Grid Service Reliability Modeling Considering Fault Recovery

Pallavi Rahinj, S. M. Sabale

Abstract: Grid computing is a recently developed technology. Although the developmental tools and techniques for the grid have been extensively studied, yet some important issues, e.g., grid service reliability and task scheduling in the grid, have not been sufficiently studied. For some grid services which have large subtasks requiring time-consuming computation, the reliability of grid service could be rather low. To resolve this problem, this paper introduces Local Node Fault Recovery (LNFR) mechanism into grid systems, and presents an in-depth study on grid service reliability modeling and analysis with this kind of fault recovery. To make LNFR mechanism practical, some constraints, i.e. the life times of subtasks, and the numbers of recoveries performed in grid nodes, are introduced; and grid service reliability models under these practical constraints are developed. Also uses new algorithm which is based on min-min algorithm for task scheduling.

Index Terms: Grid service reliability, Fault recovery, Task scheduling, Resource management system, Star topology, Local node fault recovery, Remote fault recovery.

1. Introduction

Grid computing [1] is a newly developed technology for complex systems with large-scale resource sharing, wide-area communication, multi-institutional collaboration, etc. [2]. The real and specific problem that underlies the Grid concept is coordinated resource sharing and problem solving in dynamic multi-institutional virtual organizations [1]. This is required by a range of collaborative problem solving and resource-brokering strategies. This sharing is highly controlled by resource management system (RMS) [5], with resource providers and consumers defining what are shared, who is allowed to share, and the conditions under which the sharing occurs. Unlike the traditional file exchange, as supported by the Web or peer-to-peer systems, users in the grid can access the required resource or service in a transparent way as if they were to use local resources or services. However, it gives rise to an incompatible conflict between grid users and resource providers in usage policy of the local resources. For users, in addition to simplicity and easiness, to get desirable service functionalities, some quality of service (QoS) targets associated with the service, such as grid service reliability [3], the financial cost of the resource, and the efficiency of grid service, may be specified when a service is submitted. On the other hand, resource providers receive the compensation from grid users for the consumed resources at the price of sacrificing local task executions [3]. Meanwhile, resource providers may not participate in the grid unconditionally, and they may specify different policies that govern how the resources should be used by the grid such that the resources could still meet the local resource demands [4], [5]. On behalf of grid users with multiple dimensional QoS requirements, multi-objective task scheduling with a set of resource constraints should be solved to obtain satisfied scheduling decisions. As one of the most important aspects of *quality of service* (QoS), *grid service reliability* can be defined as the probability of all of the subtasks involved in the considered service to be executed successfully [5], [4], [11]. From the point of view of grid service, it does not matter what the sources of failures are; what matters is whether the end results can return to grid RMS or not. Nevertheless, with the dramatic increasing of grid size and complexity, the grid system is much more prone to errors and failures than ever before. Moreover, the likelihood of errors occurring may be made worse by the fact that many grid services will perform long tasks that may require several days of computation [8]. Recently, much effort in fault avoidance and fault removal has been invested so as to improve grid service reliability. Recently, the modeling and analysis of grid service reliability has attracted lots of attention. Dai *et al.* [3] presented a virtual approach to modeling grid services, and derived the grid service reliability using the graphic theory. Dai *et al.* [6], and Levitin and Dai [5] studied grid service reliability for grid systems with star topology, and tree topology, respectively. Dai *et al.* [7] presented a hierarchical model from the mapping of the physical architecture, and the logical architecture in grid systems for grid service reliability analysis. Levitin *et al.* [4] studied grid service reliability taking the precedence constraints on programs execution into account. Dai and Wang [13] studied optimal resource allocation for maximizing service reliability using a genetic algorithm. Dai and Levitin [13] suggested an algorithm to study optimal resource allocation for maximizing performance while considering the service reliability factor in tree-structured grid systems. However, those works did not incorporate fault recovery, and did not investigate the influence of practical constraints of grid resources. Paul and Jie [8] developed an approach to fault tolerance based on job replication in grid systems. Affaan and Ansari [9] introduced a backup mechanism to achieve fault tolerance in grid systems. Jin *et al.* [10] put forward a fault tolerance

