Technical Report CUCS-032-93

Catastrophic Faults in Reconfigurable Linear Arrays of Processors

Roberto De Prisco* **Alfredo De Santis†**

Abstract

In regular architectures of identical processing elements, a widely used technique to improve the reconfigurability of the system consists of providing redundant processing elements and mechanisms of reconfiguration.

In this paper we consider linear arrays of processing elements, with unidirectional bypass links of length g. We count the number of particular sets of faulty processing elements. We show that the number of catastrophic faults of g elements is equal to the $(g-1)$-th Catalan number. We also provide algorithms to rank and unrank all catastrophic sets of g faults. Finally, we describe a linear time algorithm that generate all such sets of faults.

Key phrases: algorithms, catastrophic fault patterns, fault tolerance, VLSI linear arrays, reliability analysis.

*Department of Computer Science, Columbia University, New York, N.Y. 10027

†Dipartimento di Informatica ed Applicazioni, Universitá di Salerno, 84081 Baronissi (SA) - Italy

1 Introduction

Linear arrays of processors are regular architectures consisting of a large number of identical processing elements connected in a regular fashion: each processing element is connected with the subsequent processing element. Since the number of processing elements is very large, the probability that a set of processing elements becames faulty is not small, hence we have to provide fault-tolerant mechanisms to avoid that faulty processing elements take part in the computation. A widely used technique to achieve reconfigurability consists of providing redundant processing elements, called spares, and additional connections, called bypass links. Bypass links are links that connect each processor with a processor at a fixed distance greater than 1. A reconfiguration algorithm has to avoid faulty processing elements using spares and additional connections. However, there are sets of faulty processing elements for which no reconfiguration strategy is possible. Such sets are called catastrophics. If we have to reconfigure a system when a faulty set occurs, it is necessary to known if the set is catastrophic or not. Therefore it is important to study the properties of catastrophic sets.

Nayak, Santoro and Tan [9] proved that a catastrophic set must contain a number of faulty processing elements which is greater or equal than the length of the longest bypass link. They analyze catastrophic sets having the minimal number of faults and describe algorithms for constructing a catastrophic set. Nayak, Pagli and Santoro [7] describe algorithms for testing whether a set of faults is catastrophic or not.

Given a linear array with a set of bypass links, an important problem is to count the number of catastrophic sets. The knowledge of the number of catastrophic sets enables us to estimate the probability that the system operates correctly. Pagli and Pucci [10] proved tight upper and lower to the number $F^B(g)$ of catastrophic sets of size g for a linear array with one bidirectional bypass link of length g. In particular they proved that $F^B(g) = \Theta(3^g/g^{3/2})$. They also proved that $F^U(g) = O(10^g/g^{3/2})$, where $F^U(g)$ is the number of catastrophic sets of size g for a linear array with one unidirectional bypass link of length g.

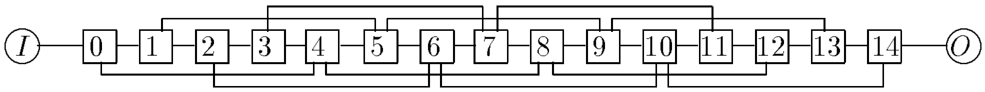
In this paper we consider linear arrays with bypass unidirectional links of length g. We compute the exact number of catastrophic sets of size g. We prove that $F^U(g)$ is equal to the $(g-1)$-th Catalan number. This enables us to prove that $F^U(g) = \Theta(4^g/g^{3/2})$. In order to characterize these catastrophic sets, we also give a classification of all the catastrophic sets: we rank and unrank all catastrophic sets and we provide an efficient algorithm that generates such sets. The rank of catastrophic sets turns to be useful, for example, when one wants to choose at random a catastrophic set.

This paper is organized as follows. In Section 3 we count the number of particular fault sets. As a special case we obtain that the number of catastrophic sets for a linear array with unidirectional bypass link of length g is the $(g-1)$-th Catalan number. In Section 4 we rank and unrank all such catastrophic sets. In Section 5 we describe and analyze a linear time algorithm that generate all the catastrophic sets.

2 Preliminaries

In this section we give preliminaries, definitions and some known results. We refer the reader to [6], [7], [9], [10], for a justification of the definitions and for proofs of the results.

Let $A = \{p_0, p_1, \ldots, p_{N-1}\}$ be a linear array of processing elements, which are connected by regular links (p_i, p_{i+1}) and by bypass links (p_i, p_{i+g}) of fixed length $g \geq 2$, both unidirectional. We refer to this structure as a linear array with link redundancy g or simply as a linear array, when g is clear from the context or immaterial. The following picture shows a linear array of 15 processing elements.



We assume the presence of an external input processor, called I, which is connected with $p_0, p_1, \ldots, p_{g-1}$, and an external output processor, called O, which is connected with $p_{N-g}, p_{N-g+1}, \ldots, p_{N-1}$. These special connections of I and O give the same degree of reconfigurability to all processing elements, and enable us to focalize our attention on that part of A beginning at the first faulty processor and ending at the last faulty processor, assuming that there are more than g processors before the first fault and after the last fault. I and O always operate correctly. In other words we can assume that A is an infinite array: no matter how many processors there are before the first fault and after the last fault. The connections with I and O, except the regular ones, are not drawn in the previous picture.

