ISSN No. 0976-5697
Volume 3, No. 3, May-June 2012
International Journal of Advanced Research in Computer Science
RESEARCH PAPER
Available Online at www.ijarcs.info

Implementation And Analysis Of Benchmarking Test Function For Genetic Operators

Puja Kumari*
M.Tech (CSE) Student
N.C.College of Engineering Panipat, India
Puja.8986@gmail.com

Vaishali Wadhwa
Assistant Professor (CSE)
N.C.College of Engineering Panipat, India
Wadhwavaishali25@yahoo.in

Abstract: The GA is a global search method that mimics the metaphor of natural biological evolution.In this paper; we have used six popular benchmark functions for studying the performance of GAs and GA operators. They are Rastrigin's function, Rosenbrock Function, Sphere Function, Ackley Function, Generalized Rastrigin and Branin Function.

Keywords: Genetic Algorithm; Benchmarking; GA; Genetic Algorithm; Matlab

I. INTRODUCTION

Genetic Algorithms (GAs) are a class of probabilistic algorithms that are loosely based on biological evolution. This paper presents experimental results on the major benchmarking functions used for performance evaluation of Genetic Algorithms (GAs). GAs relies heavily on random number generators. In addition, each of the basic genetic operators used in a simple GA (crossover, mutation) utilizes ``random choice'' to one extent or another. [1]

II. GENETIC ALGORITHM

The GA is a stochastic global search method that mimics the metaphor of natural biological evolution. GAs operates on a population of potential solutions applying the principle of survival of the fittest to produce (hopefully) better and better approximations to a solution. At each generation, a new set of approximations is created by the process of selecting individuals according to their level of fitness in the problem domain and breeding them together using operators borrowed from natural genetics. This process leads to the evolution of populations of individuals that are better suited to their environment than the individuals that they were created from, just as in natural adaptation.

Individuals, or current approximations, are encoded as strings, *chromosomes*, composed over some alphabet(s), so that the *genotypes* (chromosome values) are uniquely mapped onto the decision variable (*phenotypic*) domain. The most commonly used representation in GAs is the binary alphabet {0, 1} although other representations can also be used, e.g. ternary, integer, real-valued etc.

A. Major Elements of the Genetic Algorithm:

a. Initialization:-

Start with a population of randomly generated individuals, or use

a) A previously saved population.
b) A set of solutions provided by a human expert.
c) A set of solutions provided by another heuristic algorithm.

b. Evaluation:-

a) Solution is only as good as the evaluation function; choosing a good one is often the hardest part.
b) Similar-encoded solutions should have a similar fitness.

c. Termination condition:-

a) A pre-determined number of generations or time has elapsed
b) A satisfactory solution has been achieved
c) No improvement in solution quality has taken place for a pre-determined number of generations

The Evolutionary Cycle:

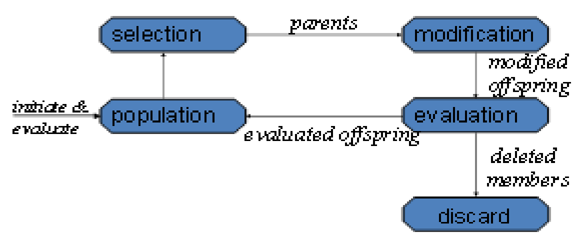

Figure: 1

B. The Objective and Fitness Functions:-

The objective function is used to provide a measure of how individuals have performed in the problem domain. The fitness function is normally used to transform the objective function value into a measure of relative fitness [2], thus:

$$F(x) = g(f(x))$$

Where f is the objective function, g transforms the value of the objective function to a non-negative number and F is the resulting relative fitness. This mapping is always necessary when the objective function is to be minimized as the lower objective function values correspond to fitter individuals.

