Adaptation to Dynamic Resource Availability in Ad-hoc Grids through a Learning Mechanism

Behnaz Pourebrahimi, Koen Bertels
Computer Engineering Laboratory, Delft University of Technology, The Netherlands
{b.pourebrahimi, k.l.m.bertels} @tudelft.nl

Abstract

Ad-hoc Grids are highly heterogeneous and dynamic networks, one of the main challenges of resource allocation in such environments is to find mechanisms which do not rely on the global information and are robust to the changes in resource availability in Grid. In this paper, we present a learning algorithm in a market-based resource allocation platform. Using this algorithm, consumer and producer agents learn the current condition of the network through their previous reward from the Grid and decide the preferred prices only based on their local knowledge. In our history-based pricing strategy, we introduce two reinforcement parameters using which the consumer and producer agents employ an aggressive or a conservative bidding strategy. Aggressive and conservative bidding strategies reinforce adaptation to the variations of resource availability in the ad-hoc Grids. Comparing our mechanism with a learning and a non-learning mechanism shows that our approach besides providing adaptable prices to the dynamic condition of the network, it also presents higher throughput.

1 Introduction

Adaptation means adjusting to the new condition. Since human operators are costly, slow and error-prone, self-adaptable systems are needed that are able to cope with resource variability, changing user needs and system faults. When the system is known beforehand, adapting to the current condition can be performed through the static information about the system. In dynamic systems where the condition of the system changes unpredictably during the time, the static information is not enough and information has to be updated. Ad-hoc Grids are instances of such systems. Ad-hoc Grids are highly heterogeneous and dynamic networks in which the availability of resources and tasks may change at any time. In ad-hoc Grids, every node in the network can act as a consumer or a producer of resources at any time when there is a need for resource or there is a resource available. In such dynamic condition, there is no global information available and decision-making process is distributed across all users and resource owners. Where there is no global information about the current condition of the network available, a way to learn about the network condition is through interaction with environment. Learning from interaction is a foundation idea underlying nearly all theories of learning and intelligence[10].

In this paper, we introduce a learning mechanism for the market-based resource allocation in ad-hoc Grids. This learning algorithm enables consumer and producer agents to adjust their prices by perceiving the current condition of the network through the interaction with environment. We assume that the agents decide based on their local knowledge and they are not aware of the other's decisions. They learn the condition of the network using the reward which they get from the environment and take a proper action to adapt to the current condition (see Figure 1). The reward is their utilization from the Grid and the action is setting the new ask or bid price. Pricing algorithm proposed in this paper introduces two reinforcement parameters which are set dynamically as the condition of the network changes. These parameters reinforce adaptation to the changes in the network condition and apply an aggressive or a conservative bidding strategy accordingly. In experimental results, we study the behavior of our learning mechanism in a dynamic network condition where the availability of resources changes during the time. We compare our learning mechanism with a non learning ZI (Zero Intelligence) and a learning ZIP (Zero Intelligence Plus) approach. The experimental results show that our approach provides more adaptability to changes of network condition and presents higher throughput compared with two others approaches.

The paper is structured as follows: In section 2, we have an overview on related work. System architecture is presented in section 3. We describe the learning mechanism with discussing bidding strategies and pricing algorithm in section 4. Performance evaluation is studied in section 5 and finally we have conclusion in section 6.

