Edge Congestion and Topological Properties of Crossed Cubes

Chien-Ping Chang, Ting-Yi Sung, Mombo, IEEE, and Lih-Hsing Hsu

Abstract—An n-dimensional crossed cube, $CQ_n$, is a variation of hypercubes. In this paper, we give a new shortest path routing algorithm based on a new distance measure defined herein. In comparison with Efe's algorithm, which generates one shortest path in $O(n^2)$ time, our algorithm can generate more shortest paths in $O(n)$ time. Based on a given shortest path routing algorithm, we consider a new performance measure of interconnection networks called edge congestion. Using our shortest path routing algorithm and assuming that message exchange between all pairs of vertices is equally probable, we show that the edge congestion of crossed cubes is the same as that of hypercubes. Using the result of edge congestion, we can show that the bisection width of crossed cubes is $2^{n-1}$. We also prove that wide diameter and fault diameter are $3^n - 2$. Furthermore, we study embedding of cycles in crossed cubes and construct more types than previous work of cycles of length at least four.

Index Terms—Crossed cubes, hypercubes, shortest path routing, wide diameter, fault diameter, edge congestion, bisection width, embedding.

1 INTRODUCTION

Network topology is a crucial factor for interconnection networks since it determines the performance of a network. Many interconnection network topologies have been proposed in the literature for connecting hundreds or thousands of processing elements. Network topology is always represented by a graph in which vertices represent processors and edges represent links between processors. Among these topologies, the binary n-cube (abbreviated as hypercube), denoted by $Q_n$, is one of the popular topologies. However, a hypercube does not make the best use of its hardware, since it is possible to fashion networks with lower diameters than that of $Q_n$. One such topology is the crossed cube, which was first proposed by Efe [1]. An n-dimensional crossed cube, denoted by $CQ_n$, is derived from $Q_n$ by changing the connection of some hypercube links. It has a diameter of $\lceil n/2 \rceil$, an improvement of approximately a factor of 2, in a trade-off of reducing high degree of symmetry in $Q_n$.

Though embedding and some topological properties of crossed cubes have been studied in the literature [1], [2], [3], [6], [7], [9], we study some different topological properties and provide different schemes from previous work. To be specific, we consider a new performance measure called edge congestion. In addition, we consider the following performance measures of crossed cubes: shortest path routing complexity, diameter, wide diameter, fault diameter, bisection width, and embedding of cycles.

Efe presented a shortest path routing algorithm of crossed cubes in [1], which generated one shortest path for any pair of vertices in $O(n^2)$ time. In this paper, we define a new distance measure which enables us to find more shortest paths for any pair of vertices in $O(n)$ time.

When transmitting or broadcasting messages, heavily congested edges will delay communication time. A network having a relatively balanced communication load of edges under the specified routing algorithm is preferred. Motivated by this observation, we introduce the notion of edge congestion, independent of Fiduccia and Hedrick's work [4]. Assuming that message exchange between all pairs of vertices is equally probable, we thus consider all-pair shortest path routing for calculating edge congestion. For each edge, we measure the number of paths of vertices that will route through this edge given a specific routing algorithm. Edge congestion of a network under a specified routing algorithm is the maximum of the congestion of all edges. We define edge congestion of a network by taking the minimum over all routing algorithms. Smaller edge congestion is preferred. In this paper, we first specify our routing strategy based on our shortest path routing algorithm and show that the edge congestion of the crossed cube $CQ_n$ is equal to that of the hypercube $Q_n$. Using the result of edge congestion, we can calculate the bisection width of crossed cubes.

Disjoint paths between a pair of vertices contribute to multipath communication between these two vertices and provide alternative routes in the case of vertex or link failures. The notion of connectivity, wide diameter, and fault diameter is defined based on multiple disjoint paths. In [6], the author showed the existence of a disjoint paths for any pair of vertices in a crossed cube $CQ_n$ to prove its connectivity, without obtaining the length of these paths. In
other words, one cannot get wide diameter from this construction. Wide diameter and fault diameter of a crossed cube $CQ_n$ are studied in this paper.

The problem of simulating one network by another can be modeled as a graph embedding problem. Embeddings of complete binary trees and cycles into crossed cubes were presented in [3], [7], [9]. In comparison with [3], we give a concrete construction of cycles of arbitrary length. In [9], the authors constructed one type of cycles for an arbitrary length, whereas we construct various types of cycles in this paper.

The rest of this paper is organized as follows: Section 2 summarizes some known results on crossed cubes and introduces notation used in this paper. In Section 3, we define a distance measure. Based on this measure, we give a new shortest path routing algorithm which runs in $O(n)$ time. Wide diameter and fault diameter are studied in Section 4. In Section 5, we define the notion of edge congestion and compare edge congestion of hypercubes and crossed cubes. In addition, we calculate bisection width of crossed cubes. Embedding of cycles into crossed cubes by constructing various types of cycles is presented in Section 6. Finally, we make concluding remarks in Section 7.

2 Preliminaries and Notation

Let $G$ be a graph. We use $V(G)$ and $E(G)$ to denote the vertex set and the edge set of $G$, respectively. Let $x$ and $y$ be two vertices. We use $d_G(x, y)$ to denote the distance between $x$ and $y$ in $G$. To define crossed cubes, we first introduce the notation of “pair related.” Let

$$R = \{(00, 00), (10, 10), (01, 11), (11, 01)\}.$$  

Two binary strings $x = x_1x_2\ldots x_n$ and $y = y_1y_2\ldots y_n$ are pair related if and only if $(x, y) \in R$.

Definition 1: An $n$-dimensional crossed cube $CQ_n$ is recursively constructed as follows: $CQ_1$ is a complete graph with two vertices labeled by 0 and 1, respectively. $CQ_n$ consists of two identical $(n-1)$-dimensional crossed cubes $CQ^0_{n-1}$ and $CQ^1_{n-1}$. The vertex $v = v_{n-2}\ldots v_0 \in V(CQ^1_{n-1})$ and the vertex $v = v_{n-2}\ldots v_0 \in V(CQ^0_{n-1})$ is an edge in $CQ_n$ if and only if

1. $v_{n-2} = v_{n-2}$ if $n$ is even, and
2. $(v_{j+i}v_{j+1}v_{j+2}) \in R$ for all $0 \leq i \leq [\frac{n-1}{2}]$.

Examples of $CQ_3$ and $CQ_4$ are illustrated in Fig. 1.

Throughout the paper, each vertex in $CQ_n$ is represented by an n-dimensional binary vector, e.g., $u = u_{n-1}u_{n-2}\ldots u_0$ and $v = v_{n-1}v_{n-2}\ldots v_0$. For $k < n$, the $k$-prefix of $u$, $p_k(u)$, is defined as $u_{n-k}u_{n-k-1}\ldots u_0$. We can thus write $u = p_k(u)u_{n-k}\ldots u_0$.

Let $x$ be an $l$-bit string with $l \leq n$. We use $CQ_n(x)$ to denote the subgraph of $CQ_n$ induced by the set of vertices with the prefix $x$. It is shown in [7] that $CQ_n(x)$ is isomorphic to $CQ_{n-l}$.

Let $x$ and $y$ be two distinct $l$-bit strings with $l \leq n$. If $CQ_n(x)$ and $CQ_n(y)$ can be joined by an edge in $CQ_n$, then $CQ_n(x)$ and $CQ_n(y)$ are called adjacent subgraphs of $CQ_n$. It can be easily verified that if $CQ_n(x)$ and $CQ_n(y)$ are adjacent subgraphs of $CQ_n$, $(x, y)$ is an edge in $CQ_n$. However, $(x, y) \in E(CQ_n)$ does not necessarily imply that $CQ_n(x)$ and $CQ_n(y)$ are adjacent subgraphs of $CQ_n$. For example, when $n = 5$, $x = 01$, and $y = 11$ (i.e., $n-l = 3$), 01 and 11 are adjacent in $CQ_3$, but $CQ_3(01)$ and $CQ_3(11)$ are not adjacent subgraphs of $CQ_5$. Note that if $(x, y)$ is an edge in $CQ_n$ and $n-l$ is even, then $CQ_{n-l}(x)$ and $CQ_{n-l}(y)$ are adjacent subgraphs of $CQ_{n-l}$.

Let $CQ_n(x, y)$ denote the subgraph of $CQ_n$ induced by $CQ_n(x) \cup CQ_n(y)$. It is proven in [1] that $CQ_n(x, y)$ is isomorphic to $CQ_{n-l}$ if $CQ_n(x)$ and $CQ_n(y)$ are adjacent subgraphs of $CQ_n$. In particular, let $|x| = |y| = 2$ and $(x, y)$ be an edge in $CQ_2$. Obviously, $CQ_2(x)$ and $CQ_2(y)$ are isomorphic to $CQ_2(x, y)$. It follows from the above discussion that $CQ_{2k}(x)$ and $CQ_{2k}(y)$ are adjacent subgraphs and that $CQ_{2k}(x, y)$ is isomorphic to $CQ_{2k-1}$. Thus, we can contract those vertices in $CQ_{2k}$ having the same prefix of length two into a vertex and obtain a graph with four vertices. It is easy to see that this four-vertex graph is isomorphic to $CQ_2$, as shown in Fig. 2a. For any two vertices $u, v$ in $CQ_n$ with
n even and n ≥ 4, the following statements can be easily observed [1]:

1. If \( p_2(u) = p_2(v) \), then \( u \) and \( v \) are in a subgraph isomorphic to \( C_{Q'\cdot 2} \).
2. If \( d_{Q'\cdot 2}(p_2(u), p_2(v)) = 1 \), then \( u \) and \( v \) are in a subgraph isomorphic to \( C_{Q'\cdot 1} \).
3. If \( d_{Q'\cdot 2}(p_2(u), p_2(v)) = 2 \), then a neighbor \( u' \) of \( u \) and \( v \) are in a subgraph isomorphic to \( C_{Q'\cdot 1} \).

