Sindh Univ. Res. Jour. (Sci. Ser.) Vol.49(2) 255-260 (2017)

SINDH UNIVERSITY RESEARCH JOURNAL (SCIENCE SERIES)

An Adaptive Mutation Strategy for Real Coded Genetic Algorithm

I. A. KOREJO, K. BROHI, A. A. CHANDIO, F. A. MEMON, A. R. NANGRAJ

Institue of Mathematics and Computer Science, University of Sindh, Jamshoro

Received 22nd June 2016 and Revised 09th March 2017

Abstract: Generally, single mutation operator is used by the conventional genetic algorithms (GAs), so it determine that all individuals in the population apply the same mechanism. Due to this mechanism, simple gas suffer from the premature convergence problem, it is easy to be trapped in to local optima.This paper proposes adaptive mutation (AMGAs) scheme for GAs to automatically select the most suitable operator while solving the problem.It also proposes different mutation strategies to deal with different situations in the search space by using amgas. To analysis the performance of propose approach, twnenty well known and applied benchmark problems are used. The experimental results show that amgas greatly balance the performance of GAs on different set problems.

Keywords: Genetic Algorithms; operator adaptation; global optimization problems.

1. INTRODUCTION

Evolutionary Algorithms (EAs), inspired by the natural evolutionary process of a population of individual over the time, are powerful search techniques in guiding evolutionary principle of "Survival of the fittest". EAs have become significant active research area over the past some decades. In recent years, differenet global optimization benchmark problems have been proposed in (Suganthan, Hansen,Liang, Deb, Auger, and Tiwari, 2005), these problems become more complex, from simple unimodal functions to rotated shifted multi-modal functions. It is challenging task to find the global optima of these functions, due this reason more efficient, effective and robust optimization algorithms can be useful for these functions (Glodberg, 1998).

Now a days, GAs have been successfully applied for various academic and real world problems due to the properties of easy-to-use and robustness with finding promising results (Glodberg, 1998), which is proposed by John Holland in the USA (Holland, 1975). GAs contains differenet components such as enconding scheme, population size, the selection scheme, the crossover and mutation operators.However, 266several experiments have been conducted by differen researchers, which show that the basic GAs easily get trapped into local optima when solving the complex problems. Single learning approach was used in the conventional GAs. This may cause easily falling into local optima with different complex situations. For example different problems may have different properties such as shapes, fitness landscapes. For the effective solution of these problems, different learning tactics might be used for dealing with different scenarios.An adaptive mutation strategy (i.e. single algorithm contains four mutation operators) may be able to overcome the drawback of a conventional strategy (i.e. single mutation operator).Duringeach evolutionary process, the best mutation operator could be selected and useful to generate new solutions,that algorithm would achieve much better than a basic GA. An adaptive scheme is used to serve the purpose of exploration or exploitation during the single iteration. Some researchers have conducted experimental results in this field observing techniques such as integeration of different mutation operators and adaptive control of mutation operators (Hong *et al*, 2000, Li *et al*, 2012, Li *et al*, 2009, Li *et al*, 2008).

2. RELATED WORK

Genetic Algorithms (GAs) areadaptive search techniques which are based on principles of natural selection and genetics. In GAs, the mechanics of natural evolution and genetic inheritance are simulated artificially in order to search for effective solutions to given problem. It works through simple cycle of phases of selection, crossover and mutation. Set of individuals of population is initialized then evoloved from generation to generation by iterative process of selection, crossover and mutation (Goldberg, 1989). In algorithm 1, the use of simple genetic algorithm is mentioned.

Algorithm 1 The basic framework of GAs.