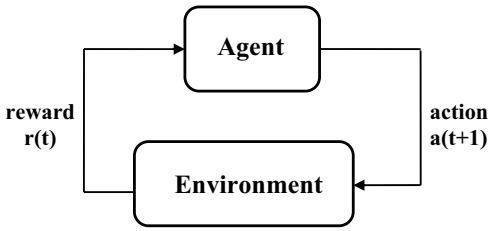

Figure 1. The agent - environment interaction in learning mechanism

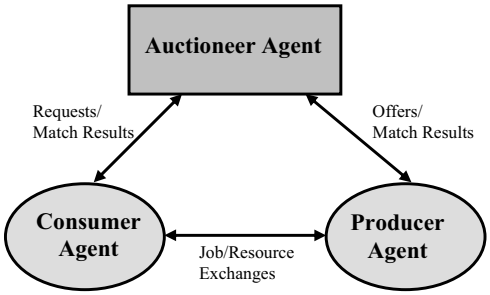

Figure 2. System Components

2 Related Work

There are several research efforts on Grid adaptability. For instance, [3] and [9] assume a centralized network architecture where a centralized server collects environment conditions from each component to make replacement decisions. Lee et al [7] present a biologically-inspired mechanism that allows grid services to autonomously adapt to dynamic changes in the network. The proposed adaptation mechanism, called the iNet artificial immune system, allows each grid service to autonomously sense its local environment conditions to evaluate whether it adapts well to the conditions, and if it does not, adaptively performs a behavior suitable for the conditions. [6] addresses adaptive query processing on the Grid in order to cope with evolving resource characteristics, such as machine load and availability.

Considering Market-based resource allocation, although there has been considerable attention given to the economic based resource allocation in the Grid [2, 11], but very few researches have addressed the problem of learning and adaptation. Preist et al [8] demonstrate the simple adaptive agents inspired by the ZIP agent work of Cliff & Bruten [4], which consist of a small number of heuristics and a simple learning rule. These agents learn to trade at an equilibrium price in a form of periodic double auction marketplace. [1] proposes an adaptation of two learning mechanisms (Zero Intelligence Plus & Gjerstad-Dickhaut) for single sellers in first price and second price auctions. Non of these researches have addressed the adaptation in ad-hoc Grid to support simultaneous participation of multiple sellers and buyers and accommodate the variations in availability of resources. In the most of economic based researches for the resource allocation in Grid, resource are assumed to be dedicated with a fixed number of servers which provide resources. In ad-hoc Grid, resources are not dedicated and their availability fluctuates during the time. In this work, we present an economic-based approach to perform adaptability to changes in the resource availability in ad-hoc Grids. We consider producer and consumer of resources as the selfish agents which can spontaneously rise and submit an offer or a request whenever a node has some idle resources or needs some extra resources. We use a continuous double auction protocol for matchmaking between these competitive consumers and producers. In our model, no global and single equilibrium price is computed. We propose a learning mechanism using which the producer and consumer agents compute the ask or bid prices and transaction price between each matched consumer and producer is calculated as the average of two corresponding prices.

3 System Architecture

The system is composed of three agents **Consumer (buyer)**, **Producer (seller)** and **Auctioneer** (see figure 2). There is one consumer and one producer agent per node. The consumer/producer agent controls the process of buying/selling resources by estimating the execution time of the job or availability of the resource. This agent also calculates the price and submits a request/offer for corresponding task/resource to the auctioneer. The consumer/producer agent is also receiving the response for the submitted request/offer from the auctioneer. The auctioneer agent controls the market using a Continuous Double Auction Protocol.

For resource allocation in ad-hoc Grids, we need a mechanism that supports simultaneous participation of producer/consumer, observes resource/request deadlines and can accommodate the variations in resource availability. Sequential and open-cry auctions are not a proper choice. As tin these type of auctions, each bid has to be broadcasted to all participants. This becomes a considerable communication overhead in the context of ad-hoc Grids. Moreover, their inability to observe time deadlines and no support for the simultaneous participation of producer/consumer are the

reasons that make them unsuitable for resource allocation in ad hoc Grid. We employ a Continuous Double Auction (CDA) as the platform for matchmaking between consumers and producers. CDA is a many-to-many auction that provides a suitable resource allocation platform for multiple sellers and buyers in ad-hoc Grid. We adopt a discriminatory pricing policy to set the transaction prices. In discriminatory policy, there is no global equilibrium price and the prices are set individually for each matched buyer-seller pair. The transaction price for each matched pair is considered as the average of two prices. The market works in the following simple manner: the buyers and sellers announce their desire to buy or sell resources to the market. The auctioneer finds the matches between buyers and sellers by matching offers (starting with lowest price and moving up) with requests (starting with 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 tasks constraints which are 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 till the time to live (TTL) for them is expired or a match is found. When a resource becomes available and several tasks are waiting, the one with the highest price bid is processed first.

