IJCSE International Journal of Computer Sciences and Engineering Open Access
Review Paper Volume-4, Special Issue-5, Jul 2016 E-ISSN: 2347-2693

Comparative Study of Simple GA & Hybrid GA for Basis Path Testing under Branch Distance Fitness Function

Mr. Deepak Garg and Dr. Pardeep Kumar

Department of Computer Science & Applications, Kurukshetra University, Kurukshetra-136119, India

Available online at: www.ijcseonline.org

Abstract-Test data generation is a key problem in software testing. Many automatic tools are already present but some are not optimal for large scale, some requires information of local or global solution of problem, some are not suitable to run time conditions. In this paper simple GA & hybrid GA have been implemented to produce automatic data set for testing under basis path testing criteria using branch distance based fitness function in MATLAB. Experimental comparison has been performed first up to twenty five iterations and second up to fifty iterations on same initial population set & then on randomly generated initial population set. After these comparisons conclusion has been made.

Keywords–Basis path coverage testing, Branch distance fitness function, Simple genetic algorithm, Hill climbing, Memetic genetic algorithm.

I. INTRODUCTION

Test data production in program testing is the job of recognizing a set of test data set, which fulfills the specified testing coverage criterion. Coverage criteria are used to check how well the program is exercised by a test. Approximately forty percent of the time and more than forty percent of the total cost in software development is caused by Manual software testing [1]. One of the type of coverage criterion is basis path based testing that can identify more than two third of errors/bugs/faults in the given program [1, 2] and many structural based test data production problems can be depicted into a path based test data production problem [3]. Branch coverage and path coverage are also covered by basis path coverage criterion. So, in this paper basis path based testing has been used as coverage criterion for program testing. However, doing this test data generation manually is a difficult task due to large number of predicated nodes, loops, infeasible paths in the program [4]. So, Simple GA and Hybrid GA have been used in this paper for generating test data set automatically [5, 6]. Furthermore related works [7, 8] show that the GA based test data set production outperforms other dynamic based methods and static based methods.

The rest of the paper has been organized as follows. In section 2, a brief introduction of simple GA is given. In section 3, Hybrid GA is given with sample program of hill climbing. In section 4, introduction of branch distance fitness function is given with its table. Section 5 includes experimental settings &section 6 includes results. Conclusions & future work has been given in the section 7.

II. SIMLPLE GENETIC ALGORITHM

It is direct, stochastic and parallel method for optimization and global search. It is used to find approximate or exact solution based on the principles of natural evolution, described by Charles Darwin [9]. It is a class of the evolutionary algorithms, because it is inspired from the biological evolution processes [10]. GA was firstly introduced in 1970s by J. Holland [11] & now used in solving search and optimization problems. GA works on a string of bits/digits called as chromosomes, each bit/digit that forms the chromosomes is called gene & a group of these chromosomes forms a population. Every chromosome has a fitness value which says the chances of survival of chromosome to the next generation set. Now the population is iteratively recombined and mutated to generate successive new populations till termination condition not satisfied.

III. HYBRID GENETIC ALGORITHM

Hybrid genetic algorithm is motivated by Dawkins notation of a meme. HybridGA is also named as memeticGA. Adding problem related local information/solution (Hill climbing, Gradient search) at any step of genetic functionin GA makes a hybrid GA [12, 13]. In the paper, hill climbing approach has been implementedas search guidance after selection method of GA.

Outline of the Hybrid GA

1. [INITIALIZATON] At first the population set is produced randomly generated of size N chromosomes.

2. [FITNESS FUNCTION] Compute fitness value F(X) for each chromosome X.
3. [NEW POPULATION SET] Until the new population is produced repeat following steps
(i) [SELECTION OPERATION] From the population set, select two parent chromosomes according to their fitnessvalue , let named as Parent1 and Parent 2

// Apply local search to each selected chromosome

Optimum1=hill_climbing (Parent 1)
Optimum2=hill_climbing (Parent 2)

(ii) [CROSSOVER OPERATION] Crossover the optimum chromosomes with a crossover probability Pc, to produce new offspring (exploitation).
(iii) [MUTATION OPERATION] Mutate new offspring with a mutation probability Pm (exploration).
(iv) [ACCEPT] Put new offspring in the new population.

