Extending a Distributed Loop Network to Tolerate Node Failures

Abdel Aziz Farrag
Faculty of Computer Science
Dalhousie University
Halifax, NS, Canada
farrag@cs.dal.ca

ABSTRACT

We examine the problem of extending a distributed loop network by adding spare nodes and links so as to make it more fault-tolerant. The optimization criterion used in finding a fault-tolerant solution is to reduce the node-degree of the overall network. This is important in practice due to the limitation on the number of links allowed per node in VLSI design. Our results indicate that the solutions obtained (numerically or analytically) are efficient, i.e., either optimal or nearly-optimal.

Categories and Subject Descriptors

C.4 [**Performance of Systems**]: Fault Tolerance.
C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design - *network topology*.

General Terms

Reliability

Keywords

Fault-tolerance, networks, digraphs, distributed loops.

1. INTRODUCTION

Loop-based topology has been used in the design of many distributed networks, and also in some parallel computer systems [3,9,15,16]. This class of networks exhibits some useful properties such as simplicity, expandability and regularity. Moreover, it includes (as special cases) several important topologies such as rings and circulant graphs, for examples.

Due to the highly regular structure of distributed loops, they tend to be vulnerable to failures, that is, the failure of even a single node or link may change or break the network. Therefore, to make the network fault-tolerant, some spare nodes and links can be added so that when a failure occurs, the network can be reconfigured to bypass the defective components. This is the main approach used to achieve fault-tolerance [1].

Extending a given network to achieve fault-tolerance has been examined for a variety of architectures (e.g., rings [8,9,10], meshes [3,5,6,9,11,21], stars [4,7,13,20] and hypercubes [3,12,19,21,22]). The optimization criterion used in building a solution is to reduce the node-degree of the overall network. This is an important objective in practice due to the limitation on the number of links allowed per node in VLSI design [2]. This is also the same criterion we use in this paper.

We examine the problem of extending a distributed loop so as to make it more fault-tolerant. Our method can be used to tolerate any desired number of node failures, and moreover, it can be applied when the parameters that define the loop are given numerically or symbolically. For the numeric case, an algorithm is proposed which finds a fault-tolerant design by first generating a family of solutions, and then selecting the one with the least node-degree. For the symbolic case, we use our method to develop analytic solutions to some well-known distributed loops [16,18].

The performance and the reliability of distributed loops have been extensively studied [2,13,16-18]. However, the problem of building a fault-tolerant extension of a distributed loop received a little attention. This papers generalizes the recent work reported in [10]. We use the same formulation, but we allow the loop to be given either numerically or symbolically.

The rest of this paper is organized as follows. We present the background material in Section 2. Then, we develop a formalism to partition the jumps of a distributed loop in Section 3. This formalism will be used in Section 4 to develop a new algorithm for finding a fault-tolerant solution of a distributed loop. This algorithm assumes that the parameters of the loop are given numerically (as constants). Finding a solution when the loop parameters are given symbolically (as variables) are discussed in Section 5. Finally, we present conclusions in Section 6.

PADTAD'11, July 17, 2011, Toronto, ON, Canada

2. BACKGROUND

The graphs or digraphs used throughout the paper denote multicomputer networks, where the nodes correspond to processors and the edges (or arcs) represent links. We deal mainly with directed graphs (also called digraphs, for short). Notice that an undirected graph is a special case of a digraph, that is, the former can be converted into the latter by replacing every edge (u,v) by two (directed) arcs, one from u to v and the other from v to u.

Definition 2.1: (Digraph isomorphism)
Two digraphs G_1 and G_2 are isomorphic iff there is a 1-to-1 correspondence between their nodes that preserves adjacency, i.e., there is a bijection (function) f from the nodes of G_1 to the nodes of G_2 such that (u,v) is an arc in G_1 iff (f(u),f(v)) is an arc in G_2.

Definition 2.2: (k-ft solution)
A digraph H is a k-fault-tolerant (or k-ft) of a digraph G, if for every set S of k nodes in H, removing the nodes of S from H will leave a remainder containing a subgraph isomorphic to G.

For example, the digraph G in Figure 1 has a 1-ft solution H shown in Figure 2, where H has one more node than G, and the thicker edges inside H identify a subgraph isomorphic to G which "excludes" one faulty node.

In designing a k-ft solution of a digraph G, we use the minimum number of spare nodes, i.e., the solution will have exactly k more nodes than G.

Definition 2.3: (Distributed loop)
A distributed loop (or a loop, for short) is a digraph defined by a set of nodes numbered {0, 1,…, n-1} and a set of integers called jumps denoted A= $\{a_1, a_2,.., a_i\}$ such that for every node x and every jump a_j, x is adjacent to node x + a_j mod n.

For example, the digraph in Figure 2 is a 6-node loop with two jumps equal to 1 and 2; respectively.

In what follows, we shall denote an n-node loop G with jumps $\{a_1, a_2,.., a_i\}$ as $G(a_1, a_2,.., a_i: n)$. For example, the distributed loop H in Figure 2 is H(1,2: 6).

It is convenient to draw the loop by arranging its nodes around a circle in a clockwise direction (as shown in Figure 1). We use only positive jumps throughout, i.e., if a negative jump a_i arises, we shall convert it into the equivalent positive jump a_i + n (mod n). Thus, to find if a given node is adjacent to another, we can find the distance between them along the circle, as defined below, and check if it is equal to one of the jumps.

