

# Consensus-Based Evolvable Hardware for Sustainable Fault-Handling

Ronald F. DeMara, *Senior Member, IEEE*, Kening Zhang, *Student Member, IEEE*, and  
Carthik A. Sharma, *Student Member, IEEE*

School of Electrical Engineering and Computer Science  
University of Central Florida  
Orlando, FL 32816-2362  
demara@mail.ucf.edu

## Abstract

An autonomous recovery approach directed towards improving the availability of reconfigurable systems is developed using *Consensus-Based Evaluation (CBE)*. Under the proposed CBE technique, a diverse population of functionally identical, yet physically distinct SRAM-based Field Programmable Gate Array (FPGA) configurations is generated at design time. At run-time, these alternative configurations are ranked using a fitness function which evaluates an individual's discrepant behavior relative to the consensus formed by the population. Hence, any physical resource exhibiting an operationally-significant fault automatically decreases the fitness of those configurations which use it. Superior individuals are preferred for selection, so the fault becomes occluded from subsequent FPGA operations. Meanwhile, offspring formed through crossover and mutation of faulty and viable configurations are selected at a controlled *re-introduction* rate. Refurbishments are evolved in-situ, with online real-time input-based performance evaluation, enhancing system availability and sustainability. Experiments demonstrate refurbishment via evolution of a completely functional configuration even when all spares are impacted by a fault, yet without needing to utilize any context-specific fitness function.

*Index Terms* - Autonomous Fault Handling, Reconfigurable FPGA Devices, Online Evaluation, Outlier Identification, Genetic Operators for Regeneration.

## 1 Introduction

Evolutionary techniques have the potential to restore mission-critical functionality in SRAM-based reprogrammable devices such as *Field Programmable Gate Arrays (FPGAs)*. Evolution might provide an alternative to device redundancy for dealing with permanent degradation due to radiation-induced stuck-at-faults, thermal fatigue, oxide breakdown, electromigration, other local permanent damage, and transient faults. Potential benefits include recovery without the increased weight and size normally associated with spares. Also, failures need not be precisely diagnosed due to intrinsic evaluation of the function on the faulty device. Hence, recent research has focused on employing the reconfigurability inherent in *Evolvable Hardware (EHW)* techniques [Torresen02] [Gallagher03] to increase reliability, availability and autonomy [Keymeulen00] [Lohn03a] [DeMara05a] [Moore05] [Zhang05]. In this

context, the potential of using intrinsic reconfiguration for fault recovery has been also assessed with respect to reconfiguration latency and real-time operating constraints [Greenwood05].

Traditional solutions rely on non-evolutionary fault handling schemes such as *Triple Modular Redundancy (TMR)* [Von Neumann56] and *N-Modular Redundancy (NMR)* [Kastensmidt05] [Lombardi01]. These methods adapted to FPGA devices rely on static spatial redundancy to increase reliability. For example in a TMR system, three identical modules evaluate the same operands simultaneously. Outputs are then assessed using a voter or comparator to ascertain the majority output which is then propagated as the validated output value. In [Vigander01], TMR is enhanced with population-based *Genetic Algorithm (GA)* to restore complete functionality among partially regenerated faulty FPGA configurations using a voting arrangement. NMR consists of  $N$  ( $N$  is odd) modules and an  $N$ -input comparator which can tolerate failures in up to  $(N-1)/2$  modules. This extension of TMR to handle more than one faulty module can provide improved reliability and availability. However, as  $N$  increases, hardware resources, power consumption, and detection logic requirements to implement such *spatial voting* approaches grow linearly.

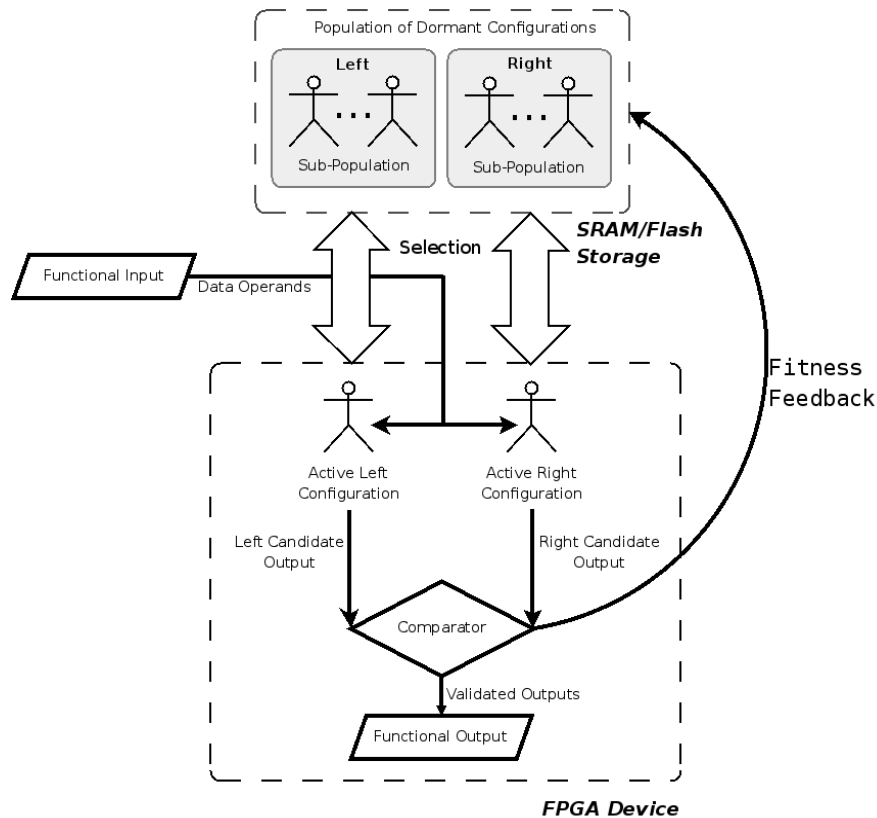


Figure 1: The Dynamic NMR Model ( $N=2$ ) used in Consensus Based Evaluation

On the other hand as shown in Figure 1, the proposed CBE approach realizes a *temporal voting* approach to *N-Modular Redundancy*. The alternative modules comprise a population of  $N$  competing FPGA configurations which are stored in off-chip memory. Without loss of generality, an  $N=2$  Duplex

scheme and an  $N=3$  TMR scheme are used to illustrate the concept and evaluate CBE with respect to an established scheme. In the *Duplex Mode* shown in Figure 1, where  $N=2$ , exactly two *active* configurations are selected from the population and loaded into the FPGA while others remain *dormant*. These two active individuals are denoted as the *Function Logic L* and the *Function Logic R*, for *Left* and *Right*, respectively. Discrepant behavior that arises during comparison of the candidate outputs is used to validate outputs and modify the relative fitness parameters of competing individuals. Discrepancies also generate fault alerts that trigger selection of alternate individuals from the dormant set. Over a period of time, accumulated discrepancy results provide fitness information to categorize individuals into subsets based on their relative reliability, including a subset of failed individuals requiring refurbishment. GA-based recovery can then gradually improve the fitness of these individuals with minimal impact on system availability by performing computations primarily with the high fitness configurations.

In this paper, we develop and evaluate an approach to manage a population of alternate designs to sustain high availability. This is achieved using a novel fitness assessment strategy of consensus that emerges from a population of diverse alternatives. Viability is demonstrated by means of fault detection and recovery experiments involving four circuits from the MCNC91 benchmark suite. Results indicate that in the majority of experimental runs conducted, a complete recovery was achieved if one was physically possible.

## 2 Related Work

Previous work on evolution-based regeneration has emphasized the use of GAs as a search mechanism driven by exhaustive test vectors. This sought to produce a single individual with the global maximal fitness determined via exhaustive evaluation of test inputs. When realized either at design-time or run-time, each implies a range of capabilities and costs as described below.

### 2.1 Offline Refurbishment Using Adaptive Regeneration

Techniques such as TMR increase reliability using physical redundancy with the option to replace voted components at a module-level granularity, but do not inherently utilize the reconfigurability available in gate array devices within each module. *Adaptive Regeneration* has been investigated as an alternative to utilizing pre-determined spare modules. For instance, Vigander's [Vigander01] regeneration approach extends TMR to utilize faulty FPGAs that have been partially regenerated using a population-based GA which utilizes *inter-modular crossover*, and *intra-modular mutation* operations. He demonstrates that FPGA-based implementations of 4-bit×4-bit multipliers can be automatically reconfigured to realize partial refurbishment. Each of the partially refurbished multipliers is deficient with respect to only selected input vectors. The approach takes advantage of this operational diversity

by using a voting arrangement of partially refurbished devices that exhibits the complete functionality. He concludes that realizing complete refurbishment is difficult, but diverse partial recoveries can be readily found using GAs in certain combinational logic circuits.