For a linear array of size N and any link redundancy, a *fault pattern* F starting at a fixed p_{f_0} is a set of integers $F = \{f_0, f_1, \ldots, f_{m-1}\}$, where $f_{i-1} < f_i$ for $1 \leq i \leq m$ and $f_{m-1} \leq N$. Processor p_{f_i} is faulty if and only if $f_i \in F$. The cardinality of F is m.
Given a linear array A, a fault pattern F is *catastrophic* for A if and only if no path exists between I and O, once the faulty processors $p_i, i \in F$, and their links are removed.

We denote a fault pattern by FP and a catastrophic fault pattern by CFP.
The special connections of I and O make indistiguishable each processor from others in the sense that any translation of a fault pattern does not affect the property of catastrophe of the pattern. Therefore we assume, without loss of generality, that the first fault of any pattern is p_0.

A catastrophic fault pattern F for A must contain at least g fault processors. As done in [9],[10], we consider only fault pattern of cardinality g, so, in general, $F = \{0, f_1, \ldots, f_{g-1}\}$.

The *width* w_F of a fault pattern F is defined to be the number of processors between and including the first and the last fault processor in F, that is, $w_F = f_{g-1} - f_0 + 1$.

A necessary condition for a fault pattern F to be catastrophic is $g \leq w_F \leq (g-1)^2 + 1$ [9].

A convenient way to represent a fault pattern F, starting at the fixed processor p_0, is the *matrix representation* [7]. The fault pattern F is represented as a boolean matrix W of size $(g-1) \times g$, defined by

$$W[i,j] = \begin{cases} 1 & \text{if } (ig+j) \in F \\ 0 & \text{otherwise.} \end{cases}$$

Example 1. Consider the case $g = 6$ and $F = \{0, 5, 10, 14, 15, 19\}$. The matrix representation of F is the following

$$W = \begin{array}{|c|c|c|c|c|c|} \hline 1 & 0 & 0 & 0 & 0 & 1 \\ \hline 0 & 0 & 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 1 & 1 & 0 & 0 \\ \hline 0 & 1 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 \\ \hline \end{array}$$

Observe that in the matrix W each regular link corresponds either to two consecutive elements in the same row or to the last element in a row with the first element in the following row, whereas each bypass link corresponds to two consecutive elements in the same columns. For a CFP F, W contains only one entry filled by 1 for each column. Indeed, if there were a column of W with two 1, then there would be a column of W with only 0 entries, as F has cardinality g. Using the bypass links of this column we can pass over the fault zone, contradicting the hypothesis that F is catastrophic. Therefore a CFP can be represented by the set of row indices corresponding to the entry 1 in columns. Formally, the *row representation* of a CFP F is the *g-upla* $(r_0, r_1, \ldots, r_{g-1})$, where each r_i is the unique integer such that $W[r_i, i] = 1$. Another convenient way to represent a CFP is the *catastrophic sequence* [10]. A catastrophic fault pattern is represented as a sequence of $g-1$ integer *moves* $(m_1, m_2, \ldots, m_{g-1})$, where m_i represents the distance from the row index of the element set to 1 in column $i-1$, to the one in column i. Formally, we have that $m_i = r_{i-1} - r_i$.

Example 2. Let $g = 6$ and $F = (0, 5, 10, 14, 15, 19)$. Its catastrophic sequence is $(-3, 1, 0, 1, 1)$ and its row representation is $(0, 3, 2, 2, 1, 0)$.

3 Counting catastrophic faults

In this section we count the number of sets of faulty processors, starting at the fixed processor p_0, that satisfy particular conditions. We will use this counting in order to rank and unrank all the CFPs (Section 4) and to design an algorithm that generates all the CFPs (Section 5). The counting gives us the number of catastrophic fault patterns, of size g, which turns out to be the $(g-1)$-*th* Catalan number. An alternative and more simple proof of this fact is also provided.

To prove our results we need the following theorem.

Theorem 1 *[9] Necessary and sufficient conditions for a fault pattern F of cardinality g to be catastrophic for a unidirectional array with link redundancy g are:*

1. $W[0, 0] = W[0, g-1] = 1$
2. *for* $1 \le k \le g-2$, *if* $W[h, k-1] = 1$ *then only one among* $W[h-1, k], W[h, k], \ldots, W[g-1, k]$ *is equal to 1*
3. *for* $1 \le k \le g-2$, *if* $W[h, k+1] = 1$ *then only one among* $W[0, k], W[1, k], \ldots, W[h+1, k]$ *is equal to 1.*

Observe that Theorem 1 is equivalent to the following proposition.

Proposition 1 *Necessary and sufficient conditions for a fault pattern F of cardinality g to be catastrophic for a unidirectional array with link redundancy g are:*

1. $W[0,0] = W[0,g-1] = 1$
2. *for* $0 \leq k \leq g-2$, *if* $W[h,k+1] = 1$ *then only one among* $W[0,k], W[1,k], \ldots, W[h+1,k]$ *is equal to 1.*

Making use of the concepts of sequence of moves and of row representation, Proposition 1 can be rewritten as follows.

Proposition 2 *[10] Necessary and sufficient conditions to have that* $(m_1, \ldots, m_{g-1})$ *is the catastrophic sequence of a CFP for a unidirectional linear array with link redundancy* g *are:*

1. $m_i \leq 1$ *for* $i = 1, \ldots, g-1$
2. $\sum_{i=1}^{k} m_i \leq 0$ *for* $k = 1, \ldots, g-2$
3. $\sum_{i=1}^{g-1} m_i = 0$.

