European Journal of Scientific Research
ISSN 1450-216X Vol. 89 No 2 October, 2012, pp.265-275
© EuroJournals Publishing, Inc. 2012
http://www.europeanjournalofscientificresearch.com

Time and Cost Optimization Algorithm for Scheduling Multiple Workflows in Hybrid Clouds

B. Arun Kumar
Research Scholar, Department Of Computer Science and Engineering
Faculty of Engineering, Karpagam University
E-mail: arunkumar.oct06@gmail.com
Tel: +91-9894096100

T. Ravichandran
Principal, Hindusthan Institute of Technology
E-mail:dr.t.ravichandran@gmail.com
Tel: +91-9944183119

Abstract

As the intent to promote Cloud Computing, evolves into genuine researches. Due to the raise in convention of many applications currently, there is a necessity for high processing and storage capacity along with the consideration of cost and instance use. To provide proficient resources, Cloud computing is been pioneered. In this paper Time and Cost Optimization for Hybrid Clouds (TCHC) algorithm is proposed to reduce the execution time and cost of multiple workflows scheduling. The users nowadays don't want to get stuck to their own cloud providers to execute or schedule the multiple workflows. Many organizations have their own private cloud, but when there is a need for extra resources they go for public cloud where they have been outlaid for their use. In case of dependent workflow scheduling, the switching between private and public cloud resources will lead to increased execution time and cost. The multiple requests made for the resources will result in increased bandwidth. As a remedy TCHC buffers the resource in the local resource pool, that might help if there is change in on demand resource price after an instant and it also reduces the requesting cost. The proposed strategy TCHC algorithm comes to the decision of deciding which resource should be chartered from public providers.

Keywords: Scheduling, Workflows, TCHC algorithm, Dependency, Hybrid cloud

1. Introduction

The usage of resources afford in grid computing is not adequate to use. As a remedy for this, Cloud computing becomes visible in a way that provides on-demand resources to the users, so as to proceed locally available computational power, delivering new computing resources whenever necessary. The Cloud computing environment provides resources dynamically whenever demanded by the user. The resource is utilized by the user without having enough knowledge about the technical details involved in the resource provider. We see Cloud Computing as a computing model, not a technology. In this model "customers" plug into the "cloud" to access IT resources which are priced and provided "on-demand". Over the last several years, virtual machines have become a standard object used to bring down the scalability of permanent resources.

Virtualization further enhances elasticity because it abstracts the hardware to the point where software stacks can be deployed and redeployed without being tied to a specific physical server. L. Grit, D. Irwin, (2006), H.N. Van (2009), M.Cardosa et al., (2009), F.Hermenier et.al.,(2009) virtualization provides a dynamic datacenter where servers provide a pool of resources that are united as needed, and where the relationship of applications to compute, storage, and network resources changes dynamically in order to meet both workload and business on demands. With application deployment decoupled from server deployment, applications can be deployed and scaled rapidly, without having to procure physical servers first.

Virtual machines have become the prevalent abstraction and unit of deployment because they are the least-common denominator interfaces between service providers and developers. Using virtual machines as object tool, it is adequate for 80 percent of application usage, and it helps to satisfy the user need to deploy and scale applications rapidly. Virtual appliances, virtual machines that include software that is moderately or fully configured to perform a specific task such as a Web or database server, further develop the ability to create and deploy applications in vertical domains.

The combination of virtual machines along with appliances as standard deployment objects is one of the key features of cloud computing. Essentially, IT resources are rented and shared among multiple leaseholders much as office space, apartments, or storage spaces used by leaseholder. Distributed over an Internet connection, the "cloud" replaces the company data center or server by providing the same service without the procurement of new permanent resources. Thus, Cloud Computing is simply IT services sold and delivered over the Internet.

Cloud offers three types of services *SaaS (Software as a Service)* provides all the functions of a sophisticated traditional application to many customers, often thousands of users, but through a Web browser, not a "locally-installed" application. It eliminates customer suspicions about application servers, storage, application development and related common concerns of IT. Highest-profile examples are Yahoo and Google, and VoIP from Vonage and Skype.

PaaS (Platform as a Service) delivers virtualized servers on which customers can run existing applications or develop new ones without having to worry about maintaining the operating systems, server hardware, load balancing or computing capacity. These vendors provide APIs or development platforms to create and run applications in the cloud – e.g. using the Internet. *IaaS (Infrastructure as a Service)* delivers utility computing capability, typically as raw virtual servers, on demand that customers configure and manage. IaaS is designed to replace the functions of an entire data center. This saves cost (time and expense) of capital equipment deployment but does not reduce the cost involved in configuration, integration or management and these tasks must be performed remotely.

