2012 11th International Conference on Machine Learning and Applications

# Increasing Efficiency of Evolutionary Algorithms by Choosing between Auxiliary Fitness Functions with Reinforcement Learning

Arina Buzdalova
St. Petersburg National Research University
of Information Technologies, Mechanics and Optics
49 Kronverkskiy prosp.
Saint-Petersburg, Russia, 197101
Email: abuzdalova@gmail.com

Maxim Buzdalov
St.Petersburg National Research University
of Information Technologies, Mechanics and Optics
49 Kronverkskiy prosp.
Saint-Petersburg, Russia, 197101
Email: mbuzdalov@gmail.com

***Abstract*—In this paper further investigation of the previously proposed method of speeding up single-objective evolutionary algorithms is done. The method is based on reinforcement learning which is used to choose auxiliary fitness functions. The requirements for this method are formulated. The compliance of the method with these requirements is illustrated on model problems such as Royal Roads problem and H-IFF optimization problem. The experiments confirm that the method increases the efficiency of evolutionary algorithms.**

## I. Introduction

This paper is dedicated to improvement of the efficiency of single-objective evolutionary computation. Usually the aim of the evolutionary algorithms (EA) is to find an individual that maximizes the objective, or the fitness function in terms of evolutionary computation.

Sometimes additional fitness functions can be used in order to enhance the efficiency of an optimization algorithm. For example, such approach is presented in [1], where some single-objective optimization problems are multi-objectivised and solved with various multi-objective evolutionary algorithms (MOEAs) [2]. This approach allows to avoid local optima and also increases the diversity of individuals [3]. It should be noted that the additional objectives are specially developed and are known to have some useful qualities. Developing such objectives, also known as helper-objectives or helpers, and choosing the most appropriate ones is a sophisticated problem [3].

Additional fitness functions can also be provided by the object domain [4]. In this case we do not have any prior knowledge about their properties. Some of them may be useful at different stages of single-objective optimization, but it is unknown which fitness function should be used actually. So it is important to have some instrument that allows to automatically choose the optimal fitness function.

In this paper, a method of increasing the efficiency of a single-objective EA by choosing between some additional, or *auxiliary*, fitness functions is investigated. The objective to be maximized is called the *target* fitness function. The function that is optimized at each particular optimization stage is chosen dynamically from the set of auxiliary fitness functions during the EA run. The proper choice leads to increase of the target fitness function. There is no prior knowledge about the auxiliary fitness functions properties. We do not aim to maximize them, they are just used to increase the efficiency of the target fitness function optimization.

The method being investigated was previously described in [5], [6]. In this article we formulate some formal requirements for this method and check its ability to ignore *obstructive* fitness functions. Choosing such functions for optimization can lead to decrease of the target fitness function. Note that multi-objective algorithms are not able to fully ignore obstructive fitness functions as long as they are designed to maximize all the objectives. This assumption is confirmed in the present article with experimental results of solving a model problem.

The choice of the optimal auxiliary fitness functions is made with reinforcement learning (RL) [7]. The method will be further referred to as *EA + RL*. To denote the particular applications of the method, we use a notation *A+L* where *A* is the name of the used EA and *L* is the particular reinforcement learning method. For example, the name of the genetic algorithm *(GA)* guided with *Q-learning* is *GA + Q-learning*. It can be seen as a method of "on-the-fly" EA adjusting. To our knowledge, in other EA-adjusting methods some fixed fitness function is usually tuned [8], [9]. What is more, RL applicability for EA adjusting is not investigated yet. There are few works that explore adjusting of such parameters as probabilities of applying evolutionary operators or some quantitative properties of individuals generation using RL [9], [10]. This work continues investigation of RL applicability for adjusting fitness functions in EAs after [5], [6].

