THE OPTIMAL SOLUTION PREDICTION FOR GENETIC AND DISTRIBUTION BUILDING ALGORITHMS WITH BINARY REPRESENTATION*

E. Sopov, O. Semenkina
Siberian State Aerospace University named after Academician M.F. Reshetnev
31 "KrasnoyarskiyRabochiy" prospect, Krasnoyarsk, 660014, Russia.

E-mail: evgenysopov@gmail.com

Abstract. Genetic and distribution building algorithms with binary representation are analyzed. A property of convergence to the optimal solution is discussed. A novel convergence prediction method is proposed and investigated. The method is based on analysis of gene value probabilities distribution dynamics, thus it can predict gene values of the optimal solution to which the algorithm converges. The results of investigations for the optimal prediction algorithm performance are presented.

Introduction.

The genetic algorithms (GAs) efficiency depends on the fine tuning of its parameters [1]. If GA-user sets arbitrary parameters values, the GA efficiency may arbitrary vary from very low to very high. The recent trends in a field of GAs lead to adaptive GAs based on complex hybrid structure and efficient GAs with reduced parameters set.

A known approach to GA parameters set reduction is Distribution Building Algorithms, also called Estimation of Distribution Algorithms (EDA) or Probabilistic Model-Building Genetic Algorithms, which replace population representation with a probability distribution over the choices available at each position in the vector that represents a population member [2, 3]. The EDAs generate solutions and control populations according to given distribution instead of using nature-inspired genetics-based operations.

A straightforward way to estimate distribution for GAs with binary representation is to use the distribution of values equal to one over the population. Many EDA algorithms are based upon such kind of distribution [4-7].

The main objective of this work is increasing the binary GAs effectiveness by analyzing and exploiting the explicit information on algorithm dynamics via distribution estimation.

In section II we discuss how distribution can be estimated for any binary-coded GA to analyze its dynamics and fulfill prediction of algorithm convergence point, hopefully global optimum point. The general prediction method and some specific realization are demonstrated

* Results were obtained in the framework of the state task № 140 of the Ministry of Education and Science of the Russian Federation

in section III. We have tested the proposed convergence method with a representative set of complex optimization problems – results are shown in section IV.

Distribution estimation for search algorithms with binary representation.

The binary representation GAs are the conventional GAs with population of fix-length solutions, which contain 0 or 1-value in each position. As solution is binary vector and its fitness (objective function value) is a real number, we can talk about pseudo-boolean optimization problem statement.

One can estimate the probabilities vector for a population to present the statistics proceeding by search algorithm:

$$P^k = (p_1^k, ..., p_n^k),\ p_i^k = P(x_i^k = 1),\ i = \overline{1, n},$$

here p_i^k is a probability of 1-value for the *i-th* position in solution, *n* is a solution length, x_j^i is a value of the *j*-th gene of *i*-th individual, *k* is a number of generation (iteration).

We can write down the general scheme of any EDA algorithm using given probability distribution in the following way:

1. Randomly generate an initial population according to the probability distribution.
2. Evaluate current population.
3. Update the distribution using the given strategy.
4. Form a new population according to the probability distribution.
5. Until stop criterion is satisfied, repeat steps 2-4.

As we can see, GAs use the same scheme. The main differences have a place on steps 3-4, where GAs update information using such operations as selection, crossover and mutation. Step 3 also defines the different EDA algorithms.

Let's consider some classical EDA realization such as the variable probabilities method ("MIVER") [4, 5], population-based incremental learning (PBIL) [6] and probabilities-based GA [7].

Variable probabilities method ("MIVER") was first proposed in 1986 [4] and improved in 1987 [5]. This is a stochastic optimization procedure for pseudo-boolean optimization problems, which works with a population of binary solutions. The distribution update strategy is based on 1-value distribution of the best and the worst solution in certain population. There exist a number of mathematical proofs of the convergence for some classes of optimization problems.

Population-based incremental learning (PBIL) was proposed in 1994 [6]. This is a combination of the evolutionary optimization and artificial neural networks learning. PBIL updates the distribution in the same way as "MIVER" does – according to the 1-value distribution of the best and the worst solution, but using similar to neural net weights update procedure.