4. [REPLACE POPULATION] By using some scheme, replace the old population with the new population.
5. [TERMINATION TEST] If the termination condition is fulfilled then stop and outputs the best solution in the current population set.
6. [LOOP AS FAILURE OF 5] Go to step 2 (as a failure of step 5).

B.Example program of Hill_climbing

```
function [ optimum ]=HILL_CLIMBING ( Parent)
fitnessOld = triangleidentifier ( Parent ); %Evaluating fitness value
neighbor = Optimum_neighbor ( Parent ); %Searching %optimum neighbor
fitnessNew = triangleidentifier ( neighbor );
    while ( fitnessNew > fitnessOld  &&  neighbor ( 1,1 ) < 4096 && neighbor ( 1,2 ) && neighbor ( 1,3 ) < 4096 )
        Parent = neighbor;
fitnessOld=fitnessNew;
neighbor=Optimum_neighbor (neighbor)
end
optimum = parent;
end
```

IV. BRANCH DISTANCE FITNESS FUNCTION

On the basis of expression in the branch (predicate node) and desired output (true, false), branch distance based functions are placed and aim is to minimize this branch distance based function value. The branch conditions are considered and computed based on the korel's table I. for branch distance based function [14]. It is used to differentiate among chromosomes/individuals that follow the identical target path [14]. Every predicate node or branch is consists of logical expressions. We have to alter or find or optimize the input data of that predicate node or branch to make branch (to be true or false) to track target path [13].

Table I: Korel's Branch Distance Based Function

Expression in branch X	F(X)=Branch Distance value If branch output=0=false	F(X)=Branch Distance value If branch output=1=true
p=q	-abs(p-q)	abs(p-q)
p~=q	abs(p-q)	-abs(p-q)
p>q	p-q	q-p
p>=q	p-q	q-p
p<q	q-p	p-q
p<=q	q-p	p-q
X1 OR X2	F(X1)+F(X2)	Min (F(X1), F(X2))
X1 ANDX2	Min (F(X1),F(X2))	F(X1)+F(X2)

V. EXPERIMENTAL SETTINGS

A. *Triangle Identification Program*

Triangle identification targets to find that if three input sides can form a triangle and what nature of triangle can be formed. Triangle identification program has been mostly utilized in the research area of software testing. Fig. 1 shows program's code in MATLAB. Fig. 4 shows flow graph of it.

```
function[]=triangleidentifier (p, q, r)
  if (p+q>r)&(q+r>p)&(r+p>q)&(p>0)&(q>0)&(r>0)
    if (p~=q)&(q~=r)&(r~=p)
      disp('Scalene Triangle');
    else
      if(p==q)&(q~=r)||(q==r)&(r~=p)||(r==p)&(p~=q)
        disp('Isosceles Triangle');
      else
        disp('Equilateral Triangle');
      end
    end
  else
    disp('Not A Triangle');
  end
end
```

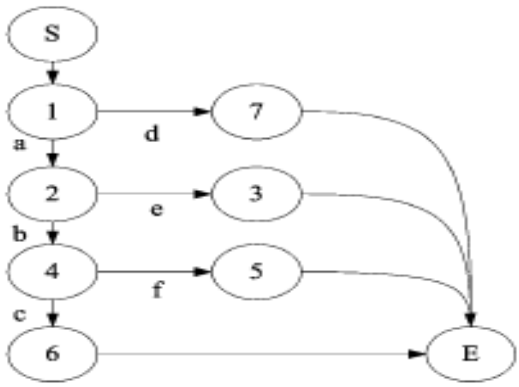

Fig. 1: Triangle Identifier ProgramFig. 2: Flow graph of the triangle Identifier

Target Path

Corresponding to Fig. 2 there are 4 linearly independent paths or basis set for triangle identifier program flow chart.
Path 1: d //Not a triangle; Path 2: a e //scalene triangle; Path 3: a b f //Isosceles triangle; Path 4: a b c //Equilateral triangle.
If every positive integer side is of 12 bits then corresponding to the probability, the probability of covering the Path4 is 2^{24} (that is $(2^{12} * 1 * 1) / (2^{12} * 2^{12} * 2^{12})$), which intends that it will take random testing 2^{24} tests to track path 4. That's why, Path 4: a b c is the most challenging path to be tracked in path based testing. Therefore, firstly the path 4: a b c is chosen as target path.

