordance with their skills and abilities so that the abilities/resources of all agents are maximally used. Task allocation has thus attracted a great deal of attention in multi-agent systems for the purpose of obtaining efficient and high-quality services.

A number of negotiation protocols have been proposed for task allocation and the contract net protocol (CNP)(Smith, 1980) has especially been implemented in various applications (Sandholm, 1993; Weyns et al., 2006). In CNP, an agent plays one of two roles: *managers* are responsible for allocating tasks and monitoring processes and *contractors* are responsible for executing the allocated tasks. A manager agent makes a task known to the contractor agents in the announcement phase, and they then bid for the task on the basis of certain values (such as cost, duration, or payment) in the bid phase. The manager awards the contractor (or *awardee*) who made the best bid in the award phase. There have been a number of studies in which the performance and characteristics of CNP have been investigated (e.g. (Gu and Ishida, 1996)). However, most have assumed CNP in small-scale, less busy environments.

Unfortunately, the performance of CNP in an MMAS is poorly understood. Clarifying this is important because interference among agents occurs with this kind of negotiation protocol if many managers have tasks to allocate. In naive CNP, a contractor agent responds to task announcements one by one,

*This research was supported in part by Kayamori Foundation of Information Science Advancement and Grant-in-Aid for Scientific Research from the Japan Society for the Promotion of Science.

Sugawara T., Hiortsu T. and Fukuda K. (2010).
PROBABILISTIC AWARD STRATEGY FOR CONTRACT NET PROTOCOL IN MASSIVELY MULTI-AGENT SYSTEMS.
In *Proceedings of the 2nd International Conference on Agents and Artificial Intelligence - Agents*, pages 165-171
DOI: 10.5220/0002712201650171

but because many managers announce tasks simultaneously in a busy MMAS, the managers may have to wait a long time to receive a sufficient number of bids: This significantly reduce the performance of the entire system. In the original conception of CNP (Smith, 1980), the use of multiple bids was proposed to concurrently handle many announcements. If a contractor is awarded multiple bids simultaneously, however, it may not be able to provide the quality or performance guaranteed in the bids. In fact, more highly capable contractor agents tend to be selected by many managers. Additionally, the task structure, meaning a task consisting of a number of different subtasks, makes this situation more complex.

In this paper, we propose the award strategy, called the *adaptive probabilistic award strategy*, to improve the overall performance of MMAS. The first key idea of the proposed strategy is the probabilistic selection of awardee according to the task loads of the system. However, the task loads of the system are hardly given to each agent. Thus the second idea is that manager agents estimate the task loads using statistical data (more precisely, the difference in the standard deviation – SD) of bid values for different subtasks, where we assume that a task consists of a number of subtasks that have different costs.

This paper is organized as follows. First, we will discuss the model of CNP, which has been slightly modified for MMASs to avoid long waits and to reduce the number of messages, the simulation environment and the issues addressed in this paper. Then, we clarify how some degree of fluctuation can improve the overall performance even if tasks have structures and, by taking advantage of this effect, we propose the adaptive probabilistic award strategy. Finally, we experimentally show how our proposed method can significantly improve overall performance.

2 MODEL AND ISSUES

2.1 Model of CNP for Massively MASs

Let $\mathcal{A} = \{1, \ldots, n\}$ be a set of agents, $\mathcal{T}$ be a task, and $\mathcal{F} = \{f^1, \ldots, f^d\}$ be the set of skills, or functions that agents can perform. We assume that task $\mathcal{T}$ consists of subtasks, $t_1, \ldots, t_l$, (therefore, we denote $\mathcal{T} = \{t_1, \ldots, t_l\}$) and that subtask $t (\in \mathcal{T})$ requires $s(t)$-th skill, $f^{s(t)}$, to perform it, where $1 \leq s(t) \leq d$. A subtask is denoted by lower-case letter t and is simply called a task unless this creates confusion. Agent i is expressed as a tuple, $(\alpha_i, L_i, S_i, Q_i)$, where $\alpha_i = (a_i^1, \ldots, a_i^d)$ is the set of the agent's capabilities (a_i^l corresponds to the l-th skill, f^l, and $a_i^l \geq 0$; $a_i^l = 0$ indicates agent i does not have skill f^l), L_i is the location of i, and Q_i is the queue where the agent's tasks are stored, waiting to be executed one by one. Set $S_i (\subset \mathcal{A})$ is i's scope, i.e., the set of agents that i knows. The *metric* between agents, $\delta(i, j)$, is based on their locations, L_i and L_j, and is used to define the communication time (or delay) of messages between i and j.

