Proceedings of the IASTED International Conference
PARALLEL AND DISTRIBUTED COMPUTING AND SYSTEMS
August 21-24, 2001, Anaheim, California, USA

A-57

Node Disjoint Shortest Paths for Pairs of Vertices in an n-Cube Network

Teofilo F. Gonzalez and F. David Serena
Department of Computer Science
University of California at Santa Barbara
{teo, dserena}@cs.ucsb.edu

ABSTRACT

In parallel and distributed systems many communications take place concurrently, so the routing algorithm as well as the underlying interconnection network plays a vital role in delivering all the messages efficiently. Fault tolerance and performance are often obtained by delivering the messages through node disjoint shortest paths. In this paper we develop an efficient message routing algorithm that generates node disjoint shortest paths in the presence of faulty nodes for a set of pairs of vertices in an n-cube. The time complexity of the algorithm is $O(m^2)$, when the input length is $m = O(n^2)$.

KEY WORDS

Parallel systems, fault-tolerance, hypercube, n-cube, node disjoint shortest paths, routing.

1. Introduction

The n-cube is a vital structure for parallel computing. Several systems with this communication architecture have been built in the past and the SGI Origin 2000 [1] is a recent computing system whose interconnection network is a variation of the n-cube. There are many algorithms for several different routing problems that arise while executing code on an n-cube connected machine. In this paper we present an efficient algorithm for a routing problem which is common to many applications.

Our node disjoint shortest paths problem is defined over the n-cube and the input consists of p pairs of vertices $X = \{X_1, X_2, \ldots, X_p\}$, where $X_i = \{s_i, t_i\}$. The problem consists of constructing node disjoint shortest paths from s_i to t_i for all $1 \leq i \leq p$. Zero length pairs or blocking nodes may also be included in the problem input. We show that under certain conditions a solution exists and we present an $O(m^2)$ time algorithm, where m is the input length, to construct a set of such paths. Node disjoint paths have been studied extensively in the context of the mesh and planar graphs problems for VLSI systems.

The present dialogue regarding the n-cube originated in the context of Rabin's paper [7] wherein three practical concerns are analyzed simultaneously and algorithms proposed: fault-tolerance, security and load balancing. While the paper is an interesting starting point note that in general, three arbitrary parameters of optimization may sometimes be contradictory or at least mutually interdependent.

Gu and Peng [4] address an interesting and very similar problem; however they study the problem of constructing set-to-set node disjoint shortest paths. The main difference between their problem and ours is that they just need to find a path from every vertex in one set to a different vertex in the other set; where as, in our problem one constructs paths for a given set of pairs of vertices. Our problem has applications when network traffic endpoints are defined by empirically observed flows; or are specifically initiated by the individual nodes in the network to designated destinations. The node-to-set node disjoint shortest paths problem reduces to our problem after finding the first link for each path. Gu and Peng's paper [4] exploits the fact that local constraints, the existence of a first step in a path, dominate in determining the existence of node-to-set node disjoint paths. Namely the existence of an SDR or System of Distinct Representatives is a necessary and sufficient condition for the existence of node disjoint paths. We have observed to some extent, similar characteristics in restricted versions of our problem. Node-to-set approaches in the n-cube are highly pragmatic in the sense that it has applications in the context of fault-tolerant distributed networks. There are several important papers which address this issue for the n-cube, [2, 6].

In the next subsection we discuss the topology of the n-cube and then in the next section we show that node disjoint shortest paths in the n-cube exist for a large set of problem instances. Then we present our algorithm, based on our constructive proofs, to construct one set of such paths. The time complexity of our algorithm is $O(m^2)$, where m, the number of input bits, is $O(n^2)$.

1.1 Topology of an n-Cube

One method of representing the set of vertices V in an n-cube is to assign to each vertex $v \in V$ an integer in the range

$$\underbrace{00\ldots0}_{n}{}_2 = 0 \leq v \leq 2^n - 1 = \underbrace{11\ldots1}_{n}{}_2 .$$

The undirected edges E are given by the set $\{\{a, b\} \mid d(a, b) = 1\}$ where $d(a, b)$ is the number of bits with value one in the binary representation of $a \oplus b$ and $\oplus$ is the bitwise "exclusive or" operation.