4 Learning Mechanism

In ad hoc Grid where the resource availability and the workloads are variable, the condition of supply and demand may change at any time. We introduce a learning mechanism by which the agents perceive the supply and demand in the system through their previous utilization from the Grid. In our learning mechanism, we define two aggressive and a conservative bidding strategies. Using these strategies, consumer and producer agents change their prices in a competitive way according to the condition of supply and demand.

4.1 Aggressive Bidding

Aggressive bidding is defined as a strategy using which agents increase or decrease their prices in a high rate. Aggressive bidding strategy is adopted by consumers when there is a high competition between consumers due to the high demand for resources in the network. The producers adopt aggressive bidding strategy when due to the high supply in the system, there is a high competition between producers. With adopting aggressive bidding, consumers and producers respectively speed up increasing or decreasing their prices to outbid their competitors.

4.2 Conservative Bidding

A bidding is conservative when agents change their prices in a low rate. Conservative bidding strategy is employed by consumers where the supply in the system is high. In this case, consumers decrease their prices to reduce their spending. They decrease the price in a low rate conservatively to not loose the competition. In the same way, when the demand is high, producers increase their prices to increase their credit. They increase the price by a conservative strategy in a low rate not to be surpassed by other producers.

The aggressive and conservative bidding strategies are taken in the networks where there is not a balance between the supply and the demand. In such condition, when one of the consumers or producers parties employ the aggressive or conservative bidding, the other party employs the conservative or aggressive bidding respectively. In following section, we present a pricing algorithm using which the agents can take a proper bidding strategy according to availability of resources and tasks in the network.

4.3 Pricing Algorithm

Consumers and producers determine their bid and ask prices using a learning pricing strategy. The pricing strategy presented simulates human intelligence to define a logical price by local analysis of the previous trade cases. The price is defined as the price of each unit of resource that consumer and producer agents are willing to buy or sell. There is an upper limit for consumer prices (bid prices) which is defined by the node's budget, as $bidPrice{*}resourceQuantity <= Budget$. Where $resourceQuantity$ is the quantity of the resource needed. For instance, it is job execution time when CPU time is considered as the resource. We also consider a minimum price for producers below which they do not sell their resources. Using the learning algorithm, the agents update the price considering their previous utilization from the Grid. If they have not been successful in buying or selling the resources , they need to update the price in a way to get more utilization in future. On the other side, if they have been successful, they conservatively continue the way of their bidding. Consumer and producer agents start in the market with their private values which is limited by minimum and maximum prices and update it over the time using the following learning pricing algorithm. Based on this algorithm, the ask and bid price are defined respectively for producers and consumers as follows:

$$p_a(t) = max\{p_{min}, p_a(t-1) + \Delta p_a\} \quad (1)$$

and

$$p_b(t) = min\{p_{max}, p_b(t-1) + \Delta p_b\} \quad (2)$$

where $p(t)$ is the new price and $p(t-1)$is the previous price. p_{min} is the minimum price for producers and p_{max} is the

maximum price affordable for consumers which is calculated as:

$$p_{max} = Budget/resourceQuantity \quad (3)$$

Δp for seller and buyer is defined respectively as below:

$$\Delta p_a = \alpha.p_a(t-1) \quad (4)$$

and

$$\Delta p_b = \beta.p_b(t-1) \quad (5)$$

where α and β are coefficients which determine the rate at which the price is increasing or decreasing. We name them as **reinforcement parameters**, as they reinforce the learning and apply two **aggressive** and **conservative** bidding strategies. These parameters are set according to variations in task or resource utilization during the time at each individual node. For each individual node, task utilization is defined as the ratio of allocated tasks to all submitted requests and resource utilization is defined as the ratio of allocated resources to all submitted offers. We define task and resource utilization in the time period $[t_1, t]$ as:

$$ResUtil(t-t_1) = \sum_{i=t_1}^{t} S(i) / \sum_{i=t_1}^{t} N_{offers}(i) \quad (6)$$

$$TaskUtil(t-t_1) = \sum_{i=t_1}^{t} P(i) / \sum_{i=t_1}^{t} N_{requests}(i) \quad (7)$$

where $\sum_{i=t_1}^{t} S(i)$ and $\sum_{i=t_1}^{t} P(i)$ are respectively the total numbers of sold and purchased resources and, $\sum_{i=t_1}^{t} N_{offers}(i)$ and $\sum_{i=t_1}^{t} N_{requests}(i)$ are respectively the total numbers of offered and requested resources at the time period $[t_1, t]$ in each individual node. The variation in resource and resource utilization is presented by the terms $\Delta ResUtil$ and $\Delta TaskUtil$ which are defined between time periods $[t_2, t_1]$ and $[t_1, t]$, where t is the current time:

$$\Delta ResUtil = ResUtil(t-t_1) - ResUtil(t_1-t_2) \quad (8)$$

$$\Delta TaskUtil = TaskUtil(t-t_1) - TaskUtil(t_1-t_2) \quad (9)$$

Now, we define reinforcement parameters α and β as follows:

$$\alpha = \begin{cases} -(K-(ResUtil(t-t_1))^2)^2 & if\ \Delta ResUtil <= 0 \\ L*(ResUtil(t-t_1))^2 & if\ \Delta ResUtil > 0 \end{cases} \quad (10)$$

$$\beta = \begin{cases} (K-(TaskUtil(t-t_1))^2)^2 & if\ \Delta TaskUtil <= 0 \\ -L*(TaskUtil(t-t_1))^2 & if\ \Delta taskUtil > 0 \end{cases} \quad (11)$$

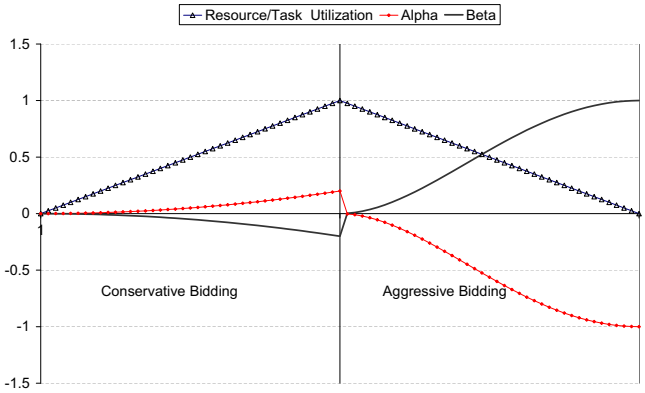

Figure 3. Reinforcement parameters (α and β) and task/resource utilization.

where K and L respectively define the maximum rate of aggressive and conservative bidding. In Our experiments, we have considered $K = 1$ and $L = 0.1$. It means that in an aggressive bidding, the maximum rate of increasing price by consumers or decreasing price by producers is 1. In the same way in a conservative bidding, the maximum rate of increasing price by producers or decreasing price by consumers is 0.1. These values can be changed individually by nodes based on the user preferences such as budget consumption, urgent tasks and etc. For instance, if a node has very limited budget, it has to choose lower level for aggressive bidding and avoid very high bid prices. On the other hand, in case of very urgent tasks, nodes can always increase level of their aggressive bidding or decrease level of their conservative bidding.

We show the theoretical analysis of these parameters using the figure 3. In this figure, the aggressive and conservative bidding strategies are presented with depicting parameters α and β together with the variation in task or resource utilization.

- **Conservative Bidding**: producer or consumer agents bid with conservative strategy when the resource or task utilization is increasing (see figure 3). In a conservative bidding, the rate of increasing price by producers (α) and the rate of decreasing price by consumers (β) are proportional to the value of recent task/resource utilization (equations 10 and 11). From figure 3, we can observe lower rate for low utilizations and higher rate for higher utilizations. The maximum rate of increasing or decreasing price is 0.1 ($L = 0.1$) where the task or resource utilization has its maximum value ($Util = 1$).

