Adaptation of a Multiagent Evolutionary Algorithm to NK Landscapes

David Chalupa
Institute of Applied Informatics
Faculty of Informatics and Information Technologies
Slovak University of Technology
Ilkovičova 3
84216 Bratislava, Slovakia
chalupa@fiit.stuba.sk

ABSTRACT

Multiagent evolutionary algorithm (MEA) is a relatively new optimization technique, where a life cycle of a population of agents, which perform local search, is simulated. The algorithm was originally intended as a method for solving the graph coloring problem and incorporates ideas such as lifespans of agents and a positive or negative reinforcement for the ability of the agent to improve fitness or its stagnation. In this paper, we propose to use MEA for optimization on NK fitness landscapes. These landscapes are popular for the tunability of their ruggedness and are a particularly interesting use case for MEA. This algorithm is especially well suited for functions, where local search tends to fail because of their multimodality. However, using many short-term local search subroutines in a well-tuned version of MEA can significantly improve the results of the same local search algorithm. Experimental results are presented for MEA with the simple (1+1) Evolutionary Algorithm ((1+1) EA) used as a local search subroutine. These results show that in large and more rugged NK landscapes, MEA outperforms the multi-start (1+1) EA with number of parallel starts equal to the initial population size of MEA. This is the first time we obtained results, which clearly indicate that solely the emergent multiagent nature of MEA, driven by the lifespans and the reinforcement mechanism, is able to improve the results of multi-start local search.

Categories and Subject Descriptors

I.2.8 [**Problem Solving, Control Methods, and Search**]: Heuristic methods

General Terms

Algorithms

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
GECCO'13 Companion, July 6–10, 2013, Amsterdam, The Netherlands.
Copyright 2013 ACM 978-1-4503-1964-5/13/07 ...$10.00.

Keywords

multiagent evolutionary algorithm, NK landscapes

1. INTRODUCTION

Population-based stochastic local search algorithms are currently a very popular topic both at the core of evolutionary computation research [1, 21] and applications such as graph coloring [14], vehicle routing [16] or protein folding [6]. Simple local search algorithms usually tend to fail on fitness landscapes with strong bias towards local optima or suffer from cycling on very large plateaus. Population-based local search is one of the most popular ways, which are chosen by researchers to overcome these difficulties. One of the alternative ways is represented by learning algorithms, such as the hill climbing with learning [11].

Multiagent evolutionary algorithm (MEA) is a general metaheuristic model, which draws inspiration from simulation of a life cycle of a population of agents, which perform local search on a fitness landscape. Each agent has its lifespan and new agents are born with certain period of time, which causes MEA to be an algorithm with variable population size. The general idea behind MEA is to perform many short-term local search subroutines, instead of long-term local search. A collection of promising solutions, which were previously found, is explicitly stored in a structure called *elite list*. From this structure, the initial solutions for the newly born agents are taken. Thus, MEA relies on a well-tuned process of highly organized restarts from promising solutions and performing short-term local search subroutines to improve these solutions [2]. We note that there also is another multiagent evolutionary algorithm with similar name, abbreviated MAEA-CmOPs, which uses a simulation of agents on a toroidal lattice, which aim to maximize their energy [12]. This approach, despite the similarity in the name, uses very different ideas than the method we study here.

In this paper, we explore the ability of MEA to optimize on *NK landscapes* [9]. These landscapes are a popular topic of both theoretical and empirical research in evolutionary computation, because of the tunability of their ruggedness [8]. A number of results in the research on NK landscapes is dedicated to measures of their hardness [13, 19], phase transitions [3, 5] and their structure [15]. NK landscapes represent a particularly interesting use case for MEA. The previous research suggests that MEA is promising especially on functions with some level of multimodality, where pure

local search tends to fail because of local optima or plateaus. However, the landscape should not be too rugged, since then we might face a problem that may not likely be solvable by any state-of-the-art method. Therefore, NK landscapes offer a promising way, how to study the optimization capabilities of MEA on functions with certain level of ruggedness.

The paper is organized as follows. In Section 2, we review the related work regarding MEA and population-based local search in general, as well as the main results regarding the NK landscapes. In Section 3, we describe the (1+1) Evolutionary Algorithm ((1+1) EA) for NK landscapes and its use within the optimization framework of MEA. In Section 4, we present the experimental results of MEA and compare it to the equivalent multi-start (1+1) EA. In Section 5, we finally present the conclusions to this work.

2. RELATED WORK

At this point, we provide a short introduction to the related research on population-based local search and MEA. Additionally, we review the foundations of NK landscapes and the most important results on them.

2.1 Population-based Local Search

The design of a solid population-based framework for local search is still a current research topic. In swarm intelligence, different metaheuristic ideas are taken from the behavior of certain multiagent systems in nature to organize the search process. The most typical example are the bee algorithms, such as the Artificial Bee Colony (ABC). ABC differentiates between several different types of bees, which have different tasks in the search process: employed bees, onlooker bees and scout bees [7].