Subtask t has an associated cost, $\gamma(t)$, which is the cost to complete it. Subtask t can be done by i in $\lceil \gamma(t)/a_i^{s(t)} \rceil$ unit times, where $\lceil x \rceil$ denotes the ceiling function. This time is also called the *execution time* of t by i. Task $\mathcal{T}$ is completed when all its subtasks are completed. The cost of $\mathcal{T}$ is defined as $\gamma(\mathcal{T}) = \sum_{t \in \mathcal{T}} \gamma(t)$.

In every unit time, $tl (\geq 0)$ tasks on average are generated according to a Poisson distribution and randomly assigned to different managers. Parameter tl is called the *task load* and denotes tl tasks per unit time, or simply tl T/t.

For CNP, we defined $\mathcal{M} = \{m_j\} (\subset \mathcal{A})$ as the set of managers, who allocate tasks, and $\mathcal{C} = \{c_k\} (\subset \mathcal{A})$ as the set of contractors, who execute the allocated tasks. Let us assume that $|\mathcal{A}|$ is large (on the order of thousands), therefore $|\mathcal{M}|$ and $|\mathcal{C}|$ are also large, and that the agents are distributed widely, like servers on the Internet.

2.2 Task Allocations for MMAS

In our experiments, we used the CNP modified for use in an MMAS to reduce the number of messages and to prevent long waits for a response. In this CNP, (1) multiple bids and *regret* and *no-bid* messages are allowed, and (2) manager m announces subtasks in $\mathcal{T}$ to restricted contractors that are selected from its scope, S_m, on the basis of an *announcement strategy*. *Regret messages* are sent in the award phase to contractors who have not been awarded the contract; *no-bid messages* are sent to managers by contractors who have decided not to bid on an announced task. These messages prevent long waits for bids and award messages (e.g., (Sandholm, 1993; Xu and Weigand, 2001)).

When manager m receives task $\mathcal{T}$, it immediately initiates the modified CNP to allocate each task $\tilde{t} (\in \mathcal{T})$ to an appropriate contractor agent. It first sends announcement messages to the contractors selected from its scope in accordance with the announcement strategy. Each contractor receiving the announcement sends back a bid message with a certain value called the *bid value*. The bid values in general might include parameters such as the price for executing the task, the quality of the result, or a combination of these values. Because we assume that agents are rational in terms

of efficiency, their bid values contain the guaranteed times for completing the task. Thus, the bid value of contractor c is $\lceil \gamma(\tilde{t})/a_c^{s(\tilde{t})} \rceil + \sum_{t \in Q_c} \lceil \gamma(t)/a_c^{s(t)} \rceil + \beta$, where β is the time required to complete the task currently being executed. With multiple bidding, c might have a number of outstanding bids. These bids are not considered because it is uncertain whether they will be accepted. Finally, manager m selects a contractor, the awardee, on the basis of the *award selection strategy*, and sends the awardee a message with the announced task. The awardee is usually the one that make the best bid (here, the lowest value).

2.3 Issue

We assume that manager agents can observe, for each subtask, the *completion time*, which is the elapsed time from the time the award message was sent to the time the message indicating that the task has been completed was received. The completion time thus includes the communication time in both directions, the queue time, and the execution time. The *overall efficiency of an MMAS* is defined as the average completion time observed for all managers; as this value is used to evaluate the system's performance, it is referred to as the *overall performance*. The issue we addressed here was to investigate the overall performance of an MMAS under a number of award strategies and to improve it by combining the advantages of a number of award strategies.

From the viewpoint of the overall performance of a MMAS, we have already tackled this issue and investigated the performance of an MMAS, especially its overall efficiency, when tasks were allocated using CNP with a variety of manager-side controls in the announcement and award phases, under the assumption that all agents were cooperative and rational in terms of efficiency (therefore, their bid values contained estimated times for completing the task)(Sugawara et al., 2008a; Sugawara et al., 2008b). This assumption is reasonable because timely responses are always of great concern in interactive and realtime services. We then found that by introducing a small fluctuation in the award phase of CNP according to the task loads the overall performance could considerably be improved. However, because they assumed that a task had no structure, i.e., a task consisted of a single subtask, their model could only be applied to limited applications.

