Available online at www.sciencedirect.com
ScienceDirect
Future Generation Computer Systems 23 (2007) 209–218
FGCS FUTURE GENERATION COMPUTER SYSTEMS
www.elsevier.com/locate/fgcs

Optimal task partition and distribution in grid service system with common cause failures

Yuan-Shun Dai[a,*], Gregory Levitin[b], Xiaolong Wang[a]

[a] *Department of Computer and Information Science, Purdue University, Indianapolis, IN 46202, USA*
[b] *The Israel Electric Corporation Ltd., Reliability Department, P.O. Box 10, Haifa 31000, Israel*

Received 25 January 2006; received in revised form 2 May 2006; accepted 3 May 2006
Available online 27 June 2006

Abstract

The grid service system is a type of large-scale distributed system, which is being widely used in many fields now. The partition of a grid service task into subtasks and the distribution of them on available resources have great influence on the extent of the service reliability and profits. This paper presents a novel optimization model for maximizing the expected grid service profit and develops a genetic algorithm to solve this optimization problem. As the basis of the objective function, the grid service reliability needs to be modeled and quantified first. However, due to the largeness and complexity of the grid service system, the existing models for small-scale distributed system reliability cannot be directly implemented for the grid. Therefore, this paper presents a virtual model with two-root tree structure that is more general than the star topology. It is because the tree structure not only can represent the virtual architecture of a grid service but also can take into account the common cause failures on the channels that are shared by multiple resources. Finally, a case of a BioGrid application is illustrated for example.

Keywords: Grid computing; Grid economy; Service reliability; Optimization; Genetic algorithm

1. Introduction

"Grid" computing systems are emerging as an important new field, different from conventional distributed computing systems by its focus on large-scale resource sharing, innovative applications, and, in some cases, high-performance orientation, e.g. [13,17]. The real and specific problem that underlies the Grid concept is to coordinate the shared resources and to solve problems through distributed programs [11]. The sharing that the grid computing is concerned with is not primarily file exchange but rather direct access to computers, software data, and other resources, as is required by a range of collaborative problem-solving and resource-brokering strategies.

The Open Grid Services Architecture (OGSA) enables the integration of services and resources across distributed, heterogeneous, dynamic, virtual organizations—whether within a single enterprise or extended to external resource-sharing and service-provider relationships [13]. Various grid services can be offered under the grid environment, which is defined as a web service that provides a set of well-defined interfaces and that follows specific conventions, see e.g. [12,24].

When the open grid system receives a user's request for a certain service, the resource management system (RMS) starts seeking available resources, then partitions the task into multiple subtasks and assigns them to those detected resources. Many services offered by the Grid need to access data from a certain source (database), such as the BioMap service using the grid system to identify the genes from open databases [8]. Marović and Jovanović [24] showed another example for a web-based grid service, in which the grid resources need to access visualization data from another remote server running on the grid. The RMS, the computational resources and the data source constitute a general model that can cover most grid services.

The partition of a service task into subtasks and their distribution among available resources are of great concern, because they significantly affect the grid service reliability,

Abbreviations: DMS, Data Management Server; MRST, Minimal Resource Spanning Tree; RMS, Resource Management System; RST, Resource spanning tree; VL, Virtual link; VN, Virtual node; VO, Virtual organization.

* Corresponding author. Tel.: +1 317 274 3473.
E-mail address: ydai@cs.iupui.edu (Y.-S. Dai).

0167-739X/$ - see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.future.2006.05.002

Notations

b	The penalty of a failed grid service,
c_i	The expected cost to use resource r_i,
d	Listed price of a service,
$element_i$	The i-th element (in VN or VL) in the MRST,
E_j	The event that $MRST_j$ successfully executes the given program,
G_j	The j-th VN in the MRST,
r_i	The i-th resource,
N_t	The total number of MRST's for a given service,
$R_c(j)$	Reliability function of the j-th VN,
$R_L(i, j)$	Reliability function of the $L(i, j)$,
R_{MRST}	Reliability of the MRST,
R_S	Grid service reliability,
$R(\sigma, \theta)$	Grid service reliability given a specific partition and distribution,
t_i	The processing time of the resource r_i,
$T(j)$	The total communication time of the j-th VN,
$T_w(element_i)$	Operating time of the i-th element,
v_i	Extra expense per time unit charged by the resource r_i,
λ_j	The failure rate of the resource j,
θ_j	Percentage of workload assigned on the j-th subtask,
σ_i	Subtask number assigned to the resource r_i,
τ_j	The allowed processing time by paying c_i on the resource r_i.

cost and profits, see e.g. [1]. C. Li and L. Li [22] described a common grid service model that allowed agents representing various grid resources, which were owned by different real world enterprises. The grid task agents buy resources to complete tasks. Grid resource agents charge the task agents for the amount of resource capacity allocated. In the meantime, the grid task agents charge users who requested the service. Buyya et al. [2] described the economical opportunities and realizations through Grid services. They identified the challenges and requirements of economy-based Grid systems, and discussed various representative systems. C. Li and L. Li [22] and Buyya et al. [3] also introduced the optimal task/resource scheduling problems and showed the significant improvement by a good schedule strategy. Some other optimization schemes, proposed for grid or cluster, include [28,14,26]. However, none of them consider the reliability factor when solving the optimization problems. In fact, service reliability significantly affects the profit, which can be viewed as the *risk cost*, i.e., if certain tasks cannot be successfully finished or wrong results are offered to the users, the service providers cannot earn money, rather, they may pay some penalty to compensate the users' loss. Thus, when the economy is studied in the grid, the service reliability should not be ignored. Therefore, this paper attempts to seek the optimal task partition and allocation on grid resources in order to maximize the economic profit of a grid service considering the effect of the reliability as well.

The reliability of grid services needs to be modeled and quantified, which contributes to the calculation of the service profit. There are many studies for the conventional small-scale distributed system reliability. However, they cannot be directly implemented in the reliability analysis of grid services due to the challenges of the grid's complexity and size [27]. For instance, Kumar et al. [20] presented the definition of Distributed Program/System Reliability in which they assumed that the probabilities for links and nodes to be operational were constant. The follow-up research, e.g. Chen and Huang [6], Kumar and Agrawal [19], Chen et al. [5], Lin et al. [23], and Dai et al. [9], continued the study based on the Kumar et al.'s [20] model. These models had thus inherited the common assumption that the operational probabilities of nodes and links are constant. This assumption might be approximately acceptable for small-scale distributed systems where the communication time could be neglected. However, for the grid computing, the operational probabilities or reliability of the nodes and links are not constant due to the largeness of the grid.

