Journal of Advances in Computer Research
Quarterly pISSN: 2345-606x eISSN: 2345-6078
Sari Branch, Islamic Azad University, Sari, I.R.Iran
(Vol. 7, No. 2, May 2016), Pages: 1-21
www.jacr.iausari.ac.ir

Reliability Evaluation in Tree-structured Grid Services with Colored Petri Net

Mirsaeid Hosseini Shirvani✉, Mahdie Khanramaki

Department of Computer Engineering, Sari Branch, Islamic Azad University, Sari, Iran

mirsaeid_hosseini@yahoo.com; m.khaanramaki@yahoo.com

Received: 2014/12/06; Accepted: 2015/08/30

Abstract

Grid computing is an emerging computing paradigm that will have significant impact on the next generation information infrastructure. Due to the largeness and complexity of grid system, its quality of service, performance and reliability are difficult to model, analyze and evaluate. This paper proposes a virtual tree-structured model of the grid service. This model simplifies the physical structure of a grid service because there are several links between RMS and resources. In this paper, the task scheduling by RMS and the task execution within grid resources considering data dependence and failure correlation are modeled using colored Petri nets (CPNs). We have also proposed a method for evaluation the grid service reliability based on the analysis of the model. In addition, an instance of the proposed model for a sample grid environment is constructed and analyzed using CPN Tools.

Keywords: *grid computing, resource management system, tree structured, reliability evaluation, colored petri nets*

1. Introduction

GRID computing is a newly developed technology for complex systems with large-scale resource sharing, wide-area communication, and multi-institutional collaboration. It is hard to analyze and model the Grid reliability because of its largeness, complexity and stiffness [1]. The service/resource integration, request, and management are controlled by the RMS (Resource Management System) that uses variety protocols helping to schedule with data replication strategy such as Condor [2] and Stork [3]. A grid service is desired to execute a certain task under the control of the RMS. The RMS receives a service request from a user, and assigns the service task to available resources for execution. After the resources finish execution of the assigned subtasks, they return the results back to the RMS; and then the RMS integrates the received results into an entire task output, which is requested by the user [3]. Miscellaneous architectures and topologies have been presented for especial purposes in literature, but for the sake of scalability, reliability and generalization we apply tree-based topology. The root of the tree virtual structure is the RMS, and the leaves are resources, while the branches of the tree represent the communication channels linking the leaves and the root. Some channels are commonly used by multiple resources. The tree structure models the common cause failures in shared communication channels. It is commonly accepted that the service time in the grid is a random variable. Finding the distribution of this variable

is important for evaluating the grid performance, and improving the RMS functioning. In this paper with use of the distribution of service time in the grid computing system can obtain reliability. The real and specific problem that underlies the grid concept is coordinated resource sharing and problem solving in dynamic, multi-institutional virtual organizations. The sharing that we are concerned with is not primarily file exchange but rather direct access to computers, software, data, and other resources [5]. The service time experienced by the users is usually random and affected by many factors. Different resources usually have respective task processing speeds. Thus, the task execution time can vary, depending on which resource is assigned to execute the task/subtasks and on the availability of those resources. Moreover, the communication links in grid service can be disconnected during the data transmission, so the communication reliability influences the service time, as well as the data transmission speed. Finally, the data dependence imposes constraints on the sequence of the subtasks' execution, which has been observed to have significant influence on the service time [6]. Multi-criteria grid scheduling can be seen as a family of complex optimization, management and decision making problems with several often conflicting criteria, which makes many standard approaches ineffective [7]. However, less attention has been paid to the modeling of the task scheduling, subtask execution and reliability evaluation in a grid environment.

Several works suggested using a star topology to simplify the grid structure [10, 11]. In a star grid, the RMS is connected with each resource by one direct communication channel (link). However, such an approximation is not accurate enough even though it simplifies the analysis and computation.

This paper proposes a virtual tree-structured model of the grid service. This model simplifies the physical structure of a grid service because there are several links between RMS and resources. Most of the existing studies assume that the failures of subtasks assigned to different resources/processors are independent. However, even though failures of different resources may be independent, the failure correlation of different subtasks exists because the resources executing the subtasks need to receive the data from the RMS and send the results back. Therefore, different independent resources may share common communication links. When failure occurs in a common link, all of the subtasks using the same link cannot transmit data that comply with a common cause failure. But that such researches there is no any modeling and only mathematic calculating done. Although in last researches mathematical solutions are proposed to evaluate the reliability of grid services, a formal model is not presented for the grid environment. We have also proposed a method for evaluation the grid service reliability based on the analysis of the model. In this paper, the task scheduling by RMS and the task execution within grid resources considering data dependence and failure correlation are modeled using colored Petri nets (CPNs).

The rest of this paper is organized as follows. In section 2 we review previous researches in relation to our study. In section 3 we describe some required definition in this paper. In section 4 describe how to calculating task execution time, its distribution and reliability. In section 5 our proposed colored petri net model for task scheduling is expressed. In section 6 examples of tree structured and calculating reliability for them is mentioned and in section 7 conclusions is expressed.

2. Related Work

Myriad works have been proposed in literature to cover reliability of grid services. Dai et al. in [4] calculated reliability and performance of tree structured grid services. The paper proposes a virtual tree model of the grid service. This model simplifies the physical structure of a grid service, allows service performance (execution time) to be estimated, and takes into account the common cause failures in communication channels. Based on the model, an algorithm for evaluating the grid service performance distribution and the service reliability indices is suggested. Illustrative examples are presented in which the results of the suggested algorithm are compared with simulation results.

Li et al. in [5] suggested reliability and performance models for grid computing. It introduces the Grid computing technology, presents different types of failures in grid system, models the grid reliability with star structure and tree structure, and finally studies optimization problems for grid task partitioning and allocation. The chapter then presents models for star-topology considering data dependence and tree-structure considering failure correlation. Then, the failure correlation and data dependence are considered in the model. Numerical examples are illustrated to show the modeling and analysis.

Dai et al. in [6] propose a virtual tree-structured model of the grid service. This model allows service performance (execution time) to be efficiently evaluated, and takes into account data dependence and failure correlation but no modeling of grid is shown in it. Only in [8] Abdollahi Azgomi et al. the task scheduling by RMS and the task execution of star topology within grid resources are modeled using colored Petri nets (CPNs). The proposed CPN-based modeling pattern formally describes the process of task distribution and execution within the grid environment. It has been also proposed a method for evaluation the star structured grid service reliability based on the analysis of the model. But the topology using in this paper is tree that rather than star topology which is used in prior researches is more reasonable and adaptable with nature word.

Dai et al. in [10] According to the optimal schedule for service task partition, and distribution among resources, one can achieve the greatest possible expected service performance (i.e. least execution time), or reliability. For solving this optimization problem, the paper suggests an algorithm that is based on graph theory, Bayesian approach, and the evolutionary optimization approach.

