DOI: 10.14529/jsfi180401

Adaptive Load Balancing in the Modified Mind Evolutionary Computation Algorithm*

***Maxim K. Sakharov*[1]*, Anatoly P. Karpenko*[1]**

The paper presents an adaptive load balancing method for the modified parallel Mind Evolutionary Computation (*MEC*) algorithm. The proposed method takes into account an objective function's topology utilizing the information obtained during the landscape analysis stage as well as the information on available computational resources. The modified *MEC* algorithm and proposed static load balancing method are designed for loosely coupled parallel computing systems and imply a minimal number of interactions between computational nodes when solving global optimization problems. A description of the proposed method is presented in this work along with the results of computational experiments, which were carried out with a use of multi–dimensional benchmark functions of various classes. Obtained results demonstrate that an effective use of available computational resources in the proposed method helps finding a better solution comparing to the traditional parallel *MEC* algorithm balancing. Further development of the proposed method requires more advanced termination criteria in order to avoid excessive iterations.

Keywords: load balancing, landscape analysis, mind evolutionary computation, global optimization.

Introduction

One of the distinct features of the real–world global optimization problems is the computational complexity of the objective functions. To cope with such problems within reasonable time, it is necessary to utilize parallel computing systems. Grid systems, made of personal computers and workstations, are widely used by scientific communities [8, 20]. Their availability is caused by a relatively low cost and easy scaling. Such systems belong to a class of loosely coupled computing systems.

When solving an optimization problem on parallel computing systems in general, and on loosely coupled systems in particular, one of the main difficulties is the optimal mapping problem [4, 20] is how to distribute groups of sub–problems over processors. It should be noted that a problem of optimal mapping of computational processes onto a parallel computing system is one of the main issues associated with parallel computations. It is well known that such a problem is NP complete and can be solved with exact methods within a very narrow class of problems [4, 5].

Most often, methods of load balancing are used to obtain an approximate solution to an optimal mapping problem. The main idea behind these methods is to distribute the computation tasks over the processors in such a way that the total computing and communication load is approximately the same for each processor.

There are two types of load balancing: static and dynamic. Static balancing is performed once, before the computational process starts. When computational complexity of the objective functions is unknown in advance, static load balancing can't be effective. In this case, dynamic load balancing should be implemented during calculation: the algorithm must re–distribute computation tasks between processors depending on their loading.

[*] The paper is recommended for publication by the Program Committee of the International Scientific Conference "Parallel computational technologies (PCT) 2018".

[1] Bauman MSTU, Moscow, Russian Federation

Regarding loosely coupled computational systems, it is required to minimize the interaction between computational nodes, hence re–distributing tasks during the calculations is highly inefficient. In this paper, we propose to use the landscape analysis (*LA*) [1, 12, 19] in order to evaluate computational complexity of objective functions and thus improve the efficiency of static load balancing.

Nowadays, the *LA* methods are widely used in global optimization problems to determine some distinct features of objective functions and, subsequently, classify these functions, for example, to adjust some free parameters of the utilized population algorithms [10]. Population algorithms represent a universal and powerful tool for solving global optimization problems. Their popularity is caused by the fact that they can be easily implemented and applied to various fields as they are based on the universal idea of evolution. However, their efficiency heavily depends on numerical values of their free parameters, which should be selected based on the characteristics of a problem in hand.

Within the landscape analysis approach, it is proposed to extract information on the objective function's landscape and topology at the cost of additional trials (approximately 1–10% of the total number of evaluations) [13]. In other words, the *LA* methods help identifying the search for sub–domains with either rugged or smooth topology, or for sub–domains where the objective function's values are almost identical, etc.

A class of Mind Evolutionary Computation (*MEC*) algorithms [2, 6, 7] is considered in this work. These algorithms belong to a family of population methods inspired by human society. Individual s is considered as an intelligent agent which operates in group S made of analogous individuals. During the evolution process, each individual is affected by other individuals within the group. This simulates the following logics. In order to achieve a high position within a group, an individual has to learn from the most successful individuals in this group. Groups themselves should follow the same principle to stay alive in the intergroup competition.

