Evaluation of Grid Computing Environment Using TOPSIS

Mahmoud Mohammaddoust[1], Ali Harounabadi[2], and Mohammadali Neizari[1]
[1]Department of Computer, Institute for Higher Education Academic Center for Education, Culture and Research Khouzestan, Iran
[2]Department of Computer, Islamic Azad University, Iran

Abstract: *Grid evaluation approaches usually focus on some special aspects of grid environment and there have been few researches on a technique which is able to comprehensively evaluate a grid system in terms of its performance. In this paper an algorithm is proposed in order to evaluate the performance of grid environment based on4 metrics of reliability, task execution time, resource utilization rate and load balance level. In the proposed algorithm, a new method for evaluating the resource utilization rate has been presented. Also, in the paper an application of Technique for Order-Preference by Similarity to Ideal Solution (TOPSIS) is presented in order to choose the most efficient system based on these 4 metrics. Algorithm and TOPSIS performances are demonstrated through analytical and numerical examples. Then, using simulation, it has been demonstrated that the proposed algorithm estimates the amount of utilization rate with high accuracy. Using the suggested approach, one can choose the most efficient algorithm so that a compromise is established between managers' and users' requests.*

Keywords: *Grid computing evaluation, reliability, task execution time, resource utilization rate, load balance level, TOPSIS.*

Received October 4, 2015; accepted March 27, 2016

1. Introduction

Grid Computing is a kind of distributed computing in which many systems are dynamically connected to each other, sharing their resources and forming a virtual and powerful set of resources [11, 20, 22]. The resource sharing is controlled by Resource Management System (RMS) based on sharing rules defined by resource providers and consumers [18].

Users or applications may request for using grid resources or services. When the RMS receives a service request, it divides the submitted service task into a set of subtasks to be executed in parallel. Then it sends the subtasks to available resources in grid environment to be executed. After completing the execution of subtasks, the resources send back the results to RMS. Then RMS integrates these results, creates the final result of task execution and sends it to user [1, 6, 17, 18]. The process of distributing subtasks between the available resources in specific times is referred to as "Task Scheduling" implemented by scheduling algorithms [1].

Grid middleware and architecture have been developed extensively during recent years, but due to the weakness of performance evaluation approaches, grid users have problem in recognition and selection an efficient grid system to fulfill their requests [21]. Since the managers and users of the system have different and sometimes contradictory requests from it, there are different performance metrics to evaluate grid performance. The metrics are of different importance from user and manager perspective. Many studies have been done on grid computing and grid environment, focusing on different subjects of grid environment including performance and scheduling algorithms evaluation. Dai *et al.* [7] provided some algorithms to evaluate the reliability of grid program and system. Levitin and Dai [18] and Dai and Levitin [5] proposed a model to evaluate task execution time and reliability of grid service with star and tree topology. Peng *et al.* [21] presented some preliminary analysis on grid performance metrics like response time and system utilization. Azgomi and Entezari-Maleki [1] provided a high-level Petri net model in order to model the grid environment and compute the reliability through colored Petri net model and Colored Petri Net (CPN) Tools. Performance metrics mentioned on these researches are not chosen as to be able to meet the expectations of both users and system managers. In addition, the evaluation methods finally leave the users and system managers just with some data tables and graphs, not answering the question of which scheduling algorithm or grid environment is better to be used. This paper tried to reply the question and evaluate grid environment from both user and system manager's perspective.

The rest of this paper is organized as follows. Section 2 represents some background information. Section 3 describes the proposed method. Section 4 provides illustrative examples. Section 5 verifies the accuracy of the proposed method for evaluating there source utilization rate metric using simulation. Section

6 summarizes the paper. All the variables used in equations, have been already described in Table 1.

Table 1. Description of the variables.

Variable	Definition	Variable	Definition
D	Decision matrix	$\mathbf{d_{ij}}$	Decision matrix element
R, R'	a set of resources which are used to execute subtask *k*.	**C**	computational complexity of requested task
N_D	Descaled matrix	$\mathbf{n_{ij}}$	Descaled matrix element
V	Weighted descaled matrix	**nos**	Number of subtasks
c_k	computational complexity of subtask *k*	**PIS**	Positive ideal solution
a_k	data to be exchanged between resource *r* and RMS	$\mathbf{T_{pro}}$	processing time of a subtask on a resource
v_{ij}	Weighted descaled matrix element	$\mathbf{r_i}$	Resource i
NIS	Negative ideal solution	$\mathbf{x_r}$	processing speed of resource *r*
J	Benefit attribute	$\acute{j}$	Cost attribute
π_r	failure rate of the communication link between RMS and the resource *r*	$\mathbf{s_r}$	data transmission speed on the link between RMS and the resource *r*
$\acute{p}$	probability of failure in subtask execution	$\boldsymbol{\lambda_r}$	failure rate of resource *r*
$T_{transfer}$	sum of the transmission times of the data between RMS and resource	$\mathbf{Spt_s}$	sum of subtasks processing times on any of active resources in each state
T	execution time of subtask *k* providing that failure of resource and its communication link does not occur	**P**	probability of a subtask processing and sending its result to RMS by the resource in time T
Nou	number of all u(z) functions in group g	$\mathbf{I_i}$	Communication link between resource i and RMS
Nop	number of all possible subtask execution states in *u(z)*or $U_g(z)$	$\mathbf{t_r}$	the time that each resource is processing the subtask
rd_k	data to be sent from RMS to resource r in order to process the subtask k	$\mathbf{R_g}$	number of active resources in each $U_g(z)$ term
GPS	average processing speed in any group		

