Short Notes

A Pairwise Substitutional Fault Tolerance Technique for the Cube-Connected Cycles Architecture

Nian-Feng Tzeng and Po-Jen Chuang

Abstract— **With all of the salient features of hypercubes, the cube-connected cycles (CCC) structure is an attractive parallel computation network suited for very large scale integration (VLSI) implementation because of its layout regularity. Unfortunately, the classical CCC structure tends to suffer from considerable performance degradation in the presence of faults. In this article, we deal with a fault-tolerant CCC structure obtained by incorporating a spare PE in each cycle and by adding extra links among PE's to realize dimensional substitutes for failed PE's in the immediate lower dimension. A unique feature of this design lies in that a faulty PE and its laterally connected PE are always replaced at the same time by their immediate vertical successor pair, achieving *pairwise substitution* to elegantly maintain the rigid full CCC structure after faulty PE's arise. The proposed structure improves reliability substantially without incurring large overhead in layout area. This design is compared with earlier fault-tolerant CCC designs in terms of normalized reliability, which takes area overhead into account. An extension to this fault-tolerant structure is also discussed.**

Index Terms—**Cube-connected cycles, fault tolerance, reconfiguration, reliability analysis, VLSI layout**

I. Introduction

A parallel system is a collection of autonomous processors interconnected according to an underlying topology. Each processor has its local memory, and is also termed a processing element (PE). A single chip or wafer may contain millions of gates that could realize a set of PE's and their interconnection links, constituting a parallel system. It is thus of practical significance now to consider how efficiently a parallel system of interest can be implemented by the very large scale integration (VLSI) or wafer scale integration (WSI) technology.

Although it supports well the communication patterns of many numerical algorithms [2], the hypercube topology is not readily suited for VLSI/WSI implementation, because the number of links to each PE grows with the hypercube dimension. Preparata and Vuillemin [2] proposed a substitute for the hypercube, known as the cube-connected cycles (CCC) architecture, which not only preserves all of the salient features of the hypercube but also enjoys a more compact and regular layout. The CCC, however, may suffer from considerable performance degradation when faults arise.

One fundamental consideration in designing a parallel system is its reliability. A parallel system tends to have lower reliability as its size grows, unless certain fault-tolerant capability is incorporated in the system design. Typically, a fault-tolerant system responds to failures by reconfiguring itself to exclude failed components while keeping all nonfaulty components. A parallel system may or may not preserve its underlying topology after reconfiguration. If it cannot preserve its underlying topology, a reconfigured system may no longer deliver the desired level of performance when executing parallel algorithms. On the other hand, if a parallel system maintains its underlying topology after reconfiguration, the system can still support applications efficiently even in the presence of failures, called a *strongly* fault-tolerant system. The basic approach to achieving strong fault tolerance is via employing systemwide redundancy and reconfiguration that ensure a rigid and full system structure even in the presence of faults. Strongly fault-tolerant designs are preferable, especially for those structures that make use of pipelining and parallelism heavily, such as the CCC.

Manuscript received September 11, 1991; revised September 23, 1992. This work was supported in part by the National Science Foundation (NSF) under Grants MIP–8807761 and MIP–9201308, and in part by the State of Lousiana under Contract LEQSF(1992–94)–RD–A–32. A preliminary version of this short note was presented at the 19th Annual International Confrence on Parallel Processing, Aug. 1990.

N.-F. Tzeng is with The Center for Advanced Computer Studies, University of Southwestern Louisiana, Lafayette, LA 70504.

P.-J. Chuang was with The Center for Advanced Computer Studies, University of Southwestern Louisiana, Lafayette, LA 70504. He is now with the Department of Electrical Engineering, Tamkang University, Taipei, Taiwan, Republic of China.

IEEE Log Number 9215378.

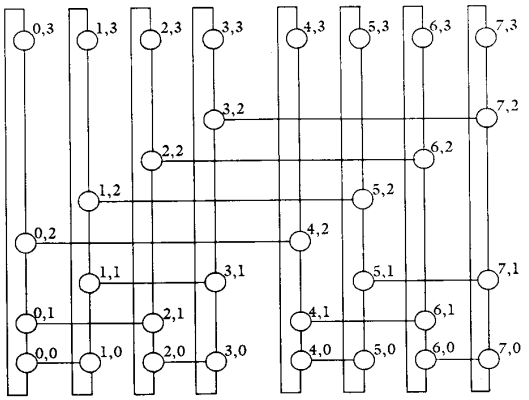

Fig. 1. A layout for CCC(4, 3).

A new strongly fault-tolerant CCC is introduced in this short note. In this new structure, a failed PE and its laterally connected PE are always replaced at the same time by their vertically successor pair, regardless of whether the laterally connected PE is faulty, realizing *pairwise substitution* that neatly achieves strong fault tolerance. The design exhibits a significant reliability improvement while maintaining its layout area overhead low, making it advantageous in VLSI/WSI environments. This design concept can readily be extended to incorporate more redundancy in the CCC to further improve its reliability.

II. Review of CCC and Prior Fault-Tolerant CCC Structures

The CCC interconnects identical PE's, each with three ports [2]. A connecting link between two PE's can be used for bidirectional data transmission. A CCC composed of 2^d cycles with each cycle involving h PE's is denoted by CCC(h, d), where h is no less than d. Each PE has an address expressed as a pair of integers, (c, p), where c and p denote the address of the cycle containing the PE and the position of the PE within the cycle, respectively. Cycles are numbered from 0 to $2^d - 1$, starting with the leftmost one, and PE's in a cycle are numbered from 0 to $h - 1$, starting with the lowest PE.