In contrast, in MEA, each agent has basically the same role in the optimization process. In a certain generation, the agent is born and some previously found promising solution, as well as an initial *lifespan* is assigned to it. Then, the agent performs several iterations of local search each generation. Each agent also records one promising solution it found during its life. This solution is the best from the solutions, at which it arrived at the end of local search for each generation. These moments at the end of local search within each generation are referred to as *milestones*. Then, when agent reaches zero lifespan and is eliminated, this "restricted best" solution is passed to the elite list. The elite list accepts it if and only if this solution is not inferior to all current solutions in the elite list. Additionally, MEA also incorporates a mechanism of *reward or punishment*, when lifespan of an agent is modified according to the change in fitness [2].

We note that MEA was originally tailored for the graph coloring problem and incorporated a specific tabu search algorithm for graph coloring. However, the ideas, which were used to improve the results of the tabu search, were quite general [2]. The results suggested that MEA benefits from several concepts, which are briefly summarized in the following.

1. The way how agents are born helped to extend the search in areas, where the agent could choose from several equivalent ways.

2. The restriction on the solutions in elite list to the milestones helped to avoid restarts from local optima.

3. The length of local search within a generation was longer for solutions with higher fitness. This was due to the fact that at the end of the evolution, an improvement tends to require many more steps than at the beginning of the evolution.

2.2 NK Model of Fitness Landscapes

We aim to adapt the previous concepts to a more general problem. We chose the NK landscapes, because their multimodality can be tuned. Thus, they offer a possibility to study the observed properties of MEA more transparently than in our previous research [2].

NK model creates binary optimization problems based on two parameters: n - the number of bits, k - the number of artificially created "interactions" per each bit [9]. The fitness is calculated as a sum of contributions of each bit, i.e.:

$$F(x) = \sum_{j=1}^{n} f_j(x_j, x_{I_{j,1}}, x_{I_{j,2}}, ..., , x_{I_{j,k}}), \qquad (1)$$

where $F(x)$ is the fitness of bitstring x, f_j is the *contribution function* for bit j and $I_{j,1}$,$I_{j,2}$,...,$I_{j,k}$ are the indexes of bits, which *interact* with bit j. These values of f_j can simply be stored in a collection of n lookup tables with 2^{k+1} cells and are generally taken from some probability distribution [18].

The advantage of NK model is that it can be tuned very well. For $k = 0$, we obtain a simple unimodal function. For $k = 1$, the problem is still in P, while for $k > 1$, the problem is NP-hard [23]. With growing k, the NK landscape tends to be more rugged and local search is more likely to get stuck in local optima. However, it should be taken into account that the interaction mapping I should not be restricted, since when the interacting bits are the next k bits in the string (with cycling if we reach the end of the string), the problem can be solved in polynomial time by dynamic programming [23].

In the unrestricted model, which is NP-hard, the general practice is that the k interactions are chosen randomly and the lookup tables for each bit are generated uniformly at random from $[0, 1)$. These unrestricted NK landscapes were previously used in experimental studies [17, 18] and will be used also in our investigation.

3. (1+1) EA AND MEA ALGORITHMS FOR NK LANDSCAPES

(1+1) EA is probably the simplest evolutionary algorithm, where we have a single individual. This individual carries a vector x of n bits at the beginning and then, it is improved by flips of each bit with probability depending on n. Most typically, probability of flipping each bit is $1/n$. The resulting solution is accepted if and only if it leads to an individual with at least as good fitness $F(x)$ as the current one.

The pseudocode of (1+1) EA is given in Algorithm 1. (1+1) EA is often a subject of theoretical research [4]. Nevertheless, it is suitable also for our experimental investigations for its simplicity and generality. In a previous study, it was shown that mutation rate $1/n$ per bit is generally better than having a constant probability of a flip for each bit. Additionally, it was shown that mutation rate $1/n$ is nearly optimal for a spectrum of pseudoboolean functions with bounded epistasis, to which NK landscapes also belong [20]. Therefore, in our experiments, we use (1+1) EA with

Algorithm 1: (1+1) EA

	(1+1) EA
	Input: input vector $x \in \{0,1\}^n$
	Output: output vector $x_o \in \{0,1\}^n$
1	while stopping criterion is not met
2	let x' be a vector created by flipping each bit of x with probability $1/n$
3	if $F(x) \leq F(x')$ then $x = x'$
4	return $x_o = x$

Algorithm 2: The Multiagent Evolutionary Algorithm (MEA) for NK Landscapes

	The Multiagent Evolutionary Algorithm (MEA) for NK Landscapes
1	generate $\|P_0\|$ agents randomly, assign q points to their lifespans and for each agent, let the local search length be γ, let m be 1
2	while stopping criterion is not met
3	decrease lifespan of each agent by 1 point
4	eliminate agents with 0 lifespan and for each eliminated agent, update the elite with the best solution from the agent's milestones
5	if we have T_b-th iteration create a new agent with q points in its lifespan, assign a random solution the elite list to it, let its local search length be $(m+1)\gamma$ and increment m by 1
6	for each agent, perform local search for the number of iterations determined by its local search length
7	for each agent, increase its lifespan to the initial value q if the agent improved its fitness or decrease its lifespan by r if stagnation occured
8	return the best state ever found

this mutation rate, including the (1+1) EA, which will be used within MEA.