This representation of the n-cube is prevalent in the literature of coding theory with the work of Frank Gray and the "Gray Codes". It is well known that paths in an n-cube may be represented with Gray codes. In fact other combinatorial objects such as permutations can be made to have Gray-Code-like orderings; for a survey see [8].

An n-cube or equivalently an n dimensional hypercube is an undirected graph $G = (V, E)$ with the set of verticies $V = \{0, 1, \ldots, 2^n - 1\}$ and the set of edges

$$E = \{\{i, j\} \mid i, j \in V, d(i, j) = 1\} .$$

Since shortest paths in the n-cube are the primary focus of this paper it is instructive to define a shortest path predicate. $\Theta(s, t, u)$ is true if and only if there exists a shortest path in the n-cube from s to t inclusive of u. Formally,

$$\Theta(s, t, u) = [s \oplus t == (s \oplus u \text{ "bitwise or" } u \oplus t)] .$$

An alternate predicate, equivalent for the n-cube, is the distance based predicate

$$\Theta(s, t, u) = [(d(s, u) + d(u, t)) == d(s, t)] ,$$

where $d(s, t)$ is the number of edges between node s and t in the n-cube.

Qualitatively the bits of value one in the "bitwise exclusive or" $(s \oplus t)$ operation can be viewed as bits which change as we move from one node to the next along a shortest path from s to t. Note by the topology of the n-cube at each step only one bit changes, as in the "Gray Code."

2. Node Disjoint Shortest Paths Problem

Let $X = \{X_1, X_2, \ldots, X_p\}$, where $X_i = \{s_i, t_i\}$, be a set of p pairs of n-cube vertices all of which are distinct. The *node disjoint shortest paths problem* is given X in the n-cube, where n is the dimension of the cube, find node disjoint shortest paths connecting each s_i to t_i such that no two paths have a node in common. Each X_i is called a *pair* and the two nodes in it are called *endpoints* of X_i.

In the following section we discuss the extreme version of this problem when $d(X_i) = n$, for all $1 \leq i \leq p$. The endpoints of every pair X_i are complements $(\oplus(2^n - 1))$ of each other because they are in the n-cube and $d(X_i) = n$, i.e., $\bar{s}_i = s_i \oplus (2^n - 1) = t_i$ We show that under certain conditions the set of shortest paths exists for the set of p pairs in X even in the presence of q blocking nodes, single vertices which can be considered as faulty or simply *blocking nodes*. Blocking nodes can be viewed as zero length shortest paths from a vertex to itself.

Our algorithm constructs the set of shortest node disjoint paths by finding for each i one link incident to an endpoint of X_i which will be part of the final path and then reduces the problem of finding p shortest paths in the presence of q blocking nodes in an n-cube into two independent problems in an $(n - 1)$-cube with a total number of p pairs and $p + q$ blocking nodes. Our approach is to provide a constructive proof for our claim, present an algorithm based on our constructive proof and produce additional claims not proved herein due to space constraints. The constructive proof appears in subsection 2.1 and the algorithm is in Subsection 2.2. Latifi, Ko and Srimani [6] use a variation of this approach, albeit for a different routing problem, for which they generate paths that are almost shortest paths. The algorithm implied by the proof of Lemma 1 in Section 2.1 finds a link for each pair and reduces the problem, using a partitioning transformation, to finding a set of node disjoint paths in an $(n - 1)$-cube.

Let us now introduce notation to represent the subcube defined by the source and destination nodes s and t by

$$K(s, t) = \{u \mid \Theta(s, t, u)\}.$$

Note when $s = t$ then the only solution is $s = t = u$. Vertices in $K(s, t)$ are all the nodes which may reside on a shortest path from s to t, inclusive of the endpoints.

