Fuzzy Logic-Based Secure and Fault Tolerant Job Scheduling in Grid*

WANG Cheng (王乘), JIANG Congfeng (蒋从锋), LIU Xiaohu (刘小虎)**

Engineering Computing and Simulation Institute, Huazhong University of Science and Technology, Wuhan 430074, China

Abstract: The uncertainties of grid sites security are main hurdle to make the job scheduling secure, reliable and fault-tolerant. Most existing scheduling algorithms use fixed-number job replications to provide fault tolerant ability and high scheduling success rate, which consume excessive resources or can not provide sufficient fault tolerant functions when grid security conditions change. In this paper a fuzzy-logic-based self-adaptive replication scheduling (FSARS) algorithm is proposed to handle the fuzziness or uncertainties of job replication number which is highly related to trust factors behind grid sites and user jobs. Remote sensing-based soil moisture extraction (RSBSME) workload experiments in real grid environment are performed to evaluate the proposed approach and the results show that high scheduling success rate of up to 95% and less grid resource utilization can be achieved through FSARS. Extensive experiments show that FSARS scales well when user jobs and grid sites increase.

Key words: fault tolerance; grid security; fuzzy logic; job scheduling; self-adaptive replication

Introduction

In a large-scale grid[1], distributed resources belong to different administrative domains. Job executions are usually carried out between many virtual organizations for faster execution or remote interaction. However, grid security problem is a main hurdle to make the job scheduling secure, reliable and fault-tolerant. A lot of algorithms have been developed for scheduling jobs in grids[2-5]. Unfortunately, most of the existing proposed scheduling algorithms had ignored security problem while scheduling jobs onto geographically distributed grid sites with a handful of exceptions. In a real life scenario, the assumption that the grid environments are safe and the resources are 100% reliable is no longer applicable for job scheduling in real grids. Thus, most existing proposed heuristics are not applicable in a risky grid environment.

Job replications are commonly used to provide fault-tolerant scheduling in grids. However, existing job replication algorithms use a fixed number replication[4, 5]. This causes two problems: (1) The algorithms utilize excessive hosts or resources when system security level is high, which makes the makespan and average waiting time of jobs rather longer; and (2) The algorithms cannot provide sufficient fault-tolerance when system security level fluctuates in risky and failure prone grid environments. Thus an adaptive job replication is necessary for real grid job scheduling.

This paper applies the concept of fuzzy set to the process from grid security conditions to replication number of each job. We offered a fuzzy-logic-based self-adaptive job replication scheduling (FSARS) algorithm for use under failure-prone and risky conditions.
We then compared the proposed algorithm with existing heuristics based on fixed number job replications. Experiment results show that security assurance performance could be achieved if we use fuzzy logic to decide the number of job replications when scheduling.

1 Related Works

Trust and security challenges within the grid environment are driven by the need to support scalable, dynamic distributed virtual organization [1]. Web service security (WS-security), open grid services architecture (OGSA) security provide mechanism for integration and interoperability solutions in a grid environment at implementation level [6]. Azzedin et al. [7] suggested integrating the trust concept into grid resource management. They proposed a trust model that incorporates the security implications into grid scheduling algorithms. In this paper, we focus on how to establish a fuzzy-logic-based self-adaptive job replication model and how the job replication number affects the overall performance of the user jobs in grids.

Abawajy [5] presented a distributed fault-tolerant scheduling (DFTS) to provide fault tolerance for job execution in a grid environment. Their algorithm uses fixed number replications of jobs at multiple sites to guarantee successful job executions. This paper uses an adaptive job replication scheme such that the number of replications could change with the security level of the grid environments.

Song et al. [8] developed a security-binding scheme through site reputation assessment and trust integration across grid sites. They applied fuzzy theory to handle the fuzziness or uncertainties behind all trust attributes. The binding is achieved by periodic exchange of site security information and matchmaking to satisfy user job demands. In this paper, we focus on how to establish a fuzzy-logic-based self-adaptive job replication model and how the job replication number affects the overall performance of the user jobs in grids.