Let us move on to the description of MEA. The pseudocode is given in Algorithm 2. In step 1, we generate agents of an initial population P_0. These agents carry random solutions and are assigned q points to their lifespan. Also, the basic local search length γ is assigned to each of them.

Then, an iterative procedure is performed. In step 3, we decrease lifespans of each agent by 1 point. In step 4, we eliminate agents with 0 lifespan and pass their recorded solutions to the elite list. The step 5 is performed only each T_b iterations and in this step, a new agent is born and a solution from the elite list is assigned to it (we will describe the elite list in more detail later). In step 6, each agent performs certain number of local search iterations, in our case (1+1) EA. Finally, in step 7, the agents' lifespans are changed according to the change in their fitness. Steps 3-7 are repeated, until a stopping criterion is met.

At this point, let us explain, how the *elite list* and the *local search subroutines* work. At each generation, agents perform a number of local search iterations (according to their local search lengths) and arrive at final solutions for that generation. These moments will be referred to as *milestones*. During its life, the agent records the best solution from the milestones.

At the moment, when an agent is eliminated, it passes the best solution from its milestones to the elite list. This solution is then accepted by the elite list if the currently worst solution in the elite list is at most as good as this new solution. If it is, the worst solution in elite list is replaced. The size of the elite list will be denoted by $|L_e|$.

Another mechanism is the *agents' birth*. This happens only each T_b iterations. At this moment, q points are assigned to the agent's lifespan and a random solution from the elite list is assigned to this agent.

In [2], the local search length for the agents in MEA was determined by a problem-specific formula for graph coloring. Thus, in this paper, we have to generalize this length specification. The basic idea was, however, that once we are in an advanced stage of the search, it is good to extend the length of local search subroutines. Therefore, we will have a constant γ, which denotes the basic length of the subroutine. The initial agents will perform γ iterations at each generation. Then, each newly born agent will perform γ more iterations, thus, m-th newly born agent will perform $(m+1)\gamma$ local search iterations.

The last issue to discuss in MEA is the *evaluation operation* in step 10. Originally, r points were given or taken from the agent's lifespan according to whether the agent improved the fitness during the local search or not [2]. However, we soon discovered that this way, agents are punished more frequently than rewarded, especially on very hard landscapes.

Thus, in this paper, we use the following modified strategy. If the final fitness of the agent is better than the initial fitness, the lifespan is reset to the initial lifespan q. Otherwise, it is decremented by r. This should improve the fairness of this mechanism, since even for a small improvement, the agent is largely rewarded, while for stagnation, it is punished only mildly.

Summarizing the parameter "jungle" of MEA, we have initial population size $|P_0|$ and elite list size $|L_e|$. For simplicity, we will assume that $|P_0| = |L_e|$. In addition, we have the initial lifespan q, birth period T_b, basic local search length γ and punishment parameter r. Therefore, we have 5 parameters to tune.

Table 1: The average fitness values for the best individuals in short-term runs of multi-start (1+1) EA with different population sizes on different NK landscapes.

n	k	$\|P\| = 1$	$\|P\| = 5$	$\|P\| = 10$	$\|P\| = 20$	$\|P\| = 50$
n = 30	k = 2	21.942	22.108	22.108	22.108	22.108
	k = 3	22.523	22.990	22.949	22.980	22.991
	k = 4	22.609	23.440	23.281	23.378	23.478
n = 50	k = 2	37.041	37.574	37.519	37.578	37.580
	k = 3	37.396	38.556	38.460	38.573	38.622
	k = 4	37.421	38.891	38.625	38.816	38.880
n = 80	k = 2	58.517	59.135	59.089	59.125	59.211
	k = 3	59.262	60.620	60.473	60.511	60.566
	k = 4	59.709	61.261	61.267	61.245	61.400

4. EXPERIMENTAL RESULTS

In this section, we present the experimental results of MEA on NK landscapes and their comparison to an equivalent multi-start (1+1) EA. To ensure fairness, this algorithm will have the same population size as is the initial population size $|P_0|$ of MEA.

4.1 Experimental Settings

In [17, 18], NK landscapes with $n \in [20, 56]$ and $k \geq 2$ were used. The interactions between bits were generated randomly and the values in the lookup tables were taken from the uniform distribution over $[0, 1)$. In [20], similar settings were considered, with the difference that a higher value of $n = 100$ was used with $k = 2$ and $k = 3$.