Proposition 3 *Necessary and sufficient conditions to have that* $(r_0, r_1, \ldots, r_{g-1})$ *is the row representation of a CFP for a unidirectional linear array with link redundancy* g *are:*

1. $r_0 = r_{g-1} = 0$
2. *for* $0 \leq c \leq g-2$, $r_c \leq r_{c+1} + 1$.

Now, we introduce the notion of (i,j)*-fault pattern.*

Definition 1 *An* (i,j)*-fault pattern, for* $i \geq 0$ *and* $j \geq 1$, *is a fault pattern of cardinality* $j+1$, *whose matrix representation satisfies*

1. $W[0,0] = 1$
2. *for* $0 \leq k \leq j-1$, *if* $W[h,k+1] = 1$ *then only one among* $W[0,k], W[1,k], \ldots, W[h+1,k]$ *is 1*
3. $W[i,j] = 1$.

Roughly speaking, an (i,j)-fault pattern, (i,j)-FP for short, is a piece of a CFP, characterized by a matrix representation equal to that of the CFP up to the j-th column, and filled by zeroes from column $j+1$ to column $g-1$. Notice that the definition of (i,j)-FP is independent from g.

Example 3. Consider the fault pattern $F = \{0, 14, 19\}$, with link redundancy $g = 6$. The matrix representing F is:

$$W = \begin{array}{|c|c|c|c|c|c|} \hline 1 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 1 & 0 & 0 & 0 \\ \hline 0 & 1 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 \\ \hline \end{array}$$

F is a $(2,2)$-FP. If we add to F the set $\{5, 10, 15\}$, F becames catastrophic, for a linear array with link redundancy g. Other $(2,2)$-FPs are $\{0, 7, 14\}$, $\{0, 1, 14\}$, $\{0, 13, 14\}$,$\{0, 14, 25\}$.

Define $N_{i,j}$, for $i \geq 0, j \geq 0$, as the number of (i,j)-FPs. Next, we derive and solve a recurrence relation for $N_{i,j}$.

Lemma 1 *Integers $N_{i,j}$ satisfy the relation*

$$N_{i,j} = N_{0,j-1} + \ldots + N_{i,j-1} + N_{i+1,j-1} \qquad for \quad i \geq 0, j \geq 2 \tag{1}$$

and

$$N_{i,1} = 1 \qquad for \quad i \geq 0. \tag{2}$$

Proof. An $(i,1)$-FP has two fault processors, namely p_0 and p_{ig+1}. Thus, there is a unique $(i,1)$-FP, which implies $N_{i,1} = 1$.
Condition *2.* in Proposition 1 tells us that by adding the processor p_{ig+j} to a $(k, j-1)$-FP, where $0 \leq k \leq i+1$, we obtain an (i,j)-FP and that any (i,j)-FP can be constructed in this way. Thus, the number of (i,j)-FPs is the sum of the number of $(k, j-1)$-FPs for $0 \leq k \leq i+1$.
□
Notice that $N_{0,0} = 1$ and $N_{i,0} = 0$ for $i > 0$, so that (1) is true for $j = 1$, too.

Lemma 2 *The solution of the recurrence (1)-(2) is*

$$N_{i,j} = (i+2)\frac{(2j+i-1)!}{(j+i+1)!(j-1)!} \tag{3}$$

for $i \geq 0$ and $j \geq 1$.

Proof. We prove the formula by induction. Let $j = 1$, we get from (3) that $N_{i,1} = 1$. Fix a row $r \geq 0$ and a column $c \geq 2$, and suppose that (3) is true for every $N_{i,j}$ in the previous column, i.e., for $i \geq 0$ and $j < c$, and for all previous elements on the column c, i.e., for $i < r$ and $j = c$. For the induction step we distinguish between two cases: $r = 0$ and $r > 0$.
If $r = 0$, from (1), we have:

$$\begin{aligned} N_{0,c} &= N_{0,c-1} + N_{1,c-1} \\ &= 2\frac{(2c-3)!}{c!(c-2)!} + 3\frac{(2c-2)!}{(c+1)!(c-2)!} \\ &= 2\frac{(2c-1)!}{(c+1)!(c-1)!}. \end{aligned}$$

Whereas if $r > 0$, from (1) we have:

$$\begin{aligned} N_{r,c} &= N_{0,c-1} + \ldots + N_{r,c-1} + N_{r+1,c-1} \\ &= N_{r-1,c} + N_{r+1,c-1} \\ &= (r+1)\frac{(2c+r-2)!}{(r+c)!(c-1)!} + (r+3)\frac{(2c+r-2)!}{(r+c+1)!(c-2)!} \\ &= (r+2)\frac{(2c+r-1)!}{(r+c+1)!(c-1)!}. \end{aligned}$$

□

Observe that there is an isomorphism between CFPs of a linear array with link redundancy g and $(0, g-1)$-FPs. This is straightforward from the definition of (i,j)-FP and from Proposition 1. Thus $N_{0,g-1}$ is equal to the number $F^U(g)$ of unidirectional CFPs for a linear array with link redundancy g. Hence

$$F^U(g) = N_{0,g-1} \tag{4}$$

and

$$F^U(g) = 2\frac{(2g-3)!}{g!(g-2)!} = \frac{1}{g}\binom{2g-2}{g-1}.$$

Next theorem provides a more simple proof of this fact.

