Applied Soft Computing 11 (2011) 1588–1599

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Autonomic fault-handling and refurbishment using throughput-driven assessment

Ronald F. DeMara, Kening Zhang, Carthik A. Sharma*

School of Electrical Engineering and Computer Science, University of Central Florida, Orlando, FL 32816-2450, United States

ARTICLE INFO

Article history:
Received 10 October 2007
Received in revised form 8 March 2009
Accepted 17 January 2010
Available online 25 January 2010

Keywords:
Evolvable hardware
Genetic Algorithms
Populational fault tolerance
Reconfigurable computing
Evolutionary algorithms
Competitive Runtime Reconfiguration

ABSTRACT

An evolvable hardware paradigm for autonomic regeneration called *Competitive Runtime Reconfiguration* (*CRR*) is developed whereby an individual's performance is assessed using the dynamic properties of the population rather than a static fitness function. CRR employs a *Sliding Evaluation Window* of recent throughput data and a periodically updated *Outlier Threshold* which avoids the extensive downtime associated with exhaustive Genetic Algorithm (GA) based evaluation. The relative fitness measure favors graceful degradation by leveraging the behavioral diversity among the individuals in the population. Throughput-driven assessment identifies configurations whose discrepancy values violate the Outlier Threshold and are thus selected for modification using Genetic Operators. Application of CRR to FPGA-based logic circuits demonstrates the identification of configurations impacted by a set of randomly injected stuck-at faults. Furthermore, regeneration of functionality can be observed within a few hundred repair iterations. The viable throughput of the CRR system during the repair process was maintained at greater than 91.7% of the fault-free throughput rate under a number of circuit scenarios. CRR results are also compared with alternative soft computing approaches for autonomous refurbishment using the MCNC-91 benchmarks.

Published by Elsevier B.V.

1. Introduction

Evolvable hardware (EH) mechanisms for self-repair seek to actively restore lost functionality of digital logic circuits realized in reprogrammable logic devices such Field Programmable Gate Arrays (FPGAs). The EH techniques aim towards providing fault-recovery capability without incurring the increased weight, size, and cost penalties incurred with redundant spares. Hence, recent research has explored the feasibility of using GA techniques to increase FPGA reliability and autonomy [6,14,2,5,19,24,32]. In particular, a detailed overview of a several dozen works which define the field of autonomous regeneration and categorize the possible approaches is presented in Ref. [21]. A key feature of the existing evolutionary techniques for this problem is their reliance upon either exhaustive functional fitness evaluation or exhaustive resource testing during regeneration.

Alternatively, using the *Competitive Runtime Reconfiguration* (*CRR*) approach developed herein, an initial population of functionally identical (same input-output behavior), yet physically distinct (alternative design or place-and-route realization) FPGA configurations is produced at design-time. At runtime, these individuals compete for selection favoring fault-free behavior. Discrepant behavior, whereby the outputs of two competing individuals do not agree on a bit-by-bit basis, is used as a metric during the fitness evaluation process. Over a period of time, as the result of successive pair-wise comparisons, performance capabilities are identified for the entire population regarding the fitness of individuals relative to one another. This fitness information can then be leveraged to select underperforming configurations to receive genetic modification such as crossover or mutation.

Conventional GA techniques employ a population-based optimization approach to the regeneration problem with the objective of producing a single best-fit configuration as the final outcome. They utilize a fitness function for each regenerated circuit which is evaluated exhaustively for all possible input values. However, given that partially complete repairs are often the best attainable with a tractably sized search [14,27], there is no guarantee that the individual with the best absolute fitness measure for an exhaustive set of test inputs will correspond to the individual that has the superior performance under a particular subset of inputs actually applied. Thus, exhaustive evaluation of regenerated alternatives is computationally expensive, yet not necessarily indicative of the optimal performing individual within a population of partially correct repairs. In this paper, we base fitness instead on the actual throughput data of the alternatives when they are placed into ser-

* Corresponding author. Tel.: +1 407 421 3062; fax: +1 407 823 5835.
E-mail addresses: demara@mail.ucf.edu (R.F. DeMara), casharma@mail.ucf.edu (C.A. Sharma).

1568-4946/$ – see front matter. Published by Elsevier B.V.
doi:10.1016/j.asoc.2010.01.020

vice. Hence, two innovations are developed in the CRR approach to facilitate self-adaptive EH regeneration:

(1) relaxation of the requirement for exhaustive assessment vectors; and
(2) fitness evaluation based on outlier identification over time.

These characteristics are used to assess the evaluation overhead and the repair capabilities, respectively, of the CRR approach defined herein.