In the next section of this article the EA + RL method is described in more details. Then the requirements for this method are formulated. In order to formulate them, the efficiency measure is defined and some auxiliary functions classification is introduced. After that, two experiments with different auxiliary

978-0-7695-4913-2/12 $26.00 © 2012 IEEE
DOI 10.1109/ICMLA.2012.32

function types are described. Both the experiments involve obstructive fitness functions. The results of the experiments confirm fulfillment of a part of the requirements. Then all the results obtained with EA + RL are reviewed and it is shown that the EA + RL is able to fulfill all the requirements. Finally, some conclusions are made.

## II. Method description

The EA + RL method is based on guiding an EA by choosing fitness functions with a RL algorithm. It is implied that there are the target fitness function that should be maximized by the EA and a set of auxiliary fitness functions.

Recall that RL algorithms are designed to find an optimal behavioral strategy in an interactive environment. An agent chooses an action, applies it to the environment and receives the reward for this action, as well as some representation of the environment state. The goal is to maximize the total reward [7].

In our method the EA is considered as the environment. The action is to choose the fitness function to be used. Either the target fitness function or an auxiliary one can be chosen. The fitness function is chosen each time when a next generation of the EA should be evolved. The reward is based on the difference of the target fitness function values in two sequential generations.

It is proved for a number of RL algorithms that the optimal strategy is eventually found and the total reward is maximized [7], [11]. So the proposed method maximizes the total target fitness difference. In the following subsections some aspects of the method will be described more formally.

### A. Optimization problem with auxiliary fitness functions

Consider the formulation of the optimization problem with auxiliary fitness functions that is solved with the EA + RL method. Let $W$ be a discrete search space. Denote all acceptable solutions contained in the search space by $X$, $X \subseteq W$. Consider the target fitness function $g\colon W \to \mathbb{R}$.

Consider an *auxiliary set* $H$ consisting of $k$ auxiliary fitness functions: $H = \{h_i(x)\}_{i=1}^{k}, h_i\colon W \to \mathbb{R}$.

The problem is to maximize the target fitness function using the auxiliary fitness functions to speed up the optimization process if it is possible: $g(x) \to \max_{x \in X}, X \subseteq W$.

The solution of the problem is $x^* \in X\colon g(x^*) \geq g(x), \forall x \in X$.

Note that there is no prior knowledge about an auxiliary fitness function properties. We do not develop auxiliary functions, they are already given. So the proposed method should be able to deal with an arbitrary set of auxiliary fitness functions.

### B. Reinforcement learning task

Let us briefly describe the problem of increasing the efficiency of EA as the RL task [7]. The more detailed description can be found in work [6].

Let $x$ be an individual evolved by the EA. Denote the $i$-th generation by $G_i$. The set of actions $A$ corresponds to the set of all fitness functions, consisting of $g$ — the target fitness function and the elements of $H$ — the set of auxiliary fitness functions. Taking an action means choosing some fitness function $f_i \in A$ to be used in the generation $G_i$.

Consider the best individual contained in the $G_i$ in terms of the currently chosen fitness function: $z_i = \arg\max_{x \in G_i} f_i(x)$. Also consider a fitness difference in two sequential generations: $\Delta(f, i) = \frac{f(z_i) - f(z_{i-1})}{f(z_i)}, f \in A$.

We map the generations of individuals to the states of the environment. The state $s_i$ corresponding to the generation $G_i$ is a vector of criteria $f \in A$ sorted in descending order of the $\Delta(f, i)$ values: $s_i = \langle f_1, f_2, \dots f_{k+1} \rangle | \Delta(f_1, i) \geq \Delta(f_2, i) \geq \dots \Delta(f_{k+1}, i)$. If $\Delta(f_a, i)$ is equal to $\Delta(f_b, i)$, then $f_a, f_b$ are placed in some predefined order.

Finally, the reward function $R\colon S \times A \to \{0, \frac{1}{2}, 1\}$, that is calculated after choosing the fitness function $f_i$ in the state $s_{i-1}$ and generating $G_i$, is defined:

$$R(s_{i-1}, f_i) = \begin{cases} 1 & \text{if } g(z_i) - g(z_{i-1}) > 0, \\ \frac{1}{2} & \text{if } g(z_i) - g(z_{i-1}) = 0, \\ 0 & \text{if } g(z_i) - g(z_{i-1}) < 0. \end{cases}$$

Note that the reward depends on the difference between the target fitness of the best individuals at sequential generations and is the highest when the target fitness increases.

