Future Generation Computer Systems 29 (2013) 1901–1908

Contents lists available at ScienceDirect

Future Generation Computer Systems

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

An approximate ϵ-constraint method for a multi-objective job scheduling in the cloud

L. Grandinetti[a,*], O. Pisacane[b], M. Sheikhalishahi[a]

[a] Dipartimento di Elettronica, Informatica e Sistemistica, Università della Calabria, Via P. Bucci, 41C, Arcavacata di Rende (CS), Italy

[b] Dipartimento di Ingegneria dell'Informazione, Università Politecnica delle Marche, Via B. Bianche, 12, Ancona (AN), Italy

HIGHLIGHTS

- We formulate the multi-objective off-line job scheduling problem in the cloud.
- We optimize the makespan, the total average waiting time and the used hosts.
- We generate a set of test instances suitable for the problem.
- We define an approximate ϵ-constraint method.
- We compare the proposed solution approach with the weighted sum method.

ARTICLE INFO

Article history:
Received 13 July 2012
Received in revised form
26 March 2013
Accepted 8 April 2013
Available online 9 May 2013

Keywords:
Jobs scheduling
Operations management
Cloud computing
ϵ-constraint method
Multi-objective optimization

ABSTRACT

Cloud computing is a hybrid model that provides both hardware and software resources through computer networks. Data services (hardware) together with their functionalities (software) are hosted on web servers rather than on single computers connected by networks. Through a device (e.g., either a computer or a smartphone), a browser and an Internet connection, each user accesses a cloud platform and asks for specific services. For example, a user can ask for executing some applications (jobs) on the machines (hosts) of a cloud infrastructure. Therefore, it becomes significant to provide optimized job scheduling approaches suitable to balance the workload distribution among hosts of the platform.

In this paper, a multi-objective mathematical formulation of the job scheduling problem in a homogeneous cloud computing platform is proposed in order to optimize the total average waiting time of the jobs, the average waiting time of the jobs in the longest working schedule (such as the makespan) and the required number of hosts. The proposed approach is based on an approximate ϵ-constraint method, tested on a set of instances and compared with the weighted sum (WS) method.

The computational results highlight that our approach outperforms the WS method in terms of a number of non-dominated solutions.

1. Introduction

Cloud computing is a revolutionary paradigm suitable to change the way of accessing both hardware and software in order to produce, price, provide and deliver services and computational resources to users. Users can run their applications (*jobs*) without paying for software licenses, using well equipped machines (*hosts*) and high performance computational resources.

This paper addresses a multi-objective job scheduling problem in a homogeneous cloud infrastructure considering the minimization of the total average waiting time of the jobs, of the total waiting time of the jobs belonging to the longest working schedule (*makespan*) and the number of used hosts. It takes into account an *off-line* job scheduling scenario and, therefore, the number of jobs to run and their resource requirements are known a-priori. The main contributions are as follows:

- a multi-objective formulation of the off-line job scheduling problem in a homogeneous cloud computing platform;
- an approximate ϵ-constraint method for solving the problem;
- a detailed experimental analysis for evaluating the quality of the proposed approach.

With reference to the last contribution, first we implement an instance generator in order to determine a set of problems considered during the experimental phase. Then, we implement an alternative solution approach based on the *weighted sum* (WS) method. Finally, we compare the two approaches on the set of generated instances.

* Corresponding author. Tel.: +39 3351244747.
E-mail addresses: lugran@unical.it (L. Grandinetti), pisacane@dii.univpm.it (O. Pisacane), alishahi@unical.it (M. Sheikhalishahi).

0167-739X/$ – see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.future.2013.04.023

This paper is organized as follows: Section 2 reviews some significant literary contributions, Section 3 provides a high level description of the problem, Section 4 describes the multi-objective mathematical formulation of the problem. Sections 4.1–4.3 detail the two solution approaches taken into account, while Section 5 describes the generated scenarios and discusses the computational results. Finally, Section 6 concludes the work and suggests some future developments.

2. Related work

This section aims at describing some significant literary contributions with reference to the job scheduling problem. Therefore, it is organized as follows: firstly, some of the more significant job scheduling algorithms in computing systems are described; secondly, several multi-objective job scheduling methods are analyzed and finally, some literary contributions based on the ϵ-constraint method are summarized.