1. Create the initial population
2. Evaluate the fitness of each individual
3. While the termination criteria is not satisfied do
4. for each individual i do
5. choose individual j by the roulette wheel method

++Corresponding Authors: E-mail:imtiaz@usindh.edu.pk. Kamran@usindh.edu.pk

6. crossover individual i with individual j applying the arithmetic crossover method
7. Mutate individual i by using random normal mutation
8. endfor
9. endwhile

2.1.Adaptation in Mutation Scheme

Typically, more research work has been done on chosing the suitable genetic operators and relevant parameters for GAs for searching the global optimimum solution(Hong *et al*, 2000, Li *et al*, 2012, Li *et al*, 2009, Li *et al*, 2008, Thierens, 2005). These parameter settings are fixed before the lunching of GAs, which are derived form user's experience or by trial-and-errors. The static parameter setting may lead to sub-optimal performance. In order to find suitable parameter setting for the good performance of GAs, the situation becomes quite complex. During various stages of the search process of single iternation, there is likelihood of finding appropriate values of parameters and operators. Therefore, reaserachers are interested in using adaptive mechanism for obtaining good results from GAs. Parameter control are adjusted in different ways (Angeline, 1995, Eiben *el at*, 2007). Deterministic adaptation modifies the values of parametes according to some deterministic rules without using any feedback information from the search space. Adaptive adaptation alters straregy variables by applying feedback information from search process. In Self-adaptive adaptation, the parameters are adjusted by EAs themselves.

3. AN ADAPTIVE STRATEGY FOR GENETIC ALGORITHMS (ASGAS)

Single crossover and mutation operator is used to produce offspring in the traditional genetic algorithms. Different mutation schemes may have different search directions in the search space. In order to help GAs to jump out of local optima, several schemes of mutation can be used. Though, the usefulness of these schemes (mutations) differs on different stages of benchmark problems, even similar on the particular problem. Various mutation operators may have finest performance at diferent level of evolutionary process of GAs, as compared to single mutation operator. Based on the simple framework of GAs explained in the previous section, ASGAs algorithm is designed in this section. The concept of ASGAs algorithm is nearly the same as that of conventional GAs algorithms, except for an extra step applied to update the ASGAs. This algorithm integrates four mutation operators.

3.1 Four mutation operators:

Four different situations are considerd in the literature (Li *et al*, 2012, Li *et al*, 2009) regarding the search space such as converging the global best solution, exploiting the best position, exploring new promising area and jumping out of local optima. Due the this reason, we define four mutation operators, which are explained as follows:

Mutation(M01): $x_i^d = \left(x_{avg_i}^d - x_i^d\right)\lambda + r_i^d \delta \quad (1)$

Mutation(M1): $x_i^d = \left(x_{avg_i}^d - x_i^d\right)\lambda \quad (2)$

Mutation(M2): $x_i^d = \left(best_i^d - x_i^d + x_{avg_i}^d - x_i^d\right)\lambda \quad (3)$

Mutation(M3): $x_i^d = \left(best_i^d - x_i^d\right)\lambda + r_i^d \delta \quad (4)$

Every solution of populatoin has four different policies for adjusting behavior respectively. The four approaches can guide a solution to move on to a particularly favorable position with a higher probability than the basic GAs. However, the location of the solution helps to determine the most suitable candidate. Furthermore, it is not possible to determine how does the sarounding environment look like. Each candidate is supposed to detect itself the shape of the environment where it located. Hence, the authors propse using the method in (Li *el at*, 2012, Li *el at*, 2008), which allow an individual to select the most appropriate operator automatically. Further details regarding the method are illustrated as under:.

3.2. GAs with Adaptive Mutation Mechansim:

An Adaptive scheme is borrowed from the probability matching (Thierens, 2005), we propose an adaptive learning framework applying above mentioned mutation operators, selection ratio is assigned to each of these operators, that selection ratio should be equally initialized to ¼ and adaptively updaed according to its relative performance.

For each individual, adaptive scheme selects one of the offered actions/operators based on the selection ratio and its offspring fitness.The selection ratios of those mutation opertaors increases who posses higher fitness values of their offsprings. On the contrary, the selectio ratio of those mutation operators deceases who posses lower fitness values of their offsprings.

The feedback information produced when each chosen action or operator responds with an answer (which is either positive or negative) is used to select its subsequent action or operator. By using this feedback information adaptive approach adjusts the probabilities by means of a learning algorithm. Gradually, the most optimal operator or action will be selected automatically and balance all the mutation behavior in the whole population. The progress, reward and probability vectors are updated at each evolutionary process of Gas, without loss of generality. We consider the minimization optimization problems in this paper.

