Grid Service Reliability Modeling and Optimal Task Scheduling Considering Fault Recovery

Suchang Guo, Hong-Zhong Huang, *Member, IEEE*, Zhonglai Wang, and Min Xie, *Fellow, IEEE*

***Abstract*—There has been quite some research on the development of tools and techniques for grid systems, 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. Based on the proposed grid service reliability model, a multi-objective task scheduling optimization model is presented, and an ant colony optimization (ACO) algorithm is developed to solve it effectively. A numerical example is given to illustrate the influence of fault recovery on grid service reliability, and show a high efficiency of ACO in solving the grid task scheduling problem.**

***Index Terms*—Ant colony optimization, fault recovery, grid service reliability, recoverability, task scheduling.**

ACRONYMS

ACO	Ant Colony Optimization
DCEs	Distributed Computing Environments
GA	Genetic Algorithms
LNFR	Local Node Fault Recovery
OGSA	Open Grid Services Architecture
QoS	Quality of Service
RMS	Resource Management System
RNFR	Remote Node Fault Recovery

Manuscript received October 11, 2009; revised April 30, 2010; accepted August 12, 2010. Date of publication January 24, 2011; date of current version March 02, 2011. This work was supported by the National Natural Science Foundation of China under the Contract 70828001 and the Specialized Research Fund for the Doctoral Program of Higher Education of China under the Contract 20090185110019. Associate Editor: G. Levitin.

S. Guo, H.-Z. Huang, and Z. Wang are with the School of Mechanical, Electronic, and Industrial Engineering, University of Electronic Science and Technology of China, Chengdu, Sichuan 611731, China (e-mail: schguo@189.cn; hzhuang@uestc.edu.cn; wzhonglai@uestc.edu.cn).

M. Xie is with the Department of Industrial and Systems Engineering, National University of Singapore, 117576 Singapore (e-mail: mxie@nus.edu.sg).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TR.2010.2104190

NOTATIONS

x_k	Recoverability of node k, the probability that a hardware failure on node k can be recovered
ξ_{ik}	The required processing time of subtask i on node k
c_i	The computational complexity of subtask i $(i = 1, 2, \cdots, m)$
s_k	The processing speed of node k $(k = 1, 2, \cdots, n)$
N_{ik}	The total number of recoverable failures occurring during the execution of subtask i on node k
TE_{ik}	The total executing time of subtask i on node k
TR_{ik}	The total recovering time of subtask i on node k
T_{ik}	The *lifetime* of subtask i executed on node k
λ_k^h	The hardware failure intensity of node k
λ_i^s	The software failure intensity of programs performing subtasks i
p_{ik}^h	The hardware reliability of node k executing subtask i
p_{ik}^s	The software reliability of node k executing subtask i
ε_k	The failure intensity of communication link between the RMS and node k
p_{ik}^l	The reliability of communication links when node k executes subtask i
l_{ik}	The amount of data exchanged when node k executes subtask i
y_k	The mean speed of communication link between the RMS and node k
T_{ik}^*	The deadline of subtask i on node k
L_k	The restriction on the allowed number of recoveries on node k
γ_k	Execution cost on node k per unit time

I. INTRODUCTION

GRID computing has emerged as the next-generation parallel and distributed computing methodology. Its goal is to provide a service-oriented infrastructure to enable easy access to and coordinated sharing of geographically distributed resources for solving various kinds of large-scale parallel applications in the wide area network. Nowadays, grid computing

has been widely accepted, researched, and given attention to by researchers [1].

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 irreconcilable 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 [2], 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 [6], [7]. Recently, the modeling and analysis of grid service reliability has attracted lots of attention. Dai *et al.* [2] presented a virtual approach to modeling grid services, and derived the grid service reliability using the graphic theory. Dai *et al.* [8], and Levitin and Dai [7] studied grid service reliability for grid systems with star topology, and tree topology, respectively. Dai *et al.* [9] 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.* [6] studied grid service reliability taking the precedence constraints on programs execution into account.

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 resource management system (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 exacerbated by the fact that many grid services will perform long tasks that may require several days of computation [10]. Recently, much effort in fault avoidance and fault removal has been invested so as to improve grid service reliability. Paul and Jie [10] developed an approach to fault tolerance based on job replication in grid systems. Affaan and Ansari [11] introduced a backup mechanism to achieve fault tolerance in grid systems. Jin *et al.* [12] put forward a fault tolerance mechanism in grid systems based on Java threads state capturing, and Mobile Agents. Jozsef and Peter [13] introduced the concept of job migration to achieve fault tolerance in grid systems.

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.

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. Therefore, LNFR can be used in conjunction with RNFR to achieve effective fault tolerance in a grid environment. However, with the introduction of LNFR, the state of resource failures may be divided into unrecoverable failures, and recoverable failures. This non-binary state property of grid resources may bring great difficulties for grid service reliability analysis [14]. Heddaya and Helal [15] studied the effect of LNFR on the reliability of distributed systems, and gave the formula for computing the reliability by analogy, i.e., the reliability where execution of the task can be interrupted by at most one failure was calculated at first, and then the reliability with any number of failures was generalized.

Without considering the influence of RNFR, we focus on the impact of LNFR on reliability, and obtain the formula of grid service reliability by directly analyzing the random execution of gird service, which is different from that in [15]. To simplify the analysis and computation, the grid service process is approximated by a structure with star topology, as done in [6] and [7]. Moreover, in favor of resource providers' customization, certain constraints on fault recovery, i.e., the life times of subtasks, and the numbers of recoveries performed in grid nodes, are imposed, which makes fault recovery mechanism more practical. Under these constraints on fault recovery, the grid service reliability model is presented. Numerical examples show that LNFR can provide an efficient method to improve grid service reliability, especially for the service with time-consuming computation.

Unfortunately, with the introduction of fault recovery and constraints on fault recovery, grid nodes may have various reliability levels, which impose more constraints when doing task scheduling optimization. Some researchers have studied the optimization on grid service reliability. Dai and Wang [16] studied optimal resource allocation for maximizing service reliability using a genetic algorithm. Dai and Levitin [17] 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 on optimization. In this paper, based on the grid service reliability model, we formulate

a multi-objective optimization approach to simultaneously maximize grid service reliability and minimize the cost.

Both the minimization of cost and the maximization of reliability are NP-hard problems [18]. There exist numerous literature works that provide on alternative solutions. Among those methods, mathematical programming approaches such as linear programming, branch-and-bound, and state-space algorithms, are computationally prohibitive if the problem size is large [19]. Recently, metaheuristic algorithms such as genetic algorithms (GA) and Ant Colony Optimization (ACO) have demonstrated successful applications with reasonable time in many fields of reliability engineering [16], [20]–[25]. In this paper, an ant colony optimization is developed to solve this multi-objective optimization problem. A numerical example shows the efficiency of ACO in solving the optimization problem.

