International Journal of Computational Intelligence and Informatics, Vol. 2: No. 3, October - December 2012

Decision Tree Learning Technique for Selection of Appropriate Grid Scheduling Algorithm

D Ramyachitra
Department of Computer Science
Bharathiar University
Coimbatore-641 046, India
jaichitra1@yaho.co.in

K Vivekanandan
Department of Management Studies
Bharathiar University
Coimbatore-641 046, India
vivekbsmed@gmail.com

***Abstract*-Different scheduling algorithms are discussed in the literature that is appropriate for the grid environment. Scheduling heuristics can be classified into constructive and improvement and we have studied and analyzed several widely used the above said heuristics. The heuristics perform according to the nature of the users' tasks and the resources. In this paper, we have proposed a strategy for finding out an appropriate scheduling heuristic from the widely used ones. Based on our study, we classified the user's tasks and resource into seven categories using the decision tree learning technique based on quantity and heterogeneity. When the user submits the job, the characteristics of the tasks and the resources given by the user will match with any one of the categories. Our proposed strategy selects a scheduling heuristic that has the maximum occurrence of best makespan and executes the user tasks. The proposed strategy shows better accuracy and this can be used to find the appropriate scheduling heuristic to execute users tasks.**

Keywords: constructive, improvement scheduling heuristics, grid, machine learning, decision tree learning.

I. INTRODUCTION

Grid computing aims to maximize the utilization of an organization's computing resources by making them shareable across applications and providing on demand computing. Grid technology enables resource virtualization, on demand provisioning and resource sharing between organizations. It can be confined to a network of computer workstations within an organization or it can be a public collaboration. By harnessing a great number of different systems in a transparent manner, a user can have access to the computer architectures that are best suited to the constraints of its applications.

In the new distributed scenario such as grid systems, traditional scheduling techniques have evolved into more complex and sophisticated approaches and other factors such as the heterogeneity of resources or geographical distribution have been taken into account [1]. Job scheduling is a NP complete problem [2], which is responsible for the management of jobs, such as allocating resources needed for any specific jobs, data management, service level management capabilities etc. Scheduling also must provide capabilities for areas such as (i) advanced resource reservation (ii) service level agreement validation and enforcement (iii) job and resource policy management and enforcement for best turnaround times within allowable budget constraints (iv) monitoring job execution and status (v) rescheduling and corrective actions of partial failover situations [3]. Different scheduling algorithms are discussed that are appropriate for the grid environment depending on the characteristics of the network connectivity, jobs, machines etc. In the case of static scheduling, information about all the resources in the grid and the characteristics of the grid are known in prior when the application is scheduled. In dynamic scheduling, only a few assumptions about the parallel program can be made before execution, and thus scheduling decisions have to be made on the fly [4]. Desirable performance goals of grid scheduling include maximizing system throughput, maximizing resource utilization, minimizing execution time, minimizing the cost on the user side and fulfilling economical constraints.

In the grid system, an end user submits the job that has to be executed with some constraints like job execution deadline, cost of execution and time required for execution. After estimating the resource requirements, Grid resource manager provides the functionality for discovery and publishing of resources as well as scheduling, submission and monitoring of jobs [5]. There are several key metrics to evaluate the

effectiveness of grid scheduling algorithms. A single measure cannot capture the whole performance of the Grid. On the user's perspective, the key measures of grid performance include time and cost. Time includes average response time, average wait time and average execution time. The cost is the economic cost that an application needs to pay for resource utilization.

Scheduling the applications not only includes the search for a suitable set of resources to run applications, but also coordinates the time slots allocated on several different resources to run the application. Also, the scheduling system must execute all these activities while balancing the optimization functions provided by the users such as cost, response time and other objectives represented by the resource providers. Grid scheduling differs significantly from conventional job scheduling on a parallel computing system. There are many grid schedulers implemented to reduce the complexity of the problem for particular application scenario. However, no common and generic grid scheduler exists and probably there will never be one as the particular scenario will require dedicated scheduling strategies to run efficiently [6].

