Introducing the run support strategy for the bison algorithm

Citation

KAZÍKOVÁ, Anežka, Michal PLUHÁČEK, Tomáš KADAVÝ, and Roman ŠENKEŘÍK. Introducing the run support strategy for the bison algorithm. In: *Lecture Notes in Electrical Engineering* [online]. vol. 554, Springer Verlag, 2020, p. 272 - 282 [cit. 2023-02-02]. ISBN 978-3-03-014906-2. ISSN 1876-1100. Available at https://link.springer.com/chapter/10.1007/978-3-030-14907-9_27

DOI

https://doi.org/10.1007/978-3-030-14907-9_27

Permanent link

https://publikace.k.utb.cz/handle/10563/1008759

Introducing the Run Support Strategy for the Bison Algorithm

Anezka Kazikova, Michal Pluhacek, Tomas Kadavy, and Roman Senkerik

Faculty of Applied Informatics, Tomas Bata University in Zlin, T. G. Masaryka 5555, 760 01 Zlin, Czech Republic {kazikova,pluhacek,kadavy,senkerik}@utb.cz

Abstract

Many state-of-the-art optimization algorithms stand against the threat of premature convergence. While some metaheuristics try to avoid it by increasing the diversity in various ways, the Bison Algorithm faces this problem by guaranteeing stable exploitation - exploration ratio throughout the whole optimization process. Still, it is important to ensure, that the newly discovered solutions can affect the overall optimization process. In this paper, we propose a new Run Support Strategy for the Bison Algorithm, that should enhance the utilization of newly discovered solutions, and should be suitable for both continuous and discrete optimization.

Keywords: Bison Algorithm, run support strategy, exploration optimization

1 Introduction

Sources of inspiration for artificial intelligence applications seem to be limitless. The optimization field employs the bases of evolution [**1**], chromosomes [**2**], or even the collective intelligence phenomenon, which created the swarm algorithms [**3**]. The swarm algorithms are powerful tools for solving both continuous and discrete minimization problems by simulating animal behavior. Though many animal species are social creatures, they do not necessarily need to have a leader to make ultimate decisions, and yet they manage to complete nontrivial optimization tasks, like foraging, hunting, mating, or protecting themselves against predators. Simulating such behavior patterns created a variation of successful optimization techniques like the Particle Swarm Optimization [**4**], Grey Wolf Optimizer [**5**], SOMA [**6**], or Cuckoo Search [**7**], which were already used to solve some challenging real-life applications [**8, 9**].

However, many swarm algorithms lean towards premature convergence due to excessive exploitation at the expense of exploration [**10**]. The so-called abnormal exploitation techniques tackle this problem by raising the population diversity [**10-12**]. One of the latest swarm algorithms, the Bison Algorithm, addresses the same problem differently - by guaranteed exploitation-exploration ratio through the whole optimization process. The Bison Algorithm is a multi-agent system, which divides its population into two groups: the swarming group, and the running group. While the first is exploiting the solutions, the second is exploring the search space.

Since the very first proposal of the Bison Algorithm, the mechanics of the running group evolved rapidly. In [**13, 14**] the groups were divided solely by the quality of the found solution. Therefore the weaker solutions explored the search space, while the stronger ones managed the exploitation. However, this approach caused a gradual scattering of the running group, as both groups switched their members after each iteration according to their objective function values. Responding to this, [**15**] proposed a new group arrangement in which the successful exploration solutions were no longer switched, but only copied to the swarming group. The worse swarming solutions were abandoned, and the running group was left intact for further exploration. This redefined the basic Bison Algorithm, as it provided a more logical model.

To enhance the exploration factor even more, [**16**] proposed a "run and seek" variation of the Bison Algorithm, in which the running group temporarily exploited the area of a promising solution on their own and then returned to the exploring behavior. This approach significantly improved the optimization of several functions, yet it was unable to affect the convergence when the swarming group was already stuck in a local optimum quite close to the global one. Also, adding the possibility of switching the exploration and exploitation methods, revoked the guarantee of stable exploration-exploitation ratio.

In this paper, we propose a new Run Support Strategy without changing the exploration behavior of the running group, by redefining the center computation of the swarming movement only. This strategy should provide a robust way of promoting the newly discovered solutions, and should be usable even for large-scale problems, discrete, and real-time optimization.

The paper is structured as follows: Sect. **2** introduces the basic Bison Algorithm, Sect. **3** proposes the new Run Support Strategy and provides an example of the bison movement in 2-dimensional space. Section **4** designs the validation experiments of the new strategy and presents the results. Section **5** discusses the achieved results, and the impact on the future research is evaluated in Sect. **6**.