Zhang et al. in [[11]11] consider a system in tree networks, and study a basic allocation problem: given a set of jobs, each demanding bandwidth and a set of cumulative computing resources and yielding profits corresponding to different machines, determine which feasible subset of jobs yields the maximum total profit.

J Akbari in [12] a parity-based striped mirroring architecture for data distribution has proposed. The specific data distribution algorithm proposed in this paper also improves the access bandwidth and data availability of the array. Same as before in this paper no simulation were found.

As Grid jobs are categorized in two broad taxonomies, i.e., data intensive and cpu intensive, for the sake of efficient processors scheduling and data management miscellaneous architectures and topologies have been presented for especial purposes in literature. Graph topology strategy has been applied by M.Tu et al.[13], Rangathatan et al. in [14] have suggested multi-tier architecture, Chen et al. have investigated free-

scale architecture in [15] in which are just a few feats. So, for the sake of scalability, reliability and generalization we apply tree-based topology.

In all of these papers, algorithms that calculate reliability in grid services with tree topology are presented, but modeling task scheduling and distribution together with calculating reliability have been less attention.

3. Literature Review

3.1 Description of the Grid Computing

Grid Computing was a vision of using and sharing computers and data. The integration, coordination, and sharing of computer resources which are geographically disperse among different physical domains and organizations became an everyday reality. This emerging infrastructure aims to provide a mechanism for sharing and coordinating the use of heterogeneous computing resources.

Up to now, this new evolving technology achieved to provide to its users the ability to utilize at maximum the existing resources across a network. Grid Computing takes collective advantage of the vast improvements that have occurred over the last few years, in microprocessor speeds, optical communications, storage capacity, the World Wide Web, and the Internet. A set of standards and protocols are being developed that completely disaggregate current compute platforms and distribute them across a network as resources that can be called into action by any eligible user or machine at any time[16].

Today, the Grid computing systems are large and complex, such as the IP-Grid (Indiana-Purdue Grid) that is a statewide grid [21]. IP-Grid is also a part of the TeraGrid that is a nationwide grid in the USA [22]. The largeness and complexity of the grid challenge the existing models and tools to analyze, evaluate, predict and optimize the reliability and performance of grid systems.

Even though all online nodes or resources are linked through the Internet with one another, not all resources or communication channels are actually used for a specific service. Therefore, according to this observation, we can make tractable models and analyses of grid computing via a virtual structure for a certain service. The grid service is defined as a service offered under the grid computing environment, which can be requested by different users through the RMS, which includes a set of subtasks that are allocated to specific resources via the RMS for execution, and which returns the result to the user after the RMS integrates the outputs from different subtasks [4].

In grid computing systems which the resource management systems (RMS) can divide service tasks into execution blocks (EB), and send these blocks to different resources. To provide a desired level of service reliability, the RMS can assign the same EB to several independent resources for parallel (redundant) execution. According to the optimal schedule for service task partition, and distribution among resources, one can achieve the greatest possible expected service performance (i.e. least execution time), or reliability. For solving this optimization problem, the paper suggests an algorithm that is based on graph theory, Bayesian approach, and the evolutionary optimization approach. A virtual tree-structure model is constructed in which failure correlation in common communication channels is taken into account. Illustrative examples are presented [5].

Grid computing systems (machines) pool together the resources of a heterogeneous collection of computing systems that are widely distributed, possibly around the globe,

to create a virtual computing organization. Users can "draw" resources either from local or from remote computing resources to execute their jobs. We only consider such a system in tree networks, and study a basic allocation problem: given a set of jobs, each demanding bandwidth and a set of cumulative computing resources and yielding profits corresponding to different machines, determine which feasible subset of jobs yields the maximum total profit [6].

3.2 Virtual Tree Structure of Grid Services

Today, the grid computing systems become increasingly large and complicated. Such largeness and complexity challenge the existed models and tools to analyze, evaluate, predict, and optimize the of the grid systems. Even though all online nodes or resources are linked through the Internet with one another, only a small portion of resources or communication channels is indeed used for a specific service. Therefore, according to this observation, we can make tractable the model and analysis of grid performance and reliability by presenting a virtual structure for a grid service [[10]17].

For the grid service, the start and terminal points are the same, at the RMS, and the service requires a set of resources that are distributed in the grid. Several works suggested using a star topology to simplify the grid structure. In a star grid, the RMS is connected with each resource by one direct communication channel (link). However, such an approximation is not accurate enough even though it simplifies the analysis and computation. For example, several resources located in a same LAN can use the same gateway to communicate outside the network. Therefore, all these resources are not connected with the RMS through independent links. The resources are connected to the gateway, which communicates with the RMS through one common communication channel.

In this paper, we present a more reasonable virtual structure which has a tree topology. The root of the tree virtual structure is the RMS, and the leaves are resources, while the branches of the tree represent the communication channels linking the leaves and the root. Some channels are commonly used by multiple resources [4].

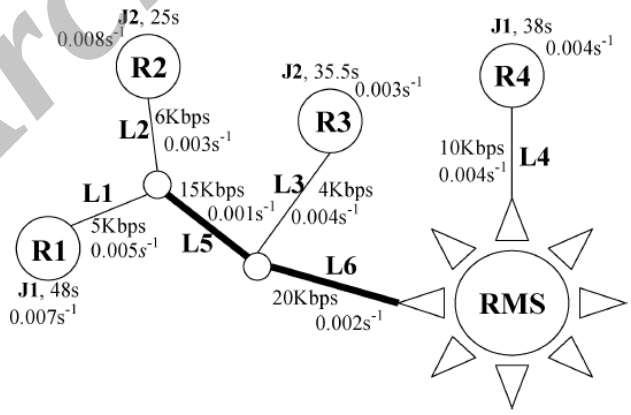

Figure1. A virtual tree structured grid service [5].

The tree structure models the common cause failures in shared communication channels. For example, in Fig. 1, the failure in channel L6 makes resources R1, R2, and R3 unavailable. This type of common cause failure was ignored by the conventional parallel computing models, and the above star-topology models. For small-area

communication, such as a LAN or a cluster, such assumption that ignores the common cause failures on communications is acceptable because the communication time is negligible compared to the processing time. However, for wide-area communication, such as the grid system, it is more likely to have failure on communication channels. Therefore, the communication time cannot be neglected. In many cases, the communication time may dominate the processing time due to the large amount of data transmitted. Therefore, the virtual tree structure is an adequate model representing the functioning of grid services [5].

