A Comparative Study on Genetic Algorithm and Ant Colony Algorithm for Testing S27 Benchmark Cyclic Sequential Circuits

J. Poornimasre, R. Harikumar, P. Saravanakumar

Abstract: *Testing is the procedure of evaluating the performance of the system with the intent to find whether the system meets its functional requirements or not. The testing procedure involves testing a predefined sets of input test data to the circuit under test (CUT) and determining the circuit responses. CUT that gives the exact output responses for all input stimuli that passes the test are said to be fault-free circuits. These failed circuits will give an exact response at any given point during the test sequence are assumed to be faulty. VLSI Testing will be done at different life cycle stages of a VLSI device, that includes the VLSI development process, the electronic system manufacturing process, and, in some cases, system-level operation. Several algorithm are exists to improve the test performance of a system also reduce the testing time. A performance analysis of Genetic algorithm and Ant colony algorithm has been carried out on S27 benchmark cyclic sequential circuits. Results shows that Genetic algorithm detects more faults with minimal number of test pattern than the Ant colony algorithm with less complexity.*

Index Terms: Test Vectors, Fault Detection, Controllability & Observability, S27 Benchmark Circuits, Cycle detection, Genetic algorithm.

I. INTRODUCTION

A system is said to be in failed state, if it deviates from its standard behavior. A fault on the system is different from failure and it is stated as a physical defect in a circuit. A fault is differentiated by its nature, value, extent, and duration of the response. The nature of a fault can be called as logical fault or non-logical fault. A logical fault is defined as the logic value becoming opposite to the given value at a specified point in a particular circuit. [1] The struck-at fault model is typical example for logical fault model. The stuck at fault affect the logical value of signal line irrespective of the signal line. The stuck at line may be any line from primary input to primary output. The struck-at fault converts the present logical value on the defected signal line, appears to be struck at a constant logic 0 or a constant logic 1, referred to as struck-at-0(SA0) or struck-at-1 (SA1), respectively. Let us consider an example given below, where the nine signal lines will point as potential fault sites and they are labelled alphabetically. There are 18 (2×9) possible faulty lines under the single-fault assumption.

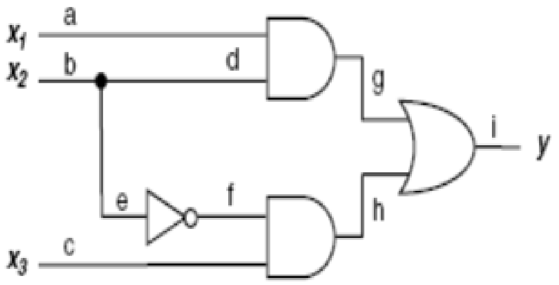


Fig:1 Example Circuit having 18 (2×9) possible faulty lines.

Below mentioned table1 will give the truth values for the fault-free circuit and the faulty circuits for all possible single stuck-at faults. Instead of assigning direct value to a logic 0 or logic 1, the struck-at-fault is determined by removing the source for the signal and assigned to a constant logic 0 or logic 1. This is shown in table 1, in which SA0 on fan out branch line ‘d’ behaves differently from SA0 on fan out branch line ‘e’, as the single SA0 fault on the fan out source line ‘b’ behaves as if both fan out branches line ‘d’ and line ‘e’ are SA0. All possible inputs and corresponding outputs is displayed in the truth table, in which the grey color highlighted denotes the faulty circuit producing an output response differ from the fault-free circuit. This makes the input values for the grey color highlighted truth table entries that represents a valid test vectors used to detect the associated struck-at fault. Other than line ‘d’ SA1, line ‘e’ SA0, and line ‘f’ SA1, all other remaining faults are found to be with two or more test vectors. Test vectors 011 and 100 must be included in any one set of test vectors, so that it will obtain 100% fault coverage in this circuit. These two test vectors will certainly detect a total of ten faults, and the left out eight faults are found with test vectors 001 and 110, making four test vectors obtaining 100% single struck-at fault coverage. Table 1 shows the different faults and their output logic for different test patterns. The a-Sao, d-Sao, g-Sao, i-Sao are detected by the test pattern 110 and 111.Those faults are called equivalent faults, either 110 or 111 used detect those four faults. Similarly fault collapsing method also used for test pattern reduction.