Similarly, we can contract those vertices in \( C_{Q'\cdot 1} \) with the same prefix of length three into a vertex and obtain a graph with eight vertices. Again, this eight-vertex graph is isomorphic to \( C_{Q'\cdot 2} \), as illustrated in Fig. 2b. We can also obtain the following observations for any two vertices \( u, v \) in \( C_{Q'\cdot 2} \) with \( u \) odd and \( n \geq 3 \) [1]:

1. If \( p_2(u) = p_2(v) \), then \( u \) and \( v \) are in a subgraph isomorphic to \( C_{Q'\cdot 3} \).
2. If \( d_{Q'\cdot 2}(p_2(u), p_2(v)) = 1 \), then \( u \) and \( v \) are in a subgraph isomorphic to \( C_{Q'\cdot 1} \).
3. If \( d_{Q'\cdot 2}(p_2(u), p_2(v)) = 2 \), then a neighbor \( u' \) of \( u \) and \( v \) are in a subgraph isomorphic to \( C_{Q'\cdot 1} \).

**Definition 2.** Let \( (u, v) \) be an edge of \( C_{Q'\cdot 2} \). When vertices \( u \) and \( v \) have a leftmost differing bit at position \( d \), we say that \( v \) is the \( d \)-neighbor of \( u \) and that the edge \( (u, v) \) is an edge of dimension \( d \) or \( \dim(u, v) = d \). We call \((u, v)\) a \( d \)-dim edge.

For example, let \( u = 10111 \). The 4-, 3-, 2-, 1-, and 0-neighbors of \( u \) are given by 01011, 11101, 10001, 10101, and 11111, respectively. We use \( N(u) \) to denote the set of neighbors of \( u \).

Let \( k \) be a positive integer satisfying \( k < n - 1 \) and \( n - k \) even. Let \( x = p_2(u) \) and \( y = p_2(v) \). Assume that

\[
(x, y) \in E(C_{Q'\cdot 2}).
\]

It follows from the definition of \( C_{Q'\cdot 2} \) that \((x, y) \in E(C_{Q'\cdot 2})\) if and only if \((v_{2i+1}, v_{2i+2}, v_{2i+3}) \in R \) for \( 0 \leq i < \frac{n-k}{2} \). Since the set \( R \) induces a one-to-one correspondence for all of the 2-bit strings, a vertex \( v \) satisfying \((x, u) \in E(C_{Q'\cdot 2})\) can be uniquely determined by \((x, y) \) and \( n \). We can thus denote \( v \) by \( f(x; y) \). For example, given \((x, y) = (101, 111) \in E(C_{Q'\cdot 2}) \) and \( u = 10111 \), the neighbor of \( u \) in \( C_{Q'\cdot 2} \) with prefix 111 can be uniquely identified and is given by \( v = 11101 \).

Let \( Q = (z_0, z_1, \cdots, z_{m-1}) \) be a path in \( C_{Q'\cdot 2} \). The length of \( Q \) is denoted by \(|Q|\). Two terminal vertices \( x \) and \( y \) of \( Q \) can be denoted by \( t(x; Q) = q \) and \( t(y; Q) = x \). In this paper, paths can be considered in \( n \) directions, e.g., the aforementioned path \( Q \) is directed from \( z_i \) to \( q \). Given path \( Q, z_i \) is called the immediate predecessor of \( z_{i+1} \). We use \( P(u; Q) \) to denote the path in \( C_{Q'\cdot n} \) induced from \( Q \), which is given by \( P(u; Q) = \{(u, z_0, w_1, \cdots, w_m) \} \), where

\[
w_i = f(w_{i-1}; z_{i-1}, z_i)
\]

for all \( 1 \leq i \leq m \). The path \( P(u; Q) \) preserves the length of \( Q \), i.e., \( |P(u; Q)| = |Q| \). For example, let

\[
Q = (10100, 01100, 01101)
\]

and \( u = 1010011 \). Then,

\[
P(u; Q) = (1010001, 0110001, 0110111).
\]

Let \( P_1 \) be a \((u, v)\)-path, and let \( P_2 \) be a \((u, w)\)-path. We define \( P_1 \cup P_2 \) as the concatenation of \( P_1 \) and \( P_2 \) which yields a \((u, w)\)-path, and \( P_1 \cup P_2 \) can also be written as \( (u, u_1', u_2', \cdots, u_m', w) \). The path obtained from \( P_1 \) by removing the subpath \((u_i, u_{i+1}, \cdots, u_m) \) with \( k < m \) is denoted by \( P_1 \backslash (u_i, u_{i+1}, \cdots, u_m) \).

**3 Shortest Path Routing**

Let \( u \) and \( v \) be two distinct vertices in \( C_{Q'\cdot n} \). The \( i \)-th double bit of vertex \( u \) is defined as a 2-bit string \( u_{k+1} u_{k+2} \) for \( 0 \leq i < \frac{n-k}{2} \) and as simply a single bit \( u_{k+2} \) for \( i = \frac{n}{2} \) and \( n \) odd. Bit \( i \) is called the most significant differing bit between \( u \) and \( v \) if \( p_{n-i-1}(u) = p_{n-i-1}(v) \) and \( u_i \neq v_i \). Let

![Fig. 2. Subgraphs of \( C_{Q'\cdot 2} \) and \( C_{Q'\cdot 1} \).](image-url)
\(v^* = \left\lfloor \frac{d}{2} \right\rfloor\), called the most significant differing double bit. We define a function \(\rho\) on \((u, v)\) as follows:
\[
\rho_j(u, v) = \begin{cases} 
0 & \text{if } u_j \neq v_j, \\
2 & \text{if } u_{j+1} v_j = v_{j+1} u_j, \\
1 & \text{otherwise}. 
\end{cases}
\]

Subsequently, we recursively define \(\rho_j(u, v)\) for \(j \leq i^* - 1\) using the notion of distance-preserving pair related (abbreviated as d.p. pair related) which is motivated from the concept of pair related.

**Definition 3.** \(u_{2j+1} u_{2j}\) and \(v_{2j+1} v_{2j}\), for \(j \leq i^* - 1\), are distance-preserving pair related if one of the following conditions holds:

1. \(\{u_{2j+1} u_{2j}, v_{2j+1} v_{2j}\} \in \{(01, 01), (11, 11)\}\) and \(\sum_{k=j+1}^{[d/2]} \rho_k(u, v)\) is even,

2. \(\{u_{2j+1} u_{2j}, v_{2j+1} v_{2j}\} \in \{(01, 11), (11, 01)\}\) and \(\sum_{k=j+1}^{[d/2]} \rho_k(u, v)\) is odd,

3. \(\{u_{2j+1} u_{2j}, v_{2j+1} v_{2j}\} \in \{(00, 00), (10, 10)\}\).

We write \(u_{2j+1} u_{2j} \sim_d v_{2j+1} v_{2j}\) if \(u_{2j+1} u_{2j}\) and \(v_{2j+1} v_{2j}\) are d.p. pair related, and \(u_{2j+1} u_{2j} \not\sim_d v_{2j+1} v_{2j}\) otherwise.

Then \(\rho_j(u, v)\) for \(j \leq i^* - 1\) is recursively defined as follows:
\[
\rho_j(u, v) = \begin{cases} 
0 & \text{if } u_{j+1} u_j \sim_d v_{j+1} v_j, \\
2 & \text{otherwise}. 
\end{cases}
\]

The pair related distance between \(u\) and \(v\) is defined, denoted by \(\rho(u, v)\), as
\[
\rho(u, v) = \sum_{j=0}^{[d/2]} \rho_j(u, v).
\]

For example, let
\[
u = \left\lfloor \frac{d}{2} \right\rfloor = 0010100011000001
\]
and \(v = 0101001110101101\) be vertices in \(G Q_4\). We recursively find \(\rho_1\) as follows: \(\rho_0(u, v) = 0\), \(\rho_1(u, v) = 2\), \(\rho_2(u, v) = 1\), \(\rho_3(u, v) = 0\), \(\rho_4(u, v) = 0\), \(\rho_5(u, v) = 1\), \(\rho_6(u, v) = 0\), and \(\rho_7(u, v) = 4\).

Now consider the relationship between \(d_{CQ_4}(u, v)\) and \(\rho(u, v)\). Let \(P\) be a shortest path from \(u\) to \(v\). If \(u_{2j+1} u_{2j} \not\sim_d v_{2j+1} v_{2j}\), it requires two steps in \(P\). Otherwise, by the definition of \(v^*\) it requires a single edge in \(P\). If \(u_{2j+1} u_{2j} \sim_d v_{2j+1} v_{2j}\) after identifying bits at positions \(p \geq 2j + 2\) with \(v\) using \(\sum_{k=j+1}^{[d/2]} \rho_k(u, v)\) steps, we can reach a vertex such that bits at positions \(2j + 1\) and \(2j\) are identical to \(v_{2j+1}\) and \(v_{2j}\). If \(u_{2j+1} u_{2j} \neq v_{2j+1} v_{2j}\), at least one step is required in \(P\) to change the \((2j + 1)\)th and the \(2j\)th bits to \(v_{2j+1}\) and \(v_{2j}\). With the above observation, we have the following remark.

**Remark 1.** \(\rho(u, v)\) is a lower bound for \(d_{CQ_4}(u, v)\), i.e.,
\[d_{CQ_4}(u, v) \geq \rho(u, v)\].

We now present a shortest path routing algorithm which can generate multiple shortest paths and is different from the one proposed in [1].

**Algorithm: Shortest Path Routing**

**Input:** Vertices \(u\) and \(v\).

**Output:** Shortest paths from \(u\) to \(v\).

**Initialization:** Determine \(\rho(u, v)\) for all \(0 \leq i \leq [d/2] - 1\). Define \(Q_1 = Q_2 = \{j \mid \rho_j(u, v) \neq 0\}\), \(i^* = \{j \mid \rho_j(u, v) \neq 0, j < i^*\}\), and either
\[
u_{2j+1} u_{2j} \sim_d v_{2j+1} v_{2j} \text{ or } u_{2j+1} u_{2j} \not\sim_d v_{2j+1} v_{2j}\).
\]

**Step 1:** If \(T = \emptyset\)
then go to Step 2;
else find a \(j \in T\) and call \(\text{ONE-STEP-ROUTE}(j, Q_1)\).

**Step 2:** If \(Q_1 = \emptyset\)
then output the sequence \(S\) which yields shortest paths, set \(S\) to be empty, and go to Step 4.