## III. Requirements for the method efficiency

In this section the requirements for the developed method are formulated. They are based on the auxiliary fitness functions classification, that divides the fitness functions into supporting and obstructive ones. The aim of the method is to increase the efficiency of the EA if it is possible. So the efficiency measure is suggested for the evolutionary algorithms.

### A. Efficiency measure

First of all, we should formulate the efficiency measure for the evolutionary algorithms. We limit the number of generations that can be evolved. The optimization is performed until the optimal solution is found or the generations limit is reached. The efficiency measure is equal to the number of actually evolved generations. The smaller it is, the higher is the efficiency of the EA. If the generations limit is reached, but the optimal solution is not found, the EA is found ineffective.

### B. Auxiliary fitness functions classification

Let us divide auxiliary fitness functions in two groups. The first group is *supporting* fitness functions. When they are being optimized, the target fitness function grows more rapidly. The rest of fitness functions are *obstructive* ones. If they are being optimized, the target fitness function can slow its growth or even start to decrease. If target fitness function behavior does not change, the auxiliary fitness function being optimized is also considered to be obstructive.

Notice that there can be three possible configuration types of the auxiliary set:

1) *obstructive only*: there are no supporting fitness functions, but at least one obstructive;
2) *supporting only*: there is at least one supporting fitness function and no obstructive ones;
3) *supporting and obstructive*: there is at least one fitness function of each type.

### C. List of requirements

The requirements for the developed EA + RL method according to the possible configurations of the auxiliary set are the following:

1) for the *obstructive only* auxiliary set, EA + RL efficiency should be asymptotically equal to that of the original EA;
2) for the *supporting only* auxiliary set, EA + RL should asymptotically outperform the original EA;
3) for the *supporting and obstructive* auxiliary set, EA + RL should also asymptotically outperform the original EA;
4) (dynamic requirement) EA + RL should respond to changing conditions of the auxiliary fitness functions.

In other words, the EA + RL should always outperform the EA when there is at least one supporting fitness function, and it should never be asymptotically worse. It also should be able to work under conditions of auxiliary fitness functions with changing properties. It means that a function may be both supporting or obstructive at different optimization stages.

EA + RL compliance with the requirements 2–4 is already illustrated in [5], [6] and also briefly presented in the last section. Requirement 1, as well as the requirement 3, which are associated with obstructive auxiliary functions, are investigated further in the present paper. Two model problems are taken and obstructive fitness functions are added to their formulations. The first model problem, Royal Roads, is used to illustrate EA + RL ability to ignore the obstructive function. It corresponds to the first requirement. The second model problem, H-IFF, is used to illustrate EA + RL ability to choose between the supporting fitness functions and obstructive one. It demonstrates fulfillment of the third requirement.

## IV. Problem with obstructive function

In this section a problem with auxiliary set of *obstructive only* type is described. It is showed that EA + RL performs equally good with EA despite the presence of the obstructive function, so it fulfills the corresponding requirement.

### A. Royal Roads problem

Consider the Royal Roads model problem [12]. In its original formulation there is only one target fitness function $f$. The individuals are bit strings of a fixed length $l$, which are split into blocks of equal length $b$. Count the number of blocks completely filled with ones in an individual, let it be $n$. The value of the target fitness function $f$ calculated on such an individual is $bn$. In other words, the target fitness function is the number of blocks filled with ones multiplied by the length of a block.

Now let us construct an auxiliary set consisting of only one obstructive fitness function. This function $\theta$ calculates the number of zeros in an individual. Notice that increase of $\theta$ leads to decrease of $f$, so $\theta$ is an obstructive fitness function. Notice that it is easier to optimize linear function $\theta$ than to optimize piecewise constant function $f$, that makes it more risky to use this obstructive function. So, the auxiliary set is $H = \{\theta\}$.