First, some explanations are given below: The progress value $\eta_i(t)$ of operator i at iteration t isconsidered as follows: [for example if no improvement (such as offspring is not better than its parent) is achieved, a null progress is allocated.

$$\eta_i(t) = \sum_{j=1}^{H_i} \left(pf_j^i(t) - cf_j^i(t)\right) \quad (5)$$

where $pf_j^i(t)$ and $cf_j^i(t)$ represent a parent and its offspring fitness produced by mutation operator i at generation t and H_i is the sum of solutions that chose mutation operator i to mutate.

The reward value $R_i(t)$ of operator i at generation t is defined as follows:

$$R_i(t) = exp\left(\frac{\eta_i(t)}{\sum_{j=1}^{K}\eta_j(t)}\alpha + \frac{h_i}{H_i}(1-\alpha)\right) + c_i p_i(t) - 1, \quad (6)$$

where h_i is the record of individuals those offspring have a better fitness than themselves after being mutated by mutation operator i at evolutionary process t, α is a forgetting factor between (0,1), K is the total number of mutation operators or actions, and c_i is the penalty factor for mutation operator i, that is stated as follows:

$$c_i = \begin{cases} 0.9, & if h_i = 0 \text{ and } p_i(t) = max_{j=1}^{K}(p_i(t)) \\ 1, & otherwise \end{cases} \quad (7)$$

Selection ratio of the current best operator gets decreased if the predecessor operator doesn't contribute in current generation.Based on the above equations, the probability or selecitn ratio of mutation operator i is updated according to the following equation:

$$P_i(t+1) = \frac{R_i(t)}{\sum_{j=1}^{K} R_j(t)}(1 - K * \gamma) + \gamma \quad (8)$$

where γ is the predefined minimum prability for each operator or action, which is set 0.01 for the experiments in this paper. The probility update equation consists four factors such as progress value, the ratio of successful mutations, previous probiblity, and predefine minimum probability.

Algorithm 2 Adaptive mutation based GAs algorithm: AMGAs

1. Create initial population of individuals
2. Evaluate the fitness of each individual
3. Set $t \coloneqq 0$
4. Initiazed some constant parameters [refer to experimental section].
5. Each mutation operator initialized by equal probability.
6. while$t < \max_gen$ do
7. for each individual $i$do
8. Choose one mutation operator according to its probability.
9. Select individual j by roulette wheel selection scheme
10. Crossover on individual i with individual j using the arithmetic crossover approach
11. Apply one mutation operator i roulette wheel selection schem
12. endfor
13. Update the probabilities of each mutation operator according to Eq. (8)
14. $t \coloneqq t + 1$
15. endwhile

By combining the four components such as progress value, the ratio of successful mutation, previous of each mutation operator and predefine minimum probability with traditional GAs, the AMGAs method is developed. The framework of AMGAs is demonstrated in algorithm 2. Some steps (i.e.4,5,8,13) are modified with respect to the conventaional GAs, due to this reason algorithm 2 differs from the simple GAs.

4. EXPERIMENTAL STUDY

In order to measure the performance of proposed approach, we take three unimodal and eighteen multi-modal functions, which are widely tested on a standard set of benchmark functions. These functions are chosen from reference (Liang, *el at.*, 2006, Suganthan *el at.*, 2005, Yao, *el at.*, 1999). The detail of these function are given in **(Table-1)**. Test problems 18 to 21 are rotated problems, where the rotation matrix $\vec{M}$ for each problem is achieved applying the method in (Salomon, 1996).

Three genetic parameters were set which are the roullete wheel method, arithmetic crossover with probability 0.8, and population size (100). For adaptive mutation,the initial probability for each operator was set to 1/4 and for adaptive mechanism minimum probability γ was set to 0.001. The mutation probility was given in the **(Table-2)**, which has differnet values for different test problems.

Table 1:The explanation of benchmark problems (Liang *el at*, 2006, Suganthan *el at*, 2005, Yao *el at*, 1999) where n is the number of dimensions of a problem, D $\in R_n$, f_{min} is thepredefine minimum value of problem. $\vec{M}$ is rotation matrix.

