Tournament-Based Convection Selection in Evolutionary Algorithms

Maciej Komosinski ✉ and Konrad Miazga

Institute of Computing Science,
Poznan University of Technology,
Piotrowo 2, 60-965 Poznan, Poland
maciej.komosinski@cs.put.poznan.pl

Abstract. One of the problems that single-threaded (non-parallel) evolutionary algorithms encounter is premature convergence and the lack of diversity in the population. To counteract this problem and improve the performance of evolutionary algorithms in terms of the quality of optimized solutions, a new subpopulation-based selection scheme – the *convection selection* – is introduced and analyzed in this work. This new selection scheme is compared against traditional selection of individuals in a single-population evolutionary processes. The experimental results indicate that the use of subpopulations with fitness-based assignment of individuals yields better results than both random assignment and a traditional, non-parallel evolutionary architecture.

Keywords: evolutionary algorithms, selection scheme, convection selection, diversity, exploration

1 Introduction

A selection scheme is one of the most important elements of evolutionary algorithms [2,6,14]. Not only it determines the selective pressure in the population, but it also controls the distribution of this pressure among all individuals. Over the years many selection schemes were proposed, some of the most popular ones being tournament selection [3,13], ranking selection [4], proportional selection [7] and sigma scaling [1]. A common element for all of them is the monotonicity of the probability of selection with respect to fitness – a sensible property in optimization, since better individuals deserve a higher chance of propagating their genes. In this paper we show that a more complex, non-monotonic selection scheme can improve the performance of evolutionary algorithms.

In a recent paper [11], Komosinski proposed two methods of dividing the population into subpopulations based only on fitness values of individuals, which

This paper appeared in *Lecture Notes in Computer Science (LNCS)* volume 10778, pp. 466–475, Springer, 2018. http://dx.doi.org/10.1007/978-3-319-78054-2_44

does not require computation of any additional, potentially complex and time-consuming, similarity measures. The performance gain obtained by these methods has been verified experimentally in a parallel setting (hence it was a *distribution* technique). The paper discussed the logic behind this way of splitting of the population and provided some explanations on why it was beneficial. This population-splitting scheme was called the *convection distribution* because it facilitates continuous evolutionary progress just like a convection current or a conveyor belt: each subpopulation always tries to independently improve genotypes of a specific fitness range which overall ensures more fitness diversity and avoids the domination of (and the convergence towards) the current globally best genotypes [5]. Occasional, short ascending trends (convections) are visible in the entire range of fitness values. As mentioned in [11], this idea can be directly implemented in a standard, single-threaded (i.e., non-parallel) evolutionary algorithm, where it becomes the *convection selection* scheme.

It is known that given the same computational cost, parallel evolutionary algorithms [18,16,12] can sometimes yield better results in optimization tasks than standard sequential evolutionary algorithms, mostly because of the local exchange of individuals between independent subpopulations. Local exchange of individuals leads to increased exploration of the search space, which is often desirable [16]. Increased exploration can also be achieved in sequential evolutionary optimization using methods such as sharing or restricted mating [15]. Such methods require however calculating of additional measures of similarity between individuals, which may be time consuming, especially in applications where individuals are complex [11], such as evolutionary design or artificial life.

Convection selection techniques may be perceived as super-selection techniques in that they determine which individual should be assigned to which subpopulation, yet within these subpopulations traditional selection schemes are still employed. Thus convection selection can be combined with any traditional selection method, constituting convection tournament selection, convection roulette selection, etc. Moreover, while in this work we will discuss one-level convection selection (i.e., a population divided into sub-populations), this technique can act on multiple levels with subpopulations recursively embedded in each other.

The experiments reported in [11] proved that convection distribution methods yielded significantly better results than random distribution of genotypes among subpopulations. In this work, we investigate when the *convection selection* (assigning individuals to subpopulations based on fitness values) can yield better results compared to standard, single-population positive selection schemes such as tournament selection. We also analyze the underlying mechanisms responsible for the success of this new approach.

