

Forming Resource-Sharing Coalitions: A Distributed Resource Allocation Mechanism for Self-Interested Agents in Computational Grids

Linli He

Department of Computer Science
Texas A&M University
College Station, TX77843-3112
linli@cs.tamu.edu

Thomas R. Ioerger

Department of Computer Science
Texas A&M University
College Station, TX77843-3112
ioerger@cs.tamu.edu

ABSTRACT

Designing efficient resource allocation mechanism for computational grids is extremely challenging because the effective agents in computational grids are inherently self-interested due to their different ownerships. Providing incentive for agents to share their resource with others is the key to make computational grids realistic. The global efficiency should be generated through the interactions among agents from the bottom up. In game theory, forming coalition is such a cooperative game among self-interested agents. We develop a distributed resource allocation mechanism for computational grids by forming resource-sharing coalitions among self-interested agents through automated multi-party negotiation. This mechanism is based on a task-oriented mechanism for measuring the economic value of computational resource usage. The simulation results show that the self-interests of agents in computational grids have considerable impact on the decisions of each agent about how to allocate their resource to appropriate tasks.

General Terms

Algorithms, Design, Measurement, Economics.

Keywords

Self-interested agent, Coalition formation, Computational grids

1. INTRODUCTION

The most challenging issue of designing resource allocation mechanism for computational grids is that the effective agents (both resource users and suppliers) are inherently self-interested because of their different ownerships [1,2,4,22,23]. Self-interested agents make their own decisions according to their budgets, capabilities, goals and local knowledge, without considering the global good [8,13,14,15,17]. Providing incentive for those self-interested agents to participate in a computational grid is the key to make computational grids feasible for all computational resource users, rather than simply remaining a research tool supported by government.

Traditionally, both centralized and decentralized resource allocation mechanisms for computational platforms have been

constructed from the top down, namely, fixed resource allocation decision rules are imposed to handle all possible situations in resource allocation for that platform. This design philosophy does not work well in computational grids, because there does not exist an omniscient designer who can develop a resource allocation mechanism that satisfies the preferences of all self-interested resource users and suppliers and maximizes the global efficiency of a grid as a computational platform. Instead, resource allocation mechanisms should be established from the bottom up, meaning that every resource user or supplier makes individual decisions based on local knowledge and preferences without considering the global good. The global efficiency is generated from the bottom up through interactions among agents [21].

The economic-based approach [2,22,23] is a promising avenue for building such a distributed resource allocation mechanism for computational grids. Wolski et al [22,23] and Buyya [2] pioneered investigation of commodity market-based resource allocation mechanisms in computational grids. However, the most obvious weakness of the commodity market-based mechanism for resource allocation in computational grids is that no such real market exists currently. A difficulty lies in verifying that the empirical results in experimental settings can be duplicated in real markets [22,23].

The most important purpose of building computational grids is to establish resource-sharing mechanisms that allow self-interested agents to complete computational tasks cooperatively so that the performance requirements of all computational tasks can be satisfied without individually increasing the amount of resources. Here, a basic assumption is that agents in a computational grid should have peak workloads at different times so that they can utilize others' resources at idle time. That is, a dynamic load balancing [10] among self-interested agents. Agents share their resources among partners. How could such partnerships be established?

How to provide incentive for self-interested agents to contribute their resource to a grid is a highly non-trivial problem. How to decide the amount of resource each agent should contribute to the grid and the amount of resource each agent can utilize are both extremely complicated issues. In game theory, coalition formation is such a cooperative game among self-interested agents [8,17]. In this paper, we develop a resource allocation mechanism for self-interested agents in computational grids by forming resource-sharing coalitions.

The rest of this paper is organized as follows: Section 2 presents the general concepts of coalition formation in game theory and describes how it fits in the architecture of the current computational grids. In Section 3, we establish a task-oriented mechanism for measuring the economic value of resource usage in computational grids. We evaluate resource-sharing coalition

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

SAC'05, March 13-17, 2005, Santa Fe, New Mexico, USA.

Copyright 2005 ACM 1-58113-964-0/05/0003...\$5.00.

formation based on this mechanism later. Section 4 formalizes the problems of forming resource-sharing coalitions in computational grids and addresses the issues necessary for solving those problems. Section 4 also gives a distributed coalition formation mechanism through automated negotiation, which we develop for investigating resource-sharing coalition formation in computational grids. Section 5 shows the simulation results. The paper ends by giving conclusions and discussing the future work.

2. COALITION FORMATION

In the real world, self-interested agents (e.g. commercial companies) can lower their costs by coordinating their activities with others [8,13,15,17]. Coalition formation has been widely studied in many different domains [3,7,11,12,17,20]. Such coordination raises questions: What coalition should the agents form? Are the coalitions stable? And how should costs and benefit be divided among agents within each coalition? [17] To our knowledge, no such study of resource-sharing coalition formation for computational grids has been undertaken so far.

Indeed, real organizations have already started to negotiate with each other to share their computational resources to execute large computational tasks. Human subjects execute all negotiation processes to establish certain resource sharing mechanisms currently. However, these negotiation processes only can be run to initialize a computational grid. Afterwards, if there are internal or external changes (e.g. software upgrades) for the grid, the resource allocation mechanisms should be changed by resetting some other negotiation processes. Furthermore, negotiations among human subjects cannot be at task level because it is impossible for human subjects to negotiate every computational task. Hence, it is impractical to build efficient and adaptive resource allocation mechanisms for computational grids through negotiations among human subjects. Distributed coalition formation [7,20] mechanisms through automated negotiation [9,14] need to be established for resource allocation in computational grids.