Both PBIL and "MIVER" use greedy linear update strategy (taking into account only the best/worst solution), so they demonstrate much more local convergence than conventional GA. Thus conventional GAs performance on average is better for many complex optimization problems.

Probabilities-based GA was proposed in 2005 [7]. This is a further evolution of ideas of variable probabilities method and PBIL. In this algorithm, genetic operators are not substituted with the distribution estimation, but the estimated distribution is used to model genetic operators. Probabilities-based GA uses the whole population to update distribution, so

it demonstrates the global performance as the conventional GA. Moreover, it contains less parameters to tune and proceeds additional information about search space (probabilities distribution), thus probabilities-based GA can exceed conventional GAs on average.

We have investigated the performance of algorithms using some visual representation of the distribution. Each component value of the probability vector, which performs distribution, varies during the run, so we can show it on diagram (figure1).

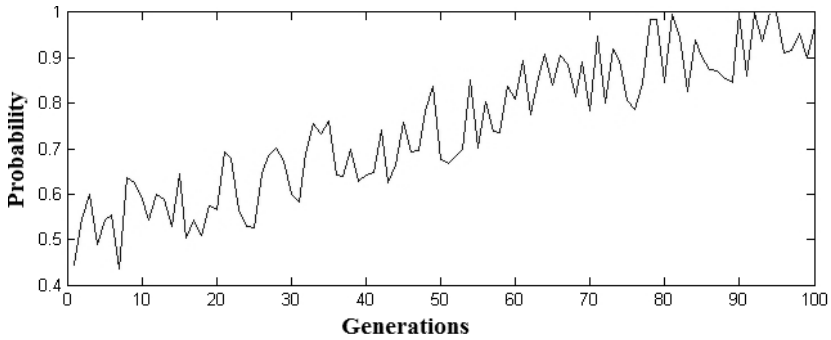

Figure 1. The value change for *j*-th component of probability vector

As we can see the probability value oscillates about $p = 0.5$ (initial steps, uniform distribution over the search space) and then converges to value one (or zero in other case) as the algorithm converges to the optimal (or suboptimal) solution.

We can estimate and visualize distribution of 1-value for any binary GA, even if GA doesn't use the distribution. The distribution estimation can be performed in a following way:

$$\overline{P} = (p_1, \ldots, p_n), \; p_j = P\{x_j = 1\} = \frac{1}{N}\sum_{i=1}^{N} x_j^i, \; j = \overline{1,n},$$

where N is a population size.

As different algorithms use different distribution update strategies, so we obtain different behavior of probabilities over the generations (figure2-5):

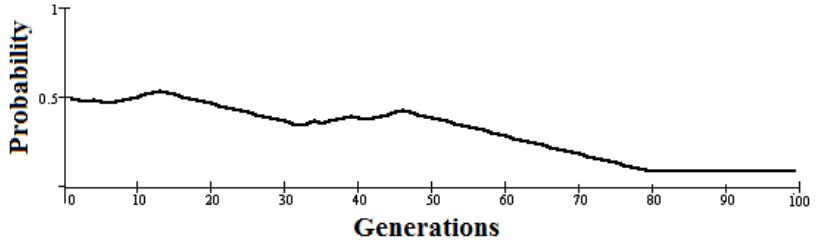

Figure 2. Evolution of the probability vector component for a run of variable probabilities method ("MIVER")

