A Dynamic Pricing and Bidding Strategy for Autonomous Agents in Grids

Behnaz Pourebrahimi[1], Koen Bertels[1], Stamatis Vassiliadis[1], and Luc Onana Alima[2]

[1] Computer Engineering Laboratory, EEMCS, Delft University of Technology, The Netherlands
{b.pourebrahimi,k.l.m.bertels,s.vassiliadis}@tudelft.nl
[2] Service des Systémes Distribués, Institut d′Informatique, Université de Mons-Hainaut, Belgium
luc.onana@umh.ac.be

Abstract. In this paper, we propose a dynamic pricing strategy which is used for a market-based resource allocation mechanism in a local Grid. We implement an agent based Grid economy in which the decision-making process regarding task and resource allocation is distributed across all users and resource owners. A Continuous Double Auction is used as the platform for matchmaking where consumers and producers meet. In this paper, we analyze the parameter regime of this pricing mechanism considering different network conditions. Our experiments described in the paper show that using the pricing parameters, the consumers and producers agents can decide the price to influence the way they contribute resources to the Grid or complete the jobs for which they need resources. These agents are individually capable of changing the degree of their task usage and resource contribution to the Grid.

1 Introduction

In recent years, intensive computational applications are becoming more and more popular. In case of a lack of computational resources in such applications, instead of consuming an extra budget to buy centralized resources, one solution is to use existing computing resources over the network that otherwise lie idle. These networks of distributed and shared resources are known as Grids.

The problem we are looking at is resource allocation and task distribution in a Grid-based environments, where the resources are heterogeneous in nature, they are owned by various individuals or organizations with different objectives and they have dynamically varying loads and availability. Such a system could be deployed to any organization having a LAN with any number of computers, in which the tasks can be processed on any node that has idle resources. Managing resources and allocating them to requested tasks in such dynamic and heterogeneous environment is a challenging task and needs to be smart, adaptable to changes in the environment and user requirements.

Conventional resource allocation schemes are based on relatively static models where a centralized controller manages jobs and resources. In fact, they focus on

Samuel R.H. Joseph et al. (Eds.): AP2PC 2007, LNAI 5319, pp. 55–71, 2010.

efficient allocation schedules which can optimize given performance metrics such as allocation time, resource utilization or system throughput. These resource allocation mechanisms may work well where resources are known in advance. However, they may fail to work in dynamic networks where jobs need to be executed by computing resources whose availability is difficult to predict. Due to the dynamic nature of such networks, mechanisms that are based on a system-wide performance metric to allocate resources, are not suitable. Therefore, resource allocation mechanisms are required that take into account both system and user performances and can adapt to variations in supply and demand of resources. Market-based mechanisms provide promising directions for building such a resource allocation mechanism. One of the promises, taken from economic theory, is that the fulfillment of individual self-interest automatically or through an unspecified mechanism called the *Invisible Hand* (proposed by Adam Smith [1]), leads to maximal generation of utility for the entire community. When transposed to the Grid environment, this implies that as long as individual nodes look after themselves, by buying or selling resources, the overall goal, namely to execute tasks, is also satisfied. Moreover, market-based mechanisms can provide adaptability in such dynamic networks by distributing decision making among the individual self-interested nodes. The self-interested nodes in a network can be presented by autonomous agents. The autonomous agents make their own decisions according to their capabilities, goals, and local knowledge without considering the global good of the entire Grid. Individual decision making is achieved through a large amount of decentralized information which is condensed into a single, simple entity, namely the price.

In this paper, we look at a particular pricing strategy and study its parameter's regime given different Grid conditions. We consider a Continuous Double Auction (CDA) mechanism for matchmaking between consumers and producers of resources. In this model, the consumers and producers of resources put their requests or offers attached with a price into the market as bids/asks. Buy orders (requests) and sell orders (offers) may be submitted at anytime during the trading period. No global and single equilibrium price is computed in this strategy; rather at any time when there are open requests and offers that match or are compatible in terms of price and requirements (e.g. quantity of resources), a trade is executed immediately.

The main contribution of this paper is to identify how an individual agent can take into account its own task loads and available resources as well as the Grid's overall condition. Using our pricing strategy, we show how the agents can adapt to a dynamic network condition where the distribution of tasks and availability of resources may change at any time. In addition, based on this strategy each agent can decide on the contribution of its resources or demanding for its tasks at any time as its available resources or its workload changes. In our pricing strategy, the price proposed by consumer and producer agents changes based on the perceived supply and demand in the network. Consumers generate aggressive bids by raising the price when supply is low and conservative bids by lowering the price when supply is high. On the other hand, producers with a conservative

or aggressive strategy respectively raise the price when demand is high and lower the price when demand is low.

The paper is structured as follows: In Section 2, we discuss market-based resources allocation based on peer-to-peer architectures. We give an overview of related works on market-based resource allocation in Section 3. Section 4 discusses the system architecture. Pricing model is presented in Section 5. The experiments are discussed in Section 6 considering different network and node conditions. Finally, conclusion and future work are discussed in Section 7.