Lohn, Larchev, and DeMara [Lohn03b] develop a bit-string representation along with mutation and two-point crossover operators for actively refurbishing FPGA interconnect as well as logic resources in a sequential circuit. Their approach regenerated a quadrature decoder circuit on a Xilinx SRAM-based Virtex XCV1000 FPGA demonstrating resolution of a stuck-at-fault on the input of a *Configurable Logic Block (CLB)*. Their GA synthesizes a new alternative configuration in the presence of a resource fault given a population of 40 initial configurations after a few hundred generations. Their GA was shown to be capable of recycling faulty components: partially-damaged CLBs were sometimes reassigned by the GA to alternative functions based on the failed parts' residual functionality. While Lohn, Larchev, and DeMara's approach demonstrated complete regeneration for a small-sized state machine, the refurbishment was performed offline and required exhaustive fitness test vectors.

Even for moderately-sized circuits, partial recoveries are often the best attainable, especially within real-time constraints [Greenwood05]. However, since solution quality asymptotically approaches complete refurbishment, the proposed CBE method is shown to maintain high system availability throughout majority of the refurbishment process. CBE successfully leverages partially-fit individuals in the population to realize one or more fully-fit individuals using evolution directed by consensus among the partially-fit members of the population. As shown in Section 3, CBE accomplishes this by using a relative fitness metric that identifies individuals that have the best fitness over a window of recently observed runtime inputs. Additionally, a robust set of partially-fit configurations is created that lends itself towards a sustainable system in the face of pervasive faults.

## 2.2 Improving Fault Tolerance at Design Time

*Design-time* fault tolerance approaches improve reliability by anticipating classes of faults and providing methods to address them effectively. The resources provided at design time might be redundant spares, as in Lach's deterministic approach [Lach98]. This approach segments the FPGA into static tiles at design time with a known functionality, some redundant resources, and a pre-designed alternate configuration. Spare tiles can be selected when needed, but their functionality is predetermined and thus limited. An alternative approach by Keymeulen, Stoica, and Zebulum [Keymeulen00] increases fault-insensitivity by developing two varieties of fitness evaluation methods in a GA-based paradigm. They develop *population-based* and *fitness-based* paradigms to design circuits that are more likely to remain functional even in presence of a wide range of faults. Their population-based fault tolerant design method evolves diverse circuits and then selects the most fault-insensitive individual after

evolution is terminated. This approach, like many others, does not inherently differentiate between transient and permanent faults. Although Keymeulen *et al* successfully utilize a population-based GA to improve fault tolerance, the technique is not capable of leveraging the diverse information contained in the evolved population once the single best-fit individual has been selected for use as the active configuration.

## 2.3 Fault Recovery Characteristics

Fault recovery characteristics of representative techniques are listed in Table I and described below. They vary with respect to their fault detection, diagnosis, and recovery methods. Most tend to focus on a single phase of the fault-handling process. Few mechanisms provide integrated fault-handling and a runtime fitness evaluation based recovery mechanism. These features are often accompanied by either higher operational costs, increased latency, or both. These characteristics motivate the proposed CBE scheme which provides a regeneration scheme with an integrated fault handling mechanism. Vigander's and Lohn's methods exhibit likelihood of recovery related to the FPGA's *Design Complexity*. In other words, they attempt to design an original refurbishment where only a single failed configuration is available for adaptation. On the other hand, the proposed CBE approach draws upon a diverse population to benefit from alternative configurations that are still operational. Ideally, recovery would be performed with the faulty device remaining online in a degraded capacity producing outputs which are unaffected by the fault, as demonstrated by CBE in Section 4.

Table I: Related Work - Recovery Characteristics of FPGA Fault-Handling Schemes

Approach	Partial Online Recovery	Basis for Recovery	Quality of Recovery	Availability	Externally-supplied Elements	Resource Recycling	Pre-determined Limits
TMR	Depends on spare modules	Requires 2 datapaths are operational	Either complete or none	100% for single fault, 0% thereafter	2 of 3 Majority Voter	No	Single datapath
[Vigander01]	No	Design complexity	Based on single individual	Non-deterministic	GA Controller, function test vectors	Yes	None
[Lohn, Larchev, DeMara03a]	No	Design complexity	Based on single individual	Non-deterministic	GA Controller, function test vectors	Yes	None
[Lach98]	No	Available spares	Either complete or none	Either complete or none	Device test vectors and controller	No	Only one faulty CLB per tile
[Keymeulen, Stoica, Zebulum00]	No	Depends on characteristics at design time	Based on single individual	Non-deterministic	None at runtime	No	Depends on redundancy during design
<i>CBE</i> ( <i>proposed herein</i> )	Yes	<i>Recovery complexity</i>	<i>Based on diverse population</i>	<i>Adaptable</i>	<i>CBE controller. RAM coverage is intrinsic. No test vectors.</i>	Yes	<i>None</i>

While the quality of recovery under evolutionary approaches cannot be guaranteed, static redundancy approaches like Lach’s are either completely recoverable or completely beyond recovery. STARS’ quality of recovery is restricted by a non-adaptive routing scheme that relies on worst-case clock dilation. As listed in Table I, active recovery approaches, such as CBE, can provide resource recycling. With regards to pre-determined recovery limits, only dynamic competitive approaches are restriction-free. Optimally, detection latency is confined to first erroneous output. Among techniques with optimal detection latency, power consumption ranges from  $2n+r$  under the proposed approach to  $3n$  under TMR, where  $n$  and  $r$  denote the number of computations during normal throughput and the refurbishment cost respectively. Significantly, unlike CBE other approaches may allow propagation of many erroneous outputs prior to detection. As with previous approaches, if a failure in the system makes reconfiguration impossible, then recovery by reconfiguration will not be possible. Also, catastrophic failures in either the critical Input-Output data paths, or the processing resources used to implement the evolutionary algorithm will adversely affect device recovery capability.

### 3 Consensus-Based Evaluation Paradigm

The proposed CBE scheme realizes regeneration by integrating all phases of fault handling within an evolutionary algorithm process flow. It employs population diversity information, partially online recovery of failed resources, and resource recycling with adaptable overheads. Two innovations are realized for self-adaptive EHW regeneration: elimination of additional test vectors and temporal assessment based on relative fitness assessment.

#### 3.1 Detecting Faults using a Population of Alternatives

CBE detects and classifies faults using a *temporal voting* approach. In the Duplex mode, the outputs of two competing active L and R *half-configurations* shown in Figure 1, are compared to detect discrepancies. Alternative pairings are considered over time to provide the robust consensus described below. Each individual in the population is represented as a *configuration bitstream* [Xilinx05] that defines the physical resources it uses and their interconnections when it is loaded onto the FPGA. An initial population of known-good individuals is created at design-time. These primordial configurations are functionally-identical, yet they utilize physically-distinct resources by having alternative design or place-and-route implementations. In the Duplex Mode, two of these competing half-configurations are instantiated on the reconfigurable FPGA device by downloading their configuration bit streams. This realizes a conventional *Concurrent Error Detection (CED)* [Mitra00] arrangement to detect at least any single resource fault with certainty. As in traditional CED approaches, comparison of the outputs of the two resident half-configurations will produce either discrepant or matching outputs to indicate the

presence or absence of faulty resources in the utilized FPGA hardware [DeMara05b]. Maintaining exclusive resource utilization for half-configurations belonging to either half ensures that under a single fault assumption, the presence of a fault implies the fault-free nature of all the half-configurations designed for the other half. An additional advantage of using pre-designed configurations is that system downtime is reduced to a minimum as potentially viable alternatives are available. Also, the use of  $L$  and  $R$  half-configurations enables the use of runtime reconfiguration technology to reconfigure a portion of the device without taking other portions offline.

The CBE process is described below using Duplex Mode depicted in Figure 1. After the device is configured with the competing configurations, the same input vector is applied to both of the functionally-equivalent logic instances. Fault detection is accomplished when there is a disparity between the outputs of the active configurations, as ascertained by the discrepancy detector. The presence or absence of discrepancy is used to adjust the *Discrepancy Values (DVs)* of both individuals without rendering any judgment at that time as to which individual is actually faulty. Succeeding pairings of alternate combinations identify those individual(s) that utilize faulty physical resources through consensus formation. Meanwhile, the fault-free configurations become exonerated over time. This is because the  $DV$  of a faulty configuration always increases regardless of its pairing, yet the  $DV$  of fault-free half-configurations which are paired together are not increased. This *temporal* testing scheme enables the use of pseudo-exhaustive testing over a period of time without the reduced availability imposed by exhaustive testing.