To solve the challenges, this paper develops a new virtual model with two-root tree structure for representing the grid service reliability in a more general manner, and then an algorithm based on Graph Theory and Bayes Theorem is developed for deriving the service reliability in an efficient way. After that, the optimization model for maximizing the expected profit of a grid service is built up. A genetic algorithm is adapted to solve this optimization problem.

The rest of this paper is organized as follows. Section 2 describes the optimization problem for task partition and distribution of grid service, and a genetic algorithm is presented to solve the problem. Section 3 presents the approaches to model and derive the grid service reliability that is the basis of the optimization problem. Section 4 shows a numerical case of the BioMap project to illustrate the procedures and applicability of the approaches.

2. Grid service and optimal task allocation

2.1. Grid computing with resource and data management

Grid computing system is a type of large-scale distributed system with the underlying objective for resource sharing and collaborative computing. Since the grid is large and complex, virtualization becomes necessary for modeling and measuring the grid. Foster et al. [10] showed that the virtualization enabled consistent resource access across multiple heterogeneous platforms. The virtual organization (VO) of the grid was initially discussed by Foster et al. [13]. Actually, the VO is made up of virtual nodes (VNs). The VN is a general unit integrated in the grid, which can execute programs or share resources, such as a computer, a processing element, an embedded system, a digital product, etc. VNs are connected with one another by the virtual links (VLs). The VL is a general connection between two VNs, which may contain various elements, such as routers, computers, the Internet, wireless communication channels, and so on. Under such virtualization, a general architecture of grid computing systems is depicted in Fig. 1.

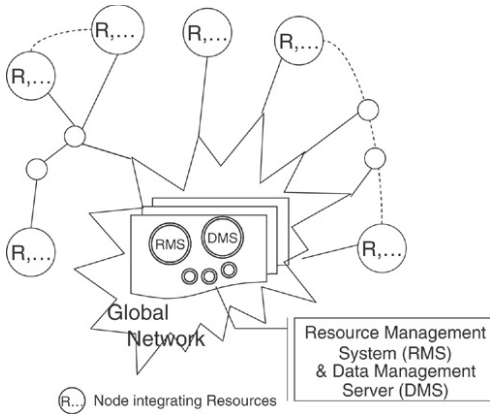


Fig. 1. Grid computing system.

The Open Grid Services Architecture [12] makes applicable the integration of services and resources by different organizations. Any users who are accessible to the grid can request those services. The sites that contain the resources are linked by the network (as Fig. 1). Each resource represents a general unit that can be a software program, a hardware equipment or a firmware product, etc.

RMS (Resource Management System), the "brain" of a grid computing system; see e.g. [17], manages all resources in the grid. Each service, offered to the grid from a certain organization, has a record at RMS, including price, description, parameters, return values and requirements that are needed.

Suppose any user of the grid sends a service request to the RMS for getting a certain service. Then, the RMS evokes a set of jobs (subtasks) to complete this service request. Each subtask is assigned to execute a portion of the service/task. These subtasks are assigned to different nodes (resources) by RMS. To reach the objectives of high service reliability and full utilization of grid resources, some subtasks may have redundancies executed by the grid. There is often a data source which manages necessary data access for a processing node. The data source can sometimes be combined with the RMS, but it can also be separated from the RMS, such as a database server. The latter model is more general than simply assuming the RMS to store all data for various grid services, because the major functions of RMS focus on the partition and distribution of the jobs/programs to the resources and other data (especially large datasets) can be directly accessed from another database during the execution of the program. When these subtasks are completed using those remote resources, the results are returned to the RMS and then the users get the final results organized by the RMS.

In this case, each resource (node), when executing jobs, communicates with not only the RMS but also the Data Management Server (DMS). Because of the distributed nature of the grid, the service task allocation problem consists of dividing the task into subtasks with corresponding percentage of workload, and assigning all necessary resources to execute the subtasks evoked by a service request. How to find the optimal solution for partition and assignment is the purpose of this section.

2.2. Modeling for grid task partition and distribution

Assumptions

(1) A grid service needs to be managed by a Resource Management System (RMS) and to access data from a Data Management Server (DMS). N remote resources, denoted by r_i $(i = 1, 2, \ldots, N)$, are available for this grid service.
(2) If the RMS receives a service request, it recognizes/translates the request into a concrete task. The entire task can be divided arbitrarily into J subtasks by the RMS, and each subtask can take a percentage of the workload of the service. The workload for processing and the data for communication are proportional to the percentage of the assignment on the subtask. Then, these subtasks are distributed over P $(P \leq N)$ resources for execution.
(3) Each available resource is able to execute one subtask. Different resources start performing their subtasks immediately after they get the assignment from the RMS. Those resources can perform their subtasks simultaneously and may request data from the DMS. The data transmission from the DMS does not start until the subtask assignment to the corresponding resources is completed by the RMS.
(4) If a site contains several resources (multi-processors) or several resources in the same LAN that communicates with the outside via the same gateway, it can be viewed as several resource nodes connected to a common virtual node which communicates with the RMS/DMS. In this case, these resources share common communication channels to the RMS/DMS.
(5) If any failure occurs on a resource or communication channel before the end of output data returned from the resource to the RMS, the subtask fails. The failure arrivals on resources or links are governed by the Poisson processes.
(6) The failure rates of the channels or resources are the same when they are either idle or loaded (hot standby mode). The failures at different resources and communication links are independent.
(7) If a subtask is processed by several resources, it is completed when the first result returns to the RMS. The task is completed when all of the subtasks are achieved and their results are returned to the RMS from the resources.
(8) The RMS and DMS are fully reliable for simplification which will not affect the optimization solutions. Actually, it is rather easy to integrate the failures of RMS and DMS because they can be viewed as two modules that are connected to the rest in series.