Fitness Function

According to Branch distance based fitness function and to track path 4: a b c for equilateral triangle, code of the adapted program (for better fitness function) of the triangle identifier is shown in fig. 3, taking individual sides as input and giving back its fitness (FIT). Fit is the fitness value, and motive is to minimize (optimize) value of Fit in Genetic Algorithm. BDBFF is array of size 3 calculating branch distance at 3 branch nodes.

```
function[ FIT ]=triangleidentifier (p, q, r)

  BDBFF(1)= - 2*(p+q+r);
  if (p+q>r)&(q+r>p)&(r+p>q)&(p>0)&(q>0)&(r>0)

    BDBFF(2)=min(min(abs(p-q),abs(q-r)),abs(r-p));
    if (p~=q)&(q~=r)&(r~=p)
      disp('Scalene Triangle');
    else

      BDBFF(3)= - abs(q-r) - abs(r-p) - abs(p-q);
      if(p==q)&(q~=r)||(q==r)&(r~=p)||(r==p)&(p~=q)
        disp('Isosceles Triangle');
      else
        disp('Equilateral Triangle');
      end
    end
```

Fig. 3: Instrumented program

Parameter Settings

Settings of Simple GA and Hybrid GA are as followings:

- Encoding: Binary string

- Chromosome Length: 12 bits*3=36 bits and every input sideis from 1 to 4096.
- Size of Population = 50
- Method of Selection: Tournament selection
- Two-point crossover probability (pc): 0.7
- Mutation probability (pm): 0.05
- Strategy of replacement: Steady state replacement
- Generation No.'s: 15

The first production of test data set was producedfrom their range randomly. For example, by running the code (Round (unifrnd (1,4096,50,3));) in MATLAB, produces 50 three dimensional vectors as the first production of test data set with values of each input variable's side is from 1 to 4096.

VI. EXPERIMENTAL RESULTS

For comparing the SGA and HGA for basis path testing under BDBFF, results were conducted with identical/same initial population in each of the 25 & 50 experiments and after that results were conducted with random initial population in each of the 25 & 50 experiments.

Path 3: a, b, f is near to the target path 4: a, b, c, so if no any method produces any test data relating to path4: a, b, c then an method who will produces more test data relating to Path3:a, b, f can come to the target path 4:a, b, c more keenly and can be said as better method as compared to others.

A. With Identical Initial Population

In this, Population was kept same in each of the two experiments of 25 iterations and 50 iterations. Result is shown in bar form for 25 iterations in fig. 4 and in numerical form in table II. For 50 iterations result is shown in fig. 5 in bar form and table III in numerical form.

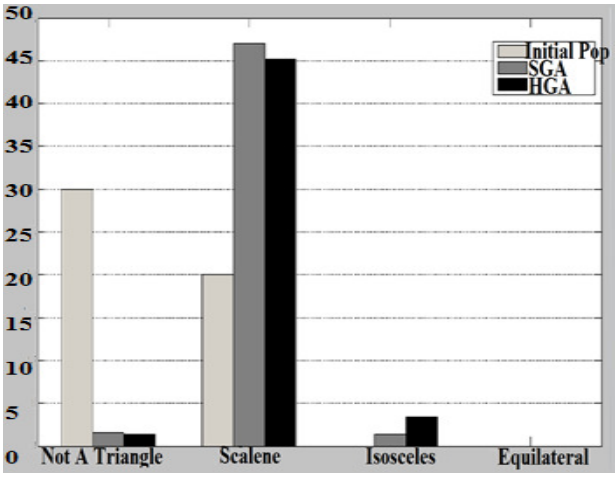

Fig. 4 Comparison after 25iterations on same population.