Manuscript published on 30 December 2018
* Correspondence Author (s)
Mrs.J.Poornimasre, Asst. Prof Level II, Dept. of Electronics & Communication Engineering, Bannari Amman Institute of Technology, Sathyamangalam, Tamilnadu, India (Email: poornimasrej@bitsathy.ac.in)
Dr.R.Harikumar, Professor, Dept. of Electronics & Communication Engineering, Bannari Amman Institute of Technology, Sathyamangalam, Tamilnadu, India (Email: harikumarr@bitsathy.ac.in)
Mr.P.Saravanakumar, Asst. Prof Level II, Dept. of Electronics & Communication Engineering, Bannari Amman Institute of Technology, Sathyamangalam, Tamilnadu, India
(Email: saravanakumarp@bitsathy.ac.in)

$x_1x_2x_3$	000	001	010	011	100	101	110	111
y	0	1	0	0	0	1	1	1
a SA0	0	1	0	0	0	1	0	0
a SA1	0	1	1	1	0	1	1	1
b SA0	0	1	0	1	0	1	0	1
b SA1	0	0	0	0	1	1	1	1
c SA0	0	0	0	0	0	0	1	1
c SA1	1	1	0	0	1	1	1	1
d SA0	0	1	0	0	0	1	0	0
d SA1	0	1	0	0	1	1	1	1
e SA0	0	1	0	1	0	1	1	1
e SA1	0	0	0	0	0	0	1	1
f SA0	0	0	0	0	0	0	1	1
f SA1	0	1	0	1	0	1	1	1
g SA0	0	1	0	0	0	1	0	0
g SA1	1	1	1	1	1	1	1	1
h SA0	0	0	0	0	0	0	1	1
h SA1	1	1	1	1	1	1	1	1
i SA0	0	0	0	0	0	0	0	0
i SA1	1	1	1	1	1	1	1	1

Table 1 Truth Table Values.

II. BACKGROUND

This section first reviews the background of Combinational SCOAP measures. Test Pattern generation for S27 cyclic sequential bench mark circuit using Ant colony algorithm presented in Section II-B. Genetic algorithm based test generation technique presented in Section III.

Combinational Scope measures

In Goldstein's method of calculating controllability, the initial step is setting the value for difficulty by resetting each primary input (PI) to 0 (called CC0) to the value 1 and each PI to 1 (called CC1) to the value 1. Then progress through the circuit in a forward pass, in level order so that the logic level of a gate is given by maximum of the distances (in logic gates) of its various inputs from its PI's.[3]

For finding the control over the logic gates in ascending level order number, processing of logic gates with input signal controllability (CC0 and CC1) have already been determined. For each logic gate traversed, add 1 to the controllability that accounts for the logic depth. In order to find the observations, a reverse pass is set starting from primary outputs (PO) moving back to the primary inputs (PIs). If a logic gate output is produced by setting only one input to a controlling value, then, Output controllability =min (input controllabilities +1). If a logic gate output can only be produced by setting all inputs to a non-controlling value, then, Output controllability =∑ (input controllabilities +1).

Ant colony algorithm

III. INTRODUCTION

The Ant Colony Optimization algorithm (ACO) is a probabilistic approach for solving any optimization problems that in turn is used to find the best possible paths. It is used to model the problem, in such a way each ant will provide a part of the solution, using a systematic decision that uses an initial information to trails. This information is updated based on a determined pheromone trails equation [2]. Lastly centralized actions that cannot be performed by single ants are applied. The steps followed in the test pattern generation are as follows, 1. Construction of a Graph 2. Test Data Generation 3. Pheromone Trail Updation and finally 4. Ant Colony Optimization Algorithm. The initial ant roams randomly till it locates the food source (F), then it returns back to its home (N), laying a pheromone trail. Like this other ants also lays different pheromone trail paths, following one of the paths randomly. The ants on the shortest path lay pheromone trails faster than longer path and make the shortest path stronger by making it more comfortable to the future ants [3].