2.1 The Simple Approach: $p \leq \lceil n/2 \rceil$

Our problem is given p pairs of nodes and q blocking nodes denoted by

$$X = \{X_1, X_2, \ldots, X_p, X_{p+1}, X_{p+2}, \ldots, X_{p+q}\}$$

where $X_i = \{s_i, t_i\}$ and $n = d(X_i)$, for $1 \leq i \leq p$, and $X_i = \{a_i\}$ for $p + 1 \leq i \leq p + q$, find node disjoint shortest paths for the pairs X_i that do not include blocking nodes and such that no two such paths have a node in common. We assume that all the endpoints and blocking nodes are different, i.e., the cardinality of the set $e(X)$ is $2p + q$, where

$$\begin{aligned} e(X) &= \{\text{all endpoints and blocking nodes in } X\} \\ &= X_1 \cup X_2 \cup \cdots \cup X_{p+q} . \end{aligned}$$

Our algorithmic approach, based on the proof of Theorem 1, is to construct the set of node disjoint shortest paths as follows. First we find an integer k which represents the position of one of the bits in the binary representation of the vertices and satisfies the property that for each pair of vertices X_i, $1 \leq i \leq p$, each of its endpoints $x \in X_i$ and its neighbor $g(x) = x \oplus 2^k$ (x and $g(x)$ differ only on bit k) in the n-cube are such that all the $g(x)$ nodes are distinct and every $g(x) \notin e(X)$. In Lemma 2 we show that a bit k that satisfies this condition always exists when $p + q < n$. The selection of this bit k is very important because we use it to reduce our original problem of finding node disjoint paths in an n-cube for p pairs of vertices in the presence of

q blocking nodes to two independent problems. One subproblem is in the sub-cube where bit k is zero and the other one in the sub-cube where bit k is one. The first subproblem consists of finding a path for one pair in the presence of a number of blocking nodes. One isolates this problem by transitioning on one pair using bit k. Lemma 1 establishes that a solution exists for this subproblem. The other subproblem has $p-1$ pairs of vertices with another set of blocking nodes. The subproblem is solved by using the inductive proof in Theorem 1.

Let us illustrate our approach with the following example. The value of n is 3, $X_1 = \{000_2, 111_2\}$, and $X_2 = \{100_2, 011_2\}$. Transitioning on the bit $k=0$ is not possible because the neighbor of 000_2 along bit 0 is $100_2 \in e(X)$, but bit 1 and bit 2 are possible choices for k. Using bit $k=1$ our approach is to select from X_1 the endpoint $e_1 = 000_2$ and from X_2 the endpoint $e_2 = 011_2$. Their corresponding neighbors are $g(e_1) = 010_2$ and $g(e_2) = 001_2$. Now the problem is to find a shortest path from 010_2 to 111_2 and one from 001_2 to 100_2. Since both paths have to go through nodes in which bit one is never changed (the bit is always one for the first and zero for the second), it follows that the resulting problems are independent of each other. Therefore, we may refer to the resulting problems as finishing up the path for X_1 in the sub-cube $K(010_2, 111_2)$ with blocking node 011_2 and finishing up the path for X_2 in the sub-cube $K(001_2, 100_2)$ with blocking node 000_2. By deleting bit k the resulting problems reduce to finding a path in $K(00_2, 11_2)$ with blocking node 01_2 and finding a path in $K(01_2, 10_2)$ with blocking node 00_2. Once we find the paths we add the deleted bit and the paths are complete in the 3-cube.

Lemma 1 *Given X consisting of one pair $X_1 = \{s_1, t_1\}$ with $d(X_1) = n$ and $q \leq n-1$ single nodes in an n-cube for all $n \geq 1$, a node disjoint shortest path exist for X_1 that do not include any of the q nodes.*

Proof omitted for brevity.