- **Aggressive Bidding**: when the resource or task utilization is decreasing, producer or consumer agents employ an aggressive bidding strategy (see figure 3). In an aggressive bidding, the rate of increasing price by consumers (β) and the rate of decreasing price by producers (α) depends on the value of recent task/resource utilization (equations 10 and 11). As seen from the figure, the lower rate are observed for high utilizations and higher rate for lower utilizations. The maximum rate of increasing or decreasing the price is 1 ($K = 1$) where the task/resource utilization has its minimum value ($Util = 0$).

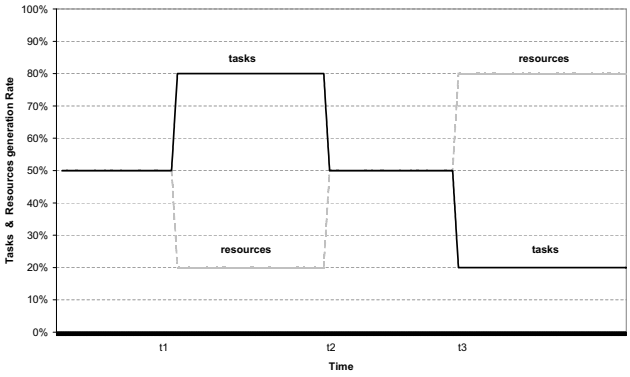

Figure 4. The generation rate of tasks and resources.

5 Performance Evaluation

In this section, we study the performance of our learning algorithm in an ad hoc Grid within a dynamic condition. Our application test-bed is developed using J2EE and Enterprise Java Beans. Auctioneer is deployed on JBoss application server. Consumer and producer of resources are buyers and sellers in the market. Whenever a node needs computational service for running its tasks, it sends a request to the auctioneer through the consumer agent and whenever a node has some computational service available, it sends an offer through the producer agent. We assume the tasks are atomic and can not be divided, therefore each request is matched with only one offer. 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 , budget}
- Offer={ resource type, resource quantity, ttl (time to live for offer validity), ask price }

The experiments are performed in a local ad-hoc Grid with 60 nodes. So, there are 60 consumer agents and 60 producer agents participating in the market. The number of created requests and offers in the network varies during the time (see section 5.1). Each request contains: resource requirement in the term of cpu time, a ttl (time to live) to determine the time during which the task has to be executed, a price that consumer is able to pay for each unit of resource, and the total amount of the budget that consumer has. An offer includes: a ttl which is the time during which the cpu is available, the type of cpu and a price below which the producer does not sell the resource. In matchmaking between consumers and producers, auctioneer considers not only the price but also the quantity of the resource and ttl constraints. TTLs and task execution times are generated randomly for each request and offer. All nodes are assigned equal budgets when joining the grid. The limited budget defined for each node can be used to trade for required resources. The nodes earn credits by devoting the idle computational resources for demanding consumers.

5.1 Experimental Condition

The results shown in this paper are from the experiments performed during four time periods with different distribution of tasks and resources. Resources and tasks are generated with the ratios 20%, 50%, and 80% in different time periods. Figure 4 depicts the system status evolving over time. In period [0,t_1] a balanced condition with more or less equal number of tasks and resources is generated. From the time $t1$ to $t2$ the number of tasks are increasing, it could simulate the busy hours during the working days (a task intensive condition). From time $t2$ onward, the tasks start decreasing and again we have a balanced condition. The resources start to increase from the time $t3$ onward and a resource intensive condition is generated which could simulate the situations with low workloads and high idle resources during the night or weekend.

In following sections, we study behavior of system in different conditions.

