Time optimizing in Economical Grid Using Adaptive Stochastic Petri Net Based on Learning Automata

Intelligent approach in Economical Grid

Mohammad Shojafar
Msc. In Software Engineering, Computer & Elec. Dept.
Qazvin Islamic Azad University (Q.I.A.U)
Qazvin, Iran
Shojafar@qiau.ac.ir

Siamak Barzegar
Msc. In Software Engineering, Computer & Elec. Dept.
Qazvin Islamic Azad University (Q.I.A.U)
Qazvin, Iran
Barzegar@qiau.ac.ir

Mohammad Reza Meybodi
Computer Dept.
Amirkabir University Tech. (A.U.T)
Tehran, Iran
Mmeybodi@aut.ac.ir

***Abstract*—in this paper, intelligent algorithm of ALATO based on Learning Automata using Adaptive Stochastic Petri Net presented to optimize the time in Economical Grid. Adaptive Stochastic Petri nets based on learning automata predicts the following optimized state through getting information from former system states and dynamic environmental reactions, makes the system current states appeared based on that, changes the probability of events occurrences during the time and causes the events to be activated based on probability of occurrence. Comparing suggested algorithm and the existed algorithm has been shown in which the mentioned algorithm acts a significant decline in considered time in 200 considered independent tasks.**

Keywords-Economical Grid; Adaptive Stochastic Petri Net(ASPN); Learning Automata; Time Optimization

I. Introduction

Stochastic Petri Nets [1] are considered suitable tools for mathematical and graphical modeling. This tool can be used for modeling, description and analyzing of the systems which have concurrent, asynchronous, distributed, parallel, infinitive or accidental natures [2]. In fact, Petri nets are parts of those models which can show a system's manner and function.

One of the intelligent algorithms is Stochastic learning automata. Learning automata can be considered as a single object which has the limit numbers of actions [3, 4]. The function of this object includes this fact that every time one action is chosen from the actions collection and then will be evaluated in a stochastic environment. The taken respond from the environment will be used for the next action selection by automata and in this way, automata gradually recognize the optimum action. The method that is used to select another action by automata that is used to select another action by automata will be defined by the used learning algorithm. Stochastic Petri nets graphical features allow the simulated model to imagine the system complexity [5]. Stochastic Petri nets have a great application to analyze the Distributed systems.

Grid computing nets can be mentioned as one of these systems [6, 7]. In simply speaking, grid computing is distributed calculations which are got to the higher development level. The purpose is to make an image from a simple virtual computer and at the same time big and powerful which has its own management ability from a vast collection of computers. This collection includes services of heterogeneous related and complex systems which share various resources combinations. To manage these kinds of complex systems, the common approaches cannot be used to manage the resources which try to optimize the performance in the whole systems. In this approach, the sources owners follow the economical methods to manage their sources and to time the users' request to guarantee the considered service quality. Competitive economical models provide algorithms, policies and tools to share or allocated the sources in grid. The most common economical model is good market model. In this model, the sources have the prices which are defined based on supply and demand and value in economical system. Grid users request limited deal (the time of finishing program execution), Budget and the strategy of time optimizing as the demands of service quality. The economical scheduling , system in grid must allocate the grids source to the user's program (which consists of the great number of independent tasks) using an efficient algorithm as these limitations are observed and according to the requested time parameter becomes minimum paying attention to optimizing strategy.

In continue, an Adaptive stochastic Petri net based on learning automata is introduced and we study its application in economical grid. In this paper, stochastic learning automata used and tested types of LA in suggested idea are described in the second part. In the third section, the concept of collation and Adaptive Stochastic Petri net(ASPN) is presented. In section four, there is an explanation about some performed algorithms to optimize the time in economical grids like BTO, AFBTO AND LATO algorithms and suggested algorithm of ALATO. The fifth section reviews the suggested algorithm with the previous methods in different situations and the results are figured out in a diagram with mentioning the details. In the last section, the achieved conclusion and suggested works are performed.

II. Learning Automata

Learning Automata is a machine with limited positions which can perform limited actions. Every selected action will be evaluated by a probable environment and a respond will be returned to the learning automata. Learning automata uses this response and choose its next action as “Fig. 1”.