Lemma 2 *Let $X = \{X_1, X_2, \ldots, X_p\}$ be pairs of vertices inside the an n-cube with $d(X_i) = n$ for $1 \leq i \leq p$, and $\{X_{p+1}, X_{p+2}, \ldots, X_{p+q}\}$ be a set of blocking nodes such that $1 \leq p \leq p+q < n$ and $n > 2$. There exist at least $n-p-q+1$ bits k such that the cardinality of the set*

$$e(X) \bigcup \{s_i \oplus 2^k \mid 1 \leq i \leq p\} \bigcup \{t_i \oplus 2^k \mid 1 \leq i \leq p\}$$

is $4p+q$. This means that no two vertices are identical and thus one may use bit k to start all the paths.

Proof: First we consider the case wherein $q = 0$ and $p = n-1$. For all $0 \leq l < n$ define the set

$$\begin{aligned} S_l = \{\{i,j\} \mid \quad & x \in X_i, y \in X_j, \text{ and}, \\ & x \oplus y = 2^l, \\ & 1 \leq i \leq p, 1 \leq j \leq p\}. \end{aligned}$$

Note that since $n > 2$, i is different j in each of the above pairs. The meaning of these sets is as follows. If $S_l \neq \emptyset$ then the neighbor of at least one endpoint $\{X_1, X_2, \ldots, X_p\}$ using the l-bit transition is an endpoint in X. Therefore bit l cannot be used by our algorithm to reduce the problem to two independent problems of the previously established form. To prove the lemma we need to show that at least $n-p+1$ sets S_l must be empty when $p \leq n-1$. The proof is by contradiction. Assume that for every l, the number of empty sets S_l is fewer than $n-p+1$.

For each l, such that $S_l \neq \emptyset$ and $0 \leq l < n$, select a set $a_l \in S_l$. Since $n > 2$ we know that if $\{i,j\} \in S_l$ then $\{i,j\} \notin S_{l'}$ for all $l' \neq l$ simply because if there exists $x \in X_i$, $y \in X_j$ such that $x \oplus y = 2^l$ and $\bar{x} \oplus \bar{y} = 2^l$ then $d(x, \bar{y}) > 1$ and $d(y, \bar{x}) > 1$. This precludes $\{i,j\}$'s membership in any $S_{l'}, l \neq l'$.

Therefore, all the sets a_l are distinct. Define the graph G' with the vertex set $V' = \{1, 2, \ldots, p\}$ and edge set $E' = \{a_i \mid 0 \leq i < n\}$. We claim that the graph G' does not have a cycle. Let us prove this claim. Suppose there is a cycle C with nodes $h_1, h_2, \ldots, h_r$. Clearly $r \leq p < n$. Since there are no self or multiple edges between node(s) in G' we know that $r > 2$. Without loss of generality assume the edge $\{h_1, h_2\}$, with label k_1, exists because $s_{h_1} \oplus s_{h_2} = 2^{k_1}$ and thus $\{h_1, h_2\} \in S_{k_1}$. Now, for $i = 2, 3, \ldots, r-2$, we claim the edge $\{h_i, h_{i+1}\}$ exists because $s_{h_i} \oplus s_{h_{i+1}} = 2^{k_i}$. The reason is that the endpoints of any pair X_i can be relabeled and if $x \in X_{h_i}$ and $y \in X_{h_{i+1}}$ are such that edge $\{h_i, h_{i+1}\}$ exists because of $x \oplus y = 2^l$ then it also is the case that $\bar{x} \oplus \bar{y} = 2^l$. Now for the last edge in the cycle $\{h_r, h_1\}$ there are two possibilities. Either it exists because of $s_{h_r} \oplus s_{h_1} = 2^{k_r}$ or $s_{h_r} \oplus t_{h_1} = 2^{k_r}$.

The former case is impossible because in an n-cube every cycle must make two transitions using the same bit, which is not possible in C because each edge is from a different set S_l. In the latter case, we know that a path composed of endpoints ($e(X)$) in the n-cube (G) yields a cycle in the graph G'. Such a path in an n-cube using distinct dimensions at each step must necessarily be less than n. In G' the constraint on the number of paths implies that $d(X_{h_1}) = r \leq p < n$, which contradicts our condition that $d(X_{h_1}) = n$.

