Applied Soft Computing 30 (2015) 501–513

Contents lists available at ScienceDirect

Applied Soft Computing

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

Extremal Optimization applied to load balancing in execution of distributed programs☆

Ivanoe De Falco[a], Eryk Laskowski[b,*], Richard Olejnik[c], Umberto Scafuri[a], Ernesto Tarantino[a], Marek Tudruj[b,d]

[a] Institute of High Performance Computing and Networking, CNR, Naples, Italy
[b] Institute of Computer Science, Polish Academy of Sciences, Warsaw, Poland
[c] Computer Science Laboratory, University of Science and Technology of Lille, France
[d] Polish-Japanese Institute of Information Technology, Warsaw, Poland

ARTICLE INFO

Article history:
Received 9 July 2014
Received in revised form 7 November 2014
Accepted 23 January 2015
Available online 12 February 2015

Keywords:
Distributed programs
Extremal optimization
Load balancing

ABSTRACT

The paper describes methods for using Extremal Optimization (EO) for processor load balancing during execution of distributed applications. A load balancing algorithm for clusters of multicore processors is presented and discussed. In this algorithm the EO approach is used to periodically detect the best tasks as candidates for migration and for a guided selection of the best computing nodes to receive the migrating tasks. To decrease the complexity of selection for migration, the embedded EO algorithm assumes a two-step stochastic selection during the solution improvement based on two separate fitness functions. The functions are based on specific models which estimate relations between the programs and the executive hardware. The proposed load balancing algorithm is assessed by experiments with simulated load balancing of distributed program graphs. The algorithm is compared against a greedy fully deterministic approach, a genetic algorithm and an EO-based algorithm with random placement of migrated tasks.

1. Introduction

Processor load balancing is a very relevant research subject in the domain of parallel and distributed systems. The importance of this subject does not require justification especially in multi-user systems where system resource availability for a particular user and computational loads of particular processors can change during application run time.

Extremal Optimization (EO) is a nature-inspired optimization technique which has moderate computational complexity and small memory requirements. Such features justify research on the use of EO in processor load balancing in distributed systems. The contribution of this paper is a dynamic load balancing method based on the use of the EO approach [1] supported by special program and runtime system behavior observation mechanisms.

The proposed load balancing algorithm is composed of iterative optimization phases which improve program task placement on processors to determine the possibly best balance of computational loads and to define periodic migration of tasks. EO is used in iterative load balancing phases which are executed in the background, in parallel with the application program. The EO algorithm discovers the candidate tasks for migration based on a special quality model including the computation and communication parameters of parallel tasks. The algorithm presented in this paper is an improved version of the EO-based load balancing algorithm introduced in [2]. In the former algorithm, the EO approach was based on the classic EO design, in which the selection of a new improved partial solution in the neighborhood of the solution being modified was fully random. The experiments performed for a big number of executive processors have shown that such a fully random approach was leading to a degradation of the quality of obtained results (parallel speedup) when compared to an equivalent fully deterministic algorithm. The comparison was carried out for the same set of program and system features including the load imbalance detection but with a deterministic selection of a processor to be the

☆ This paper is an extended, improved version of the paper "Improving Extremal Optimization in Load Balancing by Local Search" presented at EvoComNet2014 and published in: Applications of Evolutionary Computing, Proc. of 17th European Conference, EvoApplications 2014, Granada, Spain, April 23–25, 2014, LNCS 8602, pp. 51–62, Springer 2014.

* Corresponding author. Tel.: +48 223800517.
E-mail addresses: ivanoe.defalco@na.icar.cnr.it (I. De Falco), laskowsk@ipipan.waw.pl (E. Laskowski), richard.olejnik@lifl.fr (R. Olejnik), umberto.scafuri@na.icar.cnr.it (U. Scafuri), ernesto.tarantino@na.icar.cnr.it (E. Tarantino), tudruj@ipipan.waw.pl (M. Tudruj).

http://dx.doi.org/10.1016/j.asoc.2015.01.048

migration target. To alleviate this feature, we have modified the EO algorithm by replacing the fully random computing node selection with the stochastic selection in which the probability is guided by some knowledge of the problem. The support consisted in a formula which reported how a migrated task would fit to the given processor in terms of maintaining the global computational balance in the system, and in terms of influencing the communication charges of the processor. So, we have preserved the nature-inspired character of the solution improvement but in the way which speeds up the convergence of the algorithm. As a result we have obtained a correct behavior of the algorithm for the increasing cardinality of processor sets.

The algorithm is assessed by experiments with simulated load balancing of distributed program graphs. In particular, the experiments compare four algorithms: the proposed load balancing method including the EO with a guided stochastic selection of the improved solution, an EO with fully random selection of the improved solution, a genetic algorithm and an equivalent deterministic algorithm based on the similar theoretical foundations and "best fit" migration target selection. The comparison shows that the quality of load balancing with the guided EO is in most cases better than that of the fully random selection, the deterministic algorithm and the genetic algorithm.