Theorem 2 *The number of CFPs for a linear array with unidirectional bypass links of length g is the $(g-1)$-th Catalan number, i.e.,*

$$F^U(g) = \frac{1}{g}\binom{2g-2}{g-1}.$$

Proof. It is well known that the $(g-1)$-*th* Catalan number represents the number of well formed expressions over the alphabet $\{(,)\}$ of length $2g-2$ (for example see [4]). Recall that a well formed expression of length $2k$ is a sequence of k "(" and k ")" that satisfy the following property: for each $i, 1 \leq i \leq 2k$, the number of "(" among the first i letters of the sequence is greater than or equal to the number of ")". In order to prove the theorem it is sufficient to show an isomorphism between the set of CFPs and the set of well formed expressions of length $2g-2$. Let F be a CFP and $(m_1, \ldots, m_{g-1})$ its catastrophic sequence. To each integer m_i we associate the string $s(m_i) = ((\ldots(()$, consisting of $1-m_i$ "(" followed by a single ")". To the CFP F we associate the string $s(F)$ obtained by concatenating $s(m_1)s(m_2)\ldots s(m_{g-1})$. As an example, the CFP F considered in the Example 2, whose catastrophic sequence is $(-3, 1, 0, 1, 1)$, has $s(F) = (((())()))$. From Proposition 2 we have that $s(F)$ is a well formed expression. On the other hand, $s(F)$ contains exactly $g-1$ ")", so it is a well formed expression of length $2g-2$. Conversely, every well formed expression of length $2g-2$ can be viewed as a concatenation of $g-1$ strings $s(m'_i)$, $i = 1, \ldots, g-1$. From the definition of well formed expression we have that integers m'_i, $i = 1, \ldots, g-1$, satisfy Proposition 2. □

Using the well known Stirling approximation [4],

$$n! = \sqrt{2\pi n}\left(\frac{n}{e}\right)^n \left(1 + O\left(\frac{1}{12n}\right)\right)$$

we obtain the following asymptotic estimate of the number of CFPs as function of g,

$$F^U(g) = \frac{4^g}{\sqrt{\pi} g^{3/2}} \left(1 + O\left(\frac{1}{g}\right)\right).$$

A concept which will turn to be useful in Section 4 and in Section 5 is the *complement* of an (i, j)-FP.

Definition 2 *A complement of an (i, j)-fault pattern, for a linear array with link redundancy g, for $0 \leq i \leq j \leq g - 1$, is a fault pattern of cardinality $g - j$, whose matrix representation satisfies*

1. $W[0, g-1] = 1$
2. *for* $g - 2 \geq k \geq j + 1$*, if* $W[h, k+1] = 1$ *then only one among* $W[0, k], W[1, k], \ldots, W[h+1, k]$ *is 1*
3. $W[i, j] = 1$.

Informally, a complement of an (i, j)-FP is a piece of a CFP, characterized by a matrix representation filled by zeroes from the first column up to the column $j - 1$ and equal to that of a CFP from the j-th column to column $g - 1$. Notice that the definition of a complement of an (i, j)-FP depends from g.

Example 4. Consider the fault pattern $F = \{5, 10, 14, 15\}$, with bypass links of length $g = 6$. The matrix representing F is:

$$W = \begin{array}{|c|c|c|c|c|c|}\hline 0 & 0 & 0 & 0 & 0 & 1 \\ \hline 0 & 0 & 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 1 & 1 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 \\ \hline \end{array}$$

F is a complement of a $(2, 2)$-FP. If we add to F the set $\{0, 19\}$, F becames catastrophic.

We denote a complement of an (i, j)-FP with $(i, j)^C$-FP. Let $M_{i,j}$ be the number of $(i, j)^C$-FPs. Next we evaluate $M_{i,j}$.

Lemma 3 *Integers $M_{i,j}$ satisfy the relation*

$$M_{i,j} = M_{i-1,j+1} + M_{i,j+1} + \ldots + M_{g-j-1,j+1} \quad for \quad i \geq 0, j \geq 0$$

where $M_{-1,k}$, for $k \geq 0$, is assumed to be 0. Moreover

$$\begin{aligned} M_{0,g-1} &= 1 \\ M_{i,g-1} &= 0 \ for\ i > 0. \end{aligned}$$

Proof. The proof is similar to the proof of Lemma 1. ∎

Next corollary follows immediately from Lemma 3.

Corollary 1 *Integers $M_{i,j}$ satisfy the relations*

$$M_{i,j} = M_{i+1,j} + M_{i-1,j+1} \tag{5}$$

and

$$M_{i,g-1-i} = 1 \quad for \quad i \geq 0. \tag{6}$$

Observe that $M_{0,j} = M_{1,j}$ for $j = 0, 1, \ldots, g-2$, since the number of $(0,j)^C$-FPs is equal to the number of $(1,j)^C$-FPs.