This work deals with the Simple *MEC* (*SMEC*) algorithm. It belongs to a class of *MEC* algorithms and was selected for the research because it is highly suitable for parallel computations, especially for loosely coupled systems. Generally, in order to be efficient on loosely coupled systems, the basic optimization algorithm must imply a minimum number of interactions between sub–populations which evolve on separate computational nodes. Only a few currently known population–based algorithms, including the *MEC* algorithm, meet this requirement.

The authors propose a modified parallel *MEC* algorithm with an incorporated *LA* procedure accompanied with the static load balancing method. Pperformance research was carried out to evaluate efficiency of the proposed approach in comparison with the traditional parallel *SMEC* algorithm [15, 17].

1. Problem Statement and the Basic Optimization Algorithm

A global deterministic unconstrained minimization problem is considered in this work

$$\min_{X \in \mathrm{R}^{|X|}} \Phi(X) = \Phi(X^*) = \Phi^*, \tag{1}$$

where $\Phi(X)$ is the scalar objective function; $\Phi(X^*) = \Phi^*$ is the required minimal value; $X^* = (x_1, x_2, , x_{|X|})$ is $|X|$–dimensional vector of variables; $R^{|X|}$ is $|X|$–dimensional arithmetical space. Domain D_0 is defined as follows:

$$D_0 = \left\{ X | x^{min} \le x_i \le x^{max}, i \in [1 : |X|] \right\} \tag{2}$$

and used for generating the initial population of solutions. The *MEC* algorithm can be considered as a multi–population one. A multi–population consists of independent sub–populations with different instances of the *SMEC* algorithm. Each sub–population is made of leading groups $S^b = (S_1^b, S_2^b, , S_{|S^b|}^b)$ and lagging groups $S^w = (S_1^w, S_2^w, , S_{|S^w|}^w)$. The number of individuals within each group is set to be the same and equals $|S|$. Every group S_i^b or S_j^w has its own communication environment called a local blackboard and denoted C_i^b or C_j^w correspondingly. In addition, the whole sub–population $S = \{S^b, S^w\}$ has a general global blackboard C^g. The original *MEC* algorithm was presented in [2] and was named the simple Mind Evolutionary Computation algorithm (Simple *MEC*, *SMEC*). The *SMEC* algorithm is based on the operations of group initialization, similar-taxis and dissimilation. Similar-taxis and dissimilation are iteratively repeated until there is an increase in the maximum score of the leading groups. When the growth of this indicator stops, the current solution of a problem is declared a global minimum. By individuals score we mean the value of objective function $\Phi(X)$ in its current position [8].

In [16], the authors carried out a wide performance research of the *SMEC* algorithms efficiency depending on the values of the following free parameters: standard deviation σ, used for the generation of new individuals; removing frequency of lagging groups ω; ratio η between the number of leading $|S^b|$ and lagging $|S^w|$ groups in a sub–population. Obtained results were used to formulate the strategies for selecting optimal numerical values for those parameters.

2. Modified SMEC Algorithm

The modified Mind Evolutionary Computation algorithm is based on hybridization of the multi–population *SMEC* algorithm with the incorporated landscape analysis stage. The proposed *LA* method allows one to study the objective functions topology without any additional information on the problem in hand known a priori. Based on the results obtained during the landscape analysis, we can classify an objective function into one of six possible categories and form an adaptation strategy for the algorithm [14, 16]. To achieve this, the initialization stage of the *SMEC* algorithm was modified. New initialization stage with the incorporated *LA* procedure can be described as follows.