- *Miss. Pallavi Rahinj Computer Engineering Department completed M.Tech from Dr. B.A.T. University Lonere, Raigad. MH, India. Email- pallavirahinj@gmail.com*
- *Prof S.M. Sabale, Computer Engineering Department Dr. B.A.T. University Lonere, Raigad.*

mechanism in grid systems based on Java threads state capturing, and Mobile Agents. The basic approach proposed in the above researches on fault recovery in grid systems is a *Remote Node Fault Recovery* (RNFR) mechanism; i.e., when a failure occurs on a node, the state information can be migrated to another node, and the failed subtask execution is resumed from the interrupted point, or the failed subtask can be dynamically rescheduled on another node, and the node restarts the subtask from the beginning. It is very useful and effective for RNFR to recover grid tasks from failures. However, some complex tasks may require several days of computation. For those tasks, it will take a lot of time for RNFR on the transmission of state information. Furthermore, in a worst-case scenario, much time has been spent in local node execution when the execution is terminated by a failure, which brings great waste of consumed time and resource. In this case, another possible fault recovery mechanism referred as *Local Node Fault Recovery* (LNFR) could be more practical than RNFR to resume the subtask execution on the failed node once the node is recovered. Heddaya and Helal [11] studied the effect of LNFR on the reliability of distributed systems, [13] Suchang Guo and Wang studied effect of LNFR on Grid Service Reliability and Task Scheduling Considering fault recovery. LNFR offers an opportunity to resume execution from failure, and saves the migration expense compared with RNFR. Moreover, because fault recovery modules are located at grid resources, resource providers can set customizable constraints on fault recovery, which makes it easy to achieve distributed management of fault tolerance.

2 Reliability Analysis Considering Fault Recovery

2.1 RMS and Star Topology of Grid Service

Open Grid Services Architecture [1], [2] has enabled the integration of services and resources across distributed heterogeneous dynamic virtual organizations i.e. service-oriented grid. and the interaction between users and the grid is just service request and response. When a service request arrives at the RMS, a corresponding service is initiated to execute a certain task under the control of the RMS. Generally, the RMS divides the task into a set of *subtasks* so as to improve the efficiency of task execution [5]. Once the RMS determines which set of resources to use, the subtasks are assigned to the corresponding resources held on certain nodes, and are executed in parallel. When the nodes finish the assigned subtasks, they return the results to the RMS, and the RMS then integrates the received results into an entire task output and presents it to the user.

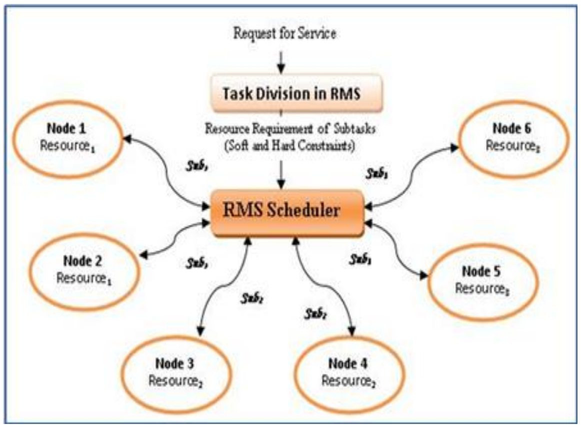

Fig.1: RMS with Star Topology

The RMS does not have complete control over all the resources in grid systems. Even though all online nodes, or resources, are linked through communication links with one another, only a small portion of nodes or resources available for a specific grid service is discovered by the RMS. At the same time, through systems selection, the RMS normally selects more than one resource from the discovered resources to assign a subtask to, so that the grid service reliability can be improved. In the case that there is only one RMS in the grid system, it can approximately regard the RMS and the selected resources as a star topology [2], [4]. In the following analysis, the assumptions of grid systems are as follows:

a) The RMS is perfect during the processing of the grid service, i.e., the RMS never fails.
b) When a service request arrives at the RMS, the RMS responds to it immediately; when a subtask is assigned to anode, the node executes the subtask immediately.
c) There is no precedence constraint on the order of execution of subtasks.
d) Each node can execute only one subtask.
e) The failure processes of nodes and communication links can be modeled by Poisson processes, respectively.
f) The failures in different elements (nodes or communication links) are independent.

