Advances in Engineering Software 84 (2015) 31–47

Contents lists available at ScienceDirect

Advances in Engineering Software

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

Balancing throughput and response time in online scientific Clouds via Ant Colony Optimization (SP2013/2013/00006)

Elina Pacini [a], Cristian Mateos [b,c,*], Carlos García Garino [a,d]

[a] ITIC – UNCuyo University, Mendoza, Argentina
[b] ISISTAN Research Institute, UNICEN University, Campus Universitario, Tandil B7001BBO, Argentina
[c] Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET), Av. Rivadavia 1917, CABA (C1033AAJ), Argentina
[d] Facultad de Ingeniería – UNCuyo University, Mendoza, Argentina

ARTICLE INFO

Article history:
Available online 31 January 2015

Keywords:
Cloud Computing
Scientific problems
Job scheduling
Swarm intelligence
Ant Colony Optimization
Genetic Algorithms

ABSTRACT

The Cloud Computing paradigm focuses on the provisioning of reliable and scalable infrastructures (Clouds) delivering execution and storage services. The paradigm, with its promise of virtually infinite resources, seems to suit well in solving resource greedy scientific computing problems. The goal of this work is to study private Clouds to execute scientific experiments coming from multiple users, i.e., our work focuses on the Infrastructure as a Service (IaaS) model where custom Virtual Machines (VM) are launched in appropriate hosts available in a Cloud. Then, correctly scheduling Cloud hosts is very important and it is necessary to develop efficient scheduling strategies to appropriately allocate VMs to physical resources. The job scheduling problem is however NP-complete, and therefore many heuristics have been developed. In this work, we describe and evaluate a Cloud scheduler based on Ant Colony Optimization (ACO). The main performance metrics to study are the number of serviced users by the Cloud and the total number of created VMs in online (non-batch) scheduling scenarios. Besides, the number of intra-Cloud network messages sent are evaluated. Simulated experiments performed using CloudSim and job data from real scientific problems show that our scheduler succeeds in balancing the studied metrics compared to schedulers based on Random assignment and Genetic Algorithms.

1. Introduction

Scientific computing is a field of study that applies computer science to solve typical scientific problems in disciplines such as Bioinformatics [44], Earth Sciences [23], High-Energy Physics [7], Molecular Science [53] and even Social Sciences [5]. Scientific computing is usually associated with large-scale computer modeling and simulation, and often requires large amounts of computer resources to satisfy the ever-increasing resource intensive nature of its experiments. An example of these experiments is parameter sweep experiments (PSEs), which we have extensively described in previous works [19,30,36].

Cloud Computing [11] is a paradigm which suits well in solving the above cited computing problems, because of its promise of provisioning infinite resources. Within a Cloud, resources can be effectively and dynamically managed using virtualization technologies. Cloud Computing comes in three flavors: infrastructure, platform, and software as services. In commercial Clouds, these services are made available to customers on a subscription basis using pay-as-you-use models. Although the use of Clouds finds its roots in IT environments, the idea is gradually entering scientific and academic ones [37].

Currently, there are several commercial Clouds that offer computing/storage resources, platform-level services or applications. Moreover, it is possible to build private Clouds (i.e., intra-datacenter) using open-source Cloud Computing solutions. This work is focused on the Infrastructure as a Service (IaaS) model, where physical resources are exposed as services. Under this model, users request virtual machines (VM) to the Cloud, which are then associated to physical resources. However, in order to achieve the best performance, VMs have to fully utilize the physical resources by adapting to the Cloud environment dynamically. To perform this, scheduling the processing units of a Cloud (hosts) is an important issue and it is necessary to develop efficient scheduling strategies to appropriately allocate the VMs in physical resources. Here, *scheduling* refers to the way VMs are allocated to run on the

* Corresponding author at: ISISTAN Research Institute, UNICEN University, Campus Universitario, Tandil B7001BBO, Argentina. Tel.: +54 (249) 4439682x35; fax: +54 (249) 4439681.
E-mail addresses: epacini@itu.uncu.edu.ar (E. Pacini), cmateos@conicet.gov.ar (C. Mateos), cgarcia@itu.uncu.edu.ar (C. García Garino).

http://dx.doi.org/10.1016/j.advengsoft.2015.01.005
0965-9978/

available computing resources, since there are typically many more VMs running than physical resources. The VM allocation is responsibility of a software component called *scheduler*. However, scheduling is an NP-complete [52] problem and therefore it is not trivial from an algorithmic perspective. In this context, scheduling may also refer to two goals, namely delivering efficient high performance computing or supporting high throughput computing. High performance computing (HPC) focuses on decreasing job execution time whereas high throughput computing (HTC) aims at increasing the processing capacity of the system. As will be shown, the studied ACO scheduler attempts to balance both aspects.

