A Strategy-proof Pricing Scheme for Multiple Resource Type Allocations *

Yong Meng Teo and Marian Mihailescu

Department of Computer Science
National University of Singapore
Computing 1, 13 Computing Drive, Singapore 117417

[teoym,marianmi]@comp.nus.edu.sg

Abstract

Resource sharing on the Internet is becoming increasingly pervasive. Recently, there is growing interest in distributed systems such as peer-to-peer and grid, with efforts being directed towards resource allocation strategies that incentivize users to share resources. While combinatorial auctions can perform multiple resource type allocations, it is computationally a NP-complete problem. Thus, allocation in large distributed resource sharing systems focuses mainly on a single resource type. We propose a strategy-proof, VCG-based resource pricing scheme for resource allocation in dynamic markets where users behave rationally in meeting their own interest. Our mechanism is designed to meet the needs of large distributed systems, delivering the following key properties: multiple resource type allocations, individual rationality, incentive compatibility for both buyers and sellers, budget balance and computational efficiency. Simulation evaluation of our prototype based on a centralized implementation demonstrates the viability of our approach, as compared to both traditional and combinatorial auctions.

1 Introduction

Currently, there is growing interest in large scale resource sharing [5]. In computational grids, users may contribute and consume resources from a large pool of shared resources. Peer-to-peer technologies are used increasingly in applications ranging from file-sharing and computational networks to VOIP and Internet messaging. Users of such systems share their resources such as compute cycles, files, and bandwidth, while benefiting from the services provided by the network. Classified as rational users, their objective is to maximize their own interest while participating in the system [17].

Previous allocation techniques, such as those based on Proportional Share [4, 19, 21], do not provide incentives for rational users to be truthful about their preferences, and result in poor efficiency by maximizing individual user welfare at the expense of low overall welfare. Traditionally, agent-based approaches have been used in distributed systems where different entities have independent aims and objectives. Mechanism design [5] provides the designers of such systems the tools needed to incentivize rational agents to act in particular ways in order to achieve the desired level of "social welfare". Increasingly, peer-to-peer systems, mobile computing, e-commerce and grid computing develop into such multi-agent systems where each entity tries to maximize its own benefit [5, 17]. Recent resource allocation systems attempt to address the problem of selfish users using rational agents and mechanism design. These solutions exploit economic objectives and mechanism design in order to achieve performance and strategy-proof, using a market-based approach where agents pay for resource usage with some common virtual currency. However, it has been shown that no budget-balanced system that provides incentives can maximize the overall welfare [10].

In this paper, we investigate how mechanism design and computational economies can be used to create a pricing scheme for strategy-proof resource allocation, both fast and efficient in the context of large distributed systems where an agent can both provide and consume resources of more than one type. Our pricing scheme is expressed in a virtual economy as a mechanism that provides incentives for truthful agents.

The main contributions of our paper are: i) formulation of the resource allocation problem with rational agents in an economic context, such that mechanism design can be applied, and ii) the design of a resource pricing scheme for multiple resource type allocations with provable properties, such as individual rationality, incentive compatibility for both buyer and seller agents, and budget balance. We perform simulation evaluation for the proposed resource pricing algorithm and compare it with traditional one-sided and combinatorial auctions.

The remainder of the paper is organized as follows. Section 2 introduces mechanism design and defines key desired properties in resource allocation. Our proposed pricing scheme is discussed in Section 3 and a comparative evaluation analysis based on simulation in Section 4. Section 5 presents a summary of related work, and Section 6 concludes this paper.

2 Preliminaries

A *market* refers to the environment, expressed in terms of rules and mechanisms, where resources within an economy are exchanged. Different markets may employ distinct mechanisms for setting the prices of the traded resources. For example, in a barter economy, resources and services are exchanged directly, based on the exchange value, without a monetary system.

*This is a revised version of the paper published in the Proceedings of 38th International Conference on Parallel Processing, pp. 172-179, IEEE Computer Society Press, Vienna, Austria, September 22-25, 2009.

The allocation with the best *economic efficiency*, also called *Pareto-optimal*, is achieved when, given an allocation, no Pareto improvement can be performed [6]. Given a set of alternative allocations, a Pareto improvement is a shift from one allocation to another that can make at least one individual better off, without making any other individual worse off. Informally, best economic efficiency is achieved when social welfare is maximized. The social welfare is derived as the total welfare of all agents in the market.

The concept of economic efficiency is different from the engineering approach commonly used in computer science. Thus, we use *computational efficiency* to indicate the algorithm complexity of our pricing mechanism.

2.1 Mechanism Design

We use mechanism design [12] as a framework to create an incentive compatible pricing scheme for resource allocation. This section provides an overview of essential concepts in mechanism design and introduces the notations used in our proofs.

Given a set of n agents, each with private information $t_i \in T_i$ (e.g. reserved price, number of resource type, etc.), a social choice function f is defined as:

$$f : T_1 \times \ldots \times T_n \rightarrow O$$

where O is a set of possible outcomes (e.g. allocation results).