Apart from implementing three subpopulation-based selection techniques in a single-threaded (non-parallel) evolutionary algorithm and comparing their performance, we also compare these three approaches against a standard, single-population evolutionary algorithm. In all comparisons we ensure that the overall computational cost is the same – in each evolutionary run, we keep the number of evaluations of individuals equal, and the computational cost of managing subpopulations and migrations between subpopulations is negligible. Moreover, we test each of the four mentioned approaches (Fig. 1) using various selective pressures and populations sizes, and for each approach we choose the best performance among its various parametrizations to ensure a fair comparison.

2 Methods

All the experiments described in this paper were performed using Framsticks software [10,9]. Framsticks allows to evolve bodies and brains of 3D designs (agents) towards a goal specified by some fitness function. This area of application of evolutionary algorithms benefits the most from selection schemes that improve the performance yet are still computationally inexpensive. This is because optimization tasks in evolutionary design are extremely difficult and solutions are very complex due to sophisticated genotype-to-phenotype mappings, so calculating sophisticated properties of such solutions or estimating their similarity is usually very costly and should be avoided if possible.

We have used two fitness functions that differ in the difficulty of optimization: *velocity* and *height*. The *velocity* criterion is used to evolve individuals that move fast on land (so body and brain are coevolved and must be coordinated), whereas *height* is used to evolve static tall structures (their neural network is disabled) with the center of mass as elevated as possible.

The "f1" genetic encoding was employed [8,9]. This encoding is a direct mapping between symbols and parts of a 3D structure: 'X' represents a rod (a stick), parentheses encode branches in the structure, and additional characters influence properties like length or rotation. Neurons are described in square brackets and index numbers in their connections are relative, so the information about connections is local and persists when a part of a genotype is cut out. The encoding is able to represent tree-like 3D body structures and neural networks of arbitrary topology. Mutations modify individual aspects of the agent by adding or removing parentheses in random locations in the genotype, by adding and removing random symbols that affect the structure, by adding and removing neurons and connections, and by adding random Gaussian-distributed values to neural weights.

For both fitness functions, evolution was started from the simplest individual (i.e., 'X' in the $f1$ encoding). The steady-state (also known as "incremental") evolutionary algorithm [17] was used. To limit the number of factors that might influence the performance of convection selection schemes, no crossover was employed in the experiments reported here. The crossover was however used in the experiments discussed in [11], where convection selection schemes provided superior results. The absence of the crossing over operator in this work and the fact that convection selection schemes still yielded superior results means that the crossover operator is not the only mechanism responsible for the efficiency of these selection techniques.

In the convection selection schemes, individuals are first sorted according to their fitness. Then each subpopulation receives a subset of individuals that fall within a range of fitness values. In our experiments, two methods of determining fitness ranges are considered. In the first method denoted $EqualWidth$ (Fig. 1c),