Machine learning, a branch of artificial intelligence concerned with the design and development of algorithms, which allows computers to progress behaviors based on experimental data, are also used in scheduling widely. Machine learning data studies recorded data and subsequent machine failures and learns prediction rules. There are many tasks where it is relatively difficult for the programmers to design and implement the necessary software. There are problems where there exists no human experts, for example, in automated manufacturing industries, machine failures has to be predicted before they occur. There are also problems where human experts exist, but they may not be able to explain their expertise, for example, speech recognition, handwriting recognition etc. Third, there are problems where phenomena are changing rapidly. For example, in stock exchange, people would like to predict the future behavior of the stock market. Last, but not the least, there are applications that need to be customized for each computer user separately. In this paper, machine learning technique is used for finding out an appropriate scheduling heuristic from the widely used ones according to the nature of the users task and resource requirements. The remaining sections of the paper is organized as follows. Section 2 deals with related work. Section 3 describes machine learning in the grid environment. Here, the appropriate scheduling algorithm is selected using machine learning concept. Section 4 deals with results and discussion. Finally section 5 contains our conclusion.

II. RELATED WORK

The scheduling heuristics can be classified into constructive and improvement heuristics. There are many scheduling algorithms discussed in the literature. Opportunistic Load Balancing (OLB) [7] assigns each task, in arbitrary order, to the next machine which is expected to be available, regardless of the task's expected execution time on that machine. Minimum Execution Time (MET) [7], assigns each task in arbitrary order, to the machine with best expected execution time for that task, regardless of that machine's availability. Minimum Completion Time (MCT) [7] assigns each task, in arbitrary order, to the machine with the minimum expected completion time for that task. Min Min [7] heuristic selects a machine with minimum completion time and sends the task with minimum completion time for execution to the machine. Max Min [7] heuristic sends the task with maximum time for completion to the machine with minimum completion time. Tabu Search (TS) is based on neighborhood search with overcoming local optimality. It performs a number of iterations and at each iteration, TS moves to the best solution that is not forbidden and thus independent of local optima [8]. Simulated Annealing (SA) origin is in statistical mechanics. In order to implement SA for grid scheduling, a number of decisions and choices have to be made. Genetic algorithm (GA) is an adaptive method that can be used to solve optimization problems, based on the genetic process of biological organisms [9]. In 1999, Ant Colony Optimization (ACO) metaheuristic was proposed by Dorigo, Dicaro and Gamberdella, which has been successfully used to solve many NP problems such as TSP, job shop scheduling etc. Particle Swarm Optimization based scheduling is a population based optimization tool, which has been used to solve various optimization problems [10].

Paolo Priore et al. in [11] have given a review of the main machine learning based scheduling approaches. It is also dealt in the paper that a common way of dynamically scheduling jobs in a flexible manufacturing system is by means of dispatching rules and the performance of these rules depends on the state of the system is in at

each moment. So appropriate dispatching rule should be used at each moment. Lee, Wen-Chiung et al. has investigated a single machine problem with the learning effect and release times with the objective of minimizing makespan [12]. Branch and bound algorithm incorporating several dominance properties and lower bounds is developed to derive optimal solution. Tamaki et al. proposed an approach based on genetics based machine learning for flow shop scheduling problem [13]. Here, a set of scheduling rules is represented as an individual of genetic algorithms, and the fitness of the individual is estimated based on the makespan of the schedule generated by the rule set. Jiangtian Li et al. use machine learning techniques to generate performance models for all tasks and have applied those models to perform automatic performance prediction across program executions [14]. This can be used where systems lack prior knowledge about the execution time of the tasks. Diego Puppin used machine learning techniques to automatically search for good orderings in convergent scheduler which consists of a vast number of legal pass orderings [15]. Here, genetic programming, s – expressions has been used to describe a particular pass sequence. Ashish Revar et al. has analyzed load balancing requirements in a grid environment and proposed an algorithm with machine learning concepts to find an efficient algorithm [16]. Daniel Vladusic et al. have employed machine learning methods to improve job scheduling over heterogeneous Grid nodes [17]. They have provided a plug in which can be added to existing frameworks for avoiding significant and time-consuming modifications. They have also used simple machine learning methods in a black box manner with internal parameter optimization. Guang Xiang et al. has proposed a hybrid machine learning and statistical model, where the machine learning component predicts the security of a service by considering the probability distribution of the past services and the statistical component evaluates the service security statistically based on its own past behaviors and users opinions [18]. Binti Abdullah et al. have proposed an approach that incorporates a machine learning model into a conversation environment for the induction of communication protocols [19]. Guopeng Zhao et al. have proposed an intelligent resource selection algorithm based on neural networks [20].

