Multi-Objective Genetic Algorithm for FPGA Testing

Vinay Chopra* and Dr. Amardeep Singh**
A.P. (Department of Computer Science)*
DAV Institute of Engg & Tech.
Jalandhar, India
Professor (Department of Computer Science)**
UCoE Punjabi University.
Patiala, India
(Received 10 March 2013 Accepted 14 March 2013)

Abstract: This paper gives a concise introduction to Multi-Objective Genetic Algorithms and FPGAs and it reveals that how Automatic Test Pattern Generation method can be formulated in terms of CNF form which in turn used to generate test patterns using Multi-Objective Genetic Algorithm. By applying a Multi-Objective Genetic Algorithm on this CNF form, it has been observed from the experiments that as the problem size are increased by increasing number of variables and clauses, the fault coverage ratio increases. It also has been shown that Multiobjective Genetic Algorithm can be applied to variety of SAT instances of FPGA test pattern generation problem, which gives competitive results.

Keywords: FPGAs, CNF, MOGA, DAG, SAT, ATPG, PODEM, FAN

1. Introduction to FPGA Testing
FPGA testing or ATPG[1] is a major research topic which is used in semiconductor electrical testing in which the vectors or input patterns are required to check a device for faults that are automatically generated by a program. The input vector patterns are sequentially applied to the device under test and its corresponding response is compared with the expected response from a good circuit. If any error occurs it means that the circuit is faulty. The effectiveness of the ATPG is considered largely by the fault coverage achieved and the cost of conducting the test. Mainly three types of approaches have been recommended for sequential circuits The Topological Approach [2, 3], Symbolic Approach [4] and the Simulation based Approach [5, 6]. Automatic Test Pattern Generation (ATPG) consist of two distinct phases, firstly a creation of the test vector and secondly an application of the test vector. In the first phase, suitable models for the FPGA circuit are developed at gate or transistor level in a way so that the output responses of a faulty device for a given set of inputs will be different from those of good devices [7]. An efficient test vector is which make an efficient use of memory space, time taken during fault coverage and minimum set of test vectors must generated to detect all the important faults of a device [8]. The parameters that should be kept in mind during the design of test vector suit are the time needed to construct the minimal test set, size of the pattern generator( hardware/software system needed to properly stimulate the devices under test),size of the testing process itself, time needed to load the test patterns ,external equipment etc. Automatic Test Pattern Generation (ATPG) methods that are mostly used include the D-Algorithm [9], the PODEM [10], and the FAN [11].

2. Introduction to Multi-objective Genetic Algorithm
Multi-objective optimization problems (MOPs) [12, 13] are common Optimization Problem. A general MOP includes a set of parameters (decision variables), a set of k objective functions, and a set of m constraints. Objective functions and constraints are functions of the decision variables.
Multi-Objective Genetic Algorithms (GA) imitate the biological processes fundamental classic Darwinian evolution for finding solutions to optimization or classification
problems. The implementation is based on finding a population of candidate solutions (or chromosomes) which is evaluated using a fitness function and ranked. From the ranking, candidates are selected from which the next generation is created [14]. The fitness function is a measure of how well a candidate solves the problem. Implementations vary in the choice and practice of the selection method; i.e. the purpose of the selection method is to choose candidates whose Multiobjective Genetic mix will tend to lead to improved candidate solutions in the next generation. The common selection methods are random, Elitist [14], Roulette Wheel Tournament [15], etc. Multiobjective genetic operators provide mixing of chromosome portions from the parent or parents to form the offspring of the next generation. Examples of Multiobjective Genetic operators include crossover, mutation, inversion, etc [16]. The process repeats until either the number of iterations is exceeded or an acceptable solution is found. A generic view of a GA includes:

a) Initialization of the initial population.
b) Ranking is determined on the evaluation of the fitness functions on each chromosome.
c) Mating rights are determined using application of the selection method on the population.
d) Application of the multi-objective genetic operators on the chromosomes selected for mating.
e) Return to Step #2.

The following flowchart shows the main phases used to simulate the working of Multi-objective Genetic Algorithm [17].

---

3. Applying Genetic Algorithm to FPGA Test Pattern Generation

For applying Genetic Algorithm to FPGA circuit, it is first converted to CNF form. For a given gate, a CNF formula \( \varphi \) is a set of clauses and is represented by conjunction of the CNF formulas of each gate. One of the most illustrious applications of SAT is FPGA routing [18], logic synthesis [19], Automatic Test Pattern Generation (ATPG) [20], testing include delay fault testing and redundancy identification and elimination, functional vector generation. To generate a test pattern for a single fault.

a) First extract a formula that defines the set of test patterns that detect the fault.
b) Then use a Boolean satisfiability algorithm to satisfy the formula.

To extract a formula, a directed acyclic graph [21] is constructed as follows that represents the topological description of the circuit:

a) The nodes of the graph are circuit inputs, outputs, gates, and fan-out points and the edges of the graph are circuit lines i.e. wires.
b) The sources of the graph are circuit outputs
c) And the sinks of the graph are the circuit inputs.

Every edge has an associated variable. Fig. 1.20 shows an example circuit and its associated DAG [22].

