FUZZY C-MEAN AND GENETIC ALGORITHMS BASED SCHEDULING FOR INDEPENDENT JOBS IN COMPUTATIONAL GRID

Siriluck Lorpunmanee[1], Mohd Noor Md Sap[2], Abdul Hanan Abdullah[3]

[1]Faculty of Science and Technology, Suan Dusit Rajabhat University,
295 Rajasima Road, Dusit, Bangkok, Thailand, Tel: (662)-2445225, Fax: (662) 6687136
[1]siriluck_lor@dusit.ac.th

[2,3]Faculty of Computer Science and Information Systems, University Technology of Malaysia,
81310 Skudai, Johor, Malaysia, Tel: (607)-5532070, Fax: (607) 5565044
[2]mohdnoor@fsksm.utm.my, [3]hanan@fsksm.utm.my

Abstract: The concept of Grid computing is becoming the most important research area in the high performance computing. Under this concept, the jobs scheduling in Grid computing has more complicated problems to discover a diversity of available resources, select the appropriate applications and map to suitable resources. However, the major problem is the optimal job scheduling, which Grid nodes need to allocate the appropriate resources for each job. In this paper, we combine Fuzzy C-Mean and Genetic Algorithms which are popular algorithms, the Grid can be used for scheduling. Our model presents the method of the jobs classifications based mainly on Fuzzy C-Mean algorithm and mapping the jobs to the appropriate resources based mainly on Genetic algorithm. In the experiments, we used the workload historical information and put it into our simulator. We get the better result when compared to the traditional algorithms for scheduling policies. Finally, the paper also discusses approach of the jobs classifications and the optimization engine in Grid scheduling.

Keywords: Jobs scheduling, Grid Computing, Fuzzy C-Mean algorithm, Genetic algorithm, High performance computing.

1. Introduction

Grid Computing is the principle that occurs for a long period of time by focusing on virtual organizations [1] to share large-scale resources, innovating applications and in some cases getting high-performance orientation. Under this principle, Grid has problem in flexible, secure, coordinated resource sharing among dynamic collections of individuals, institutions, and resources. In Grid [2] concept is a new generation of technologies combine physical resources and applications that provide vastly more effective solutions to complex problems (e.g., scientific, engineering and business).

These new technologies must be built on secure discovery, jobs allocates to resources, integration resources and services from the others. In [3] is a formal definition of Grid

concepts. There define conceptual models are abstract machines that support applications and services. Fig. 1, taken from [3], is formally defined (e.g., Organization, Virtual Organization, Virtual Machine, Programming System, etc.). Currently, Global Grid Forum [4] formulated and provided standards documents of virtual organization.

Grid Computing goes beyond distributing and sharing resources to more applications, which an applications are adapting to use Grid resources. Although, using distributed and shared resources are usefully but this is possible only if the resources and the jobs in Grid computing are scheduled which is very well. The optimal scheduler will be making high performance in grid computing as the poor scheduler will be making contrast result. Currently, the Grid scheduling is big topic in Grid environment for new algorithm model**.**

The Grid scheduling is responsible for resources discovery, resources selection, and job assignment over a decentralized heterogeneous system, which resources belong to multiple administrative domains. Normally, the resources are requested by a Grid application, which use to computing. However, Grid scheduling of applications is absolutely more complex than scheduling an applications of a single computer. Because resources information of single computer scheduling is easy to get information, such as CPU frequency, number of CPU a machine, memory size, memory configuration and network bandwidth etc. But Grid environment is dynamic resources sharing and distributing. Then an application is hard to get resources information, such as CPU load, available memory, available network capacity etc. And Grid environment also hard to classify jobs characteristic, that run in Grid. There are basically two approaches to solve this problems, the first is based on jobs characteristic and second is based on a distributed resources discovery and allocation system. It should optimize the allocation of a job allowing the execution on the optimization of resources. The scheduling in Grid environment has to satisfy a number of constraints on different problems.

We have defined a set of them to study the feasibility and the usefulness of applying fuzzy logic techniques to this field. It's worth pointing out that this is not means a complete characterization of the real world problem, the subset of constraints that we have considered provide a first insight into the usefulness of Fuzzy C-Mean Clustering techniques for Grid scheduling. As, we have defined a set of them to study the optimization of applying Genetic algorithm to Grid scheduling problem.

