An Adaptive Mutation Scheme in Genetic Algorithms for Fastening the Convergence to the Optimum

Sima (Etaner) Uyar, Gulsen (Cebiroglu) Eryigit, Sanem Sariel

Istanbul Technical University,

Computer Engineering Department

Maslak TR-34469 Istanbul, Turkey.

etaner@cs.itu.edu.tr gulsen@cs.itu.edu.tr sariel@cs.itu.edu.tr

ABSTRACT

Mutation rate parameter is considered to be one of the most sensitive of the parameters that a genetic algorithm works with. It has been shown that through using a mutation rate variation scheme that adapts the mutation rate parameter during the run of the algorithm, the time to find the optimum is decreased. In this study, a mutation rate adaptation scheme, that adapts the mutation rate separately for each gene location on the chromosome based on the feedback taken from the success and failure rates of the individuals in the current population, is proposed. Through tests using the one-max problem, it is shown that the proposed mutation adaptation scheme allows faster convergence than the other similar approaches chosen for comparisons. The results are very promising and promote further research.

Keywords : adaptive mutation rate, parameter control, convergence rate, genetic algorithms

1. Introduction

Genetic algorithms [3] (GA) belong to a class of biologically inspired optimization approaches that model the basic principles of classical Mendelian genetics and Darwinian theory of evolution. Due to their robust nature, genetic algorithms are used in a wide variety of applications. However one of the major drawbacks of working with genetic algorithms is that performance largely depends on the appropriate setting of some parameters: namely population size, crossover and mutation rates. These parameters interact with each other, making it even harder to find optimal settings. However mutation rate is considered to be one of the most sensitive of these parameters. It has been traditionally regarded as a background operator that mainly works as an insurance policy [10], protecting alleles from being lost from the population but it has been shown that the mutation rate value largely affects the general behavior of the algorithm. There has been extensive work to investigate the exact nature of mutation [1], [4] [5], [8].

The techniques developed to set these parameters are classified by Eiben et al. [2] as: parameter tuning and parameter control. For parameter tuning, the parameter values are set in advance, before the run and are kept constant during the whole execution of the algorithm. In parameter control techniques, parameters are initialized at the start of execution and are allowed to change during the run. Parameter control techniques are classified mainly into three groups based on the type of change they introduce:

- deterministic: the parameter value is updated according to some deterministic rule,
- adaptive: the parameter value is updated based on some feedback taken from the population
- self-adaptive: the parameter is evaluated and updated by the evolutionary algorithm itself.

In this study, an adaptive mutation rate strategy that uses feedback obtained from the current population and increases or decreases the mutation rate accordingly for each locus on the chromosome is introduced. Even though using feedback from the current state of the search seems to be a useful approach, it has not been studied much within the scope of canonical GAs [11]. This approach is tested against previously published methods for mutation rate control, on a simple one-max problem. The results are seen to be promising and promote further study.

The rest of this paper is organized as follows: Section 2 introduces the proposed mutation rate adaptation approach. In section 3 the different tested methods are presented. In section 4, the experimental setup and the results are given and discussed; Section 5 provides a conclusion, based on the results while providing possible directions for future work and concludes the paper.

2. Gene Based Adaptive Mutation GA

In this paper, a Gene Based Adaptive Mutation (GBAM) method is proposed. This approach experiments with varying mutation rate values during the run, using feedback from the population.

Different from other known mutation adaptation strategies, in GBAM each locus has its own mutation rate value. An adaptive approach for adjusting mutation rates for the gene locations based on the feedback obtained by observing the relative success or failure of the individuals in the population is used.

In GBAM, there are two different mutation rates defined for each locus: a mutation rate value pm1 for those genes that have an allele value of "1" at that locus and another mutation rate value pm0 for those that have a "0". In the reproduction phase, the appropriate mutation rate is applied based on the gene allele value. Initially all mutation rates are set to a default value in the specified boundaries. During the GA run, the mutation probabilities pm1 and pm0 for all loci are updated at each generation using the feedback taken from the relative success or failures of those individuals having a "1" or "0" at that locus respectively. For a maximization problem, the update rule for the mutation rate subunits for one gene location can be seen in Eq.1. This update rule is applied separately for each locus.

$$p_{m0}^{+} = \begin{cases} p_{m0} + \gamma, & \left(S_{avg} / P_{avg}\right) > 1 \\ p_{m0} - \gamma, & \left(S_{avg} / P_{avg}\right) \le 1 \end{cases}$$
$$p_{m1}^{+} = \begin{cases} p_{m1} - \gamma, & \left(S_{avg} / P_{avg}\right) > 1 \\ p_{m1} + \gamma, & \left(S_{avg} / P_{avg}\right) \le 1 \end{cases} \quad (1)$$

The pm_i (i=0,1) value for a locus corresponds to the rate of mutation subunit that will be applied when the gene value is i in the corresponding gene. S_{avg} is the average fitness of the individuals with an allele "1" for the corresponding gene location. P_{avg} is the average fitness of the population. γ is the update value for the mutation rates.