Optimization model

From the viewpoints of service providers, they aim at achieving high reliability and maximal profit from their provided services. One of the most important stages of offering a highly reliable/beneficial service is the service design stage. A service needs to access different grid resources that should charge the customers for the amount of resource capacity allocated [22]. It is inadvisable to assign too few resources to execute the task, which may increase the risk cost due to the low reliability. It is also inadvisable to consume too many

resources to execute the same subtasks in parallel, even though it can improve the service reliability. Our model is proposed to set an optimal partition and distribution of the subtasks for a service to achieve the objective that maximizes the overall profit considering the reliability.

When requesting the grid service, users pay a certain amount of money such as a listed price (d). When the service is completed successfully, the provider can make money. The service provider should also pay the expense for recruiting the online resources offered by other agents. The expected cost to use resource r_i is denoted by c_i given the usage time less than an initial period τ_j. If the time to occupy a resource exceeds the τ_j, extra money per unit of time is charged, denoted by v_i. In some cases, the service provider has to compensate the users for the failure of the service as listed on the SLA (Service Level Agreement). Suppose the penalty is b, then the expected risk cost for failed service is $b \cdot (1 - R_s)$. In order to maximize the expected profit, the optimization model for task partition and distribution is:

Decision variables

(1) Partition variable: $\theta = \{\theta_1, \theta_2, \ldots, \theta_J\}$ where θ_j represents the percentage of workload assigned on the j-th subtask.
(2) Distribution variable: $\sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_N\}$ where the integer $\sigma_i \in [0, J]$, denotes the subtask number assigned to the resource r_i and if $\sigma_i = 0$ means no subtask is assigned to the resource r_i.

Objective

Maximize $\text{Profit}(\sigma, \theta)$

$$= d \cdot R(\sigma, \theta) - \sum_{i=1}^{N} \{c_i + v_i \cdot [t(\sigma_i) - \tau_i]\} \cdot 1(\sigma_i) - b \cdot [1 - R(\sigma, \theta)] \quad (1)$$

where $1(\sigma_i) = \begin{cases} 0 & \sigma_i = 0 \\ 1 & \sigma_i \neq 0 \end{cases}$, and $t(\sigma_i) = \begin{cases} \tau_i & t_i \leq \tau_i \\ t_i & t_i > \tau_i \end{cases}$. in which t_i is the processing time of the resource r_i.

Subject to:

$$0 \leq \theta_j \leq 1, \quad (j = 1, 2, \ldots, J)$$

$$\sum_{j=1}^{J} \theta_j = 1$$

$$0 \leq \sigma_i \leq J, \quad (i = 1, 2, \ldots, N).$$

Encountering service failure, the provider cannot charge users, but it is possible not to compensate extra penalty to the user sometimes. Under this condition, b can be set to 0, which does not affect the generality of Eq. (1). The optimization problem is to find the best partition θ and distribution σ to maximize the total profit.

In Eq. (1), the service reliability function $R(\sigma, \theta)$ should be derived for calculating the risk cost as a part of the expected profit. However, it is hard to evaluate the grid service reliability without an appropriate model. Therefore, a novel model suitable for grid service reliability is required to be built. In the next Section 3, the grid service reliability will be modeled based on the specificities of the grid computing and then the algorithms for deriving the $R(\sigma, \theta)$ are developed.

2.3. Genetic algorithm

Grid service reliability $R(\sigma, \theta)$ can be obtained given the partition into subtasks θ and the distribution of subtasks on the resources σ (more details will be given in the next Section 3). Then, the objective function (1) of the optimization model is derivable. With this objective function, a genetic algorithm can be applied to optimize this subtask partition and distribution problem. A genetic algorithm is a stochastic optimization technique, which is a rapidly growing area of artificial intelligence. Genetic algorithms are inspired by Darwin's evolution theory based on the survival of the fittest species as introduced by Holland [16] and Goldberg [15]. The genetic algorithms have been widely applied in many fields of Reliability Engineering, see e.g. [7,21]. The following is our genetic algorithm designed for task partition and distribution problem.

Encoding. A chromosome should represent two basic elements of task partition and subtask distribution. The first one is a simple binary vector σ. Each tuple σ_i ($i = 1, 2, \ldots, N$) represents the subtask number assigned to the resource r_i. The vector length depends on the number of subtasks and the number of resources. Length $(\sigma_i) = \lceil \log_2 J \rceil$, where J is the number of subtasks and $\lceil X \rceil$ is the ceiling function to return the minimum integer not less than X. The second part is the percentage of workload assigned on the corresponding subtask. We used 8 bits to represent the real value of the percentage, so the total length of each chromosome is $N \cdot \lceil \log_2 J \rceil + 8 \cdot J$.

Fitness function. The evaluation of chromosomes is a critical part of the optimization procedure. The fitness function used in this problem is derived from

$$Fitness = a \cdot \exp\{b \cdot profit(\sigma, \theta)\}$$

where the $profit(\sigma, \theta)$ is derivable from Eq. (1) and a, b are two positive constants.

Crossover. Since we have chromosomes consisting of two lists of binary code representing the subtask assignment and input data size, to keep the high-performance segment in both we have to treat them separately. The most popular solution for this problem has been the use of two-point crossover. This operator is like one-point crossover, except that two cut points rather than one are selected in both lists at random, and chromosomal material is swapped separately. The frequency of crossover operators is controlled by the crossover rate that is usually a large value between 0.85 and 0.95.

Mutation. We adopt the same mutation operation that is widely used: given a mutation rate, do the mutation on a randomly selected bit to change from 0(1) to 1(0). However, current research has shown that combinational problems such as job assignment problems need high mutation rate to keep population diversity. Therefore, the mutation rate needs to be increased in this problem within the range 0.1–0.3.

Elitism. Note that the best member of the population may fail to produce offspring in the next generation. The *elitism* strategy fixes this potential of loss by copying the best member of each generation into the succeeding generation.

In summary, the generic algorithm is outlined as follows.

1. Randomly generate N binary strings (chromosomes) for the initial population.
2. Evaluate the fitness of each chromosome and use Roulette Wheel Selection strategy to select parents based on their fitness.
3. The chromosomes of the next generation are formed by applying crossover and mutation operators.
4. Continue the cycle initialized in step 2 until the termination criterion is satisfied. That is the best chromosome remains unchanged for a certain number of generations or the number of generations achieves the maximum.
5. Take the best solution and output the subtask partition and distribution.

