Knowledge-Based Systems 80 (2015) 153–162

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

Resource-efficient workflow scheduling in clouds

Young Choon Lee [a,*], Hyuck Han [b], Albert Y. Zomaya [c], Mazin Yousif [d]

[a] Department of Computing, Macquarie University, Australia
[b] Department of Computer Science, Dongduk Women's University, Republic of Korea
[c] School of Information Technologies, The University of Sydney, Australia
[d] T-Systems International, USA

ARTICLE INFO

Article history:
Received 31 October 2014
Received in revised form 11 February 2015
Accepted 11 February 2015
Available online 18 February 2015

Keywords:
Cloud computing
Scientific workflows
Resource efficiency
Resource management
Workflow scheduling

ABSTRACT

Workflow applications in science and engineering have steadily increased in variety and scale. Coinciding with this increase has been the relentless effort to improve the performance of these applications through exploiting the abundance of resources in hyper-scale clouds and with little attention to resources efficiency. The inefficient use of resources when executing scientific workflows results from both the excessive amount of resources provisioned and the wastage from unused resources among task runs. In this paper, we address the problem of resource-efficient workflow scheduling. To this end, we present the Maximum Effective Reduction (MER) algorithm, a resource efficiency solution that optimizes the resource usage of a workflow schedule generated by any particular scheduling algorithm. MER trades the minimal makespan increase for the maximal resource usage reduction by consolidating tasks with the exploitation of resource inefficiency in the original workflow schedule. The main novelty of MER lies in its identification of "near-optimal" trade-off point between makespan increase and resource usage reduction. Finding such a point is of great practical importance and can lead to: (1) improvements in resource utilization, (2) reductions in resource provisioning, and (3) savings in energy consumption. Another significant contribution of this work is MER's broad applicability. In essence, MER can be applied to any environments that deal with the execution of (scientific) workflows of many precedence-constrained tasks although MER best suits for the IaaS cloud model. Based on results obtained from our extensive simulations using scientific workflow traces, we demonstrate MER is capable of reducing the amount of actual resources used by 54% with an average makespan increase of less than 10%. The efficacy of MER is further verified by results (from a comprehensive set of experiments with varying makespan delay limits) that show the resource usage reduction, makespan increase and the trade-off between them for various workflow applications.

1. Introduction

As resouce capacity in clusters and more recently clouds becomes increasingly abundant with the prevalence of large-scale multi-core systems and advances in virtualization technologies, such resource abundance has been "relentlessly" exploited particularly to improve applications performance. Scientific workflows (such as Montage [1], CyberShake [2], LIGO [3], Epigenomics [4] and SIPHT [5]) in particular can take great advantage of abundant resources as they are mostly resource-intensive with a good degree of scalability. However, the exploitation of resource abundance is a double-edged sword in that the performance improvement from such exploitation is often achieved at the expense of resource efficiency.

The inefficient use of resources when executing scientific workflows results from both the excessive amount of resources provisioned and the wastage from unused resources (Fig. 1), including idle time[1] among task runs. The optimization of resource efficiency is of great practical importance considering its numerous benefits in the economic and environmental sustainability of large-scale computer systems like corporate data centers and clouds.

While previous work on workflow scheduling has focused on increasing the performance (makespan) with a limited amount of resources (resource scarcity), the advent of multi-core processors and cloud computing (resource abundance) has brought much

* Corresponding author.
E-mail addresses: young.lee@mq.edu.au (Y.C. Lee), hhyuck96@dongduk.ac.kr (H. Han), albert.zomaya@sydney.edu.au (A.Y. Zomaya), myousif100@gmail.com (M. Yousif).

[1] Non-negligible amount of power is drawn while CPU is idling (e.g., 50% or more of peak power in C1 state) [6].