Definition 2.4: (Circulant distance)
The circulant distance from node x to node y is defined as either y-x or y- x + n depending on whether or not y is greater than x; respectively.

For examples, in Figure 1 the circulant distance from node 1 to node 3 is 2, and the circulant distance from node 1 to node 0 is 4.

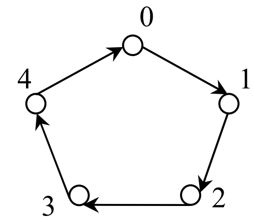

Figure 1. A digraph G.

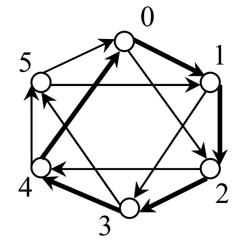

Figure 2. A 1-ft H of G.

The basic method to construct a k-ft of a distributed loop is given in the theorem below (which generalizes that of [8]).

Theorem 2.1: Given a digraph $G(a_1, a_2,.., a_i: n)$, we can construct a distributed loop H with n+k nodes that is a k-ft of G where the jumps of H consist of $\{a_1, a_1+1, a_1+2,.., a_1+k\} \cup \{a_2, a_2+1, a_2+2,.., a_2+k\} \cup \ldots \cup \{a_i, a_i+1, a_i+2,.., a_i+k\}$.

Proof: Assume here that the number of faulty nodes is k; (otherwise, if there are fewer than k faulty nodes, we can make the difference by selecting some extra healthy nodes and treat them as faulty). The remaining n healthy nodes of H will be renumbered in the same order along the circle starting at any healthy node as 0 and skipping every faulty node. That is, after this renumbering, the healthy nodes will be labeled 0, 1, .., n-1.

For every healthy node newly numbered as x, let g(x) denote the node that corresponds to x before this renumbering, i.e., g is a function that maps x to g(x). To complete the proof, we need only to show that for every such node x and every jump a_j, $1 \le j \le i$, x must be adjacent to the (healthy) node x + a_j mod n.

First, assume that none of the nodes along the circle over the interval from node x up-to node y= x+ a_j (mod n) is faulty. Then, the circulant distance from x to y is the same as that from g(x) to g(y), i.e., equal to a_j. Therefore, by the definition of the distributed loop H given in this theorem, node g(x) is adjacent to node g(y), which means that node x must be adjacent to y.

Otherwise, let M denote the number of faulty nodes (skipped) along the circle over the interval from the node x to the node y = x+ a_j (mod n), where x and y are healthy nodes. In this case, the circulant distance from node g(x) to g(y) is equal to M+ a_j, where $M \le k$. Thus, by the definition of the distributed loop H, node g(x) is adjacent to node g(y), which in turn means node x is adjacent to node y= x+ a_j (mod n). □

Example 2.1: By the above theorem, the loop G(1,9: 11) has a 1-ft of the form H(1,2,9,10: 12), and G(1,2: 7) has a 1-ft of the form H(1,2,3: 8).

Although the above theorem finds only one k-ft for a given distributed loop, a large family of k-ft solutions can be generated as will be shown later. To compare the costs of various solutions, we shall count only the number of jumps used in every solution as it relates directly to node-degree. Notice that the degree of a digraph H is the maximum degree of any node in H. (Notice also that the degree of a node x is the sum of the in-degree and the out-degree of x.)

The proof of the above theorem implies a simple reconfiguration for the loop, which amounts to renumbering of the nodes as explained before, and placing this information at the healthy nodes.

3. PARTITIONING THE JUMPS OF A DISTRIBUTED LOOP

A subset of the jumps of a distributed loop G is called a block if they are physically consecutive, i.e., of the form {j, j+1, j+2, ..}. By the (k-ft) theorem proven in the preceding section, the fewer the number of blocks in G, the more efficient the cost of its k-ft. Thus, the best-case for a k-ft constructed by this theorem occurs when the jumps of G are all consecutive (i.e., form only one block).

In this section, we develop a new formalism called an m-distance subset which generalizes the notion of a block of (consecutive) jumps. The m-distance subsets will be formed by partitioning the jumps of a given loop as explained later. In what follows, the greatest common divisor of two integers x and y is denoted gcd(x,y). If gcd(x,y)= 1, x and y will be called coprime or relatively prime.

Let x and n be relatively prime. Then, the multiplicative inverse (or the inverse, for short) of x mod n is a new element denoted x^{-1} which satisfies the condition: $x * x^{-1} \pmod{n} = 1$. For example, 2^{-1} mod 5 = 3.

Definition 3.1: (Partitioning sequences)
Let n and m be any pair of integers where gcd(n, m)= 1 and n > m > 0. We define an ordered sequence based on n and m, denoted $S(n,m)= \langle s_1, s_2, .., s_{n-1}\rangle$, where $s_i = i * m \pmod{n}$, for all $1 \leq i \leq n-1$.

For instance, when n=7 and m=1, S(7,1)= <1,2,3,4,5,6>. Similarly, for n=7 and m=3, we get S(7,3)= <3,6,2,5,1,4>.

It is not difficult to show that S(n,m) contains all integers from 1 up-to n-1, that is, it includes the whole range of valid jumps (of an n-node distributed loop). After generating S(n,m), we shall use it to partition the jumps of the distributed loop as will be explained below.