2. Background

2.1. Performance Metrics

In this research 4 performance metrics of grid environment have been surveyed. Among these 4 metrics, task execution time metric is related to user perspective, Resource Utilization Rate (RUR) and load balance level are related to manager perspective and service reliability can be considered from both manager's and final user's perspective. The aim of this type of selection is responding desirably to requests from user and manager of the grid system and the basis of the selection is their extensive usage in performed researches and studies on the area [2, 3, 4, 7, 9, 10, 13, 14, 15, 16, 18, 19, 21, 23, 28, 29].

- *Task Execution Time (Service Time)*: This is a random variable representing the task execution time through available resources in grid environment. This variable is influenced by several factors and may have different values based on them. These effective factors are:

1. Number and processing speed of available resources in grid environment.
2. How to schedule tasks.
3. Resource failure.
4. Data transmission speed (bandwidth) of communication links.
5. Communication links failure [18].

- *Service Reliability*: this is defined as the probability of successful execution of all subtasks related to a specified service [12].
- *RUR*: this is a percentage of a specific time interval in which the resource is executing the tasks. In grid environment this specific interval can be the task execution time. Due to existence of several resources, average RUR is usually used in order to evaluate overall RUR of grid environment [2, 10, 16].
- Load balance level: this metric represents the level of load balance between available resources in grid environment while executing tasks.

2.2. TOPSIS Technique

This is a technique of Multiple Attribute Decision Making (MADM) introduced by Huang and Yun in 1981. The aim of the technique is to choose an alternative with minimum distance from Positive Ideal Solution (PIS) and maximum distance from Negative Ideal Solution (NIS). The technique has six steps as following [25]:

- *Step* 1: Descaling the metrics available in decision making matrix.

$$n_{ij} = d_{ij} \Big/ \sqrt{\textstyle\sum_{i=1}^{m} d_{ij}^{2}} \tag{1}$$

- *Step* 2: Creating weighted descaled matrix by using vector *W*.

$$V = N_D \times W_{n\times n} \tag{2}$$

- *Step* 3: Determining *PIS* and *NIS*.

$$\begin{aligned} PIS &= \{(\max_i V_{ij} \mid j \in J), (\min_i V_{ij} \mid j \in J') \mid i = 1,2,...m\} \\ NIS &= \{(\min_i V_{ij} \mid j \in J), (\max_i V_{ij} \mid j \in J') \mid i = 1,2,...m\} \end{aligned} \tag{3}$$

- *Step* 4: Calculating the separation from *NIS* and *PIS* between alternatives.

$$\begin{aligned} d_{i^+} &= \{\textstyle\sum_{j=1}^{n} (V_{ij} - V_j^{+})^2\}^{0.5}; i = 1,2,...m \\ d_{i^-} &= \{\textstyle\sum_{j=1}^{n} (V_{ij} - V_j^{-})^2\}^{0.5}; i = 1,2,...m \end{aligned} \tag{4}$$

- *Step* 5: Calculating the similarities to the *PIS.*

$$cl_{i^+} = \frac{d_{i^-}}{(d_{i^+} + d_{i^-})}; 0 \le cl_{i^+} \le 1; \; i = 1,2,...m \tag{5}$$

- *Step* 6: Ranking the alternatives based on cl_{i+} descending order.

3. The Proposed Method to Choose the Best Scheduling Algorithm Using TOPSIS Approach

In order to choose the most efficient algorithm among several scheduling algorithms, Technique for Order-Preference by Similarity to Ideal Solution (TOPSIS) approach is selected. The advantages and reasons of using TOPSIS are [24, 26, 27]:

1. The concept is reasonable and easily understandable due to its simple process.
2. It is programmable.
3. it has fewer computations than the other Multiple Criteria Decision Making (MCDM) methods such as Analytic Hierarchy Process (AHP), and is easily usable.
4. The number of steps in this approach is fixed, regardless of the number of attributes
5. It has a rational logic which is the basis of human choices;
6. It simultaneously calculates a numerical value for the best and worst alternatives.

