Future Generation Computer Systems 71 (2017) 57–72

Contents lists available at ScienceDirect

Future Generation Computer Systems

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

A delay-based dynamic scheduling algorithm for bag-of-task workflows with stochastic task execution times in clouds

Zhicheng Cai [a,d,e,*], Xiaoping Li [b], Rubén Ruiz [c], Qianmu Li [a]

[a] School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, China
[b] School of Computer Science and Engineering, Southeast University, Nanjing, China
[c] Instituto Tecnológico de Informática, Acc. B. Universitat Politècnica de València, València, Spain
[d] Key Laboratory of Image and Video Understanding for Social Safety, Nanjing, China
[e] Key Laboratory of Computer Network and Information Integration (Southeast University), Ministry of Education, China

HIGHLIGHTS

- Minimizing the cloud resource renting cost of bag-of-tasks workflows.
- A bag-based delay triggering strategy is proposed to fully use the bag structure.
- Using expectation-and-variance of execution times to estimate practical times.
- A single-type based greedy method is developed for each ready BoT.

ARTICLE INFO

Article history:
Received 17 August 2016
Received in revised form
12 December 2016
Accepted 16 January 2017
Available online 23 January 2017

Keywords:
Cloud computing
Scheduling
Workflow
Bag of tasks
Stochastic

ABSTRACT

Bag-of-Tasks (BoT) workflows are widespread in many big data analysis fields. However, there are very few cloud resource provisioning and scheduling algorithms tailored for BoT workflows. Furthermore, existing algorithms fail to consider the stochastic task execution times of BoT workflows which leads to deadline violations and increased resource renting costs. In this paper, we propose a dynamic cloud resource provisioning and scheduling algorithm which aims to fulfill the workflow deadline by using the sum of task execution time expectation and standard deviation to estimate real task execution times. A bag-based delay scheduling strategy and a single-type based virtual machine interval renting method are presented to decrease the resource renting cost. The proposed algorithm is evaluated using a cloud simulator ElasticSim which is extended from CloudSim. The results show that the dynamic algorithm decreases the resource renting cost while guaranteeing the workflow deadline compared to the existing algorithms.

1. Introduction

A Bag-of-Tasks (BoT) consists of many independent tasks which can be processed in parallel [1]. BoTs are commonly found in many fields such as image processing, parameter sweeping and data mining. Applications of these fields usually consist of several (sequential, parallel or hybrid of sequential and parallel) steps rather than a single step and each step processes a BoT [2]. Data files are transferred among these BoTs which form a BoT workflow that implements complex business logic [3]. Fig. 1 shows an example of BoT workflows for an image defogging application for two image files [4]. The application consists of eight steps: image file reading, image partition of each image file, atmospheric absorption rate computing for data trunks, atmospheric scattering rate computing for data chunks, defogging model construction, image defogging for data chunks, image reconstruction and image file outputting. Each step usually processes many independent data chunks which form a BoT and the workflow is composed of eight BoTs. When the number of processed image files becomes larger, the number of tasks in each BoT becomes larger too.

Running BoT workflows requires a large amount of computing resources, such as grid clusters, supercomputers and private clusters. Nowadays cloud computing has become an available platform for many applications which provides resources on

* Corresponding author at: School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, China.
E-mail addresses: caizhicheng@njust.edu.cn (Z. Cai), xpli@seu.edu.cn (X. Li), rruiz@eio.upv.es (R. Ruiz), qianmu@njust.edu.cn (Q. Li).

http://dx.doi.org/10.1016/j.future.2017.01.020
0167-739X/

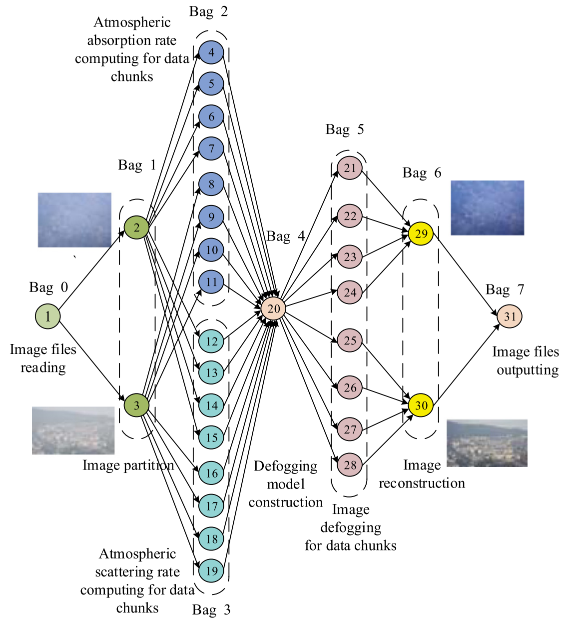

Fig. 1. An example of Bag-of-Task based workflow applications for image defogging.