### B. Experiment description

During the experiment, the original Royal Roads problem and the Royal Roads problem with an obstructive auxiliary fitness function were solved by a number of algorithms. The length of an individual was $l = 64$ and the length of a block was $b = 8$. Each algorithm were run 50 times with its parameters fixed in order to gather statistics. Each run was performed until the optimal solution (the string of one-bits) was found or the steps limit of 500000 steps were reached. The parameter values of the algorithms were chosen manually during the preliminary experiment.

The original Royal Roads problem with target fitness function $f$ was solved by (1 + 1) evolutionary strategy (ES). There was one individual in each generation. A single child was evolved at each step by flipping one randomly chosen bit. Then the parent was replaced with the child, if the child's fitness were higher than the parent's one.

The problem with the target fitness function $f$ and the obstructive auxiliary function $\theta$ was solved with the same evolution strategy adjusted by different kinds of RL algorithms. The parameter values used in these algorithms are shown in the Table I.

In R-learning algorithm [13] $\varepsilon$-greedy exploration strategy [7] was used. It chose an arbitrary fitness function with probability of $\varepsilon$ and the function with the maximal estimated "quality" with probability of $1-\varepsilon$. In Q-learning algorithm [7] greedy strategy was used, so it always chose the fitness function with the maximal estimated "quality". Both greedy and $\varepsilon$-greedy strategies were tested during the preliminary experiment and the ones that provided the corresponding algorithms with the highest efficiency were chosen. In the Delayed Q-learning algorithm a special safe exploration strategy is used [11], that is a part of this algorithm.

As it is shown in the next section, Delayed Q-learning algorithm appeared to be the most effective. So an additional experiment with individual lengths $l = 128, 256, 512, 1024$ was performed using this algorithm.

### C. Experiment results

The results of the experiment are shown in the Table II. The runs in which the optimal solution (the string of all one-bits) was found are called successful. In the column "Success" the percent of successful runs is specified. The "Average steps" column shows the average number of generations in the successful runs. For the runs during which the optimal solution was not evolved, the number of steps is denoted by the infinity sign.

TABLE I
RL PARAMETERS USED IN SOLVING ROYAL ROADS

| Parameter | Description | Value |
|---|---|---|
| Q-learning [7] | | |
| $\alpha$ | learning rate | 0.01 |
| $\gamma$ | discount factor | 0.1 |
| R-learning [13] ($\varepsilon$-greedy) | | |
| $\alpha$ | learning rate for reward $\rho$ | 1 |
| $\beta$ | learning rate for R-values | 0 |
| $\varepsilon$ | exploration probability | 0.001 |
| Delayed Q-learning [11] | | |
| $m$ | update period | 50 |
| $\gamma$ | discount factor | 0.1 |
| $\epsilon$ | bonus reward | 0.2 |

TABLE II
RESULTS OF SOLVING ROYAL ROADS WITH VARIOUS ALGORITHMS, INDIVIDUAL LENGTH = 64

| Algorithm | Success | Average steps | Max steps | Min steps | $\sigma$ |
|---|---|---|---|---|---|
| Royal Roads problem | | | | | |
| (1+1) ES | 100% | 6913.28 | 16033 | 2439 | 2925.07 |
| Royal Roads problem with obstructive fitness function | | | | | |
| (1+1) ES + Delayed | 88% | 8365.52 | $\infty$ | 3982 | 3216.13 |
| (1+1) ES + R-learning | 100% | 69881.74 | 289936 | 5254 | 67990.07 |
| (1+1) ES + Q-learning | 24% | 6964.17 | $\infty$ | 3012 | 2256.70 |

The (1+1) ES + Delayed Q-learning appeared to be the most effective EA + RL algorithm. It was successful in 88% runs with the average number of steps comparable with the number of steps in EA algorithm without obstructive fitness functions. Conceivably, the experience gathered by the Delayed Q-learning algorithm allowed it to consider the obstructive function as profitless and not to choose it. At the same time R-learning with $\varepsilon$-greedy exploration strategy was unable to eliminate the use of an obstructive fitness function because it was choosing an obstructive fitness function with probability of $\frac{1}{2}\varepsilon$. Notice that the Q-learning algorithm that performed in a greedy way was mostly unsuccessful. It can be explained by the fact that it had less chances to fully explore the environment. So the Delayed Q-learning used to combine exploration and exploitation in the most efficient way.