Test problems	pm	n	D	f_{min}
$f_1(x)=\sum_{i=1}^{n} x_i^2$	0.1	10	[-100,100]	0
$f_2(x)=\sum_{i=1}^{n}\left(x_i^2-10\cos(2\pi x_i)+10\right)$	0.01	10	[-5.12,5.12]	0
$f_3(x)=\sum_{i=1}^{n}\left(\sum_{k=0}^{k_{max}}\left[a^k\cos\left(2\pi b^k(x_i+0.5)\right)\right]\right)-n\sum_{k=0}^{k_{max}}\left[a^k\cos(\pi b^k)\right],\ a=0.5,\ b=3,\ k_{max}=20$	0.01	10	[-0.5,0.5]	0
$f_4(x)=\frac{1}{4000}\sum_{i=1}^{n}(x_i-100)2-\prod_{i=1}^{n}\cos\left(\frac{x_i-100}{\sqrt{i}}\right)+1$	0.01	10	[-600, 600]	0
$f_5(x)=-20\,exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}\right)-exp\left(\frac{1}{2}\sum_{i=1}^{n}\cos(2\pi x_i)\right)+20+e$	0.01	10	[-32, 32]	0
$f_6(x)=\sum_{i=1}^{n}(\lfloor x_i+0.5\rfloor)^2$	0.1	10	[-100,100]	0
$f_7(x)=\sum_{i=1}^{n}ix_i^4+U(0,1)$	0.05	10	[-1.28,1.28]	0
$f_8(x)=\sum_{i=1}^{n}100\left(x_{i+1}^2-x_i\right)^2+(x_i-1)^2$	0.05	10	[-100,100]	0
$f_9(x)=\sum_{i=1}^{n}-x_i\sin\left(\sqrt{\lvert x_i\rvert}\right)$	0.01	10	[-500,500]	-4189
$f_{10}(x)=418.9829.n+\sum_{i=1}^{n}-x_i\sin\left(\sqrt{\lvert x_i\rvert}\right)$	0.01	10	[-500,500]	0
$f_{11}(x)=\sum_{i=1}^{n}\lvert x_i\rvert+\prod_{i=1}^{n}\lvert x_i\rvert$	0.01	10	[-10,10]	0
$f_{12}(x)=\sum_{i=1}^{n}(\sum_{j=1}^{i}x_j)^2$	0.01	10	[-100, 100]	0
$f_{13}(x)=max_{i=1}^{n}\lvert x_i\rvert$	0.01	10	[-100,100]	0
$f_{14}(x)=\frac{\pi}{30}\left\{10sin^2(\pi y_1)+\sum_{i=1}^{n-1}(y_i-1)^2.[1+10sin^2(\pi y_{i+1})]+(y_n-1)^2\right\}+\sum_{i=1}^{n}\mu(x_i,5,100,4),y_i=1+{}^{(x_i+1)}/_{4}$	0.01	10	[-50,50]	0
$f_{15}(x)=0.1\left\{10sin^2(\pi y_1)+\sum_{i=1}^{n-1}(x_i-1)^2.[1+sin^2(3\pi x_{i+1})]+(x_n-1)^2[1+sin^2(2\pi x_n)]\right\}+\sum_{i=1}^{n}\mu(x_i,5,100,4)$	0.05	10	[-50,50]	0
$f_{16}(x)=\sum_{i=1}^{n}100\left(y_{i+1}^2-y_i\right)^2+(y_i-1)^2,\vec{y}=\vec{M}*\vec{x}$	0.05	10	[-500, 500]	0
$f_{17}(x)=\frac{1}{4000}\sum_{i=1}^{n}(y_i-100)2-\prod_{i=1}^{n}\cos\left(\frac{y_i-100}{\sqrt{i}}\right)+1,\vec{y}=\vec{M}*\vec{x}$	0.01	10	[-10, 10]	0
$f_{18}(x)=-20\,exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}y_i^2}\right)-exp\left(\frac{1}{2}\sum_{i=1}^{n}\cos(2\pi y_i)\right)+20+e,\vec{y}=\vec{M}*\vec{x}$	0.01	10	[-100,100]	0
$f_{19}(x)=\sum_{i=1}^{n}\left(y_i^2-10\cos(2\pi y_i)+10\right),\vec{y}=\vec{M}*\vec{x}$	0.05	10	[-5,5]	0
$f_{20}(x)=\sum_{i=1}^{n}\left(\sum_{k=0}^{k_{max}}\left[a^k\cos\left(2\pi b^k(y_i+0.5)\right)\right]\right)-n\sum_{k=0}^{k_{max}}\left[a^k\cos(\pi b^k)\right],a=0.5,b=3,k_{max}=20,\vec{y}=\vec{M}*\vec{x}$	0.01	10	[-0.5,0.5]	0

