ON RECONFIGURABILITY OF SYSTOLIC ARRAYS

Amiya Nayak, Nicola Santoro
and Richard Tan

SCS-TR-198, NOVEMBER 1991

School of Computer Science, Carleton University
Ottawa, Canada, KIS 5B6

On Reconfigurability of Systolic Arrays

Amiya Nayak, Nicola Santoro
Center for Parallel & Distributed Computing
School of Computer Science
Carleton University
Ottawa, Canada, K1S 5B6

Richard Tan
Dept. of Computer Science
University of Utrecht
Utrecht, The Netherlands

Abstract

This paper deals with identification, characterization, and catastrophic fault patterns that are catastrophic for systolic arrays. It is shown that for a given link configuration in the array, it is possible to identify all PE (processing element) catastrophic fault patterns. The requirement on the minimum number of faults in a fault pattern and its spectrum (spread out) for it to be catastrophic is shown to be function of the length of the longest bypass link available, and not of the total number of bypass links. This paper also discusses the effect of PE failures on computation performed on one-dimensional VLSI processor arrays. All the established properties of catastrophic fault patterns are used to study inherent limits to reconfigurability of these regular architectures. Bounds on number of faults the system can tolerate to provide guaranteed performance are derived.

Keywords and Phrases: Fault-tolerance, Systolic arrays, Reconfiguration, Catastrophic fault patterns.

1 Introduction

Fault tolerance is the survival attribute of computer systems; when a system is able to recover automatically from internal faults without suffering an externally perceivable failure, the system is said to be fault-tolerant. A common approach for achieving fault tolerance in VLSI-based systolic architectures is through the incorporation of redundancy. The popularity of this approach rests on the fact that, with modern technology, it is now possible to incorporate a large degree of redundant processing elements (PEs) and additional circuitry into a single chip. The redundant PEs are used to replace any faulty PE(s); the redundant links are used to bypass the faulty PEs and reach the redundant PEs used as a replacement. Most of the decisions made at design time with regards to fault tolerance are therefore focused on two particular aspects: amount of redundancy and reconfiguration technique. In the following, the VLSI design will be used as the leading example.

Intuitively, a system incorporating a large number of redundant PEs and "long" redundant links should be able to tolerate a large number of PE failures. A long redundant link can bypass a large block of consecutive faulty PEs. However, long wires are not always possible in such systems due to layout constraints. A long wire will introduce larger propagation delays which in turn might create synchronization problems and become the limiting factor in the performance of the system. Furthermore, note that to increase the number of redundant PEs in a chip requires an extra overhead of interconnections and switching circuitry which implies a higher likelihood of failure. Provided that these technical difficulties can be successfully overcome, it still seems natural to assume that a very high degree of tolerance can be achieved by simply providing a sufficiently large number of spare PEs with a large number of long connections.

The effectiveness of the approach of using structural redundancy to increase fault-tolerance clearly does not depend solely on the amount of redundancy. In fact, the availability of a large number of redundant PEs is useless if these PEs cannot be successfully employed to replace the faulty ones. Thus, a main measure of fault-tolerance in such redundant arrays is the *reconfiguration capability* (or reconfiguration effectiveness); that is, the ability to map faulty elements to spares (using bypass links) while preserving the high degree of regularity and locality of reference required by the system to perform correctly.

It is therefore not surprising that a large amount of research has been devoted to the design of reconfiguration algorithms for redundant arrays as well as proposing redundant architectures which facilitate the reconfiguration process [1-16,20-24].

A computer architecture (or network) can be characterized by its topological properties. It is called regular if the underlying graph is regular; that is, it is connected and every node has the same degree. Because of their properties, the regular architectures are very common and amenable to analysis unlike the irregular ones. The focus of this paper will be on the regular architectures.

The effectiveness of using redundancy to increase fault tolerance in a regular architecture clearly depends on both the amount of redundancy and the reconfiguration capability of the system. It does however depend also on the distribution of the faults in the system. In fact, faults occurring at strategic locations in a regular architecture may have catastrophic effect on the entire structure and cannot be overcome by any amount of clever design. For a given design, it is not difficult to identify a set of elements whose failure will have catastrophic consequence.

The main objective of this paper is to study the characteristics of catastrophic fault patterns; that is, patterns of faults whose occurrence has catastrophic effects on the system and cannot be overcome regardless of how "intelligent" is the employed reconfiguration algorithm. This study sheds some light on the inherent limits to reconfigurability of various regular architectures as well as on the factors limiting the effectiveness of the redundancy approach.

The catastrophic fault patterns for linear arrays are studied by treating the linear array as a set of consecutive positive integers. The results are obtained by reducing the problem of characterization of catastrophic fault patterns to the problem of identifying a special family of sets of integers, called deadly sets. Several important properties of deadly sets are established which, in turn, express important properties of catastrophic faults. For example, it is shown that the cardinality of a catastrophic fault pattern does not depend on the number of bypass links available but only on the length of the longest one; achievable bounds are established on the width of catastrophic fault patterns; results expressing the sensitivity of the width to the amount of bypass links available are proved.

All the established properties of catastrophic fault patterns and their windows are used to study limits to reconfigurability of one-dimensional arrays. The catastrophic fault patterns are employed as means to derive fault-tolerant attributes in terms of the minimum number of faults sufficient to cause a catastrophic failure. The minimum number of faults required to have an impact on reconfigurability is derived for different modes of computations. In particular, bounds for impossibility of reconfiguration are established for systems with 1) unidirectional links, 2) bidirectional links where the I/O can be at either side of the array, and 3) even for the case where each PE can act as an I/O port. The negative results hold even if faults are detectable, and transient faults never occur: they draw an impossibility map for reconfigurability in one-dimensional systolic architectures.

It should be stressed that the results established in this paper rely only on the underlying topological structure (e.g., ring, array) of the system; thus, they hold regardless of whether the basic processing elements are hardware registers, transputers, chips, workstations, software modules, etc., as long as the connectivity of the resulting system (network) is regular and uniform. The results apply directly to systems whose topologies fall in one of the classes considered here in which there are only PE failures.

2 Preliminaries

The system being considered are the one-dimensional (or linear) systolic arrays as shown in Figure 1. The basic components of such a system are the *processing elements* (PEs) and the *links*. The links can be either unidirectional or bidirectional. There are two kind of links in redundant arrays: *regular* or *bypass*. Regular links exist between neighboring PEs; bypass links connect non-neighboring PEs. The bypass links are used strictly for reconfiguration purposes when a fault is detected; in the absence of faults, the bypass links are redundant. Bypass links are not shown in Figure 1 for simplicity.

A *fault* in a PE can be best described as its inability to perform correct operation on correct inputs, possibly due to some elementary circuit or gate-level failures. The failures could be either permanent or transient. In this paper, the focus of the investigation is on permanent faults. Any non-faulty PE which is not active during a computation is said to be a *spare* for that computation.

Let $A = \{x_1, x_2, \ldots, x_N\}$ denote one-dimensional systolic array (shortly, array) of N PEs, where each $x \in A$ represents a processing element and there exists a direct link between x_i and $x_{i+1}, 1 \leq i < N$. Any link connecting x_i and x_j where $j > i + 1$ is said to be a bypass link.

Definition 1 *The length of a bypass link, connecting x_i and x_j, is the distance in the array between x_i and x_j; i.e., $|j - i|$.*

Definition 2 *Given an integer $g \in (1, N]$ and an array A of size N, A is said to have link redundancy g, if for every $x_i \in A$ with $i \leq N - g$ there exists a link between x_i and x_{i+g}.*

The above definition can be generalized as follows:

Definition 3 *The array A has link redundancy $G = \{g_1, g_2, \ldots, g_k\}$ where $g_j < g_{j+1}$ and $g_j \in (1, N]$, if A has link redundancy $g_1, g_2, \ldots, g_k$.*

In the following, it will be assumed that $g_1 = 1$ and no other link except the ones specified exists in the array; that is, G totally defines the *link structure* of A.

Given a linear array A of size N, a *fault pattern* for A is a set of integers $F = \{f_1, f_2, \ldots, f_m\}$ where $m \leq N, f_j < f_{j+1}$ and $f_j \in (1, N)$. An assignment of a fault pattern F to A means that for every $f \in F$, x_f is faulty.

Definition 4 *A fault pattern F is catastrophic for an array A with link redundancy G if the array cannot be reconfigured in the presence of such assignment of faults. That is, the removal of the faulty elements and their incident links will cause the array to become disconnected.*

3 Deadly Integer Sets and Catastrophic Fault Patterns

Consider two finite sets $F = \{f_1, f_2, \ldots, f_m\}$ and $G = \{g_1, g_2, \ldots, g_k\}$ of positive integers. Without loss of generality, let $f_i < f_{i+1}$ and $g_j < g_{j+1}$ for $1 \leq i \leq m$ and $1 \leq j \leq k$. Here, G can be visualized as the set of links including the bypass links, and F corresponds to a distribution of faults (fault pattern). The definition of the deadly integer set now follows.

Definition 5 *An integer $x \in Z$, the set of integers, is "dead" if at least one of the following conditions hold:*

1. *$x \in F$,*
2. *$\forall g \in G, x + g$ is dead,*
3. *$\forall g \in G, x - g$ is dead.*

Let D be the set of dead integers with respect to F and G.