C. Selection Methods:-

Selection is the process of determining the number of times, or trials, a particular individual are chosen for reproduction and, thus, the number of offspring that an

© 2010, IJARCS All Rights Reserved

individual will produce. The selection of individuals can be viewed as two separate processes:

a) determination of the number of trials an individual can expect to receive, and
b) Conversion of the expected number of trials into a discrete number of offspring.[3], [4]

a. Roulette Wheel Selection Methods:

Many selection techniques employ a “roulette wheel” mechanism to probabilistically select individuals based on some measure of their performance. A real-valued interval, Sum, is determined as either the sum of the individuals’ expected selection probabilities or the sum of the raw fitness values over all the individuals in the current population

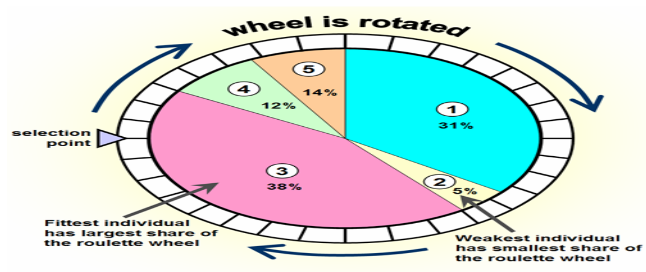

Figure: 1

b. Crossover:-

The basic operator for producing new chromosomes in the GA is that of crossover. Crossover:

a) Crossover point(s) is determined stochastically.
b) The Crossover Operator is the most important feature in a GA. [6]

Single Point Crossover Example:-

Parent 1	1 0 0	1 0 0 1 0 1 0
Parent 2	0 0 1	0 1 1 0 1 1 1
Child 1	1 0 0	0 1 1 0 1 1 1
Child 2	0 0 1	1 0 0 1 0 1 0

Double Point Crossover Example:-

Parent 1	1 1 0 1 0 0	1 0 0 1	0 1 1
Parent 2	0 1 0 1 1 0	0 0 1 0	1 0 1
Child 1	1 1 0 1 0 0	0 0 1 0	0 1 1
Child 2	0 1 0 1 1 0	1 0 0 1	1 0 1

c. Mutation:-

In natural evolution, mutation is a random process where one allele of a gene is replaced by another to produce a new genetic structure. In GAs, mutation is randomly applied with low probability, typically in the range 0.001 and 0.01, and modifies elements in the chromosomes.

a) The Mutation operator guarantees the entire state-space will be searched, given enough time.
b) Restores lost information or adds information to the population.
c) Performed on a child after crossover.
d) Performed infrequently (For example, 0.005 probability of altering a gene in a chromosome).

Child 1	1 1 0 1 0 0 0 0 1 0 0 1 1
After mutation	1 1 0 1 1 0 0 0 1 0 0 1 1

d. Termination of the GA:-

The following are the termination criteria. A solution is found that satisfies minimum criteria.

a) Fixed number of generations is reached.
b) Allocated budget (computation time/money) is reached.
c) The highest-ranking solution’s fitness is reached or has reached a plateau such that successive iteration no longer produces better result.
d) Manual setting of inspection criteria.

Combination of two or more criteria can also be used. [5]

III. BENCHMARKING

Within this theory, a benchmark is defined as a standardized test or set of tests used for comparing alternatives. A benchmark has three components, a Motivating Comparison, a Task Sample, and Performance Measures.

A. Benefits of Benchmarking:-

Benchmarking can have a strong positive effect on the scientific maturity of a research community. The benefits of benchmarking include a stronger consensus on the community’s research goals, greater collaboration between laboratories, more rigorous examination of research results, and faster technical progress.

B. Dangers of Benchmarking:-

Any discussion of benchmarking must include consideration of the costs and risks. There is a significant cost to developing and maintaining the benchmark, so there is a danger in committing to a benchmark too early. Tychy wrote: “Constructing a benchmark is usually intense work, but several laboratories can share the burden. Once defined, a benchmark can be executed repeatedly at moderate cost. In practice, it is necessary to evolve benchmarks to prevent over fitting”.

C. Optimization techniques:-

The parameters that best fit the experimental curve were determined by minimizing the difference between the experimental and the simulated data. The parameters that characterized the system were involved in non – linear partial differential equations. As an example to show how the techniques were applied, the experimental data used were the ketone concentration from Daly and Yin’s paper. If Y*i* (X*i*) represents the experimental ketone concentration values and S*i* (X*i*) are the simulated ones obtained by solving the partial differential equations, the function to be minimized was

$$F=\sum_i (S_i (X_{i)} - Y_i (X_i))^2$$

The parameters were selected by the optimization algorithm in such a manner that the function kept decreasing. The search of the best parameters was terminated when the function F fell below an acceptable value. There were numerous techniques available to solve the problem. We show two techniques we employed in determination of the best rate constants.

