Scheduling in Hybrid Clouds

Luiz F. Bittencourt, Edmundo R. M. Madeira, and Nelson L. S. da Fonseca, University of Campinas

ABSTRACT

Schedulers for cloud computing determine on which processing resource jobs of a workflow should be allocated. In hybrid clouds, jobs can be allocated on either a private cloud or a public cloud on a pay per use basis. The capacity of the communication channels connecting these two types of resources impacts the makespan and the cost of workflow execution. This article introduces the scheduling problem in hybrid clouds presenting the main characteristics to be considered when scheduling workflows, as well as a brief survey of some of the scheduling algorithms used in these systems. To assess the influence of communication channels on job allocation, we compare and evaluate the impact of the available bandwidth on the performance of some of the scheduling algorithms.

INTRODUCTION

Cloud computing has attracted an increasing number of users because it offers computational capabilities as services on a pay-per-use basis. Studies conducted by Gartner[1] estimate a potential US$150 billion market for cloud computing by 2013. Such a huge market is the consequence of a business model that offers high performance and low costs. Indeed, a survey[2] of 3645 users of cloud computing services conducted by the Computer Sciences Corporation in eight countries between October 2011 and November 2011 reported that improved data center efficiency and lower operational costs are the main reasons for the adoption of cloud computing solutions.

Cloud providers offer computing and storage resources, and platforms for software development and execution, as well as software interfaces accessible throughout the network. Three models of cloud services are commonly available: infrastructure as a service (IaaS), platform as a service (PaaS), or software as a service (SaaS) [1]. In SaaS, the clients use applications but cannot control the host environment. Google Apps and Salesforce.com are examples of this model. In PaaS, the platform is typically an application framework, and clients use a hosting environment for their applications. Examples of PaaS are the Google App Engine and Amazon Web Services. In IaaS, the clients use computing resources such as processing power and storage, and they can also control the environment and the deployment of applications. Amazon Elastic Compute Cloud (EC2), Globus Nimbus Toolkit, and Eucalyptus are good examples of this service model. In summary, clients can use/run applications from a SaaS cloud; both develop and run their applications on a development platform provided by a PaaS cloud, or extend their computational capacity by leasing computing resources from an IaaS cloud.

Moreover, clients can execute most applications using their own computing infrastructure (private cloud), and yet lease service from a cloud provider (public cloud) on demand. It was reported that 48 percent of U.S. government agencies have moved at least one workflow to a cloud provider following the federal cloud computing strategy published in February 2011. The operation of such hybrid cloud involves two fundamental questions:

- What resources should be leased?
- Which tasks should be executed on the leased resources?

These answers are provided by a scheduler, a fundamental component of distributed computing systems including clouds and grids [2].

These questions are answered considering the capacities of the communication links connecting the available resources. Slow communication channels increase delays, thus increasing the execution time (*makespan*) of applications, with bounds typically negotiated in service level agreements. Understanding the impact of network delays and costs on scheduling decisions is thus fundamental for cloud service provisioning.

In line with that, this article provides a brief survey of scheduling algorithms for hybrid clouds and the impact of communication networks on scheduling decisions. First, the problem of scheduling tasks and services in clouds is explained, which is then followed by a comparison of scheduling algorithms for hybrid clouds. At last, the impact of communication links on schedules is assessed.

SCHEDULING IN CLOUDS

Applications and services can be decomposed into sets of smaller components, called jobs. For example, an application that processes a large image can decompose this image into smaller ones for parallel processing by distinct jobs. The logical sequence of the jobs of an application is called a *workflow*, which is commonly represented by a directed acyclic graph (DAG). The nodes

[1] *http://www.gartner.com/id=914826.*

[2] *http://assets1.csc.com/newsroom/downloads/CSC_Cloud_Usage_Index_Report.pdf.*

of a DAG represent the jobs of a workflow, while arcs represent their data dependencies. A job can be executed only after the data on which it depends has been produced and sent to the resource where it will be executed. Such applications can be found in a variety of fields, such as physics (astronomy, thermodynamics), bioinformatics (DNA sequencing, proteomics), chemistry (protein dynamics), and computer science (computer vision, image processing).

