Journal of Applied Science and Engineering, Vol. 18, No. 1, pp. 67–78 (2015) *DOI: 10.6180/jase.2015.18.1.09*

An Enhanced Workflow Scheduling Strategy for Deadline Guarantee on Hybrid Grid/Cloud Infrastructure

Huimin Luo[1,2], Chaokun Yan[1]* and Zhigang Hu[2]

[1]*School of Computer and Information Engineering, Henan University, Kaifeng, P.R. China*
[2]*School of Information Science and Engineering, Central South University, Changsha, P.R. China*

Abstract

Deadline guarantee is an important QoS requirement for some critical scientific workflow applications. However, the resource heterogeneity and the unpredictable workloads make it difficult for grid system to provide efficient deadline-guarantee service for those applications. Recent IaaS providers, such as Amazon's EC2, provide virtualized on-demand computing resources on a pay-per-use model, which can be aggregated to the existing grid resource pool to enhance deadline-guarantee of scientific workflow. In this paper, a novel workflow scheduling algorithm DGESA is proposed. First, we evaluate the degree of deadline-guarantee for subtasks of workflow in grid system based on proposed probabilistic deadline guarantee model. Then, proper cloud resources are selected as an accelerator to enhance the deadline-guarantee of subtasks. The experimental results show that proposed algorithm achieves better performance than other algorithms on user's deadline guarantee.

Key Words**:** Scientific Workflow, IaaS, Grid, Scheduling, Stochastic Service Model

1. Introduction

Computational grid is an important infrastructure that provides consistent, pervasive and inexpensive access to high-end computational capabilities, which has been used to execute complex scientific workflow applications from various areas of science. The last decade has seen an unprecedented growth in grid infrastructures which nowadays enables large scale deployment and execution of scientific applications, such as TeraGrid, Earth System Grid, Enabling Grids for E-scienceE, etc [1]. Among these research domains, specially, a class of scientific applications must complete in a timely manner to generate appropriate results, such as disaster relief during crisis or specific weather predictions, neurosurgical imaging using simulation. Although grid technology enables access to computing and storage resources sharing among institutions and scientists, it does not guarantee that the scientific applications can be finished within a given time. In practice, due to scientific computing is traditionally a high-utilization workload and the production grids above-mentioned often run at over 80% utilization [2], running scientific applications in such overloaded grid environments often become latency-bound and suffers from serious deadline missing.

More recently, cloud computing platforms, such as Amazon EC2 [3] and Eucalyptus [4], has generated significant attention in HPC (high performance computing) and scientific research. Cloud computing provides on-demand resource, allowing users to execute applications without the need of new investments, converting fixed costs in expense based on current needs. The advantages of cloud computing enable many companies adopt cloud for expanding their existing infrastructures

*Corresponding author. E-mail: ckyango@csu.edu.cn

or rapidly deploying their applications. The scientific community has shown increasing interest in combining the best attributes of grid and cloud to enhance the capacity of grid resources, which mainly focused on building and developing hybrid computing environments for scientific computing, but only a few consider the QoS requirements of applications. Moreover, the research for integrating grid and cloud technique is still in infant stage. There is an urgent need to provide efficient deadline guarantee service for the execution of deadline sensitive scientific applications in hybrid grid/cloud infrastructure.

In that hybrid computing environment, providing efficient deadline guarantee service requires carefully determining the best split between public cloud and grid components. One of the problems to face that is how to split tasks of a scientific application to execute in grid or in cloud. The problem involves execution time of tasks on heterogeneous resources as well as monetary costs in the use of charged cloud resources. Existing resource mechanisms and scheduling algorithms is impractical for these scientific workflow applications, given the underlying uncertainty in various resources.

In order to solve the above-mentioned split problem, a new concept called default risk of subtask is proposed to evaluate the probability of deadline violation of subtask. Then a novel task redirecting strategy is presented, which defines how to select virtual resources from cloud providers. Last, a novel workflow scheduling algorithm called DGESA is proposed to enforce the deadline guarantee of scientific applications in hybrid computing environment. Experimental results show that our algorithm will be a good option to speed up the execution and guarantee the deadline of scientific applications.

2. Related Work

In the past years, there are few works addressing scheduling of scientific workflow on grid or cloud. Here, we review the literature related to various aspects of our work.

2.1 Workflow Scheduling with Deadline Constraint

In the context of grid and cloud computing, many efforts have contributed to the problem of resource scheduling with deadline constraint. For example, Menasc and Casalicchio [5] proposed back-tracking algorithm to optimize cost under deadline constraints. Sakellariou [6] proposed LOSS and GAIN approach that iteratively adjusts a schedule which is generated by a time-optimized heuristic or a cost-optimized heuristic to meet the budget constraints. Buyya et al. [7] tried to resolve deadline constraint optimization problem using genetic algorithm. The deadline distribution algorithm called Deadline-MDP was presented to minimize the execution cost while meeting the deadline. Xu et al. [8] proposed an algorithm for scheduling multiple workflows with multiple QoS constraints on the cloud. S. Abrishami and M. Naghibzadeh [9] proposed a QoS-based workflow scheduling algorithm based on Partial Critical Paths in cloud, which tries to minimize the cost of workflow execution while meeting user-defined deadline.

Generally speaking, the algorithms overviewed above mainly focused on transforming the structure of DAGs describing workflows to schedule tasks based on static performance of resources, while dynamicity and heterogeneity of sharing resources are always ignored by them. Moreover, some deadline constraint-based scheduling attempt to maximize (or minimize) other QoS metrics, which brings high job rejection rate and deadline missing.

2.2 Scheduling Strategy to Improve Predictability of Workflow Execution

To complete scientific workflow successfully and improve the predictability, some scheduling strategies usually adopt following techniques to cope with the problem, such as performance evaluation and prediction, scheduling based on just-in-time information from GIS, fault tolerance based scheduling, advanced reservation. Dong et al. [10] proposed a resource performance fluctuation aware workflow scheduling algorithm (called PFAS), which is an example for scheduling workflow based on just-in-time information. However, because target grid resources locate in a large scale heterogeneous, autonomous, dynamic environment, getting precise information efficiently and effectively is very difficult. Many researchers have been interested in taking advance reservation to improve predictability. For instance, Zhao et al. [11] presented two advance reservation policies for workflow in order to cope with execution time changes for tasks. Marek Wieczorek et al. [12] presented an extension to devise and implement advance reservation as part of the scheduling and resource management services of the ASKALON. OpenNebula [13] uses advance reservation in the Haizea lease manager. However, advance reservation also brings many negative effects on

resources and task scheduling, such as resource fragment and lower resource utilization. Fault tolerance is an important mechanism for grid or cloud computing environment. However, on-line profiling the failure characteristics can be a very difficult task. On the other hand, some fault tolerance mechanisms often lead to unnecessary resource consumption and degradation of the QoS of system, which cause further deterioration of deadline missing.