© 2010, IJARCS All Rights Reserved

a. Direct Search Method for Optimization:

The technique was very simple. To explain the technique of optimization, we pick the simple example of Daly and Yin's model [7]. This model was considered because it employed two reaction rate equations and thus two rate constants. This made it convenient to explain the techniques employed for the optimization of our models. The reaction system in Daly and Yin's model [7] was given as follows:

$R^* + O_2 \rightarrow RO_2^*$ (k_1)

$RO_2^* + R^* \rightarrow RCO + 2R^*$ (k_2)

We started with a guess value of the rate constant parameters. The direct search technique followed a simple step procedure.

a) It increased one of the parameters, say k_1, by a pre – set incremental value given by Δk_1. The simulated ketone concentration values were evaluated at this new rate constant and then we evaluated the function F. We stored this value as F_1 corresponding to $F(k_1 + \Delta k_1, k_2)$.
b) Similarly the technique then decreased the value of the first rate constant by the same incremental value Δk_1. The function was evaluated at this new rate constant and stored in a value F_2 corresponding to $F(k_1 - \Delta k_1, k_2)$.
c) Next, the technique selected the function that has smaller of the two values (F_1 or F_2). It retained the rate constants that gave the minimum function as the new set of refined rate constants. For example, if $F_2 < F_1$, then the new rate constants are $k_1 = k_1 - \Delta k_1$ and $k_2 = k_2$.
d) It checked whether the function $F = F_2$ had dropped below the termination criteria. If it did, then the technique stopped and the rate constants were returned as the best rate constants. If not then the technique proceeded to step 5.
e) Similarly as in step 1 and 2, the technique evaluated $F_3 = F(k_1, k_2 + \Delta k_2)$ and $F_4 = F(k_1, k_2 - \Delta k_2)$, where Δk_2 was an incremental value in F_2.
f) Given the next two function values, the technique then compares the values and selects one which is lower and updates the rate constants. It then checked whether the function had dropped below the termination criteria. If it did, then the technique stopped and the rate constants were reported as the best rate constants. If not then the procedure was repeated from step 1.
g) The Direct Search technique begins as shown with an initial guess and then proceeds by picking the rate constants that minimize the function F. The technique is very well suited for optimization of two parameters.

b. Sequential Programming:

The nature of the solution technique was sequential in nature, which meant, we needed to solve for each increment and decrement. This sequential problem was written in Fortran 77 to apply the technique to minimize the function for Daly and Yin's model.

c. Parallel Computation

Evaluation of the each function value corresponding to each increment and decrement of the rate constants (parameters) is an independent process with respect to each other. This means, $F(k_1, k_2)$, $F_1(k_1 + \Delta k_1, k_2)$, $F_2(k_1 - \Delta k_1, k_2)$, $F_3(k_1, k_2 + \Delta k_2)$ and $F_4(k_1, k_2 - \Delta k_2)$ can be evaluated independently by different processors without affecting each other and then brought together to compare the function value for one set of rate constants was now utilized for evaluating five function values simultaneously, thereby effectively reducing the time of computation.

Hence, if we had to optimize a two parameter function such as one for Daly and Yin's model [7], $F(x_1, x_2)$, then we carried out the following optimization procedure:

Evaluate

$F_1 = F(k_1 + \Delta k_1, k_2)$

$F_2 = F(k_1 - \Delta k_1, k_2)$

$F_3 = F(k_1, k_2 + \Delta k_2)$

$F_4 = F(k_1, k_2 - \Delta k_2)$

The four functions were computed by each processor simultaneously. After each function was evaluated, the values were sent to the root processor that compared the values of all the functions and kept one with the minimum value. The root processor employed MPI_REDUCE and MPI_MINLOC to find the minimum function. It checked for the termination criteria and if not met then the corresponding rate constants were broadcasted to the four processors and the process continued until the function F fell below termination criteria. [8]