3.1. Assumptions

1. Grid environment has star topology and RMS which is connected to all different resources distributed in it. There is only a single communication link between RMS and each of the resources and, whenever available, has a constant transmission speed and data transmission time is proportional to the amount of transmitted data.
2. The user requests the RMS to execute desired task and RMS divides the requested task into independent subtasks and then allocates them to the grid resources for processing.
3. Computational complexity of the requested task (number of operations) is equal to the sum of computational complexity of its subtasks.
4. Each resource, whenever available, is able to process any single subtask and starts executing allocated subtask immediately after receiving subtask data from RMS.
5. Each resource, whenever available, has a constant processing speed and the subtask processing time is proportional to computational complexity of the subtask.
6. Different resources and communication links have constant failure rate which are completely independent from each other.
7. The redundancy technique is used in order to improve reliability of task execution. So each subtask could be allocated to more than one resource (a group of resources) but each resource executes just one subtask.
8. Whenever a subtask is completed, RMS sends a message to all resources responsible for processing it to notify the task completion. The aim of sending this message is to stop the processing of those resources which are still processing the subtask, making them ready for processing next subtask. The cost of sending this message (time and network traffic) considering the resource consumption in the case of not sending this message can be ignored.
9. RMS is fully reliable and fast, so the task processing time by RMS (dividing into subtasks, sending the subtasks to the resources, receiving the results and integrating them into entire task output) is negligible when compared with subtasks' processing time. The assumption that RMS does not fail while executing service is reasonable because RMS is usually highly reliable and the task execution time is relatively short.
10. The data amount which should be sent from RMS to resource in order to process a subtask is considered as equal to the data amount sending from resource to RMS as a result.

It is obvious that failure of each resource responsible for processing subtask or their communication link will affect the average RUR. Also the task execution time may change. Indeed it can be said that task execution time and the average RUR are a function of available resources which can produce different outputs in the case of failure in resources or their communication links. A schema of the function is mentioned here as Equation (6).

$$\text{Average RUR/ task execution time} = F(r_1,r_2,r_3,...,r_k,l_1,l_2,l_3,...,l_k) \tag{6}$$

According to the above function, there are different compound states based on the probability of failure in resources and communication links, each compound has specified occurrence probability demonstrating the reliability to the state occurrence. There is an average RUR and task execution time for each state.

3.2. Proposed Algorithm

3.2.1. Stage 1: Determining The Accurate Values of Task Execution Time, Service Reliability and Average RUR

In [18], the universal generating function for modeling the execution of subtask k on a resource is stated as:

$$U(z) = p.z^{T_k} + p'.z^{\infty} \tag{7}$$

To calculate the average RUR two items should be attended:

1. Resources which have properly processed allocated subtasks and have sent the results to RMS.

2. Subtask processing time on each resource.

So Equation (7) is changed as following:

$$U(z)=p.z_{R}^{T_k}+p'.z_{R'}^{\infty}\Rightarrow U(z)=\sum_{i=1}^{nop}p_i.z_{R_i}^{T_i} \qquad (8)$$

In Equation (8) nop=2.

- *Step* 1: Calculating P, T, $\acute{p}$ for each subtask executed on resource R

According to the assumptions, RMS divides the task into independent subtask, so that:

$$C=\sum_{k=1}^{nos}c_k \qquad (9)$$

For subtask execution time (T_k), there is:

$$T_k=T_{pro}+T_{transfer} \qquad (10)$$

T_{pro} and $T_{transfer}$ are obtained as below:

$$T_{pro}=c_k/x_r \qquad (11)$$

$$T_{transfer}=a_k/s_r \qquad (12)$$

Therefore:

$$T_k=c_k/x_r+a_k/s_r \qquad (13)$$

p and $\acute{p}$are obtained as below:

$$p=p_{rpro}\times p_{rtransfer} \qquad (14)$$

$$p'=1-p \qquad (15)$$

In Equation (14), p_{rpro} is the probability that resource r does not fail at the time interval $[0, c_k/x_r\ ]$ and $p_{rtransfer}$ is the probability that communication link at the time interval$\left[0, a_k/s_r\right]$does not fail. These are calculated as following [18]:

$$p_{rpro}=e^{-\lambda_r(c_k/x_r)} \qquad (16)$$

$$p_{rtransfer}=e^{-\pi_r(a_k/s_r)} \qquad (17)$$

- *Step* 2: Achieving different subtask execution states on each group

RMS considers the first response from resources existing in any group as the result of subtask execution and after receiving the response, it will send a notification message to all available resources in the group. According to this, it can be said that the execution time of any subtask allocated to a group g of resources (including n resources), is calculated as following:

$$T_g=\min(T_k);k=1,2,...n \qquad (18)$$

It is obvious that considering the probability of failure in resource and communication links, T_g may adopt different values. So, all possible states of subtask execution resulting from these failures should be calculated. To find all possible states, $U(z)$s related to all resources in a group should be multiplied by each other using a special operator ($\otimes_{\cup}^{min}$) which covers all the purposes; therefore, there will be a new $U_g(z)$, so that:

$$U_g(z)=U_1(z)\otimes_{\bullet}^{\min}U_2(z)\otimes_{\bullet}^{\min}..U_{nou}(z) \\ =\sum_{s=1}^{nop_g}P_s.z_{R_s}^{T_s} \qquad (19)$$

$\otimes_{\cup}^{\min}$ is defined as following:

$$U_i(z)\otimes_{\bullet}^{\min}U_j(z)=\sum_{k=1}^{nop_i}\sum_{h=1}^{nop_j}P_{ik}.P_{jh}.z_{R_{ik}\bullet R_{jh}}^{\min(T_{ik},T_{jh})} \qquad (20)$$

Every term in obtained final $U_g(z)$ for any group-corresponding to a subtask execution state- implies the fact that, with the probability of P_s, R_s set of resources successfully executes allocated subtask and sends the results back to RMS. The minimal possible time to receive the results by RMS is equal to T_s at any state. In the case of failure in executing subtask, R_s is a null set and T_s is equal to ∞.

