Analyzing Scheduling Information Requirements for Deploying Kepler Workflows on the Cloud

Mr.Diddela Jagan Mohan[1]., M.Bhavana [2]

1 Assistant Professor, Department of CSE, Malla Reddy College of Engineering for Women., Maisammaguda., Medchal., TS, India

2, B.Tech CSE (20RG1A05M6),

Malla Reddy College of Engineering for Women., Maisammaguda., Medchal., TS, India

Abstract

Scientists have relied on the Kepler scientific workflow system to help them automate experiments across many different fields using distributed computing platforms. An assigned director oversees the execution of a process in Kepler. It is the responsibility of the user to provide the exact resources in the realm of computing that carry out the workflow's duties. Scientists' technical burdens may be further reduced with the help of a workflow scheduler that can distribute workflow tasks across available resources. We evaluate numerous cloud workflow scheduling methods to determine what data must be exposed for a scheduler to successfully plan for the execution of a Kepler process in the cloud. We explain the value by discussing the benefits to workflow scheduling of various forms of information relating to workflow tasks, cloud resources, and cloud providers.

1 Introduction

Management solutions for scientific workflows, like Kepler [1], have been used to make science more accessible by giving researchers the means to set up and monitor automated runs of their calculations. Workflows are able to be executed by these systems either locally or on the grid, a distributed computing infrastructure. Offers the supercomputer power required for the complex calculations inherent in scientific applications [2]. With the advent of cloud computing, it is now feasible to run scientific processes on cloud resources [3], which may be provisioned, on demand and therefore reduce the need for costly upfront hardware. Research into the execution of processes in a cloud computing environment has increased in recent years due to the many advantages of doing so. A scheduler, which determines the assignment of workflow tasks (called actors in Kepler) to cloud resources, is a crucial component of cloud-based process execution. Multiple "directors" may be applied to a Kepler process to manage its execution [4]. Users must manually designate the computing resources on which to run actors, since Kepler's directors do not automate this process. Kepler may need a scheduler in order to free up scientists from technological hassles. In this research, we analyze current cloud workflow scheduling methods to determine the sorts of data that should be sent by the Kepler environment to its scheduler in order to facilitate the construction of this scheduler.

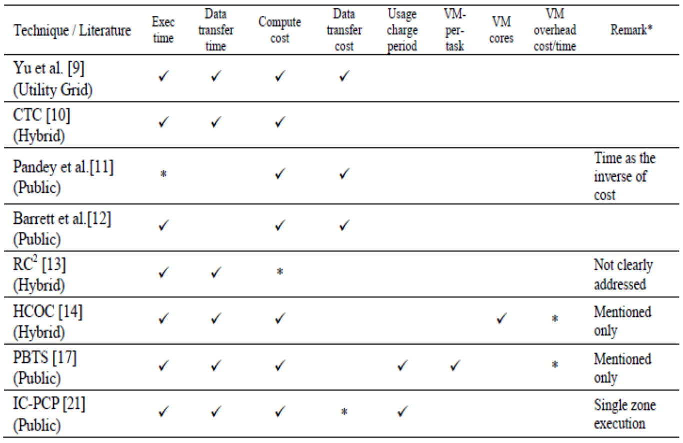
Workflow scheduling in the cloud environment differs from grid scheduling in that it must account for challenges unique to cloud computing. Schedulers in the cloud, for instance, must take into account more than just the time it takes to complete a process execution before a fee is charged. Grid resources are more varied and are not controlled by users, while cloud resources, in this instance virtual machines, are more uniform and are depending on the request from users (or from the schedulers) [3]. As a result, these changes need more data for scheduling than was previously necessary. A variety of cloud workflow scheduling methods is examined in this research, all of which make use of data unique to the cloud computing setting. Some methods (and the data they depend on) may become obsolete as cloud computing and cloud service providers continue to develop. So, we also talk about and defend the significance of each piece of data we find in the contemporary cloud computing setting. This paper will have the following outline. Methods for planning workflows in the cloud are discussed in Section 2. The third section highlights and describes how this data is used to solve cloud-specific problems. The last section of the study discusses the implications of our findings for future research.