Learning automata can have fixed or variable structure. Learning automata can be shown by four $\{\alpha, \beta, p, T\}$ that α is the automata collection actions, β is an environment response set, p is the vector of selection probability of each action contain s probabilities and The function of T is the reinforcement algorithm, which modifies the action probability vector p with respect to the performed action and received response.. Reinforcement learning algorithm of learning automata makes the automata actions probability vector up to date in the model of S-L_{RP} [8]. Learning Automata algorithm is classified as S and P models. P-model or standard model is a position that the considered reward amount (getting a favorable response) and considered penalty amount (getting an unfavorable response)is a fixed and equal amount for every action, the general structure of learning automata in standard model is come in (1) and (2).

$$\begin{aligned} &p_i(n+1) = p_i(n) + a[1 - p_i(n)] \\ &p_j(n+1) = p(1-a)p_j(n); \\ &\forall j; j \neq i \end{aligned} \qquad (1)$$

$$\begin{aligned} &p_i(n+1) = (1-b)p_i(n) \\ &p_j(n+1) = \frac{b}{1-r}(1-b)p_j(n); \\ &\forall j; j \neq i \end{aligned} \qquad (2)$$

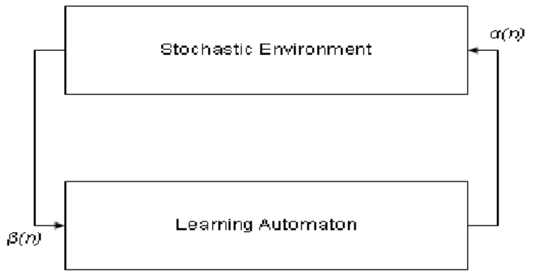

Figure 1. Learning Automata and Stochastic Environment

S-model of environment respond which is considered as a described function to improve automata has been added to the original learning automata. S-model is divided in to three classifications. a and b are reward and penalty parameters. When a and b are equal, the algorithm is called S-L_{RP}, When a>> b the algorithm is called S-L_{REP} and whenever b is zero the algorithm is called S-L_{RI} [9]. Learning Automata of S-L_{REP} is appeared with r actions and reward parameter of a and penalty parameter of b appear their vectors as following. If α_i would be chosen in nth repetition and the environment respond would be $\beta_i(n)$ to it, automata probabilities vector is appeared in (3).

$$\begin{aligned} &p_i(n+1) = p_i(n) + a(1-\beta_i(n)) \times \\ &(1-p_i(n)) - b\beta_i(n)p_i(n) \\ &p_j(n+1) = p_j(n) - a(1-\beta_i(n))p_j(n) + \\ &b\beta_i(n)[\frac{1}{r-1} - p_j(n)] \qquad \forall j \qquad j \neq i \end{aligned} \qquad (3)$$

III. Adaptive Stochastic Petri Net(ASPN)

Although, there is investigation doing to explain the quality of forcing these nets into different system , a day by day interest becomes existed in the recent years to progress a new style and method which is prepared the field in adaptive features plan in Petri net huge models [10,11].

There are various approaches presented to make the stochastic Petri net adaptive. These kinds of approaches use a general development in the field of intelligent techniques like Artificial Neural Network (ANN), Fuzzy logic (FL), Knowledge Base system (KBS) and Stochastic Learning Automata (SLA). All the attempts have been done in order to improving and developing of the samples which can transfer the adjustable feature to the Petri nets. The performed works are classified into two general classes in this field .In the first class, which is called *Hybrid Fusion*, the investigations have been done in the recent years with the purpose of defining the basic conditions under which Petri net is able to be made to describe the adaptive actions similar to existed real system. In fact, there is a real system of performance which we want to make model artificially. This artificial model which is an illustration of the real world can be easily analyzed and revised and it is possible to perform the results in the real world. The reason of being adaptive in this class is related to basic changes of Petri net structuring and its getting homogenous with the real system frame. Some changes in Petri net basic structure have caused the ability and power of learning and teaching in Petri net using intelligent techniques. This class has some disadvantages too. Learning in Petri nets of this class causes increase in operation complexity and computing loads because it takes a lot of time to model all real system features in addition using the intelligent techniques in Petri net. In the second class, which is called *Combination Hybrid*, a quiet small researcher’s group activate in the field of adaptive intelligent techniques usage and performance in addition to Petri net methodology in real world matters. This class models are simpler than the first group models. However, sometimes their performing time is more. Also, the complexity measurement in designing is less than the first method. In this method, different technologies are put next to each other and the presented model in order to perform in the environment will be exerted with more profitability and accuracy by proportional changes in each one with regard to normal condition, this method uses the basic features of Petri Net in with intelligent techniques which is exerted to internal actions and requests of every part of a real system .Here, there is a suggestion of a veal and clear combination of intelligent features with system present condition in order to improvement of current system position [12,13].