Thus, we aim to extend this approach to more general tasks that have a certain task structure. The extension of the method proposed in (Sugawara et al., 2008a; Sugawara et al., 2008b) is not trivial. The key information to apply their method is the overall task load in the environment and this was estimated from queue lengths that were directly announced by local contractors. Queue lengths cannot, however, correctly indicate the workload in general if tasks consist of a number of different subtasks, because the queued subtasks do not involve the same costs.

2.4 Simulation Environment

We set $|\mathcal{C}| = 500$ and $|\mathcal{M}| = 10,000$ in our simulation. We assumed that the contractor agents were servers running on the Internet, providing services requested by manager agents, which correspond to clients in users' sides. The agents were randomly placed on a 150×150 grid with a torus topology. The Manhattan distance between agents i and j was introduced as the metric. The communication time ranged from 1 to 14 (in *ticks*, the unit of time in the simulation), in proportion to the value of $\delta(i, j)$.

For simplicity, let us first assume that $\mathcal{T} = \{t_1, t_2\}$ where $\gamma(t_1) = 2500$ and $\gamma(t_2) = 500$. Contractor c_i was assigned different capabilities so that the values of $\gamma(t_1)/a_{c_i}^1$ ($c_i \in \mathcal{C}$) were *uniformly distributed* over the range 20–100. Since $\gamma(t_1) = 2500$, the values of $a_{c_i}^1$ ranged from 25 to 125. We also assumed that manager agents could not do the tasks ($a_m^1 = a_m^2 = 0$) (so they had to assign the tasks to agents who could), and that $a_{c_i}^1 = a_{c_i}^2$; this means that a high-performance PC can execute any task effectively (if the functions are defined).

The results presented here are the mean values from nine independent trials. In these trials, the maximal numbers of being executed $\mathcal{T}$ every tick, which were derived from the cumulative capabilities of all contractors $\sum_{c \in C} a_c$, ranged from 8.15 to 8.30 T/t, with an average of 8.25 T/t. This is the theoretical upper limit, meaning that if the task allocation was ideal, they could execute 8.25 tasks every tick.

When contractor c was awarded a task, it immediately executed it if it had no other tasks. If c was already executing a task, the new task was stored in Q_c. The tasks in the queue were executed in turn. The queue length can be finite or infinite, and we assumed that it was infinite in this paper.[2]

Manager m's scope, S_m, consists of the nearest 50 or more contractors according to this distance. More precisely, for integer $n > 0$, let $S_m(n) = \{c \in$

[2]If the queue is finite, a number of tasks are dropped when agents are busy, but this enables agents to recover from the busy state. However, we aims at investigating the overall performance by appropriate task allocations (load-balancing), we@ daringly assume that the queue length is infinite. Note that the characteristics described in this paper are almost identical to those when this is finite.

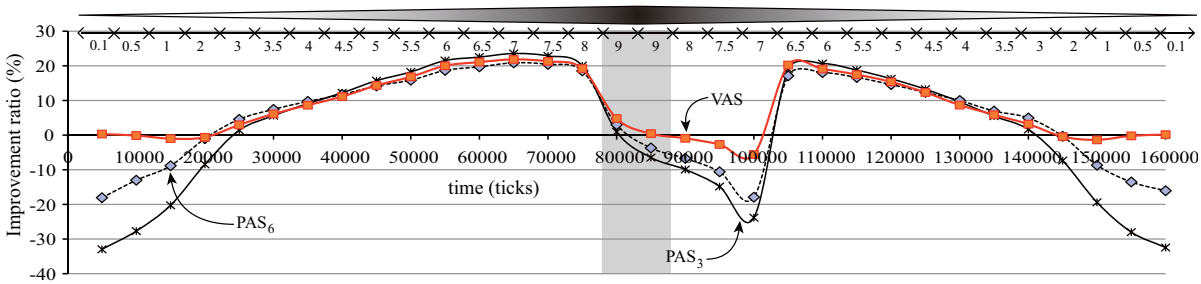

Figure 1: Ratios of completion times under PAS_k (k=3 or 6) and VAS.

$C|\delta(m,c) \leq n\}$. Then, it follows that $S_m(n) \subset S_m(n+1)$. S_m is defined as the smallest $S_m(n)$, such that $|S_m(n)| \geq 50$. Then, we fixed the announcement strategy in which m announced tasks to only 20 contractors[3] who were randomly selected from S_m.