In order to calculate the average RUR, it is necessary to calculate the sum of subtasks processing times on all resources involved in processing the task; so, in this step, the sum of subtasks processing times on any of active resources in each $U_g(z)$ term is calculated as Equation (21) and added as an index to $U_g(z)$terms.

$$spt_s=\sum_{r=1}^{R_g}t_r \qquad (21)$$

In Equation (21) $t_r$is calculated as following:

$$t_r=\min(T_s-(rd_k/s_r),c_k/x_r) \qquad (22)$$

According to assumptions, $rd_k=a_k/2$. So, the Equation (19) will be as following:

$$U_g(z)=\sum_{s=1}^{nop_g}P_s.(_{spt_s}z_{R_s}^{T_s}) \qquad (23)$$

- *Step* 3: Obtaining all different states of task execution in whole grid environment

Similar to step 2, an especial multiplying operator is used to achieve all different states of task execution in whole grid environment ($U_T(z)$). Since RMS should wait for receiving the results of all subtasks allocated to the groups in order to produce task final response, so the final time of task execution is determined by the last group sending the results of allocated subtask. The new operator which mentioned above operates as follow:

$$U_{g_i}(z)\otimes_{+,\bullet}^{\max}U_{g_j}(z)= \\ \sum_{k=1}^{nop_{g_i}}\sum_{h=1}^{nop_{g_j}}P_{g_ik}.P_{g_jh}.(_{spt_{g_ik}+spt_{g_jh}}z_{R_{g_ik}\bullet R_{g_jh}}^{\max(T_{g_ik},T_{g_jh})}) \qquad (24)$$

$U_T(z)$ obtained in this step is :

$$U_T(z)=U_{g_1}(z)\otimes_{+,\bullet}^{\max}U_{g_2}(z)\otimes_{+,\bullet}^{\max}..U_{g_n}(z) \\ =\sum_{s=1}^{n}P_{T_s}.(_{spt_{T_s}}z_{R_{T_s}}^{T_{T_s}}) \qquad (25)$$

Where n is total number of possible task execution states. Every term in final $U_T(z)$ -corresponding to a task execution state- implies that if R_T set of resources can successfully process subtasks allocated to them, and results are successfully received by RMS, then total task will be completely executed at time T_T and the occurrence probability of this state -reliability value- will be equal to P_T.

- *Step* 4: Calculating the values of average execution time, service reliability and average RUR

To obtain occurrence probability of any possible execution time, it is simply needed to add different occurrence probabilities of that time. According to [2, 10, 16, 21, 23] and the fact that the given grid environment consists of heterogeneous resources and the focus is on calculating performance metrics (especially RUR) while executing a single task, average utilization rate for each term is calculated using Equation (26):

$$U = \sum_{r=1}^{n_R} t_r \big/ T.m \qquad (26)$$

In this Equation,n_R is the number of resources in R_T set,T is the task execution time and m indicates the number of available resources in grid environment. To calculate the average utilization rate for each term in $U_T(z)$, *spt* is placed in numerator and denominator is calculated using the number of resources in grid environment and completion time T_T. Weight average is calculated to obtain average utilization rate for any possible task execution time using occurrence probability of that state. In final $U_T(z)$ Function, system reliability and average task execution time can be obtained by respectively summing probabilities of task execution time and obtaining weight average of all possible task execution time using their probabilities. Also, the average RUR in whole grid environment can be obtained by calculating weight average of average utilization rate of any possible task execution time using their probabilities.

3.2.2. Stage 2: Calculating Load Balance Level in Grid Environment

Standard deviation of workload imposed to grid environment toward Optimal Load (OL) introduced in [8] is utilized to handle load balance level in different scheduling algorithms; Contrary to the [8], in the proposed method, the grid environment based on star topology which uses resource redundancy technique is applied. Therefore, the optimal workload for each group is calculated through following Equation:

$$OL_g = \frac{GPS_g}{\sum_{i=1}^{n} GPS_i}.C \qquad (27)$$

In which, n is the number of groups. *GPS* is calculated as following:

$$GPS_g = \sum_{r=1}^{m} x_r \big/ m \qquad (28)$$

In this Equation, m is the number of available resources in the group g.

To evaluate the difference between load determined by any algorithm and optimal load of grid environment, the Standard Deviation (SD) of Determined Load (DL) toward the optimal load of each group is used which is calculated as following:

$$sd = \sqrt{\frac{\sum_{g=1}^{n}(OL_g - DL_g)^2}{n}} \qquad (29)$$

Where n is the number of groups. It is obvious that closer the *Sd* to zero, more balanced workload will be.

3.2.3. Stage 3: Choosing the most Efficient Algorithm Using TOPSIS Technique

After calculating the values of grid performance metrics for different algorithms, these algorithms can be ranked using TOPSIS technique. The values obtained for performance metrics constitute the decision making matrix for TOPSIS technique. In the given decision making matrix, the metrics of reliability and RUR are benefit metrics, and the metrics of task execution time and load balance level are cost metrics.