### 3.2 Tracking Competence using Fitness States

Competition among a diverse pool of individuals can generate robust information about their relative competence and reliability. In particular, the fitness states and health transitions of competing FPGA half-configurations during online operation are depicted in Figure 2. At any instant, each individual configuration is labeled with one of four states  $\{Pristine (C_P), Suspect (C_S), Under Repair (C_U), Refurbished (C_R)\}$  as governed by the transitions indicated by the numbered arcs in Figure 2. Initially, all of the individuals in the population begin in the *Pristine* state.

If output discrepancies are detected among the half-configurations in the FPGA then the competing  $L$  and  $R$  half-configurations undergo indicated health state transitions. A comparison can lead to one of two results, " $L=R$ " or " $L \neq R$ ." When  $L=R$  occurs, both individuals retain their *Pristine* state, as shown by transition event "1". However, when their outputs disagree, then transition "2" occurs whereby both of the configurations are demoted to the *Suspect* pool and their  $DV$  is increased. The determination of a configuration's fitness state for subsequent transitions is based on its cumulative  $DV$  relative to  $DV$  of the other individuals in the population evaluated over an *Evaluation Window*, denoted by  $E$ .

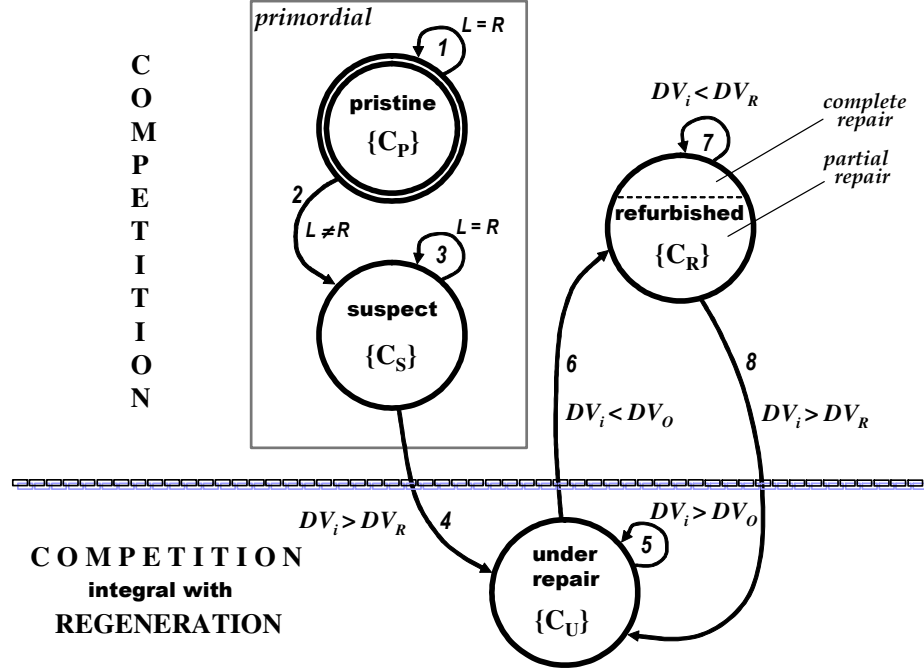


Figure 2: States in the Lifetime of the  $i^{\text{th}}$  Half-Configuration

The period  $E$  defines a fixed number of evaluations at the end of which an individual's fitness state is updated depending on its observed discrepancy history. Only after an individual has undergone such testing is its fitness state updated. The *reintroduction rate*, denoted by  $\lambda_R$ , controls the rate at which individuals are rotated for instantiation on the FPGA. By varying  $\lambda_R$ , a tradeoff between the throughput and the rate of refurbishment can be obtained. In particular, the re-introduction rate denotes the probability that an instantiated functional configuration is replaced by another from the competing pool, regardless of whether it has completed its evaluation window, or exhibits a discrepancy. Higher throughput and availability can be ensured via a low reintroduction rate which will maintain individuals that perform well on the FPGA for the length of their evaluation window, at the cost of slower refurbishment of the individuals undergoing refurbishment. Individuals that have been instantiated on the FPGA are replaced in one of three ways. They will be replaced when they articulate a discrepancy, when they have completed their evaluation window, or as dictated by the reintroduction rate.

The  $i^{\text{th}}$  half-configuration is marked as *Under Repair* if its  $DV$  increases beyond the *repair threshold* denoted by  $DV_R$  as shown in transition 4 in Figure 2.  $DV_R$  is determined by the relative fitness of the operational elements among the population, i.e. those in the *Pristine*, *Suspect* and *Refurbished* states. After successive evolutionary refurbishment operations, if an *Under Repair* individual's  $DV$  returns to the range of the outlier threshold value  $DV_o$  as a consequence of transition 6, then the configuration is *Refurbished*. Over a period of time, the  $DV$  of an individual could approach zero achieving complete

regeneration. Without exhaustive testing however, it is not possible to completely distinguish partial regeneration from complete regeneration. Competing half-configurations remain *Refurbished* unless their  $DV$  rises above the *Repair threshold*  $DV_R$ , at which time they are again demoted to the *Under Repair* state.  $DV_O$  is lower than  $DV_R$  to ensure that only individuals with  $DV$  significantly lower than  $DV_R$  are recognized as refurbished enough to be operational.

### 3.3 Self-Adaptive Fitness Assessment using Outlier Identification

Instead of using an absolute fitness function with exhaustive testing, outlier identification is achieved using statistical techniques such as the hat matrix [Rousseuw87],  $H$ , where the diagonal elements  $H_{ii}$  are used to identify the threshold to isolate faulty individuals as outliers. The hat matrix  $H$  defines the Least Squares projection matrix and is so named since it is denoted by a hat on the column vector  $y=(y_1, \dots, y_n)^t$  such that  $\hat{y}=H*y$  and  $\hat{y}$  is the LS prediction for  $y$ . The hat matrix  $H$  is defined as follows: consider that there are  $p$  explanatory variables and one response variable which will have  $n$  observations. The  $n$ -by-1 vector of responses is denoted by  $y=(y_1, \dots, y_n)^t$ . The linear model states that  $y=X*\theta+e$ , where  $\theta$  is the vector of unknown parameters,  $e$  is the error vector and  $X$  is the  $n$ -by- $p$  matrix:

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_{11} & \mathbf{x}_{12} & \dots & \mathbf{x}_{1p} \\ \mathbf{x}_{21} & \mathbf{x}_{22} & \dots & \mathbf{x}_{2p} \\ \mathbf{M} & \mathbf{M} & & \mathbf{M} \\ \mathbf{M} & \mathbf{M} & & \mathbf{M} \\ \mathbf{x}_{n1} & \mathbf{x}_{n2} & \dots & \mathbf{x}_{np} \end{bmatrix} \quad (1)$$

Then, the  $H$  matrix is composed from  $X$  as follows:

$$\mathbf{H} = \mathbf{X}(\mathbf{X}^t\mathbf{X})^{-1}\mathbf{X}^t \quad (2)$$

The diagonal elements of  $H$  have a direct interpretation as the effect exerted by the  $i^{th}$  observation on the expectation of response variable because they equal  $\partial \hat{y}_i / \partial y_i$ . The average value of the diagonal element  $H_{ii}$  is  $p/n$  and it follows that  $0 \leq H_{ii} \leq 1$  for all  $i$ . In the CBE approach, the  $DV$  of each individual can be viewed as one observation or one explanatory variable, and the observation interval can be set as the size of the entire population. Fortunately, since the  $X$  matrix consists of only one column in our application, we can see that the result of the  $X^tX$  product is a single-element vector matrix, and its inverse can be computed using a straightforward computation. In general, the computation complexity of the  $H$  matrix approach is  $2n^2+1$ . In CBE, the threshold value is determined by an analysis of the diagonal elements  $H_{ii}$  of the hat matrix generated from population statistics accumulated over an evaluation window. In order to accelerate the identification of outliers, a *Sliding Window*,  $S$ , defines the period with which the global discrepancies consensus, to which all individual values are compared, is

updated. Typically,  $S$  is selected to be an integer multiple of  $E$  such that  $S=q * E$ , where  $1 < q < |C|$  and  $|C|$  is the population size.