Coalition formation includes three activities [17]. The first is coalition structure generation, that is, formation of coalitions by the agents such that agents within each coalition coordinate their activities [16]. This means partitioning the set of agents into disjoint coalitions. In the computational grid settings, this activity is for agents to find appropriate partners and make resource-sharing agreement. The second is solving the optimization problems within each coalition. For computational grids, this involves deciding how to distribute the computational tasks of the coalition among the member agents and solving the optimization problems of each agent, given its resources and the tasks distributed to it. The third activity involves payoff division [3,8,12,17]. The value of joining a resource-sharing coalition for an agent is the difference between the cost of executing its tasks within the coalition and the cost of doing so without joining the coalition. Agents in a coalition must agree on the value that each of them obtains from joining the coalition.

These three activities in coalition formation interact with each other. An agent decides to join a coalition by comparing the benefit it would gain from each potential coalition. The value is also related to the solutions of the optimization problems within each coalition. Forming optimal coalitions is an extremely complex task [16,17].

In the computational grid setting, other problems arise during the coalition formation process. The effective agents dynamically

appear in computational grids and are distributed geographically. Who are the potential agents willing to form resource-sharing coalitions? This involves the issues about dynamic coalition formation. Who will be in charge of forming resource-sharing coalitions? This involves the issues about distributed coalition formation. How do agents negotiate with each other in order to make an agreement? This needs to establish a multi-party negotiation protocol. How do agents measure the cost of executing computational tasks with different performance requirements by using heterogeneous resources? This issue is critical for agents to evaluate their decisions to participate in resource-sharing coalitions. In this paper, we investigate these issues by establishing a distributed resource-sharing mechanism for computational grids through automated negotiation among self-interested agents.

One of the most critical preconditions of building an economic-based computational resource allocation mechanism is establishing a common currency for measuring the economic value of resource usage [1,2,22,23]. Without this measurement, agents cannot evaluate their decisions on whether to join a resource-sharing coalition or not.

3. VALUE OF RESOURCE USAGE

Previously, people assumed that a general currency existed (denoted by grid dollars [2,22,23]) for measuring the cost of using a certain resource. The problem is how to map the value of using different type of resources to grid dollars. This gap keeps computational grids from being realistic because it is difficult to convince users that participating in a computational grid is less expensive than purchasing more computational resources when obtaining the same amount of computational power for their computational tasks. Also, joining a grid will incur more security and maintenance cost than using only their own computational resources to execute their own tasks. They have to be convinced that the extra costs are worthwhile.

Currently, the price settings are based on computational units of resource usage in the market-based resource allocation strategies for computational grids [2,22,23]. Based on this price setting mechanism, in order to allocate their tasks to appropriate resource suppliers, users must be aware of the amounts and types of resource units they need for a certain task to calculate the cost and stipulate the performance.

However, unlike traditional computational platforms, even the same types of resources in computational grids vary over a wide range of capabilities. The tasks in computational grids usually need intensive resources that have different capabilities. A typical example is a task that involves a large number of CPUs with different speeds. It is not appropriate to measure the CPU usage only according to the number of time slots required, because the computational capabilities of the time slots of different CPUs used could be different. The prices should not only reflect the time slots but also the differentiation of physical capabilities of the CPUs. The above mechanism is too complicated to measure users' different performance preferences.

Thus, a new mechanism for measuring the economic value of resource usage needs to be developed to encapsulate the differences in physical capabilities of different types of resources and different user preferences. In this paper, we propose a task-oriented mechanism for measuring the value of resource usage in computational grids.

3.1 An Observation

Suppose there are three processors P_1 , P_2 , and P_3 , which have different speeds (Here, we do not specify what exact meaning of the speed of a CPU. It could be measured by MIPS, clock rates, or any other kind of standard units) from the highest to the lowest respectively. Given an identical job, these three processors would finish it in different amounts of time. Figure 1 shows the performance of each processor (We assume all other conditions are same, e.g. same amount of RAM associating with each processor.). The equivalent performance line depicts the fact that these three processors P_1 , P_2 , and P_3 finish an identical job within H_1 , H_2 , and H_3 CPU hours respectively.

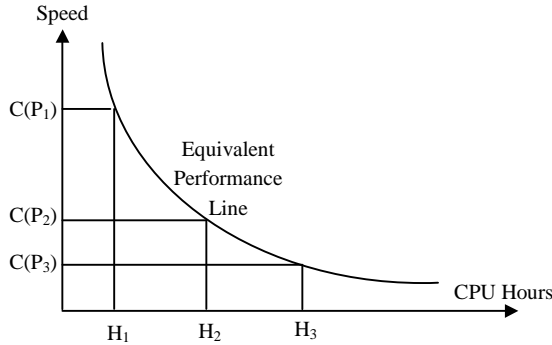


Figure 1: Equivalent Performance by Different Processors

If a resource user gives the same job to P_1 , P_2 , and P_3 , how should the processor owner charge the user for using different processors? If all processors can satisfy the deadline of a job, the resource user would prefer not to pay extra for using P_1 . But if the deadline of the job is tight, he may be willing to pay more for using P_1 . Therefore, in order to set proper prices for using P_1 , P_2 , and P_3 to execute an identical job, the resource suppliers need to consider both the different capabilities of the processors and the users' performance preferences.