Definition 6 *F is deadly for G if $D = Z$. That is, F is deadly for G if all integers are dead.*

Given $x \in Z$, let $R(x) = \{y \in Z : \exists g \in G$ such that $y = x + g\}$ be the right neighbor of x and $L(x) = \{y \in Z : \exists g \in G$ such that $y = x - g\}$ be the left neighbor of x. Also let $N(x) = R(x) \cup L(x)$ be the neighbors of x. Obviously, if $y \in R(x)$ then $x \in L(y)$.

The set of dead integers for a given F and G is now characterized with respect to their "time" of death as follows:

1). $D^0 = F$ and $A^0 = Z - D^0$; that is, initially only the "faulty" integers are dead and all others are "alive".

2a). $D^i_R = \{x \in A^{i-1} : R(x) \subseteq \bigcup_{j<i} D^j\}$

b). $D^i_L = \{x \in A^{i-1} : L(x) \subseteq \bigcup_{j<i} D^j\}$

c). $D^i = D^i_R \cup D^i_L$

d). $A^i = A^{i-1} - D^i$

3). $D = \bigcup_i D^i$

Note that 1, 2, and 3 correspond to the equally numbered conditions in Definition 5. Now define the time of death, $d(x)$, of an integer x to be

$$d(x) = \begin{cases} 0 & \text{if } x \in F \\ i & \text{if } x \in A^{i-1} \cap D^i \\ \infty & \text{otherwise (i.e., } x \notin D) \end{cases}$$

Definition 5 can now be rephrased as follows:

Definition 7 *F is deadly for G if $d(x) \neq \infty$, $\forall x \in Z$*

Now consider the set $\Psi = \{\sigma\}$ of all sequences σ of integers, "infinite at both ends", where each integer in

$$\sigma = \ldots\ x_{i-1}\ x_i\ x_{i+1} \ldots$$

is a right neighbor of the preceding one, i.e., $x_{i+1} \in R(x_i)$. Given σ and $x \in \sigma$, let $s(x)$ denote the successor of x in σ.

Lemma 1 *If F is deadly then $\forall \sigma \in \Psi$, $\sigma \cap F \neq \emptyset$. That is, if F is deadly then any "doubly-ended" infinite sequence of neighbors must contain an element of F.*

Proof: Let $x \in \sigma$. Since F is deadly for G, $d(x) = i \neq \infty$ (by Definition 6). If $i = 0$ then the lemma is proved. Suppose $i > 0$. Then $d(x) = i$ implies $x \in D^i_R \cup D^i_L$.

<u>Case 1:</u> $x \in D^i_R$

This, in turn, implies that $R(x)$ is a subset of $\bigcup_{j<i} D^j$. Since $s(x) \subseteq R(x)$, $d(s(x)) = j$ and $s(x) \in D^j$ where $j < i$.

Claim: $s(x) \in D^j$ for $x \in D^i_R$ and $j < i$
Proof: Suppose $s(x) \in D^j_L$. This implies $L(s(x)) \in \bigcup_{p<j} D^p$. Since $x \in L(s(x))$, $d(x) < j$ contradicting the fact that $d(x) = i > j$. Since $s(x) \in D^j = D^j_R \cup D^j_L$, the claim follows.

If $d(s(x)) = 0$, the lemma is proved; otherwise, define for an integer $k \geq 0$, $s^k(x) = s(s^{k-1}(x))$ where $s^0(x) = x$. What is shown that if neither x nor $s(x)$ is in F, then $d(x) > d(s(x)) > 0$. Furthermore, $s(x) \in D_R^{d(s(x))}$. Let this hold for a contiguous subsequence of σ; that is, for $1 \leq p \leq k$,

1. $d(s^{p-1}(x)) > d(s^p(x)) > 0$,
2. $s^p(x) \in D_R^{d(s^p(x))}$.

It will now be shown that

(i) $d(s^k(x)) < d(s^{k-1}(x))$ and

(ii) if $d(s^k(x)) > 0$ then $s^k(x) \in D_R^{d(s^k(x))}$.

Proof of (i): $s^{k-1}(x) \in D_R$ implies $R(s^{k-1}(x))$ is a subset of $\bigcup_{p<d(s^{k-1}(x))} D^p$. Since $s^k(x) \in R(x)$, $d(s^k(x)) < d(s^{k-1}(x))$.

Proof of (ii): Let $d(s^k(x)) > 0$. Then if $L(s^k(x))$ is a subset of $\bigcup_{p<d(s^{k-1}(x))} D^p$, $s^k(x) \in D_R^{d(s^k(x))}$.

Therefore, either $s^k(x) \in F$ or its "death time" is smaller than the preceding element in the sequence. If the latter case holds, the same process can be applied to $s^{k+1}(x)$ and so on until an element $s^r(x)$ can be found such that $d(s^r(x)) = 0$ which must be the case since the sequence

$$d(x), d(s(x)), \ldots, d(s^i(x)), \ldots$$

is decreasing.

Case 2: $x \in D_L^i$.

The argument is similar by substituting L for R.

This now completes the proof of Lemma 1. □

3.1 Cardinality of Catastrophic Fault Patterns

The following theorem establishes condition on the minimum number of faults that a fault pattern F must consist of in order for it to be catastrophic for a given link configuration G.

Theorem 1 *F is deadly with respect to G implies that the cardinality of F, $\|F\| \geq g_k$.*

Proof: Define g_k sequences σ_i ($1 \leq i \leq g_k$) as follows:

$$\sigma_i = \{i + jg_k : j \in Z^+\} \cup \{i - jg_k : j \in Z^+\}$$

It is easy to verify that

1) $\sigma_i \cap \sigma_j = \emptyset$ if $i \neq j$ for $1 \leq i, j \leq g_k$,

2) $\bigcup_{1 \leq i \leq g_k} \sigma_i = Z^+$.

Let $f(i) = \|\sigma_i \cap F\|$. Thus, from 1) and 2) above, it follows that

3) $\sum f(i) = F$.

If $\|F\| < g_k$ then by 1), 2), and 3), it follows that $\exists \sigma_i$ such that $\sigma_i \cap F = \emptyset$. That is, there exists a double-ended infinite sequence which does not contain any element of F. By Lemma 1, it follows that F would not then be deadly. That is, $\|F\| < g_k$ implies F is not deadly. □

This theorem gives the necessary condition on the minimum number of faults required for blocking a linear array. An important point to be emphasized here is that the minimum number of faults to form a catastrophic fault pattern depends solely on the length of the longest bypass link, g_k, but no other intermediate link nor the number of bypass links. In the following, all discussions will be restricted to the case where there are at least g_k faults, and catastrophic fault patterns containing exactly g_k faults will be characterized.

3.2 Width of Catastrophic Fault Patterns

Definition 8 *The width W_F of a fault pattern F (or fault window) is the number of PEs between and including the first and the last fault in F. That is, if $F = \{f_1, \cdots, f_m\}$ then $W_F = f_m - f_1 + 1$.*

The width of a catastrophic fault pattern depends on two quantities: the link configuration G, and the orientation of G. Obviously, the lower bound on the width of a catastrophic fault pattern, consisting of g_k number of faults, is g_k. This is the case when all faults occur in a cluster. In the following, the width of the largest possible fault window is given for both unidirectional and bidirectional links.

<u>Unidirectional Links</u>

Theorem 2 *Let $\|F\| = g_k$ and $G = \{g_1, g_k\}$ be a set of unidirectional links. Then the width of the fault window for G is given by $W_F = (g_k - 1)^2 + 1$.*

Proof: Partition the linear array of PEs into blocks of g_k elements and list the blocks as the rows of a matrix. Pick a block arbitrarily and mark the first element in the block by an "x" (denoting the first fault). This indicates the start of the window. Observe that going from left to right in a row is same as traversing the array using the link of length 1. Similarly, moving along a column in the matrix is same as traversing the array using the link of length g_k. The construction of a catastrophic fault pattern in this matrix be such that the following properties must hold.

- Each column of the matrix must be covered, i.e., it must have an element marked "x"; otherwise, if there is a column which does not contain an "x" then there is always an escape through that column.
- Each column must have exactly one element marked "x". This is because if there are two or more elements marked "x" in one column, then there will be at least one column which will not have any element marked "x". This implies that it is possible to escape through the column that is not covered.
- Each element below (i.e., the corresponding element in the next row) an element marked "x" must be inaccessible; otherwise, if the element is accessible, there is an escape through that element using the link of length g_k. Notice that for this property to hold, the last element in the first block must be marked "x".

Now the objective is to find a distribution of "x"s in the matrix that meets the above requirements. The obvious one is, of course, the block (row) in which all g_k elements are marked "x" as (x,x, ... ,x). This implies that $W_F = g_k$, the case of the smallest window. There are clearly other patterns that can give a larger window size than g_k. The only distribution of "x"s in the matrix that meets all the requirements and at the same time gives the largest window is the one shown in Figure 2.

Altogether, there are $(g_k - 1)$ blocks (rows) which contains elements marked "x". Each block is of size g_k. The distance between the first and the last "x" in the distribution is W_F. It is clear that

$$W_F = g_k(g_k - 2) + 2 = g_k^2 - 2g_k + 2 = (g_k - 1)^2 + 1. \quad \square$$

It should be observed that the bound established on W_F in this theorem is the upper bound for any unidirectional link configuration $G' \supseteq G = \{1, g_k\}$. This is true because the inclusion of any additional link in G will likely shrink the window size.