Apart from these, we have the following Cloud computing infrastructure models:

1.1. Public Clouds

The On-demand services are managed by third party resources provider, and the applications requested from different customers are liable to be mixed together on the cloud's servers, storage systems, and networks. If a public cloud is employed with performance, security, and data locality in mind, the existence of other applications running in the cloud should be transparent to both cloud architects and end users. In reality, one of the benefits of public clouds is that they are much larger than a company's private cloud, offering the ability to provide on demand resources, and shifting infrastructure risks from the enterprise to the cloud provider, the above services are offered temporarily.

1.2. Private Clouds

The organization permanent infrastructure can also be meant as private cloud, it was built for the limited use of one client, providing the utmost control over data, security, and Quality of Service. The company owns the infrastructure and has control over how applications are deployed on it. These types of clouds can be built and managed by a company's own IT organization or by a cloud provider. This model gives companies a high level of control over the use of cloud resources while bringing in the capability needed to establish and operate in the environment.

1.3. Hybrid Clouds

Hybrid cloud combines both the public and private cloud models to attain elasticity at the user level. They can help to provide on-demand services, externally provisioned scale. The ability to augment a private cloud with the resources of a public cloud can be used to maintain service levels in the face of rapid workload fluctuations. Sometimes called "surge computing," a public cloud can be used to perform periodic tasks that can be installed easily on a public cloud. A hybrid cloud is designed by carefully to determine the best split between public and private cloud components. One of the problems to face is to determine when and how to split a workflow, which is composed of dependent tasks, to execute in private re- sources and in public resources

Cloud Computing vendors combine virtualization (one computer hosting several "virtual" servers), automated provisioning (servers have software installed automatically) and Internet connectivity technologies to provide the service. Consequentially, acquisition costs are low but tenants never own the technology asset and might face challenges if they need to "move" i.e. expand the current resource set up or end the service for any reason. Something that is often overlooked when evaluating Cloud Computing costs is the continued need to provide LAN services that are robust enough to support the Cloud solution. The resource providing costs need not be small always.

For example, if you have 6 or more workstation computers, you will probably need to continue to maintain a server in a domain controller role (to ensure name resolution), at least one switch (to connect all the computers to each other and the router), one or more networked printers, and the router for the Internet connection. But scheduling the workflow is a challenging task in Cloud Computing.

This paper, proposes a strategy in scheduling multiple workflows to the Cloud Computing paradigm. The TCHC algorithm decides which task to be scheduled to the public cloud, by determining the dependencies among the workflows. The algorithm decides the best split between the private and public cloud and also determines the task with the least cost and execution time to be scheduled in the public cloud. The cost is minimized by choosing minimum bandwidth to the public cloud.

2. Related Work

The need for fruition of e-Science, workflows are becoming larger and more computational demanding. With this, there is a demand for a scheduler which deals with multiple workflows in the same set of resources, thus the development of multiple workflow scheduling algorithm is necessary as discussed in Bittencourt, L.F., Madeira, (2010). Although the study of cost evaluation and scheduling problem is playing a major problem in the world of cloud computing, it is well analyzed about the different algorithm considering cost by Bittencourt, L.F., Senna,(2010) It is important to decide when and how to request these new resources to satisfy deadlines and/or to get a reasonable execution time, while minimizing the monetary costs involved, a stratagem to schedule service workflows in a hybrid cloud.

The stratagem aims at determining which task should be used to paid resources and what kind of resource should be requested to the cloud in order to minimize costs and execution time and meet deadlines. Both heuristic and meta-heuristic strategies are analyzed among various scheduling algorithms. Linear programming is a general modus operandi to tackle such an optimization problem. Bossche et.al therefore analyzed and proposed a binary integer program formulation of the scheduling problem and evaluate the computational costs of this technique with respect to the problem's key parameters. He found out that this approach results in a tractable solution for scheduling applications in the public cloud Van den Bossche (2010), but that the same method becomes much less feasible in a hybrid cloud setting due to very high solve time variances.