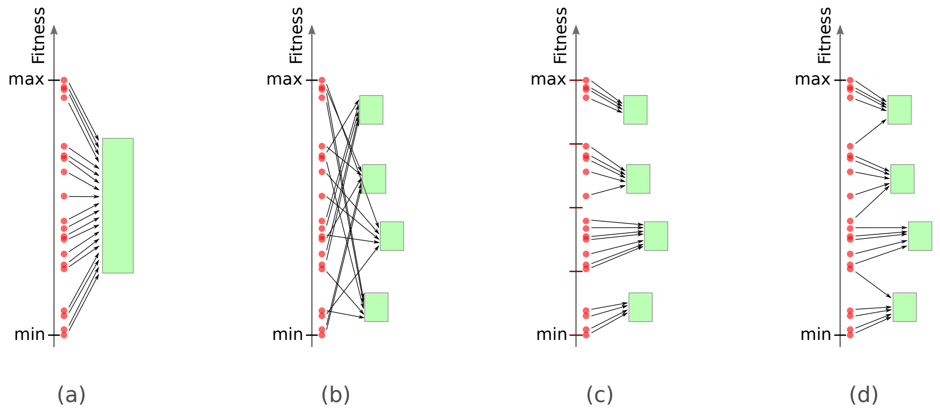


Fig. 1: An illustration of four compared selection schemes. The fitness of 20 individuals is shown as red circles, and 4 subpopulations are depicted as green boxes. (a) Standard evolutionary algorithm with a single population. (b) Random assignment of individuals to subpopulations. (c) Convection selection with fitness intervals of equal width. (d) Convection selection with fitness intervals yielding equal number of individuals.

the entire fitness range has been divided into equal intervals (as many as there are subpopulations); if there are no individuals in some fitness range, the corresponding subpopulation receives individuals from the nearest lower non-empty fitness interval. In the second method denoted $EqualNumber$ (Fig. 1d), once the individuals are sorted according to their fitness, they are divided into as many sets as there are subpopulations so that each subpopulation receives the same number of individuals.

We compare here four approaches to selection (three of which use subpopulations), and in each of them the underlying traditional selection mechanism is the tournament selection. The logic of the three evolutionary processes that use selection to assign individuals to subpopulations (i.e., $Random$, $EqualWidth$, or $EqualNumber$) is implemented as follows. Every $R \cdot \frac{N}{M}$ evaluations (where R is the migration period scaling factor which defines how frequently subpopulations should merge, N is the size of the entire population, and M is the number of subpopulations), M subpopulations are merged and then all individuals from the complete (merged) population are split again into M subpopulations according to the applied selection scheme ($Random$, $EqualWidth$, or $EqualNumber$). After that, the algorithm cycles through all subpopulations in sequence so that each subpopulation becomes "current" in turn. The steady-state evolutionary algorithm selects one individual from the current subpopulation (using tournament selection with the tournament of size t), mutates it and adds the newly mutated offspring to the current subpopulation. Once this new individual has been evaluated, the negative selection process removes randomly one individual from a random subpopulation, so the size of the complete population remains constant. Then, the next subpopulation in sequence becomes current. After all subpopulations have been processed, the cycle starts again unless it is time to merge all subpopulations and redistribute individuals to newly constructed subpopulations.

In this paper we perform two kinds of analyses. The first kind compares the quality of solutions obtained from the standard single-population evolutionary algorithms with the results yielded by the three proposed subpopulation-based selection schemes. The proper comparison between the single-population algorithm and the three subpopulation-based approaches is not simple, as each of these two concepts uses a slightly different set of parameters. Moreover, even the parameters that are shared between the four approaches can have different optimal values for each approach. If one wants to properly compare the quality of solutions achieved with each of the considered selection schemes, one should compare the best results obtained across a series of many different parametrizations for each selection scheme. Therefore, within each parametrization, the representative result for that parametrization is considered to be the average value of the best fitness values obtained across many independent runs (repetitions).

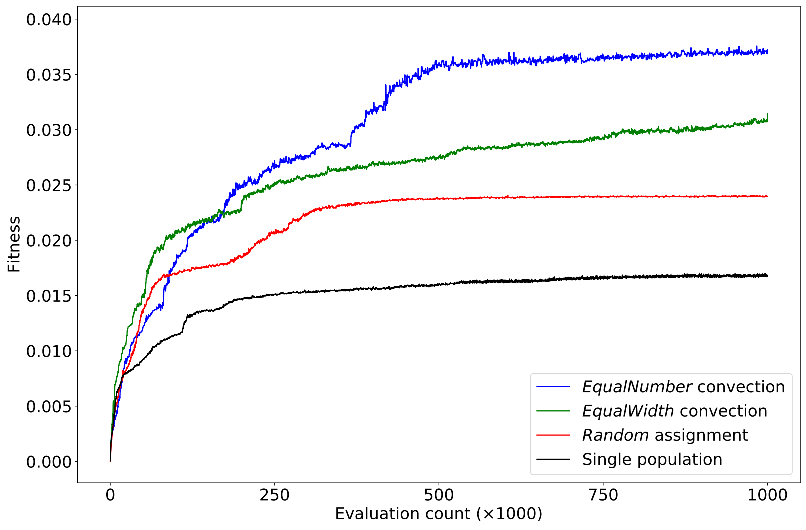
The second kind of the analysis takes a more detailed look into the results obtained by the three population-based selection schemes, two of which are convection selection schemes. We compare the average results achieved for each set of parameter values in order to understand which combinations of parameter values work well together, which combinations work poorly, and what are the potential reasons for such behavior.

