Computer Engineering and Intelligent Systems
ISSN 2222-1719 (Paper) ISSN 2222-2863 (Online)
Vol.4, No.12, 2013
www.iiste.org

QoS-aware Scientific Application Scheduling Algorithm in Cloud Environment

Ayda Mazandarani (Corresponding author)
Mirdamad University of Gorgan, Gorgan, Iran
E-mail: ayda.mazandarani@gmail.com

Hossein Momeni
Gorgan University of Agricultural sciences and Natural Resources, Gorgan, Iran
E-mail: momeni@Gau.ac.ir

Abstract

Many complex scientific applications are modeled in the form of workflows to carry out large-scale experiments. Because of complexity of scientific processes, scientific workflows need intensive computation and data requirements. Clouds make opportunity for scientific that need high performance computing infrastructure. So scientific can run their application on cloud by their desired QoS. We propose an algorithm that able scientific to select execute plan based on their preference QoS, like time and cost. Proposed algorithm ranks the tasks in workflow and then use UPFF function for select accurate resource, based on user's QoS. We compared our proposed algorithm with the same work by several scenarios and results show proposed algorithm has better efficiency.

Keywords Scientific application, Workflow scheduling, Cloud computing

1. Introduction

The field of distributed and parallel computing has seen technologies rapidly grow from desktop computing, through grid computing, and now to cloud computing. All these technologies focus on delivering computing power to a large number of end-users in a reliable, efficient and scalable manner. More and less, the trend has been to deliver the computing power as a utility, much like how water and electricity is delivered to households these days.

Cloud computing is a kind of parallel and distributed computing systems that delivers infrastructure, platform and software as a service, which are made available as services in a pay-as-you-go model to consumer. These services are referred to as infrastructure as a service (IaaS), platform as a service (PaaS), software as a service (SaaS). In [3] Buyya et al. define a cloud as a "type of parallel and distributed system consisting of a collection of interconnected and virtualized computers that are dynamically provisioned and presented as a one or more unified computing resources based on service-level agreements". Clouds try to make opportunity to the users all over the world to be able access the services on demand, according to their desired quality of service requirements. So it offers lots of benefits for companies by decreasing management and maintenance costs from leasing IT infrastructure from cloud providers.

Many scientific applications in the field of astronomy, gravitational-physics, computational biology, climate modeling, and life-sciences have used workflow technology to carry out large-scale experiments. Scientific applications are typically modeled as workflows that consist of tasks, data, control sequences and data dependencies [7].

Because of complexity of scientific process, these applications should be usually run on the large and distributed computing environments like cloud environment. Clouds present a chance for scientists whom need high-performance computing infrastructure for their experiments[9]. Most of the time, applications are represented as a scientific workflows that can manage many activities and work with lots of data. Scientific already using cloud computing that schedule these workflows onto distributed cloud resources for optimizing various objectives: minimize total makespan of the workflow, minimize cost and usage of network bandwidth, minimize cost of computation and storage, meet the deadline of application, and combination of objectives.

A data intensive computing environment consists of applications that produce, manipulate, or analyze data in the range of hundreds of megabytes (MB) to petabytes (PB) and beyond. A data intensive application workflow has

comparatively higher data workloads to manage than its computational load. In the words, the requirements of resource interconnection bandwidth for transferring data outweigh the computational requirements for processing tasks. As a consequence, demands more time to transfer and store data as compared to execution time for tasks in the workflow. It is common to characterize the distinction between data intensive and compute intensive by defining a threshold for the computation to communication ratio (CCR). Application with lower values of this ratio is distinctly data intensive in nature [8].

The rest of paper is organized as follows: section 2 presents related work. In section 3, we describe the task-resource scheduling problem. In section 4, we present our scheduling algorithm. Section 5 presents an experimental evaluation of the performance our protocol. Section 6 concludes the paper and discusses some future work.

2. Related work

Workflow applications are commonly represented as a directed acyclic graph. The mapping of jobs to the compute-resources is an NP-complete problem in the general form. A survey of field of scientific workflows and techniques for managing and scheduling them are represented in [7]. [8] is a PHD theses that represented several scientific workflow scheduling algorithms in grid and cloud environments.

In [28] proposed an optimized scheduling algorithm to achieve the optimization or sub-optimization for cloud scheduling. In this algorithm an Improved Genetic Algorithm (IGA) is used for the automated scheduling policy. It is used to increase the utilization rate of resources and speed.