**Step 3:** If either \(u_{2j+1} u_{2j} \sim_d v_{2j+1} v_{2j}\) or \(u_{2j+1} u_{2j} \not\sim_d v_{2j+1} v_{2j}\)
holds for some \(i \in Q_1\),
then choose such smallest \(i\);
else choose \(i = \max\{j \mid j \in Q_1\}\).
Call \(\text{ONE-STEP-ROUTE}(i, Q_1)\) and go to Step 2.

**Step 4:** If \(Q_2 = \emptyset\)
then output the sequence \(S\) which yields shortest paths and STOP.

**Step 5:** Choose \(l = \max\{j \mid j \in Q_2\}\) and \(m = \max\{j \mid j \in Q_2\} - 1\).
If \(\rho_l(u, v) = 1\) and \(u_{2m+1} u_{2m} \sim_d v_{2m+1} v_{2m}\),
then call \(\text{ONE-STEP-ROUTE}(m, Q_2)\);
else if \(\rho_l(u, v) = 1\) and \(u_{2m+1} u_{2m} \not\sim_d v_{2m+1} v_{2m}\)
then call \(\text{ONE-STEP-ROUTE}(m, Q_2)\) or call \(\text{ONE-STEP-ROUTE}(l, Q_2)\);
else call \(\text{ONE-STEP-ROUTE}(l, Q_2)\).

Go to Step 4.

**ONE-STEP-ROUTE**(\(j, Q\))

If \(\rho_j(u, v) = 2\)
then route to \(u'\), i.e., the \(2j\)th or \((2j + 1)\)th neighbor of \(u\),
\[\rho_j(u, v) = \rho_j(u', v) - 1;\]
else route to \( u' \), the \((2j + 1)\)th neighbor of \( u \) if
\[
u_{2j+1}v_{2j} \sim v_{2j+1}v_{2j}
\]
and the \( r \)th neighbor of \( u \) if \( u_{2j} \sim v_{2j+1}v_{2j} \),

\[
\text{set } Q = Q \setminus \{ j \}, \\
\rho_j(u,v) = \rho_j(u,v) - 1.
\]

Put \( u' \) in the sequence \( S \) and set \( u = u' \).

end

To illustrate this algorithm, we still use
\[
u = 01101101100001
\]
and \( \delta = 01010101100001 \) in \( CQ_{14} \) as an example. At initial stage, we have \( \rho_1(u,v) = 2, \rho_2(u,v) = 1 \), \( \rho_3(u,v) = 1 \), \( Q_1 = \{5,4,1\} \) and \( T = \emptyset \). Since \( T = \emptyset \) and we cannot find any \( i \in Q_1 \) satisfying \( v_{2i+1}v_i \sim v_{2i+1}v_{2i} \) or
\[
u_2v_4v_5v_3v_1v_2,
\]
in Step 3 we choose \( i = 5 \) reaching \( 0100111100001 \) or \( 0110111100001 \), and reduce \( \rho_5(u,v) \) by one. Now, the set \( Q_1 \)
is still \( \{5,4,1\} \). Repeating Step 3, we find \( i = 1 \) in this iteration and route to \( 010011110001 \) or \( 011011110001 \). It reduces \( Q_1 \) to \( \{5,4\} \). Subsequently, \( i = 4 \) and \( i = 5 \) are found in the following iterations of Step 3, which render \( Q_1 \) to be empty. In summary, performing Step 1 to Step 3, we can obtain the following two shortest paths:

\[
P_1 = \{ 01011011100101, 01000111100011, 0100001111001111, 0100001111000101 \}
\]

\[
P_2 = \{ 01010111000010, 01001111100011, 010010111100011, 011000111110101 \}
\]

Performing Step 4 to Step 5, we can obtain two shortest paths \( P_3 \) and \( P_4 \) as follows:

\[
P_3 = \{ 01011011000101, 01001111100011, 010000111100001, 01010011100011, 01010001100101 \}
\]

\[
P_4 = \{ 01011011100010, 01100111100011, 011001001110001, 01001111000101 \}
\]

The paths \( P_3 \) and \( P_4 \) are obtained by changing the fifth, fourth, fifth and first double bits sequentially.

We summarize in Table 1 the verification of the statement whether \( u_{2i+1}v_{2i} \sim v_{2i+1}v_{2i} \) or
\[
u_{2i+1}v_{2i} \sim v_{2i+1}v_{2i}
\]
for \( i \neq j \) is satisfied in Steps 3 and 5. In Table 1, “even” and “odd” represent \( \sum_{i=1}^{j} \rho_i(u,v) \) being even and odd, respectively. We use “yes” and “no” to indicate that the statement holds and does not hold, respectively, and “-” to indicate that \( u_{2i+1}v_{2i} \sim v_{2i+1}v_{2i} \). In other words, ONE_STEP_ROUTE can be performed on those cases that are marked with “yes”. Furthermore, observing from Table 1, we can obtain the following remark.

**Remark 2.** For those cases marked with “no” in Table 1, reducing a \( \rho_i(u,v) \), \( i > j \) by one enables us to perform ONE_STEP_ROUTE on the \( i \)th double bit. Furthermore, let \( k = i' \) or let it be the smallest index in \( Q_1 \) satisfying \( u_{2i+1}v_{2i} \sim v_{2i+1}v_{2i} \) or \( u_{2i+1}v_{2i} \sim v_{2i+1}v_{2i} \). After performing ONE_STEP_ROUTE(k, \( Q_1 \)) and reaching \( v' \), it can be observed that \( u_{2j+1}v_{2j} \sim v_{2j+1}v_{2j} \) or
\[
u_{2j+1}v_{2j} \sim v_{2j+1}v_{2j}
\]
is satisfied for all \( j < k \) and \( j \in Q_1 \).

**Theorem 1.**

1. The Shortest_PathRoutingModule algorithm correctly finds shortest paths from the source to the destination.

2. \( d_{\text{GIC}}(u,v) = d(u,v) \).

**Proof.** Let \( u \) be the source and \( v \) be the destination. Let \( v' \) denote a current vertex reached by the algorithm, and \( Q_1 \) and \( Q_2 \) be the subsets of \( Q_1 \) and \( Q_2 \), respectively, obtained by the algorithm. We note that ONE_STEP_ROUTE is performed at the \( i \)th double bit if \( u_{2i+1}v_{2i} \sim v_{2i+1}v_{2i} \) or \( u_{2i+1}v_{2i} \sim v_{2i+1}v_{2i} \). Therefore, at the end of \( \rho_i(u,v) \) steps of performing ONE_STEP_ROUTE, the \( i \)th double bit becomes identical to that of \( v \). Observing from Steps 1, 3, and 5, we note that

<table>
<thead>
<tr>
<th>( v_{2i+1}v_{2i} )</th>
<th>( u_{2i+1}v_{2i} )</th>
<th>( 00 )</th>
<th>( 10 )</th>
<th>( 11 )</th>
<th>( 01 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( v_{2i+1}v_{2i} )</td>
<td>( \text{yes} )</td>
<td>( \text{no} )</td>
<td>( \text{yes} )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( 00 )</td>
<td>( \text{yes} )</td>
<td>( \text{no} )</td>
<td>( \text{yes} )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( 10 )</td>
<td>( \text{yes} )</td>
<td>( \text{yes} )</td>
<td>( \text{odd} )</td>
<td>( \text{odd} )</td>
<td>( \text{odd} )</td>
</tr>
<tr>
<td>( 11 )</td>
<td>( \text{even} )</td>
<td>( \text{yes} )</td>
<td>( \text{odd} )</td>
<td>( \text{odd} )</td>
<td>( \text{odd} )</td>
</tr>
<tr>
<td>( \text{odd} )</td>
<td>( \text{odd} )</td>
<td>( \text{odd} )</td>
<td>( \text{odd} )</td>
<td>( \text{odd} )</td>
<td>( \text{odd} )</td>
</tr>
<tr>
<td>( \text{even} )</td>
<td>( \text{even} )</td>
<td>( \text{even} )</td>
<td>( \text{even} )</td>
<td>( \text{even} )</td>
<td>( \text{even} )</td>
</tr>
</tbody>
</table>

**TABLE 1**

<table>
<thead>
<tr>
<th>( v_{2i+1}v_{2i} )</th>
<th>( u_{2i+1}v_{2i} )</th>
<th>( 00 )</th>
<th>( 10 )</th>
<th>( 11 )</th>
<th>( 01 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( v_{2i+1}v_{2i} )</td>
<td>( \text{yes} )</td>
<td>( \text{no} )</td>
<td>( \text{yes} )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( 00 )</td>
<td>( \text{yes} )</td>
<td>( \text{no} )</td>
<td>( \text{yes} )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( 10 )</td>
<td>( \text{yes} )</td>
<td>( \text{yes} )</td>
<td>( \text{odd} )</td>
<td>( \text{odd} )</td>
<td>( \text{odd} )</td>
</tr>
<tr>
<td>( 11 )</td>
<td>( \text{even} )</td>
<td>( \text{yes} )</td>
<td>( \text{odd} )</td>
<td>( \text{odd} )</td>
<td>( \text{odd} )</td>
</tr>
<tr>
<td>( \text{odd} )</td>
<td>( \text{odd} )</td>
<td>( \text{odd} )</td>
<td>( \text{odd} )</td>
<td>( \text{odd} )</td>
<td>( \text{odd} )</td>
</tr>
<tr>
<td>( \text{even} )</td>
<td>( \text{even} )</td>
<td>( \text{even} )</td>
<td>( \text{even} )</td>
<td>( \text{even} )</td>
<td>( \text{even} )</td>
</tr>
</tbody>
</table>
ONE_STEP_ROUTE is performed at those double bits with \( \rho(u, v) \neq 0 \). Furthermore, ONE_STEP_ROUTE is performed \(|Q_1| = |Q_2| \) times if \( \rho(u, v) = 1 \) and \(|Q_1| + 1 \) times if \( \rho(u, v) = 2 \), i.e., exactly \( \rho(u, v) \) times. Therefore,

\[
d_{CQ_\infty}(u, v) = \rho(u, v).
\]

On the other hand, we have \( d_{CQ_\infty}(u, v) \geq \rho(u, v) \) by Remark 1. Consequently, we have \( d_{CQ_\infty}(u, v) = \rho(u, v) \), and moreover, the algorithm generates shortest paths from \( u \) to \( v \).