3 Fault Recovery In Grid System

Grid system is a complex system which spans multiple heterogeneous and disjoined organizations [6]. In Grid system failures may arise from software bugs, human operator errors, performance overload, severe congestion, or electronic component failures. Additionally, environmental disasters may shut down portions of grid systems. Generally, when the node is executing a subtask, if a grid node failure occurs, hardware failure or software failure, the output of the subtask will be incorrect or no output will be send to the RMS at all. Similarly, if a communication link failure occurs when it is transferring data, the received information will be unexpected. As an important approach of fault tolerance, LNFR Means No migration when a failure occurs on a node, the state information can be migrated to another node, and the failed subtask execution is resumed from the interrupted point, or the failed subtask can be dynamically rescheduled on same node and the node restarts the subtask from the beginning [11], [13]. LNFR can achieve fault tolerance by recovering from failures, i.e., failed components are repaired or replaced; and once they become operational, the interrupted execution may be resumed by recovery actions. It can afford an opportunity for failed nodes to resume executing from failure. In particular, for some subtasks requiring long-term execution, LNFR can save execution time and resources. However, not all failures that occur in grid systems can be recovered. According to the recoverability, we can classify the failures occurring in grid systems into following categories [13]:

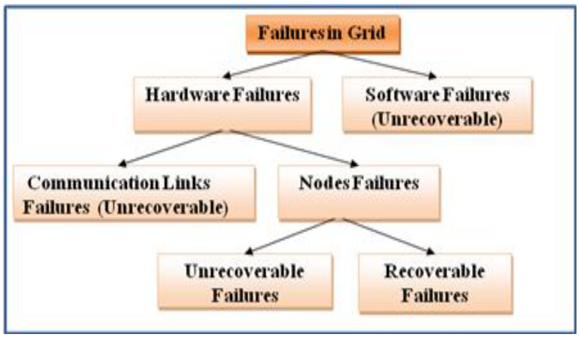

Fig.2: classification of failures in grid system

3.1 Reliability Modeling and Analysis with Fault Recovery

Denoted by *m* the number of divided subtasks by the RMS after the RMS receives a service S. Assuming that subtask is *i* assigned to node *k*, the required processing time of subtask *i* on node *k*, ξ_{ik} , is [4, 13]. $\xi_{ik} = C_i / S_k$ (1) Denoted by N_{ik} the total number of recoverable failures occurring during the execution of subtask *i* on node *k*. N_{ik} is a random variable. If $N_{ik} = n$ ($n \geq 1$) , then denoted by $TE^{(j)}_{ik}$ = (j =1,2,L,n, n+1) and $TR^{(j)}_{ik}$ = (j =1,2,L,n) the executing times, and recovering times in the execution process of subtask *i* on node *k*, respectively [13]. The *lifetime* of subtask *i* executed on node *k*, T_{ik}, is:

$$T_{ik} = TE_{ik} + TR_{ik} \quad (2)$$

If subtask *i* is successfully completed on node *k*, then its lifetime is:

$$T_{ik} = \xi_{ik} + TR_{ik} \quad (3)$$

From assumption (f), the failure occurrence process of the grid node and communication links can be modeled by Poisson processes. This assumption can be justified by the operational phase in which the software and hardware are not changed, so that the failure intensities are constant values. Because only hardware failures on nodes may be recoverable, $TE_{ik}^{(j)}$, and $TR_{ik}^{(j)}$ are actually the executing times, and recovering times of *hardware* failures on node k, respectively. According to assumptions (f) and (g), $TE_{ik}^{(j)}$ ($j = 1, 2, L,$ are –independent and identically distributed (i.i.d.) random variables, each following an exponential distribution with parameter λ_k^h [13]. It is also reasonable to make the following additional assumptions:

g) $TR_{ik}^{(j)}$ ($j = 1, 2, L,$ are i.i.d. random variables, each following exponential distribution with parameter μ_k (μ_k is often referred to as *recovery rate*).

h) $TR_{ik}^{(j)}$ are independent with $TE_{ik}^{(j)}$.

If the j[th] failure is unrecoverable, i.e. $TE_{ik}^{(l)}$=$TR_{ik}^{(l)} = 0 \; for \; all \; l > j$.