This paper is organized as follows. Section II describes the characteristics of grid systems and services, and presents the model of grid service reliability considering fault recovery. To be more practical, the reliability models under execution deadline and a constraint on number of recovery performed are presented in Section III. Section IV presents a multi-objective task scheduling model. An ACO-based algorithm is developed to solve this problem. Section V uses a numerical example to show the positive influence of fault recovery and effectiveness of our algorithms. Section VI concludes this paper, and indicates some directions for future study.

II. Reliability Analysis for Grid Systems Considering Fault Recovery

A. Star Topology of Grid Service

The Open Grid Services Architecture (OGSA) describes an architecture for a service-oriented grid computing environments for business and scientific use [26]. Namely, the service-oriented grid can be viewed as a widely distributed server, 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 [7]. 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.

Different from traditional *distributed computing environments* (DCEs), 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], [6], [7].

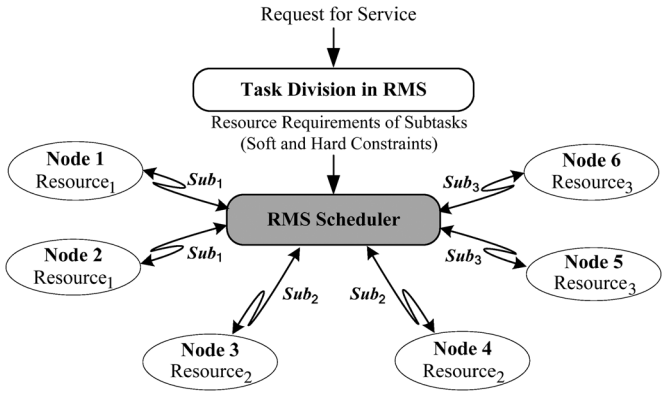

Fig. 1. Example of grid systems with star topology.

An example of a star topology is given in Fig. 1, in which three subtasks, denoted by Sub_1, Sub_2, and Sub_3, are assigned to six nodes connected with the RMS through respective communication links. Each of the three subtasks is assigned to two nodes for parallel execution, e.g., Sub_1 is assigned to node 1 and node 2.

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; and the time of task processing by the RMS is negligible when compared with subtasks' processing times.
(b) When a service request arrives at the RMS, the RMS responds to it immediately; when a subtask is assigned to a node, 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 at any time.
(e) The failure processes of nodes and communication links can be modeled by Poisson processes, respectively [2], [6], [7].
(f) The failures in different elements (nodes or communication links) are independent.

B. Fault Recovery in Grid Systems

Grid system is a complex system which spans multiple heterogeneous and disjoined organizations. It becomes increasingly difficult to guarantee that there is no failure occurring in the grid in some way [10]. Those failures may arise from software bugs, human operator errors, performance overload, severe congestion, or electronic component failures. Additionally, environmental disasters such as fires, floods, and earthquakes, 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.

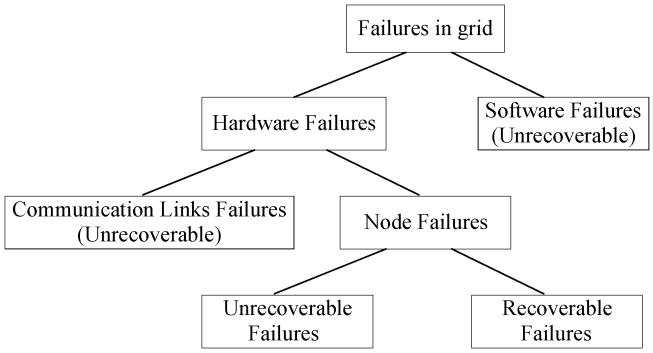

Fig. 2. Classification of failures in grid systems according to recoverability.

As an important approach of fault tolerance, 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 in some sense. 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 two categories: *unrecoverable failures, and recoverable failures.* Software failures occurring on grid nodes, which are caused by embedded faults in programs [27], are unrecoverable failures because there is no fault removal activities performed, and the source codes of the programs are not changed. For communication link failures, because it is impossible to install any recovering module on communication links, they belong to unrecoverable failures as well. However, hardware failures occurring on grid nodes can be unrecoverable or recoverable failures. For recoverable failures such as those caused by human operation errors or performance overload, once the failed node becomes operational, the interrupted execution of subtask can be resumed by recovering as much state information as needed, which can be achieved by some particular recovery procedures in grid nodes. For unrecoverable hardware failures, the subtask will be terminated. Fig. 2 illustrates the classification of failures in grid systems according to recoverability.

For ease of describing the recoverability of hardware failures on node k, a random variable $X_k^{(j)}$ is defined, which has two possible values (1, 0). If $X_k^{(j)} = 0$, it means that the jth failure on grid node k is recoverable. If $X_k^{(j)} = 1$, it means that the jth failure on grid node k is unrecoverable. Denote by x_k the probability that a hardware failure on node k is recoverable, and then it is obtained that $\Pr\{X_k^{(j)} = 0\} = x_k$, $\Pr\{X_k^{(j)} = 1\} = 1 - x_k$.

Due to the introduction of LNFR, it can be easily known that successful completion of subtasks on grid nodes can be partly achieved by a perfectly reliable node, but it can also be attained by a node with fault recovery. Therefore, to reward recovery actions, the analysis of grid service reliability has to be extended. In the following section, we will discuss the modeling and analysis of grid service reliability considering fault recovery in detail.

C. Reliability Modeling and Analysis With Fault Recovery

Denote by m the number of divided subtasks by the RMS after the RMS receives a service S. Assuming that subtask i is assigned to node k, the required processing time of subtask i on node k, ξ_{ik}, is [2], [7]

$$\xi_{ik} = c_i / s_k \tag{1}$$

With the introduction of LNFR, an operational state in grid nodes is further classified into executing or recovering. A recovering state means a state where a grid node can be operational but not quite ready to be accessed, as it performs recovery procedures. Denote 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 denote by $TE_{ik}^{(j)}$ $(j = 1, 2, L, n, n{+}1)$, and $TR_{ik}^{(j)}$ $(j = 1, 2, L, n)$ the executing times, and recovering times in the execution process of subtask i on node k, respectively. The subtask execution process goes on until the subtask is successfully completed, or it is terminated by an unrecoverable failure. In the former case, the total execution time of subtask i on node k is ξ_{ik}, which can be obtained by (1); in the latter case, the subtask is terminated. Fig. 3 gives an ideal situation, in which all of n failures occurring in the execution are recoverable failures and subtask i is completed successfully. In this example, the first failure occurs at t_1 and is recovered at t_2, the second failure occurs at t_3 and is recovered at t_4, and so on.

If $N_{ik} = n$ $(n \geq 1)$, then

$$TE_{ik} = \sum_{j=1}^{n+1} TE_{ik}^{(j)}, \quad \text{and} \quad TR_{ik} = \sum_{j=1}^{n} TR_{ik}^{(j)}.$$

The *lifetime* of subtask i executed on node k, T_{ik}, is

$$T_{ik} = TE_{ik} + TR_{ik}. \tag{2}$$

If subtask i is successfully completed on node k, then its life time is

$$T_{ik} = \xi_{ik} + TR_{ik}, \tag{3}$$

where ξ_{ik} is obtained by (1).