If initially \( T = \emptyset \) and no \( i \in Q_1 \) satisfies

\[
u_{2i+1}v_{2i} \sim_{p} v_{2i+1}v_{2i+2} \]

or \( \nu_{2i+1}v_{2i} \sim_{p} v_{2i+1}v_{2i+2} \), let \( \iota' = \max\{j \mid j \in Q_1\} \) as found in Step 3. It follows that we can choose \( \iota' = \iota^* \) first and then continue the procedure ONE_STEP_ROUTE(i, \( Q_1 \)) for all \( i \in Q_1 \) until \( Q_1 = \emptyset \). Otherwise, we can always perform ONE_STEP_ROUTE on any \( i \in T \) as specified in Step 1 and on the smallest \( i \in Q_1 \) (actually, arbitrary \( i \in Q_1 \) is allowed), satisfying \( \nu_{2i+1}v_{2i} \sim_{p} v_{2i+1}v_{2i+2} \) or

\[
u_{2i+1}v_{2i} \sim_{p} v_{2i+1}v_{2i+2},
\]

as specified in Step 3. In subsequent iterations of repeating Step 3, it follows from Remark 2 that an \( i \in Q_1 \) satisfying \( \nu_{2i+1}v_{2i} \sim_{p} v_{2i+1}v_{2i+2} \) or \( \nu_{2i+1}v_{2i} \sim_{p} v_{2i+1}v_{2i+2} \) can always be found, until \( Q_1 = \emptyset \). Thus, repeating Steps 1 to 3, we can obtain shortest paths.

Similarly, in Steps 4 and 5, the algorithm checks each double bit \( i \) with \( \rho(u, v) \neq 0 \) from left to right order and generates other shortest paths. Hence, the theorem follows.

Step 1 takes \( O(1) \) time. In Step 3, if either

\[
u_{2i+1}v_{2i} \sim_{p} v_{2i+1}v_{2i+2}
\]

or \( \nu_{2i+1}v_{2i} \sim_{p} v_{2i+1}v_{2i+2} \) is satisfied, we perform ONE_STEP_ROUTE. Otherwise, we perform ONE_STEP_ROUTE on the \( j \)-th double bit with \( j \in Q_1 \) and \( j > i \) reaching \( v' \), and then we obtain \( \nu_{2j+1}v_{2j} \sim_{p} v_{2j+1}v_{2j+2} \) or \( \nu_{2j+1}v_{2j} \sim_{p} v_{2j+1}v_{2j+2} \). Thus, we examine each double bit \( i \) with \( \rho(u, v) \neq 0 \) at most twice in increasing order of \( i \). Steps 2 and 3 are executed at most \( 2\rho(u, v) \) times. Steps 4 and 5 are executed exactly \( \rho(u, v) \) times, each taking \( O(1) \) time. Hence, the time complexity of the algorithm is \( O(n) \). The algorithm proposed in [1] can generate one shortest path in \( O(n^2) \) time, while ours can generate more shortest paths in \( O(n) \) time. Furthermore, our algorithm can be modified to generate all shortest paths in \( O(n^2) \) time.

**Theorem 2.** The diameter of \( CQ_n \), denoted by \( D(CQ_n) \), is \( \lceil n^2/2 \rceil \).

**Proof.** Let \( u, v \) be two distinct vertices. Since there are \( \lceil n^2/2 \rceil \) double bits, it follows that \( d_{CQ_\infty}(u, v) = \rho(u, v) \leq \lceil n^2/2 \rceil \). In particular, we choose \( v^* = 0 \cdots 0 \) and \( v^* = 11 \cdots 1 \) such that \( \rho(u^*, v^*) = \lceil n^2/2 \rceil \). Hence, the theorem follows.

### 4. Wide Diameter and Fault Diameter

The connectivity of a network \( G = (V, E) \), denoted by \( \kappa(G) \) or \( \kappa \), is the minimum number of vertices whose removal leaves the remaining graph disconnected or trivial. It follows from Menger's theorem that there always exist \( \kappa \) internally vertex-disjoint (abbreviated as disjoint) paths between any two vertices.

Let \( \alpha \) and \( \beta \) be two positive integers such that \( \alpha \leq \kappa \) and \( \beta \leq \kappa - 1 \). Given any two distinct vertices \( u \) and \( v \) of \( G \), let \( \kappa(u, v) \) denote the set of all \( \alpha \) disjoint paths between \( u \) and \( v \). Each element of \( \kappa(u, v) \) consists of \( \alpha \) disjoint paths. The number of elements in \( \kappa(u, v) \) is denoted by \( |\kappa(u, v)| \). Let \( \lambda(u, v) \) denote the longest length among these \( \alpha \) paths of the \( \kappa \)-th element of \( \kappa(u, v) \). We define \( d_{\alpha}(u, v) \) and \( d_{\alpha}^{\beta}(u, v) \) as follows:

\[
d_{\alpha}(u, v) = \min_{i \leq |\kappa(u, v)|} \lambda(u, v), \quad d_{\alpha}^{\beta}(u, v) = \min_{i \leq |\kappa(u, v)|} \{d(u, v) \mid u, v \not\in F\},
\]

where \( G - F \) denotes the subgraph of \( G \) induced by \( V \). In other words, \( d_{\alpha}(u, v) \) denotes the shortest distance between \( u \) and \( v \) when any \( \beta \)-vertex fault occurs.

**Definition 4.** The \( \alpha \)-wide diameter of \( G \), denoted by \( D_{\alpha}(G) \), is defined as

\[
D_{\alpha}(G) = \max_{u,v\in V} \{d_{\alpha}(u, v)\}.
\]

In particular, we call \( D_{\alpha}(G) \) simply wide diameter of \( G \).

**Definition 5.** The \( \beta \)-fault diameter, denoted by \( D_{\beta}^{\infty}(G) \), is given by

\[
D_{\beta}^{\infty}(G) = \max_{u,v\in V} \{d_{\beta}^{\infty}(u, v)\}.
\]

**Definition 6.** The \( \infty \)-fault diameter, denoted by \( D_{\infty}(G) \), is

\[
D_{\infty}(G) = \max_{u,v\in V} \{d_{\infty}(u, v)\}.
\]

Fault diameter estimates the impact on diameter when faults occur, i.e., the removal of vertices from \( G \). Small \((\kappa - 1)\)-fault diameter is also desirable to obtain smaller communication delay when vertex faults occur. Obviously, we have \( D(G) \leq D_{\infty}^{\beta-1}(G) \leq D_{\infty}(G) \).

For hypercubes, it is known that \( \kappa(CQ_n) = n \) and

\[
D_{\infty}(CQ_n) = D_{\infty}^{\infty}(CQ_n) = n + 1.
\]

In this section, we compare wide diameter and fault diameter of crossed cubes with those of hypercubes and prove that \( D_{\alpha}(CQ_n) = D_{\beta}^{\infty}(CQ_n) = \lceil \sqrt{n} \rceil + 2 \) for all \( n \geq 2 \).

**Lemma 1.** \( D_{\infty}^{\infty}(CQ_n) \geq \lceil \sqrt{n} \rceil + 2 \).

**Proof.** Let \( u, v \) be vertices of \( CQ_n \), and let \( F \) be an \((n - 1)\)-fault set given by \( F = N(u) - \{u\} \). Consider \( n = 2k, k \geq 1 \). We choose \( u = 0 \cdots 0, v = 0 \cdots 1 \). The shortest path from \( u \) to \( v \) in \( CQ_n - F \)
has the form \((u, u', P(u', v))\), where \(P(u', v)\) is a shortest path from \(u'\) to \(v\) without traversing any vertex in \(F\). Since \(d_{Q_n - F}(u, v) = \lceil \frac{n}{2} \rceil + 1\), it follows that
\[
d_{Q_n - F}(u, v) = 1 + d_{Q_n - F}(u', v) = \lceil \frac{n}{2} \rceil + 2.
\]
Therefore, \(D_{n-1}(Q_n) \geq \lceil \frac{n}{2} \rceil + 2\).

Next, let us consider \(n \) odd. Let \( u = 00000 \ldots 0 \) and \( u' = 00100 \ldots 0 \). We choose \( v = 100110110111 \ldots 1101110100 \) for \( n = 4k + 1 \), and
\[
v = 100110110110 \ldots 1101100100100 \ldots 110100100100 \ldots 110100100100 \ldots 110100100100 \ldots 11101000 \ldots 0 \]
for \( n = 4k + 3 \). The shortest path from \( u \) to \( v \) in \( CQ_n - F \) has the form \((u, u', u'', P(u', v), v)\), where \(u''\) is a neighbor of \( u'\) and \(P(u', v)\) is a shortest path from \(u''\) to \(v\) without traversing vertices in \( F \). Note that \( u''\) can be arbitrarily chosen from the set
\[
W = \{011100 \ldots 0, 11100 \ldots 0\} \cup \{00100 \ldots 00, 0000 \ldots 00, 0000 \ldots 00\}
\]
for \( 0 \leq j \leq n - 4 \).

Since any vertex \( u \) in \( W \) satisfies \( d(u, v) = \lceil \frac{n}{2} \rceil \), it follows that \( d_{Q_n - F}(u, v) = 2 + d(u', v) = \lceil \frac{n}{2} \rceil + 2 \). Therefore,
\[
D_{n-1}(Q_n) \geq \lceil \frac{n}{2} \rceil + 2.
\]
Hence the lemma follows.

Lemma 2. Let \( u \) and \( v \) be two vertices of \( Q_n \) for \( n \geq 2 \). Then, there are \( n \) disjoint paths \( P_1, P_2, \ldots, P_n \) joining \( u \) to \( v \) with nondecreasing length, i.e., \( |P_1| \leq |P_2| \leq \cdots \leq |P_n| \), such that
1. \(|P_i| \leq \lceil \frac{n}{2} \rceil + 2 \) for all \( i \).
2. \(|P_1| \leq D(Q_n) = \lceil \frac{n}{2} \rceil \).
3. \(|P_i| = \lceil \frac{n + 1}{2} \rceil \) if \(|P_i| = \lceil \frac{n + 1}{2} \rceil \) and \( n \) is even.

