Procedia Computer Science

Procedia Computer Science 101, 2016, Pages 388 – 397

YSC 2016. 5th International Young Scientist Conference on Computational Science

Automatic workflow scheduling tuning for distributed processing systems

Alexander A. Visheratin, Mikhail Melnik and Denis Nasonov

ITMO University, Saint Petersburg, Russia

alexvish91@gmail.com, mihail.melnik.ifmo@gmail.com, denis.nasonov@gmail.com

Abstract

Modern scientific applications are composed of various methods, techniques and models to solve complicated problems. Such composite applications commonly are represented as workflows. Workflow scheduling is a well-known optimization problem, for which there is a great amount of solutions. Most of the algorithms contain parameters, which affect the result of a method. Thus, for the efficient scheduling it is important to tune parameters of the algorithms. Moreover, performance models, which are used for the estimation of obtained solutions, are crucial parts of workflow scheduling. In this work we present a combined approach for automatic parameters tuning and performance models construction in the background of the WMS lifecycle. Algorithms tuning is provided by hyper-heuristic genetic algorithm, whereas models construction is performed via symbolic regression methods. Developed algorithm was evaluated using CLAVIRE platform and is applicable for any distributed computing systems to optimize the execution of composite applications.

Keywords: genetic algorithm, workflow, hyper-heuristic, parameters tuning, performance model.

1 Introduction

Complex scientific applications nowadays play a very important role in the development of different scientific domains – physics, astronomy, biology, etc. – since they allow users to solve complicated problems by combining various methods and techniques in a single solution. Execution of the composite application (CA) usually involves a large number of smaller applications, which perform some specific tasks and communicate with other parts of the CA through the signals and data transfer. One of the most common ways for representing CA is a workflows concept – directed acyclic graph, where nodes stand for the computational tasks (parts of the CA) and the edges denote dependencies between them (mostly data connections).

For performing execution of such workflows computational capacity of one computer is obviously not enough. High performance computational systems like Grid clusters and cloud environments are used for these purposes. And when speaking about the workflow execution in the distributed systems, very important issue is scheduling of the workflow. Scheduling is a process of mapping workflow tasks onto the computational resources with regard to dependencies between them. It means that task

Peer-review under responsibility of organizing committee of the scientific committee of the
5th International Young Scientist Conference on Computational Science

© 2016 The Authors. Published by Elsevier B.V.

doi:10.1016/j.procs.2016.11.045

t_2, which is dependent of task t_1, cannot be scheduled to be executed before this dependency would be satisfied, e.g. signal or data received from t_1.

Workflow scheduling is known to be an NP-hard problem [1], and there are a lot of algorithms, which are designed to perform efficient workflow scheduling. Except of design of the algorithm itself, there are two important aspects, which can greatly influence the quality of the algorithm's solution. The first is the algorithm parameters. Many algorithms have specific parameters controlling different parts of the execution, like population size, mutation and crossover probabilities for the genetic algorithm. Pertinent parameters can not only increase the quality of the result, but also have positive effect on other characteristics of the algorithm, e.g. decrease convergence time, in case of metaheuristics. The second important aspect is a proper selection of performance models. For the workflow execution in the distributed system there is a plenty of factors to consider – dependence of the application completion time on the characteristics of the computational resource (CPU cores, memory), the network characteristics (bandwidth, latency), etc. The more of these factors we take into account when creating performance model for the application, the more precise our scheduling algorithm will be.

In this paper we present a combined approach, which aims to optimize both scheduling algorithms parameters and performance models used by the algorithms. We implement this approach as a procedure for an automatic tuning of scheduling algorithms and investigate its efficiency in different situations for the previously developed MHGH algorithm [2].

2 Related works

In [3] Trelea presents a very detailed analysis on the parameters optimization for the particle swarm optimization algorithm (PSO). In the paper different aspects of the PSO are described and the influence of the algorithm parameters on its convergence behavior are presented. Author makes experimental comparison of the algorithm parameters efficiency between the set of parameters recommended in [4] and the set selected with proposed tuning heuristic after a large number of experiments using five target functions for optimization. There are several important for us results in this paper. The first is an optimal set of parameters for the PSO algorithm, which we can use as a start point for our procedure. The second is the fact that author used heuristic algorithm for searching better parameters set – despite relatively large execution time, metaheuristic algorithms tend to generate better solutions than heuristics, that is why usage of the GA for these purposes may give better results. And the third is the author's conclusion that the optimal parameters' set strongly depends on the function being optimized, which makes it very reasonable to adjust algorithms parameters for every problem (in our case workflow) independently.

Authors of [5] propose a hyper-heuristic algorithm for tasks scheduling in the cloud environments, which aims to produce results better, than metaheuristics, while not increasing the computation time. The idea of having several metaheuristic algorithms and selection of the algorithm being used on the runtime is very interesting and the results clearly show that this approach generally is more efficient. In the paper Tsai et al. use a fixed set of algorithms, where their parameters do not change over time, which may lead to the potential drawback, because parameters giving good results for one problem may give worse results for the other, and their dynamic adjustment can have significant impact on the resulting solution.

Xhafa and Abraham [6] present a comprehensive survey on computational models and heuristic methods for Grid scheduling. They make a thorough look on the different aspects of the tasks execution and scheduling in the Grid environments, such as performance models, optimization criteria and types of scheduling. A wide range of metaheuristic algorithms classes are covered, from local search and population methods to hybrid heuristics and neural networks. Hyper-heuristics presented in the paper are close to [5] and this lack of the algorithms, which can adapt their parameters to the execution environment, gives a room for the solution proposed in our paper.

Download English Version:

https://daneshyari.com/en/article/4961333

Download Persian Version:

https://daneshyari.com/article/4961333

Daneshyari.com