TABLE III
RESULTS OF SOLVING ROYAL ROADS WITH VARIOUS LENGTHS

| Length | Success | Average steps | Max steps | Min steps | $\sigma$ |
|---|---|---|---|---|---|
| (1 + 1) ES without obstructive fitness function | | | | | |
| 64 | 100% | 6913.28 | 16033 | 2439 | 2925.07 |
| 128 | 100% | 15044.62 | 29470 | 8750 | 4714,94 |
| 256 | 100% | 37419.64 | 65574 | 18428 | 12805,84 |
| 512 | 100% | 87982.58 | 160933 | 39521 | 25676.64 |
| 1024 | 100% | 216750.30 | 390530 | 132807 | 56050.36 |
| (1+1) ES + Delayed Q-learning with obstructive fitness function | | | | | |
| 64 | 88% | 8365.52 | $\infty$ | 3982 | 3216.13 |
| 128 | 96% | 17081,19 | $\infty$ | 6967 | 5423.95 |
| 256 | 100% | 40179,80 | 68589 | 21805 | 12739.42 |
| 512 | 100% | 92705.12 | 169707 | 58841 | 22133.56 |
| 1024 | 100% | 212631,84 | 361548 | 128803 | 48220.06 |

The results of the second part of the experiment are shown in Table III. The EA and EA + Delayed Q-learning efficiency was measured on different individual lengths. The results confirm that the EA + Delayed Q-learning is asymptotically equal to the EA. So the requirement for the *obstructive only* auxiliary set is fulfilled.

## V. PROBLEM WITH SUPPORTING AND OBSTRUCTIVE FUNCTIONS

In this section, the efficiency of EA + RL on a model problem with auxiliary set consisting of functions of both supporting and obstructive types is investigated. It is experimentally shown that in this case the EA + RL algorithm outperforms the EA despite the presence of an obstructive fitness function.

### A. H-FF optimization problem

Originally, the H-IFF function [1] is used to test genetic algorithms. Its target fitness function formula $f$ is given in (1), where $B$ is a bit string individual, $B_L$ and $B_R$ are its left and right halves respectively.

$$f(B) = \begin{cases} 1 & \text{if } |B| = 1, \\ |B| + f(B_L) + f(B_R) & \text{if } \forall i\{b_i = 0\} \| \forall i\{b_i = 1\}, \\ f(B_L) + f(B_R) & \text{otherwise.} \end{cases} \tag{1}$$

H-IFF optimization problem can be multi-objectivized in order to avoid getting stuck in local optima while solving it with single-objective evolutionary algorithms [1]. Multi-objectivized H-IFF optimization problem is called MH-IFF. It can be efficiently solved with multi-objective evolutionary algorithms. Additional criteria corresponding to the MH-IFF problem are $f_0$ and $f_1$ (2).

$$f_k(B) = \begin{cases} 0 & \text{if } |B| = 1 \text{ and } b_1 \neq k, \\ 1 & \text{if } |B| = 1 \text{ and } b_1 = k, \\ |B| + f_k(B_L) + f_k(B_R) & \text{if } \forall i\{b_i = k\}, \\ f_k(B_L) + f_k(B_R) & \text{otherwise.} \end{cases} \tag{2}$$

Let us construct the auxiliary set in order to solve H-IFF with EA + RL. The supporting functions are $f_0$ and $f_1$. It is showed in [1] that when they are optimized, the better solutions for the $f$ are found.

Now we add an obstructive fitness function $\theta$ to the described auxiliary set. It counts the number of overlaps with a bit mask of alternating ones and zeros: $1010\ldots10$. Optimizing such function destroys blocks of equally valued bits searched in the H-IFF problem. An experiment of dealing with this function and the experiment results are described in the following sections.