Our work is based on the related works on grid security, fuzzy theory, and fault-tolerant job scheduling. We use self-adaptive replication-based algorithm which was mostly ignored in the past [2-5].

2 Job Replication Model Based on Fuzzy Logic

In a real grid environment, a grid site’s security capabilities are varying dynamically. The security level of a grid site dynamically changes in nature, because there is no way to predict when and where a grid will be under attack or crash. Similarly, an application’s security demand also changes with time. In our work, as in Ref. [4], we first assign a security demand (SD) to a user job when user submitted it. The trust model assesses the resource site’s trustworthiness, namely, the trust level (TL). TL quantifies how much a user can trust a site for successfully executing a given job. Only the job can be successfully finished when SD and TL satisfy a security assurance condition (SD $\leq$ TL) when scheduling the jobs.

The typical attributes that the user cares in determining its security demand includes job execution success rate, data integrity, access control, etc [4,6]. These attributes and their values are dynamically changing and depend heavily on the trust model and security policy. This paper assumes that there is a central server that collects job execution success rates, firewall capabilities, grid utilization, and other performance data of the sites periodically. In the initialization of the scheduling, the trust level of the site is computed through the history performance data. Then, the trust level is updated periodically with the site operations. This can be achieved by using some network or grid services like monitoring and discovery system (MDS) [9] when scheduling.

2.1 System model

In this paper, we assume that the application has been divided into sub tasks and each sub task is independent and the tasks have no deadlines and priorities. This assumption is commonly made when studying scheduling problems for grids (e.g., Refs. [2, 4, 5]). However, scheduling jobs with priorities [10] or directed acyclic graph (DAG) topologies [11] can be found in the literature. In this paper, the terms jobs and tasks are used interchangeably.

Let $M = \{m_j | j = 1, 2, 3, ..., m\}$ denote the hosts set, and $T = \{t_i | i = 1, 2, 3, ..., n\}$ denote the tasks set. We define the following parameters:
WANG Cheng (王 成) et al: Fuzzy Logic-Based Secure and Fault Tolerant ...

(1) \( p_j \): The speed of host \( m_j \) (MFlops);

(2) \( e_{ij} \): The expected time to compute when scheduling task \( t_i \) to host \( m_j \). We assume that the estimates of expected task execution times on each machine in the grid sites are known. The assumption is commonly made when studying scheduling problems for grids or heterogeneous computing systems (e.g., Refs. [2,10,11]). Approaches such as code profiling, analytic benchmarking, and statistical prediction for doing this estimation can be found in Refs. [12,13].

(3) \( SD_i \): The security demand of task \( t_i \). \( SD_i \) is specified when the task is submitted, and \( SD_i \) is a real fraction in the range \([0, 1]\) with 0 representing the lowest and 1 the highest security requirement. In some grid environments, setting \( SD_i \) equal to 1 is unnecessary although it seems to be risk-free. If \( SD_i \) is always equal to 1, maybe there are no sites could satisfy the security demand.

(4) \( TL_j \): The trust level of host \( m_j \). \( TL_j \) is in the same range \([0, 1]\) with 0 for the most risky resource site and 1 for a risk-free or fully trusted site. And \( TL_j \) can be computed through the approach in Ref. [8].

(5) \( q_i \): The number of hosts that satisfy \( SD_i \leq TL_j \) for task \( t_i \);

(6) \( \overline{SD} \): The security demand level of the task set. \( \overline{SD} \) is in the range \([0, 1]\) with 0 representing the lowest security requirement and 1 the highest. \( \overline{SD} \) is computed as follows:

\[
\overline{SD} = \frac{1}{n} \sum_{i=1}^{n} (SD_i \times \sum_{j=1}^{n} \frac{e_{ij}}{q_i})
\]

(7) \( \overline{TL} \): The trust level of the grid environment. \( \overline{TL} \) is in the range \([0, 1]\) with 0 for the most risky grid environment and 1 for a risk-free or fully trusted grid environment. \( \overline{TL} \) is computed as follows:

\[
\overline{TL} = \frac{1}{n} \sum_{j=1}^{n} TL_j \times p_j
\]

(8) \( SE_i \) (security error ratio): The security error ratio (also called difference ratio) between \( \overline{TL} \) and \( SD_i \) for task \( t_i \), i.e., \( SE_i = \frac{\overline{TL} - SD_i}{SD_i} \), where \( SE_i \) is in the interval \([-1, +\infty)\).