2.3 Workflow Scheduling Strategy in Hybrid Computing Environment

Several efforts have addressed issues how cloud providers can be used to supply resources to scientific communities. Deelman et al. [14] demonstrated the cost of using cloud to supply the needs for resources of data intensive applications. Extending the capacity of HPC or grid to offer greater problem-solving resources for scientific applications has received attention in academia. Hyunjoo Kim et al. [15] introduced an autonomic computing engine for cloud and grid environments. Experimental results show that cloud resources can be used to complement and reinforce the scheduling and usage of traditional HPC infrastructures. Ostermann et al. [16] extended a grid workflow infrastructure to include cloud resources, and experimented with Austrian grid and Eucalyptus-based academic cloud. Similarly, Bittencourt et al. [17] propose an infrastructure managing the execution of service workflows in hybrid system composed of both grid and cloud. However, how to guarantee the application deadline is not considered in these literatures. Aimed at deadline sensitive scientific workflow applications, Ramakrishnan et al. [18] present a probabilistic QoS-based resource abstraction that enables higher-level tools to implement effective workflow orchestration. In essence, the implementation of resource abstraction model builds on queue wait time prediction technique from QBETS [19], which is a non-parametric time series method to predict bounds on the queue wait times of individual jobs. Vecchiola et al. [20] describes the provisioning mechanism in Aneka and how it supports different resources from cluster, grid or cloud. The algorithm for deadline-driven resource provisioning in Aneka is presented, which is a best-effort algorithm that only considers the time left for deadline and the average execution time of tasks that compose application to determine number of resources required. The dynamicity of service capacity of resource and multiple cloud providers are ignored in Aneka. L. F Bittencourt et al. [21] present hybrid cloud optimized cost scheduling algorithm, called HCOC. HCOC decides which resources should be leased from the public cloud and aggregated to the private cloud to provide sufficient processing power to execute a workflow within deadline. Fundamentally, HCOC is an iterative algorithm with high time complexity, which can't be applied to actual scheduler of hybrid computing environment because of the complexity of scientific applications. Moreover, the dynamicity of resource is ignored by HCOC.

3. Architecture Overview

In the section, we present the major design of scientific workflow management system in hybrid computing environment. The overall architecture is illustrated in Figure 1. Conceptually, it is a composed of Application layer and Resource layer.

For e-science application, there is usually a single system, portal system, where the application description is co-located with the scientific workflow management system. User can submit execution request by portal, which can support different levels of application parallelism. Moreover, portal can provide the necessary user interface for the workflow concepts and for the workflow application hosting environments. User can describe their workflow applications at high level of abstraction using various modeling tool, such as UML. Then, the abstract workflow representation is submitted to the Autonomic Workflow Orchestrator for transparent execution onto the underlying infrastructure.

Autonomic Workflow Orchestrator is the core of system architecture. Submitted workflow will be fetched by the workflow manager that responsible for coordinating the execution of the overall workflow, which is the central authority for all the running workflows of the platform. Resource image catalogue mainly aggregates the registration information provided by underlying resources of grid or cloud. The catalog is populated by periodically scanning the information published by the grid or cloud providers and building an inverted map, in which the key is the application URI or the resource image URI and the values are the URIs of the resources that support particular application of resource image. The Autonomic Scheduler is responsible for mapping the

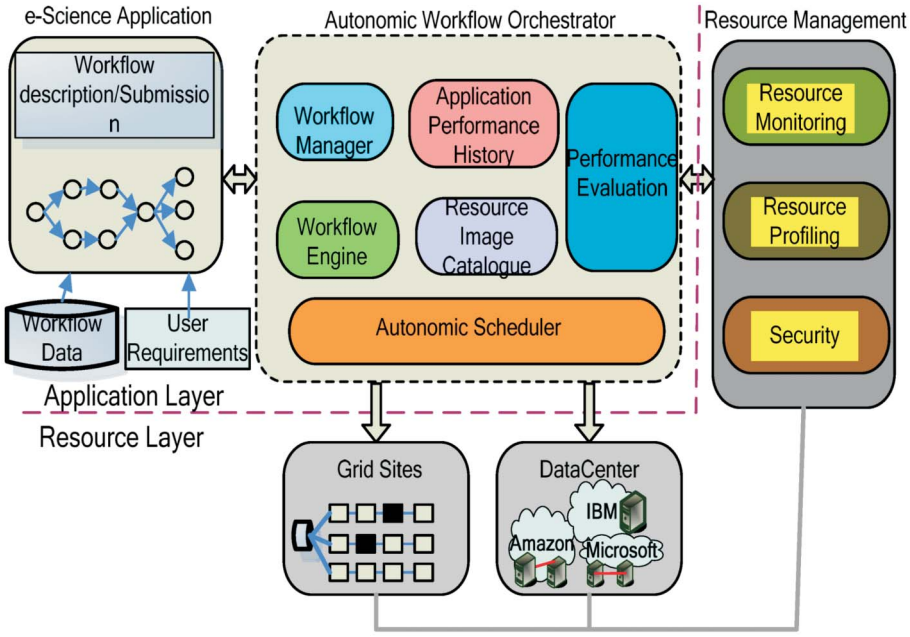

Figure 1. System architecture.

tasks of scientific workflow to underlying resources. Before making a mapping decision, it needs to send query request to the resource image catalogue service to get the list of candidate resources matching the URI. For the candidate resources, Autonomic Scheduler will use the Application Performance History Service and Performance Evaluation Service is obtain an initial hybrid grids/cloud resources that can be used to execute the workflow based on user-defined objectives and constraints. Application Performance History Service is responsible for aggregating the history information about previous similar applications and the Performance Evaluation Service estimate the execution performance for candidate resources of grid sites or datacenter according to the performance data from the Infrastructure Layer. In the paper, we implemented the Evaluation Service by proposed probabilistic deadline guarantee model based stochastic service model and Dirichlet probability distributed model, which will be explained in greater detail in next section. Workflow engine is responsible for fetching mapped tasks of workflow and deciding subsequent handling node behind each instance of the running task in the task list, which acts as the communication layer between the workflow management system and underlying resources.

Resource Layer is responsible for provisioning the resources on their specific platform, such as TeraGrid or EC2, based on the requirements and QoS constraints of scientific applications. The tasks of workflow will be sent to the Resource Layer to execute. All resource healthy status, such as availability, failure et al., are monitored by the Resource Monitoring Service, which be implemented using a heart-beating mechanism. Some performance data, such as processing capacity, service rate et al., can be aggregated by Resource Profiling Service. All workflow tasks running and producing intermediate results on remote unknown resources need to be protected. Some issues need to be stressed, such as authentication, secure File Transfer Protocol, encryption et al.

