RECONFIGURABILITY AND RELIABILITY OF SYSTOLIC/WAVEFRONT ARRAYS

Edwin Hsing-Mean Sha
Kenneth Steiglitz

CS-TR-280-90

August 1990

Reconfigurability and Reliabilty of Systolic/Wavefront Arrays[1]

Edwin Hsing-Mean Sha

Kenneth Steiglitz

Department of Computer Science

Princeton University

Princeton, NJ 08544

ABSTRACT

In this paper we study fault-tolerant redundant structures for maintaining reliable arrays. In particular we assume the desired array (*application graph*) is embedded in a certain class of regular, bounded-degree graphs called *dynamic graphs*. We define *the degree of reconfigurability* DR, and DR *with distance* DR^d, of a redundant graph. When DR (respectively DR^d) is independent of the size of the application graph, we say the graph is *finitely reconfigurable,* FR (resp. *locally reconfigurable,* LR). We show that DR provides a natural lower bound on the time complexity of any distributed reconfiguration algorithm, and that there is no difference between being FR and LR on dynamic graphs. We then show that if we wish to maintain both local reconfigurability, and a fixed level of reliability, a dynamic graph must be of dimension at least one greater than the application graph. Thus, for example, a one-dimensional systolic array cannot be embedded in a one-dimensional dynamic graph without sacrificing either reliability or locality of reconfiguration.

[1]This work was supported in part by NSF Grant MIP-8912100, and U. S. Army Research Office-Durham Grant DAAL03-89-K-0074.

1 Introduction

Highly parallel pipelined structures such as systolic or wavefront arrays are attractive architectures for achieving high throughput [HKu]. Examples of important potential applications include digital signal processing [SKu, CaSt], and large-scale scientific computation on arrays for solving partial differential equations [KAGB], or simulating lattice-gas automata [KuSt]. As such array processors become larger, the reliability of the processing elements (PE's) becomes a critical issue, and it becomes necessary to use fault-tolerant techniques — both at the time of fabrication [LeLe] and at runtime. Defective PE's must be located, and the architecture reconfigured to substitute good PE's for bad.

In certain runtime applications like avionics and space flight, fault tolerant techniques must be able to restore proper operation after failures as fast as possible. For this purpose, distributed reconfiguration algorithms executed in parallel by the PE's themselves have been studied in [KuJeCh]. In this paper, we give a lower bound on the time required to reconfigure different redundant structures.

In most literature on fault tolerance, faults are confined to processing elements only and it is assumed that all switches and connections [ChLeRo, KuLa, GrGa, SaSt] are perfect. This is not valid when the number of switches and connections becomes large. In this paper we will use a graph model that takes into account failures of switches and interconnection wires as well as PE's. PE's and switches will be represented by nodes of the graph in the obvious way, and a connection between two elements in the computational structure will be represented by a node inserted in the edge between the appropriate two nodes in the graph model. Each node of the graph will have associated with it a probability of failure ε.

To achieve fault tolerance, we add redundancy to the system. After a failure the original working architecture is *reconfigured* by replacing some nodes that were being used by redundant nodes. A good fault tolerant structure is one where the number of nodes that need to be changed after failure is as small as possible. In this paper, we define a measure of this adaptability, the ***degree of reconfigurability*** (DR), and analyze this measure on a class of very regular graphs called ***dynamic graphs*** [Or, IwSt1, IwSt2, Iwst3]. We also analyze a stricter measure, called the ***degree of reconfigurability with distance***, DR^d, which takes into account the total distance between original nodes and replacing nodes. Our goal is to investigate the relation between the structure of dynamic graphs, their reliability, and their fault-tolerant capability as measured by their degree of reconfigurability.

The case when DR is independent of the size of the system is especially important because it represents the situation when the amount of change necessary to repair the system depends only on the number of failed nodes, but not on the size of the system. In this case, we say the graph is *finitely reconfigurable.* Similarly, if DR^d, the total distance cost of changes is independent of the size of system, we say that it is *locally reconfigurable.* Actually, we show if the redundant system is a dynamic graph, it is *locally reconfigurable* if and only if it is *finitely reconfigurable.* Given a desired working structure, we will discuss what kinds of redundant structures are possible or impossible to maintain at a fixed level of reliability, while at the same time being locally reconfigurable. In particular, our main result is that if we wish to maintain both local reconfigurability, and a fixed level of reliability, the dynamic graph must be of dimension at least one greater than the application graph.

2 Definitions and Mathematical Framework

A VLSI/WSI array architecture can be represented as a graph $G = (V, E)$. Each node of the graph G can be regarded as a processor, and an edge of G is a connection between two processors. We assume that the nodes failed independently, each with probability ϵ. As mentioned above, a node in our graph model can represent a PE, a switch, or interprocessor connection.

Real working architectures are considered to be a family of graphs, $\mathcal{G}_a$, called ***application graphs***; $G_a^i = (V_a^i,\ E_a^i)$ denotes the ith application graph of $\mathcal{G}_a$. For example, $\mathcal{G}_a$ can be a family of linear arrays indexed by number of nodes, so G_a^n is an n-node linear array. We always assume each G_a^i is connected and that for each value of n, there exists a unique i. Since we need to add redundant nodes or edges to increase reliability, the embedding structures, $\mathcal{G}_r$, called ***redundant graph***, are also represented as a family of graphs; $G_r^i = (V_r^i,\ E_r^i)$ denotes the ith redundant graph of $\mathcal{G}_r$. Each pair of nodes in V_r^i is associated with a value, ***distance***, defined by a function $D^i : V_r^i \times V_r^i \rightarrow N$, where N is the set of natural numbers; $D^i(a,\ a) = 0$. This ***distance*** can be regarded as the physical distance between two nodes, or some cost, such as the communication cost.

Given two isomorphic graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$, define the isomorphism function μ: $V_1 \rightarrow V_2$ such that $(v_i,\ v_j) \in E_1$ iff $(\mu(v_i), \mu(v_j)) \in E_2$. Let $\mu(V_1)$ be the image of V_1. Given an isomorphism function $\mu : V_1 \rightarrow V_2$, let the mapping set $S(\mu)$ be the set of pairs, $\{(v, \mu(v)) |\ v \in V_1\}$. Thus, $S(\mu) - S(\mu')$ represents the difference between two isomorphism functions μ and μ'.

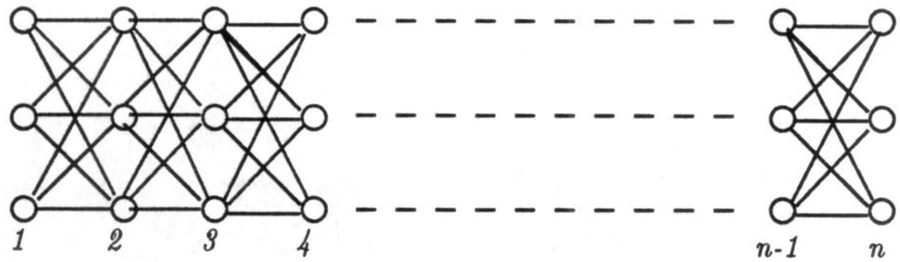
Given $\mathcal{G}_a$ and $\mathcal{G}_r$, the following function will determine which graph in $\mathcal{G}_r$ will be the redundant graph of the ith application graph.

Definition 2.1 *An Embedding Strategy for $\mathcal{G}_a$ and $\mathcal{G}_r$ is a function $ES : \mathcal{G}_a \rightarrow \mathcal{G}_r$, i.e., if $ES(G_a^i) = G_r^j$, G_r^j is the redundant graph for G_a^i.*

If $ES(G_a^i) = G_r^j$, and k nodes of G_r^j have failed, the failed nodes and all the edges incident to them will be removed and G_r^j becomes a new subgraph $\hat{G_r^j} = (\hat{V_r^j},\ \hat{E_r^j})$. The procedure of finding a new isomorphism function μ_k^i: $V_a^i \rightarrow \hat{V_r^j}$ is called ***reconfiguration***.

Definition 2.2 *Given $\mathcal{G}_a$, $\mathcal{G}_r$ and ES, the maximum fault-tolerance of G_a^i, $MFT(G_a^i)$, is the maximum number of nodes that can be allowed to fail arbitrarily in $ES(G_a^i)$ such*

G_a^n: n-node linear array



G_r^n: n-triple-modular-redundancy (TMR) array

Figure 1: Example of $\mathcal{G}_a$ and $\mathcal{G}_r$.

that $ES(G_a^i)$ can still find a subgraph isomorphic to G_a^i. In addition, $FT(G_a^i)$ is given which is some fixed number $\leq MFT(G_a^i)$ for each i.