2 Bison Algorith

The Bison Algorithm is inspired by the exploitation and exploration patterns of bison herds. The former simulates the endangered herd behavior: when attacked, bison create a circle of the strongest individuals to protect the weak. The latter replicates the persistent running behavior [**17**]. The algorithm divides the population into two groups, each simulating different behavior as outlined in Algorithm 1.

Since many swarm algorithms are based on similar principles [**18**], we would like to highlight the difference between the Bison Algorithm and other optimization techniques. The main characteristics of the Bison Algorithm is the separation of the exploration and exploitation. There is a unique group of explorers running through the search space with the sole purpose of avoiding the perils of local optima. This exploration mechanism is the main difference between the Bison Algorithm and other swarm algorithm like the PSO, GWO or SOMA (which explores the search space as an integral part of the exploitation movement in a very different manner). The exploitation movement is based on the center of several fittest solutions, while other algorithms usually use only one best solution to move to. The algorithm was compared to other metaheuristics on IEEE CEC 2017 benchmark functions in [**14-16**].

Algorithm 1: Pseudo code of the basic Bison Algorithm

```
Initialization:
  Objective function: f(x) = (x_1,...,x_d)
  Generate: swarming group randomly, running group around
            x_best and run direction vector (Eq. 4)
For every iteration do
  Compute the center of the swarming movement (Eqs. 1, 2)
  For every swarmer do
    Compute new position candidate f(x_new) (Eq. 3)
    if f(x_new) < f(x_old) then move to x_new
  End
  Adjust run direction vector (Eq. 5)
  For every runner do
    Move in the run direction vector (Eq. 6)
  End
  Copy successful runners to the swarming group
  Sort the swarming group by f(x) value
End for
```

Swarming Behavior. The swarming behavior computes the center of the strongest solutions (Eqs. **1, 2**) and then moves all the solutions from the swarming group closer to the center if it improves their quality (Eq. **3**).

$$weight = (10, 20, \ldots, 10 \cdot s) \tag{1}$$

$$center = \sum_{i=1}^{s} \frac{weight_i \cdot x_i}{\sum_{j=1}^{s} weight_i} \tag{2}$$

$$x_{i+1} = x_i + (center - x_i) \cdot rand(0, overstep)_D \tag{3}$$

Where:

- s is the elite group size parameter,
- x_i and x_{i+J} represent the current solution and the new solution candidate,
- rand(from, to) is a random number in the range of the two given arguments,
- overstep defines the maximum length of the swarming movement,
- D represents the dimensionality of the problem.

Running Behavior. Meanwhile, the running group explores the search space by shifting the whole group in the run direction vector (Eq. **6**), randomly generated during the initialization (Eq. **4**) and slightly altered in each iteration (Eq. **5**). When bison outreach the search space boundaries, they appear on the other side of the dimension.

$$run\,direction = rand\left(\frac{ub - lb}{45}, \frac{ub - lb}{15}\right) \quad (4)$$

$$run\,direction = run\,direction \cdot rand(0.9, 1.1)_D \quad (5)$$

$$x_{i+1} = x_i + run\,direction \quad (6)$$

Where:

- rand(from, to) is a random number in the range of the two given arguments,
- ub and lb are the upper and the lower boundaries of the search space,
- D represents the dimensionality of the problem,
- x_{i+1} and xi represent the current solution and its previous state.

Table **1** describes the parameters of the algorithm and their recommended values [**18**].

Table 1. Parameters of the Bison Algorithm and their recommended values

Parameter	Description	Recommended value
Population		50
Elite group size	No. of best solutions for center computation	20
Swarm group size	No. of bison performing the swarming movement	40
Overstep	The maximum length of the swarming movement 0 – no movement 1 – max to the center	3.5

3 Run Support Strategy

To enhance the impact of the running group, we propose a new strategy for the center computation, applied for successful running solutions. When a runner finds a better solution than a swarmer, the discovered solution should replace the center of the swarming movement for a certain number of iterations, specified by a new parameter called the run support and the overstep parameter is temporarily changed (Algorithm 2). This enables the swarming group to exploit the area around of the discovered solution. When the iteration limit is met, the center of the swarming movement is computed from the fittest solutions within the swarming group (Eqs. **1, 2**).