4. Probabilistic Deadline Guarantee Model

Existing grid systems is still an important infrastructure for scientific workflow. However, it is very difficult to give efficient deadline guarantee service for deadline sensitive scientific workflow due to the variability and complexity of the underlying resource. Relatively speaking, in a hybrid computing environment, the remote resources from the cloud providers can be utilized to augment the capacity of the grid resource. However, as the use of cloud resources incurs cost, the scheduler needs to find the price at which this performance improvement is achieved. In this work, a task redirection strategy is proposed to find the appropriate time-point of

task redirection. The high level architecture of our redirection strategy is shown in Figure 2.

On the whole, there are two fundamental problems towards task redirection: how to select the tasks redirecting to cloud resources and how to select the virtual resource to execute redirected tasks. The former problem exists mainly in appropriate selection of grid resources in dynamic grid system. The latter problem is mainly concerned with the effective selection of virtual resources to speed up the execution of redirecting tasks.

The redirection mechanism can be described as follows. For resource request of task, scheduler will query the available resources from grid systems firstly. Then, the resources that meet the requirements and minimize the probability of deadline violation will be selected. Otherwise, redirection mechanism will be triggered to redirect the un-dispatched tasks. In order to find suitable virtualized resource for the un-dispatched tasks, we adopt Dirichlet probability distributed model to model and evaluate the execution performance of cloud services in an IaaS provider, such as EC2 or GoCrid. Scheduler will dispatch task to the cloud service with highest reputation value when the redirection event occurs. The detailed implementation method is discussed as follows.

4.1 How to Select Subtasks Redirecting to Cloud

In this section, a probability based task redirection strategy is proposed to address the problem how to select tasks. Here, to facilitate the redirecting mechanism, two assumptions are presented and discussed as follows.

Assumption 1: A set of fine-grained deadline constraints for scientific application can be set in SLA (service level agreement) between user and resource providers, while fine-grained constraints refer to deadline assigned to individual subtasks of scientific application.

In the real world, there exists the global deadline constraint and some local deadline constraints for workflow [22]. The global deadline is set for the overall workflow. The local deadline constraint is always set for a process event composed of many tasks. In order to get the fine-grained deadline constraint for every subtask, some deadline distribution methods can be applied to overall scientific workflow in the building stage of workflow.

Assumption 2: The distribution of task arrival rate λ_i and service rate μ_i can be obtained through statistic analysis from execution log on grid sites. Here, we assume the interval between task arrivals on grid site and serving time follows exponential distribution.

Recently, many researchers have applied stochastic service model to describe the working of grid resources. These studies show that stochastic service model is capable of precisely describing the working and workload of grid resources in dynamic environments. In the paper, we adopt M/M/C queuing system to describe dynamic workloads and service capacity of resources. The performance evaluation model of grid resource is shown in Figure 3. Assumption 2 mainly depends on the monitoring and analysis for the workload on practical

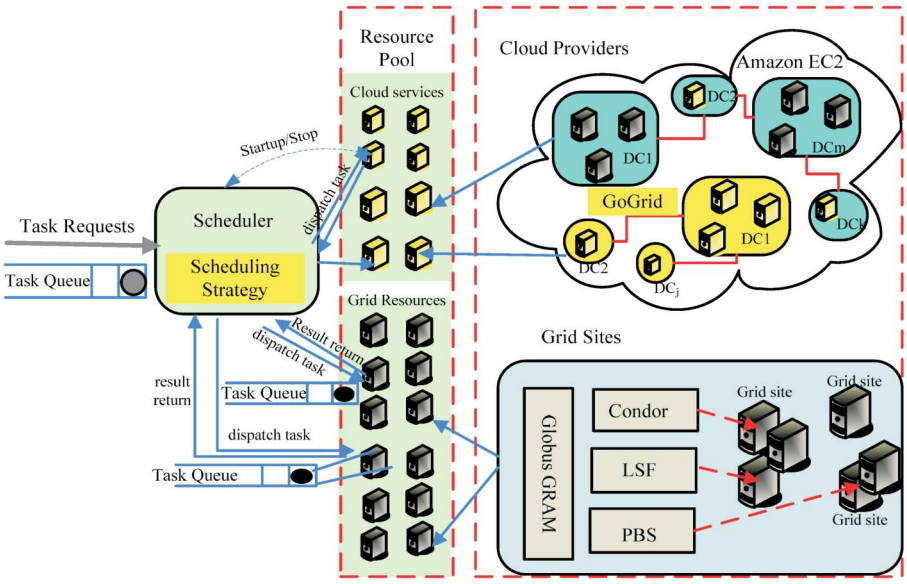

Figure 2. Task Redirection Mechanism of hybrid computing environment.

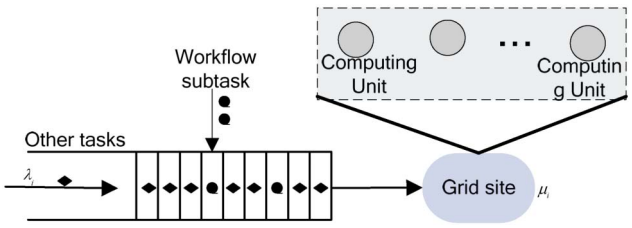

Figure 3. Performance evaluation model.

resource sites. There exist some efforts focusing on profiling the workload characteristics. Fox example, the work in [23] concluded that λ_i, μ_i follow exponential distribution based on the data of Grid'5000 sites.

Definition 1: (default risk of subtask) Given a subtask t_i was scheduled on resource r_j, then default risk of subtask t_i refer to the probability of deadline violation occurring, which can be represented as $DRS(t_i, r_j)$. Based on the value of *DRS*, scheduler can determine whether or not to redirect the subtask.

Based on definition 1, assumption 1 and 2, theorem 1 is presented to give the calculation method for probability of deadline violation used to redirect task.

Theorem 1: Given a subtask t_i was scheduled on resource r_j, d_k is the corresponding sub-deadline assigned to task t_i, if we use M/M/C queuing system to characterize the workload and service capacity of r_j, then default risk of subtask t_i, $DRS(t_i, r_j)$, can be calculated by the following equation:

$$DRS(t_i, r_j) = 1 - \sum_{n=0}^{c_i} \delta \cdot \frac{(\rho_i \cdot c_i)^n}{n!} - \sum_{k=1}^{c_i \cdot \mu_i \cdot d_k - 1} \delta \cdot \frac{\rho_i^{k+c_i} \cdot c_i^{c_i}}{c_i!} \quad (1)$$

where $\delta = \left[\sum_{n=0}^{c_i - 1} \frac{(\rho_i \cdot c_i)^n}{n!} + \frac{(\rho_i \cdot c_i)^{c_i}}{c_i! \cdot (1 - \rho_i)} \right]^{-1}$, ω is a random variable representing actual completion time of task t_i, c_i is the number of computing unit in resource r_j.