Firstly, the hardware reliability of grid nodes is modeled. The reliability of node executing subtask, considering only hardware failures is,

$$P_{ik}^h = P_1 + P_2$$

$$= Pr\,\{no\ failures\ occurs\}$$

$$+ Pr\,\{all\ the\ failures\ occuring\ are\ recoverable\} \quad (4)$$

P_1 Can be easily obtained as,

$$P_1 = \exp\left(-\lambda_k^h \xi_{ik}\right) \quad (5)$$

And P_2 can be obtained as,

$$P_2 = \sum_{n=1}^{\infty} Pr\{nrecoverablefailuresoccure, subtaskcompleted\}$$

$$= \sum_{n=1}^{\infty} \Pr\{E^n\} \quad (6)$$

Where E^n the event that subtask is i is successfully completed on node k, and n ($n \geq 1$) recoverable failures have occurred during the subtask execution. According to the execution process of subtask i on node k, we obtain

$$\Pr\{E^n\} = \frac{\left(x_k \lambda_k^h \tau_{ik}\right)^n}{n!} \exp\left(-\lambda_k^h \tau_{ik}\right) \quad (7)$$

The result obtained in (5) is the special case of (7) for which $n = 0$. Therefore, substitute (7) into (6) and (4), the reliability of node k executing subtask i, considering only hardware failures, is obtained as

$$P_{ik}^h = \sum_{n=0}^{\infty} \frac{\left(x_k \lambda_k^h \xi_{ik}\right)^n}{n!} \exp\left(-\lambda_k^h \xi_{ik}\right)$$

$$= \exp\left[(1 - x_k)\, \lambda_k^h \xi_{ik}\right] \quad (8)$$

Besides hardware failures, software failures and communication link failures may also occur on grid nodes, which are unrecoverable failures. The reliability of node k executing subtask i, considering only software failures, is

$$P_{ik}^{s} = \exp\left(-\lambda_k^h \xi_{ik}\right) \quad (9)$$

The reliability of communication links can be modeled by

$$P_{ik}^{l} = \exp\left(-\frac{\varepsilon_k l_{ik}}{y_k}\right) \quad (10)$$

Based on the above analyses, the probability that subtask i can be successfully completed on node k is

$$R_{ik} = P_{ik}^{h} P_{ik}^{s} P_{ik}^{l}$$

$$= \exp\left[-(1-x_k)\lambda_k^h \xi_{ik} - \lambda_k^h \xi_{ik} - \frac{\varepsilon_k l_{ik}}{y_k}\right] \quad (11)$$

To improve grid service reliability, a subtask is normally assigned to several nodes for parallel execution [3, 5]. Whenever a node on which a subtask is being executed returns the output to the RMS, the subtask i is considered to be completed. Denoted by $D(i)$ the node set to which subtask is assigned, the reliability of subtask, which is often referred to as *grid program reliability* [13], is

$$R(sub_i) = \Pr\{at least one node\ in\ D(i) can\ complete\ subtask\ i\}$$

$$= 1 - \prod_{k\in D(i)}(1 - R_{ik}) \quad (12)$$

When the RMS receives all the outcomes of subtasks, the grid service is considered to be completed successfully. Therefore, the *grid service reliability* is,

$$R_s = \Pr\{\ all\ subtasks\ can\ be\ completed\ successfully\}$$

$$= \prod_{i=1}^{m} R\ (sub_i)$$

$$= \sum_{i=1}^{m}\{1 - \prod_{k\in D(i)}\{1 - \exp[-(1-x_k)\lambda_k^h \xi_{ik} - \lambda_k^h \xi_{ik}\, \varepsilon_k l_{ik}]\}\} (13)$$

Given the processing times and failure intensity of grid nodes and communication links, the grid service reliability can be easily obtained.

4 Grid Service Reliability Modeling With Practical Constraints

Although the fault recovery mechanism provides an efficient way to reduce the influence of failures, some disadvantages can also be brought forward. With the introduction of fault recovery, the life time of subtasks in grid nodes is extended, especially when the mean recovery time is rather long on some nodes [13]. In grid, the service time is very critical to users because it will influence the money users are charged when the grid goes commercial. On the other hand, the resource providers in the grid may not be willing to spend a long time performing one subtask. At the same time, the process of fault recovery requires a large amount of state information so as to enable the node to execute the subtask continuously. In some particular situations, failures may occur frequently, and then be recovered again, which will impose a great burden on grid nodes, and have a strong influence on the availability of the node as well. Therefore, it is advisable to take some measures to limit the life time of any subtask, as well as the number of recoveries performed.