The data required for both of the analyses discussed above were obtained from the following experiments. In each of the evolutionary runs, 10^6 individuals were evaluated, so that even though the selection schemes were different, they did not differ significantly in the overall computational cost. Two fitness functions were considered: *velocity* and *height*. For the single-population evolution and tournament selection, we have tested all the combinations of two parameters: population size $N \in \{100, 200, 500, 1000\}$ and tournament size $t \in \{2, 3, 5\}$. For three subpopulation-based selection schemes, all the combinations of the following sets of parameter values were tested: population size $N = 1000$, tournament size $t \in \{2, 3, 5\}$, number of subpopulations $M \in \{4, 10, 25, 50\}$, and the number of individual evaluations between merging the subpopulations (given as the multiple of the size of subpopulations) $R \in \{2, 10, 50\}$. Such a setup means that to obtain one result (i.e., best fitness value from one evolutionary run) for each combination of fitness functions and parameter values, we needed to perform $2 \times ((4 \times 3) + 3 \times (3 \times 4 \times 3)) = 240$ independent evolutionary runs. Since the evolutionary process is non-deterministic, to obtain averages and standard deviations for each parametrization, these runs were repeated 10 times which yielded 2400 independent evolutionary runs.

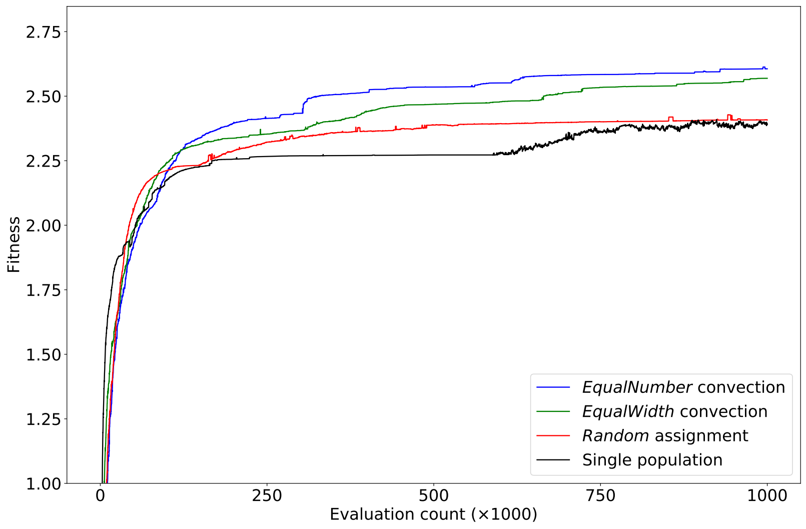
3 Results

3.1 The performance of different selection schemes

Fig. 2 shows the performance of the evolutionary algorithms in time (measured as the number of individual evaluations) for four selection schemes –



(a) *velocity* fitness function.



(b) *height* fitness function.