In this paper, we compared the performance of various jobs scheduling algorithms in Grid environment with our algorithm. We consider to three job scheduling algorithms are the First-Come-First-Served (FCFS), Largest-Job-First (LJF) and Shortest-Job-First (SJF).

In this paper we discussed in jobs characteristics for such a system. To this end, we are reviewing a related work in Section 2. Next, Section 3 we brief the clustering conceptual. Section 4, we brief the Genetic Algorithm concept. Section 5, show the design and

architecture. Section 6, we show result of experiment. Section 7 concludes the paper by summarizing this work and providing a preview of future research in this area.

2. Related Works

Most of the predictions job scheduling in Grid environment based on job execute time and job run time, such as [5] there proposed module prediction engine will be a part of such a scheduling and there module offer a history based approach for estimating the run time of job submission. In [6] proposed two modules for predicting the completion time of jobs in a service Grid and applying evolutionary techniques to job scheduling. The problem of estimating jobs run time from historical data has been in [7][8][9]. All of them adopt the method of making predictions for future jobs by applying different jobs characterization to classify similar old jobs that have been executed before and use them make the predictions. We noticed that their methods unclear definitions of jobs that what should be important features used in measuring jobs similarity and irrelevant old jobs features to take part in the prediction module.

In [10] presented an approach to predicting application performance and enabling the detection of unexpected execution behavior, typically caused by unanticipated load on shared grid resources. The other for predicting application performance on a given parallel system has been the most widely studied in the past [11, 12]. More recently those studies have been extended to distributed systems [13, 14].

Traditional applications performance prediction techniques often focus on performance models that are specific to a single architecture or a static set of resources. However, computational grid environments consist of a collection of dynamic, heterogeneous resources and a collection of dynamic jobs. Our approach, especially examine the implications of the fact that the characterization of jobs is expected to have an impact on the mentioned resource utilization and, even more interestingly for researcher on the performance quality. We use information about static workload data from the Standard Workload Archive [18] and it has been experimented in several publications [19][20][21][22]. Moreover, these workload traces were used for the evaluation of different scheduling strategies for single parallel systems [23], [24], [25], [26] and for Grid research [27], [28], [29]. These workload traces consists of information about all job submissions on a machine for a certain period of time which usually ranges over several months and several thousands of jobs. Therefore, it is reasonable to start with the available workload traces information from the compute centers to evaluate the impact of jobs characterization in Grid environment. Our approach separated workload data to three classifications base on jobs run-time historical data.

3. Clustering

Clustering algorithms refer to automatic unsupervised classification method in data set and it can be classification into different categories data set base on the type of input parameters, type of clusters they are interested.

Clustering algorithms for data sets can be found in different fields, such as statistics, computer science, bioinformatics and machine learning. The most famous clustering algorithm is the Fuzzy C-Mean algorithm.

This paper aims to introduce cluster analysis for classification of jobs characteristic, or objects, according to similarities among them, and organizing objects into groups. A cluster is a group of objects that are more similar to each other than to objects in other clusters. Similarity is often defined by means of distance based upon the length from a data vector to some prototypical object of the cluster.

The data are typically observations of some phenomenon. Each object consists of *m* measured variables, grouped into an *m*-dimensional column vector $x_i=\{x_{i1}, x_{i2,\ldots,} x_{in}\}$. A set of *n* objects is denoted by

$$U= x_i= \{x_{i1}, x_{i2,\ldots,} x_p\}$$

$$X = [x_{ij}] = \begin{pmatrix} x_{1,1} & x_{1,2} & \ldots & x_{1,n} \\ x_{2,1} & x_{2,2} & \ldots & x_{2,n} \\ \ldots & \ldots & \ldots & \ldots \\ x_{p,1} & x_{p,2} & \ldots & x_{p,n} \end{pmatrix}$$

To distinguish between labeled and unlabeled patterns we will introduce a two-valued (Boolean) indicator vector $b = [b_k], k = 1, 2, ..N$ with 0 -1 entries in the following manner.

$$b_k = \begin{cases} 1 & \text{if pattern } x_k \text{ is labeled} \\ 0 & \text{otherwise} \end{cases}$$

3.1 Fuzzy *c*-means (FCM) clustering

The Fuzzy C-Means is one of the existing clustering methods for building fuzzy partitions. This method will be used in this paper as the basic tool for building jobs characterization in Grid environment. Fuzzy C-Means (FCM) algorithm, also known as fuzzy ISODATA, was introduced by Bezdek [15] as extension to Dunn's [16] algorithm to generate fuzzy sets for every observed feature.

