Information Sciences 293 (2015) 35–58

Contents lists available at ScienceDirect

Information Sciences

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

A high performance memetic algorithm for extremely high-dimensional problems

Miguel Lastra [a,*], Daniel Molina [b], José M. Benítez [c]

[a] Depto. Lenguajes y Sistemas Informáticos, E.T.S. Ingeniería Informática y Telecomunicación, CITIC-UGR, iMUDS, Universidad de Granada, Spain
[b] Depto. Ingeniería Informática, E.S. Ingeniería, Universidad de Cádiz, Spain
[c] Depto de Ciencias de la Computación e Inteligencia Artificial, E.T.S. Ingeniería Informática y Telecomunicación, CITIC-UGR, iMUDS, Universidad de Granada, Spain

ARTICLE INFO

Article history:
Received 3 August 2013
Received in revised form 3 August 2014
Accepted 12 September 2014
Available online 22 September 2014

Keywords:
Memetic algorithm
Optimization problem
GPU
CUDA

ABSTRACT

Throughout the last years, optimization problems on a large number of variables, sometimes over 1000, are becoming common. Thus, algorithms that can tackle them effectively, both in result quality and run time, are necessary. Among these specific algorithms for high-dimensional problems, memetic algorithms, which are the result of the hybridization of an evolutionary algorithm and a local improvement technique, have arisen as very powerful optimization systems for this type of problems. A very effective algorithm of this kind is the MA-SW-Chains algorithm. On the other hand, general purpose computing using Graphics Processing Units (GPUs) has become a very active field because of the high speed-up ratios that can be obtained when applied to problems that exhibit a high degree of data parallelism.

In this work we present a new design of MA-SW-Chains to adapt it to the GPU-based massively parallel architecture. The experiments with the new GPU memetic technique, compared to the original sequential version, prove that the results are obtained with the same quality but with a reduction of the run time of two orders of magnitude. This great improvement in computing time makes our proposal suitable for future optimization problems with a dimensionality several orders of magnitude greater than current high-dimensionality standards (i.e. problems with millions of variables). The remarkable run time reduction comes at the cost of a higher design and implementation overhead compared to the CPU-based single-threaded counterpart.

1. Introduction

Evolutionary Algorithms (EAs) [3] are meta-heuristic techniques that have arisen as very good algorithms for optimization problems. These algorithms can be applied to a variety of real-world optimization problems because they do not require specific information about the problem at which they are targeted, obtaining very good results in optimization problems with computing and/or run time restrictions. Furthermore, they have shown considerable success in dealing with problems characterized by complex solution spaces.

To improve the effectivity of the search process an EA can be hybridized with a local improvement method. Memetic Algorithms (MAs) [35,36] are extensions of EAs with a separate local search process (LS) that improves the optimization process [29]. MAs are simple but flexible and powerful algorithms [32,41] that find high-quality solutions in many real-word problems [12,25,53].

* Corresponding author. Tel.: +34 958246144.
E-mail addresses: mlastral@ugr.es (M. Lastra), daniel.molina@uca.es (D. Molina), J.M.Benitez@decsai.ugr.es (J.M. Benítez).

http://dx.doi.org/10.1016/j.ins.2014.09.018
0020-0255/

Nowadays, it is very common to face real-world problems which require optimizing a rising number of variables. Typical optimization problems involve tens of variables. However, in research fields that have a growing interest it became obvious that a larger number of variables is required, leading to high-dimensional optimization problems: in data mining problems, by the huge size of available data (such as clustering in text analysis [31,6]); biomedical problems as in DNA and molecular simulation [27,65,4]; networking problems [54,18]. In fact, the requirement of high-dimensional algorithms is increasing and currently it is not unusual to tackle problems represented by a very high number of dimensions (over 10^6) like feature selection techniques [24], molecular simulation [65] or forecasting [38]. With the appearance of big data the number of huge-scale optimization problems (above dimension 10^8) is growing, as complex simulation [13], data mining [1], quantum chemistry [46], spectroscopy analysis [44], geophysical analysis [7], drug discovery [17], genomic studies [47], etc.

