Future Generation Computer Systems 63 (2016) 76–95

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Multi-objective scheduling of Scientific Workflows in multisite clouds

Ji Liu[a,*], Esther Pacitti[a], Patrick Valduriez[a], Daniel de Oliveira[b], Marta Mattoso[c]

[a] Inria, Microsoft-Inria Joint Centre, LIRMM and University of Montpellier, France
[b] Institute of Computing, Fluminense Federal University, Niteroi, Brazil
[c] COPPE, Federal University of Rio de Janeiro, Brazil

HIGHLIGHTS

- A multi-objective cost model that includes execution time and monetary costs.
- A Single Site VM Provisioning (SSVP) approach, to generate VM provisioning plans.
- ActGreedy, an efficient scheduling algorithm for SWf execution in multisite cloud.
- An extensive experimental evaluation in Microsoft Azure using the SciEvol SWf.

ARTICLE INFO

Article history:
Received 26 November 2015
Received in revised form
19 April 2016
Accepted 21 April 2016
Available online 12 May 2016

Keywords:
Scientific workflow
Scientific workflow management system
Multi-objective scheduling
Parallel execution
Multisite cloud

ABSTRACT

Clouds appear as appropriate infrastructures for executing Scientific Workflows (SWfs). A cloud is typically made of several sites (or data centers), each with its own resources and data. Thus, it becomes important to be able to execute some SWfs at more than one cloud site because of the geographical distribution of data or available resources among different cloud sites. Therefore, a major problem is how to execute a SWf in a multisite cloud, while reducing execution time and monetary costs. In this paper, we propose a general solution based on multi-objective scheduling in order to execute SWfs in a multisite cloud. The solution consists of a multi-objective cost model including execution time and monetary costs, a Single Site Virtual Machine (VM) Provisioning approach (SSVP) and ActGreedy, a multisite scheduling approach. We present an experimental evaluation, based on the execution of the SciEvol SWf in Microsoft Azure cloud. The results reveal that our scheduling approach significantly outperforms two adapted baseline algorithms (which we propose by adapting two existing algorithms) and the scheduling time is reasonable compared with genetic and brute-force algorithms. The results also show that our cost model is accurate and that SSVP can generate better VM provisioning plans compared with an existing approach.

1. Introduction

Large-scale *in silico* scientific experiments typically take advantage of Scientific Workflows (SWfs) to model data operations such as loading input data, data processing, data analysis, and aggregating output data. SWfs enable scientists to model the data processing of these experiments as a graph, in which vertices represent data processing activities and edges represent dependencies between them. A SWf is the assembly of scientific data processing activities with data dependencies among them [1]. An activity is the description of a piece of work that forms a logical step within a SWf representation [2]. Since SWf activities may process big data, we can exploit data parallelism whereby one activity corresponds to several executable tasks, each working in parallel on a different part of the input data. Thus, a task is the representation of an activity within a one-time execution of this activity, which processes a data partition or chunk [2].

A SWf Management System (SWfMS) is the tool to manage SWfs [2]. In order to execute SWfs efficiently, SWfMSs typically exploit High Performance Computing (HPC) resources in a cluster, grid or cloud environment. Because of virtually infinite resources, diverse scalable services, stable quality of service and flexible payment policies, clouds have become an interesting solution for SWf execution. In particular, the user of Virtual Machines (VMs) makes it easy to deal with elasticity and workloads that change rapidly. 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 than available at a single site or to access data at different

* Corresponding author.
E-mail address: ji.liu@inria.fr (J. Liu).

http://dx.doi.org/10.1016/j.future.2016.04.014
0167-739X/© 2016 Elsevier B.V. All rights reserved.

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 [3], a multisite cloud is a cloud with multiple data centers, 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.

To enable SWf execution in a multisite cloud, the execution of each activity should be scheduled to a corresponding cloud site (or site for short). 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 [4]. The objectives can be to reduce execution time or monetary cost, to maximize performance, reliability *etc.* Since the SWf execution may take a long time and cost much money, the scheduling problem may have multiple objectives, *i.e.* multi-objective. Thus, the multisite scheduling problem must take into account the impact of resources distributed at different sites, *e.g.* different bandwidths and data distribution at different sites, and different prices for VMs.