Updation of the pheromone trail

The testability of a circuit under test circuit is a vital one and it has direct impact on testing of device. Controllability and observability are the two used for measuring the testability of circuits. Controllability defines, the difficulty of setting logic 0 or 1 in the circuit line and observability defined as difficulty of observing the signal line as 1 or 0. The ant colony algorithm uses these measures for test pattern generation. Initially S27 circuit is converted in to graph. Each node indicates the gate and edge indicates the connection. Ant colony algorithm use probability based techniques to traversal the next node. If a node has two outgoing edges then pheromone value calculated by the formulae ρ = ω1 / ω1+ω2 (Equation 1). Based on the intensity of pheromone, ant choose the next node for traversal. Initial pheromone value between two edges i and j are taken at time t as tij (t).The pheromone valued is maintained in the shorted path or the path followed by more ants. Updated pheromone on edge is calculated at time tij (t+n) = ρ (Tij (t)) + change in Tij, where ρ is a coefficient of the trail edge's probability between time t and t + n.At a node i, an ant k uses the pheromone trail for choosing j as the next node based on the following equation:

$$\Delta\tau_{ij}^{k} = \begin{cases} Q_{cc} \times L_{cc} + Q_{cc} \times L_{co} & \text{if the } k^{th} \text{ ant uses edge } (i,j) \text{ in its tour between time } t \text{ and } t+n \\ 1 & \text{otherwise} \end{cases} \quad (3)$$

$$\Delta\tau_{ij} = \sum_{k=1}^{m} \Delta\tau_{ij}^{k} \quad (4)$$

Since ant colony algorithm is a stochastic approach, the value of Q_{cc} and Q_{co} are chosen as less than 1 and generated with uniform distribution random numbers. In this paper, the value of Qcc and Q_{co} taken as 0.001.The parameter L_{cc} and

L_{co} are denotes the sum of the controllability and the observability values of the faulty path in the circuit. CC0, CC1 are the controllability of Sa0 and Sa1 repectively[5].

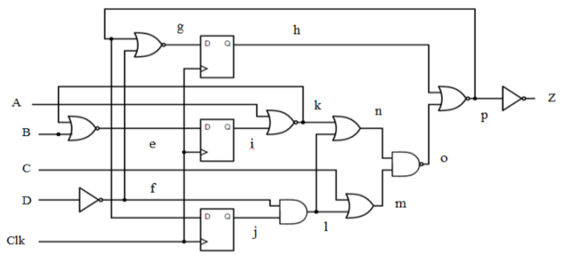


Fig. No: 2. S27 benchmark circuit

S27 benchmark is a cyclic circuit and it is converted into a acyclic circuit for calculating the controllability and observability which are the inputs to the ACO algorithm. The cyclic to acyclic conversions involves the following steps:

1. Circuit topology description
2. Cycle detection
3. Cyclic to acyclic conversion

IV. CYCLIC DETECTION AND ACYCLIC CONVERSION

The circuit C is represented as a directed graph G(V, E) | {V is the set of vertices of the graph E is the set of edges}. The CTG of S27 benchmark circuit where the primary inputs and outputs are not considered. The set of vertices (V) denotes the set of gates and the set of edges, E denotes the connections between the gates.

Cycle detection:

S27 Sequential circuit is converted in the form of graph. Node (v), Edge (E) represents the gate and connection between the gates respectively. If any loop presents in the graph it's difficult to propagate the fault to the output node. So an algorithm used to detect the loop in the graph. The main scope of algorithm is to find the feedback path R(s).It uses the GR algorithm as follows:

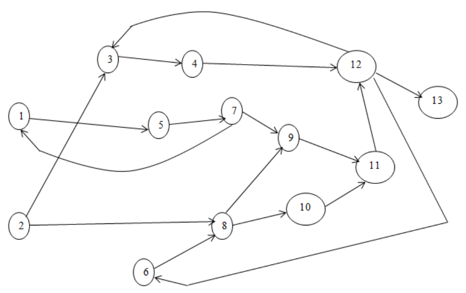


Fig. No: 3. Circuit topology graph of S27 benchmark circuit

Steps:
Step1: Initialise the S1, S2 node as 0
Step2: Remove the edge
Step3: Add the source node to S1 and sink node to S2
Step4: Repeat step 2 and 3 till the last node reached.
Step5: Concatenate S1 and S2
Step6: Find the feedback based repetition of node

