NEWS: Optimizing the Service Performance for Workflows with Replications in Virtualized Cloud Environments

Haoran Ji, Yuping Sun, Guoqing Wu, Yunpeng Wu, Sheng Xu and Huanlu Cao

Institute of NBC Defence, Beijing, 102205, China.

E-mails: ahrji@nudt.edu.cn, bzldl_fh@sina.com, cwugq2009@163.com, dypwu_cn@163.com, exsd163xx@163.com, fcaohuanlu@163.com

Abstract. Cloud computing environments facilitate applications by providing virtualized resources. It is attracting increasing interests for its elasticity and high quality of service, especially in “big time”. For applications, execution time is crucial for the quality of service, because works finished after the deadlines will be useless. Nevertheless, typical cloud environments do not present regular performance in terms of execution and data transmission. And cloud service providers cannot guarantee the performance of execution time, which would take large default risk. In this paper, a performaNce guarantee Workflow Scheduling (NEWS) algorithm is proposed to address these issues by taking replications based on reconstructed workflow. Characteristics of these issues are summarized to further describe the problems and challenges, which have always been ignored. The algorithm employs the parameter of partial critical path to optimize the workflow scheduling performance. On the analysis of challenges brought up by taking replications, NEWS also optimizes the choice of tasks and the number of replications. Simulation experiments show that the proposed algorithm reduces the variation of total execution time, which makes it possible to provide performance guarantee services.

1. Introduction
Cloud computing is a paradigm of heterogeneous computing, distributed computing and pervasive computing. It is attracting increasing interests for its scalability and high quality of service (QoS) on accomplishing the big data. Clients enjoy their services delivered by the clouds and pay for their uses by the pay-as-you-go model, while the resources in clouds seems infinite. Clouds provide dynamic scaling according to the needs of the applications. Nevertheless, typical cloud environments do not present regular performance in terms of execution and data transfer time. The variation caused by technological and strategic factors can up to 30% for execution time and 65% for data transfer time [1]. So, it is necessary for cloud service providers to employ a new kind of services with performance guarantee in the future. For the tasks with precedent constraints, which always described as a workflow, the execution time variation would be much bigger. Workflows are one kind of typical applications, especially in scientific researches, such as astronomy, biology and quantum physics. In this class of workflows, the deadline, which should be feasible for execution, is strictly requested and missing the deadline would take huge damage. Direct acyclic graph (DAG) is extensively applied in the programming paradigms available for the development of scientific applications, in which the nodes represent tasks and the vertices represent dependencies among tasks. All above make workflow scheduling with performance guarantee be one of the most challenging problems in commercial clouds.
To address these issues, one of the most efficient methods is replication. Parallel execution of some tasks can reduce the possibility of executing delay, while it takes extra expense. More replications mean smaller variation of execution time and bigger budget. Taking fewer replications leads to fewer expenses. When the possibility of execution time is higher than a threshold, it is confident of the cloud to give a regular performance. When decide to take one replication, the task choice should be deeply considered. And how many replications should be executed for a task can also be thought over. Besides, all these make challenges to scheduling. To the best of our knowledge, few works have well solved these problems. First, some existing methods in workflow scheduling ignore some special tasks with a priority. Then, some existing methods do not take the choice of multi-replications into account, which means making more than two replications to further enhance the performance stability. Thirdly, the others neglect the changes caused by replications during scheduling.

In this paper, a performance guarantee workflow scheduling (NEWS) algorithm is proposed, which optimize the makespan with as few replications as possible. The proposed algorithm utilizes the idle time of provisioned resources for replications in order to reduce the budget. And also, deadline of the workflow should be rational set. The purpose of NEWS is to ensure the possibility of execution time higher than the preset threshold while meeting the budget constraint.

The remainder of the paper is organized as follows. Section 2 introduces some related works. Section 3 gives basic descriptions of the scheduling model. Section 4 describes research motivation and proposes our contributions on optimizing the choice. In Section 5, the NEWS algorithm is proposed to cope with the problem. Experiments are used to evaluate the performance of NEWS in Section 6. Finally, a conclusion is given and further work is presented in Section 7.