### B. Experiment description

Three variations of the H-IFF problem were solved with different algorithms. Firstly, original H-IFF without any auxiliary functions was optimized with (1 + 5) evolution strategy (ES). The corresponding mutation operator flipped one randomly chosen bit of each individual.

TABLE IV
H-IFF OPTIMIZATION RESULTS

| Algorithm | Best fitness | Average fitness | $\sigma$ | Successful runs |
|---|---|---|---|---|
| H-IFF problem | | | | |
| (1+5) ES | 216 | 179.07 | 16.99 | 0% |
| H-IFF problem with supporting fitness functions | | | | |
| (1+5) ES + R-learning | 448 | 448.00 | 0.00 | 100% |
| PESA-II | 448 | 448.00 | 0.00 | 100% |
| H-IFF problem with supporting and obstructive fitness functions | | | | |
| (1+5) ES + R-learning | 448 | 439.45 | 36.32 | 92% |
| PESA-II | 312 | 277.83 | 20.07 | 0% |

Secondly, two supporting auxiliary fitness functions $f_0, f_1$ were added. The corresponding MH-IFF problem was solved with a multi-objective evolutionary algorithm PESA-II [14] and the proposed ES + R-Learning method that adjusted the same (1 + 5) evolution strategy. The parameters of R-learning algorithm were $\alpha = 0.5$ and $\beta = 0.35$ [13]. The $\varepsilon$-greedy exploration strategy with $\varepsilon = 0.25$ was used. All parameter values were chosen manually during the preliminary experiment.

Finally, the obstructive fitness function $\theta$ was added to the auxiliary set and the corresponding problem was solved with PESA-II and ES + R-learning again.

The length of an individual in all cases was 64 bits. Notice that the optimal fitness of such individual is 448. 30 runs of each algorithm were performed. In each run 500000 fitness calculations were made. The statistics shown further were based on the best individuals from the last generation of each run.

### C. Experiment results

The experiment results of optimizing H-IFF with different auxiliary sets are presented in Table IV. The runs in which the ideal individual of fitness 448 was evolved are called *successful*. (1 + 5) ES, that used no auxiliary fitness functions, appeared to be unsuccessful in all runs.

Applying the (1 + 5) ES + R-learning variation of the proposed method with supporting auxiliary fitness functions led to the increase of the ES efficiency and allowed to evolve an ideal individual in each run. So the proposed method fulfilled the requirement corresponding to the *supporting only* auxiliary set. PESA-II also appeared to be effective in this case, that is in accordance with [1], where PESA algorithm was used for the same problem.

In the last two rows of the Table IV the auxiliary set with an obstructive function is considered. This set also includes the supporting auxiliary fitness functions from the previous case. The proposed (1 + 5) ES + R-learning method is still much more effective than the (1 + 5) ES evolving an ideal individual in 92% of runs. So it fulfills the requirement for the *supporting and obstructive* auxiliary set.

Notice that PESA-II is not effective any more in the last part of the experiment. No ideal individual was evolved with it. Such results can be explained with the fact that PESA-II tried to optimize all the auxiliary fitness functions as long as it is a multi-objective algorithm. In this case optimizing of obstructive fitness function led to decrease of the target one. By contrast, the proposed EA + RL method was able to ignore the obstructive fitness function, because it learned that its application is profitless. It can be inferred that the proposed method is able to choose between the auxiliary fitness functions in order to enhance the optimization of the target one. So the proposed method can be more useful than multi-objectivization techniques when we have incomplete knowledge of the auxiliary fitness functions, or helper-objectives [3].