From assumption (e), 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 (e) and (f), $TE_{ik}^{(j)}$ $(j = 1, 2, L,$ are s-independent and identically distributed (i.i.d.) random variables, each following an exponential distribution with parameter λ_k^h. 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* in the literature).

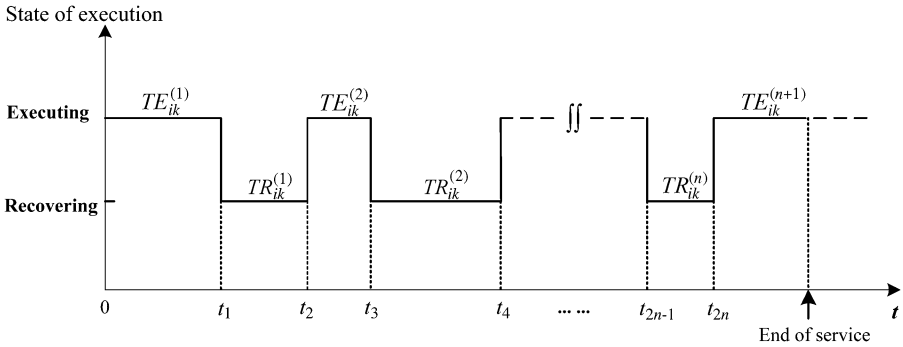

Fig. 3. Example of the execution process of subtask i on node k with fault recovery.

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

If the jth $(j \geq 1)$ failure is unrecoverable, i.e., $X_k^{(j)} = 1$, then the subtask will be terminated at the occurrence of this unrecoverable failure, 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 k executing subtask i, considering only hardware failures, is

$$\begin{aligned} p_{ik}^h &= P_1 + P_2 \\ &= \Pr\{\text{no failure occurs}\} \\ &\quad + \Pr\{\text{all the failures occuring are recoverable}\}. \end{aligned} \tag{4}$$

P_1 can be easily obtained as

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

and P_2 can be obtained as

$$\begin{aligned} P_2 &= \sum_{n=1}^{\infty} \Pr\{n \text{ recoverable failures occurring,} \\ &\qquad \text{subtask is completed}\} \\ &= \sum_{n=1}^{\infty} \Pr\left\{E^{(n)}\right\} \end{aligned} \tag{6}$$

where $E^{(n)}$ is the event that subtask 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

$$\begin{aligned} &\Pr\left\{E^{(n)}(n \geq 1)\right\} \\ &= \Pr\left\{W_{ik}^{(n)} < \xi_{ik}, W_{ik}^{(n)} + TE_{ik}^{(n+1)} > \xi_{ik}, \sum_{j=1}^{n} X_k^{(j)} = 0\right\} \end{aligned} \tag{7}$$

where $W_{ik}^{(n)} \equiv TE_{ik}^{(1)} + TE_{ik}^{(2)} + L + TE_{ik}^{(n)}$.

It is known from [28] that $W_{ik}^{(n)}$ is an Erlang random variable with parameters (n, λ_k^h). Moreover, according to assumption (f), we have that $W_{ik}^{(n)}$ is s-independent with $TE_{ik}^{(n+1)}$. Therefore, the joint density of $(W_{ik}^{(n)}, TE_{ik}^{(n+1)})$ can be written as

$$f_{ik}^{(n)} = \begin{cases} \exp\left(-\lambda_k^h x - \lambda_k^h y\right) \frac{(\lambda_k^h)^2 (\lambda_k^h x)^{n-1}}{(n-1)!}; & x \geq 0, y \geq 0 \\ 0; & \text{else.} \end{cases} \tag{8}$$

Substituting (8) into (7), we have

$$\begin{aligned} \Pr\{E^{(n)}\} &= (x_k)^n \int_0^{\tau_{ik}} \frac{(\lambda_k^h x)^{n-1}}{(n-1)!} \exp\left(-\lambda_k^h x\right) dx \\ &\quad \times \int_{\tau_{ik}-x}^{+\infty} \left(\lambda_k^h\right)^2 \exp\left(-\lambda_k^h y\right) dy \\ &= \frac{\left(x_k \lambda_k^h \tau_{ik}\right)^n}{n!} \exp\left(-\lambda_k^h \tau_{ik}\right). \end{aligned} \tag{9}$$

The result obtained in (5) is the special case of (9) for which $n = 0$. Therefore, substitute (9) into (6) and (4), the reliability of node k executing subtask i, considering only hardware failures, is obtained as

$$p_{ik}^h = \sum_{n=0}^{\infty} \frac{\left(x_k \lambda_k^h \xi_{ik}\right)^n}{n!} \exp\left(-\lambda_k^h \xi_{ik}\right) = \exp\left[(1 - x_k)\lambda_k^h \xi_{ik}\right]. \tag{10}$$

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

$$p_{ik}^s = \exp\left(-\lambda_i^s \xi_{ik}\right). \tag{11}$$

The reliability of communication links can be modeled by

$$p_{ik}^l = \exp(-\varepsilon_k l_{ik}/y_k). \tag{12}$$

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

$$R_{ik} = p_{ik}^h p_{ik}^s p_{ik}^l = \exp\left[-(1 - x_k)\lambda_k^h \xi_{ik} - \lambda_i^s \xi_{ik} - \varepsilon_k l_{ik}/y_k\right]. \tag{13}$$

To improve grid service reliability, a subtask is normally assigned to several nodes for parallel execution [2], [7]. Whenever a node on which a subtask is being executed returns the output to the RMS, the subtask is considered to be completed. Denote by $D(i)$ the node set to which subtask i is assigned, the reliability of subtask i, which is often referred to as *grid program reliability* [2], is

$$\begin{aligned} R(Sub_i) &= \Pr\{\text{at least one node in } D(i) \\ &\qquad \text{can complete subtask } i\} \\ &= 1 - \prod_{k \in D(i)} (1 - R_{ik}), \end{aligned} \tag{14}$$

where R_{ik} is given by (13).

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

$$\begin{aligned} R_S &= \Pr\{\text{all the subtasks can be completed successfully}\} \\ &= \prod_{i=1}^{m} R(Sub_i) \\ &= \prod_{i=1}^{m}\left\{1-\prod_{k\in D(i)}\left\{1-\exp\left[-(1-x_k)\lambda_k^h\xi_{ik}-\lambda_i^s\xi_{ik}-\varepsilon_k l_{ik}\right]\right\}\right\}. \end{aligned} \tag{15}$$

Given the processing times and failure intensity of grid nodes and communication links, which can be estimated by a grid monitoring system, the grid service reliability can be easily obtained.

III. Grid Service Reliability Modeling With Practical Considerations

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. 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.

A. Constraints on the Life Times of Subtasks

To prevent the lifetime of a subtask from exceeding an allowed time limit, we can set a deadline for subtask execution. Once the lifetime of subtask i executed on node k, T_{ik}, exceeds this deadline, denoted by T_{ik}^*, the node will claim failure of the subtask to the RMS.