3. Grid service reliability modeling and algorithms

The above optimization model in Eq. (1) requires the derivation of grid service reliability $R(\sigma, \theta)$ given a specific partition and distribution of subtasks among grid resources. This section presents a virtual tree structure model representing the grid service and algorithms for the derivation of service reliability.

3.1. Virtual structure of two-root tree for modeling grid service

Today, the Grid computing systems are becoming increasingly large and complex, e.g. the IP-Grid (Indiana–Purdue Grid), developed by Indiana University and Purdue University, is a state-wide grid in the US (http://www.ip-grid.org/). The size and complexity challenge the existing models and tools to analyze, evaluate, predict and optimize the reliability of grid systems. Even though all online nodes or resources are linked through the Internet with one another, not all resources or communication channels are actually used for a specific service. According to this observation, we can make the modeling and analysis more tractable for grid reliability by presenting a virtual structure corresponding to a specific grid service.

In the grid service, the start and the terminal points are identical, at the RMS. The service requires a set of resources that are distributed in the grid. During the service period, those resources may request some data (especially large datasets) from a DMS. The DMS can be a part of RMS, but can also be a separate site for generality. Several papers [18,4] suggested using star topology to simplify the grid structure. In the star grid, the RMS/DMS is connected with each resource by one direct communication channel (link). However such an approximation is not accurate even though it simplifies the analysis and computation. For example, several resources located in the same LAN can use the same gateway/router to communicate with the outside. Therefore, all these resources are not connected with the RMS/DMS through independent links. The resources are connected to the gateway which communicates with RMS/DMS through a common communication channel. Another example is a server that contains several resources (has several processors that can run different applications simultaneously). Such a server communicates with the RMS/DMS through the same links. These scenarios cannot be modeled by using only star-topology grid architecture.

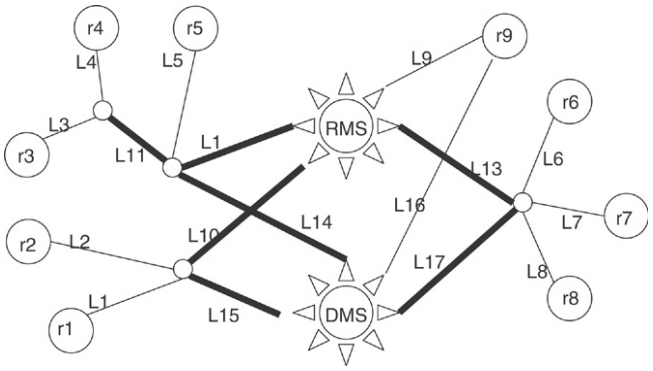


Fig. 2. Grid service topology with resource and data management.

Hereby, we present a more general virtual structure which has a tree topology with two roots. The first root of the virtual tree structure is the RMS, the other root is the DMS, and the leaves are resources, while the branches of the tree represent the communication channels linking the leaves and the root. Some channels are used in common by multiple resources. An example of the two-root tree topology is given in Fig. 2 in which 9 resources $(r_1, r_2, \ldots, r_9)$ are available for a service.

The tree structure allows the common cause failures in shared communication channels to be modeled. For example, in Fig. 1, the failure of the channel L13 makes resources r_6, r_7 and r_8 unable to communicate with the RMS simultaneously. This type of common cause failure was ignored by conventional parallel computing models or star-topology models. For small-area communication (in a LAN or a cluster), such an assumption that ignores the common cause failures on communications might be plausible, because the communication time is negligible compared to the processing time. However, for wide-area communication, such as the grid system, it is more likely to have failures on communication channels for relatively long-time data transmissions, so the communication time and link failures cannot be neglected. In many cases, the communication time is comparable to or even dominates the processing time due to the large sets of data transmitted via the Internet. Thus, the virtual 2-root tree is an adequate model representing the functioning of grid services.

The amount of data that should be transmitted between a resource j that processes a subtask i and RMS (DMS) is denoted by $a_i(\beta_i)$ respectively. If data transmission between the RMS/DMS and the resource j is accomplished through links belonging to a set γ_j, the data transmission speed is

$$s_j = \min_{x \in \gamma_j}(b_x) \tag{2}$$

where b_x is the bandwidth of the link L_x. Therefore, the reliability for the resource to successfully complete the subtask

i should be

$$R_r(i) = \exp\left(-\lambda_j\left(\tau_j + \frac{a_i}{s_j} + \frac{\beta_i}{w_j}\right)\right) \tag{3}$$

where λ_j is the failure rate of the resource j, and w_j is the bandwidth between the resource j and the DMS, which can be obtained similarly to Eq. (2). Due to the hot standby assumption, the resource has to be working through the whole process until the data is sent back to RMS. The reliability for the channel between the resource j executing the subtask i and RMS is

$$R_c(i, j) = \exp\left(-\pi_j\left(\tau_j + \frac{a_i}{s_j} + \frac{\beta_i}{w_j}\right)\right) \tag{4}$$

where π_j is the failure rate for the channel and the working time can be explained by the same reason as Eq. (3). The reliability for the channel between the same resource and the DMS is

$$R_d(i, j) = \exp\left(-\eta_j\left(\frac{\beta_i}{w_j}\right)\right) \tag{5}$$

where η_j is the failure rate for the channel, but the working time is only during the period for data transmission because the connection is not built up till the resource needs the data from DMS. Also, suppose the required data is transmitted all at once from DMS, which is the most efficient way. (Note: only the time needs to be adjusted accordingly if some other types of data transmission with the DMS are used.)

Based upon the above description of the grid computing systems, the definitions of subtask reliability and service reliability are:

Grid subtask reliability: Grid subtask reliability is defined as the probability to successfully execute a given subtask that runs on one or multiple resources and exchanges information with the RMS and the DMS, under the grid computing environment.

Grid service reliability: Grid service reliability is defined as the probability for at least a set of the subtasks required in the considered grid service to be completed successfully.

3.2. Algorithm for computing grid service reliability

To compute the reliability of the grid services, we implement the graph theory plus Bayesian approach. The concept of MRST (Minimal Resource Spanning Tree) is introduced first, and then the formulas for grid service reliability are derived.

