Reliability of Resource Allocation in Mobile Ad Hoc Grid with Tasks Replication

Sri Chusri Haryanti and Riri Fitri Sari
Department of Electrical Engineering, Faculty of Engineering, University of Indonesia, Depok, 16424 Indonesia
Email: sri.chusri@ui.ac.id, riri@ui.ac.id

Abstract—In the interest of achieving higher reliability, mobile ad hoc grid should apply a fault tolerance in resource allocation scheme. Resource allocation plays a critical role in mobile ad hoc grids due to the dynamic nature of nodes connection. Allocating appropriate resources for the tasks can avoid uncompleted tasks and ensure service reliability. We propose two approaches for avoiding tasks failure due to nodes mobility. They are selecting resource nodes, which have corresponding mobility with resource management system, and implementing task replication strategy. Task replication is expected to increase service reliability. In this paper, we study the reliability of the resource allocation scheme which considers mobile ad hoc grids with centralized architecture with the presence of dependent tasks. Eventually, numerical examples are given to illustrate resulting equations. The result shows that assigning tasks replication provides superior reliability in mobile ad hoc grid, and tasks replication could also minimize the required service time of completing the tasks.

Index Terms—resource allocation, mobile ad hoc grid, reliability, tasks replication, dependent tasks

I. INTRODUCTION

Rapid advancement in mobile computing has emerged to support grid accessibility [1]. Wireless and mobile nodes can access, join, and involve in a so called mobile ad hoc grid. Mobile nodes are able to request and receive a sharing of resources in a Grid computing system dynamically. Mobile ad hoc grid provides self-governed interaction between nodes [2].

In recent years, mobile ad hoc grid is a challenging research topic. The development of mobile ad hoc grid has shifted the objective of using grid. Initially, the grid is used for solving many scientific applications, and in the future it might be used for solving problems of everyday life [3].

The main characteristic of mobile ad hoc grid is its highly dynamic changing network topology due to nodes mobility, nodes switching on and off, and so on [1, 4]. In this environment, failures are likely to occur due to resource allocation fault. It is highly possible that a resource could not return its completed tasks. After a period, the resource node might fail to access the resource management system.

Resource allocation plays a crucial role in determining the performance of the overall grid system. Thus, it is highly critical to provide a resource allocation scheme that is suitable for the environment, and that ensures the completion of the tasks.

We propose two approaches to obtain a reliable resource allocation service. First, for avoiding tasks failure due to nodes mobility, only specific resource nodes will be allocated in completing the tasks. They are resource nodes, which have corresponding mobility with resource management system (RMS). The second is implementing task replication strategy which is expected to increase service reliability.

In an infrastructure grid, task replication is primarily used as a fault tolerance technique in case of a system outage [5]. It is assumed that the success probability of completing the task will increase when it is done by multiple resources. Task replication is a proficient technique in case of a task running on an unreliable execution environment. The goal of the replication is to ensure that at least one replica is always able to complete the computation in case the others fail [6].

In this study, we also investigate the reliability of the proposed resource allocation scheme. The mobile ad hoc grid is considered having centralized architecture. A task graph example is used to examine the reliability with the presence of dependent tasks.

The paper is structured as follows. Section 2 discusses the related works. Section 3 describes problem statement and objective. Section 4 presents the proposed scheme. Reliability analysis is presented in Section 5. In Section 6, illustrative examples are presented and discussed while Section 7 gives the conclusions to the paper.

II. RELATED WORKS

There are proposed schemes of resource allocation, which address mobile ad hoc grids [7-12]. Some of them have addressed the issues such as node mobility, energy management, and task failure. To select the most suitable node for task execution, Gomes et al. in [10] proposed a scheme which utilizes a delayed reply mechanism. It also provides load balancing and scalability. Node mobility has been addressed in [9] by profiling regular movements of a user over the time. To deal with the precedence dependencies, Shilve et al. in [11] have proposed a scheme based on a static allocation of resources in ad hoc grids; however, due to static allocation, this scheme is not adaptive to network changes and application behavior.

Hummel and Jelleschitz [9] and Litke et al. [12] propose resource allocation schemes for mobile ad hoc
grid that considers fault tolerant mechanism. Hummel and Jelleschitz [9] recommend both proactive and reactive fault tolerance mechanisms to address task failure. It is stated that, in a proactive mechanism, only stable resource nodes are selected for tasks execution. Reactive fault tolerance mechanisms aim at recovering the distributed queues and resubmitting active tasks. However, this paper [9] did not describe the method for determining a stable resource node. In [12], Litke et al. suggest to use tasks replication. They also estimate the minimum number of replicas to guarantee a specific fault tolerance level for the grid environment. However, due to the characteristic and limitation of the communication link in mobile ad hoc grid, the number of task replica should be limited.

Our literature review reveals that there is still very few proposed resource allocation algorithm for mobile ad hoc grid that considers fault tolerant mechanisms to reduce the impact of failure. This paper presents two approaches for avoiding a task failure due to node mobility. First, selecting only nodes, which have corresponding mobility as resource nodes. This will ensure only nodes with long-term connectivity will do the tasks. The second is task replication strategy, which is expected to increase service reliability.