3 EFFECT OF PROBABILISTIC AWARDING

(Sugawara et al., 2008a) reported that some degree of fluctuation in the award phase could improve overall performance when a task had no structure. Our objective of the first experiment to verify this effect occurred when a task consisted of a number of subtasks.

After a task is announced, manager m would receive bids from a number of contractors, $\{c_1, \ldots, c_p\}$. We denote the bid value from contractor c_i as $b(c_i)$. In naive CNP, m selects the contractor who submitted the best (lowest) bid. In our first award strategy, the awardee is selected according to the following probability:

$$\Pr(c_i) = \frac{1/b(c_i)^k}{\sum_{j=1}^{p} 1/b(c_j)^k}. \quad (1)$$

This *probabilistic award selection* strategy is denoted by PAS_k. Variable k is called the *fluctuation factor*. The larger the k, the smaller the degree of fluctuation; PAS_0 and PAS_∞ correspond to "random selection" and "no randomness." Therefore, PAS_∞ is the award strategy in the naive CNP.

We evaluated the overall performance for task load, tl, gradually increasing it from 0.1 (idle) to 9 (extremely busy, over the cumulative capabilities) every 5-K ticks and then returning to 0.1. The total duration was 160-K ticks. We plotted the improvement ratios from PAS_k to PAS_∞ (= CNP), which was calculated using

$$I_{CNP}(\mathrm{PAS}_k) = \frac{\wp(\mathrm{PAS}_\infty) - \wp(\mathrm{PAS}_k)}{\wp(\mathrm{PAS}_\infty)} \times 100, \quad (2)$$

where $\wp(str)$ indicates the overall performance when award selection strategy str is used.

The results are plotted in the curves labeled "PAS_3" and "PAS_6" in Fig. 1. The task loads over time are also given in this figure. The gradated "javelin" above the graph also illustrates the varied task loads. The gray area in Fig. 1 indicates the period in which the task load exceeds the cumulative capabilities. These curves indicate that (1) when tl is low (very few multiple awards occur) or tl is quite large (over the theoretical limit of cumulative capability), PAS_∞ is better than the others (PAS_k may worsen this by 35%), but (2) otherwise PAS_k ($k = 3$ or 6) can improve the overall efficiency by as much as 25%. This also clearly demonstrates the appropriate degree of fluctuation depends on the task load, tl. That is, when $tl \leq 4$, PAS_6 is better than PAS_3 but vice versa when $4 < tl \leq 7.5$. Note that the center of the curves in Fig. 1 have shifted slightly to the left because of the effect of the delayed execution of tasks queuing during the overload situation.

4 PROPOSED STRATEGY

4.1 Adaptive Probabilistic Awarding

The experiments discussed in the previous section indicate that the fluctuation factor should adaptively be controlled according to the system's task loads to utilize the capabilities of a MMAS. However, it is impossible to assess the system's task load, because this is a kind of non-local information. Instead, (Sugawara et al., 2008a) estimated the task load of the MMAS from the queue length of contractors. However, this cannot be simply applied to our case, because if the queue is long but the costs of queuing tasks are small, agents cannot conclude whether the system is busy.

[3]The overall performance varied depending on this number and was mostly optimal when it was 20. The details have been reported in (Sugawara et al., 2007).

Our idea to this issue is to estimate situations by statistically analyzing the bid values from local contractors. More precisely, we used the differences between the standard deviations (SDs) of bid values for different tasks that had different costs. Assume that, for announced task t, manager m received bids whose values are $B_m(t) = \{b_1(t), b_2(t), \dots\}$, and the SD of $B_m(t)$ is denoted by $SD_m(t)$. Let $D_m^{SD}(\mathcal{T})$ be $|SD_m(t_1) - SD_m(t_2)|$, when $\mathcal{T} = \{t_1, t_2\}$. Figure 2 shows how the average values and the standard deviation of $D_{SD}(\mathcal{T})$ vary over time.

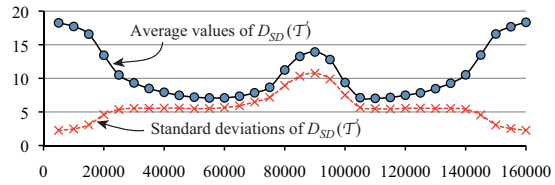

Figure 2: Average values and SDs of $D_m^{SD}(\mathcal{T})$ over time.