In fact, adaptive stochastic Petri net is a kind of a stochastic Petri net which can predict the later system position with the help of present condition with regard to environmental information in other designing steps and gradually make it up to date with environmental condition Paying attention to the definition presented in this section about Adaptive Stochastic Petri Net (ASPN), we conclude that a stochastic Petri Net can be used in intelligent systems and the systems which are dynamic and have a lot of changes. Adaptive stochastic Petri Net performed in this paper is a part of the first algorithms that is the algorithms of Fusion Hybrid. We will explain the suggested algorithms i.e. ALATO based on learning automata.

IV. Proposed Idea(ALATO)

In task scheduling in Economical grid, there are two features of resource *Time-Shared* or *Space-Shared* which have a lot of influences.

A. Time-Shared Resources

Everything which is left to a Time-shared resource immediately transfers to one of the processors for performance because time-shared processors can perform several works synchronously and as a matter of fact tasks use processors commonly.

This means that every task settles on a processor in a define time distance and then they will be stopped and the next task transfers to the processor to be performed. Every task, whose performance ends, will be exit from queue and the processor time will be equally divided between the remained tasks. One of problems that time-shared resources have is that the tasks performance finishing time cannot be unpredictable because there is always the probability of new task transferring and breaking of predicting done, one of the other problems of these resources is that while scheduling if the great number of tasks is left to them, they will be faced with overload position. In this case, because of concurrent performance of a lot of tasks in a Time-Shared processor, a small percentage of time processor allocates to every task and the stopping operation of one task and performance continuing of another task must be done a lot of times that causes to make overload in a processor and reduces the efficiency.

B. Space-Shared Resources

In a Space-Shared Resource, there is a queue of unprocessed tasks that as soon as a processor doesn't have anything to do (Idle), the first task of the queue transfers to it to be performed. So, each space-shared processor can perform one task in a time. This means first task of the queue transfers it to be performed. This means when a task transfers to processor for performance, it stands there continually up to the end and its performance will never be stopped. Here, the reviewed tasks is intelligent algorithm based on space-shared are considered independently and commonly. The tasks allocation way divides into two cases of all *at-once allocation* and *phased allocation*.

C. Phased Allocation

In phased Scheduling, scheduling operation becomes a phased operation which will continue up to the end of all tasks transferring. In this solution, task is left to every resource only equal to the number of the existed processors in it to avoid the concurrent performance of some tasks on one processor. Also, there is a queue for each user and the not transferred tasks of user program are kept in this queue. The task method is in this case that after entering a collection of independent tasks which make a a practical program belonged to the user, Scheduling system mapped all the tasks to the resources which is defined by scheduling algorithm; but the tasks allocating, tasks are left only in the number of not allocated processors. Then, all the extra tasks will be collected and returned to their own users. On the next stage, scheduling algorithm will be implemented again and the resources will be chosen to be mapped to which tasks are left in not allocated processors numbers. This operation will be continued until all tasks are transferred to the resources.

D. At-once Allocation

In At-once scheduling, the scheduler transfers the tasks to the resources which are defined by scheduling algorithm At-once and every resource takes the control of the number of performing tasks on its every process or (no processor doesn't perform more than one task at the same time) and regarding to its belonged tasks queue, it transfers them regularly to the processors which will be not allocated.

In fact, the difference between all at-once scheduling and phased scheduling is that in scheduling system which uses phased scheduling. Phased scheduling is that in scheduling system which uses phased scheduling method, the tasks are kept in laws related to their own users and are left to the resources gradually and through repeated stages. However, in at-once scheduling method, all the tasks are left to the resources at-once and tasks gradual leaving to the processor is done by the resources, themselves (every resource makes a queue for allocated tasks to itself). It means all resource Behaviors are like a space-shared resource. There are some algorithms presented in scheduling optimizing field to allocate the resource according to demands in Economical Grid. All algorithms purpose is to be able to use the defined budget by the user in maximum and decrease the program performing time as possible. BTO algorithm is suggested to time optimizing by Buyya in [14, 15] and is evaluated in [16]. BTO is a heuristic algorithm presence that shows acceptable responds to the user for independent homogenous or little heterogeneous tasks. Also, Buyya considers the resources time-shared and causes some Complexity and defects in his algorithm. In the continue, Mr. Mahdavifar has suggested some algorithms for this after studying and simulation of various ways to optimize the time. He has suggested AEBTO and LATO algorithms [17, 18]. In continue a kind of algorithm named ALATO is presented to make load balance in tasks allocation into grid computing resources. BTO and AEBTO algorithms are heuristic algorithms and LATO and ALATO algorithms are intelligent algorithms and based on Learning Automata. AEBTO algorithm makes some changes in BTO algorithm which making better scheduling. LATO algorithm uses with Learning Automata try to minimize time for desired budget. ALATO algorithm changes in getting off method and paying reward and penalty in LATO algorithm and get the best result among the mentioned algorithms.