The workflows of two real applications are illustrated: Montage [3] (Fig. 1a) and the Laser Interferometer Gravitational Wave Observatory (LIGO) [4] (Fig. 1b). Montage consists of an image application that creates mosaics of the sky in astronomy research. The size of the workflow depends on the squared degree size of the part of the sky to be generated, and it can produce an output of an 86 Tbyte data set involving 17 hierarchical workflows, each with 900 subworkflows.[3] LIGO is a project used to detect gravitational waves through a network of gravitational-wave detectors, and its workflow often requires on the order of a terabyte of data to produce meaningful results.[3]

The computational demands of applications such as Montage and LIGO can easily overwhelm the available computational power of private clouds. Moreover, their execution time can be prohibitive. The cloud computing paradigm is quite effective for dealing with such problems by providing virtually unbounded on-demand resources.

Figure 2 illustrates a hybrid IaaS cloud composed of the resources of the private cloud as well as those of one or more public IaaS clouds. A hybrid cloud scheduler must decide which resources should be leased from the public clouds to guarantee the execution of the workflow within the specified maximum execution time (deadline). After the submission of workflow by a user, a broker runs the scheduling algorithm to start the decision making process.

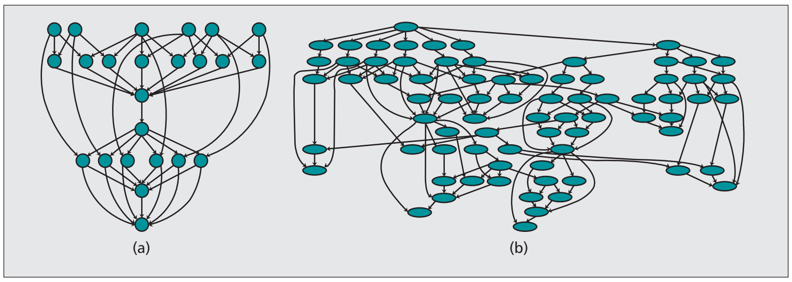

Figure 1. *Examples of workflow applications: a) Montage workflow; b) LIGO workflow.*

Besides deciding which resources will be used, the scheduler also determines which part of the workflow will run in each cloud provider.

One challenging issue in hybrid clouds is how interfaces can be provided to interact automatically with different existing public clouds so that the broker can gather information about resources, and the workflow can be executed and monitored in a variety of public cloud infrastructures. Some projects, such as the JClouds (www.jclouds.org), try to solve this problem by providing portable abstractions to several existing cloud providers. Another challenge involves the consideration of security requirements of the applications, which can reduce the pool of potential hosts for scheduling jobs.

The scheduling problem involved is known to be NP-complete in general, including the scheduling of workflows in heterogeneous computer systems discussed in this article. Scheduling algorithms often utilize heuristics and optimization techniques to try to obtain a near optimal schedule.

The input of the scheduling algorithm must include the DAG that represents the workflow of jobs and their dependencies, as well as information about the target system, including the

