Journal of ELECTRICAL ENGINEERING, VOL. 56, NO. 12/s, 2005, 123–125

PARALLEL GENETIC ALGORITHMS FOR GRAPH COLOURING PROBLEM USING MESSAGE PASSING PARADIGM

Szymon Łukasik *

This paper presents the application of Parallel Genetic Algorithms for solving Graph Colouring Problem with Message Passing Paradigm being used as an implementation tool. Master-slave and migration models are under author's consideration. Their performance and influence of various models' parameters (i.e. number of processors used, migration ratio etc.) on PGA's efficiency is discussed.

K e y w o r d s: graph colouring, parallel genetic algorithms

2000 *Mathematics Subject Classification*: 05C15, 68T20, 68W10

1 INTRODUCTION

GCP — Graph k-Colouring Problem [8] for undirected graph G=(V,E) is defined as a task of finding assignment c: $V \to N$ of colours to graph's vertices such as:

$$u, v \in E \Rightarrow c(u) \neq c(v)$$

which means that adjacent vertices are coloured differently. The problem of finding a minimum k-Colouring of a graph belongs to the class of NP-hard optimization problems and has been the subject of Second DIMACS Implementation Challenge [7] in 1993.

Due to its hard computability, GCP is often solved using heuristic methods like Genetic Algorithms(GAs). The first extensive study of the application of GAs to the GCP was made by Fleurent and Ferland [4]. Since then several heuristic algorithms have been developed for GCP — the most considerable work has been done in the field of hybrid methods (i.e. containing two heuristics: usually GA along with Tabu Search or Simulated Annealing) [5], [6].

The purpose of this paper is to present results of using simple parallelized Genetic Algorithm for GCP. This approach involves execution of an evolutionary algorithm on numerous processing units — it can significantly improve GA's performance. The general idea is well known and frequently mentioned in literature for almost 20 years [9].

Parallelization can be achieved by creating a number of separate populations which exchange genetic information during migration process (according to selected migration scheme) — this PGA variant is known as the migration model. The alternative master-slave model uses only one population on master processor which assigns some computations (usually the evaluation of individual) to slave processors. An overview on parallelized GAs and different task distribution schemes can be found in [2], [3].

The article deals with both presented models. They were implemented for GCP using Message Passing parallelization paradigm [10]. In message-passing processes exchange data with one another by sending messages. This method of implementing parallel is very flexible, universal, portable to various architectures and can be highly efficient.

The paper consists of 4 sections. The second section explains in detail chosen parallelization models and presents their application to Graph Colouring Problem. Section 3 contains the results obtained during experimentents using DIMACS standard graph instances. Some remarks are being made about prepared implementation of PGAs and its properties. Finally, the scope of further work in this area is presented.

2 PARALLEL GENETIC ALGORITHMS FOR GCP PROBLEM

For the purpose of genetic implementation, each solution (individual) p is represented by a list of colours corresponding to graph vertices. The following function can be used as the cost function for k-Colouring:

$$f(p) = \sum_{\{u,v\} \in E} q(u,v) + |k(p) - k|$$

where

$$q(u,v) = \begin{cases} 1 & \text{when } c(u) = c(v)\,, \\ 0 & \text{otherwise} \end{cases}$$

and $k(p)$ is the number of colours used in individual p, k is the desired number of colours

As the mutation and crossover operators simple one-point crossover and point mutation were used.

* Department of Automatic Control, Faculty of Electrical and Computer Engineering, Cracow University of Technology, Warszawska 24, 31 155 Kraków, Poland, E-mail: szymonl@pk.edu.pl

ISSN 1335-3632

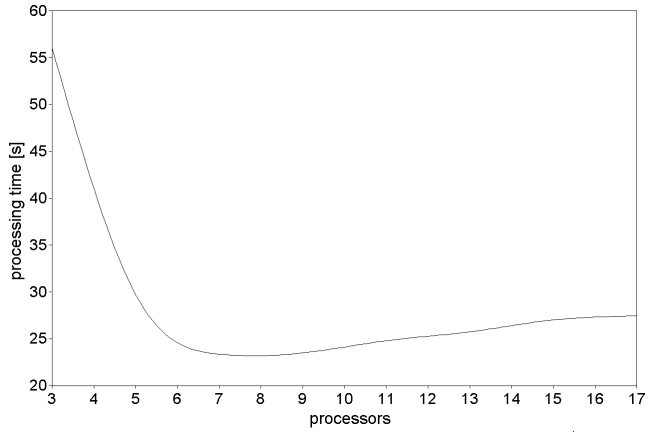

Fig. 1. Processing time in function of processors number (Master-Slave model, Mulsol.i.4 instance)