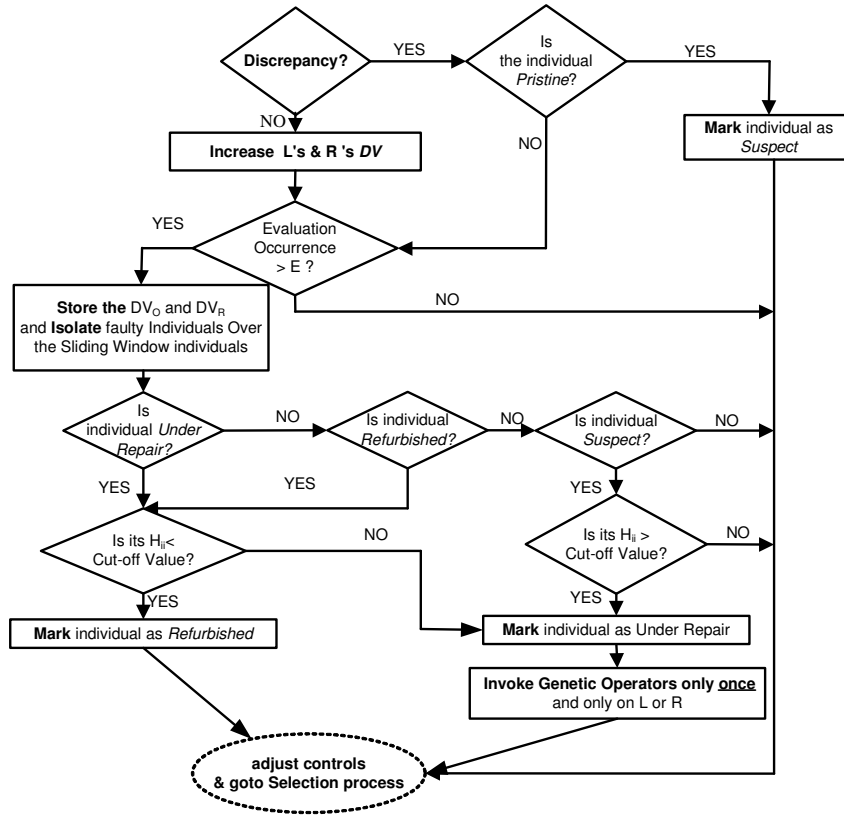


Figure 3: Fitness State Adjustment Process in the CBE Technique

Figure 3 depicts the *Fitness State Adjustment* process in CBE. Whenever a discrepancy is detected, the discrepancy values of the individuals involved are updated. The new discrepancy values are then compared to  $DV_R$  and  $DV_O$  to determine whether the individuals transition from one fitness state to another. Ideally, the repair and operational discrepancy values are updated after a sliding window width of evaluations have been completed. Under ideal conditions, as soon as all the individuals in the population have completed at least  $E$  comparisons each, new values of these thresholds are obtained. Since it may be impractical to wait for all individuals to complete the requisite iterations, the sliding window width  $S$  reduces the latency involved in updating  $DV_R$  and  $DV_O$  by considering a subset of individuals instead of the entire population. The thresholds are updated as soon as a number of individuals, as defined by the sliding window width, have completed  $E$  iterations.

### 3.4 Achieving Device Refurbishment

Conventional GAs frequently use static fitness functions to search for pre-defined globally optimal criteria in analog [Gwaltney05] or digital [Miller00] circuits. On the other hand, CBE uses a self-adaptive fitness measure that is based on consensus formation. This allows for adaptation throughout the process of solution construction involved with evolving a repair. If the realtime inputs are limited to a subset of the input space temporarily, then the relative fitness measure directs the GA towards creating individuals that perform best for this subset. However, there still remain other individuals in the population that perform optimally for other subsets. In the presence of viable alternative configurations, such *Recovery Complexity* of seeded search can be more tractable than *Design Complexity* using a blank slate, as shown in Section 4.2.

Coarse-grained functional elements are recombined into candidate repairs using CBE's inter-module crossover operator. For crossover to occur such that offspring are guaranteed to utilize only mutually-exclusive physical resources within each *L* and *R* half configuration, a two-point crossover operation is carried out with another randomly selected *Pristine*, *Suspect* or *Refurbished* individual belonging to the same *L* or *R* half, respectively. By enforcing speciation, breeding occurs exclusively in *L* or *R*, and non-interfering resource use is maintained. Crossover points are chosen along the boundaries of the FPGA's *Configuration Logic Blocks (CLBs)* so that intra-CLB crossover does not incur logic hazards. To encourage diversity and prevent stasis, an intra-modular input permutation operation performs alterations to logic cell functionality. The input permutation operator randomly changes the CLB's functionality or reconnects one of its inputs to a new randomly selected output. The *input permutation rate* defines the probability of changing the input connections and the logic functions of an LUT when the input permutation operator is applied.

## 4 CBE Performance Evaluation

The search-space complexity of a refurbishment problem is quantitatively compared to the complexity of the design problem using exhaustive analysis of the output space. Further, refurbishment experiments were conducted using two classes of benchmark circuits. The first class consists of circuits where the fan-in exceeds fan-out and the second class includes two circuits where the converse applies. The performance of CBE in TMR and Duplex modes are analyzed for both kinds of circuits. In all experiments, performance is evaluated using two different schemes which are based on the bit-weight tabulation and the hamming-distance scoring of the observed outputs, respectively.

### 4.1 Circuit Representation and Benchmark Characteristics

The FPGA structure used in the following experiments is similar to that used by Miller and

Thompson for GA-based arithmetic circuit design [Miller00]. The feed-forward combinational logic circuit uses a rectangular array of nodes with four inputs and one output. Each node represents a *Look-up Table* (LUT) in the FPGA device, and a *Configurable Logic Block* (CLB) is composed of four LUTs. There are five dyadic operators OR, AND, XOR, NOR, NAND along with the unary operator NOT, from which a function may be composed within an LUT. The LUTs in the CLB array are indexed linearly from 1 to  $n$ . Array routing is defined by the internal connectivity and the inputs/outputs of the array. Internal connectivity is specified by the connections between the array cells. The inputs of the cells can only be the outputs of cells with lower row numbers. Thus, the linear labeling and connection restrictions impose a feed-forward structure on the combinational circuit.

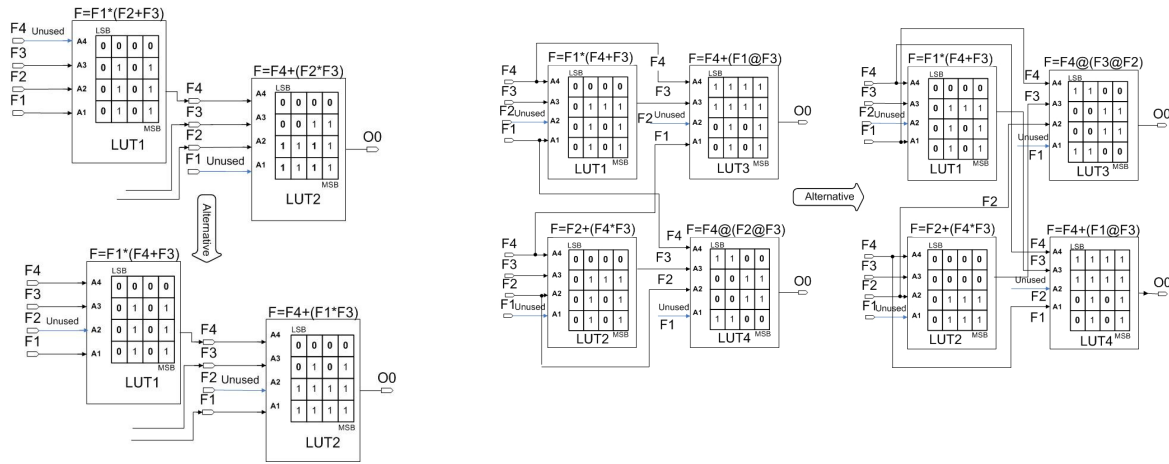


Figure 4: Generation of Alternate Configurations by –

a) Input Permutation (shown on left) and b) Cell Swapping (shown on right)

Each of the benchmark circuits was converted into a Verilog representation that preserved the described functionality. The design was then instantiated on the FPGA using Xilinx ISE version 9.1. A diverse population of configurations was created from the single Xilinx tool synthesized design using *input permutation* and *cell swapping* operators. Figure 4 shows these operators, where F1 is the Least Significant Bit (LSB) of the input to an LUT and F4 is the Most Significant Bit (MSB). As shown in Figure 4a, input permutation leverages low-level redundancy by utilizing the unused inputs of LUTs to modify the input sequence of a single LUT as well as corresponding LUT functionality to maintain identical output behavior. The cell swapping operation, shown in Figure 4b, changes interconnection sequences among LUTs. The cell-swapping operation maintains the feed forward property and re-connects the LUTs to preserve the functional logic. Together, these operations produce diverse circuits with different behavior under single or multiple physical resource failures. These circuit modification operators are also used later by the genetic algorithm to realize refurbished configurations during the repair process.