http://dx.doi.org/10.1016/j.knosys.2015.02.012
0950-7051/

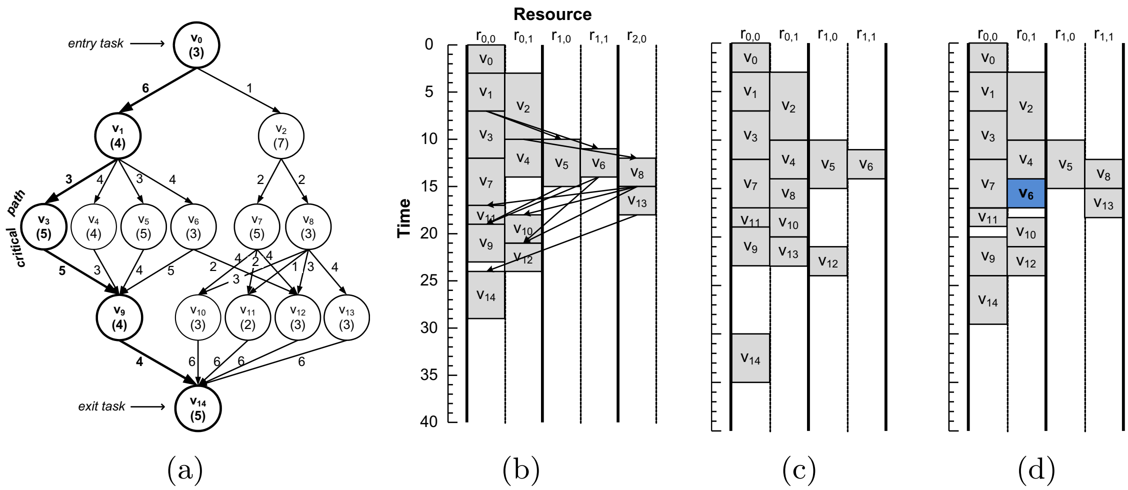

Fig. 1. Workflow scheduling example in dual-core compute resources. (a) A sample workflow. (b) The schedule generated using CPF in our previous work, [9]. (c) An example schedule with a pre-determined resource limit of 2. (d) A compacted schedule preserving the makespan of the original schedule, (the schedule in (a) in this example) with the use of the primitive solution in [9]. Computation costs and communication costs are shown in parentheses of vertices and long edges, respectively. Inter-task communications are only present (arrows in b) when they are scheduled on different resources as on-chip communication is negligible, i.e., 0.

attention to resource efficiency.[2] Existing workflow scheduling algorithms may be able to adapt to deal with resource abundance by limiting the number of resources to be used (resource limit) at the time of scheduling. However, it is only a partial and ad hoc solution; for example, the schedule in Fig. 1c with a resource limit of 2 shows that makespan increases by 21% compared to the schedule in Fig. 1b in return for the use of one less resource. Moreover, the efficacy of such a solution varies for different applications, and even with executions of a particular application with different inputs (e.g., data and/or parameter values). Dynamic resource provisioning with public clouds as in [7,8] might be an alternative; however, the poor resource utilization—sourced from uneven widths of different levels in a workflow (see Fig. 1)—within the hour still remains. Since it is very difficult, if not impossible, to find the optimal resource amount for scheduling a given workflow application, and since current workflow scheduling algorithms perform quite well in terms of makespan, the post-processing of output workflow schedules may be a practical approach to optimizing resource usage.

In this paper, we consider the problem of efficient use of resource abundance for running large-scale scientific workflows. To this end, we develop a new algorithm (*Maximum Effective Reduction* or MER) to effectively trade makespan increase for resource usage reduction. (*Maximum Effective Reduction* or MER), a workflow schedule optimization algorithm that minimizes the resource usage of a workflow schedule generated by any particular scheduling algorithm. MER is a post-optimization algorithm that takes an existing workflow schedule and consolidates tasks in the schedule into a smaller amount of resources compared to that used for the input schedule. The algorithm aims to find a minimal makespan increase that reduces the resource usage the most; this reduction is defined as effective reduction (ER) in this paper. ER is used to measure resource efficiency based primarily on the difference between the resource usage reduction and makespan increase in a resultant schedule as compared to the input schedule.

In our previous work [9], we have shown the potential that the resource usage of workflow schedule can be reduced by consolidating tasks exploiting idle/inefficiency slots; however, the degree of such reduction is mostly limited due to the preservation of makespan (Fig. 1d). MER significantly extends the primitive solution in [9] by allowing makespan increase/delay to maximize resource usage reduction. In particular, MER incorporates a simple, yet effective heuristic to estimate the makespan increase given the tasks being considered for consolidation. The estimate is used for the degree of delay allowed in makespan (makespan delay limit or simply delay limit), and is based on the estimated ER. We have verified the efficacy of this heuristic by a comparison with the best empirical delay limits found in our experiments. A preliminary version of this work can be also found in [10].

Based on results obtained from our extensive simulations using scientific workflow traces, we demonstrate MER is capable of reducing the amount of actual resources used by 54% with an average makespan increase of less than 10%. The efficacy of MER is further verified by results (from a comprehensive set of experiments with varying makespan delay limits) that show the resource usage reduction, makespan increase and the trade-off between them for various workflow applications.

The rest of this paper is organized as follows. Section 2 describes our workflow schedule optimization problem. In Section 3, we detail our solution algorithm. Section 4 demonstrates the efficacy of MER with results obtained from our extensive simulations. Section 5 provides a suvery on related work followed by our conclusion in Section 6.

2. The problem of workflow schedule optimization

In this section, we define the workflow schedule optimization problem describing workflow and system models.

2.1. Scientific workflows

A scientific workflow consists of a set of precedence-constrained tasks represented by a directed acyclic graph (DAG), $G = (V, E)$ comprising a set V of tasks, $V = \{v_0, v_1, \ldots, v_n\}$, and a set E of edges, each of which connects two tasks representing their precedence constraint or data dependency (Fig. 1a). A task is regarded as ready to run (or simply as a 'ready task'). The readiness of task v_i is determined by its predecessors (parent tasks), more specifically the one that completes the communication at the latest time. More formally, the earliest start and finish times of a task v_i are defined as:

$$EST(v_i) = \begin{cases} 0 & \text{if } v_i = v_{entry} \\ max_{v_p \in P_i}\{EST(v_p) + w_p + c_{p,i}\} & \text{otherwise,} \end{cases} \quad (1)$$

[2] Resource scarcity and abundance is relative to applications resource requirements.