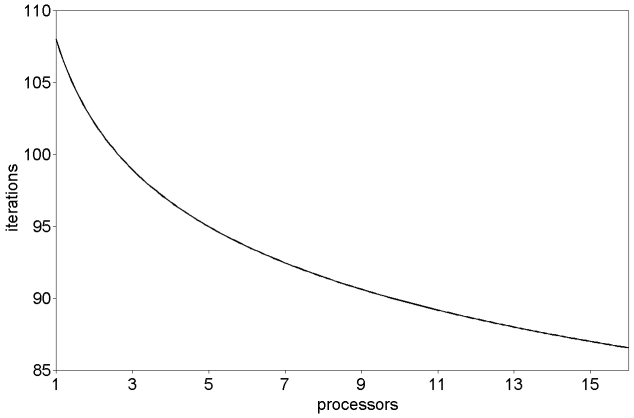

Fig. 2. Processing time in function of processors number (Migration model, Games120 instance)

Parallelization can be achieved using two different models (as presented in the previous part of the paper). Following subsections explain them in detail.

I Master-slave model

Parallelization affects only evaluation of the population. First, each slave processor receives an individual to evaluate. Master unit is fetching results and sending additional individuals to slaves. Other parts of Genetic Algorithm are executed only on main processing unit.

This method is useful especially for large-scale problems and for computationally demanding evaluation operators. Its main drawback is the idleness of slave processors during other phases of genetic algorithm. The other disadvantage of this method is the high communication ratio between master and slave units.

II Migration model

Each processor maintains its own population (island) of individuals and evaluate it using the same genetic operators. Occasionally migration occurs — every processor sends a part of its population ("best" individuals) to all other units and receives their representatives as well. On every island the "worst" individuals are replaced by newcomers. The presented approach is called the "pure" migration model. It is possible to choose other migration scheme i.e. sending to individuals only to neighbouring islands, migrating random groups of representatives or replacing random individuals.

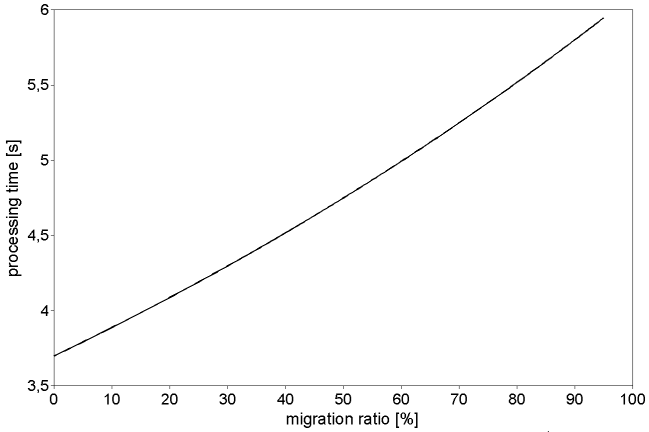

Fig. 3. Processing time in function of migration ratio (Migration model, Anna instance)

Other properties of the migration process are: migration probability P and migration ratio R which reflects the number of exchanged individuals referenced to population size and processor number as follows:

$$R = \frac{\text{processors} * \text{individuals}}{\text{population}}.$$

Migration ratio describes the diversity of populations after the migration process. If it is close or equal 1 then resulting populations are similar which has a negative effect on PGAs search space size.

3 EXPERIMENTAL RESULTS

The properties of PGA's implementation for k-Colouring problem were experimentally verified using computer network consisting of 18 Pentium© 4 running Linux operating system and LAM/MPI [1]. All computations were performed for standard DIMACS graph colouring instances. Obtained results for both models of parallel implementation are presented in the following subsections of the paper.

I Master-slave model

The influence of a number of processors on computation time was under consideration. Figure 1 presents the example result obtained for Mulsol.i.4 graph (185 vertices, 3946 edges, 31-colouring).

It can be seen that there exists an optimum number of processors used in execution of the Master-Slave algorithm. For a higher number of processors the effect of improving computing power of parallel system is reduced by increasing communication time needed in the evaluation phase.