4.1 Constraints On The Lifetime Of Subtasks

To prevent the lifetime of a subtask from exceeding an allowed time limit, we can set a deadline for subtask i execution. Once the lifetime of subtask executed on node k, T_{ik} exceeds this deadline, denoted by T_{ik}^{*}, the node will claim failure of the subtask to the RMS [13]. The lifetime T_{ik}, if the subtask is successfully completed, is given by (3). Because the required execution time ξ_{ik} is a constant, T_{ik} mainly depends on the total recovering time TR_{ik}. From assumption (h), if $N_{ik} = n\ (n \geq 1)$, then TR_{ik} follows an Erlang distribution with parameters (n, μ_k), whose cumulative distribution function (c.d.f.) is given by

$$F_{TR_{ik}}(t) \equiv \Pr\{TR_{ik} \leq t\} = \prod_{j=n}^{\infty} \frac{(\mu_k t)^j}{j!} e^{-\mu_k t},\ \ t \geq 0 \quad (14)$$

Under the constraint on deadline T_{ik}^{*}, the reliability of node k executing subtask i, considering only hardware failures, is

$$P_{ik}^{(1)} = P_1 + P_2 = \Pr\{\ no\ failures\ occurures\} + \Pr\{T_{ik}\ \leq\ T_{ik}^{*}, all\ the\ failures\ are\ recoverable\} (15)$$

According to the property of the incomplete Gamma function, we get

$$P_{ik}^{(1)} = P_{ik}^{h} - \exp\left(-\lambda_k^h \xi_{ik}\right) * \sum_{n=1}^{\infty} \frac{\left(x_k \lambda_k^h \xi_{ik}\right)^n}{n!} \Gamma[n, \mu_k T_{ik}^{*} - \mu_k \xi_{ik}] \quad (16)$$

Taking into consideration software reliability and the reliability of communication links, the reliability of subtask i executed on node k, with deadline T_{ik}^{*}, is

$$R_{ik}^{(1)} = P_{ik}^{(1)} P_{ik}^{s} P_{ik}^{l} \quad (17)$$

The grid service reliability with constraints on the life times of subtasks is,

$$R_s^{(1)} = \prod_{i=1}^{m}\{\ 1 - \prod_{k\in D(i)}\left(1 - R_{ik}^{(1)}\right)\ \} \quad (18)$$

4.2 Constraint on the Numbers of Recoveries Performed

Denoted by $L_k (L_k \geq 1)$ the restriction on the allowed number of recoveries on node k. Node k can recover L_k failures at most. When the $L_k + 1$st recoverable failure comes before the completion of subtask i, the node will claim failure of subtask i to the RMS. Thus, the reliability with a constraint on the allowed number of recoveries is the sum of the probability of no failure occurrence in subtask execution, and the probability that L_k failures occurs at most and all L_k failures are recoverable [13]. In other words, it is

$$P_{ik}^{2} = \Pr(n = 0) + \sum_{n=1}^{L_k} \Pr(E^n)$$

$$= \sum_{n=0}^{L_k} \frac{\left(x_k \lambda_k^h \xi_{ik}\right)^n}{n!} \exp\left(-\lambda_k^h \xi_{ik}\right) \quad (19)$$

Using the expression of an incomplete Gamma function, it can be written as,

$$P_{ik}^{2} = P_{ik}^{h} \Gamma[1 + L_k, x_k \lambda_k^h \xi_{ik}]/\Gamma[1 + L_k] \quad (20)$$

Considering software reliability, and the reliability of communication links, the reliability of subtask i executed on node k, with a constraint on the allowed number of recoveries L_k , is

$$R_{ik}^{(2)} = P_{ik}^{(2)} P_{ik}^{s} P_{ik}^{l} \quad (21)$$

Then the service reliability with constraints on the numbers of recoveries performed is,

$$R_s^{(2)} = \prod_{i=1}^{m} \{ 1 - \prod_{k \in D(i)} \left(1 - R_{ik}^{(2)}\right) \} \quad (22)$$