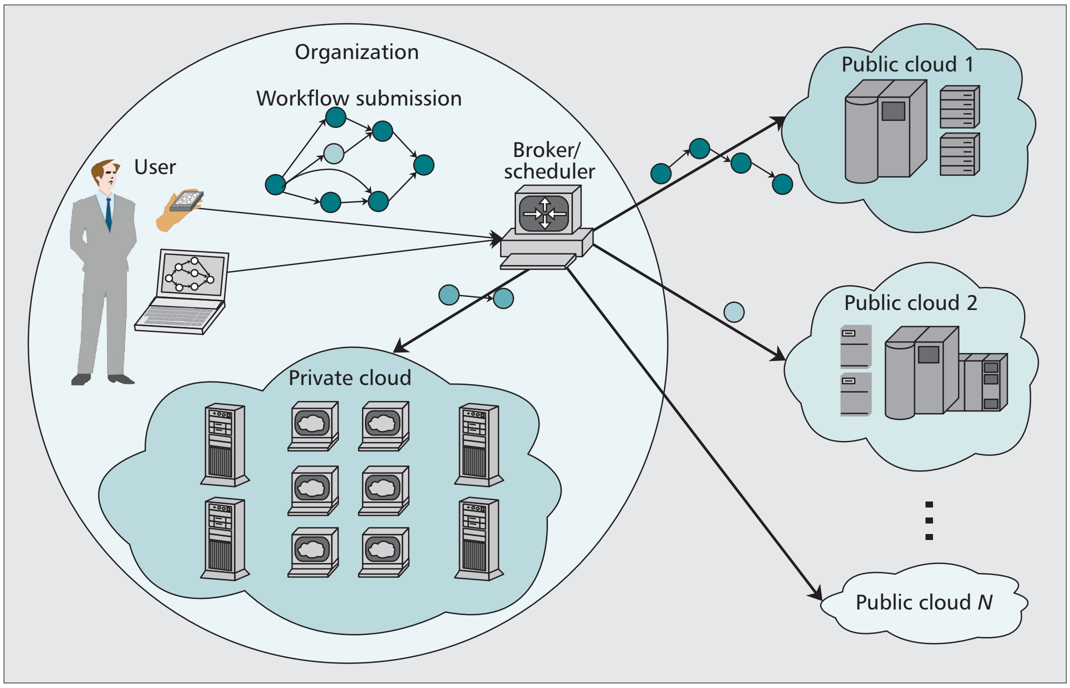

Figure 2. *Hybrid cloud infrastructure and workflow submission.*

[3] *Source: http://pegasus.isi.edu/.*

Cloud computing evolved from grid computing, service oriented computing, and virtualization paradigms. This means that scheduling algorithms developed for these type of systems can also be used in clouds.

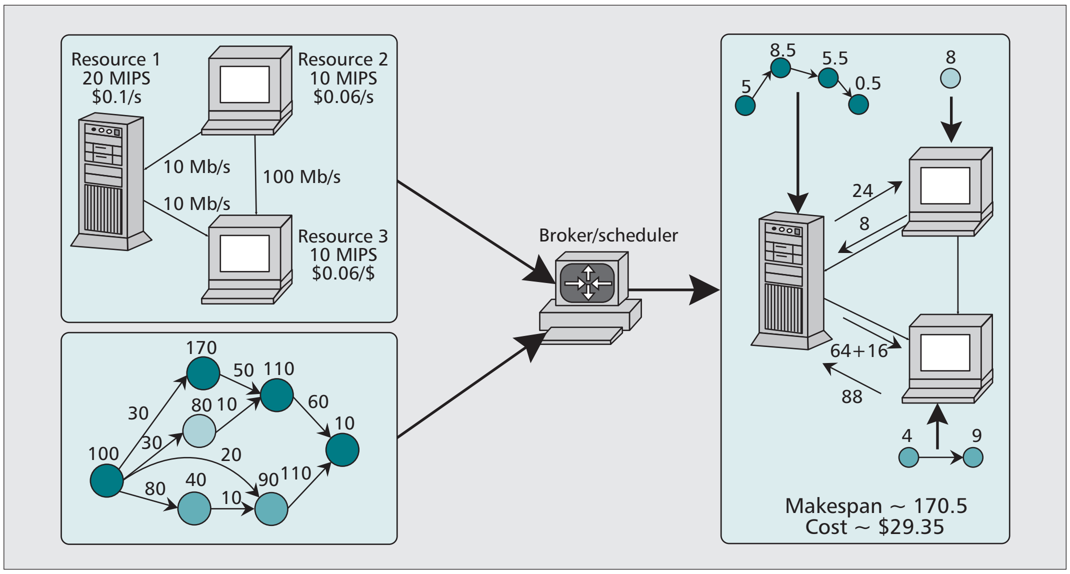

Figure 3. *Scheduler inputs, with information about the DAG and the target system, and a possible resulting schedule.*

processing capacity of each resource and the available capacities of the network links. This information is obtained from a resource information repository in the private cloud. Moreover, scheduling algorithms for clouds are usually cost-aware, so that the information about the cost per time unit of usage of each resource must be available. Providing with this information, the scheduling algorithm is capable of estimating the workflow makespan and its execution costs.