With the help of vertices, the GR algorithm computes two sequences S1 and S2. Each vertex of the graph is removed and added to the sequence S1 or S2. If the removed node is a source or sink node, the corresponding node is added to the sequence S1 or S2 respectively. Following this rule of removing all the sink or source nodes one by one, the corresponding vertices are added to the sequence S1 in the decreasing order using dist(u)= dist + (u) – dist -(u). if G(V,E)=0, then single sequence S = S1 concatenated S2. For the Fig. No: 3, the degrees of the vertices are:

dist. (1) = dist. + (1) – dist.-(1) = 1-1 = 0
dist. (3) = dist. + (3) – dist.-(3) = 1-2 = -1
dist. (4) = dist. + (4) – dist.-(4) = 1-1 = 0
dist. (5) = dist. + (5) – dist.-(5) = 1-1 = 0
dist. (6) = dist. + (6) – dist.-(6) = 1-1 = 0
dist. (7) = dist. + (7) – dist.-(7) = 2-1 = 1
dist. (8) = dist. + (8) – dist.-(8) = 2-2 = 0
dist. (9) = dist. + (9) – dist.-(9) = 1-2 = -1
dist. (10) = dist. + (10) – dist.-(10) = 1-1 = 0
dist. (11) = dist. + (11) – dist.-(11) = 1-2 = -1
dist. (12) = dist. + (12) – dist.-(12) = 3-2 = 1

Here Node 2 is Sink node and Node 3 is Source node.

S1 = {2,7,12,1,4,5,6,8,10,3,9,11} & S2 = {13}, then concatenated single sequence S ={2,7,12,1,4,5,6,8,10,3,9,11,13}

Cyclic to Acyclic conversion

This conversion algorithm receives the number of the cycles from the previous algorithm.

The algorithm steps as follows:
Step 1: Read the circuit graph
Step 2: Get the number of cycles
Step 3: Break the cycle by remove the edge
Step 4: Label the broken node point
Step 5: Choose Bin as Broken input node and Bout as broken output node
Step 6: Label the forward path and cascade with the next path.
Step 7:Repeat the step 3 to step 6 till number of cycle becomes zero

Acyclic graph is constructed by attaching the forward path to bout node of the graph.. The graph is updated with the input or output edges and the forward path copies. In order to do this, I/O nodes connection from cyclic graph are compared with corresponding nodes in acyclic graph and the corresponding edges are updated. For example, if the input node is fed into the first node, then all the clone nodes are fed by the input node. In case of a output node, the clone node of the last forward cycle copy is fed to the output node leaving other clone nodes, ensuring only the final forward path of the output node is connected.

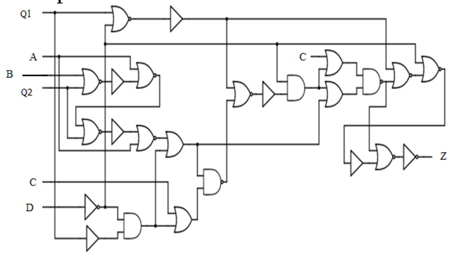
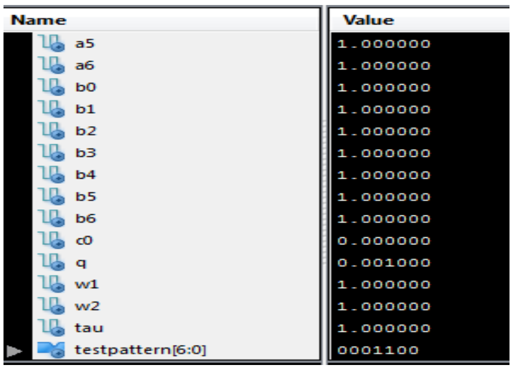


Fig. No: 4 Modified S27 Benchmark Circuit

The resulting acyclic graph contains additional copies of the original gates depending on the cycle copies. For example, if the circuit is assumed to have two cycle copies, the resulting acyclic circuit will have three forward path copies of the path in the corresponding cycle in the cyclic circuit. The number of copies will also depend on the number of cycles present in the original cyclic circuit. Thus the resulting graph forms an acyclic equivalent of a cyclic graph. By using the above procedure the S27 benchmark circuit is converted into acyclic circuit. The D flip flop is replaced by the buffer considering that clock is always in high condition as shown in figure 3.