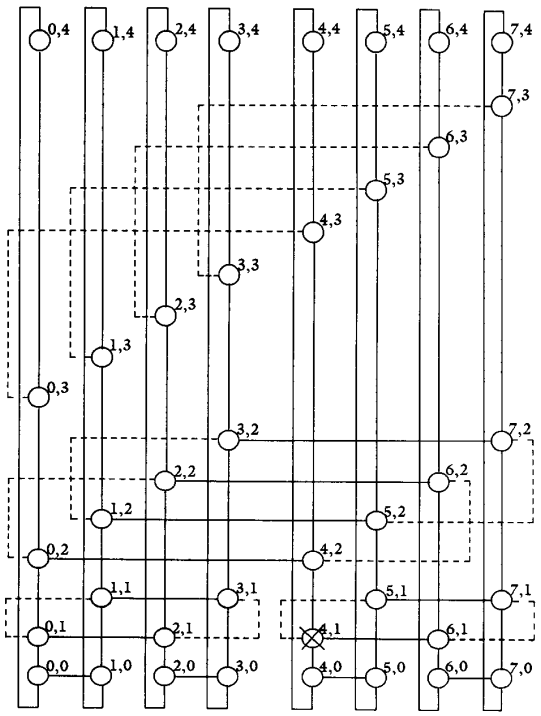

Fig. 2. The structure of DSCCC(5, 3) constructed by adding dimensional substitutes to PE's and adding an extra PE per cycle to CCC(4, 3).

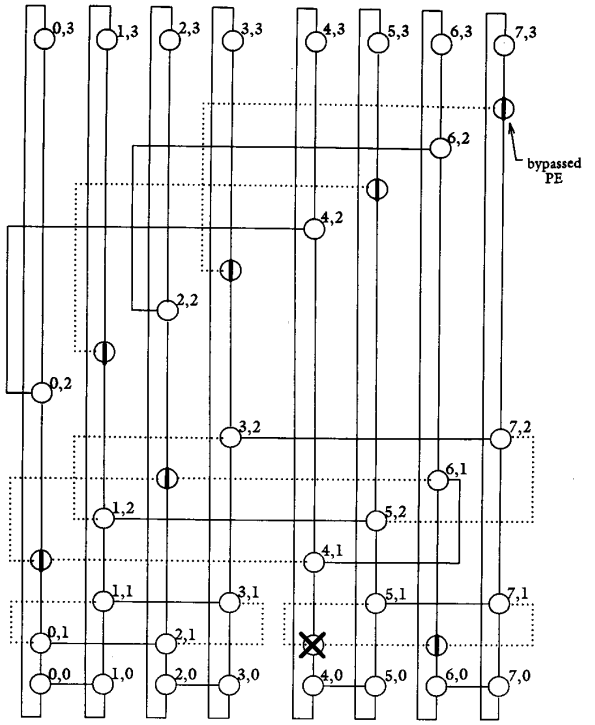

Fig. 3. The reconfigured DSCCC(5, 3) after PE(4, 1) fails. (Solid links are active connections and dotted links are inactive. A bypassed PE has a darkened line across it.)

Suppose that the three ports of each PE are called F, B, and L (mnemonic for *Forward, Backward, Lateral*), respectively. The CCC interconnection [2] is specified as follows: F of PE(c, p) is connected to B of PE$(c, (p+1) \bmod h)$; B of PE(c, p) is connected to F of PE$(c, (p-1) \bmod h)$; and L of PE(c, p) is connected to L of PE$(c+\alpha 2^p, p)$, where $\alpha = 1-2*$(the pth bit of c). All PE's inside a cycle are circularly connected by the $F-B$ links. The lower d PE's in each cycle are interconnected following the hypercube connection pattern to other PE's in different cycles through the L links, i.e., the ith PE, $0 \le i < d$, is connected to the corresponding PE in another cycle which is 2^i away from the current cycle and this L link forms the ith dimensional connection. The upper $(h-d)$ PE's do not utilize their L links. CCC(4, 3) is depicted in Fig. 1.

The Cubical Ring Connected Cycles (CRCC) [5] is a fault-tolerant CCC, obtained by adding a redundant PE and $2^{d-1}+1$ redundant lateral links to each dimension in CCC(h, d) to form a ring involving 2^d+1 PE's. Each PE requires one additional port for connecting a redundant lateral link, and the total number of cycles is increased by one. Once a fault arises in the CRCC, the cycle containing the fault is discarded, and the remaining 2^d cycles emulate the CCC. The locally reconfigurable CCC (LR–CCC) [6] is another fault-tolerant CCC, where spare PE's are included in each cycle and redundant links are added to connect every spare PE to each nonspare PE within the cycle. The degree of a spare PE is not fixed and grows as the cycle size increases, whereas a regular PE needs one additional port for each spare added to the cycle. Reconfiguration is carried out in each cycle individually.

III. Proposed Structure

A. Structure Description

To achieve fault-tolerance, we augment the connections among PE's in dimension i of CCC(h, d), $0 < i < d$, in such a way that a direct link is added to every pair of PE's whose vertical predecessors have a dimension $i-1$ connection. In other words, dimension $i-1$ connections are also realized among PE's in dimension i, as shown by dashed lines in Fig. 2. Additionally, a spare PE is added immediately above PE$(c, d-1)$ for every cycle c, with the connections among spare PE's being exactly the same as those among the PE's in dimension $d-1$.