3.2 Modeling Resource Capabilities

In this paper, we only consider the time constraints as the performance requirements for modeling the capabilities of computational resources (e.g. CPU, storage, bandwidth etc.) in grids. We can define the capability of a group of heterogeneous resources is the following:

Definition 1: Given a set of task $K = \{k_1, \dots, k_n\}$ with duration D and a group of resources $G = \{R_1, \dots, R_m\}$. R_i s in G could be heterogeneous. The capabilities of a group of resources in G for executing the tasks in K is denoted by $\text{CanGroup}(G, K, D)$. $\text{CanGroup}(G, K, D)$ is true if and only if the resources in G can finish all k_j within D .

Definition 1 implicitly indicates that there should be some scheduling algorithms for heterogeneous resources, which can schedule the tasks in K properly so that the resources in G can accomplish them within duration D . The quality of a scheduling algorithm has a strong impact on the capabilities of a group of resources [6], but all these scheduling algorithms should be hidden from users. This definition encapsulates the physical differentiation of resources to users. They allow users to ignore the different physical capabilities of resources and only consider their performance preferences.

3.3 Economic Value of Resource Usage

From a user's perspective, regardless of the type of resources are provided to execute a task, the economic values of using these resources are equivalent if they can accomplish the task while satisfying the same performance requirements. The following claim addresses the equivalent value of using two groups of resources:

Claim 1: Given a set of tasks $K = \{k_1, \dots, k_n\}$ with duration D and two group of resources $G_1 = \{R_{11}, \dots, R_{m1}\}$ and $G_2 = \{R_{11}, \dots, R_{r1}\}$. The resources in G_1 and G_2 could be heterogeneous. The economic value of using G_1 to execute the tasks in K is denoted by $V(K, D, G_1)$. We say $V(K, D, G_1) = V(K, D, G_2)$ if and only if both $\text{CanGroup}(G_1, K, D)$ and $\text{CanGroup}(G_2, K, D)$ are true.

Note that no agent in a computational grid has the super power to set the true economic value of using a resource. The value should be decided by the interaction among all agents (both the resource users and suppliers) in the grids. Whether the established value for the usage of a resource is stable depends on the relationship between the supply of resources and the demand [22,23]. The value of using a resource is not the price of using the resource, but a fair price should reflect the real value of using the resource.

Based upon the above analyses, any mechanism for measuring the economic value of resource usage in computational grids should obey the following principles:

- The economic value of using a resource is evaluated by the task executed by the resource.
- The values of using two groups of resources are equal if they can accomplish identical tasks while satisfying the same performance requirements.
- The real economic value of using a resource is established through the interactions among agents in computational grids.
- The established economic value of using a resource is stable if the relationship between the amount of resource demanded and supplied is also stable.

3.4 Task-Oriented Mechanism for Measuring the Economic Value of Resource Usage

Based upon the above capability model of the heterogeneous resources in grids, we establish a new mechanism for measuring the economic value of resource usage. In this paper, the economic value refers to the true value of a commodity that can generally be accepted by all agents in an economic system. Note that the economic value of a resource usage is not the value of a resource itself but the value of using the resource.

We use CPU as an example to illustrate our mechanism. Referring to Figure 1, three processors with different speeds deal with an identical task. We say the usage of each processor for the task is the same. Mathematically, "the usage of each processor" here refers to the area of each rectangle in Figure 1. The formal definition of the usage of a processor to execute a computational task is given as follows:

Definition 2: Given a task k and a processor P , the speed of P is $C(P)$. P needs H hours to finish k . The usage $S(k, P)$ of P for executing k is the following:

$$S(k, D, P) = C(P) \times H \quad (3-1)$$

This definition reflects the amount of processor usage to execute a task no matter what kind of processors are used. The following claim is obviously true:

Claim 2: $S(k, D, P_x) = S(k, D, P_y)$ for given P_x and P_y which are two processors with different speeds.

Based on this claim, we can establish a mechanism to translate the usage of CPUs with different speeds to a common measurement. The idea is to establish a standard speed and convert real CPU usage to the usage of a virtual CPU with the standard speed. Hence there are two directions to convert the CPU usage of a task, one is changing the duration and the other is changing number of CPUs with the standard speed.

Users in computational grids generally expect to finish their tasks as soon as possible. Given a task with certain duration, we can measure the usage of CPU for executing the task by calculating how many standard CPUs should be used to execute the task while satisfying the time constraints given by the user. We also define a standard time unit to measure the expected duration of a task. Thus, the definition of the usage of processors to execute a computational task is modified as follows:

Definition 3: Given a task k with a certain expected duration D , the standard CPU speed is $C(P_s)$ and the standard time unit is D_s . $D = m \times D_s$. In order to finish k within D , there should be n processors with speed $C(P_s)$ working simultaneously (assume k can be divided into n subtasks evenly) or a processor with speed $n \times C(P_s)$. The usage $S(k, n, P)$ of CPUs for executing k is the following:

$$S(k, D) = n \times C(P_s) \times m \times D_s \quad (3-2)$$

This definition implies that the CPU usage of any task can be measured through a standard speed and a standard time unit. The standard unit of CPU usage is given by equation 3-3:

$$S_s = C(P_s) \times D_s \quad (3-3)$$

Therefore, the CPU usage for executing a task can be measured through S_s by changing the number of CPUs with the standard speed or the number of the standard time units to finish a task. Thus equation 3-2 becomes:

$$S(k, D) = n \times m \times S_s \quad (3-4)$$

If the economic value of S_s is V_s , it is easy to calculate the corresponding economic value $V(k, D)$ of CPU usage for executing the task.