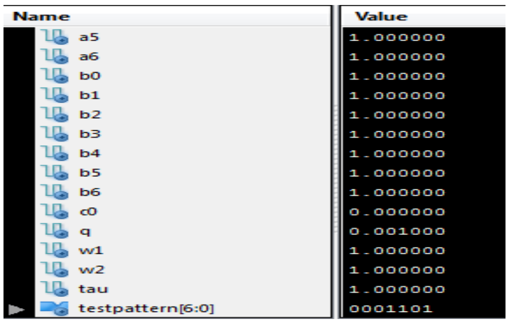
D. Simulation results

The test patterns for various faults are introduced by calculating the controllability and observability of the corresponding faults which act as an input to the ACO algorithm. The test patterns obtained for the faults (p – sa0) & (m – sa0) are shown below.



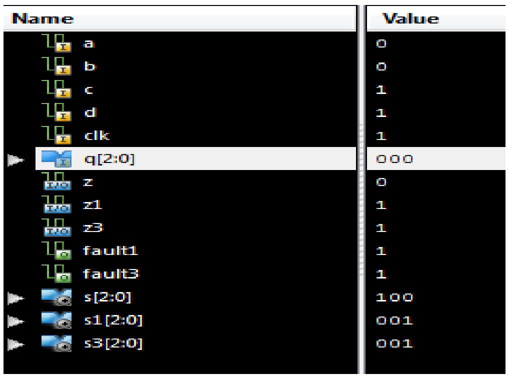
Name	Value
a5	1.000000
a6	1.000000
b0	1.000000
b1	1.000000
b2	1.000000
b3	1.000000
b4	1.000000
b5	1.000000
b6	1.000000
c0	0.000000
q	0.001000
w1	1.000000
w2	1.000000
tau	1.000000
testpattern[6:0]	0001100

The test patterns obtained for the faults (o – sa0) , (n – sa1) &(g – sa0) are shown below.



Name	Value
a5	1.000000
a6	1.000000
b0	1.000000
b1	1.000000
b2	1.000000
b3	1.000000
b4	1.000000
b5	1.000000
b6	1.000000
c0	0.000000
q	0.001000
w1	1.000000
w2	1.000000
tau	1.000000
testpattern[6:0]	0001101

The faults (p – sa0) & (m – sa0) are considered as fault1 and fault3 here.



Name	Value
a	0
b	0
c	1
d	1
clk	1
q[2:0]	000
z	0
z1	1
z3	1
fault1	1
fault3	1
s[2:0]	100
s1[2:0]	001
s3[2:0]	001

From the above figure it is clear that the primary output varies for the faults 1 and 3, when the pattern 0001100 obtained using ACO algorithm is applied to the S27 benchmark circuit under fault free and faulty condition. Since the output varies the faults 1 and 3 are detected. Simulation results shows that 70% of fault detected with the help of Ant colony algorithm.

Genetic algorithm

V. INTRODUCTION

Now a days, technology related to integrated circuits booming more and more. Most of current industries working with 5nm scale technology. Since the complexity of integrated circuit increases proportionally with technology scaling, the demand of quality based testing also indeed. The limited number of input/output pins represent a bottleneck for testing of complex embedded cores where apply large amounts of test patterns and test results between the automatic test equipment and the circuits under-test are required. Testing become more complex if number of inputs and outputs are increased also it increases the test cost and test time. Some cases, even human life span is not enough to test the single IC. So an optimization algorithm is needed to optimize the test pattern. Genetic algorithm plays a vital role to reduce the test time, test cost too. In this paper, genetic algorithm used to find the minimal test patterns for S27 bench mark cyclic sequential circuit. Genetic algorithm is most popular stochastic, random based technique and used in many applications. It follows the principle of natural evolution process and more suited for optimization based applications. Most of research has successfully carried out with the help of genetic algorithm. Since it is an evolutionary process, the data to be processed is represented in terms of binary strings also called Population. Each individual of population is called as chromosomes. The values of 0's and 1's in the chromosomes represents the features of individual string (chromosomes). The analogy of genetic algorithm mostly connected to the genes in biological organisms. The structural entity of each gene is completely independent of other genes. Genetic Algorithm is a non-deterministic random approach described by Goldberg [7] is to solve largescale combination optimization problem. Certain sequence of steps needed to solve a problem through genetic algorithm. The steps are

1. Choose the initial population
2. Representation of population individual by binary strings
3. Mathematical representation of optimization function(objective function or Fitness function)
4. Evaluate the fitness of individual
5. Applying genetic operators on selected individual

VI. POPULATION

The population is nothing but an input data. The range of input data may vary based on the constraints related to specific problems. For S27 circuit, the group of input combination taken as the initial population. All individual populations are taken as in the form of binary string (chromosomes.)

