Category: Fault tolerant hardware

Title: Fault tolerance in evolved sorting networks: the mechanism responsible for inherent robustness

Authors:
   Rob Shepherd
   Dr. James Foster

First author address:
   529 Pennsylvania Ave.
   Dover, DE 19901

First author email address:
   rob_shepherd@bigfoot.com

First author phone number:
   302.399.3691
Abstract

Research has indicated that evolutionary design confers a certain degree of fault tolerance on circuits created by these methods, but the mechanism responsible for this effect has been elusive. We postulate that this effect is apparent in evolved sorting networks and seek to isolate the underlying method behind it. We compared evolved and hand-designed sorting networks, using a measure of robustness called bitwise stability. Our results show the metric is slightly biased toward networks sorting larger numbers of inputs, but gives evidence of error correction patterns when used to compare networks with the same number of inputs.

Introduction

The field of evolvable hardware seeks to enhance electronic circuit capabilities by allowing them to adapt using simulated evolution as a guide. This can be done completely in simulation, called extrinsic, or by immediately reconfiguring the hardware in response to the evolutionary algorithm, called intrinsic. Additionally, evolution may design a totally static system before any hardware implementation, known as off-line, or may continue to shape the system during its operation in response to faults or external inputs, called on-line.

One way of simulating evolved hardware for analysis is implementation as a sorting network, since this type of comparison-swap algorithm can be hard-wired. Such a network that sorts \( k \) values is called a \( k \)-input sorting network. Figure 1 shows an example of a 4-input network. The horizontal lines represent the inputs and a connection between two lines indicates a comparator (detailed in Figure 2), which compares the two inputs and puts the larger one on the lower line. The input to the network is depicted at the left of the figure and the sorted output is at the right. Using the sample network, the binary input 1001 would be sorted to 0011. This particular network is a minimum sorting network (MSN), which is one with the fewest number of compare-exchange (CE) gates required to correctly sort all inputs. Sorting networks provide a good framework for studying evolved hardware because they can easily simulate electronic circuits and their associated faults, and there is a large literature with examples of hand-designed networks for comparison.

![Figure 1. 4-input sorting network.](image)

Masner et al. used sorting networks to examine the robustness of evolved hardware in the presence of local faults (Masner et al., 1999). They were able to use one model to create the networks (CE gates) while generating faults at the logic gate level, eliminating the representation’s impact on their observations. In order to reduce confusion between CE gates and logic gates in the remainder of this paper, we’ll refer to a CE gate as a column based on the graphical representation in Figure 1. The term gate will signify a logic gate, limited to AND and OR gates in our circuits. We also need to clarify terms of size for these networks. We will use size to indicate the number of input bits, while length will signify the number of columns present regardless of number of inputs. Therefore, a larger circuit would be one sorting more inputs and a longer one would have more columns than another.

Masner et al. examined networks of different size, length and representation (i.e., chromosomes represented by array and tree structures). Their evolved 4-, 6- and 9-bit circuits were subjected to pass-through (PT), stuck-on-zero (SO0) and stuck-on-one (SO1) faults and the number of correctly sorted bits counted. The circuits, slightly larger than known MSNs, were then compared with some published MSNs that had been subjected to the same error model to determine whether evolution produced more robust circuits. Their results indicated that some increase in network size (i.e., number of columns) added to the robustness, but only to a certain degree beyond which additional columns...
served only to have adverse reaction to faults. However, the mechanism behind this improved robustness was not readily evident.

Figure 2. Compare-exchange (CE) gate (courtesy Masner et al.).

Designing circuits using evolutionary methods holds promise of superiority in several aspects. First, these circuits demonstrate a certain level of generality, allowing them to perform tasks for which they were not explicitly designed (Masner, 2000; Thompson, 1996) or operating in environments they may not have actually experienced (Keymeulen et al., 1997). Design time is reduced for small networks as well (Koza et al., 1998), providing a direct costs savings. We also believe this time may be used for further exploration in search of better circuits.

