ANNALS of Faculty Engineering Hunedoara
International Journal of Engineering
ISSN: 1584 - 2665

THE USE OF MDCE ENVIRONMENT FOR PARALLEL PROCESSING OF STOCHASTIC ALGORITHMS

Martin VESTENICKÝ[1)], Peter VESTENICKÝ[2)]

[1)] University of Žilina, Faculty of Electrical Engineering,
Department of Telecommunications and Multimedia, Žilina, SLOVAKIA
[2)] University of Žilina, Faculty of Electrical Engineering,
Department of Control and Information Systems, Žilina, SLOVAKIA

ABSTRACT:
This paper deals with basic architectures of computing system for parallel processing of stochastic algorithms. The architecture client – server, peer – to – peer and their advantages and disadvantages are described in details. In the next part of paper an application of the MDCE (Matlab Distributed Computing Engine) environment as an integral part of the Matlab software package from the MathWorks, Inc. is presented. The MDCE has been applied to optimize selective filters used in telecommunication industry with using of parallel processed evolutionary algorithm. Results of experiments are discussed from the viewpoint of the parallel processing contribution to calculation speed.
KEYWORDS:
Parallel processing, evolutionary algorithm, Matlab, population matrix, client – server, peer – to – peer.

1. INTRODUCTION

A high requirement on the computing power is one of the greatest disadvantages of stochastic algorithms [1, 5, 6, 7]. These requirements are closely linked with the amount of time needed to find a solution. A possible answer to this problem is the parallel processing of particular calculating operations. Positive property of the stochastic algorithms is that the majority of calculations can be processed in parallel. The most suitable operation for parallel processing is the costing of individuals because the number of individuals in population is relatively high and the costing of one individual is independent on the costing of remaining individuals. Basically the whole stochastic algorithm can be executed in parallel with a certain information exchange mechanism about the best achieved solutions, optimum solution etc. among particular cooperating workstations.

The theoretical limit of the parallel processed algorithm speed up coefficient specifies Amdahl's law which is defined by formula (1) if N identical computers are used:

$$K_Z = \frac{1}{F + \frac{1-F}{N}} \tag{1}$$

where F is fraction of operations that must be executed in serial only mode (i. e. then 1-F is fraction of operations that can be parallelized) and N is the number of parallel working computers (processors).

Resulting from formula (1) the coefficient of calculation speed up K_Z approaches the value of F^{-1} for big N so the stochastic algorithm must contain minimum number of serially performed operations.

For parallel processing of stochastic algorithm a suitable architecture of computing system must be selected to process particular calculation effectively. It is necessary to take into account the fact that the separate workstations must inform each other about the results of particular calculations. The workstations must be interconnected; the simplest way is to utilize the local area network (LAN) which has some limitations for example:

- The data transfer rate is not infinite – this is limitation for transfer of big data volumes,
- The network delay is not zero – this is limitation for frequent transfer of small data volumes.

These facts must be taken into account when parallel stochastic algorithm is designed especially the chromosome representation of individual and population matrix dimension must be properly selected. From the logical topology point of view the following architectures are suitable:

- Client – server architecture,
- Peer – to – peer architecture.

More detailed information about architectures and operating mode of parallel computing systems can be found in [3, 8].

2. CLIENT – SERVER ARCHITECTURE

The logical topology of the computing system with client – server architecture is presented in Fig. 1. The computing system contains one server and *N* clients. The server manages entire computing system. Its functions are defined in the next list:

- The generation of initial population matrix,
- Splitting of population matrix into particular population submatrices randomly or deterministically,
- The generation of descendants can also be performed by client stations,
- Sending out the particular population submatrices to the clients for further processing,
- Reception of the new generation population submatrices from the clients and reassembling of the new generation population matrix,
- Check of the algorithm termination criterion,
- Management of the whole computing system (workstation login, logout, detection of workstation fail etc.).