Yair Amir developed an Ad-Hoc method that helps in the job assignment and reassignment. Luiz F. Bittencourt, developed a dynamic approach that is applied to the Bittencourt, L.F., (2008) Path Clustering Heuristic, and introduces the concept of rounds, which take turns sending tasks to execution and evaluating the performance of the resources. In Topcuoglu et al. (2002) a heuristic Dynamic Critical Path (DCP) based workflow scheduling algorithm that determines efficient mapping of tasks and priority is assigned by calculating the critical path in the workflow task graph at every step.

Yu-Kwong Kwok (2004) has made a pair-wise comparison among seven scheduling algorithm under various conditions. But the drawback of this algorithm is that it has a set of several procedures that takes too much time to compile.

A hybrid heuristic scheduling algorithm was implemented on heterogeneous system that comprised of three phases given by Sakellariou (2004). The key idea of the hybrid heuristic is to use a standard list scheduling approach to rank the nodes of the DAG and then use this ranking to assign tasks to groups of tasks that can be subsequently scheduled independently. Rahman, M., (2007).: Haluk Topcuoglu has provided two performance-effective and low complexity task scheduling algorithms, namely HEFT and CPOP algorithms for heterogeneous system.

Edwin.S.H.Hou has developed a genetic algorithm for multiprocessor scheduling Hou, (1994). The algorithm is based on the precedence relations between the tasks in the task graph. He has compared the genetic algorithm with the list scheduling and optimal schedule using random task graphs and a robot inverse dynamics computational task graphs are presented. But this existing algorithm does not provide an optimal solution to the scheme.

Although cloud computing is playing a major role, many problems exist currently. One among them is, which resource to select based on cost and execution time. So the proposed TCHC algorithm is designed in such a way that it provides an optimized scheduling to the public cloud.

3. Preliminary Information for TCHC Algorithm

A private cloud contains multiple heterogeneous resources

$H_{ri} = \{Hr_1, Hr_2, Hr_3, \ldots .H_{rn}\}$,

Their computing capabilities are H_{r1}, C_iH_{r2}, $C_i\, H_{r3}, \ldots .C_iH_{rn}\, \varepsilon H_{ri,}$,

where i=1, 2, 3…n, ‘C'_i is the computing capability. The heterogeneous resource has set of associated links $L_{ri\ldots n} = \{l_{i\ldots n,r1}, l_{i\ldots n,r2}, l_{i\ldots n,r3}, \ldots l\ {}_{i}\ldots n,\mathrm{rm}\}$. where $l \leq m \leq n$. each link is associated with respective workflow.

$$w_{\,ri\ldots n} = \{l_1 \varepsilon w_1, l_2 \varepsilon w_2, l_3 \varepsilon w_3, \ldots l_n \varepsilon w_n\}.$$

Initially the link between any resources is ∞ (i.e) $l_{i,i} = \infty$, the on demand resources from the public cloud is acquired on pay per basis

$v = v_{1,}\, v_{3,}\, v_{3,} \ldots v_{n,}$

Similarly their computing capabilities of public cloud are $C_jv_{1,}\, C_jv_{2,}\, C_jv_{3,} \ldots .C_jv_{r,\varepsilon}V^{+}$, where j=1, 2, 3…n. The links associated with each public resource are $L_{vj\text{-}n} = \{l_{j\text{-}n,v1}, l_{j\text{-}n,v2}, l_{j\text{-}n,v3}, - l_{j\text{-}n,\mathrm{vd}}\}$ where $l \leq d \leq r$, each link is associated with their respective workflow.

$w_{vj\text{-}n} = \{l_1 \varepsilon w_{1,}\, l_2 \varepsilon w_{2,}\, l_2 \varepsilon w_{2,} \ldots l_n \varepsilon w_n\}$.

In-order to improve the scalability and capability of computing we combine both the private and resources leased from the public cloud.

$$H_c = \rho_{wf} \left[\prod\nolimits_1^n H_{ri} + \prod\nolimits_1^n V_{ri} \right]$$

As evaluated above the algorithm can easily allocate the resource to the workflows.