4.2 Exerimental Result and Analsis:

The average result of 30 independent trails of the GAs with four mutation operators and the GAs with adaptive mutation on the standard test problems are presented in **(Table-2)**. From **(Table-2)**, it can be seen that different mutation operator performance varies on different problems. The performance of M1-M4 and adaptive mutation (AM) are identicall on few of the test problems. The best result of all mutation operators is highlighted in the **(Table-1)**. On some of the functions,

M1 and M2 operators are better than other three operators. AM operator gets the balance result on all standered test problems. Optimum result is achieved by all mutation operators on f_2, f_3, and f_6. Both mutation algorithms however obtain the global minimum optimum result on f_4 and f_8.

Table 2: Average best fitness result over 30 independent tarils of algorithms on benchmark problems.

Funtion	f_1	f_2	f_3	f_4	f_5	f_6	f_7
AMGAs	4.869e-163	0	0	0.00601385	3.878e-015	0	0.000172
M0	7.589e-082	0	0	0.0720698	1.590e-013	0	0.000270
M1	2.422e-181	0	0	0	4.441e-016	0	0.000116
M2	2.987e-173	0	0	0	4.441e-016	0	0.000142
M3	1.835e-125	0	0	0.0882026	1.062e-014	0	0.000235

Funtion	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}
AMGAs	0	11.9415	-2528.24	1732.7	2.118e-024	0.266751	1.936e-005
M0	0.00453	19.503	-2502.1	1775.69	1.760e-016	0.173153	0.0009037
M1	0	8.64308	-2436.47	1749.55	1.229e-026	15.4378	6.983e-007
M2	0	15.212	-2507.82	1785.69	1.198e-028	16.3872	3.280e-007
M3	0.00518	21.6417	-2523.67	1698.92	1.627e-019	0.209032	0.000711

Funtion	f_{15}	f_{16}	f_{17}	f_{18}	f_{19}	f_{20}
AMGAs	0.00300	0.099308	1.510e-165	76.5802	0.104473	3.641e-015
M0	0.00425	0.0901201	9.195e-082	75.9182	0.15912	1.342e-013
M1	0.00550	0.106859	1.703e-185	73.0674	0.0960584	4.440e-016
M2	0.00177	0.0430605	2.266e-175	93.9993	0.0791828	4.440e-016
M3	0.00224	0.0607695	4.231e-128	77.6174	0.174947	1.003e-014

Fig.1 and **Fig. 2** both show the evolutionary process for all mutation opeators on all standared benchmark problems, and the results on these functions are displayed in a log scale. When compared the result of each mutation during the evolutionary process, all five mutation operators achieved good results on the test problems. **(Fig. 1)** shows that M2 and M3 obtain best results on most of the problems. M1 operator has the capablility to produce high diversity, due to this reason, it gives worst result on most of the benchmark problems.