4.3 Constraints on Both the LifeTime of Subtasks and the Number of Recoveries Performed

Furthermore, we can obtain the reliability of node k with the restriction in both the life times of subtasks, and the numbers of recoveries performed. Given the constraint on the life time limitation T_{ik}^{*} and the numbers of recoveries performed $L_k (L_k \geq 1)$, using the above methods discussed, we can easily get the reliability of subtask i executed on node k.

$$R_{ik}^{(3)} = P_{ik}^{s} P_{ik}^{l} *$$

$$\left[P_{ik}^{(2)} - \exp(-\lambda_k^h \xi_{ik}) \sum_{n=1}^{L_k} \frac{(x_k \lambda_k^h \xi_{ik})^n}{n!} \Gamma[n, \mu_k T_{ik}^{*} - \mu_k \xi_{ik}\right] \quad (23)$$

From (23), when $T_{ik}^{*} \rightarrow \infty$, and $L_k \rightarrow \infty$, the result of (23) is the same as that of (11). It can be seen that the situation of no constraints on recovery is the special case of (21). Then, the service reliability with both the life times of subtasks and the numbers of recoveries performed is

$$R_s^{(3)} = \prod_{i=1}^{m} \{ 1 - \prod_{k \in D(i)} \left(1 - R_{ik}^{(3)}\right) \} \quad (24)$$

5 Task Scheduling Based On Multidimensional Requirements

After the grid service is divided into some subtasks, the RMS should quickly and effectively schedule those subtasks to the appropriate nodes according to the particular requirements of those subtasks, and the QoS demands of grid users [15]. In the scheduling, it needs to take into account not only the *hard constraints* of a subtask (OS type, available CPU, memory, disk space, etc.) but also *software constraints* such as the demanded reliability level of grid service, and the constraints on total financial cost. Moreover, with the introduction of LNFR, there should be many other factors that may also be taken into account, such as the recoverability of grid nodes, the allowed number of recovery performed, and the life time of subtask execution in located nodes. In this section, a multi-objective task scheduling model, minimizing cost and maximizing reliability, is presented and a search algorithm based on Min-Min algorithm [15] is proposed to solve this problem.

5.1 Task Scheduling Algorithm

We consider the problem of task scheduling with the following scenarios.

1) The nodes involved in the grid are heterogeneous. Hence, the nodes may have different units of memory and computation resources, processing speeds and failure rates. Also, the communication links may have different bandwidths and failure rates.
2) The cost of subtask execution in grid nodes is mainly dependent upon the execution time, and the charging of resource providers per unit time.
3) The nodes involved in the grid may have the different failure recoverability, and the constraints on the life times of subtasks and the numbers of recoveries performed may be different as well.
4) To simplify the problem formulation, one subtask is allowed to be assigned at one node, and one node can only be allowed to execute one subtask at most.

The goal is to search a task scheduling that minimizes the total cost of grid service and maximizes the grid service reliability simultaneously while satisfying all of the resource constraints as called to be *soft constraints*, the RMS needs a specific subtask scheduling mechanism [15]. Here, denoted by σ a vector of $\{\sigma_{ik} \mid i \in [1, m], k \in [1, k]\}$, which represents a scheme of subtasks on grid nodes. σ_{ik} also has two possible values (1, 0): $\sigma_{ki} = 0$ means that subtask *i* is not assigned on node *k* by the RMS, while $\sigma_{ki} = 1$ means that subtask *i* is assigned on node *k*. The grid service reliability can be determined in term of σ [13] Then, the grid service reliability, and the total cost can be functions of σ_{ik} , which are written respectively as:

$$R_S(\sigma) = \prod_{i=1}^{m} \{ 1 - \prod_{k=1}^{K} (1 - R_{ik}^{(3)})^{\sigma_{ik}} \} \quad (25)$$

$$Cs(\sigma) = \sum_{i=1}^{m} \sum_{k=1}^{K} \sigma_{ik} C_{ik} = \sum_{i=1}^{m} \sum_{k=1}^{K} \sigma_{ik} \gamma_k \xi_{ik} \quad (26)$$

Where γ_k is execution cost in node *k* per unit time, and C_{ik} is the product of γ_k and the required execution time ξ_{ik}.