Proof: Let stochastic variable ψ represent the number of tasks waiting in the job queue of r_j. According to the theory of stochastic service system, the probability that the number of tasks waiting on r_j is k should be:

$$\Pr\{\psi = k\} = \begin{cases} \delta \cdot \frac{\rho_i^{k+c_i} \cdot c_i^{c_i}}{c_i!}, & k > 0 \\ \sum_{n=0}^{c_i} \delta \cdot \frac{(\rho_i \cdot c_i)^n}{n!}, & k = 0 \end{cases} \quad (2)$$

According to the property of M/M/C queuing system, $c_i \cdot \mu_i$ is the service rate of r_j, i.e. the average number of tasks completed on r_j in a unit of time. Furthermore, the average number of tasks that r_j can finish in d_k units of time should be $c_i \cdot \mu_i \cdot d_k$. Therefore, deadline of task t_i may occur violation means that the number of tasks waiting in the queue of r_j should be larger than $c_i \cdot \mu_i \cdot d_k - 1$. So the default risk of subtask t_i is equal to the probability that the number of waiting tasks on r_j is larger than $c_i \cdot \mu_i \cdot d_k - 1$. So we have

$$DRS(t_i, r_j) = \Pr\{\psi > c_i \cdot \mu_i \cdot d_k - 1\} \quad (3)$$

Combining (2) and (3), we have that

$$\begin{aligned} DRS(t_i, r_j) &= \Pr\{\psi > c_i \cdot \mu_i \cdot d_k - 1\} \\ &= 1 - \Pr\{\psi \le c_i \cdot \mu_i \cdot d_k - 1\} = 1 - \sum_{k=0}^{c_i \cdot \mu_i \cdot d_k - 1} \Pr\{\psi = k\} \\ &= 1 - \sum_{n=0}^{c_i} \delta \cdot \frac{(\rho_i \cdot c_i)^n}{n!} - \sum_{k=1}^{c_i \cdot \mu_i \cdot d_k - 1} \delta \cdot \frac{\rho_i^{k+c_i} \cdot c_i^{c_i}}{c_i!} \end{aligned}$$

4.2 How to Select the Cloud Service for Redirected Subtasks

In real world, each cloud provider has multiple datacenters called *region* distributed across the world. For example, Amazon has 7 regions in many cities across Asia, Europe, South America and United States [24]. Each *region* can provide various types of cloud services with different prices. Obviously, user could have completely different experience when requesting same cloud services from different regions because of the geographical distribution of regions and different network conditions. Therefore, selecting suitable cloud service has important impact on the performance of scientific workflow management system. In the section, a reputation-based cloud service selection strategy is proposed to address the problem. First, to facilitate the selection method, a basic assumption is presented.

Assumption 3: For each cloud service, we assume all task execution information in a time interval T can be obtained from task interaction logs of the cloud service. After finishing one task interaction with a cloud service, user will give an evaluation level for the interaction based on the following metric:

$$\xi = \frac{TRT}{EET} \quad (4)$$

where *TRT* stands for actual **t**ask **r**esponse **t**ime, *EET*

stands for the **e**xpected **e**xecution **t**ime of task.

Each feedback ξ will be gathered and recorded in the interaction log by cloud system. Here, user feedback is divided into five different ratings: excellent, very good, good, fair and poor. For example, the procedure of task execution can be considered "excellent" when $\xi \leq 0.5$. It can be deemed "poor" when $\xi \geq 2$. In real world, the value of ξ can be determined according to actual applications. Because task response time is very important for deadline sensitive applications, so assumption 3 is rational and easy to understand.

According to overall user feedbacks, we will obtain the performance data of a cloud service in a time interval *T*, which can be used to guide the selection for cloud service. Here, the concept of reputation is adopted to model and evaluate the behavior of a cloud service. Dirichlet reputation model [25] was proposed in the realm of e-commerce to construct reputation system. More recently, it has been widely applied in the context of several diverse domains such as internet, ad-hoc wireless networks because of it' flexibility, simplicity and great expression capacity [26]. Therefore, based on assumption 3, we employ Dirichlet reputation model to evaluate *region*' behavior in a time interval *T*.

The Dirichlet distribution captures a sequence of observations of the *k* possible outcomes with *k* positive real parameters α_i, $i = 1,\ldots, k$, each corresponding to one of the possible outcomes. Let $\Theta = \{\theta_1,\ldots\theta_k\}$ be a state space consisting *k* mutually disjoint events. We define vector $\vec{p} = \{p(\theta_i)|1 \leq i \leq k\}$ to denote continuous random vector in the k-dimension simplex, vector $\vec{\alpha} = \{\alpha_i|1 \leq i \leq k\}$ to denote the k-dimension random observation variable. The probability density of the Dirichlet distribution is given as

$$f(p \mid \alpha) = \frac{\Gamma(\sum_{i=1}^{k} \alpha(\theta_i))}{\prod_{i=1}^{k} \Gamma(\alpha(\theta_i))} \prod_{i=1}^{k} p(\theta_i)^{\alpha(\theta_i)-1} \quad (5)$$

where $\Gamma(x) = \int_0^\infty t^{x-1} e^{-t} dt$ is the Gamma function.