3.3 Colored Petri Nets

Petri nets provides a more accurate modeling of complex asynchronous process. Each petri net use four elements, place, transition, arc, token for modeling. In colored petri nets, tokens can vary with each other and each token is added a feature called color. The mean of color is a special attribute that can be assigned to a token and distinguishes it from other. With use of this feature a new dimension was given to token and thus tokens CPNs combine the strengths of Petri nets with the facilities of high-level programming languages, which allow the definition of powerful data types and permit the manipulation of their values. The informal and formal definitions of CPNs are presented, as follows:

Colored Petri nets (CPNs or CP-nets) are a class of high-level nets that extend ordinary Petri nets. In CPNs, tokens can carry arbitrarily complex data, arcs can be annotated with input inscriptions influencing the enabling of a transition, or output inscriptions stating the production rule of tokens when a transition fires. Input/output inscriptions can be functions or variables [[18]18].

4. Calculating Task Execution and Reliability

4.1 Assumptions

All service requests from the users reach the RMS. Each service presumes performing a certain computational task. The RMS is able to divide the entire service task into subtasks. Resources with different characteristics are distributed in the grid, and are registered or automatically detected by the RMS. Some of the available resources can be assigned by the RMS to perform certain subtasks when the RMS gets a request for service [9]. In a grid service, the structure of a virtual network (consisting of RMS and resources involved in performing the service) can be represented by the tree topology.

1. The service request reaches the RMS and is served immediately, that is, the starting time point is defined as the RMS starts processing the request. The RMS divides the entire service task into a set of subtasks. The data dependence may exist among the subtasks. The order is determined by precedence constraints and controlled by the RMS.

2. Each available resource is able to process any single subtask. Different resources start performing their tasks immediately after they get the input data from the RMS through communication channels. The RMS can assign the same subtasks to several resources to provide fault-tolerance of the service.

3. Each resource has a given constant processing speed when it is available, and has a given constant failure rate. Each communication channel has a constant failure rate, and a constant bandwidth/speed of data transmission. This assumption is a first-order approximation which can be justified by the fact that the service time is usually much

shorter than the time during which the component remains in the same state, so the expected performance within the short intervals can be viewed as constants.

4. The total computational complexity of the entire task (number of operations) is equal to the sum of the computational complexities of subtasks. The subtask processing time is proportional to the subtask's computational complexity.

5. If a resource or communication channel failure occurs before the end of the output data transmission from the resource to the RMS, the subtask fails.

6. If a subtask is processed by several resources, it is completed when the first result is returned to the RMS. The task is completed when all of the subtasks are completed, and their results are returned to the RMS from the resources [4-6, 9].

4.2 Distribution of Task Completion Time

It is observed in many grid projects that the service time experienced by the users is a random variable. Finding the distribution of this variable is important for evaluating the grid performance and improving the RMS functioning. The service time is affected by many factors. First, various available resources usually have different task processing speeds online. Thus, the task execution time can vary depending on which resource is assigned to execute the task/subtasks. Second, some resources can fail when running the subtasks, so the execution time is also affected by the resource reliability. Similarly, the communication links in grid service can be disconnected during the data transmission. Thus, the communication reliability influences the service time as well as data transmission speed through the communication channels. Moreover, the service requested by a user may be delayed due to the queue of earlier requests submitted from others. Finally, the data dependence imposes constraints on the sequence of the subtasks' execution, which has significant influence on the service time [5].

A set of n non preemptive tasks are to be scheduled on m parallel dedicated machines with a regular criterion. Chain precedence constraints among the tasks, deterministic processing times and processing machine of each task are given. Computational complexity results and solution algorithms for some special cases are shown in [19]. When the precedence relations among the tasks are given by two chains, efficient solution algorithms are provided for the minimization of the weighted sum of task completion times and the number of tardy jobs.

Based on an interesting model for tasks with continuous processor, two natural assumptions for tasks are considered: the processing time of any task is non-increasing in the number of processors allotted, and the speedup is concave in the number of processors. Under these assumptions the work function of any malleable task is non-decreasing in the number of processors and is convex in the processing time [20].

The data dependence on task execution can be represented by m×m matrix H such that $h_{ki}=1$ if subtask i needs its execution output data from subtask k and $h_{ki}=0$ otherwise (the subtasks can always be numbered such that k < i for any $h_{ki}=1$). Therefore, if $h_{ki}=1$, then the execution of subtask i cannot begin before completion of subtask k. For any subtask i, one can define a set $\emptyset_i$ of its immediate predecessors $k \in \emptyset_i$ if $h_{ki}=1$.

The random total completion time for subtask j assigned to resource k is equal to sum of the subtask processing time T_{kj} and τ_{kj} the random time of communication between the RMS and the resource k that process subtask j.

$$Time_{kj} = T_{kj} + \tau_{kj} \quad (1)$$

τ_{kj} can take two values:

$$\tau_{kj} = \frac{a_j}{s_k} \quad (2)$$

Where a_j is the amount of data that should be transmitted for the subtask j (input data from the RMS to the resource, and output data from the resource to the RMS). If data transmission between the RMS and the resource k is accomplished through links belonging to a set γ_k , the data transmission speed is

$$S_k = \min_{y \in \Upsilon_k}(b_y) \quad (3)$$

Where b_y is the bandwidth of the link L_y [4] .

For a constant failure rate, the probability that resource k does not fail until the completion of subtask j can be obtained as

$$P_{kj} = \exp[-\lambda_k (T_{kj} + \frac{a_j}{s_k})] \quad (4)$$

These give the distribution of the random subtask processing time: $\Pr(T_{kj}) = P_{kj}$ and $\Pr(T_{kj} = \infty) = 1 - P_{kj}$.

For constant failure rate, the probability that the communication path between the RMS and the resource k does not fail until the completion of subtask j is

$$q_{kj} = \exp[-\pi_k (T_{kj} + \frac{a_j}{s_k})] \quad (5)$$

Where π_k is the failure rate of the communication path between the RMS and the resource k, which can be calculated by $\pi_k = \sum_{y \in \gamma_k} \lambda_y$ where λ_y is the failure rate of the y-th link. This gives the distribution of the random subtask data transmission time:

$$pr(\tau_{kj} = \frac{a_j}{s_k}) = q_{kj} \quad (6)$$

The random total completion time $\theta_{j,\{k\}}$, for subtask j assigned to resource k is equal to $T_{kj} + \tau_{kj}$. It can be easily seen in [1] that the distribution of this time is [11]:

$$pr(\theta_{j,\{k\}} = T_{kj} + \frac{a_j}{s_k}) = p_{kj} q_{kj} \quad (7)$$

Assume that each subtask i is assigned by the RMS to resources composing set ω_i . The RMS can initiate execution of any subtask j (that is, send to all the resources from ω_i) only after the completion of each subtask $k \in \phi_i$. Thus, the random time of the start of subtask i is

$$T_i = \max(V_k), k \in \phi_i \quad (8)$$

Where V_k is the random completion time for subtask k. If $\phi_i = 0$ that is, subtask i does not need data produced by any other subtask, then the subtask execution starts without delay $T_i = 0$. If $\phi_i \neq 0$ then Ti can have different realizations. Having the time Ti when the execution of subtask i starts and the time Tij of subtask i execution by resource j, one obtains the time for subtask i executed by resource j as [6]

