Performance Evaluation of the SOMA Asynchronously Parallel Distribution

Jan Kolek, Pavel Varacha and Ivo Motyl

***Abstract*—**Self-Organising Migrating Algorithm (SOMA) is a very effective tool of evolutionary optimization. This paper deals with performed evaluation of asynchronous parallel version of SOMA, based on the strategy AllToOne. The experiment was focused on the dependence of the solution quality (finding the global extreme value or approximating to global extreme value) on the number of used algorithm threads. In total, 10 different test functions were employed. Each function had 100 dimensions to optimize and the optimization was always repeated 100 times for 1,2,3 and 4 threats. Overlay 4000 independent runs of SOMA were rune and statistically evaluated.

***Keywords*—**SOMA, asynchronous, optimization, evolutionary algorithm, parallel, AllToOne.

I. Introduction

THE Self Organizing Migration Algorithm (SOMA) was created in 1999 and has many successful applications on the various problems of optimization, e.g. [1-5]. SOMA is ranked among evolution algorithms although there are not created new individuals during running the algorithm, in contradiction with typical evolution algorithms. It changes only the coordinates of individuals in the area of possible solution. More accurately, SOMA can be classified as memetic algorithm or among swarm algorithms.

SOMA is inspired by the intelligent behaviour of groups of individuals in the nature, e. g. while searching for food or finding the shortest way towards it. As it was already mentioned, the new induviduals are not created by hybridizing but the algorithm uses cooperative search of the area of possible solutions. That is why the evolution cycle that is called by other Genetic Algorithms „generation“ was renamed to „migration loop“. The run of the algorithm is influenced by settings of parameters indicated in Table 1.:

Parametr	Recommended range	Comment
PathLength	[1,1 ; 5>]	Control parameter
Step	[0,11 ; PathLengt]	Control parameter
PRT	[0 ; 1]	Control parameter
D	dimension	Dimension of the problem
PopSize	[10 ; define the user]	Control parameter
Migrations	[10 ; define the user]	Termination parameter
MinDiv	[± arbitrary ; define the user]	Termination parameter

Table 1.: Algorithm SOMA parameters

II. Performance evaluation of soma parallel imlementation

The aim of the experiment is a performance evaluation of asynchronous parallel algorithm SOMA based on classical (one-thread) version AllToOne. The algorithm SOMA was modified so that it not used only one thread but as many threads as available processor cores. The proposal and description of such SOMA implementation was firstly published and can be found in [6].

The asynchronous algorithm SOMA (contrary to synchronous version of the algorithm) does not wait for all individuals to finish their paths. If any individual gets better position than the current Leader, this individual becomes immediately the Leader and the other individuals continue migrating towards this new Leader immediately, not towards the original one. This arrangement significantly increases the opportunity to find global extreme value.

The experiment was performed on the personal computer from the ACER company with two-core processor AMD Athlon 4960 (64-bit) and the operating memory 3 GB DDR2 with operating system Windows Vista (64-bit).
The algorithm was tested for these 10 test functions, proposed as the benchmark in [7].

For better clarity the functions are shown in the Table 2.

Manuscript received July 30, 2012: Revised version received August 16, 2012. This paper is supported by the Internal Grant Agency at TBU in Zlin, project No. IGA/FAI/2012/041 and project No. IGA/FAI/2012/019 and by the European Regional Development Fund under the project CEBIA-Tech No. CZ.1.05/2.1.00/03.0089.

J. Kolek is with the Department of Informatics and Artificial Intelligence, Faculty of Applied Informatics, Tomas Bata University in Zlin, nam. T. G. Masaryka 5555, 760 01 Zlin Czech Republic (e-mail:. kolek@fai.utb.cz).

P. Varacha is with the Department of Informatics and Artificial Intelligence, Faculty of Applied Informatics, Tomas Bata University in Zlin, nam. T. G. Masaryka 5555, 760 01 Zlin Czech Republic (e-mail: varacha@fai.utb.cz).