Algorithm 2: Center computation of the Run Support Strategy

if $f(x_{runner}) < f(x_{swarmer})$ **do**
  for next *run support* iterations
    $center = x_{runner}$
    $overstep = rand(0.95, 1.05)$
  end for
end if

Where:

- x_{runner} and xswarmer are the running and swarming solutions,
- f(x) is the objective function value,
- run support is the number of iterations for the planned exploitation of the promising solution,
- rand(from, to) is a random number in the range of the two given arguments,
- center is the center of the swarming movement,
- overstep is the overstep parameter.

3.1 Run Support Strategy Movement Example

Figure **1** shows the application of the Run Support Strategy on population distribution showing the movement on 2-dimensional Schwefel's function. In the first picture, the center is computed from the fittest solutions within the swarming group (elites).

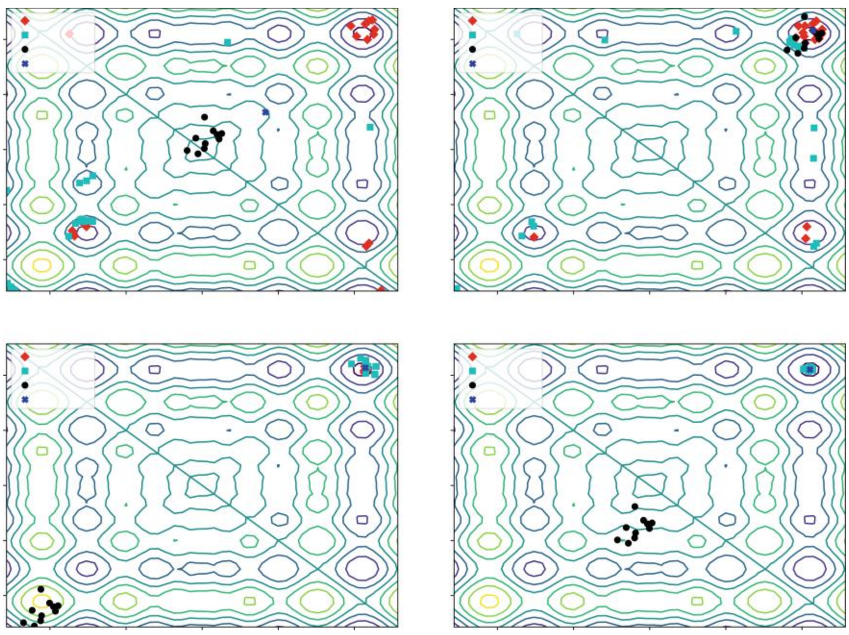


Fig. 1. Movement of the Bison Algorithm with the Run Support Strategy on 2-dimensional Schwefel's function

When the running group reaches the optimum location area (iteration 17), the center is shifted towards the newly discovered solution. When the run support limit is reached, the center is computed again from the best swarming solutions (iteration 20), and ultimately improves the final solution (iteration 25).

4 Methods and Results

As we were particularly interested in the cases, where the running group found a promising solution and employed the Run Support Strategy, we carried out a success simulation experiment. During this experiment, we placed one member of the running group exactly one run direction vector over the global optimum location (Eq. **7**) and generated the rest of the running group around in the original formation. From there the running group explored the search space in the run direction vector, as expected.

$$x_{runner} = x_{opt} + run\ direction \tag{7}$$

Where:

- x_{runner} represents one solution of the running group,
- x_{opt} is the known optimum location,
- run direction is the run direction vector.

All of the experiments were held on 30 independent runs, each consisting of 10000 • dimension evaluations of the objective function solving 10, 30 and 50dimensional problems with well-known locations of global optima (Eqs. **8-11**). The parameter configuration used for the experiments are described in Table **2**.

Due to the exploration emphasis, the Bison Algorithm was originally intended to solve functions with a particularly narrow decreasing neighborhood around the global optimum. This description notably fits the Easom's function, as can be seen in Fig. **2**.

Table 2. Parameter configurations applied for the experimen

Parameter	Basic Bison Algorithm	Success Simulation Run Support Strategy	Tuning of the Run Support Parameter
Population	50	50	50
Elite group size	20	20	20
Swarm group size	40	40	40
Overstep	3.5	3.5	3.5
Run support	0	2	0, 1, 2, 3, 5, 10

Rastrigin's Function

$$f(x) = 10dim + \sum_{i=1}^{dim} x_i^2 - 10\cos(2\pi x_i) \qquad (8)$$

Function minimum for E_n $(x_1; x_2...x_n) = (0, 0,...,0)$. Value for E_n y =0.