The benefit of primary interest to us is the apparently inherent fault tolerant qualities present in evolved circuits. Masner et al. cited previous research (Layzell & Thompson, 2000; Masner et al., 1999) pointing to this effect and demonstrated it through analysis (Masner et al., 1999; Masner, 2000). They defined fault tolerance, or robustness, of the circuits as the ability to degrade gracefully in the presence of local faults. This is different than some interpretations (Avizienis, 1997; Keymeulen et al., 2000; Layzell & Thompson, 2000), which are more along the lines of requiring a system to not fail catastrophically when faults occur, whether they are variations in temperature, radiation damage to the circuit, silicon defects, etc. We continue to use the former definition of robustness throughout this paper since it better reflects the fault tolerance needs of digital logic systems.

Other researchers investigated fault tolerance as well (Harvey & Thompson, 1997; Yu & Miller, 2001), but their works have focused on slightly different aspects. Layzell and Thompson (Layzell & Thompson, 2000) postulated that populations confer robustness by evolving individuals who perform well in the presence of faults that render the previously best individual ineffective. Their *populational fault tolerance* stems from the evolutionary design’s incremental nature and suggests that ancestral configurations remain largely intact in final genotypes, which in turn provides individuals in the population who can “step up” following the fault event described above. Harvey and Thompson (Harvey & Thompson, 1997) explored the effect of *neutral ridges* through the search landscape. They define a *neutral network* as “a set of connected points of equivalent fitness, each representing a separate genotype.” (Harvey & Thompson, 1997) The importance here is that genetic operations (crossover and mutation) do not have any effect on the fitness of individuals traversing such a network. Yu and Miller explicitly used this neutrality effect to improve the evolvability of their evolutionary computation system (Yu & Miller, 2001).

**Overview**

Our goal for this project was to test a hypothesis that evolution confers a degree of fault tolerance, or robustness, on sorting networks absent any selective pressure or other explicit design specification. Furthermore, we wanted to identify the mechanism responsible for this inherent quality and possibly isolate it for use as selective pressure to further enhance the robustness of evolved circuits.

Masner et al. developed a metric, called bitwise stability (*BS*) (Masner et al., 1999), to measure the relative robustness of each network. This metric counted the number of bits sorted correctly in the presence of certain local faults. Formally, the *BS* of a network is given by
\[ BS(Err, N) = \sum_{e \in Err} \sum_{x \in X_k} \Delta(C(x), N_e(x)) \cdot \frac{1}{|Err| \cdot |X_k| \cdot k}, \]

where \( Err \) is the previously enumerated set of faults (PT, SO0 and SO1), \( N_e(x) \) is the output of network \( N \) with fault \( e \) on input \( x \), \( C(x) \) is the output of a correct sorting network \( C \) on input \( x \), and \( X_k \) is the set of all \( 2^k \) \( k \)-bit strings. The number of correctly sorted bits is given by the “\( \Delta(x, y) \)” term and is the inverse of the Hamming distance between \( x \) and \( y \). One of the features of this metric is in the definition of \( Err \), which can be changed to measure different properties in the circuits, whether evolved or hand-designed. We have taken extensive advantage of this feature in our research. Since it sums the results over the entire set of faults, this metric provides an exact evaluation of a circuit, and will be synonymous with robustness throughout this paper.

Our research was threefold: 1) validate the BS metric as an accurate, unbiased measurement, 2) identify the cause of the inherent robustness in Masner et al.’s work, and 3) evaluate issues of scale by examining larger evolved and hand-designed sorting networks.

**Methodology**

Our research differed from that of Masner et al. in one important way: we were only concerned with comparing a single evolved circuit with a single hand-designed circuit of the same size (i.e., sorting the same number of inputs). Masner et al. eliminated the network length as the major contributor of robustness in evolved networks, so we chose only the most robust individual from each evolved population (regardless of its length) for comparison with a hand-designed circuit found in the literature. Our vision for the use of this research is the generation of a single, high quality network using a minimum of resources (primarily time) and we can think of few practical applications for sub optimal evolved networks in this pursuit.