4. Illustrative Examples

In this section two analytical and numerical examples in [18] have been used to show the function of the proposed algorithm.

4.1. Analytical Example

A grid service which utilizes four resources is considered and the requested service task has complexity of 4000 mega operation (C=4000). Data amount exchanged between RMS and the resource to each subtask is proportional to computational complexity of this subtask: a_k=5%c_k. Parameters relevant to resources and communication links are mentioned in Table 2. RMS divides the task into two subtasks with the computational complexity of 2000 and allocates subtask 1 to resources 1 and 2 and subtask 2 to resources 3 and 4. The aim is to compute values of each performance metric.

Table 2. Parameters of grid resources and communication links.

No of resource (link) k	X_r(mega operations/s)	$\lambda_r(s^{-1})$	$\pi_r(s^{-1})$	Sr(megabytes/s)
1	10	0,0006	0,002	3
2	12	0,001	0,003	5
3	15	0,001	0,002	3
4	15	0,0015	0,001	7

According to problem description c_1= c_2=2000, DL_1=DL_2=2000 and rd_i=50; c_1 is allocated to resources 1 and 2; $c_2$is allocated to resources 3 and 4.

4.1.1. Stage 1

- *Step* 1:

$$U_1(z) = 0.8297.z_{\{1\}}^{233.33} + 0.1703.z_{\{\}}^{\infty}$$
$$U_2(z) = 0.7972.z_{\{2\}}^{186.67} + 0.2028.z_{\{\}}^{\infty}$$
$$U_3(z) = 0.8187.z_{\{3\}}^{166.67} + 0.1813.z_{\{\}}^{\infty}$$
$$U_4(z) = 0.8071.z_{\{4\}}^{147.62} + 0.1929.z_{\{\}}^{\infty}$$

- *Step* 2:

$$U_{g_1}(z) = 0.6614._{336.67}z_{\{1,2\}}^{186.67} + 0.1683._{200}z_{\{1\}}^{233.33} + 0.1358._{166.67}z_{\{2\}}^{186.67} + 0.0345._{0}\,z_{\{\}}^{\infty}$$

$$U_{g_2}(z) = 0.6608._{264.29}z_{\{3,4\}}^{147.62} + 0.1579._{133.33}z_{\{3\}}^{166.67} + 0.1463._{133.33}z_{\{4\}}^{147.62} + 0.035._{0}\,z_{\{\}}^{\infty}$$

- *Step* 3:

$$\begin{aligned}U_T(z) &= U_{g_1}(z) \otimes_{+,\cup}^{\max} U_{g_2}(z)\\ &= 0.4371._{600.96}z_{\{1,2,3,4\}}^{186.67} + 0.1044._{470}z_{\{1,2,3\}}^{186.67} +\\ &\quad 0.0968._{470}z_{\{1,2,4\}}^{186.67} + 0.0231._{336.67}z_{\{1,2\}}^{\infty} +\\ &\quad 0.1112._{464.29}z_{\{1,3,4\}}^{233.33} + 0.0266._{333.33}z_{\{1,3\}}^{233.33} +\\ &\quad 0.0246._{333.33}z_{\{1,4\}}^{233.33} + 0.0059._{200}z_{\{1\}}^{\infty} +\\ &\quad 0.0897._{430.96}z_{\{2,3,4\}}^{186.67} + 0.0214._{300}z_{\{2,3\}}^{186.67} +\\ &\quad 0.0199._{300}z_{\{2,4\}}^{186.67} + 0.0048._{166.67}z_{\{2\}}^{\infty} +\\ &\quad 0.0228._{264.29}z_{\{3,4\}}^{\infty} + 0.0054._{133.33}z_{\{3\}}^{\infty} +\\ &\quad 0.005._{133.33}z_{\{4\}}^{\infty} + 0.0012._{0}\,z_{\{\}}^{\infty}\end{aligned}$$

- *Step* 4:

$$P(T_1 = 186.67) = 0.7693, P(T_2 = 233.33) = 0.1624,$$
$$P(T_3 = \infty) = 0.0683$$

Values of the average utilization rate for each term of $U_T(z)$ are calculated as following:

$$U_{T_1} = \frac{600.96}{186.67 \times 4} = 0.8048, U_{T_2} = 0.6295,$$
$$U_{T_3} = 0.6295, U_{T_5} = 0.4974, U_{T_6} = 0.3571,$$
$$U_{T_7} = 0.3571, U_{T_9} = 0.5772, U_{T_{10}} = 0.4018,$$
$$U_{T11} = 0.4018$$

Since the executing of requested task in other terms has been failed, the average utilization rate for them is equal to zero. Weight average of average utilization rate for the execution time 233.33 $(\overline{UT_1})$ and 186.67 $(\overline{UT_2})$ are:

$$\overline{UT_1} = \frac{P_5.U_{T_5} + P_6.U_{T_6} + P_7.U_{T_7}}{P_5 + P_6 + P_7} = 0.4532$$
$$\overline{UT_2} = 0.7108$$