The 2nd De Jong s Function (Rosenbrock's Valley)

$$f(x) = \sum_{i=1}^{dim-1} 100(x_i^2 - x_{i+1})^2 + (1 - x_i)^2 \qquad (9)$$

Function minimum for En: $(x_1, x_2...x_n) = (1, 1,...,1)$. Value for En: y =0. **Schwefel's Function**

$$f(x) = 418.9829 - \sum_{i=1}^{dim} -x_i \sin\left(\sqrt{|x|}\right) \qquad (10)$$

Function minimum for E_n: $(x_1, x_2...x_n) = (420.96,., 420.96)$. Value for E_n: y = 0. **Easom's Function**

$$f(x) = -\prod_{i=1}^{dim} \cos(x_i) \cdot e^{\sum_{i=1}^{dim} -(x_i-\pi)^2} \qquad (11)$$

Function minimum for En: $(x_1, x_2...x_n) = (\pi, \pi, ..., \pi)$. Value for E_n: y = –dim + 1.

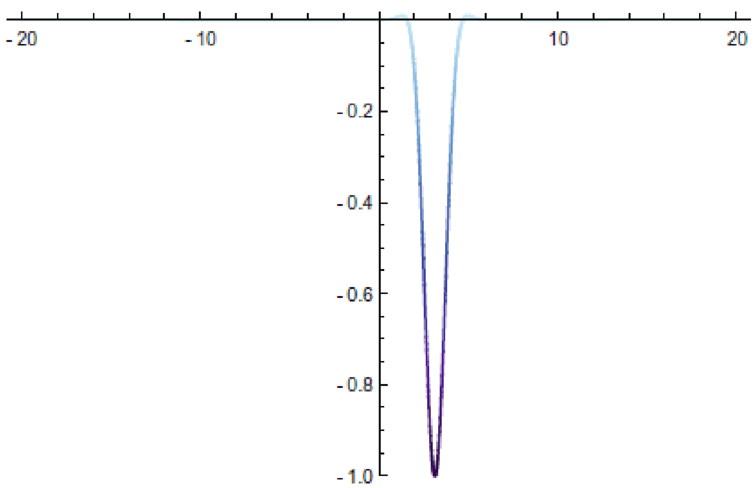


Fig. 2. Easom's function in 2 dimensions

To decide the ideal value of the run support parameter, we compared five values of the parameters on all the functions with the Friedman Rank Test ($p < 0.05$). It is worth mentioning that this experiment considered even the basic Bison Algorithm (with run support = 0). The results are shown in Table **3**.

The results of the success simulation experiments are presented as follows: Table **4** compares the Run Support Strategy and the basic Bison Algorithm with the Wilcoxon rank-sum test ($\alpha = 0.05$). Table 5 shows the statistics of the algorithms. The optimum find rate represents the percentage of all the runs, where the algorithm was able to find a solution with the error f (x) — f($x_{giobaioptimum}$) \<E — 8.

Figure **3** shows a selection of mean convergences of both the success simulation experiments and the standard runs of the algorithms.

Table 3. The Run Support parameter values on functions with simulated success ranked by the Friedman Rank Test ($p < 0.05$). The best ranks are bold.

Run support parameter	0	1	2	3	5	10	P-Value
10 dimensions	5.00	3.00	3.50	**2.75**	3.25	3.50	0.66
30 dimensions	4.75	4.50	3.00	**1.75**	3.25	3.75	0.22
50 dimensions	5.00	**2.25**	3.25	3.25	4.00	3.25	0.47