Figure 3 illustrates the inputs necessary for a scheduler and gives an example of a schedule. The DAG contains information on the computational requirements of its jobs as well as information on the amount of bytes to be transmitted to resolve each data dependency. In the example, we use millions of instructions (MI) for specifying job computational costs, MIs per second for resource processing capacities, megabytes for data dependencies, and megabits per second for link bandwidths. The scheduler combines all of this information to compute how long each job takes to run on each resource, how long each data transmission would take according to the resource assigned to each job, and how much a given job assignment would cost. The scheduling algorithm runs when a workflow is submitted for execution, and the necessary resources are allocated on demand to run the workflow.

The right side of Fig. 3 illustrates a hypothetical schedule for this example. The critical path of the DAG was scheduled on resource 1 ($R1$), which is the fastest available. Data transmission takes 24 s between $R1$ and $R2$ in order to fulfill data requirements for the job scheduled on $R2$. This job returns the results to $R1$ so that it can be utilized by the third job on the critical path, which demands a transmission of 8 s. Longer data transmissions occur between $R1$ and $R3$, since jobs running on those resources have stronger data dependencies. By considering all the computations performed and all the transmission delays, the resulting makespan is 170.5 s and cost is $29.35. The consideration of communication delays is important for the minimization of costs since a job that receives data from its predecessors needs to be active in that resource, thus consuming processing time and money.

Schedulers for Clouds

Cloud computing evolved from grid computing, service oriented computing, and virtualization paradigms. This means that scheduling algorithms developed for these types of systems can also be used in clouds. Scheduling algorithms can be distinguished by their main characteristics, such as:

- Target system: The system for which the scheduling algorithm was developed, which can be a heterogeneous system, a grid, or a cloud computing system.
- Optimization criterion: Makespan and cost are the main metrics specified by cloud user and considered by schedulers in the decision making process.
- Multicore awareness: Computer systems can have multiple cores, which should be considered by scheduling algorithms in resource selection.
- On-demand resources: Resources can be leased either on-demand or for long terms. The on-demand leasing of resources is treated by the scheduling algorithm as a "single expense" during the execution of the workflow.
- Reserved resources: The algorithm should consider the use of a resource reserved for a long term.
- Levels in a service level agreement (SLA): The scheduling algorithm should consider that SLAs can be organized hierarchically. SLAs with a single level allow clients and providers to interact directly to negotiate resource capacities and prices. When there are multiple levels, the scheduling algo-

Algorithm	Target system	Optimization criteria	Multi-core aware	On-demand resources	Reserved resources	SLA levels
HEFT [7]	Heterogeneous	minimize makespan	No	No	No	No
MDP [8]	Utility Grid	minimize cost within deadline	No	Yes	No	Single-level
PCP [9]	Utility Grid	minimize cost within deadline	No	Yes	No	Single-level
Pandey [10]	Cloud	minimize cost	No	Yes	No	Single-level
Wu [11]	Cloud	minimize cost	No	Yes	No	Single-level
HCOC [5]	Cloud	minimize cost within deadline	Yes	Yes	No	Single-level
Genez [6]	Cloud	minimize cost within deadline	Yes	Yes	Yes	Two-level

Table 1. Table 1. *Scheduling algorithms characteristics.*

rithm can run in an intermediate facility between the IaaS cloud provider and the final client. By doing so, costs can be decreased.

Table 1 lists various workflow scheduling algorithms and compares their characteristics and applicability for cloud scheduling. Although not all scheduling algorithms used in clouds were conceived for these systems, recently some scheduling algorithms specially designed to hybrid clouds have been proposed [5, 6]. Next, we briefly describe some scheduling algorithms commonly used.