A mechanism M is represented by the tuple $(f, p_1, \ldots, p_n)$, where p_i is the payment received from agent i when the social choice is f. Agent's i valuation for a particular outcome o is denoted by $v_i(t_i, o(t_i^d, t_{-i}^d))$, where t_i^d is the *declared* information of agent i, as opposed to t_i which is the private (reserved) information. Similarly, t_{-i}^d is the declared private information of all other agents. Agent utility or welfare is measured using the function:

$$u_i(t_i, t_i^d, t_{-i}^d) = v_i(t_i, o(t_i^d, t_{-i}^d)) + p_i(t_i^d, t_{-i}^d)$$

The objective of a mechanism is to choose a desirable outcome o and a set of payments p_i. The criteria used to choose the outcome is defined by some desirable properties of that mechanism. In the following, we define four desirable properties for a resource allocation mechanism.

Definition 1 (Individual Rationality - IR)**.** *In an individual-rational mechanism, rational agents gain higher utility from actively participating in the mechanism than from avoiding it.*

Thus, by ensuring that agents stand to gain from participation, the mechanism provides incentives for rational agents to get resources and requests into the system.

Definition 2 (Economic Efficiency - PO)**.** *The best economic efficiency, also called Pareto-optimal, is achieved when, given an allocation, no Pareto improvement can be performed. Given a set of alternative allocations, a Pareto improvement is a shift from one allocation to another that can make at least one participant better off, without making any other participant worse off.*

A Pareto-optimal resource allocation is achieved when the total welfare of all agents is maximized. This ensures that resources are allocated to the agent that values them the most. The social choice function used to achieve economic efficiency is:

$$f_{PO}(o, t) = max_o \sum_i u_i$$

Definition 3 (Incentive Compatibility - IC)**.** *A mechanism is incentive compatible if the dominant strategy for each agent is to reveal its true valuation: $u_i(t_i, t_i^d, t_{-i}^d) = u_i(t_i, t_i, t_{-i}^d)$. Thus, agents have no incentives to declare false information, and $t_i^d = t_i$ is a truthful strategy.*

A mechanism that is both incentive-compatible and individual rational is said to be *strategy-proof*.

Definition 4 (Budget Balance - BB)**.** *In a budget balanced mechanism, the sum of all agent payments is $\sum_i p_i = 0$.*

Budget balance ensures that allocations do not result in budget deficit or surplus.

Vickrey, Clarke and Groves (VCG) introduce a class of mechanisms that are both economic efficient and strategy-proof [7]. Suitable for any utilitarian design problem, VCG mechanisms are characterized by the following payment function:

$$p_i = \sum_{j \neq i} v_j(t_j, o) + h_i(t_{-i})$$

where h_i is an arbitrary function of private information of the other agents, t_{-i}. Informally, VCG mechanisms are incentive compatible because agent payments are determined independently of their declared private information, hence there are no incentives for a rational agent to declare false information. Furthermore, they are efficient because the payment is a function of all other agents' valuations, and individual rational because all agents' welfare is positive, and agents involved in an exchange have welfare greater than 0.

The Myerson-Satterthwaite impossibility theorem [10] is a well-known result, which verifies that no mechanism can achieve all four properties at the same time. Indeed, VCG mechanisms are not budget-balanced [13] and require a third-party agent (called a market-maker) to mediate between seller and buyer agents and to provide the surplus or deficit budget. Parkes et al. [13] argue that budget-balance is possible in a Vickrey-Clarke-Groves mechanism if payments are implemented on one side of the exchange, and that side has no aggregation. In resource allocation, an aggregation involves a bundle containing more than one resource type.

3 Proposed Pricing Mechanism

Resource allocation is a complex process, which can be divided in several independent steps, such as resource location, pricing, allocation administration, etc. In this paper, we focus on the design and evaluation of the pricing mechanism.

Pricing represents the process of computing the economic exchange value of resources relative to some common currency. Thus, we consider a virtual economy in which common currency provides the basis for a utilitarian function that can be exploited by our mechanism. This qualifies the common currency to measure the welfare of market participants, to provide the means for

incentives, and to enable the trading of multiple resource types. Other mechanism design incentive-compatible solutions, such as network-of-favors or tit-for-tat, are not able to provide a virtual economy in which resources of different types can be traded.

In a resource market, the process of matching market participants, e.g. buyers and sellers, to engage in an exchange is called *pricing mechanism*. The objective of the pricing mechanism is to choose an outcome, also known as winner determination, and a set of payments. The winner determination problem resolves the agents which participate in an exchange. The payments for that exchange are facilitated by the use of common currency. Given a fixed set of alternative choices, with subjectively known distributions of outcomes for each alternative, a rational agent selects an alternative to maximize the expected value of his utility function. In the context of our virtual economy, the goal of the agents is to maximize the amount of currency they possess in order to secure resources in the system. Thus, rational agents are incentivized to share resources in order to gain currency such that their request benefits from the best possible allocation.

A mechanism design problem consists of an outcome specification and a set of agent utilities. The output specification maps each vector of private information $t_1, \ldots t_n$ to the set of allowed outputs, $o \in O$. We formulate the resource allocation problem in a resource sharing system where users can both share and consume resources as a general mechanism design problem as follows.

Market-based Resource Allocation Problem

Given a market containing requests submitted by buyers and resources offered by sellers, each participant is modeled by a rational agent i with private information t_i. A seller agent has private information t_s^r, the underlying costs for the available resource r, such as power consumption, bandwidth costs, etc. The buyer's agent private information is t_b^R, the maximum price the buyer is willing to pay such that resources are allocated to satisfy its request R. Agent's i valuation is $t_i^{r/R}$ if the resource r (request R) is allocated, and 0 if not. For a particular request R, the goal is to allocate resources such that the underlying costs are minimized.