Definition 3.2: (m-distance subset)
A subset J of the jumps of an n-node distributed loop G is called an m-distance subset, where m is any integer such that gcd(n,m) =1 and n > m > 0, if there is a subsequence of consecutive elements in S(n,m) that contains every jump in J and has the same number of elements as in J. Further, an m-distance subset is called maximal if it is not contained in any other m-distance subset.

Example 3.1: The two jumps of the distributed loop G(1,3: 9) form a 2-distance subset. This is because S(9,2)= <2,4,6,8,1,3,5,7>, i.e., S(9,2) has a subsequence <1,3> that contains these jumps.

Definition 3.3: (m-distance partition)
Let P(A,n,m) denote a collection of m-distance subsets defined over a set A of jumps of an n-node distributed loop. Then, P(A,n,m) is called an m-distance partition of A, if every m-distance subset in P(A,n,m) is maximal and every jump in A appears (inside one subset) in P(A,n,m).

Example 3.2: Consider the distributed loop G(1,2,5:7). Since S(7,1)= <1,2,3,4,5,6>, a 1-distance partition of the jumps of G is equal to {{1,2},{5}}. Similarly, since S(7,3)= <3,6,2,5,1,4>, a 3-distance partition of the jumps of G has only one subset {{1,2,5}}.

We present below the algorithm for partitioning the jumps of a distributed loop G in which A denotes the set of jumps in G, n denotes the number of nodes of G, and m is any integer that is coprime to n, where n > m > 0.

Algorithm Partition(A,n,m) {

1. Construct the sequence $S(n,m)= \langle s_1, s_2, .., s_{n-1}\rangle$ as defined before;
2. For every element s_i in S(n,m), if s_i appears as a jump in A, then keep s_i in S(n,m); otherwise, replace s_i in S(n,m) by a special separation symbol, say "&";
3. For every (maximal) subsequence in S(n,m) that does not include the separation symbol "&", form an m-distance subset which has the same elements as those in the subsequence;
4. Return a partition P(A,n,m) consisting of all m-distance subsets formed above;

}

Since S(n,m) contains n-1 elements, and each of them can be checked in Step 2 in only O(log |A|) time (if A were sorted first), therefore, the time-complexity of the above algorithm is O(n log |A|).

It is not difficult to show that the sequences S(n,m) and S(n,n-m) are the reverse of each other, and therefore, if we partition the jumps of an n-node digraph using S(n,m) or S(n,n-m), we obtain the same results. Accordingly, we shall use only sequences S(n,m) that satisfy the condition m < n/2.

The table below shows all possible ways to partition the jumps of the distributed loop G(1,4,7,9: 11); each of them is based on a different value of m.

Table 1. Every row below gives a value of m that is coprime to n (where n=11), the sequence S(11,m) and the corresponding partition of set of jumps {1,4,7,9}.

m	S(11,m)	Partition of {1,4,7,9}
1	<1,2,3,4,5,6,7,8,9,10>	{{1}, {4}, {7}, {9}}
2	<2,4,6,8,10,1,3,5,7,9>	{{1}, {4}, {7,9}}
3	<3,6,9,1,4,7,10,2,5,8>	{{1,4,7,9}}
4	<4,8,1,5,9,2,6,10,3,7>	{{1}, {4}, {7}, {9}}
5	<5,10,4,9,3,8,2,7,1,6>	{{1,7}, {4,9}}

4. THE ALGORITHM

Given a distributed loop G, we would like to construct a k-ft for it. This can be done by applying Theorem 2.1 as explained before, but the solution obtained this way may be unnecessarily expensive. Instead, we build over the formulation of the preceding section to develop a new algorithm which generates a large family of k-ft solutions that can be compared to select the one with the least node degree. This algorithm assumes that the jumps of G are specified numerically (as constants). Otherwise, if the jumps of G are given symbolically (as variables), we can use the method explained in the next section.

Theorem 4.1: If n and p are coprime, then the digraph $G(a_1, a_2,.., a_i: n)$ must be isomorphic to $H(a_1 * p \bmod n, a_2 * p \bmod n,.., a_i * p \bmod n: n)$.

Proof: We define a mapping "f" which transforms each node x in G into a node f(x) in H such that f(x)= x ∗ p (mod n). To prove that "f" is 1-to-1, it suffices to show that "f" cannot map distinct nodes to the same value. To see why, let f(x) = f(y), then x ∗ p (mod n) = y ∗ p (mod n), and therefore, by multiplying each side by p^{-1} we get $x * p * p^{-1}$ (mod n) = $y * p * p^{-1}$ (mod n). But, since $p * p^{-1} = 1$ (mod n), this will imply x= y.

The function "f" also preserves adjacency. This is because for any edge (x,y) in G, we must have some jump a_j such that $y = x+a_j$ (mod n). This, in turn, implies $f(y) = f(x + a_j) = x * p + a_j * p$ (mod n), that is, $f(y) = f(x)+ a_j * p \bmod n$. This implies (f(x),f(y)) is an edge in H.

The two properties of "f" proven above establishes the isomorphism between G and H. □

We have shown (in the preceding section) how to partition the jumps of a loop G into m-distance subsets (where m is coprime to the number of nodes in G). Following this, we can use the following lemma to convert every such subset into a block of consecutive integers. This is needed before we can construct a new k-ft solution for G.

Lemma 4.1: Multiplying the jumps of the distributed loop $G(a_1, a_2,.., a_i:n)$ by m^{-1} transforms each m-distance subset of its jumps into a block of consecutive integers of the form {j, j+1, j+2, ..}.