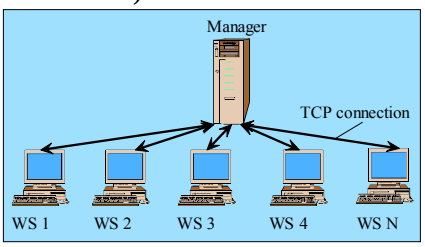

Figure 1. Client – server system

The method of working with population matrix and population submatrices is presented in Fig. 2 and Fig. 3. At the first step (Fig. 2) the population matrix is randomly or deterministically split into particular population submatrices which are sent out to the client stations for processing. At the second step (Fig. 3) the population submatrices received from the client stations are reassembled into one unit – population matrix. The use of random or deterministic method of population matrix splitting depends on whether or not the client station generates the new individuals. If the client station generates the new individuals it is suitable to split the population matrix randomly to prevent early convergence of algorithm.

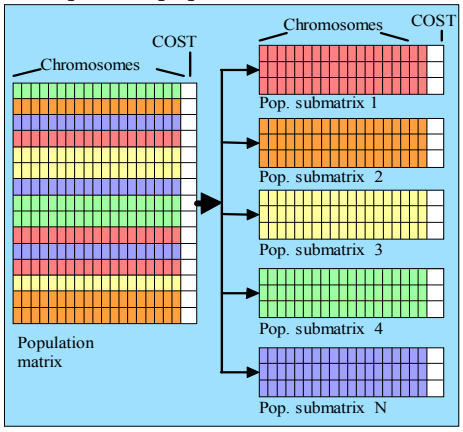

Figure 2. Splitting of population matrix into submatrices

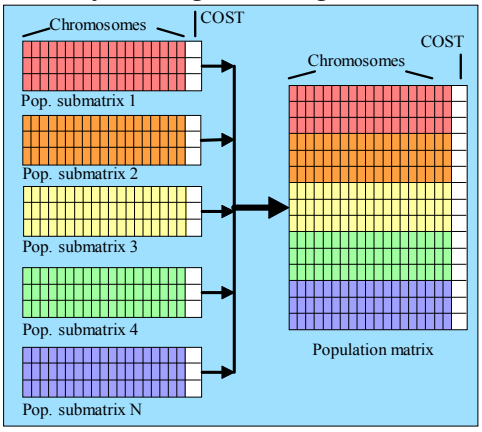

Figure 3. Reassembling of population submatrices into population matrix

The functions of client station are:

- Reception of population submatrix from the manager,
- The generation of descendants – also can be done by the management station,
- Costing of individuals,
- Decision about insertion of individual into the new generation,
- Sending out the population submatrix to the manager.

Described method of working with population matrix requires certain synchronization of workstations. It is implicitly assumed that the new generation will start after the reception and costing of the population submatrices from all participating workstations. The use of workstations with equal computing power is recommended to enable effective work of the whole system. If workstations with various computing power are used the population matrix must be split unequally to keep the time of costing process approximately identical. The powerful workstation would process a larger population submatrix and less powerful ones would process a smaller submatrix.

The work with the population matrix can be partially modified to enable asynchronous mode of workstation operation but their management would be more difficult.

3. PEER – TO – PEER ARCHITECTURE

The computing system logical topology of peer – to – peer architecture is shown in Fig. 4. The workstations are logically interconnected by polygonal network. The method of working with population matrix is presented in Fig. 5. Every workstation keeps its own population matrix and performs complete stochastic algorithm. Moreover every workstation performs the tasks to control entire system. These tasks can be defined as:

- To search for other workstations which cooperate on calculations,
- To notify about finding of new workstation,
- To notify that given workstation is leaving the computing system,
- Creation of list with defined number of the most successful solutions,
- Distribution of the most successful solutions list to other workstations,
- Reception of the most successful solutions lists from other workstations,
- To incorporate the received solutions in accordance with selected criterions into own population,
- To check the algorithm termination criterion,
- To notify the other workstations about solution finding.

For effective work it is suitable to split the space of solutions to the parts which are proportional to the computational power of particular workstations. It can be done by proper selection of population size and boundary of solutions. All these parameters would be dynamically scalable. The purpose of these precautions is to prevent that the workstations search the same space of solutions. It is essentially imitation of organism evolution in various isolated areas.