We leave the proof to Appendix A. Note that Lemma 1 and Lemma 2 immediately imply \( \kappa(Q_n) = n \) since each vertex has a degree of \( n \). Since Lemma 2 implies that \( d_n(u, v) \leq \lfloor \frac{n}{2} \rfloor + 2 \), we then have the following corollary.

Corollary 1. \( D_n(Q_n) \leq \lceil \frac{n}{2} \rceil + 2 \) for all \( n \geq 2 \).

Since \( D_{n-1}(Q_n) \leq D_n(Q_n) \), we can easily obtain the following theorem from Corollary 1 and Lemma 1.

Theorem 3. \( D_{n-1}(Q_n) = D_n(Q_n) = \lfloor \frac{n}{2} \rfloor + 2 \).

5 Edge Congestion and Bisection Width

In this section, we treat a path as a directed routing from a source to a destination. To distinguish different orientation of an edge \((u, v)\), we write \([u, v]\) and \([v, u]\) as traversing from \(u\) to \(v\) and from \(v\) to \(u\), respectively. A path \( P = (x = x^0, x^1, \ldots, x^m = y) \) is treated as a directed path from \(x\) to \(y\) consisting of \([x^i, x^{i+1}]\) for \( 0 \leq i \leq m - 1 \). For convenience, we also treat a path as a set of edges and write \([x^i, x^{i+1}] \in P\) to mean that \([x^i, x^{i+1}]\) is in \( P \). We say that an edge \( e = (u, v) \) is incident on \( P \) if \([u, v] \in P \) or \([v, u] \in P \), i.e., \( u = x^i, v = x^{i+1} \) or \( v = x^i, u = x^{i+1} \) for some \( i \). When considering edge congestion, we treat any routing algorithm \( A \) for a network \( G = (V, E) \) as a function assigning each \((x, y) \in V \times V \) to only one path from \( x \) to \( y \), denoted by \( P_A(x, y) \). We consider only the shortest path routing algorithms.

Definition 6. The edge congestion of an edge \( e \in E \) under the routing algorithm \( A \), denoted by \( c_A(e) \), is defined as the number of \((x, y)\) pairs such that \( e \) is incident on \( P_A(x, y) \), i.e.,
\[
c_A(e) = |\{(x, y) \mid x, y \in V, e \text{ is incident on } P_A(x, y)\}|
\]
The edge congestion of the network \( G \) under the routing algorithm \( A \) and the edge congestion of the network \( G \) denoted by \( c_A(G) \) and \( c(G) \), respectively, are defined as follows:
\[
c_A(G) = \max\{c_A(e) \mid e \in E \},
c(G) = \min\{c_A(G) \mid \text{all routing algorithms } A \text{ for } G\}.
\]

Edge congestion can also provide lower bound on the area and the longest wire length required by VLSI layout of networks. A routing algorithm achieving \( c(G) \) is called an optimal routing algorithm.

Definition 7. The bisection width of \( G \), denoted by \( \omega(G) \), is the minimum number of edges to be removed to disconnect the graph into two (not necessarily connected) subgraphs with \( \lceil \frac{|V|}{2} \rceil \) and \( \lceil \frac{|V|}{2} \rceil \) vertices, respectively.

The problem of finding the bisection width of a graph is NP-hard. The bisection width of an interconnection network is a critical factor in determining the speed with which the network can perform a computation and the area needed to layout this network [8].

In this section, we will prove that \( c(CQ_n) = c(Q_n) = 2^n \) and \( \omega(CQ_n) = 2^{n+1} \).

Theorem 4. \( c(Q_n) = 2^n \).

Proof. For any vertex \( u \) in \( Q_n \), it is known that exactly \( \binom{n}{i} \) vertices at distance \( i \) from \( u \). Thus for any routing algorithm \( A \), we have
\[
\sum_{e \in E(Q_n)} c_A(e) = \sum_{(x, y) \in \mathcal{C} \times \mathcal{V}} |P_A(x, y)|
\geq \sum_{x \in \mathcal{V}} \sum_{y \in \mathcal{V}} \left( \begin{array}{c} n \\begin{array}{c} \end{array} \end{array} \right)
\geq \sum_{i=0}^{n} \left( \begin{array}{c} n \\begin{array}{c} \end{array} \end{array} \right)
\geq 2^n, 2^{n+1}.
\]
Since there are exactly \( n \cdot 2^{n-1} \) edges in \( E(Q_n) \),
\[
c_A(Q_n) = \max\{c_A(e) \mid e \in E(Q_n)\} \geq 2^n.
\]
Thus, \( c(Q_n) = 2^n \).

On the other hand, let \( C \) be the routing strategy which routes \( x \) to \( y \) by changing the rightmost differing bit iteratively. To be precise, assume that \( x \) differs from \( y \) at \( k \) bits, say, the \( l_1 \), \( l_2 \), ..., \( l_k \)th bits with
\[
0 \leq l_1 < l_2 < \ldots < l_k \leq n - 1.
\]
Then, \( P_C(x, y) \) is given by \((x = x^0, x^1, \ldots, x^m = y)\), where \( x^i \) differs from \( x^{i+1} \) at the \( l_i \)th bit. Let \( e = (u, v) \) be an edge of \( Q_n \) with \( u \) and \( v \) differing at the \( j \)th bit such that \( (u, v) \) is incident on \( P_C(x, y) \). That is, either \([u, v] \in P_C(x, y)\) or \([v, u] \in P_C(x, y)\). Consider \([u, v] \in P_C(x, y)\). It
We first consider \( u_0 = 0 \). That is, \( u_0, v_0 = 00 \) and \( v_1, v_9 = 10 \). First, assume that \([u, v] \in P_n(x, y)\). It follows from Step 3 of Algorithm \( B \) that the change of the zeroth double bit is first executed, i.e., \( x = x^0 = u \). When \( y_{10} \) is either 00 or 01, it follows from Algorithm \( B \) that \( x^1 \neq u \). Thus,

\[
\{(x, y) \mid [u, v] \in P_n(x, y), \; y_{10} = 00 \; \text{or} \; 01\} = \emptyset.
\]

When \( y_{10} = 10 \), \([u, v] \in P_n(x, y)\) always holds. Thus,

\[
\{(x, y) \mid [u, v] \in P_n(x, y), \; y_{10} = 00 \; \text{or} \; 10\} = 2^{k-2}.
\]

When \( y_{10} = 11 \), it follows from Algorithm \( B \) that \( y \) is exactly the vertex \( u_{-1} u_{-2} \cdots u_{11} \). Therefore, there are \( 2^{k-2} + 1 \) \((x, y)\)-pairs routing through \([u, v]\).

Next, consider \([u, v] \in P_n(x, y)\). Similarly, we first change the 0th double bit and thus, \( x = x^0 = u \). When \( y_{10} = 10 \) or 11, it follows from Algorithm \( B \) that \( x^1 \) cannot be \( u \). Thus,

\[
\{(x, y) \mid [u, v] \in P_n(x, y), \; y_{10} = 10 \; \text{or} \; 11\} = 0.
\]

When \( y_{10} = 00 \), \( P_n(x, y) \) always routes through \([u, v]\).

Thus,

\[
\{(x, y) \mid [u, v] \in P_n(x, y), \; y_{10} = 10 \; \text{or} \; 11\} = 2^{k-2}.
\]

When \( y_{10} = 01 \), it follows from Algorithm \( B \) that \( y \) is exactly the vertex \( u_{-1} u_{-2} \cdots v_{01} \). Therefore, there are \( 2^{k-2} + 1 \) \((x, y)\)-pairs routing through \([u, v]\). From the above discussion, \( c_0(e) = 2^{k-2} + 1 + 2^{k-2} + 1 = 2^{k-1} + 2 \).

Next, we consider \( u_0 = 1 \), i.e., \( u_0, v_0 = 01 \) and \( v_1, v_9 = 11 \).

Again, we have either \([u, v] \in P_n(x, y)\) or \([u, v] \in P_n(x, y)\). First, assume that \([u, v] \in P_n(x, y)\). Suppose \( u = x^1 \). It follows that \( x_1 x_9 = 11, x_1 x_9 = x_9, \) and equivalently \( x_1 x_9 = x_9 = v_1 v_9 = v_1 v_9 \), which contradicts \([u, v] \in P_n(x, y)\). Therefore, we have \( u = x^2 = x \). When \( y_{10} = 01 \), \([u, v] \in P_n(x, y)\) implies that \( p_0 x \neq (u, p_0 x) \) is odd. When \( y_{10} = 11 \), \([u, v] \in P_n(x, y)\) implies that \( p_0 x \neq (u, p_0 x) \) is even. Therefore,
\[ y = p_{k-1}t - (I_1 - 2) \]

Since \([u, v] \in P_B(x, y)\), it follows that \(y_{10} \neq 0\). When \(y_{10} = 10\), \(y\) can be only \(u_{k-1}u_{k-2} \cdots u_2 10\) and, thus,
\[
|\{(x, y) \mid [u, v] \in P_B(x, y), u_{10} = 01\}| = 2^{k-2} + 1.
\]

Furthermore, using similar arguments we can also obtain
\[
|\{(x, y) \mid [u, v] \in P_B(x, y), u_{10} = 11\}| = 2^{k-2} + 1.
\]

Thus, \(c_{B}(e) = 2^{k-2} + 2^{k-2} + 2 = 2^{k-1} + 2\). The lemma now follows. \(\square\)

**Lemma 4.** \(c_{B}(e) = 2^{k}\) for any dim-0 edge in \(CQ_k\).

**Proof.** Let \(e = (u, v) = (u_{k-1}, \ldots, u_1, u_0, v_{k-1}, \ldots, v_0, u_{10} = v_{10} = v_{20} = 0)\) be any dim-0 edge in \(CQ_k\). Obviously, \(y_{k-2}(u) = v_{k-2}(v)\) and \(u_{k-1} = v_{k-1} = v_{20} = 0\). Without loss of generality, we can assume that \(u_0 = 0\). Let \(x = x_{k-1}x_{k-2} \cdots x_0 x_1\) and
\[
y = y_{k-1}y_{k-2} \cdots y_0 y_1
\]
be any two vertices such that \(e\) is incident on \(P_B(x, y)\), where \(P_B(x, y) = (x = x_0, x_1, \ldots, x_m = y)\).