The Heterogeneous Earliest Finish Time (HEFT) [7] scheduling algorithm was designed for heterogeneous computing systems. Since it was developed before the advent of cloud computing and utility grids, it does not consider monetary costs. Its objective is to minimize the workflow makespan.

The deadline-driven cost-minimization algorithm [8] or Deadline-Markov Decision Process (MDP) breaks the DAG into partitions, assigning a maximum finishing time for each partition according to the deadline set by the user. Based on this time, each partition is scheduled for that resource, which will result in the lowest cost and earliest estimated finishing time. This algorithm works with on-demand resource reservation.

Abrishami *et al.* [9] presented the Partial Critical Paths (PCP) algorithm, which schedules the workflow in a backward fashion. Constraints are added to the scheduling process when such scheduling of jobs in a partial critical path fails so that the algorithm will be restarted. This algorithm presents the same characteristics as does MDP, although it involves greater time complexity, since a relatively large number of reschedulings can be demanded during the execution of the algorithm.

The self-adaptive global search optimization technique called particle swarm optimization (PSO) is utilized to schedule workflows in the algorithm proposed in [10]; it was developed to work in clouds with single-level SLAs and on-demand resource leasing. It considers neither multicore resources nor workflow deadlines, but focuses solely on monetary cost minimization.

The Hybrid Cloud Optimized Cost (HCOC) algorithm [5] schedules workflows in hybrid clouds by first attempting costless local scheduling using HEFT. If the local scheduling cannot meet the deadline, the algorithm selects jobs for scheduling in resources from the public cloud. When selecting resources from the public cloud, the HCOC algorithm considers the relation between the number of parallel jobs being scheduled and the number of cores of each resource as well as deadlines, performance, and cost. As with the MDP algorithm, the objective is to minimize the financial cost, obeying the deadlines stipulated by the user in a single-level SLA contract.

In [6], the workflow scheduling problem was formulated as an integer linear program that considers the leasing of reserved and on-demand resources from multiple IaaS providers according to a two-level SLA. The scheduler can run in either a SaaS or PaaS cloud provider, and receive workflow execution requests with deadlines from its clients (first SLA level), but it can also lease resources from multiple IaaS providers (second SLA level).

A common characteristic of the above mentioned algorithms is that they do not take into account the fluctuation of the prices of resource allocation due to the varying demand of resources. In order to reduce costs, a scheduler could allocate or even reallocate jobs when prices are low.

Impact of Available Bandwidth on Scheduling

The available bandwidth in channels connecting processing resources of the hybrid cloud impacts the makespan and cost of a schedule. This section discusses this impact as well as the effectiveness of the HEFT [7], MDP [8], and HCOC [5] in scheduling workflows in hybrid clouds. HEFT is a well-known scheduling algorithm for heterogeneous computing systems and aims at makespan minimization. The MDP scheduling algorithm was designed for utility grids and is often used in the literature for comparisons with cost-based algorithms. Moreover, the HCOC scheduling algorithm is a recent development specially for hybrid clouds. By evaluating these algorithms, we can analyze the adequacy of

