Applied Soft Computing 63 (2018) 249–267

Contents lists available at ScienceDirect

Applied Soft Computing

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

A bi-objective load balancing model in a distributed simulation system using NSGA-II and MOPSO approaches

Shuxin Ding[a,b,c], Chen Chen[a,b,*], Bin Xin[a,b], Panos M. Pardalos[c]

[a] School of Automation, Beijing Institute of Technology, Beijing 100081, China
[b] State Key Laboratory of Intelligent Control and Decision of Complex System, Beijing 100081, China
[c] Center for Applied Optimization, Department of Industrial and Systems Engineering, University of Florida, Gainesville, FL 32611, USA

A R T I C L E I N F O

Article history:
Received 13 April 2017
Received in revised form 6 August 2017
Accepted 5 September 2017

Keywords:
Load balancing
High level architecture
Multi-objective optimization
Non-dominated sorting genetic algorithm with elitist strategy
Multi-objective particle swarm optimization

A B S T R A C T

High level architecture (HLA) is a software-architecture specification of a distributed simulation system which does not involve load balancing. As a result, problems of long simulation time and distortion caused by unequally distributed simulation tasks cannot be solved in the simulation process of HLA. In this paper, the simulation system based on the component level will be studied, and problems of load imbalances will be tackled. The goal is to find a set of Pareto optimal solutions to minimize the imbalance of the computation load and the total communication load with load limitation constraints in the model. To formulate this problem, a new integer programming model is presented. Both a non-dominated sorting genetic algorithm with elitist strategy (NSGA-II) and a multi-objective particle swarm optimization (MOPSO) are adopted to solve the problem. Different global best selection methods (crowding distance, adaptive grids and comprehensive ranking) and perturbation methods (rapidly decreasing and elitist learning strategy) for MOPSO are analyzed. Since the parameters of the algorithms have significant effects on their performance, the Taguchi method with a novel response value is utilized to tune the parameters of the proposed algorithms. Five performance metrics are used to evaluate the results of these algorithms. Based on the results, a networked control simulation platform will be used to test and verify the schemes. Numerical results show that the proposed MOPSO with ELS operator outperforms other proposed algorithms in solving the problem. In addition, the proposed strategies could solve problems of load imbalances in systems based on HLA.

1. Introduction

With the advantages of high utilization ratio, high scalability and parallelism, distributed systems can be run in parallel to handle problems that are computationally intensive. In addition, several difficult problems are derived from the distributed structure, such as the communication problems among the nodes, the mapping and assignment problems of system resources and computing nodes.

High level architecture (HLA) is widely used for distributed simulation development. Although HLA supports large-scale distributed simulation, the load balancing mechanism in the simulation process is not standardized [1]. When the simulation system runs under the condition of non-optimized allocation, load imbalance problems will occur due to the shortage of computing resources, which will seriously affect the simulation efficiency and confidence.

Load balancing is a problem related to the resource allocation problem, which is a fundamental combinatorial optimization problem. Many fields are concerned such as portfolio management [2], sensor network management [3], node deployment [4], etc. However, load balancing problem focuses on the equilibrium of load. In the distributed simulation system based on HLA, the load is divided into the computation load and the communication load. The main purpose of load balancing is to minimize the negative effects of the two kinds of loads, that is to say, try to even out the distribution of computation load among the nodes, and minimize the communication load among them. The results of load balancing are shown in the following two aspects: firstly, shortening the average response time of tasks effectively, that is to say, trying to make the runtime of the simulation process as short as possible, so that the system could run steadily and smoothly; secondly, maximizing the use of simulation resources of the whole system, and reducing the unnecessary waste of resources. However, there are conflicts between the dis-

* Corresponding author at: School of Automation, Beijing Institute of Technology, Beijing 100081, China.
E-mail addresses: shxding@bit.edu.cn (S. Ding), xiaofan@bit.edu.cn (C. Chen), brucebin@bit.edu.cn (B. Xin), pardalos@ufl.edu (P.M. Pardalos).

https://doi.org/10.1016/j.asoc.2017.09.012
1568-4946/© 2017 Published by Elsevier B.V.

tributions of these two kinds of loads, as a result, the load balancing problem is a multi-objective problem.

According to the load distribution mode, methods of load balancing are divided into static task allocation method and dynamic task migration method. Dynamic load balancing is concerned with the migration of the load, that is, the migration of the federates [5]. However, the migration operation will cause many problems, such as transforming the federates developed in traditional methods. Besides, the entire simulation process needs to be paused sometimes, and the majority operations of the migration process are complex, so that the federal implementation will be of low efficiency, which betrays the original intention of load balancing. In addition, the migration operation can't be realized unless operated by the third party software or components. The operation will increase the difficulty of its development and use [5–7]. The static load balancing method mainly studies the allocation algorithm of simulation tasks. It can be directly used without any additional software. Therefore, this paper uses the static load balancing method to optimize the simulation of the distribution simulation system based on HLA.

