Assessing the Impact of Supporting Information on the Scheduling of Scientific Workflows on Clouds

Eduardo Cotrin Teixeira[1], **Daniel Cordeiro**[2], **Kelly Rosa Braghetto**[3]

[1]Universidade Tecnológica Federal do Paraná (UTFPR)
Campus Cornélio Procópio – PR – Brazil

[2]Escola de Artes, Ciências e Humanidades
[3]Departamento de Ciência da Computação - Instituto de Matemática e Estatística
Universidade de São Paulo (USP) – SP – Brazil

cotrin@utfpr.edu.br, daniel.cordeiro@usp.br, kellyrb@ime.usp.br

Abstract. *Executing scientific workflows in high-performance cloud computing platforms requires the use of scheduling algorithms that allow workflows execution as fast as possible, while minimizing the monetary cost of such executions. In this work we study how the use of supporting information can offer guidance to scheduling algorithms, helping them to devise more efficient execution plans in terms of the total execution time (makespan) and the total monetary cost. Using two large-scale scientific workflows, our experiments showed that simple modifications on a classical scheduling algorithm (HEFT), in conjunction with the appropriate supporting information, could reduce the monetary cost of an execution in up to 59% and reduce the makespan in up to 8.6%.*

1. Introduction

Cloud computing platforms are a viable alternative for running scientific workflows. However, the scheduling on this type of platform generally must consider specific constraints such as a limited budget and the type of computational resources required for the execution. To design effective scheduling algorithms under such constraints that will ensure efficient workflow executions, one must rely on *supporting information*, such as estimated duration of workflow activities or execution time and cost constraints.

This work explores the use of supporting information that can be added to scientific workflow models to support their scheduling and execution on clouds. To assess the impact of use of supporting information, we performed experiments with large-scale, real scientific workflows executed through the workflow management system (WfMS) Pegasus [Deelman et al. 2009]. To be able to consider the supporting information associated to the workflows in the scheduling phase, we modified a classical scheduling algorithm (HEFT) [Topcuouglu et al. 2002] existent in Pegasus to implement two new ones. The first algorithm minimizes the workflow total execution time (makespan), while the second minimizes the total monetary cost to execute the workflow in the cloud computing under a deadline constraint. In the experiments, we observed a reduction in the monetary cost of the workflow execution in the cloud of up to 59% and a reduction in the makespan of up to 8.6%, when compared to the scheduling with no supporting information available.

2. Basic Definitions

A scientific workflow is the automation of a scientific experiment or process, expressed in terms of the activities and their interdependencies [Cuevas-Vicenttin et al. 2012]. Sci-

entific workflow models can be expressed as Directed Acyclic Graphs (DAGs), where each node represents an activity and each directed edge represents a precedence relation between two activities. A *scheduling algorithm* uses this information to choose, for each activity, the resource and execution time that optimizes a specific objective function.

The essential information required to schedule and execute a workflow is: description of the computational resources available for the execution, scheduling objective to be optimized and specification of the activities with their precedence relations (i.e., the workflow DAG). The specification of the activities includes the procedure to be performed (program, script, service, etc.), and the parameters required for the execution (including input and output data). Any additional information related to the workflow or to the execution environment can be considered a *supporting information* for the scheduling.

3. Experimental Evaluation of the Supporting Information Impact

Support information impact on scheduling and execution of scientific workflows on cloud was evaluated using InterNuvem[1], a pay-per-use academic IaaS cloud platform. Ten virtual machines with Ubuntu 12.04 operating system were used, namely, 4 Standard VMs (2 CPUs and 4GB RAM - R$ 0.112/h), 3 Advanced VMs (4 CPUs and 8GB RAM - R$ 0.202/h) and 3 High Performance VMs (8 CPUs and 32 GB RAM - R$ 0.542/h). We used the WfMS Pegasus version 4.5.0 to manage the execution of two large-scale scientific workflows — Montage (four instances, with degrees 0.5, 1.0, 2.0, and 4.0) and Epigenomics (two instances, using TAQ and HEP datasets) [Juve et al. 2013].