4. TCHC Data Flow

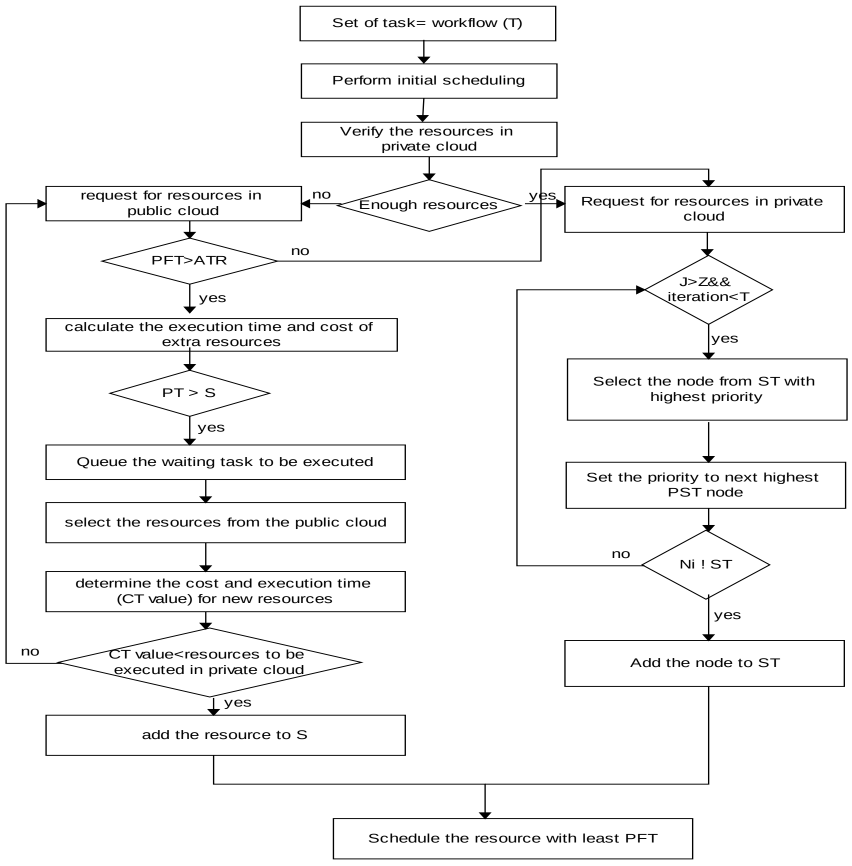
The two main steps of the algorithm are the selection of tasks to reschedule and the selection of resources from the public cloud to create the hybrid cloud. While the former decides which tasks can have their execution time reduced by using more powerful resources from the public cloud, the latter determines the execution time and costs involved in the new schedule.

Once the task is selected, the following are the process involved as shown in the above data flow diagram [figure 5.1], initial scheduling, which involves verification of available resources in the private cloud. If enough resources are available to do the task, the scheduler will schedule the task inside the private cloud.

Now the algorithm will check whether the private resource pool (J) is less than deadline (Z) the loop continues till all the tasks (T) are completed. Inside the loop, first a node is selected from a set of task with the highest priority and its PST and PFT are calculated along with dependency ratio. Secondly the priority is set to the next highest Predetermined Start Time (PST). Finally all the nodes are added and the resources are allocated to each set of task.

But if the resources are not enough, rescheduling is done in which it requests the public resource pool (S). The algorithm will now check whether the Predetermined Finish time (PST) is greater than the Application Time Remaining (ATR), if yes then calculate the execution time and cost for the extra resources, based on the Path clustering Heuristic (PCH) algorithm, else goes to the private resource pool itself. Next if the Pending Task (PT) is more than the available Public Resource Pool (S), then queue the tasks for execution. Each one of the tasks is selected and their cost and execution time are premeditated for the new resource. While the CT value is less than the resource in the private pool, it is continued, else the algorithm is made to request again from the public pool. Finally the algorithm schedules the resource with the least Predetermined Finish Time (PFT) and the task is continued.

Figure 1: Flow Diagram for TCHC


Set of task= workflow (T)
Perform initial scheduling
Verify the resources in private cloud
Enough resources
no
yes
request for resources in public cloud
Request for resources in private cloud
PFT>ATR
no
yes
calculate the execution time and cost of extra resources
PT > S
yes
Queue the waiting task to be executed
select the resources from the public cloud
determine the cost and execution time (CT value) for new resources
CT value<resources to be executed in private cloud
no
yes
add the resource to S
J>Z&& iteration<T
yes
Select the node from ST with highest priority
Set the priority to next highest PST node
Ni ! ST
no
yes
Add the node to ST
Schedule the resource with least PFT

5. TCHC Algorithm Description

The number of multiple workflows have to be scheduled properly, so that the resources are effectively utilized without consuming a higher price, when it comes to the public cloud resource. All the workflows have to be scheduled initially based on the following criteria

$$SC_{i,j} = \sum_{\forall ni \in s \rightarrow \min BW} Pwf \frac{Ir}{Pn} \qquad \text{(model 1)}$$

Where $SC_{i,j}$ is the final scheduled cost for nodes I to j.