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 (g), 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

$$F_{TR_{ik}}(t)\equiv \Pr(TR_{ik}\leq t)=\sum_{j=n}^{\infty}\frac{(\mu_k t)^j}{j!}e^{-\mu_k t};\quad t\geq 0. \tag{16}$$

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

$$\begin{aligned} p_{ik}^{(1)} &= P_1+P_3=\Pr\{\text{no failure occurs}\} \\ &\quad +\Pr\{T_{ik}\leq T_{ik}^*, \text{all the failures are recoverable}\} \end{aligned} \tag{17}$$

where P_3 can be obtained by

$$P_3=\sum_{n=1}^{\infty}\Pr\left\{E^{(n)},T_{ik}\leq T_{ik}^*\right\}. \tag{18}$$

The probability being summed in (18) can be calculated by

$$\begin{aligned} &\Pr\left\{E^{(n)},T_{ik}\leq T_{ik}^*\right\} \\ &\quad=\Pr\left\{TR_{ik}\leq T_{ik}^*-\xi_{ik}|E^{(n)}\right\}\Pr\{E^{(n)}\} \\ &\quad=\exp\left(-\lambda_k^h\xi_{ik}\right)\frac{\left(x_k\lambda_k^h\xi_{ik}\right)^n}{n!} \\ &\qquad-\exp\left(-\mu_k T_{ik}^*+\mu_k\xi_{ik}-\lambda_k^h\xi_{ik}\right)\frac{\left(x_k\lambda_k^h\xi_{ik}\right)^n}{n!} \\ &\qquad\times\sum_{j=0}^{n-1}\frac{\left(\mu_k T_{ik}^*-\mu_k\xi_{ik}\right)^j}{j!} \end{aligned} \tag{19}$$

Substitute (19) into (18) and (17), we have

$$\begin{aligned} p_{ik}^{(1)} &= p_{ik}^h-\exp\left(-\mu_k T_{ik}^*+\mu_k\xi_{ik}-\lambda_k^h\xi_{ik}\right) \\ &\quad\times\sum_{n=1}^{\infty}\left[\frac{\left(x_k\lambda_k^h\xi_{ik}\right)^n}{n!}\sum_{j=0}^{n-1}\frac{\left(\mu_k T_{ik}^*-\mu_k\xi_{ik}\right)^j}{j!}\right] \end{aligned} \tag{20}$$

where p_{ik}^h is given by (10).

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

$$\begin{aligned} p_{ik}^{(1)} &= p_{ik}^h-\exp\left(-\lambda_k^h\xi_{ik}\right) \\ &\quad\times\sum_{n=1}^{\infty}\frac{\left(x_k\lambda_k^h\xi_{ik}\right)^n\Gamma\left[n,\mu_k T_{ki}^*-\mu_k\xi_{ik}\right]}{\Gamma[n]\Gamma[n+1]} \end{aligned} \tag{21}$$

where $\Gamma[n,\mu_k T_{ik}^*-\mu_k\xi_{ik}]=\int_{\mu_k T_{ik}^*-\mu_k\xi_{ik}}^{\infty}\exp(-x)\cdot x^{n-1}dx$, and $\Gamma[n]\equiv(n-1)!$ for any $n\geq 1$.

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

$$R_{ik}^{(1)}=p_{ik}^s p_{ik}^l p_{ik}^{(1)}. \tag{22}$$

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

$$R_S^{(1)}=\prod_{i=1}^{m}\left\{1-\prod_{k\in D(i)}\left\{1-R_{ik}^{(1)}\right\}\right\}. \tag{23}$$

B. Constraint on the Numbers of Recoveries Performed

Denote 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. In other words, it is

$$p_{ik}^{(2)} = \Pr(n=0) + \sum_{n=1}^{L_k} \Pr(E^{(n)}) = \sum_{n=0}^{L_k} \frac{\left(x_k \lambda_k^h \xi_{iki}\right)^n}{n!} \exp\left(-\lambda_k^h \xi_{iki}\right). \quad (24)$$

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

$$p_{ik}^{(2)} = \frac{p_{ki}^h \Gamma\left[1+L_k, x_k \lambda_k^h \tau_{ki}\right]}{\Gamma[1+L_k]}. \quad (25)$$

where p_{ik}^h is given by (10).

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

$$R_{ik}^{(2)} = p_{ik}^{(2)} p_{ik}^s p_{ik}^l = \frac{p_{ik}^s p_{ik}^l p_{ik}^h \Gamma\left[1+L_k, x_k \lambda_k^h \xi_{ik}\right]}{\Gamma[1+L_k]}. \quad (26)$$

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

$$R_S^{(2)} = \prod_{i=1}^{m} \left\{ 1 - \prod_{k \in D(i)} \left\{ 1 - R_{ik}^{(2)} \right\} \right\}. \quad (27)$$

C. Constraints on Both the Life Times of Subtasks, and the Numbers 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

$$R_{ik}^{(3)} = p_{ik}^s p_{ik}^l \times \left[p_{ik}^{(2)} - \exp\left(-\lambda_k^h \xi_{ik}\right) \sum_{n=1}^{L_k} \frac{\left(x_k \lambda_k^h \xi_{ik}\right)^n \Gamma\left[n, \mu_k T_{ik}^* - \mu_k \xi_{ik}\right]}{\Gamma[n+1]\Gamma[n]} \right]. \quad (28)$$

where $p_{ik}^{(2)}$ is given by (25). From (28), when $T_k^* \to \infty$, and $L_k \to \infty$, the result of (28) is the same as that of (13). It can be seen that the situation of no constraints on recovery is the special case of (26). Then, the service reliability with both the life times of subtasks and the numbers of recoveries performed is

$$R_S^{(3)} = \prod_{i=1}^{m} \left\{ 1 - \prod_{k \in D(i)} \left\{ 1 - R_{ik}^{(3)} \right\} \right\}. \quad (29)$$

IV. Optimal Task Scheduling Based on Multi-Dimensional 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. In the scheduling, it needs to take into account not only the *hard constraints* of a subtask (the operating system 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 ACO is proposed to solve this problem.

A. Problem Formulation

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

1) The nodes involved in the grid are heterogeneous. Hence, the nodes may be capacitated with various units of memory and computation resources, and they may have different 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.

In this study, 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. Suppose that a service S is divided into m subtasks which can be assigned on K accessible nodes. When scheduling the subtasks, the RMS must satisfy *hard constraints* of subtasks in the first place. Hence, we use a_{ik} to describe the influence of those hard constraints on subtask scheduling. a_{ik} has two possible values (1, 0): if $a_{ik} = 1$, it means that subtask i can be allowed to be allocated on node k; and if $a_{ik} = 0$, it means that subtask i can not be allowed to be allocated on node k. Meanwhile, to satisfy users' demands, i.e., maximizing the grid service reliability and minimizing the cost, as called to be *soft constraints*, the RMS needs a specific subtask scheduling mechanism. Here, denote by $\boldsymbol{\sigma}$ a vector of $\{\sigma_{ik} | i \in [1,m], k \in [1,k]\}$, 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 $\sigma_{ik} = 1$ means that subtask i is assigned on node k.