Fig. 2: Comparison of the performance of single-population tournament selection and three proposed meta-selections. Each series consists of the high bound (i.e, best) of the average fitness value obtainable for a given selection scheme, for any of the tested sets of parameter values. The band around each series represents 25% of the standard deviation for that series (25% is used instead of 100% to avoid overlapping bands and improve the readability of the plots).

one single-population tournament selection, and three subpopulation-based algorithms with super-selection schemes. Since the influence of parameter values for the single-population approach and the three subpopulation approaches is not directly comparable (even for the same parameters), in order to provide a fair comparison we show the best average fitness value achieved by any parametrization for each approach, computed separately for each point in time. This means that the chart is a high-level comparison of the best performance of the four selection schemes that can be achieved over all parametrizations.

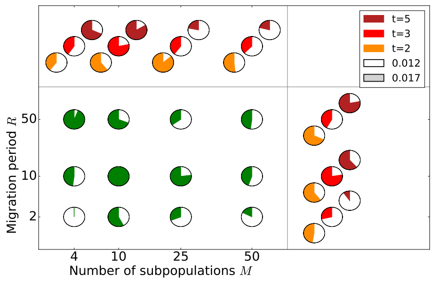
For the *velocity* fitness function, the performance of subpopulation-based selection schemes is clearly superior to the single-population tournament selection. While the fitness values for single-population evolution stabilize near the value of 0.017, the convection-based schemes manage to overtake it by a significant margin. The *Random* assignment selection scheme stabilizes only around the value of 0.024, whereas the convection schemes continue to improve in time (see Fig. 9 in [11] for the distributions of fitness values that illustrate the convection effect), ultimately reaching 0.031 for the $EqualWidth$ method and 0.037 for the $EqualNumber$ method.

The plot for the *height* fitness function presents similar, although less pronounced relationships. Once again the subpopulation-based schemes yield better results than the single-population selection, with convection selection schemes outperforming the *Random* assignment of individuals to subpopulations. It is worth noting however that for the first few thousand evaluations, single-population selection leads to better individuals than the subpopulation-based schemes – in this phase the optimization is relatively easy, and so population diversity (exploration) is not as beneficial as intensive, fast exploitation. Once the solutions reach the fitness values above 2 it is much harder to produce better individuals, at which point the subpopulation-based schemes overtake the single-population selection.

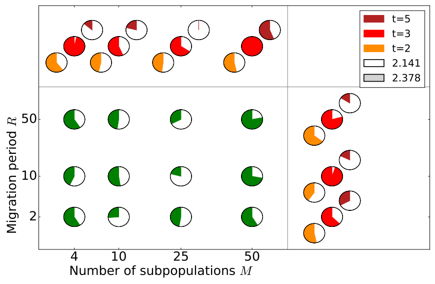
3.2 The influence of parameters of the convection selection

Fig. 3 presents the effect that parameter values of convection selection have on the quality of solutions that were found by the evolutionary algorithm. Depending on the selection scheme, various trends can be seen. For *Random* assignment of individuals to subpopulations (Figs. 3a and 3b) no clear patterns emerged – parameter values do not demonstrate any direct influence on fitness, which may suggest that without any specific logic like fitness-based selection, the algorithm cannot fully exploit the advantages of working with multiple subpopulations.

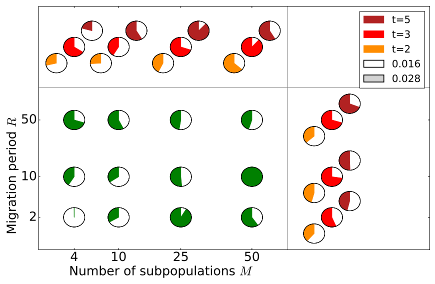
The opposite is however visible for the $EqualNumber$ convection selection scheme (Figs. 3e and 3f). For the *height* fitness function (Fig. 3f), high selective pressure yields better results, as represented by darker circles being more filled up than the light ones. For both fitness functions, increasing the migration period scaling factor R (the vertical axis) leads to better results. The increase in the value of R allows each of the subpopulations to significantly increase the quality of its solutions before the subpopulations are merged; for longer migration periods, the contents of each subpopulation can change considerably between