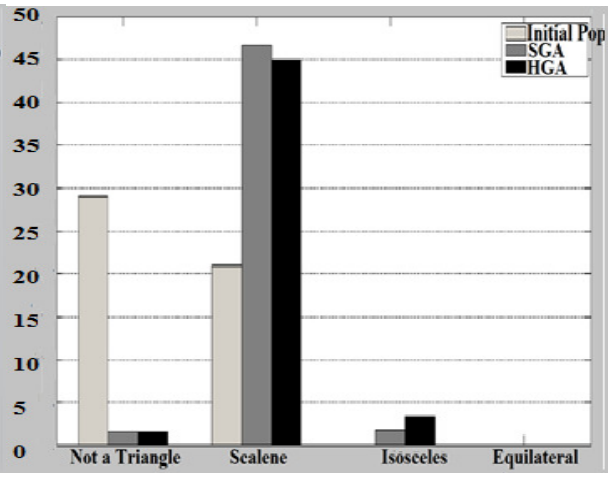

Fig. 5 Comparison after 50 iterations on same population

Table II. Numeric comparison after 25 iteraions

	Not-a-Triangle	Scalene Triangle	Isosceles Triangle	Equilateral Triangle
Same Pop 25	30	20	0	0
SGA with BDBFF	1.60	47.04	1.36	0
HGA with BDBFF	1.40	45.20	3.40	0

Table III. Numeric comparison after 50 iterations.

	Not-a-Triangle	Scalene Triangle	Isosceles Triangle	Equilateral Triangle
Same Pop 50	29	21	0	0
SGA with BDBFF	1.54	46.70	1.76	0
HGA with BDBFF	1.54	45.06	3.40	0

B. With Random Initial Population

In this also, Population was generated at random but was kept different in each of the two experiments of 25 iterations and 50 iterations. Result is shown in bar chart form for 25 iterations in fig. 6 and in numerical form in table IV. For 50 iterations result is shown in fig. 7 in bar form and table V in numerical form.

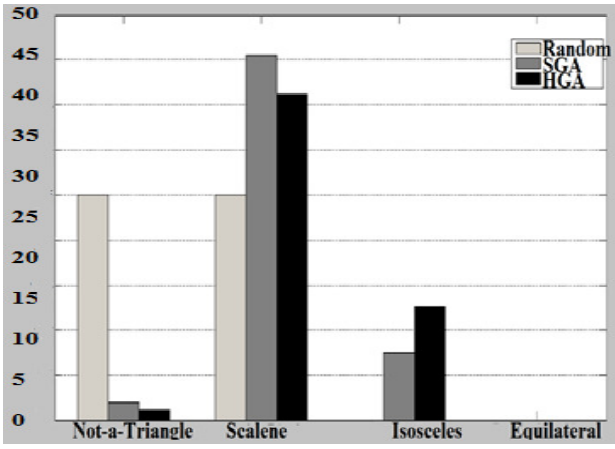

Fig. 6 Comparison after 25 itterations on diff. population

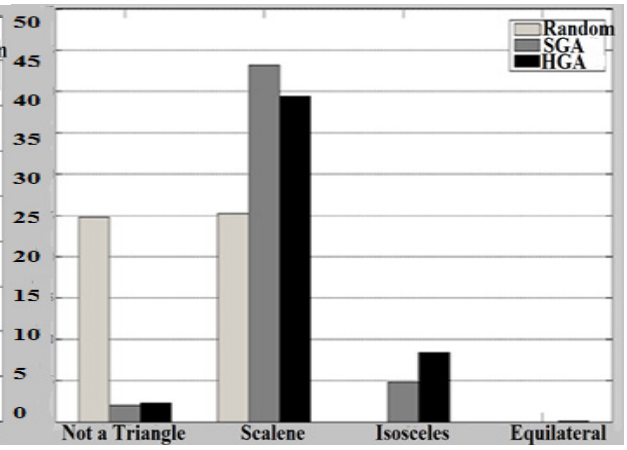

Fig. 7 Comparison after 50 iterations on diff. population

Table IV. Numeric comparison after 25 iterations.