Definition 2.3 *Given $\mathcal{G}_a$, $\mathcal{G}_r$, ES and Fault Tolerance $FT(G_a^i) \leq MFT(G_a^i)$ for each i, the quadruple (G_a, G_r, ES, FT) is called an Embedding Architecture, EA.*

For example in figure 1, $\mathcal{G}_a$ is a family of linear arrays, and $\mathcal{G}_r$ is a family of triple-modular-redundancy(TMR) arrays obtained by triplicating each node of a linear array to be three nodes, called a *module.* Let $G_r^n = ES(G_a^n)$ be the n-module array, and let its corresponding $FT(G_a^n)$ be 2 for all n.

For simplicity, if the context is clear, we will always assume the ith application graph maps to the ith redundant graph, i.e., $ES(G_a^i) = G_r^i$. Let $\mu_0^i : G_a^i \to G_r^i$, be the initial isomorphism function for the ith application graph G_a^i.

Definition 2.4 *Given an Embedding Architecture, define the Initial Embedding, IE, to be a set of μ_0^i for all G_a^i in the family.*

For the above example in figure 1, an *initial embedding* can be a set of μ_0^i such that each node of G_a^i maps to the bottom node of each module of G_r^i.

Given an *embedding architecture* for a G_a^i, after k nodes have failed , obviously there may be many different isomorphism functions μ_k's. But, the difference between $S(\mu_0^i)$ and $S(\mu_k^i)$ should be as small as possible for the purpose of real-time fault-tolerance.

Suppose that the number of nodes in G_a^i is n. Given EA, IE and that $k \leq FT(G_a^i)$ nodes have failed, let the *cost of reconfiguration* of G_a^i, $\Delta(k,n)$, be the minimum of $|S(\mu_0^i) - S(\mu_k^i)|$ over all the possible isomorphism functions μ_k^i, i.e.,

$$\Delta(k,n) = \min_{\mu_k^i} |S(\mu_0^i) - S(\mu_k^i)|.$$

When there is no μ_k^i, $\Delta(k,n) = \infty$. We also want to measure the total distance between original nodes and replacing nodes after reconfiguration. The total *distance* cost of reconfiguration for G_a^i, $\Delta^d(k,n)$ is similarly defined to be the following:

$$\Delta^d(k,n) = \min_{\mu_k^i} \sum_{(a,b)\in S(\mu_0^i)-S(\mu_k^i)} D^i(\mu_k^i(a),\ b).$$

When there is no μ_k^i, $\Delta^d(k,n) = \infty$. Under a given EA and IE, let $DR(k,\ n)$, the *Degree of Reconfigurability for* G_a^i , be the maximum of $\Delta(k,n)$ over all possible k failures in G_r^i, $k \leq FT(G_a^i)$; i.e.,

$$DR(k,\ n) = \max_{\substack{\text{failures of k nodes} \\ k\leq FT(G_a^i)}} \Delta(k,n).$$

The *Degree of Reconfigurability with distance*, $DR^d(k,\ n)$, is defined similarly (change Δ to be Δ^d in the above equation).

Return to the example in figure 1. Let the *distance* between two nodes in the same module be one, and the distance between two nodes, one in module i and the other in module j, be $|i-j|+1$. In this case $DR(k,\ n)$ and $DR^d(k,\ n)$ for G_a^n are both k, since for any $k \leq FT(G_a^n) = 2$ faults, we need only change k nodes in the same modules as the k faulty nodes, and the distance between two nodes in the same module is one.

Definition 2.5 *An Embedding Architecture, EA is finitely reconfigurable (resp. locally reconfigurable), if there exists an Initial Embedding, IE, such that for all the* $G_a^i \in \mathcal{G}_a$, $DR(k,n)$*(resp.* $DR^d(k,n)$*), can be bounded from above by a function of* k *but not* n.

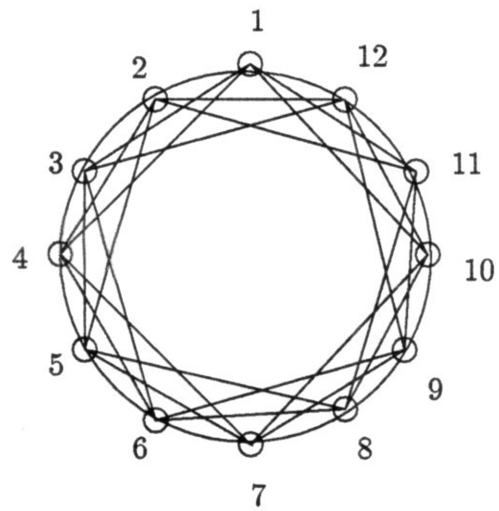


Figure 2: Hayes' 4-FT single loop.

For example, the *embedding architecture* for linear arrays in the example above is both LR and FR, since for each G_a^i, $DR(k,n) = DR^d(k,n) \leq k$.

We show in the following lemma that Hayes' h-FT $(n+h)$-node single loop [Ha], which is an h-fault-tolerant graph for an n-node loop application graph, is not finitely reconfigurable.

The nth application graph G_a^n is an n-node single loop, and the *embedding strategy* is to map G_a^i to its so-called Hayes' h-FT $(n+h)$-node single loop. Thus, G_r^n is defined by the following procedure, where we assume for this example that h is even.

1) Form a single-loop graph C_{n+h} with $n+h$ nodes.

2) Join every node x_i of C_{n+h} to all nodes at index distance j from x_i, for all j satisfying $2 \leq j \leq \frac{h}{2}+1$.

The resulting graph G_r^n is an h-FT $(n+h)$-node single-loop graph. Hayes [Ha] shows that its $MFT(G_a^n) = h$. Let the distance between node x_i and x_j be $|i-j|$ mod $n+h$. All the computations in the proof are based on indices mod $n+h$, and all the indices are in G_r. The graph in Figure 2 is an example for $n=8$, $h=4$.

Lemma 2.1 The above *embedding architecture* with $FT = MFT = h$, mapping the n-node single loop to Hayes' h-FT $(n+h)$-node single-loop graph, is neither FR nor LR if h is $o(n^{\frac{1}{2}})$.

Proof: We assume there is an adversary A who always tries her best to select failures that show that $DR(k,\, n)$ is not bounded by a function of k only. No matter what the initial μ_0^n is, n working nodes must be distributed among the $n+h$ nodes of G_r^n. Define a *segment* S to be a sequence of consecutively numbered working nodes $(x_i, x_{i+1}, \ldots, x_j)$ in G_r^n , where x_{i-1} and x_{j+1} are non-working redundant nodes. Denote the length of the segment S by $l(S) = j - i + 1$, and suppose the h non-working nodes, ordered by their indices, form the sequence $(x_{i_1}, x_{i_2}, \ldots, x_{i_h})$. For each x_{i_j} there is a segment S_j (it may be null) starting from x_{i_j+1}. Thus,

$$\sum_{j=1}^{h}(l(S_j)+1) = n+h.$$

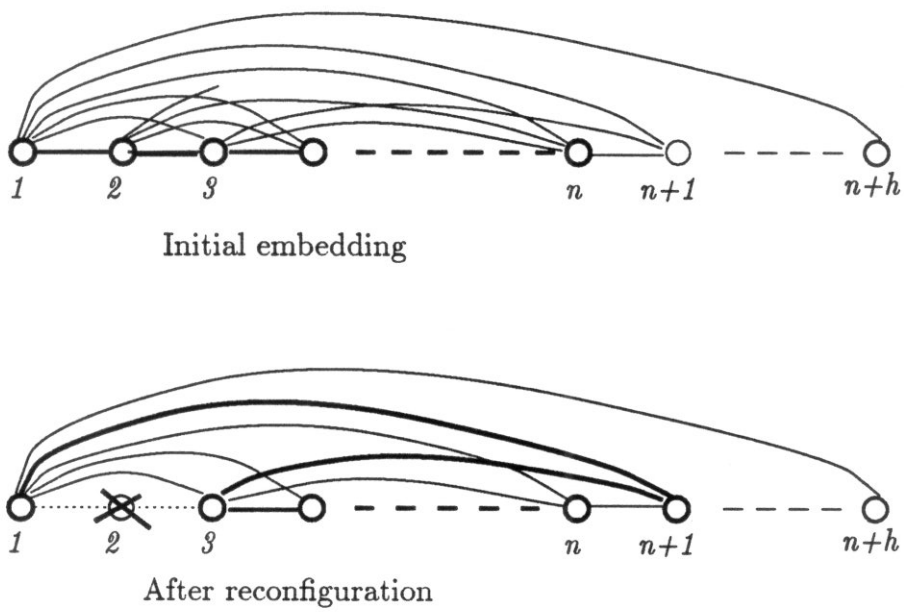
There must exist a segment S^* such that $l(S^*)+1 \geq \dfrac{n+h}{h}$; i.e., $l(S^*) \geq \dfrac{n}{h}$. Without loss of generality, assume that S^* is from node x_1 to node $x_{l(S^*)}$.