5.2 Reinforcement Parameters

We analyzed the parameters α and β theoretically in section 4.3. In this section, we study these two parameters practically when running experiments in the dynamic condition described in the previous section. Figure 5 shows reinforcement parameters evolving during the dynamic condition of the network. As seen from the figure, in the time periods [0, t_1] and [$t2$,$t3$], where there is a balance situation between tasks and resources, the consumer and producer agents do not take any aggressive bidding. However,

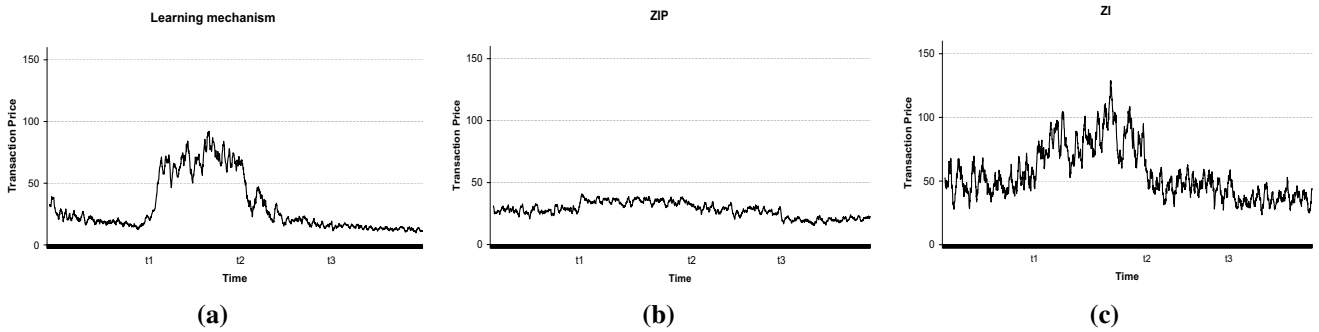

Figure 6. Transaction price evolution (a)our learning approach, (b)ZIP, (c)ZI.

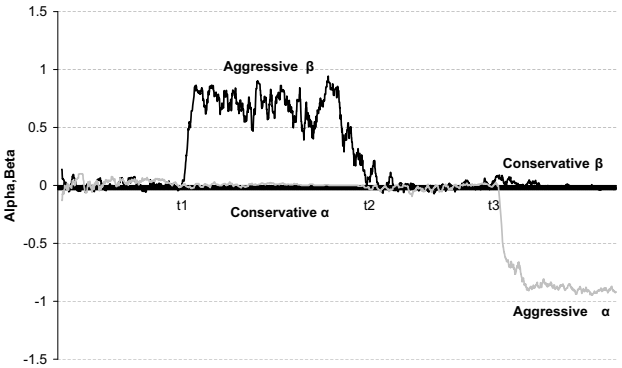

Figure 5. Reinforcement parameters variations during the experiment.

they increase or decrease their prices more conservatively in this condition. In time period $[t1, t2]$, where there is a task intensive condition, the consumer agents take an aggressive bidding strategy (aggressive β)and on the other side producers employ a conservative bidding (conservative α). As In such network condition, consumers are abundant and they have to increase their prices aggressively to win the competition. In task intensive network, producers increase their prices conservatively to increase their credit. When a resource intensive condition is created in the time period $t3$ onward, resources become abundant. Therefore, producers take an aggressive strategy (aggressive α) for decreasing their prices. In such condition, consumers take a conservative strategy (conservative β) to decrease their prices.

As discussed before(section 4.3), the maximum rate for aggressive and conservative bidding can be set by each agent individually. This is feasible by modifying the constant coefficients K and L provided in equations 10 and 11. These values can be modified according to consumer/producer preferences. For example, a node for which there is no worry about the budget consumption and it can afford high transaction prices, the level of aggressive bidding can be increased and vise versa. Our mechanism provides autonomy for consumer and producer agents to decide about the level of their aggressive or conservative bidding.

5.3 Price Adaptation

Consumer and producer agents adapt to the new condition of the network by learning from their previous trades. They accordingly adopt a proper bidding strategy and modify their ask and bid prices. The adaptation can be observed with studying the evolution of transaction prices as the condition of the network changes. Transaction price is the price paid by consumer to producer for each unit of resource. The transaction price is considered as the average of bid and ask prices.