Clearly, the graph G' is a tree. Since there are p paths and hence p nodes in V', then the maximum number of edges is $p-1$. Thus, the number of empty sets is at least $n-p+1$. Now one may for each blocking node transform the problem such that it consists only of paths and therefore the fact that the Lemma holds for $q=0$ and $p=n-1$ implies that it holds for $1 \leq p \leq p+q < n$. The constraint $n-p-q+1$ follows immediately by viewing blocking nodes as length n paths. This completes our proof of the Lemma 2. $\square$

Our proof of Theorem 1 and algorithmic approach are based on the *1-bit step* for finding disjoint paths for p pairs and q blocking nodes in an n-cube. First we find using Lemma 2 an integer k that represents the position of one of the bits; such that, for every pair there is a shortest path that on one end makes a transition on bit k without conflict.

The reduction uses a transition for one pair from 0 to 1 along the k^{th} bit and $p-1$ transitions from 1 to 0 for the remaining pairs. Therefore, the subproblem in which all the k^{th} bit positions are 1 will have one pair that makes transitions from 0 to 1, plus the blocking nodes that have a one in the k bit plus one endpoint of the remaining $p-1$ pairs (the ones that make the transition from 1 to 0). Therefore the resulting problem is $(n-1, 1, q')$, where $q' \leq p+q-1$. This problem can be solved by using a constructive proof of Theorem 1. The other subproblem in which all the k^{th} bit positions are 0 will have the $p-1$ pairs that make transitions from 1 to 0, plus the blocking nodes that have a zero in the k^{th} bit plus one endpoint of the pair that makes the transition from 0 to 1. This resulting problem is $(n-1, p-1, q')$, where $q' \leq q+1$. The solution for this problem is obtained inductively.

Theorem 1 *Given X consisting of p pairs of nodes and q blocking nodes in an n-cube for $n > 1$. Node disjoint shortest paths exist for all i and j such that $1 \leq i \leq p < j \leq p+q$, $d(X_i) = n$, $d(X_j) = 0$, $p \leq \lceil n/2 \rceil$ and $2p + q \leq n + 1$.*

Proof: The proof is by induction on n. The base case is when $n = 2$ which is covered by Lemma 1. The case when $p = 0$ is trivial and when $p = 1$ it falls in the conditions of Lemma 1 since $2p + q \leq n + 1$ which implies that $q \leq n - 1$. So assume for all $p \geq 2$ the Theorem holds for all $n - 1$ and let us prove it for n.

Since $p \geq 2$ then $p+q < n$ and by Lemma 2 we know that a bit k for a valid transition exists and we can apply the 1-bit step to reduce the problem into two subproblems. The first problem is on $n-1$ dimensions and it has $p' = 1$ and $q' \leq n-2p+1+p-1 = n-p \leq n-1$ which fall into the conditions of Lemma 1. The second subproblem is on $n-1$ dimensions and it has $p' = p-1$ and $q' \leq n-2p+1+1$. Therefore, $p+q \leq n-2p+1+1+p-1 \leq n-1$ which holds for $p \geq 2$; hence, the resulting problem falls into the induction hypothesis. The theorem follows by induction. []

In the following we will write (n, p, q) which represents all problem instances in an n-cube with p paths and q blocking nodes. Note that these equivalence classes do not correlate with the underlying decision problem of whether or not node disjoint paths exist between the endpoints. For example note that some placements for $(4, 2, 3)$ have node disjoint paths and others do not. Note that input $\{\{0000_2, 1111_2\}, \{0011_2, 1100_2\}, \{0101_2\}, \{0110_2\}, \{1001_2\}\}$ has no solution, but $\{\{0000_2, 1111_2\}, \{0011_2, 1100_2\}, \{0001_2\}, \{0110_2\}, \{1001_2\}\}$ does.

For the case when $n = 4$ it is impossible to strengthen our results because for $p \leq \lceil n/2 \rceil + 1 = 3$ all the problem instances cannot be be solved, i.e., disjoint paths do not exist. One such problem instance is given in Example 1.