The adversary can choose the middle node x_d of segment S^* to be faulty, that is $d = \lceil \dfrac{n}{2h} \rceil$. Pick a reconfiguration that is optimal in the sense that the fewest possible number of nodes in G_r^{n+h} are changed. Let m be the number of nodes in S^* which are changed in this reconfiguration, Let C be such a sequence of m nodes, $(x_{j_1}, x_{j_2}, \ldots, x_{j_m})$, ordered by their indices. We know x_d must be replaced by one node, say x'_d, and if x'_d is a working node, it must be replaced by another node. Thus, there is a sequence $\subseteq C$ of working nodes in S^* in this sequence of replacements, starting with x_d and ending at a working node that is replaced by the first node x_r outside S^*. First, we divide S^* into many small subsegments with length w, where $w = 2 \cdot (\dfrac{h}{2}+1)$, and represent them as a sequence $(S_1^*, \ldots, S_k^*)$. Let x_d be in subsegment S_i^*. Without loss of generality, assume that the index of x_r is larger than the largest index of a node in C; i.e., $r > j_m$.

We claim that there must exist at least one node in C in the subsegment S_k^* or S_1^*. Suppose not. Let x_r replace x_i in C and let a and b be the two nodes connected to x_i

in the initial working subgraph. Since connections must be of length at most $\frac{h}{2}+1$ and the distance between x_i and the last node in S^* (and also the first node in S^*) is $> w$, we know a and b must be in S^*. If a or b is not in C, say a, because a is not replaced, x_r must be connected to a after the reconfiguration. But we know that $i \leq j_m$ and $r > l(S^*)$ from the assumption, so it is impossible that x_r is connected to a. Thus, we know that a and b are in C, say that a is replaced by a'. Denote the sequence of original working nodes starting from x_i toward one direction in the original working subgraph by $\{x_i, a, a_1, a_2, \ldots\}$, and the sequence after reconfiguration by $\{x_r, a', a'_1, a'_2, \ldots\}$. If $a' \in S^*$, because a' replaces a, a' must be in C. Since the index of a' is $\leq j_m$, it is impossible for a' to be connected to x_r. Thus, a' is not in S^*. In summary, we know that if $x_i \in C$ and $x_r \notin S^*$, then a is in C and a' is not in S^*. Repeating the argument, using a instead of x_i and a' instead of x_r, we can get the result that a_1 is in C and a'_1 is not in S^*. Continuing in this way, it follows that all the nodes $a, a_1, a_2, \ldots$ are in C and nodes $a', a'_1, a'_2, \ldots$ are not in S^*, but this is impossible, since there are only finite number of nodes in C. Thus, our claim is correct.

We claim next that in each pair of the subsegments $(S^*_l,\ S^*_{k-l+1})$, where $l = 1, \ldots, i$, there exists at least one node in C. We have proved that it is true for the first pair of subsegments $(S^*_1,\ S^*_k)$. Assume it is true for all the pairs of subsegments from $l = 1$ to $k - j$, and $i < j$. We represent $C' = \{x_j |\ x_j \in C, x_j \text{ not in } S^*_1, \ldots, S^*_{k-j}, \text{ and } S^*_{j+1}, \ldots, S^*_k\}$. Since $x_d \in C'$, from the way that x_r is chosen we know there must exist one node in C' which is replaced by a node outside of C'. If, in S^*_{k-j+1} and S^*_j, there does not exist a node in C', the same argument as above results in the same contradiction. Thus, in each pair of subsegments in S^*, there is at least one node which has been replaced. The number of nodes in C must therefore be at least $n/2hw = \Omega(n/h^2)$. If $h = o(n^{\frac{1}{2}})$, a number of nodes that is an unbounded function of n need to be changed. Thus, $DR(k,\ n)$ is not bounded by a function of k only, under any initial isomorphism function μ^n_0, and therefore the Hayes' *embedding architecture* is not finitely reconfigurable. It is obvious that the total distance between original nodes and their replacing nodes



G_r^n: $(n+h)$-node complete graph

Figure 3: An example that is FR but not LR.

is also an increasing function of n, so it is not LR either. □

Our next example is an *embedding architecture* that is *finitely reconfigurable*, but not *locally reconfigurable*. Choose $\mathcal{G}_a$ as in figure 1 to be a family of linear arrays, and $\mathcal{G}_r$ as in figure 3 to be a family of complete graphs on a row. Let ES map G_a^n to G_r^{n+h} and let $FT(G_a^n) = h$, for each G_a^n in $\mathcal{G}_a$. The *distance* between node i and node j is defined to be $|i - j|$. After one node has failed, say node 2, we can take any spare node to replace it , say node $n+1$, as shown in figure 3.

Lemma 2.2 If h is $o(n)$ the above *embedding architecture* is FR, but not LR.

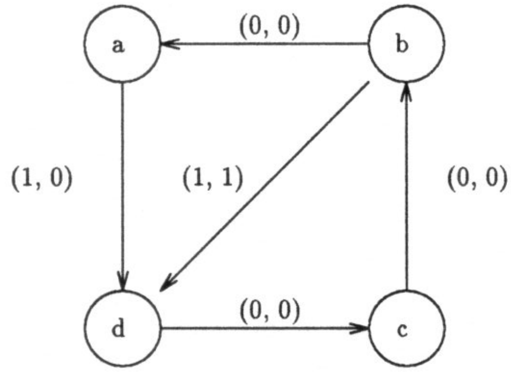
Proof: It is obvious that such an EA is *finitely reconfigurable*, since any spare node can replace any other node, so that only k faulty nodes need be changed after k nodes fail. Considering G_a^n and G_r^{n+h}, under any initial embedding, there must exist a sequence of working nodes in G_r^{n+h} with consecutive indexes of length $\geq n/(h+1)$, by the same argument as in lemma 2.1. Choosing the middle node of such a path to be faulty, the distance between any spare node and the faulty node must be $\geq n/(2(h+1))$. Since $h = o(n)$, the distance is an increasing function of n. Thus, this EA is not *locally reconfigurable*. □

3 Degree of Reconfigurability for Dynamic Graphs

In applications we are interested in graphs which are very regular and of bounded degree. An interesting and useful class of such graphs are called *dynamic graphs* [Or, IwSt1, IwSt2, IwSt3], which model regular systolic and wavefront arrays in a natural way. An undirected k-dimensional *dynamic graph* $G^k = (V^k, E^k, T^k)$ is defined by a finite digraph $G^0 = (V^0, E^0)$, called *the static graph*, and a k-dimensional labeling of edges $T^k : E^0 \to Z^k$. The vertex set V_x is a copy of V^0 at the integer lattice point x and V^k is the union of all V_x, where $x \in Z^k$. Let a_x be the copy of node $a \in V^0$ in the vertex set V_x and let b_y be the copy of node $b \in V^0$ in the vertex set V_y. Nodes a_x and b_y are connected if $(a, b) \in E^0$ and the difference between the two lattice point y and x is equal to the labeling $T^k(a, b)$. Therefore, the dynamic graph is a locally-finite infinite graph consisting of repetitions of the basic cell V^0 interconnected by edges determined by the labeling T^k. In figure 4, we show an example of a two-dimensional static graph G^0 and its corresponding dynamic graph G^2.

For $x, y \in Z^k$, let $E_{x,y} = \{(a_x, b_y) \mid (a, b) \in E^0\}$. The graph with vertex set V_x and edges with both end points only in V_x is called the *x-th cell* of G^k, $C_x = (V_x, E_{x,x})$. Given a dynamic graph, we can contract all the nodes in the same cell to one node and delete the edges totally within the cell. This contracted graph is called the *cell-dynamic graph*, $G_c = (V_c, E_c)$, where $V_c = Z^k$ and $E^c = \bigcup_{x \neq y} E_{x,y}$. We give an example in figure 5, which is the cell-dynamic graph corresponding to G^2 in figure 4.