In [27] proposed an improved cost-based scheduling algorithm for making efficient mapping of tasks to available resources in cloud. This scheduling algorithm measures both resource cost and computation performance, it also Improves the computation/communication ratio.

In [29] proposed an SHEFT workflow scheduling algorithm to schedule a workflow elastically on a Cloud computing environment. The experimental results show that SHEFT not only outperforms several representative workflow scheduling algorithms in optimizing workflow execution time, but also enables resources to scale elastically at runtime.

In [18] presented a particle swarm optimization (PSO) based heuristic to schedule applications to cloud resources that takes into account both computation cost and data transmission cost. It is used for workflow application by varying its computation and communication costs. The experimental results show s that PSO can achieve cost savings and good distribution of workload onto resources.

In [19] proposed a market-oriented hierarchical scheduling strategy which consists of a service-level scheduling and a task-level scheduling. The service-level scheduling deals with the Task-to-Service assignment and the task-level scheduling deals with the optimization of the Task-to-VM assignment in local cloud data centers.

In [20] worked on multiple workflows and multiple QoS. They has a strategy implemented for multiple workflow management system with multiple QoS. The scheduling access rate is increased by using this strategy. This strategy minimizes the make span and cost of workflows for cloud computing platform.

In [21] presented the HEFT algorithm. This algorithm first calculates average execution time for each task and average communication time between resources of two successive tasks. Then tasks in the workflow are ordered (non-increasing) on a rank function. The task with higher rank value is given higher priority. In the resource selection phase tasks are scheduled in the order of their priorities and each task is assigned to the resource that can complete the task at the earliest time.

In [24], data resource replication and parallel retrieving from several location are used to reach efficient scheduling plan. ESMH algorithms based on retrieving from several data resources are presented that mapped tasks on resources, according to data retrieving time and task compute time.

3. Problem definition

A scheduling system model consists of an application, a target cloud computing environment, and a performance criteria for scheduling. We denote an application workflow as a directed acyclic graph (DAG) represented by G=(V,E), where V={T_1, … , T_n} is the set of tasks, and E represents the data dependencies between these tasks. The number of tasks are considered n. G is a n*n matrix of Directed acyclic graph that gained by scientific application. if the amount of $G_{i,j}$ is one, represent task T_i and task T_j are dependable and T_i is the parent of T_j, else if the amount of $G_{i,j}$ is zero, shows they aren't dependable. if tasks were dependable, the data of parent task T_i should be transmitted to a child task T_j. a task without any parent is called an entry task and a task without any

child is called an exit task. The size of output data of tasks are given in array D as D_i represent size of output data task T_i.

We assumed that we want to use some of resources in cloud for scheduling. We have a set of resources R = {1,… , m}. here, we consider the number of resources is m. Each resource has its feature like cost of running tasks or Cost_exe(R_i), cost of incoming data by resources or Cost_in(R_i), cost of sending data from resources or Cost_out(R_i), resource availability or Availability(R_i) and resource reliability or Reliability(R_i).

The estimated times for compute each task on each resource are given. W is the n*m matrix that represent the estimated times for execute each task on each resource. $W_{i,j}$ shows the estimated time for execute task T_i on resource R_j .

The objective function of our workflow scheduling problem is to determine the schedule plan to assign tasks of a given scientific application to target cloud resources such that it's been done in user's desired time and cost.

4. Proposed approach

We present our proposed approach as a scheduling algorithm. The key idea of our approach is based on HEFT algorithm [21]. We named the proposed approach as a QoS-aware Scientific Application Scheduling Algorithm (QSASA). In QSASA we try to preserve benefits of HEFT and also apply the data-intensive scheduling features in cloud environment. QSASA considers varied aspects of scientific workflow scheduling. Aspects like tasks dependency, tasks data size, compute time of tasks, data transfer time from parent task to child, workflow makespan, resources bandwidth, resources cost for compute, resources cost for input or output data, availability and reliability parameters.

Before presenting the proposed approach in section 4-1, we formulate the parameters and attributes that are used in proposed protocol. And in the section 4-2 we present our protocol.

4.1 Problem formulation

In this section, we present the metrics of comparison, the experiment setup and the results.In section 3 we explained the problem and the parameters and information that gained by given scientific application and target cloud resources. Also there are several attributes that we may use in the proposed protocol. In this section, we formulate the attribute that may used in proposed protocol.