(9) \( K_i \): The replication number of each job when scheduling. We set \( K_i \) in the interval \([0, 4]\) according to our previous application experiences. We choose 4 as the maximum number of replicas because in our real experiments when the number of replications becomes larger than 4, the system performance degrades heavily.

2.2 Fuzzy inference process

A fuzzy set expresses the degree to which an element belongs to a set. The characteristic function of a fuzzy set is allowed to have values between 0 and 1, which denotes the degree of membership of an element in a given set. If \( X \) is a collection of objects denoted set of ordered pairs: \( A = \{(x, \mu_X(x)) \mid x \in X\} \), where \( \mu_x(x) \) is called the membership function for the fuzzy set \( A \). The membership function \( \mu_j(x) \) for a fuzzy element \( x \) maps element \( x \) into the interval \([0, 1]\), while 1 for full membership and 0 for no membership. Let \( x_1, x_2, \ldots, x_n \) be the elements of the fuzzy set \( A \) and \( \mu_1, \mu_2, \ldots, \mu_n \) denote the membership values of the elements. The fuzzy set \( A \) is commonly denoted as follows:

\[ A = \mu_1/x_1 + \mu_2/x_2 + \cdots + \mu_n/x_n \]

For the transformation from grid security conditions to fuzzy set, we provide the empirical membership functions in Fig. 1. In this paper, fuzzy inference is a process to decide the replication number of each job in four steps as showing in Table 1.

<table>
<thead>
<tr>
<th>Step</th>
<th>Process description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 1</td>
<td>Compute the initial values of ( SD_i, TL_j, \overline{SD}, \overline{TL}, ) and ( SE_i )</td>
</tr>
<tr>
<td>Step 2</td>
<td>Use the membership functions to generate membership degrees for ( SD_i, \overline{SD}, ) and ( SE_i )</td>
</tr>
<tr>
<td>Step 3</td>
<td>Apply the fuzzy rule set to map the input space onto the output space (( K_i ) space) through fuzzy operations</td>
</tr>
<tr>
<td>Step 4</td>
<td>Derive the replication number of each job through a defuzzification process</td>
</tr>
</tbody>
</table>

For example, we consider initial values: \( SD_i = 0.7, \)
Two semi-empirical example fuzzy inference rules are as follows for use in the inference process:

**Rule 1**

IF SD$_i$ is high, SD is medium, and SE$_i$ is medium, THEN $K_i$ is medium.

**Rule 2**

IF SD$_i$ is very low, SD is medium, and SE$_i$ is very low, THEN $K_i$ is very high.

The reason why these particular membership functions and rules are chosen is that in our experiments, the grid system can achieve best performance when we use these membership functions and rules. In a real grid environment, particular membership functions and rules should be chosen according to the system architecture or network topologies, grid site’s dynamic security capabilities, etc.

Through the job replication number inference process using the membership functions in Fig. 1, we can deduce that $K_i$ is 2.3. Then we round the result into an integer 2. Thus, the real job replication number of task $t_i$ is 2.

All selected rules are inferred in parallel. Initially, the membership is determined by assessing all terms in the premise. The fuzzy operator ‘AND’ is applied to determine the support degree of the rules. The final job replication number $K_i$=2 is generated by defuzzification.

There are many other fuzzy inference rules that can be designed using various conditional combinations of the fuzzy variables SD$_i$, SD, and TL.