Minimal resource spanning tree (MRST)

Generally, the set of nodes and links involved in running a given set of programs and exchanging information with the resources form a tree diagram. There are many such tree diagrams, some of which may dominate some others. The grid service reliability is determined by the reliability of these trees. In order to effectively implement those trees to analyze the reliability, the RST and MRST are defined.

Resource spanning tree (RST): A spanning tree connecting the RMS and DMS for executing the given service under consideration, such that its vertices hold all the required resources for exchanging information.

Minimal resource spanning tree (MRST): For a given service, an MRST_i is an RST_i such that there exists no other resource spanning tree, say RST_j, which is a subset of RST_i, i.e., $\neg(\exists j)\mathrm{RST}_j \subseteq \mathrm{RST}_i$.

The reliability of an MRST is defined as the probability for the MRST to be operational to execute the given service, which can be derived from

$$R_{\mathrm{MRST}} = \prod_{i\in\mathrm{MRST}} R(element_i) = \prod_{i\in\mathrm{MRST}} \exp\{-\lambda(element_i) \times T_w(element_i)\} \tag{6}$$

where the $R(element_i)$ can be calculated from Eqs. (3) to (5). With Eq. (6), the reliability of an MRST can be computed if the working time (T_w) and failure rate (λ) of all the elements are known. Hence, finding all the MRST's and determining the working time of their elements are the first step in deriving the grid service reliability.

According to the specificity of the two-root tree and the definition of MRST, it is efficient to obtain all the MRSTs as follows. The combinations of the resources that run a set of subtasks form the MRSTs. The links between corresponding resources and RMS/DMS should also be included in each MRST. The working time for resources and communications can be calculated given the amount of data needed by a subtask, the processing speed of a resource, and the bandwidth for communication.

Grid service reliability

Note that failures of all the MRSTs lead to the failure of the given service, and any one of the MRST's can successfully complete the service if all of its elements are good. The grid service reliability of a given set of subtasks can be described as the probability of having at least one of the MRST's operational,

$$R(P_m) = \Pr(\text{at least one MRST of a given service is reliable}).$$

Let N_t be the total number of MRST's and E_j be the event in which the MRST_j is reliable to successfully complete the given service, $(j = 1, 2, \ldots, N_t)$. Thus, the grid service reliability can be written as

$$R_s = \Pr\left(\bigcup_{j=1}^{N_t} E_j\right). \tag{7}$$

By using the concept of Bayesian conditional probability, the events considered in (7) can be decomposed into mutually exclusive events as

$$R_s = \Pr(E_1) + \Pr(E_2)\cdot\Pr(\overline{E}_1|\overline{E}_2)\cdots + \Pr(E_{N_t}) \times \Pr(\overline{E_1}, \overline{E_2}, \cdots \overline{E}_{N_t-1}|E_{N_t}) \tag{8}$$

where $\Pr(\overline{E_1}|E_2)$ denotes the conditional probability that MRST_1 is in the failure state given that MRST_2 is in the operational state.

Hence, the grid service reliability can be evaluated in terms of the probability of two distinct events. The first event indicates that the MRST_i is in the operational state while the

Table 1
Characteristics of resources

Resources	r_1	r_2	r_3	r_4	r_5	r_6	r_7	r_8	r_9
Proc. speed: (KB recs/s)	21	18	19	20	15	13	19	20	15
Initial cost: ($)	12.5	11	9.5	13.5	10	9	14.5	10.5	20
Initial period (s)	11	17	13	15	17	19	21	23	25
Extra cost: ($/s)	2	2.5	3.5	3	2.5	2	1.5	1	0.8
Failure rate $\times 10^{-3}$ s^{-1}	1.7	2.6	1.3	1.4	1.5	2.6	1.8	2	1.5

second indicates that all of its previous trees MRST_j ($j = 1, 2, \ldots, i-1$) are in the failure state given that MRST_i is in the operational state. The probability of the first event, $\Pr(E_i)$ is straightforward, which can be calculated through Eq. (6). The probability of the second event, $\Pr(\overline{E}_1 \Lambda \overline{E}_2 \Lambda \cdots \overline{E}_{i-1}|E_i)$, can be computed using the following Algorithm 2.

Algorithm 2 identifies all the conditional elements that can lead to the failure of any MRST_j ($j = 1, 2, \ldots, i-1$) while keeping MRST_i to be operational. Such an identified conditional element, say $element_k$ (contained in any MRST_j, $j = 1, 2, \ldots, i-1$), has starting time and end time. To keep MRST_i operational, its starting time must be greater than the working time of $element_k$ in MRST_i (if MRST_i does not contain $element_k$, consider its working time to be 0). If any failure occurs on the $element_k$ between its starting time and end time, it can lead one MRST_j to fail while MRST_i is operational. Then using a binary search tree, Algorithm 3 seeks the possible combinations of these identified elements that can make all the MRST_j ($j = 1, 2, \ldots, i-1$) fail, and computes the probabilities of those combinations. The sum of these probabilities is the result of $\Pr(\overline{E}_1 \Lambda \overline{E}_2 \Lambda \cdots \overline{E}_{i-1}|E_i)$. More detailed procedures and Pseudo Code of the algorithms are given in Appendix.

After computing all the $\Pr(\overline{E}_1 \Lambda \overline{E}_2 \Lambda \cdots \overline{E}_{i-1}|E_i)$ and $\Pr(E_i)$ ($i = 1, 2, \ldots, N_t$), we can calculate the grid service reliability $R(\sigma, \theta)$ by substituting them into Eq. (8). In the meantime, the processing time for all resources can also be obtained. Thus, the objective function (1) of the optimization model presented in Section 2 can be calculated, and following that the Genetic Algorithm is able to be implemented.

4. Illustrative examples

A project named BioMap is being done at Indiana University, which implements the grid computing system in solving some bioinformatics problems. Data in the bioinformatics research area has gone beyond human capability to comprehend effectively due to its sheer volume. With more than 15 million literatures publicly available at MEDLINE/PubMed (http://www.ncbi.nlm.nih.gov), it is just impossible for a human to extract information and gain a complete knowledge, see e.g. [25]. BioMAP (Biomedical Association and Pathways) is an intelligent biomedical literature mining system that provides biologists with new hypotheses that may lead to a new discovery, see e.g. [8]. By using the entire MEDLINE/PubMed literature collection, along with other well-known biological databases (Uniprot, Unigene, UMLS, Gene Ontology), BioMap has become a knowledge-base data warehouse. It finds not only contextual information about a specific gene or protein, but also discovers how those genes and/or proteins interact with other object(s). There are several steps involved in the knowledge discovery process, such as biological object identification, synonym resolution (several names for the same object), association and directionality finding. Biological object identification, which has been the most time consuming process is considered as a case study in this paper.