The probability expectation value of any of the *k* random variables is defined as:

$$E(p(\theta_i) \mid \vec{\alpha}) = \frac{\alpha(\theta_i)}{\sum_{i=1}^{k} \alpha(\theta_i)} \quad (6)$$

Since the Dirichlet distributed is a conjugate priori of the multinomial distribution, the posteriori distribution is also Dirichlet and can be calculated as follows:

$$f(\vec{p} \mid \vec{r}, \vec{\alpha}) = \frac{\Gamma(\sum_{i=1}^{k} r(\theta_i) + C\alpha(\theta_i))}{\prod_{i=1}^{k} \Gamma(r(\theta_i) + C\alpha(\theta_i))} \prod_{i=1}^{k} p(\theta_i)^{(r(\theta_i)+C\alpha(\theta_i)-1)} \quad (7)$$

where $\alpha(\theta_i)$ is a base rate vector over the state space Θ satisfying $\alpha(\theta_i) \geq 0$ and $\sum_{i=1}^{k} \alpha(\theta_i) = 1$, *C* is a priori constant which is equal to the cardinality of the state space over which a uniform distribution is assumed, and the vector $r(\theta_i)$ is a posteriori evidence over the state space Θ. Given the Dirichlet distribution of (7), the probability expectation of any of the *k* variables can be written as:

$$E(p(\theta) \mid \vec{r}, \vec{\alpha}) = \frac{r(\theta_i) + C\alpha(\theta_i)}{C + \sum_{i=1}^{k} r(\theta_i)} \quad (8)$$

According to assumption 3, *k* is equal to 5. In *T*, let R_y be the total accumulated ratings of cloud service *y*, which can be described as a vector: $\vec{R}_y = <r_y(1), r_y(2),\ldots, r_y(5)>$, where $r_y(i)$ is the sum of all ratings of cloud service *y* by all users in *T*, expressed by

$$s_y(i) = \frac{r_y(i) + C\alpha_i}{C + \sum_{i=1}^{5} r_y(i)} \quad (9)$$

where *M* is the set of all users who rated cloud service *y* during period *T*. Obviously, $r_y(i)$ corresponds to the posteriori evidence in (7). According to (8), the evaluating score of rating *i* for cloud service *y*, denoted by $s_y(i)$, can be calculated by following equation:

$$s_y(i) = \frac{r_y(i) + C\alpha_i}{C + \sum_{i=1}^{5} r_y(i)} \quad (10)$$

Hence, we can obtain a continuous evaluation score in overall set *M*. Let vector $S_y = <s_y(1), s_y(2),\ldots, s_y(5)>$ be evaluation score vector in overall ratings for cloud service *y*. However, in real world, it is not practical that a cloud service has multiple scores. Therefore, we combine the discrete scores into a single integrated score by the weighted sum method. Assuming the 5 ratings corresponds to the 5 integrator in set {0, 1, 2, 3, 4}, let $w(i)$ is the value of rating *i*, then the total reputation value can be expressed as:

$$\mathrm{Re}_y = \sum_{i=1}^{5} w(i) \cdot s_y(i) \quad (11)$$

In (10), Re_y is calculated based on current $r_y(i)$ and base rate α_i, which not considering the history evaluation score. Therefore, in order to improve the accuracy of reputation value, equation (8) can be modified as follows:

$$s_{y,\Delta}(i) = \frac{r_y(i) + C\alpha_{i,\Delta}}{C + \sum_{i=1}^{5} r_y(i)}$$
$$\alpha_{i,1} = \alpha_i \quad (12)$$
$$\alpha_{i,\Delta} = s_{y,\Delta-1}(i)$$

where Δ and $\Delta + 1$ are two continuous evaluation periods.

5. Scheduling Algorithm

5.1 Application and Resource Model

Scientific applications can be represented by DAG. In the paper, a deadline sensitive scientific workflow can be represented as a 2-tuple <*D*, *d*>, where $D = (\Gamma, \Lambda)$ is the workflow structure, *d* denotes the overall deadline. Let set $\Gamma = \{T_i, 1 \le i \le m\}$ be the collection of tasks in the workflow, corresponding to the vertex set in DAG.

In a hybrid computing environment, the resources include not only dynamic grid resources, but also cloud services from datacenters (regions) in an IaaS provider. Given that a grid is composed of *N* resource nodes, represented as a set (R_1,…, R_N). The computing capacity of each resource r_i is represented as P_{r_i}. There exists a link between each pair of resources. We assume that there exist *w* datacenters in a hybrid computing environment, represented as a set (DC_1,…, DC_w). Each datacenter has a set of charged cloud services, represented as set (VR_{i1},…, VR_{ik}). Additionally, each cloud service VR_{ik} has a price per time unit $price_{vki}$. Once the resources executing scientific application have been selected from grid system or datacenters of cloud, a hybrid computing environment is built. Therefore, the resources of hybrid computing environment can be represented as set $H = (\bigcup_{i=p}^{q} R_i) \bigcup (\bigcup_{j=1}^{z} DC_j)$, where $p, q \ge 1$ and $z \ge 1$.

5.2 Scheduling Strategy

In the section, a **d**eadline **g**uarantee **e**nhanced **s**cheduling **a**lgorithm (DGESA) for deadline sensitive scientific workflow is proposed. The pseudo-code of DGESA is shown in Figure 4.

Before dispatching subtasks of workflow to appropriate resource, DGESA needs to obtain the information about the sub-deadline assignment of all subtasks in workflow. Some existing methods of workflow deadline distribution can be used to get it, such as distribution strategy of Pegasus [27], DTL (deadline top level) [28], DBL (deadline bottom level) [29]. Comparatively, DTL is a simple and easy-to-implement method with low time-complexity, which has been applied to Gridbus [30] Scheduler. Here, DTL is adopted to distribute overall deadline of workflow into each subtask and get a sub-deadline vector of all workflow tasks. Then, scheduler starts to schedule workflow level by level.

DGESA computes the value of *DRS* of each task according to the theorem 1. From the entry task, algorithm schedules each task to appropriate resource to execute. If all *DRS* of a subtask are bigger than threshold *v**, then the subtasks need to redirect to cloud service of data center. Here, the value of *v** is set by scheduler through mining the knowledge from the real execution log of e-science applications. This procedure schedules all unscheduled parents of its input node. As it has been called for the exit task, it will schedule all workflow tasks.

5.3 Time Complexity

To compute the time complexity of DEGEA, suppose that workflow ς has *m* subtasks. We assume that the number of grid resources is *n* and the maximum number

DGESA (ς , *H*)

INPUT: Workflow ς submitted by user at time τ and the computation requirement of each subtask, *cr*(*task*); Grid resources R_i ($1 \le i \le n$) and <λ_i, μ_i>; processing capacity pc_{jk} for each instance type and price $price_{jk}$ in region DC_j ($1 \le j \le m$).

1. **Begin**
2. Put the entry task t_{entry} into ready task queue Q.
3. $t_i \leftarrow t_{entry}$.
4. **while** t_i is not the exit task do
5. $t_i \leftarrow$ take the first task from Q.
6. **for** $j = 1$ to N **do**
7. Calculate $DRS(t_i, r_j)$ for task t_i using Theorem (1).
8. get the candidate resource R_{T_i} with minimum $\{DRS(T_i, r_j)\}$;
9. **if** min $\{DRS(T_i, r_j)\}$ <= *v**
10. Schedule T_i in the resource with min $\{DRS(T_i, r_j)\}$;
11. **Else**
12. **For** available cloud service do
13. Compute the reputation value of each cloud service using formula (12);
14. **End for**
15. Construct service set meeting deadline constraint of subtask.
16. Select cost-effective cloud service for T_i with high reputation;
17. **end for**
18. Put ready child tasks into Q if their parent tasks have been scheduled;
19. **end while**
20. **End**
21. **OUTPUT**: task-resource mapping strategy.