In order to evaluate different networks (i.e., evolved and hand-designed), we needed to be able to measure the BS of each one. The code used to generate the evolved networks in Masner et al.’s work automatically provided this data. We extracted the applicable software modules to allow for the measurement of any given circuit, so we could make direct comparisons regardless of the source of the subject circuit. We also used parts of Masner’s code to generate larger evolved networks.

**Figure 3: Array representation of a single chromosome.**

The evolutionary method used a traditional genetic algorithm with each chromosome represented by a one-dimensional array of integer pairs. These integer pairs signify the CE gates in the network. Figure 3 shows an example array for the network depicted in Figure 1. We used a two-point crossover with a probability of 0.75. The mutation operator exchanged CE gates with a probability of 0.05. It randomly chose two CE gates and swapped them as shown in Figure 4. We generally allowed evolution to continue until we discovered 10 correct and unique networks to sort a given number of inputs and then chose the most robust one for comparison with other networks of the same input size. Since larger networks took significantly longer to evolve, we limited the number of unique individuals to one or two when we began to generate circuits sorting 10 or more inputs. We allowed the networks to contain as many as twice the number of columns as similar size MSNs and we address the effects of this decision later.
Analysis of the Bitwise Stability Metric

An essential element in finding the cause of enhanced robustness is a measurement of that quality. The BS metric was designed to allow comparisons between circuits of differing input sizes by normalizing the measurement over input size as well as circuit length. Unfortunately, our analysis indicated a subtle bias toward circuits with more inputs. We discovered this by studying the BS of circuits when only one column was subjected to errors. We’ll use the notation $BS_i$ to indicate the BS with $Err$ being the set of all PT, SO0 and SO1 errors in column $i$. The average of the $BS_i$ values in the circuit represents the bitwise stability. In the cases we examined, regardless of input size, the first few columns (between one and six in our experiments) generally had the same $BS_i$ values, while subsequent columns’ $BS_i$ varied, resulting in different BS values among different networks. The number bits of incorrectly sorted when the early columns were subjected to errors followed a pattern based on the input size, given by

$$I_{early} = 2^k \cdot |Err|$$

for $k \leq 10$ inputs. We also examined circuits with 12 and 13 inputs and these followed this pattern closely, though not exactly (within .03 percent). The total number of bits evaluated in one pass (i.e., one column subjected to faults) is

$$B_{column} = 4 \cdot |Err| \cdot 2^k \cdot k,$$

which represents two gates with two input wires each being subjected to $|Err|$ faults over an input set of $2^k$ values and comparing $k$ bits to the correctly sorted outputs. The $BS_i$ is the ratio of the number of correctly sorted bits to the total number of bits, or

$$\frac{B_{column} - I_{early}}{B_{column}} = 1 - \frac{1}{4k}$$

for the first few columns. This clearly shows that early columns’ $BS_i$ will be higher for networks that sort more inputs, which will lead to higher overall BS for these larger circuits and make them appear more robust than smaller sorting networks.

Analysis of Enhanced Robustness in Sorting Networks

Despite the bias toward larger sorting networks, the BS metric is still a valuable tool to compare circuits of the same number of inputs. We used this measurement and its column-wise variant ($BS_i$) to evaluate some sorting networks in an effort to discover the mechanism behind evolved circuits’ inherent robustness. We suspect that certain columns are setting up error correction patterns for previous faults and have done extensive examination of evolved and hand-designed networks applying the BS and $BS_i$ measurements in an effort to discover whether this is the basis. Column-wise ($BS_i$) analysis yielded some interesting data suggesting error correction patterns may indeed exist in most sorting networks, whether evolved or designed by hand. Looking at two selected plots of 9-bit
circuits, a trend of increasing $BS_i$ from the first through the last column is evident (Figure 5). Early analysis suggest this effect may be slightly more pronounced in smaller networks, as measured by the relative increase of $BS_i$ across the network. Figure 6 illustrates this point. Note the steeper slope of the $BS_i$ trend in the smaller 6-bit network compared with that of the trend in the 13-bit circuit. The details of this phenomenon are still under investigation.