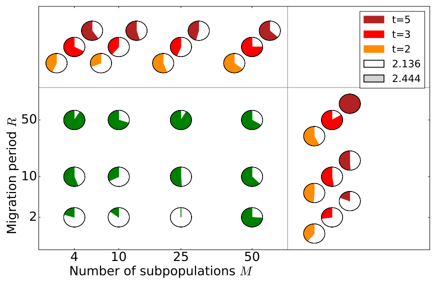
(a) Fitness: *velocity*
Selection: *Random* assignment



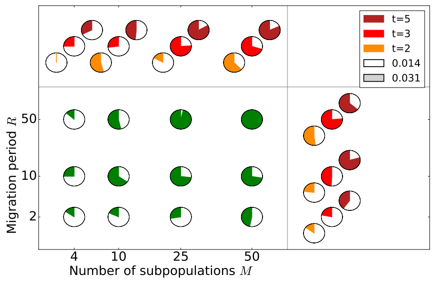
(b) Fitness: *height*
Selection: *Random* assignment



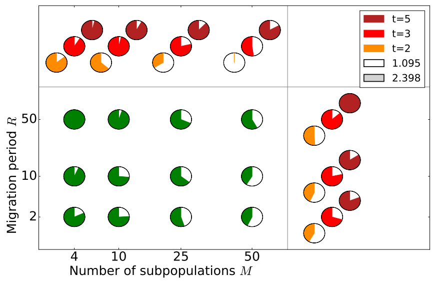
(c) Fitness: *velocity*
Selection: *EqualWidth* convection



(d) Fitness: *height*
Selection: *EqualWidth* convection



(e) Fitness: *velocity*
Selection: *EqualNumber* convection



(f) Fitness: *height*
Selection: *EqualNumber* convection

Fig. 3: Average best fitness values for each combination of parameter values after 10^6 evaluations, additionally averaged along each of the three dimensions (parameters). Empty circles represent the minimal fitness value present in each chart, and full circles represent the maximal fitness value in each chart. The minimal and maximal fitness values are shown in the legend.

migrations which facilitates diversity, and this is a desired property for hard optimization problems.

The number of subpopulations M has a different effect on fitness values for each fitness function. For *velocity* (Fig. 3e), increasing the number of subpopulations (and hence reducing their size) has a positive effect on the quality of results, which is indicated by the circles filling up along the horizontal axis, while for *height* (Fig. 3f) and for early evolution of *velocity* (around the first 50k evaluations) it has a negative effect. While it is not clear what causes this difference, one possible explanation is related to different properties of these fitness functions, as demonstrated in Fig.2. While the *velocity* criterion allows the algorithm to continuously improve the quality of solutions by exploring new ideas of "how to be fast" (i.e., more possibilities for exploration), the evolution of *height* quickly leads to a plateau, where the improvement can be achieved mostly by fine-tuning of existing solutions ("local optima") that are easy to break.

Although no obvious trends are visible for the $EqualWidth$ selection scheme (Figs. 3c and 3d), it is worth noting that the combination of a small number of big subpopulations and frequent migrations is unfavorable for both fitness functions, as indicated by primarily empty circles in the bottom-left part of these plots. The most likely explanation of this is the low level of exploration that results from such parametrization.

4 Conclusions

In this article, we investigated the concept of convection distribution and convection selection [11] in single-threaded (non-parallel) evolutionary algorithms and demonstrated that dividing the population into subpopulations based on fitness values of individuals can significantly improve the quality of optimized solutions. We have discussed potential mechanisms responsible for superior results of the convection-based methods, the most important ones being the diversification of the population and the ability to constantly explore diverse paths in fitness landscape [11]. If many subpopulations are allowed to evolve independently for longer periods of time, we can expect that each of them will produce unique, fit solutions which can then compete and cooperate every time the subpopulations are merged.