Lemma 4 *The number of $(i,j)^C$-FPs and the number of (i,j)-FPs, for $0 \leq i \leq j \leq g-1$, are related by*

$$M_{i,j} = N_{i-1,g-i-j}.$$

Proof. Define $D_{i,j}$, for $i \geq 0$ and $j > 0$, as follows

$$D_{i,j} = M_{i+1,g-1-(i+j)}.$$

Then, one gets $M_{i,j} = D_{i-1,g-(i+j)}$. Using this fact and (5) we have that

$$\begin{aligned} D_{i,j} &= M_{i+1,g-1-(i+j)} \\ &= M_{i+2,g-1-(i+j)} + M_{i,g-(i+j)} \\ &= D_{i+1,j-1} + M_{i+1,g-(i+j)} + M_{i-1,g-(i+j)+1} \\ &= D_{i+1,j-1} + D_{i,j-1} + M_{i,g-(i+j)+1} + M_{i-2,g-(i+j)+2} \\ & \quad \ldots\ldots \\ & \quad \ldots\ldots \\ &= D_{i+1,j-1} + D_{i,j-1} + D_{i-1,j-1} + \ldots + M_{2,g-j-1} + M_{0,g-j} \\ &= D_{i+1,j-1} + D_{i,j-1} + D_{i-1,j-1} + \ldots + D_{1,j-1} + M_{1,g-j} \\ &= D_{i+1,j-1} + D_{i,j-1} + D_{i-1,j-1} + \ldots + D_{1,j-1} + D_{0,j-1}. \end{aligned}$$

From (6) it is easy to see that $D_{i,1} = M_{i+1,g-1-(i+1)} = 1$ for $i \geq 0$. Hence, by Lemmas 1 and 2 we conclude that $D_{i,j} = N_{i,j}$. ∎

From Lemmas 2 and 4 it follows that, for $0 \leq i \leq j \leq g-1$,

$$\begin{aligned} M_{i,j} &= N_{i-1,g-i-j} \\ &= (i+1)\frac{(2g-2i-2j+i-1-1)!}{(g-i-j+i-1+1)!(g-i-j-1)!} \\ &= (i+1)\frac{(2g-i-2j-2)!}{(g-j)!(g-i-j-1)!}. \end{aligned}$$

Notice that fixing an entry (i,j) of the matrix W, the number of CFPs which contain the processor represented by (i,j), i.e., the processor p_{ig+j}, is $N_{i,j}M_{i,j}$. Since any CFP must contain one and only one of the processors represented by the elements of a fixed column c of W, with $1 \leq c \leq g-2$, we have that

$$F^U(g) = \sum_{i=0}^{g-1-c} N_{i,c}M_{i,c}.$$

Example 5. The following tables show the $N_{i,j}$'s and the $M_{i,j}$'s for $g = 8$.

$N =$

1	1	2	5	14	42	132	429
0	1	3	9	28	90	297	...
0	1	4	14	48	165	...	...
0	1	5	20	75	...	...	...
0	1	6	27	...	...	...	...
0	1	7	...	...	...	...	...
0	1	...	...	...	...	...	...
0	...	...	...	...	...	...	...

$M =$

429	132	42	14	5	2	1	1
429	132	42	14	5	2	1	0
297	90	28	9	3	1	0	0
165	48	14	4	1	0	0	0
75	20	5	1	0	0	0	0
27	6	1	0	0	0	0	0
7	1	0	0	0	0	0	0
1	0	0	0	0	0	0	0

4 Ranking and unranking catastrophic faults

In this section we provide an invertible mapping defined over the set of CFPs for a linear array with unidirectional link redundancy g, which assumes values in the set of integers $0, 1, \ldots, F^U(g) - 1$. This function enables us to rank all CFPs. The inverse of the ranking function is given as an algorithm. In the following we consider the row representation of a CFP.

The rank of a CFP F whose row representation is $(r_0, r_1, \ldots, r_{g-1})$ is the integer given by the sum of the $N_{i,j}$'s that satisfy $i < r_j$, for $j = 0, \ldots, g-1$. Formally, the *rank* of F is

$$\mathcal{R}(F) = \sum_{c=0}^{g-1} \mathcal{R}_c(F) \tag{7}$$

where

$$\mathcal{R}_c(F) = \begin{cases} \sum_{r=0}^{r_c-1} N_{r,c} & \text{if } r_c > 0 \\ 0 & \text{if } r_c = 0. \end{cases}$$

Observe that $\mathcal{R}_0(F) = \mathcal{R}_{g-1}(F) = 0$ since for any CFP F we have $r_0 = r_{g-1} = 0$.
In order to prove that $\mathcal{R}$ is an isomorphism between the set of CFPs for a linear array with unidirectional bypass links of length g and the set of integers $\{0, 1, \ldots, F^U(g) - 1\}$, we need the following lemma.

Lemma 5 *The integers $N_{i,j}$'s, for $i, j \geq 0$, satisfy the following equality*

$$N_{i,j} = \sum_{c=0}^{j-1} \sum_{r=0}^{i+j-c-1} N_{r,c}. \tag{8}$$

Proof. From (1) we have that

$$\begin{aligned} N_{i,j} &= \sum_{r=0}^{i} N_{r,j-1} + N_{i+1,j-1} \\ &= \sum_{r=0}^{i} N_{r,j-1} + \sum_{r=0}^{i+1} N_{r,j-2} + N_{i+2,j-2} \\ &\ldots \\ &= \sum_{c=0}^{j-1} \sum_{r=0}^{i+j-c-1} N_{r,c}. \end{aligned}$$