Proof: By the definition of an m-distance subset given earlier, the elements of any such subset can be written as {b, b+m, b+2m, ..} for some integer b, where additions are mod n. Thus, if we multiply the jumps by m^{-1}, we obtain $\{b * m^{-1}, b * m^{-1} + m * m^{-1}, b * m^{-1} + 2m * m^{-1}, ..\}$ where the additions and the multiplications are done mod n. Since $m * m^{-1} = 1$ (mod n), this subset can be simplified to $\{b * m^{-1}, b * m^{-1} + 1, b * m^{-1} + 2, ...\}$. Moreover, if we substitute $j = b * m^{-1}$, we get {j, j+1, j+2, ..}. □

The above results together with the formalism given earlier can be combined to develop an algorithm which constructs a k-ft solution for G by partitioning its jumps in many ways; each is based on a different selection of an integer (m) that is coprime to the number of nodes (n) and each leads to a k-ft solution for G. These solutions can be compared to select the one with the least node degree. The complete details of this algorithm are given below.

Algorithm Fault-tolerance(G,k) {

1. Generate all integers $(m_1, m_2,.., m_j\}$ such that for each m_j we have $\gcd(n, m_j) = 1$ and $1 \le m_j < (n/2)$;
2. For every m_j generated above, find an m_j-partition of the jumps of the distributed loop G as shown before;
3. For every m_j-partition generated above, convert its subsets into blocks by multiplying its jumps by m_j^{-1} (mod n); and let T_j denote the distributed loop whose jumps consists of the union of these blocks;
4. Use Theorem 2.1 to construct a k-ft solution of each digraph T_j defined in Step(3);
5. Compare each k-ft solution once constructed in Step(4) to select the one with the least node-degree;

}

The table below shows an example of applying the above algorithm to find a 1-ft of G(1,4,7,9: 11).

Table 2. This table records the various 1-ft solutions generated by above algorithm for the loop G(1,4,7,9:11) and identifies the final solution selected.

m	1-ft solution of G(1,4,7,9: 11)
1	H(1,2,4,5,7,8,9,10: 12)
2	H(2,3,6,7,9,10,11: 12)
3	**H(3,4,5,6,7: 12) ← selected solution**
4	H(1,2,3,4,5,6,10,11: 12)
5	H(3,4,5,8,9,10: 12)

Lemma 4.2: Algorithm Fault-tolerance(G,k) runs in $O(n^2 \log|A| + n\,k\,|A|)$ time.

Proof: We trace the steps of the algorithm for each m_j generated. Checking if m_j and n are relatively prime in Step(1) can be done in $O(\log^3 n)$ time as shown in reference [14], and finding a partition corresponding to m_j in Step(2) requires O(n log|A|) time as was shown earlier. Converting the jumps into blocks in Step(3) needs O(|A|) time, and constructing a k-ft in Step(4) requires at most O(k|A|) time. Thus, the time for each m_j is O(n log|A| + k|A|), and since the number of integers (m_j's) to be generated is bounded by O(n), therefore, the whole algorithm can be done in O(n^2 log|A| + n k|A|) time. □

The table below shows more examples of k-ft solutions generated by the above algorithm for some distributed loops.

Table 3. Every row in the table below records a distributed loop G, its 1-ft and 2-ft solutions.

Graph	1-ft of G	2-ft of G
G(1:5)	H(1,2:6)	H(1,2,3:7)
G(1,6:7)	H(3,4,5:8)	H(3,4,5,6:9)
G(1,3:7)	H(4,5,6:8)	H(4,5,6,7:9)
G(1,3:9)	H(5,6,7: 10)	H(5,6,7,8:11)
G(1,4,7,9:11)	H(3,4,5,6,7:12)	H(3,4,5,6,7,8:13)

5. FAULT-TOLERANCE OF SOME DISTRIBUTED LOOPS

The algorithm developed in the preceding section to design a k–ft solution for a distributed loop G assumes that the parameters of G (i.e., its size and jumps) are given numerically as constants. Otherwise, if (all or some of) these parameters are given symbolically as variables, we can still apply our formulation to develop a k-ft "analytically". We illustrate the method by developing solutions for the following three loops:

- G(1, m: m^2),
- G(1, n–2: n), and
- G(1, n –1: n).

The above loops are examples of the so called double-loop networks analyzed before in [16]. The theorem below finds a solution for G(1, m: m^2).

Theorem 5.1: For any k ≥ 1, G(1, m: m^2) has a k-ft solution of the form H(m^2 –m –1, m^2 –m,.., m^2 –m –1+k : m^2+k).

Proof: Since, for any m > 1, m^2 and m^2–1 are coprime and m^2–1=(m –1)(m +1), therefore, m^2 and m –1 must also be copime. Thus, we can group the two jumps of G into an (m –1)-distance subset, and then convert it into a block (of consecutive jumps) by forming a digraph Q of the form Q(1 ∗ $(m-1)^{-1}$, m ∗ $(m-1)^{-1}$: m^2). It is not hard to see that $(m-1)^{-1}$= m^2–m–1. This is because (m –1) ∗ (m^2 –m –1)= m^3 –2m^2+1 = 0 –0 +1 (mod m^2)= 1. Thus, Q has the form Q(1 ∗ (m^2– m –1), m ∗ (m^2–m –1): m^2) = Q(m^2 –m –1, m^3 –m^2 –m): m^2).