2 Market-Based Resource Allocation and Peer-to-Peer Architectures

The limitations of client/server mechanisms for resource allocation have become evident in large scale distributed environments. In such systems, individual resources are concentrated on one or a small number of nodes. In order to provide access to resources with an acceptable response time, sophisticated load balancing and fault-tolerant algorithms have to be applied. These limitations have motivated researchers to suggest approaches to distribute processing loads and network bandwidth among all nodes participating in a distributed system. Peer-to-peer systems offer an alternative to traditional client/server systems that solve bottleneck problems and improve the Grid scalability. Different resource allocation mechanisms can be employed based on peer-to-peer architectures ranging from fully centralized (centralized indexing) to fully decentralized (blind flooding) [2]. Fully centralized mechanisms can be efficient for small scale systems and may take less time in finding a required resource. However, these mechanisms are not scalable and the centralized resource broker becomes a performance bottleneck. In contrast, fully decentralized mechanisms do not have a single point of failure and may have better scalability. The drawback is that fully decentralized mechanisms are computationally expensive and may take more time to find a resource. Fully decentralized mechanisms do not also guarantee finding a resource. Fully centralized and fully decentralized mechanisms can be considered to be part of a continuum where the system should be capable of restructuring itself in either of these states or any intermediate state between the two extremes.

To understand appropriate mechanisms for self-organization in the range from fully decentralized to fully centralized, system wide information on the basis of the individual states of the participating nodes, is needed. Economics may provide one way of doing so. It is an accepted axiom in economic markets that all the available information which may reside at the level of the individual nodes and which is not necessarily shared among them, is consolidated into a simple global metric, named the price. In this paper, we do not address such a self-organizing system but rather the results of this paper can be used for building such a system. This work is part of the research that will address such scalable system that can organize itself according the system status [3]. For instance, a mechanism can be designed to organize the system structure in the continuum between fully centralized to fully decentralized by introducing more/less central

servers whenever it is required. This can be done based on the global state of the network which is presented by the price of resources. This paper studies the price in a market-based resource allocation mechanism which is built based on a centralized peer-to-peer architecture.

3 Related Work

Economic models have been used widely in resource allocation algorithms [4] [5]. Price-based economic models are classified into two main categories: auctions and Commodity Markets. In Commodity Markets, allocations are done based on reaching an equilibrium price where demand equals the supply. For instance, Wolski et al [6] have used the commodity market mechanism to allocate two types of resources (CPU and disk storage) in Grid. The auction protocols are either one-to-many or many-to-many. The strategy in auctions is to grant the resources to the buyers that bid the highest prices. In one-to-many auctions one agent initiates an auction and a number of other agents can make a bid. The English auction, Dutch auction, First-price auction, Second-price (Vickrey auction) belong to this category. Popcorn [7] and Spawn [8] are examples that use these types of auctions. In many-to-many auctions, several agents initiate an auction and several other agents can bid in the auction. The double auction is the most widely used auction protocol for many-to-many auctions. In these auctions, buyers and sellers are treated symmetrically with buyers submitting requests and sellers submitting offers. In the literature, we find several studies on double auction based resource allocation. The works presented in [9], [10], and [11] are examples which use the double auction model. There are two types of double auctions: Continuous Double Auction (CDA) and periodic double auction. CDA matches buyers and sellers immediately on detection of compatible bids. In this type of auction, the transaction prices are set individually for each matched buyer-seller as a function of corresponding seller and buyer prices. Gomoluch et al [12] investigate a market-based resource allocation using CDA and compare it with Proportional Share Protocol and Round-robin mechanism. A periodic version of the double auction instead collects bids over a specified interval of time, then clears the market at the expiration of the bidding interval [13]. Weng et al. [14] present a periodic double auction mechanism with a uniform price for resource allocation in Grids. In this work, auction takes place in rounds and all exchanges are performed with the same price. The Proportional Share Protocol (PSP) is a similar protocol to CDA, as both use a centralized scheduling algorithm. In this mechanism, the amount of resources allocated to a task depends on its price bid in relation to the sum of price bids of all tasks executing on that server. Proportional Share Protocol is proposed for the scheduling of tasks in computational clusters [15].

What distinguishes our work from the others is using a dynamic pricing strategy in which consumer and producer agents are able to use aggressive or conservative bids and adapt to the current condition of the network. The economic mechanism used in this work is not novel but the main novelty is applying

the dynamic pricing algorithm to the Grid environment. Our experiments are performed in a local Grid (a LAN) with different network conditions regarding distribution of tasks and resources. We investigate the pricing behavior of the consumer and producer agents and study the influence of this behavior on the system efficiency in terms of task/resource utilization and average matching time. Eagerness of the agents for contribution to the Grid has been also applied in this strategy by adopting different levels of agent's activity.

4 System Architecture

The system is composed of three entities: Consumer (buyer), Producer (seller) and Auctioneer. The system is modeled as a market and works in the following simple manner: the buyers and sellers announce their desire to buy or sell processing power to the market. The auctioneer finds the matches between buyers and sellers by matching offers (starting with the lowest price and moving up) with requests (starting with the highest price and moving down). When a task query arrives at the market place, the protocol searches all available resource offers and returns the best match which satisfies the task's constraints (such as resource quantity, time frame and price). If no match is found, the task query object is stored in a queue. The queries are kept in the queue until their Time To Live (TTL) expire or matching resources are found. When a resource becomes available and several tasks are waiting, the one with the highest bid price is processed first.