2. Related Work
In the field of scheduling in heterogeneous environment, Bajaj et al. introduced a task duplication-based scheduling algorithm for network of heterogeneous systems (TANH), which provides optimal results for applications represented by directed acyclic graphs (DAGs). They provided a simple set of conditions on task computation and network communication time that could be satisfied [2]. Topcuoglu presented two novel scheduling algorithms, which were called the Heterogeneous Earliest-Finish-Time (HEFT) algorithm and the Critical-Path-on-a-Processor (CPOP) algorithm [3]. Rahman et al. proposed a dynamic critical path (DCP) based workflow scheduling algorithm that determines efficient mapping of tasks by calculating the critical path in the workflow task graph at every step [4]. Abrishami et al. proposed a new QoS-based workflow-scheduling algorithm named partial critical paths (PCP), which makes efforts on minimizing the cost of workflow execution while meeting deadline constraints [5]. After that he extended the PCP algorithm and proposed two workflow scheduling algorithms, which called Infrastructure as a Service (IaaS) Cloud Partial Paths (IC-PCP), and IaaS Cloud Partial Critical Paths with Deadline Distribution (IC-PCPD2) [6].

In theory, the VMs should present regular ability for the same task, while all above works are set on this assumption. To check the knowledge, Jackson et al. [1] provided the broadest evaluation of the application performance on virtualized cloud computing platforms and find that there is a 30% variability seen in computing time and 65% variability in communication. Therefore, it is hard for cloud provider to promise the clients an exact execution time. For these issues, Calheiros and Buyya [7] proposed an algorithm that uses idle time of provisioned resources and budget surplus to replicate tasks to mitigate effects of performance variation of resources. Then, Reynolds et al. [8] took the method of replicating slow tasks to increase the chance of an early completion of the workflow. Sirvent [9] made use of task replication techniques for workflow applications, trying to achieve both fault tolerance and speeding up of the computation. These works demonstrate the effectiveness of replication on optimizing the makespan performance.

This paper aims to address the same issues as [7] and employ the ideas in workflow scheduling as [9]. However, [7] doesn't cope with the issues of taking multi-replications and [9] cannot deal with the challenges brought up by taking replications.
3. Problem Formulation
The problem can be normally described by four aspects of tasks, services, objectives and constraints.

3.1. Task Model
A workflow application is modelled as a DAG, denoted as $G=(T,E)$, where $T = \{t_1, t_2, ..., t_n\}$ is a set of tasks and $E$, denoted as $e_{ij} = (t_i, t_j)$, is a set of dependencies between tasks. If $t_i$ is parent task of $t_j$ and $t_j$ is child task of $t_i$, denoted as $t_i = \text{parent}(t_j)$ and $t_j = \text{child}(t_i)$. There are two important parameters as the number of parent tasks, denoted as $N(\text{parent}(t_i))$, and the number of child tasks, denoted as $N(\text{child}(t_i))$. A workflow has an entry task, denoting as $t_{\text{entry}}$, and an exit task, denoting as $t_{\text{exit}}$, which sometimes may be a “dummy” task. Each task has a start time ($ST$), a finish time ($FT$) and an execution time ($ET$). For tasks in workflow, the input and output will be well defined. So it is quit feasible to give a predicted $ET$. Besides, transfer time ($TT$) corresponds to the communication between tasks. The time duration from $ST$ of $t_{\text{entry}}$ to $FT$ of $t_{\text{exit}}$ is the makespan of the workflow. Also, there are several theoretical time as the earliest start time, denoting as $EST$, and the latest finish time, denoting as $LFT$. Link length ($L$) is used to describe the number of tasks to $t_{\text{exit}}$. Each $G$ has a deadline, $d(G)$, which limits the LFT of workflow. During scheduling in this work, $t_i$ may have replications, denoted as $R(t_i) = \{t^1_i, t^2_i, ..., t^n_i\}$. Each $t^j_i$ has the same properties as $t_i$.

3.2. Service Model
Normally, clouds provide services, denoted as $S = \{s_1, s_2, ..., s_n\}$, by a set of virtual machines (VMs), denoted as $\text{VM} = \{v_{m_1}, v_{m_2}, ..., v_{m_n}\}$, to clients. A service $s_{ij}$ describes the assignment of $v_{m_j}$ to $t_i$ as $s_{ij} = (t_i, v_{m_j})$. Services are charged by the time intervals and partial utilization of a time interval incurs charge for the whole interval. We set the price of per resource use during one time unite as $\omega$. There exists an initial charge for providing the services in forms of money. The expense of keeping services would be much cheaper than that of initializing the services.