I. Motyl is with the Department of Informatics and Artificial Intelligence, Faculty of Applied Informatics, Tomas Bata University in Zlin, nam. T. G. Masaryka 5555, 760 01 Zlin Czech Republic (e-mail: motyl@fai.utb.cz).

$$\sum_{i=1}^{Dim-1}(20+E-20E^{-0.2\sqrt{0.5(x_i^2+x_{i+1}^2)}}-E^{-0.5(\cos(2\pi x_i)+\cos(2\pi x_{i+1}))}) \quad (1)$$

$$\sum_{i=1}^{Dim-1}(-x_i\sin(\sqrt{|x_i-(x_{i+1}+47)|})-(x_{i+1}+47)\sin(\sqrt{\left|x_{i+1}+47+\frac{x_i}{2}\right|})) \quad (2)$$

$$-\sum_{i=1}^{Dim-1}(E^{\frac{-(x_i^2+x_{i+1}^2+0.5x_ix_{i+1})}{8}}\cos(4\sqrt{x_i^2+x_{i+1}^2+0.5x_ix_{i+1}})) \quad (3)$$

$$\sum_{i=1}^{Dim-1}(-1(\sin(x_i)\sin(\frac{x_i^2}{\pi})^{20}+\sin(x_{i+1})\sin(\frac{2x_{i+1}^2}{\pi})^{20})) \quad (4)$$

$$1+\sum_{i=1}^{Dim}\frac{x_i^2}{4000}-\prod_{i=1}^{Dim}\cos(\frac{x_i}{\sqrt{i}}) \quad (5)$$

$$\sum_{i=1}^{Dim-1}(x_i\sin(a)\cos(b)+(x_{i+1}+1)\sin(b)\cos(a)) \quad (6)$$

$$where\ a=\sqrt{|x_{i+1}+1-x_i|}\ \ and\ \ b=\sqrt{|x_{i+1}+1+x_i|}$$

$$(\mathbf{n}*10)\sum_{i=1}^{Dim}(x_i^2 - 10\cos(2\pi_i)) \quad (7)$$

$$\sum_{i=1}^{Dim-1}(100(x_i^2 - x_{i+1}^2)^2 + (1 - x_i^2)^2) \quad (8)$$

$$\sum_{i=1}^{Dim-1} - x_i \sin(\sqrt{|x_i|}) \quad (9)$$

$$-\sum_{i=1}^{Dim-1}(x_i^2 + x_{i+1}^2)^{0.25}(\sin(50(x_i^2 + x_{i+1}^2)^{0.1})^2 + 1) \quad (10)$$

Table 2.: Test functions mathematical formulation

All tests were performed in 100 dimensional space and the process of optimization was 100-times repeated. In all cases, new initial population was generated as starting point of the optimization.

Table 3. describes used borders and expected optimal results of used functions (1) – (10):

Function	Borders		Extreme(minimum)
	Minimum	Maximum	
Ackley (1)	-20	20	0
EggHolder (2)	-512	512	not known
Masters (3)	-50	50	0
Michalewicz (4)	-5	5	-1xD
Griewangk (5)	0	3	1,00089x(D-2)
Rana (6)	-512	512	not known
Rastrigin (7)	-5	5	-200xD

Rosenbrock (8)	-3	3	0
Schwefel (9)	-512	512	-418,983xD
Sine Wave (10)	-10	10	0

Table 3.: Test functions

III. Evaluation of the Dependence of Solution Quality on the Number of Used Threads

As the algorithm runs asynchronously and Leader can already be exchanged after migration of the first individual is finished (if this individual has better value of purpose-function than Leader), there is a hypothesis to be evaluated that that the number of threads has an influence on the quality of searching the area of possible solutions.

Following subchapters describe used SOMA control parameters and acquired experimental results for functions (1) - (10).

Function	PopSize	Step	PathLength
Ackley (1)	100	0,11	3
EggHolder (2)	60	0,11	3
Masters (3)	60	0,11	3
Michalewicz (4)	60	0,11	0,5
Griewangk (5)	100	0,11	3
Rana (6)	100	0,11	3
Rastrigin (7)	100	0,11	3
Rosenbrock (8)	60	0,011	0,5
Schwefel (9)	60	0,11	3
Sine Wave (10)	100	0,11	3