Using values above, average execution time, overall reliability and average RUR are:

$$\text{Reliability} = 0.1624 + 0.7693 = 0.9317$$

Average execution time=

$$\frac{186.67 \times 0.7693 + 233.33 \times 0.1624}{0.1624 + 0.7693} = 194.8$$

Average RUR=

$$\frac{0.4532 \times 0.1624 + 0.7108 \times 0.7693}{0.1624 + 0.7693} = 0.6659$$

4.1.2. Stage 2

$$GPS_1 = (x_1 + x_2)/2 = 11, GPS_2 = 15$$
$$OL_1 = \frac{GPS_1}{GPS_1 + GPS_2} \times C = 1692, OL_2 = 2308$$
$$Load\ balance\ level:$$
$$sd = \sqrt{\frac{(OL_1 - DL_1)^2 + (OL_2 - DL_2)^2}{2}} = 308$$

4.2. Numerical Example

Consider a grid environment that uses six resources with service task of 6000 mega operation complexity. The parameters of resources and communication link are presented in Table 3.

Table 3. Parameters of grid resources and communication links.

No of resource (link) *j*	$\lambda_j(s^{-1})$	X_j(mega operation/s)	$\pi_j\ (s^{-1})$	S_j(megabyte /s)
1	0.0006	10	0.002	3
2	0.001	12	0.003	5
3	0.001	15	0.002	3
4	0.0015	15	0.001	7
5	0.001	20	0.003	4
6	0.0015	20	0.002	7

The amount of transmitted data between each resource and RMS is 5% of the complexity of the subtask allocated to that resource. There are four different algorithms considered to divide and distribute subtasks between the resources, demonstrated in Table 4. The average of performance metrics values evaluated using the proposed method is mentioned in Table 5. According to the values obtained, the most efficient algorithm can be obtained using TOPSIS as following. Assuming the weight values of metrics of reliability, task execution time, RUR and load balance level are 0.3, 0.2, 0.3 and 0.2 respectively. In Table 4, *R* refers to the resources sets.

Data of Table 5 constitute the values of decision making matrix for TOPSIS technique.

- *Step* 1 : Descaling data matrix (N_D Matrix)

$$N_D = \begin{bmatrix} 0.5156 & 0.5047 & 0.5082 & 0.0855 \\ 0.5156 & 0.6156 & 0.4743 & 0.7700 \\ 0.4837 & 0.4575 & 0.4997 & 0.5456 \\ 0.4841 & 0.3962 & 0.5167 & 0.3197 \end{bmatrix}$$

- *Step* 2: Creating weighted descaled matrix (V)

$$V = N_D \times W_{n\times n} = \begin{bmatrix} 0.1547 & 0.1009 & 0.1525 & 0.0171 \\ 0.1547 & 0.1231 & 0.1423 & 0.1540 \\ 0.1451 & 0.0915 & 0.1499 & 0.1091 \\ 0.1452 & 0.0792 & 0.1550 & 0.0639 \end{bmatrix}$$

- *Step* 3: Determining *PIS* and *NIS*
  PIS={0.1547, 0.0792, 0.1550, 0.0171}
  NIS={0.1451, 0.1231, 0.1423, 0.1540}
- *Step* 4: The separation from and *NIS* and *PIS* (d_{i+}, d_{i-} in Table 6).
- *Step* 5: The similarities to the *PIS* (cl_{i+} in Table 6).
- *Step* 6: Ranking the alternatives based on cl_{i+} descending order (Rank in Table 6).

Table 4. Distribution algorithms.

Algorithm	*c and R*	*c and R*	*c and R*
A	c=3000,*R*={1,3,5}	c=3000,*R*={2,4,6}	
B	c=3000,*R*={1,2,3}	c=3000,*R*={4,5,6}	
C	c=2000,*R*={1,2}	c=2000,*R*={3,4}	*c*=2000,*R*={5,6}
D	c=2000,*R*={1,4}	c=2000,*R*={2,5}	c=2000,*R*={3,6}

Table 5. Evaluated average of the performance metrics values.

Algorithm	Reliability	Service Time	RUR	Load Level
A	0.9670	214	60%	92.23
B	0.9671	261	56%	830.08
C	0.9072	194	59%	588.16
D	0.9079	168	61%	334.67

Table 6. Results of four to six steps of TOPSIS technique.

	A	B	C	D
d_{i+}	0.0218	0.1443	0.0934	0.0478
d_{i-}	0.1394	0.0096	0.0554	0.1010
cl_{i+}	0.8645	0.0623	0.3723	0.6788
Rank	1	4	3	2

5. Results and Discussion

Since the evaluating method of average RUR in proposed algorithm is a new method, in this section the accuracy and correctness of this evaluating method have been examined. A simulated application is implemented in MATLAB R2015a to survey the accuracy of average RUR which is obtained from the proposed algorithm. The simulation is performed based on the data available in numerical example of section 4.2 for scenarios *A* and *D*. To achieve a more real estimation for average RUR, random normal distribution with the average of parameters values in section 4.2 and variance 0.05 is used instead of constant values for resources processing speed and communication link bandwidth. 100 values have been generated for each resource and communication link, and their average is used as the resource processing speed parameters as well as communication link bandwidth. The corresponding parameters presented in section 4.2 are used for both resources and communication links failure rate parameters.