Hence, given the structure of the nodes and links involved in the service, the grid service reliability can be determined in term

of $\boldsymbol{\sigma}$. Then, the grid service reliability, and the total cost can be functions of σ_{ik}, which are written respectively as

$$R_S(\boldsymbol{\sigma}) = \prod_{i=1}^{m}\left\{1 - \prod_{k=1}^{K}\left(1 - R_{ik}^{(3)}\right)^{\sigma_{ik}}\right\}, \text{ and} \quad (30)$$

$$C_S(\boldsymbol{\sigma}) = \sum_{i=1}^{m}\sum_{k=1}^{K}\sigma_{ik}C_{ik} = \sum_{i=1}^{m}\sum_{k=1}^{K}\sigma_{ik}\gamma_k\xi_{ik} \quad (31)$$

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}.

To find the optimal solution of $\boldsymbol{\sigma}$ so as to maximize the grid service reliability and minimize the total cost simultaneously, a weighting summation to optimize the two criteria simultaneously is proposed, while satisfying all the system resource constraints. The optimization model is given as follows.

Decision variables: $\boldsymbol{\sigma} = \{\sigma_{ik} = 0, 1 | i \in [1, m], k \in [1, K]\}$

Objective function:

$$\text{Min} \quad C_S(\boldsymbol{\sigma}) = \sum_{i=1}^{m}\sum_{k=1}^{K}\sigma_{ik}C_{ik} = \sum_{i=1}^{m}\sum_{k=1}^{K}\sigma_{ik}\gamma_k\xi_{ik} \quad (32)$$

$$\text{Max} \quad R_S(\boldsymbol{\sigma}) = \prod_{i=1}^{m}\left\{1 - \prod_{k=1}^{K}\left(1 - R_{ik}^{(3)}\right)^{\sigma_{ik}}\right\} \quad (33)$$

Subject to:

$$\sum_{k=1}^{K}\sigma_{ik} = 1, \quad i = 1, 2, L, m \quad (34)$$

$$\sum_{i=1}^{m}\sigma_{ik} \le 1, \quad k = 1, 2, L, K \quad (35)$$

$$\sigma_{ik} = a_{ik} \text{ when } a_{ik} = 0, \quad i = 1, 2, L, m \quad \text{and} \quad k = 1, 2, L, K \quad (36)$$

Constraints (34) and (35) mean that one subtask can only be allowed to be assigned at one node, and one node can only be allowed to execute one subtask at most, respectively. Constraint (36) is the hard constraint.

For the above multi-objective optimization problem, there are two most common approaches [29]. One is to combine the individual objective functions into a single composite function by the weighted sum method or utility functions. The other one is to use Pareto optimal sets. Although the Pareto set includes all rational choices, it is heuristic, and it is hard to tell which scheduling is really good for practical scenarios [30]. Despite the deficiencies with respect to depicting the Pareto optimal set, i.e., the weighted sum method seeks Pareto optimal solutions one by one by systematically changing the weights between the objective functions, the weighted sum method continues to be used extensively not only to provide multiple solution points by varying the weights consistently, but also to provide a single solution point that reflects the user-applied preferences [31]. Yin *et al.* [19]used the weighted sum method for multi-objective task allocation in distributed system. Proos *et al.* [32] applied the method to topology optimization. In this paper, a weighted sum method is proposed to optimize the two criteria simultaneously. The mathematical formulation combining the two optimization objectives is as follows.

$$\text{Min} \quad Z_S(\boldsymbol{\sigma}) = C_S(\boldsymbol{\sigma}) + \chi / R_S(\boldsymbol{\sigma}) \quad (37)$$

where χ is a scaling factor. The scaling factor χ plays two roles. First, it normalizes the values of $C_S(\boldsymbol{\sigma})$ and $R_S(\boldsymbol{\sigma})$ to comparative ranges such that $Z_S(\boldsymbol{\sigma})$ will not be dominated by a single objective. Second, χ can be used as a weighting parameter which controls the relative significance of each objective.

This task scheduling optimization is a typical combinatorial optimization problem, and the whole solution space is 2^{K-m} because $\boldsymbol{\sigma} = \{\sigma_{ik} = 0, 1 | k \in [1, K], i \in [1, m]\}$. Meanwhile, the solution space increases exponentially with the number of nodes. To solve this optimization problem, the exhaustive search algorithm is not effective due to its time-consuming characteristic. Hence, some intelligent optimization methods, e.g., ant colony optimization, are suitable for such complex combinatorial optimization problems.

B. Ant Colony Optimization

Ant colony optimization (ACO) is a meta-heuristic optimization technique inspired by research on real ant foraging behavior. The ACO paradigm was first proposed by Dorigo and Blum [33], and has been successfully applied to diverse combinatorial optimization problems including traveling salesman [33], telecommunication networks [34], and scheduling [35].

1) Construction of Solutions: A permutation of subtask scheduling subject to the constraint is termed a feasible solution. In the algorithm, the process of constructing a feasible solution can be divided into m steps. In the ith $(0 < i \le m)$ step, ant j finds one accessible node for subtask i, and then moves to the $i + 1$st step for subtask $i + 1$ scheduling until all the subtasks have been assigned, which can guarantee all the subtasks are scheduled as stated in (34). In the scheduling, all infeasible moves of ant j must be stored into a $Tabu$ list denoted by $Tabu_j$. This list is the memory of ant j saving the index of infeasible allocations, and is initialized according to the hard constraint, as shown in (36). Once node k is selected for subtask i by ant j, node k will be added to the $Tabu_j$ to avoid being selected again by ant j in the following scheduling, which is (35). In addition to the list $Tabu_j$, all the nodes selected by ant j in m steps are stored in a memory represented by $Path_j$. A $Path_j$ is one feasible solution of subtask scheduling, and will be used to update the pheromones of moves selected by ants at the end of iteration.

2) Selection Probability: In each step, an ant in a subtask chooses a node as the location of the corresponding subtask. This move can be represented by edge (i, k), which is the assignment of subtask i to node k. Ant j chooses a node based on a combination of two factors, namely the desirability of that move, and the quantity of pheromone on the edge which is to be traversed. There are different methods in literature to combine these factors. According to the method proposed by [36], ant j chooses edge (i, k) for traversing by the probability

$$p_{ik}^{j} = \begin{cases} \dfrac{\alpha\tau_{ik} + (1-\alpha)/\eta_{ik}}{\sum_{s \notin Tabu_j}[\alpha\tau_{is} + (1-\alpha)\eta_{is}]}, & \text{if } k \notin Tabu_j \\ 0, & \text{else} \end{cases} \quad (38)$$

