Applied Soft Computing 38 (2016) 530–547

Contents lists available at ScienceDirect

Applied Soft Computing

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

Studying the effect of population size in distributed evolutionary algorithms on heterogeneous clusters

Pablo García-Sánchez[a,*], Gustavo Romero[a], Jesús González[a], Antonio Miguel Mora[a], Maribel García Arenas[a], Pedro Ángel Castillo[a], Carlos Fernandes[b], Juan Julián Merelo[a]

[a] Department of Computer Architecture and Computer Technology and CITIC-UGR, University of Granada, Granada, Spain
[b] LaSEEB-ISR-IST, Technical University of Lisbon (IST), Lisbon, Portugal

ARTICLE INFO

Article history:
Received 8 August 2013
Received in revised form
17 September 2015
Accepted 29 September 2015
Available online 19 October 2015

Keywords:
Evolutionary Algorithms
Genetic algorithms
Distributed computing
Parameter tuning
Parameter control

ABSTRACT

Distributed Evolutionary Algorithms are traditionally executed on homogeneous dedicated clusters, despite most scientists have access mainly to networks of heterogeneous nodes (e.g., desktop PCs in a lab). Fitting this kind of algorithms to these environments, so that they can take advantage of their heterogeneity to save running time, is still an open problem. The different computational power of the nodes affects the performance of the algorithm, and tuning or fitting it to each node properly could reduce execution time.

Since the distributed Evolutionary Algorithms include a whole range of parameters that influence the performance, this paper proposes a study on the population size. This parameter is one of the most important, since it has a direct relationship with the number of iterations needed to find the solution, as it affects the exploration factor of the algorithm. The aim of this paper consists in validating the following hypothesis: fitting the sub-population size to the computational power of the heterogeneous cluster node can lead to an improvement in running time with respect to the use of the same population size in every node.

Two parameter size schemes have been tested, an offline and an online parameter setting, and three problems with different characteristics and computational demands have been used.

Results show that setting the population size according to the computational power of each node in the heterogeneous cluster improves the time required to obtain the optimal solution. Meanwhile, the same set of different size values could not improve the running time to reach the optimum in a homogeneous cluster with respect to the same size in all nodes, indicating that the improvement is due to the interaction of the different hardware resources with the algorithm. In addition, a study on the influence of the different population sizes on each stage of the algorithm is presented. This opens a new research line on the fitting (offline or online) of parameters of the distributed Evolutionary Algorithms to the computational power of the devices.

1. Introduction

Evolutionary Algorithms (EAs) are a general method for solving optimization and search problems inspired by the evolution of species and its underlying mechanism, natural selection. These algorithms work on a population of encoded possible solutions, called *individuals*, every one competing with others according to their *fitness* (quality or cost of the solution they encode). The population evolves, with iterations of the algorithm (or *generations*), by means of selection and recombination/mutation operators, creating a new set of candidates. This goes on until a *stopping criterion* (e.g. number of generations) is met. The fitness function, which describes the target problem, expresses the quality of the solution, giving the grade of adaptation of an individual.

The performance of an EA can be measured by the running time required to find the optimal value and/or by the quality of the solution attained in a fixed time. The performance is affected by a set of parameters, such as the crossover rate, or population size, among others. Population size has been studied as a fixed [27] or adaptive [19,29] value, as it has a significant impact on the efficiency of an EA [28].

[*] Corresponding author. Tel.: +34 958241778; fax: +34 958248993.
E-mail addresses: pablogarcia@ugr.es (P. García-Sánchez), gustavo@ugr.es (G. Romero), jesusgonzalez@ugr.es (J. González), amorag@geneura.ugr.es (A. Miguel Mora), maribel@ugr.es (M.G. Arenas), pacv@ugr.es (P.Á. Castillo), cfernandes@laseeb.org (C. Fernandes), jmerelo@geneura.ugr.es (J. Julián Merelo).

http://dx.doi.org/10.1016/j.asoc.2015.09.052
1568-4946/© 2015 Elsevier B.V. All rights reserved.

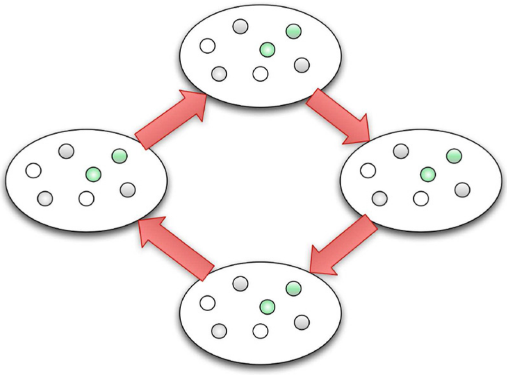

Fig. 1. Island model scheme using a neighborhood ring topology.