According to these results, we have chosen to use NK landscapes with parameters $n = 30$, 50, 80 and 120 and $k = 2$, 3 and 4. The interactions between bits and the values in the lookup tables were generated uniformly at random, as in [18]. For our empirical investigations, we used multi-start (1+1) EA and confronted it with MEA with equivalent values both for initial population size and the predicted population size, on which the value would stabilize if there was no reinforcement mechanism. In the next section, we discuss the way how we determined the parameter values for both MEA and the multi-start (1+1) EA.

4.2 Tuning the Performance of MEA

The tuning process for parameters of MEA is partially based on investigations in [2] and partially on some new ideas. The general approach is based on observing the influence of changing one parameter, while the other parameters remain fixed. However, reasonable values for some of the parameters can be determined indirectly.

For determining the initial population size and the size of the elite list $|P_0| = |L_e|$, we simply perform several short runs of multi-start (1+1) EA on several NK landscapes. The values for population sizes were 1, 5, 10, 20 and 50. For all population sizes, we performed 10 runs on 10 different independently generated NK landscapes with the same parameters (since NK landscapes are stochastic functions), where each run was limited to 1 second. CPU time limit was chosen as a relatively fair stopping criterion for these experiments and it also gives us the control of the length of the experiment. The results are given in Table 1.

The results in Table 1 suggest that even the small value $|P| = 5$ leads to solid improvement of results, when we compare it to the simple (1+1) EA. For $|P| = 10$ and $|P| = 20$, we interestingly often obtain a worsening, which can perhaps be explained by the fact that the local search for a single individual is shorter. However, for $|P| = 50$, the results start to be competitive again. The high number of regions, where the search is performed, seems to outweigh the shorther length of the local search. We add that this investigation is useful also for determining good values for the initial lifespan q and the period of birth T_b. It can be easily shown that if reward / punishment factor was not taken into account, the value, on which the population size would stabilize over time, would be q/T_b.

For the rest of our tuning experiments, we chose $|P_0| = |L_e| = q/T_b = 50$. The remaining values for parameters of MEA have to be tested by changing the value of one parameter and fixing the values for other parameters. After a series of preliminary experiments, we chose a generic configuration of MEA with the following parameters: initial lifespan $q = 50$, period of birth $T_b = 1$, basic local search length $\gamma = 10$ and $r = 1$. Practically, only 3 of these parameters have to be tuned, since q/T_b is fixed to a constant. In these second series of experiments, we extended the time limit to 10 seconds. Both the number of generated NK landscapes and the number of repeated runs of MEA for a single NK landscape remained 10. The following tuning were all performed on instances with $n = 80$ and $k = 4$.

Table 2 summarizes the experiments, where we changed the value T_b, which automatically led to $q = 50T_b$. We merged the results of these experiments with results for the basic local search length parameter γ, since there clearly is a correlation between these parameters, which together influence the number of iterations that are performed by an agent during its life. From the values in Table 2, we can see that the results obtained here (fitness values $61.827 - 62.160$) are a little better than the results obtained in the first series of experiments (fitness values $59.709 - 61.400$). However, this could be due to the higher time limit. Thus, this will be discussed further later, when we compare MEA to multi-start (1+1) EA with the same population size and time limit. MEA seems to give slightly better results for lower values of γ. This can possibly be explained by the fact that for NK landscapes with higher level of multimodality, a search algorithm, which explores the search space in more possible ways can have a better chance to escape local optima. However, changing the values of q and T_b does not seem to have a significant impact. With small values of γ, the quality

Table 2: The average fitness values for the best individuals obtained by MEA with different values of T_b and γ on NK landscapes with $n = 80$ and $k = 4$.

	$\gamma = 1$	$\gamma = 2$	$\gamma = 5$	$\gamma = 10$	$\gamma = 20$	$\gamma = 30$
$T_b = 1$, $q = 50$	62.136	62.160	62.055	61.893	62.031	62.033
$T_b = 2$, $q = 100$	62.056	61.827	61.896	61.908	61.987	61.960
$T_b = 3$, $q = 150$	62.037	61.936	61.930	61.970	62.017	61.900

Table 3: The average fitness values for the best individuals obtained by MEA with different values of r on NK landscapes with $n = 80$ and $k = 4$.

$r = 0$	$r = 1$	$r = 2$	$r = 3$	$r = 5$	$r = 10$
62.171	62.340	62.316	62.174	62.134	61.881

seems to grow very slightly but for larger values of γ, the obtained fitness declines. Therefore, it seems to be safer to choose lower values for T_b, since (1+1) EA does not seem to need many iterations for an improvement but it has to choose the right sequence of random decisions in the search process. This observation differs from the results in graph coloring, where a different local search subroutine was used [2].