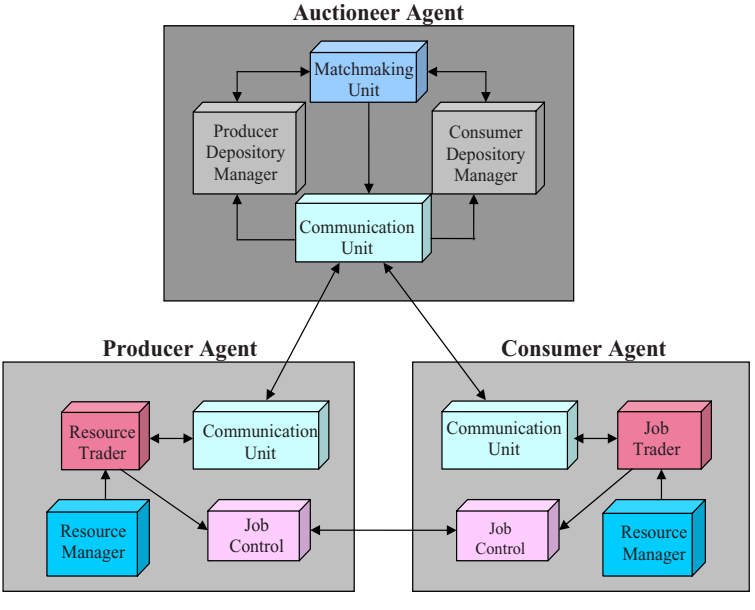

Fig. 1. System components

The system components can be summarized as follows (see Figure 1):

- **Consumer(Buyer)/Producer(Seller) Agent:** Every node in the network can play two different roles either as a consumer or as a producer of resources. A node is a consumer whenever it requests some resources from the Grid, and it is a producer whenever it offers some resources to the Grid. There is one consumer/ producer agent per node. A consumer/producer agent controls the process of buying/selling resources by estimating the execution time of tasks or availability of resources (*Resource Manager*), calculating the price (*Job/Resource Trader*) and generating and submitting a request/offer for the corresponding task/resource (*Communication Unit*). Transferring tasks and results between matched consumer and producer agents (*Job Controller*) is also performed by these agents.
- **Matchmaker(Auctioneer) Agent:** The matchmaker agent is a middle agent between consumer and producer agents. It controls assigning resources to tasks in the network using a matchmaking mechanism. In our model, the matchmaker agent controls the market as an auctioneer using a continuous double auction protocol. Based on this protocol, every consumer and producer sends its request and offer to the auctioneer. Auctioneer inserts each received request or offer in its depositories (*Consumer/Producer Depository Manager*). The requests are sorted from high price to low price and the offers are sorted from low price to high price. A request is matched with an offer if the resource offered by the producer meets the consumer requirements regarding the quantity, time and price (*Matchmaking unit*).

5 Pricing Algorithm

In a price based system, the resources are priced based on the demand, supply, and the wealth in the economic system. In each market, the objective of a seller is to maximize its earning as much as possible and the objective of a buyer is to spend as little money as possible. Based on these objectives, the strategy of resource producers is to raise the price when the demand for associated resource is high and lower the price when the demand is low. On the other hand, the strategy of resource consumers is to lower the price when supply is high and raise the price when the supply is low. Based on these strategies, we define consumer and producer pricing functions through which the consumers and producers perceive the supply and demand in the system and act accordingly. Consumer and producer prices are called respectively bid and asks prices. Consumers and producers start in the market with an initial bid/ask price and update it over the time based on their local knowledge. Each consumer or producer agent keeps a history of its previous experiences in buying or selling resources and defines a bid/ask price as follows:

$$p(t) = p(t-1) + \Delta p \tag{1}$$

Where $p(t)$ is the new price and $p(t-1)$ is the previous bid/ask price offered by the consumer/producer agent. The value of Δp determines whether the price is

increasing or decreasing. To change the price according to the supply or demand in the system, Δp is defined based on the past resource or task utilization for this particular producer/consumer agent. Δp is calculated for a producer and a consumer respectively in Equations 2 and 3 as below:

$$\Delta p = \alpha(u(t) - u_{thR})p(t-1) \tag{2}$$

$$\Delta p = \beta(u_{thT} - u(t))p(t-1) \tag{3}$$

where α and β are the coefficients that control the rate of changing the price. $u(t)$ is resource or task utilization, respectively, at the corresponding producer or consumer agent. $u(t)$ is defined as:

$$u(t) = \sum_{i=t_0}^{t} x(i) / \sum_{i=t_0}^{t} N(i) \tag{4}$$

Where $\sum_{i=t_0}^{t} x(i)$ is the total numbers of sold/purchased resources in the time period $[t_0, t]$ and $\sum_{i=t_0}^{t} N(t)$ is the total numbers of offered/requested resources in the time period $[t_0, t]$ where t is the current time. Based on these definitions, $u(t)$ has always a value between 0 and 1. u_{thT}/u_{thR} are the threshold values below which, the task/resource utilization ($u(t)$) should not go. These values are constant and set by the consumer and producer agents. u_{thT} and u_{thR} could be interpreted as the degree of agent activity. If activity is low, it implies that the agent is satisfied with a low usage of its resources or a low completion rate of its tasks. If it is high, the agent is being more demanding of itself by imposing higher satisfaction thresholds.

Consumers and producers submit their bid/ask prices along with the quantity of requested or offered resources to the auctioneer. The auctioneer finds matched pairs and the trade between each pair is made at the average of the corresponding consumer and producer prices. This price is called **transaction price**.

6 Experiments in a Local Grid

We evaluate the pricing mechanism by simulation. In compared to real-world experiments, simulation models provide control over different system settings, and they can compress time and allow faster execution of experiments. However, they can not resemble the real-world completely and some assumptions have to be made. We constructed a simulation platform in which a Grid like environment is set up based on a local LAN. Our application test-bed is developed using J2EE and Enterprise Java Beans. A JBOSS application server is used to implement the auctioneer. We consider Java Message Service (JMS) for communication between clients and auctioneer. MySQL server is used as a database server to store the results.