Table 4.: Parameters for test

A. *Ackley*

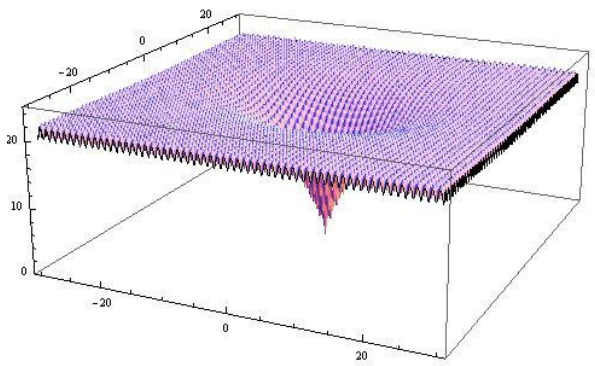

Fig. 1.: Ackley function (1)

As can be seen in Fig. 1., the function has very complex space to searched. Fig. 2. shows that the solution quality decreases with the increasing number of used threads. When using only one thread the algorithm did not find the global minimum (0), it has just approximated to global minimum value 232.92. When using 8 threads the algorithm stopped when reaching the value 451.32.

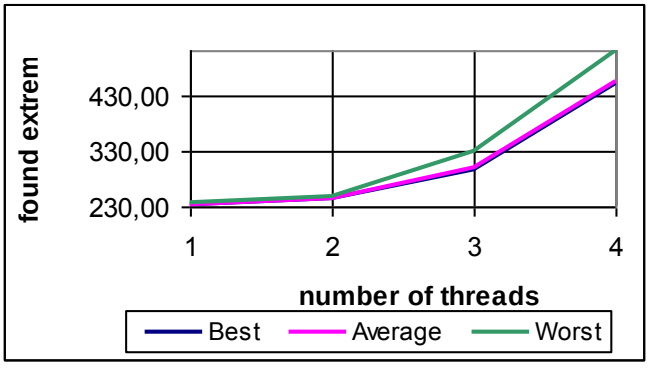

Fig. 2.: Dependence of solution quality on the number of used threads for Ackley (1)

B. *Egg Holder*

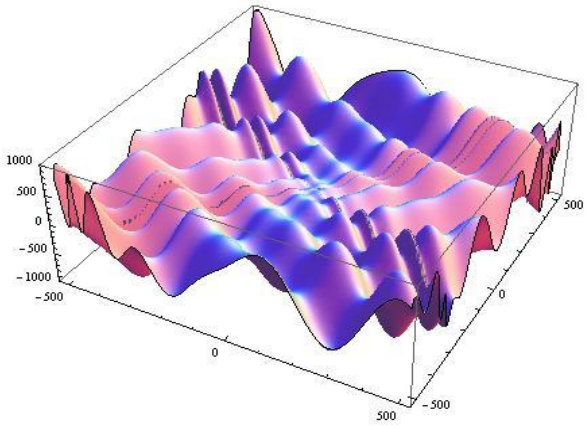

Fig. 3.: Egg Holder function (2)

From the Fig. 4 it is obvious that best results were provided by the employment of only one threat. Although exact value of global extreme (minimum) for this function is not known, in this experiment -57844.6618 was found as best result. With increasing number of the used threats quality of obtained solution considerably decreased. While using four threats only -30335.6880 was found as a possible global minimum.

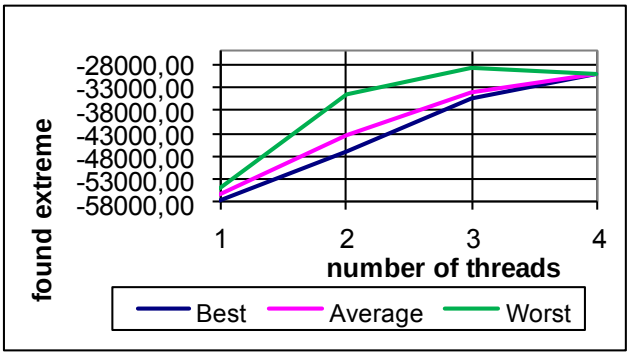

Fig. 4.: Dependence of solution quality on the number of used threads for Egg Holder (2)

C. *Masters*

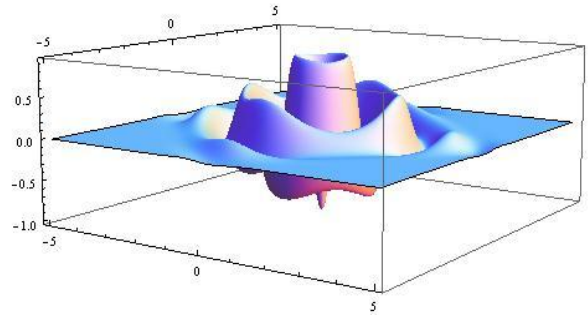

Fig. 5.: Masters function (3)