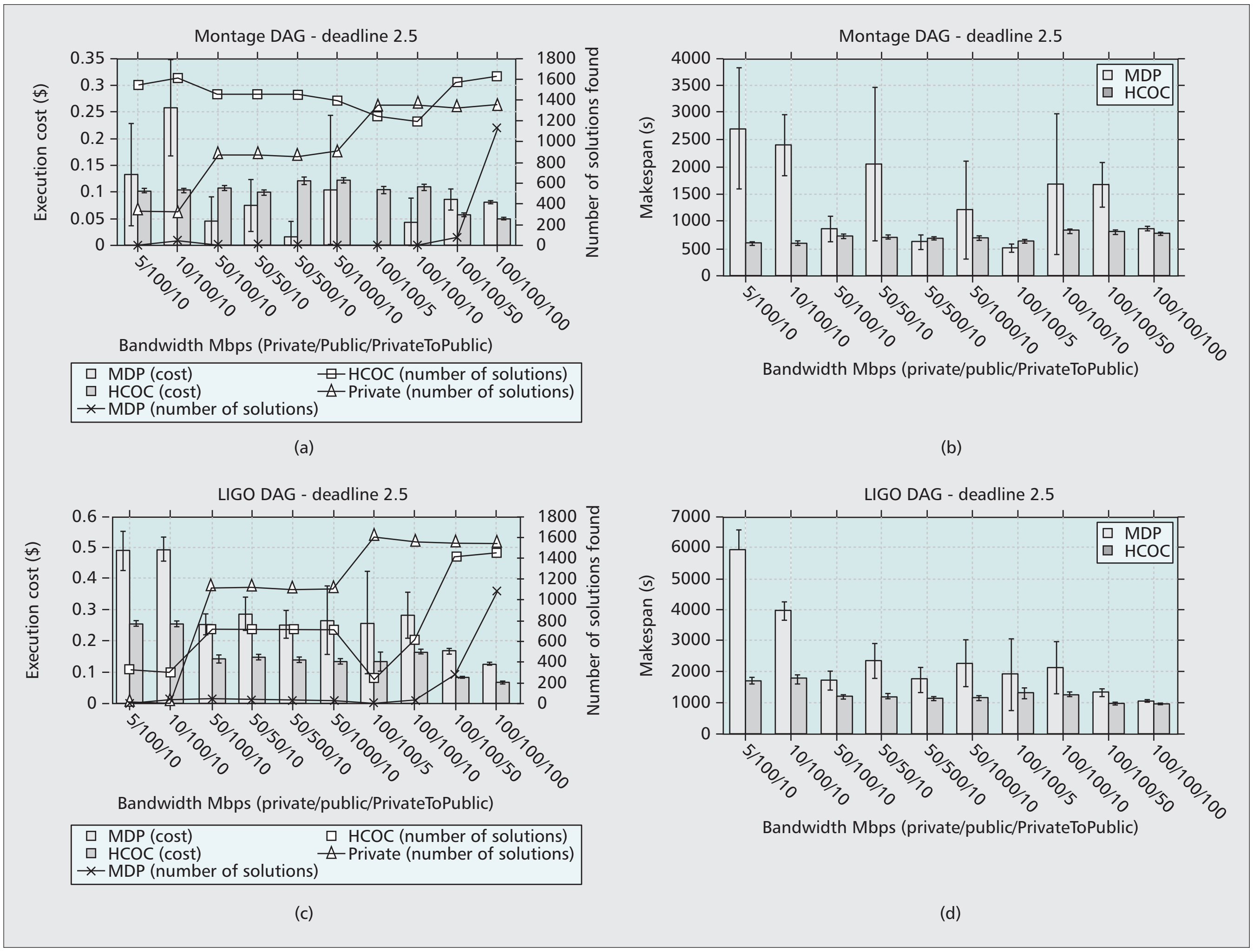

Figure 4. *Results for the Montage and LIGO DAGs: a) Montage — average cost; b) Montage — average makespan; c) LIGO — average cost; d) LIGO — average makespan.*

scheduling algorithms designed for clouds with those designed for other systems but used for scheduling in cloud systems.

In the evaluation, the capacity of intracloud channels as well as the intercloud channels were varied so that their impact on the scheduling efficacy could be evaluated.

Three thousand different Montage and LIGO DAGs were generated with computation demands varying in the interval [5×10^5, 4×10^6] MI, and communication demands varying randomly in the interval [60,500] Mbytes. The deadline for completion of the workflow was set to 2.5 times the duration taken to compute the critical path of the DAG. Such value was set since it was observed in previous experiments that it leads to the highest number of workflow completions by the three scheduling algorithms. If the predicted makespan is shorter than the deadline, the solution given by HEFT is adopted, avoiding cost increase due to the leasing of resources.