1. Generate N quasi–random $|X|$–dimensional vectors within domain D_0. Here, N is the total number of all groups in a multi–population (free parameter of the algorithm).
2. For every X_r, $r \in [1\colon N]$, calculate the corresponding values of objective function Φ_r and sort those vectors in ascending order of values Φ_r, $r \in [r\colon N]$.
3. Equally divide a set of vectors $(X_1, X_2, \ldots, X_N)$ into $|K|$ sub–populations (another free parameter).
4. For every sub–population K_l, $l \in [1\colon |K|]$, calculate a value of its diameter d_l, the maximum Euclidean distance between any two individuals.
5. Build a linear approximation for the dependency of diameter $d(l)$ on group number l, using the least squares method [14].
6. Calculate an estimation of the size of domain D_0 using the formula

$$d_0 = \sqrt{|X|(x^{max} - x^{min})^2}.$$

7. Classify objective function $\Phi(X)$ into one of the six categories provided in Tab. 1 based on the calculated parameters and determine the corresponding numerical values for the

SMEC algorithm for each sub–population. Numerical parameters in Tab. 1 are based on the empirical results [16].

Table 1. Classification of objective functions based on the *LA* results

	$d(l)$ increases	$d(l)$ constant	$d(l)$ decreases
$\frac{d_0}{d_1} > 2.5$	Nested sub–populations with dense first sub–population (category I)	Non–intersected sub–populations of the same size (category III)	Distributed sub–populations with potential minima (category V)
$\frac{d_0}{d_1} \leq 2.5$	Nested sub–populations with sparse first sub–populations (category II)	Intersected sub–populations of the same size (category IV)	Highly distributed sub–populations with potential minima (category VI)

In Fig. 1, an example of calculating a diameter is demonstrated for the first sub–population of individuals generated for two–dimensional Composition Function 1 from CEC 14 [11]. Circles represent an arbitrary neighborhood of the individuals; different colors of those circles correspond to different sub–populations. Additional examples of the landscape analysis procedure for various Composition Functions from the same set of benchmark problems are presented in Fig. 2.

Each classification category describes a specific topology of an objective function with the following pre–determined rules for calculating numerical values of the free parameters.

1. For objective functions from categories I and II, there is a high probability to find a global optimum within sub–population K_1. In such a case, numerical values for parameter σ are defined to increase the search intensification for the first sub–populations and the search diversification for the last sub–populations:

$$\sigma(l) = 0.25 + 0.75\frac{(l-1)}{(|K|-1)}.$$

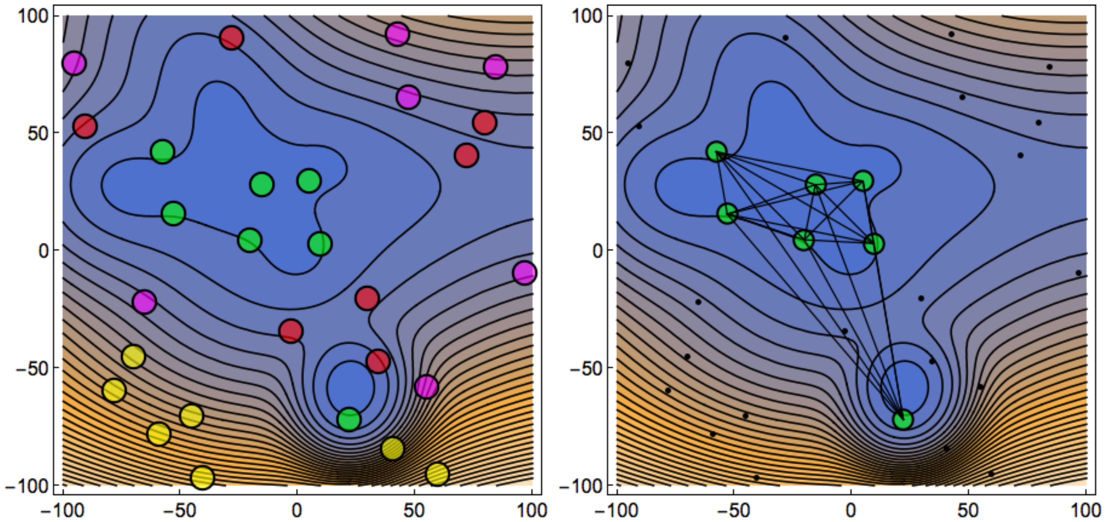
2. For objective functions from categories III and IV, numerical values for parameter σ are defined randomly from a specified range:

$$\sigma(l) = rand(0.1, 0.9).$$

3. For objective functions from categories V and VI, the first sub–population usually covers a large part of initial domain D_0. In such a case, the goal is to increase the search diversification for the first sub–populations:

$$\sigma(l) = 1 - 0.75\frac{(l-1)}{(|K|-1)}.$$

4. For categories with odd numbers, the removing frequency of lagging groups $\omega = 20$, while ratio between the number of leading and lagging groups in a sub–population $\eta = 50$. For categories with even numbers: $\eta = 75$, and $\omega = 25$ in order to provide enough time for lagging groups in a sub–population to explore their search sub–domains.



(a) Distribution of individuals for four sub–populations

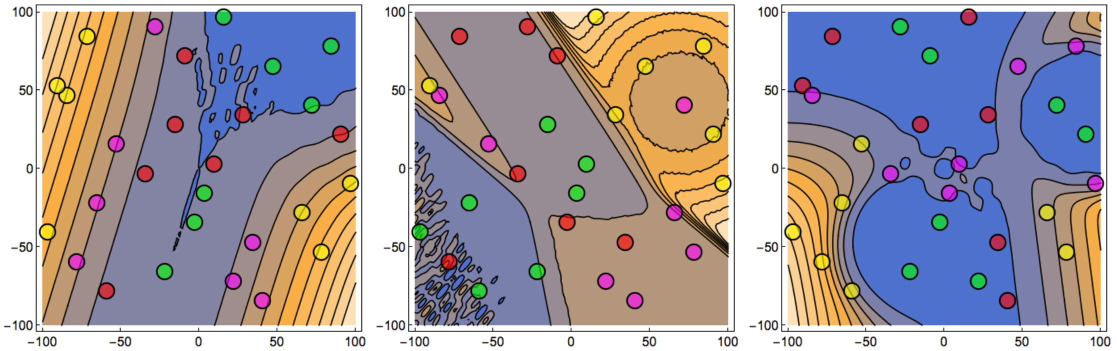
(b) Determining a diameter of the first sub–population

Figure 1. Determining a diameter of the first sub–population for the benchmark Composition Function 1 from CEC 14

3. Adaptive Load Balancing

As mentioned above, parallel optimization methods should be used in order to efficiently solve the real–world global optimization problems. Multi–population version of the *SMEC* algorithm can be naturally used to decompose the problem and map it onto computational nodes of the loosely coupled system. In such a case, each sub–population or a group of sub–populations evolves independently on separate computational nodes. This approach lies behind the idea of the traditional Parallel *MEC* algorithm and was used by the authors in [9] in order to design multi–memetic algorithms.

When performing landscape analysis, it is reasonable to use all extracted information in order to use available computational resources as efficiently as possible and, subsequently, increase the efficiency of the algorithm in general.



(a) Composition function 3 from CEC 14 (Group I)

(b) Composition function 4 from CEC 14 (Group II)

(c) Composition function 5 from CEC 14 (Group VI)

Figure 2. Results of the landscape analysis procedure for a few benchmark functions

We modified the initialization stage described above so that during step 2, the time required for calculations t_r is measured together with calculating the values of objective function Φ_r. The proposed adaptive load balancing method can be described as follows.