More formally:

- *The possible outputs of the mechanism are all partitions $x = x_1 \ldots x_n$ of resources that satisfy r, where x_i is the set of resources that agent i contributes to the allocation.*
- *The objective function is $g(x,t) = \sum_{|x_i|>0} \sum_{j \in x_i} t_i^j$.*
- *Agent's i valuation is $v_i(x,t_i) = \sum_{j \in x_i} t_i^j$.*

This is an optimization problem, since the output specification is given by a positive real valued objective function, $g(x,t)$, and the output o minimizes g. Moreover, it is utilitarian since the objective function satisfies the relation $g(o,t) = \sum_i u_i(t_i, o)$. This allows us to apply a VCG mechanism to our design problem.

Alongside the seller and buyer agents, we introduce the market-maker agent, which acts as resource broker to mediate between sellers and buyers (see Figure 1).

An allocation is determined by a market-maker agent and represents an exchange between one buyer agent and at least one seller agent. Payments are made by the buyer agent for the resources allocated to satisfy its request, and received by seller agents for the resources provided for the allocation.

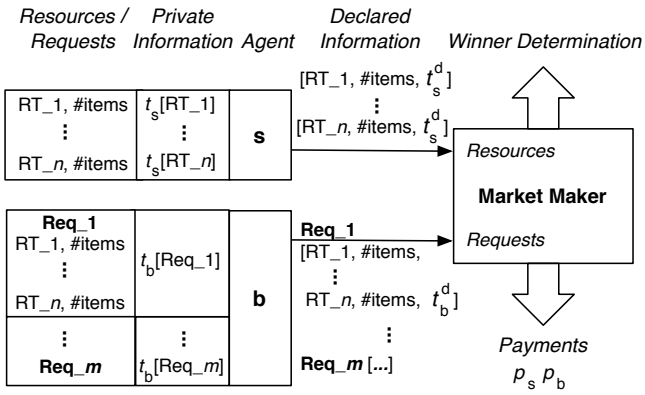

Figure 1. Market-based Resource Allocation

3.1 Winner Determination for Multiple Resource Types

Our scheme is designed for pricing and allocation of multiple resource types in dynamic markets, where buyer and seller agents may join and leave at any time. In this section, we describe the buyer request and seller available resource, and our strategy for selecting winners.

A buyer request consists of one or more resource types and its associated number of items requested. A simplified request description shown below include the buyer identifier, each resource type (RT) with its number of items (#items), and the buying price (t_b^{Rd}):

```
request[buyer_id, (RT_1, #items) and
                   ... and
                  (RT_n, #items), price]
```

Seller agents publish its resources as and when they become available. A seller agent publishes multiple resource descriptions, one for each resource type. The resource description includes its identifier, resource type and number of items, and the price per item (t_s^{rd}) :

```
resource[seller_id, RT, #items, price/item]
```

Resource descriptions give the market-maker the flexibility to allocate a number of seller items for each resource type to more than one buyer and thus maximize resource utilization. We use t_b^{Rd} to denote the buyer's maximum declared price, as opposed to t_b^R which is the private maximum price. Similarly, we use t_s^{rd} for sellers. We consider that the seller's private price for an item of a resource type is equal to the underlying costs for that resource, t_s^r.

An agent forwards its buy and sell requests to the market-maker. The buyer requests and seller available resources contain the declared information (see Figure 1) and represent the input for our pricing mechanism. Firstly, the market-marker selects a buyer request and determines the set of successful sellers. Buyer requests are selected on a First-Come-First-Serve basis to minimize agent waiting time. The winning sellers are determined such that all items of all resource types in the buyer request are

allocated and the underlying resource costs are minimized. This strategy maximizes the overall welfare of sellers. Next, payments to buyer and sellers are determined using the payment functions detailed below. Furthermore, we present the proofs for the properties achieved by our mechanism.

3.2 Seller Payment Function

The payment p_s for a seller agent s after the allocation of a request R is determined by the function:

$$p_s = \begin{cases} 0, & \text{if } s \text{ does not contribute} \\ & \text{resources to satisfy } R \\ -c_{M|s=\infty} + c_{M|s=0} & \\ & \text{if } s \text{ contributes} \\ & \text{resources to satisfy } R \end{cases} \quad (1)$$

where:

$c_{M|s=\infty}$ is the lowest cost to satisfy R without the contribution of agent s;

$c_{M|s=0}$ is the lowest cost to satisfy R when the cost of agent's s resources is 0.

We can see that the above function is a VCG payment for seller agents, since $c_{M|s=\infty}$ corresponds to $h_i(t_{-i})$, the arbitrary function of resource prices of the other agents, and $c_{M|s=0}$ corresponds to $v_s(t_s, o)$, the valuation of agent s.