□

The maximum value of $\mathcal{R}$ is reached when each r_c, $c = 1, \ldots, g-2$, assumes its maximum value. From Proposition 3 we have that $r_c \leq g - c$. Therefore the maximum value of $\mathcal{R}$ is reached for the CFP whose row representation is $(0, g-1, g-2 \ldots, 2, 1, 0)$.
From (4) and (8) one gets

$$F^U(g) = N_{0,g-1} = \sum_{c=0}^{g-2} \sum_{r=0}^{g-c-2} N_{r,c}. \tag{9}$$

From (7) and (9) and from the fact that $N_{0,0} = 1$ it follows that the maximum value of $\mathcal{R}$ is $F^U(g) - 1$. The function $\mathcal{R}$ is clearly non negative. It is easy to see that it assumes the value 0 for the CFP whose row representation is $(0, 0, \ldots, 0)$.
The following lemma shows that different CFPs have different rank.

Lemma 6 *Let* $(r_0, r_1, \ldots, r_{g-1})$ *and* $(s_0, s_1, \ldots, s_{g-1})$ *be two row representations of CFPs* F_1 *and* F_2*, respectively. If* $F_1 \neq F_2$ *then* $\mathcal{R}(F_1) \neq \mathcal{R}(F_2)$.

Proof. Let k be the greatest index for which $r_k \neq s_k$. Without loss of generality, we can assume that $r_k < s_k$. We have that $\mathcal{R}_j(F_1) = \mathcal{R}_j(F_2)$ for $j = k+1, k+2, \ldots, g-1$, and $\mathcal{R}_k(F_2) - \mathcal{R}_k(F_1) \geq N_{r_k,k}$. By Proposition 3 we have that $r_c \leq r_k + k - c$, for $0 \leq c \leq k$. Hence $\mathcal{R}_c(F_1) \leq \sum_{r=0}^{r_k+k-c-1} N_{r,c}$ which implies that $\sum_{c=1}^{k-1} \mathcal{R}_c(F_1) \leq \sum_{c=1}^{k-1} \sum_{r=0}^{r_k+k-c-1} N_{r,c}$. Since $N_{0,0} = 1$ and $N_{i,0} = 0$ for $i > 0$, by (8) we have that $\sum_{c=1}^{k-1} \mathcal{R}_c(F_1) \leq N_{r_k,k} - 1$. Since each term $\mathcal{R}_j(F_2)$ is non negative we have that $\sum_{j=1}^{k-1} \mathcal{R}_j(F_1) - \sum_{j=1}^{k-1} \mathcal{R}_j(F_2) < N_{r_k,k}$. As $\sum_{j=k}^{g-1} \mathcal{R}_j(F_2) - \sum_{j=k}^{g-1} \mathcal{R}_j(F_1) \geq N_{r_k,k}$ we conclude that $\mathcal{R}(F_2) - \mathcal{R}(F_1)$ is greater than 0. $\square$

Theorem 3 $\mathcal{R}$ *is an isomorphism between the set of row representations of CFPs for a unidirectional linear array with link redundancy* g *and the set* $\{0, 1, \ldots, F^U(g) - 1\}$.

Proof. By Lemma 6 we have that $\mathcal{R}$ is an injective function between the set of row representations of CFPs for a unidirectional linear array with link redundancy g and the set $\{0, 1, \ldots, F^U(g) - 1\}$. On the other hand these two sets have the same cardinality. Hence the theorem. $\square$

Next we describe an algorithm UNRANK which takes in input an integer n, $0 \leq n \leq F^U(g) - 1$, and gives as output the row representation of the CFP whose rank is n.

```
UNRANK(n)
v = n
for i = 0 to g − 1 r_i = 0
for c = g − 2 to 1 step −1
     i = 0
     while v ≥ N_{i,c} do
          v = v − N_{i,c}
          i = i + 1
     r_c = i
return (r_0, r_1, ..., r_{g−1})
```

Next lemmas prove the correctness of the algorithm.

Lemma 7 UNRANK*(n) is the row representation of a CFP.*

Proof. Let $(r_0, r_1, \ldots, r_{g-1})$ be the list returned by UNRANK. Since we never change the initial value of r_0 and r_{g-1}, we have that $r_0 = r_{g-1} = 0$. Fix s, $0 < s < g-1$. Consider the iteration of the second **for** with $c = s+1$. Let q be the value assigned to r_{s+1} at the end of the **while**. Clearly at this point we have that $v < N_{q,s+1}$. Consider the iteration of the second **for** with $c = s$. By contradiction suppose that at the end of the **while** the value assigned to r_s is greater than $q+1$. During the **while** v has been decreased by a value greater than $\sum_{j=1}^{q+1} N_{j,s} = N_{q,s+1}$. Since before the execution of the **while** v was less than $N_{q,s+1}$, then, at the end of the **while** v will be less than zero. This is a contradiction because in the algorithm v is always non negative. Hence UNRANK(n) satisfies the conditions of Proposition 3. $\square$

Lemma 8 *Let F be the CFP whose row representation is* UNRANK*(n). Then $\mathcal{R}(F) = n$.*

Proof. Let $(r_0, r_1, \ldots, r_{g-1})$ be the list returned by UNRANK, which by Lemma 7 is a row representation of a CFP. Consider the last iteration of the second **for**, that is the iteration for which $c = 1$. Since $N_{i,1} = 1$ for $i \geq 0$, the algorithm decreases v by one until v is 0. Therefore, at the end of the algorithm v is equal to zero. Since the rank of the CFP represented by this list, is equal to the sum of the $N_{i,j}$'s used to decrease v in the **while**, the lemma follows. ∎