A grid system has been developed for the BioMap project, e.g. [8], where the biological database of Unigene is analyzed first. Thus, the RMS will call the available grid nodes/resources to work for the complicated discovery process on different records in the Unigene. After those nodes receive the subtasks assigned by the RMS, they will start working, during which the data in the Unigene database will be read by those nodes, respectively. Thus, the Unigene database becomes the DMS in our model of Fig. 2.

This experiment possesses the same two-root tree as Fig. 2 in Section 2 for example. The virtual structure for a grid service involves 9 nodes/resources ($r_1, r_2, \ldots, r_9$) with the RMS and DMS. RMS evokes at most 3 subtasks for a service request, denoted by J_1, J_2 and J_3. Suppose the expected income to complete a user's service request for this complicated "Gene Discovery" can earn $250, but the penalty cost for the failure of the service is $100. In our experiment, there are in total 800 KB records to proceed from the DMS. The processing speed (KB recs/s) of the 9 heterogeneous resources are described in Table 1 as well as the cost for initial period, extra charge over the initial period and the resource failure rates. The whole task needs to communicate 800 KB with DMS and 200 KB with the RMS, respectively. The communication channels' characters are depicted by Table 2.

A program for the genetic algorithm was written in C and executed to optimize the partition and distribution problem by a Pentium IV 1.5 G processor. The values of parameters for the GA are selected based upon the experiment results: the population size is 40, the number of generations is 200, the crossover rate is 0.85, and the mutation rate is 0.1. This takes about 20 min on average to run the GA. The results for the optimal partition of the task and distribution on the nine resources are given in Table 3. The expected profit for this solution is $133.42, and the grid service reliability is 0.9255. In this assignment, we can find the resource r_9 is not recruited because to use r_9 is not cost effective. The performance of

Table 2
Characteristics of communication channels

Link	L1	L2	L3	L4	L5	L6	L7	L8	
Failure rate $\times 10^{-3}$ s^{-1}	1	0.5	0.4	0.6	1.3	2.4	2.2	1.5	
Speed (Kbps)	60	50	20	80	10	40	80	10	
Link	L9	L10	L11	L12	L13	L14	L15	L16	L17
Failure rate $\times 10^{-3}$ s^{-1}	2.3	0.4	3.3	2.7	0.9	1.	0.4	0.8	1.2
Speed (Kbps)	100	90	130	30	50	30	70	120	90

Table 3
Optimal solution

Jobs	Partition (%)	Distribution
J_1	10.76	r_1, r_5, r_8
J_2	37.57	r_3, r_7
J_3	51.66	r_2, r_4, r_6

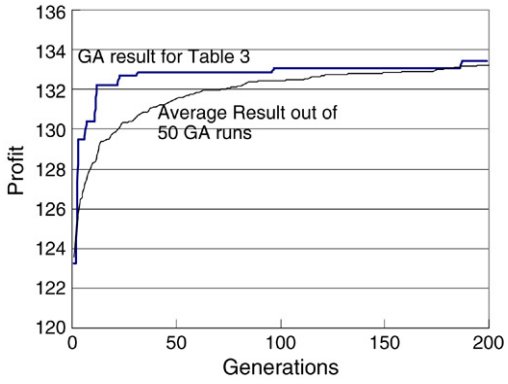


Fig. 3. Performance of GA for 200 generations.

the Genetic Algorithm is further depicted by Fig. 3 at each generation.

The average result out of 50 GA runs at each generation is also plotted in Fig. 3. The standard deviation for 50 runs of the GA for this problem is depicted by Fig. 4 which shows a good convergence of the algorithm.

Some other cases for initially unavailable resources were also studied, as summarized in Table 4. Case A: resource r_2 is initially unavailable; Case B: r_2 and r_3 are unavailable; and Case C: lack of resources r_2, r_3 and r_4. In Table 4, the partition represents the percentage of the workload allocated on the three subtasks, and the distribution contains the subtask number (1, 2, 3) assigned to the resources $(r_1, r_2, \ldots, r_9)$ in which "X" means unavailable resources and "0" represents non-recruited

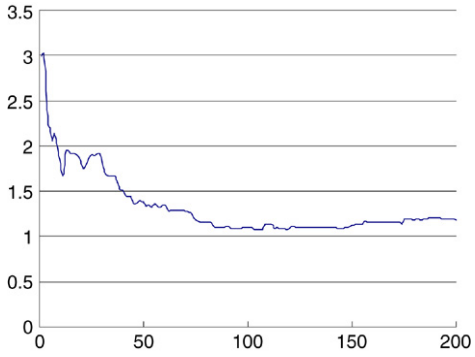


Fig. 4. Standard deviation out of 50 GA runs at each generation.

available resources. It can be observed from Table 4 that the unavailable resources reduce the expected profit compared to the optimal solution from Table 3.

5. Conclusion

The grid service system is a newly developed computing system. This paper solved an important optimization problem for grid service, which was different from other prior works. The problem was to maximize the expected profit by partitioning the service task into subtasks and by distributing them among the available resources. A genetic algorithm was presented to solve this type of optimization problem.

In this optimization model, the grid service reliability is a critical component as the basis of the profit function. However, due to the largeness and complexity of the grid service system, the existing models for distributed systems cannot be directly applied. Thus, this paper first presented a virtual model of two-root tree structure. This tree structure is more general than the star topology that has been assumed in many works, because the tree structure can take into account the failure correlation on the common channels shared by multiple resources.