2 Cloud Workflow Scheduling Techniques

Prior to the advent of cloud computing, scientific workflows were often implemented in the grid, and many years of research and development went into grid workflow scheduling, yielding a variety of methodologies. However, it is possible that these methods can't be used to schedule operations in the cloud due to the dissimilarities between the grid and the cloud environment. That's why they're focusing on cloud-based process scheduling instead. In light of the constant development of cloud infrastructure and cloud-based software, new scheduling methods have emerged. This is because the scheduling procedure now requires more data about processes and cloud resources. Scheduling tasks for a grid-based workflow typically depends on three metrics: task execution time, grid resource wait time, and file staging or data transfer time [5]. Specific problems, such as resource competition [6], fault tolerance [7], and dependability [8], may need additional metrics or information, which may be used by various strategies. Users are paid for using the cloud's shared pool of computing resources, which may include both physical servers and virtual machines, on an as-needed basis. With the entire execution cost now becoming a more crucial statistic, the three traditional KPIs are no longer adequate.

In order to determine what scheduling data is unique to cloud computing, this section examines numerous cloud workflow scheduling strategies. In order to show how various methods have developed through time, we provide them in chronological sequence. Fewer strategies address execution cost in utility grid, although several concentrate on lowering make span in grid workflow scheduling. One such method was presented by Yu et al. [9], and it involves splitting a workflow into sub-processes based on synchronization tasks and then allocating a common deadline to each of these sub-processes. As a result, we aim to keep each partition's deadline in mind while minimizing its associated costs. The data transport and grid resource utilization costs are taken into account in this early method. Prior to the advent of cloud computing, scientific workflows were often implemented in the grid, and many years of research and development went into grid workflow scheduling, yielding a variety of methodologies. However, it is possible that these methods can't be used to schedule operations in the cloud due to the dissimilarities between the grid and the cloud environment. That's why they're focusing on cloud-based process scheduling instead. In light of the constant development of cloud infrastructure and cloud-based software, new scheduling methods have emerged. This is because the scheduling procedure now requires more data about processes and cloud resources. Scheduling tasks for a grid-based workflow typically depends on three metrics: task execution time, grid resource wait time, and file staging or data transfer time [5]. Specific problems, such as resource competition [6], fault tolerance [7], and dependability [8], may need additional metrics or information, which may be used by various strategies. Users are paid for using the cloud's shared pool of computing resources, which may include both physical servers and virtual machines, on an as-needed basis. With the entire execution cost now becoming a more crucial statistic, the three traditional KPIs are no longer adequate. In order to determine what scheduling data is unique to cloud computing, this section examines numerous cloud workflow scheduling strategies. In order to show how various methods have developed through time, we provide them in chronological sequence. Fewer strategies address execution cost in utility grid, although several concentrate on lowering make span in grid workflow scheduling. One such method was presented by Yu et al. [9], and it involves splitting a workflow into sub-processes based on synchronization tasks and then allocating a common deadline to each of these sub-processes. As a result, we aim to keep each partition's deadline in mind while minimizing its associated costs. The data transport and grid resource utilization costs are taken into account in this early method.

Bettencourt and Madeira [14] devised the "HCOC" method in 2011 to plan cloud processes within deadlines while decreasing compute cost. The method is based on a hybrid cloud architecture, which consists of both a private cloud with diverse resources and a public cloud (unlimited) resources. As a first step in developing a strategy for the private cloud, we use "PCH," a grid workflow scheduling approach [15] that accounts for execution time and data transfer. If the current schedule seems like it may miss the deadline, a rescheduling event is triggered to choose and redistribute workloads to public cloud resources. When choosing resources from the public cloud, we look at things like cost, performance, and the number of available CPU cores. Cloud resources that are idle while waiting for incoming data to be sent incur an opportunity cost, as the authors point out [16]. However, this was an oversight in the HCOC algorithm. In Table 1, we compare and contrast the various cloud workflow scheduling strategies and the cloud execution model they recommend. Due to their importance in determining both cost and make span, the metrics of task execution time, compute cost, and data transfer time are widely used throughout the majority of the approaches shown here. The next section discusses the significance of the remaining five to contemporary cloud computing.