3.3. Objectives
In this case, the primary objective, denoted as $O$, is providing stable performance with as short makespan as possible. Makespan is the time between $EST(t_{\text{entry}})$ and $LFT(t_{\text{exit}})$, which can be computed by calculating the sum of $ET$ of tasks in critical path ($CP$). It can be described as:

$$makespan = \sum_{t_i \in CP} ET(t_i)$$  \hspace{1cm} (1)

Although replications will make service performance more regular, the performance guarantee can be described as the possibility of makespan shorter than a certain value is bigger than a threshold:

$$O = P(\text{makespan} \leq d) \geq \xi_d$$  \hspace{1cm} (2)

where $d$ is the certain value and $\xi_d$ is the threshold of performance stability required, which is related with $d$. Normally, in scheduling, we endeavour to meet the constraint by controlling the performance of each task. For a task, there is a theoretic $ET$, denoted as $TET$, which can be computed. For $t_i$, whose link to $t_{\text{exit}}$ as $L_i$ with $k$ tasks, there is $d'(t_{\text{entry}}) = d - TET(G)$ and assume

$$\tau_i = \frac{TET_i}{\sum_{k} TET_i}$$  \hspace{1cm} (3)

Similarly, there is $d'(t_i) = d - \text{EFT}(t_i) - TET(\sum_{k} t_i)$. The workflow’s deadline can be satisfied by meeting following objective for each task:

$$O_i = P(ET_i \leq \tau_i \cdot d'(t_i)) \geq \xi$$  \hspace{1cm} (4)
3.4. Constraints

As mentioned above, the budget constraint should also be considered. Based on the pricing model of most clouds, clients employ clouds by paying for services. Clients pay for time intervals of cloud resources they reserve. Set the number of reserved resources as \( N \) and the reserved time intervals of \( v_{m_i} \) as \( RT_i \). So we can get the budget constraint:

\[
\sum_{i} \omega \cdot RT_i \leq \psi
\]  

(5)

where \( \psi \) is the budget limitation, which can be treated as a limitation of replication number.

4. Motivation

For replications, one of the most challenging issues is making optimization on the choice of tasks for replications and the number of replications. Assume that the distribution of possible \( ET \) for each task obeys probability density function \( f(x) \). To simplify the analysis, assume that, for tasks in workflow, \( f(x) \) obeys the same function as:

\[
f(x) = \frac{1}{\theta_H - \theta_L}
\]  

(6)

where \( \theta_H \) is the upper limit and \( \theta_L \) is the lower limit of \( x \).

4.1. Replication task choosing

In this paper, four kinds of tasks are set with higher priority, which should be executed with replications. These tasks influence the total performance more than normal ones. We set the priority by following rules:

- Rule 1: The tasks with more children tasks. For example, in Figure 1(a), the lateness of task \( A \) would delay the start time of \( B_1 \) and \( B_2 \). Therefore, with the growing number of children tasks, it would be more likely to delay the total performance.
- Rule 2: The parent task of which has more parent tasks. For example, in Figure 1(b), the latest LFT(\( A_i \)), which \( i=1,2 \), decides the EST(\( B \)). As the number of parent tasks would influence the possibility distribution of their child task as the \( ET \) tends to last longer. The more parent tasks \( B \) has, the higher probability \( B \) has a later start time.
- Rule 3: The tasks with longer \( LL \). Tasks with longer \( LL \) means longer time before FT(\( t_{exit} \)). Its lateness would be more likely to delay the LFT(\( t_{exit} \)), which equals to the makespan of workflow. As described in Figure 1, on the assumption of task \( C \) and \( D \) with the same task length, task \( C \) would influence the total performance more than task \( D \).
- Rule 4: The tasks with longer \( ET \). As shown in Figure 1(c), \( ET(A) \) is longer than \( ET(B) \). As introduced in [1], the performance variation is related with the task length. Assuming the clouds generate 20% performance fluctuation, task \( A \) would have bigger lateness than task \( B \).

![Figure 1. Example of important tasks which should take replications.](image)

4.2. Challenges of taking replications

Among the existing approaches, the EIPR algorithm [8] works with the closest assumptions to the system and application models. But it doesn't take the issues of how replications affect the scheduling into account. For example, in Figure 1(b), assuming there are services \( s(s_{A_1}, v_{m_1}) \) and \( s(s_{A_2}, v_{m_2}) \). If \( A_1 \) is finished earlier than \( A_2 \), it is better to execute \( B \) on \( v_{m_1} \).
The replications can be treated as parallel tasks [10], of which, unlike the DAG model, the minimum FT decides the EST of following tasks. This paper solves the problems aforementioned by reconstructing the workflow with replications. For instance, a sample workflow modeled as Figure 2, Task A is a virtualized task whose ET=0. Based on solving the issues of determining the tasks for replications and the number of replications, the problem can be describes as provisioning services to the tasks of the developed DAG (DDAG) model.

