Simulation Modelling Practice and Theory 70 (2017) 120–134

Contents lists available at ScienceDirect

Simulation Modelling Practice and Theory

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

Different aspects of workflow scheduling in large-scale distributed systems

Georgios L. Stavrinides[a,*], Francisco Rodrigo Duro[b], Helen D. Karatza[a], Javier Garcia Blas[b], Jesus Carretero[b]

[a] Department of Informatics, Aristotle University of Thessaloniki, Thessaloniki,54124, Greece
[b] Computer Architecture and Communication Area, University Carlos III, Avda. Universidad, 30,Madrid,28911, Spain

ARTICLE INFO

Article history:
Received 12 October 2016
Revised 26 October 2016
Accepted 27 October 2016

Keywords:
Workflow scheduling
Large-scale distributed systems
Ultrascale systems
Quality of Service
Data locality

ABSTRACT

As large-scale distributed systems gain momentum, the scheduling of workflow applications with multiple requirements in such computing platforms has become a crucial area of research. In this paper, we investigate the workflow scheduling problem in large-scale distributed systems, from the Quality of Service (QoS) and data locality perspectives. We present a scheduling approach, considering two models of synchronization for the tasks in a workflow application: (a) communication through the network and (b) communication through temporary files. Specifically, we investigate via simulation the performance of a heterogeneous distributed system, where multiple soft real-time workflow applications arrive dynamically. The applications are scheduled under various tardiness bounds, taking into account the communication cost in the first case study and the I/O cost and data locality in the second. The simulation results provide useful insights into the impact of tardiness bound and data locality on the system performance.

© 2016 Elsevier B.V. All rights reserved.

1. Introduction

The constant progress in computing and communication technologies has led to the emergence of *ultrascale computing*. Ultrascale computing systems are envisioned as large-scale complex systems, joining parallel and distributed computing resources that may be located at multiple sites. The applications in these environments usually concern complex problems and feature coarse-grained parallelism. That is, their component tasks do not require any communication with each other during processing, but only before or after their execution, forming a *workflow*. In order to determine the result of such complex applications, their individual tasks are scheduled and executed coordinately at various nodes of the system, according to their precedence constraints [1].

1.1. Motivation

The scheduling of workflow applications with multiple requirements in such computing platforms has become a crucial area of research. Workflow applications usually have *soft deadlines*, which may be missed by a bounded amount of time (i.e.

* Corresponding author.
E-mail addresses: gstavrin@csd.auth.gr (G.L. Stavrinides), frodrigo@arcos.inf.uc3m.es (F.R. Duro), karatza@csd.auth.gr (H.D. Karatza), fjblas@inf.uc3m.es (J.G. Blas), jcarrete@inf.uc3m.es (J. Carretero).

http://dx.doi.org/10.1016/j.simpat.2016.10.009
1569-190X/© 2016 Elsevier B.V. All rights reserved.

the applications are *real-time*). Thus, the *Quality of Service (QoS)* of the system often concerns parameters like the completion rate, the timeliness, the makespan and the tardiness of the applications.

Furthermore, workflow applications often require specific data in order to start execution, which may be available at different nodes of the system. Current systems from both high performance computing (HPC) and cloud domains do not efficiently support data-intensive workflows. On the one hand, typical HPC systems use monolithic parallel file systems for data sharing, such as GPFS [2] and Lustre [3]. On the other hand, in order to provide portability and scalable I/O, one of the alternatives for mass data storage in cloud platforms is the use of remote shared storage systems. Usually, the aim of this approach is to provide a unified interface and a scalable storage solution for cloud-based applications through storage services, such as Amazon S3.

Consequently, *QoS* and *data locality* are two important aspects of workflow scheduling in ultrascale systems, that should be taken into account in order to achieve good performance [4,5].

1.2. Contribution

In this paper, we investigate the workflow scheduling problem in large-scale distributed systems, from the QoS and data locality perspectives. We present a scheduling approach, considering two models of synchronization for the tasks in a workflow application: (a) communication through the network and (b) communication through temporary files. Specifically, we investigate via simulation the performance of a heterogeneous distributed system, where multiple soft real-time workflow applications arrive dynamically. The applications are scheduled under various tardiness bounds, taking into account the communication cost in the first case study and the I/O cost and data locality in the second.

The remainder of the paper is organized as follows: Section 2 gives an overview of related literature. Section 3 describes the QoS objectives during workflow scheduling in ultrascale systems and presents a relevant case study. Section 4 presents an alternative case study, from the data locality perspective, where data awareness is incorporated into the scheduling algorithm. Section 5 presents the performance evaluation of the system from the QoS and data locality perspectives and discusses the insights obtained by the simulation experiments in each case study. Finally, Section 6 summarizes and concludes the paper.

2. Related work

As the scheduling and execution of complex applications in large-scale distributed systems continues to gather significant attention from the research community, QoS, data locality and workflow-oriented storage systems are some of the most crucial topics.

2.1. Quality of Service

The scheduling problem of complex applications with various constraints and requirements has been studied extensively in the literature [6–13]. For workflow applications in particular, list scheduling heuristics are the simplest, most practical, easiest to implement and often outperform other scheduling approaches [14]. According to this method, all the component tasks of the applications are prioritized according to a particular common parameter (e.g. deadline, computational cost etc.) and then arranged in a list, ordered according to their priority. Subsequently, each task is allocated to the processor that minimizes a specific cost parameter, such as the estimated start time of the task.

Genez et al. in [15], propose the Integer Linear Program (ILP) policy, in an attempt to solve the problem of scheduling a workflow application in a Software as a Service (SaaS) or Platform as a Service (PaaS) cloud with two levels of service level agreements (SLAs). The first SLA level is between the end-user and the SaaS/ PaaS provider and concerns the deadline within which the user's workflow must be completed. The second SLA level is between the SaaS/ PaaS provider and multiple Infrastructure as a Service (IaaS) providers, concerning the characteristics and pricing of the virtual machines on which the user's workflow will be executed. The ultimate goal of the ILP strategy is to find a feasible mapping between the tasks of the workflow and the virtual machines from multiple IaaS providers, so that the total monetary cost is minimized and the makespan of the workflow is at most equal to the deadline required by the user. It is shown that the proposed policy can provide low-cost solutions, while meeting the deadline of the workflow. However, it only deals with the static scheduling of a single workflow, without utilizing any schedule gaps.

In modern large-scale distributed computing platforms, executing simultaneously multiple workflow applications of different users, sharing the same underlying resources, is an inevitable requirement. Multiple workflow applications have been used in service-oriented and cloud computing environments. For example, the Amazon Simple Workflow Service (SWF) can be used for the creation and deployment of multiple workflow applications in the Amazon EC2 cloud [16]. Recent research showed that utilizing gaps in the schedule of the compute nodes, formed by the inter-task dependencies and data communication costs, is a promising approach for efficient multiple workflow scheduling [17,18]. Towards this direction, Jiang et al. in [19] present the Path Clustering Heuristic with Distributed Gap Search (PCH-DGS), for the scheduling of multiple workflow applications in a heterogeneous cloud. According to their proposed strategy, the tasks of a workflow are first partitioned into groups, in an attempt to minimize the communication cost between them. Subsequently, each group of tasks is inserted into the first available time gap in a processor's schedule. In case the gap cannot accommodate all of the tasks of

Download English Version:

https://daneshyari.com/en/article/4962739

Download Persian Version:

https://daneshyari.com/article/4962739

Daneshyari.com