With this provision, after a PE (e.g., PE(4, 1)) becomes faulty, the link between PE(4, 2) and PE(6, 2) is activated to serve as the dimension 1 connection upon reconfiguration, whereas the link between PE(4, 1) and PE(6, 1) is deactivated (i.e., PE(6, 1), although healthy, is removed from the reconfigured system). Because PE(4, 2) and PE(6, 2) now take over the roles of PE(4, 1) and PE(6, 1), respectively, PE(0, 2) and PE(2, 2) are removed from the reconfigured system, whereas the PE(0, 3)–PE(4, 3) spare pair and the PE(2, 3)–PE(6, 3) spare pair are then brought in for replacing the roles of the PE(0, 2)–PE(4, 2) pair and the PE(2, 2)–PE(6, 2) pair, respectively. As a result, PE(0, 3) becomes the vertical successor to PE(0, 1). Likewise, PE(2, 3), PE(4, 2), and PE(6, 2) are the vertical successors, respectively, to PE(2, 1), PE(4, 0), and PE(6, 0). The reconfigured system in response to the PE(4, 1) failure is illustrated in Fig. 3, where solid links are active connections and dotted links are inactive. Notice that the added PE's act as standby spares and are not used under normal fault-free circumstances; they may be activated only when a failure arises. This new fault-tolerant cube-connected cycles structure achieves fault tolerance through dimensional substitution, and is thus referred to as the DSCCC (where DS stands for dimensional substitution). Dimensional substitution is also referred to as pairwise substitution because substitution is always carried out pairwise, and they are used interchangeably.

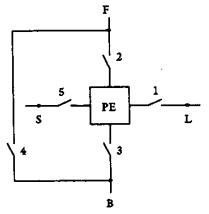

Fig. 4. A PE with its control switches to allow various connecting states settings.

In practice, links are more reliable than PE's, because they involve lower hardware complexity and are passive components. Faults at an L link may be viewed as the failure of any one of the two PE's connected by the L link, and thus a failed L link is tolerable. Although failures at the B/F link cannot be tolerated, such a link is expected to have a relatively low failure rate, because it is often shorter and connects two neighboring PE's. In addition, redundant wires may be provided in such a fault-intolerant link for trivially replacing failed wires [4], making it fairly reliable. A faulty PE is assumed to allow data to pass successfully between its B and F links. This may be accomplished by adding switches to every PE so that data can get around any PE when those switches are set appropriately, as depicted in Fig. 4. For an active PE, switches 2 and 3 are on, whereas switch 4 stays off. If a PE is to be removed from the system, its switches 2 and 3 are set off, with switch 4 set on to make a direct connection between its B and F links.

Each cycle of the proposed fault-tolerant CCC(h, d) contains $h+1$ PE's, which are numbered from 0 to h starting with the lowest one. This fault-tolerant CCC is denoted as DSCCC($h+1$, d). The spare PE in cycle c of DSCCC($h+1$, d) is addressed by PE(c, d). Every PE in the DSCCC has four ports, with three of them making the CCC interconnection style and the remaining one augmenting the structure for fault tolerance. The extra connection from each PE is made as follows. If PE(c_1, p) connects directly to PE(c_2, p), $0 \le p < d$, then the upward successor of PE(c_1, p) in cycle c_1, PE(c_1, $p+1$), also connects to the upward successor of PE(c_2, p) in cycle c_2, PE(c_2, $p+1$). The top $(h-d)$ PE's in each cycle do not have any extra connections, nor do PE's in the lowest dimension, dimension 0. These unused ports may be employed for I–O purposes.

Formally, DSCCC($h+1$, d) is defined as a collection of $(h+1)2^d$ PE's, which forms 2^d cycles, each with $(h+1)$ PE's. In addition to the F, B, and L ports, each PE has the S (mnemonic for substitution) port. The interconnection of F, B, and L among PE's is the same as that given in Section II, and the interconnection of S is characterized below.

S of PE(c, p) is connected to S of PE($c+\gamma 2^{p-1}$, p) for $1 \le p \le d$,

where γ equals $1-2*$(the $(p-1)$th bit of c).

Compared with CCC(h, d), DSCCC($h+1$, d) requires 2^d more PE's and $(d-1)2^{d-1}+3(2^{d-1})$ more links. The DSCCC achieves strong fault tolerance with respect to any single fault.

B. Reconfiguration Process

Consider DSCCC($h+1$, d), where a faulty PE, e.g., PE(c, i), arises in dimension i, $0 \le i < d$. The reconfiguration process works from dimension i onward, dimension-by-dimension upward, until necessary spares are brought into the system. It discards certain healthy PE's to neatly accomplish dimensional substitution. A PE is described as being at the "bypassed" state if its control switches 2 and 3 are off and switch 4 is on (see Fig. 4). It is described as being at the "regular" state if its switches 1, 2, and 3 are set on and switches 4 and 5 are off. It is described as being at the "substitute" state if its switches 2, 3, and 5 stay on and switches 1 and 4 are off. After the said fault arises, the reconfiguration process removes the failed PE, together with its dimensionally connected PE, PE($c+\alpha 2^i$, i), from the system, *no matter whether* PE $(c+\alpha 2^i, i)$ *is faulty*, to elegantly keep the strict full CCC structure, where α is $1-2*$(the ith bit of c) as defined in Section II. In other words, this pair of PE's are set to the "bypassed" state at the same time. In dimension j $(i<j<d)$, 2^{j-i+1} PE's change their connecting states during the reconfiguration process, namely, one half of them are set to the "substitute" state for realizing dimension $j-1$ connections, whereas the other half are set to the "bypassed" state. Those PE's involved can be identified iteratively from dimension $i+1$ upward. Specifically, in dimension $i+1$, PE(c, $i+1$) and PE(c_1, $i+1$) are set to the "substitute" state, whereas PE($c+\alpha_2 2^{i+1}$, $i+1$) and PE($c_1+\alpha_3 2^{i+1}$, $i+1$) are set to the "bypassed" state, where c_1 equals $c+\alpha 2^i$, α_2 is $1-2*$(the $(i+1)$th bit of c), and α_3 is $1-2*$(the $(i+1)$th bit of c_1). Similarly, in dimension $i+2$, the four immediate vertical successors of the four mentioned PE's in dimension $i+1$ are set to the "substitute" state, with their laterally connected PE's being set to the "bypassed" state. This process repeats until all involved PE's are determined and set up properly. The reconfigured system after PE(4, 1) fails in DSCCC(5, 3) is shown in Fig. 3.