$$V_{ij} = T_i + T_{ij} \quad (9)$$

Therefore, the conditional distribution of the random time V_{ij}, given that execution of subtask i starts at time τ_{il}, takes the following form:

$$pr(V_{ij} = \tau_{il} + t_{ij}) = p_{il}\, q_{il}\, p_{ij} q_{ij} \quad (10)$$

The random time of subtask i completion V_i is equal to the shortest time when one of the resources from ω_i completes the subtask execution [9]:

$$V_i = \min(V_{ij}), j \in \omega_i \quad (11)$$

4.3 Service Reliability

In applications where the execution time of each task (service) is of critical importance, the system reliability $R(\theta^*)$ is defined as a probability that the correct output is produced in time less than θ^*. This index can be obtained from [4-6, 9-11]

$$R(\theta^*) = \sum_{i=1}^{k} Q_i .1(\Theta_i < \theta^*) \quad (12)$$

4.4 Determining the PMF (Probability Mass Function) of the Task Execution Time

After the subtasks have be assigned to corresponding resources, it is easy to find all combinations of resources such that each combination contains exactly m resources executing m different subtasks that compose the entire task. Each combination determines exactly one MTST consisting of links that belong to paths from the m resources to the RMS.

Having the MTST (Minimal Task Spanning Tree), and the times of their elements involvement in performing different subtasks, one can determine the PMF of the entire service time.

First, we can obtain the conditional time of the entire task completion given only MTST S_i is available as [6]:

$$Y_{\{i\}} = \max_{1 \le j \le m}(y_{ij}) \text{ , } for\ any\ 1 \le i \le N \quad (13)$$

For a set of available MTST, the task completion time is equal to the minimal task completion times among the MTST [9].

$$Y_\psi = \min_{i \in \psi}(Y_{\{i\}}) = \min_{i \in \psi}[\max_{1 \le j \le m}(y_{ij})] \quad (14)$$

Now, we can sort the MTST in an increasing order of their conditional task completion times $Y_{\{i\}}$, and divide them into different groups containing MTST with identical conditional completion time. Suppose there are k such groups and any group contains MTST with identical conditional task completion times. Then it can be seen

that the probability can be obtained as $Q_i = \Pr(\mathrm{E}_i, \overline{\mathrm{E}}_{i-1}, \overline{\mathrm{E}}_{i-2}, ..., \overline{\mathrm{E}}_1)$ where E_i is the event when at least one of MTST from the group G_i is available, and $\overline{\mathrm{E}}_i$ is the event when none of MTST from the group G_i is available [4-6, 9].

The proposed model with petri net

In this section, first we present a general framework for execution a task on tree structure grid service with precedence using colored Petri net and then to verify aforementioned claim, we express an example that is used in previous research and calculate reliability for it.

For modeling execution a task, use CPN TOOLS version4.0. The petri net that is used in this study is colored petri net because for calculating the reliability need the elements that is transferred in net. Therefore a feature named color that presents the set of possible token values is assigned to it. Also, the proposed algorithm flowchart is illustrated in figure 2.

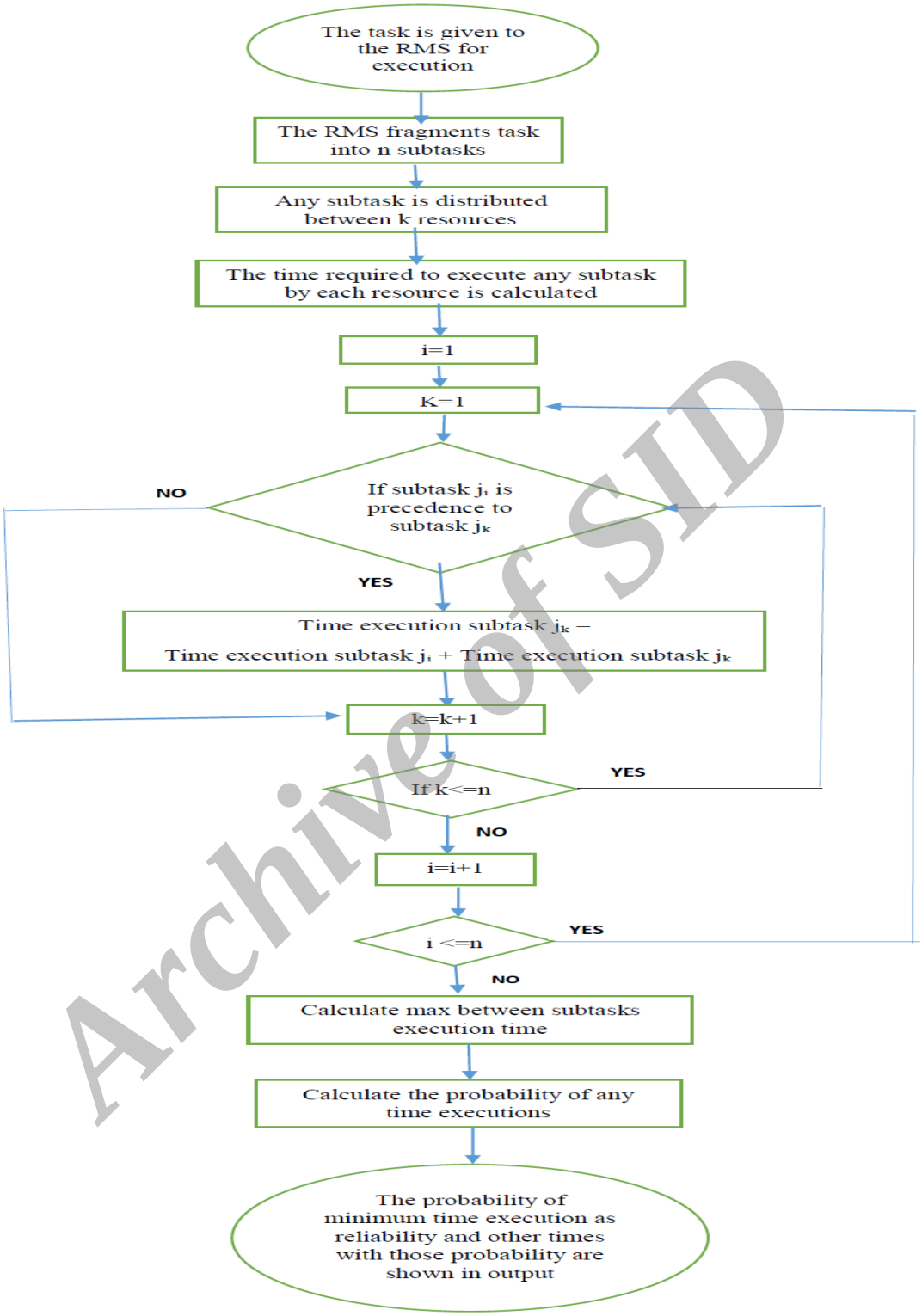

Figure 2. The proposed algorithm flowchart for resource scheduling

The tasks are given to RMS, so the iterative algorithm for scheduling n subtasks can be implemented by means of multi-threaded via distribution fashion.