Dabrowski noted that there are studies and efforts in developing reliability methods for grid environments [13]. Some investigations on the reliability of infrastructure and management services that are in charge of critical functions for grid systems to operate, such as resource allocation and scheduling in traditional grid have been conducted [14-17]. Nevertheless, there is still minor attention to the reliability of a mobile ad hoc grid [12, 18].

Jang [18] analyzes a mobile resource reliability-based job scheduling model in order to overcome the unreliability of wireless mobile devices and to guarantee stable and reliable job processing. The mobile resource reliability of each resource is calculated and predicted by using connectivity and availability metrics. Litke et al. [12] use tasks replication to guarantee a fault tolerance level for the Grid environment.

The main contribution of this paper is not only proposing a resource allocation scheme that is exhaustive fault tolerant but also performing a reliability study of mobile ad hoc grid.

III. PROBLEM CONTEXT AND OBJECTIVES

In a mobile ad hoc grid, if a node does not have enough resource to execute an application or the performance of an application is not good enough, it will submit the application to resource allocation service through application submission service. Once application is submitted, resource allocation service activates resource discovery service which in turn broadcasts a discovery message to nodes within a wireless range. The nodes willing to share computing resources will reply. They will be noted as the potential resources.

In this paper, the mobile ad hoc grid is considered to have a centralized architecture. It means that there is a single node which is responsible as an RMS. Although the centralized architecture suffers from the scalability and probability of a single point of failure, it gratifies an effective resource allocation decisions due to network-wide view [7].

It is critical to implement effective fault tolerant technique in resource allocation, due to the vulnerability of a mobile ad hoc grid environment. To prevent and reduce the potential errors that may occur due to node mobility, it is necessary to consider the relative mobility in selecting resource nodes. Unpredictable node mobility across the coverage area in mobile ad hoc grid may lead to a task failure. Node mobility across the coverage area will affect not only the task execution on a node but also the dependent tasks execution on the other nodes. Therefore, mobility of a single node can have a large effect on an application performance.

Implementation of a fault tolerant mechanism will also increase service reliability in a mobile ad hoc grid. An effective fault resilient mechanisms are mandatory, due to the unreliable execution environment of such a grid.

In this paper, we present a comprehensive solution for fault resilient mechanisms in a mobile ad hoc grid. We propose two supportive methods. First, a resource selection that considers nodes mobility. The second is task replication. Furthermore, the reliability of mobile ad hoc grid is analyzed. The analysis is based on the hierarchical model such as presented by Dai et al. [15]. In the hierarchical modeling, the reliability analysis considers the probability of different kinds of failures at each layer.

IV. RESOURCE ALLOCATION SCHEME

In grid computing systems, the resource management system (RMS) controls sharing resources among nodes [19]. The resource allocation service in a star topology is depicted in Fig. 1. The RMS is at the center and is directly connected to the resources through a communication channel.

![Figure 1. Mobile ad hoc grid with a star architecture](image)

A number of nodes, that could connect and communicate in a mobile ad hoc network, are considered potentially constructing a mobile ad hoc grid. When a user needs other user resources to execute an application, it will send the application to resource allocation service. Afterward, resource allocation service will enable resource discovery service which subsequently broadcasts a discovery message. Nodes within a range that are willing to share the computing resources will send a reply message. In the reply message, the nodes characteristics
such as processing power, memory, and remaining battery power are enclosed. These attributes will be used to select resource nodes that will execute the tasks. When the tasks execution has finished, the results will be gathered by the requesting node which submit the application.

We use a procedure to obtain a stable cluster of resource nodes among all nodes that are willing to share resources. Stable means having the ability to communicate with the node that requests help to complete the tasks. The task will only be submitted to the nodes which belong to this cluster. This is conducted to ensure that most of the tasks will be properly resolved.

Subsequently, to consolidate the fault tolerant mechanism, tasks replication is also applied. It is assumed that each resource can process only a single task at a time. The same task can be replicated and assigned to several resources for parallel execution. It is assumed that different tasks replication number is assigned, which are computed on the condition of the number of available resource nodes. If the same task is processed by several resources, it is completed when first output is returned to the RMS. All the tasks are completed when their results are returned to the RMS from the resources. Some tasks require outputs from previous tasks for their execution. The order of tasks’ execution is determined by the precedence constraints.

V. RESOURCE ALLOCATION RELIABILITY ANALYSIS

Dai et al. originally introduce a hierarchical model for the grid service reliability analysis and evaluation in [15]. They map the physical and logical architecture of the grid service system. The model considers various types of failures in the grid computing environment, such as blocking failures, time-out failures, matchmaking failures, network failures, program failures, and resource failures. Using the hierarchical model makes the evaluation and calculation tractable by identifying the independence among layers. Later, Zhang presents an analysis of grid service reliability subject to failures similar to Dai’s approach [17].

In association with the resource allocation in a mobile ad hoc grid, we present a reliability analysis using the hierarchical model approach. In this respect, the reliability analysis is comprised of reliability of matchmaking process, reliability of network and reliability of task execution. The reliability analysis will be described in the following sub section.

A. Reliability of Matchmaking Process

The management layer can be considered as the global resource allocation layer [15]. It has the function of detecting new resources, monitoring the resources, removing failed resources, and matching the resource requests of a service to the registered resources. If the resource requests are fit to the available resources, the network layer will handle the next process.