Given a static graph G^0, we define F_j to be the finite subgraph of G^k such that



A static graph $G^0 = (V^0, E^0)$

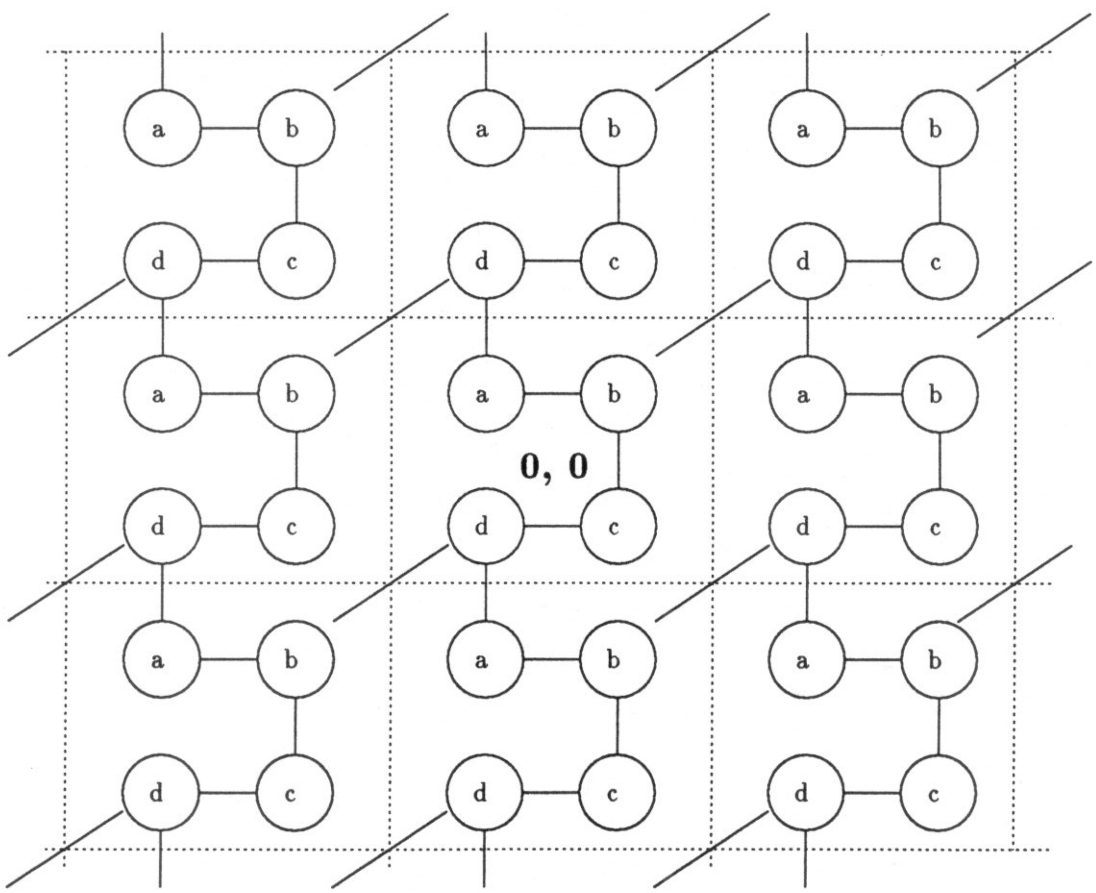


Figure 4: An example of G^0 and the corresponding dynamic graph G^2.

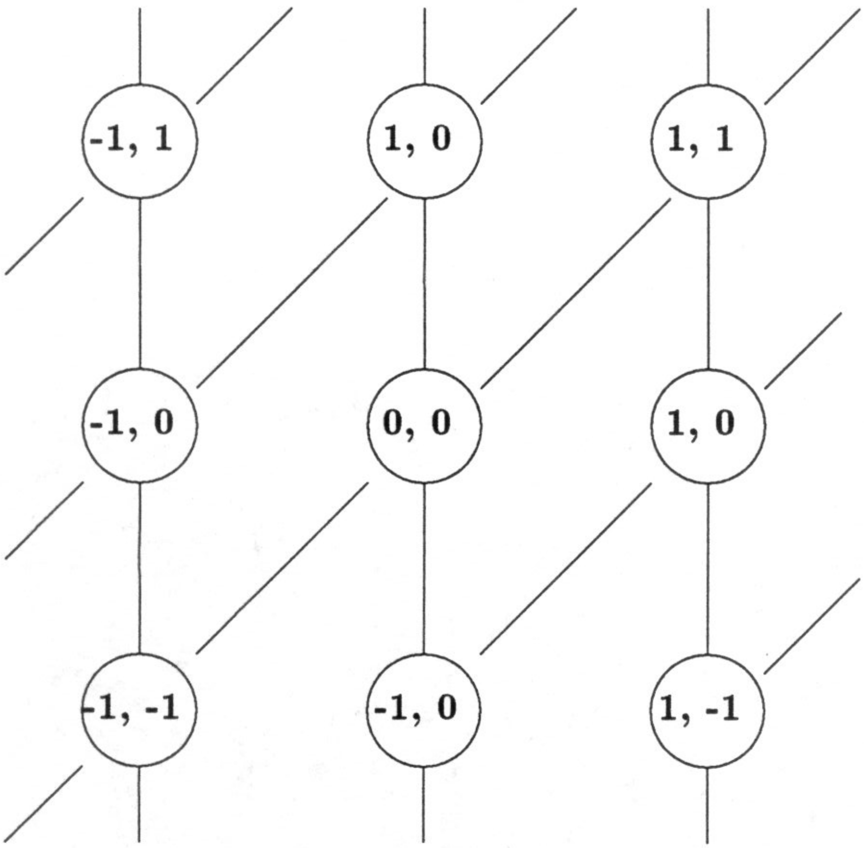


Figure 5: The cell-dynamic graph G_c of G^2.

each dimension of F_j has j cells, i.e., $F_j = (\bigcup_x V_x, \bigcup_{x,y} E_{x,y})$, where $x = (x_1, x_2, \ldots, x_k)$, $1 \leq x_i \leq j$, and $y = (y_1, y_2, \ldots, y_k)$, $1 \leq y_i \leq j$. We define the family $\mathcal{F}$ of k-dimensional dynamic graphs to be the set of F_j, where $j \geq 1$.

There are different ways to define distance in dynamic graphs. For example, one reasonable definition of the distance function D is to define the distance between two nodes, one in vertex set V_x, and the other in V_y, to be the Euclidian distance in k-dimensional space between point x and point y if x and y are in different cells, and one if they are in the same cell. We say that a distance function D satisfies property $\bigtriangledown$ (triangle inequality), if the distance between nodes a and b is less than or equal to the total distance of any path from a to b. Of course Euclidian distance satisfies $\bigtriangledown$. The following lemma will show that when the set of redundant graphs $\mathcal{G}_r$ is a family of dynamic graphs and the distance function satisfies $\bigtriangledown$, then any *embedding architecture* is LR if and only if it is FR. In the rest of paper, we assume that D satisfies property $\bigtriangledown$.

Lemma 3.1 When $\mathcal{G}_r$ is a *family of dynamic graphs* and its distance function satisfies $\bigtriangledown$, the *embedding architecture* is *locally reconfigurable* if and only if it is *finitely reconfigurable.*

Proof: Given an EA, if this EA is LR, we know by definition that the *total distance cost* of any k failures can be expressed as a function $f(k)$, where f is a function of k only. We know the distance between any two nodes is at least one, so the number of nodes changed must be $\leq f(k)$. Thus, this EA is also FR.

Suppose that it is FR. We know that for each $G_a^n \in \mathcal{G}_a$, after k nodes have failed, at most a function of k, say $f(k)$, nodes must be changed in the original working subgraph. Let a_1 be the node in G_a^n such that the distance in G_r^n between $\mu_k^n(a_1)$ and $\mu_0^n(a_1)$ is the maximum over all the nodes in V_a^n.