The simulation application has been run 500 times for each scenario and the average utilization rate is calculated for different execution time. In Figures 1and 2, the values obtained for average utilization rate for each possible execution time -for case A&D- are mentioned beside the values resulted from analytical approach which implies the accuracy and proximity of analytical results to the real results.

As Figures 1 and 2 show, it's obvious that the estimated average RUR by presented algorithm is closely similar to simulation results. So, the method can be used for the RUR evaluation in grid environment.

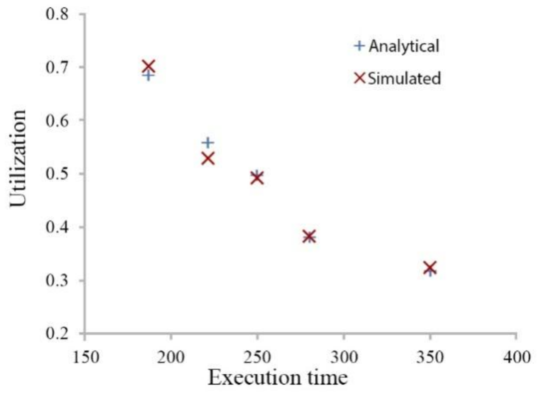

Figure 1. Simulated results (500 runs) vs. analytical results for case A.

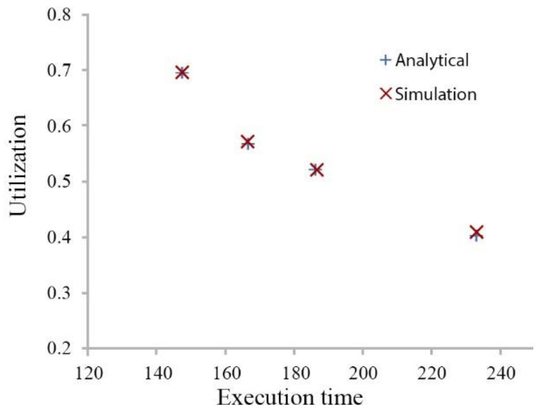

Figure 2. Simulated results (500 runs) vs. analytical results for case D.

6. Conclusions

This paper represents four metrics of reliability, task execution time, and RUR and load balance level as the metrics of performance. It provides an algorithm to show the way of evaluating these metrics. One of the innovations of this article is the method of evaluating RUR in the proposed algorithm. Using simulation, it is proved that average RUR is estimated in high accuracy. Using TOPSIS technique, this paper also shows how to choose an algorithm (as the most efficient) to be used in RMS in order to divide and distribute tasks in grid environment; so that, a compromise between performance metrics in the chosen algorithm is established and the requests from system users and managers are properly replied.

References

[1] Azgomi M. and Entezari-Maleki R., "Task Scheduling Modelling and Reliability Evaluation of Grid Services Using Coloured Petri Nets," *Future Generation Computer Systems*, vol. 26, no. 8, pp. 1141-1150, 2010.

[2] Cao J., Spooner D., Jarvis S., and Nudd G.,"Grid Load Balancing Using Intelligent Agents," *Future Generation Computer Systems*, vol. 21, no. 1, pp. 135-149, 2005.

[3] Chauhan S. and Joshi R., "QoS Guided Heuristic Algorithms for Grid Task Scheduling," *International Journal of Computer Applications*, vol. 2, no. 9, pp. 24-31, 2010.

[4] Dabrowski C., "Reliability in Grid Computing Systems," *Concurrency and Computation Practice and Experience*, vol. 21, no. 8, pp. 927-959, 2009.

[5] Dai Y. and Levitin G., "Reliability and Performance of Tree-Structured Grid Services," *IEEE Transactions on Reliability*, vol. 55, no. 2, pp. 337-349, 2006.

[6] Dai Y., Levitin G., and Trivedi K., "Performance and Reliability of Tree-Structured Grid Services Considering Data Dependence and Failure Correlation," *IEEE Transactions on Reliability*, vol. 56, no. 7, pp. 925-936, 2007.

[7] Dai Y., Xie M., and Poh K., "Reliability Analysis of Grid Computing Systems," *in Proceedings of Pacific Rim International Symposium on Dependable Computing*, Tsukuba-City, pp. 97-104, 2002.

[8] El-Zoghdy S. and Aljahdali S., "A Two-Level Load Balancing Policy for Grid Computing," *in Proceedings International Conference on Multimedia Computing and Systems*, Tangier, pp. 617-622, 2012.

[9] Etminani K. and Naghibzadeh M., "A Min-Min Max-Min Selective Algorihtm for Grid Task Scheduling," *in Proceedings of 3rd IEEE/IFIP International Conference in Central Asia on Internet*, Tashkent, pp. 1-7, 2007.