$$V(k, D) = n \times m \times V_s \quad (3-5)$$

However, equation 3-5 does not reflect the common sense that the CPU usage for executing a task in a shorter duration might have higher value than the one for executing the same task with a longer duration. In order to catch this fact, we modify equation 3-5 as follows:

$$V(k, D) = (n + \lambda_1 n_0) \times (m + \lambda_2 m_0) \times V_s \quad (3-6)$$

Where, n_0 and m_0 refer to the increasing number of CPUs with the standard speed and the number of the standard time units respectively. n_0 and m_0 can be negative. The coefficients λ_1 and λ_2 in equation 3-6 imply that changing the number of CPUs with standard speed and the number of the standard time units to finish the same task could result in different economic values of CPU usage. If only either m_0 or n_0 is equal to 0, then we can have:

$$V(k, D) = nmV_s + \lambda_1 m n_0 V_s \quad (3-7)$$

$$\text{or} \quad V(k, D) = nmV_s + \lambda_2 n m_0 V_s \quad (3-8)$$

If $\lambda_1 \neq \lambda_2$, then changing the number of CPUs with standard speed and the number of the standard time units to finish a same task causes different economic values of CPU usage.

We have now established a task-oriented mechanism for measuring the economic value of CPU usage for executing a task in a computational grid. It can be extend to a group of heterogeneous resources through defining a standard

computational unit based on the definition of the capability of a group of heterogeneous resources. We leave this for the future work.

4. FORMING COALITIONS

Game theorists did not provide algorithms for forming coalitions but establish a bunch of solution concepts for evaluating coalition games [8,17]. Researchers in multi-agent systems society have been developing algorithms for forming buyer coalitions in electronic markets so that buyers can obtain greater discount from sellers without purchasing more than they really want to buy [7,11,12]. As mentioned in Section 2, in order to form resource-sharing coalitions in computational grids, three activities should be undertaken: generating coalition structure, solving the optimization problem within each coalition and dividing payoff within each coalition.

4.1. Problem Formalization

Agents in the grid should have peak workloads at different times so that they can utilize others' resources at their idle time. Thus, we need to define the relationship between the workload of each agent and time. We call it workload distribution function[6].

4.1.1. Workload Distribution Function Using CPUs as the resource example, based on the mechanism of measuring resource usage in Section 3, we can measure the computational capability of an agent by converting all its CPUs with different speeds to the CPUs with the standard speed. The entire CPU capacity of a group of CPUs is defined as follows:

Definition 4: Given a group of processors $U = \{P_1, \dots, P_m\}$, different P_i in U could have different speeds. The standard CPU speed is $C(P_s)$. The speed $C(P_i)$ of P_i is equal to $n_i \times C(P_s)$. The entire CPU capacity $C(U)$ of a group of processors in U within time interval $[t_1, t_2]$ is:

$$C(U) = N \times C(P_s) \times |t_1 - t_2| \quad (4-1)$$

where $N = \sum_{i=1}^m n_i$. Hence, the workload of an agent at a certain

time point is decided by how many standard CPU P_s s are needed to run its tasks at that time.

Definition 5: The workload of agent A at certain time point t is defined as $M \times C(P_s)$, if the agent needs M standard CPU P_s s to run its tasks at time t .

Definition 6: The workload distribution function is defined as a function of time $w(t): T \rightarrow R$, which is the workload of the agent at time t (T is the set of time points).

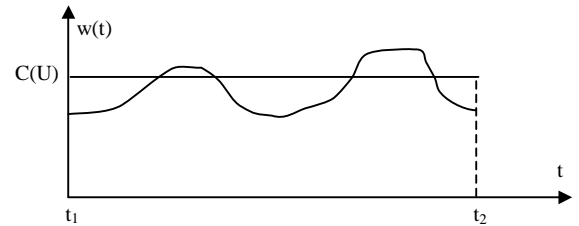


Figure 2: Workload Distribution Function

The workload at a certain time in a workload distribution function can be larger than the entire computational capability of an agent because the tasks require more resources. Figure 2 shows the example of workload distribution function. To evaluate the cooperation among agents, we also need to define an idle resource distribution function.

Definition 7: Given a workload distribution function $w(t)$ of agent a within time interval $[t_1, t_2]$ and the entire CPU capacity of a is $C(U)$. The idle CPU distribution function $g(t)$ is defined as:

$$g(t) = C(U) - w(t), t \in [t_1, t_2] \quad (4-2)$$

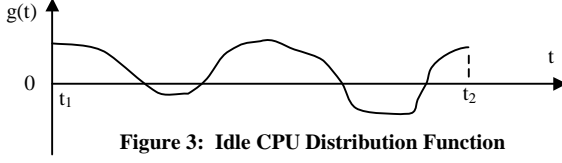


Figure 3: Idle CPU Distribution Function

Figure 3 shows an example of $g(t)$ if $w(t)$ is given by Figure 2. When $g(t) > 0$, the agent has idle CPUs. When $g(t) < 0$, the agent is overloaded. Agents are self-interested in computational grids. Considering only the idle resource distribution is not enough to form a resource-sharing coalition. Each agent needs to consider the economic value of contributing its idle resource to others and the cost of using others' idle resource.