Because there are at most $f(k)$ nodes which are changed by μ_k^n, there exists a path in the application graph G_a^n with at most $f(k)$ edges from a_1 to an unchanged node a_2, i.e. $\mu_0^n(a_2) = \mu_k^n(a_2)$. Let c be the maximum distance between any two nodes connected by an edge, which is a constant independent of k and n by definition. The distance D between node $\mu_0^n(a_1)$ and node $\mu_0^n(a_2)$ is at most $c \cdot f(k)$ by property $\bigtriangledown$, the triangle ineqality. Similarly, the distance between node $\mu_k^n(a_1)$ and node $\mu_k^n(a_2)$ is at most $c \cdot f(k)$. Since $\mu_k^n(a_2) = \mu_0^n(a_2)$, the distance between $\mu_0^n(a_1)$ and $\mu_k^n(a_1)$ is at most $2c \cdot f(k)$. Therefore the total distance of the $f(k)$ changed nodes is at most $2c \cdot f(k)^2$ because there are at most $f(k)$ pairs that are changed. EA is therefore locally reconfigurable from the definition. □

Finite reconfigurability is desirable in practice, especially for real-time fault tolerance, because it shows that after k nodes have failed, at most a function of k nodes need to be changed, independent of the size of the application graph. Lemma 3.2 will show that the degree of reconfigurability DR provides a lower bound on the time complexity of any distributed reconfiguration algorithm, and shows one reason this measure DR is important. We assume in what follows that it takes one time step to send a message through an edge.

Lemma 3.2 When G_a^i is an n-node application graph and $\mathcal{G}_r$ is a family of d-dimensional dynamic graphs, the time complexity of any distributed reconfiguration algorithm, is $\Omega(\left(\frac{DR}{k}\right)^{\frac{1}{d}})$, where k is the number of nodes that have failed.

Proof: After k nodes have failed, we must change at least DR nodes to reconfigure. We can assume that a distributed reconfiguration algorithm is initiated by a neighbor node, called a source node, of each faulty node after this neighbor node has detected the failure. We need to inform at least DR nodes in G_r^i that they are assigned different nodes in G_a^i. Thus, the time to broadcast this fault information is a lower bound on the time complexity of any distributed reconfiguration algorithm.

Let the corresponding *static graph* be $G^0 = (V^0, E^0)$, and its labelling be T^d. The maximum edge distance c in one dimension is the max $\{|t_i| \mid (t_1, \ldots, t_i, \ldots, t_d) \in T^d(e), e \in E^0\}$. Let m be equal to $(|V^0| \times 2c)^d$. We can always contract the nodes of G^d into groups of at most m nodes to obtain a d-dimensional *reduced graph* $G_c' = (V_c', E_c')$, such that $V_c' = Z^d$ and $E_c' = \{(x,y) \mid x, y \in V_c', x \neq y, y - x = (e_1, \ldots, e_i, \ldots, e_d)$ where $e_i = 0$ or 1 $\}$. Each node of V_c', called a *class* here, represents at most m nodes of the dynamic graph. Note that m is a constant by definition.

After t time steps, one source node can inform at most $(2 \cdot t)^d$ classes in a d-dimensional reduced graph, so at most $(2 \cdot t)^d \cdot m$ nodes have been reached. Since there are at most $c_1 k$ source nodes, where c_1 is the maximum degree in G_r, the total number of nodes that can be informed after t time steps is at most $(2 \cdot t)^d \cdot mk$. There are DR nodes that need to be informed, so t should be at least $\Omega(\left(\frac{DR}{k}\right)^{\frac{1}{d}})$. □

4 Impossibility of an LR-reliable Embedding of Dynamic Graphs from Dimension d to d

In this section we restrict attention to dynamic graphs, and consider the relationship between reconfigurability and reliability. In particular, we ask whether a given *embedding architecture* can be finite and locally reconfigurable, and at the same time maintain a given level of reliability. Without the constraint of being FR or LR, we can simply construct a redundant graph to be many replications of the application graph, achieving high reliability, but at the price of using large amounts of hardware and being difficult to reconfigure. Our main result is Theorem 4.5: when mapping from d-dimensions to d-dimensions, we cannot maintain both local reconfigurability and reliability simultaneously.

As lemma 3.1 shows, there is no difference between *local* and *finite reconfigurability* for dynamic graphs, and thus we consider only *local reconfigurability*, without the loss of generality. We define *LR-reliability* in our framework as follows. Given an EA which is LR, the probability, for each i, that G_r^i contains an isomorphic image of G_a^i is

$$P(G_a^i) = \sum_{k=0}^{FT} \epsilon^k (1-\epsilon)^{n-k} \binom{n}{k},$$

where $n = |V_r^i|$. The following definition replaces definition 2.5 in the statistical case.

Definition 4.1 *An Embedding Architecture is LR-reliable with reliability β, if $P(G_a^i) \geq \beta$ for all the $G_a^i \in \mathcal{G}_a$.*

The following lemma is useful in what follows.

Lemma 4.1 Given $\mathcal{G}_a$, $\mathcal{G}_r$ and ES, for each i, let $MFT(G_a^i)$ be the maximum number of failures that allows the corresponding EA to be LR. If this MFT is upper-bounded by a constant as $n \to \infty$, there exists a constant β such that EA cannot be *LR-reliable* with reliability β.

Proof: Let the upper bound on MFT be c. By the definition of MFT in the hypothesis of the lemma, there exist $c+1$ nodes in the redundant graph G_r^i such that after they have failed, for any IE, EA cannot be LR. Therefore $P(G_a^i) < \sum_{k=0}^{c+1} \epsilon^k (1-\epsilon)^{n-k} \binom{n}{k}$. We know n can be chosen large enough to make $c+1 < \epsilon n$, so the term corresponding to $k = c+1$ is the largest in the summation. Thus, the probability $P(G_a^i) < (c+1)(1-\epsilon)^{n-c-1} \binom{n}{c+1}$. Since $(1-\epsilon)^{n-c-1} \leq e^{-\epsilon(n-c-1)}$, and $\binom{n}{c+1} \leq \frac{n^{c+1}}{(c+1)!}$, it is obvious that when n goes to ∞, $P(G_a^i)$ goes to 0. Thus, for some i, we always can pick a $\beta > P(G_a^i)$. Therefore, such an *Embedding Architecture* cannot be *LR-reliable* with reliability β. □

We want to study some properties of dynamic graphs if we insist on local reconfigurability after some nodes have failed, since local reconfigurability is desirable in practical implementations. The following lemma tells us that one-dimensional *dynamic graphs* cannot be *LR-reliable* when the application graphs are linear arrays.

Lemma 4.2 When $\mathcal{G}_a$ is a family of one-dimensional linear arrays and $\mathcal{G}_r$ is a family of one-dimensional dynamic graphs, there exists a constant β such that no *Embedding Architecture* is *LR-reliable* with reliability β.

Proof: As in the proof of lemma 3.2, we can always build a *reduced graph* $G_c' = (V_c', E_c')$ by contracting sets of size at most m nodes in G_r^n to produce a one-dimensional linear array. Each node of G_c' now represents a *class* of a finite number of nodes. Note that m is a constant number, since G^0 is a finite graph by definition.

For any *initial embedding*, the n nodes of G_a^n are distributed into at least n/m contiguous classes in G_c'. If the adversary chooses all the nodes in the middle class of the above n/m classes to be faulty, the initial working subgraph is separated into two halves. We must shift at least half of the G_a^n and therefore change $\Omega(n)$ nodes to get a new working subgraph. Thus, if an *embedding architecture* is locally reconfigurable, its FT must be bounded by a constant m. From lemma 4.1, we know there exists a constant β, such that EA cannot be *LR-reliable* with reliability β. □

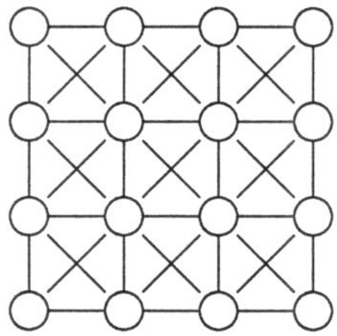


Figure 6: Example of a 2-dimensional 16-node web.

To generalize lemma 4.2, we define an n^d-node d-dimensional *web* to be a d-dimensional graph $G_l = (V_l, E_l)$ such that $V_l = \{x = (x_1, x_2, \ldots, x_d) |$ where $x_i = 0, \ldots, n-1\}$ and $E_l = \{(x, y) \mid x, y \in V_l, x \neq y, y - x = (e_1, \ldots, e_i, \ldots, e_d)$ where $e_i = 0$ or 1 $\}$. Thus, we connect all adjacent points in the d-dimensional Euclidian space. For example, figure 6 shows a 2-dimensional 16-node web. The family of d-dimensional webs is indexed by n.

Theorem 4.3 If $\mathcal{G}_a$ is a family of d-dimensional webs and $\mathcal{G}_r$ is a family of d-dimensional dynamic graphs, there exists a constant β such that no *Embedding Architecture* is *LR-reliable* with reliability β.