In this section, we study the transaction price evolution in a dynamic network and compare our mechanism with a learning and a non-learning mechanism. We consider ZIP (Zero Intelligence Plus) approach as learning and ZI (Zero Intelligence) as a non learning approach to compare with. In ZIP (Zero Intelligence Plus) approach [1], the agents learn a bidding strategy based on information provided by the results of their previous shout and their private value. A ZIP agent adopts a linear bid function given by:

$$b(t) = x * (1 - \mu(t)) \tag{12}$$

Where μ represents the fraction above or below its value at which the agent bids and x is the private value. In this approach, the problem of learning an optimal strategy is reduced to learning the optimal margins. The margin $\mu(t)$ is adjusted to $\mu(t+1)$ using the rule:

$$\mu(t+1) = \mu(t) + \Delta(t) \tag{13}$$

Where $\Delta(t)$ is calculated based on the previous bid. For a detailed presentation of this approach, please refer to [1].

In [1], ZIP has been used to compute bid prices for the single seller auctions. We compute ask prices with the same strategy using this approach. As a non-learning algorithm, we consider ZI (Zero Intelligence) approach [5]. In this strategy, agents generate random order prices, ignoring the state of the market. We consider a ZI-C (Zero Intelligence-Constraint) model in which traders are subject to a budget constraint and consumer bids and producer offers are limited between a maximum and a minimum price.

Within a completely similar condition, we perform our experiments for the three approaches (our method, ZIP and ZI). In all the three approaches, every node receives an equal amount of budget when it joins the Grid. The bid and ask prices are limited between a minimum and a maximum price. There is a minimum price below which producers do not submit an ask. We consider this minimum price equal to one for all producers in the network. The maximum price for consumers is limited by their budget. Each consumer or producer agent has a private value. The private values are generated with a random function in the range between minimum and maximum prices. Private values in our approach are the prices with which agents start in the market and private values in ZIP approach are used to calculate the bid and ask prices (equation 12). We study the transaction prices for matched consumers and producers during the time in each approach.

Figures 6(a), 6(b) and, 6(c) show how transaction prices are evolving as the condition of the network changes. As seen from the figure 6(a), in our approach, the transaction price is evolving according to the availability of resources in the network. In the time period [$t1$,$t2$] where resources are becoming scarce, we observe high transaction prices. Because in this condition, consumers become more competitive and with an aggressive bidding strategy increase their bid prices. Therefore, transaction prices increase. From the time t2 onward, as resources are getting more available, the transaction prices start to decrease. As in this condition, producers become competitive and decrease the ask prices and in consequence transaction prices go down. In ZIP approach (see figures 6(b)), transaction price is changing very slightly as the condition of network change. ZI approach, in which the bid and ask prices are generated randomly, shows high values and high variations in transaction prices in all conditions (figure 6(c)).

With comparing the transaction price evolution in three approaches, we observe that in our approach the consumer and producer agents can adapt their prices to the condition of the network more than in two other approaches. In the next section, we study if our approach keeps a promising throughput for the system.

	Resource Utilization	Task Utilization
Learning Algorithm	%61	%56
ZIP	%48	%44
ZI	%56	%52

Table 1. Task and resource utilization in the three approaches.

5.4 Throughput

We measure the throughput of the system regarding task and resource utilization and compare three methods in this regard. Task utilization is defined as the ratio of allocated tasks to the all submitted requests in the system. The ratio of allocated resources to all submitted offers in the system is defined as resource utilization. Table 1 shows the task and resource utilization in three approaches with performing the experiments in the dynamic network condition described in section 5.1. Having higher resource utilization than task utilization is due to unbalanced number of generated tasks and resources during the simulation. Results show that our approach provides highest task/resource utilization. The task and resource utilization in our approach is around %4 more compared to ZI approach and %12 more compared to ZIP approach. Although, the simple and non intelligence approach ZI provides a throughput comparable with our approach, but in the same time it presents high and non-stable transaction prices (figure 6(c)). ZIP approach presents lower and more stable transaction price (figure 6(c)), but it gives a low throughput. It can be concluded that our approach besides providing a promising throughput, it presents adaptable transaction prices. As in our approach, transaction prices are low and stable when the resources are available, and are increasing when the resources become scarce.