Experimental testing of the algorithm based on Masters test function provides different results in comparison with previous three functions. While measuring the algorithm performance for 1 and 2 threats best and worsts results were almost identical so founded solutions were close to equality

On the other hand for 3 and 4 threats the algorithm was not so effective and it diverged comparatively fare form global minimum. Best founded solution was -91.5888 for 1 threat only while the worst was -54.9272 running on 8 threats. Real global extreme of this function for 100 dimensions is -100.

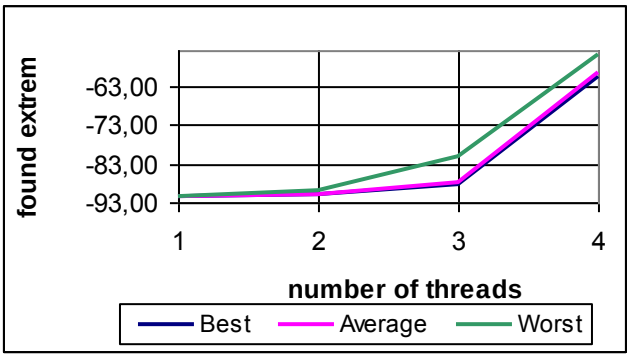

Fig. 6.: Dependence of solution quality on the number of used threads for Masters (3)

D. *Michalewicz*

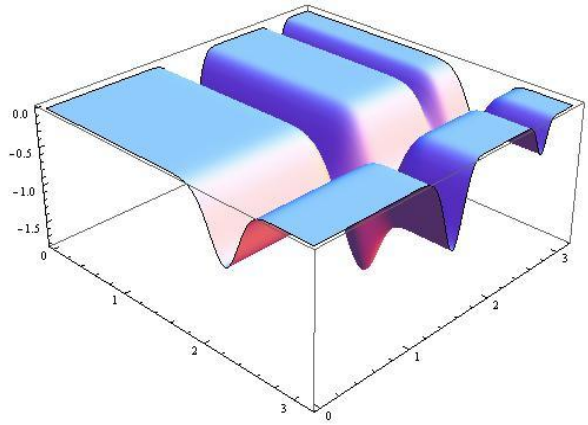

Fig. 7.: Michalewicz function (4)

This function is different from the others, it contains a flat surface. That is a problem for all deterministic methods. Also it took much longer to evaluate this function than the other functions.

Michalewicz function is the only function which had better solution accuracy when increasing the number of threads. When using one thread the algorithm found the solution -89.09 but when using eight threads it found minimum value 90.15.

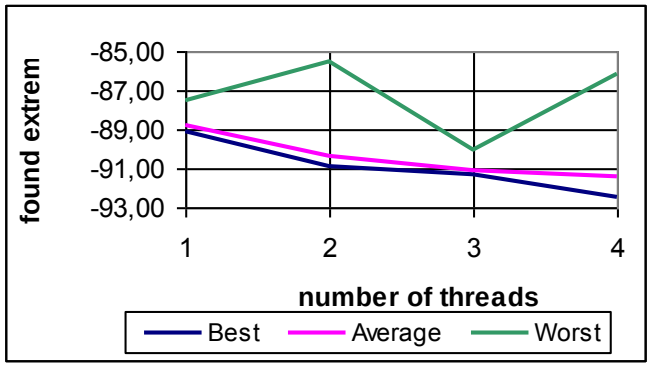

Fig. 8.: Dependence of solution quality on the number of used threads for Michalewicz (4)

E. Griewangk

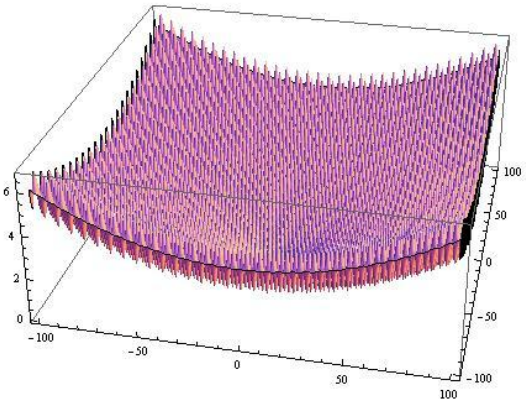

Fig. 9.: Griewangk function (5)

Global extreme of Griewangk function is 0. Surface of this function results is very structurally complicated so a number of individuals in the population was set equal to the function dimension. (PopSize = 100). The experiment provided for this function similar results as in previous case.