We consider the scheduling of scientific workflows as a bicriteria optimization problem, where the performance of the workflow execution is related to two different (and somehow contradictory) performance objectives: the makespan (finishing time of the last activity) and the monetary cost of execution. To schedule and execute the workflow instances considering supporting information, we developed and evaluated two scheduling algorithms: HEFTData and HEFTDeadline.

3.1. HEFTData

HEFT (Heterogeneous Earliest-Finish-Time) [Topcuouglu et al. 2002] is an algorithm for scheduling dependent activities into heterogeneous machines that minimizes makespan. HEFT assigns to each activity a rank (i.e., the expected distance from the end of the execution) defined as $rank(n_i) = \overline{w_i} + \max_{n_j \in succ(n_i)}(\overline{c_{i,j}} + rank(n_j))$, where n_i represents the i^{th} activity, $\overline{w_i}$ is an average computation cost of activity i among all the resource types, $succ(n_i)$ is the set of all activities that immediately depend on activity n_i, and $\overline{c_{i,j}}$ is the average communication cost from activity n_i to n_j considering all pairs of machines types. After that, HEFT assigns the prioritized activities (in descending order of rank) to the machines. Pegasus' HEFT implementation considers only the activity's expected execution time, ignoring communication time. For workflows with large volumes of data, data transfer times may actually exceed the activity durations.

We have implemented in Pegasus a new version of the HEFT algorithm, called HEFTData, which considers supporting information about *input data volume*, besides the *activity estimated duration*. HEFTData works similarly to HEFT, except that tasks' rank

[1]InterNuvem: `https://internuvem.usp.br/`

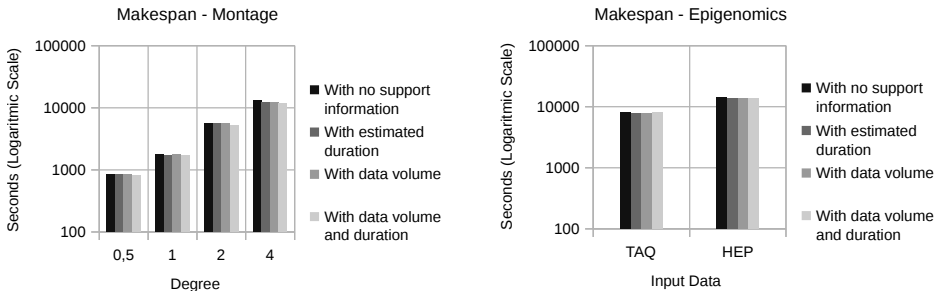

Figure 1. Average makespan for Montage and Epigenomics.

and expected completion time also consider data transfer times. To evaluate HEFTData, additional supporting information was needed: the *estimated average duration of each activity on each type of VM* used in the experiments (measured with at least 20 executions in each type of VM) and the *volume of data to be transferred between the activities*, to calculate the data transfer times. All supporting information was defined as part of the workflow model itself and was determined for each edge in the workflow DAG.

3.2. HEFTDeadline

Monetary cost is an important issue to execute workflows on clouds. Virtual machines may have different prices depending on their configuration, so the choices the scheduler makes have a direct impact on the cost of the execution. On the one hand, faster machines tend to be more expensive, and algorithms like HEFT tend to use more of this type of machine to minimize the makespan. On the other hand, if the scheduler uses cheaper machines, the workflow execution can take more time than the desired by the scientist. HEFTDeadline was designed to prioritize the execution of workflow activities in machines that provide the lowest monetary cost, considering as a constraint the supporting information *deadline*, that must be defined by the scientist for the workflow.

Activities and ranks are computed as in HEFTData. However, when mapping the activities to virtual machines, HEFTDeadline considers only suitable machines resulting on the cheapest execution costs, considering estimated duration of the activity execution multiplied by the price per use time of the machine. In this way, a more expensive machine can be selected if the cost of the activity execution becomes lower due to a shorter estimated duration in this machine. The execution cost also considers that a machine can have multiple execution nodes (e.g., multiple cores or hyper-threading capabilities). If a VM is allocated to an activity, it will be fully billed regardless of how many of its nodes are actually used. Therefore, an activity mapped to an execution node only has a cost if its completion time is greater than the latest completion time of the activities mapped to the other execution nodes of the same VM. The scheduler will not allocate a new VM while execution nodes on active VMs are still available and are more cost effective.