In this paper, we propose a general solution based on multi-objective scheduling in order to execute SWfs in a multisite cloud. The solution includes a multi-objective cost model, a Single Site VM Provisioning approach (SSVP) and ActGreedy, a multisite scheduling approach. The cost model includes two objectives, namely reducing execution time and monetary costs, under stored data constraints, which specify that some data should not be moved, because it is either too big or for proprietary reasons. Although useful for fixing some activities, these constraints do not reduce much the complexity of activity scheduling. We consider a homogeneous cloud environment, *i.e.* from single provider. The case of federated clouds (with multiple cloud providers) is beyond the scope of this paper and not a reality (although there are some recent proposals). ActGreedy handles multiple objectives, namely reducing execution time and monetary costs. In order to schedule a SWf in a multisite cloud, the SWf should be partitioned to SWf fragments, which can be executed at a single site. A SWf fragment (or fragment for short) is a subset of activities, dependencies and associated input data of the original workflow [3]. Then, each fragment can be scheduled by ActGreedy to the site that yields the minimum cost among all available sites. When a fragment is scheduled to a site, the execution of its associated activities is scheduled to the site. ActGreedy is based on our dynamic VM provisioning algorithm, called Single Site VM Provisioning (SSVP), which generates VM provisioning plans for the execution of fragments with minimum cost at the scheduled site based on a cost model. The cost model is used to estimate the cost of the execution of SWfs [5] according to a scheduling plan, which defines the schedule of fragments to execution sites. A VM provisioning plan defines how to provision VMs. For instance, it determines the types, corresponding number and the order of VMs to provision, for the execution of a fragment. The VM type determines some parameters such as the number of virtual CPUs, the size of memory and the default storage size of hard disk. The main contributions of this paper are:

1. The design of a multi-objective cost model that includes execution time and monetary costs, to estimate the cost of executing SWfs in a multisite cloud.
2. A single site VM provisioning approach (SSVP), to generate VM provisioning plans to execute fragments at each single site.
3. ActGreedy multisite scheduling algorithm that uses the cost model and SSVP to schedule and execute SWfs in a multisite cloud.
4. An extensive experimental evaluation, based on the implementation of our approach in Microsoft Azure, and using a real SWf use case (SciEvol [6], a bioinformatics Scientific workflow for molecular evolution reconstruction) that shows the advantages of our approach, compared with baseline algorithms.

This paper is organized as follows. Section 2 discusses related work. Section 3 introduces the problems for multisite SWf execution. Section 4 describes the system architecture for SWf execution in a multisite cloud. Section 5 describes our multi-objective optimization approach. Section 6 describes our scheduling approaches including the SciEvol SWf use case, the approaches for SWf partitioning and three scheduling approaches, *i.e.* ActGreedy, LocBased and SGreedy. Section 7 is our experimental evaluation in Microsoft Azure cloud [7]. Section 8concludes.

2. Related work

To the best of authors' knowledge, there is no solution to execute SWfs in a multisite cloud environment that takes into account both multiple objectives and dynamic VM provisioning. The related work either focuses on static VM provisioning [8], single objective [3,9,10,10–16] or single site execution [5,17,18]. Static VM provisioning refers to the use of the existing VMs (before execution) for SWf execution without changing the types of VMs during execution. However, existing cost models are not suitable for the SWfs that have a big part of the sequential workload. For instance, the dynamic approach proposed in [19] ignores the sequential part of the SWf and the cost of provisioning VMs, which may generate VM provisioning plans that yield high cost.

Many solutions for workflow scheduling [9–14] focus on a single objective, *i.e.* reducing execution time. These solutions address the scheduling problem in a single site cloud. Classic heuristics have been used in scheduling algorithms, such as HEFT [15], min–min [16], max–min [16] and Opportunistic Load Balancing (OLB) [10], but they only address the single objective. Furthermore, they are designed for static computing resources in grid or cluster environments. In contrast, our algorithm handles multiple objectives, which are reducing execution time and monetary costs, with dynamic VM provisioning support. Although some general heuristics, *e.g.* genetic algorithms [15], can generate near optimal scheduling plans, it is not always feasible to design algorithms for every possible optimization problem [15] and it is not trivial to configure parameters for the problem. In addition, it may take much time to generate scheduling plans. A brute-force method can generate an optimal scheduling plan, but its complexity is very high.

Some multi-objective scheduling techniques [5,17,18] have been proposed. However, they do not take the distribution of resources at different sites into consideration, so they are not suitable for a multisite environment. De Oliveira et al. [5] propose a greedy scheduling approach for the execution of SWfs at a single site. However, this approach is not appropriate for multisite execution of SWfs as it schedules the most suitable activities to each VM, which may incur transferring of big data. Rodriguez and Buyya [18] introduce an algorithm for scheduling dynamic bags of tasks and dynamic VM provisioning for the execution of SWfs with multiple objectives in a single site cloud. Rather than using real execution, they simulate the execution of SWfs, thus missing the heterogeneity among the activities of the same SWf, to evaluate their proposed approaches. In real SWf execution, the activities generally correspond to different programs to process data. However, in simulations of SWf execution, the activities are typically made homogeneous, namely, they correspond to the same program. Different from the existing approaches, our approach is suitable for multisite execution and is evaluated by executing a real-life SWf on a multisite cloud (Azure).