Pwf is the Parallel workflow for the Instruction (Ir) to the processed node (Pn)

The model 1 determines the task based on the criteria that has the minimum bandwidth for the instruction to be executed for ‘n’ nodes in the multiple workflows

$$CC_{n\rightarrow i,j} = \frac{dataPn \rightarrow i,j}{lr \rightarrow P} \qquad \text{(model 2)}$$

where $\mathbf{CC_{n\rightarrow i,j}}$ is the communication cost, Pn is the parallel node and *l* is the link between resource (r) to resource provider (P).

The model 2, determines the communication cost for ‘n’ nodes, it determines the number of request to the pubic cloud. The bandwidth utilization (Bw) determines the link cost of the node $i\rightarrow j$. As long as the link cost is low, the scheduling task execution result is within the stipulated cost.

$$P_i = \begin{cases} SC_{i,j,} & , P(ni) = 0 \\ SC_{i,j} + \max\limits_{\forall n_{f \in (n1)}} [(CC_{n\rightarrow i,j} + P_{j+1}]) & , otherwise \end{cases} \qquad \text{(model 3)}$$

The task that has to be scheduled in the public cloud is the task that cannot be executed in the private cloud, though there is a serious expansion in the resource establishment. The task is determined from the pool of parallel workflow by using the priority to the task.

The priority is set to the node ‘n’, at the scheduling instance. Based on the priority that is set to the node ‘i’, we can determine the Predetermined Start Time (PST) and Predetermined Finish Time (PFT).

5.1. TCHC Optimization Algorithm

The scheduler’s intention is clear-cut: minimize the makespan without creating too much communication overhead between multiple workflows. We developed an approach to dynamically schedule multiple workflows, verifying the dependency range and finally trying to evaluate a minimized execution time and cost.

The proposed algorithm makes an initial schedule that is based on the equation model 1, model 2 and model 3. This initial schedule considers only the private resources and check if they meet the desired deadline. If the deadline is not met, the algorithm starts the process of deciding which resources it will request to the public cloud. This decision is based on performance, cost, and the number of tasks to be scheduled in the public cloud.

The TCHC algorithm is based on the following general steps:

1. Initial schedule: schedule the workflow in the private cloud R;
2. While the makespan is larger than the deadline:

- Select tasks to reschedule.
- Verify the dependency range among the workflows.
- Select resources from the public clouds to compose the hybrid cloud.
- Reschedule the selected tasks in H.

The two main steps of the algorithm are the selection of tasks to reschedule and the selection of re-sources from the public cloud to compose the hybrid cloud. While the former decides which tasks can have their execution time reduced by using more powerful resources from the public cloud, the latter determines the performance and costs involved in the new schedule.

Workflow **T**

Dependency **De=0 to 5**
Deadline **Z**
Resource **H**
Predetermined Start Time **PST**
Predetermined Finish Time **PFT**
Public resource pool **S**
Private Resource Pool **J**
Priority **Pi**
Pending task **PT**
Node **Ni**
Application Time Remaining **ATR**
Node set **ST**
Cost & Time value **CT**

```
1.      TW=Workflows==set of tasks TS==single task T
2.      Perform initial schedule
3.      Dependency De=0-5
4.      For each W in TW
        For each T in TS do
            If T < De Do
5.              If H Є J then
6.              Schedule T in J
7.          While (J > Z && iteration =T) do
8.              Select node from ST with ↑Pi
9.                  If T Э ST then
10                  Add T to ST
11.                 Iteration=iteration+1
12.         End while
13.       else select next task from TS
14.       else select next workflow from WT
15.               Schedule the H with ↓ PFT
16.             Else
17.             While Request for H in S
18.                 If PFT > ATR then
19.                   Queue WT to execute
20.             For each W in TW
                For each T in TS do
                  If T < De Do
21.                   Select H Є S then
22.                   Calculate CT for new H
23.                     If CT < ( H Є J ) then
24.                             Add H to S
25.             else select next task from TS
26.         else select next workflow from WT
27.                 Schedule H with ↓ PFT
28.             Else
29                  Continue Request in J
30.               End while
```

Workflow consists of set of tasks; In this paper the algorithm considers a set of workflows. Basically a single workflow consists of a set of tasks which is dependent on one another. The main task is to reduce the dependency. We have assigned a range for dependency for instance: dependency De value is between 0 - 5.