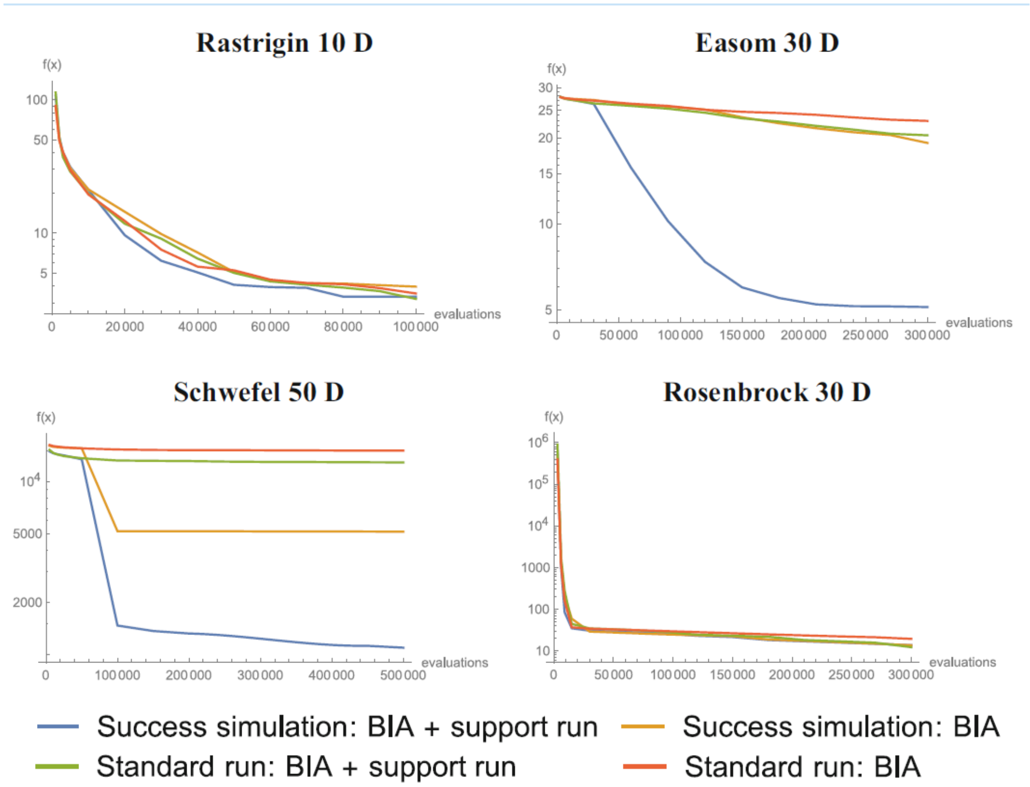
Table 4. Winning algorithms on functions with simulated success (Wilcoxon a = 0.05)

Dimension	Rastrigin	Rosenbrock	Schwefel	Easom
10 D	–	–	Run Support	–
30 D	–	–	Run Support	Run Support
50 D	–	–	Run Support	Run Support

Table 5. Performance of the Run Support Strategy and the basic Bison Algorithm on functions with simulated success (mean error, standard deviation, and optimum find rate)

	Run Support Strategy			Basic Bison Algorithm		
	avg	*std*	*opt*	*avg*	*std*	*opt*
Rastrigin						
10D	3.33	1.96	(7%)	3.97	3.39	(0%)
30D	19.27	6.38	(0%)	22.00	14.69	(0%)
50D	45.24	12.49	(0%)	44.64	10.39	(0%)
Rosenbrock						
10D	1.06	0.80	(0%)	1.23	1.50	(0%)
30D	13.65	5.15	(0%)	13.54	2.94	(0%)
50D	29.84	5.12	(0%)	35.67	16.19	(0%)
Schwefel						
10D	**203.52**	330.81	**(53%)**	741.07	610.45	(27%)
30D	**578.11**	1070.85	**(23%)**	3022.16	1156.57	(3%)
50D	**1091.92**	1785.85	**(10%)**	5129.62	1349.55	(0%)
Easom						
10d	1.84	3.15	(73%)	1.84	2.95	(70%)
30d	**5.11**	9.48	**(73%)**	19.19	10.87	(20%)
50d	**4.37**	11.94	**(80%)**	41.40	9.87	(3%)

Fig. 3. Mean convergences of the basic Bison Algorithm and the Run Support Strategy both in the success simulation experiment and the standard run (with no bias of the running group)



5 Discussion

The success simulation experiment uncovered the supremacy of the Run Support Strategy over the basic version in all of the tested dimensions of Schwefel's function and high dimensional Easom's function (Wilcoxon $\alpha = 0.05$). At the 50-dimensional Easom's function, the new strategy was ultimately able to find the global optimum in 80% of all the runs. The Run Support Strategy also converged better than the basic algorithm even when comparing the standard run and success simulation alone.

The Friedman Rank Test ($p = 0.05$) experiment compared the run support parameter values. Even though that none of the configurations significantly outperformed the other ones, the best results were usually achieved when the run support parameter was set to 3 iterations. In contrast, when the run support was set to 0 (therefore when the basic algorithm was applied), the results were steadily ranked the worst. However, the results also suggest, the trend might change for higher dimensions.

6 Conclusion

We proposed a new Run Support Strategy for the Bison Algorithm and proved, that it is beneficial for the optimization process when compared to the basic version of the algorithm. The strategy was especially successful when solving the Easom's function, a prototype of a function with the global optimum hidden in a very narrow decreasing neighborhood. These results manifest the asset of the boosted exploration. Accordingly, we highly recommend the employment of the Run Support Strategy.