IV. TEST FUNCTIONS

We use six popular benchmark functions for studying the performance of GAs and GA operators. They are Rastrigin's function, Rosenbrock Function, Sphere Function, Ackley Function, Generalized Rastrigin and Branin Function.
These functions and the fitness functions are described below:

A. Rastrigin's Function:-

Rastrigin's function is based on the function of De Jong with the addition of cosine modulation in order to produce frequent local minima. Thus, the test function is highly multimodal. However, the location of the minima is regularly distributed.
Function has the following definition: [13], [9], [10]

$$F(x) = 10n + \sum_{i=1}^{n} x_i^2 - 10 \cos(2\pi x_i)].$$

Test area is usually restricted to hypercube $\boldsymbol{-5.12 <= x_I <= 5.12,\ i=1\dots n}$.
Global minimum $f(x) = 0$ is obtainable for $\boldsymbol{x_I = 0,\ I = 1\dots n}$.

B. Rosenbrock's Valley:-

Rosenbrock's valley is a classic optimization problem, also known as banana function or the second function of De Jong. The global optimum lies inside a long, narrow, parabolic shaped flat valley. To find the valley is trivial, however convergence to the global optimum is difficult and hence this problem has been frequently used to test the

© 2010, IJARCS All Rights Reserved

performance of optimization algorithms. Function has the following definition [9], [13]

$$F(x) = \sum_{i=1}^{n-1} [100(x_i+1-x^2_i)^2 + (1-x_i)^2].$$

Test area is usually restricted to hypercube $-2.48<=x_i 2.48$, $i=1\ldots n$.

Its global minimum equal $f(x) = 0$ is obtainable for x_i, i=1… n.

C. The Sphere Function:-

The Sphere function is defined as follows: [13], [9]

$$F(x) = \sum_{i=1}^{D} x^2_i$$

Where D is the dimension and x = ($x1$, $x2$… x D) is a D-dimensional row vector (i.e., a $1\times D$ matrix). The Sphere function is very simple and is mainly used for demonstration. In this test suite this function serves as separable part when using a naturally no separable function to form some partially no separable functions.

D. Ackley Function:-

Ackley's is a widely used multimodal test function. It has the following definition: [9], [10]

$$F(x) = -a.\exp(-b.^2_i\ ) - \exp(1/n \sum_{i=1}^{n} \cos(cx_i)) + a + \exp(1)$$

It is recommended to set $a = 20$, $b = 0.2$, $c = 2\pi$ Test area is usually restricted to hypercube $-32.768 <= x_i <= 32.768$, i = 1… n. Its global minimum $f(x) = 0$ is obtainable for $x_i = 0$, i = 1… n.

E. Generalized Rastrigin Function:-

The Generalized Rastrigin Function (Equation 1) is a typical example of non-linear multimodal function. It was first proposed by Rastrigin as a 2-dimensional function and has been generalized by Miihlenbein et al in. This function is a fairly difficult problem due to its large search space and its large number of local minima. [9]

$$F(x) = A.\ n + \sum_{i=1}^{n} x\ 2i - A.\ \cos(\omega.\ x_i)$$

$$A=10,\ \omega=2.\pi,\ x_i \ \square\ [-5.12, 5.12]$$

The Rastrigin function has a complexity of O (n1n (n)), where n is the dimension of the problem. The surface of the function is determined by the external variables A and ω, which control the amplitude and frequency modulation respectively.

F. Branins's Function:-

The Branin function is a global optimization test function having only two variables. The function has three equal-sized global optima, and has the following definition: [9], [10]

$$F(x1, x2) = a(x_2 - bx^2_1 + cx_1 + d)^2 + e(1-f)\cos(x_1) + e.$$

It is recommended to set the following values of parameters: $a = 1$, $b = 5.1/4\pi^2$,

$c = 5/\pi$, $d = 6$, $e = 10$, $f=1/8\ \pi$. Three global optima equal $f(x_1, x_2) = 0.397887$ are located as follows: $(x_1, x_2) = (-\pi, 12.275), (\pi, 2.275), (9.42478, 2.475)$.