Definition 8: Given the idle CPU distribution function $g(t)$ of agent a within time interval $[t_1, t_2]$, the amount C_R of CPU capacity required for agent A is:

$$C_R(a, g(t)) = \int_{t_1}^{t_2} g^-(t) \quad (4-3)$$

where, $g^-(t)$ is equal to $g(t)$ when $g(t) < 0$ and $g^-(t)$ is equal to 0 otherwise. The economic value V_R of using others' idle CPUs to finish agent A 's tasks is:

$$V_R(a, g(t)) = \frac{C_R(a, g(t)) \times V_s}{S_s} \quad (4-4)$$

Definition 9: Given the idle CPU distribution function of agent A within time interval $[t_1, t_2]$, the amount C_I of CPU capacity of agent A is idle:

$$C_I(a, g(t)) = \int_{t_1}^{t_2} g^+(t) \quad (4-5)$$

where, $g^+(t)$ is equal to $g(t)$ when $g(t) > 0$ and $g^+(t)$ is equal to 0 otherwise. The economic value V_I of agent A 's idle CPUs is:

$$V_I(a, g(t)) = \frac{C_I(a, g(t)) \times V_s}{S_s} \quad (4-6)$$

4.1.2. Resource-Sharing Coalition Formation. From an economic perspective, forming resource-sharing coalitions in computational grids is multi-party bartering. Each agent tries to use its idle resource capacity to exchange for others' idle resource capacities. Agents try to obtain as much resource capacity as possible from the grid by contributing its own idle resource capacity to the grid.

The formal problem definition of forming resource-sharing coalitions in a computational grid is as follows:

Definition 10: Given a set of all agents in a computational grid, $A = \{a_1, a_2, \dots, a_r\}$. Each agent a_i also has its CPU capacity $N \times C(P_s) \times [t_1, t_2]$ within time interval $[t_1, t_2]$. Each agent a_i needs to run a set of tasks $K_i = \{k_{i1}, k_{i2}, \dots, k_{il}\}$ which generates its idle resource distribution function $g_i(t)$ within $[t_1, t_2]$. The objective is to form appropriate coalitions such that each agent a_i can have the optimal idle resource capacity exchanging with other agents in A within time interval $[t_1, t_2]$.

For each agent in A , the best exchange is to contribute minimally of its own idle resource capacity and to obtain the maximum idle resource capacity from a resource sharing coalition while finish as many of its own tasks as possible. To evaluate the optimal idle resource capacity exchanging for each agent, we need to define the utility function of an agent. The utility is composed

of two parts. One is the difference between the resource capacity the agent obtains from the coalition and the one it contributes to the coalition. The other is the economic value of finishing tasks by using the resource capacity it obtains.

Definition 11: Suppose that agent a_i obtains idle CPU capacity $C_{\text{gain}}(a_i, CL_j)$ by joining a resource coalition CL_j and it contributes its own idle CPU capacity $C_{\text{give}}(a_i, CL_j)$. By obtaining $C_{\text{gain}}(a_i, CL_j)$, agent a_i can finish a subset of tasks K_i' in K_i while satisfying the time constraints. The utility $U(a_i, CL_j)$ that agent a_i obtains by joining coalition CL_j is given by:

$$U(a_i, CL_j) = \left(\frac{C_{\text{gain}}(a_i, CL_j) - C_{\text{give}}(a_i, CL_j) + S(K_i')}{S_s} \right) \times V_s \quad (4-7)$$

where $S(K_i')$ is the amount of CPU capacity that agent a_i obtaining from CL_j for finishing tasks in K_i' . Each agent a_i in A uses this utility function to evaluate its decision on which coalitions in the grid it should join by maximizing its utility.

4.1.3. Definition of Terms. We give the formal definition of terms related to forming resource-sharing coalitions in computational grids as follows:

Resource-Sharing Coalition: A resource-sharing coalition (CL) is defined as a subset of A . Each member in CL should contribute part of its idle resource capacity to the CL and it can use idle resource capacity contributed by other members CL.

Coalition Structure: A coalition structure (CS) is defined as a partition of the agents in A into disjoint coalitions in previous work [17].

Coalition Value: The coalition value CV of a coalition CL is defined as the sum of the utilities that all members obtain through joining the coalition.

Value of Coalition Structure: The value of a coalition structure CS is defined as the sum of the values of all coalitions in the coalition structure. The value of the coalition structure is equal to the sum of utilities of all agents in A . This term is used to evaluate the social welfare of the computational grid.

4.1.4. Assumptions.

- Agents' decisions about whether joining a coalition or not depend on whether they can maximize their own individual utilities.
- Agents know the idle resource distribution functions of other agents in A before they evaluate their decisions (Agents can obtain this information from some grid information service providers [1,5]).
- Agents do not avoid opting out [9], namely, if the utility derived from joining a coalition and from opting out are the same, they will not join it.
- After a coalition is formed, each member of the coalition must reserve the corresponding idle resource capacity to the coalition.
- Agents use a global clock.

4.2. A Distributed Resource-Sharing Coalition Formation Mechanism Through Negotiation

We develop a distributed resource-sharing coalition formation mechanism for computational grids based on the DCF-EN (which stands for distributed coalition formation through explicit negotiation) mechanism for forming buyer coalitions in electronic markets in [7]. The DCF-EN mechanism is composed of agent strategies for searching out the best coalitions to join and an automated multi-party negotiation mechanism.

4.2.1. Agent Strategies for Finding Appropriate Resource-Sharing Coalitions. In the DCF-EN mechanism, all agents are assumed to take same strategies for finding appropriate resource-sharing coalitions in a computational grid. Every agent greedily searches for the best coalition in a computational grid. The best coalition to an agent is the one that maximizes his own utility with the smallest size.