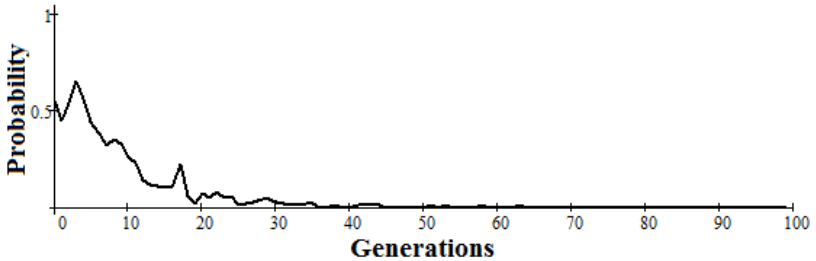

Figure 3. Evolution of the probability vector component for a run of PBIL

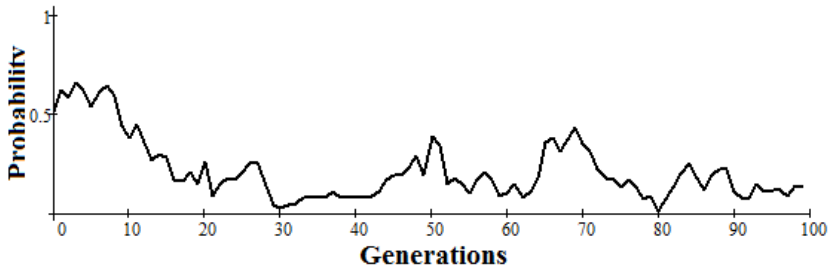

Figure 4. Evolution of the probability vector component for a run of probabilities-based GA

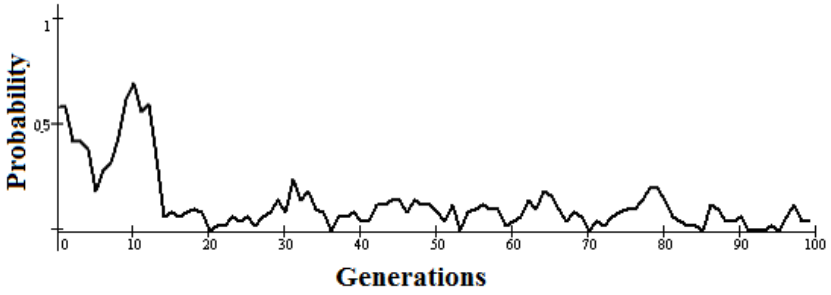

Figure 5. Evolution of the probability vector component for a run of conventional GA

Analyzing these diagrams for different optimization problems we can find that the components of the probabilities vector frequently converge to value one if optimal solution contains 1-value in corresponding position (or to zero if optimal solution contains 0-value). It means that if the probability converges to one (or zero) then the value of the corresponding gene of the optimal solution (or a solution to which the algorithm intends to converge) most probably is equal to one (or zero). The higher algorithm performance the more often probabilities converge to the correct value.

The optimal solution prediction method.

We can express the previously mentioned feature of stochastic binary algorithms as:

$$\begin{cases} p_i \to 1, \; if \; x_i^{opt} = 1, \\ p_i \to 0, if \; x_i^{opt} = 0, \end{cases}$$

when the algorithm's iteration number approaches infinity. Here x_i^{opt} is the *i*-th position value of problem optimal solution, $i = \overline{1,n}$.

So we can use this feature to predict an "optimal" solution of the given problem. The convergence prediction method is as follows:

1. Choose the binary stochastic algorithm (e.g. GA, PBIL or other), set the iteration number $i = 1,...,I$ and the number of independent algorithm runs $s = 1,...,S$.
2. Collect the statistics $(p_j)_i^s$, $j = 1,...,n$. Average $(p_j)_i$ over *s*. Determine the tendency for p_j change.
3. Set $x_j^{opt} = \begin{cases} 1, \; if \; p_j \to 1, \\ 0, \; if \; p_j \to 0 \end{cases}$.
4. Add "optimal" solution into the current population.

A simple way to determine the convergence tendency is the use of the following integral criterion on the step 3:

$$x_j^{opt} = \begin{cases} 1, & if \sum_{i=1}^{I} \left((p_j)_i - 0.5 \right) > 0, \\ 0, & otherwise. \end{cases}$$

The main idea is that the more often probability value is greater than 0.5, the higher probability of being "optimal" solution coordinate equal to one.

In many practical problems, a situation exists when the binary stochastic algorithm collects not enough information on the early steps and the gene value of the certain position is equal to one (or zero) for almost every current solution. At the later stage algorithm can find the much better solution with inverted value of genes and it means that the probability vector values will change their convergence direction. But the previously mentioned prediction method will give us the primary value, because the value of the probabilities vector was greater than 0.5 (or less than 0.5 for zero values) for too long time.