As we said, W is the n*m matrix that represent the estimated times for execute each task on each resource and $W_{i,j}$ shows the estimated time for execute task T_i on resource R_j. before scheduling, the tasks are labeled with the average execution times. The average execution time of a task T_i is defined as

$$\overline{w_i} = \sum_{j=1}^{m} w_{i,j} / m \qquad (1)$$

Target cloud resources are connected and so considering the communication time should be occurred. Resource bandwidth or data transfer rates are stored in matrix B of size m*m. the amount of $B_{i,j}$ represents resource bandwidth between resource R_i and R_j. the communication time of the edge (i,k), which is for transferring data from task T_i (scheduled on R_g) to task T_k (scheduled on R_h), is defined by

$$communicate_{i,k} = \frac{D_i}{B_{g,h}} \qquad (2)$$

When both T_i and T_k are scheduled on the same resource, $communicate_{i,k}$ becomes zero since we assume that the intra-resource communication time is negligible when it is compared with the inter-resources communication time. Before scheduling, average communication times are used to label the edges. The average communication time of an the edges out from T_i, is defined by

$$\overline{communicate}_i = \frac{D_i}{\overline{B}} \qquad (3)$$

Where $\overline{B}$ is the average transfer rate among the target cloud resources.

It is necessary to reword some attributes like EST(T_i, R_j) and EFT(T_i, R_j) in [21] that represent earliest start time and earliest finish time task T_i on resource R_j, respectively. For the entry task T_{entry},

$$EST(T_{entry}, R_j) = 0 \qquad (4)$$

For the rest tasks in workflow, the EST and EFT values are computed respectively, starting from the entry task, as shown in (5) and (6), respectively. In order to compute the task T_i, all immediate predecessor tasks of T_i must

$$EST(T_i, R_j) = \max \{avail[j], \max (EFT(T_K, R) + Communicate_{i,k})\} / (Availability(R_j) * Reliability(R_j)), T_k \in pred(T_i) \quad (5)$$

$$EFT(T_i, R_j) = w_{i,j} + EST(T_i, R_j) \quad (6)$$

have been scheduled.

Where pred(T_i) is the set of immediate predecessor tasks of task T_i and avail[j] is the earliest time at which resource R_j is ready for task execution. The inner max block in the EST equation returns the ready time, means the time when all data needed by T_i has arrived at resource R_j.

EST_{min} shows minimum value of EST for task on target resources and EST_{max} shows maximum value of EST. To

$$EFC(T_i) = \sum_{k=1}^{p} \{((w_{i,j} * cost_{exe(R_j)}) + (communicate_{k,i} * cost_{in}(R_j)) + (communicate_{k,i} * cost_{out}(R_l))) / (Availability(R_j) * Reliability(R_j))\} \quad (11)$$

compute average values, we used $\overline{B}$ instead of B for bandwidth and shown by $\overline{EST}(T_i)$ and $\overline{EFT}(T_i)$.

EFC is another attribute that represent total cost for execute tasks containing run cost, input data cost and output data cost that are calculate by (11).

Where p is the number of parent tasks of task T_i. $communicate_{k,l}$ is the required time for transmitting data from parent tasks of T_i, T_k, that is mapped on resource R_l, to task T_i that mapped on resource R_j. EFC_{min} represents minimum value of EFC of task T on all target resources and EFC_{max} shows maximum value of EFC.

Tasks are sorted by their priority. Upward ranking [21] of task T_i are calculated recursively by (12).

$$Rank_u(T_i) = \overline{w}_i + \max (\overline{communicate}_i + Rank_u(T_j)), T_j \in succ(T_i) \quad (12)$$

Where succ (T_i) is the set of immediate successors of task T_i. the rank is computed recursively by traversing the task graph upward, starting from exit task. For exit task T_{exit}, the upward rank is calculated by (13).

$$Rank_u(T_{exit}) = \overline{w}_{exit} \quad (13)$$

4.2 Proposed Algorithm

We present QoS-aware scientific application scheduling algorithm (QSASA) as a proposed approach. We aimed to schedule scientific workflow on target cloud resources based on user's preferences. QSASA algorithm is shown in figure 1. and have two phases, ranking tasks and selecting resources. Average time values of execute tasks and edge communication are computed. Then values for EST, EFT and EFC for each task on all resources are calculated. According to calculated attributes and the parameters, we use upward ranking for ranking tasks in workflow and use User preference Fitness Function (UPFF) for accurate resource selection. And last tasks dispatched on resources for workflow execution.