6 Conclusion

In this paper, we propose a learning algorithm to adjust the asks and bids prices according to availability of resources in an ad-hoc Grid. In the proposed pricing algorithm, we introduce two parameters that reinforce learning and adaptation. These parameters are set dynamically according to the network condition. Using these parameters, the consumer and producer agents employ aggressive or conservative bidding strategy to adapt to the dynamic availability of resources. In this strategy, nodes are also able to control the level of aggressiveness or conservativeness individually based on their preferences and requirements when bidding the prices.

We study the behavior of our learning algorithm in a dy-

namic ad hoc Grid in which the supply of resources and demand for resources is variable. We assume there is no global information about the supply and demand of the resources available and agents do not exchange any information and just relay on their local knowledge. They learn the condition of the network through the interaction with the environment using their previous experiences. The consumer and producer agents decide for the price and their bidding strategy based on the their utilization rewarded by the Grid. We compare our algorithm with a learning approach (ZIP) and a non-learning approach (ZI). Studying transaction prices in the three approaches shows that using our approach, the consumer and producer agents can adjust their prices according to the availability of resources and tasks. ZI approach shows high variations and high values in prices in any network condition and ZIP shows very slight changes in prices as the condition of the network changes. Moreover, our approach provides higher throughput compared to the two other approaches.

References

[1] A. J. Bagnall and I. E. Toft. Zero intelligence plus and gjerstad-dickhaut agents for sealed bid auctions. In *Workshop on Trading Agent Design and Analysis, part of international conference on autonomous agents and multiagent systems (AAMAS-2004)*, pages 59–64, 2004.

[2] R. Buyya, D. Abramson, J. Giddy, and H. Stockinger. Economic models for resource management and scheduling in grid computing. *Concurrency and Computation: Practice and Experience*, 14(13-15):1507–1542, 2002.

[3] S.-W. Cheng, D. Garlan, B. Schmerl, P. Steenkiste, and N. Hu. Software architecture-based adaptation for grid computing. In *Proceedings of the 11th IEEE Int. Symposium on High Performance Distributed Computing*, page 389, Washington, DC, USA, 2002. IEEE Computer Society.

[4] D. Cliff and J. Bruten. Less than human: Simple adaptive trading agents for cda markets. Technical Report HPL-97-9 I, Hewlett Packard Labs, Bristol, England, 1997.

[5] D. Gode and S. Sunder. Allocative efficiency of markets with zero-intelligence traders: Market as a partial substitute for individual rationality. *The Journal of Political Economy*, 101(1):119–137, 1993.

[6] A. Gounaris, N. W. Paton, R. Sakellariou, A. A. A. Fernandes, J. Smith, and P. Watson. Practical adaptation to changing resources in grid query processing. In *ICDE '06: Proceedings of the 22nd Int. Conf. on Data Engineering*, page 165, USA, 2006. IEEE Computer Society.

[7] C. Lee and J. Suzuki. An immunologically-inspired autonomic framework for adaptive and survivable network applications. In *ACM Transactions on Autonomous and Adaptive Systems, Special Issue on Autonomic Communications*, 2007.

[8] M. Preist C., Van Tol. Adaptive agents in a persistent shout double auction. In *Proc. of 1st International Conference on the Internet Computing and Economics*, pages 11–17, 1998.

[9] K. Shirose, S. Matsuoka, H. Nakada, and H. Ogawa. Autonomous con.guration of grid monitoring systems. In *Proceedings of the 2004 Symposium on Applications and the Internet-Workshops*, page 651, USA, 2004. IEEE Computer Society.

[10] R. Sutton and A. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

[11] R. Wolski, J. Brevik, J. S. Plank, and T. Bryan. Grid resource allocation and control using computational economies. In F. Berman, G. Fox, and A. Hey, editors, *Grid Computing: Making The Global Infrastructure a Reality*. John Wiley & Sons, 2003.