The number of processing resources in the private cloud was varied from 1 to 10 with the processing capacities for each randomly taken from the interval [10^4, 10^5] MI per second (MIPS). For the public cloud, there were four types of resources with computing unit capacities and leasing costs equivalent to the Amazon EC2 *small*, *large*, *extra large*, and *extra large high CPU* on-demand instance types. In the experiments undertaken, each computing unit was randomly drawn from the interval [10^4,7×10^4] MIPS, referring to the small instance type computing power. The four types of resources were: small (1 core of 1 computing unit, \$0.085 per hour); large (2 cores of 2 computing units each, \$0.34/h); extra large (4 cores of 2 computing units each, \$0.68/h); extra large high CPU (8 cores of 2.5 compute units each, \$0.68/h). Moreover, in the experiments, the topologies of private cloud and public cloud networks were fully connected graphs, and the private cloud was connected to each public cloud by an intercloud link.

Figure 4 shows the makespan, execution cost, and number of solutions found for the Montage and LIGO DAGs by the three scheduling algorithms evaluated. The bars show the average cost and average makespan with 95 percent confidence interval. The three curves represent the total number of successful schedules (i.e., the number of schedules with a makespan lower than the deadline) achieved by each algorithm. The increase in the number of solutions found by the HEFT algorithm as a function of the increase in available bandwidth in a private cloud reveals that the private bandwidth is of major importance for scheduling workflows in the private cloud. Moreover, an increase in bandwidth reduces the costs when the target system is a hybrid cloud.

The number of solutions found by the HCOC and MDP algorithms increases when the bandwidth between the private and public clouds increases, since new solutions for problem instances that cannot be satisfied in the private cloud can now be found using the resources of the public cloud. The small influence of public cloud bandwidth in the number of solutions found by the HCOC and MDP algorithms can be explained by the tendency of these algorithms to group dependent jobs to minimize communication costs.

The intercloud available bandwidth reduces the makespan. Moreover, when numerous solutions are found in hybrid cloud processing, the average cost of the schedule decreases with an increase in the bandwidth between the private and public clouds. Schedules that are unfeasible in private cloud processing can be implemented at a low cost when the intercloud bandwidth increases. The average makespan value tends to be close to the deadline, since the scheduling algorithms try to minimize cost as long as the makespan is kept lower than the deadline. Makespan closer to the deadline can use less expensive (and slower) resources from public clouds and more resources from the private cloud, which hardly cost anything compared to using public clouds. Results using other DAGs (Random, AIRSN spatial normalization, the example DAG in the MDP paper, and Chimera virtual data), not discussed here, reinforce this conclusion.

Comparison of the makespan produced by the MDP algorithm with that given by the HCOC algorithm shows that the former, designed for utility computing, underperforms the HCOC, which was specially designed for hybrid clouds.

Moreover, the cost demanded by HCOC for processing the LIGO workflow is lower than that demanded by MDP. The low number of solutions found by MDP when processing the Montage DAG does not allow a significant comparison of the costs for this DAG.

Conclusion

This article has discussed the problem of scheduling applications and services in hybrid clouds. A brief comparison of some of the scheduling algorithms used in hybrid grids has been provided, with the results suggesting a prominent importance of communication capacity when scheduling workflows in hybrid clouds, especially that of the communication channels between private and public clouds. Such channels are usually located in the Internet backbone, which increases the challenges involved in the development of communication-aware scheduling algorithms since the available bandwidth of these links fluctuates widely. Moreover, given the importance of the intercloud communication channels, the development of communication-aware or even communication-driven scheduling algorithms is of paramount importance to provide quality of service and competitive costs for hybrid clouds. Furthermore, an efficient algorithm developed for efficient use on utility grids (MDP) may not be as efficient for hybrid clouds as are those algorithms developed specially for such systems (HCOC). This superior performance of the HCOC scheduler may partially be due to its multicore awareness, which is clearly a characteristic requiring consideration in hybrid cloud computing.

Given the importance of the intercloud communication channels, the development of communication-aware or even communication-driven scheduling algorithms is of paramount importance to provide quality of service and competitive costs for hybrid clouds.