Bidirectional Links

Theorem 3 *Let $\|F\| = g_k$ and $G = \{g_1, g_k\}$ be a set of bidirectional links. Then the width of the fault window for G is*

$$W_F = (\lceil \frac{g_k}{2} \rceil - 1)g_k + \lfloor \frac{g_k}{2} \rfloor + 1.$$

Proof: The proof is similar to the proof of Theorem 2. Like before, partition the linear array of PEs into blocks of g_k elements and list the blocks as the rows of a matrix. Pick a block arbitrarily and mark the first element in the block by an "x" (denoting the first fault). The construction of a catastrophic fault pattern in this matrix be such that all the previous properties (in Theorem 2) must hold. In addition, since the links are bidirectional, each element below (i.e., the corresponding element in the next row) an element marked "x" must now be inaccessible from left as well as from right. The only distribution of "x"s in the matrix that meets all the requirements and at the same time gives the largest window is the one shown in Figure 3.

Altogether, there are $\lceil g_k/2 \rceil$ blocks (rows) which contains elements marked "x". Each block is of size g_k. The distance between the first and the last "x" in the distribution is W_F. From Figure 3, it follows that

$$W_F = (\lceil \frac{g_k}{2} \rceil - 1)g_k + \lfloor \frac{g_k}{2} \rfloor + 1. \quad \square$$

It should be observed that the bound established on W_F in Theorem 3 is the upper bound for any bidirectional link configuration $G' \supseteq G = \{1, g_k\}$.

3.3 Effect of G on W_F

Obviously, the width of the fault window varies with the size of G. Consider k, the size of G. When $k = 1$, there are no bypass links; hence, any single fault is catastrophic. When $k = g_k$ (a case which is very much impractical from a circuit manufacturing point of view) only a cluster of g_k faults can be catastrophic; the probability of having a cluster of g_k faults is quite small. Thus, the focus of attention is on the case where $k < g_k$. The following corollary shows that the window size decreases as the size of G increases.

Corollary 1 *Let $G' \subseteq G$ and W'_F, W_F be the corresponding widest fault windows. Then $W'_F \geq W_F$.*

Proof: There are two cases to be considered here.

- Case 1: G' is unidirectional. Theorem 2 gives the width of the largest window for $G' = \{1, g_k\}$. Since it is the upper bound on the size of W'_F, when an additional link is introduced in G', the corresponding W'_F will not increase. Thus, for any $G \supseteq G'$, $W_F \leq W'_F$.
- Case 2: G' is bidirectional. The argument is similar and follows from Theorem 3. $\square$

Some of the fundamental results which show the sensitiveness of the width of the largest fault window to the variation in the size of G in both unidirectional and bidirectional links are presented below.

Unidirectional Links

Corollary 2 *Let $\|F\| = g_k$ and $G = \{1, 2, g_k\}$. Then the width of the widest fault window W_F is given by*

$$W_F = \begin{cases} (\frac{g_k}{2} - 1)g_k + 2 & \text{if } g_k \text{ is even} \\ (\lfloor \frac{g_k}{2} \rfloor - 1)g_k + 3 & \text{if } g_k \text{ is odd} \end{cases}$$

The corollary follows from Algorithm 1, given in Section 4, for the construction of the largest fault window. Figure 4 gives an example of a catastrophic fault pattern for $G = \{1, 2, 12\}$ that is contained in the largest window.

Corollary 3 *Let $\|F\| = g_k$ and $G \supseteq \{g_k - 1, g_k\}$ in which $3 \leq \|G\| \leq g_k - 1$. Then the width of the largest fault window $W_F = g_k$.*

The corollary follows from Algorithm 1, given in Section 4, for the construction of the largest fault window.

Corollary 4 *Let $\|F\| = g_k$ and $G \supseteq \{g_k - 2, g_k\}$ in which $\|G\| \leq g_k - 1$ but the link of length $g_k - 1$ is not available. Then the width of the fault window $W_F \leq 2g_k - 1$.*

Equality in Corollary 4 is achievable. Figure 5 gives an example of a G for which this is true.

Corollary 5 *Let $\|F\| = g_k$ and $G = \{1, \lceil g_k/2 \rceil, g_k\}$. Then the width of the fault window is*

$$W_F = \begin{cases} (\frac{g_k}{2} - 1)g_k + 2 & \text{if } g_k \text{ is even} \\ (\lfloor \frac{g_k}{2} \rfloor - 1)g_k + \lceil \frac{g_k}{2} \rceil + 1 & \text{if } g_k \text{ is odd} \end{cases}$$

This corollary gives the size of the fault window when bypass links of length $\lceil g_k/2 \rceil$ and g_k are available in the linear array. Figure 6 shows an example of a fault window when $G = \{1, \lceil g_k/2 \rceil, g_k\}$. The following corollary is the generalization of Corollaries 2, 3, 4, and 5.

Corollary 6 *Let $\|F\| = g_k$ and $G = \{1, \alpha, g_k\}$. Also let $\beta = \alpha \bmod (g_k - \alpha)$. Then the width of the fault window is given by*

$$W_F \leq \begin{cases} (g_k - \alpha)g_k & \text{if } \beta = 0 \\ \lfloor \frac{g_k - \alpha}{\beta} \rfloor g_k + (\beta - 1)g_k & \text{if } \beta \neq 0 \end{cases}$$

In Corollary 6, equality is achievable. The example in Figure 7 demonstates the case.

Bidirectional Links

In the case of bidirectional links, the results which show the sensitiveness of the width of the largest fault window to the variation of the size of G are now shown.

Corollary 7 *Let $\|F\| = g_k$ and $G = \{1, 2, g_k\}$. Then the width of the largest fault window W_F is given by*

$$W_F = \lfloor \frac{g_k}{4} \rfloor (g_k - 2) + g_k$$

The corollary follows from Algorithm 2, given in Section 4, for the construction of the largest fault window. Figure 8 gives an example of a catastrophic fault pattern for $G = \{1, 2, 12\}$ that is contained in the largest window.

Corollary 8 *Let $\|F\| = g_k$ and $G \supseteq \{g_k - 1, g_k\}$ in which $\|G\| \leq g_k - 1$. Then the width of the fault window $W_F = g_k$.*

This corollary states that when the link set G includes two links, one of length g_k and other of length $g_k - 1$, then any cluster of g_k faults can be catastrophic for G.

Corollary 9 *Let $\|F\| = g_k$ and $G \supseteq \{g_k - 2, g_k\}$ in which $\|G\| \leq g_k - 1$ but the link of length $g_k - 1$ is not available. Then the width of the fault window $W_F \leq 2g_k - 1$.*

Note that Corollaries 8 and 9 are identical to Corollaries 3 and 4 respectively.

4 Construction of Widest Fault Pattern

In this section, algorithms for constructing a catastrophic fault pattern, contained in the largest possible fault window, are given for both unidirectional and bidirectional links.

4.1 Unidirectional Links

Algorithm 1: Construction of a Catastrophic Fault Pattern for Unidirectional Links

Input: A link configuration G
Output: A catastrophic fault pattern F

Step 1: Partition the linear array of PEs into blocks of g_k elements and list the blocks as rows of a matrix. Select a block arbitrarily, and mark the first element in the block by an "x".

Step 2: Repeat

- Select the next block.
- Moving from left to right in the selected block, pick the unmarked element directly below the element that is marked "x" or "y" in the previous block. Mark it by a "y". If the element in the previous block is marked by an "x" then mark all unmarked elements that are to the left at distance $g_1, g_2, \ldots, g_{k-1}$ of the current element by an "x". Note that this process can span the current as well as the previous block.

Step 3: Until all elements in the current block are marked.

Note that the elements marked "x" represent faulty PEs. The distribution of all the "x"s constitutes a catastrophic fault patten. The algorithm assigns exactly g_k number of "x"s. The number of PEs between and including the first "x" and the last "x" is the width of the fault pattern (window).

Example 1: Figure 9 shows a catastrophic fault pattern consisting of 8 faults corresponding to the unidirectional link configuration $G = \{1, 2, 4, 8\}$ as generated by Algorithm 1. The width of this pattern is 18.

Theorem 4 *Algorithm 1 is correct, i.e., it generates a catastrophic fault pattern.*

Proof: The following simple observations can be made on the algorithm:

- Any PE marked with an y is inaccessible, i.e., there is no way to reach this PE using any of the bypass links.
- There is only one x per column. Thus, there can be only g_k x's.
- For each column the x occurs before the y's.

Clearly, if the algorithm terminates then the entire row is marked with x and y, i.e., either the PE is faulty or inaccesible. Thus the algorithm is correct if it ever terminates.

Observe that location 1 and g_k of the first block have x's. Consequently, on the second row, the last element is marked with y, so location $g_k - 1$ must be an x (because of the 1 link) unless it is already a y, i.e., location $g_k - 1$ in row 1 already has an x. Continuing row by row, it can be seen that each location from the right is gradually marked with an x. Consequently, all the PEs in the column under the x are also marked with y. Thus, eventually there is a row that is all marked. □

Intuitively, by examining Algorithm 1, it can be seen that there are more x's and y's for G' in each row, so the algorithm terminates sooner.