Machine learning addresses many of the research problems. Grid environment consists of distributed heterogeneous resources, with each resource having their own characteristics; it would not be efficient to use one scheduling heuristics for all the applications, as each application/problem will be of different nature. In this paper, an attempt is made to use decision tree learning technique to find and use an appropriate scheduling heuristic to execute an application.

III. Machine Learning in Grid Environment

In this paper, we have implemented several widely used constructive and improvement scheduling heuristics. To find an appropriate scheduling heuristic from the list of widely used heuristics for executing the user's task, we have used decision tree learning technique.

The decision tree learning technique uses decision tree as a predictive model which maps observation about an item to conclusions about the items target value. A decision tree is a hierarchical data structure implementing the divide and conquers strategy. It is an efficient non parametric method, which can be used for both classification and regression [21]. A tree can be learned by splitting the source set into subsets based on an attribute value test. This process is repeated on each derived subset in a recursive manner called recursive partitioning. The recursion is completed when the subset at a node all has the same value of the target variable, or when splitting no longer adds value to the predictions. Decision tree has number of advantages over other learning techniques such as simple to understand and interpret, requires little data preparation, able to handle both numerical and categorical data, uses a white box model, possible to validate a model using statistical tests, robust and performs well with large data in a short time. Hence, this technique has been chosen to find the suitable scheduling heuristics for a given users job.

The internal nodes of the decision tree are tests on input patterns and leaf nodes are categories of patterns and each branch corresponds to attribute value. Each test has mutually exclusive and exhaustive outcomes. The conventional decision tree algorithms are ID3, C4.5, C5.0, CART, Random Forest etc. In decision tree learning, ID3 (Iterative Dichotomiser 3) algorithm is used to generate a

1. Take all unused attributes and count the ir entropy concerning test samples.
2. Choose attributes for which entropy is minimum or information gain is maximum.
3. Make node containing that attribute.

Figure 1 ID3 algorithm

decision tree invented by Ross Quinlan which is a precursor to C4.5 algorithm. Summarization of the ID3 algorithm is given in the Figure 1.

For creation of root node, entropy and information gain has to be computed.
Entropy is computed using the formula

$$I(P)=-(p_1 \times \log_2(p_1) + p_2 \times \log_2(p_2) + \ldots\ldots\ p_n \times \log_2(p_n)) \quad (1)$$

where I(P) is the entropy of P
P is the given probability distribution of the scheduling algorithms, P = (p1, p2……..pn).
Gain for each attribute is computed using the formula

$$Gain(no\_of\_tasks, P)=I(P) - I(no\_of\_tasks,P) \quad (2)$$
$$Gain(tasklength,P)=I(P)-I(tasklength,P) \quad (3)$$
$$Gain(taskhet,P)=I(P)-I(taskhet,P) \quad (4)$$
$$Gain(no\_of\_resources,P)=I(P)-I(no\_of\_resources,P) \quad (5)$$
$$Gain(resource\_speed,P)=I(P)-I(resource\_speed,P) \quad (6)$$
$$Gain(process\_het,P)=I(P)-I(process\_het,P) \quad (7)$$

Gain(no_of_tasks) and Gain(no_of_resources) has highest value and hence, number of tasks and resources will be assigned to the root node. Next, processor heterogeneity and task heterogeneity has the higher value and so they are assigned to the next level of number of tasks and resources. Adaptive machine prediction procedure for scheduling is given in detail in the Figure 2.