Here, we consider *Minimum* algorithm which is calculated with the least time. To get the minimum of calculation time, the presented algorithm can be used to optimize the time that we use scheduling operation for instructions as the smallest executive unit instead of scheduling the tasks. It means, we allocate some budget to every million instructions and leave them to the suitable resource to be presented. Another way to get the minimum of calculation time is to use the performed method to get the minimum of cost. To do this, we carry out the minimum cost method considering the performance respite and study the getting expresses.

Then, we lose some of the respite and perform the cost minimum method again. We repeat this operation as many times to get the time in which tasks doing cost minimum become almost equal to the defined budget. In this case, the used time in cost minimum method is calculation time minimum. We use this method to get the minimum of tasks performing time spending the defined budget by the user.

Suggested algorithms (ALATO) in this section is in the case of *Space-Shared* and uses all *At-once Scheduling* method and regulate the tasks queue regarding to length in a descending position to decrease the not allocated time of more expensive resources. At user's time entrance to the system, if his practical program doing is possible in the defined time and budget area, it will accept the user and guarantee the program ending obeying these areas. To do this, at the end of suggested algorithm and after distinguishing the tasks doing time total and program performing expenses which are approximately gained in scheduling.

In the case of user's acceptance, the distribution stage will be done in which the tasks are transferred to the defined resources.

Therefore, ALATO algorithm tries to fulfill the complicated operation of time optimizing with better efficiency than the heuristic algorithms. This algorithm uses a collection of learning automates that are attributed to the tasks. These learning automates have variable structures and their actions are in agreement with grid resources. In fact, learning automata belonged to a task selects the recourse to which the task must be allocate. When all the words choose their resources regularly, they will get reward or penalty from the environment. Then again, all the woks fulfill the resource selection to get the environment response. This operation repeats in definite numbers up to the time that all tasks find their suitable resources and be allocating them.

ALATO algorithm cannot catch the final scheduling with studying the tasks and scheduling for only once, but it does this operation by a lot of repetition and makes a new scheduling in every repetition. Environment responds the tasks (reward or penalty) regarding the resources they have chosen. Tasks use this environmental response in the following repetition in order to make the new existed scheduling closer to optimum scheduling. These repetitions occur in a great number till all the tasks find their suitable resources and don't have any other change in their decisions. Actually, repetitions continue up to the time that all the learning automatas become convergent and a harmonic scheduling is made in repetitions. This homogenous scheduling is final scheduling in whose base the tasks are transferred to the resources.

Also in ALATO algorithm, the made scheduling must be temporary done to the resources in every repetition to evaluate it. To do this, every resource include a temporary queue called *relation queue* which is emptied in the beginning of every repetition and then stands immediately in the relation queue when the tasks are selecting regularly.

The environment response to the learning automata belonged to tasks occurs regarding to ideal conditions in a time optimizing. To do this, environment uses the following points to give reward or penalty to the tasks:

1) Tasks executive expense total must not be more than budget defined by the user.

2) The execution time for left tasks to the busier resource must not be allotted different from performance time in other resources.

3) Every tasks mast is given to a resource which can finish its performance sooner than the other resources.

In this case, an environment punishes the tasks using two first points and rewards to them regarding the third point. According to the first point, it tasks performance total costs on their selected resources is more than the define budget, the environment will punish the tasks which need a lot of expenses to perform. In fact, punishing operation starts from tasks queue end in the most expensive used resource (the most expensive resource to whose one tasks is related at least once).

To punish every task and omit it from the selected resource related queue, the execution difference on this resource with its performance expense on the cheapest existed resource deducts from the total costs. The punishing operation continues till the total cost is more than the defined budget.

As a matter of fact, tasks are gaited toward using the cheaper recourses with this penalty. The purpose in this stage is decreasing the cost fast and takes it to a less amount measurement than the considered one to transfer the tasks from the most expensive resource in to the cheapest resource. To regard the second point, environment considers the tasks punishment in related queue of the most activated resource.

Pay attention to this point that the most activated resource is not necessarily the most expensive resource. We start from the last activated queue end.

We chose Y number tasks from the end queue of the most activated resources. If one of the used resources in scheduling (the resource which has at least one related task) except the last one (the most expensive) can process this task sooner than the most activated resource, environment will punish this task then it will choose more suitable resources for itself in the following repetitions unless this task won't be punished. This operation will be done for all Y tasks from the most activated resources. This operation causes that woks distribute with more balance and equilibrium among the recourses. The amount of Y is calculated according to "TABLE I" from (4) and (5).

$$X = (Task\_No - A)/(Res\_No - 1) \tag{4}$$

$$Y = (B - X) * \frac{Res\_No - 2}{Res\_No - 1} \tag{5}$$

Finally environment rewards tasks according to the third point. The tasks which can get less finishing time or equal in compare with the previous repetition will get rewards. Of course, if they haven't punish in this repetition. So. Every task encourages choosing a source for its transferring which ends its performance in the least time.