Benchmark circuits from the MCNC91 benchmark suite [Yang91] were used to analyze CBE performance. Table II lists the characteristics of these circuits. As listed in Table II, the *z4ml* and *cm85a* circuits have a fan-in greater than the fan-out, and the *cm138a* and *2x-decod* circuits have a fan-out greater than the fan-in value. To verify CBE performance on a circuit that utilizes more resources than the circuits provided by the MCNC91 suite, the *2x-decod* circuit was created by appending multiple copies of the *decod* benchmark circuit. The resulting *2x-decod* circuit utilizes approximately four times the LUTs used by the other circuits. The circuits were described using VHDL for synthesis on a Xilinx Virtex-II Pro VP7 FPGA to estimate the gate count and the number of LUTs used. The *input pin redundancy* is calculated as the ratio of the number of unused LUT input pins to the total number of LUT input pins. Table II also lists the percentage of aberrant outputs produced by each circuit under a single stuck-at fault for the entire output space, across all possible fault locations to indicate the demands of each refurbishment task.

Table II: Characteristics of Benchmark Circuits

Type of Circuit	Circuit	Functionality	No. of Inputs	No. of Outputs	Gate Count	LUT Count	Input Pin Redundancy (%)	Aberrant Outputs (%)
Fan-in > Fan-out	z4ml	2-bit adder	6	4	20	8	25	28.6
	cm85a	Logic	11	3	38	12	16.7	19.9
Fan-out > Fan-in	cm138a	Logic	6	8	17	10	22.5	6.6
	2x-decod	Decoder	10	32	44	40	25	3.7

Results from experiments conducted on the MCNC91 circuits also provide insights into the relative merits of operating CBE in the Duplex and TMR modes, and the effect of the performance evaluation method used. To examine more demanding failure scenarios the following experiments consider multiple resource faults.

## 4.2 Quantifying Search Space Complexity under Fault

In order to evaluate the effect of a single stuck-at fault at the inputs of a circuit, the *Correctness-Under-Fault (CUF)* search space characteristics for the various circuits are generated. The CUF characteristics for a circuit are obtained by inserting a single stuck-at fault at each of the inputs of the circuit, and then applying all possible input combinations to the instantiated circuit. The deviation of the observed output from the correct, expected output completely describes the response of the circuit to all possible stuck-at faults for its entire input space. Using this data, a three-dimensional representation of the refurbishment search space can be plotted as shown in Figure 5.

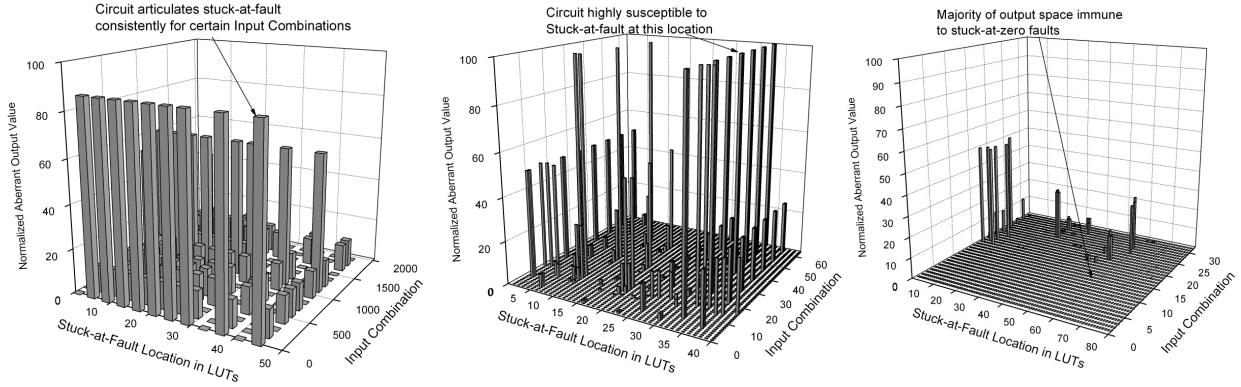


Figure 5: MCNC91 Benchmark Circuit Sensitivity to Stuck-at Faults

– a) *cm85a*, b) *cm138a* and c) *decod* Circuit

The single stuck-at fault CUF search space of the benchmark circuits are shown in Figure 5, which show the Root Mean Squared discrepancy observed for all combinations of input and stuck-at-fault locations. Vertical bars depict representative aberrant outputs, with one sample taken from every 300 data points of the entire search space to enhance readability. In the above figures, the  $x$ -axis represents a particular stuck-at fault identified by the input pin at which the fault is introduced, and the  $y$ -axis represents the input combination applied to the circuit. The  $z=0$  plane represents input combinations for which the output response of the circuit is ideal, in the presence of a stuck-at-fault. The percentage of aberrant outputs for the various circuits listed in Table II are obtained as the percentage of such points in the output space that are affected by the various stuck-at faults. The peaks and troughs in the 3-dimensional plot represent deviations from the expected output due to the presence of a fault. The search space may be sparse, as in Figure 5c, which represents the CUF space of the  $2x$ -*decod* circuit, or dense as in the case of the *cm85a* circuit shown in Figure 5a.

In the case of a refurbishment problem, the evolutionary algorithm is assisted a-priori by the presence of points in the search space where the deviation from the expected behavior is null, as represented by the set of points for which Normalized Aberrant Output is zero. For example, if a particular LUT input is unused, a stuck-at fault at this pin will not adversely affect the outputs of the circuit. This characteristic can be used by the cell-swapping and input permutation operator during the search for a refurbished configuration. In a design problem, the search for a solution starts from a population of arbitrary individuals which provide no such partial functionality. Yet, a refurbishment problem can leverage diversity of partially working spares.

### 4.3 Prioritizing Individuals for Refurbishment

Under CBE, individuals are prioritized for refurbishment operations based on their discrepancies. In particular, individuals whose DV's deviate the most from the average DV of the population are given

more opportunities to undergo refurbishment. This is implemented by reloading the individual under repair with a frequency exceeding that of individuals who have a higher relative fitness. Figure 6 shows the measured performance of an individual over 28 iterations during the repair process for the *z4ml* circuit. In this particular experiment, the reintroduction rate used was 20, with both the cell-swapping rate and the input permutation rate set to 20%. As shown in Figure 6a, whenever the discrepancy of the individual rises above the average discrepancy of the population, the individual is reloaded onto the FPGA, as evidenced by Figure 6b. This can be clearly seen for the first and the next to last iterations shown in Figure 6a and Figure 6b. Conversely, when the individual discrepancy is equal to, or less than the average discrepancy of the population, the individual is not reloaded, or reloaded less than the average member of the population. This ensures steady improvement in the average fitness of the population, while ensuring that individuals are prioritized for refurbishment operations based on their relative fitness arrived at by using a consensus-based evaluation method.

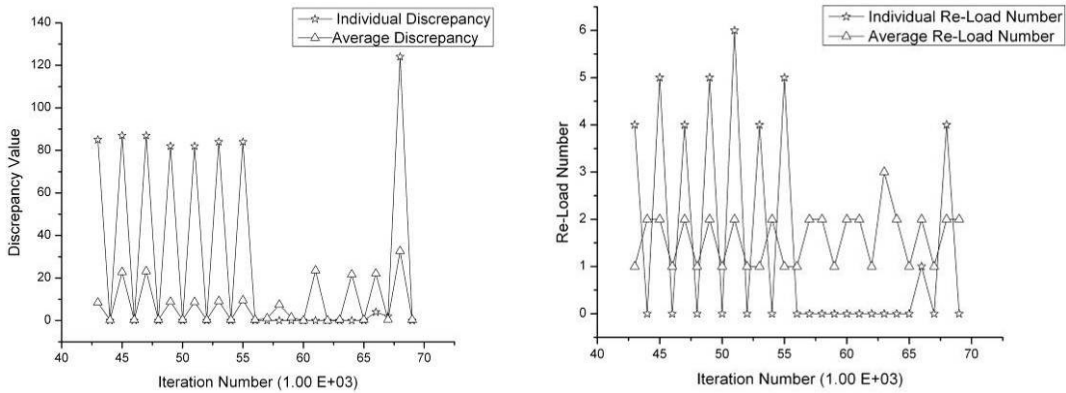


Figure 6: Prioritizing Individuals for Refurbishment  
a) Discrepancy Values, and b) Number of Iterations the Individual is Reloaded

#### 4.4 Comparing CBE performance in Duplex and TMR Modes

Figure 7 and Figure 8 show the performance of CBE under the Duplex and TMR modes when using bit-weights to calculate the fitness of individuals. Figure 7 shows the results of refurbishing circuits in a population of 20 individuals in the Duplex mode, with ten individuals each comprising the Left- and Right-half configuration populations. The Duplex experiment begins when a fault is inserted into two resources, one on the Left-half and one on the Right-half, which impact 18 of the 20 individuals in the population. In the TMR mode, a population of 30 individuals are used, with three resource faults distributed across each voting component affecting 27 out of the 30 individuals. However, as opposed to the Duplex mode, in the TMR mode, outputs from three individuals are compared for the input vector applied to realize throughput, and the majority outcome is asserted as the output of the system.