3.3 Buyer Payment Function

Given the set S of sellers with resources to satisfy a request R, the buyer payment function p_b is:

$$p_b = -\sum_{s \in S} p_s \quad (2)$$

3.4 Achieved Properties

Theorem 1. *The proposed mechanism is individual rational.*

Proof. We consider the allocation of a request R from agent b. The output of our mechanism is $x = x_1 \dots x_n$, where x_i is the set of resources that seller i contributes to the allocation. Formally, the IR property verifies that $u_i \geq 0$ for any winner agent i. The winner agents in the allocation are: b, the buyer agent, and S, a set of seller agents such that $s \in S \iff |x_s| > 0$. When determining the winners, our mechanism chooses the sellers with the lowest underlying costs. Let σ be the winner seller with the highest underlying costs, and ς the seller with the lowest underlying costs which is not a winner. Consequently:

$$v_\sigma = \max_{s \in S} v_s, v_\varsigma = \min_{j \notin S} v_j; v_\sigma \geq v_\varsigma$$

The payment for seller σ is:

$$p_\sigma = -c_{M|\sigma=\infty} + c_{M|\sigma=0} = -(v_{S-\sigma} + v_\varsigma) + v_{S-\sigma} = -v_\varsigma$$

Thus, the utility of the seller σ is:

$$u_\sigma = v_\sigma - v_\varsigma \geq 0 \quad (3)$$

Given the buyer payment, $p_b = -\sum_{s \in S} p_s$, buyer utility is:

$$u_b = v_b - \sum_{s \in S} p_s \quad (4)$$

The market-maker implements our mechanism by solving the winner determination problem and computing agent payments. From Equation 3, we see that seller utility is always positive. In addition, the market-maker implementation verifies that the result in Equation 4 is positive, to ensure that the utility of the participating buyer is positive (see Figure 3, line 14). □

Theorem 2. *The proposed mechanism is budget-balanced.*

Proof. It can be seen from Equation 2 that our mechanism uses the buyer payment function to achieve budget balance. Indeed, given a buyer request R, and the set of sellers S which provide resources for R, the sum of all agent payments is:

$$\sum_i p_i = p_S + p_b = \sum_{s \in S} p_s - \sum_{s \in S} p_s = 0 \qquad \square$$

Theorem 3. *The proposed mechanism is incentive compatible.*

Lemma 1. *Seller payment function is incentive compatible.*

Proof. From Equation 1, we can see that the seller payment function is based on VCG and is inherently IC, thus we omit the proof and express the property in Equation 5.

$$u_s(t_s, t_s^d, t_{-i}^d) = u_s(t_s, t_s, t_{-i}^d) \qquad \square \quad (5)$$

Lemma 2. *Buyer payment function is incentive compatible.*

Proof. From Equation 2, the buyer payment function depends on the seller agents' payments p_s, and is independent of the buyer's valuation, v_b. Seller agent payments depend on the private information of other seller agents, t_{-s}, and its own valuation for an outcome, $v_s(t_s, o)$. Thus:

$$p_b(t_b^d, t_{-i}^d) = p_b(t_b, t_s) = p_b(t_b, t_{-i}^d) \quad (6)$$

We require the buyer agents to have the same valuation independent of the outcome selected by our pricing mechanism, $v_b(t_b, o)$. This is achieved by selecting buyer requests using a strategy that is independent of the buyer's valuation, such as First Come First Serve. Thus, requests are considered by a market-maker agent based on the time of their arrival, and not the intrinsic value for the buyer agents valuation, such that:

$$v_b(t_b, o(t_b^d, t_{-b}^d)) = v_b(t_b, o(t_b, t_{-b}^d)) = v_b(t_b, o(t_b, t_{-b})) \quad (7)$$

Furthermore, market-maker agents do not take into consideration the valuation of sellers when computing the payment for a buyer agent, as long as the condition for IR is satisfied. This allows us to rewrite the above equation as:

$$v_b(t_b, o(t_b^d, t_{-b}^d)) = v_b(t_b, o(t_b, t_{-b}^d)) =$$
$$= v_b(t_b, o(t_b, t_{-b})) = v_b(t_b, o(t_b, t_{-i})) \quad (8)$$

where t_{-b} represents the private information of all other buyers, and t_{-i} the private information of all other agents, buyer and seller.

Adding Equation 8 and Equation 6, we obtain the equality:

$$v_b(t_b, o(t_b^d, t_{-b}^d)) + p_b(t_b^d, t_{-i}^d) = v_b(t_b, o(t_b, t_{-i})) + p_b(t_b, t_{-i}^d)$$

which stands for:

$$u_b(t_b, t_b^d, t_{-i}^d) = u_b(t_b, t_b, t_{-i}^d) \quad (9)$$

This relation expresses the IC property of the buyer payment function. □

The results of Lemma 1 and Lemma 2 prove Theorem 3, as $t_i^d = t_i$.

3.5 Generalized Algorithm and Example

The proposed pricing algorithm is outlined in Figure 3. Assume a market consisting of buyer requests and seller available resources published to a market-maker agent, together with their reserved prices, t_i (private information).

Each request is dequeued according to a policy independent of the buyer's valuation, such as FCFS (line 3). For each resource type (line 4), the market-maker sorts the seller queue for that resource type based on the reserved price, t_s (line 5). Next, it selects the winning sellers from the head of the sorted queue such that all items in the request may be allocated (lines 6–7), solving the winner determination problem. Subsequently, we determine the payments for each seller winner p_s by computing the two associated costs, $c_{M|s=\infty}$ and $c_{M|s=0}$ (lines 8–11). Finally, we compute the buyer payment p_b (line 13) and inform the winners of the allocation if their welfare is greater than 0 (lines 14–15).

```
1  MarketMaker (buyer_queue, seller_queue)
2  while buyer_queue is not empty
3      select next request from the buyer_queue (FCFS)
4      for each resource type rt in request
5          sort seller resources of the same type by cost t_s
6          for all items of resource type rt in request
7              determine winning sellers
8              determine c_{M|s=0}
9          for each winning seller
10             remove seller resources from seller_queue
11             determine c_{M|s=∞}
12         compute winning seller payment p_s
13     compute buyer payment p_b
14     if buyer welfare ≥ 0
15         notify successful sellers and buyer of payments
```