Assume that \([u, v] \in P_B(x, y)\). It follows from Step 3 of Algorithm \(B\) that there are two types of \((x, y)\)-pairs routing through \([u, v]\). The shortest path of the first type of \((x, y)\)-pairs is first to change the zeroth double bit, while the second type is first to change some \(t\)th double bit, \(1 \leq t \leq \lfloor \frac{k}{2} \rfloor\), and then the zeroth double bit. In other words, we have \([u, v] = [x^0, x^1]\) in the former and \([u, v] = [x^1, x^0]\) in the latter.

Consider that \([u, v] = [x^0, x^1]\), i.e., \(x^0 = u\) and \(x^1 = v\). If \(y_{10} = 0\) or \(y_{20} = 1\), \(x^1 = v\) cannot be \(v\), given routing under Algorithm \(B\), since \(u_0 = 0\). When \(y_{10} = y_{20} = 0\), it follows that \([u, v] \in P_B(x, y)\) iff \(p_{k-2}(v), p_{k-2}(y))\) is even. When \(y_{10} = y_{20} = 0\), it follows that \([u, v] \in P(x, y)\) iff \(p_{k-2}(v), p_{k-2}(y))\) is odd. Thus,
\[
|\{(x, y) \mid [u, v] \in P_B(x, y) \text{ and } x = u\}| = 2^{k-2}.
\]

Thus, \(c_{B}(e) = 2^{k-2} + 2^{k-2} + 2 = 2^{k-1} + 2\). The lemma now follows. \(\square\)

Next, consider that \([u, v] = [x^1, x^0]\). Clearly, \(x \in N(u)\) and \(x\) is not the 0-neighbor of \(u\). When \(x\) is the 1-neighbor of \(u\), it follows from Algorithm \(B\) that \(y\) is the vertex \(v\). Now let \(x\) be the \(i\)-neighbor of \(u\) with \(i > 1\). It follows that \(p_{n-i-1}(x) = p_{n-i-1}(u)\), \((x_{2i+1}, x_{2i+2}, y_{2i+1}, y_{2i+2}) \in R\) for all \(0 < i < \lfloor \frac{k}{2} \rfloor\), and \(x_{2i} = \bar{u_i}\) for \(i\) even and \(x_{2i+1} = u_i - u_i \bar{u_i}\) for \(i\) odd. It follows from Algorithm \(B\) that \(p_{0}(x, y) = \bar{y}_{10}\) and \(p_{0}(v, y) = 0\), i.e., \(u_{k-1} \bar{y}_{10} = \bar{y}_{10} y_{10}\). Furthermore, using Remark 2, if \(p_{B}(x, y) = 1\) and \(1 \leq j < \lfloor \frac{k}{2} \rfloor\), it follows that \(x_{2j+1}, x_{2j+2} \neq y_{2j+1}, y_{2j+2}\) and
\[
\sum_{j=0}^{k-1} \frac{d_{j-1}}{d_{j+1}} x_{2j+1} \bar{y}_{2j+2} \neq y_{2j+1} y_{2j+2}.
\]

by Algorithm \(B\). For \(p_{B}(x, y) = 0\) and \(1 \leq j < \lfloor \frac{k}{2} \rfloor\), we have \(x_{2j+1}, x_{2j+2} \neq \bar{y}_{2j+1}, y_{2j+2}\). Therefore, given \(y_{k-2} \cdots y_{10} \neq y_{10}\), there are \(2^{k-1}\) choices of \(y\) satisfying \([u, v] \in P_B(x, y)\). Note that in ONE_STEP_ROUTE when \(p_{B}(u, v) = 2\), we route to the \((2j + 1)\)th neighbor of \(u\). When \(l\) is even, it implies \(p_{B}(y, y) = 1\). It follows that when \(l\) is even, we have
\[
|\{(x, y) \mid x \text{ is the } l\text{-neighbor of } u \text{ and } [u, v] \in P_B(x, y)\} = \begin{cases} 2^{k-1} & \text{if } l < k - 1, \\ 2^{k-1} & \text{if } l = k - 1. \end{cases}
\]

When \(l > 1\) is odd, then
\[
|\{(x, y) \mid x \text{ is the } l\text{-neighbor of } u \text{ and } [u, v] \in P_B(x, y)\} = \begin{cases} 2^{k-1} & \text{if } p_{B}(x, y) = 2, \\ 2^{k-1} & \text{if } p_{B}(x, y) = 1. \end{cases}
\]

Therefore, for even \(k\), we have
\[
|\{(x, y) \mid [u, v] \in P_B(x, y) \text{ and } [u, v] = [x^1, x^0]\}| = 1 + \sum_{i=1}^{\lfloor \frac{k}{2} \rfloor} (2^{k-i-2} + 2^{k-i-2}) = 1 + \sum_{i=1}^{\lfloor \frac{k}{2} \rfloor} 2^{k-1} = 2^{k-2}.
\]

Similarly, for odd \(k\), we have
\[
|\{(x, y) \mid [u, v] \in P_B(x, y) \text{ and } [u, v] = [x^1, x^0]\}| = 1 + \sum_{i=1}^{\lfloor \frac{k}{2} \rfloor} (2^{k-i} + 2^{k-i}) = 1 + \sum_{i=1}^{\lfloor \frac{k}{2} \rfloor} 2^{k-1} = 2^{k-2}.
\]

Assume that \([u, v] \in P_B(x, y)\). Again, \([u, v] \in \{[x^0, x^1]\} \) or \([x^1, x^0]\). If \(x^0 = u\) and \(y_{10} = 0\) or \(y_{20} = 1\), \(x^1 = v\) cannot be \(v\), given routing under Algorithm \(B\) that \(x^1 \neq u\), i.e., \([u, v] \notin P_B(x, y)\). Furthermore, \([u, v] = [x^0, x^1]\) \(P_B(x, y)\) if and only if \(y_{10} = y_{20} = 0\). Thus,
\[
|\{(x, y) \mid [u, v] \in P_B(x, y) \text{ and } x = u\}| = 2^{k-2}.
\]

Sapphire that \([u, v] = [x^1, x^0]\). Using similar arguments for the case \([u, v] = [x^0, x^1]\), we have
\[
|\{(x, y) \mid [u, v] \in P_B(x, y) \text{ and } x \text{ is a neighbor of } u\}| = 2^{k-2}.
\]

Thus, \(c_{B}(e) = 2^{k-2} + 2^{k-2} + 2^{k-2} + 2^{k-2} = 2^k\). \(\square\)

Let \(Q(x, y) = \{x = x^0, x^1, \ldots, x^m = y\}\) be a path in \(CQ_k\), \(k > 2\), that traverses from \(x\) to \(y\). Based on \(Q(x, y)\), we define a new path \(Q(p_{k-2}(x), p_{k-2}(y))\) in \(CQ_k\) by replacing all \(x^0\) in \(Q\) with \(p_{k-2}(x^0)\) and deleting \(p_{k-2}(x^{1+})\) if
\[
p_{k-2}(x^{1+}) = p_{k-2}(x^0).
\]
Definition 8. We say that \((x, y)\) and \((x', y')\) are in an equivalence class, denoted by \([x, y] \equiv [x', y']\), if and only if \(P_0(x, y) = P_0(x', y')\).

Let \((u, v)\) be a dim-1 edge in \(CQ_8\) where \(l \geq 2\). Given an edge \((u, v) \in E(CQ_8)\), let

\[
\chi(u, v) = \{(x, y) \mid [u, v] \in P_0(x, y) \text{ or } [u, v] \in P_0(x, y)\}
\]

We can partition each set \(\chi[u, v]\) into different equivalence classes. It is obvious that each equivalence class contains at most 16 elements since \(x', y', z', \beta' \in \{0, 1\}\). However, in the following lemma we show that each equivalence class contains exactly four elements.

Lemma 5. Let \((u, v)\) be a dim-d, \(d \geq 2\), edge of \(CQ_n\). Each equivalence class in \(\chi[u, v]\) contains exactly four elements.

We leave the proof of Lemma 5 to Appendix B.

Lemma 6. Let \(e\) be an edge of dimension \(d\) in \(CQ_n\), for \(n \geq 2\). Then,

\[
e_{\delta}(e) = \begin{cases} 2^{d-1} + 2^d & \text{if } d \text{ is odd,} \\ 2^d & \text{if } d \text{ is even.} \end{cases}
\]

Proof. The proof is by induction on \(n\). The theorem is true for \(n = 2, 3\), as illustrated in Fig. 3. Suppose that this theorem holds for \(CQ_{n-2}\). It follows from Lemmas 3 and 4 that the statement is also true for \(d = 0, 1\). Let \(e = \{(u, v)\}\) be an edge of \(CQ_n\), where \(n \geq 2\). Obviously, \((p_{n-2}, p_{n-2}(u), p_{n-2}(v))\) is an edge of dimension \((d - 2)\) in \(CQ_{n-2}\). By induction,

\[
e_{\delta}(p_{n-2}, p_{n-2}(u), p_{n-2}(v)) = \begin{cases} 2^{d-3} + 2^d & \text{if } d \text{ is odd,} \\ 2^{d-2} & \text{if } d \text{ is even.} \end{cases}
\]

Since \(P_0(x, y) = P_0(p_{n-2}(x), p_{n-2}(y))\), it follows from Lemma 5 that we have

\[
e_{\delta}(u, v) = \begin{cases} 2^{n-1} + 2^d & \text{if } d \text{ is odd,} \\ 2^n & \text{if } d \text{ is even.} \end{cases}
\]

Hence, the lemma follows.

Lemma 6 immediately yields the following corollary.

Corollary 2. \(e_{\alpha}(CQ_n) \leq 2^n\).

We use a similar proof technique of finding \(e_{\alpha}(CQ_n)\) to find \(e_{\alpha}(CQ_n)\), which is stated in the following theorem.