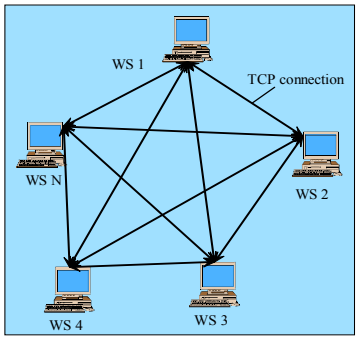

Figure 4. Peer – to – peer architecture of computing system

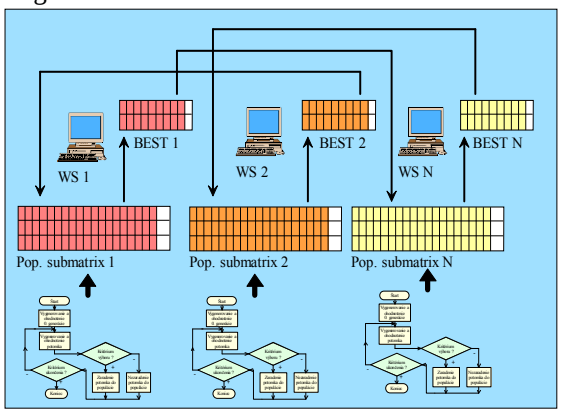

Figure 5. Processing of stochastic algorithm by peer – to – peer architecture

4. IMPLEMENTATION IN THE MATLAB ENVIRONMENT

The Matlab software package contains as its own integral part the Matlab Distributed Computing Toolbox which creates environment intended for parallel processing of computational tasks. Logical architecture of MDCE environment is presented in Fig. 6. The MDCE contains three basic parts:

- The client which is created by running MATLAB application. The client performs given algorithm, communicates with the task manager to transfer the computing tasks and to receipt the results.
- The task manager is a software entity which manages the workers and communicates with the client, transfers the received tasks from the client to workers, collects the results of the calculations from workers and sends them to the client.
- The worker is a software entity which performs the tasks received from the manager and sends back the results. Every worker must have defined the superior manager.

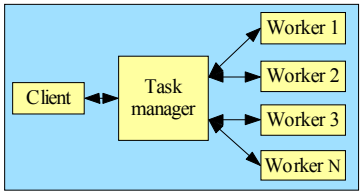

Figure 6. Logical architecture of the MDCE environment

Detailed information about MDCE environment configuration can be found in [2].

Resulting from Fig. 6 and from previous text the MDCE environment allows the implementation of parallel working evolutionary algorithm based on the client – server topology. Basically the MDCE environment can be used to evaluate *N* identical functions for *N* files of various arguments. This fact can be preferably used at parallel processing of stochastic algorithm which has been partially modified. The changes concern about parallel costing of individuals and minimization of communication among particular entities of the MDCE environment. Simplified situation is explained in the flowchart in Fig. 7.

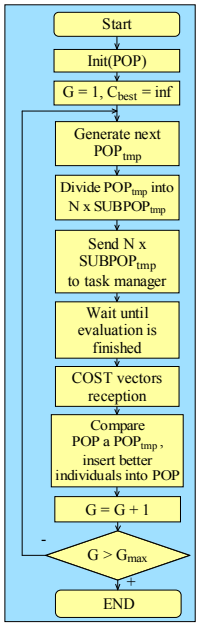

Figure 7. Parallelized differential evolutionary algorithm

At the start the zero generation individuals are generated which are stored in the population matrix **POP**. In the next step the first generation individuals are generated by corresponding operator and they are stored in the population matrix $\mathbf{POP_{tmp}}$. The matrix $\mathbf{POP_{tmp}}$ is then split into *N* identical population submatrices $\mathbf{SUBPOP_{tmp}}$ which serve as input arguments for *N* parallel evaluated instances of cost function. The result of every evaluation is a cost vector the algorithm is waiting for. It is necessary to keep the cost assignment of individuals. This is ensured by correct sequencing of cost vectors. In the next step the costs are compared with costs of individuals from the matrix **POP** and the individuals with better costs will replace the individuals from **POP**. At the same time the best individual is found. Then the algorithm end is checked and potential new generation or end of algorithm follow.

The Fig. 8 presents the utilization of various MDCE entities at processing of stochastic algorithm where data flows and roles of entities are shown. Exactly the possibility of parallel evaluation of cost functions for different parameters is used. The cost function is adapted to process the solutions group consisted of population submatrix and to return the cost vector whereby the correct cost assignment to the individuals is ensured. This solution has been selected to reduce the network delay influence and losses of time caused by entities of the MDCE environment (start of calculation, end of calculation, acquisition of results etc.). The splitting of population matrix into submatrices is much more advantageous as sending out the individuals to evaluate one after another. The whole computing system works synchronously i. e. the cost vectors are sent by manager to the client as late as the evaluation of given population submatrices set are completed by all workers. The MDCE can run asynchronously, too but it would require the changes of population matrix processing algorithm.