Furthermore, thanks to the robust design of the Run Support Strategy, the outcomes of this research will certainly be used in future research. We would like to exploit the uncovered benefits in a discrete version of the Bison Algorithm, large-scale, and realtime optimization.

References

1. Back, T.: Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms. Oxford University Press, Oxford (1996)

2. Goldberg, D.E., Holland, J.H.: Genetic algorithms and machine learning. Mach. Learn. **3**(2), 95-99 (1988)

3. Chakraborty, A., Kar, A.K.: Swarm intelligence: a review of algorithms. In: Nature-Inspired Computing and Optimization, pp. 475-494. Springer (2017)

4. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings of the IEEE International Conference on Neural Networks, vol. 4 (1995)

5. Mirjalili, S., Mirjalili, S.M., Lewis, A.: Grey wolf optimizer. Adv. Eng. Software. **69**, 46-61 (2014)

6. Zelinka, I.: SOMA—self-organizing migrating algorithm. In: New Optimization Techniques in Engineering, pp. 167-217. Springer, Heidelberg (2004)

7. Yang, X.-S., Deb, S.: Cuckoo search via Levy flights. In: Proceedings of World Congress on Nature & Biologically Inspired Computing (NaBIC 2009), December 2009, India, pp. 210214. IEEE Publications, USA (2009)

8. Pluhacek, M., Senkerik, R., Viktorin, A., Kadavy, T., Zelinka, I.: A review of real-world applications of particle swarm optimization algorithm. In: Lecture Notes in Electrical Engineering, pp. 115-122. Springer, Cham (2018). ISSN 1876-1100

9. Mohamad, A., Zain, A.M., Bazin, N.E.N., Udin, A.: Cuckoo search algorithm for optimization problems-a literature review. In: Applied Mechanics and Materials, vol. 421, pp. 502-506. Trans Tech Publications (2013)

10. Chen, J., Xin, B., Peng, Z., Dou, L., Zhang, J.: Optimal contraction theorem for exploration-exploitation tradeoff in search and optimization. IEEE Trans. Syst. Man Cybern. Part A Syst. Hum. **39**(3), 680-691 (2009)

11. Pluhacek, M., Senkerik, R., Viktorin, A., Zelinka, I.: Chaos Enhanced Repulsive MC-PSO/DE Hybrid. Springer, Cham (2016)

12. Riget, J., Vesterstr0m, J.S.: A diversity-guided particle swarm optimizer-the ARPSO. Dept. Comput. Sci., Univ. of Aarhus, Denmark, Technical report (2002)

13. Kazikova, A., Pluhacek, M., Viktorin, A., Senkerik, R.: Proposal of a new swarm optimization method inspired in bison behavior. In: Matousek, R. (ed.) Recent Advances in Soft Computing (MENDEL 2017), Advances in Intelligent Systems and Computing. Springer, Cham (2018)

14. Kazikova, A., Pluhacek, M., Senkerik, R.: Performance of the Bison Algorithm on benchmark IEEE CEC 2017. In: Silhavy, R. (ed.) Artificial Intelligence and Algorithms in Intelligent Systems, CSOC2018, Advances in Intelligent Systems and Computing, vol. 764. Springer, Cham (2018)

15. Kazikova, A., Pluhacek, M., Viktorin, A., Senkerik, R.: New Running Technique for the Bison Algorithm. In: Rutkowski, L., Scherer, R., Korytkowski, M., Pedrycz, W., Tadeusiewicz, R., Zurada, J. (eds.) Artificial Intelligence and Soft Computing, ICAISC 2018. Lecture Notes in Computer Science, vol. 10841. Springer, Cham (2018)

16. Kazikova, A., Pluhacek, M., Senkerik, R.: Regarding the behavior of bison runners within the bison algorithm. In: Mendel Journal Series 2018, vol. 24, pp. 63-70 (2018)

17. Berman, R.: American Bison. Nature Watch. Lerner Publications, Minneapolis (2008)

18. Sorensen, K.: Metaheuristics—the metaphor exposed. Int. Trans. Oper. Res. **22**(1), 3-18 (2015)

19. Kazikova, A., Pluhacek, M., Senkerik, R.: Tuning of The Bison Algorithm control parameters. In: 32nd European Conference on Modelling and Simulation, 22nd May-26th May. European Council for Modeling and Simulation (2018)