Fuzzy clustering methods allow for uncertainty in the cluster assignments. Rather that partitioning the data into a collection of distinct sets (where each data point is assigned to exactly one set), fuzzy clustering creates a fuzzy pseudo partition, which consists of a collection of fuzzy sets. Fuzzy sets differ from traditional sets in that membership in the set is allowed to be uncertain. A fuzzy set is formalized by the following definitions.

Let $X=\{x_1,x_2,...,x_n\}$ be a set of given data, where $x_i \in \Re^n$ is a set of feature data. The minimization objective function of FCM algorithm is frequently used in pattern recognition follow as.

$$J(U,V)=\sum_{j=1}^{C}\sum_{i=1}^{N}(\mu_{ij})^m \|x_i-v_j\|^2; \quad 1\le m<\infty \tag{1}$$

Where m is any real number greater than 1, $V=\{v_1,v_2,...,v_C\}$ are the cluster centers. $U=(\mu_{ij})_{NxC}$ is the degree of membership of vector x_i in the cluster j. The values of matrix U should satisfy the following conditions.

$$\mu_{ij} \in [0,1]; \forall_i = 1,2,...,N; \forall_j = 1,2,...,C \tag{2}$$

$$\sum_{j=1}^{C}\mu_{ij}=1; \forall_i = 1,2,...,N \tag{3}$$

The Euclidean distance $d_{ij} = ||x_i - v_j||$ is any norm expressing the similarity between any measured data and the center and calculate the cluster centers $\boldsymbol{V}$ according to the equation:

$$v_j=\frac{\sum_{i=1}^{N}(\mu_{ij})^m x_i}{\sum_{i=1}^{N}(\mu_{ij})^m}; \forall_j = 1,2,...,C \tag{4}$$

Fuzzy partitioning is carried out through an iterative optimization of the objective function shown above, with the update the fuzzy partition matrix U by:

$$u_{ij}=\frac{1}{\sum_{k=1}^{C}\left(\frac{d_{ij}}{d_{ik}}\right)^{\frac{2}{m-1}}} \tag{5}$$

This algorithm will stop if $\max_{ij}\left\|\mu_{ij}^{(k+1)}-\mu_{ij}^{(k)}\right\|<\phi$; $\phi \in [0,1]$ and k is the iteration step. All equation shown above, converge to Fuzzy C-Mean algorithm by following step:

Algorithm: Fuzzy *C*-Means
Step 1: Generate an initial U and V
Step 2: At k-step: calculate the centers vectors $C^{(k)}=[v_j]$ with $U^{(k)}$:

$$v_j=\frac{\sum_{i=1}^{N}(\mu_{ij})^m x_i}{\sum_{i=1}^{N}(\mu_{ij})^m}$$

Step 3: Update $U^{(k)}$, $U^{(k+1)}$

$$u_{ij}=\frac{1}{\sum_{k=1}^{C}\left(\frac{d_{ij}}{d_{ik}}\right)^{\frac{2}{m-1}}}$$

Step 4: If $||U^{(k+1)} - U^{(k)}|| < \phi$ then STOP;
otherwise return to step 2.

Figure 1: Fuzzy *c*-means [17]

4. Genetic Algorithms

Genetic algorithms (GA) are a class of stochastic search algorithms which based on biological evolution [18], [19]. A basic GA can be represented as in Fig 7.2 taken from [20]. Genetic algorithms combine the exploitation of past results with the exploration of new areas of the search space by using survival of the fittest techniques combined with a structured of randomized information exchange, a GA can mimic some of the innovative intelligence of human search. A generation is a collection of artificial creatures (strings). In every new generation a set of strings is created using information from the previous times. Occasionally a new part is tried for good measure. GAs are randomized, but they are not simple random walks. They efficiently exploit historical information to speculate on new search points with expected improvement.