Property 1 *Algorithm 1 has time complexity $O(W_F + kg_k)$ and space complexity $O(W_F)$.*

Proof: The algorithm will consider and mark elements only in the first $\lceil W_F/g_k \rceil + 1$ rows of the matrix (denote it as M) as defined by the algorithm; in particular, the last row will be entirely marked (either with "x" or "y"). This marking is performed by a linear scan of these rows together with a backtracking operation. The backtracking operation involves checking (and possibly marking) all the elements in the array at distance g_l $(1 \leq l < k)$ from any element $M[i, j]$ such that $M[i-1, j]$ is marked "x". Since there are exactly g_k elements $M[i, j]$ such that $M[i-1, j]$ is marked "x", the backtracking will require exactly $(k-1)g_k$ operations in addition to the scan which requires exactly $(\lceil W_F/g_k \rceil + 1)g_k$ operations. Hence, the algorithm requires $g_k(k + \lceil W_F/g_k \rceil)$ operations. The space complexity is linear in the size of the fault window W_F. This is because the algorithm operates on $\lceil W_F/g_k \rceil + 1$ rows, each of size g_k, hence the space complexity of $O(W_F)$. It is important to note that W_F is the width of the fault window which could, in the worst case, be of $O(g_k^2)$. □

Theorem 5 *The catastrophic fault pattern generated by Algorithm 1 for unidirectional links has the widest window, W_F.*

Proof: Suppose there is a catastrophic fault pattern with a wider window W_F. Position the pattern into blocks of size g_k with f_1 at location 1 of the first block. Note that since there are only g_k faults, each column can only contain one faulty PE. This means that each column that contains a faulty PE (x) must have the rest of the column be inaccessible (y). This, in turn, forces the PEs $g_1, \ldots, g_k$ distance back to be faulty or inaccessible. In other words, F must contain the same pattern as that in the algorithm. □

4.2 Bidirectional Links

In the following, an algorithm is outlined for the construction of a catastrophic fault pattern F from a given set G when links are bidirectional, and backtracking is allowed in attempts to bypass faulty PEs.

Algorithm 2: Construction of a Catastrophic Fault Pattern for Bidirectional Links

Input: A link configuration G
Output: A catastrophic fault pattern F

Step 1: Partition the linear array of PEs into blocks of g_k elements and list the blocks as rows of a matrix. Select a block arbitrarily, and mark the first element in the block by an "x".

Step 2: Repeat

- Select the next block.
- Moving from left to right in the selected block, pick the unmarked element directly below the element that is marked "x" or "y" in the previous block. Mark it by a "y". If the element in the previous block is marked by an "x" then consider all unmarked elements that are to the left at distance $g_1, g_2, \ldots, g_{k-1}$ of the current element: mark each of them "x" or "y" depending on whether the corresponding element in the row above is unmarked or marked respectively. Now mark those unmarked elements that are to the right in the same row at distance $g_1, g_2, \ldots, g_{k-1}$ of the current element by an "x", ignoring the ones which appear in a column that already has an "x" above it.

Step 3: Until all elements in the current block are marked.

Example 2: Figure 10 shows a catastrophic fault pattern consisting of 9 faults corresponding to $G = \{1, 3, 9\}$ when the links are bidirectional using the construction outlined in Algorithm 2. The width of the widest catastrophic fault pattern is 23.

Theorem 6 *Algorithm 2 is correct, i.e., it generates a catastrophic fault pattern.*

Similar to the proof of Theorem 4.

Theorem 7 *The fault pattern generated by Algorithm 2 for bidirectional links is catastrophic and has the widest window, W_F.*

Similar to the proof of Theorem 5.

Property 2 *Algorithm 2 has time complexity $O(W_F + kg_k)$ and space complexity $O(W_F)$.*

Proof: The only difference between this algorithm and the one for the unidirectional case is that in addition to scanning and backtracking it also checks (and possibly marks) the elements at distance g_l $(1 \leq l \leq k)$ to the right of each element $M[i,j]$ such that $M[i-1,j]$ is marked "x". This requires an additional kg_k operations. The time complexity still remains at $O(W_F + kg_k)$. The space complexity for Algorithm 2 is comparable to that of Algorithm 1 since both algorithms operate on exactly the same number of rows of the matrix. □

5 Limits to Reconfigurability of Systolic Architectures

In systolic arrays, the peripheral PEs are normally responsible for I/O operations. Faults occurring at strategic locations can prevent the computation to proceed, by disrupting I/O operations or even segmenting the structure into pieces which can no longer be suitable for any practical purpose. In this section, the properties of catastrophic fault patterns and their windows are used to study limits to reconfigurability of one-dimensional arrays. The catastrophic fault patterns are employed as means to derive fault-tolerant attributes in terms of the minimum number of faults sufficient to cause a catastrophic failure. The minimum number of faults required to have an impact on reconfigurability is derived for different modes of computations, requiring a minimum number of connected fault- free processors. All negative results established in this section hold even in the most benign situation; i.e., even if faults are detectable, reconfiguration is instantaneous, only correct information is generated, no transient failures occur, etc.

Some additional definitions now follow.

Definition 9 *Given a problem X, let $\|X\|$ denote the minimum number of connected PEs required to compute X. $\|X\|$ is called the size of the problem.*

Definition 10 *A "$(k,m)-$ Fault Tolerant" system is one, where k PEs have failed, but it is still possible to solve problems of all size n, for which $n < m < N$ (total number of PEs).*

Definition 11 *The system is operating normally if it is $(0,N)-$ Fault Tolerant.*

Definition 12 *A "catastrophic" failure is said to have occured if the system is $(k,0)-$ Fault Tolerant.*

The analysis will be based on the direction of information flow in and out of the systolic arrays. As a special case, the systolic structures where it may be possible to download data and upload result in some miraculous way are considered. The three models are the following:

a) Systems with unidirectional links

b) Systems with bidirectional links (I/O can be on one side or both)

c) Systems with bidirectional links as in (b), and also has miracle I/O facilities

Theorem 8 *For a problem of size,* $\|X\| = n$, *a one-dimensional array with* N *PEs and* $G = \{1, 2, \ldots, g_k\}$ *unidirectional links is* $(g_k, 0)-$ *Fault Tolerant.*

Proof: Since $G = \{1, 2, \ldots, g_k\}$, the only fault patterns that can be catastrophic are the clusters of g_k faults. Consider a block fault of size g_k, as shown in Figure 11, occurring somewhere in a linear array. The faulty region is shadowed. Let x and y be two PEs occurring in immediate left and right of the faulty region respectively. There is no bypass link from x to y since the longest bypass link from x takes to a PE in the faulty region. In this case, flow of data from x cannot reach y and hence the output. Therefore, occurrence of a block fault of size g_k (i.e., g_k faults) is enough to disconnect the entire array when the bypass links are of size g_k or less.

The result still holds if the problem size is specified, i.e., $\|X\| = n$. By Definition 9, n fault-free connected PEs are needed to solve the problem. But, the fault block could just happen to be in the input side (see Figure 12), not leaving enough fault-free PEs for the computation to proceed. □

Theorem 9 *For a problem of size,* $\|X\| = n$, *a one-dimensional array with* N *PEs and* $G = \{1, 2, \ldots, g_k\}$ *bidirectional links is* $(Min\{2g_k, N - 2(n-1)\}, 0)-$ *Fault Tolerant.*

Proof: Two cases will be considered here.

- Case 1: $\{N \geq 2(n - 1 + g_k)\}$. Position two fault blocks, each of size g_k, in both side of the array (as shown in Figure 13) such that there are less than n fault-free PEs in either end. By the previous theorem (Theorem 8), these two blocks of faults would stop the flow of data beyond the faulty region into the array. In addition, fault-free PEs in either end, by Definition 9, are insufficient for the problem. The total number of faults sufficient to disconnect the array is equal to $2g_k$.
- Case 2: $\{N < 2(n-1+g_k)\}$ In this case less than $2g_k$ faults, i.e., $N - 2(n-1)$, occurring in one large block, leaving fewer than n non- faulty PEs (see Figure 14) on either side of the array which are insufficient for a problem of size, $\|X\| = n$. Therefore, the total number of faults is simply $N - 2(n-1)$. □

The result in the above theorem will still hold even if faults do not occur in blocks of size g_k. This can be seen in the following corollary.

Corollary 10 *Let* $G' \subset G = \{1, 2, \ldots, g_k\}$. *For a problem of size* $\|X\| = n$, *a one-dimensional array with* N *PEs and the bidirectional link configuration* G' *is* $(Min\{2g_k,\ N - 2(n-1)\}, 0)-$ *Fault Tolerant.*

Proof: Since $G' \subset G = \{1, 2, \ldots, g_k\}$, there are other fault patterns besides the block (or cluster) faults that can be catastrophic. The only difference in the case of G' is that the number of fault patterns is at least one, and each fault pattern has the window size $W_F \geq g_k$. The window containing the fault pattern is equivalent to a fault block in the sense that both contain the same number of faulty PEs. This, in turn, does not change the minimum number of faults required to logically disconnect the array. □

The last remaining model is the case of miraculous I/O. In this model it is assumed that each PE is accessible and can provide I/O facilities. The number of faults required in this case to make such a structure fail will certainly be higher than in the case of normal I/O. Furthermore, the number of faults also depends on the size of the fault window W_F. The following theorem demonstrates that.

