Performance Evaluation of Two Allocation Schemes for Combinatorial Group Testing Fault Isolation

Rawad N. Al-Haddad, Carthik A. Sharma, and Ronald F. DeMara

School of Electrical Engineering and Computer Science
University of Central Florida
Orlando, FL 32816-2450
demara@mail.ucf.edu

Abstract

Two fault isolation approaches based on Combinatorial Groups Testing (CGT) are presented. Although they both share the basic principle of grouping suspect resources into subgroups for testing, they differ in the allocation strategy used to achieve that grouping. Equal share CGT approach is analyzed first, results shown successful fault isolation in 85% of the runs. An average of 3.65 groups and 362 test vectors were needed to isolate a single fault with an average of 2.95 discrepant outputs in each run. After that, Interleaved CGT approach is proposed along with analytical demonstration of its strengths and weaknesses. Finally, Future research framework is suggested to experimentally compare the two approaches.

1 Introduction

Adaptive fault detection and isolation methods for *Field Programmable Gate Arrays (FPGAs)* have been growing in importance especially in mission-critical domains. The design of dependable systems optimized for availability, reliability, and performance is crucial for applications that require fast recovery with minimal human intervention. For example, deep space satellites such as Stardust utilize over 100 FPGA devices to support multi-purpose tasks [1] and need to operate in various environmental extremes such as high radiation bursts and thermal fatigue conditions. Under such circumstances, the need for an efficient fault detection and isolation scheme becomes inevitable.

While most conventional approaches like redundancy and BIST provide fault detection and isolation capabilities, they suffer from the following two major limitations:

I. Offline isolation: In which the hardware has to be taken off service during fault isolation. This implies suspending the functional throughput during the isolation process, which in turn reduces device availability for mission-related tasks.
II. Exhaustive testing: By stress-testing individual resources with non-functional test vectors. Input/output pairs must be generated beforehand in such a way that guarantees best coverage of all resources under test. Fault detection latency might be increased by extensive testing thus degrading device reliability.

This paper presents a fault isolation approach based on CGT algorithm that mitigates the shortcomings of the conventional approaches. Availability is maximized by keeping the devices online during the fault isolation process. In addition, the usage of functional dataflow inputs makes it possible to detect and isolate faults without altering the normal throughput of the device, hence increasing device reliability.

Combinatorial group testing was first used during World War II to isolate a number of defective blood samples out of millions of samples under test. These types of applications, where a small set of defective individuals are to be isolated from a large population, are typical candidates for CGT approaches. The algorithm relies on the combinatorial aspects of the problem [2] by performing tests on groups of individuals at each step of the solution. The presented approach has shown that faulty resources in FPGA device can be isolated with the aid of CGT based approach.

The solution starts by initially considering each resource as suspect, and then proceeds by narrowing the suspect pool with each group of testing, until eventually isolating the single faulty resource. Two alternative versions of the method are presented:

I. Equal Share Allocation: In which the suspected resources are divided into equal subsets, each assigned to one individual in the population, meaning that each suspected resource is exclusively covered by exactly one individual.
II. Interleaved Allocation: where the suspect resource subsets are shared among individuals. The sharing degree is controlled by a *Coverage Factor (CF)* which

determines how many individual utilize each resource in the suspect pool.

The remainder of the paper is organized as follows: Section 2 provides details about the equal share allocation scheme along with experimental analysis of its performance. Section 3 proposes the new interleaved allocation scheme. Section 4 concludes the paper by comparing the two method in term of efficiency and best/worst case scenarios and suggests a direction for future research.

2 Equal Share Allocation Scheme

The Equal Share Allocation Scheme divides the resource suspects pool into equal subgroups (shares) and assigns each share to a different individual in the population. Therefore, each *Look Up Table (LUT)* in the suspect pool is said to be "exclusively covered" by one individual. An important note here is that an LUT can be utilized by many individuals in different ways, this means one physical resource having many logical realizations. This scheme however opts to map each suspect LUT to exactly one individual in the population preset. The selection of which individual to configure on the device in order to apply test vectors is done randomly, for a population of *R* individuals, each will have an equal probability to be picked and tested.