TABLE I
ATTRIBUTES OF GRID NODES AND COMMUNICATION LINKS

Node k	1	2	3	4	5	6	7	8	9	10
Processing capability s_k (Mega Operations/s)	10	20	15	25	12	20	13	10	29	32
Link Bandwidth y_k (*Mbits/s*)	2	3	4	5	6	6	3	4	3	4
Hardware failure intensity λ_k^h (10^{-4}/*s*)	1.0	1.8	1.2	1.5	1.9	0.8	0.9	1.2	1.2	0.8
Failure intensity of link ε_k (10^{-5}/*s*)	0.1	0.2	0.3	0.4	0.1	0.2	0.3	0.1	0.2	0.3
Recovering rate μ_k (/s)	0.5	0.6	0.5	0.7	0.5	0.8	0.9	0.5	0.4	0.3
Recovery constraint L_k	3	1	2	1	1	1	1	2	1	1
Failure recoverability x_k	0.1	0.3	0.4	0.2	0.4	0.3	0.2	0.3	0.2	0.3
Cost per unit time γ_k (10^{-1}\$/*s*)	0.8	1.3	1.1	1.1	0.7	1.0	0.8	0.9	1.8	1.9

where τ_{ik} is the quantity of pheromone on the edge (i,k), and η_{ik} is the desirability of assigning subtask i to node k. Here, according to the objective function (30), the constant heuristic information is adopted, which is written as

$$\eta_{ik} = C_{ik} + \chi / R_{ik} \tag{39}$$

where C_{ik} is execution cost of Sub_i in Node k, and R_{ik} is the reliability of Sub_i in node k, which can be calculated by (27). χ is the scaling factor.

There are two reasons for adopting the method of [36]. The first is that only one control parameter, i.e., α $(0 < \alpha < 1)$, is used to map the relative importance of quantity of pheromone, and the desirability of each move. The second reason is the computational efficiency of this method as multiplication operations are used instead of exponentiations in the original ACO algorithm [33].

3) Pheromone Updating Rule: Pheromone updating is a process of changing the quantity of pheromone over time on each edge. Before activating the next iteration, the quantity of pheromone on each edge is updated by the pheromone updating rule so as to avoid local convergence, and explore more search space as well. According to the original ACO algorithm [33] in the single objective function problems, the pheromone updating rule on each edge is

$$\tau'_{ik} = (1-\rho)\tau_{ik} + \Delta\tau_{ik} \tag{40}$$

where τ'_{ik} is the updated quantity of pheromone on the edge (i,k), and ρ $(0 < \rho \leq 1)$ is the evaporation rate of pheromone. $\Delta\tau_{ik}$ is the quantity of pheromone laid on edge (i,k) by the ants during the current iteration. It can be obtained by

$$\Delta\tau_{ik} = \sum_{j=1}^{\beta} \Delta\tau_{ik}^{j} \tag{41}$$

where β is the total number of ants, and $\Delta\tau_{ik}^{j}$ is the amount of pheromone laid on edge (i,k) by ant j, which can be computed as

$$\Delta\tau_{ik}^{j} = \begin{cases} Q/fit_j, & \text{if } edge(i,k) \text{ is in } Path_j \\ 0, & \text{else} \end{cases} \tag{42}$$

where Q is a constant parameter. fit_j can be viewed as the fitness of $Path_j$, which can be obtained by

$$fit_j = C(Path_j) + \chi / R_S(Path_j). \tag{43}$$

4) Ant Colony Algorithm: The pseudo code of the ant algorithm to solve the subtask scheduling in the grid is as follows.

1) *Initialization*

Initialize $\tau_{ik}(0)$, R_{ki}, C_{ki}, α, β, ρ, a, *and* Max_Iter *(number of iteration).*

2) *Construction*

For each ant in each step, choose the node to move into in probability.Update the $Tabu_j$ *and* $Path_j$.

3) *Pheromone update*

For each $Path_j$ *do compute* fit_j*update Pheromone*

4) *Terminating condition*

if $total\_iteration < max\_Iter$ *got to step* 2 *otherwise stop.*

V. NUMERICAL EXAMPLE

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

Firstly, we investigate the influence of fault recovery on grid service reliability. To achieve this aim, a fixed task scheduling is given, i.e., $\boldsymbol{\sigma}_{fixed} = \{2,4,5,9,10\}$. A subtask is limited to be allocated at one node for the sake of computational simplification. Meanwhile, the same failure recoverability and constraint on recovery time are assumed, i.e., $x_k = x$, $TR_{ik} = TR = 5s$ for $1 \leq k \leq 10$. Fig. 4 gives the comparison of grid service reliability among that with constraint L, that with constraints both L_k and TR, and that without any constraint. In Fig. 4, it can be seen that, when failure recoverability $x = 0$, the values of grid service reliability with different constraints are the same as would be expected, i.e., $x = 0$, $R_S = R_S^{(2)} = R_S^{(3)} = 0.7445$. With the increase of fault recoverability x, the values of grid

TABLE II
ATTRIBUTES OF SUBTASKS

Subtask i	1	2	3	4	5
Software failure intensity λ_i^s ($10^{-5}/s$)	1.0	1.5	1.4	1.6	1.0
Subtask complexity c_i (*Giga Operations*)	6.0	8.0	7.0	10	8.5
Exchanged data b_{ik} (*Mbit*)	12	16	14	20	17

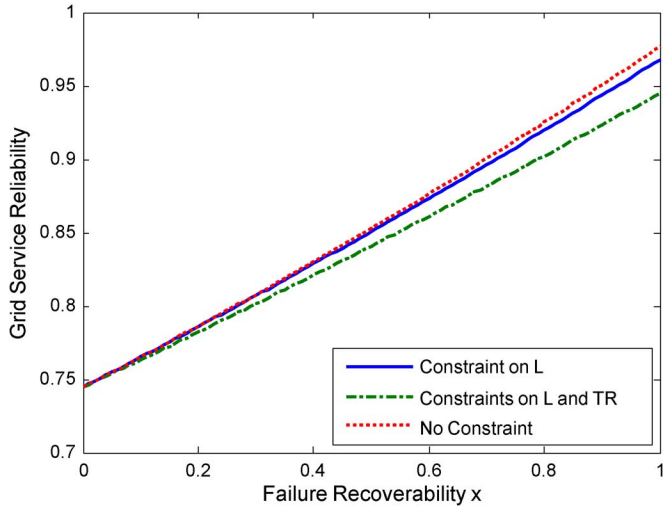

Fig. 4. Grid service reliability with respect to failure recoverability in difference situations.

service reliability are increasing while the increasing of grid service reliability without any constraint is the fastest among three situations. In particular, when $x = 1$, $R_S = 0.9767$, $R_S^{(2)} = 0.9678$, and $R_S^{(3)} = 0.9447$.

From the above analysis, we can see that fault recovery has a positive influence on grid service reliability. However, the constraints on both the life times of subtasks and the numbers of recoveries performed, which is customized by resource providers, have a negative impact on grid service reliability. To balance the contradiction of them, the optimization of task scheduling is necessary so as to satisfy the demand of users.

The hard constraints matrix $\mathbf{a}$, in which a_{ik} is the relationship between the subtask i and node k, is

$$\mathbf{a} = [a_{ik}]_{5\times 10} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}.$$

According to the hard constraints matrix $\mathbf{a}$, and the constraints of (34) and (35), the whole set of feasible solutions numbers $6 \times 7 \times 6 \times 2 = 504$. Firstly, the exhaustive search is used to find the best solution of (30). When the scaling factor χ is 211.3961, the minimum cost of 504 feasible solutions, the best solution is $\boldsymbol{\sigma}_{best} = \{7, 4, 6, 10, 9\}$. The corresponding grid service reliability, and total cost are 0.8527, and \$213.3192, respectively.