Theorem 10 *Given a problem of size, $\|X\| = n$, an one-dimensional array with a total of N PEs, $G = \{1, 2, \ldots, g_k\}$ bidirectional links, and provision of miraculous I/O facilities is $(\alpha, 0)-$ Fault Tolerant, where $\alpha = \lfloor N/(n-1+g_k) \rfloor g_k + Min\{0, [N mod(n-1+g_k)] - n + 1\}$.*

Proof : Consider the linear array given in Figure 15. In this structure, first $(n-1)$ PEs are fault-free, and the next g_k PEs are faulty. Let the pattern of $(n-1)$ fault-free and g_k faulty PEs continue until $\lfloor N/(n-1+g_k) \rfloor$ such patterns are seen. Out of the remaining $[N \ mod \ (n-1+g_k)]$ PEs, if this number is greater than $(n-1)$, there should be a block of $(n-1)$ fault-free PEs with the remaining as a block of faulty PEs. The structure now has blocks of $(n-1)$ fault-free PEs that are disconnected from other fault-free blocks. By Definition 9, it is not possible to compute $f(X)$ for which $\|X\| = n$. The total number of faults is, therefore, $\lfloor N/(n-1+g_k) \rfloor g_k + Min\{0, [N \ mod \ (n-1+g_k)] - n + 1\}$. □

Now what remains for investigation is the the case where $G' \subset G = \{1, 2, \ldots, g_k\}$; that is, the link configuration does not contain all the bypass links up to size g_k. Let W_F and W'_F be the largest fault window for G and G' respectively. By Corollary 1, $W'_F \geq W_F$. Consider the case where $W'_F > W_F$, and let F_i be the catastrophic fault pattern consisting of g_k faults contained in the window W'_F. Since $W'_F > g_k$, there exists at least one non-faulty PE in F_i. Let l and r be the number of non-faulty nodes that can be accessed in F_i from left side and right side respectively. Obviously, $l, r \geq 0$.

Corollary 11 *Let $G' \subset G = \{1, 2, \ldots, g_k\}$. For a problem of size, $\|X\| = n$, an one-dimensional array with a total of N PEs, bidirectional link configuration G', and provision of miraculous I/O facilities is $(\beta, 0)-$ Fault Tolerant, where*

$$\beta = \lfloor \frac{N-n+l+1}{n-1+g_k} \rfloor g_k + Min\{0, [(N-n+l+1) mod(n-1+g_k)] - n + r + 1\}.$$

Proof: From left to right of the one-dimensional array, label the array as follows. Mark the first $n-l-1$ PEs be marked non-faulty. In the remaining array of size $N-n+l+1$ mark alternatinely the pattern in W'_F and $n+g_k-1$ fault-free PEs until $\lfloor (N-n+l+1)/(n-1+g_k) \rfloor$ such patterns are seen (see Figure 16). Out of the remaining $[(N-n+l+1) \ mod \ (n-1+g_k)]$ PEs, if this number is greater than $(n-r+1)$, there should be a block of $(n-r+1)$ fault-free PEs with the remaining as a block of faulty PEs. The array now has blocks of $(n-1)$ fault-free PEs that are disconnected from other fault-free blocks. By Definition 9, it is not possible to compute $f(X)$ for which $\|X\| = n$. Counting the number of faults in the labelled array carefully, it is clear that the total number of faults is

$$\lfloor \frac{N-n+l+1}{n-1+g_k} \rfloor g_k + Min\{0, [(N-n+l+1) mod(n-1+g_k)] - n + r + 1\}. \quad \square$$

6 Conclusions

Identification, characterization and construction of fault patterns that are catastrophic for linear systolic arrays are discussed in this paper. The paper also discusses the effects of processor failures in one-dimensional processing arrays on computations requiring a fixed number of processors. Catastrophic fault patterns are used to derive fault-tolerant characteristics in terms of the minimum number of faults required to cause a catastrophic failure in such structures. Some of the preliminary results of this paper appear in [18]. Results for two-dimensional arrays can be found in [17,19].

From a practical viewpoint, the results allow to prove some answers to questions about the "guaranteed" level of fault tolerance of a design, something not previously considered. Guaranteed fault tolerance indicates positive answers to questions such as

a) will the system withstand up to k faults always regardless of how and where they occur?

b) will the system withstand up to k faults with probability ϵ regardless of how and where the faults occur?

The results in this paper provide a set of tools which can be employed in

1. assessing the fault tolerance effectiveness of a design; this can be done by specifying the minimum number of faults which the design cannot be guaranteed to withstand,
2. testing whether a design meets the specified fault tolerance requirements; this can be achieved by comparing the requirements with the ones derived using the properties of the catastrophic fault patterns, and
3. determining redundancy requirement for the designer to meet a desired level of fault tolerance; this can be done by determining the minimal link configuration for which no catastrophic fault patterns exist below the specified amount of failure.

Furthermore, the results presented here can help to usefully incorporate knowledge of the application field into the design process. In particular, knowledge of the type and distribution of faults occurring in the application field can be used to determine for which design those patterns are catastrophic; thus, the designer can remove those designs from further consideration (even though, without that knowledge, they might have been viable choices).

7 References

[1] K. P. Belkhale and P. Banerjee, "Reconfiguration Strategies in VLSI Processor Arrays," in *Proc. Int'l Conf. on Computer Design*, 1988, pp. 418-421

[2] M. Chean and J. A. B. Fortes, "A Texanomy of Reconfiguration Techniques for Fault-Tolerant Processor Arrays," in *IEEE Computer*, Vol. 23, No. 1, Jan. 1990, pp. 55-69.

[3] F. Distante, F. Lombardi, and D. Sciuto, "Array Partitioning: A Methodology for Reconfigurability and Reconfiguration Problems," in *Proc. Int'l Conf. on Computer Design*, 1988, pp. 564-567.

[4] P. Franzon, "Interconnect Strategies for Fault Tolerant 2D VLSI Arrays," in *Proc. Int'l Conf. on Computer Design*, 1986, pp. 230-233.

[5] J. W. Greene and A. E. Gamal, "Configuration of VLSI Arrays in the Presence of Defects," *Journal of the ACM*, Vol. 31, No. 4, Oct. 1984, pp. 694-717.

[6] S. H. Hosseini, "On Fault-Tolerant Structure, Distributed Fault-Diagnosis, Reconfiguration, and Recovery of the Array Processors," *IEEE Trans. on Computers*, Vol. C-38, No. 7, July 1989, pp. 932-942.

[7] J. H. Kim and S. M. Reddy, "On the Design of Fault-Tolerant Two-Dimensional Systolic Arrays for Yield Enhancement," *IEEE Trans. on Computers*, Vol. C-38, No. 4, April 1989, pp. 515-525.

[8] I. Koren and D. K. Pradhan, "Introducing Redundancy into VLSI Designs for Yield and Performance Enhancement," in *Proc. 15th Int'l Conf. on Fault-Tolerant Computers*, 1985, pp. 330-335.

[9] R. Kumar and F. G. Gray, "Fault Tolerant 1-Dimensional Cellular Structure," in *1984 Conf. on Distributed Computing*, May 1984, pp. 472-483.

[10] S. Kuo and W. K. Fuchs, "Efficient Spare Allocation for Reconfigurable Arrays," *IEEE Design and Test*, Feb. 1987, pp. 24-31.

[11] H. T. Kung and M. Lam, "Fault-Tolerant VLSI Systolic Arrays and Two-Level Pipelining," *Journal of Parallel and Distributed Processing*, Aug. 1984, pp. 32-63.

[12] S. Y. Kung, S. N. Jean, and C. W. Chang, "Fault-Tolerant Array Processors Using Single-Track Switches," *IEEE Trans. on Computers*, Vol. C-38, No. 4, April 1989, pp. 501-514.

[13] T. Leighton and C. E. Leiserson, "Wafer-Scale Integration of Systolic Arrays," *IEEE Trans. on Computers*, Vol. C-34, No. 5, May 1985, pp. 448-461.

[14] F. Lombardi, R. Negrini, M. Sami, and R. Stefanelli, "Reconfiguration of VLSI Arrays: A Covering Approach," in *Proc. 17th Int'l Conf. on Fault-Tolerant Computers*, 1987, pp. 251-256.

[15] T. E. Mangir and C. S. Raghavendra, "Issues in the Implementation of Fault-Tolerant VLSI and WSI Systems," in in *Proc. Int'l Conf. on Computer Design*, 1984, pp. 95-100.

[16] W. R. Moore, "A Review of Fault-Tolerant Techniques for the Enhancement of Integrated Circuit Yield," *Proc. of the IEEE*, Vol. 74, No. 5, May 1986, pp. 684-698.

[17] A. Nayak, "On Reconfigurability of some Regular Architectures," Ph.D Thesis, Dept. Systems & Computer Engineering, Carleton University, Ottawa, Canada, 1991.

[18] A. Nayak, N. Santoro and R. Tan, "Fault-Intolerance of Reconfigurable Systolic Arrays," in *Proc. 20th Int'l Conf. on Fault-Tolerant Computers*, 1990.

[19] A. Nayak and N. Santoro, "Bounds on Performance of VLSI Processor Arrays," in *5th Int'l Parallel Processing Symposium*, Anaheim, California, May 1991.

[20] R. Negrini and R. Stefanelli, "Algorithms for Self-Reconfiguration of Wafer-Scale Regular Arrays," in *Proc. Int'l Conf. on Circuits and Systems*, 1985, pp. 190-196.

[21] S. P. Popli and M. A. Bayoumi, "Fault Diagnosis and Reconfiguration for Reliable VLSI Arrays," in *Proc. Conf. on Computers and Communications*, Phoenix, 1988, pp. 69-73.

[22] C. S. Raghavendra, "Fault Tolerance in Regular Network Architectures," *IEEE Micro*, Vol. 4, No. 6, Dec. 1984, pp. 44-53.

[23] A. L. Rosenberg, "The Diogenes Approach to Testable Fault-Tolerant Arrays of Processors," *IEEE Trans. on Computers*, Vol. C-32, No. 10, Oct. 1983, pp.902-910.