2. Related work

The existing schemes for EH repair can be broadly classified in terms of their refurbishment strategy, resource coverage, and the granularity at which faults are isolated.

2.1. Offline refurbishment with exhaustive functional testing

Several approaches to GA-based fault-handling in FPGAs utilize exhaustive testing for fault isolation and offline regeneration mechanisms. As listed in Table 1, conventional Triple Modular Redundancy (TMR) [15], Vigander's approach [27] which combines TMR with GAs, and other *n*-plex *spatial voting* techniques [17] deliver real-time fault-handling, but increase power consumption *n*-fold during fault-free operation. These techniques also require exhaustive evaluation of the function being refurbished. On the other hand, STARS [1] is an example of exhaustive evaluation of a resource-oriented diagnostic that performs *Built-in Self-Tests* (*BISTs*) on sub-sections of FPGA. Under this paradigm, the test area roves across all FPGA resources. Portions of the FPGA are continually taken offline in succession for testing while the functionality is moved to a new location within the reprogrammable fabric. One limitation is that detection latency can be large since tests must sweep through all intervening resources before a fault is detected. Potential throughput unavailability due to diagnostic reconfigurations when no faults have yet occurred is also a consideration.

As listed in Table 1, methods proposed by Lohn [14], Larchev [11], and Lach [10] either rely on offline regeneration supported by exhaustive functional testing, or pre-determined spares defined at design-time. Other soft computing-based approaches to fault tolerance include *Jiggling* [4,5] which combines TMR with a (1 + 1) GA, *genetic-learning based models* [24], an *Asexual GA* for FPGA self-repair [22], and *bio-inspired systems* for evolving dependability [26] which apply immune-based approaches on circuit case studies similar to those herein. In addition, Liu et al. [13] present another biologically inspired model for transient faults.

Of the methods in Table 1, only Keymeulen et al. [6] investigate the possibility of using a population-based approach to desensitize circuits to faults. They exploit evolutionary techniques at design-time so that a circuit is more likely to be designed to remain functional even in presence of various a-priori envisioned faults. Their *population-based* fault tolerant design method evolves diverse circuits and then selects the single most fault-insensitive individual. While their population-based fault tolerance approach provides passive runtime fault tolerance, the approach developed herein adapts dynamically to environmental demands through intrinsic evaluation of a fitness consensus from the entire population at runtime.

2.2. Forming a robust consensus using diversity

To form a robust consensus, Layzell and Thompson [12] dealt with fitness evaluation by exploiting *populational fault tolerance* (*PFT*). Under a PFT strategy, the creation of the best-fit individual proceeds incrementally by incorporating additional elements into partially correct prototypes as they adapt to faults. They speculate that evaluation becomes focused on the precise regions of relevance within the search space during the execution of online processes. This provides motivation to explore CRR's first goal mentioned in Section 1 of relaxing exhaustive input evaluation, especially before returning a partially repaired configuration back to service.

Yao et al. [30] further emphasize that in evolutionary systems the population as a whole contains more robust information than any one individual alone. They demonstrate the utility of information contained within the population using case studies from the domains of artificial neural networks and rule based systems. In both cases, the final collection of individuals outperforms any single individual. Yao and Liu [31] further extend this concept by presenting four methods for combining the different individuals in the final population to generate system outputs. While the authors devise a method to utilize the information contained in the population to improve the final solution, they did not attempt to use the information in the population to improve the optimization process itself. More recently in Ref. [29] the authors describe using fitness sharing and negative correlation to create a diverse population of solutions. A combined solution is then obtained using a gating algorithm that ensures the best response to the observed

Table 1
Characteristics of related FPGA fault-handling schemes.

Approach	Fault-handling method	Latency	Fault detection		Resource coverage		Fault isolation
			Distinguish transients	Logic	Inter-connect	Comparator	Granularity
TMR	Spatial voting	Negligible	No	Yes	Yes	No	Voting element
[27]	Spatial voting and offline evolutionary regeneration	Negligible	No	Yes	No	No	Voting element
[14]	Offline evolutionary regeneration	Negligible	No	Yes	Yes	No	Unnecessary
[10]	Static-capability tile reconfiguration	Relies on independent fault detection mechanism					
STARS [1]	Online BIST	Up to 8.5M erroneous outputs	Test pattern transients	Yes	Yes	No	LUT function
[6]	Population-based fault-insensitive design	Design-time prevention emphasis	No	Yes	Yes	No	Not addressed at runtime
CRR (*proposed herein*)	*Competitive runtime-input fitness evaluation and evolutionary regeneration*	*Negligible*	*Transients are attenuated automatically*	*Yes*	*Yes*	*Yes*	*Unnecessary, but can isolate functional components*