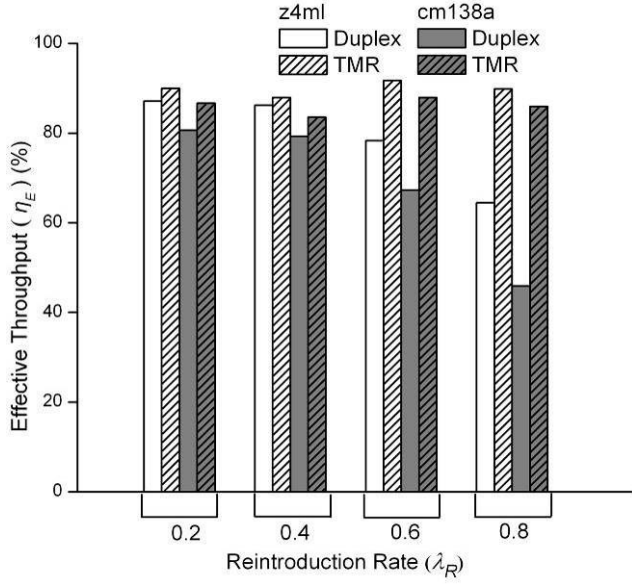


Figure 7: Effective Throughput  $\eta_E$  during Regeneration Under Duplex and TMR Modes of Operation

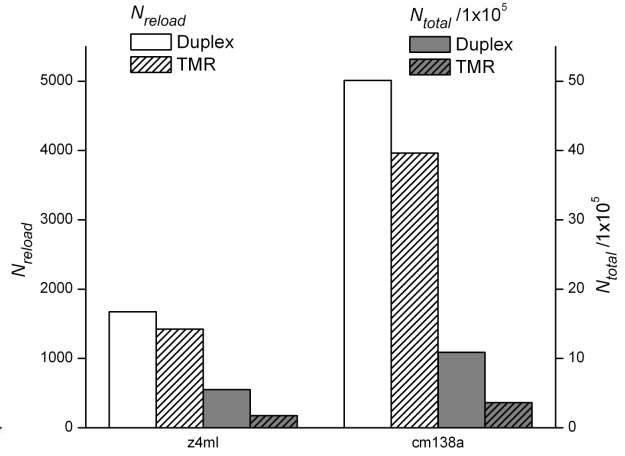


Figure 8: Comparison of Performance Characteristics under Duplex and TMR Modes

In all these experiments, the cell-swap rate and the input permutation operation rate were maintained at 80%. In Figure 8, performance metrics from the experiment refurbishing the population with re-introduction rate  $\lambda_R = 0.4$  are presented, in order to compare the overheads of the two modes. Detailed results obtained from the implementation of the two modes are listed in Table III which tabulates several parameters listed in Equation 3. The *effective throughput*,  $\eta_E$  is measured using the following relationship:

$$\eta_E = \frac{N_{total} - N_{evolution} - N_{reload} - N_{incorrect} - (N_{reload} \times \beta_{reload})}{N_{total}} \quad (3)$$

where,  $N_{total}$  is the total number of iterations required to refurbish the population,

$N_{evolution}$  is the number of iterations in which the genetic recovery operators are invoked,

$N_{reload}$  is the number of iterations where the individuals currently evaluated are replaced by other members from the population,

$N_{incorrect}$  is the number of iterations yielding discrepant outputs verified during the experiment to be incorrect,

$\beta_{reload}$  is the *reload penalty*, which is the ratio of the time taken to reload a configurations and the time taken to compute the outputs for a single input.

Thus,  $\eta_E$  measures effective throughput during refurbishment by accounting for the number of iterations, and the time spent in refurbishment-related operations.

As shown in Figure 7, for low values of  $\lambda_R$ ,  $0.2 \leq \lambda_R \leq 0.4$ , the effective throughput of CBE in the

Duplex mode is only 2% to 6% lower than TMR mode. For example, with the *z4ml* benchmark circuit, from Table III, CBE in TMR mode provides 2.9% higher effective throughput when compared to the Duplex mode. The difference in effective throughput is greater across different values of  $\lambda_R$  for the *cm138a* circuit. Performance varies depending on the fan-in / fan-out ratio of circuit as shown by the *z4ml* circuit, where fan-in > fan-out, and the *cm138a* circuit where fan-in is less than fan-out.

Table III: CBE Performance under Duplex and TMR Modes for Two Different Circuits

<i>Circuit</i>	<i>Mode</i>	$\lambda_R$	$N_{evolution}$	$N_{incorrect}$	$N_{reload}$	$N_{total}$	$\eta_E$	<i>Fully Refurbished Individuals</i>
<b>Z4ml</b>	Duplex	0.2	144	$3.9 \times 10^4$	1594	$4.4 \times 10^5$	87.1	5
		0.4	166	$5.7 \times 10^4$	1674	$5.4 \times 10^5$	86.2	11
		0.6	133	$5.3 \times 10^4$	1671	$3.3 \times 10^5$	78.3	13
		0.8	131	$5.7 \times 10^4$	1907	$2.2 \times 10^5$	64.5	12
	TMR	0.2	132	$3.9 \times 10^3$	1554	$2.1 \times 10^5$	90.0	5
		0.4	150	$5.9 \times 10^3$	1422	$1.8 \times 10^5$	87.9	12
		0.6	125	$1.5 \times 10^3$	1002	$1.5 \times 10^5$	91.7	13
		0.8	121	$2.3 \times 10^3$	1237	$1.6 \times 10^5$	89.9	13
<b>Cm138a</b>	Duplex	0.2	187	$1.1 \times 10^5$	4771	$8.7 \times 10^5$	80.6	4
		0.4	231	$1.7 \times 10^5$	5011	$1.1 \times 10^6$	79.3	11
		0.6	165	$1.6 \times 10^5$	5002	$6.5 \times 10^5$	67.3	12
		0.8	161	$1.7 \times 10^5$	5710	$4.3 \times 10^5$	45.9	12
	TMR	0.2	1362	$1.2 \times 10^4$	4229	$4.3 \times 10^5$	86.6	5
		0.4	1398	$1.8 \times 10^4$	3965	$3.7 \times 10^5$	83.6	11
		0.6	1348	$4.6 \times 10^3$	3125	$3.2 \times 10^5$	88.0	13
		0.8	1340	$6.8 \times 10^3$	3595	$3.2 \times 10^5$	86.0	14

However for  $\lambda_R \geq 0.6$ , the difference in the effective throughput becomes pronounced in favor of the TMR mode. This occurs because a higher re-introduction rate replaces active configurations with configurations from the under repair pool more frequently. TMR throughput is less adversely affected because it ensures throughput whenever any two of three configurations' outputs agree, giving  ${}_3C_2 = 4$  ways for agreement, as opposed to the Duplex mode where there is only one combination to realize agreement. In both Duplex and TMR modes, disagreements trigger reloading of configurations as well as re-computation of the outputs.

Figure 8 quantifies the time vs. space tradeoff during recovery when utilizing 50% fewer physical resources in Duplex mode as opposed to TMR. It shows the number of reloads and the total number of iterations required to refurbish the population for the *z4ml* and *cm138a* circuits when  $\lambda_R = 0.4$ . Under Duplex mode, up to 1.6 times as many reloads and 1.3 to 3 fold total iterations are required to achieve refurbishment of the population. This correlates with the lower effective throughput observed under the Duplex mode. From Table III, with higher values of  $\lambda_R$ , such as  $\lambda_R = 0.8$ , the increased number of

reloads required for Duplex mode skews throughput in favor of TMR mode. The number of fully refurbished individuals increases with an increasing introduction rate, as explained in detail in Section 5.6.