We propose to use the following modification of the convergence prediction method to avoid the above mentioned shortcoming:

1. Set the prediction step *K*.

2. Every *K* iteration use the given statistics $\overline{P}_i, i = \overline{1, N_K}$, $N_K = t \cdot K$, $t \in \{1,2,\ldots\}$, to evaluate the probability vector change: $\Delta\overline{P}_i = \overline{P}_i - \overline{P}_{i-1}$.

3. Set the weights for every iteration accordingly to its number:
$\sigma_i = 2i / N_K (N_K + 1),\ i = 1,\ldots, N_K$.

4. Evaluate probability vector weighted change as: $\Delta\tilde{P} = (\Delta\tilde{p}_j) = \sum_{i=1}^{N_K} \sigma_i \cdot \Delta\overline{P}_i$.

5. Set the "optimal" solution:

$$x_j^{opt} = \begin{cases} 1, & if\ \Delta\tilde{p}_j \geq 0, \\ 0, & otherwise. \end{cases}$$

6. Add the "optimal" solution into the current population.

The main idea of the given modification is that the probability values on the later iterations have the greater weights as algorithm collects more information about search space. The weights should have values such that $\sigma_{i+1} > \sigma_i$ and $\sum_{i=1}^{N_K} \sigma_i = 1$.

The strategies of the predicted "optimal" solution usage may vary: use it as the final solution, add it to the population and continue search, and so on. In this work we just added it in the current population without additional heuristics which can be applied here.

Test Problems and Numerical Experiments.

We have tested the proposed convergence prediction methods for discussed EDA and conventional GA with the representative set of complex optimization problems.

We have included 12 known test problems of continuous optimization, 6 of them are from the test function set approved at Special Session on Real-Parameter Optimization (CEC 2005) [8]. All these functions are known as complex optimization problems and represent the challenge for GAs and other evolutionary algorithms. We haven't included any well-known binary problem as they are less complex for fine-tuned GAs than real-value problems after the binarization.

The test problem set contains the following functions: Rastrigin and Shifted Rotated Rastrigin, Rosenbrock, Griewank, De Jong 2, "Sombrero" ("Mexican Hat"), Schekel, Katkovnik, Additive Potential function, Multiplicative Potential function and 2 one-dimensional multi-extremal functions.

Some functions are listed below:

Shifted Rotated Rastrigin Function.

$$F(x,y) = (0.1 \cdot K_x \cdot A)^2 + (0.1 \cdot K_y \cdot B)^2 -$$
$$- 4 \cdot \cos(0.8 \cdot K_x \cdot A) - 4 \cdot \cos(0.8 \cdot K_y \cdot B) + 8$$

$A = x \cdot \cos(\alpha) - y \cdot \sin(\alpha)$,

$B = x \cdot \sin(\alpha) + y \cdot \cos(\alpha)$,

K_x, K_y - scaling rate over x, y axis,

α - rotation angle.

$\alpha = \pi/2, kx = 1.5, ky = 0.8$

$x, y \in [-16;16]$, $\min = F(0,0) = 0$

Griewank function.

$$F(x,y) = \frac{-10}{0.005 \cdot (x^2 + y^2) - \cos(x) \cdot \cos(y/\sqrt{2}) + 2} + 10$$

$x_1, x_2 \in [-16;16]$, $\min = F(0,0) = 0$

"Sombrero" function.

$$F(x,y) = \frac{1 - \sin^2(\sqrt{x^2 + y^2})}{1 + 0.001 \cdot (x^2 + y^2)}$$

$x_1, x_2 \in [-10;10]$, $\min = F(0,0) = 0$

Additive potential function.

$F(x_1, x_2) = -z(x_1) z(x_2)$,

$$z(x) = -\frac{1}{(x-1)^2 + 0.2} - \frac{1}{2(x-2)^2 + 0.15} - \frac{1}{3(x-3)^2 + 0.3}$$

$x_1, x_2 \in [0;4]$, $\min = F(2,2) = -60.8$

One-dimensional function N1.

$$F(x) = 0.05(x-1)^2 + (3 - 2.9e^{-2.77257x^2}) \times$$
$$\times (1 - \cos(x(4 - 50e^{-2.77257x^2})))$$

$x \in [-1;1]$, $\min = F(0.9544) = 0.00017$.