Before an agent starts a negotiation with others, it needs to calculate the best possible coalitions it can find based on the idle resource distributions functions of available agents in the grid. In practice, immigrating tasks from one agent to another agent in computational grids would cause communication cost. Therefore, an agent tries to find such a coalition that the agent can immigrate as less tasks as possible while maximize its utility by joining it. As a result, each agent prefers to join in coalitions with smaller sizes.

Even though we assume agents can know others' idle resource distribution functions in advance, agents cannot access the local scheduling mechanisms of other agents. Agents have to form a resource-sharing coalition through negotiation. We assume that each agent greedily proposes the coalition with the highest utility it can obtain to the corresponding agents at first. If it cannot get accepted, the agent proposes the second best coalition and so on.

4.2.2. Multi-Party Negotiation Mechanism for the Distributed Coalition Formation Process. There are three main issues in defining a negotiation mechanism: the space of possible deals, the negotiation process, and the negotiation strategy [14].

Space of Possible Coalitions: An agent only considers possible coalitions that include itself. The negotiation space is 2^{R-1} for each agent.

Negotiation Process. Agents negotiate with each other by sending messages (through grid middlewares [1,5]). The negotiation process for an agent handles every message from outside of the agent appropriately. Any decision during a negotiation process is made based on the negotiation strategy of the agent.

Negotiation Strategy. In our current DCF-EN mechanism, all agents use the same negotiation strategies for resource-sharing coalition formation.

- Each agent can propose multiple coalitions without waiting for the confirmation of the coalitions that have been sent out. However, all these coalitions that have been sent out parallel should result in same utility for the agent at any given time. It is like a leveled commitment [18].
- Each agent joins the coalition that is confirmed earliest.
- Each agent greedily accepts the best coalition that it can find.
- Each agent who receives multiple coalition proposals can only accept to one proposal at any given time (no regret decision [9]).
- Agents terminate their negotiation process when coalitions are formed (accepted), failed (refused) or time out.

Agents have no incentive to leave the coalition because they use greedy strategies to search for the best coalition. They always agree the best coalition that they can find first.

Figure 4 gives the agent algorithm in the DCF-EN mechanism for implementing the resource-sharing coalition formation. Due to the space limitation, we cannot show all algorithms for handling different negotiation processes. The detail algorithms and complexity analyses are given in [7]. The simulation results showed in Section 5 are generated through implementing these algorithms.

```

Agent Algorithm {
  Initialize Message Listener;
  While (true) {
    If (the coalition space is empty) {
      Calculate all possible resource-sharing coalitions;
      Process negotiation messages from other agents {
        If (has not found the best coalition) {
          Keep negotiating with other agents based on its strategies;
        }
        Else {Tell others "finish forming coalition in this round";}
      }
    }
    If (has not found the best coalition) {
      If (has not accepted any proposal){
        If (has not sent any proposal
          or proposals sent previously has been refused) {
          If (there exist possible coalitions) {
            Sending the best possible coalition proposals;
          }
          Else {The best coalition is agent self;}
        }
        Else {
          If (there exist possible coalitions) {
            Waiting for acceptance or refusing messages;
          }
          Else {The best coalition is agent self;}
        }
      }
      Else {Waiting for confirmation or proposal failure messages;}
    }
  }
  Else {
    Update the partnership databases;
    Sleep until next round of coalition formation;
  }
}

```

Figure 4: Agent Algorithm in DCF-EN

5. SIMULATIONS

In our current experimental settings, the idle resource distributions for agents are generated randomly in uniform distribution. There are 10 segments per round for the idle resource distribution function of an agent. During each segment, the agent has a constant idle resource capacity. It is not hard to keep this assumption true in real applications by using the lower bound for the idle resource capacity and the upper bound for overload capacity. The number of agents in the experiments is not large, because we are studying agents that represent organizations, which own large-scale computational resources and have computational tasks that require large amount of resources.

By running the experiments based on the DCF-EN mechanism, we want to study the following issues: What is the difference between the desired utility and the real utility that an agent can obtain by joining the best coalition it can find (The desired utility is the one that an agent can obtain by joining the best coalition in the entire possible coalition space.)? How frequently can an agent join the coalition that makes the agent have its desired utility? How much communication load do the multi-party negotiation processes cause during a coalition formation game for each agent? What is the difference between the global resource utilization of a grid composed of self-interested agents and the one of a grid composed of cooperative agents?

Table 1 shows the desired coalition that each agent wants to join and the real coalition structures generated in resource-sharing coalition formation games with 5, 6, 7, 8, 9 and 10 agents. The desired coalitions of different agents are generally conflicted with each other during one game. No coalition formed in our experiments has size larger than 2. The reason is that every agent tried to join smaller size coalitions for reducing the potential cost caused by immigrating tasks.

Table 1: Desired Coalition for Each Agent vs. Real Coalition Structure

Number of Agents		5	6	7	8	9	10
Desired Coalition for Each Agent	Agent0	[0, 3]	[0, 2, 5]	[0, 1, 2]	[0, 2, 3, 7]	[0, 4]	[0, 1]
	Agent1	[0, 1]	[1, 5]	[1, 5, 6]	[1, 4, 6]	[1, 4, 6]	[1, 4]
	Agent2	[1, 2]	[2, 1]	[2, 0, 3]	[2, 7]	[2, 0, 6]	[2, 0, 5, 8]
	Agent3	[2, 0]	[3, 0]	[3, 4]	[3, 2, 6]	[3, 1, 8]	[3, 0, 5, 8]
	Agent4	[4, 1]	[4, 2]	[4, 5]	[4, 0, 2]	[4, 8]	[4, 5, 9]
	Agent5		[5, 3]	[5, 4, 6]	[5, 3, 7]	[5, 6, 8]	[5, 7, 8]
	Agent6			[6, 2, 4]	[6, 0, 1]	[6, 1]	[6, 0, 4, 7]
	Agent7				[7, 0]	[7, 0]	[7, 0, 1]
	Agent8					[8, 4, 5]	[8, 1, 9]
	Agent9						[9, 5, 8]
Real Coalition Structure		[0,] [1,5] [2] [3] [4]	[0] [1] [2,4] [3,5]	[0,3] [2] [4] [5,6]	[0] [1] [2] [3] [4,7] [5] [6]	[0] [1,5] [2,8] [4,7] [6]	[0] [1] [2,7] [3,5] [6,8] [9]