Then, an ACO-based program was composed in Matlab 7.5, and executed on a ''Xeon(R) 5120 1.86G" processor with 3G RAM. Based on some experimental tests, experimentally derived rules of ACO are proposed to set the value of parameters:

$$\beta = m = 6; \quad \alpha = 0.4; \quad \rho = 0.4;$$
$$\text{Max\_Iter} = 20; \quad \tau_{ik}(0) = 0.01; \quad Q = 1.$$

The ACO program has been executed 100 times. Of the 100 obtained results, 95 are the same as $\boldsymbol{\sigma}_{best}$, and the average execution time is about 0.0016s. Some statistics out of the 100 results of grid service reliability, and total cost are given in Table III.

TABLE III
STATISTICS OF GRID SERVICE RELIABILITY AND TOTAL COST FOR 100 ACO RUNS

	Max	Min	Ave	Std. Deviation
Grid service reliability	0.8527	0.8501	0.8527	3.8257×10^{-4}
Total cost (\$)	215.3961	213.3192	213.3648	0.2927
Execution Time (s)	0.0214	0.0146	0.0157	0.0016

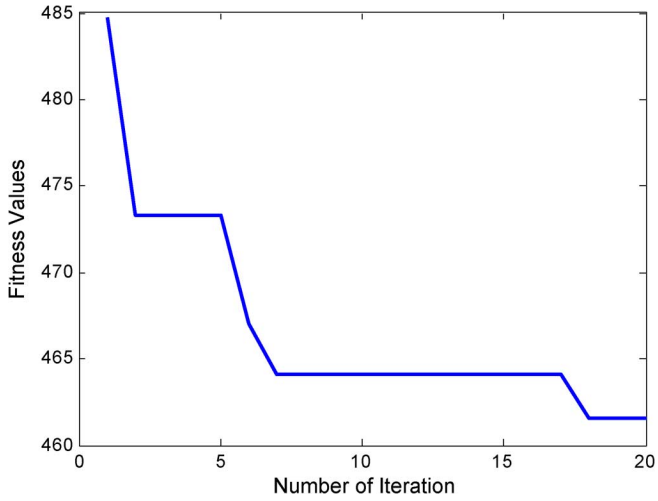

Fig. 5. Trend of fitness function at different iterations in one run.

From Table III, a slight standard deviation of grid service reliability means mostly surrounding the average value or in equivalence often near the best solution out of all experiments. Thus, the ACO can often find a near optimal solution even though it may not guarantee the optimum every time. The average execution time is about 0.0157 seconds per run, which is much efficient. The trend of fitness function at different iterations in one run is plotted in Fig. 5 which shows that the ACO has a good convergence rate.

Finally, the influence of scaling factor χ is studied. In the above analysis, the scaling factor χ is fixed at the minimum

TABLE IV
INFLUENCE OF SCALING FACTOR ON THE TASK SCHEDULING DECISION

χ	Grid service reliability	Cost ($)	Scheduling Decision
1~65	0.8318	211.3961	{5,4,6,10,9}
66~287	0.8527	213.3192	{7,4,6,10,9}
288~1192	0.8543	213.9192	{7,6,4,10,9}
1193~1500	0.8586	220.9961	{3,6,4,10,9}

cost of 504 feasible solutions. To analyze the influence of the scaling factor on the scheduling solution, we change the value of χ, and the corresponding solution is given in Table IV. From Table IV, we can see that χ is critical for the determination of the scheduling decision. When the value of χ increases, it means that the importance of reliability increases, and the importance of cost decreases. However, grid service reliability increases and the cost also increases in Table IV, which is not expected. The reason is that the price of resources given in Table I is arbitrary, while in practice the price is closely related to the performance of grid resources, such as CPU processing capability, and reliability. Therefore, the price of grid resources also plays an importance part in grid task scheduling.

VI. CONCLUSION AND DISCUSSIONS

In this paper, a fault recovery mechanism is introduced into the grid, and the modeling of grid service reliability considering fault recovery is presented. In order to make it more practical, a constraint on recovery amount is discussed in the modeling of grid service reliability. As for the implementation of fault recovery in grid resources, it can be achieved by embedding a fault recovery module in grid clients. In the module, there are options such as the allowed life times of grid subtasks, and the allowed numbers of recoveries performed. By those options, resource providers can be free to choose appropriate fault recovery strategies according to the local situations. Based on that, a task scheduling optimization model to maximize grid service reliability and minimize the total cost simultaneously is proposed, and an ACO algorithm is used to solve this task scheduling problem. A numerical example is shown to illustrate the advantage of fault recovery mechanism, and the effectiveness of ACO.

Some future research can be carried out. In our paper, it is assumed that failures occurring on both nodes and links satisfy Poisson processes, which is not true in all cases. For example, some software may have periodic updates, while some software systems running continuously for a long time may tend to show degraded performance [37]. Thus, some other random processes may also be implemented. Furthermore, the proposed model assumes that the grid has one RMS, and can be viewed as a star topology, which sometimes is impractical for the complexity of the grid. Also, it is assumed that one subtask can be allowed to be assigned at only one node due to the applicability of the original ACO algorithm in the scheduling. Thus, some modifications of the original ACO algorithm may be further made if they are cost effective. Finally, the price-setting of the grid resources in a market-oriented environment also needs substantial attention in future research.

ACKNOWLEDGMENT

The authors would like to thank three anonymous referees and the Associate Editor for their comments on an earlier version of this paper.

REFERENCES

[1] I. Foster, "The Grid: A new infrastructure for 21st century science," *Physics Today*, vol. 55, no. 2, pp. 42–47, 2002.
[2] 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.
[3] S. C. Guo, H. Wan, G. B. Wang, and M. Xie, "Analysis of grid resource compensation in market-oriented environment," *Eksploatacja I Niezawodnoœæ—Maintenance and Reliability*, vol. 45, no. 2, pp. 36–42, 2010.
[4] K. Krauter, R. Buyya, and M. Maheswaran, "A taxonomy and survey of grid resource management systems for distributed computing," *Software—Practice and Experience*, vol. 32, no. 2, pp. 135–164, 2002.
[5] C. L. Li and L. Y. Li, "Multiple QoS modeling and algorithm in computational grid," *Journal of Systems Engineering and Electronics*, vol. 18, no. 2, pp. 412–417, 2007.
[6] 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.
[7] 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.
[8] 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.
[9] 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.
[10] T. Paul and X. Jie, "Fault tolerance within a grid environment," in *Proceedings of UK e-Science All Hands Meeting*, 2003.
[11] 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.
[12] 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.
[13] K. Jozsef and K. Peter, "A migration framework for executing parallel programs in the grid," in *Proceedings of European across Grids Conference*, 2004.
[14] L. Xing and Y. S. Dai, "A new decision diagram model for efficient analysis on multi-state systems," *IEEE Trans. Dependable and Secure Computing*, vol. 6, no. 3, pp. 161–174, 2009.
[15] A. Heddaya and A. Helal, Reliability, Availability, Dependability and Performability: A User-Centered View 1997, Technical Report.
[16] 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.
[17] 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.
[18] M. Gen and Y. S. Yun, "Soft computing approach for reliability optimization: State-of-the-art survey," *Reliability Engineering and System Safety*, vol. 91, no. 9, pp. 1008–1026, 2006.
[19] P. Y. Yin, S. S. Yu, P. P. Wang, and Y. T. Wang, "Multi-objective task allocation in distributed computing systems by hybrid particle swarm optimization," *Applied Mathematics and Computation*, vol. 184, no. 2, pp. 407–420, 2007.
[20] D. Coit and A. Smith, "Reliability optimization of series-parallel systems using genetic algorithm," *IEEE Trans. Reliability*, vol. 45, no. 2, pp. 254–260, 1996.
[21] Y. C. Liang and A. E. Smith, "An ant colony optimization algorithm for the redundancy allocation problem (RAP)," *IEEE Trans. Reliability*, vol. 53, no. 3, pp. 417–423, 2004.
[22] H. Z. Huang, J. Qu, and M. J. Zuo, "Genetic-algorithm-based optimal apportionment of reliability and redundancy under multiple objectives," *IIE Transactions*, vol. 41, no. 4, pp. 287–298, 2009.