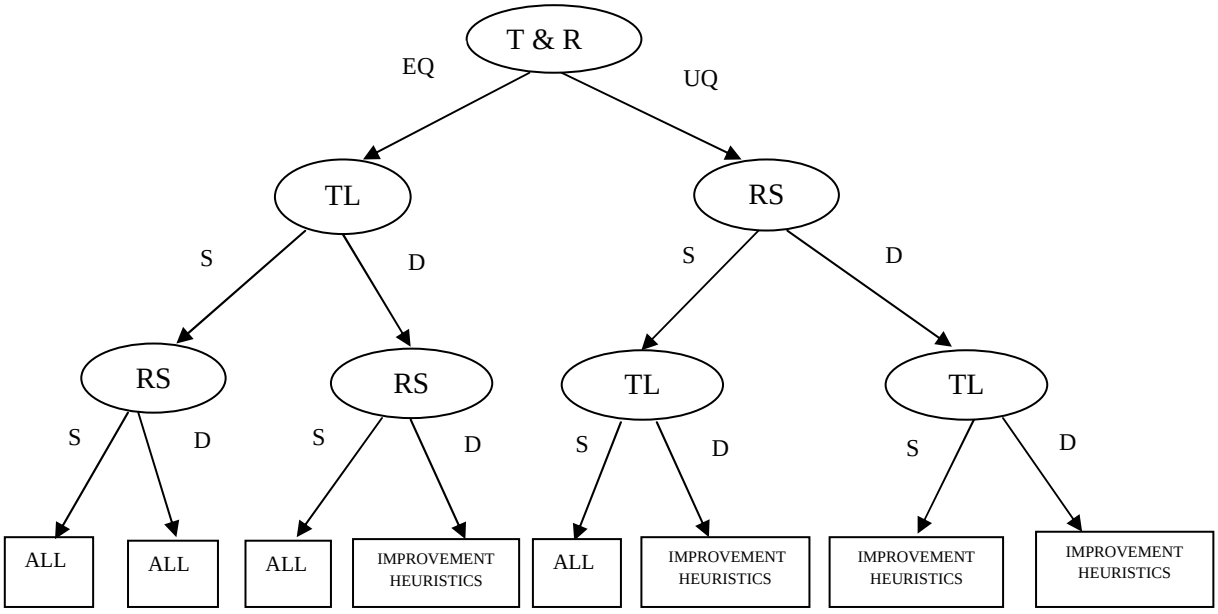

Figure 2 Adaptive machine prediction procedure for scheduling

When the number of tasks and resources are high, in some cases, all the scheduling heuristics perform well and in the remaining cases, BFO and its hybrids [22] perform better. When we have low number of tasks and resources, in some cases, all the scheduling heuristics perform well and in the remaining cases GA or PSO or ACO perform better. The decision tree for scheduling is given in Figure 3. In figure 3, T refers to number of tasks, R refers to number of resources, TL refers to task length, RS refers to resource speed, EQ refers to equal

number of tasks and resources, UQ refers to unequal number of tasks and resources, S refers to same task length/resource speed and D refers to different task length/resource speed.

1. Categorization of problem scenario
    a. Number of tasks = number of resources
        I. Same task length, same resource speed
        II. Same task length, different resource speed
        III. Different task length, same resource speed
        IV. Different task length, different resource speed
    b. Number of tasks ≠ number of resources
        I. Same task length, same resource speed
        II. Same task length, different resource speed
        III. Different task length, same resource speed
        IV. Different task length, different resource speed
2. Match making mechanism
    a. Read task length and resource speed
    b. Compare the user input with the eight classifications
    c. Return the scheduling algorithm whose occurrence exceeds the threshold value
3. Execution of scheduling algorithm
    a. Call the appropriate scheduling algorithm
    b. Execute the scheduling algorithm with user given input
    c. Return the makespan

Figure 3 Decision Tree for Scheduling

IV. RESULTS AND DISCUSSION

The scheduling algorithms in the simulated grid environment were run for 1000 data sets and the results were stored in a database. The data sets consist of ETC matrices from benchmark problems given by Tracy D. Braun et al. [7]. Gridsim toolkit has been used for simulating the grid environment.

4.1 Experimental Setup

Based upon the classification using the decision tree learning technique, seven tables were created for each category. The tables are as follows

Table 1: Equal number of tasks and resources, same task length
Table 2: Equal number of tasks and resources, different task length and same resource speed
Table 3: Equal number of tasks and resources, different task length and different resource speed
Table 4: Unequal number of tasks and resources, same resource speed and same task length
Table 5: Unequal number of tasks and resources, same resource speed and different task length
Table 6: Unequal number of tasks and resources, different resource speed and same task length
Table 7: Unequal number of tasks and resources, different resource speed and different task length

The characteristics of the tasks and resources given by the user will fall in any one of the above Tables 1 to 7. The scheduling heuristics used for simulation includes constructive heuristics such as OLB, MET, MCT, Min-min, Max-min and improvement heuristics such as SA, TS, GA, ACO, ABC and some hybrid heuristics such as Improved ABC, GA+TS, BFO, Min-min BFO and Max-min BFO. The users' inputs consisting of task length and resource speed were got through a common control panel and it was checked whether it exists in the database based on the number, equality and the value of tasks and resources. If the same type of task and resource exists, then the system returns the algorithm that was used to run the users' task. If it is not available,

the system returns a better algorithm with minimum makespan that occurred maximum number of times using the past history. The scheduling algorithm returned by our system was executed and the actual makespan was found out. Finally, the database consists of the task lengths, resource speeds and their corresponding makespan and the scheduling algorithm which gave the minimum makespan.