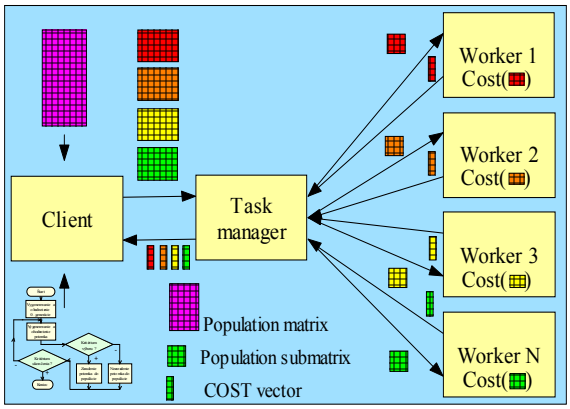

Figure 8. Utilization of MDCE entities to run the differential evolutionary algorithm

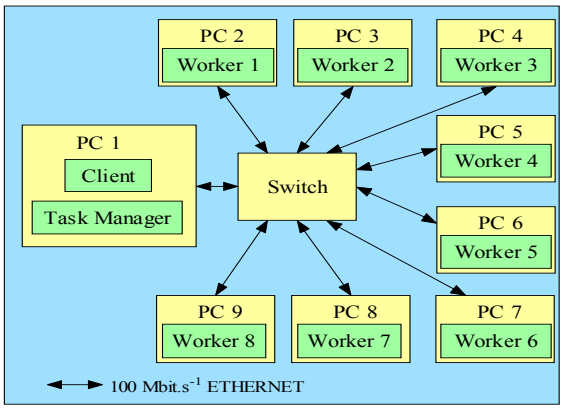

Figure 9. Hardware of computing system

5. APPLICATION OF MDCE IN THE CIRCUIT THEORY

For experimental verification of designed method of stochastic algorithm parallelization the computing system consisting of nine identical computers has been built. The configuration of hardware and placement of MDCE entities are shown in the Fig. 9. The computers have been interconnected by a standard Fast Ethernet network (faster network has not been available). The used computers contained CPU Intel Celeron 2.4 GHz, 512 MB RAM and Windows XP Home Edition. Both the client and the task manager have been run on the computer PC 1 to reduce the number of computers in the system.

The experimental verification of parallel processing in the MDCE environment has been performed by application of the differential evolutionary algorithm for solving of various optimization problems. In the experiments no. 1, 2 and 3 the elements of biquadratic circuit with given transfer function have been found whereby various degrees of used operational amplifier nonlinearity, optimization of dynamic relations in the circuit structure and minimization of minimum and maximum value ratio of used circuit elements have been taken into account. In the experiment no. 4 the filter for zero intermediate frequency receiver has been optimized to minimize bit error rate at data transfer. All the optimizations are in detail described in [2]. The basic algorithm of differential

evolution [1] has been partially adapted for parallel running in accordance with the flowchart in Fig. 7. The control constants of algorithm have been set as it is given in the table 1.

Table 1. Setting of control constants

Experiment		1	2	3	4
Number of unknown quantities	D	9	9	9	14
Number of generations	G_{max}	300	300	300	100
Number of population individuals	NP	900	900	900	280
Crossover ratio	CR	0.9	0.9	0.9	0.9
Weighting coefficient of differential mutation 1	F1	0.5	0.5	0.5	0.5
Weighting coefficient of differential mutation 2	F2	0.5	0.5	0.5	0.5

Described optimization tasks have been run on one, two, four and eight parallel working computers respectively. The results of these experiments are described in the tables 2 up to 5. Every table contains absolute time duration of experiment *t*, normalized time duration of experiment with respect of time duration on one computer t_{norm} (in %), speed up coefficient K_Z and fraction of operations that must be executed in serial only mode *F*. The tables are sorted by cost function complexity.