[23] H. Z. Huang, M. J. Zuo, and Z. Q. Sun, "Bayesian reliability analysis for fuzzy lifetime data," *Fuzzy Sets and Systems*, vol. 157, no. 12, pp. 1674–1686, 2006.
[24] H. Z. Huang, Z. Tian, and M. J. Zuo, "Intelligent interactive multiobjective optimization method and its application to reliability optimization," *IIE Transactions*, vol. 37, no. 11, pp. 983–993, 2005.
[25] C. Lu, H. Z. Huang, J. Y. H. Fuh, and Y. S. Wong, "A multi-objective disassembly planning approach with ant colony optimization algorithm," *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacturing*, vol. 222, no. 11, pp. 1465–1474, 2008.
[26] I. Foster, C. Kesselman, and J. M. Nick, "Grid services for distributed system integration," *Computer*, vol. 35, no. 6, pp. 37–46, 2002.
[27] M. Xie, *Software Reliability Modeling*. New York: World Scientific Publishing Company, 1991.
[28] E. P. C. Kao, *An introduction to Stochastic Processes*. Belmont: Wadsworth Publishing Company, 1997.
[29] D. Kalyanmoy, *Multi-Objective Optimization Using Evolutionary Algorithms*. New York: John Wiley & Sons, 2001.
[30] H. A. Taboada, F. Baheranwala, D. W. Coit, and N. Wattanapongsakorn, "Practical solutions for multi-objective optimization: An application to system reliability design problems," *Reliability Engineering and System Safety*, vol. 92, no. 3, pp. 314–322, 2007.
[31] R. T. Marler and J. S. Arora, "The weighted sum method for multi-objective optimization: new insights," *Structural and Multidisciplinary Optimization*, vol. 41, no. 6, pp. 853–862, 2010.
[32] K. A. Proos, G. P. Steven, O. M. Querin, and Y. M. Xie, "Multicriterion evolutionary structural optimization using the weighted and the global criterion methods," *AIAA Journal*, vol. 39, no. 10, pp. 2006–2012, 2001.
[33] M. Dorigo and C. Blum, "Ant colony optimization theory: A survey," *Theoretical Computer Science*, vol. 344, no. 2-3, pp. 243–278, 2004.
[34] G. D. Caro and M. Dorigo, "AntNet: distributed stigmergetic control for communications networks," *Journal of Artificial Intelligence Research*, vol. 9, no. 2, pp. 317–365, 1998.
[35] D. Merkle, M. Middendorf, and H. Schmeck, "Ant colony optimization for resource-constrained project scheduling," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 4, pp. 333–346, 2002.
[36] V. Maniezzo, "Exact and approximate nondeterministic tree-search procedures for the quadratic assignment problem," *INFORMS Journal on Computing*, vol. 11, no. 4, pp. 358–369, 1999.
[37] M. Grottke and K. S. Trivedi, "Fighting bugs: Remove, retry, replicate, and rejuvenate," *IEEE Computer*, vol. 40, no. 2, pp. 107–109, 2007.

Suchang Guo is currently an Engineer in No.30 Institute of China Electronics Technology Group Corporation, Chengdu, Sichuan, 610041, China. He received a Ph.D. degree in Mechatronics Engineering from University of Electronic Science and Technology of China, China. His research areas include software reliability modeling, trusted networks, and distributed and Grid systems.

Hong-Zhong Huang (M'10) is a Professor and the Dean of the School of Mechanical, Electronic, and Industrial Engineering, University of Electronic Science and Technology of China. He has held visiting appointments at several universities in the USA, Canada, and Asia. He received a Ph.D. degree in Reliability Engineering from Shanghai Jiaotong University, China and has published 150 journal papers and 5 books in fields of reliability engineering, optimization design, fuzzy sets theory, and product development. He is a fellow of ISEAM (International Society of Engineering Asset Management), and a member of ESRA (European Safety and Reliability Association) Technical Committee on System Reliability, a Regional Editor of International Journal of Reliability and Applications, an Editorial Board Member of International Journal of Reliability, Quality and Safety Engineering, International Journal of Quality, Statistics, and Reliability, International Journal of Reliability and Quality Performance, Advances in Fuzzy Sets and Systems, and The Open Mechanical Engineering Journal. He received the William A. J. Golomski Award from the Institute of Industrial Engineers in 2006, and the Best Paper Award of the 8th International Conference on Frontiers of Design and Manufacturing in 2008. His current research interests include system reliability analysis, warranty, maintenance planning and optimization, computational intelligence in product design.

Zhonglai Wang is an Assistant Professor in the School of Mechanical, Electronic and Industrial Engineering, University of Electronic Science and Technology of China. He has been a visiting scholar in the Department of Mechanical and Aerospace Engineering, Missouri University of Science and Technology from Sept. 2007 to Sept. 2008. He received a Ph.D. degree in Mechatronics Engineering from University of Electronic Science and Technology of China, China. His research interests include reliability-based design, robust design, and life cycle reliability-based design.

Min Xie (M'91–SM'94–F'06) received his Ph.D. in Quality Technology in 1987 from Linkoping University in Sweden. He was awarded the prestigious LKY research fellowship in 1991, and currently he is a Professor at National University of Singapore. He has authored or co-authored numerous refereed journal papers, and six books on quality and reliability engineering, including Software Reliability Modelling by World Scientific Publisher, Weibull Models by John Wiley, Computing Systems Reliability by Kluwer Academic, and Advanced QFD Applications by ASQ Quality Press. He is an Editor of International Journal of Reliability, Quality and Safety Engineering, Department Editor of IIE Trans., Associate Editor of IEEE TRANS. RELIABILITY, and on the editorial board of a number other international journals. He is a fellow of IEEE.