A case study related to a BioMap project was presented to show the efficiency and effectiveness of the novel models

Table 4
Optimal solutions for initially unavailable resources

Cases	Unavailable	Profit ($)	Reliability (%)	Partition	Distribution
Case A	r_2	129.3	88.23	0.45, 0.19, 0.36	1X3122320
Case B	r_2 and r_3	120.0	88.59	0.47, 0.32, 0.21	2XX231113
Case C	r_2, r_3, r_4	114.0	83.52	0.14, 0.78, 0.08	3XXX11232

and the genetic algorithm in solving this optimization problem. The optimization scheme is applicable not only to the BioGrid services, but also to other grid services, e.g., the Knowledge Grid [29] which provides large-scale intelligent service of mapping the queries to their keys. In this system, both Data resources (DMS) and Computational resources are involved. Hence, our two-root tree is applicable if there is one DMS. Moreover, if there are multiple data sources, the tree structure can be expanded to include multiple roots. Then, the optimization model can be built and algorithms can run in a similar way.

Appendix. Pseudo code for the deriving the service reliability

Algorithm 1. Identifying all the conditional elements that can lead to the failure of any one tree of MRST_j ($j = 1, 2, \ldots, i-1$) while keeping MRST_i operational.

Continued with Algorithm 1, record the working time of their elements in MRSTE$[j, k]$ and MRSTW$[j, k]$ ($j = 1, 2, \ldots, i; k = 1, 2, \ldots, K$), where K is the total number of elements in the grid computing system.

begin
$m \leftarrow 0$ //accumulate the number of conditional elements in *CEV*
for all $k \in [1, K]$ **do**
$OS = ASCEND(\text{MRSTW}[j, k]|(j = 1, 2, \ldots, i))$
//Ascend the working time of the k-th element among the MRST_j ($j = 1, 2, \ldots, i$)
$n \leftarrow 0$
for all $t \in OS$ **and** $t >$ MRSTW$[i, k]$ **do** //Select the working time in OS that is more than the working time of the k-th element of the MRST_i.
$m \leftarrow m + 1$
$n \leftarrow n + 1$
$T_b(m) = t(n-1)$ // Starting time: the immediately previous value of t and $t(0) =$ MRSTW$[i, k]$.
$T_e(m) = t$ // End time: the current considered working time t
$R(m) = e^{-\lambda(k)\cdot(T_e(m)-T_b(m))}$ //Reliability of the considered condition element
$CEV[m] \leftarrow \{k, T_b(m), T_e(m), R(m)\}$
//*CEV* saves the four informations of each conditional element
od
od
end (*Algorithm 1*)

Based on the conditional elements listed in *CEV* identified by Algorithm 1, the following Algorithm 2 can compute the probability of $\Pr(\overline{E_1}, \overline{E_2}, \ldots \overline{E_i} - 1|E_i)$.

Algorithm 2. Evaluating the probability of $\Pr(\overline{E_1}\Lambda\overline{E_2}\Lambda\cdots\overline{E}_{i-1}|E_i)$.

Continued with Algorithm 1 that has identified all the conditional elements listed in the vector *CEV*.

begin
$Y \leftarrow 0$ //This is a global variable to calculate the $\Pr(\overline{E_1}\Lambda\overline{E_2}\Lambda\cdots\overline{E}_{i-1}|E_i)$
for all $k \in [1, m]$ **do** //m is the total number of conditional elements listed in *CEV*
CV$[k] \leftarrow d$ // All the tuples of the initial CV are set to d
od
for all $j \in [1, i-1]$ **do**
OV$[j] \leftarrow 1$ // The initial value of OV is set to be 1 at first
od
BinomialTree(CV,OV,0) //Calculate the probability using Binomial tree
$\Pr(\overline{E_1}\Lambda\overline{E_2}\Lambda\cdots\overline{E}_{i-1}|E_i) \leftarrow Y$//Y is computed in the evaluation of binomial tree
end (*Algorithm 2*)

procedure*BinomialTree*(CV,OV, z)
begin
if OV=0 **then** $Y \leftarrow Y + PR(CV)$, **return** //Find an available leaf
elseif $z = m$ **then return, endif** // Find an unavailable leaf
z $\leftarrow$ z+1 //The point for the current position in CV
$CV_l \leftarrow CV, CV_r \leftarrow CV, CV_l[z] \leftarrow 0, CV_r[z] \leftarrow 1$
// Update CV by binomial values
$OV_l \leftarrow OV, OV_r \leftarrow OV, OV_l[Fail(z)] \leftarrow 0$ //$Fail(z)$ represents the MRST number that is failed by the z-th conditional element $CEV[z]$
BinomialTree(CV_l, OV_l, z), *BinomialTree*(CV_r, OV_r, z)
//Fork the current binomial tree
end(**BinomialTree**)

function *PR*(CV) //*PR*(CV) is the function to calculate the probability of a CV
begin
$x \leftarrow 1$
for all $k \in [1, m]$ **do**
if $CV[k] = 1$, **then** $x \leftarrow x \times R(k)$ //$R(k)$ is saved in $CEV[k]$
elseif $CV[k] = 0$, **then** $x \leftarrow x \times (1 - R(k))$, endif
od
$PR \leftarrow x$
end (**PR* *)

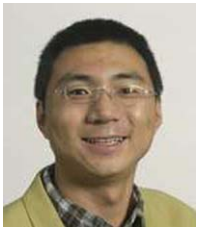
References

[1] L. Bölöni, D. Turgut, D.C. Marinescu, Task distribution with a random overlay network, Future Generation Computer Systems 22 (6) (2006) 676–687.

[2] R. Buyya, D. Abramson, S. Venugopal, The grid economy, Proceedings of the IEEE 93 (3) (2005) 698–714.

[3] R. Buyya, M. Murshed, D. Abramson, S. Venugopal, Scheduling parameter sweep applications on global grids: A deadline and budget constrained cost-time optimisation algorithm, Software: Practice and Experience Journal 35 (5) (2005) 491–512.

