Optimalisation of a Job Scheduler in the Grid Environment by Using Fuzzy C-Mean

Siriluck Lorpunmanee¹, Mohd Noor Md Sap², Abdul Hanan Abdullah²

¹Faculty of Science and Technology, Suan Dusit Rajabhat University, Dusit, Bangkok, Thailand; ²Faculty of Computer Science and Information Systems, University Technology of Malaysia, Johor, Malaysia

Abstract: Grid computing is the principle in utilizing and sharing large-scale resources to solve complex scientific problems. Under this principle, Grid environment has problems in flexible, secure, coordinated resource sharing among dynamic collections of individuals, institutions, and resources. However, the major problems include optimal job scheduling, and which grid nodes allocate the resources for each job. This paper proposes the model for optimizing jobs scheduling in Grid environment. The model presents the results of the simulation of the Grid environment of jobs allocation to different nodes. We develop the results of job characteristics to three classifications depending on jobs run time in machines, which have been obtained using the optimization of jobs scheduling. The results prove the model by using Fuzzy c-mean clustering technique for predicting the characterization of jobs and optimization of jobs scheduling in Grid environment. Simulation runs demonstrate that our algorithm leads to better results than the traditional algorithms for scheduling policies used in Grid environment.

Keywords: Job scheduling, Grid Computing, Fuzzy c-mean technique, job characteristics, large-scale resources.

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 a problem in flexible, secure, coordinated resource sharing among dynamic collections of individuals, institutions, and resources. In Grid 2 concept
[2], a new generation of technologies combines 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 allocated to resources, and integration of resources and services from the others. In [3], there is a formal definition of Grid concepts. They are defined as conceptual models in abstract machines that support applications and services. In Figure 1, taken from [3], they are formally defined (e.g., Organization, Virtual Organization, Virtual Machine, Programming System, etc.). Currently, Global Grid Forum [4] formulates and provides standard documents of virtual organization.

Algorithm 2 Fuzzy C-Means

Step 1: Generate an initial $U$ and $V$

Step 2: At k-step, calculate the center vectors $C^{(k)} = \{v_j\}$ with $U^{(k)}$:

$$
V_j = \frac{\sum_{i=1}^{N} (u_{ij})^m x_i}{\sum_{j=1}^{m} (u_{ij})^m}
$$

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

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

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

Figure 1. Fuzzy C-Means [17]
allocation system. It should optimize the allocation of a job allowing the execution on the optimization of resources. The scheduling in Grid environment has therefore to satisfy a number of constraints on different problems. We have defined a number of them to study the feasibility and the usefulness of applying fuzzy logic techniques to this field. It’s worth pointing out that this does not mean a complete characterization of the real world problems. The subset of constraints that we have considered provides a first insight into the usefulness of Fuzzy C-Mean Clustering techniques for Grid scheduling.

In this paper, we compare the performance of various job scheduling algorithms in Grid environment with our algorithm. We consider three job scheduling algorithms, the First-Come-First-Served (FCFS), Largest-Job-First (LJF) and Shortest-Job-First (SJF). We also discuss job characteristics for such a system. To this end, we review related work in Section 2. In Section 3 we briefly discuss the clustering concept. Section 4 shows the design and architecture. In Section 5, we show the results of experiments. Section 6 concludes the paper by summarizing this work and providing a preview of future research in this area.

Related Work

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

In [10] an approach is presented to predict application performance enabling the detection of unexpected execution behavior, typically caused by unanticipated load on shared grid resources. Another approach for predicting application performance on a given parallel system, which has been the most widely studied, is presented in [11] and [12]. More recently those studies have been extended to distributed systems [13] and [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 examines 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 researchers, on the performance quality. We use information about static workload data from the Standard Workload Archive [18], which has been experimented in several publications [19], [20], [21], and [22]. Moreover, these workload traces were used for the evaluation of different scheduling strategies for single parallel systems [23], [24], [25], and [26], and for Grid research [27], [28], and [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 of information from the compute centers to evaluate the impact of job characterizations in Grid environment. Our approach separates workload data to three classifications based on jobs run-time historical data.

**Clustering**

Clustering algorithms refer to automatic unsupervised classification methods in data sets, which can be classified into different categories based on the type of input parameters and type of clusters. 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 job characteristics, or objects, according to similarities among them, and for 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 \( U = \{ x_{i1}, x_{i2}, \ldots, x_p \} \)

\[
X = [x_{ij}] = \begin{pmatrix}
  x_{11} & x_{12} & \cdots & x_{1n} \\
  x_{21} & x_{22} & \cdots & x_{2n} \\
  \vdots & \vdots & \ddots & \vdots \\
  x_{p1} & x_{p2} & \cdots & x_{pn}
\end{pmatrix}
\]

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

\[
b_k = \begin{cases}
  1 & \text{if pattern } x_i \text{ is labeled} \\
  0 & \text{otherwise}
\end{cases}
\]