It is clear that the number of PE's that change their connecting states during the reconfiguration process depends on the dimension in which the faulty PE is located. The number decreases when it falls in a higher dimension. If a fault arises in dimension i, the total number of PE's involved amounts to $\sum_{j=i}^{d-1} 2^{j-i+1} + 2^{d-i} = 3*2^{d-i} - 2$, where the last term 2^{d-i} is the number of spare PE's brought into the system. It should be noted that following this process, when a second or subsequent fault occurs, a bypassed PE may be brought into the system again upon reconfiguration. In Fig. 3, for example, if PE(0,1) fails, two bypassed PE's, which are the immediate vertical successors to PE(0,1) and PE(2, 1), are then reconfigured into the system to replace the dimension 1 connection that originally exists between PE(0, 1) and PE(2, 1).

The reconfiguration process can proceed in a distributed manner if 1) the PE is self-testable, allowing it to determine its own status [7], and 2) every PE can read the status of any of its directly connected PE's. Before the process starts, every PE performs a self-test individually. Then every healthy PE in dimension i, $0 \le i < d$, carries out the following process, one dimension after another, starting with dimension 0.

1) If a PE finds its B-link connected neighbor to be failed or bypassed (by reading the status of the neighbor), the PE is set to the "substitute" state; otherwise, it is set to the "regular" state.
2) If a PE finds its L-link connected neighbor to be failed or at the "substitute" state, the PE is set to the "bypassed" state by reconfiguring itself accordingly; otherwise, its connecting state is unchanged.

Notice that PE's in dimension 0 execute only step 2). Every spare PE then performs step 1). Finally, a failed PE is set to the "bypassed" state.

IV. Reliability Analysis

We first examine what kinds of multiple faults are tolerable in DSCCC($d+1$, d). When a fault arises in dimension i, $0 \le i < d$, from the reconfiguration process, δ_1 PE's are set to the "substitute" state, where $\delta_1 = \sum_{j=i+1}^{d} 2^{j-i} = 2^{d-i+1}-2$. If any subsequent fault happens to these δ_1 PE's, then it is not tolerable, and DSCCC($d+1$, d) fails. Moreover, to keep the structure operational, a subsequent fault

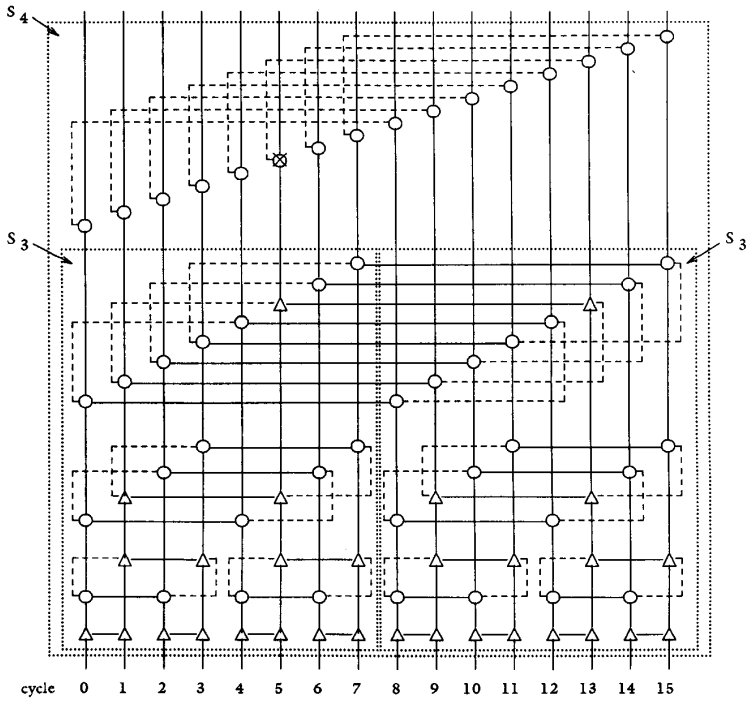

Fig. 5. The structure of S_4, comprising three parts: X_4 and two S_3's. (The wraparound link of every cycle is omitted for clarity.)

cannot occur at certain PE's in every dimension below i as well. In Fig. 5, for example, the failure of PE(13, 3) also excludes any subsequent fault at PE(9, 2) and PE(13, 2) in dimension 2; at PE(9, 1), PE(11, 1), PE(13, 1), and PE(15, 1) in dimension 1; and at the eight rightmost PE's in dimension 0. This is because any such subsequent fault, on reconfiguration, needs PE(13, 3), the first faulty PE, to be set to the "substitute" state, a surely unsuccessful attempt. Note that all of the triangles and circles in this figure denote identical PE's, and they are so drawn simply for ease of reference. In general, a fault in dimension i also disallows δ_2 PE's in dimensions below i to become faulty, where $\delta_2 = \sum_{j=0}^{i-1} 2^{j+1} = 2^{i+1} - 2$. These δ_2 PE's are referred to as *critical-B* PE's, whereas those δ_1 PE's are called *critical-A* PE's (where B and A stand for below and above, respectively). The union of *critical-B* PE's and *critical-A* PE's are called *critical* PE's.