Optimization of high-dimensional problems, also called large-scale optimization, implies a higher complexity in EAs, not only because many techniques are not suitable for higher dimensions but because the error increases with dimensionality. EAs try to achieve a compromise between accuracy and invested effort (in terms of number of evaluations or run time), thus, when applied to high-dimensional problems, EAs do not always produce good results in an affordable time.

High-dimensional optimization problems have several characteristics that makes them difficult to solve. First, when the dimensionality increases the search domain size increases exponentially, while the number of evaluations that can be performed usually increases lineally as computational resources are added. Also, techniques that use gradient information (or approximation, such as Quasi-Newton techniques), present problems in high-dimensional systems [55]. Additionally, proximity and neighborhood relationships are difficult to assess because, although it is widely used, the Euclidean distance may not be an appropriate measure in high-dimensional problems [1].

In recent years different EAs specifically designed for large-scale optimization have been proposed to tackle the aforementioned issues. In particular, MAs have proven to be very competitive in this field because they are based on stochastic algorithms that do not require any gradient information and because the use of LS methods may improve the performance of the algorithm by quickly exploring around the best solutions (especially useful in high-dimensions). Unfortunately, applying traditional LS techniques to high-dimensional problems increases the computational intensity because of the high number of variables and the additional fitness function evaluations added by the local search process. MA-SW-Chains [34] is a specially interesting MA because it allows a greater computational intensity to be applied only to the most promising solutions until they do not improve, while maintaining a fixed effort ratio (ratio of number of evaluations) invested in the LS method. With this combination the algorithm exhibits a good trade-off between exploration and exploitation in high-dimensional problems. The fact that MA-SW-Chains was the winner of the large-scale global optimization session in the IEEE Congress on Evolutionary Computation in 2010 proves it is a very competitive algorithm in this field.

However, MA-SW-Chains is a sequential algorithm, where the process is run by one CPU and each step of the algorithm is run after the previous one. On the other hand, parallel algorithms are able to perform several computations at the same time and can greatly reduce the required processing time. For high-dimensional optimization problems this kind of algorithms are specially interesting because they could reduce the high computing time due to the huge amount of variables [2,43,45].

One parallel architecture that has obtained very good results for certain kind of parallel algorithms is the one provided by Graphics Processing Units (GPUs). General Purpose computation on GPUs (GPGPU) allows algorithms to perform parallel computations over different data using the general purpose computing capabilities of modern GPUs. Recently, several parallel EAs for optimization using GPUs have been published, as Particle Swarm Optimizations (PSO) [11] or Differential Evolution (DE) [26,20].

In this work we propose a GPU-based design and implementation of the MA-SW-Chains algorithm. The objective we have sought was to improve the performance of this algorithm to make it adequate for huge-dimensional problems without reducing the quality of the results.

The GPU version of MA-SW-Chains is compared to the original version over several well-known optimization problems which are frequently used in different high-dimensional benchmarks. This empirical study includes both the run time and error measure analysis. Furthermore, the scalability achieved using GPUs is also studied and experiments are not limited to usual high-dimensional values in the order of 1000 variables and reach dimensionality values up to 3,000,000. The goal was to analyze if GPU-based implementations could be ready even for future high-dimensional problems. The results that were obtained show that GPUs allow tackling optimization problems with dimensionality levels not affordable with the original MA-SW-Chains version in a reasonable time.

This paper has the following structure: in Section 2, GPGPU programming is introduced, in Section 3, a brief review of EAs designed for high-dimensionality, paying special attention to memetic algorithms, is shown. In Section 4 the original MA-SW-Chains algorithm is explained. In Section 5, our parallel algorithm is described, remarking the differences with respect to the CPU version. In Section 6, the experiments using the GPU and CPU version are presented. Finally, in Section 7 the results are analyzed and in Section 8 the main conclusions and future work are detailed.

2. GPGPU programming

The GPGPU computation discipline (General Purpose computation on Graphics Processing Units) has been a very active research topic in the last years, especially since computing frameworks such as CUDA [14] or openCL [42] were introduced. These platforms have allowed using the great computing capabilities of modern Graphics Processing Units for general