VII. FITNESS FUNCTION

It is necessary to derive the mathematical model to meet the objective of the given problem. The mathematical model of the specific objective is called as fitness function.[8] The objective function may be a maximization function or minimization function.

VIII. SELECTION

Once the fitness of individual is calculated, the best individuals are chosen from the initial population. The best individuals are used to produce the new population for next iteration. The selection process used to choose the best individuals. Rank and Roulette wheel selection approach are more popular. Here Roulette wheel method used for further reproduction.

IX. CROSS OVER

One best individuals are selected, genetic operations are applied on those individual to produce new parents. Cross over is one such operation, where genes (binary bits) are swapped to produce new parents. The most common crossover types [9].

1) One-point crossover
2) Two-point crossover
3) Uniform crossover

X. MUTATION

The next important genetic operator is Mutation. The process of inverting or flipping the bit/bits of individual is called mutation. Since it is a random process, mutation takes a random binary bit of individuals. The figure 4 illustrates the mutation operation of single bit of the individual. The mutation rate should be less than one. The new population will be produced after the mutation process. Again fitness function is evaluated on the individual population to find the best individual from the new population and crossover, mutation operator is applied for next generation of individuals. This process is repeated until reach the optimum result.

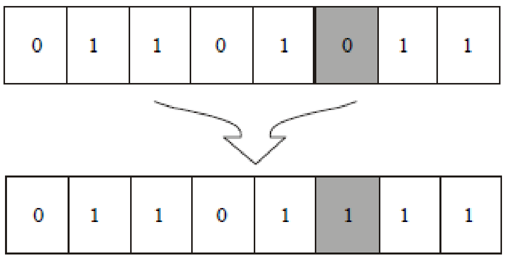


Figure4. Mutation

XI. TEST GENERATION:

A set of chromosomes (test patterns) is selected for initial population. The fitness value of each chromosome is evaluated to choose fittest individual for next generation. If the fitness function is low, then reproduction operations like mutation and crossover are applied directly on these test patterns to produce new offspring. [10].

The Fitness function for the S27 sequential circuit chosen as

F = (FD * Nf) / (FT * Ng)
F = Fitness Function
FD = Total number of Fault detected
Nf = Number of FF used
Ng = Number of logic gate used
Ft = Total number of fault injected

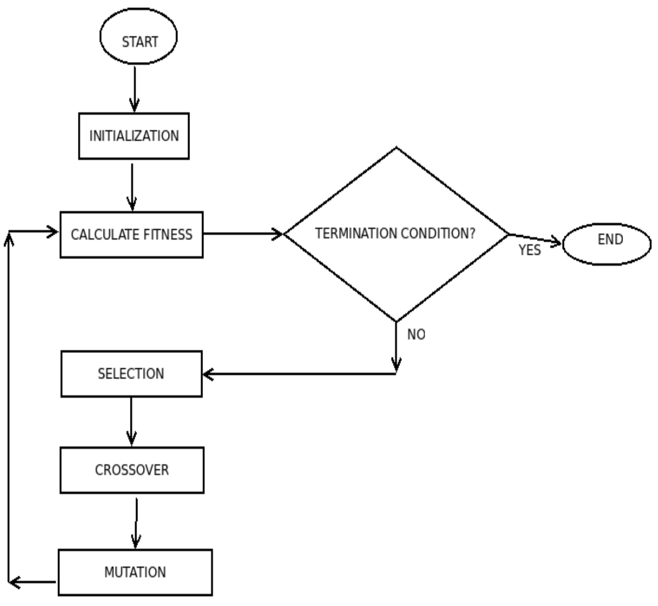
Fitness values are calculated for total population of 128 patterns. After first iteration, following new chromosomes are generated and their corresponding fitness values are calculated. Depends upon their highest fitness values, chromosomes are selected for the next operation of crossover and mutation operation. To generate the test, the circuit has been divided to sub circuits. By using XILINX ISim simulatorVHDL code is developed for test pattern generation and cross-over and mutation functions and test patterns applied for cyclic sequential (S-27) circuits. Almost 80%of the fault has been detected by using Genetic algorithm.