[24] M. Wang, M. Cutler, and S. Y. H. Su, "Reconfiguration of VLSI/WSI Array Processors with Two-Level Redundancy," *IEEE Trans. on Computers*, Vol. C-38, No. 4, April 1989, pp. 547-554.

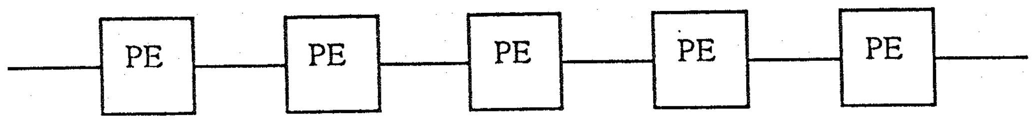

Figure 1: A linear systolic array of PEs

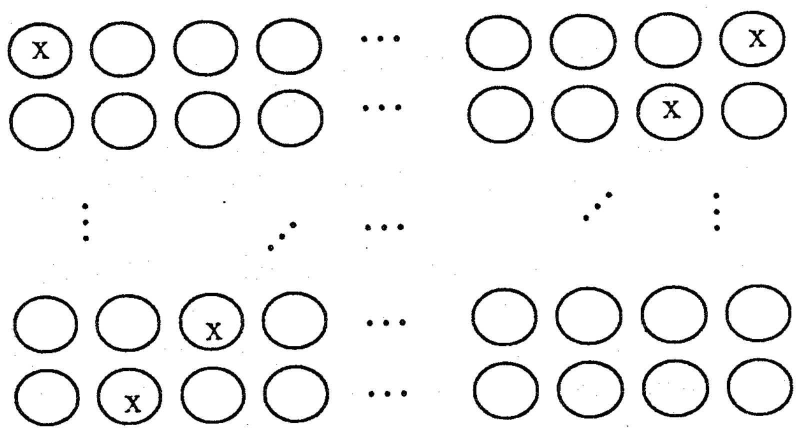

Figure 2: Distribution of "x"s for unidirectional $G = \{1, g_k\}$

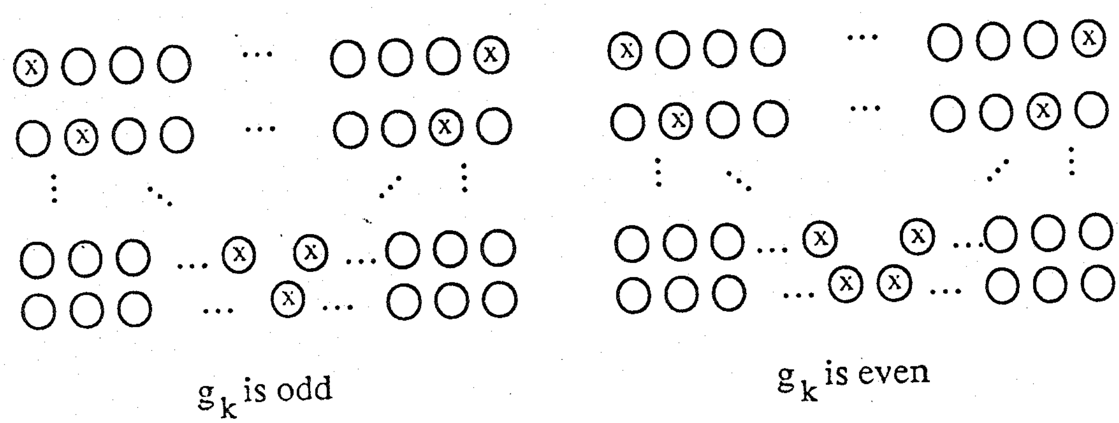

Figure 3: Distribution of "x"s for bidirectional $G = \{1, g_k\}$

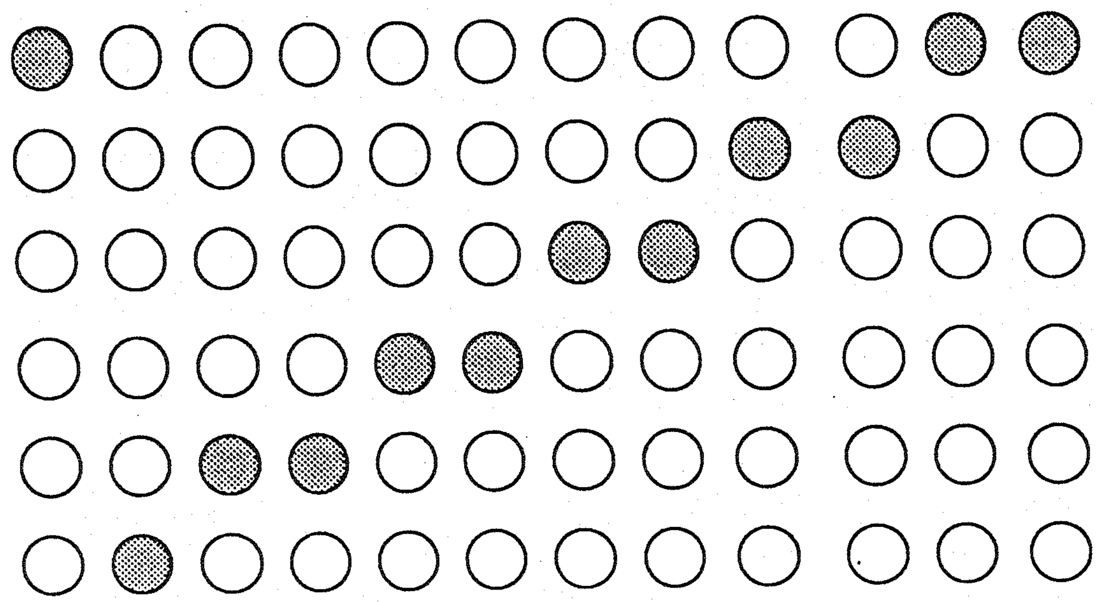

Figure 4: A catastrophic fault pattern for $G = \{1, 2, 12\}$

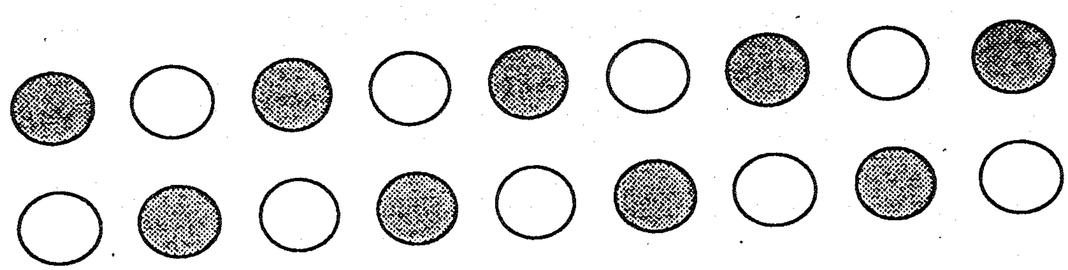

Figure 5: A catastrophic fault pattern for $G = \{1, 7, 9\}$

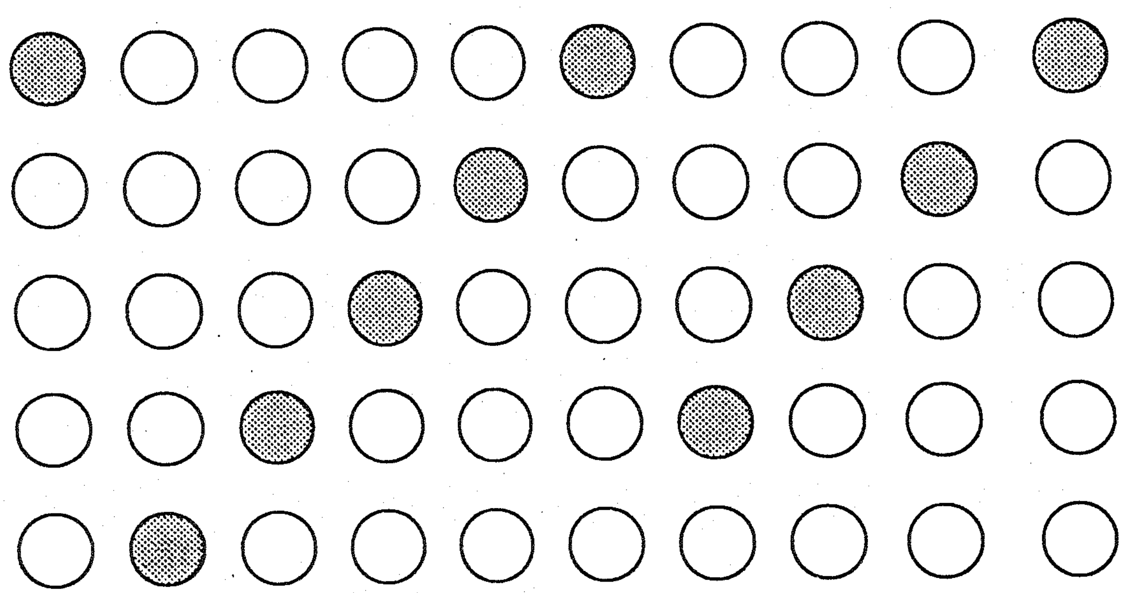

Figure 6: A catastrophic fault pattern for $G = \{1, 5, 10\}$

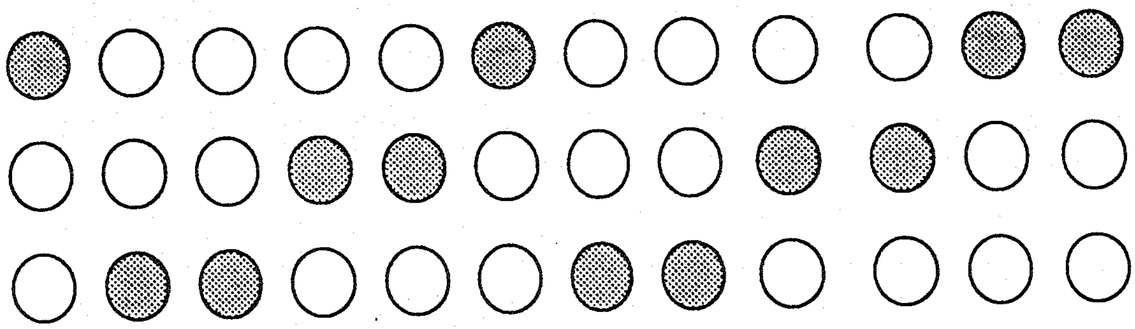

Figure 7: A catastrophic fault pattern for $G = \{1, 7, 12\}$

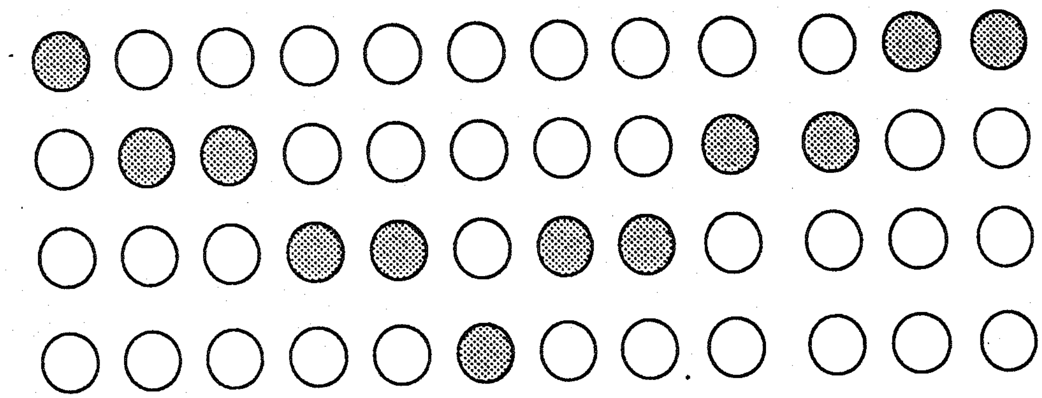

Figure 8: A catastrophic fault pattern for bidirectional $G = \{1, 2, 12\}$

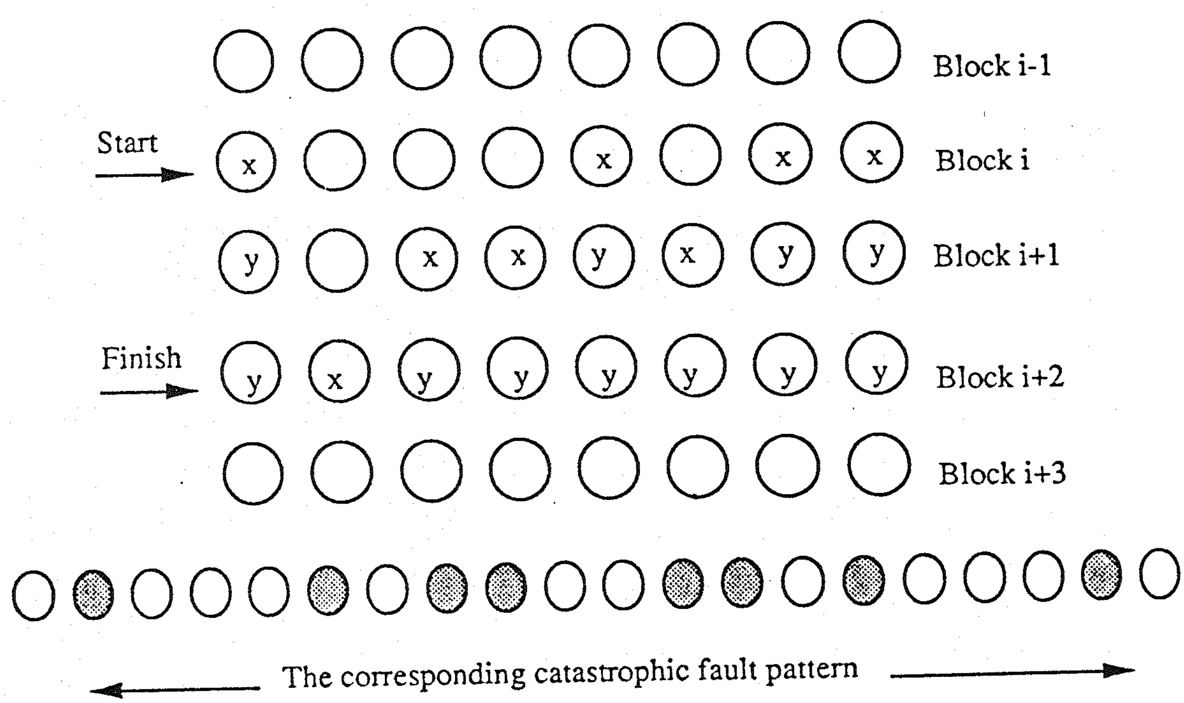

Figure 9: A catastrophic fault pattern for unidirectional $G = \{1, 2, 4, 8\}$

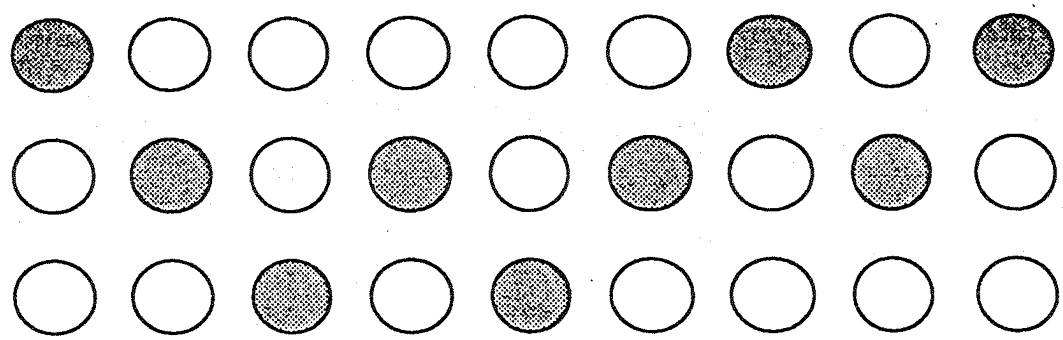

Figure 10: A catastrophic fault pattern for bidirectional $G = \{1, 3, 9\}$

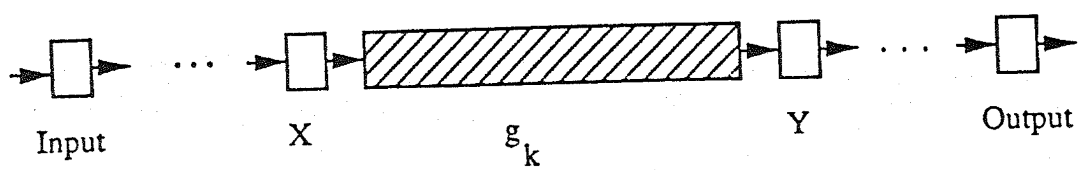

Figure 11: 1-D array of PEs with a block fault

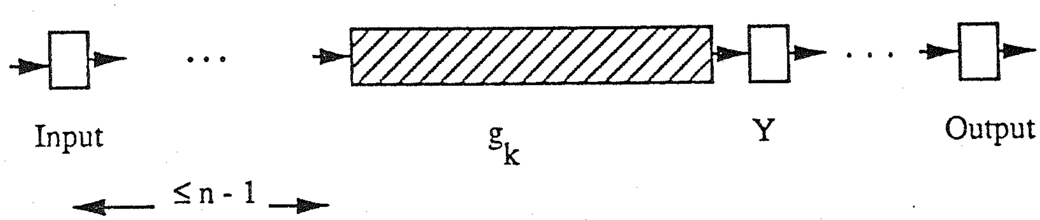

Figure 12: 1-D array with block fault in input side

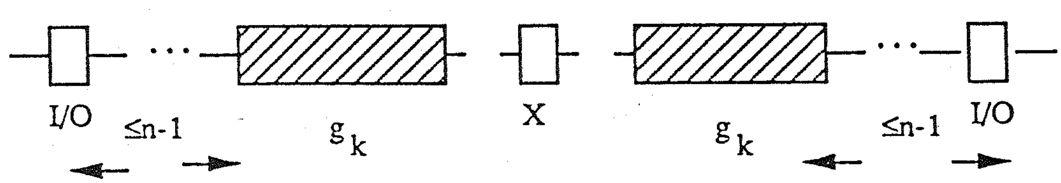

Figure 13: 1-D bidirectional array with blocked I/O

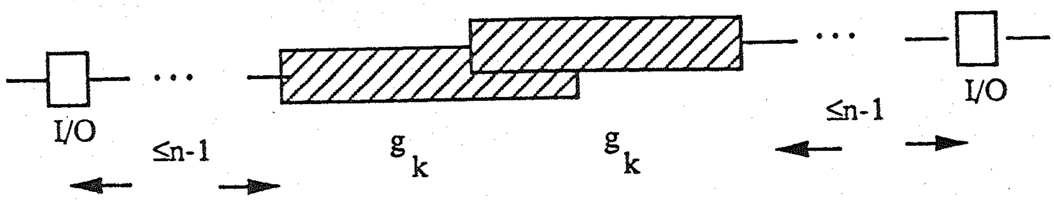

Figure 14: 1-D bidirectional array with $< 2g_k$ faults

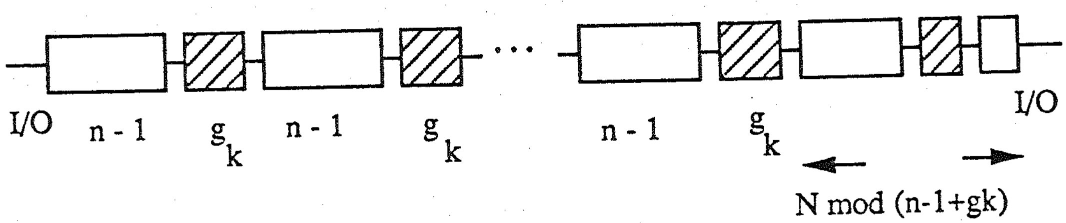

Figure 15: 1-D array with alternating block faults

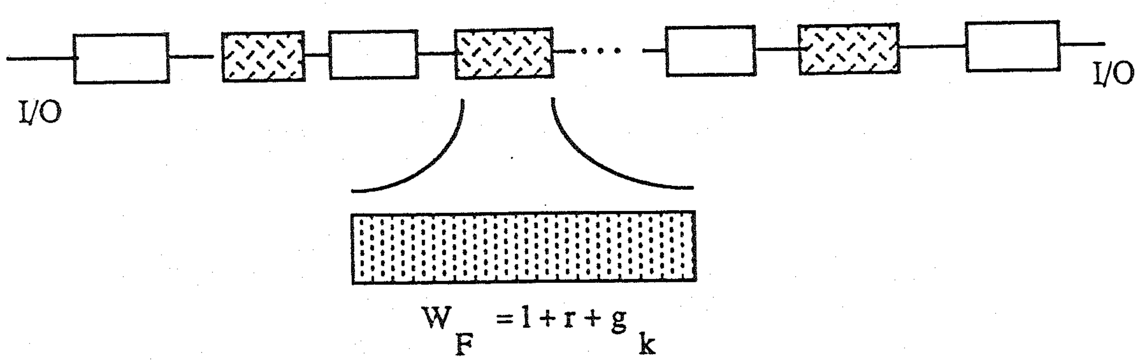

Figure 16: 1-D array with alternating fault patterns

School of Computer Science, Carleton University
Recent Technical Reports

SCS-TR-158 (revised)
Parallel Fractional Cascading on a Hypercube Multiprocessor
Frank Dehne, Afonso Ferreira and Andrew Rau-Chaplin, May 1989 (Revised April 1990)

SCS-TR-159
Epsilon-Optimal Stubborn Learning Mechanisms
J.P.R. Christensen and B.J. Oommen, June 1989

SCS-TR-160
Disassembling Two-Dimensional Composite Parts Via Translations
Doron Nussbaum and Jörg-R. Sack, June 1989

SCS-TR-161 (revised)
Recognizing Sources of Random Strings
R.S. Valiveti and B.J. Oommen, January 1990