In the last series of tuning experiments, we investigate the influence of r, i.e. the punishment parameter on the quality of results. We used 6 different values of r and compared it on the instances with $n = 80$ and $k = 4$. The results are presented in Table 3. For $r = 0$, we evidently have a situation, when the population size can only increase, since we have only positive reinforcement for the agents and their lifespans. From the values of fitness, it seems that for low non-zero values of r, the model is quite balanced and the agents are able to obtain a very good fitness. On the other hand, for $r = 10$, we clearly obtain the worst result, which indicates that the agents are eliminated too soon.

4.3 Comparing the Performance of MEA and Multi-start (1+1) EA

This section is dedicated to a comparison between simple (1+1) EA, multi-start (1+1) EA and MEA on different NK landscapes. This is the core of our experiments, where we show that it makes sense to introduce the multiagent nature and the lifespans to the search process.

We used the simple (1+1) EA with probability of bit flip $1/n$ for each bit. The multi-start (1+1) EA was the same implementation with 50 starts from different randomly generated initial solutions. MEA was used with parameters $|P_0| = |L_e| = 50$, $q = 50$, $T_b = 1$, $r = 1$ and $\gamma = 1$. As in the tuning experiments, all algorithms were run 10 times on 10 independently generated NK landscapes with the same values of n and k. Each run of each algorithm was limited to 10 seconds.

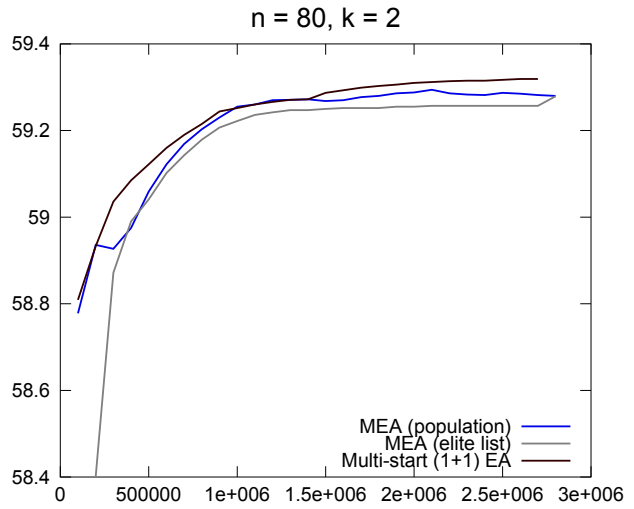
Table 4 shows the average fitness values obtained over all runs of each algorithm on all NK landscapes with particular values of n and k. Thus, all the three algorithms were tested on the same instances, with the same fitness values of local and global optima. This, along with the stopping criterion defined by time limit, makes the comparison very fair, since the algorithms have the same practical conditions.

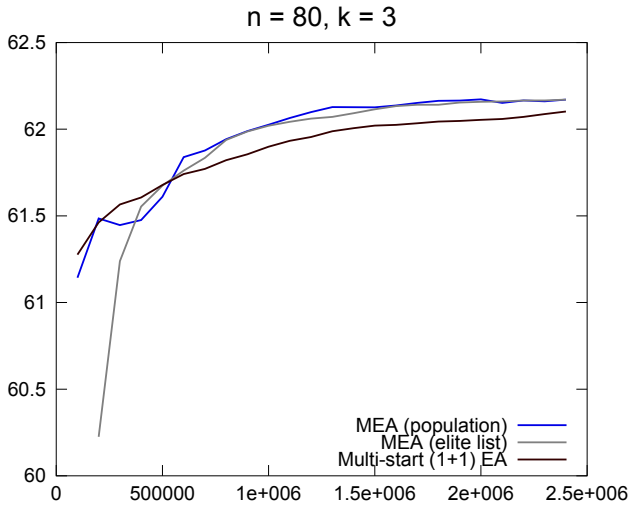
Table 4: A comparison of average fitness values obtained by simple (1+1) EA, multi-start (1+1) EA with population size 50 and MEA with the same initial population size on different NK landscapes.

n	k	simple (1+1) EA	multi-start (1+1) EA	MEA
n = 30	k = 2	22.283	**22.413**	**22.413**
	k = 3	22.397	22.778	**22.779**
	k = 4	23.121	**23.826**	23.819
n = 50	k = 2	37.026	**37.433**	37.424
	k = 3	37.849	**38.761**	38.716
	k = 4	37.680	**39.178**	39.165
n = 80	k = 2	58.882	**59.619**	59.604
	k = 3	59.812	61.620	**61.632**
	k = 4	60.155	61.879	**61.881**
n = 120	k = 2	88.242	89.287	**89.430**
	k = 3	89.828	91.607	**91.844**
	k = 4	89.974	91.866	**92.294**

From Table 4, we can see that the average improvement obtained by using a multi-start version of (1+1) EA ranges from 0.13 to 1.892. By introducing the multiagent nature of MEA, we obtain a change in the average fitness, which seems to depend on the values of n and k. For $n = 30$, the results of multi-start (1+1) EA and MEA are comparable, which is probably due to the easiness of the instances. For $n = 50$, MEA is slightly worse, with the average decline ranging from 0.013 to 0.045. However, for $n = 80$, we have a situation, where for $k = 2$ MEA is still worse but for $k = 3$ and 4, it becomes dominant. Finally, for $n = 120$, we obtain that MEA clearly beats multi-start (1+1) EA by a factor ranging from 0.143 to 0.428.