Similar to critical-A PE's, critical-B PE's can be identified dimension-by-dimension, but from dimension $i-1$ downward. In the presence of the said fault, any subsequent fault in DSCCC$(d+1, d)$ is tolerable, as long as it is not a critical PE. After a tolerable subsequent fault arises, the critical PE's are redetermined, and the number of the critical PE's now depends upon the positions of the two failed PE's. A third fault is tolerable, provided that it does not belong to the set of the newly determined critical PE's. Similarly, this characteristic can be employed to decide whether an arising fault is tolerable, given any number of faults present in the structure. This characteristic serves as the basis of our reliability analysis.

The following assumptions are made to facilitate reliability evaluation:

1) The PE becomes faulty randomly and independently, with a constant failure rate λ, which includes the faults at its L link.
2) The B/F link is very reliable and therefore is not treated individually; instead, a "lump" failure rate is assigned to the set of all B/F links.

Let $Q_d(k)$ be the probability that DSCCC$(d+1, d)$ can still function after having k faulty PE's, then the reliability of DSCCC$(d+1, d)$ is as follows:

$$R_{\mathrm{DSCCC}(d+1,d)}(t) = \sum_{k=0}^{N_d} Q_d(k) \times \binom{N_d}{k} \times (e^{-\lambda t})^{N_d-k} \times (1-e^{-\lambda t})^k, \quad (1)$$

where N_d is the total number of PE's in DSCCC$(d+1, d)$. An exact evaluation of $R_{\mathrm{DSCCC}(d+1,d)}(t)$ is extremely difficult, because it depends not only on the number but also on the positions of faults. Therefore, lower and upper bounds on $R_{\mathrm{DSCCC}(d+1,d)}(t)$ are pursued instead. The derivation of the two bounds is provided in [8]. A program has been written to calculate the two bounds of any given DSCCC$(d+1, d)$, and the two reliability bounds calculated are always close to each other.

In Fig. 6, the lower and upper reliability bounds on DSCCC(5, 4) versus time are plotted, where the PE failure rate λ is assumed to be 1.0 per unit time (e.g., 10^6 hr) and the "lump" B/F links failure rate for DSCCC$(d+1, d)$ is $d\lambda$. These two bounds give very close values throughout the time period of interest, and thus the exact DSCCC reliability can be stated reasonably precisely. As expected, DSCCC(5, 4) exhibits a significant reliability improvement over CCC(4, 4), in which all of the CCC links are assumed to be totally fault-free.

Since the CCC is suitable for VLSI implementation and a whole CCC system can be fabricated on one chip or wafer, the cost of a system is dictated mainly by its layout area. Preparata and Vuillemin [2] presented a layout strategy for CCC(h, d) based on the VLSI grid model. This layout takes area $O(N^2/\log^2 N)$ for CCC(h, d), $N = h * 2^d$. (See Fig. 1 for an example.) The DSCCC can be laid out in a way that the added links and spares incur low area overhead. Assume that the ratio of PE width to link width is g. In practice, g is greater than 1 because the PE is expected to be thicker than the link

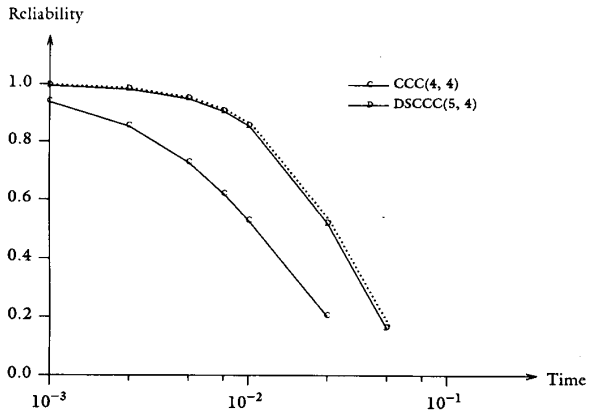

Fig. 6. Reliabilty comparison between the CCC and the DSCCC. (The dotted curve shows the DSCCC reliabilty upper bound.)

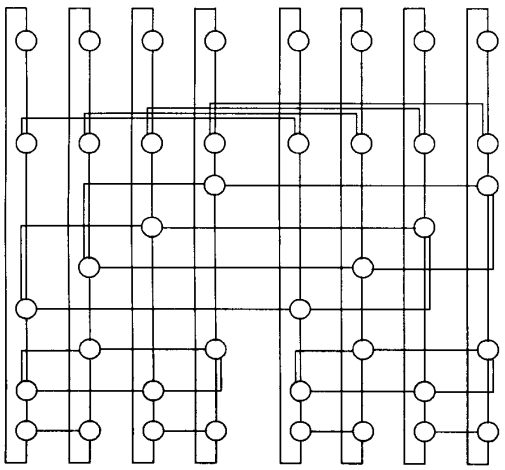

Fig. 7. A layout for DSCCC(5, 3).

as it involves more components [3]. For $g \geq 2$, the DSCCC layout requires no extra vertical spacing to accommodate the added links, as shown in Fig. 7. One extra row is needed for spare PE's, and the L connections between pairs of spares take 2^{d-1} extra horizontal tracks, one for a pair. The height of the DSCCC$(h+1, d)$ layout is thus increased by $(g+2^{d-1})$ as compared with that of the CCC(h, d) layout. It is easy to show that the DSCCC$(h+1, d)$ layout takes area of $[(1+g)2^d] * [(2^d + h - d)g + 2^{d-1}]$, where the term $(1+g)2^d$ accounts for the layout width and the second term is the layout height. When $g = 4$, for example, DSCCC(5, 4) involves an area overhead of 20% in comparison with CCC(4, 4). The area overhead decreases as the system size grows.