BEGIN : QoS-aware Scientific Application Scheduling Algorithm (QSASA)

INPUT:

A matrix G represented scientific workflow graph

A array D represented data produced by tasks

A matrix B represented bandwidth between resources

A matrix W represented times for execute tasks on resources

A array Cost_exe represented cost of execute tasks on resources

A array Cost_in represented cost of input required tasks on resources
A array Cost_out represented cost of output required tasks on resources
A array Reliability represented reliability of resources
A array Availability represented availability of resources
PRE-COMPUTE:
Compute average time of execution for each task, $\overline{w_i}$
Compute average time of communication for each edge (task dependency), $\overline{communicate_{i,k}}$
Compute average value of earliest start time for each task, $\overline{EST}(T_i)$
Compute average value of earliest finish time for each task, $\overline{EFT}(T_i)$
Ranking tasks by upward ranking in [21] as Rank_upward list
While all the tasks aren't scheduled
Start for all resources in R
Calculate EFT for task T on all resources in R
Calculate Costs for all resources in R that their EFT was computed (EFC)
Determinate EST_{min} (T), EST_{max} (T), EFC_{min} (T), EFC_{max} (T)
Use UPFF function for select accurate resource
End
Map task T on the resource Rthat have minimum value of UPFF function
End
Distributed tasks on resources
END : QoS-aware Scientific Application Scheduling Algorithm (QSASA)

Figure 1.QoS-aware Sientific Application Scheduling algorithm (QSASA)

As we said, we used upward ranking [21] to make a priority list as Rank_upward list. UPFF function is used to find resources for tasks in Rank_upward list, respectively. based on the user's preference, we use different values for weight of time $weight_t$ and weight of cost $weight_c$. this two weight are in the range of [0, 1] and the sum of them is 1. For example if $weight_c$ be equal to 0.7, represents 70% user is concerned with cost and 30% is concerned with time. So the UPFF is a two variable function. As the time and cost aren't the same kind, we normalize these values in the range of [0, 1]. UPFF is defined by (14).

$$UPFF(T_i, R_j) = \left(weight_t \times \frac{EST(T_i,R_j) - EST_{min}}{EST_{max} - EST_{min}} + weight_c \times \frac{EFC(T_i,R_j) - EFC_{min}}{EFC_{max} - EFC_{min}} \right) / \left((Availability(R_j) * Reliability(R_j)) \right) \quad (14)$$

UPFF represents fitness of resource R_j for task T_i.the resourcewith minimum UPFF is selected and the task is mapped on it. the $weight_t$ and $weight_c$ make a chance to scientific for their desired QoS requirements.

5. Experiment evaluation

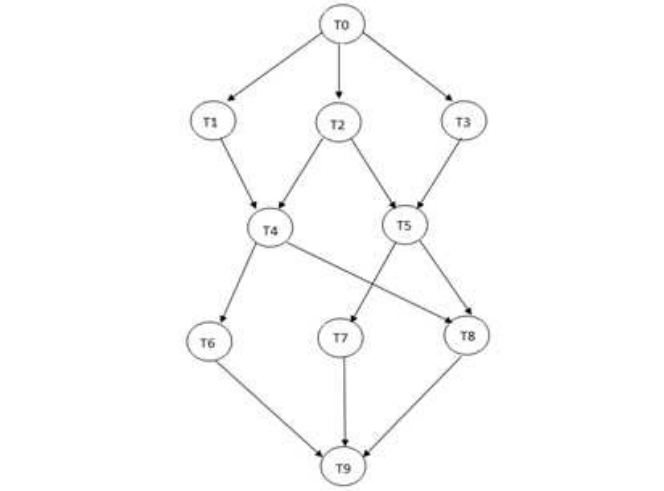
In this section, we present the metrics of comparison, the experiment setup and the results. In this evaluation, the value for weight of cost 0.7 and the value for weight time 0.3 are considered. We assume user is most interested in minimizing cost.

5.1 Performance metrics

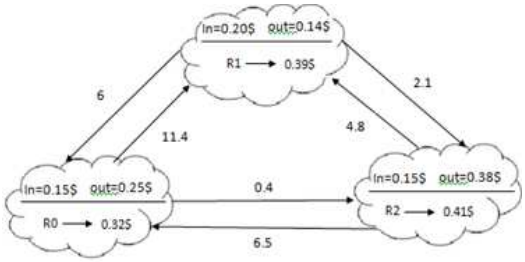
As a measure of performance, we used time and cost for complete execution of application as a metrics. We computed the total cost of execution of a workflow using two approaches: QSASA protocol and HEFT algorithm.