As it is figured out in "Fig 2", ALATO algorithm firstly arranges the tasks based on their length in a descending way. Then in a lot of repetitions, finds the suitable resource to perform each task. This algorithm finally transfers tasks to those resources and uses all at once scheduling method.

All models of learning automata (Standard Model and S-Model Methods) are studied to achieve the best answer. In standard automata environment uses two penalty response and one reward response during algorithm learning. Since a task isn't punished in any of two cases and also gets less or equal ending in compare with the previous repetition, probably it had found a suitable source for itself. In this case, the reward amount is (0.1) more than the amount of penalties in the

TABLE I. THE ABBREVIATION SIGNS FOR ALATO

Acronym	Details Types
Tasks_No	Task Number
Res_No	Resource Number
A	Task Numbers Allocated To The Most Expensive Resource
B	Task Numbers Allocated To busiest Resource

```
1. Arrangement: Arrange the tasks according to their length in a descending way.
2. Learning: Repeat 10000 times:
    a. Make the resource relation queue empty
    b. Selection: Do this for each task in order:
        i : choose resource to transfer using related learning automata
        ii: Put the task in selected in source relation queue.
    c. Penalty (1): Repeat since the total cost of tasks performance on selected resources are more than the defined budget:
        i: Divide one task at the relation queue ending in the most expensive used resource.
        ii: Decrease the difference of the task performance expenses on this resource with the task executive cost on the cheapest existed resource from the total cost of tasks performance.
        iii : Fine the task (Penalty amount : 0.01)
    d. Penalty (2): Repeat Y times:
        i: Consider the last related task to the most activated resource.
        ii: Fine the task in one of the used resources lout the last one (the most expensive used resource) in able to finish this task performance sooner (Penalty amount : 0.05)
    e. Reward: Do for every task:
If it isn't fined: if the finishing time of its performance on selecting resource is less than or equal to the finishing time in previous repetition, reward it. (Reward amount: 0.1)
3. Scheduling: choose a resource for each task in order using the learning automata be longed to it and record the task to that resource.
```

Figure 2. ALATO Algorithm

First case, it's (0.01) more and in the second case, it is (0.05) more. The cause for being more in penalty amount in the second case is because of putting the smaller tasks in more activated resources and increasing in that recourse accumulation. In fact, the tasks go towards the putting in less activated resources and increasing in load balance in every step by increasing the tasks penalty and automata convergent speed is increased significantly. We have tested the learning automata S-model with various amount for reward and penalty rate which is come in simulation section.

In the next part, Minimum, LATO, AEBTO, BTO and ALATO suggested algorithm are studied in different cases of Learning Automata.

V. SIMULATION AND PERFORMANCE EVALUATION

We describe a task model in CSPL language of SPNP software in this performance. A Sample Formatting is used in tests for grid resources named GRI that of course is not existed in the outside world.

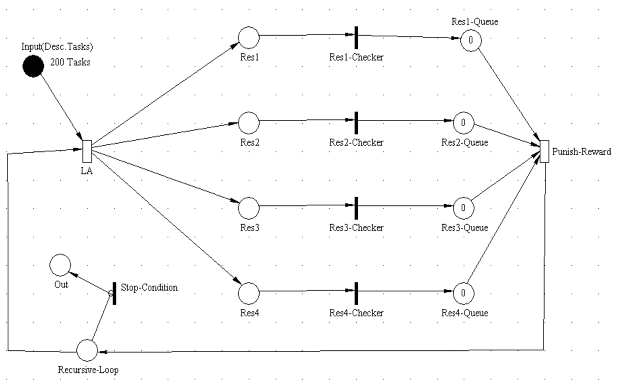

Figure 3. To allocate the tasks to the resources and applying the reward, Penalty and performance 10000 times

A sample of a real formatting named WWG is come in [19] which are also used in experiments. In this formatting four resources with different performance rate and useful prices are used that are arranged ascend through cost. In this simulation. 200 tasks amount with the various lengths is studied in a defined time by the user in various non-homogeneous budgets for four resources.

In presented adaptive Stochastic Petri Net (ASPN), every token shows one task of user's tasks. Every token include the information like task length, priority number based on its length and learning automata.