Proof: We can always find a d-dimensional *reduced graph* $G'_c = (V'_c, E'_c)$ by contracting the dynamic graph G^n_r as we did in the proof of lemma 3.2. Without loss of generality, we consider the most general case with all possible edges present, where $V'_c \subset Z^d$ and $E'_c = \{(x, y) \mid x, y \in V'_c, x \neq y, y - x = (e_1, \ldots, e_i, \ldots, e_d)$ where $e_i = 0$ or 1 $\}$. Each node of V'_c represents a class of m nodes of G^n_r, where m is the constant in the proof of lemma 3.2.

First, we prove that there cannot be an *embedding strategy* that maps a d-dimensional web to $(d-1)$-dimensional dynamic graph. Suppose first an $n \times n$ two-dimensional lattice is projected to a one-dimensional dynamic graph. Among the n^2 nodes in the web, the vertices on the path from vertex $(0, 0)$ to $(0, n-1)$ must be projected to at most n consecutive classes. Similarly, each of the n paths horizontally from $(0, 0)$ through

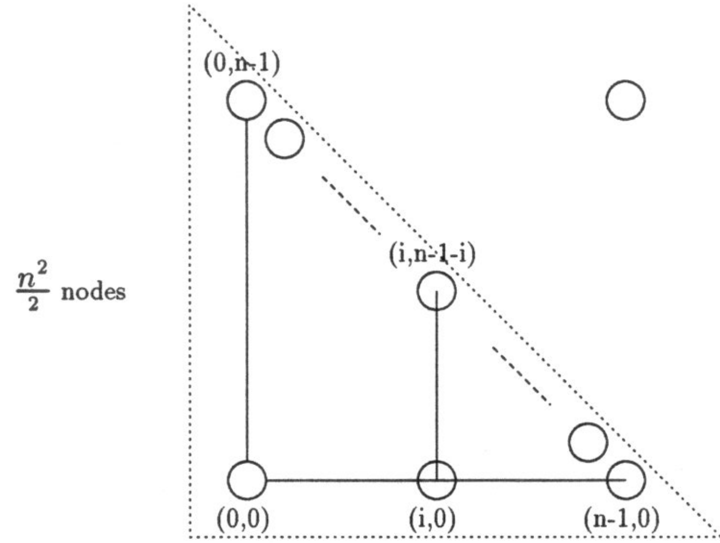


Figure 7: The n paths in the proof of theorem 4.3.

$(i, 0)$ and vertically to the diagonal vertices $(i, n-1-i)$ where $0 \leq i \leq n-1$ also must be projected to at most n consecutive classes. We show these n paths in figure 7. Thus, all the $n^2/2$ nodes on the paths must be in at most $2n$ classes, and there must exist one class to which at least $n/4$ nodes are mapped. This is impossible, since each class only has finite number of nodes. The same argument can be generalized easily to d-dimensional lattices. Thus, we can restrict attention to the possibility of mapping a d-dimensional web mapping to a d-dimensional dynamic graph.

We say a class in G'_c is *empty* if there is no working node in it. In the application graph the nodes which are adjacent must be mapped to one or adjacent classes. It is not hard to see that in the initial embedding there cannot be an empty class surrounded by non-empty classes. Consider a line of $\geq n$ nodes in the n^d-node d-dimensional web, as in the proof of lemma 4.2. For any *initial embedding* these n nodes are distributed into at least n/m classes that are linearly connected in G'_c. These images of lines may zig-zag in G'_c, but must map to at least n/m contiguous classes. Therefore, there is a well-defined *inner central class* which is $\Omega(n/m)$ classes away from the border in the image of the web, as shown in figure 8. Note that a line between the *inner central* class and the border may not be the image of a line along one dimension in the web, but the line must contains $\Omega(n)$ nodes in the web, as figure 8 shows.

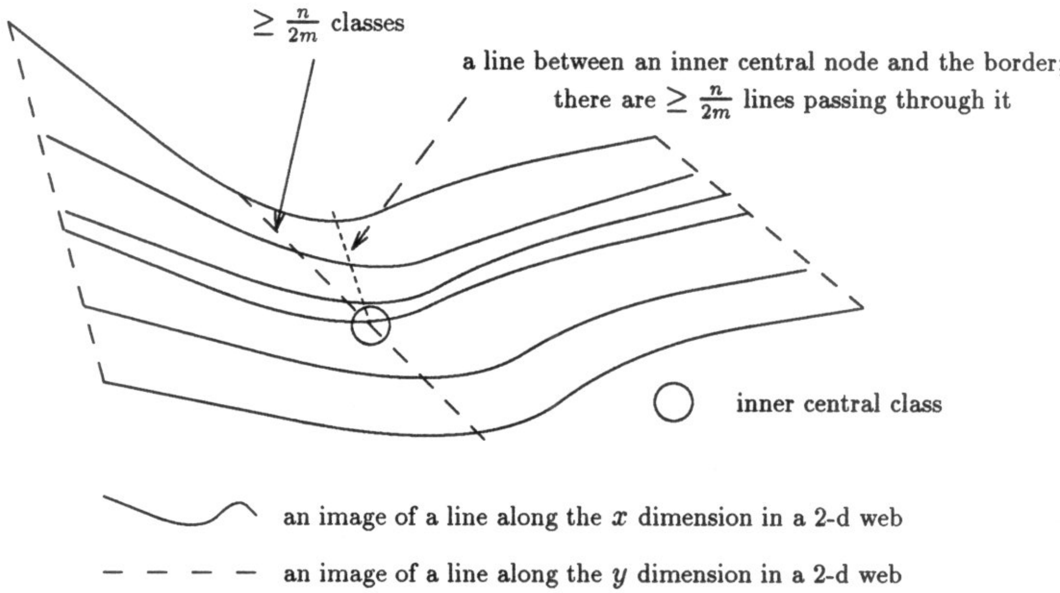


Figure 8: The inner central class in the proof of theorem 4.3.

If the adversary chooses all the nodes, at most m, in the *inner central* class to be faulty, the original working subgraph has a central inner hole. We must change $\Omega(n)$ nodes in one direction to get a new isomorphic subgraph in G_r^n. Therefore, to maintain local reconfigurability, for any *embedding architecture*, FT must be upper-bounded by m. From Lemma 4.1, we then know there exists a constant β, such that EA cannot be *LR-reliable* with reliability β □

We next modify the application graph so that each node $x = (x_1, x_2, \ldots, x_d)$ is connected only to nodes $y = (x_1, \ldots, x_i \pm 1, \ldots, x_d)$, $i = 1, \ldots, d$. We call such a d-dimensional graph a d-dimensional *orthogonal lattice*. To develop intuition for the general case of d-dimensional dynamic graphs, the following lemma extends theorem 4.3 to two-dimensional orthogonal lattices.

Lemma 4.4 If $\mathcal{G}_a$ is a family of two-dimensional orthogonal lattices and $\mathcal{G}_r$ is a family of two-dimensional dynamic graphs, there exists a constant β such that no *embedding architecture* is *LR-reliable* with reliability β.

Proof: As in the proof of theorem 4.3, we know that a two-dimensional *orthogonal lattice* cannot be embedded in a one-dimensional dynamic graph (we made no use of

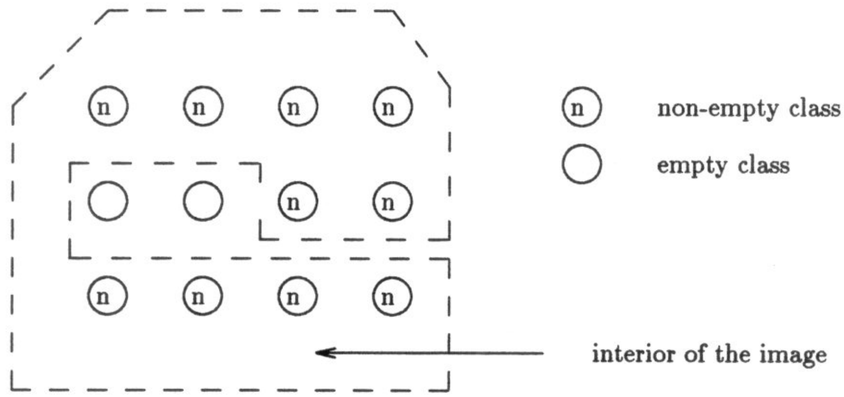


Figure 9: A pseudo hole.

diagonal edges in that proof). Without diagonal edges, however, the rest of the proof is a bit more complicated.