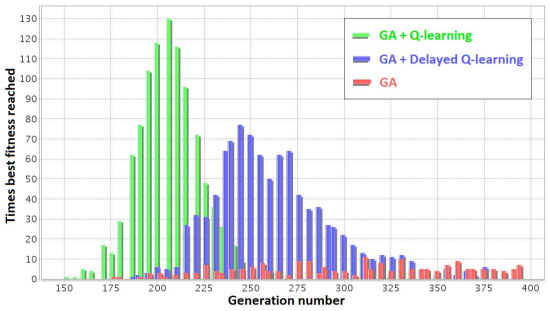


Fig. 1. Results for the auxiliary set with dynamically varying properties

## VI. RESULTS OVERVIEW

Let us sum up all the results (including the ones from the previous works) in accordance with the requirements formulated in the present paper. The method was applied to a number of model problems. In all the problems individuals were encoded as bit strings.

The very first model problem used to test the proposed method was the problem described in [5]. It has auxiliary functions which can be both obstructive and supporting depending on the optimization stage. There are two optimization stages. The results of solving this problem are illustrated in Fig. 1. EA + RL manages to choose the most efficient auxiliary fitness function at both optimization stages. The proposed method noticeably outperforms the genetic algorithm used to optimize the target fitness function. The crossover operator in this algorithm exchanges parts of individuals with shift. The mutation operator flips each bit of the individual with some probability.

The second considered model problem was H-IFF with supporting functions $f_0$, $f_1$ only. It was used to compare EA + RL method with multi-objectivization [6] approach. The proposed method performed equally well with the multi-objective PESA algorithm and outperformed all other single and multi-objective evolutionary algorithms, such as different kinds of evolution strategies, DCGA and PAES [1]. The maximal possible efficiency was achieved. It means that in all EA + RL runs the best individual was evolved.

Finally, the experiment with the Royal Roads model problem described in the present paper showed that EA + RL performs asymptotically equally well with EA in the case of

TABLE V
OVERVIEW OF THE EXPERIMENTS WITH EA + RL

| Target fitness function | Auxiliary fitness functions | Type of the auxiliary set | Best RL algorithm | Implemented EAs and MOEAs | Results |
|---|---|---|---|---|---|
| $\lfloor \frac{x}{d} \rfloor$ | $min(x,p); max(x,p)$ | *supporting and obstructive*, dynamically changing | Q-learning with $\varepsilon$-greedy exploration; Delayed Q-learning | GA with shift crossover and homogeneous mutation | EA + RL $>$ EA |
| H-IFF | $f_0$ = zero-bit blocks number; $f_1$ = one-bit blocks number | *supporting only* | R-learning with $\varepsilon$-greedy exploration | (1 + 1) ES; (1 + 5) ES; (1 + 10) ES; GA with one-point crossover; MOEAs (PAES and PESA) | EA + RL $>$ all EAs; EA + RL $>$ PAES; EA + RL $=$ PESA |
| Royal Roads | number of zeros | *obstructive only* | Delayed Q-learning | (1 + 1) ES | EA + RL $=$ EA |
| H-IFF | $f_0$; $f_1$; number of matches with $1010\ldots10$ mask | *supporting and obstructive* | R-learning with $\varepsilon$-greedy exploration | (1 + 5) ES; PESA-II MOEA | EA + RL $>$ EA; EA + RL $>$ PESA-II |

the auxiliary set consisting of the obstructive fitness function only. It is also confirmed that EA + RL outperforms the EA in the case of auxiliary set of both supporting and obstructive fitness functions on the example of the H-IFF optimization problem variation. Results of solving this problem showed that the proposed method outperforms multi-objective optimization algorithm PESA-II in the case of presence of the obstructive fitness function.

The summary of all the experiments taken with EA + RL is presented in Table V. The "$>$" sign means "outperforms", the "=" sign means "performs asymptotically equally with". The efficiency measure is equal to the number of generations taken to evolve the best individual, as described previously in this paper. The table demonstrates that the proposed EA + RL method outperforms the adjusted EA for all the auxiliary sets with supporting fitness functions and it never performs worse than the EA. It is also able to work properly with the auxiliary set with dynamically varying properties. So the experiment results confirm that the proposed method fulfills the formulated requirements for all the model problems tested so far.