1. For each sub–population K_l, $l[1 : |K|]$, we analyzed all time measurements t_r for the corresponding vectors X_r, $r \in [1 : \frac{N}{|K|}]$ in order to determine whether there are outliers or not. All found outliers t^* are excluded from sub–populations. A new sub–population is composed from those outliers; it can be studied upon users request after the computational process is over.
2. If the number of sub–populations $|K|$ is equal to the number of available computational nodes M, then we send the first sub–population K_1 to one node. Individuals in other sub–populations are re–distributed between neighboring sub–populations starting from K_2, so that the average calculation time would be approximately the same for every sub–populations. Balanced sub–populations K_l, $l \in [2 : |K|]$ are then mapped onto the computational nodes. Then we go to step 5.
3. If the number of sub–populations $|K|$ is greater than the number of available computational nodes M, then we send the first sub–population K_1 to one node. The rest sub–populations are divided into equal groups of sizes $\frac{(|K|-1)}{(M-1)}$ in ascending order; the last group contains all remaining sub–populations.
4. Individuals in sub–populations are re–distributed between neighboring sub–populations starting from K_2, so that the average calculation time would be approximately the same for every sub–populations. Balanced sub–populations K_l, $l \in [2 : |K|]$ are then mapped onto the computational nodes.
5. The modified *SMEC* algorithm is launched on each node with the specific values of free parameters in accordance with Tab. 1.

4. Performance Investigation

The modified *SMEC* algorithm and the proposed load balancing method were implemented in Wolfram Mathematica. A set of numerical experiments with the use of multi–dimensional ($|X| = 16$) benchmark functions of various classes [3] was carried out to estimate performance of the proposed approach. During the study, efficiency of the parallel modified *SMEC* algorithm with the adaptive load balancing (*SMEC/LA*) and the parallel multi–population *SMEC* algorithm without the landscape analysis were compared.

The benchmark optimization functions considered in this paper are presented in Tab. 2 along with their known global optimal values. An original domain for generating initial population equals

$$D_0 = \{X : -5 \leq x_i \leq 5, i \in [1 : |X|]\}.$$

All numerical experiments were carried out using the multi–start method with 100 launches. The best obtained value of objective function Φ^* as well as its average value $\bar{\Phi}$ based on the results of all launches were utilized as the performance indices for comparison of the two algorithms and their software implementations along with maximum iteration number N_{max} among all computational nodes and the number or iterations N_{opt} from the node, where the optimal value was obtained.

In this work, LP_τ–sequence was used to generate the initial vectors for the LA procedure, as it provides a uniform coverage of the search domain [18]. Other parameters had the following values for both algorithms: the total number of groups in one sub–population $\gamma = 40$; the number of individuals in each group $|S| = 40$; the number of sub–populations $|K| = 8$.

The number of stagnation iterations $\lambda_{stop} = 50$ was used as a termination criterion for the algorithms. Tolerance used for identifying stagnation was equal to $\epsilon = 10^{-5}$. All computations were performed with a use of computational network made of $M = 8$ computational nodes and one master node. All computational nodes did not communicate with each other. Each node represents a personal computer with Intel Core i5–6600 CPU and 8GB RAM.

Table 2. Definitions of benchmark functions

Function	Definition	Global minimum
Griewank	$\Phi_1(X) = \sum_{i=1}^{\lvert X \rvert} x_i^2 4000 - \prod_{i=1}^{\lvert X \rvert} \cos\left(x_i \sqrt{i}\right) + 1$	$\Phi_1(X^*) = 0$ $X^* = (0, \ldots, 0)$
Rastrigin	$\Phi_2(X) = \sum_{i=1}^{\lvert X \rvert}(10 + x_i^2 - 10\cos(2\pi x_i))$	$\Phi_2(X^*) = 0$ $X^* = (0, \ldots, 0)$
Rosenbrock	$\Phi_3(X) = \sum_{i=1}^{\lvert X \rvert}\left(100(x_{i+1} - x_i^2)^2 + (1 - x_i^2)\right)$	$\Phi_3(X^*) = 0$ $X^* = (1, \ldots, 1)$
Zakharov	$\Phi_4(X) = \sum_{i=1}^{\lvert X \rvert} x_i^2 + \left(\sum_{i=1}^{\lvert X \rvert} 0.5\ i\ x_i\right)^2 + \left(\sum_{i=1}^{\lvert X \rvert} 0.5\ i\ x_i\right)^4$	$\Phi_4(X^*) = 0$ $X^* = (0, \ldots, 0)$