Theorem 5. \(e_{\alpha}(CQ_n) = 2^{n-1}\).

Proof. Note that \(CQ_n\) is constructed from two identical \((n - 1)\)-dimensional crossed cubes \(CQ_{n-1}\) and \(CQ_{n-1}\), which are connected by \(\text{dim}(n - 1)\) edges of \(CQ_n\). Since these \(\text{dim}(n - 1)\) edges form a perfect matching and removal of these edges disconnects \(CQ_n\), it follows that \(\omega(CQ_n) \leq 2^{n-1}\).

We define an embedding of a directed complete graph of \(2^n\) vertices, denoted by \(K_n\), into \(CQ_n\), where each edge from \(u\) to \(v\) in \(K_n\) is embedded by \(P_0(u, v)\) in \(CQ_n\).

Suppose \(\omega(CQ_n) \geq 2^{n-1}\). It follows that \(CQ_n\) can be partitioned into two subgraphs of equal size by removing a cut of \(n\) edges. This cut of \(CQ_n\) also induces a bisection of \(K_n\). Since each edge of \(CQ_n\) is contained in at most \(2^n\) shortest paths following from Lemma 6, it follows that \(\omega(K) \leq 2^n < 2^{n-1}\), which is contradictory to the known fact that \(\omega(K) \geq 2^{n-1}\). We will use the following lemma (originally stated in [4]) to find a lower bound for \(e_{\alpha}(CQ_n)\).

Lemma 7. Let \(G\) be an \(n\)-vertex graph such that the removal of a cut of \(k\) edges partitions \(G\) into two, not necessarily connected, subgraphs of \(t\) vertices and \(n - t\) vertices, respectively. Then,

\[
e_{\alpha}(G) \geq \frac{1}{k}(n^2 - 1)\alpha(G).
\]

Moreover, this bound is tight.

In particular, we can choose a cut that bisects \(G\) in (1), which can be restated as follows:

\[
e_{\alpha}(G) \geq \begin{cases} \frac{n^2}{2\alpha(G)} & \text{for } n \text{ even,} \\ \frac{(n^2 - 1)}{2\alpha(G)} & \text{for } n \text{ odd.} \end{cases}
\]

Theorem 6. \(e_{\alpha}(CQ_n) = 2^n\).

Proof. It follows from (2) and Theorem 5 that \(e_{\alpha}(CQ_n) \geq 2^n\). On the other hand, Corollary 2 states \(e_{\alpha}(CQ_n) \leq 2^n\). Thus the theorem follows.

Since our proposed shortest routing Algorithm \(B\) achieves \(e_{\alpha}(CQ_n)\), Algorithm \(B\) is an optimal routing algorithm. Theorem 6 implies that \(e_{\alpha}(CQ_n) = e_{\alpha}(Q_n)\). Furthermore, since hypercubes have an optimal routing algorithm which generates uniform congestion of all edges, Theorem 6 implies that each edge of crossed cubes has smaller congestion than or equal congestion to that of hypercubes.

6 Embedding of Cycles

A cycle is often used as a connection structure for local area networks, and can also be used as a control/data flow structure for distributed computation in arbitrary networks. In this section, we present embedding of cycles into \(CQ_n\) that our embedding of cycles is different from the one proposed in [9]. A cycle of length \(k\) is denoted by \(Q_k\). Let \(u, v\) be binary strings of length \(n - 2\) satisfying \((u, v) \in E(CQ_{n-2})\). In order to construct large cycles in \(CQ_n\), we define two types of primitive paths from \(u00\) or \(u10\) to \(v00\) as follows:

Type 1: Four primitive paths from \(u00\) to \(v00\) given by

\[
P_{1,1} = \{(u00, u01, u10, v01, v00)\},
P_{1,2} = \{(u00, u10, v01, u11, v00)\},
P_{1,3} = \{(u00, u10, u11, v01, v00)\},
P_{1,4} = \{(u00, u01, v01, u11, v10, v00)\},
\]

as shown in Fig. 4a;

Type 2: Four primitive paths from \(u10\) to \(v00\) given by

\[
P_{2,1} = \{(u10, u11, v00, v01)\},
P_{2,2} = \{(u10, v10, v11, v01, v00)\},
P_{2,3} = \{(u10, u11, v00, v01, v00)\},
P_{2,4} = \{(u10, u10, v11, v01, v00)\},
\]

as shown in Fig. 4b.
as shown in Fig. 4b. Each $P_{i,j}$ has the length of $j + 3$ for $i = 1, 2$.

**Theorem 7.** $CQ_n$ is a pancyclic network for $n \geq 2$, i.e., $CQ_n$ contains $C_l$ for all $4 \leq l \leq 2^n$ as subgraphs.

**Proof.** The theorem can be easily proven for $n = 2, 3$ by using type-1 primitive paths. For $n = 4$, since $CQ_4$ is a subgraph of $CQ_4$, it suffices to construct cycles $C_l$ for $9 \leq l \leq 16$ as follows:

$$
C_9 = \langle 0000, 0001, 0011, 0100, 0110, 1110, 1111, 0101, 0100, 0000 \rangle,
$$

$$
C_{10} = \langle 0000, 0001, 0011, 0100, 0110, 1110, 1111, 1101, 1100, 0000 \rangle,
$$

$$
C_{11} = \langle 0000, 0001, 0011, 0100, 0110, 1110, 1111, 1101, 1100, 0000 \rangle,
$$

$$
C_{12} = \langle 0000, 0001, 0011, 0100, 0110, 1110, 1111, 1101, 1100, 0000 \rangle,
$$

$$
C_{13} = \langle 0000, 0001, 0011, 0100, 0110, 1110, 1111, 1101, 1100, 0000 \rangle,
$$

$$
C_{14} = \langle 0000, 0001, 0011, 0100, 0110, 1110, 1111, 1101, 1100, 0000 \rangle,
$$

$$
C_{15} = \langle 0000, 0001, 0011, 0100, 0110, 1110, 1111, 1101, 1100, 0000 \rangle,
$$

$$
C_{16} = \langle 0000, 0001, 0011, 0100, 0110, 1110, 1111, 1101, 1100, 1000, 0000 \rangle.
$$

We next prove the theorem for $n \geq 5$ by induction. Since $CQ_{n-1}$ is a subgraph of $CQ_n$, it suffices to construct cycles of length $2^{n-1} + 1 \leq l \leq 2^n$. Let $s$ be an arbitrary integer satisfying $2^{n-3} + 1 \leq s \leq 2^{n-2}$. By the induction hypothesis, there is a cycle $C_i$ in $CQ_{n-2}$ given by

$$
C_i = \langle u^i, \cdots, u^i, u^i \rangle.
$$

When $s$ is even, construct $C_j$ based on $C_i$ as follows:

$$
C_j = \langle u^i, u^i, u^i, u^i, u^i, u^i, u^i, u^i, u^i, u^i, u^i, u^i, u^i, u^i, u^i, u^i, u^i, u^i, u^i, u^i, u^i \rangle,
$$

where $P'$ is a type-1 primitive path from $u^{i-10}$ to $u^0$. It follows that $j = 4(s - 2) + |P'| + 1$ and to be precise, $j = 4s - 3, 4s - 2, 4s - 1, 4s$. When $s$ is odd, we use $C_i$ to construct $C_j$ similar to $C_i$ as follows:

$$
C_j = \langle u^i, u^i, u^i, u^i, u^i, u^i, u^i, u^i, u^i, u^i, u^i, u^i, u^i, u^i, u^i, u^i, u^i, u^i, u^i, u^i, u^i \rangle,
$$

where $P''$ is a type-2 primitive path from $u^{i-11}$ to $u^0$. It follows that $j$ can be $4s - 3, 4s - 2, 4s - 1, 4s$. Therefore, we have obtained cycles $C_i$ such that

$$
4(2^{n-3} + 1) - 3 \leq l \leq 4(2^{n-2}),
$$

i.e., $2^{n-1} + 1 \leq l \leq 2^n$. Hence, the theorem follows.
A

Based on the above proof idea, we can easily construct a cycle of arbitrary length. We here illustrate an example of constructing $C_{12}$ in $CQ_5$. First, we find a cycle of length $s = \lceil \frac{19}{2} \rceil = 10$ in $CQ_5$, which is given by

$$C_6 = (000, 001, 011, 101, 100, 000).$$

Since $s$ is odd and $19 = 4s - 1$, we can use $C_6$ and a $P_{2,3}$ from $10110$ to $10000$ to construct a $C_{12}$ as follows:

$$C_{12} = (00000, 00001, 00010, 00101, 00111, 01101, 01110, 10101, 10111, 10001, 11001, 11101, 11110, 11111, 00000).$$

Embedding cycles presented in [9] can construct one type of cycles for an arbitrary length, while ours can construct various types of cycles since we can start with any vertex and make modification of type-1 and type-2 primitive paths to construct cycles. For example, to construct $C_{9}$ in $CQ_5$, we define a different cycle of length 5, which is given by

$$C_9 = (011, 010, 000, 130, 101, 011).$$

We can use $C_9$ and a path $P$ from 10011 to 10101 of length six to construct $C_{12}$ as follows:

$$C_{12} = (01111, 01110, 01100, 01101, 01011, 01020, 01000, 00101, 00100, 00010, 00001, 01011, 10000, 10010, 10100, 10101, 10111, 01111).$$

7 Concluding Remarks

Though topological properties of crossed cubes have been studied in the literature, we introduce a new measure in this paper called pair related distance. Using this measure, we can easily find the shortest distance between any pair of vertices. Furthermore, we use this measure to give a shortest path routing algorithm in $O(n^2)$ time rather than $O(n^3)$ time in comparison with previous work. In this paper, we also define a new measure performance called edge congestion. Given the shortest path routing algorithm presented in this paper, we show that the edge congestion of crossed cubes is $2^n$, equal to that of hypercubes. Bisection width of crossed cubes is $2^{n-1}$. We also prove that the wide diameter and the fault diameter of crossed cubes are approximately half of those of hypercubes. Furthermore, crossed cubes are shown to be pancyclic networks with more types of cycles constructed.

Appendix A

Proof of Lemma 3. We prove this lemma by induction on $n$.