Each request or offer submitted by consumers or producers has the following specifications:

- Request={ resource type, resource quantity, TTL (time to live for request validity), bid price}
- Offer={ resource type, resource quantity, TTL (time to live for offer validity), ask price }

CPU time is considered as the resource in our experiments. For simplicity, we consider a reference 1.4GHz CPU based on that consumer agents indicate the quantity of their required resources. For a request, *resource quantity* is indicated in terms of CPU time required to execute a task on the reference CPU. *TTL* in a request is the time by which the task has to be executed after submitting a request. For an offer, *resource quantity* and *TTL* are the same and they represent the time during which the CPU is available. *Resource type* determines the type of requested/offered resource. In our case that CPU time is the resource, resource type determines the CPU speed since experiments are performed in an environment with nodes having various CPU speeds. *Bid/ask* prices are defined using the presented pricing mechanism in Section 5. Consumer and producer agents start in the market with a random bid/ask price between 10 and 30 Grid$ and update their price using the pricing mechanism. The values of resource quantity, speed, and TTL are generated by a uniform random distribution function between a maximum value and a minimum value. We have considered the CPU speed in the range [700MHz, 4GHz], resource quantity for a request in the range $[5000, 20000]$, and TTL for requests or offers in the range $[50000, 100000]$.

The simulation is performed in an environment with 50 nodes each having one consumer agent and one producer agent. Therefore, there exist 50 consumer and 50 producer agents in the network. Some of these agents, called consumers, have tasks to perform for which they are looking for additional resources while others called producers have resources to sell. During a simulation time, every node creates a number of requests and offers in a random order in different time intervals. A node either creates a task request and activates a consumer agent or creates a resource offer and activates a producer agent depending on its workload or its idle resources. An imbalance between number of tasks and resources in the network leads to a task or a resource intensive condition. The experiments are performed under three different network conditions. These network conditions are namely **the balanced network** which is a type of network where there are more or less an equal number of tasks and resources, **the task intensive network** where there are more tasks than resources and **the resource intensive network** where there are more resources than tasks. The tasks and resources are generated with the probability 50%-50% in a balanced condition, 80%-20% in a task intensive condition, and 20%-80% in a resource intensive condition. For instance, a node may creates 20 requests and 80 offers in a random order during the simulation time in a resource intensive condition.

6.1 Adapting to Different Network Conditions

In this section, we want to show how the agents decide on price changes when updating their prices in each network condition. The behavior of the price is

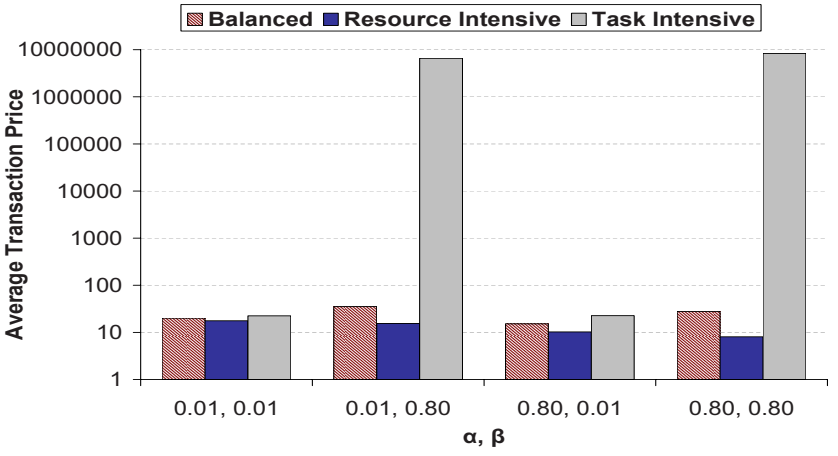

Fig. 2. Average transaction price with various values for α and β in different network conditions (Logarithmic Y-scale)

discussed in these networks and the efficiency of the system is measured in the terms of task and resource utilization and the average time of finding matches. In this set of experiments, we study the impact of α and β parameters in different network conditions. These parameters, as already discussed in Section 5, determine the rate at which the price is changing. We consider the value of $u_{thT} = u_{thR} = 0.9$ for all consumers and producer agents, which means the agents who have tasks or resources are contributing with the same degree of activity in the grid(these parameters are studied in Section 6.2).

Transaction Price. The average transaction price is studied in three network conditions with various values of α and β. The producers increase or decrease the price with the rate of α and consumers increase or decrease the price with the rate of β. As seen from figure 2, the lowest prices are observed in a resource intensive network. This type of network is similar to what is called a buyer market. In a buyer market, there are more sellers than buyers and low prices result from the exceedance of supply over demand. The average transaction price has the highest values in a task intensive network comparing to the other networks. In the task intensive network which has more buyers than sellers (a seller market), high prices are the result of exceedance of demand over supply. In such networks, buyers enter into competition with each other in order to obtain scarce resources. In a balanced network as the supply equals the demand, no very high or low prices are expected.

As seen from figure 2, the average transaction price in a task intensive network increases to very high values when the value of β is high. This is expected, as in task intensive networks, resources are scarce and consumers increase their prices in a competitive way with the rate of β. Therefore, high values of β speeds up the rate of increasing bid prices and leads to the high transaction prices. The high prices can be prevented by applying a budget constraint.