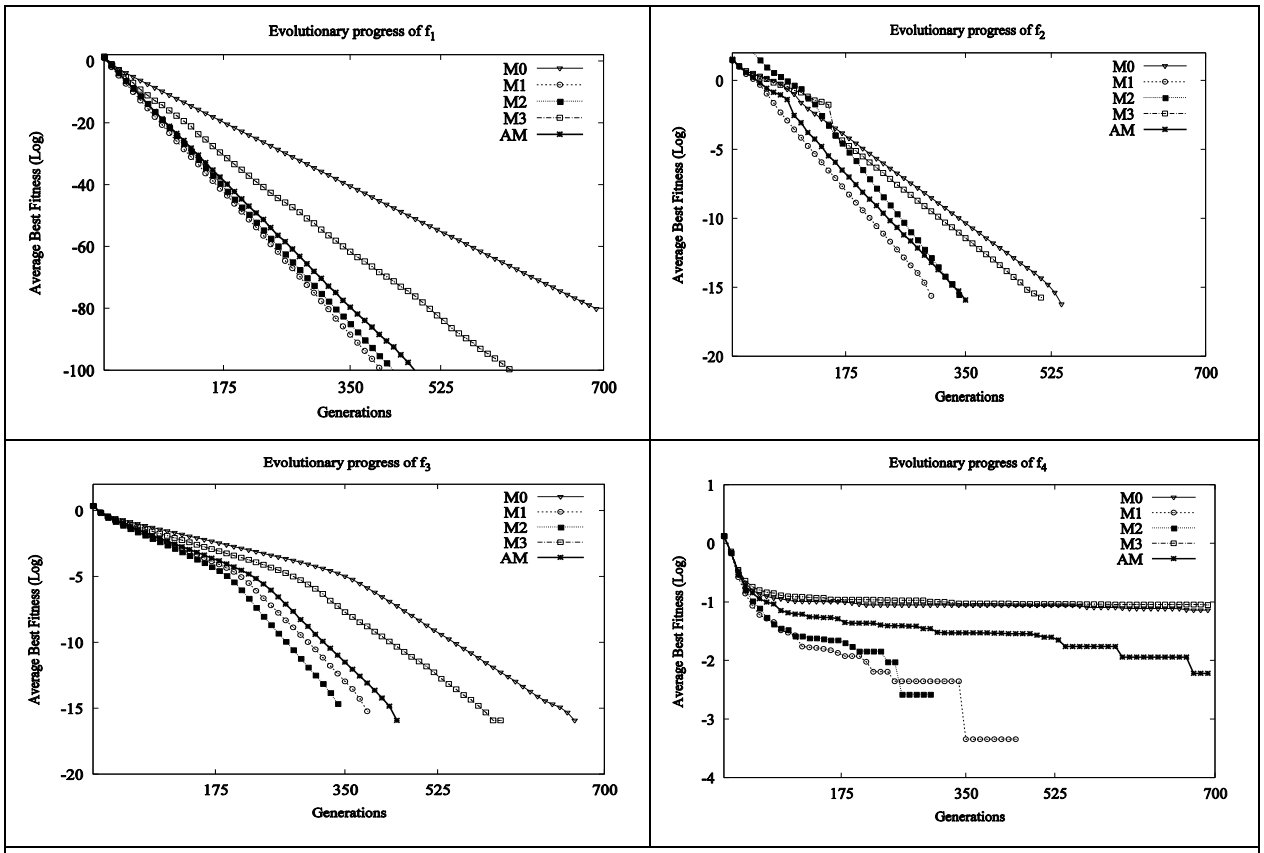

Fig.1. Evolutionary process of GAs with Single Mutation and AMGAs on Selected Functions.