References

[1] M. Dikaiakos *et al.*, "Cloud Computing: Distributed Internet Computing for IT and Scientific Research," *IEEE Internet Computing*, vol. 13, no. 5, Sept.–Oct. 2009, pp. 10–13.

[2] D. M. Batista and N. L. S. da Fonseca, "A Survey of Self-Adaptive Grids," *IEEE Commun. Mag.*, vol. 48, no. 7, July 2010, pp. 94–100.

[3] Y. Gil, E. Deelman *et al.*, "Examining the Challenges of Scientific Workflows," *IEEE Computer*, vol. 40, no. 12, Dec. 2007, pp. 24–32.

[4] E. Deelman *et al.*, "Griphyn and Ligo, Building A Virtual Data Grid for Gravitational Wave Scientists," *11th IEEE Int'l. Symp. High Perf. Distrib. Computing*, 2002, pp. 225–34.

[5] L. F. Bittencourt and E. R. M. Madeira, "HCOC: A Cost Optimization Algorithm for Workflow Scheduling in Hybrid Clouds," *J. Internet Svcs. and Apps.*, vol. 2, no. 3, Dec 2011, pp. 207–27.

[6] T. A. L. Genez, L. F. Bittencourt, and E. R. M. Madeira, "Workflow Scheduling for SaaS/PaaS Cloud Providers Considering Two SLA Levels," *IEEE/IFIP NOMS*, Apr. 2012.

[7] H. Topcuoglu, S. Hariri, and M.-Y. Wu, "Performance-Effective and Low-Complexity Task Scheduling for Heterogeneous Computing," *IEEE Trans. Parallel and Distrib. Sys.*, vol. 13, no. 3, 2002, pp. 260–74.

[8] J. Yu, R. Buyya, and C. K. Tham, "Cost-based Scheduling of Scientific Workflow Applications on Utility Grids," *Int'l. Conf. e-Science and Grid Computing*, July 2005, pp. 140–47.

[9] S. Abrishami, M. Naghibzadeh, and D. Epema, "Cost-Driven Scheduling of Grid Workflows Using Partial Critical Paths," *11th IEEE/ACM GRID*, Oct. 2010, pp. 81–88.

[10] S. Pandey *et al.*, "A Particle Swarm Optimization-Based Heuristic for Scheduling Workflow Applications in Cloud Computing Environments," *24th IEEE AINA*, Apr. 2010, pp. 400–07.

[11] Z. Wu *et al.*, "A Revised Discrete Particle Swarm Optimization for Cloud Workflow Scheduling," *Int'l. Conf. Computational Intelligence and Security*, Dec. 2010, pp. 184–88.

Biographies

LUIZ F. BITTENCOURT (bit@ic.unicamp.br) received his Bachelor's degree in computer science from the Federal University of Parana, Brazil, in 2004. He received his Master's degree in 2006 and Ph.D. degree in 2010 from the University of Campinas (UNICAMP), Brazil. He is currently an assistant professor at UNICAMP, and his main interests are in the areas of virtualization and scheduling in grids and clouds.

EDMUNDO R. M. MADEIRA (edmundo@ic.unicamp.br) is a full professor at UNICAMP. He received his Ph.D. in electrical engineering from UNICAMP. He has published over 120 papers in national and international conferences and journals. He was General Chair of the 7th Latin American Network Operation and Management Symposium (LANOMS '11), and is Technical Program Co-Chair of the IEEE LatinCloud '12. His research interests include network management, future Internet, and cloud computing.

NELSON L. S. DA FONSECA (nfonseca@ic.unicamp.br) obtained his Ph.D. degree from the University of Southern California in 1994. He is a full professor at the State University of Campinas. He has published 300+ refereed papers and supervised 50+ graduate students. He is currently ComSoc Vice President — Member Relations. He served as EiC of *IEEE Communications Surveys and Tutorials*, Member-at-Large of the ComSoc Board of Governors, Director of Latin America Region, and Director of Online Services.