The best results were found for 1 threat (0.0006). The nest and worst founded solution were practically even in this case so the whole population ended around global extreme. For two employed threats the acquired solution quality was only slightly worst and it start decreasing with usage of additional threats.

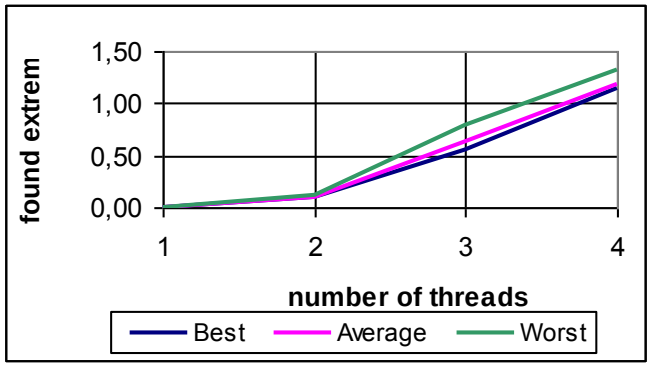

Fig. 10.: Dependence of solution quality on the number of used threads for Griewangk (5)

F. Rana

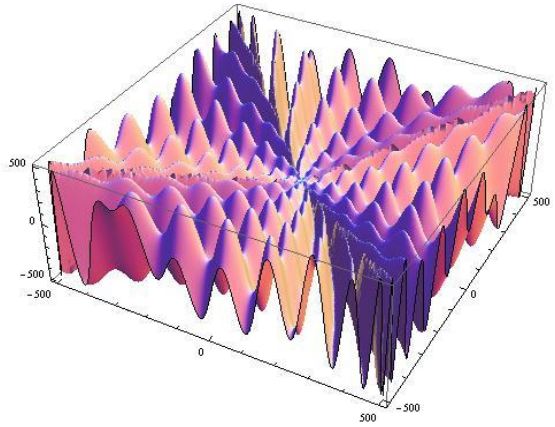

Fig. 11.: Rana function (6)

Similarly to Griewangk, Rana test function has very complicated structure so the number of the population was again set equal to the function dimension. For this function the experiments also shown that quality of founded solution is decreasing with increasing number of employed threats. The best founded solution was -32709.8767 for 1 threat and was 18490.2969 for 8 threats. (The exact value of global extreme is for this function unknown.)

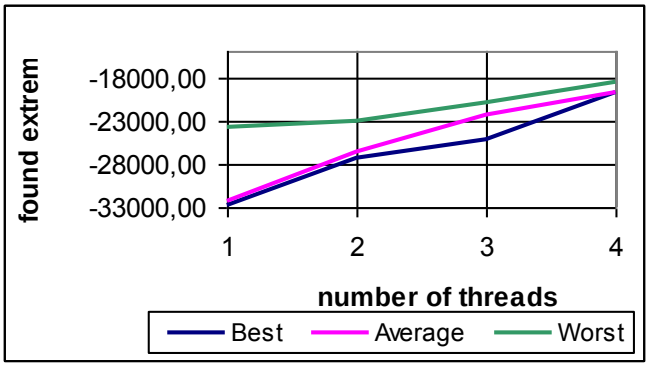

Fig. 12.: Dependence of solution quality on the number of used threads for Rana (6)

G. Rastrigin

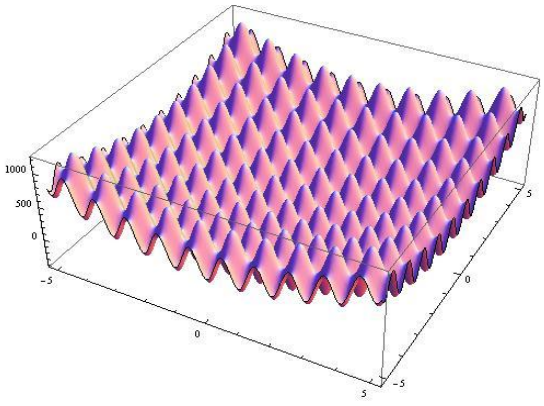

Fig. 13.: Rastrigin function (7)

In the case of Rastigin function, which also very complicated one as can be seen on Fig. 13., the algorithm performance for 1, 2 and 3 threats was almost similar. For 2 threats the local extreme with cost value -194313.6487 was founded. This solution is not far from the number -200000 which is know global extreme (minimum) of the function in 100 dimensional space. The population will be probably able to reach global extreme if larger amount of cost function evaluation is available.