In the job scheduling problems in computer systems, the number of machines can be assumed finite and fixed. However, this assumption does not hold for the number of jobs. The scheduling problems in computer systems also differ from system to system (e.g., manufacturing and project planning). For example, the jobs could be dynamic (i.e., their arrival time is not known a-priori as well as their characteristics and duration). Moreover, in computer systems, the aim of these problems is to provide a better resource utilization satisfying the job requirements. A very popular research topic is the *parallel job scheduling*. There are many different ways for scheduling parallel jobs and threads that make them up [1]. However, only a few mechanisms are used in practice and studied in detail. Two approaches that have dominated over the last decade are the *Backfilling* and the *Gang Scheduling*. In particular, the *Backfilling*, introduced in [2], aims at balancing the resource utilization and at maintaining a *First Come First Served* (FCFS) order. It also allows small jobs to move ahead and run on processors that would otherwise remain idle. However, this is subject to some restrictions so that the situations in which the FCFS order is completely violated and some jobs are never run (i.e., *starvation* phenomenon) are avoided. In particular, a reservation for some future times is usually given to jobs that need to wait and its use was included in several early *Batch Schedulers* [3].

In [4], a decentralized dynamic scheduling approach (Community Aware Scheduling Algorithm—CASA) is proposed. CASA consists of a two-phase scheduling solution approach and of some heuristics to achieve optimized performances in the grid/cloud platform. The authors show that it yields a 30%–61% better average job slowdown if compared to the centralized scheduling scheme based on the BestFit meta-scheduling policy. It also yields a 68%–86% shorter average job waiting time if compared to a decentralized scheduling approach without requiring detailed real-time processing information from nodes.

In [5], a hierarchical framework and a job scheduling algorithm (*Hierarchical Load Balanced Algorithm*—(HLBA)) are proposed with reference to a grid platform. The system load is used to determine a balance threshold. The scheduler dynamically adapts the balance threshold according to the system load changes. The aim of the proposed scheduling algorithm is to balance the system load and minimize the *makespan* of jobs. In [6], the authors analyze and investigate the effectiveness of rescheduling using cloud resources in order to increase the job completion reliability. The jobs are scheduled using grid resources and then, cloud resources are used only for rescheduling to cope with a delay in job completion. The computational results demonstrate that the proposed rescheduling guarantees a delay reduction in job completion.

In [7], a credit based scheduling is used to evaluate the entire group of tasks in the job queue and to find their minimal completion time. In [8], a mathematical model for distributing workload among a minimum number of servers is proposed as a set partitioning formulation and two solution approaches are described. In the former, a set of candidate blocks are generated and then composed together for having schedules through an integer programming problem. Instead, in the latter, the set partitioning problem is solved by performing a column generation technique. After performing a test phase, the authors conclude that the second method outperforms the first one.

Moreover, due to the high dynamism of cloud environments that leads to a time-varying resource utilization, cloud providers can potentially accommodate secondary jobs with the remaining resource. In [9], the problem of secondary job scheduling with deadlines under time-varying resource capacity is taken into consideration. However, the scheduling scheme used on many distributed memory parallel supercomputers is *variable partitioning*. In this context, each job receives a machine partition with its desired number of processors [1]. Such partitions are allocated in an FCFS manner to incoming jobs.

While the job scheduling problem in distributed computing has attracted a lot of interest, less attention has been given to the multi-criteria version. In fact, there are a few works to address this special issue. In [10], some novel taxonomies of the multi-criteria grid workflow scheduling problem are proposed, mainly considering workflow, resource and task model, scheduling criteria and process. In [11], a multi-cost scheme of polynomial complexity is proposed in order to perform reservations and select computational resources to execute tasks. This scheme is also used to determine the path to route input data. The authors also describe some multi-cost algorithms with the aim of performing more advance reservations and finding the starting times for data transmission and tasks' execution. In [12], a modular broker architecture is proposed and described. It works with different scheduling strategies in order to optimally deploy virtual services across multiple clouds. These scheduling schemes are mainly based on different criteria to be optimized (either cost or performance optimization). Some user constraints to be taken into consideration (budget, performance, instance types, placement, reallocation or load balancing constraints) together with certain environmental conditions (static vs. dynamic conditions, instance prices, instance types, service workload, etc.). In [13], a particle swarm optimization based heuristic is designed and proposed in order to schedule applications on cloud resources, taking into account both computation cost and data transmission cost.

Nonetheless, one important criticism to be addressed for multi-objective models mainly concerns the definition of an efficient solution approach. In the literature, several alternative solution approaches for multi-objective optimization problems have been proposed. In particular,

- the *weighted global criterion method* in which the objectives are jointly optimized using a weighted function (usually a weighted exponential sum);
- the *weighted sum method* in which the objectives are summed in one function by introducing appropriate weights;
- the *lexicographic method* in which the objectives are arranged in order of importance and relevance.

For an exhaustive study about the multi-objective optimization methods, the reader may refer to [14]. However, an innovative solution approach, the ϵ-constraint method, has recently been introduced in [15]. In [16], a combined procedure of a previously developed single-objective optimization approach together with the ϵ-constraint method is proposed in order to provide an approximation of the Pareto front in multi-objective optimization. In [15], an exact ϵ-constraint method for the bi-objective combinatorial optimization problems with integer objective values is described. The authors also show how the Pareto front can be efficiently