At this point, let us further discuss how the fitness values change both in MEA and multi-start (1+1) EA at some point of the algorithms. This provides some insight into the difference between behavior of these algorithms. In Figure 1 and Figure 2, we plot the profiles of average obtained fitness values during the optimization over 10 indepedent runs on 10 different NK landscapes with the same values of n and k. The x axis contains the number of fitness calls, while the y axis contains the currently best fitness values. For MEA,





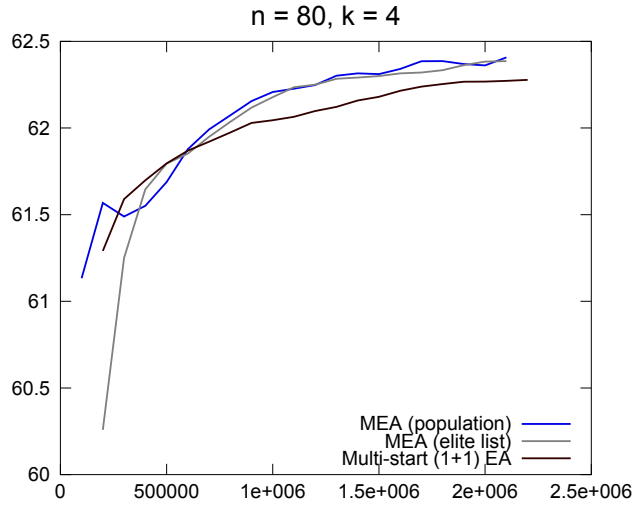
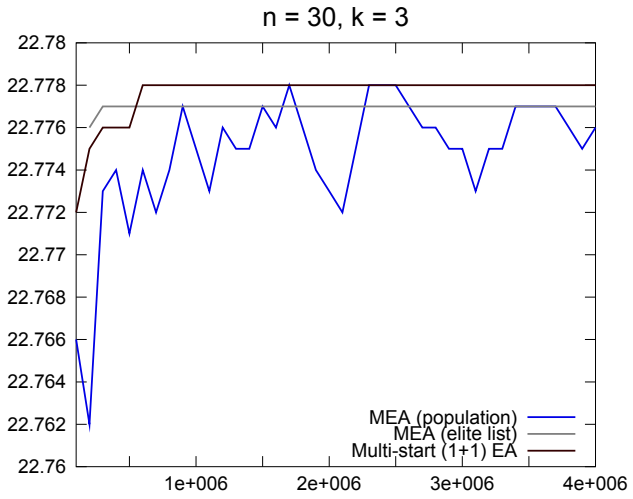
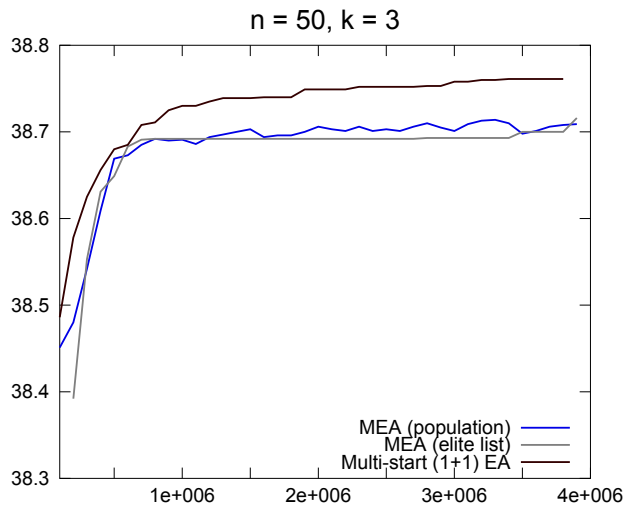


Figure 1: The illustration of the change in fitness of the best solutions for MEA (both the population and the elite list) and multi-start (1+1) EA on NK landscapes with $n = 50$, $2 \leq k \leq 4$. The x-axis denotes the number of fitness calls and the y-axis denotes the average value of best fitness obtained over the runs of the algorithms.