Matchmaking failure means that the requests fail to match with the correct resources [15]. During the matchmaking process, certain requests may be mismatched to the wrong resource, which causes matchmaking failure. The assumptions for matchmaking failures are:

- The failure rate of matchmaking process is \( \lambda(k) \), which is a function to \( k \) number of faults.
- When a matchmaking failure happens, the program will notify the RMS and then they will try to remove the faults.
- Removing the fault may result in other new faults. The occurrence of a new generating fault follows a Poisson distribution. The rate of new faults is constant, \( v \).

\[
R_{match} = \exp (-\lambda \cdot H \cdot T) = \exp \left( -\frac{n_r}{n_t} H \right) = \exp(\lambda_m H) \quad (2)
\]

Figure 2. Markov chain for matchmaking reliability

To model this process, according to those assumptions, it fits a continuous time Markov chain (CTMC) [15]. The model is depicted in Fig.2. It is an infinite number of states of a birth-death Markov process. State \( k \) represents \( k \) matchmaking faults.

Matchmaking time for a request can be regarded smaller enough than the whole life of the resource allocation service. Accordingly, the failure rate can be considered as constant, as in [15], which can be estimated as

\[
\tilde{\lambda} = \frac{n_r}{n_t} \tilde{T}
\]

Where \( n_r \) is the number of mismatched requests, \( n_t \) is the total number of requests, and \( \tilde{T} \) is the expected time for completing each request. The probability that all requests are correctly fitted to their required resources if the given \( H \) requests can be yielded by

\[
R_{match} = \exp (-\tilde{\lambda} \cdot H \cdot \tilde{T}) = \exp \left( -\frac{n_r}{n_t} \tilde{T} \right) = \exp(\lambda_{m} \tilde{T}) \quad (2)
\]

B. Reliability of Network

If the matchmaking process has worked well, the programs can connect to the resources through the network. However, during the period of connection, network failures may occur. Since the communication of the nodes occur through a mobile ad hoc network, the network reliability follows the reliability of mobile ad hoc network accordingly. Consequently, the discussions of a link in a mobile ad hoc network need to be associated with node’s mobility.

In [20], Hwang and Kim present a Markov model of link connectivity for mobile ad hoc networks. Link connectivity is analyzed for a non-steady state mobility model. Thus, apparent condition of mobile ad hoc network can be attained.

Figure 3. Connectivity states of nodes and their transition probability
Fig. 3 shows the connectivity and transition probability between two nodes in an ad-hoc network. The state D denotes that two nodes are disconnected, and the state C indicates that they are connected with a bidirectional wireless channel. $p$ refers to the probability of the transition from D to C, and $q$ is the probability of the transition from C to D.

If $a$ and $\beta$ are transition rates of $D \rightarrow C$ and $C \rightarrow D$, respectively, a continuous-time Markov chain will yield to (3) as the probability of the connection of the two nodes will break in time $t$ [20].

$$\lim_{t \to \infty} P_b(t) = \frac{\beta}{a + \beta}$$

(3)

Thus, the reliability of the network

$$R_n(t) = p_i(t)$$

(4)

Where the values of $a$ and $\beta$ depend on the mobility model of the nodes [20].

C. Reliability of Task Execution

In the presence of dependent tasks, the reliability of tasks execution depends not only on the reliability of each task, but also on the reliability of the execution dependencies on them [16]. In the tasks execution phase, tasks will be processed in different resources. Each resource can only process one task when it is available. However, in case of implementing tasks replication, the same task may be assigned to several resources. If the same task is done by several resources, it is considered completed when first output has arrived back to the RMS.

To facilitate the discussion, it is assumed that RMS is fully reliable. The time of task processing by the RMS is also assumed to be insignificant as it is considered very small compared to the tasks processing and network connection time. As it is assumed that the grid service has the star topology, the discussion of task execution reliability will follow [16].

In this work, a job is composed of $m$ tasks with $c_i$ complexity and the amount of the total data that will be interchanged between RMS and a resource is $a_j$ ($1 \leq j \leq m$). The matrix $H$, $m \times m$, describes the precedence dependency of task execution. If the task $i$ needs an output data from the task $k$, then $h_{ki} = 1$, otherwise, $h_{ki} = 0$. For $k < i$, $h_{ki} = 1$ means that the task $i$ can be executed only if the task $k$ has been completed.

The task execution time is defined as the time from the beginning of the input data transmission from the RMS to a resource, to the end of output data transmission from the resource to the RMS [16].

The random time of task $i$ executed by resource $j$ ($t_{ij}$) has two possible values. If the task has been completed successfully, then

$$t_{ij} = \frac{c_i}{s_j} + \frac{a_i}{s_j}$$

(5)

and otherwise $t_{ij} = \infty$.

The probability of a successful task execution when the failure rate of the resource ($\lambda_j$) is constant is:

$$p_j(T_{ij}) = e^{-\lambda_j t_{ij}}$$

(6)

with $Pr(t_{ij} = \infty) = p_j(T_{ij})$ and $Pr(t_{ij} = \infty) = 1 - p_j(T_{ij})$.