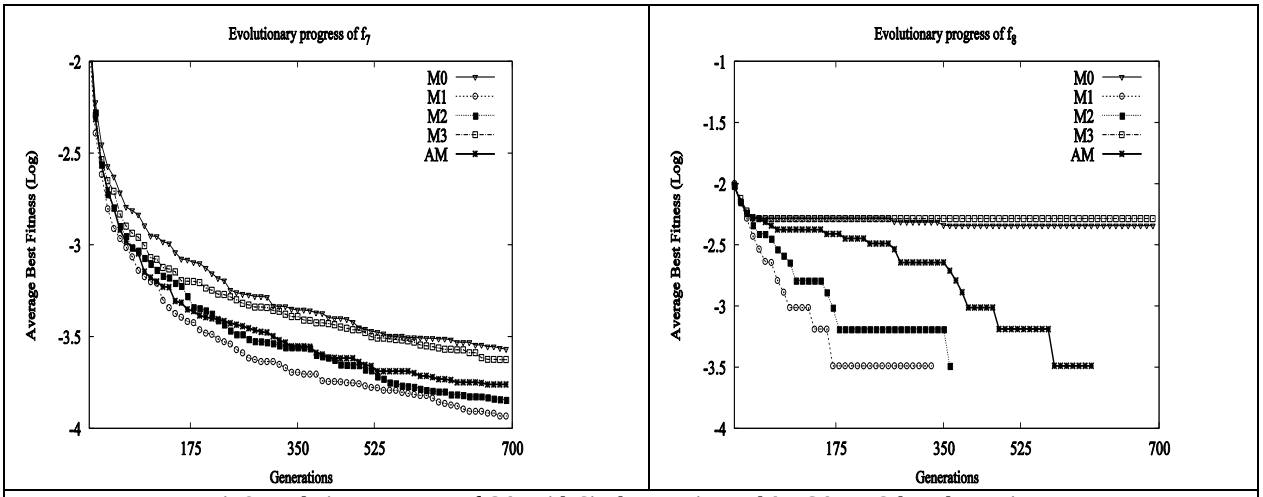

Fig.2. Evolutionary process of GAs with Single Mutation and AMGAs on Selected Functions.

5. CONCLUSION

This paper presents adaptive mutation operator for AMGAs. In this approach, selection method and relative fitness improvement, based on the progress value ,are incorporated in the simple GAs. By implementing this adaptive approach paradiam, which enables an individual to adaptivebly adjust its search behavior during the evolutionary process for global optimization problems,balanced performance is achieved for all standard benchmark problems. Single solution has multiple learning options produced by multiple mutation operators, which have different properties to guide individuals to converge on the current global best position, exploiting a local optimum, exploring a new desirable area and moving away from the local optimum. As compared to any other mutation operator the proposed adaptive approach has been found more effective than all. Thus, the addition of AMGAs mechanism is aencouraging work for enhancing the ability of GAs.

REFERENCES:

Angeline, P. J., (1995). Adaptive and self-adaptive evolutionary computations. In M. Palaniswami, Y Attikiouzel, R. J. Marks, D. B. Fogel, and T. Fukuda, (eds.) Computational Intelligence: A Dynamic Systems perpective, IEEE Press New York, 152-163.

Eiben, A. E, Z. Michalewics, M. Schoenauer, J. E. Smith, (2007). Parameter control in evolutionary algorithms, ch.2,19-46.

Goldberg, D. E., (1989). Genetic Algorithm in Search, Optimization. Addison-Wesley.

Holland, H. J., (1975). Adaptation in Natural and Artificial Systems. Ann Arbor, Univeristy of Michigan.

Hong, T. P., H. S. Wang, W. C. Chen, (2000). Simultaneously applying multiple mutation operators in genetic algorithms. Journal of Heuristics, 6: 439-455.

Li, C., S. Yang, T. T. Nguyen, (2012). A self-learning particle swarm optimizer for global optimization problems, IEEE Transaction on Systems Man and Cybernetics 42(3).

Li, C., S. Yang, (2009). An adaptive learning particle swarm optimizer for function optimization, Congr. Evol. Comput, 381-388.

Li, C., S. Yang, I. A. Korejo, (2008). An adaptive mutation operator for particle swarm optimization, 2008 UK Workshop Computational. Intelligence, 165-170.

Salomon, R., (1996). Reevaluating genetic algorithm performance under coordinate rotation of benchmark functions: A survery of some theoretical and practical aspects of genetic algorithms. BioSystem. 39(3), 263-278.

Liang,J. J., A. K. Qin, P. N. Suganthan, S. Baska, (2006). Comprehensive learning particle swarm optimizer for global optimization of multimodal functions, IEEE Transaction on 10(3), 281-295.

Suganthan, P. N., N. Hansen, J. J. Liang, P. C. K. Deb, A. Auger, S. Tiwari, (2005). Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization,Nanyang Technological Univeristy, Singapore, Technical Report.

Thierens, D., (2005). An Adaptive pursuit Strategy for allocating operator probabilities. In Proc. Genetic Evol, Compu, Conf, 1539-1546.

Yao, X., Y. Liu, G. Lin, (1999). Evolutionary programming made faster, IEEE Transaction on 3(2), 82-90