#### 4.5 Effect of Reintroduction Rate on Refurbishment Performance

Table IV lists the number of individuals that were fully refurbished from adverse effects of a single fault inserted into 18 out of 20 individuals under CBE in Duplex mode. A Refurbished individual might be partially or fully refurbished. An individual is fully refurbished if and only if its output response to the entire set of possible input vectors implements the correct truth-table in its entirety. The fitness of the individuals was evaluated using a bit-weight scoring scheme. The stopping criterion for all refurbishment experiments was the condition wherein none of the individuals remain in the Under-Repair pool. Nonetheless, the effectiveness of the refurbishment can also be measured by exhaustively testing each individual under all possible input combinations. Such exhaustive testing is not required for CBE to refurbish individuals; it was conducted only to evaluate performance at the end of a refurbishment cycle.

Table IV: Number of Fully Refurbished Individuals vs. Effect of Reintroduction Rate ( $\lambda_R$ ) for Four Circuits

Reintroduction rate ( $\lambda_R$ )	Circuit	Fully Refurbished Individuals
20	z4ml	8
	cm85a	6
	cm138a	5
	2x-decod	12
40	z4ml	11
	cm85a	12
	cm138a	12
	2x-decod	14

Table IV indicates that as  $\lambda_R$  increases from 0.2 to 0.4, the number of individuals that are fully refurbished in the population rises, irrespective of the circuit used. The improvement depends on not just the fan-in to fan-out ratio, but also on the particular circuit. The *cm138a* circuit shows the best improvement – from three recovered individuals with the lower re-introduction rate to 10 fully refurbished individuals. In the *2x-decod* circuit, which is also a circuit with a fan-in greater than the fan-out, there is an improvement of only two additional fully refurbished individuals.

A higher reintroduction rate increases the probability that more individuals are evaluated, evolved, and therefore improved. This improvement occurs at the cost of the greater number of re-computations and re-loads necessitated by individuals under repair which are instantiated on the FPGA for evaluation,

leading to an increased number of discrepancies. If any individual in the population expresses very low fitness as expressed by a higher discrepancy count, the individual will be demoted to the Under Repair pool to be improved. This refurbishes individuals with low fitness, leading to a higher number of fully recovered individuals.

An additional insight provided by these results is that even though all individuals are not fully recovered, after successive evaluation, the individuals in the population were promoted from the Under Repair pool to the Refurbished pool by virtue of their fitness to inputs observed in practicality. In this manner, CBE emphasizes sustainability by improving the robustness of the entire population in the process of achieving complete recovery.

## 4.6 Comparing Discrepancy Scoring Schemes

Figure 9 and Figure 10 show the relative performance of two different discrepancy scoring schemes. In the Hamming distance method, the fitness of individual configurations was measured using the Hamming distance of the outputs produced by the competing individuals. The bit-weight scheme measures the arithmetic difference between outputs produced by the individuals. Experiments were conducted under the Duplex mode for the *cm85a* circuit and the *2x-decod* circuit. Results from the experiments, both of which were conducted with CBE in the Duplex Mode, are listed in Table V.

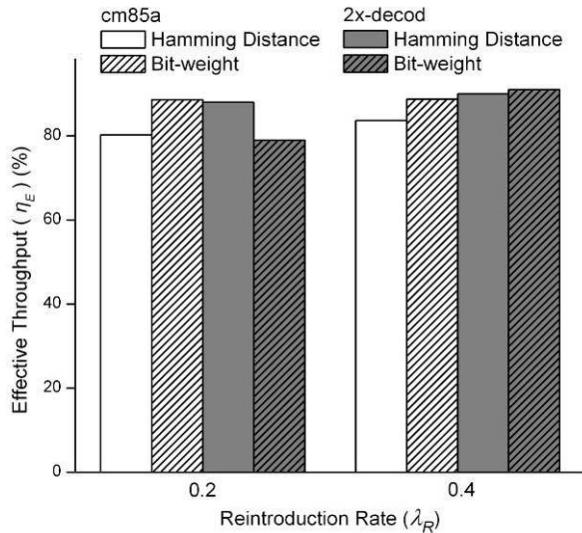


Figure 9: Effective Throughput with Hamming Distance and Bit-weight Schemes

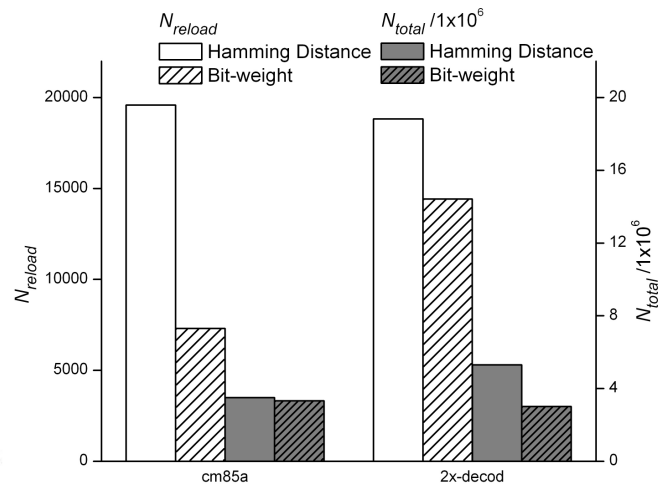


Figure 10: CBE Performance Characteristics with Hamming Distance and Bit-weight Schemes

As shown in Figure 9, the bit-weight evaluation scheme leads to higher effective throughput for the *cm85a* circuit for both values of  $\lambda_R$ , while for the *2x-decod* circuit, the hamming-distance based evaluation scheme seems to lead to a higher throughput. This is due to the fact that unlike the *cm85a*

circuit, the fan-out of the  $2x\text{-decod}$  circuit is greater than the fan-in. Thus, a fault nearer the inputs of the circuit will affect a larger number of outputs for the  $2x\text{-decod}$  circuit. Under these circumstances, the Hamming distance of the output from the ideal output will provide a much better indicator of the fitness of an individual configuration. From Figure 10, it can be seen that the Hamming-distance scheme reports a greater discrepancy value resulting in more refurbishment operations than the bit-weight scheme. As listed in Table V for either performance evaluation scheme, the effective throughput as well as the number of individuals that are fully refurbished for a constant  $\lambda_R$  do not vary significantly. From the results in Table V, it is clear that refurbishment can benefit from the selection of an appropriate fitness-evaluation scheme for the target circuit.

Table V: CBE Performance under Hamming Distance and Bit-weight Performance Evaluation Schemes

<i>Circuit</i>	<i>Performance Evaluation Scheme</i>	$\lambda_R$	$N_{evolution}$	$N_{incorrect}$	$N_{reload}$	$N_{total}$	$\eta_E$	<i>Fully Refurbished Individuals</i>
<b>cm85a</b>	Hamming Distance	0.2	1987	$2.8 \times 10^5$	70387	$8.0 \times 10^6$	87.5	5
		0.4	2120	$3.6 \times 10^5$	19593	$3.5 \times 10^6$	83.6	10
	Bit-weight	0.2	1913	$3.3 \times 10^5$	7270	$4.1 \times 10^6$	87.9	4
		0.4	1684	$2.4 \times 10^5$	7300	$3.3 \times 10^6$	88.7	10
<b>2x-decod</b>	Hamming Distance	0.2	13100	$4.4 \times 10^5$	16676	$5.1 \times 10^6$	88.0	11
		0.4	14420	$3.2 \times 10^5$	18821	$5.3 \times 10^6$	90.0	13
	Bit-weight	0.2	10115	$5.9 \times 10^5$	13362	$3.7 \times 10^6$	79.0	10
		0.4	12750	$1.2 \times 10^5$	14429	$3.0 \times 10^6$	91.0	12

## 4.7 Recovery from Pervasive Faults

The impact of simultaneous resource failures may completely deplete all viable spares from the dormant population. The worst case scenario occurs when all individuals in the  $N$  mutually exclusive resource pools allocated to each module are affected, creating a pervasive hardware failure. However, the residual functionality of each individual can be utilized by the CBE approach to fully refurbish one or more individuals. As depicted in Figure 5, the CUF search space characteristics of the circuits demonstrate the viability of refurbishing individuals using the genetic operators. When affected by pervasive faults, the functionality of each of the diverse individuals remains partially intact. The less affected individuals will then be favored by CBE to remain on board longer and used to generate the consensus output. Conversely, the worst affected individuals will, by virtue of their discrepancy with the majority vote, be forced to undergo evolutionary repair to improve their performance.