Swarm Intelligence (SI) metaheuristics have been suggested as interesting techniques to solve combinatorial optimization problems – e.g., job scheduling – by simulating the collective behavior of social insects swarms [10]. Within these, the ACO metaheuristic proposed by Marco Dorigo [16] was inspired by the ability of real ant colonies to efficiently organize the foraging behavior of the colony using external chemical pheromone trails for communication. Since then, ACO algorithms have been widely used for solving many combinatorial optimization problems [17], many of them closely related to the problem at hand. A review of the literature about the uses of ACO algorithms for scheduling problems can be found in the work of Tavares Neto and Godinho Filo [46]. Moreover, since scheduling in Clouds is also a combinatorial optimization problem, some schedulers in this line that exploit ACO have been surveyed in our previous work [35]. In this paper, we describe a scheduler based on ACO to allocate VMs to physical Cloud resources.

Unlike previous work of our own [19,30], the aim of this paper is to experiment with the ACO scheduler in an online Cloud (non-batch) scenario in which multiple users connect to the Cloud at different times to execute their PSEs. In this paper, by extending the preliminary results first reported in a previous work presented at the Pareng 2013 Conference [36], we have deepened the experimental analysis by incorporating two new pure HTC and HPC scenarios. Moreover, we measure network resources consumed by the scheduler and its competitors when handling VM requests issued by users.

Experiments have been conduced in order to evaluate the trade-off between the number of serviced users (which relates to throughput) among all users that are connected to the Cloud, and the total number of VMs that are allocated by the scheduler (which relates to response time). The more the users served, the more the executed PSEs, and hence throughput increases. Moreover, when more VMs can be allocated, more physical resources can be taken advantage of, and hence PSE execution time decreases. The main performance metric to study in this paper is a weighted metric in which the results obtained from different scheduling algorithms have been normalized and weighted in order to determine, from the evaluated algorithms, which one better balances the aforementioned metrics. For this, two weights have been assigned to the individual metrics, i.e., a weigh for the number of serviced users (*weightSU*) and a weight for the number of created VMs (*weightVMs*). Each pair of weight combinations (weightSU, weightVMs) represent a different scenario. In this paper we evaluate two pure HTC and HPC scenarios by assigning the weight combinations (1, 0) and (0, 1), and a mixed HTC/HPC scenario by assigning weights (0.5, 0.5) with the aim of balancing these two basic metrics.

In addition, similarly to the preliminary results reported in [36], we study how the number of serviced users and created VMs is affected when using an exponential back-off strategy to retry allocating failing VMs. Experiments were performed with job data obtained from a real-world PSE [21] based on 3D finite element study whereas our previous results [19,30,36] were computed from 2D finite element simulations. In computational terms, this problem led to much more computing intensive jobs. It is worth mentioning that we have deliberately included some of the explanations from [36], specially the description of our ACO scheduler, so as to make this paper self-contained.

The comparisons have been performed against alternative Cloud schedulers, namely a Random allocation algorithm and a Cloud scheduler based on Genetic Algorithms [1]. Results show that our ACO scheduler performs competitively with respect to the number of serviced users and allows for a fair assignment of VMs. In other words, our scheduler provides a good balance to the number serviced users, i.e., the number of Cloud users that the scheduler is able to successfully serve, and the created VMs. The common ground for comparison is an ideal scheduler that always achieves the best possible allocation of VMs to physical resources according to these metrics. Experiments were performed by using CloudSim [12], a Cloud simulator that is widely employed for assessing Cloud schedulers.

The rest of the paper is structured as follows. Section 2 gives some background necessary to understand the concepts underpinning our scheduler. Then, Section 3 presents the scheduler. Section 4 reports the experimental evaluation. Then, Section 5 surveys relevant related works. Lastly, Section 6 concludes the paper and delineates future research opportunities.

2. Background

Cloud Computing [11] is a computing paradigm that has been recently incepted in the academic community [4]. Within a Cloud, *services* that represent computing resources, platforms or applications are provided across (sometimes geographically) dispersed organizations. Moreover, a Cloud provides resources in a highly dynamic and scalable way and offers to end-users a variety of services covering the entire computing stack. Particularly, within IaaS Clouds, slices of computational power in networked hosts are offered with the intent of reducing the owning and operating costs of having such resources in situ. Besides, the spectrum of configuration options available to scientists, such as PSEs scientific users, through Cloud services is wide enough to cover any specific need from their research.

2.1. Cloud Computing basics

The growing popularity of Cloud Computing has led to several definitions, as previously indicated by Vaquero et al. [48]. Some of the definitions given by scientists in the area include:

- Buyya et al. [11] define Cloud Computing in terms of its utility to end users: "A Cloud is a market-oriented distributed computing system consisting of a collection of interconnected and virtualized computers that are dynamically provisioned and presented as one or more unified computing resource(s) based on service-level agreements established through negotiation between the service provider and consumers".
- On the other hand, Mell and Grance [32] define Cloud Computing as "a model for enabling ubiquitous, convenient, on demand network access to a shared pool of configurable computing resources (i.e. networks, servers, storage, applications and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This Cloud model is composed of five essential characteristics, three services models (Software/Platform/Infrastructure as a Service), and four deployment models, whereas the five characteristics are: on-demand self-service, broad network access, resource pooling, rapid elasticity, and measured services. The deployment models include private, community, public and hybrid Clouds".