Figure 4. Pseudo-code of DGESA algorithm.

of available grid resources for a task is n. We also assume that the maximum width of ς is l, the number of virtual resource is k. The detail analysis of time complexity for DTL can be found in [28], which is omitted here.

In essence, DGESA algorithm is a recursive procedure. It starts to schedule workflow from the entry task (line 2). The algorithm has a while loop (line 4–15) that processes all tasks. Inside the while loop, the first task will be taken from queue Q firstly. Then, algorithm computes all value of *DRS* for the task according to theorem (1) (line 6–7), the time complexity is O(n), where n represents the number of grid resources. The minimum value of *DRS* can be obtained (line 8), the time complexity is O(n). There exists a decision (line 9–17) that processes the task redirection whose time complexity is O(k). When the number of tasks is m, the while loop will process m times to generate result whose time complexity is O(mn). Once all parents of a task have been scheduled, the tasks are ready for scheduling and then the scheduler put it into ready queue Q (line 18), the time complexity can be considered as a constant time.

To sum up, the time complexity of DGESA is O(mn). Comparatively speaking, the number of grid resource and virtual resource is much less than the number of tasks of workflow. So the execution time of algorithm will not take a long time. As a result, DGESA is a practical solution in real hybrid computing environments.

6. Performance Evaluation

6.1 Experimental Parameter Setup

We have set up a simulation environment of hybrid computing to evaluate the performance of DGESA. The simulation environment is composed of 3 grid sites based on valid data of Grid'5000 in [31] and 4 cloud services that could be requested by the scheduler. According to [24], we know One EC2 Compute Unit (ECU) provides the equivalent CPU capacity of a 1.0–1.2 GHz 2007 Opteron or 2007 Xeon processor. Therefore, we assume that IaaS provider offers four different computation services with different processor speeds and prices, which is close to the standard instance of EC2. The deployment delay of virtual instances and data storage cost are considered to be zero in the experiments. To evaluate the reputation of cloud service, a set of interacting log for each type cloud service are generated randomly. Table 1 shows the setup of our simulation environment.

Nowadays, many scientific applications are represented by DAG. Juve et al. [32] study the structure of five realistic workflows from diverse scientific applications, which are Montage, CyberShake, Epigenomics, LIGO, SIPHT. Moreover, they developed a workflow generator, which can create synthetic workflows of arbitrary size. We choose three sizes Montage workflow instance, which are: small (25 tasks), medium (50 tasks), large (100 tasks) to evaluate the performance of DGESA. Detail workflow description is available in DAX format from website (https://confluence.pegasus.isi.edu/display/pegasus/WorkflowGenerator). As user's deadline parameter is not recorded in the trace, we adopt a similar experimental methodology in [33] to model the parameter as: $D = \sigma T$, where T is the execution time in the fastest computing resource of all tasks in critical path of the currently scheduled workflow. Therefore, T is just a lower bound for the makespan of executing the workflow. We let σ ranges from 1.2 to 3.0 in our experiments with a step length equal to 0.6. According to the analysis of workload for clusters and real grid systems in [23], the interval between task arrivals λ and task execution time μ follow exponential distribution. Their values are generated from the interval (0, 1) and (1, 2) respectively. The value of threshold v^* is 0.2.

In the simulation experiments, DGESA is compared with HCOC and the scheduling algorithm of Aneka for different deadline scenarios. For each scenario, simulation was repeated 50 times and the average makespan and average cost was calculated to generate result graph.

6.2 Experimental Results and Analysis

Figure 5 show average makespan comparison of DGESA, HCOC and Aneka' algorithm with respect to the increase of deadline. We can observe that DGESA is very efficient in deadline guarantee. The average make-

Table 1. Resources in the hybrid computing environment

Resource name	Computing capacity (GHz)	Price (USD/h)	Number of computing units
Luxembourg	2.0	0	44
Reims	1.7	0	88
Lyon	2.4	0	158
Small	1.0	0.115	1
Medium	1.7	0.23	1
Large	2.2	0.46	2
Xlarge	4.3	0.92	4

span of DGESA is always lower than average makespan of other algorithms. Moreover, when deadline is very tight (e.g. $\sigma = 1.2$), average makespan of DGESA is closer to deadline compared with HCOC and Aneka' algorithm. For Montage-100 workflow, average makespan of DGESA will be better than HCOC by 9.9% and Aneka by 13.1%. As we can see from Figure 5, with the increase of task number of Montage workflow, average makespan increases rapidly. This is easy to understand. As deadline become looser, three algorithms present an increase in the makespan too. This is due to the fact that more subtasks of workflow can be completed by grid resources. As a consequence, DGESA leases less cloud service from the IaaS provider.

Figure 6 shows average cost for three algorithms. Note that the average costs in three cases have a different behavior when deadline increases. The higher the deadline, the lower the cost for DGESA schedules. We can note that the average cost of HCOC and Aneka's algorithm are better than DGESA. This is because HCOC optimizes the execution cost in selecting cloud services from public clouds. Aneka uses EC2 large virtual instance to speed up the execution of subtasks. Large instance means higher cost. However, comparatively, the average cost of DGESA is higher than the algorithm of Aneka system and HCOC. There are two reasons for the result. One is DGESA redirects more subtasks to cloud for providing efficient deadline guarantee, and the other is cost optimizing methods are not considered. As a consequence, more cloud services are leaned by DGESA, which increase the execution costs accordingly.

Besides, we investigate the impact of different thresholds on scheduling performance for DGESA algorithm when deadline is equal to 1.8 T. Figure 7 shows the impact of threshold v^* on scheduling Montage-100 workflow. It can be see that, with the decrease of threshold v^*, the costs of execution are increasing while the makespan is decreasing. The execution cost is the highest when v^* equals to 0.2, which is 2.79 times of the execution cost when v^* equals to 0.5. This shows that threshold v^* has impact on DGESA algorithm. In this experimental scenario, the deadline constraint of workflow can be guaranteed when the value of v^* is less than 0.5.