Obtain task scheduling decision so as to maximize the grid service reliability and minimize the total cost simultaneously, a weighting summation method is proposed, while satisfying all the system resource constraints. We proposed a new algorithm which is based on min-min algorithm [15]. In this, Pr_{ik} is the desirability of assigning subtask i to node k. Here, according to the objective function, the constant heuristic information is adopted, which is written as,

$$Pr_{ik} = C_{ik} + \frac{1}{R_{ik}} \qquad (27)$$

Where C_{ik} is execution cost of subtask *i* on Node *k*, and R_{ik} is the reliability of subtask *i* on node *k*.

The pseudo code of the minimum cost maximum reliability algorithm to solve the subtask scheduling in the grid is as follows:

1. For all subtasks T_{ik} in meta-task M
2. For all resources R_{ik}
3. Selection probability of Jobs is evaluate :
   $$Pr_{ik} = C_{ik} + \frac{1}{R_{ik}} \qquad (27)$$
4. Do until all subtasks in M are mapped
   a. For each subtask in M find the minimum probability of the jobs and resources that obtains it.
   b. Assign subtask T_{ik} to the resource R_{ik} that gives the earliest completion time.
   c. Delete Subtask T_{ik} from M.
   d. Update Status of Resource R_{ik}
5. End do.

6 RESULTS

Suppose a service is divided into four subtasks by RMS. Those four subtasks have no precedent constraints, and can be assigned to four nodes. The information of grid nodes and communication links is shown in Table I, and the attributes of subtasks are shown in Table II.

Table I: Attributes of Nodes and Communication Link

Node *k*	**1**	**2**	**3**	**4**
Hardware Failure Intensity λ_k^h(10^{-4}/s)	1.31	0.87	0.58	1.12
Link Failure Intensity ε_k (10^{-5}/s)	0.92	0.29	0.47	0.39
Link B/W y_k (Mbits/s)	6	2	7	7
Processing Capabilities s_k (Mega Operations/s)	27	22	40	20
Recovering Rate μ_k (/s)	0.78	0.52	0.93	1.19
Recovery Constraint L_k	3	2	1	2
Failure Recoverability x_k	0.45	0.18	0.75	0.58

Table II: Attributes of Subtask

Subtasks i	**1**	**2**	**3**	**4**
Subtask Complexity c_i (Giga Operations)	5.5	6.0	7.0	4.0
Subtask Execution Time (/s)	0.31	0.22	0.44	0.16

Figure 3 gives the comparison of grid service reliability among that with constraint, that with constraints both lifetime and number of recoveries performed, and that without any constraint. With the increase of fault recoverability, the values of grid service reliability are increasing. Probability all Subtasks completed successfully without constraints is calculated using equation (13) GSR = 0.8441 * 0.9573 * 0.99937 * 0.9999 = 0.8075. Grid service reliability with practical constraints is calculated using equation (24), GSR = 0.9662 * 0.9379 * 0.9892 * 0.8954 = 0.8026. Selection probability of Jobs is evaluate using $Pr_{ik} = C_{ik} + \frac{1}{R_{ik}}$ taking this heuristic values we have done task scheduling. Since The Minimum Cost Maximum Reliability algorithm we have used, sufficient number of runs should be taken.

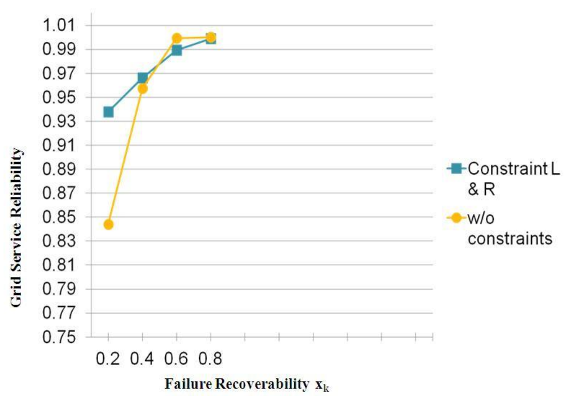

Fig 3: Grid Service Reliability With respective Failure Recoverability.