Because different fault-tolerant designs incur different amounts of area overhead, in order to ensure a fair comparison between designs, it seems meaningful to take the overhead amounts into account, instead of using reliability alone as a measure. We adopt the ratio of reliability to area, referred to as *normalized reliability*, as a measure for comparing various designs. The area of a standard CCC layout is assumed to be one unit. Normalized reliability indicates the effectiveness of a fault-tolerant design.

The DSCCC is compared with two earlier fault-tolerant designs, namely, the CRCC [5] and the LR–CCC [6] in terms of normalized reliability, as depicted in Fig. 8, where g is assumed to be 4 (i.e.,

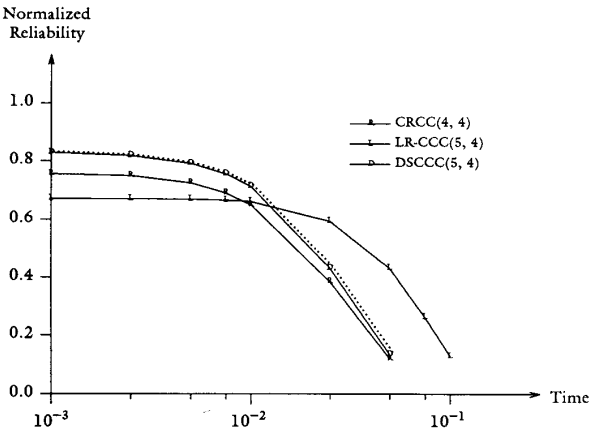

Fig. 8. Comparison among various fault-tolerant designs. (The dotted curve shows the DSCCC upper bound result.)

relatively simple PE's) in every design. All links in the CRCC and the LR–CCC are assumed to be totally fault-free (an optimistic assumption), and the LR–CCC has one spare PE per cycle. With $g = 4$, the area overheads of CRCC(4, 4) and LR–CCC(5, 4) [6] are, respectively, 32% and 49%, which are much higher than the DSCCC(5, 4) overhead. The DSCCC is consistently more effective than the CRCC and is better than the LR–CCC if the service time is not very long. In the case of an extended service time, one can incorporate two rows of spares in the DSCCC to further improve its reliability, as will be illustrated in the next section, with the involved area overhead being less than 40%, because the second spare row occupies less area than the first spare row, which incurs 20% overhead. Although the PE is assumed to take the same area for every fault-tolerant design in our comparison, it should be noted that the degree of spares in LR–CCC(5, 4) equals 6 and increases as the system size grows, implying that the PE actually occupies more area in the LR–CCC than in the DSCCC.

V. Extension to Multiple Spare Rows

For a system designed for a long mission time, it might be necessary to consider an even more reliable structure than the DSCCC discussed so far. Specifically, we may need a structure that can tolerate more than one fault per cycle. The scheme for constructing the DSCCC can be extended for incorporating more than one spare row in the CCC to further increase its reliability. Although we can have as many rows of spares as needed in general, we investigate here only the case of adding two spare rows to the CCC. A general case can be analyzed similarly.

Consider CCC(h, d) where one spare row, called SR_1, is placed right above the highest dimension, as in the DSCCC. Assume that the second spare row, called SR_2, is put right above the ith dimension, $0 \leq i < d$. The resultant structure, denoted by DSCCC$_i^2(h+2, d)$, can be viewed as composed of two sections, with the upper section involving dimensions $i+1, i+2, \cdots, d-1$, and SR_1, whereas the lower one involves dimensions $0, 1, \cdots, i$, and SR_2. In the lower section, PE's in dimension 1 have additional connections added to serve as substitutes for emulating dimension 0 connections. Similarly, PE's in dimension $j, 2 \leq j \leq i$, are equipped with extra links as dimension $j-1$ substitutes. The spare PE's in SR_2 have the connecting pattern identical to that among PE's in dimension i. In the upper section, PE's in dimension $i+1$ have no extra connections added, just like PE's in dimension 0. Substitute links are added to