2.1 Allocation Strategy

Assuming a suspect pool of *N* LUTs, Equal share scheme divides the *N* LUTs equally on *R* individuals (Population preset). Each individual will comprise *M* suspect resources, where $M = N/R$. After this allocation is accomplished, CGT can start by applying test vectors to each individual and comparing the output to a predetermined set of correct outputs. If a discrepancy is detected in the output of one individual, this means under a single-fault assumption that the faulty LUT is utilized by that individual. The suspect pool can drastically drop from *N* LUTs to *M* LUTs. New groups can then be formed by dividing the new suspect pool with *M* resources into *R* individuals again, and so on until the single faulty LUT is isolated.

On the other hand, no reduction in the suspects pool can be gained if no discrepancy is detected. This case can happen under two possible scenarios, when the faulty resource is not utilized in such a way to signify erroneous output in the individual (falling into a redundant path or unused logic), or when the applied test vectors do not articulate the fault which may happen since testing is driven by the functional input vectors.

Figure 1 shows *N* suspect LUTs divided among a population of *R* individuals, where each individual exclusively covers *M* suspect LUTs.

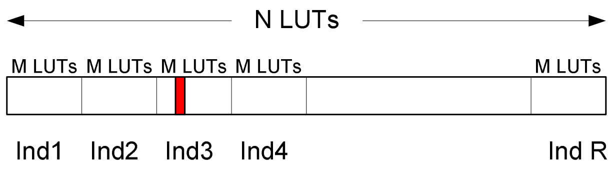

Figure 1: Equal Share Allocation Scheme

If the faulty LUT is used exclusively by Ind3 as shown in Figure 1 (denoted by the highlighted rectangle), and the applied test vectors articulated the existence of the fault, the current group can be terminated and new one can be formed with a new suspects pool consisting of all the *M* LUTs used by Ind3.

Based on the previous discussion, it is apparent that this allocation scheme can have two contrasting effects on the performance of the CGT:

- Maximal possible gain if the fault is articulated by the test vectors. This case is what makes the technique very efficient since the suspect list is reduced by a factor of *R* (from *N* suspect resources to *M*), in the best case leading to the formation of only $Log_R N$ groups.
- Minimal possible gain if the fault is not articulated by the test vector. In this case, the algorithm will finish testing all individuals in the population without detecting and discrepancy, and thus new groups must be formed out of the same *N* suspected LUTs, without achieving any gain from the previous stage.

The best case for this allocation scheme is to reduce the suspect list by a factor of *R* for each testing group, while the worst occurs when the fault is not articulated and the suspect list remains unchanged.

2.2 Experimental Setup

The circuit used to evaluate the Equal share CGT approach is a DES-56 encryption circuit. The circuit was implemented using VHDL. Xilinx ISE design tools were used to place and route the design, targeting Virtex II Pro FPGA device.

The *Fault Injection and Analysis Toolkit (FIAT)* [3], which has been developed for conducting fault isolation experiments, was used to model faults and evaluate testing results in order to assess the performance of the CGT approach. FIAT is a set of *Application Programmer Interfaces (APIs)* implemented using Python language to interact with the Xilinx ISE tools to inject and evaluate faults, instead of leaving this burdensome task to the developer himself. Fault injection is achieved by editing the design file rather than the configuration bitstreams. In addition, the tool exposes methods for editing *User Constraint Files (UCF)*, these files are used by Xilinx tools to set the physical constraint of the design, such as LUT locations, physical design dimensions, and most importantly, whether a specific LUT can be used in the logical-physical mapping of the design or not. These features tune the experiment completely for the Xilinx Virtex II Pro FPGA device and minimize the effort needed when the approach is to be implemented on the real device. Taking the physical aspect of the device into consideration is what makes this research more valuable in the field of reconfigurable hardware rather than being a theoretical research for a mathematical algorithm such as CGT.

The experiment addressed population size tweaking and its effect on the performance of the CGT. Three population sizes were tried: 15, 20, and 25. Each individual required 304 LUTs to be implemented on a Virtex II Pro FPGA device. The area under test was square-shaped with dimensions of 25X25 LUTs. Thus, a total of 625 LUTs were available to place and route the design, making the resource utilization approximately 50% (304 LUTs out of 625 available).