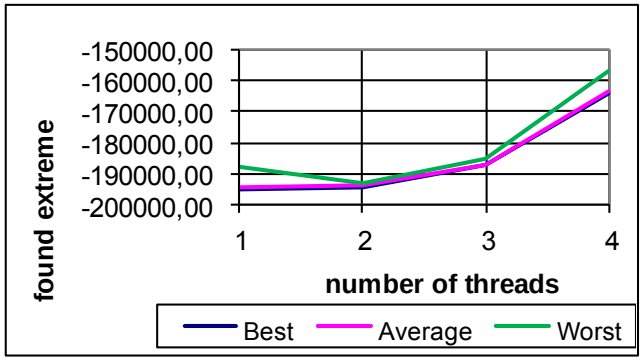

Fig. 14.: Dependence of solution quality on the number of used threads for Rastrigin (7)

H. Rosenbrock

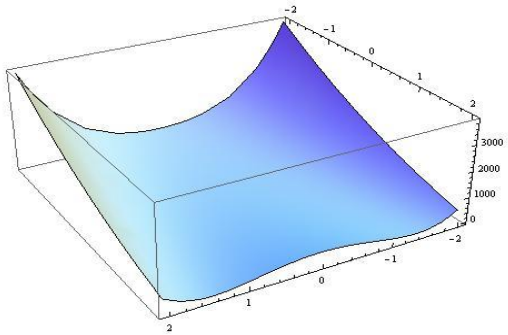

Fig. 15.: Rosenbrock function (8)

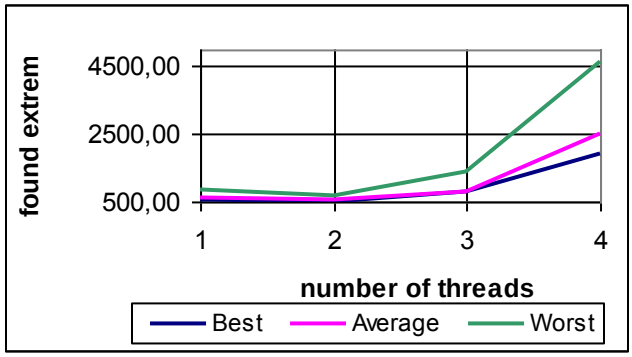

Fig. 16.: Dependence of solution quality on the number of used threads for Rosenbrock (8)

Experiments on the Rosenberk function were inconclusive. This function in contrast with others is not too much complicated so founding of global extreme was expected. Nevertheless this was not the case even though the step control parameter was set to 0.011 in contrast with other experiments. This was due to small search interval issue.

These results were probably caused by the fact that starting population is always randomly generated. Even if the algorithm was able to found better and better solution it does not reach global extreme (0).

The best solution was founded for 2 threats 497.4972. While running on 4 threats the algorithm even ends on the number 1895.2562.

I. Schwefel

The global extreme (minimum) of Schwefel function is -41898,3. This function belongs between those with less complicated structure even if it is multimodal function. Experiment on this function provided best results for 2 threats, closely followed by 1 threat performance, similarly with precious case. The best founded solution was -41540.0430. For 4 and 8 threats the quality of solution again decreased (-39789.0647 and -37157.8522). The algorithm probably ends in local extreme in the other side of searched space.