The approach used in this work generates a set of initial scheduling, evaluates the scheduling to obtain a measure of fitness, selects the most appropriate and combines then together using operators (crossover and mutation) to formulate a new set of solutions. The basic type of GAs, known as the simple GA (SGA), uses a population of binary strings, single point crossover, and proportional selection [18], [19]. Many other modifications to the SGA have been proposed; some of these are adopted in our work. The following subsections explain the steps in our proposed approach;

```
GA( )
    {
        Generate initial population of Jobs individuals
        Evaluate individuals according to fitness function;
        While stopping condition is satisfied.
        {
            Count from 1 to amount generation;
            Select two parents from initial population
             (Population →Parent1 and Parent2);
            Crossover (Parent1 and Parent2) →Child;
            Mutation (Parent p, Parent q, Child);
            Fitness (Child c, Best Chromosome bc);
            Improvement (Child c);
            Replace (Chromosome chrom, Child c);
            Scheduling(best chromosome);
        }
    } return set of the best chromosome in population for jobs scheduling.
```

5. Design and Algorithm

This section will briefly describe how workload data will separate to three classifications and will compute in Grid environment by using our algorithm. Let us consider a set of jobs of workload, a set of heterogeneous computing system. Jobs are subject to constraints on the Grid environment then we selected a set of properties of jobs and descriptions relevant to real world situations, an example dataset is presented as in Table 1.

Table 1: data set of the jobs workload.

Instance	Properties type	Decision field
1	Job Number	Yes
2	Submit Time	No
3	Run Time	Yes
4	Number of allocated processors	No
5	User ID	Yes
6	Group ID	No

Let $J = \left[j_{xy} \right] = \begin{bmatrix} j_{11} & j_{12} & \cdots & j_{1y} \\ j_{21} & j_{22} & \cdots & j_{2y} \\ \cdots & \cdots & \cdots & \cdots \\ j_{x1} & j_{x2} & \cdots & j_{xy} \end{bmatrix}$ be a set of given jobs, where y is the y^{th} of User ID and x is the x^{th} of Job Number. As a Grid environment consists of the vector value of n nodes and a node consists of m resources then following as:

$$\vec{G} = \begin{bmatrix} N_1 \\ N_2 \\ N_3 \\ \vdots \\ N_n \end{bmatrix} \text{ and } \vec{N} = \begin{bmatrix} R_1 & R_2 & R_3 & \cdots & R_m \end{bmatrix} \text{ then } \vec{G} = \begin{bmatrix} R_{11} & R_{12} & \cdots & R_{1m} \\ R_{21} & R_{22} & \cdots & R_{2m} \\ \cdots & \cdots & \cdots & \cdots \\ R_{n1} & R_{n2} & \cdots & R_{nm} \end{bmatrix}$$

In previous assumptions, we discussed jobs workload in Grid environment. And we concentrated in classification of jobs workload based on each job utilized resource. We separated jobs workload to three classifications, first is heavy workload, second is medium workload and third light workload by using Fuzzy C-Mean algorithm. In the same time, we use the same algorithm calculate the appropriate resources within three classifications based mainly on the performance of Grid nodes. Finally, we arrange suitable jobs to appropriate resources in Grid environment by using Genetic algorithm. Our simulation, we simulated different performance nodes in Grid environment and we use the jobs workload traces from [18].

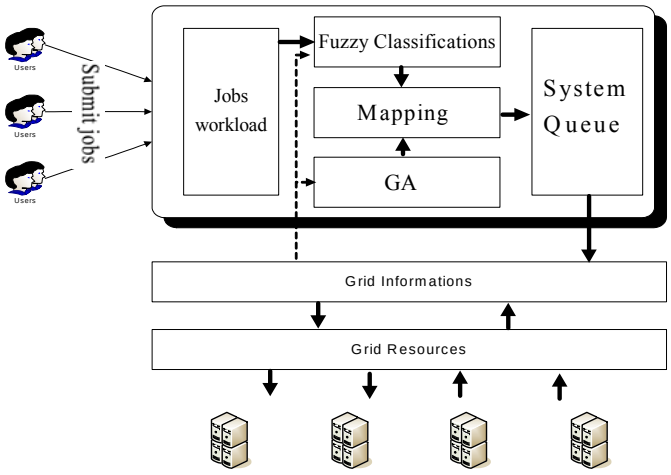

Figure 2: Our jobs scheduling algorithm Model.

5.1 Scheduling algorithms

We consider to three job scheduling algorithms are FCFS, LJF and SJF. A job J is $\lfloor j_{xy} \rfloor$, where each of j_{xy} is specified as t_{xy}, where t_{xy} is the job's run-time at x^{th} of job number and y is the y^{th} of User ID. We are interested in scheduling that jobs j_{xy} allocate to n nodes in Grid environment.