The First line of the algorithm initializes the set of workflows to a variable TW. The third line performs initial schedule which considers only the Private resource pool and schedule these workflow in the Private resource pool itself based on some attributes like communication cost, Priority and time. Initial scheduling involves evaluating the cost and time value for each workflow based on some parameters like communication cost, Priority; cost for each resource allocation is done.

Basically every workflow has a set of tasks which is less or more dependent on each other. The main task is to verify that each task must have the defined dependency range only. The fourth line checks the range and once if the dependency value is less than the range, the allocation or request to the resource is done else it is not. Next the algorithm checks whether the available resources are enough or not. If it is sufficient enough to finish the job, the workflow is requested in the private cloud itself else it is requested in the public cloud.

Once the workflows are scheduled in the private cloud, until the deadline is met the task is running inside the private cloud. The iteration is repeated until the deadline Z is met, where the algorithm continues by selecting a node Ni from the node set ST with the highest priority. Once the iteration is over it checks in the public resource pool S. The line18 in the algorithm verifies whether the predetermined finish time PFT is greater than ATR, then queue the tasks to execute. In line 22 calculate the cost and time value for the new resource to allocate. Finally allocate the resource with the lowest PFT.

Finally schedule the resource with the lowest PST. Once the resources are not enough in the private cloud, the request is done to the public cloud. If the evaluated PFT is greater than the ATR, then the waiting tasks are put in the queue to execute. Next select the resource from the public cloud and evaluate the new CT value for new resource allocation. Once the value of CT is less than the available resource in the private cloud, then only the public cloud is requested. Since the CT value, is considered to be less than the old CT value the resource is added to the set S. Now schedule the resource with the lowest PFT, suppose the CT value is larger than verify inside the private cloud itself. Finally allocate the resource with the lowest PFT.

6. Simulation Results

The figure 1 shows the comparison between HCSA and TCHC algorithm in terms of scheduled cost for node $i \rightarrow j$. For ‘n’ nodes in a workflow, the HCSA algorithm has an edge when compared to TCHC algorithm. The concave matrix played a vital role in determining the efficiency of the TCHC algorithm. By providing minimum concave to each of the nodes associated with the workflow and least dependency ratio, our algorithm out performs the HCSA algorithm.

Initially, when compared with the HCSA and TCHC, both execute at a steady rate, these results are considered when they are compared at the scheduling instance of a particular processing time for a particular node. This is done to avoid dependency ratio of the workflow associated with the task to be completed. The efficiency of TCHC algorithm has to be determined in order to showoff, how it works when it comes to parallel nodes. The figure 2 shows the throughput of TCHC against the parallel number of nodes. The graph plots shows that, the TCHC when compared to HCSA has a same throughput for an instant.