![Figure 5: Increasing column-wise stability ($BS_i$).](image1)

![Figure 6: $BS_i$ increase in small vs. large network.](image2)
Analysis of Larger Sorting Networks

We examined circuits that sort 9, 10, 12 and 13 inputs. Unfortunately, we could only compare the 9- and 10-bit networks to one another due to the lack of 12-bit evolved and 13-bit hand-designed circuits. The BS metric’s bias discussed earlier made us suspicious of the results when comparing networks with different input sizes. The small number of published networks of these input sizes further limited our scope. Knuth (Knuth, 1998) provided two examples of hand-designed 10-bit networks and our evolved 9-bit circuits were from (Masner, 2000). The evolved 10-bit circuits were created using Masner’s evolutionary process, though he did not report these data in (Masner, 2000).

Our analysis yielded mixed results. Evolved networks sorting smaller number of inputs seemed to demonstrate more robustness than hand-designed ones, but the gap narrowed as the number of inputs grew (Figure 7). In fact, when we allowed Masner’s evolutionary method to evolve 10-bit circuits with twice the number of CE gates as in Knuth’s MSN (58 vs. 29 comparators), Knuth’s network had a higher BS value than the best of 10 evolved ones. Masner noted that networks tend to become more robust when allowed to grow beyond the size of an MSN, but that advantage only held with the addition of a few extra columns (Masner, 2000). A concern associated with increased network size is that all columns, whether present in the MSN or not, are subject to the faults injected in our evaluation. The additional columns in the evolved network may have contributed less to the circuit’s overall robustness than the degradation they experienced under the fault set. Therefore, simple redundancy beyond these few additional columns is likely not the cause of increased robustness.

When we evolved circuits with an upper bound of 150 percent of that of an MSN (43 CE gates for a 10 inputs), they again gained an advantage over Knuth’s hand-designed MSN. This small degree of redundancy allows the evolutionary process to search a larger pool of potentially robust circuits while limiting the circuits’ exposure to a greatly increased number of faults. This is consistent with one of Masner’s observations that parsimony pressure is required to generate more robust circuits. In fact, their results indicate a peak in robustness with circuits containing only about 20 percent more columns than the associated MSN, noting a negative correlation between additional columns and robustness above that point (Masner, 2000). The downside of applying this parsimony pressure is that the time required to find correct networks increases dramatically as the maximum circuit length is reduced. We speculate this additional time requirement is still marginal compared with time spent designing networks using traditional methods and yields superior performance in the presence of faults.
We feel these data are noteworthy but not conclusive due to the small number of different network sizes. We are currently evolving some 12-bit networks (limited to 120 percent of the number of CE gates in Knuth’s MSN) for comparison with Knuth’s hand-designed circuits and feel this will give more support to our hypothesis. Unfortunately, this process is extremely time consuming on our single processor workstation.

Issues of scale become important in this area because of the time required to design networks with more inputs, whether by evolutionary or traditional methods. We considered that creating a larger set of evolved circuits might have yielded one or more superior networks. This may be the advantage evolution would retain with respect to the design of larger networks, since advances in computational power (parallel processing, Moore’s law, etc.) would enable engineers to generate sizable sets of circuits more quickly than traditional methods could create one correct network by hand.

Conclusions

We have found evidence to support our hypothesis that evolutionary methods create more robust sorting networks than traditional design means, as measured by the bitwise stability metric. Our finding that the metric is subtly biased toward those circuits that sort larger numbers of inputs did not allow us to compare networks with different input sizes, but does not reduce its effectiveness when comparing networks with the same number of inputs. We had indications that a sort of error correction scheme may be at work in most sorting networks, based on a generally increasing column-wise stability across a network. We also noted a diminishing degree of superiority of evolved circuits as the input sizes increased.

Future Work

We are still in search of the mechanism responsible for the relative robustness of evolved networks. It may stem from the evolutionary process producing redundant columns that set up error correction patterns, akin to the *useful junk* that Harvey and Thompson hypothesized (Harvey & Thompson, 1997). We would also like to use this mechanism as a form of selective pressure during evolution and evaluate its effect on finding even more robust networks.
References