We choose on the initial order of the jobs (run-time) in J. We consider three job scheduling algorithms, namely, FCFS, SJF and LJF. Following as:

1. First-Come-First-Serve (FCFS) : jobs are arranged such that $j_1, j_2, j_3, \ldots, j_x$.
2. Shortest-Job-First (SJF) : jobs are arranged such that $t_1 \le t_2 \le t_3 \le \ldots \le t_x$.
3. Largest-Job-First (LJF) : jobs are arranged such that $t_1 \ge t_2 \ge t_3 \ge \ldots \ge t_x$.

Then the algorithms of there, which are.

Input: $J = \lfloor j_{xy} \rfloor = \begin{bmatrix} j_{11} & j_{12} & \cdots & j_{1y} \\ j_{21} & j_{22} & \cdots & j_{2y} \\ \cdots & \cdots & \cdots & \cdots \\ j_{x1} & j_{x2} & \cdots & j_{xy} \end{bmatrix}$, where as,

$j_1, j_2, j_3, \ldots, j_x$ for FCFS;

$t_1 \le t_2 \le t_3 \le \ldots \le t_x$ for SJF;

$t_1 \ge t_2 \ge t_3 \ge \ldots \ge t_x$ for LJF.

Output: waiting time and make span time.

5.2 Our Scheduling algorithms

Fuzzy C-Mean and GA( ) {

Input: $J = \lfloor j_{xy} \rfloor = \begin{bmatrix} j_{11} & j_{12} & \cdots & j_{1y} \\ j_{21} & j_{22} & \cdots & j_{2y} \\ \cdots & \cdots & \cdots & \cdots \\ j_{x1} & j_{x2} & \cdots & j_{xy} \end{bmatrix}$, where as each of jobs order in job number.

Find classification of workload by using Fuzzy C-Mean, such that

1. Heavy workload: Group of jobs is large job run-time. Such as $t_{H1}, t_{H2}, \ldots, t_{Ha}$; $1 \le a \le xy$
2. Medium workload: Group of jobs is medium job run-time. Such as $t_{M1}, t_{M2}, \ldots, t_{Mb}$; $1 \le b \le xy$
3. Light workload: Group of jobs is short job run-time. Such as $t_{L1}, t_{L2}, \ldots, t_{Lc}$; $1 \le c \le xy$

while (jobs ≤xy) {

if (Heavy workload) {

order jobs run-time:

$t_{H1} \le t_{H2} \le t_{H3} \le \ldots \le t_{Hx}$

}

if (Medium workload){

order jobs run-time:

$$t_{M1} \le t_{M2} \le t_{M3} \le \ldots \le t_{Mx}$$

}

if (Light workload) {

order jobs run-time:

$$t_{L1} \le t_{L2} \le t_{L3} \le \ldots \le t_{Lx}$$

}

New order jobs :

$$t_{H1} \le t_{H2} \le t_{H3} \le \ldots \le t_{Hx} ,$$

$$t_{M1} \le t_{M2} \le t_{M3} \le \ldots \le t_{Mx} ,$$

$$t_{L1} \le t_{L2} \le t_{L3} \le \ldots \le t_{Lx}$$

}

Mapping new order jobs to Grid Environment by using GA. As, searching system utilization and system performance classifications in Grid environment are calculated by using Fuzzy C-Mean as well the same algorithm.

}

Output: $WaitingTime_{j_1} = WaitingTime_{j2} = WaitingTime_{j3} = \ldots = WaitingTime_{jxy}$

Min(Makespan$_n$), where as n is each of node in Grid environment.

$$Min\left(\frac{\sum_{j=1}^{xy} WaitingTim\ \ e_j}{xy}\right), \text{ and Min(Makespan}_n\text{)}$$

6. Experimental setup and results

In this experiment, we used jobs workload data from the Standard Workload Archive [18]. This data consists of 18,239 jobs, each of which has 18 properties field, however we focused on some properties that previous mention. In our experiments we assumed that each of jobs is allowed to run in each node by using space-sharing mechanism. Our simulation, we simulated 500 different performance nodes in Grid environment.

Our experiments showed classification of jobs workload to three groups, first is heavy workload, second is medium workload and third is light workload. We compared the performance of Grid system using our method with traditional method, such as FCFS, LJF and SJF.

Figure 3 shows the jobs characterization by using Fuzzy C-Mean algorithm separated workload data to three groups, (a) is heavy workload, (b) is medium workload and (c) is light workload.