Table 2. Achieved times in experiment 1

N	1	2	4	8
t [s]	736	1065	1750	3444
t_{norm} [%]	100	144.70	237.77	467.93
K_z	1	0.691	0.421	0.214
F	1	1.894	2.834	5.198

Table 3. Achieved times in experiment 2

N	1	2	4	8
t [s]	17353	9479	5749	4571
t_{norm} [%]	100	54.62	33.13	26.34
K_z	1	1.831	3.018	3.796
F	1	0.092	0.108	0.158

Table 4. Achieved times in experiment 3

N	1	2	4	8
t [s]	30574	16110	9115	6169
t_{norm} [%]	100	52.69	29.81	20.18
K_z	1	1.898	3.354	4.956
F	1	0.054	0.064	0.088

Table 5. Achieved times in experiment 4

N	1	2	4	8
t [s]	103364	53558	29203	16270
t_{norm} [%]	100	51.81	28.25	15.74
K_z	1	1.930	3.539	6.353
F	1	0.036	0.037	0.038

6. CONCLUSION

Some facts can be formulated on the base of results from the previous tables: fraction of operations that must be executed in serial only mode (*F)* is not constant but increases when the number of parallel working computers increases. This phenomenon is caused by these factors: the number of needed operations to split the population matrix increases (performed by client), data transfer among MDCE entities increases and the number of needed operations to assign the costs to the individuals increases (performed by client). If the computing requirements of cost function increase the speed up coefficient K_Z also increases except of the experiment no. 1 when the time consumption increases despite the fact that the number of computers increases. It is in conflict with formula (1). Analyzing this phenomenon it is caused by some reasons: if the *N* is big the management requirements of entire system are being increased, time delays increase because the task manager is able to communicate on physical layer only with one worker at given moment, the number of operations needed to split the population matrix increases and especially if the cost function is simple in comparison with the one in experiments 2, 3, 4 the total management time becomes dominant.

The maximum values of speed up coefficient K_Z cannot be estimated because *F* is not constant but it can be assumed that K_Z will start to decrease if number of cooperating computers increases. In described experiments this phenomenon has not been verified because only 9 computers were available. The parallelization effect dominates only when the calculation of the cost function consumes more computing power.

ACKNOWLEDGEMENT

This work has been supported by the Grant Agency VEGA of the Slovak Republic, grant No. 1/0023/08 "Theoretical apparatus for risk analysis and risk evaluation of transport telematic systems"

REFERENCES

[1] CORNE, D., DORIGO, M., GLOVER, F.: New Ideas in Optimization. McGraw-Hill, London 1999. ISBN 001-709506-5

[2] *VESTENICKÝ, M.: The Use of Evolutionary Algorithms in Selective Systems Synthesis for Communication Systems. Dissertation thesis, University of Žilina, Faculty of Electrical Engineering, 2007 (in Slovak)*

[3] *JELŠINA, M.: Architectures of Computing Systems. ELFA, Košice, 2002. ISBN 80-89066-40-2 (in Slovak)*

[4] *The MathWorks, Inc.: Parallel Computing Toolbox™ 4 User's Guide. [online], URL <http://www.mathworks.com/access/helpdesk/help/pdf_doc/distcomp/distcomp.pdf>*

[5] *LOHN, J. D., LINDEN, D. S., HORNBY, G. S., KRAUS, W. F., RODRÍGUEZ-ARROYO, A., SEUFERT, S.: Evolutionary Design of an X-Band Antenna for NASA's Space Technology 5 Mission. Proceeding of 2004 IEEE Antenna and Propagation Society International Symposium and USNC/URSI National Radio Science Meeting, Vol. 3, pp. 2313-2316, 2004. ISBN 0-7803-8302-8*

[6] *LOHN, J. D., COLOMBANO, S. P.: A Circuit Representation Technique for Automated Circuit Design. IEEE Transactions on Evolutionary Computation, Vol. 3, no. 3, 1999, pp. 205-219. ISSN 1089-778X*

[7] *KROKAVEC, D., FILASOVÁ, A.: Optimal Stochastic Systems. 2nd edition. Elfa, Košice, 2002. ISBN 80-89066-52-6 (in Slovak)*

[8] *HOTTMAR, V.: Network Model of The Processor System. Híradástechnika, info - communications - technology, vol. 60, no. 12/,2005, pp. 28-31. HU ISSN 0018 - 2008*

ANNALS OF FACULTY ENGINEERING HUNEDOARA
– INTERNATIONAL JOURNAL OF ENGINEERING