In “Fig. 3”, every token choose an action (one of the exited resource in grid net) based on its probability vector. Then all the allocated tasks in each of the resources based on the capacity of that resource, the time user defined to do that task and the algorithm suggested in previous section will get reward or penalty. Our four resources are *Res1, Res 2, Res3,* and *Res4* for allocating action that each resource has distinct queue that located in order *Res1_Queue, Res2_Queue, Res3_Queue* and *Res4_Queue*. In every tasks allocation to every resource the tasks number located in its resource allocated queue and transition Punish_Reward is for penalty/reward action for each allocated resources. This may is done up to 10000 times on the resources in a returning manner and the gained results of every performance go to the next executive stage after storing in *Recursive-Loop* state.

In Transition LA, S-model Learning Automata is considered for all four resources. We have tested the S model LA with various amounts for reward and penalty rate. In S model methods, if action α_i is chosen in the n^th repetition the environment response (undesired response) is $\beta_i(n)=1$ to it and environment response (desired response) to it illustrated in (6).

$$B_i(n)\frac{1}{1+\frac{\Pr evous\,time}{New\,time}} \qquad (6)$$

It means it gains better response in Compare with the cheaper resource selecting.

According to the results we will observe that S-L $_{\text{Rep}}$ method will have the best result with the reward rate equals to 0.1 and penalty rate equals to 0.05.

Every task has its own specific length which is stated according to million instructions (MI). We always consider the tasks length valid, in this case that on area (Maximum ... Minimum) will be chosen for tasks length. The for every task length, an amount is chosen from this area as the tasks length distribution is UNIFORM. For the most homogeneous case, we consider the tasks length distribution area of (100.000 ... 110000) and for the most heterogeneous case the area of (10000...200000). The conditions of doing simulation experiment are figured out in “TABLE II”. The tasks length area parameter is various and the experiments are done in different areas (it means different Heterogeneous). The parameter of the number of experiments distinguishes that 20 tests are done to gain the time amount of algorithms performance in a definite Heterogeneous and them the average amount is selected.

In “Fig. 4”, there is a comparison between the mentioned automates in LATO and ALATO algorithms in addition to Minimum algorithm. Here, the best time to perform each automaton is considered 150000 for the budget. ALATO algorithm is improved a lot according the time and is very close to the optimum case comparing with LATO algorithm. S-L$_{\text{Rep}}$ model spends less time in compare with the other automates.

TABLE II. TIME OPTIMIZING TEST CONDITIONS IN DIFFERENT HETEROGENEOUS AND 150000 BUDGET

Parameter	Value	Parameter	Value
Algorithm Type	Alternative	Time	20000
Res. Config.	GR1	Budget	150000
User No.	1	Task Length	Alternative
Exp. No	20	Task No.	200

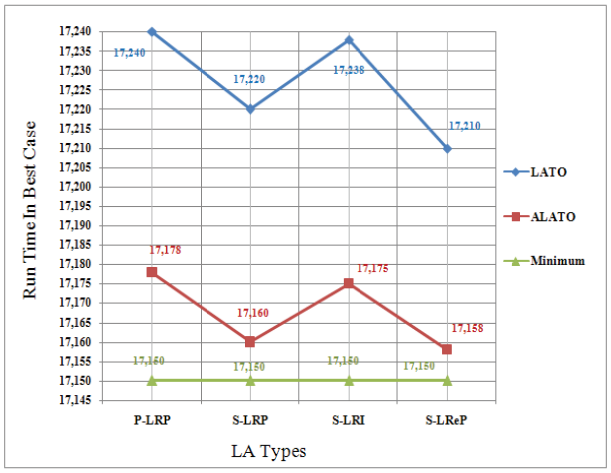

Figure 4. Comparing theRun Time of LA’s with the Fixed Budget of 150000 in the Best Situation of Every LA

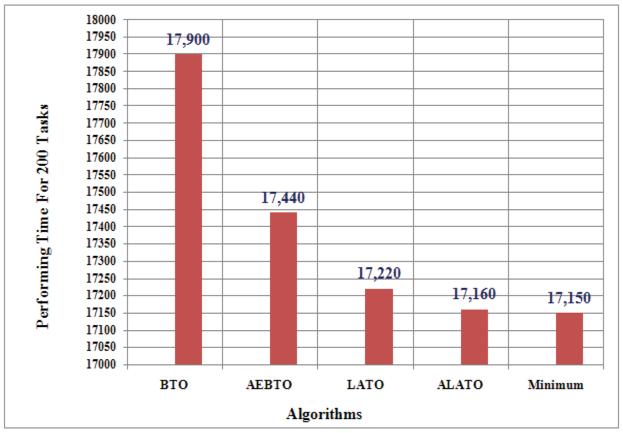

Figure 5. The Performing Time in an Acceptable Situation of S-L_{RP} Automata with te Fixed Budget of 150000