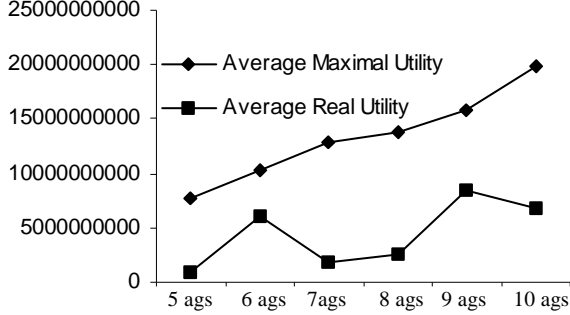


Figure 5: Average Real Utility vs. Average Desired Maximal Utility for Each Agent in Different Number of Agents Resource-Sharing Coalition Formation Games

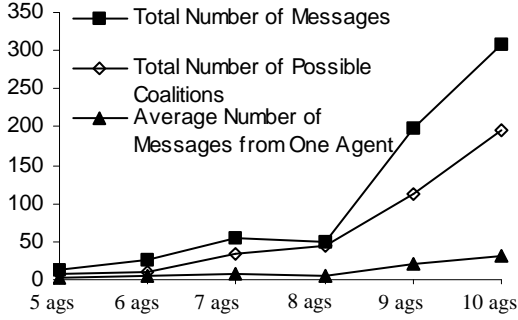


Figure 6: Number of Messages vs. Number of Possible Coalitions for Each Agent in Different Number of Agents Resource-Sharing Coalition Formation Games

Figure 5 shows the comparison between the average desired utility and the average real utility of each agent in games with 5, 6, 7, 8, 9 and 10 agents respectively. The desired utilities increase with the number of agents increasing. But the real utilities do not show this feature.

The space of possible negotiation deals is exponential increasing with the number of agents in a coalition formation game. Figure 6 shows the average total number of messages accepted by each agent increasing with the number of possible coalitions and the number of agents in a game.

Figure 7 depicts the fact that the global resource utilization of a grid composed of self-interested agents is worse than the utilization of a grid composed of cooperative agents, given same idle resource distribution for each agent. Here, a grid composed of cooperative agents refers to a grid in which agents pool all their resource together without considering their own utilities. The grid with self-interested agents has much lower global utilization than

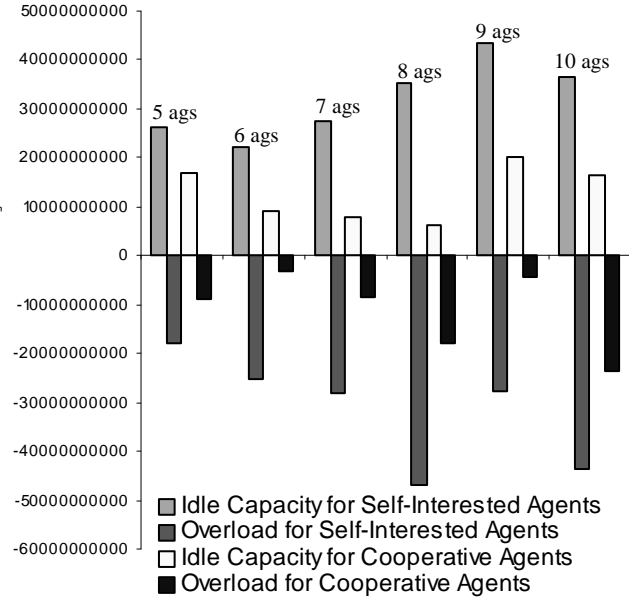


Figure 7: Global Resource Utilization of Self-Interested Agents vs. The One of Cooperative Agents in Different Number of Agents Resource-Sharing Coalition Formation Games

the grid with cooperative agents, when the self-interested agents use strategies in the DCF-EN mechanism.

The above experimental results prove that the self-interests of agents in computational grids have considerable impact on the decisions of each agent about how to allocate resource to appropriate tasks. The resource allocation mechanism for a computational grid must be designed with considering the self-interests of agents.

6. CONCLUSION AND FUTURE WORK

Building computational grids is intended to establish resource-sharing mechanisms through cooperation among self-interested agents so that the performance requirements of all agents can be satisfied without individually increasing the amount of computational resources. In this paper, we introduced Coalition Formation in Game Theory as a new theoretic base for designing resource allocation mechanisms for self-interested agents in computational grids.

Having appropriate mechanism for measuring the economic value of resource usage by a common currency is critical for building an economic-based computational resource sharing mechanism. We proposed a task-oriented mechanism for measuring the value of resource usage in computational grids by converting the heterogeneous resource usages to a standard measurement so that the economic value of using resources can be measured by a common currency (i.e. a standard economic unit).