Comparing Figs. 1 and 2, $D_m^{SD}(\mathcal{T})$ can be used as the key to determining the degree of fluctuation; more precisely, the fluctuation factor k is established by using the following strategy,

$$\begin{array}{lll} k = \infty & \text{if} & D_m^{SD}(\mathcal{T}) \geq 12.0, \\ k = 6 & \text{if} & 12.0 > D_m^{SD}(\mathcal{T}) \geq 8.8, \\ k = 3 & \text{if} & D_m^{SD}(\mathcal{T}) < 8.8. \end{array} \quad (3)$$

This is called the *variable award strategy* and is denoted by *VAS* in this paper. The aim of this strategy is to combine the best in strategies, PAS_∞, PAS_3, and PAS_6.

The results of performance for VAS, $I_{CNP}(\text{VAS})$, are also indicated by the curve labeled "VAS" in Fig. 1. All the curves in the figure clearly indicate that VAS can usually provide better overall performance than other individual strategies. The improvement ratios are particularly large just before the task load reaches the theoretical limit of MMAS and right after the contractors overcome the overload caused by the huge number of queuing tasks. We believe that this characteristic is important and will be discussed in Section 4.4.

4.2 Learning Probabilistic Awarding

Although strategy VAS can provide better performance to an MMAS, it assumes a number of fixed threshold values as shown in Eqs. (3). We propose that agents learn these threshold values for applying it to other tasks as follows: First, manager m first calculates the SDs of bid values for each $t_i \in \mathcal{T}$ and the maximum difference between these SDs. This is denoted by $D_m^{SD}(\mathcal{T})$. Manager m also retains the maximum and minimum values of $D_m^{SD}(\mathcal{T})$ (denoted by *maxSDdiff*, and *minSDdiff*), thus far. Then, m estimates the current task load using *maxSDdiff*, *minSDdiff*, and $D_m^{SD}(\mathcal{T})$. We will call the award strategy according to this algorithm the *adaptive probabilistic awarding strategy*, or *AAS* after this.

We will investigate whether AAS can provide the good performance comparable to VAS for task $\mathcal{T} = \{t_1, t_2\}$ and whether it also provides the acceptable overall performances for other tasks that have different task structures.

Improvement ratios $I_{CNP}(\text{AAS})$ over time is plotted in Fig. 3. The duration of this experiment was double that of the previous experiment accomplished by repeating it twice because agents had to learn the maximum and minimum differences between the SDs of bid values for individual subtasks. The changes in task loads are also illustrated as gradated javelins. Improvement ratios $I_{CNP}(\text{PAS}_6)$ and $I_{CNP}(\text{VAS})$ have also been shown as benchmarks. Figure 3 indicates that AAS can performs as efficiently as VAS. Note that the performance of AAS is slightly lower than VAS only in the beginning (from 0-K to 20-K ticks), because the learning of threshold values Th_1 and Th_2 is not sufficient.

4.3 Applying Strategy to Other Tasks

Finally, we have to show whether or not the proposed strategy can provide the better performance for tasks with other cost structures. The AAS strategy relies on the difference of costs of subtasks, we examined the case when a task has the different cost ratio, other than 2500:500. Instead, we set the sum of the costs of these tasks to 3000, in order to standardize the theoretical upper limit number of task executions by all agents.

Let us denote the costs of subtasks as superscripts. For example $\mathcal{T}^{25-3-2} = \{t_1^a, t_2^a, t_3^a\}$ means $(\gamma(t_1^a), \gamma(t_2^a), \gamma(t_3^a)) = (2500, 300, 200)$. Thus, the task used in the previous experiments is denoted by $\mathcal{T}^{25-5}$. The results are plotted in Fig. 4. Note that, because we intended to compare their performance under AAS and PAS_∞, we fixed the changes in task loads over time in those experiments. Figure 4 indicates that the overall performance for $\mathcal{T}^{25-3-2}$ $\mathcal{T}^{20-10}$ $\mathcal{T}^{18-12}$ $\mathcal{T}^{20-8-2}$ and $\mathcal{T}^{15-8-5-2}$ under AAS are generally better than those under PAS_∞. Of course, VAS is not applicable to these tasks that have different task structures.