5. Scheduling Algorithm
To provide performance guarantee services, a performaNce guaranteE workflow Scheduling (NEWS) algorithm is proposed. The goal of proposed algorithm is to shorten the execution time of workflow applications in actual unstable virtualized clouds while meeting the deadlines, which typically offers high availability but significant performance variation. The algorithm performs two distinct steps:

- Workflow Reconstruction: the algorithm chooses tasks for replications and reconstructs the workflow (normally as a DAG) to a developed workflow (DDAG).
- Resources Assignment: the algorithm proposes an enhanced PCP algorithm to optimize the total performance on makespan by assigning higher performance VMs to the critical tasks.

5.1. Workflow Reconstruction

Algorithm 1. Workflow Reconstruction.

```
Input: Workflow G
Output: Developed Workflow G'
for each t_i \in G do
    Scan the workflow and compute the parameters:
    V(rules_1) \leftarrow N(parent(t_i));
    V(rules_2) \leftarrow N(child(t_i));
    V(rules_3) \leftarrow ET(t_i);
    V(rules_4) \leftarrow Li(t_i);
    V(t_i) \leftarrow u_1 \cdot V(rules_4) + u_2 \cdot V(rules_2) + u_3 \cdot V(rules_3) + w_4 \cdot V(rules_4);
    Get \pi_G of all tasks;
    Compute replication number \{n_1, n_2, n_3, n_4\} based on V(t_i);
    Add replication(t_i) to T, T';
    Add e(parent(t_i), replication(t_i)) to E, E';
end
Add dummy entry task e_{entry} and exit task e_{exit} to G';
return Developed workflow G' = (T', E');
```

More replications present more regular ET performance while taking more expense. Additionally, we find that taking an extra replication at a small replication number would make better optimization than that at a bigger number. Therefore, there are an optimal number for replications. In this paper, we employ the tactic of taking \{n_1, n_2, n_3, n_4\} replications. The algorithm makes decision by giving each rule a priority weight, denoted as w_1, w_2, w_3 and w_4. The process has been presented in Algorithm 1,
5.2. Resources Assignment

After generating the DDAG, NEWS would make an initial resource assignment. Based on related works, our algorithm develops the partial critical path method (PCP), which has been demonstrated to be very efficient and effective, for resource assignment. One of the core concepts is critical parent: the critical parent of $t_i$, denoted as $\text{CriticalParent}(t_i)$, is the unscheduled parent task of $t_i$ that has the latest data arrival time, described as $EFT(t_p) + TT(e_{pl})$, at $t_i$ [6]. Then we get the concept of PCP:

**Definition 1.** Partial Critical Path of task $t_i$ is:
- $t_i$ if $t_i$ does not have any unassigned parents.
- consists of $t_i$, the Critical Parent $t_p$ of $t_i$ and the Partial Critical Path of $t_p$ if it has any unassigned parents.

NEWS searches the PCP and assigns it with the best resources. Once tasks in a PCP are scheduled, NEWS deletes them from the task list and repeats the search based on the rest of tasks. For instance, in Figure 2, NEWS gets a PCP as $A_1 \rightarrow C \rightarrow F_2 \rightarrow G$. After removing these tasks, the workflow would be translated as Figure 3. Then we get the second PCP as $A_2 \rightarrow B \rightarrow E$. Similar until all tasks have been scheduled. For each $PCP$, the proposed algorithm assigns the same $vm$.