It is assumed that, for every task $i$, RMS allocates a set of resources $\omega_i$ ($\omega_i \cap \omega_j = \emptyset$). RMS will send the data of the task $i$ to $\omega_i$ after the previous task $k \in \omega_i$ is successfully completed. Thus, it can be determined that the random time of the start of task $i$ execution, $T_i$ is

$$T_i = \max_{k \in \omega_i} (T_k)$$

(7)

where $\hat{T}_k$ is the random completion time for task $k$. $\omega_i = \emptyset$ means that the execution of the task $i$ does not depend on the data produced from other task and it can be started without delay: $T_i = 0$. If $\omega_i \neq \emptyset$, so the value becomes varied $T_i (1 \leq l \leq L_i)$.

With the start of execution $T_i$ and execution time $t_{ij}$ of task $i$ by resource $j$, the random execution of task $i$ by resource $j$ is

$$\hat{t}_{ij} = T_{ij} + t_{ij}$$

(8)

To obtain the distribution of $\hat{t}_{ij}$, the probability of any realization of $\hat{t}_{ij} = \hat{T}_{ij} + \hat{t}_{ij}$ has to be considered. According to [16], it is equal to the product of three occurrences:

1. Task $i$ execution starts at $\hat{T}_{ij}$: $q_{ij} = Pr(T_i = \hat{T}_{ij})$
2. Resource $j$ has not failed before the task $i$ execution starts: $p_j(\hat{T}_{ij})$
3. Resource $j$ has not failed during task $i$ execution: $p_j(\hat{t}_{ij})$

Consequently, the distribution of $\hat{t}_{ij}$, with the task $i$ execution at $\hat{T}_{ij}$ ($T_i = \hat{T}_{ij}$) is:

$$Pr(\hat{t}_{ij} = \hat{T}_{ij} + \hat{t}_{ij}) = p_j(\hat{T}_{ij} + \hat{t}_{ij}) = e^{-\lambda_j (T_{ij} + t_{ij})}$$

$$Pr(\hat{t}_{ij} = \infty) = 1 - p_j(\hat{T}_{ij} + \hat{t}_{ij}) = 1 - e^{-\lambda_j (T_{ij} + t_{ij})}$$

(9)

The random execution time of task $i$, $\hat{T}_{i}$ is equal to the fastest time that one of resource from the $\omega_i$ set can finish the task,

$$\hat{T}_{i} = \min_{j \in \omega_i} (\hat{t}_{ij})$$

(10)

In case that the last task is the task $m$, the start time of task $m$ is $T_m$. Knowing the distribution of $T_m$ in the form of $q_{ml} = Pr(T_m = \hat{T}_{ml})$ for $1 \leq l \leq L_m$, the reliability of the service performance can be evaluated.

In estimating the service reliability and the performance, one can use different measures depending on the application [16]. In the application where the execution time of each task is critical, the task execution reliability $R(T^*)$ is defined as the probability of the correct output in time less than $T^*$.

$$R(T^*) = \sum_{l=1}^{L_m} q_{ml} \cdot 1(\hat{T}_{ml} < T^*)$$

(11)

In applications where the number of executed tasks over a fixed time is important, the service reliability can be defined as the probability that the correct output is produced without respect to the service time.

$$W = \sum_{l=1}^{L_m} \hat{T}_{ml} q_{ml} / R(T, \infty)$$

(12)

The universal generating function ($u$-function) is used to evaluate the distribution of service time. This method
is effective for evaluating the reliability of the multi state system [21].

\( u(z) = \sum_{k=1}^{N} a_k z^k \)  
(13)

\( Y \) variable has \( K \) possible values and \( a_k \) is the probability that the \( Y \) value is equal \( y_k \).

To obtain \( u \) function that represents a \( pmf \) of a function with two independent random variables \( \varphi(Y_i, Y_j) \):  

\[
U(z) = \sum_{k=1}^{N} a_k z^k = \sum_{k=1}^{N} a_k z^{y_k} \quad \sum_{k=1}^{N} a_k z^{y_k}
\]

\( \text{where} \quad \varphi(Y_i, Y_j) \quad \text{is the probability of} \quad \varphi(Y_i, Y_j) \)

The function \( U(z) \) describes all possible combination from all variables by means of connecting the probability of each combination to the value of the function \( \varphi(Y_i, Y_j) \). In the case of the grid system, \( u \) function \( u(z) \) defines the \( pmf \) of execution time of service \( i \) that is executed in the resource \( j \).

\[
u(u)(z) = p(z) z \cdot (1 - p(z)) z\infty
\]

(15)

Where \( \hat{T}_i \) and \( \hat{p}(z) \) are determined by (5) and (6).

\( U(z) \) function for \( \text{random start time} \ T_i \) for \( \text{task} \ i \) can be described as the \( \text{u function} \ U(z) \) :  

\[
U(z) = \sum_{i=1}^{N} q_{il} z^{T_{il}}
\]

(16)

\( q_{il} = Pr(T_i = T_{il}) \).