5.2Illustrative example

Figure 4(a) depicts a workflow structure with ten tasks, which are represented as nodes. The dependencies between tasks are represented as arrows. Each task generates output data after it has completed. These data are used by the task's children, if any. Figure 4(b) depicts three resources interconnected with varying bandwidth and having its own execution, input and output costs. Also the estimated average time for each tasks on all resources are given. In this example we consider that the resources are completely reliable and available.



$Data_0$	$Data_1$	$Data_2$	$Data_3$	$Data_4$	$Data_5$	$Data_6$	$Data_7$	$Data_8$	$Data_9$
100	56	512	418	98	540	890	99	245	645

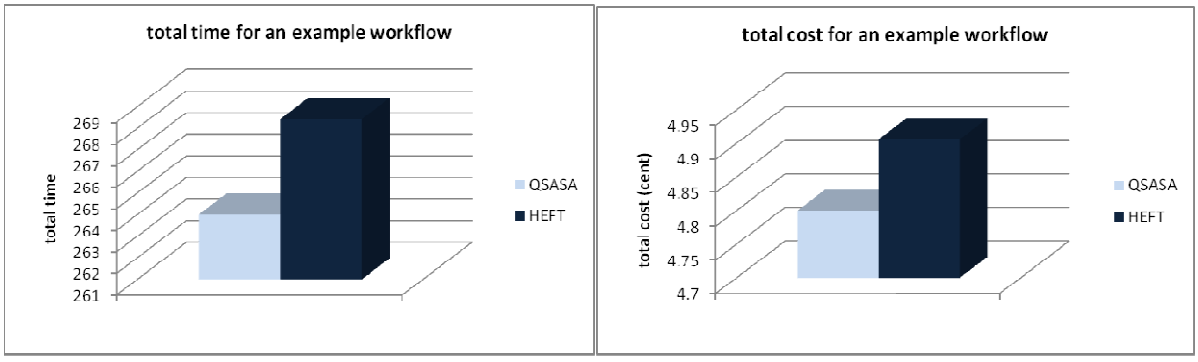


Resources / tasks	R0	R1	R2
T0	13	18	8
T1	24	13	21
T2	24	14	18
T3	21	16	21
T4	8	14	18
T5	18	11	18
T6	7	11	20
T7	8	9	13
T8	12	21	18
T9	25	25	13

(a) Workflow and the output data of each tasks (b) Resources and the estimated times for execute tasks

Figure 4. Illustrative example of workflow and target cloud resources

We use QSASA algorithm and HEFT algorithm on the example and the results are gained as shown in Figure 5 (a) and Figure 5(b). As you can see, QSASA algorithm has a higher efficiency than HEFT algorithm, both in time and cost perspective.



(a) Comparison of total cost (b) Comparison of total time

Figura 5. Comparison between QSASA and HEFT for an illustrative example workflow and target resources

5.3Data and Implemention

We have used several matrix and array that store the values like average time of execute tasks on resources, average communication cost between resources, data size of tasks, execute cost, input/output cost and so on.

The values for Cost_in(R_i) and Cost_out(R_i) resemble the cost of unit data transfer between resources given by Amazon CloudFront. We randomly use that values for each iterative of experiment. In some experiments, we use the Amazon EC2's pricing policy for different classes of virtual machine instances. As each task has its own data, the sum of all the data values varies according to the size of data we experiment from 64 MB to 1024 MB.

We assumed in this work, the $weight_c$ is equal to 0.7 and the $weight_t$ is equal to 0.3 and are fixed in all experiment. So we assumed user wants more to cost saving than time saving.

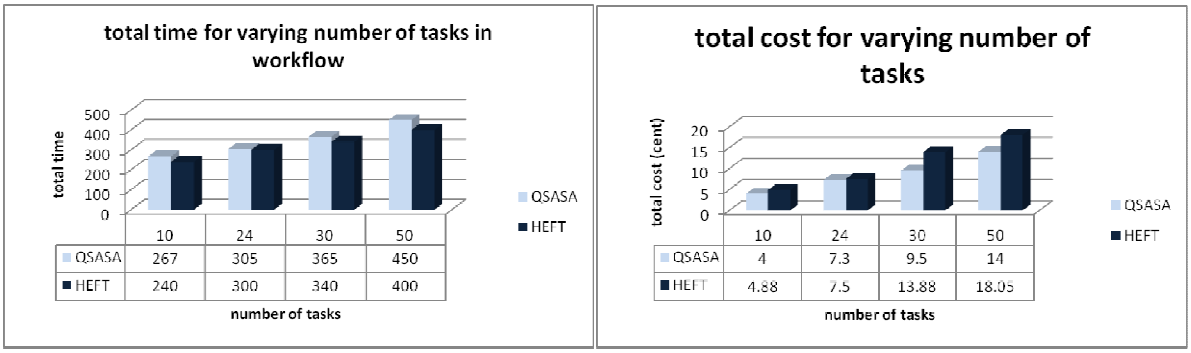
We use C++ environment to conduct our simulation experiment. In addition we implement HEFT algorithm, too. Also we have done twenty independent executions for all scenarios to gain better results.