4. Results and Conclusion

Each instance was executed at least 8 times for each scheduler. The results were evaluated in terms of makespan (total duration in seconds) and monetary cost. Figure 1 shows the average makespan obtained using HEFTData and four different configurations of supporting information: (i) using no information, (ii) using only the *estimated activity durations*,

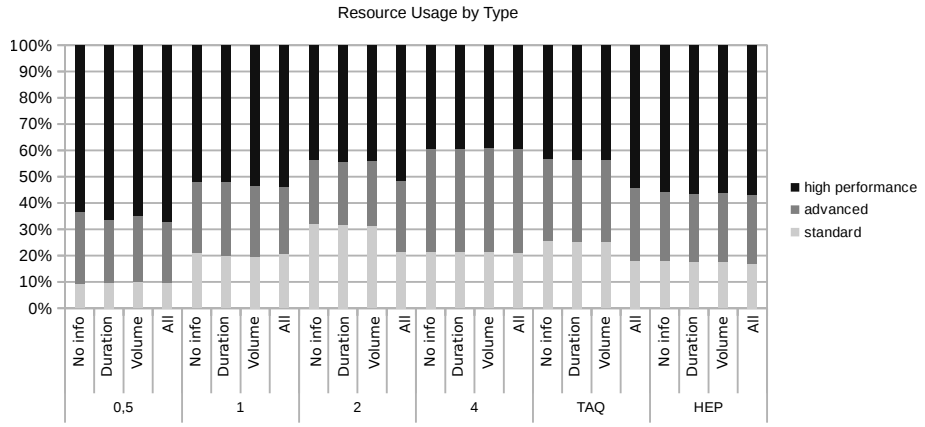

Figure 2. Average distribution of the virtual machines by type.

Table 1. Average monetary costs (in R$) using HEFTDeadline.

	HEFTData		HEFTDeadline			
Workflow	**Makespan**	**Cost**	**Deadline**	**Makespan**	**Cost**	**Cost Reduction**
Montage 0.5	819	0.17	983	906	0.07	59.1%
Montage 1.0	1711	0.39	2053	1924	0.16	59.2%
Montage 2.0	5139	1.22	6167	5678	0.55	55.4%
Montage 4.0	11818	2.56	14182	13543	1.34	47.7%
Epigenomics TAQ	7897	2.83	9476	9187	2.45	13.4%
Epigenomics HEP	13602	5.82	16322	16138	2.37	59.2%

(iii) using only the *input data volumes*, and (iv) using both duration and data volume. In all instances, the use of some supporting information reduced the average makespan. The greatest reduction (8.6%) was observed in the executions of Montage with degree 2.0.

Figure 2 shows the distribution of the average makespan by VM types used in the execution. High-performance VMs tended to be used more often when supporting information was used. Increased parallelism (machines with more processing nodes) and execution speed of the workflow activities provided improvement in the makespan.

Table 1 shows the average monetary costs of the executions with the HEFTDeadline. The deadline used for each instance was 120% of the average makespan obtained for the same instance with HEFTData using the support information *estimated activity durations* and *input data volumes*. The table shows that, with an increase of 20% in the makespan, the HEFTDeadline reduced the workflow average execution cost up to 59.2%.

References

Cuevas-Vicenttin, V. et al. (2012). Scientific workflows and provenance: Introduction and research opportunities. *Datenbank-Spektrum*, 12(3):193–203.

Deelman, E., Gannon, D., Shields, M., and Taylor, I. (2009). Workflows and e-Science: An overview of workflow system features and capabilities. *Future Generation Computer Systems*, 25(5):528–540.

Juve, G., Chervenak, A., Deelman, E., Bharathi, S., Mehta, G., and Vahi, K. (2013). Characterizing and profiling scientific workflows. *Future Generation Computer Systems*, 29(3):682–692.

Topcuouglu, H., Hariri, S., and Wu, M.-Y. (2002). Performance-effective and low-complexity task scheduling for heterogeneous computing. *IEEE Trans. Parallel Distrib. Syst.*, 13(3):260–274.