For each realization of \( T_i \), the conditional distribution of the completion time \( \hat{T}_i \) of the task \( i \) which is executed by the resource \( j \) if \( T_{il} = T_i \) can be represented by \( \varphi(Y_i, Y_j) \) :

\[
\hat{u}(z, \hat{T}_i) = p(z) z \cdot (1 - p(z)) z\infty
\]

(17)

The total completion time from the task \( i \) which is executed by the resource \( j \) is equal to the minimum time of completion time for those resource as in (7). To get \( u \) function that represent the \( pmf \) of that time, if \( T_i = T_{il} \), it has to use a composition operator with \( \varphi(Y_i, Y_j) = \min(Y_i, Y_j) \)

\[
\hat{u}(z, \hat{T}_i) = \min\hat{u}(z, \hat{T}_i) \quad \varphi \hat{u}(z, \hat{T}_i)
\]

(18)

\( u \) function of \( \hat{u}(z, \hat{T}_i) \) from (15) represents the conditional \( pmf \) from the completion time of \( T_i \) for the task \( i \) that is executed by all resources in \( \omega_i = j_1, ..., j_l \), where:

\[
\hat{u}(z, \hat{T}_i) = \min\hat{u}(z, \hat{T}_i) \quad \varphi \hat{u}(z, \hat{T}_i)
\]

(19)

\( \hat{u}(z, \hat{T}_i) \) can be obtain recursively:

\[
\hat{u}(z, \hat{T}_i) = \hat{u}(z, \hat{T}_i) \quad \varphi \hat{u}(z, \hat{T}_i)
\]

(20)

With the realization of the probability that exclusively mutual from the start time \( T_i = T_{il} \) and \( u \) function of \( \hat{u}(z, \hat{T}_i) \) which represent the conditional \( pmf \) of the completion time \( T_i \), can be represented as:

\[
\hat{U}(z) = \sum_{i=1}^{N} q_{il} z^{T_{il}}
\]

(21)

\( U(z) = \hat{U}(z) \quad u \) for \( e = j_1, ..., j_l \)

(22)

\( \hat{U}(z) \) can be obtained recursively as:

\[
\hat{U}(z) = \hat{U}(z) \quad u \) for \( e = j_1, ..., j_l \)
\]

(23)

It can be seen that if \( o_i = \emptyset \), then \( U(z) = z^0 \).

The final result of the \( u \) function \( U(z) \) represents the \( pmf \) of random completion time task \( T_m \) in the form of:

\[
U(z) = \sum_{i=1}^{N} q_{ml} z^{T_{ml}}
\]

(24)

C. Grid Service Reliability

Following the analysis above, with the assumption that the condition of the three levels of the hierarchical modeling is independent, the resource allocation service reliability can be calculated by:

\[
R_{\text{service}} = R_{\text{match}} \cdot R_{a} \cdot R_{i}
\]

(25)

\( R_{\text{match}}, R_{a}, \) and \( R_{i} \) are determined by (2), (4) and (11) respectively.

VI. ILLUSTRATIVE EXAMPLES

This example illustrates the analysis of the reliability, the performance of a resource allocation grid service that has been discussed above.

A. Computing the Reliability of Matching Process

Suppose that the RMS records the number of mismatched requests \( n_l = 10 \) out of the total number of requests that have been completed (\( n_r = 1000 \). This means \( a_{m} = 10/1000 = 0.001 \), we can get the probability for the matchmaking failure that is not occurring for the fifteen request by (2):

\[
R_{\text{match}} = \exp\left(-\frac{n_{l}}{n_{r}}\right) = \exp(-0.001 \times 15) = 0.985
\]
B. Computing the Reliability of Network

It has been explained in the previous sections that the tasks will only be done by a cluster of resource nodes which have corresponding mobility with resource management system. Accordingly, the mobility model of the resource nodes can be assumed following the Random Point Group Mobility (RPGM). In [20], we obtain the parameter $\alpha = 0.00074$ and $\beta = 0.00164$ which is satisfied the RPGM. Consequently, the reliability of the communication channel is:

$$R_{\text{match}} = \frac{\beta}{\alpha + \beta} = \frac{0.00164}{0.00074 + 0.00164} = 0.6874$$

C. Computing the Reliability of Task Execution

It is assumed that there are only 15 resources which satisfy the condition and are included in the cluster of resource nodes. The entire service job consists of eight tasks with precedence constraints presented in Fig. 4. Task 9 corresponds to the final task processing by the RMS. The first scenario (Case A) of the task assignment and the parameter used can be seen in Table I. The values of the time of task $i$ executed in resource $j$ are in the range of $10 - 20$ seconds referred to [18]. The assumption of the failure rate of resource $j$ is referred to [16].