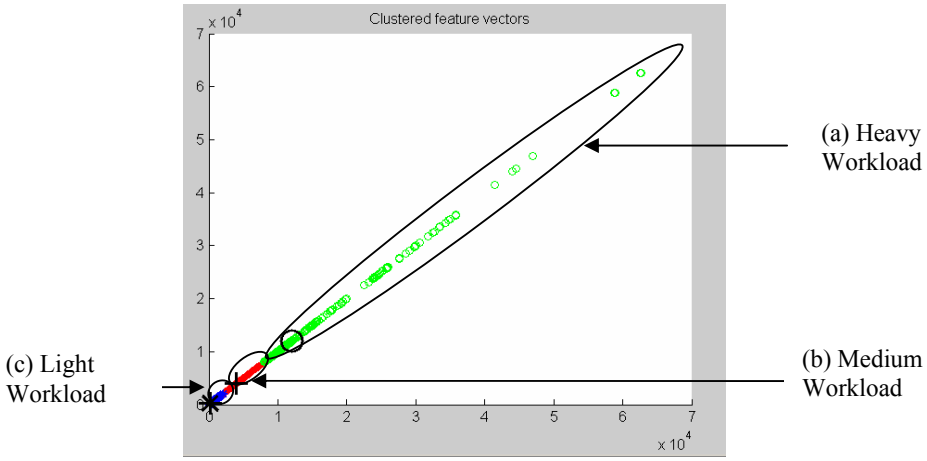

Figure 3: the jobs characteristics by using Fuzzy C-Mean

In this experiment, the jobs in workload data is predicted to three classification, heavy workload is Run-Time from 62,643 to 8,021 and total jobs are 503 jobs, Medium workload from 7,979 to 2,069 and total jobs are 1,056 jobs and Light workload from 2,061 to 1 and total jobs are 16,507, can be seen Table 2.

Table 2: the jobs classifications in Grid environment

No.	Classification	Total
1	Heavy Workload	503
2	Medium Workload	1,056
3	Light Workload	16,507

In Figure 4, shows the distribution of waiting time for each of jobs in Grid environment, and shows the comparisons of waiting time of different algorithms (FCFS, SJF, LJF and Fuzzy C-Mean + GA). It can be seen from Figure 4 that our algorithm was mush faster than the traditional algorithms.

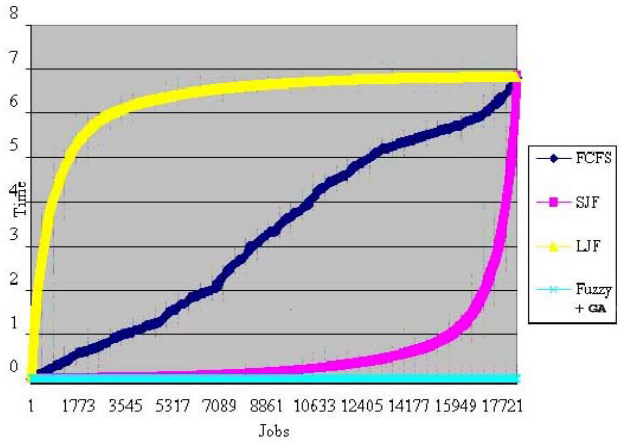

Figure 4: The waiting time of jobs in Grid environment

In Figure 5, shows the make-span time of jobs run in each of nodes of Grid environment. There trends may indicate the fastest of each nodes by using our algorithm.

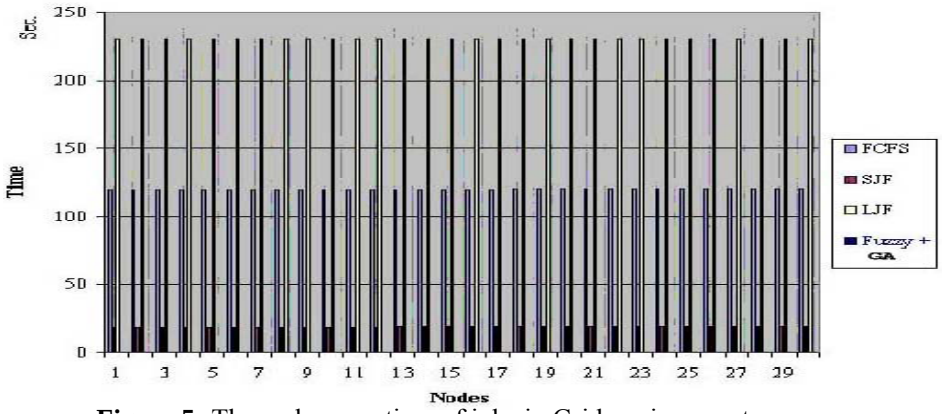

Figure 5: The make span time of jobs in Grid environment

7. Conclusions

We have studied the jobs scheduling problem for Grid environment as a combinatorial prediction and optimisation based on Fuzzy C-Mean and GA technique. Several observations are in order.