Technique / Literature	Exec time	Data transfer time	Compute cost	Data transfer cost	Usage charge period	VM-per-task	VM cores	VM overhead cost/time	Remark*
Yu et al. [9] (Utility Grid)	✓	✓	✓	✓					
CTC [10] (Hybrid)	✓	✓	✓						
Pandey et al.[11] (Public)	*		✓	✓					Time as the inverse of cost
Barrett et al.[12] (Public)	✓		✓	✓					
RC^2 [13] (Hybrid)	✓	✓	*						Not clearly addressed
HCOC [14] (Hybrid)	✓	✓	✓				✓	*	Mentioned only
PBTS [17] (Public)	✓	✓	✓		✓	✓		*	Mentioned only
IC-PCP [21] (Public)	✓	✓	✓	*	✓				Single zone execution

Table 1: Summary of the information utilized by cloud workflow scheduling techniques

3 Discussion on Information Utilized in Workflow Scheduling

Not as much focus has been placed on the fourth statistic, which is the cost of moving data into and out of the cloud, as there has been on the other three. The assumption made by each method may determine whether or not this data is included. Infrastructure-as-a- Windows Azure [22] and Amazon Data transfers to and from EC2 [20] cloud data centres are cheap. Assuming that a process execution only occurs in a single cloud, this cost will be spent mostly when taking data back out of a cloud data centre once the workflow has completed. Because data may need to be transported from cloud centre to private resources during execution and vice versa, the cost of executing a workflow in a hybrid cloud might vary depending on the data dependencies between jobs. Costs associated with transporting data across providers become more convoluted when operating in an "interclub" [23] environment, when a process is completed employing resources from numerous cloud providers. However, the volume of data delivered during an operation might also affect the final price. Therefore, the cost of data transmission must be included into the scheduling mechanism of a system that primarily supports data-intensive activities.

Two of the more complex methods discussed in this study (PBTS and IC-PCP) account for the time value of money spent on virtual machines. If the majority of the jobs in a workflow have execution periods that are less than the charging period, then this is an important decision to make. In such case, the assigned VMs result in underutilization and greater costs for customers during non-peak times [17, 21]. In contrast, as of the year 2013, both Windows Azure and Google Compute Engine impose a per-minute fee on their virtual machines. Due to the reduced charge duration, the benefits of taking charge period into account [21] are less, and it may not be worthwhile to build such sophisticated scheduling algorithms. However, other cloud providers, such Amazon EC2, continue to charge customers on an hourly basis for their service. Until the majority of cloud providers switch to a shorter billing cycle, thinking about the charge term may be helpful. Shorter billing cycles make it more convenient to assign a virtual machine for a limited time to complete a modest activity at a lower overall cost. However, this might result in unnecessary costs. The time and resources needed to boot up a virtual machine are factors to think about, as noted in [3, 16, and 17]. Despite the fact that a virtual machine's image may be stored in the cloud, it still needs some time to boot the operating system before it can be used to perform a job [3]. There may also be a period of idle waiting as the machine awaits the transmission of input data before execution can begin [16]. These complications add unnecessary expense and delay to an operation. When deciding whether to create a new virtual machine (in order to achieve a deadline) or to terminate one, the scheduler must take this overhead into account (to reduce cost when it is no longer required).