![Task execution precedence constraints](image)

**Figure 4.** Task execution precedence constraints for the analytical example

**TABLE I. PARAMETERS USED FOR THE ANALYTICAL EXAMPLE**

<table>
<thead>
<tr>
<th>No. of resource $j$</th>
<th>Failure rate of resource $j$</th>
<th>Time of task $i$ executed in resource $j$</th>
<th>No of task that will be executed $(i)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.0005</td>
<td>20</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>0.0007</td>
<td>17</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>0.0001</td>
<td>16</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>0.0010</td>
<td>12</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>0.0002</td>
<td>13</td>
<td>3</td>
</tr>
<tr>
<td>6</td>
<td>0.0006</td>
<td>10</td>
<td>3</td>
</tr>
<tr>
<td>7</td>
<td>0.0004</td>
<td>20</td>
<td>4</td>
</tr>
<tr>
<td>8</td>
<td>0.0003</td>
<td>14</td>
<td>5</td>
</tr>
<tr>
<td>9</td>
<td>0.0008</td>
<td>10</td>
<td>4</td>
</tr>
<tr>
<td>10</td>
<td>0.0010</td>
<td>15</td>
<td>5</td>
</tr>
<tr>
<td>11</td>
<td>0.0007</td>
<td>18</td>
<td>6</td>
</tr>
<tr>
<td>12</td>
<td>0.0002</td>
<td>10</td>
<td>6</td>
</tr>
<tr>
<td>13</td>
<td>0.0000</td>
<td>12</td>
<td>7</td>
</tr>
<tr>
<td>14</td>
<td>0.0004</td>
<td>14</td>
<td>8</td>
</tr>
<tr>
<td>15</td>
<td>0.0003</td>
<td>17</td>
<td>8</td>
</tr>
</tbody>
</table>

Only tasks number 2, 4, and 7 are executed by a single resource; 4, 7, and 12, respectively. The rest of the tasks are executed by 2 or 3 resources in parallel. It is assumed that the RMS is in node 13 and it is fully reliable ($\mu = 0$). The failure rate of the other resources is assumed 0.0001.

Referring to Fig 4., the tasks 1, 2, 3 and 4 get the input data directly from the RMS. Task 5 needs input data from task 1, task 6 needs input data from tasks 2 and 3. Task 7 needs input data from tasks 5 and 6 and task 8 needs input data from tasks 4 and 6. The task is completed when the RMS gets the outputs of both tasks 8 and 7. It can be stated that $\eta_j = \eta_1 = \eta_2 = \eta_3 = \eta_4 = 0$. In consequence, the realization of the start time of tasks 1, 2, 3 and 4 is 0, and therefore $\mathcal{U}_1(z) = \mathcal{U}_2(z) = \mathcal{U}_3(z) = \mathcal{U}_4(z) = z^0$. We can obtain the $u$-functions representing the pmf of completion times $\mathcal{F}_{11}, \mathcal{F}_{12}, \mathcal{F}_{13}, \mathcal{F}_{24}, \mathcal{F}_{35}, \mathcal{F}_{36}$, and $\mathcal{F}_{47}$.

Task execution time distributions for the individual resources, define the $u$-functions $\mathcal{U}_i(z)$, according to Table I, and (15):

$$\mathcal{U}_{11}(z,0) = e^{-0.0005x20}z^{20} + (1-e^{-0.0005x20})z^{\infty} = 0.99z^{20} + 0.01z^{\infty}$$

In a similar method, we obtain

$$\mathcal{U}_{12}(z,0) = 0.9882z^{17} + 0.0118z^{\infty}$$
$$\mathcal{U}_{13}(z,0) = 0.9984z^{16} + 0.0016z^{\infty}$$
$$\mathcal{U}_{135}(z,0) = 0.9974z^{13} + 0.0026z^{\infty}$$
$$\mathcal{U}_{136}(z,0) = 0.994z^{10} + 0.006z^{\infty}$$

For task 1 which is executed by resources 1, 2 and 3, the $u$-function representing the pmf of the completion time is:

$$\mathcal{U}_1(z) = \mathcal{U}_{11}(z,0) \otimes \mathcal{U}_{12}(z,0) \otimes \mathcal{U}_{13}(z,0) = (0.99z^{20} + 0.01z^{\infty}) \otimes (0.9882z^{17} + 0.0118z^{\infty}) \otimes (0.9984z^{16} + 0.0016z^{\infty})$$

$$= 0.9984z^{16} + 0.0016z^{17}$$

For task 3 executed by both resources 5 and 6, the $u$-function representing the pmf of the completion time is:

$$\mathcal{U}_3(z) = \mathcal{U}_{35}(z,0) \otimes \mathcal{U}_{36}(z,0) = (0.9974z^{13} + 0.0026z^{\infty}) \otimes (0.994z^{10} + 0.006z^{\infty})$$

$$= 0.994z^{10} + 0.006z^{13}$$

Task 2 and task 4, each is executed by one resource. The pmf of the completion time of task 2 and task 4:

$$\mathcal{U}_2(z) = \mathcal{U}_{24}(z,0) = 0.9881z^{12} + 0.0119z^{\infty}$$
$$\mathcal{U}_4(z) = \mathcal{U}_{47}(z,0) = 0.992z^{20} + 0.008z^{\infty}$$

The execution of task 5 begins after the completion of task 1. Therefore,

$$\mathcal{U}_5(z) = \mathcal{U}_1(z) = 0.9984z^{16} + 0.0016z^{17}$$

The $u$-functions representing the conditional pmf of the completion times for the task 5 executed by resource number 8, 9 and 10 are obtained as follows.