As a result of the updates at each generation, pm_i values are allowed to oscillate within the limits defined by lower and upper bounds. If an update causes a mutation rate to exceed the upper limit, the corresponding mutation rate is set to the upper bound value and if it causes a mutation rate to go below the lower limit, the corresponding mutation rate is set to the lower bound value. Another parameter that GBAM uses is the initial mutation rate value. All parameters are determined empirically.

As will be shown in the analysis of the experiments, GBAM provides rapid convergence. For unimodal objective functions, this rapid convergence provides a valuable refinement. However, the fast convergence feature may cause the program to get stuck at local optima, especially for the multimodal objective functions. This problem is explored in detail in [6] for self-adaptive mutations, however the results can easily be extended to adaptive mutation schemes too. For the purposes of this paper, a unimodal function is used for the tests. The problems that may arise as a result of fast convergence are not within the scope of this paper and will be explored in detail in a future study.

3. Approaches Chosen for Comparisons

The aim of the experiments is to show that GBAM provides faster rates of convergence, as well as exploring its convergence behavior compared to other similar parameter control approaches found in literature.

As given in Section 1, parameter control approaches are categorized based on the type of change that is applied to the parameter. There are different formulations and implementations of each type of parameter control found in literature. A representative scheme, which is shown to give good performance, is chosen from each category and used for the comparisons.

3.1. Deterministic Approach

The deterministic mutation rate schedule provides the mutation rate to decrease from a value (generally, 0.5) to the optimum mutation rate (generally 1/L) without using any feedback from the population. The deterministic mutation rate schedule implementation proposed in [7] was reported in [11] as having the most succeful results for hard combinatorial problems. Based on this method, the time-varying mutation rate is calculated using the formula given in Eq. 2. In this formula, t is the current generation number and T is the maximum number of generations. In the original proposal for Eq. 2, the k value is chosen as 1.

$$p_t = \left(2 + \frac{L-2}{T-1} * t\right)^{-k} \quad (2)$$

3.2. Self-Adaptive Approach

In the self-adaptive approach, the parameters are encoded into the chromosomes and undergo mutation and recombination. The basic idea is that better parameter values lead to better individuals and these parameter values will survive in the population since they belong to the surviving individuals. Bäck et al. [3] refer to this approach also as on-line learning. In their work, they propose a self-adaptation mechanism of a single mutation rate per individual. The mutation of this mutation rate value gives the new mutation rate through Eq. 3. In this equation, γ is the learning rate and controls the adaptation speed. It is taken as 0.22 in [3] and also in this study.

$$p' = (1 + \frac{1-p}{p}.\exp(-\gamma.N(0,1)))^{-1} \quad (3)$$

3.3. Individually Adaptive Approach

In this study, an individually adaptive GA method (AGA) [29] is chosen for the comparisons. In this method, the probabilities of crossover and mutation are adapted depending on the fitness values of the individuals. The adaptation of the p_c and p_m allows the individuals having fitness values of over-average to maintain their genetic material, while forcing the individuals with sub-average fitness values to disrupt. The mutation rate adaptation rule is given in Eq. 4. In this equation, f denotes the fitness value of the individual, $f_{\max}$ denotes the best fitness value of the current generation, and f_{avg} denotes the average fitness value of the current generation. In [9], the constants k_2 and the k_4 are chosen as 0.5.

$$\begin{aligned} p_m &= k_2(f_{\max} - f)/(f_{\max} - f_{avg}), \quad f \geq f_{avg} \\ p_m &= k_4 \quad f < f_{avg} \end{aligned} \quad (4)$$

4. Experiments

The GBAM approach is expected to reduce the number of generations (or fitness evaluations) to locate an optimal individual. To investigate this effect, the one-max problem, which is unimodal and easy for a simple GA, is used for the tests. The main aim of this problem is to maximize the number of 1s in a binary represented string of length L. The optimum for this function is L. More formally the fitness function can be defined as in Eq. 5 where x_i represents the ith character in the string.

$$f = \sum_{i=1}^{L} x_i \quad (5)$$

Tests are performed for four different string lengths: L=200, L=400, L=800 and L=1600 to explore the effects of the length of the string on the number of generations required to first locate the optimum. For all tests, the program implementation for each chosen approach is run 50 times. All parameter settings are determined empirically to provide the best performance for each approach. Some settings are the same for all approaches:

- number of generations: 1500
- population size: 250 individuals
- parent selection: tournament selection with tournament sizes of two
- recombination: two-point cross over with p_c=1.0
- population dynamics: strictly generational

Some extra parameters are used by the methods chosen for comparisons. The settings for these values are given in Table-1 where DET is used for the deterministic approach, SA for the self adaptive approach, AGA for the adaptive GA approach and GBAM for the gene based adaptive mutation approach proposed in this study.