Revised version of SCS-TR-161 "On the Data Analysis of Random Permutations and its Application to Source Recognition", published June 1989

SCS-TR-162
An Adaptive Learning Solution to the Keyboard Optimization Problem
B.J. Oommen, R.S. Valiveti and J. Zgierski, October 1989

SCS-TR-163
Finding a Central Link Segment of a Simple Polygon in O(N Log N) Time
L.G. Alexandrov, H.N. Djidjev, J.-R. Sack, October 1989

SCS-TR-164
A Survey of Algorithms for Handling Permutation Groups
M.D. Atkinson, January 1990

SCS-TR-165
Key Exchange Using Chebychev Polynomials
M.D. Atkinson and Vincenzo Acciaro, January 1990

SCS-TR-166
Efficient Concurrency Control Protocols for B-tree Indexes
Ekow J. Otoo, January 1990

SCS-TR-167
A Hierarchical Stochastic Automaton Solution to the Object Partitioning Problem
B.J. Oommen, January 1990

SCS-TR-168
Adaptive List Organizing for Non-stationary Query Distributions. Part I: The Move-to-Front Rule
R.S. Valiveti and B.J. Oommen, January 1990

SCS-TR-169
Trade-Offs in Non-Reversing Diameter
Hans L. Bodlaender, Gerard Tel and Nicola Santoro, February 1990

SCS-TR-170
A Massively Parallel Knowledge-Base Server using a Hypercube Multiprocessor
Frank Dehne, Afonso Ferreira and Andrew Rau-Chaplin, April 1990

SCS-TR-171
Parallel Processing of Quad Trees on the Hypercube (and PRAM)
Frank Dehne, Afonso Ferreira and Andrew Rau-Chaplin, April 1990

SCS-TR-172
A Note on the Load Balancing Problem for Coarse Grained Hypercube Dictionary Machines
Frank Dehne and Michel Gastaldo, May 1990

SCS-TR-173
Self-Organizing Doubly-Linked Lists
R.S. Valiveti and B.J. Oommen, May 1990

SCS-TR-174
A Presortedness Metric for Ensembles of Data Sequences
R.S. Valiveti and B.J. Oommen, May 1990

SCS-TR-175
Separation of Graphs of Bounded Genus
Ljudmil G. Aleksandrov and Hristo N. Djidjev, May 1990

SCS-TR-176
Edge Separators of Planar and Outerplanar Graphs with Applications
Krzystof Diks, Hristo N. Djidjev, Ondrej Sykora and Imrich Vrto, May 1990

SCS-TR-177
Representing Partial Orders by Polygons and Circles in the Plane
Jeffrey B. Sidney and Stuart J. Sidney, July 1990

SCS-TR-178 **Determining Stochastic Dependence for Normally Distributed Vectors Using the Chi-squared Metric**
R.S. Valiveti and B.J. Oommen, July 1990

SCS-TR-179 **Parallel Algorithms for Determining K-width-Connectivity in Binary Images**
Frank Dehne and Susanne E. Hambrusch, September 1990

SCS-TR-180 **A Workbench for Computational Geometry (WOCG)**