V. RESULT AND CONCLUSION

Table1, Table 2, Table 3 shows fitness with 2 variables, 10 variables and 20 variables when Population Type is changing while other options remain default which are shown as:-

Fitness scaling function=Rank, Selection function=Stochastic uniform, Mutation function=Gaussian, Crossover function=Scattered

Table 1: Fitness with 2 variable when population type changes

Functions	**Population Type**			
	Double Vector		**Bit String**	
	Best Fitness	**Mean Fitness**	**Best Fitness**	**Mean Fitness**
Rastrigin's function	0.04066	3.78650	0	0
Rosenbrock Function	0.01149	18.8125	0	5.0000
Sphere Function	0.00275	0.34290	0	0
Ackley Function	0.01972	1.17570	8.8818e-16	8.8818e-16
Generalized Rastrigin	0.02989	8.42150	0	0
Branin Function	0.39799	0.95491	0	0

Table 1 show the results for fitness when we vary the population type. The table shows the result for two variables. The results show that when we change the population type from Double Vector to Bit String, there is change in fitness (best fitness, mean fitness) and the fitness is decrease. Remember that we are minimizing the benchmarks functions, so lower the fitness value, better is the performance. The table shows when we choose Bit String as population type, the performance of genetic algorithm is much better than double Vector.

Table 2: Fitness with 10 variable when population type changes

Functions	**Population Type**			
	Double Vector		**Bit String**	
	Best Fitness	**Mean Fitness**	**Best Fitness**	**Mean Fitness**
Rastrigin's function	16.8745	55.0668	0	0.05
Rosenbrock Function	39.4568	465.619	9	9
Sphere Function	0.19307	1.22630	0	0.05
Ackley Function	0.76575	1.94870	-8.8818e-16	0.06128
Generalized Rastrigin	15.1837	52.0701	2	2
Branin Function	0.39813	0.97380	0	0.05

Table 2 shows the result for best fitness and mean fitness value when we varies the population type and use the different benchmarks functions with ten variables. The fitness value is increased as compared to fitness value when we use the function with two variables. As a comparative study when we compare the two of population type using function with ten variables the result shows that the performance of bit string population type is better than double vector.

© 2010, IJARCS All Rights Reserved

Table 3: Fitness with 20 variable when population type changes

Functions	Population Type			
	Double Vector		Bit String	
	Best Fitness	Mean Fitness	Best Fitness	Mean Fitness
Rastrigin's function	27.4594	98.786	1	1.15
Rosenbrock Function	145.9792	1014.52	105	105
Sphere Function	1.37	3.6269	1	1.15
Ackley Function	1.8748	2.6855	1.2257	1.2257
Generalized Rastrigin	42.7229	114.3473	1	1.15
Branin Function	9.718	17.4735	4.2813	4.2813

Table 3 shows the results for different benchmarks functions with twenty variables when we change the population type. The results confirm the conclusion of table 1 and table 2. Like, table 1 and table 2, the result of table 3 shows that performance of genetic algorithm is much better when we use the bit string as population type as compare to double vector.

Table for fitness when Scaling Function is changing while other options remain default which is shown as:-
Population type=Double vector, Selection function=Stochastic uniform,
Mutation function=Gaussian, Crossover function=Scattered

Table 4: Fitness when scaling function changes

Function	Selection Function									
	Stochastic		Remainder		Uniform		Roulette Wheel		Tournament	
	Best	Mean	Best	Mean	Best	Mean	Best	Mean	Best	Mean
Rastrigin's function	0.0122	6.5048	0.0105	5.2067	0.0825	14.195	0.0008	8.9790	1.0245	6.2878
Rosenbrock Function	0.0586	68.258	0.0061	24.265	0.0849	143.70	0.5238	25.273	0.4904	32.758
Sphere Function	0.0002	0.2378	0.0003	0.2564	0.0014	0.6107	0.0002	0.3090	0.0110	0.0567
Ackley Function	0.0311	1.5726	0.0766	1.7593	0.2349	2.9226	0.1223	1.0676	0.2763	0.0799
Generalized Rastrigin	1.0062	10.139	0.0097	11.752	0.1039	13.816	1.0073	11.743	1.3382	5.0775
Branin Function	0.3979	2.3684	0.3994	1.2960	0.4044	2.3745	0.3991	0.8085	0.4149	1.2595