Based upon this measurement mechanism, we formalized the resource sharing among self-interested agents in computational grids as a problem of forming resource-sharing coalitions. In order to solve this problem, distributed coalition formation mechanisms through multi-party negotiation should be developed. In such a distributed coalition formation mechanism, every agent has its own searching strategies for finding the best coalition with maximizing its utility. Every agent has its own negotiation strategies to make an agreement on desired coalitions. The experimental results show that the self-interests of agents have considerable impact on the decisions of each agent about how to allocate resource to appropriate tasks.

To make resource-sharing coalition formation feasible in computational grids, plenty of further investigation needs to be done. We name a few here. How does the local resource allocation mechanism of an agent interact with the resource-sharing coalition formation module? How does the grid middleware interact with the coalition formation module of the agent? How do we evaluate the social welfare of a coalition formation-based resource allocation mechanism mathematically?

Since there exist different economic-based resource allocation mechanisms [2,22,23], another important future work is to investigate combining all those economic-based mechanisms together in a grid. We believe that it is most likely that the real computational grid in the future is the one in which different economic resource allocation mechanisms exist simultaneously.

7. ACKNOWLEDGMENTS

This work was supported in part by MURI grant #F49620-00-1-0326 from DoD and AFOSR.

8. REFERENCES

- [1] Berman, F., Fox, G. and Hey, T. (Editors), *Grid Computing: Making The Global Infrastructure a Reality*, John Wiley & Sons, 2003.
- [2] Buyya, R. *Economic-Based Distributed Resource Management and Scheduling for Grid Computing*. Ph.D. Dissertation, 2002.
- [3] Caillou, P., Aknine, S. and Pinson, S., *A Multi-Agent Method for Forming and Dynamic Restructuring of Pareto Optimal Coalitions*, in proceedings of the First International Joint Conference on Autonomous Agents and Multi-agent Systems, 2002.
- [4] Clearwater, S.H.(Editor), *Market-Based Control: A Paradigm for distributed resource allocation*, World Scientific, 1996.
- [5] Foster, I. and Kesselman, C., *The Grid : Blueprint for a New Computing Infrastructure*, 1st edition, Morgan Kaufmann, 1998.
- [6] He, L. and Ioerger, T.R., *A Quantitative Model of Capabilities in Multi-Agent Systems*, in Proceedings of the International Conference on Artificial Intelligence, 2003.
- [7] He, L. and Ioerger, T.R., *Combining Bundle Search with Buyer Coalition Formation in Electronic Markets: A Distributed Approach through Explicit Negotiation*. The Sixth International Conference on Electronic Commerce, 2004.
- [8] Kahan, J.P. and Rapoport, A., *Theories of Coalition Formation*, Lawrence Erlbaum Associates, Inc., London, 1984.
- [9] Kraus, S., *Strategic Negotiation in Multiagent Environments*, The MIT Press, 2001.
- [10] Lan, Z., Taylor, V. and Bryan, G., *A Novel Dynamic Load Balancing Scheme for Parallel Systems*, *Journal of Parallel and Distributed Computing*, Vol 62/12, pp.1763-1781, 2002.
- [11] Lerman, K. and Shehory, O., *Coalition formation for largescale electronic markets*, in proceedings of the International Conference on Multi-Agent Systems, 2000.
- [12] Li, C. and Sycara, K., *Algorithm for combinatorial coalition formation and payoff division in an electronic marketplace*, in proceedings of the First International Joint Conference on Autonomous Agents and Multi-agent Systems (AAMAS), 2002.
- [13] Proter, M. E., *Competitive Strategy*, The Free Press, New York, 1980.
- [14] Rosenschein, J. S. and Zlotkin, G., *Rules of Encounter: Designing Conventions for Automated Negotiation among Computers*, The MIT Press, 1994.
- [15] Sandholm, T. *Making Markets and Democracy Work: A Story of Incentives and Computing*, in Proceedings of IJCAI, 2003.
- [16] Sandholm, T., Larson, K., Andersson, M., Shehory, O. and Tohme F., *Coalition structure generation with worst case guarantees*, *Artificial Intelligence Journal*, Vol (111), 209-238, 1999.
- [17] Sandholm, T., *Negotiation among Self-Interested Computationally Limited Agents*, Ph.D. Dissertation, 1996.
- [18] Sandholm, T. and Lesser, V., *Leveled Commitment Contracting: A Backtracking Instrument for Multiagent Systems*. *AI Magazine* 23 (3): 89-100, 2002.
- [19] Shehory, O. and Kraus, S., *Feasible formation of coalitions among autonomous agents in non-super-additive environments*, *Computational Intelligence*, vol. 15(3), 218-251, 1999.
- [20] Shehory, O. and Kraus, S., *Methods for task allocation via agent coalition formation*, *Artificial Intelligence Journal*, Vol. 101 (1-2), May, pages 165-200, 1998.
- [21] Tesfatsion, L., *Agent-based computational economics: Growing economies from the bottom up*. *Artificial Life*, 8:55--82, 2002.
- [22] Wolski, R., Brevik, J., Plank, J., and Bryan, T., *Grid Resource Allocation and Control Using Computational Economics*, In *Grid Computing: Making the Global Infrastructure a Reality* Berman, F, Fox, G., and Hey, T. editors, Wiley and Sons, pp. 747--772, 2003.
- [23] Wolski, R., Plank, J.S., Brevik, J. and Bryan, T. *Analyzing market-based resource allocation strategies for the computational Grid*. *International Journal of High Performance Computing Applications*, 15(3):258--281, 2001.