![Figure 3. Remaining unscheduled tasks after PCP of $A_1 \rightarrow C \rightarrow F_2 \rightarrow G$ is scheduled.](image)

**Algorithm 2.** Resources Assignment.

```
Input: Developed Workflow $G'$
Output: Scheme of services $S$
1 Set $LFT(t_{exit}) = \text{deadline}$;
2 Set $EST(t_{entry}) = 0$;
3 Set $Status(t_{entry}) = \text{scheduled}$, $Status(t_{entry}) = \text{scheduled}$;
4 while all tasks in $G$ are scheduled do
5 for each $t_i \in G$ do
6 Compute $ET(t_i), LFT(t_i), EST(t_i)$;
7 end
8 for each $t_i$ do
9 if $LFT(t_i)$ is latest then
10 Set scheduling task $t = t_i$;
11 end
12 end
13 while there exists unscheduled parent of $t$ do
14 Add $\text{CriticalParent}(t_i)$ to the partial critical path, $PCP$;
15 $t = \text{CriticalParent}(t_i)$;
16 end
17 if $t$ has scheduled replication $t_j$ then
18 if for $s(t_i, vm_i)$, there is $p_{ik} > p_{jk} + \kappa$ then
19 $vm_i$ is unavailable for $t_i$;
20 end
21 end
22 Scan all available $VMs$ for $PCP$;
23 Find $vm_j$ with the highest computing ability while meeting task requirements on $w_{i\lambda_j}$ and $w_{i\mu_j}$;
24 Mark $vm_j$ as busy during the $ET$ of $CP$;
25 Set $Status(CP) = \text{scheduled}$;
26 Add $s(\text{PCP}, vm_j)$ to the scheme;
27 end
28 return Scheme $S$;
```
6. Performance Evaluation

In this section, we conduct experiments to evaluate the performance of NEWS. And a competitive algorithm of EIPR is used, of which has been well evaluated in [7]. The performance of resource selection has been demonstrated in our previous work [11].

6.1. Experiment Setup

Experiments are conducted with the CloudSim toolkit. In these simulation environments, cloud consists of 100 hosts, which meets the capability distribution as Google cloud [12]. The number of VMs on a host is random generated while each VM meets the Amazon AWS EC2 standard instance types. To test the algorithm, we take three classic workflows [13] separately as Montage, Cybershake and LIGO. All of them are abstracted from actual applications. We adjust the workflow size by changing mProject, mdiffFit and mbackground. The workloads are set as the Amazon EC2 standard instance types by random. For each application, a soft deadline would be generated as defined in [6], which benefits the performance evaluation. First, a theoretical makespan will be computed as the total ET obtained with a scheduling policy that assigns each task to the most powerful VM. The deadline is then defined as 112.5% of the proportional theoretical runtime. In the following experiments, we set \( \omega = \{\omega_1, \omega_2, \omega_3, \omega_4\} = \{10^3, 10^2, 10^1, 10^0\} \). The tasks in the experiments present variable ETs with at most 30% positive variation. In this section, tasks finish at a random time between the variation range and each experiment is repeated 50 times.

6.2. Performance for Different Workflows

In this part, the workflows of Montage, Cybershake and LIGO are separately used to evaluate the performance of NEWS on the objective of equation (1). And the experiments are conducted on each workflow with the size from 200 to 1000. The replication tactic is set as \{1, 3, 5, 7\}. The experimental results are presented in Figure 4. In these figures, x axis is the workflow size, z axis is the makespan and y axis describes the distribution within repeated 50 times execution.

In Figure 4(a), makespan of NEWS on Montage is normally smaller than that of the competitive algorithm. We find that makespan grows less when workflow size is growing. That is because we change the workflow size by adding some kind of tasks which make little influence on the longest link length of the workflow. In Figure 4(b), the makespan of NEWS ranges from 975 to 982 while competitive algorithm ranges from 98 0 to 994. Obviously, NEWS presents more stability. In Figure 4(c), NEWS shows not so much better. It just makes around 5 time units optimizing. That is because of the special topological structure of LIGO. The path length of LIGO is much shorter than that of Montage and the topological characteristics of each task in LI GO are generally similar. So NEWS will prefer to take few replications as it treats all tasks with same significance. For these reasons, the execution performance brought up by taking replications decreases. To cope with this case, one effective method is to change the tactics of replication.

![Figure 4. Performance for three classic workflows.](image-url)
6.3. Performance for Different Replication Schemes

In this part, we study the influence of different replication tactics. The experiment is conducted on Montage workflow in size of 206 tasks. Tactic 1 denotes taking replications of \{5, 10, 15, 20\}. Tactic 2 denotes taking replications of \{1, 3, 5, 7\}. And, Tactic 3 denotes taking replications as the EIPR algorithm. x axis describes the distribution of makespan during 50 times experiments.