Figure 3. Generalized Market-Maker Algorithm

For a better understanding of our mechanism, we present a simple example. Consider a simple market with two resource types, CPU and disk space, with three sellers and two buyers. Seller S1 provides one unit of CPU for $1/hour, S2 provides one CPU at $2/hour and disk space at $2/hour, and S3 provides disk space for $1/hour. Buyers B1 and B2 both offer to buy one unit of CPU and disk space for one hour at $5 and $6, respectively (Figure 2).

Assuming FCFS policy for requests and B1's request arrives before B2's request, B1 is selected by the market-maker first. We sort the resource list based on the reserved price, and determine the winners: CPU from S1 and disk space from S3. Table 1 shows the individual payment computation for S1 and S3, together with B1, the owner of the request.

Agent	$c_{M\|s=\infty}$	$c_{M\|s=0}$	Payment
S1	2 + 1 = 3	0 + 1 = 1	-3 + 1 = -2
S3	1 + 2 = 3	1 + 0 = 1	-3 + 1 = -2
B1	N/A	N/A	- (-2 - 2) = 4

Table 1. Proposed Payment Scheme

Using this simple example, we can observe both the properties achieved by our mechanism, strategy-proof and budget-balance, and the trade-off in terms of economic efficiency. A solution that is pareto-optimal, but not budget-balanced, may be achieved using VCG payments for both buyers and sellers, as below. First, we compute the total welfare for all possible exchanges in Table 2, where winner agents are shown in bold font.

Total Welfare		Exchange
w/o S1	**6 - 2 - 1 = 3**	**B2 buys from S2, S3**
w/o S2	6 - 1 - 1 = 4	B2 buys from S1, S3
w/o S3	**6 - 1 - 2 = 3**	**B2 buys from S1, S2**
w/o B1	6 - 1 - 1 = 4	B2 buys from S1, S3
w/o B2	**5 - 1 - 1 = 3**	**B1 buys from S1, S3**
maximum	*6 - 1 - 1 = 4*	*B2 buys from S1, S3*

Table 2. Total Welfare and Resource Allocation

Since Pareto-optimal allocation maximizes the sum of agents utilities, the winners selected in our example are S1, S3 and B2 with the final payments shown in Table 3.

Agent	Payment
S1	-1 - (4 - 3) = -2
S3	-1 - (4 - 3) = -2
B2	6 - (4 - 3) = 5

Table 3. Computation of VCG Payments

It is clear that total welfare is greater with VCG payments than when using our algorithm (*B1 buys from S1, S3 vs. B2 buys from S1, S3* in Table 2). However, one thing to note is that achieving pareto-optimality usually requires an algorithm with exponential complexity, making the trade-off both in terms of budget-balance (-2-2+5 yields in $1 deficit in the case of VCG payments), and computational efficiency.

4 Evaluation

We evaluate the performance of our mechanism using efficiency as a performance measure. Welfare measures the *economic efficiency* and the algorithm runtime represents the *computational efficiency*. *Global efficiency* for buyers is defined as the total number of successful buyer requests, and for sellers as the average resource utilization of all seller agents, i.e. total resources utilized over total available seller resources.

Using simulation, we first compare our mechanism with traditional auctions. We evaluate the impact of untruthful users in a balanced market under different market conditions. Second, we compare the performance of our mechanism with combinatorial auctions [9, 11]. For simplicity, we consider a centralized implementation characterized by a single market-maker agent to which sellers and buyers submit their requests and available resources.