2.3 Results

Three experiments were conducted with population sizes of 15, 20, and 25 individuals. The ability of the CGT algorithm to isolate a single stuck-at fault was measured for the different three population sizes. Each experiment featured 20 different runs with the same experimental settings, this is mandated by the fact that CGT is probabilistic by nature and multiple runs are needed to verify its behavior. **Table 1** shows the experimental results for the three runs, The following four findings can be extracted by studying the table to analyze the impact of population size on the performance of the Equal share CGT scheme:

- Isolation ability: Equal share CGT was able to isolate the single fault in 85% of the experiments. Varying the population size did not affect the ability of the CGT to isolate the fault at the end. However, it might be possible that the reason is the narrow adjustment of the population size in each consecutive experiment. Further exploration of this aspect is encouraged in future work to cover particularly high and low population sizes.
- Average number of groups: The selection of the population size was crucial at one point to reduce the minimum possible number of groups needed to isolate the fault. This case is clear in the 15-individual population size as no solution could be achieved in less than four groups, while it became possible when the population size was increased to 20 individuals. On the other hand, increasing the population size from 20 to 25 did not contribute to reducing the minimum number of groups. These results are evident from the way in which the equal share approach allocates suspect LUTs into available individuals. Four groups represents a lower bound for the best case solution in a population size of 15 individuals. This bound becomes three in the 20 and 25 individuals case. **Figure 2** shows a bar graph depicting the total number of runs for each group count.

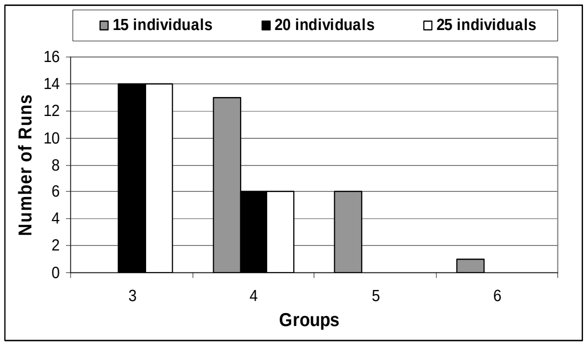

Figure 2: Total number of runs for each group count

It can be noticed from **Figure 2** that no runs in the 15-individual case has achieved a solution in three groups, most of the runs ended with four and five groups, only one run required six groups. On the other hand, the 20 and 25-individual case achieved isolation with four runs at maximum, where three groups being the frequent case. The CGT was able to isolate the single stuck-at fault in an average of 4.35 groups for the 15-individual case, this number was reduced to 3.3 for the 20 and 25 individuals cases. The fact that both cases required the same number of groups precludes the hypothesis that increasing the number of physical implementations is always beneficial to enhance the performance of the CGT.

- Average number of test vectors: The average number of test vectors needed in each experiment is proportional to the population size. The algorithm divides the suspect list into all available individuals and apply test vectors to each physical realization of the application. Consequently, more individuals imply more test vectors. This is due to the fact that the algorithm needs to apply more test vectors before giving up and declaring the fault as unarticulated. **Figure 3** shows the number of test vectors required in all runs for the three population sizes.

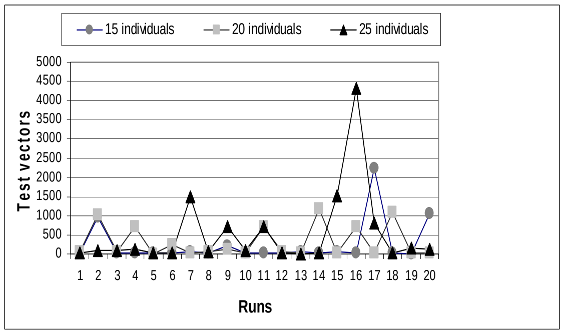

Figure 3: Number of test vectors required in each run.