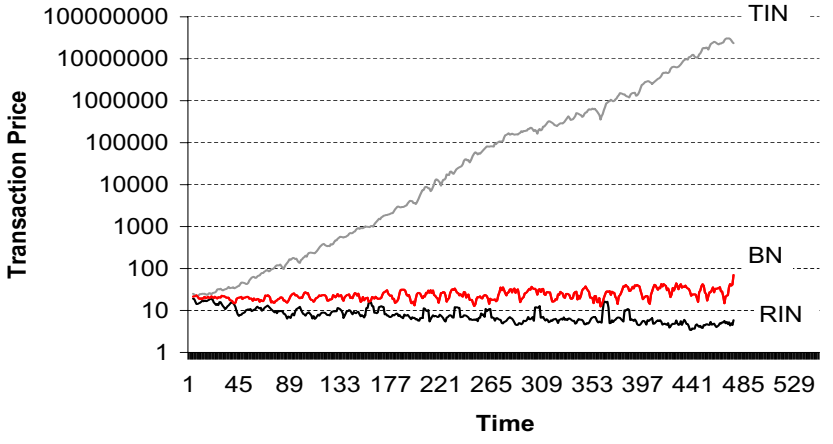

Fig. 3. Transaction price evolution in a Task Intensive Network (TIN), a Resource Intensive Network (RIN), and a Balanced Network (BN) with the values of $\alpha = 0.8$ and $\beta = 0.8$ (Logarithmic Y-scale)

To see how the prices evolve over time under each network condition, transaction price evolution is shown in figure 3 in the three networks with the values of $\alpha = 0.8$ and $\beta = 0.8$. The upward, downward and stable trends of transaction price curve in task intensive, resource intensive and balanced networks respectively is a consequence of what we have already discussed about these markets.

System Efficiency. In this set of experiments, system efficiency is measured in three network conditions with varying the values of α and β.

– **Task/Resource Utilization.** Task/resource utilization is defined as the ratio of allocated tasks/resources to all sent resource requests/offers. As figures 4 and 5 show, task and resource utilization in a balanced network for all values of α and β is around 90%, except when α and β are very low. In case of $\alpha = 0.01$ and $\beta = 0.01$, task and resource utilization are around 75% (in the balanced condition). As with low values of α and β, producer and consumer update their prices in a slower rate that leads to a lower utilization of tasks and resources. In a resource intensive network, a global observation is apparent from the figures as the task completion is close to 100% in most of the cases and only around 25% of the available resources are used. This is expected as we are now looking at a Grid condition where there are abundant resources. In such a network, as figures 4 and 5 show, the highest task/resource utilization is obtained when $\alpha = 0.8$ ($\beta = 0.01$ or $\beta = 0.8$). As in case of high competition between producers, if producers update their prices at a higher rate, they will be more successful in selling their resources. A global view on the task/resource utilization in a task in-

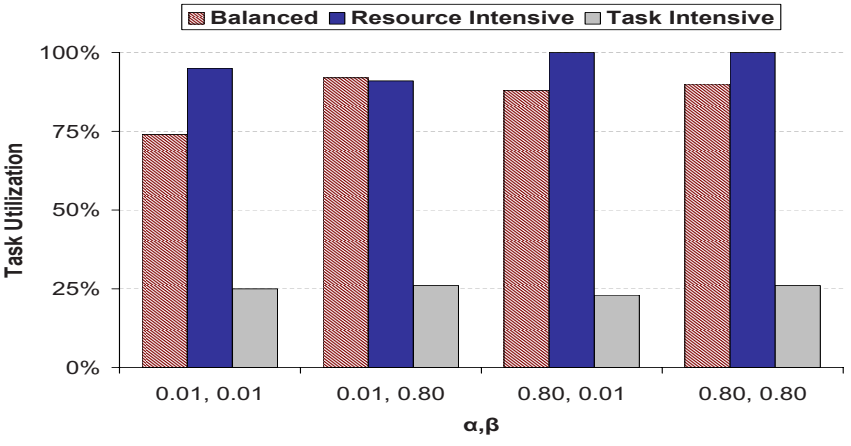

Fig. 4. Task utilization with various values for α and β in different network conditions

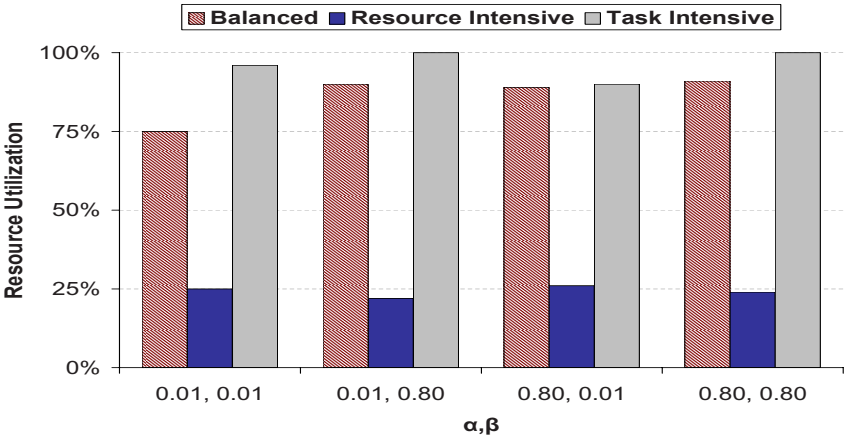

Fig. 5. Resource utilization with various values for α and β in different network conditions

tensive network determines a usage of almost 100% for resources and 25% for allocated tasks. These results are a consequence of higher number of tasks than resources. The highest task/resource utilization in this type of network is obtained when $\beta = 0.8$ ($\alpha = 0.01$ or $\alpha = 0.8$). A higher rate in updating the consumer price helps competitive consumers to find more matches which leads to more task/resource utilization.