Static load balancing method is a kind of off-line optimization method, and its goal is to obtain an optimal allocation scheme, which can be applied to assigning tasks to different simulation nodes before running the simulation system. In the research, heuristic algorithms, such as genetic algorithm, simulated annealing algorithm, and hybrid algorithms, are commonly used to obtain the task allocation scheme. The optimum solution obtained can assign simulation tasks to different nodes.

Wu et al. [8] proposed a heuristic algorithm to schedule the simulation tasks. Martino et al. [9] used an improved genetic algorithm to realize static allocation of simulation tasks in the grid environment. According to the static scheduling problem of grid simulation, Wei et al. [10] proposed an entity scheduling strategy based on the interaction priority algorithm. This strategy can transform the multi-objective problem into a single objective optimization problem by using the weighted method, and divided the scheduling strategy into two periods. Zhao et al. [11] used an improved simulated annealing algorithm to optimize the mapping of the federates to the computing nodes. Pan et al. [12] focused on the static load balancing of the multi-level distributed real-time simulation system, and they took the real-time performance of the simulation tasks into consideration, and set up a heuristic function to realize the task map. Ban et al. [13] designed a sub-phase load balancing method based on the federal level of HLA simulation system. Yue et al. [14] studied the static load balancing and took the dynamic process into consideration at the same time. They took the total communication and computation load as well as the simulation time as two objectives. They used the Monte Carlo method and sequential optimization method to obtain the optimal allocation scheme.

According to the analysis above, the static load balancing research based on the HLA simulation system mostly focuses on the single objective optimization of the federate. However, the computation load and the communication load are two different kinds of loads, and there is no reference value to measure them. As a result, there is a lack of accurate basis for their weight, the common basis is subjective and unreasonable so that ideal task allocation schemes are extremely hard to obtain. On the basis of module design of the federates in the simulation system, this paper will consider a bi-objective model to describe the load balancing problem.

In order to solve the bi-objective load balancing model, multi-objective evolutionary algorithms (MOEAs) are usually adopted. In recent years, a non-dominated sorting genetic algorithm with elitist strategy (NSGA-II) [15], and a multi-objective particle swarm optimization (MOPSO) [16], etc. are widely used in many fields. Many of recent works on bi-objective or multi-objective optimization problems [17–22] are analyzed by comparisons of these algorithms. Salimi et al. [23] solves a task scheduling problem with load balancing by NSGA-II with fuzzy operators for computational grids. In [24], a bi-objective generalized assignment problem (GAP) with equilibrium function is solved using an integer enhanced version of NSGA-II. Dahmani et al. [25] use a discrete MOPSO to solve a bi-objective load balancing problem in aircraft cargo transportation. Total weight and the total priority of loaded cargo are maximized with load balancing constraints. In [26], a multi-objective load balancing system that avoids virtual machine (VM) migration and achieves system load balancing is developed and particle swarm optimization algorithm is applied. Alkayal et al. [27] uses a multi-objective particle swarm optimization algorithm with ranking strategy to solve the task scheduling problem in cloud computing. Cloud computing is a parallel and distributed system with virtual machines. Computational resources are allocated to multiple independent execution environments. Zhu et al. [28] use a multi-objective ant colony optimization algorithm based on load balance for virtual machine placement. The reviewed literature related to this study is summarized in Table 1.

Here are the main contributions of this paper. First, unlike many researches in which a single objective load balancing model is considered. In this paper, a novel bi-objective load balancing model with load limitation is presented. The first objective function aims to minimize the imbalances of the computation load and the second objective function, minimize the total communication load. Second, in this paper, several MOEAs based on NSGA-II [15] and MOPSO are adopted to solve the bi-objective load balancing model, and the idea of dynamic equilibrium will also be applied to these algorithm. We consider some improved strategies in MOPSO, they are global best selection methods (crowding distance [29], adaptive grids [16] and comprehensive ranking [30]) and perturbation methods (rapidly decreasing [16] and elitist learning strategy [31]). Crowding distance and adaptive grids are two popular global best selection methods. Besides, this paper also considers comprehensive ranking. Since the meta-heuristic algorithms are sensitive to the parameters, a Taguchi method is conducted to calibrate the parameters of the algorithms with a novel metric which uses a combination of convergence and diversity as the response value. Finally, several performance metrics are used to evaluate the efficiency of the proposed algorithms. The ideal allocation scheme will be verified on a networked control simulation platform.

The remainder of this paper is organized as follows: In Section 2, the problem is formulated. In Section 3, solution algorithms are presented. Simulation experiment results are provided in Section 4. In Section 5, the schemes verification is given. Conclusions and future works are presented in Section 6.

2. Problem formulation

In this paper, a bi-objective load balancing problem with computation and communication load limitation is considered. As stated before, this problem has been inspired from the load distribution problem in distributed simulation system, i.e. HLA. In the distributed simulation system based on HLA, simulation tasks can be divided into federate level and component level. The location of the federates remain unchanged during the simulation process. The load related to the simulation tasks is divided into computation load and communication load. Computation load refers to the system overhead of each simulation task (federate/component) that can be executed in parallel [12]. In this system, the computation load is mainly derived from the internal calculation of the simulation model and propulsion of the simulation process. Communication load refers to the amount of data exchange between nodes in the unit simulation period. HLA provides the distributed simulation