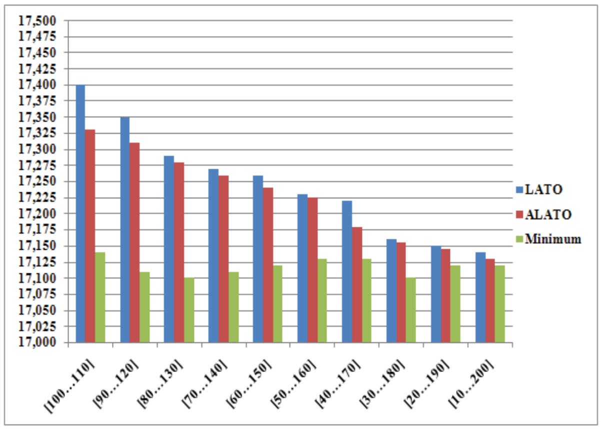

Figure 6. Comparing LATO, ALATO and Minimum Algorithms in Different Non-homogeneity

This is because of this fact that whatever the reward amount is more than penalty, the time length is more possible to make update and choose resource with the less repetition. S model shows the better result comparing with P-Model. Because, Considering the smaller changes amount in β_i makes the automata more accurate and proving time of resources selection probabilities will be possible with less repetition and as a result the time for automata to be constant is more that the P case.

"Fig. 5" illustrates the performing time duration in an acceptable situation of S-L_{Rp} LA and the case which penalty is 0.05 and the reward is 0.1. As you see,

ALATO algorithm acts better than LATO algorithm about 60 units and it has just 10 units difference from Minimum or Optimum case.

Because S-Model automata present a better time comparing with Standard LA, this automata is chosen to study. In S-Model types, S-L_{Rep} automata have presented a better result, so, we evaluate the results in these automata. We have done 20 tests for every heterogeneous and considered the average amount implemented for each algorithm as the enrage time to perform the tasks.

LATO and ALATO algorithms that use the learning automata and ALATO as the best suggested algorithm wants to improve LATO algorithm. Here, we review the tasks heterogeneous effect on improvement measure resulted from LATO and ALATO algorithm.

"Fig. 6" shows the results of these two algorithms in addition to minimum algorithm next to each other and different homogeneities. It is observed that the improvement resulted from ALATO and LATO algorithms scheduling becomes more with increasing the tasks Non-homogeneity. The time decreasing trend in ALATO algorithm is more than LATO algorithm.

In fact, the speed of getting Minimum in ALATO is more than LATO.

As you see, ALATO algorithm is very close to optimized algorithm (Minimum) Comparing with LATO heterogeneous.

VI. Conclusion and Future Work

In this paper an Adaptive Stochastic Petri Net is presented to improve the time in economical grid helping learning automata. This presented net is a part of fusion Hybrid class. As it is state in previous sections, there are some disadvantages in this class. Most of the time learning in Petri nets from this class cause increasing in operation complexity and rise in overhead calculation. As it is also reviewed in section two of this paper, Adaptive Stochastic Petri Net use intelligent techniques like artificial neural nets, Fuzzy logic and knowledge based system, but the Adaptive Stochastic Petri Net that we are presented is based on learning automata. Adaptive Stochastic Petri Net based on learning automata doesn't need any training with some pre-defined models in contrast with Adaptive Petri Net that use artificial neural nets. In the systems in which the environment is unknown and changing every time, there can't be any model with which the artificial neural net can be trained lout suggested adaptive model in these environments are completely flexible because learning automata doesn't need any training with the previous data, so, it doesn't have the fusion Hybrid class problem which is increasing in calculating overhead. Besides, in Adaptive Stochastic Petri Net based on learning automata, every token has its specific automata and in contrast with previous adaptive models that need to make extra place and passes to make artificial neural net, there is no need to any newer place and pass and it causes non-complexity in planning and raise in calculating overhead.

Also, using time optimizing algorithms simulation, it is distinguished that ALATO learning algorithm in a lot of tasks heterogeneous gets better results comparing with the heuristic algorithms and it can perform the practical plan be longed to user on the grid in less time comparing with the other algorithms spending the defined budget. Also, regarding to the small amount of gained time from this algorithm with the minimum of calculating time, there is no significant possibility to make the ALATO algorithm improved.

The future tasks which will be stated in this field we can mention using presented Adaptive Stochastic Petri Net in demands scheduling and the quality of their allocation in economical grid base on techniques like bargaining, auctions, calling for tenders and barter trade.