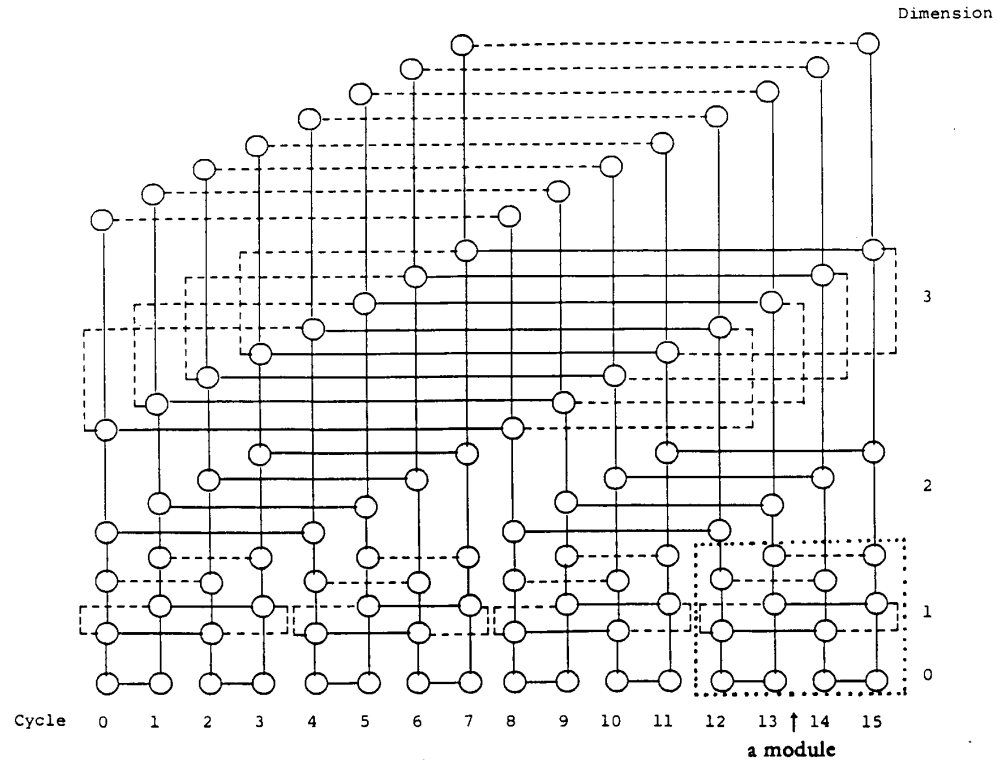

Fig. 9. The structure of $\mathrm{DSCCC}_1^2(6, 4)$, where SR_2 is added above dimension 1. For the sake of clarity, the wraparound connections are not shown and only one module is illustrated.

PE's in dimension $j, i+2 \leq j < d$, and the spare PE's in SR_1 are interconnected in the same way as PE's in dimension $d-1$. With this provision, a cycle may tolerate as many as two faults, one in the upper section and the other in the lower section, because the fault in the upper section is taken care of by the spare in SR_1, whereas the fault in the lower section is by the spare in SR_2. $\mathrm{DSCCC}_i^2(6, 4)$ with $i = 1$ is illustrated in Fig. 9.

The overall reliability of $\mathrm{DSCCC}_i^2(h+2, d)$ depends on the position of the second spare row, i. The larger the i value, the more reliable the upper section, but the less reliable the lower section. Intuitively, it appears that if SR_2 is added to the middle of the CCC, then the overall reliability is maximized. The exact reliability of the DSCCC_i^2 is fairly difficult to evaluate. However, we may utilize the lower bound expression derived earlier for estimating a lower bound on the $\mathrm{DSCCC}_i^2(d+2, d)$ reliability. The lower bound is calculated on the following basis. The reliability of the lower $\mathrm{DSCCC}_i^2(d+2, d)$ section is the product of 2^{d-1-i} *independent* modules, each with its reliability lower bounded by that of $\mathrm{DSCCC}(i+2, i+1)$. In Fig. 9, for example, the lower section involves four modules (as $d = 4$ and $i = 1$), and each module can be viewed as a DSCCC(3, 2). Likewise, the reliability of the upper section is the product of 2^{i+1} *independent* modules, each with the reliability lower bounded by that of $\mathrm{DSCCC}(d-i, d-1-i)$. (Note that a module now involves the set of cycles: $\{\text{cycle } (2^{i+1})j + k \mid 0 \leq j < 2^{d-1-i}$, for any $k, 0 \leq k < 2^{i+1}\}$.) The overall reliability of $\mathrm{DSCCC}_i^2(d+2, d)$ is bounded below by the product of the lower bounds of its two sections.

The reliability lower bound of $\mathrm{DSCCC}_i^2(10, 8)$ for i, ranging from 1 to 5, is shown in Fig. 10, where the PE failure rate λ is 1.0 per unit time (e.g., 10^6 hr), and the "lump" B/F link failure rate is $d\lambda$ for $\mathrm{DSCCC}_i^2(d+2, d)$. As expected, the highest lower bound happens as $i = 3$, namely, when SR_2 is placed in the middle. As i moves away from 3, the reliability value consistently falls off. The structure with SR_2 placed in the middle is the best. The dashed curve in the figure shows the reliability lower bound of a system with one single spare row, i.e., DSCCC(9, 8). As can be seen, the second spare row improves system reliability considerably.

VI. Concluding Remarks

A fault-tolerant cube-connected cycles (CCC) structure has been introduced. This structure achieves strong fault-tolerance by using a pair of PE's in the immediate higher dimension to replace a failed PE and its dimensionally connected PE, and is dubbed the DSCCC. A distributed reconfiguration process is provided to carry out pairwise dimensional substitution when a fault arises. The DSCCC exhibits a significant reliability improvement over its CCC counterpart without incurring large area overhead, making it favorable in the VLSI/WSI environments. It is also illustrated that more than one row of spare PE's can be added to the CCC to obtain an even more reliable structure aimed at a long mission time.

A "folded" layout for the CCC has been proposed recently by Shen and Koren [1], where the right half of the 2^d cycles in $\mathrm{CCC}(h, d)$ is placed above the left half, saving the area needed for the PE's and the L links in the highest dimension, i.e., dimension $d-1$. The DSCCC can readily take advantage of the "folded" layout strategy to reduce already low area overhead incurred by spare PE's and their connections, adding another potential benefit to the DSCCC design [8].