Figure 2: Cost Optimization

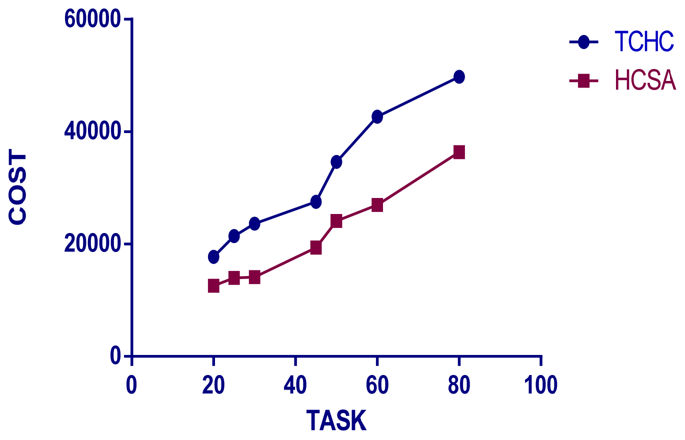


Figure 3: Throughput

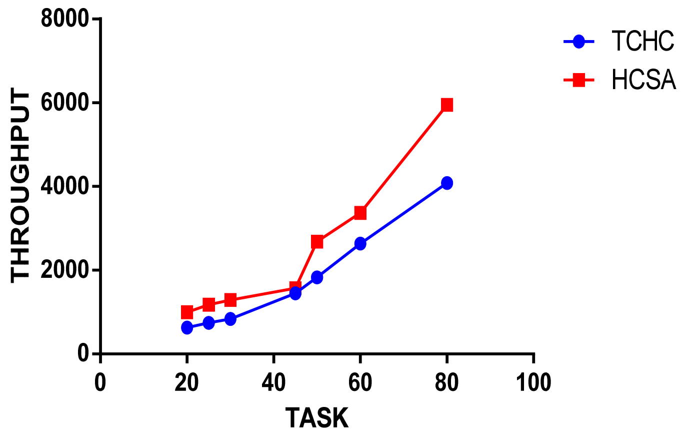
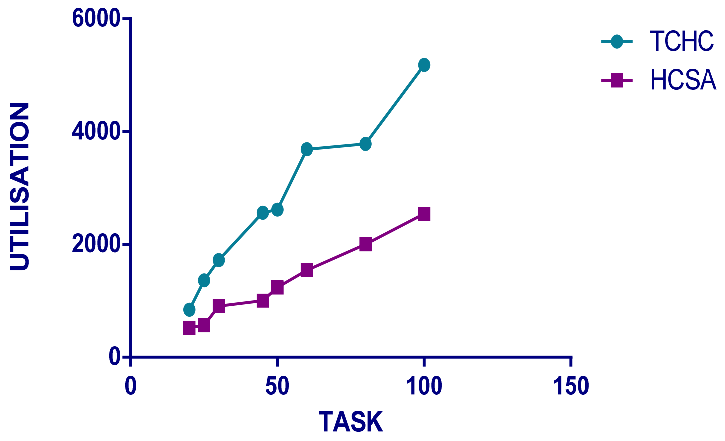
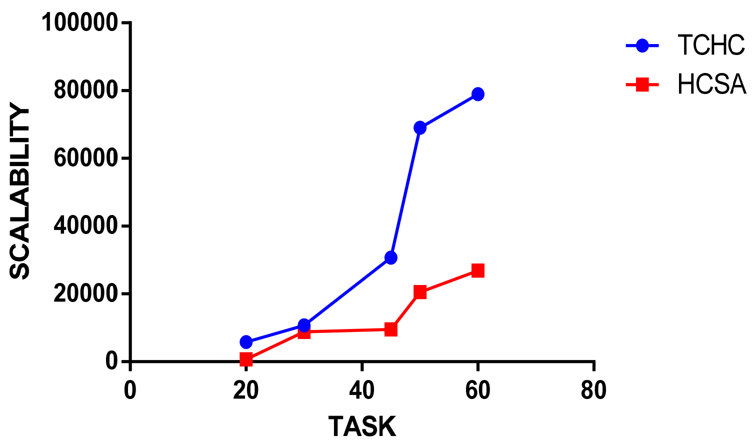


Figure 4: Utility usage of resources



At some point in time, the throughput of TCHC starts to degrade a bit when there are more parallel task request. This is where TCHC comes into play, it avoids the dependency constraints and also more importantly the minimum concave metric constraints that played a vital role in determining the throughput of TCHC algorithm.

Figure 5: Scalability factor



The figure 4 shows the utility curve of the HCSA and TCHC algorithms. The HCSA algorithm has a very high computation overhead, when it is loaded with multiple workflows and impossible for it to meet the desired deadline. However, for the purpose of comparison, TCHC has an optimal utilization of resources requested from the public resources and also less overhead in terms of multiple requests made to the public cloud. The resource price of the public cloud may change at any instant, so requesting a released resource may increase the bandwidth. The resources are buffered in the local resource pool for the demanded time will outperform the HCSA algorithm.

When it comes to scaling the extra resource to the current scheduling instant, TCHC outperforms the HCSA algorithm. The figure 5 shows that HCSA scalability factor grows linearly, whereas TCHC scalability factor grows exponentially. The aim TCHC is to determine the right cloud vendor to have high scalability ratio. The suitable interface provided by TCHC enables a high scalability in terms of new resources.

7. Conclusion

Hybrid clouds are being used to execute different kinds of applications. Among them, workflows have an important role in processes of many fundamental science fields, such as Physics, Chemistry, Biology, and Computer Science. To speedup science advancements, it is important to provide efficient resource utilization as well as to make application executions affordable. This paper has presented TCHC: The Time and Cost optimized scheduling algorithm for Hybrid Cloud. TCHC is an algorithm to speed up the execution of multiple workflows obeying a desired execution time, but also reducing costs when compared to the HCSA approach.