5 Generation of the catastrophic faults

In this section we describe ad analyze an algorithm for the generation of all the catastrophic fault patterns for a linear array with link redundancy g. The problem of the generation of the objects of a given set has been widely studied [11]. Our algorithm, as in many algorithms for the systematic generation of a set of objects, has three components: the initialization, the transformation from an object of the set to the next one, and the end condition telling when to stop. In our case, the set of objects is the set of all the catastrophic fault patterns. We want generate the CFPs according to the order established by the rank, i.e., we want to start with the CFP whose rank is 0, and then proceed by generating the CFPs in order of increasing rank. The initialization is the generation of the CFP which has the smallest rank. The transformation from a catastrophic fault pattern F whose rank is $\mathcal{R}(F)$, yields the CFP G whose rank is $\mathcal{R}(G) = \mathcal{R}(F) + 1$. Let $(r_0, r_1, \ldots, r_{g-1})$ be the row representation of a catastrophic fault pattern F. The CFP with rank $\mathcal{R}(F) + 1$ is obtained by increasing the row index r_c, $c \leq g - 2$, such that $r_c \leq r_{c+1}$ and $r_j > r_{j+1}$, for $1 \leq j \leq c$, and by setting to 0 the row indexes $r_1, r_2, \ldots, r_{c-1}$. Observe that if such index r_c does not exist, then the CFP has row representation $(0, g-2, g-3, \ldots, 2, 1, 0)$ and it has the biggest rank. Procedure NEXT uses the dummy row index $r_g = 0$ to detect this situation which constitute the end condition.
Procedure GENERATE uses procedure INIT to perform the initialization, and calls procedure NEXT until the end condition is reached, to obtain all the CFPs. The algorithm follows.

```
INIT(F,flag)
flag=false
for j = 0 to g
      r_j = 0
return

NEXT(F,flag)
j = 1
while r_j > r_{j+1} do
      r_j = 0
      j = j + 1
if j = g then flag = true
      else r_j = r_j + 1
return

GENERATE()
INIT(F,flag)
while flag=false do
      NEXT(F,flag)
return
```

Notice that the procedure NEXT, in order to obtain the next CFP, modifies only a subset of the row indexes $r_1, r_2, \ldots, r_{g-2}$, without rewriting those that remain unchanged. The correctness of procedures INIT and GENERATE is straightforward. Next lemma proves the correctness of NEXT.

Lemma 9 *Given as input a catastrophic fault pattern F, with $\mathcal{R}(F) < F^U(g)$, a call to* NEXT *returns the catastrophic fault pattern G whose rank is $\mathcal{R}(G) = \mathcal{R}(F) + 1$.*

Proof. Let $(0, r_1, \ldots, r_{c-1}, r_c, r_{c+1}, \ldots, r_{g-2}, 0)$ be the row representation of F. First, observe that the procedure yields always a CFP. Indeed, let c be the value of j at the end of the **while**. Then c is the smallest index for which $r_c \le r_{c+1}$. Procedure NEXT increases the row index r_c and, if $c > 1$, it sets to zero the row indexes $r_1, r_2, \ldots, r_{c-1}$. Hence the obtained row representation satisfies Proposition 3.
If $c = 1$ it is easy to see that $\mathcal{R}(G) = \mathcal{R}(F) + 1$.
Assume $c > 1$. Then $(0, 0, \ldots, 0, r_c + 1, r_{c+1}, \ldots, r_{g-2}, 0)$ is the row representation of G. Since $\mathcal{R}_0(H) = \mathcal{R}_{g-1}(H) = 0$ for any CFP H for a linear array with link redundancy g, from (7) we have that

$$\begin{aligned}\mathcal{R}(F) &= \mathcal{R}_1(F) + \ldots + \mathcal{R}_c(F) + \ldots + \mathcal{R}_{g-2}(F) \\ &= \sum_{r=0}^{r_1-1} N_{r,1} + \ldots + \sum_{r=0}^{r_c-1} N_{r,c} + \mathcal{R}_{c+1}(F) + \ldots + \mathcal{R}_{g-2}(F)\end{aligned}$$

whereas

$$\mathcal{R}(G) = \sum_{r=0}^{r_c} N_{r,c} + \ldots + \mathcal{R}_{g-2}(G).$$

Since $\mathcal{R}_k(F) = \mathcal{R}_k(G)$, for $g-2 > k > c$, from (8) we have that

$$\mathcal{R}(G) - \mathcal{R}(F) = N_{r_c,c} - \left(\sum_{r=0}^{r_1-1} N_{r,1} + \ldots + \sum_{r=0}^{r_{c-1}-1} N_{r,c-1} \right) = N_{0,0} = 1.$$

Hence the lemma. □

Now, we analyze the complexity of GENERATE. The execution of GENERATE requires one call to INIT and exactly $F^U(g)$ calls to NEXT. The complexity of INIT is clearly $\Theta(g)$. Procedure NEXT yields the next catastrophic fault pattern by increasing an index j from 1 up to the first value $c < g-1$ for which $r_c \le r_{c+1}$. Hence, a simple upper bound is $O(g)$. Next lemma, by using an ammortized analysis, characterizes the complexity of NEXT.