[10] Farooq U., Majumdar S., and Parsons E., "Engineering Grid Applications and Middleware for High Performance," *in Proceedings of 6th International Workshop on Software and Performance*, Buenos Aires, pp. 141-152, 2007.

[11] Foster I., Kesselman C., and Tuecke S., "The Anatomy of the Grid: Enabling Scalable Virtual Organizations," *The International Journal of High Performance Computing Applications*, vol. 15, no. 3, pp. 200-222, 2001.

[12] Guo S., Huang H., Wang Z., and Xie M., "Grid Service Reliability Modeling and Optimal Task Scheduling Considering Fault Recovery," *IEEE Transactions on Reliability*, vol. 60, no. 1, pp. 263-274, 2011.

[13] Hamscher V., Schwiegelshohn U., Streit A., and Yahyapour R., "Evaluation of Job-Scheduling Strategies for Grid Computing," *in Proceedings of First IEEE/ACM International Workshop on Grid Computing*, Bangalore, pp. 191-202, 2000.

[14] Huedo E., Montero R., and Liorente I., "An Evaluation Methodology for Computational Grids," *in Proceedings of International Conference on High Performance Computing and Communications*, Sorrento, pp. 499-504, 2005.

[15] Huedo E., Montero R., and Liorente I., "Evaluating the Reliability of Computational Grids from the End User's Point of View," *Journal of Systems Architecture*, vol. 52, no. 12, pp. 727-736, 2006.

[16] Izakian H., Abraham A., and Ladani BT.,"An Auction Method for Resource Allocation in Computational Grids," *Future Generation Computer Systems*, vol. 26, no. 2, pp. 228-235, 2010.

[17] Levitin G. and Dai Y., "Optimal Service Task Partition and Distribution in Grid System with Star Topology," *Reliability Engineering and System Safety*, vol. 93, no. 1, pp. 152-159, 2008.

[18] Levitin G. and Dai Y., "Service Reliability and Performance in Grid System with Star Topology," *Reliability Engineering and System Safety*, vol. 92, no. 1, pp. 40-46, 2007.

[19] Li H. and Buyya R., "Model-based simulation and Performance Evaluation of Grid Scheduling Strategies," *Future Generation Computer Systems*, vol. 25, no. 4, pp. 460-465, 2009.

[20] Ong S., "Grid Computing: Business and Policy Implications," Master Thesis, Cambridge United States, 2003.

[21] Peng L., See S., Jiang Y., Song J., Stoelwinder A., and Neo H., "Performance Evaluation in Computational Grid Environments," *in Proceedings of 7th International Conference on High Performance Computing and Grid in Asia Pacific Region*, Tokyo, pp. 54-62, 2004.

[22] Selvarani S. and Sadhasivam G., "An Intelligent Water Drop Algorithm for Optimizing Task Scheduling in Grid Environment," *The International Arab Journal of Information Technology*, vol. 13, no. 6, pp. 627-643, 2016.

[23] Shan H., Oliker L., and Biswas R., "Job Superscheduler Architecture and Performance in Computational Grid Environments," *in Proceedings of ACM/IEEE SC2003 Conference on Supercomputing*, Phoenix, pp. 44-44, 2003.

[24] Shih H., Shyur H., and Lee E., "An Extension of TOPSIS for Group Decision Making," *Mathematical and Computer Modelling*, vol. 45, no. 7, pp. 801-813, 2007.

[25] Tzeng G. and Huang J., *Multiple Attribute Decision Making: Methods and Applications*, CRC Press, 2011.

[26] Velasquez M. and Hester P., "An Analysis of Multi-Criteria Decision Making Methods," *International Journal of Operations Research*, vol. 10, no. 2, pp. 56-66, 2013.

[27] Vimal J., Chaturvedi V., and Dubey A., "Application of TOPSIS Method for Supplier Selection in Manufacturing Industry," *International Journal of Research in Engineering and Applied Sciences*, vol. 2, no. 5, pp. 25-35, 2012.

[28] Wu Y., Liu L., Mao J., Yang G., and Zheng W., “An Analytical Model for Performance Evaluation in A Computational Grid,” *in Proceedings of the Asian Technology Information Program's 3rd Workshop on High Performance Computing in China: Solution Approaches to Impediments for High Performance Computing*, Reno, pp. 145-151, 2007.

[29] Yin F., Jiang C., Deng R. and Yuan J., “Grid Resource Management Policies for Load-Balancing and Energy-Saving By Vacation Queuing Theory,” *Computers and Electrical Engineering*, vol. 35, no. 6, pp. 966-979, 2009.

Mahmoud Mohammaddoust is currently a M.S. student in Computer engineering in Institute for Higher Education ACECR Khouzestan. His main research interests include grid computing, performance evaluation, and task scheduling algorithms.

Ali Harounabadi is currently an associate professor in Computer Engineering at the Department of Computer in Islamic Azad University, Central Tehran Branch. His main research interests include grid computing, performance evaluation, software engineering, formal models, data base, and artificial intelligence.

Mohammadali Naizari is currently a faculty at Department of Computer Engineering in Institute for Higher Education ACECR Khouzestan. His main research interests include data base, grid computing, cloud computing, task scheduling algorithms, and parallel programing.