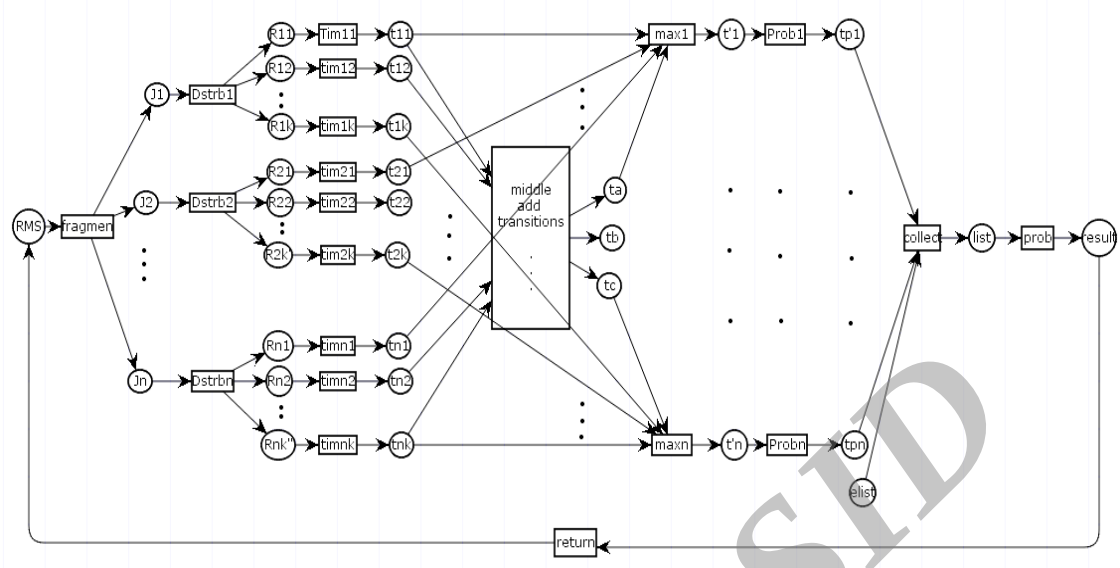

Figure 3. Proposed colored petri net model for tree structured grid service when nk resources are available.

As can be seen from Fig.3 the RMS is modeled by place RMS and contains one token that is our task. Tasks are available from RMS and RMS should divide them into subtasks. Transition fragmentation fires and divides task j into subtasks j1 to jn that is shown with places j1, j2,…, jn and one token is added to each of these places. Then several resources is allocated to each of these subtasks. According to assumptions, assign more than one resource to each subtask so that desired level of reliability is obtained. Transition distributei allocate resources Ri1, Ri2,…,Rik which is represented by places Ri1, Ri2,…, Rik to each subtask ji.

Since that in the tree topology there are several links between RMS and resources, the data transmission speed for each path between the resource and the RMS is limited by the lowest bandwidth of links belonging to the path. Thus we define function min(bw) that calculates minimum bandwidth of communication links.

Now we must calculate entire task execution time and its distribution. According to formula numbered 1, subtask execution time that is sum of the subtask processing time and time of communication between the RMS and the resource. This time is represented by transition TIME. By transition TIME, entire subtask execution time is transferred to next place.

Since we consider data dependence that some successor subtasks use the results from some predecessor subtasks, according to the data dependence that any problem assume, execution time of successor subtask is added with execution time of predecessor subtask and consider as execution time of successor subtask. Furthermore some transitions needed to calculate this addition. Since data precedence is variety in different problems, we can't show this step explicitly and name this middle transition. After this step, total execution time of any subtask by any resource corresponding to it considering the time needed for dependence is calculated.

Now we should find minimal task spanning tree. We should find all combinations of resources such that each combination contains exactly n resources executing n different

subtasks that compose the entire task. Each combination determines exactly one MTST. The entire task completion is depended on maximum of subtask completion time. Since there is no redundancy in any single MTST, the sequence of subtask execution by the resources from this MTST is determined by the data dependence/precedence Constraints. If subtask jk is prior to subtask jl, we should consider a middle transition that calculate subtask jl time with adding time of subtask jk with jl in any MTST. Transition add do this and is transferred this time into next place. Note that an MTST completes the entire task if all of its elements do not fail during the maximal time needed to complete subtasks in execution. Thus, when calculating the element reliability in a given MTST, one has to use the corresponding record with the maximum time. Thus max transitions calculate maximum execution time in any MTST considering time required for subtask priority. Task execution time is restricted to the slowest subtask's execution. Therefore for obtain optimized time, select minimum time between all calculated times from MTSTs and for simplify calculating probability in next step, collect all times ascending with transition collect. Now we should calculate time distribution. According to the above described, distribution of subtask execution time is depending on the probability that corresponding resource does not fail until the completion of subtask and the probability that the communication path between the RMS and the resource does not fail. Thus we define function sum() that calculates sum of link failures and use that in probability distribution. Therefore first element in this list is time for executing entire task and can calculate probability of it in transition prob and transfer time and its probability in place result.

Now should return results to the RMS to deliver it to user. Transition return does it.

This is a general form for modeling execution of task in tree structured grid services considering priority ordering among subtasks but we need to calculate reliability of this model. We present this with an example.

5. Case Study

5.1. Calculating Reliability Using Mentioned Algorithms

Consider the virtual tree structure of a grid service presented in Fig. 4(a) and assume that the service task is partitioned into three subtasks: J1 (assigned to resources R1and R2), J2 (assigned to resources R3 and R4), and J3 (assigned to R5). J1, J2, and J3 require transmission of 150, 90, and 70 Kbits of data, respectively, between the RMS and the corresponding resources.

The subtask processing time of the resources, bandwidths, and failure rates of the links are shown in Fig. 4(a), near the elements. Subtasks 1 and 2 get the input data directly from the RMS. Subtask 3 needs the output of subtask 2. The service task is completed when the RMS gets the outputs of both subtasks 1 and 3: $\varphi_1 = \varphi_2 = 0$, $\varphi_3 = \{2\}$, $\varphi_4 = \{1,3\}$.These subtask precedence constraints can be represented by the directed graph in Fig.4(b).