4 Information Requirements for Cloud Workflow Scheduler

Ultimately, the execution time, data transfer time, and compute cost is the three most important metrics that must be supplied for a Kepler system workflow scheduler implementation in order to estimate cost and make span. In addition to equating execution time with virtual machine ratio of performance to job size (in MIPS) and overall task size (in millions of instructions) [16]. A second option is for the scheduler to keep track of the amount of time it takes to complete each job on each virtual machine internally (or as part of provenance) in order to calculate estimated task completion times. Schedulers may provide conservative estimates of transfer times by recording information about the amount of the data being transmitted and the capacity of network connections [6]. To aid in planning of data-intensive operations, it is helpful to have the costs associated with transporting data to and from each cloud provider in each availability zone (or area). The scheduler needs access to the number of cores set for each virtual machine instance type in order to take advantage of workflow parallelism. To facilitate a distributed programming paradigm, it is necessary to have as many virtual machines as the number of tasks. Users may set this value in a parameter associated with the task actor. Although not explicitly addressed in any study provided here, it is important to keep tabs on the time it takes to start various kinds of virtual machines in order to rationalize resource allocation and release [3]. The scheduler is a potential location for including this monitoring. Finally, we think that it may not be essential throughout the use fee time. As more cloud service providers provide shorter billing cycles, the benefits of this feature may no longer be sufficient to cover the costs associated with developing and implementing it.

5 Conclusions and Future Work

In this research, we analyze the requirements of a workflow scheduler and draw conclusions on what the Keller system environment should provide. The value of various forms of data in the present-day environment of cloud computing is examined. Based on our work developing a prototype scheduler in the Kepler-based tool "Nimrod/K" [6, 24, 25], "Nimrod Director" [26] uses this scheduler to take care of scheduling process actors onto computational resources. A cloud-based process scheduler may be implemented in the Kepler system, and this may be a viable alternative. The prototype scheduler, however, was designed for a grid environment and hence did not take into account this novel characteristic of cloud computing. Furthermore, it is possible that the Kepler system may not immediately provide access to some of the information described in this work, such as compute cost, data transmission cost, and the number of cores in each virtual machine. In the real implementation, it may be necessary to add an extra part to collect such data. Several approaches, including those discussed in this research, makes assumptions about the information and concerns unique to the workflow and cloud execution settings, such as task need and user interaction [10]. As a follow-up, we're doing a thorough literature review to compile a taxonomy that may guide future innovation in cloud workflow scheduling methods and tools.

References

[1] B. Ludäscher, I. Altintas, C. Berkley, D. Higgins, E. Jaeger, M. Jones, E. A. Lee, J. Tao, and Y. Zhao, "Scientific workflow management and the Kepler system," Concurr. Comput. : Pact. Expert. vol. 18, pp. 1039-1065, 2006.

[2] E. Deelman, D. Gannon, M. Shields, and I. Taylor, "Workflows and e-Science: An overview of workflow system features and capabilities," Future Generation Computer Systems, vol. 25, pp. 528-540, 2009.

[3] G. Mates, W. Gentzsch and C. J. Ribbens, "Hybrid Computing—Where HPC meets grid and Cloud Computing," Future Generation Computer Systems, vol. 27, pp. 440-453, 2011.

[4] Kepler User Manual. Available: https://kepler-project.org/users/documentation Accessed January, 2014.

[5] J. Yu, R. Buyya and K. Ramamohanarao, "Workflow Scheduling Algorithms for Grid Computing," in Metaheuristics for Scheduling in Distributed Computing Environments, ed, 2008, pp. 173-214.

[6] S. Smanchat, M. Indrawan, S. Ling, C. Enticott, and D. Abramson, "Scheduling parameter sweep workflow in the Grid based on resource competition," Future Generation Computer Systems, vol. 29, pp. 1164-1183, 2013.

[7] K. Plankensteiner, R. Prodan and T. Fahringer, "A new fault tolerance heuristic for scientific workflows in highly distributed environments based on resubmission impact," in Proceedings of the 5th IEEE International Conference on e-Science (e-Science '09), Oxford, UK, 2009, pp. 313-320.

[8] Y. Tao, H. Jin, S. Wu, X. Shi, and L. Shi, "Dependable Grid Workflow Scheduling Based on Resource Availability," J. Grid Comput., vol. 11, pp. 47-61, 2013.

[9] J. Yu, R. Buyya and C. K. Tham, "Cost-based scheduling of scientific workflow applications on utility grids," in Proceedings of the First International Conference on e-Science and Grid Computing, 2005, pp. 140-147.

[10] K. Liu, "Scheduling algorithms for instance-intensive cloud workflows," PhD Thesis, Faculty of Information and Communication Technologies, Swinburne University of Technology, 2009.