### 3 Experiment Results and Performance Analysis

#### 3.1 Experiment setup and parameters settings

We test the performance of our fuzzy-logic-based self-adaptive job replication scheduling algorithm on remote sensing-based soil moisture extraction (RSBSME) application workload. The RSBSME workload is a typical data-intensive and high-throughput computing application on grid system. The RSBSME is defined as a set of independent parallel jobs and each job computes a part of a large remote sensing image containing soil moisture information. Here, we use an approximate 165 000 m$^2$ remote sensing image of Hubei Province of China.

Table 2 lists the key experiment parameters. The job security demands and site trust levels dynamically and randomly change during the experiments.

As far as we know, there is no adaptive-replication-based scheduling algorithm proposed or applied in grid environments. Thus, we studied and compared the performance of the simple and frequently used heuristics such as Min-min$^2$, R-Min-min$^4$, and DT-Min-min$^4$, with FSARS scheduling algorithm.
Table 2 Experiment parameters and settings

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of jobs</td>
<td>20,000</td>
</tr>
<tr>
<td>Number of sites</td>
<td>128</td>
</tr>
<tr>
<td>Site processing speed</td>
<td>8 × 8 nodes with 300 MFlops and 8 × 4 nodes with 600 MFlops and 8 × 4 nodes with 200 MFlops</td>
</tr>
<tr>
<td>Initial job security demands</td>
<td>Normally distributed in [0.3, 0.9]</td>
</tr>
<tr>
<td>Initial site trust level</td>
<td>Uniform distributed in [0.2, 1]</td>
</tr>
<tr>
<td>Sites failure rate</td>
<td>Poisson distribution with failure rate 0.10</td>
</tr>
<tr>
<td>Expected execution times</td>
<td>Normally distributed in [120,400] (s)</td>
</tr>
<tr>
<td>Nominal bandwidth between sites</td>
<td>100 Mb/s</td>
</tr>
</tbody>
</table>

To evaluate FSARS, we use the following metrics:
- Makespan: the total running time of all jobs;
- Scheduling success rate: the percentage of jobs successfully completed in the system;
- Grid utilization: defined by the percentage of processing power allocated to user jobs out of total processing power available over all grid sites;
- Average waiting time: the average waiting time spent by a job in the system.

3.2 Experiment results and relative performance

Due to space limitations, only a subset of the results is presented. The experiment results are shown in Fig. 2 and Fig. 3. All the data in the figures are mean values of 15 experiment results, the waiting factor in DT-Min-min heuristic is 0.2, and the number of replication in

R-Min-min is 2 for all experiments. In our experiments, a task will be dropped if it could not finish successfully after five times. Thus, the scheduling success rate cannot reach to 100%.

From Fig. 2 we can see that no single algorithm achieves the best performance for all metrics. However, FSARS exhibits relatively better performance with highest success rate, moderate level of makespan, grid utilization, and average waiting time due to its adaptive job replication scheme.

The comparison of scalability in Fig. 3 shows that FSARS has the highest successful scheduling rate and scales well when the number of tasks increases. It is due to the adaptive replication execution in a real grid environment.

In summary, FSARS changes the number of replications dynamically according to the dynamicity of the grid security conditions. Thus FSARS is applicable to the grid where the security conditions change frequently and can reduce the number of total job replications.
4 Conclusions

A fixed number of job replications in scheduling strategies may utilize excessive hosts or resources. This makes the makespan and average waiting time of tasks rather longer. Thus an adaptive replication strategy is necessary in a real grid with dynamic security conditions. In this paper we applied fuzzy theory to handle the fuzziness and uncertainties when deciding the replication number of user jobs. We have compared makespan, success rate, grid utilization, and average waiting time of FSARS with other strategies. The results show that the task scheduling in a real grid can dramatically be improved by introducing fuzzy-logic based self-adaptive job replication scheduling algorithm. Thus, FSARS is applicable to security-driven and fault-tolerant grid job scheduling.

References