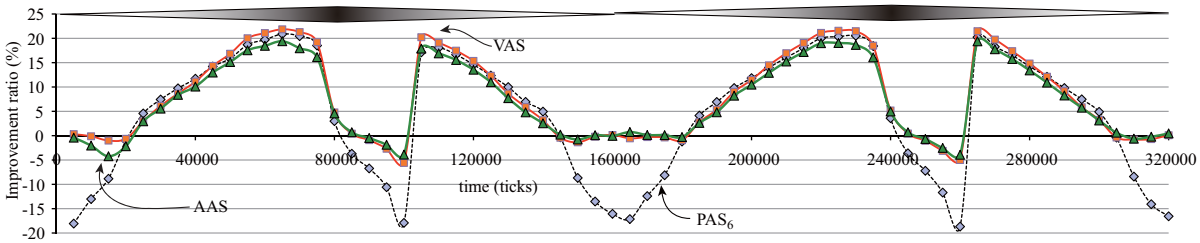

Figure 3: Improvement ratios of AAS compared with PAS_6 and VAS over time.

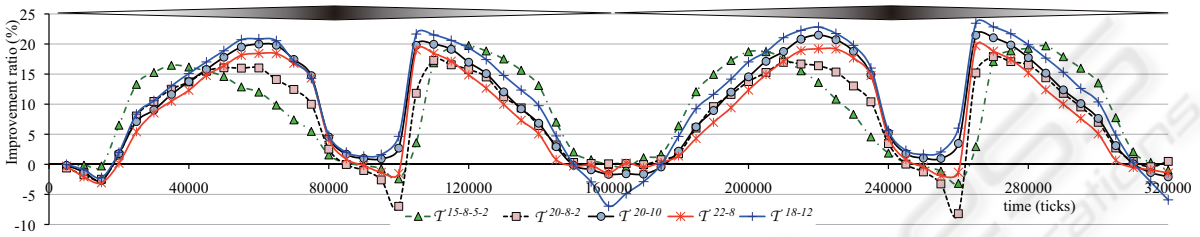

Figure 4: Improvement ratios of AAS for various tasks.

4.4 Cases for Simpler Tasks

We have to discuss two simpler cases in which (1) the costs of all subtasks are quite similar and (2) each subtask is allocated to one of disjoint sets of contractors. In these cases, we believe that the simple extension of (Sugawara et al., 2008a; Sugawara et al., 2008b) can be applied, because the queue length reflects the task loads (or more precisely, multiple awarding because of task congestion) in these cases. It is also clear that when a task cannot be divided into subtasks, the method in (Sugawara et al., 2008a; Sugawara et al., 2008b) is applicable without any extension.

sectionDISCUSSION The improvement in the proposed algorithm was maximum just before the task load reached the theoretical upper limit and right after the contractors overcame the overload caused by the huge number of queuing tasks. These corresponded to situations during 50 to 75-K ticks, 105 to 120-K ticks, 210 to 235-K ticks and 265 to 280-K ticks in Figs. 3 and 4. We want to emphasize that this is one of its quite important characteristics; an MMAS should perform at its full potential near the theoretical limit. If the task load is quite low, any task allocation strategy can provide satisfactory services. However, if it is extremely heavy and over the theoretical limit, no strategy can accomplish acceptable performance. In other situations, the system must yield maximum performance. Our experimental results revealed that the proposed strategy can provide excellent improvements in these situations.

Note that the centers of the curves in Figs. 3 and 4 are slightly shifted to left. Moreover, if we look at Fig. 4 more carefully, the values of improvement ratios at around 100-K ticks and 260-K ticks peaked on the upper of lower direction (but these are excessive values). While the task load was over the cumulative capabilities of MMAS, their queues became longer. After the extremely busy situation, their lengths were getting shorter and they returned to unbusy states. Of course, the better strategy can execute tasks more efficiently, thus can enable all agents to return to unbusy states a little earlier. This results in the peaks to lower direction but their differences is not significant because the PAS_∞ is better when the task load is the extremely busy and under *AAS*, all managers intended to adopt the same strategy.

As the communications link-bandwidth of the Internet is getting increasingly broader, the bottleneck in network computing has shifted to nodes (hosts) in which many tasks are done. Here, it is important that tasks should be distributed to appropriate servers that can complete them in the shortest time. Communication delay, on the other hand, decreases and becomes insignificant. However, our experiments showed that if many managers determine the contractors according to bids from local contractors, a high-performance contractor is likely to be selected as the best bidder; thus, tasks are concentrated on this contractor. Therefore, some degree of fluctuation has beneficial effects on enabling this concentration to be avoided. If we more carefully consider the reason for the concentration, it must be multiple awarding due to the small communication delay. As the network broadens, the delay decreases, but we cannot eliminate the latency caused by the finite speed of light and the finite pro-

cessing speed of the switching fabric. Our experiments suggest that this small delay significantly affects the overall performance of busy MMASs. Recent research has focused on overlay networks that reflect the application-layer links between agents and ignore the physical network topology. However, this is insufficient in an MMAS to elicit agents' capabilities. It is necessary to take into account the physical topology in designing an MAS to minimize interference among agents.