Table III shows the result of algorithm runs for grid service reliability, cost and scheduling decision.

Table III: GSR, Cost, Scheduling Decision for Minimum Cost Maximum Reliability Algorithm runs

GSR	**Cost ($)**	**Scheduling Decision**
0.8026	142.48	(P1, P3, P2, P4)
0.7852	119.94	(P4, P1, P2, P3)
0.8171	138.13	(P1, P2, P3, P4)
0.8441	146.88	(P3, P4, P2, P1)

7 CONCLUSION

Local node fault recovery mechanism is introduced into the grid. Moreover a constraint on recovery amount is considered in the modeling of grid service reliability with star topology. We calculated the grid service reliability using practical constraints i.e. increasing lifetime to specific limit and performing recoveries. So resource provider is free to choose appropriate fault recovery strategy according to local situation. Based on min-min algorithm minimum cost maximum reliability task scheduling algorithm we proposed to maximizing grid service reliability and minimizing cost. Final result of grid service reliability is around 0.8123. We plan to obtain grid service reliability using Tree Structure Topology considering fault recovery and task scheduling.

REFERENCES

[1]. I. Foster, C. Kesselman and S. Tuecke, "The anatomy of the grid: Enabling scalable virtual organizations", International Journal of High Performance Computing Applications, vol. 15, pp. 200-222, 2001.

[2]. I. Foster and C. Kesselman, The Grid: Blueprint for a New Computing Infrastructure, Morgan-Kaufmann, 1998.

[3]. Y. S. Dai, M. Xie, and K. L. Poh, "Reliability of grid service systems," Computers and Industrial Engineering, vol. 50, no. 1–2, pp. 130–147, 2006.

[4]. G. Levitin, Y. S. Dai, and B. H. Hanoch, "Reliability and performance of star topology grid service with precedence constraints on subtask execution," IEEE Trans. Reliability, vol. 55, no. 3, pp. 507–515, 2006.

[5]. G. Levitin and Y. S. Dai, "Grid service reliability and performance in grid system with star topology," Reliability Engineering and System Safety, vol. 92, no. 1, pp. 40–46, 2007.

[6]. Y. S. Dai, G. Levitin, and X. L. Wang, "Optimal task partition and distribution in grid service system with common cause failures," Future Generation Computer Systems, vol. 23, no. 2, pp. 209–218, 2007.

[7]. Y. S. Dai, Y. Pan, and X. K. Zou, "A hierarchical modeling and analysis for grid service reliability," IEEE Trans. Computers, vol. 56, no. 5, pp. 681–691, 2007.

[8]. T. Paul and X. Jie, "Fault tolerance within a grid environment," in Proceedings of UK e-Science All Hands Meeting, 2003.

[9]. M. Affaan and M. A. Ansari, "Distributed fault management for computational grids," in Proceedings of the Fifth International Conference on Grid and Cooperative Computing, 2006.

[10]. L. Jin, W. Q. Tong, J. Q. Tang, and B.Wang, "A fault-tolerance mechanism in grid," in Proceedings of IEEE International Conference on Industrial Informatics, 2003.

[11]. A. Heddaya and A. Helal, Reliability, Availability, Dependability and Performability: A User-Centered View 1997, Technical Report.

[12]. Y. S. Dai and X. L. Wang, "Optimal resource allocation on grid systems for maximizing service reliability using a genetic algorithm," Reliability Engineering and System Safety, vol. 91, no. 9, pp. 1071–1082, 2006.

[13]. Suchang Guo, Hong-Zhong Huang, Member IEEE, Zhonglai Wang, and Min Xie, Grid Service Reliability Modeling and Optimal Task Scheduling Considering Fault Recovery, IEEE Trans. on Reliability, Vol. 60, no. 1, March 2011.

[14]. Y. S. Dai and G. Levitin, "Optimal resource allocation for maximizing performance and reliability in tree-structured grid services," IEEE Trans. Reliability, vol. 56, no. 3, pp. 444–453, 2007.

[15]. Fangpeng Dong and Selim G. Akl, "Scheduling Algorithms for Grid Computing: State of the Art and Open Problems", Technical Report No. 2006-504, School of Computing, Queen's University Kingston, Ontario January 2006.