Since m^3 –m^2= 0 (mod m^2), Q has the form Q(m^2 –m –1, –m : m^2), i.e., after representing each jump in a positive form, we obtain Q(m^2 –m –1, m^2 –m : m^2). Consequently, a k-ft of Q (which is also a k-ft of G) is equal to H(m^2 –m –1, m^2 –m,.., m^2 –m –1+k : m^2+k). □

To solve the loop G(1, n–2: n), (which is known in the literature as a daisy-chain network [13]), we shall need the results developed in the following 3 lemmas. (In the rest of the paper, all operations will be done mod n.)

Lemma 5.1: Let n be divisible by 3. Then, either n/3 - 1, or n/3 +1 or both must be coprime to n.

Proof: Since every pair of consecutive integers is coprime, then each of the two pairs (n/3 -1, n/3) and (n/3, n/3 +1) must be coprime. This, in turn, implies that "3" is the only possible common factor between the pair (n/3 -1, n), and between the pair (n, n/3 +1). However, since exactly one out of any 3 consecutive integers (e.g., n/3 -1, n/3, n/3 +1) is divisible by 3, therefore, at least one of the two pairs (n/3 -1, n) or (n, n/3 +1) must be coprime. □

Lemma 5.2: Let n be divisible by 3, and let n/3 -1 be coprime to n. Then, the inverse of n/3 -1 (mod n) is equal to either (2n/3 -1) or (n/3 -1) depending on whether or not n is divisible by 9; respectively.

Proof: Since 3 divides n and n/3 -1 is coprime to n, then n/3 -1 is not divisible by 3. Consequently, either n/3 or n/3 +1 is divisible by 3.

Suppose first n/3 is divisible by 3, i.e., n is divisible by 9. Then, (n/3 -1)(2n/3 -1) = 2n(n/9)-n+1. Since n/9 is an integer in this case, then 2n(n/9) -n +1 (mod n) = 0 -0 +1 (mod n) =1. Thus, $(n/3-1)^{-1}$ = 2n/3 -1.

Otherwise, let n/3 +1 be divisible by 3, then (n/3 -1)(n/3 -1) = (n/3)(n/3 +1) -n+1= n (n/3 +1)/3 -n+1 = 0-0+1 (mod n) =1. That is, $(n/3-1)^{-1}$ = n/3 -1 (mod n). □

Lemma 5.3: Let n be divisible by 3 and let n/3 +1 be a coprime to n. Then, the inverse of n/3 +1 (mod n) is equal to either (2n/3 +1) or (n/3 +1) depending on whether or not n is divisible by 9; respectively.

Proof: The proof methodology for this lemma is similar to that of Lemma 5.2, and therefore omitted. □

Theorem 5.2: For any k ≥ 1, we can form a k-ft H for the distributed loop G(1, n-2: n) as follows.

a) If 3 divides (n+1), the solution H will be equal to H((n-2)/3, (n+1)/3, (n+1)/3 +1,.., (n+1)/3 +k : n+k).

b) If 3 divides (n-1), the solution H will be equal to H((2n-2)/3,(2n+1)/3,(2n+1)/3 +1,..,(2n+1)/3 +k: n+k).

c) If 3 divides n and gcd(n, (n/3 +1)(n/3 -1))=1, H will be equal to H((2n/3)-1, 2n/3,.., (2n/3)+k-1, (2n/3)+2, (2n/3)+3,.., (2n/3)+2+k : n+k).

d) If 3 divides n and gcd(n, n/3 +1) ≠ 1, H will be H((n/3) -1, (n/3),.., (n/3) -1+k, (n/3) +2, (n/3) +3,.., (n/3) +2+k : n+k).

e) If 3 divides n and gcd(n, n/3 -1) ≠ 1, H will be H((n/3)-2, (n/3)-1,.., (n/3)-2+k, (n/3)+1, (n/3)+2,.., (n/3)+1+k : n+k).

(Thus, in all cases, H has either k+2 or k+4 jumps.)

Proof: Suppose first that Case(a) is true, i.e., let (n+1) be divisible by 3. Then, 3^{-1} (mod n) will be equal to (n+1)/3, i.e., we can group the two jumps of G into a 3-distance partition, and then form a (block) digraph Q for G of the form Q(1 ∗ 3^{-1}, (n-2) ∗ 3^{-1}: n)= Q(1 ∗ (n+1)/3, (n-2) ∗ (n+1)/3: n)= Q((n-2)/3, (n+1)/3: n), where the multiplication ∗ is done mod n. Thus, a k-ft H of Q, (which is also a k-ft of G), is equal to H((n-2)/3, (n+1)/3, (n+1)/3 +1,.., (n+1)/3 +k : n+k).

Similarly, suppose that Case(b) is true, i.e., let (n-1) be divisible by 3, then it is not difficult to verify that 3^{-1} =(2n+1)/3 in this case. This is because 3 ∗ (2n+1)/3 = 3 ∗ (2n-2+3)/3 = 3 ∗ 2(n-1)/3 +3 =2n−2+3 =1 mod n. Thus, we can group the jumps of G into a 3-distance partition, and then form a (block) digraph Q for G of the form Q(1 ∗ 3^{-1}, (n-2) ∗ 3^{-1}: n) = Q(1 ∗ (2n+1)/3, (n-2) ∗ (2n+1)/3: n) = Q((2n-2)/3, (2n+1)/3: n), (where the multiplication is done mod n). Accordingly, a k-ft solution H of G is equal to H((2n-2)/3, (2n+1)/3, (2n+1)/3 +1,.., (2n+1)/3 +k : n+k).