An image of an application graph can be regarded as a polygon. We say an embedding in G'_c has a *hole of size* k, if there exist k consecutive empty classes in a line along one dimension which are inside the polygon and surrounded by non-empty classes. Thus, the example in figure 9 is excluded from our definition of *hole.*

We claim that after any embedding of a two-dimensional *orthogonal lattice* in a two-dimensional dynamic graph, it is impossible that there is a hole of size 2. Assume our claim is false, and denote the empty classes in a hole of size 2 by A and B. Index the nodes in the two-dimensional orthogonal lattice G_a by x_{ij}. For notational convenience, choose the origin so that x_{00} is a particular node which is mapped to the nonempty class immediately above A in G'_c. We will refer to the vertical line in G_a passing through x_{ij} as the vertical line Lx_i.

We have the following observations about the images in G'_c of vertical lines in the orthogonal lattice G_a. First, the images of the vertical lines Lx_i and Lx_{i+1} cannot be more than one class apart along one dimension. Because the image of each pair of nodes x_{ij} and $x_{i+1,j}$ is in the same class or adjacent classes, this follows by induction on j. Second, the vertical line Lx_0 and Lx_1 (resp. Lx_0 and Lx_{-1}) must pass on the same side of A and B, as in figure 10, since there is no edge passing between A and

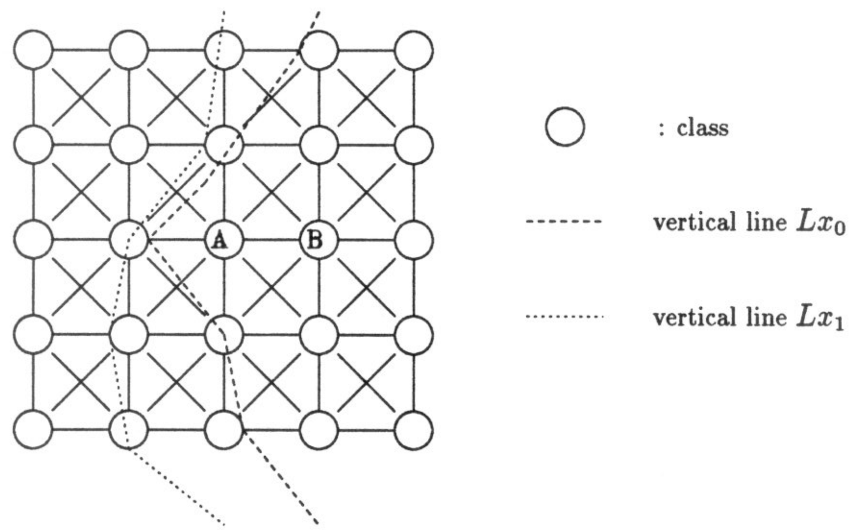


Figure 10: The image of vertical lines Lx_0 and Lx_1.

B. According to the above two observations, by induction on i, all the vertical lines Lx_i must be on the same side of A and B (either left or right), so A and B cannot be in the interior of the image of G_a. This contradiction proves that it is impossible to have a *hole of size two*. As we did in theorem 4.3, the adversary can choose the two *inner central classes* in one dimension to be faulty, and as before, there is no way to reconfigure G_r so that those two faulty classes are surrounded by non-empty classes. Thus, we must change $\Omega(n)$ nodes in one dimension to get a new working subgraph. □

Finally, we can extend this result to d dimensions. The line containing classes A and B will be replaced by a $(d-1)$-dimensional hyperplane in a d-dimensional dynamic graph.

Theorem 4.5 If $\mathcal{G}_a$ and $\mathcal{G}_r$ are families of d-dimensional dynamic graphs, there exists a constant β such that no *embedding architecture* can be *LR-reliable* with reliability β.

Proof: Given an application graph G_a which is a dynamic graph, a *reduced graph* can be built as before. Since the application graph is connected and a class is connected only to its neighboring classes, there exists at least one edge along each dimension from one class to its neighboring class. Therefore, any d-dimensional *reduced graph* contains a subgraph which is isomorphic to a d-dimensional *orthogonal lattice.* We therefore need only prove the theorem for the case of the application graph being a family of d-dimensional *orthogonal lattices.* Again, the proof of theorem 4.3 shows that d-dimensional *orthogonal lattices* cannot be embedded in $(d-1)$-dimensional dynamic graphs.

We claim that it is impossible that there exist a hole of size 2^{d-1} in one hyperplane H along $(d-1)$ dimensions (one coordinate is fixed) in the reduced graph. Assume our claim is false. Call the above 2^{d-1} classes an *obstacle* O. The *obstacle* is composed of two empty classes along each of the $(d-1)$ dimensions in H. Call the fixed dimension of H "vertical." By the same reasoning as in lemma 4.4, no vertical lines can pass through the *obstacle* O, and the images of any two adjacent vertical lines must lie on the same side of the *obstacle* O in the reduced graph. Therefore, the obstacle cannot be in the interior of the reduced graph, so our claim is correct. The adversary then chooses the *inner central* 2^{d-1} classes in H to be faulty. There is no way to reconfigure the redundant graph such that those faulty classes are surrounded by non-empty classes. Thus, we must change $\Omega(n)$ nodes in one dimension to get a new isomorphic subgraph. □

5 Possibility of an LR-reliable Embedding of Dynamic Graphs from Dimension d to d+1

Finally, we want to show that we really can embed d-dimensional dynamic graphs in $(d+1)$-dimensional dynamic graphs, while maintaining any desired high reliability and local reconfigurability. We begin with the one-dimensional case.

Lemma 5.1 When $\mathcal{G}_a$ is a family of linear arrays, there exists an *Embedding Architecture* where $\mathcal{G}_r$ is a family of two-dimensional dynamic graphs, which can be *LR-reliable* with any given β.

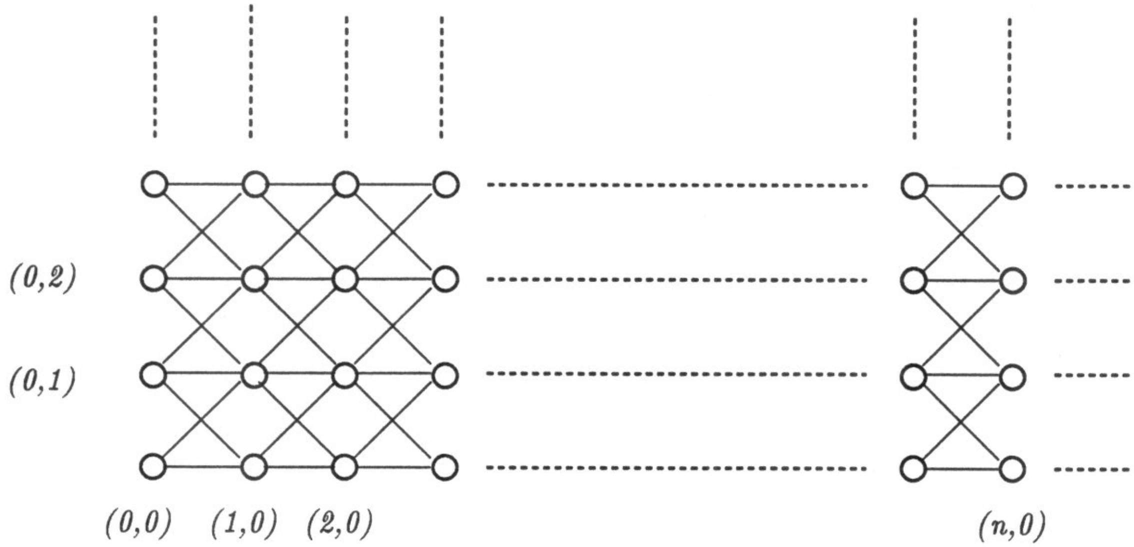


Figure 11: An LR-reliable 2-dimensional dynamic graph.

Proof: We prove this by constructing a redundant graph G_r^n for an n-node linear array G_a^n as shown in figure 11. G_r^n has n columns and each column has s nodes. Let $FT(G_a^n) < s$.

The initial embedding allocates each node of G_a^n to a distinct column of G_r^n, i.e. let the initial isomorphic subgraph be the sequence $(0,0)$, $(1,0)$, ..., $(n,0)$. If one node $(i,0)$ has failed , we choose $(i,1)$ as the replacing node, and if nodes $(i,0)$ and $(i,1)$ have failed, we use $(i-1,1)$, $(i,2)$ and $(i+1,1)$ to replace nodes $(i-1,0)$, $(i,0)$ and $(i+1,0)$. By using the above reconfiguration procedure, we change at most $2k-1$ nodes after any $k < s$ nodes have failed. Since $DR(k,n) = O(k)$, G_a^n with respect to such an EA and IE is locally reconfigurable.