Table 5: Fitness when selection function changes

Function	Scaling Function							
	Rank		Proportion		Top		Shift Linear	
	Best	Mean	Best	Mean	Best	Mean	Best	Mean
Rastrigin's function	0.0351	6.3410	0.0171	6.3242	0.2162	4.0440	0.1479	1.5511
Rosenbrock Function	0.0151	42.474	0.0066	25.532	0.3895	9.4210	0.0355	30.445
Sphere Function	0.0006	0.3179	0.0003	0.2083	0.0018	0.1147	0.0010	0.2313
Ackley Function	0.0040	1.3805	0.0942	1.0969	0.9159	1.3704	0.09123	0.8008
Generalized Rastrigin	0.0666	5.3002	0.0974	5.1373	0.0513	3.3444	0.0220	4.3985
Branin Function	0.3984	0.6950	0.3983	0.9330	0.4085	0.7495	0.4000	1.5352

Table 4 shows the performance of six benchmarks function in the form of mean fitness and best fitness. The fitness shown is evaluated using benchmarks function with four variables. The performance shows the comparison between all four types of scaling function (i.e. Rank, Proportion, Top and Shift Linear. The results show that performance of *proportion* using scaling function is much better than performance of *Rank* scaling function followed by *top* scaling function.

Table for fitness when Selection function is changing while other options remain default which are shown as: -
Population type=Double vector, Scaling function=Rank,
Mutation function=Gaussian, Crossover function=Scattered
The table 5 shows the results when selection function is changing (i.e. stochastic, Remainder, Uniform, Roulette Wheel, Tournament). The table shows the comparison between performances of genetic algorithm by using different types of scaling function in the genetic algorithm. The table shows that when is used *Remainder* as a scaling function the best fitness and mean fitness is comes better than *Roulette wheel* when it is used as a scaling function followed by the performance of *Stochastic Function*.
Table for fitness when Mutation function is changing while other options remain default as:-
Population type=Double vector, Scaling function=Rank, Selection function= Stochastic uniform, Crossover function=Scattered

Table 6 shows the comparison in the performance of genetic algorithm when we change the mutation function. The mutation function is change due to comparative study of performance evaluation by using the different benchmarks function with four variables. The results from tables 6 show that when we used *Gaussian* as mutation function the performance of genetic algorithm is improved than other two functions. The performance of *Gaussian* is superior than performance of *Adaptive Feasible* followed by *Uniform* mutation function.

Table for fitness when Crossover function is changing while other options remain default which is shown as:-

© 2010, IJARCS All Rights Reserved

Population type=Double vector, Scaling function=Rank, Selection function=Stochastic uniform, Mutation Function=Gaussian

Table 7 shows the best fitness and mean fitness of the all six benchmarks functions. The results shows that when we use two points crossover function the results are always superior to other types of crossover function. After two points crossover function the performance of single point crossover is much better followed by scattered crossover function.

Table 6: Fitness when mutation function changes

Function	Mutation Function					
	Gaussian		Uniform		Adaptive Feasible	
	Best	Mean	Best	Mean	Best	Mean
Rastrigin's function	0.2285	9.6641	0.4066	0.4066	0	0
Rosenbrock Function	0.0060	69.693	0.1577	0.1577	0.0060	5.4151
Sphere Function	0.0011	0.2050	0.0018	0.0146	1.7946e-015	1.1542e-013
Ackley Function	0.0612	1.1228	0.2212	0.2212	6.5582e-008	3.6219e-007
Generalized Rastrigin	0.1090	5.4176	1.7482	1.7482	5.2047e-013	3.4807e-012
Branin Function	0.4042	0.9490	30.646	30.646	0.4058	0.4442

