Parallel Computing 48 (2015) 40–58

Contents lists available at ScienceDirect

Parallel Computing

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

Mapping of time-consuming multitask applications on a cloud system by multiobjective Differential Evolution

Ivanoe De Falco, Umberto Scafuri, Ernesto Tarantino*

Institute of High Performance Computing and Networking, National Research Council of Italy, Naples 80131, Italy

ARTICLE INFO

Article history:
Received 8 March 2013
Revised 30 March 2015
Accepted 5 April 2015
Available online 10 April 2015

Keywords:
Cloud computing
Mapping
Differential Evolution
Multiobjective problems

ABSTRACT

Cloud computing is on-demand provisioning of virtual resources aggregated together so that by specific contracts users can lease access to their combined power.

Here we hypothesize a new form of service contract by means of which users do not explicitly require resources, but simply supply information about their time-consuming multitask applications and specify their needs through some quality of service (QoS) parameters. The individuation of the virtual machines (VMs) onto which map and execute them is left to the cloud manager. Unfortunately the task/node mapping, already known as NP-hard for conventional parallel systems, becomes more challenging when application tasks must be run on VMs hosted on heterogeneous and shared cloud nodes, and when it must comply with QoS requests too. To support this new cloud service, a novel mapper tool, based on a multiobjective Differential Evolution algorithm, is proposed. Such a tool defines the mapping of the tasks on the VMs with the aim to exploit as much as possible the available cloud resources without penalizing the execution time of the submitted applications and, at the same time, to respect users' QoS requests.

To reveal the robustness of this evolutionary tool, an experimental analysis on artificial time-consuming parallel applications, modeled as task interaction graphs, has been effected.

1. Introduction

In the last years high interest has been kindled by the advent of cloud technology [1–3], widely publicized and commercially supported by important firms and projects as for instance IBM, Amazon, Microsoft, and so on [4–8]. Cloud computing is increasingly explored as an effective alternative (and addition) to supercomputers for some high performance computing (HPC) applications [9–11]. In fact, the cloud allows benefits in terms of elasticity, maintenance costs, economics of scale and virtualization flexibility. Furthermore, many studies have been effected to find the nature of the HPC applications suitable to be executed on cloud platforms [12,13].

At the moment most of the cloud systems offers on-demand virtualized services ranging from the hardware to the application level [1]. Generally, these services are classified into three main service delivery models: infrastructure as a service (*IaaS*), platform as a service (*PaaS*), and software as a service (*SaaS*). *IaaS* refers to the practice of delivering on demand IT infrastructure as a commodity to customers. *PaaS* provides a development platform in which customers can create and execute their own applications. *SaaS* endows the user with an integrated service comprising hardware, development platforms, and applications.

* Corresponding author. Tel.: +39 081 6139525; fax: +39 081 6139531.
E-mail address: ernesto.tarantino@na.icar.cnr.it (E. Tarantino).

http://dx.doi.org/10.1016/j.parco.2015.04.001
0167-8191/

Typically, a cloud service provider signs contracts with his customers in the form of service-level agreements (SLAs), which can concern many aspects of a cloud computing service. The contract defines the agreed-upon-service fees for the total virtual resources negotiated by the client as well as the associated service credit if the provider fails to deliver the level of service.

Cloud systems are usually endowed with a very high aggregated computational power, so they could represent a viable solution to execute in parallel time-consuming multitask applications. This is why, although cloud systems currently seem unfit to efficiently solve compute-intensive parallel applications [9,14,15], many efforts are being dedicated by manufacturers and scientists to endow them with new functionalities in order to execute in reasonable times small- and medium-sized HPC applications [13,16,17]. In this way customers can have at their disposal virtual instances of all the needed resources without having any knowledge about the numbers, the characteristics, and the location of the cloud physical resources used to provide the requested services. Thus, in the absence of their own resources, developers of HPC parallel applications can negotiate their leasing from a cloud manager by a canonical *IaaS* or a *PaaS* SLA. Both these contract forms suppose that the customers, on the basis of their application requirements, first must individuate and bargain over the virtual machines (VMs) needed and then they have to establish the task/VM mapping.