Although the proposed method can provide better performance than the naive CNP, it is possible that more tailored controls can improve more for complicated tasks. In addition, from Fig. 4, the duration in which the improvement ratios $I_{CNP}(AAS)$ are large moves to slightly less busier situations if $|\mathcal{T}|$ is larger. We believe that this is caused by the increased chance of multiple awards. Our experiments show that CNP in an MMAS exhibits quite different features from CNP in a small-scale MAS and thus a simple strategy like the naive CNP cannot elicit the potential capability of MMASs. More research is needed for this purpose; this paper represents the first step toward this direction.

5 CONCLUSIONS

We proposed a probabilistic award strategy in CNP for a massively MAS to elicit the potential capabilities of all agents. In this strategy, a manager agent (a) announces subtasks, (b) statistically analyzes the bids for each of these, (c) estimates the current local task load, and (d) introduces an adaptive degree of fluctuation in the award phase. We then experimentally demonstrated that this strategy provides considerably better performance than the naive CNP.

In this paper, we focused on CNP because it is the well-known protocol, but CNP is not the only approach to task allocation. It is necessary to investigate other protocols (with some modification) or create a new protocol for busy MMAS. This is our future research. Meanwhile, when there are many agents, it is crucial to consider the inter-agent structure like (Gaston and desJardins, 2005; Abdallah and Lesser, 2007). Thus, to investigate the relationship between network structures and the performance of the CNP in MMASs is also another future research issue.

REFERENCES

Abdallah, S. and Lesser, V. (2007). Multiagent Reinforcement Learning and Self-Organization in a Network of Agents. In *Proceedings of the Sixth International Joint Conference on Autonomous Agents and Multi-Agent Systems*, pages 172–179, Honolulu. IFAAMAS.

Foster, I. (2002). What is the grid? a three point checklist. *Grid Today*, 1(6).

Gaston, M. E. and desJardins, M. (2005). Agent-organized networks for dynamic team formation. In *Proceedings of 4th Int. Joint Conf. on Autonomous Agents and Multiagent Systems (AAMAS2005)*, pages 230–237.

Gu, C. and Ishida, T. (1996). Analyzing the Social Behavior of Contract Net Protocol . In de Velde, W. V. and Perram, J. W., editors, *Proceedings of 7th European Workshop on Modelling Autonomous Agents in a Multi-Agent World (MAAMAW 96), LNAI 1038*, pages 116 – 127. Springer-Verlag.

Sandholm, T. (1993). An Implementation of the Contract Net Protocol Based on Marginal Cost Calculations. In *Proceedings of the Eleventh National Conference on Artificial Intelligence*, pages 256–262.

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

Sugawara, T., Hirotsu, T., Kurihara, S., and Fukuda, K. (2007). Performance Variation Due to Interference Among a Large Number of Self-Interested Agents. In *Proceedings of 2007 IEEE Congress on Evolutionary Computation*, pages 766–773.

Sugawara, T., Hirotsu, T., Kurihara, S., and Fukuda, K. (2008a). Adaptive Manager-side Control Policy in Contract Net Protocol for Massively Multi-Agent Systems. In *Proceedings of 7th Int. Joint Conf. on Autonomous Agents and Multiagent Systems (AAMAS2008)*, pages 1433–1436. IFMAS.

Sugawara, T., Hirotsu, T., Kurihara, S., and Fukuda, K. (2008b). "controling contract net protocol by local observation for large-scale multi-agent systems". In *Cooperative Information Agents XII (CIA2008)*, pages 206–220. LNCS 5180, Springer.

Weyns, D., Boucké, N., and Holvoet, T. (2006). Gradient Field-Based Task Assignment in an AGV Transportation System. In *Proceedings of 5th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS2006)*, pages 842 – 849.

Xu, L. and Weigand, H. (2001). The Evolution of the Contract Net Protocol. In *Proceedings of WAIM 2001*, volume LNCS 2118, pages 257–264.