HAL archives-ouvertes.fr

# Scientific Workflow Execution with Multiple Objectives in Multisite Clouds

Ji Liu, Esther Pacitti, Patrick Valduriez, Daniel de Oliveira, Marta Mattoso



## To cite this version:

Ji Liu, Esther Pacitti, Patrick Valduriez, Daniel de Oliveira, Marta Mattoso. Scientific Workflow Execution with Multiple Objectives in Multisite Clouds. BDA: Gestion de Données - Principes, Technologies et Applications, LIAS / ISAE-ENSMA, Poitiers, Nov 2016, Poitiers, France. lirmm-01409092

## HAL Id: lirmm-01409092

## https://hal-lirmm.ccsd.cnrs.fr/lirmm-01409092

Submitted on 5 Dec 2016

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution| 4.0 International License

# Scientific Workflow Execution with Multiple Objectives in Multisite Clouds

Ji Liu
MSR-INRIA Joint Centre
Inria and LIRMM
University of Montpellier
Ji.Liu@inria.fr

Esther Pacitti
University of Montpellier
Inria and LIRMM
Esther.Pacitti@lirmm.fr

Patrick Valduriez
MSR-INRIA Joint Centre
Inria and LIRMM
University of Montpellier
Patrick.Valduriez@inria.fr

Daniel de Oliveira
Institute of Computing
Fluminense Federal University
danielcmo@ic.uff.br

Marta Mattoso
COPPE
Federal University of Rio de Janeiro
marta@cos.ufrj.br

## 1. INTRODUCTION

Large-scale *in silico* scientific experiments typically take advantage of Scientific Workflows (SWfs) to model data operations. A SWf is the assembly of scientific data processing activities with data dependencies among them. An activity is the description of a piece of work that forms a logical step within a SWf representation. A SWf Management System (SWfMS) is the tool to manage SWfs [3]. In order to execute SWfs efficiently, SWfMSs typically exploit High Performance Computing (HPC) resources in a cluster, grid or cloud environment. Clouds have become an interesting solution for SWf execution because of various advantages. A cloud is typically made of several sites (or data centers), each with its own resources and data. Thus, in order to use more resources at different sites, SWfs could also be executed in a distributed manner at different sites. Nowadays, the computing resources or data of a cloud provider such as Amazon or Microsoft are distributed at different sites and should be used during the execution of SWfs. As a result, a multisite cloud is an appealing solution for large scale SWf execution. As defined in [2], a multisite cloud is a cloud with multiple sites, each at a different location (possibly in a different region) and being explicitly accessible to cloud users, typically in the data center close to them for performance reasons. In addition, there is a stored data constraint, *i.e.* some data cannot be moved to other sites because of big amounts or proprietary issues, in a multisite cloud.

To enable SWf execution in a multisite cloud, the execution of each activity should be scheduled to a corresponding site. Then, the scheduling problem is to decide where to execute the activities. In general, to map the execution of activities to distributed computing resources is an NP-hard problem. The scheduling problem may have multiple objectives, *i.e.* multi-objective, *e.g.* reducing execution time or monetary cost *etc.* Thus, the multisite scheduling problem must take into account the impact of resources distributed at different sites, *e.g.* different bandwidths, data distribution and costs to use Virtual Machines (VMs) at different sites. To the best of authors' knowledge, there is no solution to execute SWfs in a multisite cloud while considering both multiple objectives and dynamic VM provisioning. The related work either focuses on static VM provisioning, single objective or single site execution. Static VM provisioning refers to the use of the existing VMs (before execution) for SWf execution without modifying VMs during execution. In addition, existing cost models are not suitable for the SWfs that have a big part of sequential workload.

In this short paper (see [4] for the extended version), we propose a general solution based on multi-objective scheduling to execute SWfs in a multisite cloud with the following main contributions: the design of a multi-objective cost model, SSVP VM provisioning approach, ActGreedy scheduling algorithm and an extensive experimental evaluation.

(c) 2016, Copyright is with the authors. Published in the Proceedings of the BDA 2016 Conference (15-18 November, 2016, Poitiers, France). Distribution of this paper is permitted under the terms of the Creative Commons license CC-by-nc-nd 4.0.
(c) 2016, Droits restant aux auteurs. Publié dans les actes de la conférence BDA 2016 (15 au 18 Novembre 2016, Poitiers, France). Redistribution de cet article autorisée selon les termes de la licence Creative Commons CC-by-nc-nd 4.0.
*BDA* 2016, 15 au 18 Novembre 2016, Poitiers, France.

## 2. FRAGMENT SCHEDULING

In this section, we present the multisite SWfMS architecture, cost model, SSVP and ActGreedy.

The architecture of a multisite SWfMS is composed of four modules: workflow partitioner, multisite scheduler, single site initialization, and single site execution. The workflow partitioner partitions a SWf into fragments. After SWf partitioning, the fragments are scheduled to sites by the multisite scheduler. After scheduling, in order to avoid restarting VMs for the execution of continuous activities, all the activities scheduled at the same site are grouped as a fragment to be executed. Then, the single site initialization module prepares the execution environment for the fragment, using two components, *i.e.* VM provisioning and multisite data transfer. At each site, the VM provisioning component deploys and initializes VMs for the execution of SWfs. Finally, the single site execution module starts the execution of the fragments at each site. This can be realized by an existing single site SWfMS, *e.g.* Chiron [5].