CHROMOSOMES	FITNESS
0000000	0.270
0010000	0.270
0110000	0.270
0111000	0.270
0010100	0.270
0011100	0.270
0110100	0.270
0111100	0.270

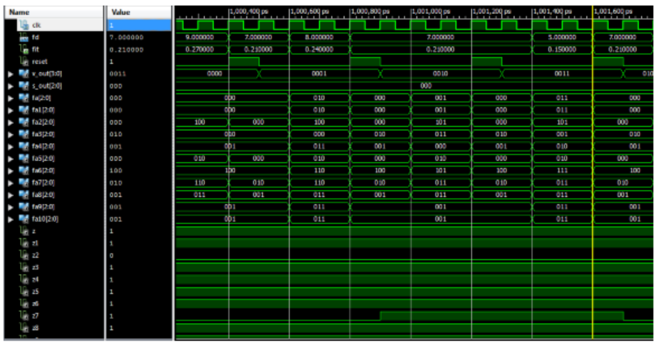
Figure 5. Chromosome Fitness value

XII. SIMULATION RESULTS

Steps followed in genetic algorithm are
Step 1: Choose the initial population
Step 2: After the injection of stuck-at faults in S-27 circuit, calculate the fittest individual chromosome from the population.
Step 3: Roulette wheel selection of best individual
Step 4: Select two fittest chromosomes from the population
Step 5: Cross over with a cross over probability, intersect the parents to form new offspring (children).
Step 6: Inverting the random bits as per the mutation probability
Step 7: Generate the new population.
Step 8: Use new produced population fitness calculation.
Step 9: Repeat from step 2 to 8 till stopping criteria met.



Output for fitness values of various chromosomes and their primary and secondary outputs are shown below



XIII. CONCLUSION

Testing of sequential circuits is a complex task. The basic S27 benchmark sequential circuit has taken to compare the performance of Genetic algorithm and Ant colony algorithm. The probabilistic approach of Ant colony algorithm uses controllability and observability to propagate the fault. . In some case the fault is not propagated to either the primary or secondary output. This is because of reconvergent fan out the controllability and observability values not able to propagate the fault. In future these values can be improved in such a way that the fault is propagated to the output. The complexity of Ant colony algorithm is very high than the Genetic algorithm. Simulation results shows that Genetic algorithm detects more fault than Ant colony algorithm with less complexity.

REFERENCES

1. Book: M. Bushnell, V. Agrawal: Essentials of Electronic Testing for Digital, Memory and Mixed Signal VLSI Circuits, Published by Kluwer academic press, 2000.
2. Rana Farah and Haidar M. Harmanani, “An Ant colony optimization approach for test pattern generation”, Computer Science and Mathematics Department, American University Byblos, 2010
3. Kelson Gent and Michael S. Hsiao, “Dual-Purpose Mixed-Level Test Generation Using Swarm Intelligence”, 2014 IEEE 23rdAsian Test Symposium.
4. M. Dorigo and T. St¨utzle, Ant Colony Optimization, MIT Press, 2004.
5. Kewen Li, Zilu Zhang, Wenying Liu, “Automatic Test Data Generation Based On Ant Colony Optimization”, 2009 Fifth International Conference on Natural Computation.
6. H. Harmanani and B. Karablieh, “A Hybrid Distributed Test Generation Method Using Deterministic and Genetic Algorithms,” in Proc. 5thInt. Workshop on System on-Chip for Real-Time Applications, pp. 317- 322, 2005.
7. Goldberg, D.E,”Genetic algorithms in search, optimization and Machine Learning. Reading”, MA: Addison-Wesley.1989.
8. l.Pomeranz, SudhakarM.Reddy, "On improving genetic optimization based test generation, " Proceedings of European design and test conference, 1997.
9. Yung-Chieh Lin, Kwang Ting Cheng, "Multiple-fault diagnosis based on single fault activation and single output observation, " Proceedings of Design, Automation and Test, 2006.
10. H. Takahashi, K. O. Boateng, K. K. Saluja, Y. Takamatsu, "On diagnosing multiple stuck at faults using multiple and single fault simulation in combinational circuits, " EEE Transactions on Computer Aided Design of Integrated Circuits and Systems,VoL21,no. 3, pp. 362-368, 2002.