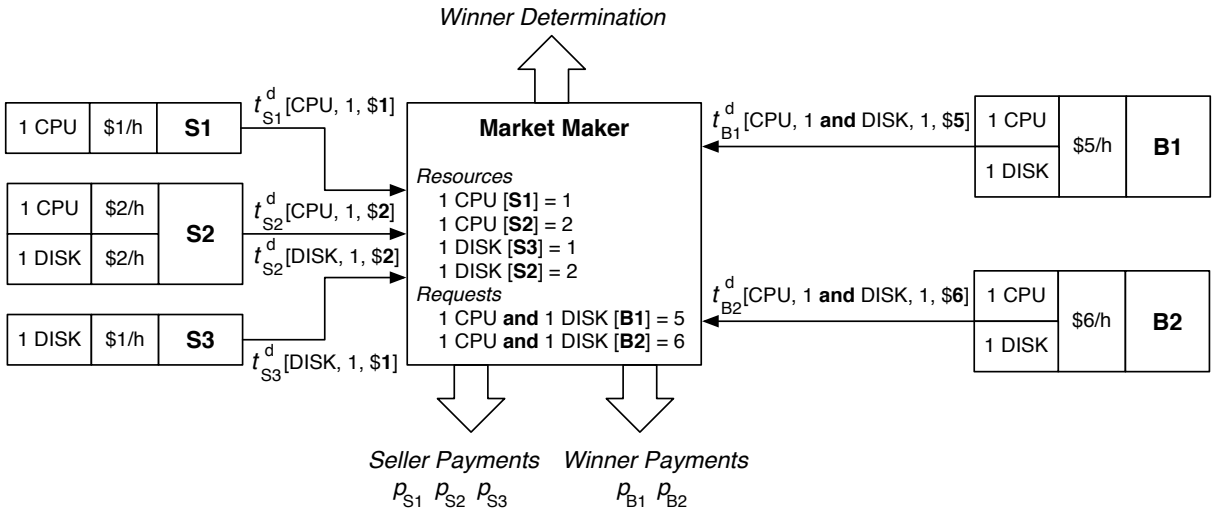

Figure 2. Market with 3 Sellers and 2 Buyers

4.1 Comparison with Traditional Auctions

For comparison with traditional one-sided auctions, we have developed a discrete-event auctions simulator with a request queue holding all outstanding buyer requests, and a resource queue containing seller agents published resources. Both queues are ordered FCFS. One-sided auctions take place by matching a buyer request with all available resources in the seller resource queue.

As in a real market, we model three market conditions: *balanced* where resource demand matches supply, *under-demand* where resource demand is less than supply, and *over-demand* where resource demand exceeds supply. First, we consider a balanced market to study the impact of untruthfulness on global efficiency of buyers, i.e. total number of successful buyer requests. We use *untruthful* to specify the percentage of untruthful agents, and *price change* for the different between the truthful price (private information) and the declared price. The truthful price is sampled from an exponential distribution. Our experiments summarized in Figure 4 show that in traditional auctions global efficiency degrades proportionally with the number of untruthful agents in the system. This result motivates the need of a strategy-proof mechanism in which participants are incentivized to be truthful.

Next, we compare the proposed scheme with traditional one-sided auctions under different market scenarios. Market conditions are modeled by varying the arrival rates of buyers and sellers. In a *balanced market*, both rates are equal and are sampled from an exponential distribution with a mean of 1 arrival/minute. For *under-demand* and *over-demand* markets, we set the mean seller arrival rate at 2 and 0.5 arrivals per minute, respectively.

The difference between the buyers and sellers private information, expressed as percentage, is captured by *market diversity*. Greater market diversity means that the difference between buyers prices and sellers costs is higher, which leads to a smaller number of possible allocations due to individual rationality.

Table 4 compares 10,000 buyer requests under different market scenarios and with different market diversity. For simplicity, each buyer request consists of one resource type and one item

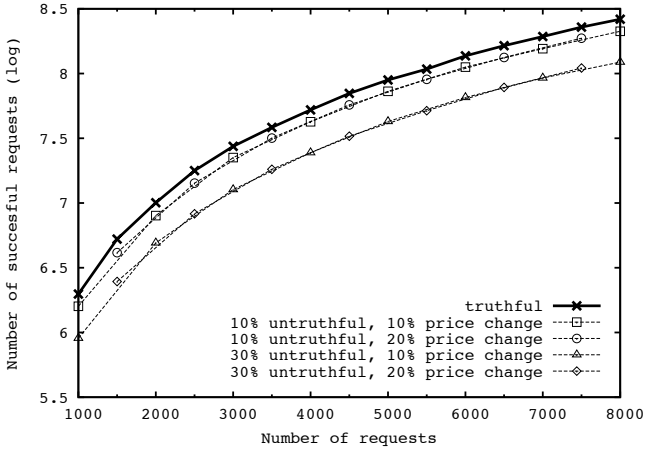

Figure 4. Varying the Degree of Untruthfulness in a Balanced Market

Market Diversity(%)	Number of Successful Requests		
	Auctions	Proposed	Increase(%)
Balanced Market			
10	5,475	6,918	26.4
20	5,454	6,933	27.1
40	5,452	6,903	26.6
Under-Demand			
10	6,637	7,894	18.9
20	6,645	7,904	19.0
40	6,637	7,909	19.2
Over-Demand			
10	3,348	3,906	16.7
20	3,380	3,906	15.6
40	3,364	3,912	16.3

Table 4. Varying Market Diversity

Our scheme achieves an overall improvement of 15% in terms of the total number of successful buyer requests over traditional auctions and over 25% for a balanced market.

Lastly, we vary the number of resource types in both buyer and seller requests between 1 and 16. Figure 5 shows that for multiple resource type allocations our algorithm achieved much higher global efficiency for buyer agents, i.e., total number of successful buyer requests.

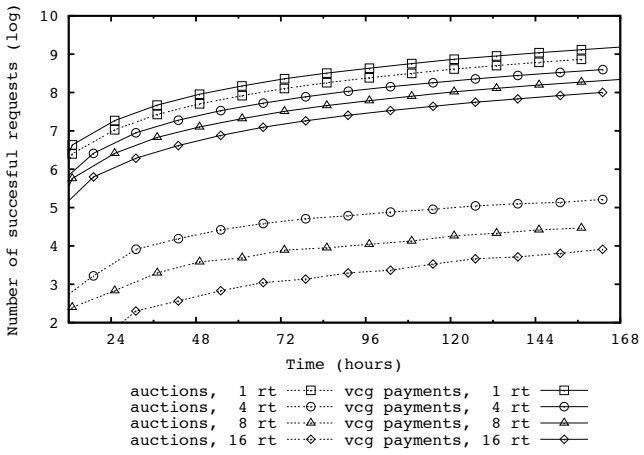

Figure 5. Varying Number of Resource Types

4.2 Comparison with Combinatorial Auctions

In this section we compare the proposed scheme with combinatorial auctions using the open-source combinatorial auctions simulator jCase [16]. We select for comparison the pure VCG and the Threshold algorithms proposed by Parkes et al. [13]. We assume a balanced market with equal number of buyers and sellers (*#Agents*). Each buyer request consists of many resource types sampled from a uniform distribution between 1 and 10. In each simulation run, we perform 50 allocation rounds and generating a total of $50 * \#Agents$ buyer requests. As shown in Table 5, we vary *#Agents* from 20 to 80 for combinatorial auctions, and up to 1000 for our proposed algorithm. *IC* denotes the incentive compatible schemes, *BB* the sum of all agent payments, and *Welfare* the economic efficiency.