## VII. CONCLUSION

A method of increasing the efficiency of single-objective evolutionary algorithms is described. It is based on choosing efficient auxiliary fitness functions with reinforcement learning. The auxiliary fitness functions are divided into supporting and obstructive ones. Requirements based on this classification are formulated. It is shown in the previous works that a part of them is fulfilled for some model problems. Dealing with obstructive functions is illustrated by the present experiment results. The experiments are based on the modifications of Royal Roads and H-IFF optimization model problems. Thus, compliance with all the requirements has been illustrated on a number of model problems. The proposed method is shown to be effective.

## VIII. ACKNOWLEDGMENTS

The research was supported by Ministry of Education and Science of Russian Federation in the context of Federal Program "Scientific and pedagogical personnel of innovative Russia".

## REFERENCES

[1] J. D. Knowles, R. A. Watson, and D. Corne, "Reducing local optima in single-objective problems by multi-objectivization," in *Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimization*, ser. EMO '01. London, UK: Springer-Verlag, 2001, pp. 269–283.

[2] K. Deb, *Multi-objective Optimization Using Evolutionary Algorithms*. John Wiley & Sons, 2001.

[3] M. T. Jensen, "Helper-objectives: Using multi-objective evolutionary algorithms for single-objective optimisation: Evolutionary computation combinatorial optimization," *Journal of Mathematical Modelling and Algorithms*, vol. 3, no. 4, pp. 323–347, 2004.

[4] M. Buzdalov, "Generation of Tests for Programming Challenge Tasks Using Evolution Algorithms," in *Proceedings of the 2011 GECCO Conference Companion on Genetic and Evolutionary Computation*, New York, US, ACM, 2011, pp. 763–766.

[5] A. Afanasyeva and M. Buzdalov, "Choosing best fitness function with reinforcement learning," in *Proceedings of the Tenth International Conference on Machine Learning and Applications, ICMLA 2011*, vol. 2. Honolulu, HI, USA: IEEE Computer Society, 2011, pp. 354–357.

[6] A. Afanasyeva and M. Buzdalov, "Optimization with auxiliary criteria using evolutionary algorithms and reinforcement learning," in *Proceedings of 18th International Conference on Soft Computing MENDEL 2012*, Brno, Czech Republic, 2012, pp. 58–63.

[7] A. Gosavi, "Reinforcement learning: A tutorial survey and recent advances," *INFORMS Journal on Computing*, vol. 21, no. 2, pp. 178–192, 2009.

[8] A. E. Eiben, Z. Michalewicz, M. Schoenauer, and J. E. Smith, "Parameter control in evolutionary algorithms," in *Parameter Setting in Evolutionary Algorithms*, 2007, pp. 19–46.

[9] A. E. Eiben, M. Horvath, W. Kowalczyk, and M. C. Schut, "Reinforcement learning for online control of evolutionary algorithms," in *Proceedings of the 4th international conference on Engineering self-organising systems ESOA'06*. Springer-Verlag, Berlin, Heidelberg, 2006, pp. 151–160.

[10] S. Müller, N. N. Schraudolph, and P. D. Koumoutsakos, "Step size adaptation in evolution strategies using reinforcement learning," in *Proceedings of the Congress on Evolutionary Computation*. IEEE, 2002, pp. 151–156.

[11] A. L. Strehl, L. Li, E. Wiewiora, J. Langford, and M. L. Littman, "PAC Model-free Reinforcement Learning," in *Proceedings of the 23rd International Conference on Machine Learning (ICML 2006)*, 2006, pp. 881–888.

[12] M. Mitchell, *An Introduction to Genetic Algorithms*. Cambridge, MA: MIT Press, 1996.

[13] A. Schwartz, "A reinforcement learning method for maximizing undiscounted rewards," in *Proceedings of the Tenth International Conference on Machine Learning*, 1993, pp. 298–305.

[14] D. W. Corne, N. R. Jerram, J. D. Knowles, M. J. Oates, and M. J., "Pesa-II: Region-based selection in evolutionary multiobjective optimization," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2001)*. Morgan Kaufmann Publishers, 2001, pp. 283–290.