$$\mathcal{U}_{58}(z,16) = e^{-0.0003x(16+14)}z^{(16+14)} + (1-e^{-0.0003x(16+14)})z^{\infty}$$

$$= 0.991z^{30} + 0.009z^{\infty}$$
The conditional pmf of task 5 completion time for each realization time are
\[
\bar{u}_{5}(z,16) = \bar{u}_{58}(z,16) \otimes \bar{u}_{59}(z,16) \otimes \bar{u}_{510}(z,16) \\
= 0.9794 z^{26} + 0.0204 z^{31} + 0.0002 z^{31}
\]
\[
\bar{u}_{5}(z,17) = \bar{u}_{58}(z,17) \otimes \bar{u}_{59}(z,17) \otimes \bar{u}_{510}(z,17) \\
= 0.9786 z^{27} + 0.0212 z^{31} + 0.0002 z^{32}
\]

According to (21), we can obtain the u-function representing the unconditional pmf of the task 5 completion time is
\[
\bar{U}_{5}(z) = 0.9984 \bar{u}_{5}(z,16) + 0.0016 \bar{u}_{5}(z,17) \\
= 0.9984(0.9794 z^{26} + 0.0204 z^{31} + 0.0002 z^{31}) + \\
0.0016(0.9786 z^{27} + 0.0212 z^{31} + 0.0002 z^{32}) \\
= 0.9978 z^{26} + 0.0016 z^{27} + 0.0204 z^{31} + 0.0002 z^{31}
\]

Next, the task 6 needs the input data from task 2 and 3. Therefore, the u-function representing the pmf of the task 6 is obtained as
\[
\bar{U}_{6}(z) = \bar{U}_{2}(z) \otimes \bar{U}_{3}(z) \\
= (0.9881 z^{12} + 0.0119 z^{e}) \\
\otimes (0.994 z^{10} + 0.006 z^{13}) \\
= 0.9822 z^{13} + 0.0059 z^{13} + 0.0119 z^{e}
\]

The u-functions representing the conditional pmf of the completion times for the task 6 executed by individual resources are obtained as follows.
\[
\bar{u}_{611}(z,12) = e^{-0.0007(z+12)}(z+12)^{12} - 1 + e^{-0.0007(z+12)} z^{e} \\
= 0.9792 z^{20} + 0.0208 z^{e} \\
\bar{u}_{612}(z,12) = 0.9785 z^{31} + 0.0215 z^{e} \\
\bar{u}_{611}(z,13) = 0.9785 z^{31} + 0.0215 z^{e} \\
\bar{u}_{612}(z,13) = 0.9954 z^{23} + 0.0046 z^{e} \\
\bar{u}_{612}(z,∞) = z^{e}
\]

The u-functions representing the conditional pmf of the completion times for the task 6 for each realization time are
\[
\bar{u}_{6}(z,12) = \bar{u}_{611}(z,12) \otimes \bar{u}_{612}(z,12) \\
= 0.9956 z^{23} + 0.0043 z^{30} + 0.0001 z^{30} \\
\bar{u}_{6}(z,13) = \bar{u}_{611}(z,13) \otimes \bar{u}_{612}(z,13) \\
= 0.9954 z^{23} + 0.0043 z^{30} + 0.0001 z^{30} \\
\bar{u}_{6}(z,∞) = z^{e}
\]

According to (21), the u-function represented by the unconditional pmf of the completion time of task 6 is
\[
\bar{U}_{6}(z) = 0.98822 \bar{u}_{6}(z,12) + 0.0059 \bar{u}_{6}(z,13) + 0.0119 \bar{u}_{6}(z,∞) \\
\bar{U}_{6}(z) = 0.9797 z^{23} + 0.0059 z^{23} + 0.0042 z^{30} + 0.0112 z^{e}
\]

The entire task is completed when tasks 7 and 8 return their outputs to the RMS, which corresponds to the beginning of task 9. After performing the calculation for task 7 and task 8 using the same method, we can obtain the u-function representing the pmf of the entire service time which is
\[
U_{S}(z) = \bar{U}_{7}(z) \otimes \bar{U}_{8}(z) \\
= 0.9049 z^{26} + 0.0015 z^{27} + 0.0227 z^{24} + 0.0150 z^{31} + 0.0233 z^{34} + 0.0002 z^{45} + 0.0004 z^{47} + 0.0316 z^{e}
\]

From the obtained pmf, we can calculate the service reliability using (11),
\[
R(T^*) = 0.9049 	ext{ for } 40 < T^* \leq 41 \\
R(T^*) = 0.9064 	ext{ for } 41 < T^* \leq 42 \\
R(T^*) = 0.9291 	ext{ for } 42 < T^* \leq 43 \\
R(T^*) = 0.9441 	ext{ for } 43 < T^* \leq 44 \\
R(T^*) = 0.9674 	ext{ for } 44 < T^* \leq 45 \\
R(T^*) = 0.9676 	ext{ for } 45 < T^* \leq 47 \\
R(∞) = 0.9996
\]

According to (25), with \(R_{\text{match}} = 0.985\) and \(R_s = 0.6874\) as calculated in the subsection A and B in the section IV, we can obtain the reliability of the service of the Case A as
\[
R(T^*) = 0.6127 	ext{ for } 40 < T^* \leq 41 \\
R(T^*) = 0.6137 	ext{ for } 41 < T^* \leq 42 \\
R(T^*) = 0.6290 	ext{ for } 42 < T^* \leq 43 \\
R(T^*) = 0.6550 	ext{ for } 43 < T^* \leq 44 \\
R(T^*) = 0.6552 	ext{ for } 44 < T^* \leq 45 \\
R(T^*) = 0.6554 	ext{ for } 45 < T^* \leq 47 \\
R_{\text{service}}(∞) = 0.6768
\]