Suppose Case(c) is true, i.e., let 3 divides n and gcd(n, (n/3 +1)(n/3 -1)) =1. Then, we can group the jumps of G into either an (n/3 +1)-distance partition or (n/3 -1)-distance partition. Both lead to solutions of equal cost, and therefore, we chose the latter, i.e., we can form a (block) digraph Q for G of the form Q(1 ∗ $(n/3 -1)^{-1}$, (n-2) ∗ $(n/3 -1)^{-1}$: n). Since neither (n/3 +1) or (n/3 -1) is divisible by 3 in this case, therefore, n must be divisible by 9, i.e., by Lemma 5.2, the inverse of (n/3 -1) will be equal to (2n/3 -1). Thus, the digraph Q will be equal to Q(1 ∗ (2n/3 -1), (n-2) ∗ (2n/3 -1): n) = Q((2n/3 -1), (2n/3 +2): n) where multiplication is done mod n. Therefore, a k-ft H of G is equal to H((2n/3)-1, 2n/3,.., (2n/3)+k-1, (2n/3)+2, (2n/3)+3,.., (2n/3)+2 +k : n+k).

Similarly, suppose that Case(d) is true, i.e., let 3 divides n and gcd(n, n/3 +1) ≠ 1. Then, by Lemma 5.1, n and (n/3)-1 must be coprime, i.e., we can group the jumps of G into an (n/3 -1)-distance partition, and then form a (block) digraph Q for G of the form Q(1 ∗ $(n/3 -1)^{-1}$, (n-2) ∗ $(n/3 -1)^{-1}$: n), where the inverse of (n/3 -1) in this case will be equal to (n/3 -1) as was shown in Lemma 5.2. Thus, the (block) digraph is equal to Q(1 ∗ (n/3 -1), (n-2) ∗ (n/3 -1) : n) = Q((n/3 -1), (n/3 +2) : n) (where multiplication is done mod n). Therefore, a k-ft H of G is equal to H((n/3)-1, (n/3),.., (n/3)-1+k, (n/3)+2, (n/3)+3,.., (n/3)+2+k : n+k).

Finally, suppose that Case(e) is true, i.e., let 3 divides n and gcd(n, n/3 -1) ≠ 1. Then, by Lemma 5.1, n and (n/3 +1) must be coprime, i.e., we can group the jumps of G into an (n/3 +1)-distance partition, and form a (block) graph for G of the form Q(1 ∗ $(n/3 +1)^{-1}$, (n-2) ∗ $(n/3 +1)^{-1}$: n), where the inverse of (n/3 +1) in this case is equal to (n/3 +1) as was shown in Lemma 5.2. Thus, the digraph Q is equal to Q(1 ∗ (n/3 +1), (n-2) ∗ (n/3 +1) : n) = Q((n/3 +1), (n/3 -2) : n) (mod n). Accordingly, a k-ft H of G is H((n/3)-2, (n/3)-1,.., (n/3)-2+k, (n/3)+1, (n/3)+2,.., (n/3)+1+k : n+k). □

The k-ft solution for the bidirectional ring G(1, n −1: n) depends on whether n is even or odd as shown below.

Lemma 5.4: Let n be odd, then for any k ≥ 1, the loop G(1,n − 1: n) has a k-ft solution of the form H((n −1)/2, (n −1)/2 +1 ,.., (n −1)/2 +1+k : n+k).

Proof: Since n is odd, then it is relatively prime to 2, that is, we can group the jumps of G into a 2−distance partition, and then convert each 2−distance subset of this partition into a block by forming a (block) digraph Q of the form: Q(1 ∗ 2^{-1}, (n −1) ∗ 2^{-1}: n), where 2^{-1} (mod n)= (n +1)/2 in this case. Thus, Q has the form Q(1 ∗ (n +1)/2, (n −1) ∗ (n +1)/2: n) = Q((n +1)/2, n ∗ (n −1)/2 + (n −1)/2: n), which is equivalent to Q((n +1)/2, (n −1)/2: n) after we apply the mod n operation (i.e., n ∗ (n −1)/2 = 0 mod n). Therefore, a k-ft solution of Q (which is also a k-ft of G) is H((n −1)/2, (n −1)/2 +1 ,.., (n −1)/2 +1+k : n+k). □

For example, by the above lemma, G(1,8: 9) has a 2-ft of the form H(4,5,6,7: 11). The solution for the ring G(1, n-1: n), when n is even, depends on whether n is divisible by 4 as shown below.

Lemma 5.5: Let n be divisible by 4, then for any k ≥ 1, G(1, n − 1: n) has a k-ft solution of the form H(n/2 −1, n/2,.., n/2 +1+k : n+k).

Proof: Since n is divisible by 4, therefore n/2 has the same prime factors as n. Further, since every pair of consecutive integers, such as n/2 −1 and n/2, are coprine, therefore, n/2 −1 and n must also be coprime.