The obtained results (Tab. 3) demonstrate superiority of the proposed *SMEC/LA* algorithm in comparison with the simple parallel *MEC* in terms of accuracy.

Table 3. Results of numerical experiments

Function	*SMEC*	*SMEC/LA*
Griewank	$\bar{\Phi}_1 = 1.63\text{E}{-2}$	$\bar{\Phi}_1 = 2.44\text{E}{-5}$
	$\Phi_1^* = 9.19\text{E}{-3}$	$\Phi_1^* = 1.21\text{E}{-5}$
Rastrigin	$\bar{\Phi}_2 = 7.27\text{E}{+1}$	$\bar{\Phi}_2 = 7.69\text{E}{-3}$
	$\Phi_2^* = 5.25\text{E}{+1}$	$\Phi_2^* = 3.54\text{E}{-6}$
Rosenbrock	$\bar{\Phi}_3 = 2.73\text{E}{+1}$	$\bar{\Phi}_3 = 8.91\text{E}{-1}$
	$\Phi_3^* = 2.12\text{E}{+1}$	$\Phi_3^* = 9.45\text{E}{-2}$
Zakharov	$\bar{\Phi}_4 = 2.79\text{E}{+1}$	$\bar{\Phi}_4 = 1.16\text{E}{+0}$
	$\Phi_4^* = 2.10\text{E}{+1}$	$\Phi_4^* = 1.13\text{E}{-1}$

For all benchmark functions, results obtained with the use of *SMEC/LA* are better than ones obtained using just parallel *SMEC* by several orders of magnitude both for average values $\bar{\Phi}$ and least found values Φ^*.

On the other hand, the *SMEC/LA* algorithm requires more iterations N_{max} than *SMEC* algorithm (Fig. 3). This can be explained by the fact that both parallel algorithms wait until all computations are over; and in case of the *SMEC/LA* algorithm, it contains groups made of individuals with large values of the objective function. Such a group can require many iterations to converge to some local optimum. The comparison of the number of iterations N_{opt} required to find a global optimum proves our assumption (Fig. 4). The results demonstrate that the global optimum with the use of *SMEC/LA* method can be found relatively rapidly and, therefore, some unnecessary iterations can be avoided with the use of more advanced termination criteria.