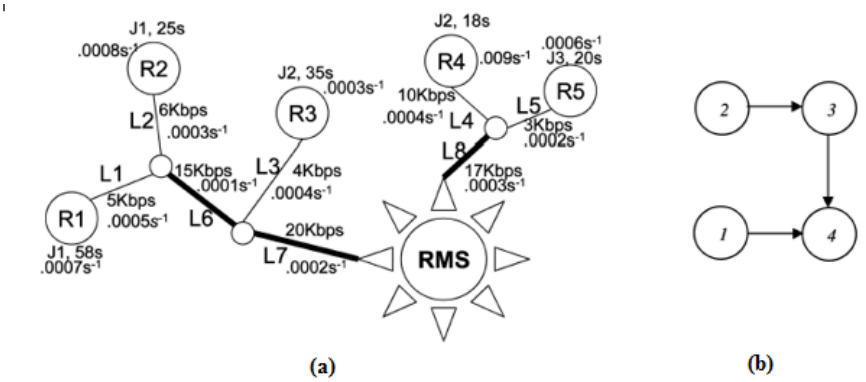

Fig4. (a) an example of tree structured grid service and (b) its precedence.

Considering four MTSTs:
S_1={R1,R3,R5,L1,L3,L5,L6,L7,L8}, S_2={R1,R4,R5,L1,L4,L5,L6,L7,L8}, S_3={R2,R3,R5,L2,L3,L5,L6,L7,L8}, S4={R2,R4,R5,L2,L4,L5,L6,L7,L8},

We can obtain the data transmission speed for each path between the resource and the RMS and calculate the data transmission time and the time of subtasks' completion. At this step of calculating parameters, we do not consider the data dependence, which will be complemented later. These parameters are presented in Table 1.

Table1. Parameters of MTSTS' path

Elements, subtasks	R1,J1	R2,J2	R3,J3	R4,J4	R5,J3
Data transmission speed (Kbps)	5	6	4	10	3
Data transmission time (s)	30	25	22.5	9	23.3
Processing time (s)	58	25	35	18	20
Time to subtask completion (s)	88	50	57.5	27	43.3

In accordance with the subtask precedence constraints, we obtain the subtask completion time for any MTST. The entire task and subtask completion times for different MTSTs are presented in Table 2.

Table2. The entire task and subtask execution times.

	J1	J2	J3	Entire task
S1	88	57.5	100.8	100.8
S2	88	27	70.3	88
S3	50	57.5	100.8	100.8
S4	50	27	70.3	70.3

The conditional times of the entire task completion by different MTST are Y1=100.8 , Y2=88, Y3=100.8, Y4=70.3.

Therefore, the MTST compose three groups and obtain the PMF of service time presented in Table 3.

Table3. PMF of execution times

Θ_i	Q_i
70.3	0.6695
88	0.0336
100.8	0.1884

Calculating the reliability: From Table 3 we obtain the probability that the service time is not greater than a pre-specified value of $\theta^* = 90$ seconds as

$$R(\theta^*) = \sum_{i=1}^{3} Q_i .1(\Theta_i < \theta^*) = 0.6695 + 0.0336 = 0.7032$$

5.2 Reliability Modeling of Above Example with Colored Petri Net

As can be seen in Fig.4 an initial step is similar to the proposed model in Fig.3. According to precedence constraint between subtasks, subtask j3 need the output of subtask j2. Therefore in any MTST the time of executing subtask j3 should add with time of executing subtask j2 and consider for subtask j3 execution time. This is done with transition *add, add1* .Therefore place *t3, t3'* represent execution time for subtask j3. Diagnosis of MTST is important. For example transition *max1* calculates maximum time between execution time of subtask j1 performed by resource R1 and execution time of subtask j2 performed by resource R3 and execution time of subtask j3 (which described above) that is task execution time in this term and the time of any MTST is transferred into next places. Thus places *tt1, tt2, tt3, tt4* represent all times that task can be execute. Maximum time of execution those subtasks (*t*), *P1* the probability that resource R1 doesn't fail during execution j1 and probability that communication path between the RMS and the resource R1 does not fail (*q1*), the probability that resource R3 doesn't fail during execution j2 (*P2*) and probability that communication path between the RMS and the resource R2 does not fail (*q2*) are parameters that in the form of a record, can be transferred from transition *prob1* to place *tp1* when *max1* fired. For achieving efficient execution time of entire task should calculate minimum time between these times. Namely, to achieve efficient execution of task, we select the best alternative between variety scheduling in terms of resource consumptions and subtask's execution time pertained to especial task. Hence integrate records to a list and sort it ascending. Therefore first element in the list that is minimum existing time for completing task is task execution time and can use other parameters (probabilities) to calculate distribution of this time and reliability. So that consider an empty list presented with place *elist* and define *addtolist* function to achieve a list with records with five parameters *(t,p1,q1,p2,q2)*. After firing transition *collect* one token will be added to place *list* and such list will be provided. By transition *prob* probabilities of execution task in each of task execution times are calculated and in form of a list containing records with parameters *(t,p)* is transferred to place *result*. Finally a list containing all possible task execution times and corresponding probabilities will be achieved. We use a write-in-file monitor in place *result*. A monitor is a mechanism in CPN Tools that is used to observe, inspect, control, or modify a simulation of a CP-net.

Many different monitors can be defined for a given net. Write-in-file monitors are one kind of monitors that are used to update files during simulations.

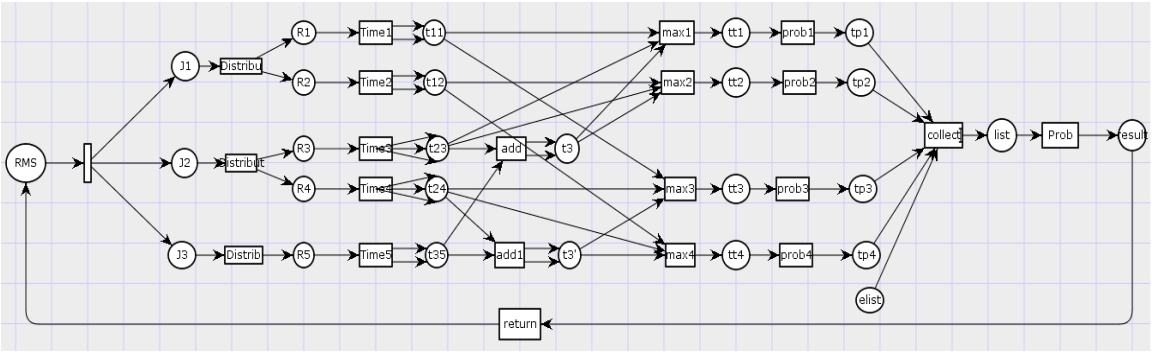

Figure 5. Colored petri net for modelling mentioned example

According to description above, reliability in grid computing is the probability that the correct output is produced in time less than certain time. We use a text file containing a list from three integers as times which we have an intention to calculate reliability in those. Write-in-file monitor will read this file as an input then presents output in text file named result. Minimum time needed for execution task and its probability, maximum time needed for execution task and its probability and reliability in each of times that reads from text input file. For example output file when input file is containing list [60, 90, 120] is as the form below:

Task is completed in 70.3 seconds with probability 0.6695
and completed in 100.8 seconds with probability 0.1884.
The Reliability or probability that output is produced in time less than 60 is equal to 0.0.
The Reliability or probability that output is produced in time less than 90 is equal to 0.7032.
The Reliability or probability that output is produced in time less than 120 is equal to 0.8916.