To estimate algorithms efficiency we have carried out series of independent runs of each algorithm for each test problem. We used the reliability as a criterion for evaluation of the algorithm efficiency. Reliability is the mean number of successful algorithm runs (i.e. the exact solution of the problem was found) over the whole number of independent runs. So we have computed the expected value of algorithms reliability.

We have set the following parameters values for algorithms:

- Accuracy (the whole binary solution length) – 40,
- Population size – 50,
- Generation number – 50,

- Independent runs number – 100.

The settings and parameters for each algorithm (e.g. selection type in GA, learning rate in PBIL and others) were chosen in advance to be efficient on average over the test function set.

The results are validated through ANOVA method, namely Mann–Whitney–Wilcoxon test. All result differences are statistically significant.

First we have estimated the efficiency of convergence prediction algorithms using original integral criterion. Results are shows in a Table 1. The last column contains the values for the conventional GA without prediction. The grey cells show the best on problem value.

Table 1. The reliability estimation for convergence prediction algorithm using integral criterion

Problem	**MIVER**	**PBIL**	**Prob.-based GA**	**GA**	**Conventional GA (no prediction)**
Rastrigin	23	72	**88**	69	85
Shifted Rotated Rastrigin	4	**44**	36	42	24
Rosenbrock	14	20	**50**	24	26
Griewank	0	42	**53**	40	14
De Jong 2	0	**23**	12	21	22
Sombrero	20	**45**	33	39	18
Schekel	8	40	42	**50**	48
Katkovnik	14	80	52	**81**	76
Additive potential	66	62	71	**72**	**72**
Multiplicative potential	8	**92**	46	**92**	**92**
One-dimensional N1	**68**	47	60	41	42
One-dimensional N2	28	43	**60**	48	45

As we can see, results vary: algorithms are good in some cases and less efficient in others. The variable probabilities method demonstrates the worst performance as it is most greedy algorithm and has local behavior. PBIL, probabilities-based GA and conventional GA performances are almost similar. So we can say that the average reliability of integral criterion in convergence prediction is low although this method is useful as GA with the prediction method shows higher performance comparing to conventional GA.

Table 2 presents efficiency estimation results for the modified convergence prediction algorithm. The results show that the modified convergence prediction can essentially improve performance of search algorithms. The best results are achieved using the probabilities-based GA as it collects and proceeds more statistical information about the search space. As we can see in Table 2, probabilities-based GA "wins" in 7 of 12 problems (GA – 5, PBIL – 4, MIVER – 0). In cases when the probabilities-based GA exceeds, its reliability essentially better, in other cases its reliability is also high.

Table 2. The reliability estimation for modified convergence prediction algorithm

Problem	MIVER	PBIL	Prob.-based GA	GA	Conventional GA (no prediction)
Rastrigin	61	88	**98**	94	85
Shifted Rotated Rastrigin	28	40	**72**	50	24
Rosenbrock	24	41	**80**	38	26
Griewank	10	51	**100**	58	14
De Jong 2	8	28	**70**	22	22
Sombrero	22	55	**60**	56	18
Schekel	28	62	30	**70**	48
Katkovnik	36	**100**	**100**	**100**	76
Additive potential	76	**100**	98	**100**	72
Multiplicative potential	28	**100**	86	98	92
One-dimensional N1	70	98	75	**100**	42
One-dimensional N2	86	**100**	80	**100**	45

As we can see, the modified convergence prediction algorithm demonstrates the better performance. It also can essentially improve the performance of simple algorithms like PBIL or MIVER. The GA-based techniques can increase the reliability up to 100% even with complex problem.

We have also established experimentally that deceptive problems [9, 10] are not difficult for GA with prediction, but GA requires a slight modification to intense mutation [11].

We have investigated the following mostly used trap-functions: Ackley Trap, Whitley Trap, Goldberg Trap, Trap-4, Trap-5, DEC2TRAP, Liepins and Vose, which are known as GA-hard problems.