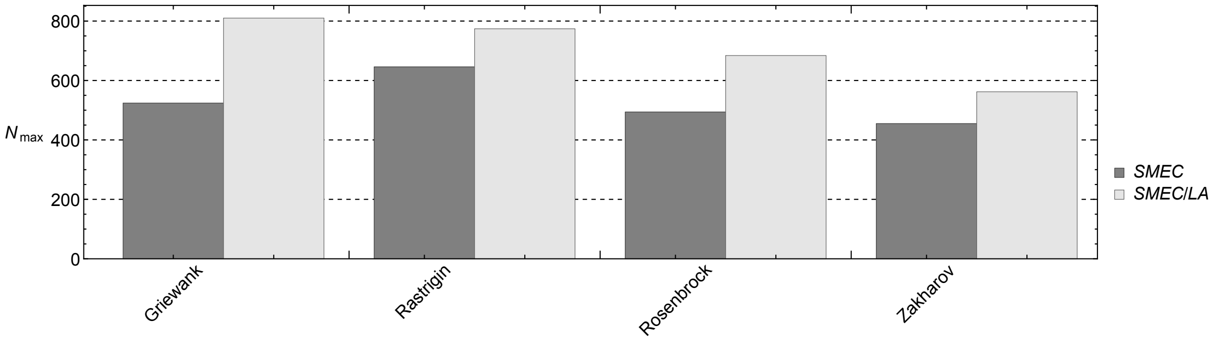


Figure 3. Overall iteration number N_{max} for different benchmark functions

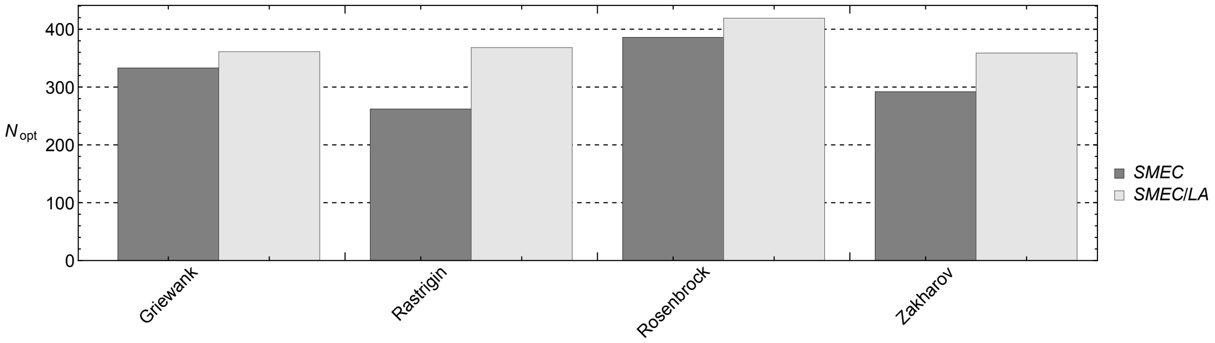


Figure 4. Iteration number N_{opt} required to find a global optimum for different benchmark functions

Conclusions

This paper presents a new adaptive load balancing method designed for the multi–population *SMEC* algorithm with the incorporated landscape analysis procedure. The algorithm is capable of adapting to various objective functions using an adaptation based on the results of objective functions classification.

Performance investigation was carried out in this work with the use of multi–dimensional benchmark optimization functions of various classes. The proposed algorithm along with the proposed load balancing technique was capable of localizing all known global optima with high precision. The parallel algorithm worked in the synchronous mode; that led to a significant increase in the maximum number of iterations. This disadvantage can be overcome using either asynchronous mode or some advanced termination criteria.

Further research will be devoted to a more extensive study of various objective functions and their classification as well as the efficiency of adaptation.

References

1. Bischl, B., Mersmann, O., Trautmann, H., Preub, M.: Algorithm selection based on exploratory landscape analysis and cost-sensitive learning. In: Proceedings of the fourteenth

international conference on Genetic and evolutionary computation conference, New York, NY, USA. pp. 313–320 (2012), DOI: 10.1145/2330163.2330209

2. Chengyi, S., Yan, S., Wanzhen, W.: A Survey of MEC: 1998-2001. In: IEEE International Conference on Systems, Man and Cybernetics IEEE SMC2002, Hammamet, Tunisia. October 6-9. Institute of Electrical and Electronics Engineers Inc., vol. 6, pp. 445–453 (2002), DOI: 10.1109/ICSMC.2002.1175629

3. Floudas, A.A., Pardalos, P.M., Adjiman, C., Esposito, W.R., Gumus, Z.H., Harding, S.T., Klepeis, J.L., Meyer, C.A., Schweiger, C.A.: Handbook of Test Problems in Local and Global Optimization. Kluwer, Dordrecht (1999)

4. Foster, I.: Designing and building parallel programs: concept and tools for parallel software engineering. Boston, Addison–Wesley (1995)

5. Gubenko, G.: Dynamic load Balancing for Distributed Memory Multiprocessors. Journal of parallel and distributed computing, 7, 279–301 (1989)