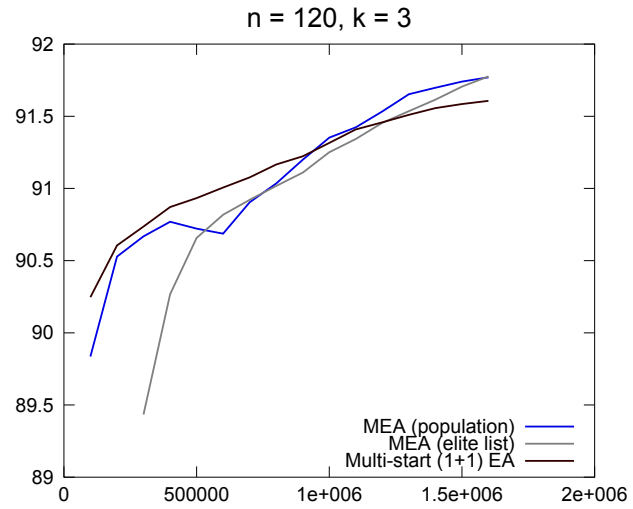


Figure 2: The illustration of the change in fitness of the best solutions for MEA (both the population and the elite list) and multi-start (1+1) EA on NK landscapes with $n = 30$, 50 and 120 and $k = 3$. The x-axis denotes the number of fitness calls and the y-axis denotes the average value of best fitness obtained over the runs of the algorithms.

we include the best values both for the population and the elite list. Figure 1 illustrates the fitness profiles for instances with $n = 80$. These pictures show that MEA is better than multi-start (1+1) EA for $k = 3$ and 4, while for $k = 2$, the situation is reversed. Also, the profile of MEA indicates that at the beginning, its behavior is similar to multi-start (1+1) EA. However, a mild decline in fitness tends to occur after the initial population starts to be eliminated. Then, MEA slowly starts to beat multi-start (1+1) EA.

In Figure 2, we illustrate the influence of n on the results, where we have the profiles for instances $n = 30$, 50 and 120 and $k = 3$. For $n = 30$, the fitness profile of MEA clearly indicates that unlike multi-start (1+1) EA, our algorithm naturally attempts to diversify the search. For $n = 120$, we can also see that MEA starts to obtain better solutions for large instances slowly, however, reasonably. In all cases, the observed behavior of MEA indicates that this is an algorithm, which relies on a temporary worsening of fitness, similarly to simulated annealing [10, 22].

Thus, MEA not merely improves the results of simple (1+1) EA, but it also is able to significantly improve the results of a population-based version of (1+1) EA with the same initial population size. Additionally, the improvement occurs in harder instances of the problem. Hence, MEA clearly does not profit only from the parallel search in more different regions. This is a highly encouraging result, since for the first time, we managed to show very clearly that the improvement is obtained purely on the basis of lifespans, well-tuned restarts and variable population size. When we compare these results to those published in [2], we can see more clearly that the mechanisms used in MEA are not only a collection of techniques, which were somehow able to obtain solid results in graph coloring. In addition, NK landscapes with the chosen values of k are quite representative fitness functions, since the problem is NP-hard but in some sense, it is "easier" because of the low values of k, which induce only a low level of interactions between components of the problem. Despite this fact, we can see that even on such problems, multi-start (1+1) EA can be beaten by using the emergent multiagent nature of MEA.

5. CONCLUSIONS

We proposed an adaptation of a relatively recently introduced multiagent evolutionary algorithm (MEA), which was originally intended as a metaheuristic approach to graph coloring problem. However, MEA uses relatively general mechanisms, where the core idea is based on simulation of a life cycle of agents, which perform local search. Each agent has its lifespan, which is influenced by its ability to improve fitness. Thus, MEA is an algorithm with variable population size. In this paper, MEA was adapted to search on NK fitness landscapes, which are very popular in evolutionary computation for the tunability of ruggedness. Additionally, unrestricted NK landscapes, which we used in this paper, are a typical example of an NP-hard problem.

Our experimental results confirmed that the mechanisms used in MEA are beneficial in optimization on NK fitness landscapes. This is the first time, when we were able to show that the improvement by MEA is definitely caused by the lifespans, reinforcement mechanisms and the well-tuned restarts, which can be considered to be a variant of "swarming" effect. For harder instances of the problem, MEA achieved higher fitness than an equivalent multi-start (1+1) EA with the same population size. The improvement factor achieved by MEA on NK landscapes for high values of n and k was encouraging. This definitely shows that the emergent multiagent nature of MEA is beneficial in optimization, especially on larger and more rugged landscapes.

Acknowledgement.

The author would like to thank Jiří Pospíchal and the anonymous referees, for their helpful comments on this work. This contribution was supported by Grant Agency VEGA SR under the grants 1/0553/12 and 1/0458/13.

6. REFERENCES

[1] I. Araya, L. Pérez, and M. C. Riff. Towards a population-based framework for improving stochastic local search algorithms. In T. Soule and J. Moore, editors, *Proceedings of the 14th international conference on Genetic and evolutionary computation conference*, GECCO '12, pages 337–344, New York, NY, USA, 2012. ACM.