a. We used jobs workload from the Standard Workload Archive [18] on space-sharing mechanism. We show that the proposed model captures the jobs characterization of real workload in three different classifications. This model can be used in our algorithm to simulate and evaluate scheduling policies for simulating Grid environment.

b. Many scheduling algorithms for Grid environment depend on static information provided by the Standard Workload Archive [18] and the different performance nodes in Grid, simulating by us. We observe that this information is often unreliable. However, it is a useful way.

c. The experimental results on the waiting time and the make-span time have shown that the scheduling using our algorithm can allocate the best results as compared with traditional scheduling such as FCFS, SJF, LJF.

d. The results provided here suggest that the researcher look forward to new method for such problems should consider combine them with their method.

Our future works will focus on the following:

a. Our simulation environment will include critical parameters, such as submit time, Grid network cost, job migrations overhead, fault tolerance and workload of network.

b. We plan to investigate the evolution mechanism, such as genetic algorithm, ant colony algorithm and the nature's heuristic algorithms for Grid scheduling.

c. We will include a more complex characterization of the constraints for Grid scheduling and will improve the complexity problems in Grid environment.

ACKNOWLEDGMENT

We would like to thank Suan Dusit Rajabhat University for supported us.

References

[1] Foster I, Kesselman C, Tuecke S. The anatomy of the Grid: Enabling scalable virtual organizations. International Journal of Supercomputer Applications 2001.

[2] I. Foster and C. Kesselman, Eds., The Grid 2: Blueprint for a New Computing Infrastructure. San Francisco, CA: Morgan Kaufmann, 2004.

[3] M. Parashar, J. Browne, Conceptual and Implementation Models for the Grid.

[4] Global Grid Forum [Online]. Available: http://www.gridforum.org.

[5] ArshadAli, Ashiq Anjum, Julian Bunn, R. Cavanaugh, Frank van Lingen, R. McClatchey, Muhammad Atif Mehmood, H. Newman, C. Steenberg, M. Thomas, I. Willers. PREDICTING THE RESOURCE REQUIREMENTS OF A JOB SUBMISSION.

[6] Y. Gao, H. Rong, Joshua Zhexue Huang. Adaptive grid job scheduling with genetic algorithms.

[7] Downey, A, Predicting Queue Times on Space-Sharing Parallel Computers, in International Parallel Processing Symposium, 1997.

[8] Gibbons, R., A Historical Profiler for Use by Parallel Schedulers, Master's thesis, University of Toronto, 1997.

[9] Smith, W., I. Foster, and V. Taylor, Predicting Application Run Times Using Historical Information, Proceedings of the IPPS/SPDP '98 Workshop on Job Scheduling Strategies for Parallel Processing, 1998.

[10] Vraalsen, F., R.A. Aydt, C.L. Mendes, and D.A. Reed, Performance Contracts: Predicting and Monitoring Grid Application Behavior, GRID 2001: Proceedings of the Second International Workshop on Grid Computing, 2001.

[11] Mehra, P., Schulbach, C. H., and Yan, J. C. A Comparison of Two Model-Based Performance-Prediction Techniques for Message-Passing Parallel Programs. In Proceedings of the ACM Conference on Measurement & Modeling of Computer Systems - SIGMETRICS'94 (Nashville, May 1994), pp. 181–190.

[12] Saavedra-Barrera, R. H., Smith, A. J., and Miya, E. Performance prediction by benchmark and machine characterization. IEEE Transactions on Computers 38, 12 (December 1989), 1659–1679.

[13] Kapadia, N., Fortes, J., and Brodley, C. Predictive Application-Performance Modeling in a Computational Grid Environment. In Proceedings of the Eight IEEE Symposium on High-Performance Distributed Computing (Redondo Beach, California, August 1999), pp. 47–54.

[14] Petitet, A., Blackford, S., J.Dongarra, Ellis, B., Fagg, G., Roche, K., and Vadhiyar, S. Numerical Libraries and The Grid: The GrADS Experiments with ScaLAPACK. Tech. Rep. UT-CS-01-460, University of Tennessee, April 2001.

[15] J. Bezkek, "Pattern Recognition with Fuzzy Objective Function Algorithms", Plenum Press, USA, 1981.