A slightly different fault-tolerant CCC results if 1) two rows of spare PE's are added to the immediate top of the highest dimension,

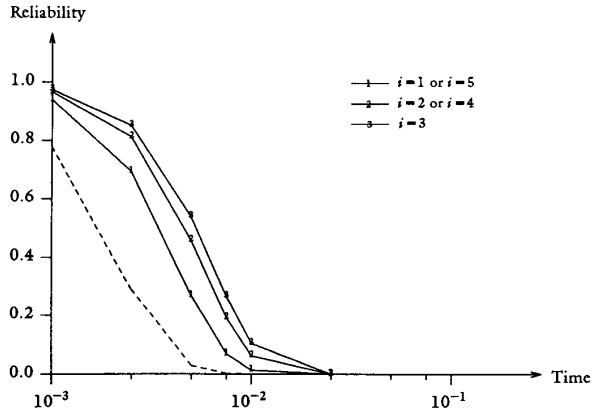

Fig. 10. Reliability comparison among $DSCCC_i^2(10, 8)$'s, $1 \leq i \leq 5$. (The dashed curve denotes the DSCCC(9, 8) result.)

and 2) PE's in dimension i are equipped with extra links that emulate dimensions $i-1$ and $i-2$ connections. The resulting structure is expected to have higher reliability than the $DSCCC_i^2$, but every PE then needs one more extra port, and its layout takes larger area. It may be interesting to contrast the cost-effectiveness of these two structures.

REFERENCES

[1] J.-J. Shen and I. Koren, "Yield enhancement designs for WSI cube connected cycles," *Proc. Int. Conf. Wafer Scale Integration,* 1989, pp. 289–298.

[2] F. P. Preparata and J. Vuillemin, "The cube-connected cycles: A versatile network for parallel computation," *Commun. ACM,* vol. 24, pp. 300–309, May 1981.

[3] M. S. Krishnan and J. P. Hayes, "A normalized-area measure for VLSI layouts," *IEEE Trans. Comput.-Aided Design,* vol. 7, pp. 411–419, Mar. 1988.

[4] I. Koren, Z. Koren, and D. K. Pradhan, "Designing interconnection buses in VLSI and WSI for maximum yield and minimum delay," *IEEE J. Solid-State Circuits,* vol. 23, pp. 859–866, June 1988.

[5] P. Banerjee, "The cubical ring connected cycles: a fault-tolerant parallel computation network," *IEEE Trans. Comput.*, vol. 37, pp 632–636, May 1988.

[6] S.-Y. Kuo and W. K. Fuchs, "Reconfigurable cube-connected cycles architectures," *J. Parallel Distrib. Computing*, vol. 9, pp. 1–10, May 1990.

[7] P. R. Lala, *Fault Tolerant and Fault Testable Hardware Design.* Englewood Cliffs, NJ: Prentice-Hall, 1985.

[8] N.-F. Tzeng, S. Bhattacharya, and P.-J. Chuang, "Fault-tolerant cube-connected cycles structures through dimensional substitution," *Proc. 1990 Int. Conf. Parallel Processing,* vol. I, Aug. 1990, pp. 433–440.

Optimal Processor Assignment for a Class of Pipelined Computations

Alok N. Choudhary, Bhagirath Narahari, David M. Nicol, and Rahul Simha

***Abstract*— The availability of large-scale multitasked parallel architectures introduces the following processor assignment problem. We are given a long sequence of data sets, each of which is to undergo processing by a collection of tasks whose intertask data dependencies form a series-parallel partial order. Each individual task is potentially parallelizable, with a known experimentally determined execution signature. Recognizing that data sets can be pipelined through the task structure, the problem is to find a "good" assignment of processors to tasks. Two objectives interest us: minimal response time per data set, given a throughput requirement, and maximal throughput, given a response time requirement. Our approach is to decompose a series-parallel task system into its essential "serial" and "parallel" components; our problem admits the independent solution and recomposition of each such component. We provide algorithms for the series analysis, and use an algorithm due to Krishnamurti and Ma for the parallel analysis. For a p processor system and a series-parallel precedence graph with n constituent tasks, we give a $O(np^2)$ algorithm that finds the optimal assignment (over a broad class of assignments) for the response time optimization problem; we find the assignment optimizing the constrained throughput in $O(np^2 \log p)$ time. Our techniques are applied to a task system in computer vision.**

I. INTRODUCTION

In recent years, much research has been devoted to the problem of mapping large computations onto a system of parallel processors. Various aspects of the general problem have been studied, including different parallel architectures, task structures, communication issues, and load balancing [8], [13]. Typically, experimentally observed performance (e.g., speedup or response time) is tabulated as a function of the number of processors employed, a function sometimes known as the *execution signature* [10], or the *response time* function. In this short note, we use such functions to determine the number of processors to be allocated to each of several tasks when the tasks are part of a pipelined computation. This problem is natural, given the growing availability of *multitasked* parallel architectures, such as PASM [29], the NCube system [14], and Intel's iPSC system [5], in which it is possible to map tasks to processors and allow parallel execution of multiple tasks in different logical partitions.

We consider the problem of optimizing the performance of a complex computation applied to each member of a sequence of data sets. This type of problem arises, for instance, in imaging systems, where each image frame is analyzed by a sequence of elemental tasks, e.g., fast Fourier transform or convolution. Other applications include network software, where packets are pipelined through well-defined functions such as check-sum computations, address decoding, and framing. Given the data dependencies between the computation's

Manuscript received August 3, 1991; revised April 27, 1992, and March 6, 1993. This work was supported in part by the National Science Foundation (NSF) under Grants MIP–9110810, ASC–8819393, and NCR–8907909, and in part by the National Aeronautics and Space Administration (NASA) under Grant NAG–1–995.

A. N. Choudhary is with the Department of Electrical and Computer Engineering, Syracuse University, Syracuse, NY 13244.

B. Narahari is with the Department of Electrical Engineering and Computer Science, George Washington University, Washington, DC 20052.

D. M. Nicol and R. Simha are with the Department of Computer Science, College of William and Mary, Williamsburg, VA 23185.

IEEE Log Number 9215379.