Example 1 *The instance of* $(n, p, q) = (4, 3, 0)$ *with*

$$\begin{aligned} X \;=\; & \{\{0000_2, 1111_2\}, \\ & \{1100_2, 0011_2\}, \\ & \{1010_2, 0101_2\}\} \end{aligned}$$

has no solution in the 4-cube. i.e., node disjoint shortest paths do not exist for this problem instance.

However note that there is a bit $k = 3$ (i.e. $0000_2 \oplus 2^3 = 1000_2$) as noted in the proof of Lemma 2. Applying the 1-bit step we end up with the problems $\{X_1 = \{000_2, 111_2\}, X_2 = \{100_2\}, X_3 = \{010_2\}\}$, and $\{X_1 = \{100_2, 011_2\}, X_2 = \{100_2, 011_2\}, X_3 = \{000_2\}\}$, The former problem can be solved, but not the latter one simply because there are not enough vertices in a 3-cube for two paths of length 3 and a blocking node.

As will be shown Theorem 1 can clearly be much improved. Example 2 gives a problem instance with $n = 5$ and $p = 4$ for which node disjoint shortest paths exist while Theorem 1 does not cover this scenario.

Example 2 *(n,p,q)=(5,4,0) with the pairs defined by*

$$\begin{aligned} X \;= & \{\{00000_2, 11111_2\}, \{00001_2, 11110_2\}, \\ & \{00010_2, 11101_2\}, \{00100_2, 11011_2\}\}\,. \end{aligned}$$

In binary, a set of disjoint paths for this problem is given by

$$\begin{aligned}
&00000 \leftrightarrow 01000 \leftrightarrow 01001 \leftrightarrow 01011 \leftrightarrow 01111 \leftrightarrow 11111 \\
&00001 \leftrightarrow 00011 \leftrightarrow 00111 \leftrightarrow 00110 \leftrightarrow 01110 \leftrightarrow 11110 \\
&00010 \leftrightarrow 01010 \leftrightarrow 11010 \leftrightarrow 11000 \leftrightarrow 11001 \leftrightarrow 11101 \\
&00100 \leftrightarrow 00101 \leftrightarrow 10101 \leftrightarrow 10111 \leftrightarrow 10011 \leftrightarrow 11011
\end{aligned}$$

It is interesting to note that our proof technique (algorithmic approach) will not work for the above problem instance. The reason is that the only transition that can be made is on bit $k = 0$ or $k = 1$. Since both cases are similar, under interchange of dimensions, we only discuss the case when the transition is on bit $k = 0$.

When a transition on bit $k = 0$ is made according to our 1-bit step, we advance as follows

$$\begin{aligned}
11111_2 &\rightarrow 01111_2 \\
00001_2 &\rightarrow 10001_2 \\
00010_2 &\rightarrow 10010_2 \\
00100_2 &\rightarrow 10100_2
\end{aligned}$$

In the resulting problem the sub-cube in which bit 0 is always a one has the pairs

$$X = \{\{0001_2, 1110_2\}, \{0010_2, 1101_2\}, \{0100_2, 1011_2\}\}\,.$$

"Exclusive or"ing the input with 0001_2 yields the equivalent problem

$$X = \{\{0000_2, 1111_2\}, \{1100_2, 0011_2\}, \{1010_2, 0101_2\}\}$$

Which is Example 1 and does not have a solution. As in general there may be problem instances that are not amenable to the 1-bit change approach, in the sequel we seek to improve the sufficiency conditions required to insure that node disjoint paths exist.

As the number of edges increases exponentially in the n-cube as n increases linearly, one would expect that a much stronger sufficiency condition exists.

2.2 The Algorithm

The following algorithm assumes that solutions are calculated for dimension $n = 2$, forming a base case of the recursion. The algorithm uses Theorem 1 to partition the problem. The program is called initially with Find_P aths(n, X).