5.4Experiment and Result

Each task in workflow has input/output data in varying sizes. We evaluate proposed protocol in different scenarios. We plot the graphs by varying the results obtained after twenty independent executions. In almost every execution, the x-axis parameters present number of tasks in workflow, total data size and costs range. The y-axis parameters present performance criteria like time and cost.

5.4.1 Variation in number of workflow tasks

In this scenario, we try to compare QSASA with HEFT algorithm when the number of tasks in workflow is varying from 10-50 tasks. We fixed the compute resource cost in the range 0.30-0.80$/hr and the communicate cost in the range 0.14-0.38$/hr, in the sub-section 5-4-1 and 5-4-2. by varying number of tasks in workflow, we compared SWASP and HEFT algorithm, time and cost perspective and the results depicted in Figure 6(a). and Figure 6(b). In both figures, x-axis represents number of tasks in workflow. Based on the results, QSASA algorithm has better performance.

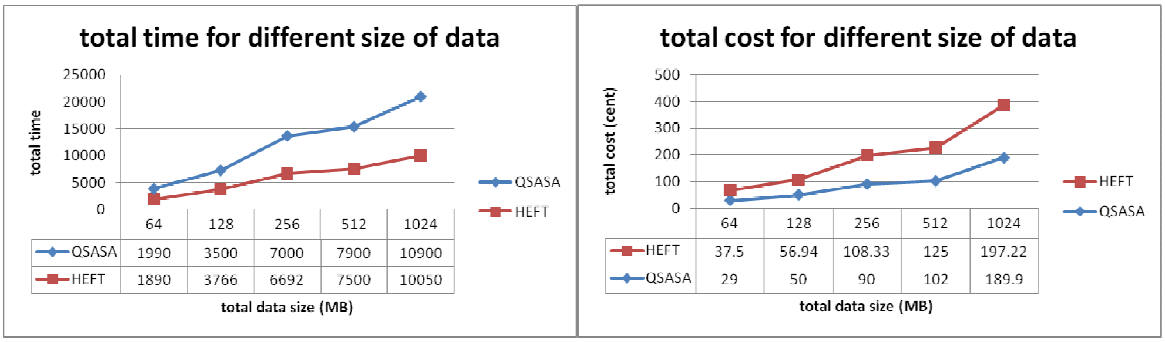


(a) Comparison of total time (b) Comparison of total cost

Figure 6. Comparison between QSASA and HEFT when while varying number of tasks in workflow

5.4.2 Variation in total data size of a workflow

We varied the size of total data processed by the workflow in the range 64—1024MB. By varying the data size, we compared the variance in total time and cost of execution, for two approaches as depicted in Figure 7(a) and Figure 7(b). Results show that QSASA has better performance. In some experiment, the QSASA's time criteria may be higher than HEFT, but usually the cost criteria is lower than.



(a) Comparison of total cost (b) Comparison of total time

Figure 7. Comparison between QSASA and HEFT when while varying number of tasks in workflow

5.4.3 Variation in resource cost

We experiment the performance of QSASA by varying the cost of computation of all target resources. This variation is practically justifiable as different cloud service providers (e.g. Amazon) can have varying pricing policies depending on the type and capabilities of their resources. Figure 8 depicts the change in total cost of

computation for different range of resource price. The workflow processed a total of 128MB of data and the x-axis represents range of resources cost. As QSASA consider the resource cost, Results show by varying resource cost, it has low variance.