Computational efficiency of the pricing algorithm is measured by the wallclock time (*Runtime*, where h = hours, m = minutes, and s = seconds) to execute an algorithm. Because winner determination in combinatorial auctions is a NP-complete problem [9, 15], each experiment with 80 agents takes more than 60 hours[1] to execute. In contrast, our algorithm takes just 5 seconds. Global efficiency for sellers is indicated by *Util*, and the percentage of successful buyer requests by *Success*.

We have shown that our scheme is strategy-proof and budget balanced, and simulation results further show our scheme is efficient. Overall, our scheme is comparable to combinatorial auctions both in economic efficiency, measured in terms of overall welfare, and global efficiency, average seller resource utilization and percentage of successful buyer requests. More importantly, our scheme achieves computational efficiency that is essential for deployment in large and dynamic distributed systems.

[1]Our experiments are performed on a 8-core Intel Xeon, 1.86 GHz server with 4GB RAM.

5 Related Work

Spawn [18] is one of the first implementations of a market-based system that utilizes idle computer resources in a distributed network. More recent work includes Popcorn [14], Berkley Millennium Project [4], Nimrod/G [2], LibraSLA [21], Tycoon [8], G-commerce [20], Bellagio [1] and Mirage [3]. These approaches are either based on Proportional Share [4, 21], or use some types of auction or negotiation to determine the price of resources. Some disadvantages of these systems are discussed below.

In a system with a finite amount of virtual currency, rational users are incentivized to use it based on their expected utility. By maximizing the total user utility, these systems aim to reach Pareto efficiency [6]. However, typical auction-based approaches have difficulties when an agent request consists of multiple resource types. Thus, many of such systems [8, 14, 18] focus on allocating only one type of resource such as CPU cycles. An alternative to auctions, commodity markets rely on a trusted third-party negotiator (the market-maker) that periodically queries buyers and sellers to determine the supply and demand, and compute the resource prices. This approach improves allocation results in computational grids, as in Nimrod/G [2] and G-commerce [20]. However, in many distributed systems, users are both consumers and providers of shared resources. By periodically refreshing the users budgets, these systems provide incentives only for buyers to be truthful. In contrast, our scheme is designed to incentivize both buyers and sellers.

The combinatorial auction scheme [9, 11] addresses resource allocation for multiple resource types. Systems designed to allocate "bundles" of resources include Bellagio [1] and Mirage [3]. However, to the best of our knowledge, there is no algorithm to find the best combinatorial auction allocation in polynomial time [15]. Approximation algorithms [13] are used, but are not incentive compatible and, inherently, economic efficiency is degraded. Furthermore, combinatorial auctions assume all bids are present at the same time. Our scheme is designed for dynamic markets, where the number of buyer and sellers change dynamically over time and require dynamic allocations to achieve good service levels.

A strategy-proof pricing model is required for both buyers and sellers in a dynamic market where rational users trade resources. Our model is designed for such dynamic markets and achieves key properties such as *incentive compatibility*, *budget balance* and *computational efficiency*.

6 Conclusions and Future Work

We have presented a new strategy-proof resource pricing scheme suitable for large distributed systems where rational users share and utilize resources. We formulate the resource allocation problem in an economic context, allowing mechanism design to be applied. As a trade-off for economic efficiency, we achieve individual rationality, incentive compatibility and budget balance. Unlike traditional auctions, our scheme is designed to perform multiple resource types allocation. In comparison with combinatorial auctions, simulation results show that our proposed scheme is computationally efficient and suitable for dynamic markets with rational users. Overall, our pricing mech-

Pricing Mechanism	#Agents	Overall			Performance		
		IC	BB	Welfare	Runtime	Success (%)	Util (%)
Combinatorial - VCG	20	✓	-1402	2,470	9.6m	44.5	44.8
	40	✓	-1544	6,321	2.5h	52.5	57.2
	80	✓	-1,557	14,384	67.4h	54.2	64.0
Combinatorial - Threshold [13]	20	✘	5	2,491	9.8m	44.4	48.3
	40	✘	9	6,223	2.5h	49.6	59.8
	80	✘	6	14,567	49.5h	54.8	65.1
Proposed Mechanism	20	✓	0	1,871	1s	36.3	32.5
	40	✓	0	5,483	3s	48.5	55.3
	80	✓	0	11,561	5s	52.8	68.0
	100	✓	0	14,369	7s	54.1	71.6
	200	✓	0	28,564	20s	53.5	76.5
	500	✓	0	65,948	1.9m	52.6	80.2
	1000	✓	0	132,729	7.7m	52.8	81.5

Table 5. Comparison of Combinatorial Auctions and Proposed Scheme

anism also achieved better global efficiency when compared to both traditional and combinatorial auctions.