As a result, the value of parameter v^* has impact on the deadline guarantee capacity of workflow management system and execution cost. Before scheduling scientific workflow in hybrid computing environment, v^* must be set by scheduler. In real world, the value of v^* can be adjusted based on the actual requirement for

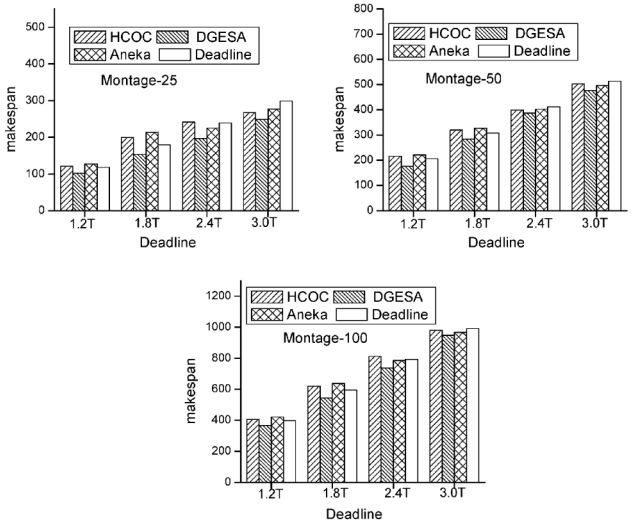

Figure 5. Makespan comparison.

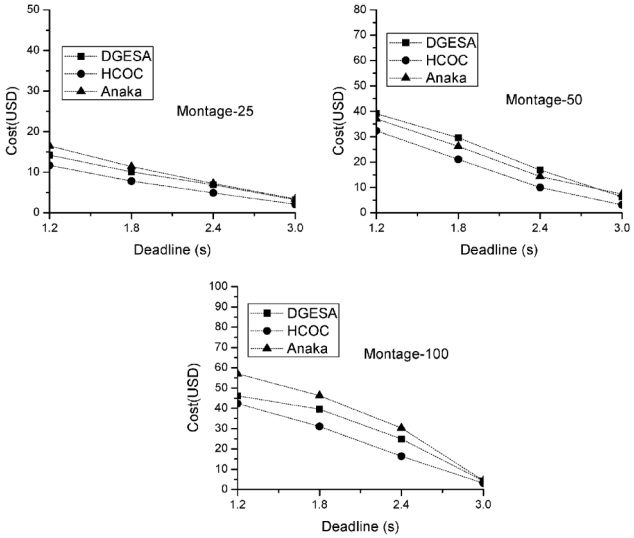

Figure 6. Cost comparison.

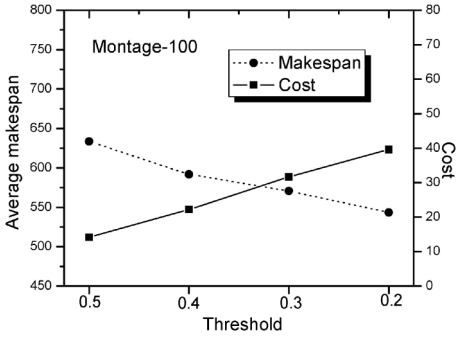

Figure 7. Threshold impact on performance.

deadline guarantee degree. Flexible and accurate threshold can be adopted by scheduler through analyzing and learning from the execution logs of resources.

7. Conclusions

In a hybrid grid-cloud computing environment, the user has elasticity provided by public cloud resources that can be aggregated to the existing grid resource pool as necessary. Aimed at scheduling of deadline-sensitive scientific workflow, stochastic service model is adopt to model dynamic service capacity of grid resource and a metric called default risk of subtask is provided to judge whether virtual resources should be leaned from cloud providers. Then, a scheduling algorithm, DGESA, is put forward. The experimental results show algorithm achieves better performance than other algorithms on user's deadline guarantee. Overall, our work can be viewed as an optimization module that can be incorporated with any hybrid computing infrastructures.

Acknowledgements

This work presented in this work is supported by the Natural Science Foundation of China (grants No. 60970038 and 61272148), the Natural Science Foundation of Henan Province (Grant No. 14A520042) and the Science Foundation of Henan University, China (Grant No. 2012YBZR040). We are grateful for support from the High Performance Computing Center of Central South University.

References

[1] Gentzsch, W., Girou, D., Kennedy, A., et al., "DEISA-Distributed European Infrastructure for Supercomputing Applications," *Journal of Grid Computing*, Vol. 9, No. 2, pp. 259–277 (2011). doi: 10.1007/s10723-011-9183-2

[2] Iosup, A., Dumitrescu, C., Epema, D. H. J., Li, H. and Wolters, L., "How are Real Grids Used? The Analysis of Four Grid Traces and Its Implications," *Proc. of 2006 IEEE/ACM International Conference on Grid Computing*, Barcelona, Spain, Sep. 28–29, pp. 262–269 (2006). doi: 10.1109/ICGRID.2006.311024

[3] Information on http://aws.amazon.com/ec2/

[4] Information on http://www.eucalyptus.com/

[5] Menasc, D. A. and Casalicchio, E., "A Framework for Resource Allocation in Grid Computing," *Proc. of 2004 International Workshop on Modeling, Analysis, and Simulation of Computer and Telecommunications Systems*, Vollendam, The Netherlands, pp. 259–267 (2004). doi: 10.1109/MASCOT.2004.1348280

[6] Tsiakkouri, H. Z. E., Sakellariou, R. and Dikaiakos, M. D., "Scheduling Workflows with Budget Constraints," *Proc. of 2005 CoreGRID Workshop Integrated Research in Grid Computing*, Pisa, Italy Gorlatch, Nov. 28–30, pp. 347–357 (2005). doi: 10.1007/978-0-387-47658-2_14

[7] Yu, J. and Buyya, R., "A Budget Constrained Scheduling of Workflow Applications on Utility Grids Using Genetic Algorithms," *Proc. of 2006 IEEE International Symposium on High Performance Distributed Computing*, Paris, France, Jun. 19–23, pp. 19–23 (2006). doi: 10.1109/WORKS.2006.5282330

[8] Xu, M., Cui, L., Wang, H. and Bi, Y., "A Multiple QoS Constrained Scheduling Strategy of Multiple Workflows for Cloud Computing," *Proc. of IEEE Symposium on Parallel and Distributed Processing with Applications*, Aug. 9–12, Chengdu, China, pp. 629–634 (2009). doi: 10.1109/ISPA.2009.95

[9] Abrishami, S. and Naghibzadeh, M., "Deadline-Constrained Workflow Scheduling in Software as a Service Cloud," *Scientia Iranica, Transactions D: Computer Science and Engineering and Electrical Engineering*, pp. 680–689 (2012). doi: 10.1016/j.scient.2011.11.047

[10] Dong, F. and Akl, S. G., "PFAS: A Resource-Performance-Fluctuation-Aware Workflow Scheduling Strategies for Grid Computing," *Proc. of IEEE Symposium on Parallel and Distributed Processing*, Mar. 26–30, Long Beach, California, pp. 1–9 (2007). doi: 10.1109/IPDPS.2007.370328