	Not-a-Triangle	Scalene Triangle	Isosceles Triangle	Equilateral Triangle
Random Pop 25	24.96	25.04	0	0
SGA with BDBFF	2.00	40.48	7.52	0
HGA with BDBFF	1.16	36.20	12.64	0

Table V. Numeric comparison after 50 iterations

	Not-a-Triangle	Scalene Triangle	Isosceles Triangle	Equilateral Triangle
Random Pop 50	24.72	25.28	0	0
SGA with BDBFF	2.03	43.15	4.82	0
HGA with BDBFF	2.24	39.42	8.32	0.02

VII CONCLUSION

On the basis of Comparisons after twenty five and fifty experiments with same initial random population and with random initial population, two conclusions have been made. First conclusion is that Simple GA and Hybrid GA with branch distance fitness, both are better than random test data generation and second conclusion is Hybrid GA with branch distance fitness is better than Simple GA with branch distance fitness.

Future work will be to upgrade branch distance fitness function and to study the performance of Hybrid GA after applying hill climbing at different level of genetic operations.

REFERENCES

[1] B. Antonia. "Software Testing Research: Achievements, Challenges and Dreams".*Future of Software Engineering, IEEE Computer Society*, (2007): 85-103.

[2] B. W. Kernighan and P. J. Plauger."The Elements of Programming Style".*McGraw-Hill, Inc.New York, NY, USA* (1982).

[3] M. Alzabidi, A. Kumar and A. D. Shaligram. "Automatic Software Structure Testing by Using Evolutionary Algorithms for Test Data Generations".*IJCSNS: International Journal of Computer Science and Network Security* on 9, no. 4 (2009).

[4] D. Garg and P. Garg."Comparison of BDBFF & ALBFF for Basis Path Testing Using GA".*International Journal of Advanced Research in Computer Science and Software Engineering* on 5, no. 7 (2015).

[5] T. K. Wijayasiriwardhane, P. G. Wijayarathna and D. D. Karunarathna."An Automated Tool to Generate Test Cases for Performing Basis Path Testing".*Proc. International Conference on Advances in ICT for Emerging Regions, IEEE Computer Society* (2011): 95-101.

[6] G. L. Latiu, O. A. Cret and L. Vacariu. "Automatic Test Data Generation for Software Path Testing using Evolutionary Algorithms".*Proc. Third International Conference on Emerging Intelligent Data and Web Technologies, IEEE Computer Society* (2012): 1-8.

[7] G. M. C. Michael and M. Schatz. "Generating software test data by evolution".*IEEE Transactions on Software Engineering* on 27 (2001):1085-1110.

[8] W. Joachim and S. Harmen. "Suitability of Evolutionary Algorithms for Evolutionary Testing".*Proc. of the 26th Conf. on Prolonging Software Life: Development and Redevelopment, IEEE Computer Society* (2002).

[9] D. E. Goldberg. "Genetic Algorithms in Search Optimization and Machine Learning".*Addison Wesley Longman, Inc.,* ISBN 0-201- 15767-5 (1989).

[10] N. Singh and K. Aggarwal."Software Testing using Evolutionary approach".*International Journal of Scientific and Research Publications* on 3, no. 6 (2013).

[11] J. Holland. "Adaptation in Natural and Artificial Systems".*University of Michigan Press* (1975).

[12] P. Mascato and P. C. Cotta."A gentle introduction to memetic algorithms".*handbook of Metahuristics* (2003):105-144.

[13] D. Garg and P. Garg. "Basis Path Testing Using SGA & HGA with ExLB Fitness Function". *Elsevier Procedia Computer Science* on 70 (2015): 593-602.

[14] B. Korel. "Automated software test data generation". *IEEE Transactions on Software Engineering* on 16 (1990): 870-879.