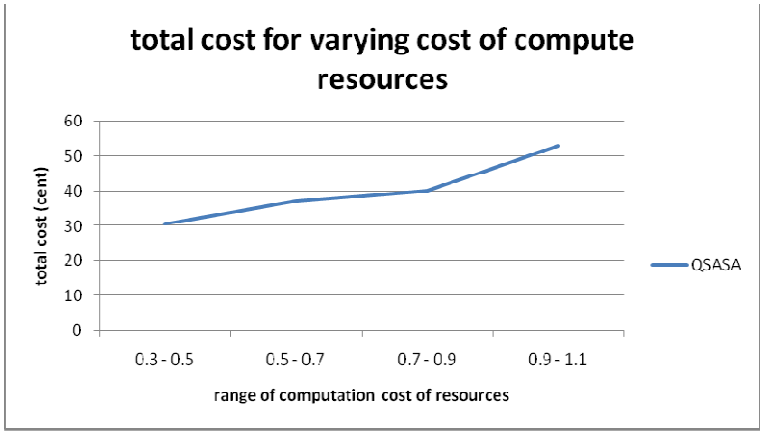


Figure 8. Total cost by QSASA when while varying cost of compute resources

5.4.4 Scheduling Failure variance

There are some communications between resources to execute workflow. This communication is done by resource bandwidth. Communications may encounter delay. We want to use a approach that consider this condition. According to the history of resources, values as reliability and availability factor are assigned to the resources. QSASA considers these factors. In Figure 9 amount of scheduling variation are shown. X-axis shows multiply of average values of reliability and availability.

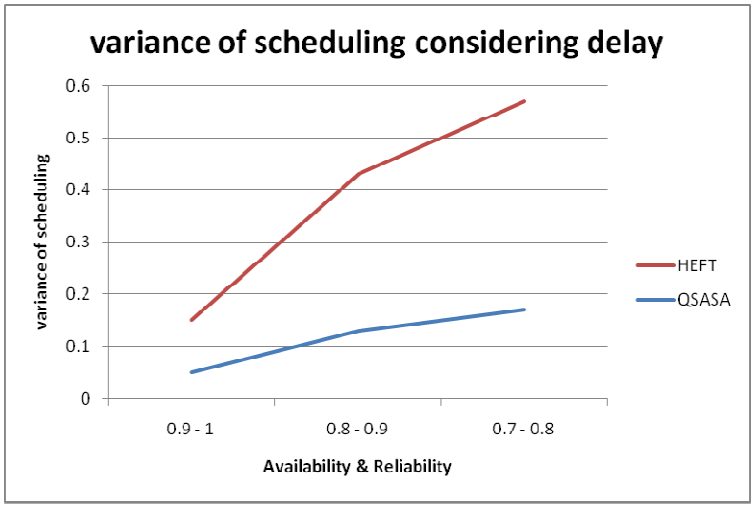


Figure 9. Comparison of scheduling failure variance between QSASA and HEFT when while varying availability and reliability of resources.

6. Conclusion and future work

In this work, we presented a scientific workflow scheduling algorithm on cloud resources as called QSASA, based on HEFT algorithm. QSASA contains two phases for ranking tasks and selecting resources. We used QSASA algorithm to minimize the total cost of execution of application workflows on cloud environment, based user's preference. In evaluation we consider the user is more interested to minimize cost. We compared the results obtained by our algorithm against HEFT algorithm. We found that QSASA achieve at least 15 percent improvement on cost saving and at last 4 percent more time as compared to HEFT for our experiments. In addition, QSASA considers reliability and availability factor of resources while scheduling.

As part of our future work, we would like to expand our work for independent compute resource and store resource. Also we want to work on scheduling scientific application on BIG DATA.