Thus, we can form an (n/2 −1)-distance partition of the jumps of G, and then convert each (n/2 −1)-distance subset into a block by forming a (block) digraph Q of the form: Q(1 ∗ $(n/2 -1)^{-1}$, (n − 1) ∗ $(n/2 -1)^{-1}$: n). It is not hard to see that $(n/2 -1)^{-1}$ = (n/2 −1). This is because (n/2 −1) ∗ (n/2 −1) mod n = n(n/4) −n +1, and since n is divisible by 4, this formula evaluates to 1 (mod n). That is, Q has the form Q(1 ∗ (n/2 −1), (n −1) ∗ (n/2 −1): n) = Q(n/2 −1, n∗(n/2) −3(n/2) +1: n) = Q(n/2 −1, −3(n/2) +1: n) since n∗(n/2) mod n = 0. Further, if we represent each jump in Q in a positive form, we obtain Q(n/2 −1, −3(n/2) +1: n) = Q(n/2 −1, 2n −3(n/2) +1: n) = Q(n/2 −1, n/2 +1: n). Thus, a k-ft of Q (which is also a k-ft of G) is equal to H(n/2 −1, n/2, n/2 +1,.., n/2 +1+k: n+k). □

For example, by the above lemma, G(1,11: 12) has a 2-ft of the form H(5,6,7,8,9: 14). The solution for the remaining case is given below.

Lemma 5.6: Let n be even and not divisible by 4. Then, if k ≤ 3, G(1, n −1: n) has a k-ft of the form H(n/2 −2, n/2 −1,.., n/2 −2+k, n/2 +2,.., n/2 +2+k : n+k); and if k > 3, G(1, n −1: n) has a k-ft of the form H(n/2 −2, n/2 −1 ,.., n/2 +2+k : n+k).

Proof: Since n is even and not divisible by 4, then both $(n + 2)$ and $(n - 2)$ are multiple of 4, and moreover, either $(n + 2)/4$ or $(n - 2)/4$ must be odd. Therefore, let $(n + 2)/4$ be odd. (The proof for the other case where $(n - 2)/4$ is odd is similar, and therefore it is omitted.)

The only factor common to both n and n+2 is 2, and since (n+2)/4 is odd, then it must be coprime to n. That is, we can form an ((n+2)/4)-distance partition of the jumps of G, and then convert each ((n+2)/4)-distance subset into a block by forming a (block) digraph Q for G that has the form $Q(1 * ((n+2)/4)^{-1}, (n-1) * ((n+2)/4)^{-1}: n)$, where the inverse is computed mod n.

It is not difficult to see that $((n+2)/4)^{-1} = (n/2 +2)$. This is because the product $(n+2)/4 * (n/2 +2)$ is equal to the formula $n[(n+2)/4 +1]/2 +1$, and since $(n+2)/4$ is odd, then $[(n+2)/4 +1]$ must be even, i.e., $n[(n+2)/4 +1]/2 +1 \pmod n = 0 +1 = 1$. That is, the digraph Q has the form $Q(1 * (n/2 +2), (n-1) * (n/2 +2): n)$. Consequently, after applying the mod n operation in computing the jumps of Q, we obtain $Q(n/2 +2, n/2 -2: n)$. Thus, for $k \le 3$, a k-ft of Q, which is also a k-ft of G, is equal to $H(n/2 -2, n/2 -1,.., n/2 -2+k, n/2 +2,.., n/2 +2+k : n+k)$; and for $k > 3$, a k-ft of Q will be equal to $H(n/2 -2, n/2 -1 ,.., n/2 +2+k : n+k)$. □

The results developed for the ring $G(1, n-1: n)$ are grouped together in the following theorem.

Theorem 5.3: For any $k \ge 1$, we can form a k-ft H for the ring $G(1, n -1: n)$ as follows.

a) If n is odd, then H will have the form $H((n -1)/2, (n -1)/2 +1,.., (n -1)/2 +1+k : n+k)$.

b) If n is divisible by 4, H will have the form $H(n/2 -1, n/2,.., n/2 +1+k : n+k)$.

c) If n is even and not divisible by 4, then H will have the form $H(n/2 -2, n/2 -1,.., n/2 -2+k, n/2 +2,.., n/2 +2+k : n+k)$ when $k \le 3$ and will have the form $H(n/2 -2, n/2 -1 ,.., n/2 +2+k : n+k)$ when $k > 3$.

Thus, in all cases, the number of jumps in H is between k+2 and k+5.

Proof: The proof is as given in preceding 3 lemmas. □

Actually, the solutions developed in the above three theorems are either optimal or nearly-optimal as implied by the lower-bound proven in the following theorem.

Theorem 5.4: Given a distributed loop G with d jumps, any k-ft H of G must have a degree $\ge 2(d+k)$.

Proof: Suppose to the contrary a k-ft solution H has a smaller degree than 2(d+k). Then, for any node u, either its in-degree(u) or out-degree(u) is less than d+k. Suppose, for example, in-degree(u) < (d+k) and select any set of k nodes "adjacent to u" as being faulty. Excluding or removing these k nodes from H will make the new value of in-degree(u) less than d, i.e., the digraph obtained by excluding these k nodes from H cannot contain a subgraph isomorphic to G, which implies that H cannot be a k-ft of G leading to a contradiction. □

By the above lower-bound, the k-ft solutions developed in Theorem 5.1 for the loop $G(1, m: m^2)$ are optimal in node-degree. Similarly, the k-ft developed in Theorem 5.2 for the (daisy-chain) loop $G(1, n\text{-}2: n)$ are optimal when n is not divisible by 3 and nearly-optimal when 3 divides n. Further, the solutions developed above for the bidirectional ring $G(1, n -1: n)$ are optimal (when n is odd) or nearly-optimal (when n is even).