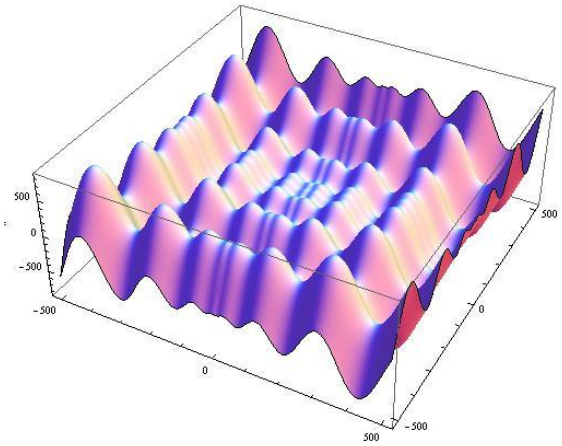

Fig. 17.: Schwefel function (9)

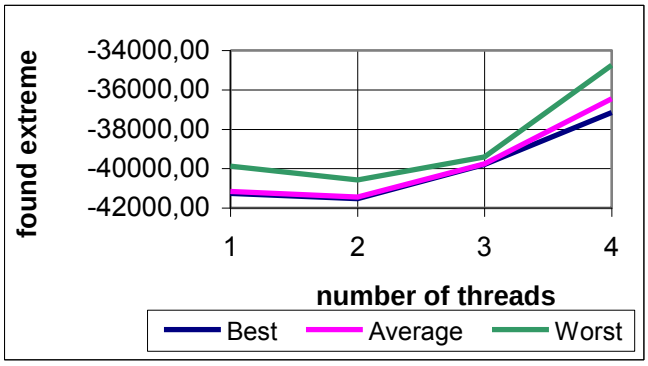

Fig. 18.: Dependence of solution quality on the number of used threads for Schwefel (9)

J. *Sine Wave*

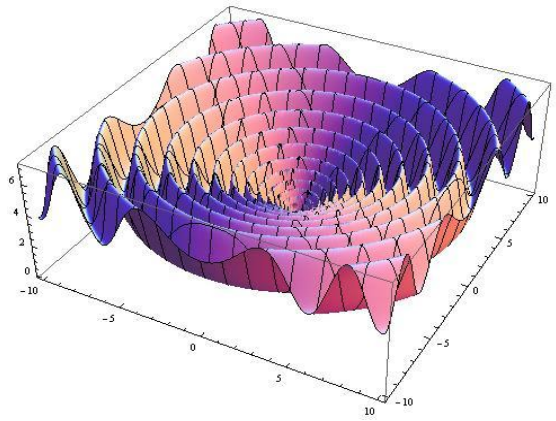

Fig. 19.: Sine Wave function (10)

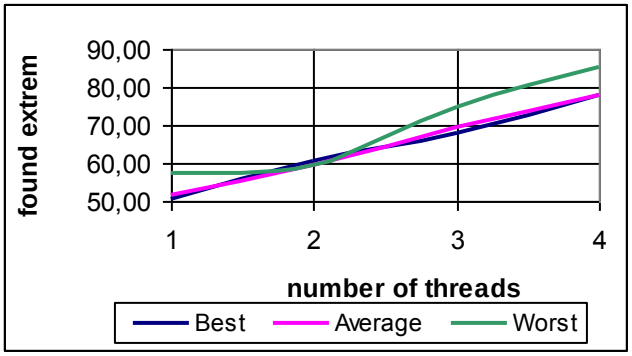

Fig. 20.: Dependence of solution quality on the number of used threads for Sine wave (10)

Experiments considering Sine Wave function have similar development as several cases on the beginning of the chapter. With increasing number of employed threats the overall quality of obtained solution decreased. The best founded solution was 50.3595. The global extreme for 100 dimensional case of the function is 0.

IV. Conclusion

Evaluation was performed on ten functions indicated in Table 2.

Within testing dependence of solution quality on the number of used threads, the solution quality decreased with increasing number of the used threads. The only exception was the function Michalewicz mentioned above where the solution quality increased with the increasing number of used threads. It was probably caused by the fact that this function contains too many straight surfaces.

The total result of 100 repeating demonstrates that the solution quality is not very dependent on the number of used threads. The determinative factor is generating initial (starting) population.

In the tables 5, 6, and 7, the overall results of the algorithm for 1, 2, 3 and 4 employed threats are visible. These tables show perceptual performance of the algorithm for different number of threats in accordance with the performance of 1 threat only. It is obvious that the algorithm frequently lose its performance with increased number of employed threats.

However in several cases the algorithm performance even increases while more threats employed. This interesting behavior should by further studied and experimental evaluated.