Figure 6. Output file content in mentioned example

As can be seen results such that obtained from proposed petri net modelling are same as results in previous section with formal algorithm.

5.3 Evaluating Reliability

For evaluating reliability, use proposed model for a case study .In our case study, 11 databases are involved. Some databases have some data overlapped partially or totally. The subtasks are partitioned according to the different portions of the data that they work on. One identical subtask may be executed on different databases, which means that those databases have common data that are being processed by the programs in a redundant manner. Multiple subtasks may also be assigned on the same data server to operate on different portions of the data. For example, here, J1 and J2 are both assigned to data server R1, J4 and J5 to R4, and J5, J6, and J8 to R6. We adopt the sequential execution of different subtasks on the same resource. The virtual tree structure is depicted in Fig. 5 and some precedence constraints are imposed on the subtasks, as depicted in Fig. 7.

This is a project named Biomedical Association and Pathways (BioMap) at Indiana University and Purdue University based on a grid computing system. BioMap is an

intelligent biomedical literature mining system that provides biologists with new hypotheses that may lead to a new discovery.

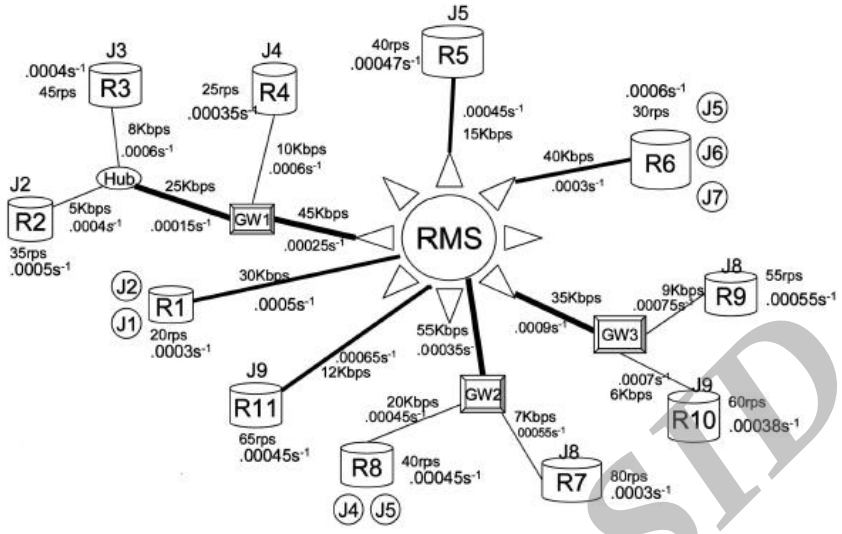

Figure7.Tthe tree structure of case study

Nine different subtasks, J1; J2; . . . ; J9, for mining the databases need to be completed in this BioMap grid service. The assignment of the nine subtasks on the 11 resources (data servers) is also depicted in Fig.8. Other parameters about the subtasks are given in Table 4, where "# of records" means the number of records for each subtask to complete (that is, the amount of work), and Kbits mean the amount of data transmitted between each subtask with the RMS.

Table4. Parameters of case study

subtask	J1	J2	J3	J4	J5	J6	J7	J8	J9
# of records	700	800	1000	500	1200	650	350	600	900
Kbits	70	80	75	50	95	65	35	60	90

Three different cases of data dependence on the nine subtasks are analyzed. The corresponding precedence constraints are presented by the directed acyclic graphs in Fig.8 , where J10 is the pseudosubtask as the final step on the RMS.

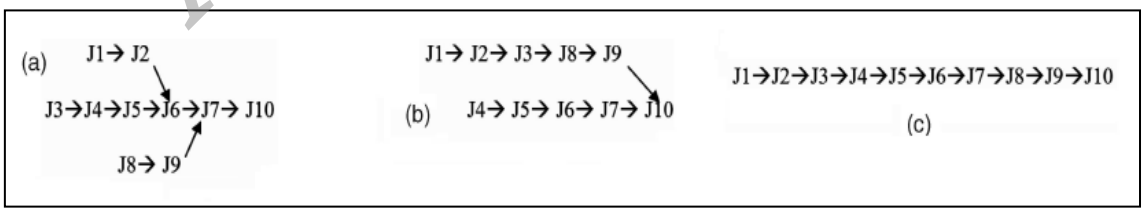

Figure 8. Precedence constraints on subtasks.

We use proposed model for each case A to C. our supposed middle transition is long therefore we irrespective mentioning those models. The output file of those are as below:

Case A:

Task is completed in 117 seconds with probability 0.6832
and completed in 134 seconds with probability 0.0098.
The Reliability or probability that output is produced in time less than 116 is equal to 0.0.
The Reliability or probability that output is produced in time less than 120 is equal to 0.7201.
The Reliability or probability that output is produced in time less than 134 is equal to 0.7774.

Case B:

Task is completed in 145 seconds with probability 0.5068
and completed in 159 seconds with probability 0.0107.
The Reliability or probability that output is produced in time less than 146 is equal to 0.0.
The Reliability or probability that output is produced in time less than 150 is equal to 0.6213.
The Reliability or probability that output is produced in time less than 159 is equal to 0.6929.

Case C:

Task is completed in 230 seconds with probability 0.2951
and completed in 262 seconds with probability 0.0594.
The Reliability or probability that output is produced in time less than 229 is equal to 0.0.
The Reliability or probability that output is produced in time less than 250 is equal to 0.4432.
The Reliability or probability that output is produced in time less than 262 is equal to 0.5205.

We show this values in Fig.9.

It can be seen that different subtask precedence constraints cause different service time and reliabilities. Less data dependence will lead to higher performance and higher reliability. Therefore, reduction of the data dependence in the scheduling/division of tasks by the RMS is one of the possible ways to improve the service performance and reliability.

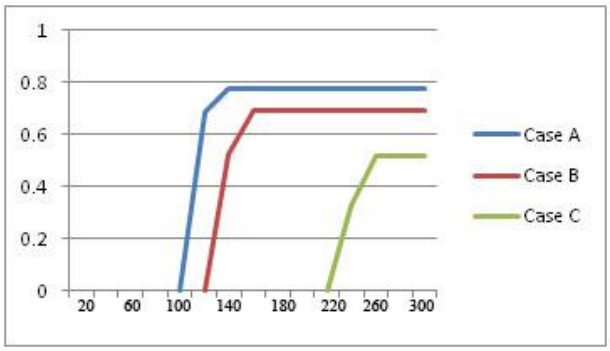

Figure 9.Service reliabilities for cases A to C.