– **Average Time of Finding Matches.** Figures 6 and 7 show the average time that it takes for consumers and producers to find their required matches. In a resource intensive network, a global observation is that the average time

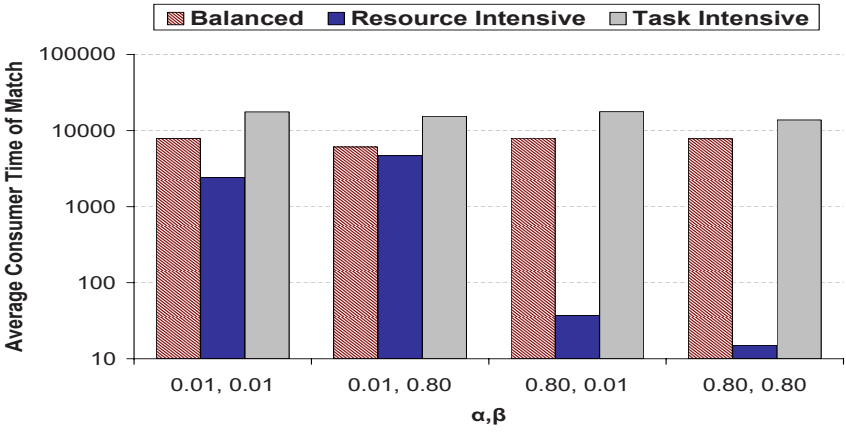

Fig. 6. Average time of finding matches for consumers with various values of α and β in different network conditions (Logarithmic Y-scale)

for producers to find a task for their available resources is at least 4 times higher than for consumers. This is a consequence of the fact that there are more resources available than tasks to perform. In a task intensive network, the average time of finding a match for consumers is at least 6 times higher than for producers. As in this kind of network there are more tasks than resources. However, the lowest consumer matching time in a resource intensive network is obtained when $\alpha = 0.8$ and the lowest producer matching time in a task intensive network is obtained when $\beta = 0.8$, which are corresponding to the highest task/resource utilization in the respective networks.

In a balanced network, we do not see much difference in the average matching time of consumers and producers. However the matching time is higher for both consumers and producers when $\alpha = 0.01$ and $\beta = 0.01$. This is because of a slower rate of updating producer and consumer price that concludes with a longer time for finding proper matches.

An overall study of system efficiency shows that the highest task/resource utilization and lowest matching time in different network conditions is provided when α and β have bigger values. The question is how can agents recognize the current network condition? The answer is that agents can sense the condition through the way their price is evolving. For instance, when the price is increasing, a consumer agent knows that resources become scarce. Therefore it has to adapt its bidding strategy to become more aggressive by increasing the value of β. On the other hand when the price is decreasing, it shows the demand for the resources is low then a producer agent bidding strategy should be converted to a aggressive strategy by increasing the value of α. Lower values of α and β implies a conservative bidding strategy for consumers and producers respectively where they decrease or increase the price in a low rate.

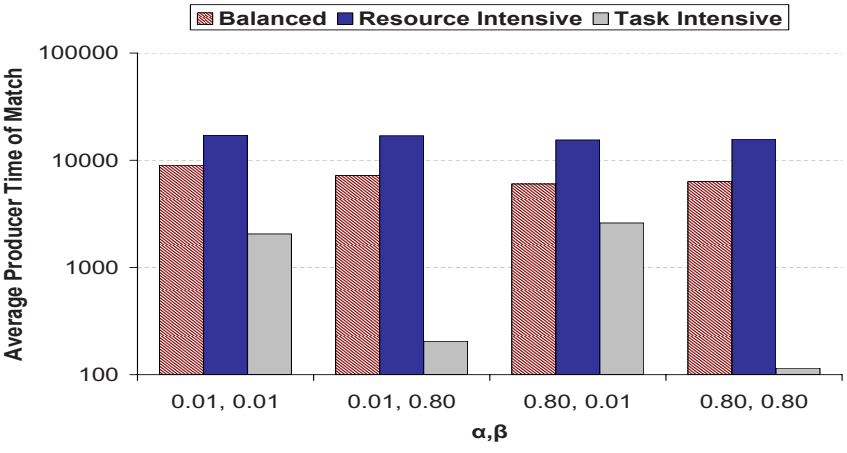

Fig. 7. Average time of finding matches for producers with various values of α and β in different network conditions (Logarithmic Y-scale)

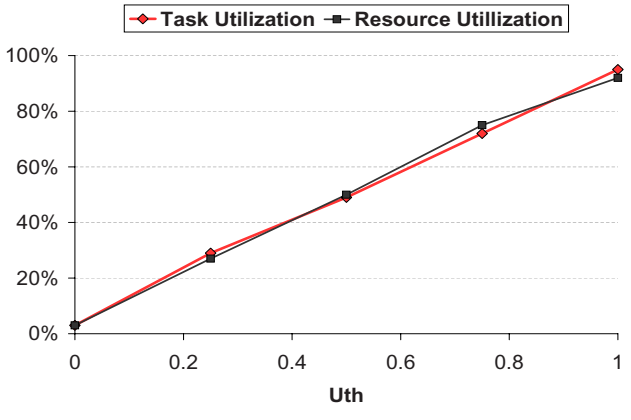

Fig. 8. Task/resource utilization (with different values of u_{th} in a balanced network (u_{th}=u_{thT}=u_{thR})