Function	1 thread	2 threads	3 threads	4 threads
Ackley (1)	100 %	95,3 %	78,3 %	51,6 %
EggHolder (2)	100 %	81,1 %	60,9 %	52,4 %
Masters (3)	100 %	99,5 %	96,3 %	66,5 %
Michalewicz (4)	100 %	102,0 %	102,5 %	103,8 %
Griewangk (5)	100 %	0,6 %	0,1 %	0,0 %
Rana (6)	100 %	83,8 %	77,2 %	60,6 %
Rastrigin (7)	100 %	99,7 %	96,2 %	84,3 %
Rosenbrock (8)	100 %	117,5 %	75,5 %	30,9 %
Schwefel (9)	100 %	100,7 %	96,4 %	90,0 %
Sine Wave (10)	100 %	83,1 %	74,1 %	64,8 %

Table 5.: Results of test for best

Function	1 thread	2 threads	3 threads	4 threads
Ackley (1)	100 %	94,9 %	77,9 %	51,3 %
EggHolder (2)	100 %	77,0 %	60,6 %	53,9 %
Masters (3)	100 %	99,5 %	95,9 %	65,4 %
Michalewicz (4)	100 %	101,8 %	102,5 %	102,9 %
Griewangk (5)	100 %	0,9 %	0,1 %	0,1 %
Rana (6)	100 %	82,2 %	69,1 %	60,7 %
Rastrigin (7)	100 %	99,7 %	96,4 %	83,9 %
Rosenbrock (8)	100 %	104,4 %	75,8 %	24,2 %
Schwefel (9)	100 %	100,7 %	96,6 %	88,6 %
Sine Wave (10)	100 %	86,1 %	74,0 %	65,9 %

Table 6.: Results of test for average

Function	1 thread	2 threads	3 threads	4 threads
Ackley (1)	100 %	95,3 %	72,0 %	46,4 %
EggHolder (2)	100 %	62,8 %	52,6 %	55, 3 %
Masters (3)	100 %	98,5 %	89,0 %	60,2 %
Michalewicz (4)	100 %	97,7 %	102,9 %	98,4 %
Griewangk (5)	100 %	2,3 %	0,4 %	0,2 %
Rana (6)	100 %	97,5 %	87,8 %	78,0 %
Rastrigin (7)	100 %	102,9 %	98,8 %	83,3 %
Rosenbrock (8)	100 %	126,2 %	62,8 %	18,5 %
Schwefel (9)	100 %	101,8 %	98,8 %	87,2 %
Sine Wave (10)	100 %	96,5 %	76,4 %	67,0 %

Table 7.: Results of test for worst

REFERENCES

[1] SENKERIK R., OPLATKOVA Z., ZELINKA I., DAVENDRA D. Synthesis of feedback controller for three selected chaotic systems by means of evolutionary techniques: Analytic programming, Mathematical and Computer Modelling, Available online 27 May 2011, ISSN 0895-7177, 10.1016/j.mcm.2011.05.030.

[2] CHRAMCOV B., BERAN P., DANÍČEK L., JAŠEK R.. A simulation approach to achieving more efficient production systems. International Journal of Mathematics and Computers in Simulations, 2011, year 5, issue 4, page 299-309. ISSN 1998-0159.

[3] KRÁL E., VAŠEK, L., DOLINAY V., ČÁPEK P. Usage of Peak Functions in Heat Load Modeling of District Heating System. In Recent Researches in Automatic Control. Montreux : WSEAS Press, 2011, p. 404-406. ISBN 978-1-61804-004-6.

[4] POSPÍŠILÍK M., KOUŘIL L., MOTÝL I., ADÁMEK M. Single and Double Layer Spiral Planar Inductors Optimisation with the Aid of Self-Organising Migrating Algorithm. In Proceedings of the 11th WSEAS International Conference on Signal Processing, Computational Geometry and Artificial Vision. Venice : WSEAS Press (IT), 2011, p. 272 - 277. ISBN 978-1-61804-027-5

[5] ZELINKA I., Studies in Fuzziness and Soft Computing, New York : Springer-Verlag, 2004.

[6] VAŘACHA P. Innovative Strategy of SOMA Control Parameter Setting. In 12th WSEAS International Conference on Neural Networks, Fuzzy Systems, Evolutionary Computing & Automation. Timisoara : WSEAS press, 2011, p. 70-75. ISBN 978-960-474-292-9.