References

1. Kleinrock L (2003) An Internet Vision: The Invisible Global Infrastructure. Ad-Hoc Networks vol 1, no 1, pp 3–11
2. Vaquero LM, Rodero-Merino l, Caceres J, Lindner M (2008) A break in the clouds: towards a cloud definition. *SIGCOMM Computer Communication Review*, vol 39, pp 50–55
3. Buyya R, Yeo CS, Venugopal S, Broberg J, Brandic I (2009) Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the5th utility. Future Generation Computer Systems, vol 25, no 6, pp 599–616
4. Duncan D, Chu X, Vecchiola C, Buyya R (2009) The Structural of the New IT Frontier: Cloud Computing - Part I. CLOUDS Laboratory, Department of Computer Science and Software Engineering, the University of Melbourne, Australia
5. Vecchiola C, Duncan D, Buyya R (2009) The Structural of the New IT Frontier: Market Oriented Cloud Computing - Part Π. CLOUDS Laboratory, Department of Computer Science and Software Engineering, the University of Melbourne, Australia
6. Foster I, Zhao I, Raicu I, Lu S (2008) Cloud Computing and Grid Computing 360-Degree Compared.In : Grid Computing Environments Workshop, 2008 GCE '08, pp1-10
7. Pandy S, Buyya R (2009)Scheduling and Management Techniques for Data-Intensive Application Workflows. IGI Global, USA
8. Pendy S (2010) Scheduling and Management of Data Intensive Application Workflows in Grid and Cloud Computing Environments**.** Ph.D Thesis, University of Melbourne, Australia
9. Lee J, Tierney B, Johnston WE (1999) Data Intensive Distributed Computing; AMedical Application Example**.** in HPCN Europe ’99: Proceedings of the 7th InternationalConference on High-Performance Computing and Networking. London,UK:Springer-Verlag, pp 150–158
10. Zhao Y, Raicu I, Foster I (2008) ScientificWorkflow Systems for 21st Century, NewBottle or New Wine?.in SERVICES ’08: Proceedings of the 2008 IEEE Congress onServices - Part I. Washington DC USA: IEEE, pp 467–471
11. Buyya R, Pandey S, Christian V (2009) Cloudbus Toolkit for Market-Oriented Cloud Computing. Proceeding of the 1st International Conference on Cloud Computing (CloudCom 2009, Springer, Germany), Beijing, China
12. Juve G, Deelman E, Vahi K, Mehta G, Berriman B, Maechling P (2009) Scientific workflow applications on Amazon EC2. in E-Science Workshops, 2009 5th IEEE International Conference pp 59-66
13. Esteves R (2011) A Taxonomic Analysis of Cloud Computing. 1st Doctoral Workshop in Complexity Sciences, ISCTE-IUL/FCUL
14. Taylor IJ, Deelman E, Gannon DB, Shields M (2007)Workflows for e-Science: Scientific Workflows for Grids. 1 ed. Springer
15. Oliveira D, Baião F, Mattoso M (2010) Towards a Taxonomy for Cloud Computing from an e-Science Perspective. Cloud Computing: Principles, Systems and Applications, Heidelberg: Springer- Verlag
16. Juve G, Deelman E (2010) Scientific workflows and clouds. Crossroads v 16 (Mar.), p 14–18
17. Deelman E, Singh G, et al. (2008) The Cost of Doing Science on the Cloud: The Montage Example. in *SC'08* Austin, TX2008
18. Pandey S, Wu L, Guru S, Buyya R (2010) A Particle Swarm Optimization (PSO)-based Heuristic for Scheduling Workflow Applications in Cloud Computing Environments**.** Proceedings of the 24th IEEE International Conference on Advanced Information Networking and Applications (AINA 2010) Perth Australia
19. Wu Z, Liu X, Ni Z, Yuan D, Yang Y(2011)A market-oriented hierarchical scheduling strategy in cloud workflow systems. The Journal of Supercomputing pp 1–38
20. Xu M, Cui L, Haiyang W, Yanbing B(2009) A Multiple QoS Constrained Scheduling Strategy of Multiple Workflows for Cloud Computing. in Parallel and Distributed Processing with Applications 2009 IEEE International Symposium pp. 629-634
21. Topcuoglu H, Hariri S, Wu M. Performance effective and low-complexity task scheduling for heterogeneous computing IEEE Transactions on Parallel and Distributed Systems 13(3):260–274
22. Sakellariou R, Zhao H (2004) A Hybrid Heuristic for DAG Scheduling on Heterogeneous Systems. The 13th Heterogeneous Computing Workshop (HCW 2004) Santa Fe New Mexico USA

23. Bajaj R. Agrawal DP (2004) Improving Scheduling of Tasks in a Heterogeneous Environment. IEEE Transactions on Parallel and Distributed Systems 15:107-118
24. Pendy S, Buya R (2010) Scheduling Data Intensive Workflow Applications based on Multi-Source Parallel Data Retrieval in Distributed Computing Networks. the Computer Journal
25. Hoffa C, Mehta G, Freeman T, Deelman E, Keahey K, Berriman B, Good J (2008) On the Use of Cloud Computing for Scientific Workflows. in IEEE Fourth International Conference on eScience pp 640-645
26. Lin C, Shiyong L (2011) Scheduling Scientific Workflows Elastically for Cloud Computing. in Cloud Computing (CLOUD) 2011 IEEE International Conference on pp 746-747
27. Selvarani S, Sahasivam GS (2010) Improved Cost-Based Algorithm for Task Scheduling in Cloud Computing. Department of Information Technology Tamilnadu Colledge of Engineering Coimbatore India IEEE
28. Ge Y, Wei G (2010) GA-Based Task Scheduler for the Cloud Computing Systems. International Conference on Web Information Systems and Mining
29. Bajaj R, Agrawal DP (2004) Improving Scheduling of tasks in a Heterogeneous Environment. IEEE Transactions on Parallel and Distributed systems vol 15 pp 107-118