Performance can also be improved using Distributed EAs (dEAs) [6], where the algorithm is run on a set of nodes simultaneously, working with different sub-populations (or islands) at the same time. One or more individuals are interchanged (migrated) periodically between sub-populations, which are connected following a specific topology. Fig. 1 shows the island model with a ring topology.

Cantú-Paz, in [8], published a theoretical study about population size, migration rate and the degree of the topology. This kind of algorithms has been usually executed on homogeneous clusters [6,42], being used with the same parameters in all nodes (homogeneous dEAs), or with different parameters (heterogeneous dEAs) [40,38,24].

As previously said, population size is a key factor for the performance of dEAs [28], so this paper investigates whether adapting the sub-population size of each node in a distributed Evolutionary Algorithm can leverage the computational capabilities of a heterogeneous cluster.

This is motivated by new trends in distributed computing, such as Cloud Computing [11], GRID [5] or Service Oriented Science [20], which are leading to heterogeneous computational devices, such as laptops, virtual machine instances, GPUs [1], tablets or desktop PCs, working in the same environment. Thus, many laboratories, which do not include classic clusters but the usual desktop and laptop computers, should be able to leverage this motley set as a heterogeneous cluster.

It has been shown that [3] asynchronous dEAs with the same parameter configuration are even more efficient in time and the number of evaluations in reaching the optimal solution on this kind of environments than on clusters with homogeneous devices. This can be explained by different reasons, such as different memory access times, cache sizes, or even implementation languages or compilers in each machine, leading to a different exploitation/exploration rate of the search space [3].

These facts have motivated the study of a combination of both ideas in this paper: asynchronous dEAs on a heterogeneous set of nodes with different parameter values fitted to each node. The concept *parameter fitting* is proposed in this work to distinguish with respect the term *parameter adaptation* defined by Eiben in [15]. Our concept specifies that the parameter values are set (online or offline) taking into account the computational power of the nodes, while the parameter adaptation implies a modification (online) of the parameters taking into account the information provided by the algorithm itself (for example, a variation in the average fitness value).

In this case, the parameter to be fitted to the computational power of each node has been the sub-population size of each island. The population size is of capital importance in the accuracy and convergence speed of an EA. The population is a repository of the building-blocks that an evolutionary computational process recombines and mutates in order to improve the average fitness of the population and, eventually, find the optimal solution to the target-problem. If the supply of building-blocks is insufficient, selection and genetic drift will probably guide the search towards premature convergence to a local optimum (or the EA becomes too dependent on the randomness of the mutation operator). Larger populations guarantee a more diverse supply of building-blocks (higher genetic diversity) and are more likely to overcome premature convergence. However, in large populations, convergence speed is the payoff for accuracy: in general, when the population size of an EA grows, the convergence speed is reduced [9]. When having different processors with different computational power computing sub-populations of an EA, it may be a good policy to distribute the individuals in a way that less efficient nodes are assigned with smaller populations, having smaller diversity. This solution not only integrates within an EA two different types of evolutionary search (fast, with the smaller populations, and accurate, with the larger), but also puts on the most efficient nodes the burden of computing larger sub-populations. This combination can reduce computation time as slower nodes can exploit solutions without premature convergence and, at the same time, benefit from the diversity of the faster ones.

In this work, a heterogeneous distributed system has been used to answer the following research questions:

- Can an asynchronous distributed EA be fitted to leverage the capability of a heterogeneous cluster?
- How does the fitting of the sub-population size to the computational power affect the running time and number of evaluations?
- Is there any difference between using the same sub-population sizes on a homogeneous and a heterogeneous cluster?
- How is each stage of the algorithm (selection, recombination, mutation, replacement and migration) affected by the different configurations?

The rest of the work is structured as follows: after a background in the algorithm parameter fitting to computational substrate in dEAs, the developed algorithms and experimental setting (Section 3) are presented. Then, the results of the experiments are shown (Section 4), followed by conclusions and suggestions for future work lines.

2. Background on tuning and fitting EAs on heterogeneous clusters

Fitting algorithm parameters to available heterogeneous computational resources usually leads to improved performance [36]. An easy way to take advantage of the available resources is balancing the load [21,30] so that workloads are distributed across multiple elements. However, assigning equal tasks to each node in heterogeneous clusters may result in suboptimal performance [7].

One of the problems in parameter tuning and adaptation on heterogeneous clusters is the computational load representation. It depends on the algorithm, size of the problem, language, compiler or hardware characteristics, so the results obtained from artificial benchmarks, such as Linpack [18], should not be extolled as identificative of the system performance [14]. For example, in the work of Garamendi et al. [21], a small benchmark was executed on all nodes at the beginning of the algorithm in order to distribute individuals of an Evolutionary Strategy (ES), following a master-slave model. The computational load by artificial benchmarks may not