In this paper a new form of a PaaS contract is hypothesized, by means of which customers only have to submit the information about their time-consuming multitask MPI application and to indicate quality of service (QoS) parameters they are mostly interested in. In our view, the values of these parameters cannot be negotiated. The cloud management software is instead in charge of detecting the available VMs onto which map and execute in parallel the application tasks, by suitably deducting the resources already used for other instanced VMs and services. This is accomplished by endowing the software with a novel mapping tool that does not face the classical mapping problem of parallel applications on multicomputers or on computational grids. Rather, such a tool, on the basis of the residual cloud resources, of the characteristics of the submitted applications and of users' QoS requests, is able to individuate the VM allocation on the cloud physical nodes that supports at best the application execution.

It is quite simple to design a web interface to support the application submission phase. Unfortunately the finding of an efficient task/VM mapping becomes even more challenging when it also has to deal with multiple conflicting optimization objectives [18], such as performance and QoS under limited budget constraints [19]. Not only is this an NP-complete problem [20,21], it is also not-approximable, i.e., it cannot be approximated in polynomial time with arbitrarily good precision by deterministic algorithms [22]. As shown in [23,24], given its NP-complete nature, the non-deterministic metaheuristic algorithms are the most appropriate to attain approximate solutions that meet the requirements in a reasonable time [25,26].

Here we propose a multiobjective version of Differential Evolution (DE) [27,28], relying on the Pareto method [29], to provide a set of mapping solutions (Pareto front), each with its different balance between use of resources and QoS constraints. More precisely, when a user submits a multitask application, the cloud manager determines the set of the VMs that can be instanced in parallel on the currently available physical resources. Successively, the manager executes the proposed evolutionary mapping tool to find the task/VM mapping solutions which allow exploiting the cloud resources that meet at best the needs formulated by QoS parameters.

Differently from other approaches [23,30], our tool is used to tackle the allocation of communicating tasks of time-consuming parallel applications modeled as task interaction graphs (TIGs) [31,32].

Within this paper, according to the *minimax* model [33,34], for each mapping solution, the cost incurred by each VM, i.e., the amount of time spent for the computation and the communication of all the tasks mapped on it, is estimated, and the maximum of these costs is to be minimized. This optimization criterion is chosen because, even in the case with the highest degree of overlapping, i.e., if no task waits for communications, the time required to complete the execution of a parallel application is at least equal to the greatest amount of time. Such a view leads toward the discovery of solutions which do not use the most powerful available VM if, due to task overlapping, its use does not contribute to a reduction in the above greatest amount of time. In such a way, only the minimal cloud resources needed for the task requirements are used, and this permits the cloud manager to more fruitfully exploit powerful resources for further applications. Therefore, each user can be charged for the amount of time she/he has effectively used the resources on the basis of the pricing model established by the cloud administrator.

TIGs are more flexible to describe different program structures than direct acyclic graphs (DAGs) [35,36], therefore our tool can also deal with cases for which other mapping algorithms, like Min–min [37,38], Max–min [39], and XSufferage [40], are not suitable [41]. Other evolutionary approaches map the application on locally distributed resources only [42]. Our tool, instead, can choose from even geographically-distributed VMs which, on the basis of their features, turn out to be the fittest for the application tasks to be allocated.

To evaluate the effectiveness of our evolutionary tool, the mapping of artificial applications on a cloud infrastructure at different workload operating conditions has been performed.

Paper structure is as follows: Section 2 reports on the related research; Section 3 presents the working environment; Section 4 summarizes the evolutionary technique investigated, while Section 5 explains our multiobjective mapper. In Section 6 the test problems experienced are reported and the results attained are commented. Finally in Section 7 conclusions are given.

2. Related research

The selection of the cloud resources which, on the basis of physical characteristics (computational power, frequency, memory, bandwidth, ...) and load, better support the services as they are negotiated by the customers is nearly always a problem of considerable difficulty. Unfortunately, even if the primary objective for *IaaS*, *PaaS*, and *SaaS* models remains to map as well as