We now want to show that given β, we can find an s and G_r^n with the desired properties. Let $\hat{G}_r^n$ be a square piece of G_r^n, an $n \times n$ dynamic graph. Let $p(n)$ be the probability that $\hat{G}_r^n$ contains G_a^n. We form a vertical pile of s/n such blocks to obtain $s \times n$ such dynamic graphs as in figure 12. After we connect each two adjacent squares, the resulting graph is the same as G_r^n.

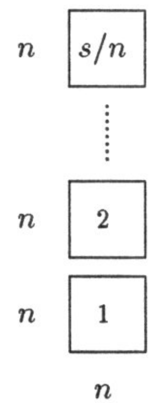


Figure 12: A pile of $\hat{G}_r^n$ for the proof of lemma 5.1.

Since connections between two squares can only increase the reliability, the probability that there does not exist a working linear array in this big graph is $< (1-p(n))^{s/n}$. For any c, if $s > \frac{cn \cdot logn}{-log(1-p(n))}$, the above probability will be $< 1/n^c$. Therefore, for any reliability β, we can find a sufficient large s to achieve reliability β. □

We can now prove the main result in this section.

Theorem 5.2 When $\mathcal{G}_a$ is a family of d-dimensional dynamic graphs, there exists an *embedding architecture* where $\mathcal{G}_r$ is a family of $(d+1)$-dimensional dynamic graphs, which can be *LR-reliable* with any given β.

Proof: As before, we construct a reduced graph from the given dynamic application graph G_a. The most general form of a reduced graph is a web. Thus, without loss of generality, we need only prove the theorem for the case of the application graph being a family of d-dimensional webs. We can use the same construction and reconfiguration method as we did in the previous lemma. □

From the above reconfiguration method, after $k \leq FT(G_a^n)$ nodes have failed, we need to change at most $2 \cdot k$ nodes. The following corollary shows that when $d = 1$, we can reduce this to exactly k nodes.

Corollary 5.3 When $\mathcal{G}_a$ is a family of linear arrays, there exists an *embedding architecture* where $\mathcal{G}_r$ is a family of two-dimensional dynamic graphs with edge degree

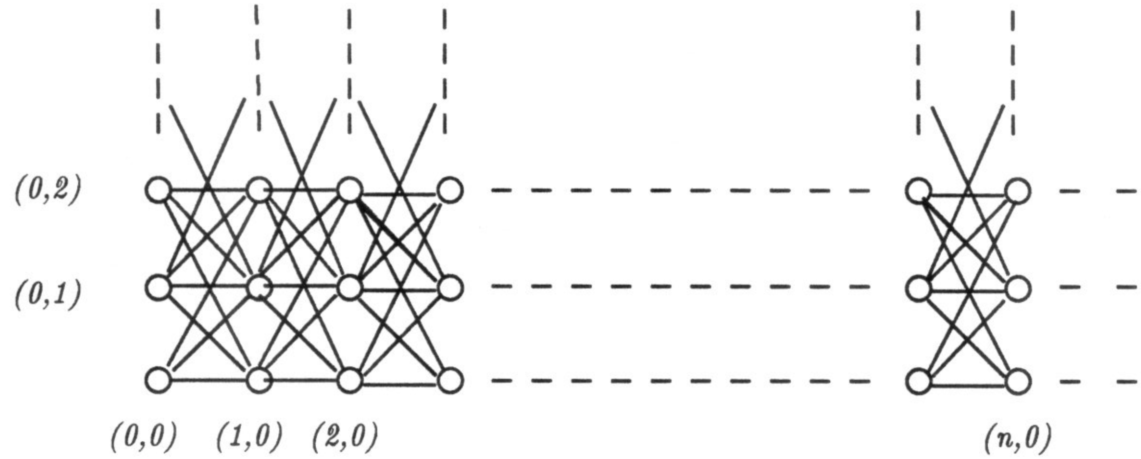


Figure 13: Dynamic graph construction for corollary 5.3.

$4m+2$, where m is any constant ≥ 2, such that after any $k \leq FT(G^n_a)$ nodes have failed, we only need to change k nodes.

Proof: First construct the dynamic graph as shown in figure 13, where there are s nodes in each column: each node (i,j) connects to $(i+1, j+m), (i+1, j+m-1), \ldots, (i+1, j), \ldots, (i+1, j-m+1), (i+1, j-m)$.

The reconfiguration method is the same as in lemma 5.1. Let $FT(G^n_a) < s$ for each G^n_a in the family, and allocate nodes of G^n_a to different columns as before. The number of nodes which need to be changed after k nodes in one column have failed is at most $\lceil \frac{k}{m} \rceil \times 2 - 1$. This is the worst case, so $DR(k,n) = \max(\lceil \frac{k}{m} \rceil \times 2 - 1, \; k) = k$, if $m \geq 2$. □

Similar constructions work for d dimensions.

References

[ChLeRo] Fan R.K. Chung, F.T. Leighton, and A.L. Rosenberg, "Diogenes: A methodology for designing fault-tolerant VLSI processing arrays," *Proc. IEEE FTCS,* Milano, 1983, pp. 26-32.

[CaSt] P. R. Cappello and K. Steiglitz, "Digital signal processing applications of systolic algorithms," *CMU Conf. on VLSI Systems and Computations*, H.T. Kung, B. Sproull, and G. Steele (eds.), Computer Science Press, Rockville, MD, Oct. 1981, pp. 19-21.

[GrGa] J.W.Greene and A.E. Gamal, "Configuration of VLSI arrays in the presence of defects," *J. Asso. Comp. Mach.,* vol. 31, Oct. 1984, pp. 694-717.

[Ha] J.P. Hayes, " A graph model for fault-tolerant computing systems, " *IEEE Transactions on Computers*, vol. C-25, no. 9, September 1976, pp. 875-884.

[IwSt1] K. Iwano and K. Steiglitz, "Testing for cycles in infinite graphs with periodic structure," *Proc. 19th Annual ACM Symposium on Theory of Computing*, New York, NY, May 1987, pp. 46-55.

[IwSt2] K. Iwano and K. Steiglitz, "Planarity testing of doubly periodic infinite graphs," *Networks*, vol. 18, no. 3, Fall 1988, pp. 205-222.

[IwSt3] K. Iwano and K.Steiglitz, "A semiring on convex polygons and zero-sum cycle problems," *SIAM J. Computing*, to appear.

[HKu] H.T. Kung, " Why systolic architectures ?," *Computer Magizine*, vol. 15 , no. 1, January 1982, pp. 37-46.

[KuLa] H.T. Kung and M.S.Lam, "Fault tolerant VLSI systolic arrays and two-level pipelines," *J. Parall. and Distr. Proc.,* vol. 8, 1984, pp. 32-63.

[SKu] S.Y. Kung, *VLSI Array Processors,* Prentice Hall, Englewood Cliffs, NJ, 1988.

[KAGB] S.Y. Kung, K.S. Arun, R.J. Gal-Ezer, and D.V. Bhaskar Rao, "Wavefront array processor: Languages, architecture, and applications," *IEEE Transactions on Computers,* vol. C-31, Nov. 1982, pp. 1054-1066.

[KuJeCh] S.Y. Kung, S.N. Jean and C.W. Chang, "Fault-tolerant array processors using single track switches," *IEEE Transactions on Computers,* vol. C-38, no. 4, April 1989, pp. 501-514.

[KuSt] S.D. Kugelmass and K. Steiglitz, "A scalable architecture for lattice-gas simulation, " *J. Computational Physics*, vol. 84, Oct. 1989, pp. 311-325.

[LeLe] T. Leighton and C. E. Leiserson, "Wafer-scale integratrion of systolic arrays," *IEEE Transactions on Computers,* vol. C-34, no. 5, 1989, pp. 448-461.

[Or] Orlin, J., "Some problems on dynamic/periodic graphs," *Progress in Combinatorial Optimization*, W. R. Pulleyblank (ed.), Academic Press, Orlando, Florida, 1984, pp. 273-293.

[SaSt] M. Sami and R. Stefenelli, "Reconfiguration architecture for VLSI processing arrays," *Proc. IEEE FTCS,* 1986, pp. 712-722.