The extensive evaluation carried out in this work provides sufficient data to support the conclusion that the TCHC algorithm can provide efficient scheduling in a hybrid cloud scenario with a low dependency ratio between the tasks. Its multicore awareness, along with the cost knowledge, can provide makespans as low as the user needs. As a consequence, the user is able to control costs by adjusting the desired workflow execution time Z, if the dependency ratio among the workflow is less. In general, the proposed algorithm has the ability of reducing the execution costs and time in the public cloud with the increase of the workflow desired execution time. Besides that, in some cases where the desired execution time is too low, TCHC finds better schedules than the HCSA approach by taking advantage of multicore resources, reducing the number of violations of Z.

References

[1] Amir, Y., Awerbuch, B., Barak, A., Borgstrom, R.S.,Keren,(2000).: An opportunity cost approach for job assignment in a scalable computing cluster. IEEE Transaction on Parallel and Distributed Systems 11(7), 760–768

[2] Bittencourt, L.F., Madeira, (2008) E.R.M.: A performance oriented adaptive scheduler for dependent tasks on grids. Concurrency and Computation: Practice and Experience 20(9), 1029–1049.

[3] Bittencourt, L.F., Madeira, (2010) E.R.M.: Towards the scheduling of multiple workflows on computational grids. Journal of Grid Computing 8,419–441

[4] M. Cardosa, M.R. Korupolu, and A. Singh (2009), "Shares and Utilities Based Power Consolidation in Virtualized Server Environments," Proc. IFIP/IEEE 11th Int'l Conf. Symp. Integrated Network Management (IM '09).

[5] L. Grit, D. Irwin, A. Yumerefendi, and J. Chase (2006), "Virtual Machine Hosting for Networked Clusters: Building the Foundations for Autonomic Orchestration," Proc. IEEE Int'l Workshop Virtualization Technology in Distributed Computing.

[6] Hou, E.S.H., Ansari, N., Ren, H(1994): A genetic algorithm for multiprocessor scheduling. IEEE Transactions on Parallel and Distributed Systems 5(2), 113–120.

[7] F. Hermenier, X. Lorca, and J.-M. Menaud (2009)., "Entropy: A Consolidation Manager for Clusters," Proc. ACM SIGPLAN/ SIGOPS Int'l Conf. Virtual Execution Environments (VEE '09).

[8] Kwok, Y.K., Ahmad, I (1996).: Dynamic critical-path scheduling: An effective technique for allocating task graphs to multiprocessors. IEEE Transactions on Parallel and Distributed Systems 7(5), 506–521.

[9] "The NIST definition of cloud computing 15",National Institute of Standards and Technology (NIST), Tech. Rep., July 2009. [Online]. Available: http://csrc.nist.gov/groups/SNS/cloud-computing/cloud-def-v15.doc

[10] Rahman, M., Venugopal, S., Buyya, R (2007).: A dynamic critical path algorithm for scheduling scientific workflow applications on global grids. In: Third IEEE International Conference on e-Science and Grid Computing, pp. 35–42. IEEE Computer Society, Washington, DC, USA.

[11] Sakellariou, R., Zhao, H (2004): A Hybrid Heuristic for DAG Scheduling on Heterogeneous Systems. In: 13th Heterogeneous Computing Workshop, pp. 111–123. IEEE Computer Society.

[12] Bittencourt, L.F., Senna, C.R., Madeira, and E.R.M (2010): Scheduling service workflows for cost optimization in hybrid clouds. In: 6th International Conference on Network and Service Management (CNSM 2010). Toronto, Canada.

[13] Topcuoglu, H., Hariri, S., Wu, M.Y (2002): Performance effective and low-complexity task scheduling for heterogeneous computing. IEEE Transactions on Parallel and Distributed Systems 13(3), 260–274.

[14] H.N. Van, F.D. Tran, and J.-M. Menaud, (2009), "SLA-Aware Virtual Resource Management for Cloud Infrastructures," Proc. IEEE Ninth Int'l Conf. Computer and Information Technology.

[15] Van den Bossche, R., Vanmechelen, K., Broeckhove, J235 (2010).: Cost-Optimal Scheduling in Hybrid IaaS Clouds for Deadline Constrained Workloads. In: 3rd International Conference on Cloud Computing, pp 228-235.