6. Conclusions and future works

The basic goal in grid computing is use of processing power of computers in net for solving complicated time consuming problems. For achieving this distribution subtasks to resources and scheduling them have important role. This scheduling is done with respect Quality of Service (QoS). Reliability is one of important QoS measures in grid services. To provide the desired level of service reliability, the RMS can assign the same subtasks to several independent resources. We use tree structure of grid computing in this paper that is more reasonable than star structure which is used in formal researches because failure in communication links and communication time cannot be ignored. Moreover, the communication links in grid service can be disconnected during the data transmission, so the communication reliability influences the service time, as well as the data transmission speed. Finally, the data dependence imposes constraints on the sequence of the subtasks' execution, which has been observed to have significant influence on the service time.

The RMS is able to divide the entire service task into subtasks. Some of the available resources can be assigned by the RMS to perform certain subtasks. In this paper task fragmentation and distribution subtasks among resources for calculating reliability in tree structured grid service considering priority in sequence of subtasks is modeled by colored petri net and for modelling CPNTOOLS version 4.0 is used. Then the time of execution task and its distribution in order to calculating reliability is obtained. Finally for evaluating reliability a case study with 11 subtasks and 9 resources with three conditions in precedence is mentioned and found that less data dependence will lead to higher performance and higher reliability.

There are different task scheduling methods in grid computing in order to reliability evaluation in prior researches, but modeling task scheduling and distribution together with calculating reliability notice that precedence in subtasks, have been less attention. In addition the topology using in this paper is tree that rather than star topology which is used in prior researches is more reasonable and adaptable with nature world.

Despite of current approach's merits, it has demerits such as:

1- Using some assumptions may digress model from nature network.
2- If number of nodes are increased, with the same ratio network is greater and net modelling is complicated.

Future works

1- Reliability evaluation for other grid topologies.
2- Performance measuring for tree structured grid services as one of important measures of QoS which is interact with reliability.
3- Reliability calculating in tree structured grid services with use of queue structure.
4- Using other kind of petri net such as hierarchical petri net and fuzzy petri net for evaluating reliability of grid services.

References

[1] I. Foster and C.Kesselman, The Grid 2: Blueprint for a New Computing Infrastructure. : Morgan-Kaufmann, 2003.
[2] T. Tannenbaum, D. Wright, K. Miller, M. Livny, Condor—a distributed job scheduler, in: T. Sterling (Ed.), Beowulf Cluster Computing with Linux, MIT Press, 2001. http://www.cs.wisc.edu/condor/ [20 August 2015].
[3] T. Kosar, M. Livny, Stork: Making data placement a first class citizen in the grid, in: Proceedings of the 24th International Conference on Distributed Computing Systems, ICDCS2004, Tokyo, Japan, March 2004, pp. 342–349.
[4] Yuan-S un Dai, Gregory Levitin, "Reliability and Performance of Tree-Structured Grid Services", IEEE TRANSACTIONS ON RELIABILITY, VOL. 55, NO. 2, JUNE 2006, 337-349.
[5] Kuan-ching Li, ching-Hsien Hsu, Laurence Tianruo Yang, Jack Dongarra, Hans Zima, Reliability and Performance Models for Grid Computing, Handbook of Research on Scalable Computing Technologies,2009,219-245.
[6] Yuan-S un Dai, Gregory Levitin, Kishor S. Trivedi, " Performance and Reliability of Tree-Structured Grid Services Considering Data Dependence and Failure Correlation" , IEEE TRANSACTIONS ON COMPUTERS, VOL. 56, NO. 7, JULY 2007.
[7] Joanna Kołodziej, Samee Ullah Khan, Lizhe Wang, Marek Kisiel-Dorohinicki , Sajjad A. Madani, Ewa Niewiadomska-Szynkiewicz , Albert Y. Zomaya, Cheng-Zhong Xu, "Security, energy, and performance-aware resource allocation mechanisms for computational grids", Future Generation Computer Systems 31 (2014) 77–92.
[8] Mohammad Abdollahi Azgomi, Reza Entezari-Maleki, " Task scheduling modelling and reliability evaluation of grid services using coloured Petri nets", Future Generation Computer Systems 26 (2010) 1141_1150.
[9] Gregory Levitin, Yuan-Shun Dai, Hanoch Ben-Haim, "Reliability and Performance of Star Topology Grid Service With Precedence Constraints on Subtask Execution", IEEE TRANSACTIONS ON RELIABILITY, VOL. 55, NO. 3, SEPTEMBER 2006.
[10] Yuan-Shun Dai, Gregory Levitin, "Optimal Resource Allocation for Maximizing Performance and Reliability in Tree-Structured Grid Services" IEEE TRANSACTIONS ON RELIABILITY, VOL. 56, NO. 3, SEPTEMBER 2007.
[11] Shaoqiang Zhang, Chunyong Ding, Gang Hou, "Resource and Bandwidth Allocation on a Computational Grid with Tree Topology" , Proceedings of the Fifth International Conference on Grid and Cooperative Computing (GCC'06), 2006.
[12] Javad Akbari Torkestani, "A highly reliable and parallelizable data distribution scheme for data grids", Future Generation Computer Systems 29 (2013) 509–519.
[13] T. Manghui, L. Peng, I. L. Yen, B. Thuraisingham, and L. Khan, "Secure Data Objects Replication in Data Grid," Dependable and Secure Computing, IEEE Transactions on, vol. 7, pp. 50-64, 2010.
[14] K. Ranganathan and I. Foster, "Identifying Dynamic Replication Strategies for a High-Performance Data Grid," in Grid Computing — GRID 2001. vol. 2242, C. Lee, Ed., ed: Springer Berlin Heidelberg, 2001, pp. 75-86.
[15] D.-w. Chen, S.-t. Zhou, X.-y. Ren, and Q. Kong, "Method for replica creation in data grids based on complex networks," The Journal of China Universities of Posts and Telecommunications, vol. 17, pp. 110-115, 8// 2010.
[16] preve, NP. (ED), " Grid Computing Towards a Global Interconnected Infrastructure", 2011.

[17] Yue Chen, Ying Li, Zhengxian Gong, Qiaoming Zhu, " A Framework of a Tree-based Grid Information Service", Proceedings of the 2005 IEEE International Conference on Services Computing (SCC'05), 2005.
[18] K. Jensen, L.M. Kristensen, L. Wells, Coloured Petri nets and CPN tools for modeling and validation of concurrent systems, International Journal on Software Tools for Technology Transfer (STTT) (2007) 213_254.
[19] Alessandro Agnetis , Hans Kellerer , Gaia Nicosia , Andrea Pacifici, "Parallel dedicated machines scheduling with chain precedence constraints", European Journal of Operational Research 221 (2012) 296–305.
[20] Klaus Jansen , Hu Zhang ," Scheduling malleable tasks with precedence constraints", Journal of Computer and System Sciences 78 (2012) 245–259.
[21] http://www.ip-grid.org/ [6 March 2015].
[22] http://www.teragrid.org/ [10 April 2015].