**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 job characterizations in Grid environment. Fuzzy C-Means (FCM) algorithm, also known as fuzzy ISODATA, 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 than 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 $X = \{x_1, x_2, \ldots, x_n\}$ be a set of given data, where $x_i \in \mathbb{R}^n$ is a set of feature data. The minimization objective function of FCM algorithm is frequently used in pattern recognition as follows:

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

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

$$\mu_{ij} \in [0,1] ; \forall i = 1,2,\ldots,N ; \forall j = 1,2,\ldots,C \text{ .............................................. (2)}$$

$$\sum_{i=1}^{N} \mu_{ij} = 1 ; \forall j = 1,2,\ldots,N \text{ ................................................. (3)}$$

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

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

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

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

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

**Design and Algorithm**

This section will briefly describe how workload data will be classified into three categories and will be computed in Grid environment by using our algorithm. Let us consider a set of jobs of workload, and a set of heterogeneous computing systems. Jobs are subject to constraints on the Grid environment. We selected a set of properties of jobs and descriptions relevant to real world situations. An example of dataset is presented in Table 1.
Table 1: Dataset of job properties representation

<table>
<thead>
<tr>
<th>Instance</th>
<th>Properties type</th>
<th>Decision field</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Job Number</td>
<td>Yes</td>
</tr>
<tr>
<td>2</td>
<td>Submit Time</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>Run Time</td>
<td>Yes</td>
</tr>
<tr>
<td>4</td>
<td>Number of allocated</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>processors</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>User ID</td>
<td>Yes</td>
</tr>
<tr>
<td>6</td>
<td>Group ID</td>
<td>No</td>
</tr>
</tbody>
</table>

Let 
\[
\begin{bmatrix}
  j_{11} & j_{12} & \ldots & j_{1y} \\
  j_{21} & j_{22} & \ldots & j_{2y} \\
  M & M & M & M \\
  j_{x1} & j_{x2} & \ldots & j_{xy}
\end{bmatrix}
\]

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, it follows that:

\[
\begin{bmatrix}
  N_1 \\
  N_2 \\
  M \\
  N_3
\end{bmatrix}
\]

and

\[
\begin{bmatrix}
  R_1 & R_2 & R_3 & \ldots & R_m
\end{bmatrix}
\]

Then

\[
\begin{bmatrix}
  R_{11} & R_{12} & \ldots & R_{1m} \\
  R_{21} & R_{22} & \ldots & R_{2m} \\
  M & M & M & M \\
  R_{n1} & R_{n2} & \ldots & R_{nm}
\end{bmatrix}
\]

In previous assumptions, we discussed job workloads in Grid environment. We concentrated on classification of job workloads based on each job utilized resource. We classified job workloads into three categories, heavy workload, medium workload, and light workload by using the Fuzzy C-Mean algorithm, after using the SJF (Short Job First) policy for scheduling job mapping to Grid environment. We simulated different performance nodes in Grid environment (Figure 2).
1- Scheduling algorithms

We consider three job scheduling algorithms: FCFS, LJF and SJF. A job $J$ is $\{j_{xy}\}$, where each of $j_{xy}$ is specified as $t_{xy}$, where $t_{xy}$ is the job’s run-time at xth of job number and y is the yth of User ID. We are interested in scheduling such that jobs $j_{xy}$ allocate to n nodes in Grid environment. We choose the initial order of the jobs (run-time) in J. We consider three job scheduling algorithms, namely, FCFS, SJF and LJF, as follows:

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

Then the algorithms of these are:

Input: $J = \begin{bmatrix} j_{1y} & j_{1y} & \ldots & j_{1y} \\ j_{2y} & j_{2y} & \ldots & j_{2y} \\ \vdots & \vdots & \ddots & \vdots \\ j_{xy} & j_{xy} & \ldots & j_{xy} \end{bmatrix}$, whereas $j_1, j_2, j_3, \ldots, j_x$ for FCFS;
$t_1 \leq t_2 \leq t_3 \leq \ldots \leq t_x$ for SJF;
$t_1 \geq t_2 \geq t_3 \geq \ldots \geq t_x$ for LJF.

Output: waiting time and make-span time.