[16] J.C. Dunn, "A Fuzzy Relative of the ISODATA process and its Use in Detecting Compact", Well Separated Clusters, Journal of Cybernetics, Vol. 3, No.3, 1974, pp. 32-57.

[17] A Tutorial on Clustering Algorithms: "Fuzzy C-Means Clustering".

http://www.elet.polimi.it/upload/matteucc/Clustering/tutorial_html/index.html.

[18] Parallel Workloads Archive. http://www.cs.huji.ac.il/labs/parallel/workload/, Juni 2004.

[19] J. Jann, P. Pattnaik, H. Franke, F. Wang, J. Skovira, and J. Riordan. Modeling of Workload in MPPs. In D. Feitelson and L. Rudolph, editors, IPPS'97 Workshop: Job Scheduling Strategies for Parallel Processing, pages 94–116. Springer–Verlag, Lecture Notes in Computer Science LNCS 1291, 1997.

[20] C. Ernemann, B. Song, and R. Yahyapour. Scaling of Workload Traces. In D. G. Feitelson, L. Rudolph, and U. Schwiegelshohn, editors, Job Scheduling Strategies for Parallel Processing: 9th International Workshop, JSSPP 2003 Seattle, WA, USA, June 24, 2003, volume 2862 of Lecture Notes in Computer Science (LNCS), pages 166–183. Springer-Verlag Heidelberg, October 2003.

[21] D. G. Feitelson and A. M. Weil. Utilization and Predictability in Scheduling the IBM SP2 with Backfilling. In Proc. 12th Int'l Parallel Processing Symp. and the 9th Symp. on Parallel and Distributed Processing, pages 542–547, Los Alamitos CA, 1998. IEEE Computer Society Press.

[22] D. G. Feitelson and B. Nitzberg. Job Characteristics of a Production Parallel Scientific Workload on the NASA Ames iPSC/860. In D. G. Feitelson and L. Rudolph, editors, IPPS'95 Workshop: Job Scheduling Strategies for Parallel Processing, pages 337–360. Springer, Berlin, Lecture Notes in Computer Science LNCS 949, 1995.

[23] J. Krallmann, U. Schwiegelshohn, and R. Yahyapour. On the Design and Evaluation of Job Scheduling Systems. In D. G. Feitelson and L. Rudolph, editors, IPPS/SPDP'99 Workshop: Job Scheduling Strategies for Parallel Processing. Springer, Berlin, Lecture Notes in Computer Science, LNCS 1659, 1999.

[24] U. Schwiegelshohn and R. Yahyapour. Analysis of First-Come-First-Serve Parallel Job Scheduling. In Proceedings of the 9th SIAM Symposium on Discrete Algorithms, pages 629–638, January 1998.

[25] U. Schwiegelshohn and R. Yahyapour. Improving First-Come-First-Serve Job Scheduling by Gang Scheduling. In D. Feitelson and L. Rudolph, editors, IPPS'98 Workshop: Job Scheduling Strategies for Parallel Processing, pages 180–198. Springer–Verlag, Lecture Notes in Computer Science LNCS 1459, 1998.

[26] D. G. Feitelson, L. Rudolph, U. Schwiegelshohn, and K. C. Sevcik. Theory and Practice in Parallel Job Scheduling. Lecture Notes in Computer Science, 1291:1–34, 1997.

[27] V. Hamscher, U. Schwiegelshohn, A. Streit, and R. Yahyapour. Evaluation of Job-Scheduling Strategies for Grid Computing. In Proc. 7th Int'l Conf. on High Performance Computing, HiPC-2000, pages 191–202, Bangalore, Indien, 2000. Springer, Berlin, Lecture Notes in Computer Science LNCS 1971.

[28] C. Ernemann, V. Hamscher, U. Schwiegelshohn, A. Streit, and R. Yahyapour. On Advantages of Grid Computing for Parallel Job Scheduling. In Proc. 2nd IEEE/ACM Int'l Symp. on Cluster Computing and the Grid (CCGRID2002), Berlin, May 2002. IEEE Press.

[29] C. Ernemann, V. Hamscher, U. Schwiegelshohn, A. Streit, and R. Yahyapour. Enhanced Algorithms for Multi-Site Scheduling. In Proceedings of the 3rd International Workshop on Grid Computing, Baltimore. Springer–Verlag, Lecture Notes in Computer Science LNCS, 2002.