demand to improve application performance and to reduce the costs by renting only the resources required. When BoT workflow applications are migrated to public clouds, resource provisioning and scheduling algorithms are needed. They decide the time, the type of resources and the number of resources to rent in order to reduce the renting cost by as much as possible. In this paper, we aim to minimize the cloud resource renting cost of a BoT workflow with deadline constraints in which task execution times are stochastic. The problem of optimal scheduling tasks with precedence relationships, in general cases, is NP-hard [5]. For example, the problem of minimizing the cost of workflows under the workflow deadline constraint considered by Möhring et al. [6], and Demeulemeester et al. [7] can be modeled as the Discrete Time-Cost Tradeoff Problem (DTCTP), which has been proved to be NP-hard [8]. In this paper, the problem of minimizing the cost of a BoT workflow under the workflow deadline constraint with stochastic task execution times and interval-based cloud resource pricing models is much more complex than DTCTP. However, most existing elasticity mechanisms provided by commercial public clouds are designed for web-server based applications and few of them focus on computation-intensive workflow applications [9].

Unlike a single BoT or multiple independent BoTs, there are complex dependencies among different BoTs in the same BoT workflow. In the academic work, most existing scheduling algorithms of BoT applications are designed for a single BoT or multiple independent BoTs [10–12]. These methods are not suitable for BoT workflows with many connected and constrained BoTs, for example the workflow shown in Fig. 1.

Most existing workflow scheduling algorithms are designed according to deterministic task execution times by assuming that task execution times can be predicted accurately [3,13–18]. However, practical task execution times are stochastic and have different probability distributions because of Virtual Machine (VM) performance uncertainty and complicated task properties [19]. When algorithms based on deterministic task execution times are put into practice, algorithm performance degenerates because of task execution time uncertainty [20], e.g., deadlines are violated or additional resource rental costs are incurred. Two strategies are usually adopted to deal with task execution time uncertainty [21]: (i) Static algorithms which tolerate to some extent task execution time uncertainty [22–24] and (ii) Dynamic algorithms which schedule tasks according to real time execution states as much as possible [20]. In these methods, task execution times with different probability distributions are usually fixed to known values. Maximum task execution times are usually adopted. However, these usually overestimates the practical task execution times which leads to resource over provisioning and additional costs. The sum of task execution time expected value and standard deviation is used by Tang et al. [24] for task scheduling with stochastic task execution times on private clusters and performs well.

Most existing static workflow scheduling methods considering task execution time uncertainty are designed to minimize workflow makespans on clusters with fixed capacities which are not suitable for the considered cloud resource rental cost minimization problem. In this paper, cloud resources are rented and released dynamically in intervals during the workflow runtime (called resource runtime auto-scaling). Scheduled tasks are more likely to exceed the planned resource interval horizons which incurs an additional resource renting cost. In other words, resource runtime auto-scaling makes workflow scheduling with task execution time uncertainty more complex. Therefore, developing static algorithms which are robust for task execution time uncertainty for BoT workflows is much more complicated.

Dynamic algorithms are widely used to handle task execution time uncertainty. However, the performance of dynamic algorithms is usually worse than that of static algorithms because of myopic optimization and a lack of complete information [20]. Therefore, it is crucial to determine when to make resource renting decisions for a dynamic algorithm. For the dynamic algorithms developed by Malawski et al. [20], VM renting decisions are made whenever the utilization is above or below given thresholds and it has been proved that making dynamic algorithms aware of workflow structures is beneficial for improving performance. However, the algorithms proposed by Malawski et al. [20] are tailored for maximizing the number of completed workflows rather than minimizing the rental cost of a single BoT workflow.

In this paper, a delay-based dynamic (online) algorithm is proposed to deal with stochastic task execution times. The main contributions are the following:

(1) A bag-based delay triggering strategy for VM renting processes is proposed to use the bag structure to improve the performance of the dynamic algorithm. Tasks are delayed and not scheduled until the ratio of the number of ready tasks to the number of total tasks in the BoT is larger than a given delay threshold, i.e., virtual machine renting decisions are only made whenever a batch of tasks in a BoT are ready in order to decrease the resource rental cost.
(2) An expectation-and-variance based VM selection method is proposed to handle the task execution time uncertainty. The sum of task execution time expected value and standard deviation is adopted to estimate the practical task execution times on VMs properly, which is beneficial for improving the utilization of rented VM intervals.
(3) A single-type based greedy method for VM renting of each ready BoT is developed to improve the effectiveness and efficiency simultaneously. Resource provisioning for each BoT under deadline constraints at each step of the dynamic algorithm is NP-hard. Since tasks of the same BoT have the same function they have the same virtual machine performance requirements. The type and the amount of required VMs are determined based on the assumption that VMs with the same type are provisioned.