6. CONCLUSIONS

Distributed loop networks form the underlining topology of many local area networks and some parallel computers (such as the ILLIAC machine). This class of networks satisfies many useful properties and includes (as special cases) important topologies such as rings and circulant graphs, for examples.

Due to the highly regular structure of distributed loops, they tend to be vulnerable to node failures. Accordingly, we have examined the problem of extending these networks so as to make them more fault-tolerant. Our method can be used to tolerate any desired number of node failures; and moreover, it can also be applied when the parameters that define the distributed loop are given numerically or symbolically. Finally, our k-ft solutions appear to be efficient (i.e., either optimal or nearly-optimal as was proven for many cases in this paper).

7. ACKNOWLEDGMENTS

My thanks go to the reviewers for their useful suggestions, and to our system manager, Mr. Chris Maxwell, for helping with the formatting.

REFERENCES

[1] Anderson, T., and P. Lee, "Fault-Tolerance Principles and Practice", Prentice-Hall International, London, 1981.

[2] J. Bermond, F. Comellas and D. Hsu, "Distributed Loop Computer Networks: A Survey", J. of Parallel and Dist. Computing, 24, 1995, pp. 2-10.

[3] Bruck, J., R. Cypher and C. Ho, "Fault-Tolerant Meshes and Hypercubes With Minimal Number of Spares", IEEE Trans on Comp, 42, no. 9, September 1992, pp. 1089-1103.

[4] Choudum, A., and S. Sivagurunathan, "Optimal fault-tolerant networks with a server", *in Netowrks*, V. 35, N. 2, Mar 2000, pp. 157-=160.

[5] Chuang, Y., L. Hsu and C. Chang, "Optimal 1-edge Fault-Tolerant Designs for Ladders", Info. Proc. Letters, 84, no. 2, October 2002, pp. 87-92.

[6] Chung, F., F. Leighton, and A. Rosenberg, "Diogenes: a Methodology for Designing Fault-Tolerant VLSI Processor Arrays", in Proc. IEEE 13th Conf on Fault Tolerant Computing Symp, June 1993, Chicago, IL, pp. 26-32.

[7] Dawson, R. and A. Farrag, "Fault-tolerant Extensions of Star Networks," *in Networks*, V. 21, July 1991, pp. 373-385.

[8] Dutt, S., and J. Hayes, "Designing Fault-Tolerant Systems Using Automorphisms", J. on Parallel and Dist Computing, 12, no. 3, July 1991, pp. 249-268.

[9] Farrag, A., "Algorithm for Constructing Fault-Tolerant Solutions of the Circulant Graph Configuration", in Proc. of 5th IEEE Symp. on Frontiers of Massively Parallel Computations, February 1995, McLean, Virginia, pp. 514-520.

[10] Farrag, A., "Developing Fault-Tolerant Distributed Loops", *Information Processing Letters*, V. 111, N. 2, December 2010, pp. 97-101.

[11] Farrag, A., and S. Lou, "Applying Fault-Tolerant Solutions of Circulant Graphs to Multi-Dimensional Meshes" in Computers & Mathematics Journal, 50, no. 8-9, November 2005, pp. 1383-1394.

[12] Farrag, A., S. Lou and Y. Qi, "Fault-tolerance and Reconfiguration of Circulant Graphs and Hypercubes", *in Proc. of High Performance Computing Symposium (HPCS-08)*, Ottawa, Ontario, Canada, Apr 2008, pp. 475-481.

[13] Grnanov, A., L. Kleinrock, and M. Gerla, "A highly Reliable Distributed Loop Network Architecture", in Proc. IEEE Symp. Fault Tolerant Computing, Kyoto, Japan, Oct 1980, pp. 319-324.

[14] Lipson, J., "Elements of Algebra and Algebraic Computing", Benjamin/Cummings Publisher, Melno Park, California, 1981.

[15] Liu, M. T., "distributed Loop Computer Networks", Advances in Computers, Vol. 17, Academic Press, New York, 1978, pp. 163-221.

[16] Peha, J., and F. Toubagi, "Analyzing the Fault Tolerance of Double-Loop Networks", IEEE/ACM Trans. on Networking, V. 2, N. 4, August 1994, pp. 363-373.

[17] Raghavendra, C., M. Gerla and A. Avizienis, "A Reliable Loop topologies for Large Local Computer Networks", *IEEE Trans. Computer*, V. 34, N. 1, Jan 1985, pp. 46-55.

[18] Raghavendra, C., and J. Silvester, "Double Loop Network Architectures - A Performance Study", *IEEE Trans. Comm.*, V. 33, N. 2, Feb 1985, pp. 185-187.

[19] Rennels, D., "On Implementing Fault-Tolerance in Binary Hypercubes", Digest of papers of IEEE Symp on Fault-Tolerant Comp, (July), Vienna, Austria, 344-349.

[20] Schmitter, E., and P. Baues, "The Basic Fault-Tolerant System", IEEE Micro, 4, no. 1, February 1984, pp. 66-74.

[21] Snyder, L., "Introduction to the Configurable, Highly Parallel Computer," IEEE Computer, 15, no. 1, January 1992, pp. 47-56.

[22] Sung, T., M. Lin, and T. Ho, "Multiple-Edge-Fault Tolerance with respect to Hypercubes", IEEE Trans. Parallel and Dist Systems, 8, no. 2, February 1997, pp. 187-192.