4.2 Results

Table 1 Accuracy Of The Simulation Tool For Selection Of Better Scheduling Strategy

S. No.	Category of Task-Resource		Accuracy (%)
1	No. of Tasks = No. of Resources	Same TL, Same RS	98%
2		Same TL, Diff. RS	97%
3		Diff. TL, Same RS	96%
4		Diff. TL, Diff. RS	93%
5	No. of Tasks ≠ No. of resources	Same RS, Same TL	94%
6		Same RS, Diff. TL	90%
7		Diff. RS, Same TL	91%
8		Diff. RS, Diff. TL	85%

The simulation was conducted using most of the normal and some hybrid scheduling heuristics. The simulation results are shown in the table 1 and figures 4 and 5. Accuracy is the performance measure used to evaluate our suggested strategy. The strategy suggested in this paper is only for finding out which heuristic is appropriate for certain nature of tasks and resources. In table 1, TL refers to Task Length and RS refers to Resource Speed.

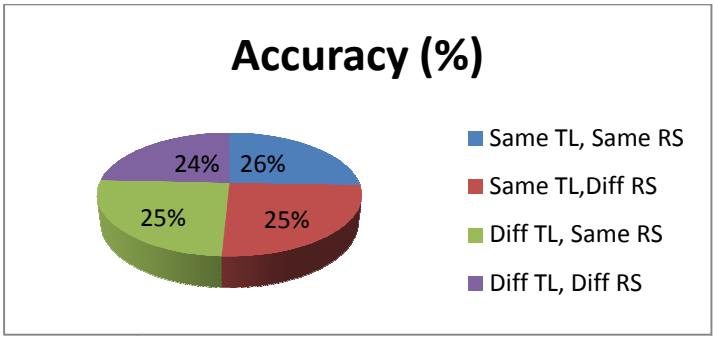

Figure 4. Accuracy of the simulation tool for selection of better scheduling strategy for number of tasks = number of resources

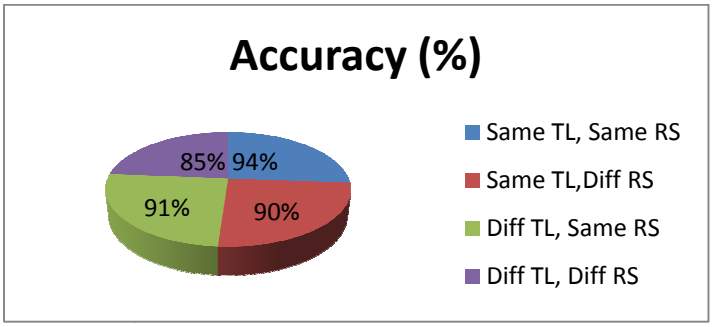

Figure 5. Accuracy of the simulation tool for selection of better scheduling strategy for number of tasks ≠ number of resources

V. Conclusion

Decision tree learning technique has been used to study the behavior of the scheduling heuristics under different circumstances. The scheduling heuristics has been classified based on the nature of the tasks and resources. It was found that for larger number of tasks and resources, BFO and its hybrids performed very well and for smaller number of tasks and resources, the remaining heuristics performed well. All the scheduling heuristics performed well when all the tasks are of same length and all the resources are of the same speed.

References

[1] Ivan Rodero, Francesc Guim, Julita Corbalan, Liana Fong, S.Masoud Sadjadi, Grid broker selection strategies using aggregated resource information, Future Generation Computer Systems 26(2010) 72-86.