In this paper, we demonstrate results only for the probability-based GA as it shows the highest efficiency on average. We investigated two versions of algorithms: standard and one with the optimal solution prediction (the predicted solution is evaluated every 5 generations and added to population).

The binary solution length is 100. The results are evaluated over 100 independent run. The parameters for algorithm (selection and mutation) are chosen in advance to be efficient on average over the test function set.

In addition to the standard mutation operation, we implement the inversion operation, which inverts all genes in chromosome with very low probability. As numerical experiments shows it can help to overcome the trap attraction.

The results are shown in a table 3.

Table 3. The reliability estimation on trap-functions

Problem	Prob.-based GA with inversion	Prob.-based GA with inversion and prediction
Ackley Trap	42	**48**
Whitley Trap	**41**	40
Goldberg Trap	53	**57**
Trap-4	61	**62**
Trap-5	38	**49**
DEC2TRAP	78	**98**
Liepins and Vose	**51**	50

As known traps lead search algorithms in a direction away from optima, the efficiency of the standard search techniques (e.g. GAs) is about zero. As we can see in a table 3, the implementation of the inversion operator can help to avoid the traps and the usage of proposed prediction method improves the results.

Conclusion.

As we demonstrate in the paper, stochastic search procedures with binary representation (GA, PBIL and others) can be analyzed through visualization of its estimated distribution dynamics. We have proposed the efficient method to predict the point of convergence ("optimal" solution) using the results of the distribution dynamic analysis.

Two approaches were investigated: a straightforward way of using the integral criterion and more complex strategy with the contribution weights calculation. As numerical experiments show the modified convergence prediction provides better performance, even for algorithms with greedy search strategy and local behavior.

In further work we suppose to develop the prediction method and its implementation with other EDA and evolutionary algorithms.

References

[1] Goldberg D.E. 1989 Genetic algorithms in search, optimization and machine learning *Addison-Wesley*.
[2] Muehlenbein H., Paas G. 1996 From recombination of genes to the estimation of distribution. part 1, binary parameter *Lecture notes in computer science 1141, parallel problem solving from nature* pp. 178-187.
[3] Larraenaga P., Lozano J. A. 2001 Estimation of distribution algorithms: a new tool for evolutionary computation *Kluwer academic publishers*.
[4] Antamoshkin A. 1986 Brainware for search pseudoboolean optimization *Pr. of 10th Prague Conf. on Inf. Theory, Stat. Dec. Functions, Random Processes*.
[5] Antamoshkin A.N, Saraev V.N. 1987 The method of variable probabilities *Problemy sluchaynogo poiska (Problems of random search)* (Riga: Zinatne) pp. 26-34.
[6] Baluja S. 1994 Population-based incremental learning: a method for integrating genetic search based function optimization and competitive learning *Technical report Carnegie Mellon University* Report number CMU-CS-94-163.
[7] Semenkin E.S., Sopov E.A. 2005 Probabilities-based evolutionary algorithms of complex systems optimization *Intelligent Systems (AIS'05) and Intelligent CAD (CAD-2005)* Vol.1. pp. 77-79.

[8] Suganthan P. N., Hansen N. et al. 2005 Problem Definitions and Evaluation Criteria for the CEC 2005 Special Session on Real-Parameter Optimization *Technical Report, Nanyang Technological University, Singapore and KanGAL* Report Number 2005005, Kanpur Genetic Algorithms Laboratory, IIT Kanpur.
[9] Deb, K. and D. E. Goldberg 1992 Analysing deception in trap functions *Foundations of Genetic Algorithms 2*, D. Whitley (Ed.), San Mateo: Morgan Kaufmann.
[10] Whitley, L. D. 1991 Fundamental principles of deception in genetic search *Foundations of Genetic Algorithms*, G. J. E. Rawlins (Ed.), San Mateo, CA: Morgan Kaufmann.
[11] Sopov E.A., Semenkin E.S. 2011 Effectiveness investigation of modified probabilities based genetic algorithm on deceptive “trap” functions *Sistemy upravleniya i informatsionnye tekhnologii* (Control systems and information technologies) **3(45)** pp. 90-95.