6.2 Adaptation at the Node Level

In the experiments presented in Section 6.1, we studied the behavior of the price and efficiency of the system in different network conditions. We showed how the agents adapt to the current condition of the network. In the current experiments, we want to show how the agents can adapt based on the current condition of their own tasks and resources. For instance, if a node has more resources than tasks, it should generate an active producer agent and a lazy consumer agent. We need a way to represent this information and to incorporate it into the agent's

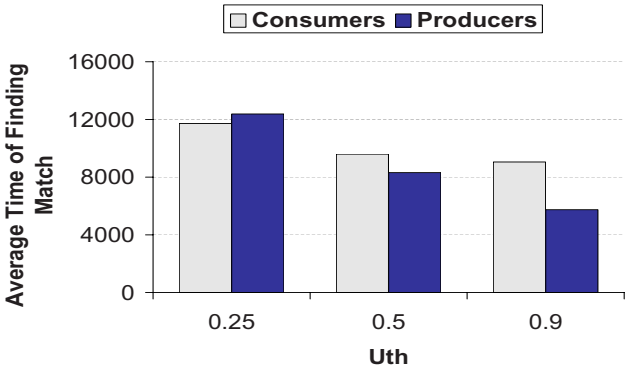

Fig. 9. Average time of finding matches with different values of u_{th} in a balanced network (u_{th}=u_{thT}=u_{thR})

behavior. As already discussed (Section 5), u_{thT}/u_{thR} could be interpreted as the degree of agent activity in the Grid. If it is low, it implies that the agent is contributing with a low usage of its resources or is demanding a low completion rate of its tasks. If it is high, the agent is contributing to the Grid by offering more resources or is demanding more resources from the Grid. To study the impact of u_{thT} and u_{thR}, we consider the fixed value of 0.8 for both α and β.

Impact of u_{thT}/u_{thR} on system efficiency. In the first set of experiments, we study the impact of varying u_{thT} and u_{thR} on system efficiency. We consider the same value of utilization threshold for the consumers and producers ($u_{thT} = u_{thR} = u_{th}$) and perform the experiment in a balanced network.

- Task/Resource Utilization: We measure the task and resource utilization in the network considering different values for u_{th}. The result of these experiments shows that task and resource utilization is linearly proportional to this threshold value (see figure 8). Agents with low value of u_{th} represent lazy agents and agent with high value of u_{th} show the agents which are active in Grid. Seen from the figure 8, as u_{th} increases, the Grid utilizes more from the agent's tasks or agent's resources.
- Average Time of Finding Matches: In the same experiment, the average time that it takes to find a match is measured for both producers and consumers. With increasing degree of the activeness (u_{th}), the agents become more active in the Grid, so the time to find a match for them is decreasing. As Figure 9 shows, with increasing the value of u_{th}, producers and consumers spend less time to find their required matches.

Lazy/active agents. To show in a Grid how consumer and producer agents can become lazy or active by modifying u_{thT} and u_{thR} parameters, we undertake one other experiment. Assume that in a Grid some nodes have heavy workloads and

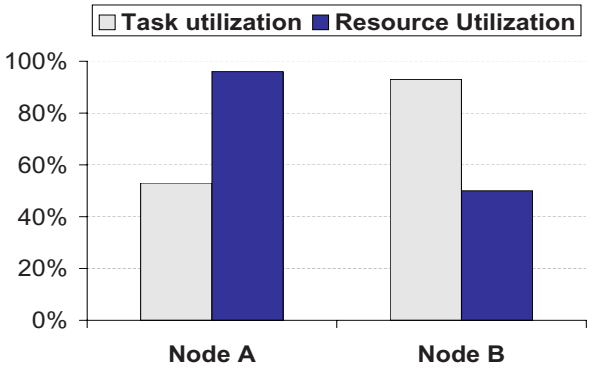

Fig. 10. Task/resources utilization for lazy/active agents. Node type A: $u_{thR} = 0.9$ and $u_{thT} = 0.25$; Node type B: $u_{thR} = 0.25$ and $u_{thT} = 0.9$.

need extra resources. These kinds of nodes prefer to complete their tasks rather than offering their resources, so these nodes can be regarded as lazy producers but active consumers. Other nodes are more willing to contribute their resources as they have idle resources or low workloads. That means that these nodes are active producers but lazy consumers. In this experiment, we consider 40 nodes in the network. The values of $u_{thR} = 0.9$ and $u_{thT} = 0.25$ are set for 20 nodes which are assumed to be active producers and lazy consumers. Other half of the nodes have values that are set to $u_{thR} = 0.25$ and $u_{thT} = 0.9$ to create active consumers and lazy producers. We consider a balanced condition where each node generates more or less the same number of tasks and resources during the experiment. We study the task and resource utilizations of the individual nodes from these two categories. Figure 10 shows the average of resource and task utilizations of two typical nodes from each category. Node A is an instance of the first category with low workloads which has more idle resources, and Node B is an instance of the nodes with high workloads. As seen in Figure 10, nodes of type A contribute more as producers than as consumers and Grid utilizes more resources (93%) from this group compared to the tasks (56%). On the other hand, more tasks are utilized from the nodes of type B compared to the resources, which is 96% for the tasks and 59% for the resources. These nodes contribute to Grid more as consumers than as producers. Therefore, consumer and producer agents can decide on their task usage and resource contribution to Grid individually by setting the parameters u_{thT} and u_{thR}.