We also evaluate the effect of removing some of resources from the grid. The summary of different cases is presented in Table II. In Case B, resource number 3 and 10 are removed. In Case C, more resources are removed, i.e resource number 3, 10, 12 and 15. In Case D, there are also four resources removed from the list as in Case C, but not all of the removed resources are the same. Case E is the resource allocation service without task replication. In Case F, we apply the same number of replica for each task, i.e. two, where the set of resources for each task is shown in Table II.

<table>
<thead>
<tr>
<th>Case</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>The same as in Table 1</td>
</tr>
<tr>
<td>B</td>
<td>Removing resource number 3 &amp; 10</td>
</tr>
<tr>
<td>C</td>
<td>Removing resource number 3, 10, 12 &amp; 15</td>
</tr>
<tr>
<td>D</td>
<td>Removing resource number 6, 10, 12 &amp; 15</td>
</tr>
<tr>
<td>E</td>
<td>No task replication, using resource number 1, 4, 5, 7, 10, 11, 13 &amp; 15</td>
</tr>
<tr>
<td>F</td>
<td>The number of replica for each task is two, with (ω_r = 1(2), ω_r = 3(4), ω_r = 5(6), ω_r = 7(8), ω_r = 9(10), ω_r = 11(12), ω_r = 13, ω_r = 14(15).)</td>
</tr>
</tbody>
</table>
The comparison of reliability for each case is presented in Table III. It is shown that the minimum of required service time for Case F is the minimum among all the cases. The less of resources will extend the minimum of required service time ($T_{min}$). It denotes that the implementation of task replication in resource allocation of mobile ad hoc grid could decrease the required service time and increase the reliability.

The comparison of reliability of Case A, B and C is presented in Fig. 5. It is shown that if more resources participate in completing the tasks, the reliability will increase. Although $T_{min}$ of Case A and Case B is the same, Case A is more reliable than Case B. It can be inferred that all the value of $R(T_i)$ in Case A is higher than in Case B. Case C uses less resources than Case B. Although the value of $R(T_{min})$ of Case C is higher than Case B, but the reliability for the same $T_i$ values in Case B is always higher than Case C.

Fig. 6 presents the reliability of full replication condition (Case A) compared to no replication condition (Case E). It is shown that the reliability of Case A is better than Case E. The required service time of Case A is also smaller than Case E. It can be concluded that the task replication will increase the reliability and decrease the required service time of resource allocation in mobile ad hoc grid.

The same number of resources that finish the tasks but with different condition of replication number in the graph will result in different reliability as shown in Fig. 7. Case C and D use the same number of resources but with different replication tasks. It is shown that Case C has less $T_{min}$ and superior reliability compared to Case D. This indicates that for a job that include dependent task, the scenario of replication number of each task will greatly affect the reliability of the whole job.

Fig. 8 presents the comparison of service reliability of Case A, E, and F. It is interesting to note that the reliability of Case F (the same number of replication for all tasks) is better than Case A (different number of replication for each task). This result provides further support for the hypothesis that performing the same replication number for all the tasks will give higher reliability and shorter required service time of resource allocation in mobile ad hoc grid. However, further studies should be done to investigate the optimum number of task replication that satisfy mobile ad hoc grid environment.
VII. CONCLUSION

A fault-tolerant resource allocation is highly needed in the severe environment of mobile ad hoc grid. In this work, we propose two approaches in resource allocation of mobile ad hoc grid for avoiding tasks failure due to nodes mobility. The first is selecting resource nodes, which have corresponding mobility with resource management system, is expected to ensure that the tasks will be properly completed. The second is implementing task replication, which is expected to increase the service reliability.

In this paper, we also analyze the reliability of the resource allocation with tasks replication in mobile ad hoc grid using the hierarchical model. By means of the model, the reliability of mobile ad hoc network could be taken into account. The result of an analytical example shows that assigning tasks replication provides superior reliability in mobile ad hoc grid. Tasks replication could also minimize the required service time for completing all the tasks. However, it turns out that a certain combination of task replica number will give different reliability. In the future, we will study the optimization of task replica for a certain failure probability in mobile ad hoc grid.

REFERENCES


Sri Chusri Haryanti received her B.Sc and Master degrees in Electrical Engineering from Universitas Indonesia. She is a lecturer at Universitas YARSI, Jakarta, and currently she is a PhD candidate at Electrical Engineering Department of Universitas Indonesia. Her research interest includes mobile computing and mobile ad hoc grid.

Riri Fitri Sari, PhD. is a professor at Electrical Engineering Department of Universitas Indonesia. She received her BSc degree in Electrical Engineering from Universitas Indonesia. She received her MSc in Computer Science and Parallel Processing from The University of Sheffield, UK, and her PhD in Computer Science from The University of Leeds, UK. Riri Fitri Sari is a senior member of the Institute of Electrical and Electronic Engineers (IEEE).