Table-1 Extra parameter settings

DET	k=1.2 (Eq. 2)
SA	initial mutation rate = 1/L lower mutation rate limit = 0.0001
AGA	k_2=1/L (Eq. 4)
GBAM	initial mutation rate = 0.02 mutation rate lower limit = 0.0001 mutation rate upper limit = 0.2 mutation update amount = 0.001

The statistical calculations for the number of generations required to locate the optimum individuals are given in Table-2, where μ is the mean number of generations needed to locate the best individual, σ is the standard deviation of this value, CI is the 99% confidence interval calculated for the location of the mean. Since one-max is a unimodal function, all approaches except for AGA are able to find the optimum for all string lengths. The plots of the number of generations needed to find the optimum for all methods averaged over 50 runs are given in Fig. 1.

As can be seen from Fig. 1 and Table-2, GBAM reduces the number of steps required to find the optimum solution. Based on the results in Table-2, GBAM seems to generate

promising results for all of the L values. SA and SGA seem to generate very close results, which is to be expected since the advantage of using SA comes from not having to find optimal mutation rates before the run. However in this study, SGA is implemented using optimal rates for each test problem, causing SA and SGA to perform similarly. The drawback of deterministic approach is the high generation number needed to locate the best fitness value. However, when L value is increased, the results become acceptable. Because the initial mutation rate value is 0.4 approximately, which is unnecessarily high for small L values. Although the AGA performs well for the small values of L compared to SA, SGA, and DET, when the L value increases, its performance decreases. The reason of this is that when the L value increases the k_2 value in Eq. 4 becomes very small.

Table-2 Statistical calculations for number of generations required to find the best individual

	L=200			**L=400**		
	μ	**σ**	**CI**	**μ**	**σ**	**CI**
GBAM	45.56	5.41	43.51 47.61	82.66	23.27	73.84 91.48
SGA	79.16	5.12	77.22 81.10	147.02	9.35	143.48 150.56
SA	80.84	6.22	78.48 83.20	149.92	10.97	145.76 154.08
DET	486.36	24.79	476.96 495.76	450.72	20.77	442.85 458.59
AGA	73.26	6.12	70.94 75.58	167.48	20.04	159.88 175.08
	L=800			**L=1600**		
	μ	**σ**	**CI**	**μ**	**σ**	**CI**
GBAM	190.52	38.42	175.96 205.08	354.96	39.25	340.08 369.84
SGA	289.84	17.67	283.14 296.54	612.16	31.41	600.26 624.06
SA	288.66	20.54	280.87 296.45	610.54	37.90	596.17 624.90
DET	427.04	14.13	421.68 432.40	568.50	32.82	556.06 580.94
AGA	506.40	95.91	470.05 542.75	*	*	*