P. Epstein, A. Knight, J. May, T. Nguyen, and J.-R. Sack, September 1990

SCS-TR-181 **Adaptive Linear List Reorganization under a Generalized Query System**
R.S. Valiveti, B.J. Oommen and J.R. Zgierski, October 1990

SCS-TR-182 **Breaking Substitution Cyphers using Stochastic Automata**
B.J. Oommen and J.R. Zgierski, October 1990

SCS-TR-183 **A New Algorithm for Testing the Regularity of a Permutation Group**
V. Acciaro and M.D. Atkinson, November 1990

SCS-TR-184 **Generating Binary Trees at Random**
M.D. Atkinson and J.-R. Sack, December 1990

SCS-TR-185 **Uniform Generation of Combinatorial Objects in Parallel**
M.D. Atkinson and J.-R. Sack, January 1991

SCS-TR-186 **Reduced Constants for Simple Cycle Graph Separation**
Hristo N. Djidjev and Shankar M. Venkatesan, February 1991

SCS-TR-187 **Multisearch Techniques for Implementing Data Structures on a Mesh-Connected Computer**
Mikhail J. Atallah, Frank Dehne, Russ Miller, Andrew Rau-Chaplin, and Jyh-Jong Tsay, February 1991

SCS-TR-188 Out of print **Generating and Sorting Jordan Sequences**
Alan Knight and Jörg-Rüdiger Sack, March 1991

SCS-TR-189 **Probabilistic Estimation of Damage from Fire Spread**
Charles C. Colbourn, Louis D. Nel, T.B. Boffey and D.F. Yates, April 1991

SCS-TR-190 **Coordinators: A Mechanism for Monitoring and Controlling Interactions Between Groups of Objects**
Wilf R. LaLonde, Paul White, and Kevin McGuire, April 1991

SCS-TR-191 **Towards Decomposable, Reusable Smalltalk Windows**
Kevin McGuire, Paul White, and Wilf R. LaLonde, April 1991

SCS-TR-192 **PARASOL: A Simulator for Distributed and/or Parallel Systems**
John E. Neilson, May 1991

SCS-TR-193 **Realizing a Spatial Topological Data Model in a Relational Database Management System**
Ekow J. Otoo and M.M. Allam, August 1991

SCS-TR-194 **String Editing with Substitution, Insertion, Deletion, Squashing and Expansion Operations**
B John Oommen, September 1991

SCS-TR-195 **The Expressiveness of Silence: Optimal Algorithms for Synchronous Communication of Information**
Una-May O'Reilly and Nicola Santoro, October 1991

SCS-TR-196 **Lights, Walls and Bricks**
J. Czyzowicz, E. Rivera-Campo, N. Santoro, J. Urrutia and J. Zaks, October 1991

SCS-TR-197 **A Brief Survey of Art Gallery Problems in Integer Lattice Systems**
Evangelos Kranakis and Michel Pocchiola, November 1991

SCS-TR-198 **On Reconfigurability of Systolic Arrays**
Amiya Nayak, Nicola Santoro, and Richard Tan, November 1991