![Figure 5. Influence of ET variation.](image)

In Figure 5, makespan with replication tactic 1 presents much more stable performance than that with tactic 2 and 3. However, more replications consume more resources. Using tactic 3, NEWS can generally give a performance guarantee of 910 time units for the Montage used in this experiment. Obviously, within the budget, suitable tactics can be made to provide different quality of performance guarantee service (QoPS). In addition, workflow should also be considered since the influence of replication tactics on different workflows differs. NEWS can be further optimized by taking replication tactics fitting the applications.

7. Conclusion and Future Work

Clouds provide on-demand services to clients while hardly give performance guarantee. The paper addresses the issues of workflow scheduling with replications, which would be especially important in emergency. A novel algorithm, NEWS, is proposed to optimize the performance containing the stability and the makespan. The contributions of this paper can be enumerated as follows:

- In this paper, we make efforts on providing performance guarantee services for cloud computing with the method of replications. A mechanism of which tasks executed with replications and how many replications be taken is proposed, which has always been ignored in former researches, to further optimize the performance of clouds.
- In this paper, we introduce a workflow reconstruction method to solve the problem of ensuring the critical path for different relationships.
- In this paper, a novel workflow scheduling algorithm of NEWS is proposed which presents well performance on scheduling the services under the objective of performance guarantee while meeting the deadlines.

As future work, we will develop the NEWS algorithm by optimizing the taking dynamic replication tactics, which is based on the agreement of QoPS and workflows.

References

[1] K.R. Jackson et. al: Performance analysis of high performance computing applications on the Amazon Web Services cloud. 2nd International Conference on Cloud Computing Technology and Science (CloudCom), Indianapolis, USA, Nov.30-Dec. 1, 2010.
[2] Rashmi Bajaj. Improving scheduling of tasks in a heterogeneous environment, IEEE Transactions on Parallel and Distributed Systems, 15(2):107-118, Feb. 2004.
[3] H. Topcuoglu: Performance-effective and low-complexity task scheduling for heterogeneous computing. IEEE Transactions on Parallel and Distributed Systems, 13(3):260-274, Mar. 2002.
[4] M. Rahman, S. Venugopal, R. Buyya: A Dynamic Critical Path Algorithm for Scheduling Scientific Workflow Applications on Global Grids. 3rd IEEE International Conference on e-Science and Grid Computing, pp.35-42, Bangalore, Dec. 10-13, 2007.
[5] J. Saeid Abrishami et. al: Cost-driven Scheduling of Grid Workflows Using Partial Critical Paths. IEEE Transactions on Parallel and Distribution Computing, 23(8):1400-1414, 2012.
[6] Saeid Abrishami et. al: Deadline-constrained workflow scheduling algorithms for infrastructure as a Service clouds. Future Generation Computer Systems, 29(1):158-169, 2013.
[7] Rodrigo N. Calherios, Rajkumar Buyya: Meeting Deadlines of Scientific Workflows in Public Clouds with Tasks Replication. IEEE Transactions on Parallel and Distributed Systems, 25(7):1787-1796, 2014.
[8] C.J. Reynolds, et. al: Scientific workflow makespan reduction through cloud augmented desktop grids. in Proceedings of the 3rd International Conference on Cloud Computing Technology and Science(CloudCom), Athens, Greece, Nov.29-Dec.1, 2011.
[9] R. Sirvent, R.M. Badia, J. Labarta: Graph-based task replication for workflow applications. 11th IEEE International Conference on High Performance Computing and Communications, pp.20-28,Seoul, Korea, June 25-27, 2009.
[10] Jinghui Zhang et. al: Scheduling of scientific workflow in non-dedicated heterogeneous multicenter platform. The Journal of Systems and Software. 86(7):1806-1818, July 2013.
[11] Haoran Ji, Weidong Bao, Xiaomin Zhu, Su Yin: Profit-Oriented Scheduling Optimization for Workflow in Clouds. 2013 IEEE 10th International Conference on High Performance Computing and Communications, pp. 1922-1929, Nov. 10-13, 2013.
[12] Zitao Liu, Sangyeun Cho: Characterizing Machines and Workloads on a Google Cluster, 2012 41st International Conference on Parallel Processing Workshops, USA, Sep. 10-13, 2012.
[13] S. Bharathi, A. Chervenak, E. Deelman: Characterization of scientific workflows. 3th Workshop on Workflows in Support of Large-Scale Science, pp.1-10, Austin, TX, Nov.17, 2008.