Lemma 10 *Let c be an integer, $1 \le c \le g-2$. During the $F^U(g)$ calls, procedure* NEXT *ends the computation executing c iterations in the* **while** *statement exactly $M_{2,c-1}$ times.*

Proof. From Proposition 3, if in a CFP $r_j > r_{j+1}$ then $r_j = r_{j+1}+1$. Therefore if the algorithm ends the **while** with $j = c$, then $r_{k-1} = r_k + 1$ for $k = 2, 3, \ldots, c-1$ and $r_c \le r_{c+1}$. How many times does this situation occur? This situation occurs whenever the CFP input of NEXT has row representation $(0, r_c + c, \ldots, r_c + 1, r_c, \ge r_c, *, \ldots, *)$, where $\ge r_c$ means an index unspecified but greater than r_c and $*$ means an unspecified row index. For fixed values of c and r_c, there are exactly $M_{r_c,c+1} + M_{r_{c+1},c+1} + \ldots + M_{r_{g-2-c},c+1} = M_{r_c+1,c}$ such CFPs. To obtain the number of times in which NEXT ends the **while** with $j = c$ we have to sum all previous quantities over all possible values of r_c for which $M_{r_{c+1},c}$ is not zero, that is for $r_c = 0, 1, ..., g-c-2$. Hence NEXT exits from the **while** with $j = c$ exactly

$$M_{1,c} + M_{2,c} + \ldots + M_{g-1-c,c} = M_{2,c-1}$$

times. □

Lemma 10 enables us to estimate the total running time (i.e., the time needed to generate all the CFPs) of GENERATE. By using (5) we have that

$$\begin{aligned} M_{1,0} &= M_{2,0} + M_{1,1} \\ &= M_{2,0} + M_{2,1} + M_{1,2} \\ &\ldots\ldots \\ &= M_{2,0} + M_{2,1} + \ldots + M_{2,g-3} + M_{1,g-2}. \end{aligned}$$

Since $N_{0,g-1} = M_{0,0} = M_{1,0}$ and $M_{1,g-2} = 1$ we have that

$$\sum_{c=1}^{g-2} c \; M_{2,c-1} = N_{0,g-1} - 1.$$

Therefore the total running of time GENERATE is proportional to

$$
\begin{aligned}
\sum_{c=1}^{g-2} c\ M_{2,c-1} &= N_{0,g-1} - 1 + \sum_{c=1}^{g-3} cM_{2,c-1} \\
&= N_{0,g-1} - 1 + N_{0,g-2} - 1 + \sum_{c=1}^{g-4} cM_{2,c-1} \\
& \quad \ldots \\
& \quad \ldots \\
&= N_{0,g-1} + N_{0,g-2} + \ldots + N_{0,1} - (g-1).
\end{aligned}
$$

From (4), since $F^U(k) = \Theta(4^k/k^{3/2})$, the above expression is clearly $\Theta(4^g/g^{3/2})$. Therefore, the algorithm generates all the CFPs in time linear in the number of the CFPs.

Acknowledgements. We thank L. Pagli for useful discussions and for pointing out the references on the subject. The first author would like to thank A. De Bonis for helpful suggestions.

References

[1] K. P. Belkhale and P. Banerjee, "Reconfiguration strategies in VLSI processor arrays", in *Proc. Int'l Conf. on Computer Design*, pp. 418–421, 1988.

[2] M. Chean and J. A. B. Fortes, "A taxonomy of reconfiguration techniques for fault-tolerant processor arrays", *IEEE Computer*, Vol. 23, n.1, pp. 55-69, Jan 1990.

[3] J. W. Greene and A. E. Gamal, "Configuration of VLSI arrays in presence of defects", *Journal of the ACM*, Vol. 31, n. 4, pp. 694–717, Oct 1984.

[4] D. E. Knuth, *The art of computer programming*, vol I, Addison-Wesley, 1973.

[5] H.T. Kung, "Why systolic architecture?", *IEEE Computer*, vol. 15, n. 1, pp. 37–46, Jan 1982.

[6] A. Nayak, "On reconfigurability of some regular architectures", Ph.D Thesis, Dept. System and Computer Engineering, Carleton University, Ottawa, Canada, 1991.

[7] A. Nayak, L. Pagli, and N. Santoro, "Recognition of catastrophic faults in reconfigurable arrays with arbitrary link redundancy", School of Computer Science, Carleton University, Technical Report 202, March 1992.

[8] A. Nayak and N. Santoro, "Bounds on performance of VLSI processor arrays", in *5th Int'l Parallel Processing Symposium*, Anaheim, California, May 1991.

[9] A. Nayak, N. Santoro, and R. Tan, "Fault-intolerance of reconfigurable systolic arrays", In *Proc. 20th Int. Symp. on Fault Tolerant Computing, FTCS'20*, pp. 202–209, 1990.

[10] L. Pagli and G. Pucci, "Reliability analysis of redundant VLSI arrays", Preprint 1992.

[11] E.M. Reingold, J. Nievergelt, and N. Deo, *Combinatorial Algorithms: theory and practice*, Prentice-Hall, 1977.

[12] A. L. Rosenberg, "The diogens approach to testable fault-tolerant arrays of processors", *IEEE Trans. on Computer*, vol. C-32, n. 10, pp. 902–910, Oct 1983.