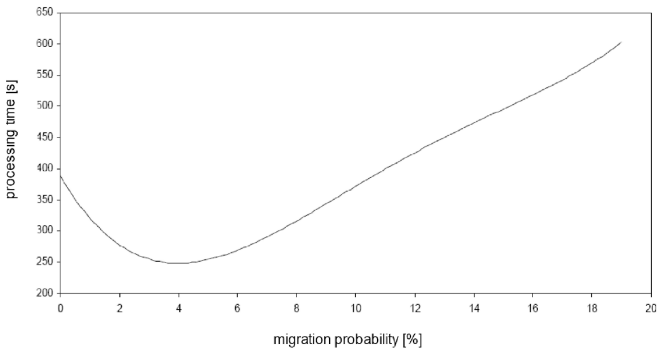

Fig. 4. Processing time in function of migration probability (Migration model, Queen5 instance)

II Migration model

The relation between the number of processors and algorithm execution time for migration model calculated for Games120 graph instance (120 vertices, 638 edges, 9-colouring) is presented in Fig. 2. The time decreases with the number of processors involved being increased. Even though it is worth mentioning that due to network saturation the income of using more processing units becomes less significant for a large number of processors used.

The cost of migration process grows exponentially with increasing migration ratio (and constant processors number) as presented in Fig. 3 for Anna graph instance (138 vertices, 493 edges, 11-colouring).

Figure 4 presents the relation between computation time and selected migration probability for constant number of islands and Queen5 graph instance (25 vertices, 160 edges, 5-colouring). It can be seen that choosing the right (about 4%) migration probability value significantly improves algorithm performance. Too high migration probability leads to the effect of reducing the diversity of populations. On the other hand, setting this coefficient at very low level decreases the income of exchanging individuals between islands.

4 CONCLUSIONS

As presented in the paper, parallelization of genetic algorithms for GCP can be effectively used to improve their performance. Migration model is especially well suited for large scale Colouring problems. It is though worth mentioning that choosing the right model parameters (migration ratio, migration rate, processors number) is crucial for algorithm efficiency.

Further work referring to the subject of this contribution will concern the application of custom Graph Colouring genetic operators and applying different migration schemes. The application of PGAs for non-classical Graph Colouring problems like harmonious colouring, exact colouring are yet to be presented as well.

Acknowledgement

The author acknowledges many useful suggestions of Zbigniew Kokosiński (PhD) during preparation of the paper and would also like to thank Zenon Cyganek for his help during implementation phase of the presented contribution.

References

[1] BURNS, G.—DAOUD, R.—VAIGL, J.: LAM: An Open Cluster Environment for MPI, Proceedings of Supercomputing Symposium, http://www.lam-mpi.org/download/files/lam-papers.tar.gz, 1994.

[2] CANTU-PAZ E.: A survey of parallel genetic algorithms, Calculateurs Paralleles, Reseaux et Systems Repartis, vol 10, number 2, pp. 141-171, 1997.

[3] CANTU-PAZ, E.—GOLDBERG, D.E.: Parallel genetic algorithms: theory and practice, Computer Methods in Applied Mechanics and Engineering, Elsevier, 2000.

[4] FLEURENT, C.—FERLAND, J.A.: Genetic and hybrid algorithms for graph colouring, Annals of Operations Research, pp. 437-461, 1995.

[5] FOTAKIS, D.A.—LIKOTHANASSIS, S.D.—STEFANAKOS, S.K.: An Evolutionary Annealing Approach to Graph Colouring, EvoWorkshop 2001, LNCS 2037, pp. 120-129, Springer-Verlag, 2001.

[6] GALINIER, P.—HAO, J.K.: Hybrid evolutionary algorithms for graph colouring, Journal of Combinatorial Optimization, 1998.

[7] JOHNSON, D.S.—TRICK, M.A.: Cliques, colouring and satisfability: Second DIMACS Implementation Challenge, DIMACS Series in Discrete Mathematics and Theoretical Computer Science Vol.26, 1996.

[8] KUBALE, M.: Optymalizacja dyskretna. Modele i metody kolorowania grafów, WNT, 2002.

[9] PETTEY C.S. et al.: A Parallel Genetic Algorithm, Proceedings of the Second International Conference for Genetic Algorithms, Morgan Kaufmann Publishers, p. 155-161, 1987.

[10] SNIR, M.—OTTO, S.—HUSS-LEDERMAN, S.—WALKER, D.—DONGARRA, J.: MPI: The Complete Reference (Vol. 1) - 2nd Edition, MIT Press, 1998.

Received 11 July 2005

Szymon Łukasik (MSc) is a student-assistant at the Faculty of Electrical and Computer Engineering of Cracow University of Technology. He graduated in computer science and is currently studying control engineering at CUT.