(*) AGA is not able to find the optimum in 1500 generations

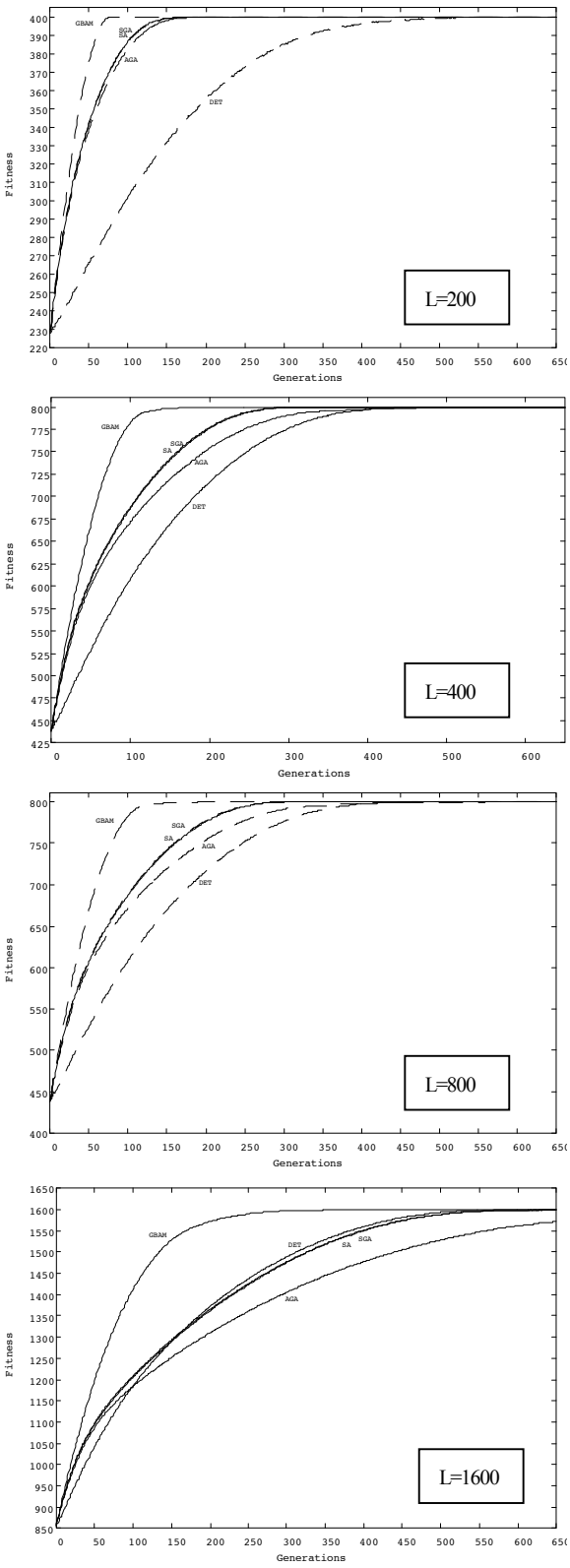

Fig.1. Best fitness values observed through generations for all methods averaged over 50 runs

5. Conclusion and Future Work

In this study, a mutation rate adaptation approach (GBAM) for each gene location in a binary representation, based on the relative performance of individuals is proposed. Because each gene location has a different parameter for controlling the rate of mutation at that location, the proposed approach is more suited for problems where the epistasis between genes is low to none. Since the mutation rate is adapted based on the fitness values of the best performing individuals, it is expected that GBAM fastens the convergence of the GA to the optimum. This has been confirmed through tests performed on a simple one-max problem. Since this problem is a unimodal function fast convergence to the optimum is not a problem. However for multimodal fitness landscapes, to escape from local optima, extra convergence tests should be applied, and necessary precautions to restore diversity should be taken.

Even though as a result of these preliminary tests, the overall performance of GBAM seems to be very promising for the chosen type of problems, there is still more work to be done to be able to make healthy generalizations. First of all, the parameter settings for GBAM, such as the lower and upper bound values, initial mutation rate and the mutation update values have been determined experimentally. More experiments need to be performed to see the effects of these parameters on performance more thoroughly. Secondly, the test problem set can be extended to include different types of problem domains. This also would allow the effects of different degrees of epistasis to be explored. Thirdly, a convergence control mechanism should be added to GBAM.

References

[1] Bäck T., "Optimal Mutation Rates in Genetic Search", *Proceedings of 5th International Conference on Genetic Algorithms*, Morgan Kaufmann (1993).

[2] Eiben A. E., Smith J. E., *Introduction to Evolutionary Computing*, Springer-Verlag, Berlin Heidelberg New York (2003).

[3] Goldberg D. E., *Genetic Algorithms in Search Optimization and Machine Learning*, Addison Wesley (1989).

[4] Hinterding R., Gielewski H., Peachey T. C., "The Nature of Mutation in Genetic Algorithms", *Proceedings of the 6th International Conference on Genetic Algorithms*, pp. 65-72, Morgan Kaufmann (1995).

[5] Ochoa G., "Setting the Mutation Rate: Scope and Limitations of the 1/L Heuristic", *Proceedings of Genetic and Evolutionary Computation Conference*, Morgan Kaufmann (2002).

[6] Rudolph G., "Self-Adaptive Mutations May Lead to Premature Convergence", IEEE Transactions on Evolutionary Computation, Vol. 5., No. 4, pp. 410-414, IEEE (2001).

[7] Smith J. E., Fogarty T. C., "Operator and Parameter Adaptation in Genetic Algorithms", Soft Computing 1, pp. 81-87, Springer-Verlag (1997).

[8] Spears W. M., "Crossover or Mutation", *Proceedings of Foundations of Genetic Algorithms 2*, Morgan Kaufmann (1993).

[9] Srinivas, M., Patnaik, L. M., "Adaptive Probabilities of Crossover and Mutation in Genetic Algorithms", IEEE Transactions on Systems, Man and Cybernetics, Vol. 24, No. 4., pp. 656-667, IEEE (1994).

[10] Stanhope S.A., Daida J.M., "An Individually Variable Mutation-Rate Strategy", *Proceedings of the Sixth Annual Conference on Evolutionary Programming* (1997).

[11] Thierens D., "Adaptive Mutation Control Schemes in Genetic Algorithms", *Proceedings of Congress on Evolutionary Computing*, IEEE (2002).