[11] Zhao, H. and Sakellariou, R., "Advance Reservation Policies for Workflows," *Proc. of 2007 Workshop on Job Scheduling Strategies for Parallel Processing*, Jun. 17, Seattle, WA, USA, pp. 46–67 (2007). doi: 10.1007/978-3-540-71035-6_3

[12] Wieczorek, M., Siddiqui, M., Villazón, A., Prodan, R. and Fahringer, T., "Applying Advance Reservation to Increase Predictability of Workflow Execution on the Grid," *Proc. of 2006 E-Science and Grid Computing Conference*, Dec. 4–6, Amsterdam, The Netherlands, p. 82 (2006). doi: 10.1109/E-SCIENCE.2006.261166

[13] Information on http://www.opennebula.org/doku.php

[14] Deelman, E., Singh, G., Livny, M., Berriman, B. and Good, J., "The Cost of Doing Science on the Cloud:

The Montage Example," *Proc. of ACM/IEEE Supercomputing Conference*, Nov. 15–21, Austin, Texas, USA, pp. 1–12 (2008). doi: 10.1109/SC.2008.5217932

[15] Kim, H., Khamra, Y. el, Rodero, I., Jha, S. and Parashar, M., "Autonomic Management of Application Workflows on Hybrid Computing Infrastructure," *Scientific Programming*, Vol. 19, No. 2, pp. 75–89 (2011).

[16] Ostermann, S., Prodan, R. and Fahringer, T., *Cloud Computing: Computer Communications and Networks*, Springer Press, Berlin, pp. 179–184 (2010). doi: 10.1007/978-1-84996-241-4_11

[17] Bittencourt, L. F., Senna, C. R. and Madeira, E. R. M., "Enabling Execution of Service Workflows in Grid/Cloud Hybrid Systems," *Proc. of 2010 IEEE Symposium on Network Operations and Management Symposium*, Apr. 19–23, Osaka, Japan, pp. 343–349 (2010). doi: 10.1109/NOMSW.2010.5486553

[18] Ramakrishnan, L., Koelbel, C., Kee, Y., Wolski, R., Nurmi, D., Gannon, D., Obertelli, G., YarKhan, A., Mandal, A., Huang, T. M., Thyagaraja, K. and Zagorodnov, D., "VGrADS: Enabling E-Science Workflows on Grids and Clouds with Fault Tolerance," *Proc. of 2009 ACM/IEEE Conference on High Performance Computing, Networking, Storage and Analysis*, Nov. 14–20, Portland, OR, USA, pp. 369–376 (2009). doi: 10.1145/1654059.1654107

[19] Nurmi, D., Brevik, J. and Wolski, R., "QBETS: Queue Bounds Estimation from Time Series," *Proc. of 2007 ACM International Conference on Measurement and Modeling of Computer Systems*, Jun. 12–16, San Diego, CA, USA, pp. 379–380 (2007). doi: 10.1145/1254882.1254939

[20] Vecchiola, C., Calheiros, R. N., Karunamoorthy, D. and Buyya, R., "Deadline-Driven Provisioning of Resources for Scientific Applications in Hybrid Clouds with Aneka," *Future Generation Computer Systems*, Vol. 28, No. 1, pp. 58–65 (2012). doi: 10.1016/j.future.2011.05.008

[21] Bittencourt, L. F., Senna, C. R. and Madeira, E. R. M., "HCOC: a Cost Optimization Algorithm for Workflow Scheduling in Hybrid Clouds," *Journal of Internet Services and Applications*, Vol. 2, No. 3, pp. 207–227 (2012). doi: 10.1007/s13174-011-0032-0

[22] Liu, X., Yang, Y., Jiang Y. C. and Chen, J. J., "Preventing Temporal Violations in Scientific Workflows: Where and How," *Transactions on Software Engineering*, Vol. 37, No. 6, pp. 805–825 (2010). doi: 10.1109/TSE.2010.99

[23] Iosup, A., Jan, M., Sonmez, O. O. and Epema, D. H. J., "The Characteristics and the Performance of Groups of Jobs in Grids," *Lecture Notes on Computer Science*, Vol. 4641, pp. 382–393 (2007). doi: 10.1007/978-3-540-74466-5_42

[24] Information on http://aws.amazon.com/ec2/

[25] Jøsang, A. and Haller, J., "Dirichlet Reputation Systems," *Proc. of 2007 International Conference on Availability, Reliability and Security*, Vienna, Austria, Apr. 10–13, pp. 112–119 (2007). doi: 10.1109/ARES.2007.71

[26] Wang, X., Ding, L. and Bi, D. W., "Reputation-Enabled Self-Modification for Target Sensing in Wireless Sensor Networks," *IEEE Transactions on Instrumentation and Measurement*, Vol. 59, No. 1, pp. 171–179 (2010). doi: 10.1109/TIM.2009.2022445

[27] Deelman, E., Singh, G., Su, M. H., Blythe, J., Gil, Y., Kesselman, C., Mehta, G., Vahi, K., Berriman, G. B., Good, J., Laity, A., Jacob, J. C. and Katz, D. S., "Pegasus: a Framework for Mapping Complex Scientific Workflows onto Distributed System," *Scientific Programming Journal*, Vol. 13, No. 3, pp. 219–237 (2005).

[28] Yu, J., Buyya, R. and Tham, C. K., "Cost-Based Scheduling of Scientific Workflow Applications on Utility Grids," *Proc. of 2005 International Conference on E-Science and Grid Computing*, July 5–8, Melbourne, Australia, pp. 140–147 (2005). doi: 10.1109/E-SCIENCE.2005.26

[29] Yuan, Y. C., Li, X. P., Wang, Q. and Zhu, X., "Deadline Division-Based Heuristic for Cost Optimization in Workflow Scheduling," *Information Science*, Vol. 179, No. 15, pp. 2562–2575 (2009). doi: 10.1016/j.ins.2009.01.035

[30] Information on http://www.cloudbus.org

[31] Information on https://www.grid5000.fr/

[32] Juve, G., Chervenak, A., Deelman, E., et al., "Characterization and Profiling Scientific Workflows," *Future Generation Computation Systems*, Vol. 29, No. 3, pp. 682–692 (2013). doi: 10.1016/j.future.2012.08.015

[33] Yeo, C. S. and Buyya, R., "Pricing for Utility-Driven Resource Management and Allocation in Clusters," *International Journal of High Performance Computing Applications*, Vol. 21, No. 4, pp. 405–418D (2007). doi: 10.1177/1094342007083776

Manuscript Received: Feb. 18, 2013
Accepted: Jan. 12, 2015