Table 1: Results of three CGT experiments with different population size

Population	Isolation results		Number of groups						Required Test vectors	Discrepancies
	Success	Fail	3	4	5	6	Mean	SD		
15	17	3	0	13	6	1	4.35	0.587	247.4	3.7
20	17	3	14	6	0	0	3.3	0.470	311.9	2.55
25	17	3	14	6	0	0	3.3	0.470	525.3	2.6

One run with 25 individuals required a total number of 4,321 test vectors until the fault was acknowledged as unarticulated, only one discrepant output appeared in this run making the system highly reliable although the CGT failed to isolate it. It can be suggested that the error hit a rarely used LUT so that 1 out of 4,000 functional test vectors produced incorrect output. No relation can be drawn between the number of test vectors and the exact time-unit latency unless the rate of input/output evaluation is known. Nevertheless, the number of test vectors can give an adequate measure about the relative fault isolation latency in this case.

- Discrepancies: This column in Table I represents the average number of discrepant outputs in each experiment. Note that this number is upper-bounded by the number of groups required to isolate the fault as no suspect list reduction can happen without receiving one discrepant output. The number of discrepant output quantifies the reliability of the system because it tells what percentage of the time the system was able to respond with good output to some input. For example, a run with 100 applied test vectors and 4 discrepancies indicates that the system responded correctly to 96% of the input and yet was able to isolate the fault. **Figure 4** shows the number of discrepant outputs detected in each run for the three population sizes.

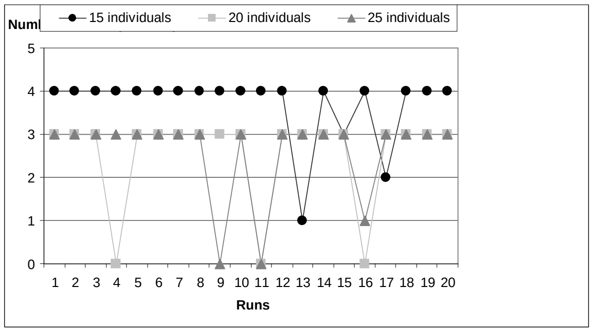

Figure 4: Number of discrepant outputs in each run

Although Zero discrepant outputs are marked as failed runs, the fact that the reliability was 100% in these runs makes it clear that the fault occurred in unused or redundant LUT in all physical configurations of the circuit under test, alleviating the fact that CGT failed to isolate the fault.

3 Interleaved Allocation Scheme

In this scheme, each LUT in the suspect pool is used by more than one individual in the population. This represents multiple logical mappings for the same physical resource by realizing the same functionality with different implementations. The fact that each suspect LUT is utilized by more than one individual means that the suspect resource groups used by individuals have to be interleaved.

The rationale behind adopting this scheme is to reduce the chance that a faulty LUT is not articulated by test vectors within a specific configuration. If the probability that a faulty LUT is not articulated by one configuration is *P*, then the probability that it is not articulated by two independent configurations becomes *P*P*. This improvement is attained at the expense of the maximal isolation progress rate achieved in the equal share approach. As the suspect list is shared among more than one individuals, a discrepant output will not reduce the number of suspects by as great a factor as the equal share approach.

3.1 Allocation Strategy

The Coverage Factor (CF) indicates the number of individuals covering each resource in the suspects pool. For example, *CF = 2* means that each suspected LUT is covered by two different individuals. In order to match this constraint, the subgroups of suspects used by different individuals have to interleave. Figure 5 shows an example of the Interleaved allocation scheme with CF = 2 and population size = 5.

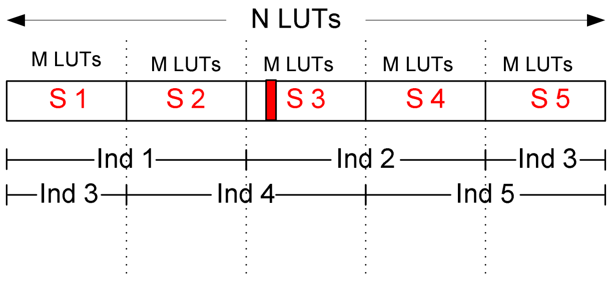

Figure 5: Interleaved Allocation scheme with CF=2

In the interleaved allocation scheme, the *N* LUTs have to be divided into *M* subgroups where *M* = *N/R*, and each individual will utilize $2\times M$ LUTs in order to meet the coverage factor requirement *CF*=2. In this case, a discrepancy will reduce the number of suspects to *2M* rather than *M* as in the equal share approach. However, a two-pass special technique can be used to improve the performance of the Interleaved approach while maintaining a low probability that a fault LUT is not articulated in case fault occurs. The two passes are described below:

Pass one: In this pass, the algorithm tries to reduce the suspect list from *N* to $CF \times N/R$ where *CF* is the coverage factor. In the *CF*=2 example shown in Figure 5, the target is to reduce the suspect list from *N* LUTs to *2M* LUTs, this can happen as early as one discrepant output is detected. The advantage here is that there is less probability that a fault in an LUT is not articulated by a test vector as there are two individuals utilizing each suspect LUT. For instance, if the highlighted LUT in Figure 5 is the faulty one, two individuals (Ind2 and Ind4) have a chance to produce discrepant output, making it a better choice in this aspect over the equal share approach in which each LUT is only utilized by one individual.

Pass two: The goal of this pass is to reduce the size of the suspect list from 2M to M, and thus achieving the same maximal reduction attained in the Equal share approach. In order to achieve this target efficiently, a new data structure called *Interleaved Individuals Set (IIS)* is introduced to the CGT program to keep track of the interleaved individuals in a specific CGT configuration. **Figure 6** shows the IIS for the configuration of Figure 5 with CF=2 and R=5.

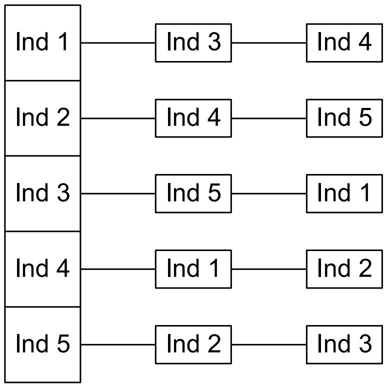

Figure 6: Interleaved Individual Set for configuration in Figure 5.

This simple data structure can be constructed at the beginning of each group when the individuals are configured and the suspect list is distributed among them. The purpose of the IIS is to improve pass 2 by expediting the process of narrowing the suspect list from *2M* to *M*. This can be achieved by testing only those individuals that were interleaved with the one that showed a discrepancy in Pass 1. In a single fault model, once an individual shows a discrepant output, only the individuals which share the same physical set of LUTs with it can articulate the fault. The algorithm can proceed and test these two individuals, and once a discrepancy is shown the suspect list can be reduced further to *M*.

4 Conclusion

The equal share approach relies on a high-risk high-gain technique, in which each resource in the suspect list is utilized by one individual. In such case, a discrepancy will reduce the suspect list from *M* to *M/R* where *R* is the population size, achieving maximal reduction in the suspect list. This high-gain is achieved at the expense of a higher probability of unarticulated faults as each faulty LUT is utilized by one individual only. Groups containing individuals with unarticulated faults lead to no improvement in suspect isolation.

On the other hand, the interleaved allocation scheme adopts a low-risk approach by covering each suspected resource with two or more configurations, making it less probable that a group of testing yields no improvement in fault isolation. Yet, it still can achieve the same fault isolation improvement as the equal share approach by utilizing a design-time IIS data structure to keep track of the individuals that should be tested whenever a discrepancy is detected.

It is also important to mention here that the equal share approach is a special case of the interleaved approach in which CF=1, this might be useful to design one generic model for both approaches that can be controlled by setting CF to 1 (equal share) or more (interleaved approach). This represents our first recommendation for future work in this domain. In addition, population size effect should be studied more to explore the effect of considerably high and low population sizes on the performance of both allocation schemes. Finally, Interleaved approach should be investigated more for different coverage factors to see if it really adds more robustness to the conventional equal share CGT approach.

5 References

[1]. Sharma, C. A. and R. F. DeMara (2006), “A Combinatorial Group Testing Method for FPGA Fault Location,” in *Proceedings of the International Conference on Advances in Computer Science and Technology (ACST 2006), Puerto Vallarta, Mexico,* 2006.

[2]. Du D and Hwang, F. K (2000), "Combinatorial Group Testing and its Applications," *Series on Applied Mathematics volume 12,* World Scientific.

[3]. Sharma, C. A. (2007), "FPGA Fault Injection and Analysis Toolkit (FIAT)."