Table 7: Fitness when crossover function changes

Function	Crossover Function											
	Scattered		Single Point		Two Points		Intermediate		Heuristic		Arithmetic	
	Best	Mean	Best	Mean	Best	Mean	Best	Mean	Best	Mean	Best	Mean
Rastrigin's Function	0.1121	6.9208	0.0028	5.6924	0.0766	6.4230	0.9950	7.5540	0.9949	5.8189	0.9949	8.6786
Rosenbrock Function	0.0103	21.646	0.1497	47.808	0.0023	24.716	0.0058	42.109	0.0005	5.8141	0.0016	32.281
Sphere Function	0.0005	0.0848	0.0003	0.2586	0.0001	0.1360	5.5e-06	0.1425	6.4e-10	0.1731	4.9e-08	0.1481
Ackley Function	0.0324	1.2293	0.0361	1.7253	0.0305	1.1409	0.0002	1.3888	8.2e-05	0.8012	0.0006	1.1908
Generalized Rastrigin	0.0977	8.9253	0.1455	5.8766	0.2650	5.3562	0.9949	4.9667	0.9949	5.0541	0.0037	8.4358
Branin Function	0.3997	1.3886	0.3991	0.7614	0.3991	2.2225	0.3980	1.0070	0.3978	0.7386	0.3978	0.5975

VI. REFERENCES

[1]. http://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol4/tcw2/report.html#Internet

[2]. Pohlheim, H. (1997). ''GEATbx: Genetic and Evolutionary Algorithm Toolbox for use with MATLAB'', http://www.systemtechnik.tuilmenau.de/pohlheim/GA_Toolbox

[3]. Tom V.Mathew ,"Genetic Algorithm", http://www.civil.iitb.ac.in/tvm/2701_dga/2701-ga-notes/gadoc.pdf

[4]. Fundamentals of genetic algorithms:AI course lecture 39-40,notes,slides, http://www.myreaders.info/09_Genetic_Algorithms.pdf June 01,2010

[5]. Andrew chipperfield,Peter Flemming,Hartmut and Carlos,"Genetic Algorithm Toolbox", http://www.shef.ac.uk/polopoly_fs/1.60188!/file/manual.pdf

[6]. J.J.Liang, P.N.Suganthan and K.Deb,"Novel Compostion Test Functions For Numerical Global Optimization",2005,IEEE

[7]. More, J. J., Garbow, B. S., and Hillstrom, K. E. (1981). ''Testing Unconstrained Optimization Software, {ACM} Transactions on Mathematical Software, vol 7, pp. 17–41

[8]. Edwin S.H. Hou,Nirwan Ansari and Hong Ren,"A Genetic Algorithm For Multiprocessor Scheduling",1994,IEEE transactions on parallel and distributed processing,Vol.5,No.2,p113

[9]. Dr.Elgasim Elamin Elnima Ali,"A Proposed Genetic Algorithm Selection Method",2000

[10]. Saroj and Devraj,"A Non-Revisiting Genetic Algorithm For Optimizing Numeric Multidimensional Functions",2012,IJCSA,Vol.2,p-92

[11]. Md.Sakahawat Hossen,Fazle Rabbi and Md.Mainur Rahman."APSO For Multimodal Function Optimization",2009,IJET,Vol.1(3)

[12]. Chapter 09.03,Multidimensional Direct Search Method, *numerical**methods**.eng.usf.edu/.../mws_gen_opt_txt_multi**direct**.doc*

[13]. J.G.DIGALAKIS and K.G.MARGARITIS," On Benchmarkig For Genetic Algorithms ",2000,p-3

[14]. Salomon. R. (1995). ''Reevaluating Genetic Algorithm Performance under Coordinate Rotation of Benchmark Functions'', BioSystems vol. 39, pp. 263–278, Elsevier Science

[15]. Baeck, T. (1991). ''Optimization by Means of Genetic Algorithms''. In Internationales Wissenschaftliches Kolloquium (pp. 1–8). Germany: Technische Universit''

© 2010, IJARCS All Rights Reserved

[16]. De Jong, K. A. (1975). ''An analysis of the behaviour of a class of genetic adaptive systems'', University of Michigan, Ann Arbor. (University Microfilms No. 76-9381)

[17]. Goldberg, D. E. (1989). ''Genetic algorithms in search, optimization and machine learning'', New York: Addison-Wesley, pp. 40–45

© 2010, IJARCS All Rights Reserved