7 Conclusion and Future Work

In this paper, a dynamic pricing and bidding strategy is introduced where the consumer and producer agents determine the price of the tasks/resources that they contribute to the Grid. In this strategy, the pricing function is adaptive to

changing supply and demand of resources. Adaptation is achieved by increasing or decreasing the price when the supply or demand is low. For instance, if the demand for resources are high, prices start to increase so as to discourage users from demanding resources thus maintaining equilibrium of supply and demand of resources. We study the parameter regime of the pricing equations in three network conditions. There are four parameters that can be manipulated by the consumer and producer agents: α, β, u_{thT} and u_{thR}. The parameters α and β are used to define the rate of changing ask and bid prices. The parameters u_{thT} and u_{thR} determine the degree of activity of agents in the Grid.

Our experiments show that a resource intensive network is more influenced by α while a task intensive network is more influenced by β. In a resource intensive network, agents can decrease their asking prices using a aggressive bidding strategy by increasing the α-value. In a task intensive network, agents can speed up their bidding prices by increasing the β-value using an aggressive bidding strategy to make more use of the Grid. In a balanced network both α and β parameters have the same affect. Furthermore, producers and consumers can change their degree of activity in the Grid using u_{thR} and u_{thT} parameters. The producers and consumers can decide how much they are contributing to the Grid considering their capabilities and their workloads. They become less/more active in the Grid by decreasing/increasing these values.

As the results show, in all conditions higher values of α or β provide higher resource/task utilization and lower matching time. On the other hand, high values of β causes transaction prices to grow infinitely in a task intensive condition. Infinite prices do not exist in real markets since buyers can not bid beyond their limited budget. To avoid unlimited prices and to be consistent with real markets, in future work we aim to implement our model considering a given budget for each node which provides an upper boundary for prices. Different auction models with different pricing strategies are to be studied in the future work.

References

1. Minowitz, P.: Adam smith's invisible hands. Econ. Journal Watch 1, 381–412 (2004)
2. Pourebrahimi, B., Bertels, K., Vassiliadis, S.: A survey of peer-to-peer networks. In: Proceedings of the 16th Annual Workshop on Circuits, Systems and Signal Processing, ProRisc 2005 (2005)
3. Abdullah, T., Sokolov, V., Pourebrahimi, B., Bertels, K.: Self-organizing dynamic ad hoc grids. In: 2nd IEEE International Conference on Self-Adaptive and Self-Organizing Systems Workshops, SASOW 2008 (2008)
4. Wolski, R., Brevik, J., Plank, J.S., Bryan, T.: Grid resource allocation and control using computational economies. In: Grid Computing: Making The Global Infrastructure a Reality. John Wiley & Sons, Chichester (2003)
5. Buyya, R., Abramson, D., Giddy, J., Stockinger, H.: Economic models for resource management and scheduling in grid computing. Concurrency and Computation: Practice and Experience 14, 1507–1542 (2002)
6. Wolski, R., Plank, J., Brevik, J., Bryan, T.: G-commerce: Market formulations controlling resource allocation on the computational grid. In: Proceedings of International Parallel and Distributed Processing Symposium (IPDPS) (2001)

7. Nisan, N., London, S., Regev, O., Camiel, N.: Globally distributed computation over the internet - the popcorn project. In: Proceedings of the 18th International Conference on Distributed Computing Systems (ICDCS 1998), p. 592. IEEE Computer Society, Los Alamitos (1998)
8. Waldspurger, C.A., Hogg, T., Huberman, B.A., Kephart, J.O., Stornetta, W.S.: Spawn: A distributed computational economy. Software Engineering 18, 103–117 (1992)
9. Lalis, S., Karipidis, A.: Jaws: An open market-based framework for distributed computing over the internet. In: Buyya, R., Baker, M. (eds.) GRID 2000. LNCS, vol. 1971, pp. 36–46. Springer, Heidelberg (2000)
10. Preist, C., Van Tol, M.: Adaptive agents in a persistent shout double auction. In: Proceedings of 1st International Conference on the Internet Computing and Economics, 1998, pp. 11–17 (1998)
11. Ogston, E., Vassiliadis, S.: A peer-to-peer agent auction. In: Proceedings of the first international joint conference on Autonomous agents and multiagent systems Part I, pp. 151–159 (2002)
12. Gomoluch, J., Schroeder, M.: Market-based resource allocation for grid computing: A model and simulation. In: Proceedings of the First International Workshop on Middleware for Grid Computing, Rio de, pp. 211–218 (2003)
13. Wurman, P., Walsh, W., Wellman, M.: Flexible double auctions for electronic commerce: Theory and implementation. Decision Support Systems, 17–27 (1998)
14. Weng, C., Lu, X., Xue, G., Deng, Q., Li, M.: A double auction mechanism for resource allocation on grid computing systems. In: Jin, H., Pan, Y., Xiao, N., Sun, J. (eds.) GCC 2004. LNCS, vol. 3251, pp. 269–276. Springer, Heidelberg (2004)
15. Waldspurger, C.A., Weihl, W.E.: Lottery scheduling: Flexible proportional-share resource management. In: Operating Systems Design and Implementation, pp. 1–11 (1994)