2- Our Scheduling algorithms

Fuzzy C-Mean and SJF() {

Input: $J = \begin{bmatrix} j_{1y} & j_{1y} & \ldots & j_{1y} \\ j_{2y} & j_{2y} & \ldots & j_{2y} \\ \vdots & \vdots & \ddots & \vdots \\ j_{xy} & j_{xy} & \ldots & j_{xy} \end{bmatrix}$, whereas each jobs order is in job number.
Find classification of workload by using Fuzzy C-Mean, such that
Heavy workload: Group of jobs is large job run-time, such as $t_{H1}, t_{H2}, \ldots, t_{Ha}$;
   \[ 1 \leq a \leq xy \]
Medium workload: Group of jobs is medium job run-time, such as $t_{M1}, t_{M2}, \ldots, t_{Mb}$;
   \[ 1 \leq b \leq xy \]
Light workload: Group of jobs is short job run-time, such as $t_{L1}, t_{L2}, \ldots, t_{Lc}$;
   \[ 1 \leq c \leq xy \]

while (jobs $\leq$ $xy$) {
    if (Heavy workload) {
        order jobs run-time: $t_{H1} \leq t_{H2} \leq t_{H3} \leq K \leq t_{Hx}$
    }
    if (Medium workload) {
        order jobs run-time: $t_{M1} \leq t_{M2} \leq t_{M3} \leq K \leq t_{Ms}$
    }
    if (Light workload) {
        order jobs run-time: $t_{L1} \leq t_{L2} \leq t_{L3} \leq K \leq t_{Lx}$
    }
    New order jobs
    \[ t_{H1} \leq t_{H2} \leq t_{H3} \leq K \leq t_{Hx}, \]
    \[ t_{M1} \leq t_{M2} \leq t_{M3} \leq K \leq t_{Ms}, t_{L1} \leq t_{L2} \leq t_{L3} \leq K \leq t_{Lx} \]
    Mapping new order jobs to Grid environment
}
Output: $\text{WaitingTime}_{j} = \text{WaitingTime}_{j2} = \text{WaitingTime}_{j3} = K = \text{WaitingTime}_{jxy}$

Min(Makespan$_n$), whereas $n$ is node number:

\[
\text{Min} \left( \sum_{j=1}^{xy} \frac{\text{WaitingTim}_{e_j}}{xy} \right)
\]

**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 field properties; however, we focused on some properties previously mentioned. In our experiments we assumed that each of the jobs is allowed to run in each node by using space-sharing mechanism. We simulated 500 different performance nodes in Grid environment.

Our experiments showed classification of jobs workload into three groups: heavy workload, medium workload and light workload. We compared the performance of Grid system using our method with that using the traditional methods, such as FCFS, LJF and SJF.
Figure 3 shows that the jobs characterization by using Fuzzy C-Mean algorithm separated the workload data into three groups, (a) heavy workload, (b) medium workload and (c) light workload.

In this experiment, the jobs in workload data are predicted to fall into three classifications, heavy workload where Run-Time is from 62,643 to 8,021 and total jobs are 503, Medium workload where Run-Time is from 7,979 to 2,069 and total jobs are 1,056, and Light workload where Run-Time is from 2,061 to 1 and total jobs are 16,507 (see Table 2).

**Table 2: Job classifications in Grid environment**

<table>
<thead>
<tr>
<th>No.</th>
<th>Classification</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Heavy Workload</td>
<td>503</td>
</tr>
<tr>
<td>2</td>
<td>Medium Workload</td>
<td>1,056</td>
</tr>
<tr>
<td>3</td>
<td>Light Workload</td>
<td>16,507</td>
</tr>
</tbody>
</table>

Figure 4 shows the distribution of waiting time for each of the jobs in Grid environment, and shows the comparisons of waiting time of different algorithms (FCFS, SJF, LJF and Fuzzy + SJF). It can be seen from Figure 3 that our algorithm was much faster than the traditional algorithms.

**Figure 3: Jobs characterization**

**Figure 4: The waiting time of jobs in Grid environment**
Figure 5 shows the make-span time of jobs running in each node of the Grid environment. These trends may indicate the fastest of each node by using our algorithm.

![Figure 5: The make-span time of jobs in Grid environment](image)

**Conclusions**

1. We have proposed a prediction of jobs characterization on Grid environment and used it for our algorithm by using Fuzzy C-Mean.
2. We used jobs workload from the Standard Workload Archive [18] on space-sharing mechanism. We showed that the proposed model captures the jobs characterization of real workload in three different classifications. This model can be used in our algorithm for simulating and evaluating scheduling policies for simulating Grid environment.
3. Many scheduling algorithms for Grid environment depend on static information provided by the Standard Workload Archive [18] and the different performance nodes in the Grid simulated by us. We observe that this information is often unreliable. However, it is a useful way.
4. The experiment results on the waiting time and the make-span have shown that the scheduling using our algorithm can allocate the best results.
5. The results provided here suggest that the researchers look forward to new methods for handling such problems and consider combining them with their method.

**Future Work**

1. Our simulation environment will include critical parameters, such as submit time, Grid network cost, job migrations overhead, and fault tolerance.
2. We plan to investigate the evolution mechanism, such as genetic algorithm, colony algorithm for Grid scheduling, and tabu searching for Grid scheduling.
3. We will include a more complex characterization of the constraints for Grid scheduling and will improve the complexity of problems in Grid environment.
References