The focus of the paper is primarily on studying the possibility of using a nature-inspired EO in the context of load balancing. In this respect the contribution of the paper is the way how to use EO for load balancing. The second aspect of the paper consists in developing foundations for the general load balancing methods. In this aspect, the proposed program and system metrics are based on particular heuristics which have originality features. The third aspect of the paper concerns how to combine the EO-based search for efficient processor load balance with the use of proposed metrics.

When we compare the proposed load balancing method to the current parallel computing environments with load balancing, like CHARM++ [3] or Zoltan [4], we notice that none of them includes EO as a load balancing algorithm component. So, the proposed approach is clearly original and enables using the EO advantages such as low computational complexity and limited use of memory space. The presented approach leverages some our earlier works on metrics for processor load balancing [5,6], and our experience in using EO to static task scheduling in distributed programs in Java [7,8].

The paper is organized as follows. In Section 2, the state of the art in load balancing based on nature-inspired algorithm is reported. In Section 3, the EO principles are shortly explained. Section 4 describes the theoretical foundations for the discussed algorithm, explains how the EO is applied to the dynamic processor load balancing during execution of distributed programs, and illustrates the principles of the deterministic load balancing reference algorithm. In Section 5, the experiments which assess the proposed algorithms are presented. Section 6 presents a statistical analysis of the results achieved by the investigated algorithms. The conclusions are exposed in the last section.

2. State of the art

A huge quantity of papers exist in literature dealing with dynamic load balancing in parallel and distributed systems, too many to be listed here. Good reviews and classifications of classic load balancing methods are presented in [9–13]. In this section we wish to point out that nature-inspired algorithms have been and are being used to fulfill this task, earlier with reference to parallel and distributed systems and more recently to grids and clouds.

Genetic algorithms (GAs) have been the first nature-inspired optimization methods to be used with reference to this issue. Therefore, we start this section by giving a few examples about their use for dynamic load scheduling.

Munetomo et al. [14] are among the first to present a genetic algorithm for stochastic environments and show its application to dynamic load balancing in distributed systems. The stochastic GA (StGA) has an evaluation mechanism for fitness values based on the reinforcement learning in order to adapt to stochastic environments. StGA is applied to the decision phase of task migration requests in dynamic load balancing.

In [15] Lee and Hwang propose a GA method for improved sender-initiated load balancing in distributed systems, and define a suitable fitness function. This algorithm decreases unnecessary request messages and increases acceptance rate. Compared with the conventional sender-initiated algorithms, the proposed algorithm performs better.

Zomaya and Teh [16] investigate how a GA can be employed to solve the dynamic load balancing problem. A dynamic load balancing algorithm is developed whereby optimal or near-optimal task allocations can evolve during the operation of a parallel computing system. The algorithm considers other load balancing issues such as threshold policies, information exchange criteria, and inter-processor communication.

In [17] Suresh, Huang and Kim describe a hybrid real-coded genetic algorithm for data partitioning in multi-round load distribution and scheduling in heterogeneous systems. The scheduling problem solution is based on multi round load balancing of divisible loads in a single-level tree of processors with the round participation defined by loads inter-dependence relation.

Greene [18] designs a GA scheduling routine in which incoming tasks (of varying durations) accumulate, then are periodically scheduled, in small batches, to the available processors. Two important priorities for the scheduling work are that loads on the processors are well balanced, and that scheduling per se remains cheap in comparison to the actual productive work of the processors.

To address the problem of dynamic load balancing in a processing pool, Uyar and Harmanci [19] apply an improved genetic algorithm called damGA (diploidy-aging-meiosis genetic algorithm). Promising results for load balancing in a network of processing units are reported.

Xhafa et al. [20] present GA-based schedulers for efficiently allocating jobs to resources in a Grid system, when makespan and flowtime are to be minimized. Two encoding schemes have been considered and most of GA operators for each of them are implemented and empirically studied. The extensive experimental study showed the efficiency of their GAs when makespan and flowtime are minimized either in a hierarchical or a simultaneous optimization mode; previous approaches considered only the minimization of the makespan.

Sahoo et al. [21] investigate a dynamic load balancing algorithm for heterogeneous distributed systems where half of the processors have double the speed of the others. Two job classes are considered for the study, the jobs of first class are dedicated to fast processors. While second job classes are generic in the sense they can be allocated to any processor. The performance of the scheduler has been verified under scalability.

A model based on GAs is proposed by Prakash and Vidyarthi [22] to achieve better dynamic load balancing in computational grids. To study the performance of the model, experiments have been conducted by simulating the model.

Recently, Lin and Deung [23] solve dynamic load balancing in cloud-based multimedia system (CMS) using a GA. They consider a centralized CMS consisting of a resource manager, cluster heads, and server clusters. Their dynamic load balancing algorithm spreads the multimedia service task load on servers with the minimal cost for transmitting multimedia