REFERENCES

[1] J. Peterson, Petri Net Theory and the Modeling of Systems, Perantic Hall, Englewood Cliffs, NJ, Vol. 1, 1981.

[2] W. Reisig, Petri Nets: An Introduction, EATCS Monographs on Theoretical Computer Science, Springer Verlag, Vol. 1, 1985.

[3] R. Boppana and M. M. Halldorsson, "Approximating maximum independent sets by excluding subgraphs," BIT Magazine, Published By BIT Computer Science and Numerical Mathematics, ISSN:0006-3835, Vol. 32, Issue 2, 1992, pp.180-196. DOI: http://dx.doi.org/10.1007/BF01994876

[4] T. N. Bui and P. H. Eppley, "A hybrid genetic algorithm for the maximum clique problem," Proceedings of 6th International Conference on Genetic Algorithms, Published By Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, ISBN: 1-55860-370-0, 1995, pp. 478-484.

[5] C. Hirel, S. Wells, R. Fricksy and K. S. Trivedi, "ISPN: An Integrated Environment for Modeling Using Stochastic Petri Nets,", Center for Advanced Computing and Communication Department of Electrical and Computer Engineering Duke University, Durham, NC 27708-0291, 2000.

[6] R. D. Venkataramana and N. Ranganathan, "Multiple Cost Optimization for Task Assignment in Heterogeneous Computing Systems Using Learning Automata," Heterogeneous Computing Workshop (HCW'99), 1999, pp. 137-145.

[7] J. Moore and L. Hahn, "Petri Net Modeling of High-Order Genetic Systems Using Grammatical Evolution," Bio Systems 72, 2003, pp. 177–186.

[8] K. S. Narendra and M. A. L. Thathachar, Learning automata: An introduction, Prentice Hall, 1989.

[9] A.S. Poznyak and K. Najim, Learning Automata and Stochastic Optimization, ISBN 3-540-76154-3 Springer-Verlag Berlin Heidelberg New York, 1997.

[10] T. Murata, "Some recent applications of high-level Petri nets," IEEE International Symposium on Circuit and System, ISBN: 0-7803-0050-5, Vol. 2, Old Version (1991), 2002, pp. 818-821. DOI: http://dx.doi.org/10.1109/ISCAS.1991.176488

[11] M. C. Zhou and M. D. Jeng, "Modeling, analysis, simulation, scheduling, and control of semiconductor manufacturing systems: A Petri net approach," IEEE Transaction on Semiconductor Manufacturing, ISSN: 0894-6507, Vol. 11, Issue 3, Old Version (1998), 2002, pp. 333-357. DOI: http://dx.doi.org/10.1109/66.705370

[12] R. Khosla and T. Dillon, "Intelligent hybrid multi-agent architecture for engineering complex systems," International Conference on Neural Networks, Houston, TX, and ISBN: 0-7803-4122-8, Vol. 4, Old Version (1997), 2002, pp. 2449-2454. DOI: http://dx.doi.org/10.1109/ICNN.1997.614540

[13] L. C. Jain and N. M. Martin, Fusion of Neural Networks, Fuzzy Sets, and Genetic Algorithms: Industrial Applications, ISBN: 0849398045, First edition, CRC Press, FL, USA, 1998. DOI: http://www.amazon.com/exec/obidos/ASIN/0849398045/acmorg-20

[14] R. Al-Ali, O. Rana, D. Walker, S. Jha and S. Sohail, "G-QOSM: Grid Service Discovery Using QOS Properties," Computing and Informatics Journal, Special Issue on Grid Computing, Vol. 21, No. 4, 2002, pp. 363–382.

[15] R. Buyya, Economic-Based Distributed Resource Management and Scheduling for Grid Computing, Ph.D. Thesis, School of Computer Science and Software Engineering, Monash University, Melbourne, Australia, 2002.

[16] K. Czajkowski, I. Foster, C. Kesselman, V. Sander and S. Tuecke, "SNAP: A Protocol for Negotiating Service Level Agreements and Coordinating Resource Management in Distributed Systems," 8Th Workshop on Job Scheduling Strategies for Parallel Processing, 2002.

[17] Y. Mahdavifar and M. R. Meybodi,"Time Optimization in Economic Computational Grids Using Learning Automata," Proceedings of the First Iranian Data Mining Conference, Amirkabir University of Technology, Tehran, Iran, 2007.

[18] Y. Mahdavifar and M. R. Meybodi, "Cost-Time Optimization in Economic Computational Grids," Proceedings of the Third Information and Knowledge Technology, Ferdowsi University of Mashad, Mashad, Iran, 2007.

[19] R. Buyya, "The World-Wide Grid (WWG)," 2002. http://www.buyya.com/ecogrid/wwg/