The diverse failure behavior under a pervasive fault can be exploited to generate a completely

functional individual even if all individuals in the population are faulty. Experiments conducted on the *2x-decod* circuit, which is the most resource-intensive of the benchmark circuits yield completely refurbished individuals. The Hamming distance based fitness metric produces a majority-indicative vote when the outputs of the three modules are compared on a bit-by-bit basis. In these experiments, all of the 30 individuals across the three modules are negatively affected by a single fault in the resources used by each of the TMR modules. In a sample experiment, CBE realizes three completely refurbished individuals after  $N_{total} = 6 \times 10^5$  iterations with a reintroduction rate  $\lambda_R = 0.4$ . To realize refurbishment, the configurations were reloaded  $N_{reload} = 1121$  times, and a total of  $N_{evolution} = 552$  evolutionary operations were completed by CBE.

In all the experiments, the majority voted output produced by the three modules was asserted as the output. The throughput was observed to be maintained at 95% throughout the refurbishment experiment. High throughput is maintained during refurbishment because even partially-fit individuals can arrive at the correct result for many subsets of inputs encountered at runtime. For measuring throughput and evaluating the absolute fitness of the individuals, the outputs were verified against the truth table of the circuit. However, the correctness information provided by these comparisons was not made available to the refurbishment process. Of course, successful resolution of a pervasive fault still relies on having a population large enough and diverse enough to make recovery tractable by consensus.

## 5 Conclusion

Leveraging the property that even partially-fit individuals respond correctly to some subset of inputs, CBE is shown to maintain adaptable levels of system availability in the presence of defective configurations. This allows for graceful degradation using population characteristics without requiring a circuit-specific fitness function. Additionally, the proposed approach requires no specially constructed test vectors, as the response of individuals to real-time inputs forms the basis for evaluation. This recasts the emphasis in EHW for repair from exhaustive testing to a focus on functionality based only on the relevant inputs which are encountered in the embedded application.

Rather than trying to anticipate operating conditions, CBE utilizes runtime information to adapt to the subset of possible faults which are actually present and being articulated. Even pervasive faults that may completely deplete all viable spares from the dormant population are shown to be recoverable, given adequate population size and diversity. This focus on Recovery Complexity emphasizes use of a diverse population of previously correct alternatives as compared to a single failed seed configuration in the case of Design Complexity. Current work includes the development of a self-contained System-on-Chip implementation of self-healing EHW using the Multi-Layer Runtime Reconfigurable architecture [Tan07] as a partial reconfiguration framework for Xilinx SRAM-based FPGAs.

## Acknowledgements

This research was supported in part by NASA Intelligent Systems NRA Contract NNA04CL07A.

## References

- [Abramovici01] M. Abramovici, J. M. Emmert, and C. E. Stroud, "Roving STARS: An integrated approach to on-line testing, diagnosis, and fault tolerance for FPGAs in adaptive computing systems," *NASA/DoD Workshop on Evolvable Hardware*, 2001.
- [DeMara05a] R. F. DeMara and K. Zhang, "Autonomous FPGA fault handling through competitive runtime reconfiguration," in *Proceedings of the NASA/DoD Conference on Evolvable Hardware (EH'05)*, Washington D.C., U.S.A., pp. 109 – 116, June 29 – July 1, 2005.
- [DeMara05b] R. F. DeMara and C. A. Sharma, "Self-checking fault detection using discrepancy mirrors," in *Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'05)*, Las Vegas, Nevada, U.S.A, pp. 311 – 317, June 27 – 30, 2005.
- [Kastensmidt05] F. L. Kastensmidt, L. Sterpone, L. Carro, and M. S. Reorda, "On the optimal design of triple modular redundancy logic for SRAM-based FPGAs," *Proceedings of Design, Automation and Test in Europe*, pp. 1290 – 1295, 2005.
- [Keymeulen00] D. Keymeulen, A. Stoica, and R. Zebulum, "Fault-tolerant evolvable hardware using field programmable transistor arrays," *IEEE Transactions on Reliability*, vol. 49, No. 3, September 2000.
- [Gallagher03] J. C. Gallagher, "The once and future analog alternative: evolvable hardware and analog computation," in *Proceedings of the 2003 NASA/DoD Conference on Evolvable Hardware*, 2003.
- [Greenwood05] G. W. Greenwood, "On the practicality of using intrinsic reconfiguration for fault recovery," *IEEE Transactions on Evolutionary Computation*, vol. 9, Issue 4, pp. 398 – 405, August 2005.
- [Gwaltney05] D. A. Gwaltney, M. I. Ferguson. "Enabling the on-line intrinsic evolution of analog controllers," *2005 NASA/DoD Conference on Evolvable Hardware (EH'05)*, pp. 3-11, 2005.
- [Lach98] J. Lach, W. H. Mangione-Smith, and M. Potkonjak, "Low overhead fault-tolerant FPGA systems," *IEEE Transactions on VLSI Systems*, vol. 6, No. 2, pp. 212-321, June 1998.
- [Lohn03a] J. D. Lohn, G. Larchev, and R. F. DeMara, "A genetic representation for evolutionary fault recovery in Virtex FPGAs," *Proceedings of the 5th International Conference on Evolvable Systems*, Trondheim, Norway, March 2003.
- [Lohn03b] J. D. Lohn, G. Larchev, and R. F. DeMara, "Evolutionary fault recovery in a Virtex FPGA using a representation that incorporates routing," *Proceedings of 17th International Parallel and Distributed Processing Symposium*, Nice, France, April 2003.
- [Lombardi01] F. Lombardi, N. Park, M. Al-Hashimi, and H. H. Pu, "Modeling the dependability of N-modular redundancy on demand under malicious agreement," *Proceedings of the Pacific Rim International Symposium on Dependable Computing*, pp. 68 – 75, December 2001.
- [Miller00] J. F. Miller, P. Thomson, "Cartesian genetic programming," *Proceedings of the Third European Conference on Genetic Programming (EuroGP2000)*.LNCS, Vol. 1802, pp.121-132, Springer-Verlag, 2000.
- [Mitra00] S. Mitra and E. J. McCluskey, "Which concurrent error detection scheme to choose?," *Proceedings of 2000 International Test Conference*, Atlantic City, NJ, pp. 985-994, October 2000.
- [Moore05] P. Moore, G. K. Venayagamoorthy, "Evolving combinational logic circuits using a hybrid quantum evolution and particle swarm inspired algorithm," *Proceedings of NASA/DoD Conference on Evolvable Hardware*, Washington D.C., U.S.A., pp. 97 – 102, June 29 – July 1, 2005.
- [Rousseuw87] P. J. Rousseuw and A. M. Leroy, "Robust regression and outlier detection," *Wiley Series in Probability and Mathematical Statistics*, pp. 216 – 227, 1987.
- [Tan07] H. Tan and R. F. DeMara, "A Multi-layer Framework Supporting Autonomous Runtime Partial

Reconfiguration,” accepted to *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* on 17 July 2007, in-press.

[Torresen02] J. Torresen, “Evolvable hardware as a new computer architecture,” *International Conference on Advances in Infrastructure for Electronic Business, Education, Science, and Medicine on the Internet (SSGRR 2002W)*, L`Aquila, Italy, January 2002.

[Vigander01] S. Vigander, “Evolutionary fault repair of electronics in space applications,” M.S. *Thesis*, Norwegian University Science and Technology, Trondheim, Norway, February 28, 2001.

[Von Neumann56] Von Neumann, J., “Probabilistic Logics and the Synthesis of Reliable Organisms from Unreliable Components,” *Automata Studies, Ann. of Math. Studies*, no. 34, C. E. Shannon and J. McCarthy, Eds., Princeton University Press, pp. 43-98, 1956.

[Xilinx05] Xilinx Inc., “Virtex-II Pro and Virtex-II Pro X platform FPGAs: Complete data sheet,” Oct. 2005. Available: <http://www.xilinx.com/bvdocs/publications/ds083.pdf>.

[Yang91] S. Yang, “Logic Synthesis and Optimization Benchmarks, Version 3.0,” Tech. Report, Microelectronics Centre of North Carolina, 1991.

[Zhang05] K. Zhang, R. F. DeMara, C. A. Sharma, “Consensus-based evaluation for fault isolation and on-line evolutionary regeneration,” in *Proceedings of the International Conference in Evolvable Systems (ICES'05)*, Barcelona, Spain, , pp. 12 – 24, September 12 - 14, 2005.