It is difficult to estimate the execution time and monetary costs for the whole SWf even with a scheduling plan according to existing cost models [1] since it is hard to generate a VM provisioning plan for each site with global desired execution time and monetary costs. As shown in Formula 2.0.1, we decompose the cost model as the sum of the cost of executing each fragment.

$$Cost(Sch(SWf, S)) = \sum_{wf_i \in SWf}^{Schedule(wf_i, s_j)=1} Cost(wf_i, s_j) \tag{2.0.1}$$

In the cost model, we also take the cost to provision VMs and the sequential workload into consideration in order to estimate the cost more precisely. In addition, we take advantage of Amdahl's law to estimate the execution time.

We propose a single site VM provisioning algorithm, called SSVP, to generate VM provisioning plans, which minimizes the cost to execute a fragment at Site $s$. First, SSVP calculates a near-optimal number of virtual CPUs to instantiate based on the cost model. Then, it optimizes the provisioning plan to reduce the cost to execute a fragment at Site $s$. Afterward, SSVP calculates the cost to execute the fragment at the site based on the cost model and improves the provisioning plan by inserting a new VM, modifying an existing VM or removing an existing VM. If the cost to execute the fragment at Site $s$ can be reduced by improving the provisioning plan, the provisioning will be updated, and the improvement of provisioning plan continues.

Finally, we propose our multisite scheduling algorithms, *i.e.* *ActGreedy*, which schedules the most suitable site to each fragment. In ActGreedy, all the fragments of a SWf are not scheduled at beginning. During the scheduling process, all the fragments are scheduled at a corresponding site, which takes the minimum cost with the consideration of the stored data constraint while the cost is the minimum compared to other sites. As a result, the cost of executing a SWf in a multisite cloud is minimized. ActGreedy schedules fragments of multiple activities. ActGreedy can schedule a pipeline of activities to reduce data transfer between different fragments, *i.e.* the possible data transfer between different sites. A pipeline is a group of activities with a one-to-one, sequential relationship between them. In addition, ActGreedy makes a trade-off between time and monetary costs by using SSVP. ActGreedy schedules the available fragments, while choosing the best site for an available fragment rather than choosing the best fragment for an available site.

# 3. EXPERIMENTAL EVALUATION

In this section, we present an experimental evaluation of ActGreedy algorithm. All experiments are based on the execution of a SciEvol SWf in Microsoft Azure multisite cloud with three sites. We compare ActGreedy with LocBased [2] and SGreedy [1], as well as with two general algorithms, *i.e.* Genetic and Brute-force. In the experiments, workflow partitioner, multisite scheduler and single site initialization are simulated, but the execution of fragments is performed in a real environment by Chiron.

The results show that LocBased corresponds to up to 21.75% higher cost than ActGreedy and SGreedy takes up to 74.51% higher cost than ActGreedy based on our proposed cost model. Based on an existing cost model, the advantage of ActGreedy can reduce the total cost up to 10.7% compared with LocBased and 17.2% compared with SGreedy. In addition, when the weight of reducing execution is low, ActGreedy may correspond to bigger execution time but when the weight is high, ActGreedy corresponds to less execution time. The execution with ActGreedy always takes less monetary cost compared with LocBased (up to 14.12%) and SGreedy (up to 17.28%). In addition, the experiments also show that the scheduling time of Genetic and Brute-force is much longer than ActGreedy (up to 577 times and 128 times). For instance, with more than 22 activities or 10 sites, the scheduling time of both Genetic and Brute-force exceeds the execution while the scheduling time of ActGreedy remains small.

# 4. CONCLUSION

In this paper, we proposed a general solution based on multi-objective scheduling to execute SWfs in a multisite cloud (from the same provider). We first proposed a novel multi-objective cost model, based on which, we proposed a dynamic VM provisioning approach, namely SSVP, to generate VM provisioning plans for fragment execution. The cost model aims at minimizing two costs: execution time and monetary costs. Our proposed fragment scheduling approach that is ActGreedy, allows for considering stored data constraints while reducing the cost based on our multi-objective cost model to execute a SWf in a multisite cloud. We evaluated our approaches by executing SciEvol in Microsoft Azure cloud. The results show that since it makes a good trade-off between execution time and monetary costs based on SSVP, ActGreedy leads to the least total normalized cost, which is calculated based on our multi-objective cost model, than LocBased (up to 21.75%) and SGreedy (up to 74.51%) approaches. Additionally, ActGreedy scales very well, *i.e.* it takes a very small time to generate the optimal or near optimal scheduling plans when the number of activities or sites increases, compared with general approaches, *e.g.* Genetic and Brute-force.

# References


[1] D. de Oliveira, K. A. C. S. Ocaña, F. Baião, and M. Mattoso. A provenance-based adaptive scheduling heuristic for parallel scientific workflows in clouds. *Journal of Grid Computing*, 10(3):521–552, 2012.

[2] J. Liu, V. Silva, E. Pacitti, P. Valduriez, and M. Mattoso. Scientific workflow partitioning in multi-site clouds. In *BigDataCloud'2014: 3rd Workshop on Big Data Management in Clouds in conjunction with Euro-Par 2014*, page 12, 2014.

[3] J. Liu, E. Pacitti, P. Valduriez, and M. Mattoso. A survey of data-intensive scientific workflow management. *Journal of Grid Computing*, pages 1–37, 2015.

[4] J. Liu, E. Pacitti, P. Valduriez, D. de Oliveira, and M. Mattoso. Multi-objective scheduling of scientific workflows in multisite clouds. *Future Generation Computer Systems*, 63:76–95, 2016.

[5] E. S. Ogasawara, J. Dias, V. Silva, F. S. Chirigati, D. de Oliveira, F. Porto, P. Valduriez, and M. Mattoso. Chiron: a parallel engine for algebraic scientific workflows. *Concurrency and Computation: Practice and Experience*, 25(16):2327–2341, 2013.