[2] M.R. Garey, D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W. H. Freeman & Co., New York, 1979.
[3] Joshy Joseph, Craig Fellenstein, Grid Computing, IBM Press, 2004.
[4] Ahmad I, Ghafoor A, Semi distributed load balancing for massively parallel multi computer systems, IEEE Trans. on Software Engineering, 1991, 987 – 1004.
[5] Ajith Abraham, Hongbo Liu, Weishi Zhang,and Tae –Gyu Chang, Scheduling jobs on computational grids using particle swarm algorithm, 10th International Conference on Knowledge Based Intelligent Information and Engineering Systems, UK, 2006, 500-507.
[6] N. Tonellotto, R.Yahyapour, Ph.Wieder, A proposal for a generic grid scheduling architecture, CoreGRID Technical Report, Number TR-0015, Jan. 2006.
[7] Tracy D.Braun, Howard Jay Siegel, Noah Beck, A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems, Journal of Parallel and Distributed Computing, 2001.
[8] Dalibor Klusacek, Ludek Matyska, Hana Rudova, Local search for deadline driven grid scheduling, In Third Doctoral Workshop on Mathematical and Engineering Methods in Computer Science, 2007.
[9] Vincenzo Di Martino, Sub optimal scheduling in a grid using genetic algorithms, Parallel and Nature Inspired Computational Paradigms and Applications, Elsevier Science Publishers, 2004, 553-565.
[10] Dr.K.Vivekanandan, D.Ramyachitra, A study on performance of job scheduling algorithms in grid environment, International Journal of Computer Science and Information Technology, vol. 3, no. 1, Jan – June 2010.
[11] Paolo Priore, David De La Fuente, Alberto Gomez, Javier Puente, A review of machine learning in dynamic scheduling of flexible manufacturing systems, Journal of Artificial Intelligence for Engineering Design, Analysis and Manufacturing, vol. 15, issue 5, June 2001.
[12] Lee, Wen-Chiung Wu, Chin-Chia Hsu, Peng-Hsiang, A single – machine learning effect scheduling problem with release times, Omega, vol. 38, issue 1-2, Feb. 2010, pages 3 – 11.
[13] Tamaki, H., Kryssanov, V.V., and Kitamura, S. (1999), A simulation engine to support production scheduling using genetics based machine learning, In proceedings of APMS 1999, pp. 482-489.
[14] Jiangtian Li, Xiaosong Ma, Singh K, Schulz M, de Supinski B.R, McKee S.A., Machine learning based online performance prediction for runtime parallelization and task scheduling, IEEE International Symposium on Performance Analysis of Systems and Software, ISPASS 2009, Boston, MA, 26-28 April 2009, pages 89-100.
[15] Diego Puppin, Mark Stephenson, Saman Amarasinghe, Martin Martin and Una-May O'Reilly, Adaptive convergent scheduling using machine learning, Languages and Compilers for Parallel Computing, LNCS, 2004, vol. 2958/2004, pages 17-31.
[16] Ashish Revar, Malay Andhriya, Madhuri Bhavsar, Load balancing in grid environment using machine learning-innovative approach, International Journal of Computer Applications, vol. 8, no. 10, Oct 2010.
[17] Daniel Vladušic, Aleš Cernivec, Boštjan Slivnik, Improving Job Scheduling in GRID Environments with Use of Simple Machine Learning Methods, ITNG, Sixth International Conference on Information Technology: New Generations, pp.177-182, 2009
[18] Guang Xiang, Ge Yu, Xiangli Qu, Xiaomei Dong and Lina Wang, A hybrid machine learning/statistical model of grid security, Grid and Cooperative Computing – GCC 2004, LNCS, vol. 3251/2004, 2004, 348 – 355.Binti Abdullah, N.N, Liquiere, M., and Cerri, S.A. *GRID* Services as Learning Agents: Steps towards the induction of communication protocols. ITS'04: First International Workshop on GRID Learning Services Proceedings. Maceio, Brazil, 2004, pp: 64-77.
[19] Naillah Binti Abdullah, Liquiere M, Cerri S.A., Grid services as learning agents: Steps towards the induction of communication protocols. In proceedings of the 1st Workshop on Grid Learning Service (GLS'04), Maceio, Brazil, 2004.
[20] Guopeng Zhao ,Zhiqi Shen, Chunyan Miao, ELM-Based intelligent resource selection for grid scheduling, International Conference on Machine Learning and Applications, ICMLA'09, Miami Beach, Dec 2009, pages 398 – 403.
[21] Ethem Alpaydin, Introduction to machine learning, MIT Press, Cambridge, MA, U.S.A, 2010.
[22] K.Vivekanandan, D.Ramyachitra, Bacteria Foraging Optimization for Protein Sequence Analysis on the Grid, Future Generation Computer Systems, vol. 28, issue 4, 2012.