There are a number of issues that should still be examined. Even though the experiments reported in this paper were computationally highly expensive due to a large number of combinations of parameter values and very complex evolutionary goals, it would be worthwhile to extend the ranges of parameters to test the space of possible parameter combinations more comprehensively. It would be advantageous to test the proposed approaches on more fitness functions, including well-known benchmark optimization problems. Apart from convection selection and random assignment of individuals to subpopulations, we would like to additionally test the policy that ensures the best individual is placed in each subpopulation. For larger populations, the convection selection may have multiple levels so that it is applied recursively and subpopulations are nested in each other – this concept is worth testing too, along with dynamic, adaptive strategies of splitting and merging subpopulations and recursion levels. It would be useful to devise a formal statistical model behind convection selection to understand its mechanisms and causes of its success. Finally, it will be interesting to investigate to what extent can crossover benefit from convection-based schemes where fitness of parents is in most cases similar.

Acknowledgments The research presented in the paper received support from Polish National Science Center (DEC-2013/09/B/ST10/01734).

References

1. Back, T.: Selective pressure in evolutionary algorithms: A characterization of selection mechanisms. In: Evolutionary Computation, 1994. IEEE World Congress on Computational Intelligence., Proceedings of the First IEEE Conference on. pp. 57–62. IEEE (1994)
2. Back, T.: Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms. Oxford university press (1996)
3. Blickle, T., Thiele, L.: A mathematical analysis of tournament selection. In: ICGA. pp. 9–16. Citeseer (1995)
4. Blickle, T., Thiele, L.: A comparison of selection schemes used in evolutionary algorithms. Evolutionary Computation 4(4), 361–394 (1996)
5. Črepinšek, M., Liu, S.H., Mernik, M.: Exploration and exploitation in evolutionary algorithms: a survey. ACM Computing Surveys (CSUR) 45(3), 35 (2013)
6. Dasgupta, D., Michalewicz, Z.: Evolutionary algorithms in engineering applications. Springer Science & Business Media (2013)
7. Goldberg, D.E., Deb, K.: A comparative analysis of selection schemes used in genetic algorithms. Foundations of genetic algorithms 1, 69–93 (1991)
8. Komosinski, M., Rotaru-Varga, A.: Comparison of different genotype encodings for simulated 3D agents. Artificial Life Journal 7(4), 395–418 (Fall 2001)
9. Komosinski, M., Ulatowski, S.: Framsticks: Creating and understanding complexity of life. In: Komosinski, M., Adamatzky, A. (eds.) Artificial Life Models in Software, chap. 5, pp. 107–148. Springer, London, 2nd edn. (2009)
10. Komosinski, M., Ulatowski, S.: Framsticks web site (2016), `http://www.framsticks.com`
11. Komosinski, M., Ulatowski, S.: Multithreaded computing in evolutionary design and in artificial life simulations. The Journal of Supercomputing 73(5), 2214–2228 (2017), `http://www.framsticks.com/files/common/MultithreadedEvolutionaryDesign.pdf`
12. Luque, G., Alba, E., Dorronsoro, B.: Parallel genetic algorithms. Parallel metaheuristics: A new class of algorithms pp. 107–126 (2005)
13. Miller, B.L., Goldberg, D.E.: Genetic algorithms, tournament selection, and the effects of noise. Complex systems 9(3), 193–212 (1995)
14. Sastry, K., Goldberg, D.E., Kendall, G.: Genetic algorithms. In: Search methodologies, pp. 93–117. Springer (2014)
15. Spears, W.M.: Simple subpopulation schemes. In: Proceedings of the Evolutionary Programming Conference. vol. 3, pp. 296–307. World Scientific River Edge, NJ (1994)
16. Sudholt, D.: Parallel evolutionary algorithms. In: Springer Handbook of Computational Intelligence, pp. 929–959. Springer (2015)
17. Syswerda, G.: A study of reproduction in generational and steady state genetic algorithms. Foundations of genetic algorithms 2, 94–101 (1991)
18. Tomassini, M.: Parallel and distributed evolutionary algorithms: A review (1999)