2.2.1 Algorithm

```
Find_P aths($n$, $X = \{X_1, X_2, \ldots, X_p,$
                      $X_{p+1}, \ldots, X_{p+q}\}$)
 {
 assume that $d(X_1) = d(X_2) = \ldots = d(X_p) = n$,
   $d(X_{p+1}) = d(X_{p+2}) = \ldots = d(X_{p+q}) = 0$,
   all $X_i$ are in the same $n$ dimensional sub-cube,
   $p \leq \lceil n/2 \rceil$ and $2p + q \leq n + 1$
 if $p = 0$ return;
 if $p = 1$ construct path using the
      proof of Lemma 1, output and return;
 if $n \leq 2$ return appropriate base case;
 $k \leftarrow$ Find_Bit($X$); // using the proof of Lemma 2
   //find an unobstructed bit.
 $X' \leftarrow$ OneToZeroTransition($k, X$);
 $X'' \leftarrow$ ZeroToOneTransition($k, X$);
 Output_P aths(k,X);
 Output path for $X'$ using the proof of Lemma 1.
 Find_P aths($n - 1, X''$);
 }
```

Output_P aths(k,X) outputs the selection of the particular path. OneToZeroTransition selects one path per the proof of Theorem 1 for transition from 0 to 1 on bit k as well as the blocking nodes with a 1 in the k^{th} bit position. ZeroToOneTransition selects $p - 1$ paths for transition from 1 to 0 on bit k as well as the blocking nodes with a 0 in the k^{th} bit position.

2.3 Proof of Correctness

The proof of correctness follows directly from the constructive proof of Theorem 1.

2.3.1 Analysis: Polynomial Order

Find_Bit is to be implemented using a binary trie for set insertion. The call finds a bit k with no conflicts for the $4p+q$ insert operations. For the n bits each insert is length n bits and the number of such operations is $4p+q$. Therefore this operation will take at most $O(n^2(4p+q))$ steps. There are at most p such calls. Therefore the order of the algorithm is $O(pn^2(4p+q))$. The condition of Theorem 1 requires that $p \leq \lceil n/2 \rceil$. Therefore the algorithm runs in $O(pn^2(4p+q))) = O(pn^2(4p+q)) = O(n^4)$. Since the input length is $m = O(n^2)$ the overall time complexity is $O(m^2)$.

3. Conclusion

The paper presents a solution for a restricted version of the node disjoint pairs in an n-cube. Our results for an improved sufficiency conditions appear in [3]. This improvements are based on an extension to Lemma 2 and a more sophisticated transformation step. The transformation step ends dividing a problem into four subproblems all of which belong is different $n - 2$ cubes. But this new algorithm is slower than the one in this paper.

REFERENCES

[1] J. Ammon. Hypercube Connectivity within the ccNUMA Architecture. Silicon Graphics, May 1998.

[2] S. Gao, B. Novick, and K. Qui. From Hall's matching theorem to optimal routing on hypercubes. *Journal of Combinatorial Theory, Series B*, 74(2):291–301, Nov. 1998.

[3] T. F. Gonzalez and F. D. Serena. n-Cube Network: Node Disjoint Shortest Paths for Pairs of Vertices. Technical Report TRCS-12.01, University of California at Santa Barbara, July 2001.

[4] Q.-P. Gu and S. Peng. Node-to-set and set-to-set cluster fault tolerant routing in hypercubes. *Parallel Computing*, 24:1245–1261, 1998.

[5] D. F. Hsu. On Container Width and Length in Graphs, Groups and Networks. *IEICE Trans. Fundamentals*, E77-A(4):668–680, 1994.

[6] S. Latifi, H. Ko, and P. K. Srimani. Node-to-Set Vertex Disjoint Paths in Hypercube Networks. *Computer Science Technical Report, Colorado State University*, CS-98-107, 1998.

[7] M. O. Rabin. Efficient Dispersal of Information for Security, Load Balancing, and Fault Tolerance. *Journal of the Association of Computing Machinery*, 36(2):335–348, April 1989.

[8] C. Savage. A Survey of Combinatorial Gray Codes. *SIAM Review*, 39(4):605–629, Dec. 1997.