[4] M.S. Chang, D.J. Chen, M.S. Lin, K.L. Ku, The distributed program reliability analysis on star topologies, Computers & Operations Research 27 (2000) 129–142.
[5] D.J. Chen, R.S. Chen, T.H. Huang, A heuristic approach to generating file spanning trees for reliability analysis of distributed computing systems, Computers and Mathematics with Application 34 (1997) 115–131.
[6] D.J. Chen, T.H. Huang, Reliability analysis of distributed systems based on a fast reliability algorithm, IEEE Transactions on Parallel and Distributed Systems 3 (1992) 139–154.
[7] D. Coit, A. Smith, Reliability optimization of series-parallel systems using genetic algorithm, IEEE Transactions on Reliability 45 (1996) 254–266.
[8] Y.S. Dai, M. Palakal, S. Hartanto, X. Wang, Y. Guo, A grid-based pseudo-cache solution for MISD biomedical problems with high confidentiality and efficiency, International Journal of Bioinformatics Research and Applications (2006) (in press).
[9] Y.S. Dai, M. Xie, K.L. Poh, G.Q. Liu, A study of service reliability and availability for distributed systems, Reliability Engineering and System Safety 79 (1) (2003) 103–112.
[10] I. Foster, J. Insley, G. von Laszewski, C. Kesselman, M. Thiebaux, Distance visualization: Data exploration on the Grid, IEEE Computer Magazine 32 (12) (1999) 36–43.
[11] I. Foster, C. Kesselman, The Grid: Blueprint for a New Computing Infrastructure, Morgan-Kaufmann, 1998.
[12] I. Foster, C. Kesselman, J.M. Nick, S. Tuecke, Grid services for distributed system integration, Computer 35 (6) (2002) 37–46.
[13] I. Foster, C. Kesselman, S. Tuecke, The anatomy of the grid: Enabling scalable virtual organizations, International Journal of High Performance Computing Applications 15 (2001) 200–222.
[14] V.P. Gergel, R.G. Strongin, Parallel computing for globally optimal decision making on cluster systems, Future Generation Computer Systems 21 (5) (2005) 673–678.
[15] D.E. Goldberg, Genetic Algorithms in Search of Optimization and Machine Learning, Addison Wesley, 1989.
[16] J.H. Holland, Adaptation in Natural and Artificial Systems, Univ. of Michigan Press, Ann Arbor, MI, 1975.
[17] K. Krauter, R. Buyya, M. Maheswaran, A taxonomy and survey of grid resource management systems for distributed computing, Software—Practice and Experience 32 (2) (2002) 135–164.
[18] A. Kumar, Adaptive load control of the central processor in a distributed system with a star topology, IEEE Transactions on Computers 38 (11) (1989) 1502–1512.
[19] A. Kumar, D.P. Agrawal, A generalized algorithm for evaluating distributed-program reliability, IEEE Transactions on Reliability 42 (1993) 416–424.
[20] V.K.P. Kumar, S. Hariri, C.S. Raghavendra, Distributed program reliability analysis, IEEE Transactions on Software Engineering SE-12 (1986) 42–50.
[21] G. Levitin, Y.S. Dai, M. Xie, K.L. Poh, Optimizing survivability of multi-state systems with multi-level protection by multi-processor genetic algorithm, Reliability Engineering & System Safety 82 (2003) 93–104.
[22] C. Li, L. Li, Competitive proportional resource allocation policy for computational grid, Future Generation Computer Systems 20 (6) (2004) 1041–1054.
[23] M.S. Lin, M.S. Chang, D.J. Chen, Efficient algorithms for reliability analysis of distributed computing systems, Information Sciences 117 (1999) 89–106.
[24] B. Marović, Z. Jovanović, Web-based grid-enabled interaction with 3D medical data, Future Generation Computer Systems 22 (4) (2006) 385–392.
[25] M. Palakal, M. Stephens, S. Mukhopadhyay, R. Raje, S. Rhodes, Identification of Biological Relationships from text documents using efficient computational methods, Journal of Bioinformatics and Computational Biology 1 (2) (2003) 1–34.
[26] M. Parashar, H. Klie, U. Catalyurek, et al., Application of Grid-enabled technologies for solving optimization problems in data-driven reservoir studies, Future Generation Computer Systems 21 (1) (2005) 19–26.
[27] D.A. Reed, C. Lu, C.L. Mendes, Reliability challenges in large systems, Future Generation Computer Systems 22 (3) (2006) 293–302.
[28] J. Schneider, Searching for Backbones—a high-performance parallel algorithm for solving combinatorial optimization problems, Future Generation Computer Systems 19 (1) (2003) 121–131.
[29] H. Zhuge, X. Sun, J. Liu, E. Yao, X. Chen, A scalable P2P platform for the knowledge grid, IEEE Transactions on Knowledge and Data Engineering 17 (12) (2005) 1721–1736.



Yuanshun Dai is a faculty member with the Computer Science Department of Purdue University School of Science, at IUPUI. He received his Ph.D. from National University of Singapore, and Bachelor from Tsinghua University, China. He has published 4 books, and over 50 articles. He was awarded the "Most Favourite Professor Award 2005" for teaching, voted by all the Computer Science Students at IUPUI. He was featured by "Industrial Engineer Magazine" (December, 2004), the most important magazine in industry for his research. His research is in Dependability, Grid computing, Security and Autonomic Computing. Dr. Dai is a Program Chair for the 12th Pacific Rim Symposium on Dependable Computing (PRDC2006), and a General Chair for the 2nd IEEE Symposium on Dependable Autonomic and Secure Computing (DASC06). He also chairs many conferences and is on the editorial board of some journals. He is a member of IEEE.

Gregory Levitin received the B.S. and M.S. degrees in Electrical Engineering from Kharkov Politechnic Institute (Ukraine) in 1982, the B.S. degree in Mathematics from Kharkov State University in 1986 and Ph.D. degree in Industrial Automation from Moscow Research Institute of Metalworking Machines in 1989. From 1982 to 1990 he worked as a software engineer and research associate in the field of industrial automation. From 1991 to 1993 he worked at the Technion (Israel Institute of Technology) as a postdoctoral fellow at the faculty of Industrial Engineering and Management. Dr. Levitin is presently an engineer-expert at the Reliability Department of the Israel Electric Corporation and adjunct senior lecturer at the Technion. His current interests are in operations research and artificial intelligence applications in reliability engineering. In this field Dr. Levitin has published more than 100 papers and two books. He is a senior member of IEEE. He serves on the editorial boards of IEEE Transactions on Reliability and Reliability Engineering and System Safety.

Xiaolong Wang is a Graduate Student in the Computer Science Department at the Indiana University Purdue University Indianapolis. Mr. Wang received a B.S. in Computer Science and Technology at Zhejiang University, China and a M.S. in Computer Science at IUPUI. His research interest includes Grid Computing and System Reliability.