6. Jie, J., Han, C., Zeng, J.: An Extended Mind Evolutionary Computation Model for Optimizations. Applied Mathematics and Computation, 185, 1038–1049. (2007), DOI: 10.1016/j.amc.2006.07.037

7. Jie, J., Zeng, J.: Improved Mind Evolutionary Computation for Optimizations. In: Proceedings of 5th World Congress on Intelligent Control and Automation, 15–19 June 2004, Hang Zhou, China, pp. 2200–2204 (2004), DOI: 10.1109/WCICA.2004.1341978

8. Karpenko, A.P.: Modern algorithms of search engine optimization. Nature–inspired optimization algorithms. Moscow, Bauman MSTU Publ. (2014) (in Russian)

9. Karpenko, A.P., Sakharov, M.K.: Multi–memetic global optimization based on MEC. Information Technologies, 7, 23–30 (2014)

10. Kerschke, P., et al.: Cell mapping techniques for exploratory landscape analysis. EVOLVE A Bridge between Probability, Set Oriented Numerics and Evolutionary Computation V, Springer, 115–131 (2014), DOI: 10.1007/978-3-319-07494-8_9

11. Liang, J.J., Qu, B.Y., Suganthan, P.N.: Problem Definitions and Evaluation Criteria for the CEC 2014. Special Session and Competition on Single Objective Real–Parameter Numerical Optimization. Technical Report, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou, China and Technical Report, Nanyang Technological University, Singapore (2013)

12. Mersmann, O., et al.: Exploratory landscape analysis. In: Proceedings of the 13th annual conference on Genetic and evolutionary computation, July 12–16, Dublin, Ireland. ACM, pp. 829–836 (2011), DOI: 10.1145/2001576.2001690

13. Munoz, M.A., Kirley, M., Halgamuge, S.K.: Exploratory landscape analysis of continuous space optimization problems using information content. IEEE Transactions on Evolutionary Computation, 19(1), 74–87 (2015), DOI: 10.1109/TEVC.2014.2302006

14. Sakharov, M., Karpenko, A.: A New Way of Decomposing Search Domain in a Global Optimization Problem. In: Proceedings of the Second International Scientific Conference Intelligent Information Technologies for Industry (IITI17). Springer, pp. 398–407 (2018), DOI: 10.1007/978-3-319-68321-8_41

15. Sakharov, M., Karpenko, A.: New Parallel Multi–Memetic MEC–Based Algorithm for Loosely Coupled Systems. In: Proceedings of the VII International Conference on Optimization Methods and Application Optimization and applications OPTIMA–2016, September 25 – October 2, Petrova, Montenegro. pp. 124–126 (2016)

16. Sakharov, M., Karpenko, A.: Performance Investigation of Mind Evolutionary Computation Algorithm and Some of Its Modifications. In: Proceedings of the First International Scientific Conference Intelligent Information Technologies for Industry (IITI16), Springer, pp. 475–486 (2016), DOI: 10.1007/978-3-319-33609-1_43

17. Sakharov, M.K., Karpenko, A.P., Velisevich, Ya.I.: Multi–Memetic Mind Evolutionary Computation Algorithm for Loosely Coupled Systems of Desktop Computers. Science and Education of the Bauman MSTU, 10, 438–452 (2015), DOI: 10.7463/1015.0814435

18. Sobol, I.M., Distribution of points in a cube and approximate evaluation of integrals. USSR Comput. Maths. Phys., 7, 86–112 (1967)

19. Vassilev, V., Fogarty, T., Miller, J.: Smoothness, ruggedness and neutrality of fitness landscapes: from theory to application. Advances in evolutionary computing. New York, NY, USA, Springer, 3–44 (2003)

20. Voevodin V.V., Voevodin Vl. V.: Parallel Computations. SPb.: BHV–Peterburg, (2004) (In Russian)