[2] D. Chalupa. Population-based and learning-based metaheuristic algorithms for the graph coloring problem. In N. Krasnogor and P. L. Lanzi, editors, *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, GECCO '11, pages 465–472, New York, NY, USA, 2011. ACM.

[3] S. S. Choi, K. Jung, and J. H. Kim. Phase transition in a random NK landscape model. In H. G. Beyer and U. M. O'Reilly, editors, *Proceedings of the 7th international conference on Genetic and evolutionary computation*, GECCO '05, pages 1241–1248, New York, NY, USA, 2005. ACM.

[4] S. Droste, T. Jansen, and I. Wegener. On the analysis of the (1+1) evolutionary algorithm. *Theor Comput Sci*, 276(1-2):51–81, 2002.

[5] Y. Gao and J. C. Culberson. An analysis of phase transition in NK landscapes. *J Artif Intell Res*, 17:309–332, 2002.

[6] L. Kapsokalivas, X. Gan, A.A. Albrecht, and K. Steinhöfel. Population-based local search for protein folding simulation in the MJ energy model and cubic lattices. *Comput Biol Chem*, 33(4):283 – 294, 2009.

[7] D. Karaboga and B. Basturk. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. *J Global Optim*, 39(3):459–471, 2007.

[8] S. Kauffman. Adaptation on rugged fitness landscapes. In D. L. Stein, editor, *Lecture Notes in the Sciences of Complexity*, pages 527–618. Addison Wesley, 1989.

[9] S. Kauffman and S. Levin. Towards a general theory of adaptive walks on rugged landscapes. *J Theor Biol*, 128(1):11–45, 1987.

[10] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

[11] V. Kvasnička, M. Pelikán, and J. Pospíchal. Hill climbing with learning (an abstraction of genetic algorithm). *Neural Netw World*, 6:773–796, 1995.

[12] J. Liu, W. Zhong, and L. Jiao. A multiagent evolutionary algorithm for combinatorial optimization

problems. *IEEE T Syst Man Cy B*, 40(1):229 –240, 2010.

[13] P. Merz. Advanced fitness landscape analysis and the performance of memetic algorithms. *Evol Comput*, 12(3):303–325, 2004.

[14] C. Morgenstern. Distributed coloration neighborhood search. In D. S. Johnson and M. Trick, editors, *Cliques, Coloring, and Satisfiability: Second DIMACS Implementation Challenge*, pages 335–358. 1996.

[15] G. Ochoa, M. Tomassini, S. Vérel, and C. Darabos. A study of NK landscapes' basins and local optima networks. In C. Ryan and M. Keijzer, editors, *Proceedings of the 10th international conference on Genetic and evolutionary computation*, GECCO '08, pages 555–562, New York, NY, USA, 2008. ACM.

[16] J. M. Pasia, K. F. Doerner, R. F. Hartl, and M. Reimann. A population-based local search for solving a bi-objective vehicle routing problem. In C. Cotta and J. Hemert, editors, *Evolutionary Computation in Combinatorial Optimization*, volume 4446 of *Lecture Notes in Computer Science*, pages 166–175. Springer Berlin Heidelberg, 2007.

[17] M. Pelikan. Analysis of estimation of distribution algorithms and genetic algorithms on NK landscapes. In C. Ryan and M. Keijzer, editors, *Proceedings of the 10th international conference on Genetic and evolutionary computation*, GECCO '08, pages 1033–1040, New York, NY, USA, 2008. ACM.

[18] M. Pelikan. NK landscapes, problem difficulty, and hybrid evolutionary algorithms. In M. Pelikan and J. Branke, editors, *Proceedings of the 12th annual conference on Genetic and evolutionary computation*, GECCO '10, pages 665–672, New York, NY, USA, 2010. ACM.

[19] A. M. Sutton, D. Whitley, and A. E. Howe. A polynomial time computation of the exact correlation structure of k-satisfiability landscapes. In F. Rothlauf, editor, *Proceedings of the 11th international conference on Genetic and evolutionary computation*, GECCO '09, pages 365–372, New York, NY, USA, 2009. ACM.

[20] A. M. Sutton, D. Whitley, and A. E. Howe. Mutation rates of the (1+1)-EA on pseudo-boolean functions of bounded epistasis. In N. Krasnogor and P. L. Lanzi, editors, *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, GECCO '11, New York, NY, USA, 2011. ACM.

[21] M. Tairan and Q. Zhang. Population-based guided local search: Some preliminary experimental results. In *IEEE Congress on Evolutionary Computation 2010 (CEC)*, pages 1–5, 2010.

[22] V. Černý. Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm. *J Optimiz Theory App*, 45(1):41–51, 1985.

[23] A. H. Wright, R. K. Thompson, and J. Zhang. The computational complexity of N-K fitness functions. *IEEE T Evolut Comput*, 4(4):373–379, 2000.