---

**Fig 1:** Flow Chart of Multi-objective genetic Algorithm

**Fig 2:** Example Circuit [22]
3.1 SAT Representation for FPGA Testing

An individual is represented for a SAT instance with variables is a string of bits where each variable is associated to one bit [21]. The search space consist of the set $S = \{0,1\}^n$ (i.e. all the possible strings $n$ of bits) and an individual $X_i$ clearly corresponds to an assignment $X_i$ represents the truth value of the $i$th atom and $X_i \leftarrow \alpha$ represents an individual $X_i$ where the $i$th atom has been set to the value $\alpha$. Given an individual $X_i$ and a clause $c$, we use $\text{SAT}(X, c)$ to denote the fact that the assignment associated $X_i$ satisfies the clause $c$.

3.2 Fitness Function

Given a formula $\phi$ and an individual $X$, the fitness of $X$ is defined to be the number of clauses which are not satisfied by $X$ [119].

$$\text{eval}: S \rightarrow N$$

$$\text{eval} (X) = \text{card} \{ c \mid \neg \text{sat}(X, c) \land c \in \phi \}$$

Where card represents the cardinality of a set. This fitness function will be used in the selection process and stimulate an order on the population [124].

3.3 Selection Process

The selection process [23] takes as input a given population and extracts some individual’s assignments according to that selection criterion. The selected individuals are the elected as parents for the evolution process and evolve by crossover operations. It is necessary to keep some diversity in the population to insure an efficient search so that if the selected parents are too close, some region of the search space will not be explored. This diversity of the selected population is achieved by introducing the notion of hamming distance between strings of bits. This distance gives the number of different bits between two strings and can be defined here as:

$$\text{ham}: S \times S \rightarrow N$$

$$\text{ham} (X, Y) = \text{card} \{ i \mid X_i \neq Y_i \}$$

Therefore the function is defined as select:

$$\text{select}: 2S \times N \times N \rightarrow 2S$$

such that select $(P, n, d)$ is the set of the $n$ best $X$ in $P$ according to eval and such that $\forall X, Y \in \text{select} (P, n, d)$, $\text{ham}(X, Y) \geq d$. typically, the first generation consists of randomly generated individuals. Next generation individuals are selected on the basis of their fitness values i.e. the more the individual is fit the more it has probability to come into the next generation. Then do crossover operation between the two individuals of the chosen ones. In this work two point and three point crossover is used to obtain the next 10 individuals. Fitness proportionate selection also known as roulette wheel selection is a genetic operator used in Genetic Algorithm for selecting potentially useful solutions for recombination.

In this method fitness function assigns fitness to possible solutions or chromosomes which is used to associate a probability of selection with each individual chromosome. While candidate solution with a higher fitness will be less likely to be eliminated, there is still a chance that they may be.
roulette wheel with each slice proportional in size to the fitness, [see table 1] shows the selection probability for 11 individuals, linear ranking and selective pressure of 2 together with the fitness value. Individual 1 is the fit individual and occupies the largest interval, whereas individual 10 as the second least fit individual has the smallest interval on the line. Individual 11, the least fit interval, has a fitness value of 0 and get no chance for reproduction.

Table 1: Selection probability and fitness value

<table>
<thead>
<tr>
<th>Number of individual</th>
<th>Fitness Value</th>
<th>Selection Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2.0</td>
<td>0.18</td>
</tr>
<tr>
<td>2</td>
<td>1.8</td>
<td>0.16</td>
</tr>
<tr>
<td>3</td>
<td>1.6</td>
<td>0.15</td>
</tr>
<tr>
<td>4</td>
<td>1.4</td>
<td>0.13</td>
</tr>
<tr>
<td>5</td>
<td>1.2</td>
<td>0.11</td>
</tr>
<tr>
<td>6</td>
<td>1.0</td>
<td>0.09</td>
</tr>
<tr>
<td>7</td>
<td>0.8</td>
<td>0.07</td>
</tr>
<tr>
<td>8</td>
<td>0.6</td>
<td>0.06</td>
</tr>
<tr>
<td>9</td>
<td>0.4</td>
<td>0.05</td>
</tr>
<tr>
<td>10</td>
<td>0.2</td>
<td>0.02</td>
</tr>
<tr>
<td>11</td>
<td>0.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

For selecting the mating population the appropriate number of uniformly distributed random numbers (uniform distributed between 0.0 and 1.0) is independently generated [24].

Fig 4: Roulette wheel selection method [25]

After selection the mating population consists of the individuals: 1, 2, and 3,5,6,9. The roulette-wheel selection algorithm provides a zero bias but does not guarantee minimum spread.

4. Conclusion

This work has revealed that FPGA testing in the form of Boolean SAT, which is a NP complete problem can be proficiently solved using the Multi-Objective Genetic Algorithm by first converting it into the Boolean SAT and representing it as a constraint satisfaction problem. This simulation work reads the BLIF files representing the Xilinx MCNC Benchmark circuits as word file and dynamically circuit graphs are created for above used circuits. Then these graphs are used for FPGA circuit testing using GA procedure and results have shown that as the problem size are increased by increasing number of variables and clauses, the fault coverage ratio increases. Also as the numbers of generations are increased, fault coverage ratio and CPU time to test the circuit for each FPGA circuit has improved.

5. References


2) Jie Qin “A Brief Introduction to Application-Dependent FPGA Testing” Dept. of Electrical and Computer Engineering 200 Broun Hall, Auburn University, AL 36849-5201.


20) Fadi A. Aloul and Assim Sahyoon “SAT-Based Techniques in Test Vectors...