To improve the scalability of our scheme, we are developing a fully decentralized pricing scheme for large distributed systems. Because users are able to both utilize and share resources, we are leveraging on peer-to-peer networks as an underlying overlay for both users and resources to support resource discovery based on resource type and market-based resource pricing and allocation. In such a highly dynamic and distributed system with decentralized pricing information, the main challenge remains to preserve the key properties of incentive compatibility, budget balance, multiple resource type allocation and global efficiency.

Acknowledgments

This work is supported by the Singapore Ministry of Education under AcRF grant number R-252-000-339-112.

References

[1] A. AuYong, B. N. Chun, A. C. Snoeren, and A. Vahdat, Resource Allocation in Federated Distributed Computing Infrastructures, *Proc. of the 1st Workshop on Operating System and Architectural Support for the On-demand IT InfraStructure*, Boston, USA, 2004.

[2] R. Buyya, D. Abramson, and J. Giddy, Nimrod/G: An Architecture for a Resource Management and Scheduling System in a Global Computational Grid, *Proc. of the 4th Intl. Conference on High Performance Computing in Asia-Pacific Region*, pp. 283-289, Beijing, China, 2000.

[3] B. Chun, P. Buonadonna, A. AuYoung, C. Ng, D. Parkes, J. Shneidman, A. Snoeren, and A. Vahdat, Mirage: A Microeconomic Resource Allocation System for SensorNet Testbeds, *Proc. of the 2nd IEEE Workshop on Embedded Networked Sensors*, pp. 19-28, Sydney, Australia, 2005.

[4] B. Chun, and D. Culler, Market-based Proportional Resource Sharing for Clusters, Millenium Project Research Report, University of California at Berkeley, 1999.

[5] R. K. Dash, N. R. Jennings, and D. C. Parkes, Computational Mechanism Design: A Call to Arms, *IEEE Intelligent Systems*, Vol. 18(6), pp. 40-47, 2003.

[6] D. Fudenberg, and J. Tirole, Game Theory, MIT Press, 1983.

[7] T. Groves, Incentives in Teams, *Econometrica*, Vol. 41(4), pp. 617-631, Econometric Society, 1973.

[8] K. Lai, B. A. Huberman, and L. R. Fine, Tycoon: A Distributed Market-based Resource Allocation System, Technical Report DC/0404013, Helwett-Packard, 2004.

[9] D. Lehmann, R. Muller, and T. Sandholm, The Winner Determination Problem, *Combinatorial Auctions*, Chapter 12, MIT Press, 2006.

[10] R. Myerson, and M. A. Satterthwaite, Efficient Mechanisms for Bilateral Trading, *Journal of Economic Theory*, pp. 265-281, Elsevier, 1983.

[11] N. Nisan, Bidding and allocation on combinatorial auctions, *Proc. of the 2nd ACM Conference on Electronic Commerce*, pp. 1-12, Minneapolis, USA, 2000.

[12] N. Nisan, and A. Ronen, Algorithmic Mechanism Design (Extended Abstract), *Proc. of ACM Symposium on Theory of Computing*, pp. 129–140, Atlanta, USA, 1999.

[13] D. C. Parkes, J. R. Kalagnanam, and M. Eso, Achieving Budget-Balance with Vickrey-Based Payment Schemes in Exchanges, *Proc. of the 17th Intl. Conference on Artificial Intelligence*, pp. 1161-1168, Seattle, USA, 2001.

[14] O. Regev and N. Nisan, The Popcorn Market: Online Markets for Computational Resources *Proc. of the 1st Intl. Conference on Information and Computation Economies*, pp. 148-157, Charleston, USA, 1998.

[15] T. Sandholm, An Algorithm for Optimal Winner Determination in Combinatorial Auctions, *Proc. of the 16th Intl. Joint Conference on Artificial Intelligence*, pp.542–547, Stockholm, Sweden, 1999.

[16] B. Schnizler, Java Combinatorial Auction Simulation Environment, `http://www.schnizler.com/jcase.html`

[17] J. Shneidman, and D. C. Parkes, Rationality and Self-Interest in Peer to Peer Networks, *Proc. of 2nd Intl. Workshop on Peer-to-Peer Systems*, pp. 139–148, Berkeley, USA, 2003.

[18] C. A. Waldspurger, T. Hogg, B. A. Huberman, J. O. Kephart, and W. S. Stornetta, Spawn: A Distributed Computational Economy, *IEEE Transactions on Software Engineering*, Vol. 18, pp. 103-117, 1992.

[19] C. A. Waldspurger, and W. E. Weihl, Lottery Scheduling: Flexible Proportional-Share Resource Management, *Proc. of the 1st USENIX Symp. on Operating Systems Design and Implementation*, pp. 1-11, California, USA, 1994.

[20] R. Wolski, J. S. Plank, J. Brevik, and T. Bryan, G-commerce: Market Formulations Controlling Resource Allocation on the Computational Grid, *Proc. of the 15th Intl. Parallel & Distributed Processing Symposium*, pp. 46, San Francisco, USA, 2001.

[21] C. S. Yeo and R. Buyya, Service Level Agreement Based Allocation of Cluster Resources: Handling Penalty to Enhance Utility, *Proc. of the 7th Intl. Conference on Cluster Computing*, pp. 27–30, Boston, USA, 2005.