Obviously, the lemma is true for $n = 2$. Assume that such disjoint paths exist for any two distinct vertices in $CQ_n$ for $l < n$ and $n \geq 3$. Let $u = u_{n-1}u_{n-2} \cdots u_1u_0$ and $v = v_{n-1}v_{n-2} \cdots v_1v_0$ be two vertices in $CQ_n$.

We first consider $n$ odd and distinguish the following cases.

Case 1.1. $d_{CQ_n}(p_1(u), p_1(v)) < 1$ or $u_{n-1} = v_{n-1}$.

It follows that $u, v$ belong to a subgraph $G_1$ which is isomorphic to $CQ_{n-1}$. By the induction hypothesis, there are $n - 1$ disjoint paths $P_1, P_2, \cdots, P_{n-1}$ joining $u$ to $v$ such that $|P_i| \leq \frac{|u_n|}{2} + 1$ for all $1 \leq n - 1$ and $|P_i| \leq |u_n|$. There exist two binary strings $x' = x_{n-1}x_{n-2} \cdots x_1x_0$ and $y' = y_{n-1}y_{n-2} \cdots y_1y_0$ such that

1. $(p_2(u), x') \in E(CQ_n)$ and $(p_3(v), y') \in E(CQ_n)$,
2. $x' \neq p_2(u)$ and $y' \neq p_3(v)$,
3. $x'_{n-3} = u_{n-3}, y'_{n-3} = y_{n-3}$ if $u_{n-1} = v_{n-1}$,
4. $x'_{n-3} \neq u_{n-3}, y'_{n-3} \neq y_{n-3}$ if $d_{CQ_n}(p_2(u), p_3(v)) \leq 1$,
5. $x' = y'$ if $p_2(u) = p_3(v)$.

Fig. 5a illustrates an example of choice for $x'$ and $y'$. Those figures in Fig. 5 illustrate an example of choice of specified vertices whose positions are not unique.

Let $u'$ and $v'$ be neighbors of $u$ and $v$ given by $u' = f(u, p_2(u), x')$ and $v' = f(v, p_3(v), y')$, where in particular $u' = x'$ and $v' = y'$. Since $u'$ and $v'$ are chosen from $G_2 = CQ_n - G_1$, which is also isomorphic to $CQ_{n-1}$. Let $S_1$ be a shortest path in $G_2$ joining $u'$ and $v'$ and satisfying $|S_1| < |u_n|$. Construct a path $P_n$ as

$$P_n = (u, u', s_1, v', v),$$

which is disjoint from $P_1, P_2, \cdots, P_{n-1}$. Thus

$$(P_1, P_2, \cdots, P_n)$$

satisfy Conditions 1, 2, and 3.

Case 1.2. $d_{CQ_n}(p_2(u), p_3(v)) = 2$ and $u_{n-1} \neq v_{n-1}$.

Since $p_{n-2}(u)$ and $p_{n-2}(v)$ are two different vertices in $CQ_{n-2}$, by the induction hypothesis there are $n - 2$ disjoint paths $P_1', P_2', \cdots, P_{n-2}'$ joining $p_{n-2}(u)$ and $p_{n-2}(v)$ such that $|P_i'| \leq \frac{|u_n|}{2} + 1$ for all $i \leq n - 2$. We use these $n - 2$ paths to construct $P_n'$ as follows:

$$P_n' = P(u; P_1') = (u, u', u'' \cdots u^{(k)}; v) = w_1.$$ (3)

Note that all of $P_i'$ are disjoint paths in $CQ_n$ and that $|P_i'| = |P_i|$ for all $1 \leq i \leq n - 2$. Furthermore, since

$$p_n(2(u)) = p_{n-2}(v),$$

it follows that $w_1$ and $v$ are in a subgraph isomorphic to $CQ_2$. However, $P_n'$ are not necessarily $(u, v)$-paths.

We make the following modification of $P_n'$ for $2 \leq i \leq n - 2$ in order to obtain $n - 3$ $(u, v)$-paths $P_i$. If $w_i = v$, we simply let $P_i = P_i'$. Now consider $w_i \neq v$. Let $z_i$ be the immediate predecessor of $w_i$, and let $z_i'$ be the immediate predecessor of $z_i$ on $P_i'$. Let $v'$ be a neighbor of $v$ in $CQ_n(p_{n-2}(z_i))$ and let $v''$ be a neighbor of $v'$ in $CQ_n(p_{n-2}(z_i'))$. Since $CQ_n(p_{n-2}(z_i))$ and $CQ_n(p_{n-2}(z_i'))$ are adjacent subgraphs,

$$CQ_n(p_{n-2}(z_i), p_{n-2}(w_i)) \cong CQ_n(p_{n-2}(z_i), p_{n-2}(w_i'))$$

is isomorphic to $CQ_2$. If $v'$ is also a neighbor of $z_i$ (as illustrated in Fig. 5b), we construct
Consider that all of $w_1, w_{n-1},$ and $w_n$ are distinct from $v$, say, $d_{CQ_1}(w_1, v) = d_{CQ_1}(w_{n-1}, v) = 1$ and $d_{CQ_1}(w_n, v) = 2$.

We construct
\[ P_1 = (u, P^*_1, v), \quad \text{and} \quad P_{n-1} = (u, u', S_{n-1}, w_{n-1}, v), \]
where $|P_1| = |P^*_1| + 1$ and $|P_{n-1}| = |S_{n-1}| + 2$. Since
\[ CQ_1(p_{n-2}(z_n), p_{n-2}(w_n)) \]
is isomorphic to $CQ_2$, there exists a neighbor $v'$ of $z_n$ in $CQ_1(p_{n-2}(z_n))$ such that $v'$ is the common neighbor of $v$ and $z_n$ (as illustrated in Fig. 5d). It follows that $d_{CQ_2}(z_n, v') = 2$. We construct
\[ P_n = (u, u'_k) \cup (S_n \setminus \{z_n, w_n\}) \cup \{z_n, v', v\}, \]
where $|P_n| = |S_n| + 2$.

Now consider that one of $w_1, w_{n-1},$ and $w_n$ is exactly $v$, say, $w_1 = v$, $d_{CQ_1}(w_{n-1}, v) = 1$, and $d_{CQ_1}(w_n, v) = 2$. We simply let $P_1 = P^*_1$ and construct $P_{n-1}$ as above. There exists a common neighbor $v'$ of $w_n$ and $v$ in $CQ_1(p_{n-2}(w_n))$ with $v' \neq w_{n-1}$ (as illustrated in Fig. 5e). We construct...
we construct $P_n = \langle u, w^0, s_n, w_n, v', v \rangle$, where $|P_n| = |S_n| + 3$. The other cases of relative positions of $w_1, w_0, u$, and $v$ can be similarly treated.

Now we have constructed $n$ $(u, v)$-paths satisfying Conditions 1, 2, and 3. To prove disjointness of these paths, let $x$ be any internal vertex of $P_i$ with $2 \leq i \leq n - 2$ and let $y$ be any internal vertex of $P_i, P_{i+1}$, and $P_n$. Clearly,

$p_n \circ x(x) \neq p_{n-2}\circ y(y)$.

Moreover, any internal vertex of $P_i, P_{i+1}$, and $P_n$ differs from each other at the last two bits. Thus, we have constructed $n$ disjoint paths $P_1, P_2, \ldots, P_n$ which satisfy Conditions 1, 2, and 3.

Next, consider $n$ is even. We distinguish the following cases.

Case 2.1. $d_{CQ_n}(p_2(u), p_2(v)) \leq 1$.

The proof is similar to that of Case 1.1.

Case 2.2. $d_{CQ_n}(p_2(u), p_2(v)) = 2$.

Let $P_i$ be defined as in Case 1.2, and let $P_2$ be defined as in (3) for $1 \leq i \leq n - 2$. The first $n - 4$ $(u, v)$-paths are constructed by $P_i$ for all $3 \leq i \leq n - 2$ as in Case 1.2, which satisfy Conditions 1, 2, and 3.

In the following discussion, we use $P_i$ and $P_2$ to construct the remaining four $(u, v)$-paths $P_1, P_2, P_{i+1}, P_n$. Let $u'$ be defined as in Case 1.2. If $|P_i| \leq D(CQ_n, a_2) - 1$, then $P_1, P_{i-1}, P_n$, and $P_2$ are constructed in the same way as in Case 1.2. Now, consider $|P_i| = D(CQ_n, a_2)$. It follows from Condition 3 that $|P_i| = |P_2| = D(CQ_n, a_2)$. We define $S_{i-1} = P(u'; P_i)$, $S_i = P(u', P_2)$, $w_{i-1} = t(u'; S_{i-1})$, and $w_0 = t(u'; S_0)$. Note that $|S_{i-1}| = |S_i| = |P_i| = |P_2|$. We note that $u_1$ and $u_2$ are in $CQ_2$. Furthermore, $P_i$ and $P_2$ start from the same vertex and $|P_i| = |P_2|$. It follows that $u_1 = u_2$. Moreover, $w_{i-1} \neq w_0$ and $p_{i-2}(w_{i-1}) = p_{i-2}(w_0) = p_{i-2}(w_0) = p_{i-2}(u_0)$.

Therefore, all of $w_1, w_0, w_{i-1}, v_0$, and $v_0$ are in a subgraph isomorphic to $CQ_2$. Let $z_i$ be the immediate predecessor of $w_i$ and $z_i$ be the immediate predecessor of $z_i$ on $P_i$ for $i = 1, 2$. Let $z_i$ be the immediate predecessor of $w_i$ and $z_i$ be the immediate predecessor of $z_i$ on $S_i$ for $i = 1, n - 1$. Let $u'$ and $v'$ be neighbors of $v$ in $CQ_n(p_{i-2}(z_i))$ and $CQ_n(p_{i-2}(z_i))$, respectively.

Suppose that all of $w_1, w_0, w_{i-1}, v_0$, and $v_0$ are distinct from $u'$ and $v'$, any $d_{CQ_n} (w, v) = d_{CQ_n} (w, v) = d_{CQ_n} (w, v) = 1$ and $d_{CQ_n} (w, v) = 2$. We construct $P_n$ as in (4) and $P_3 = \langle u, P_i, P_2, v \rangle$, where $|P_3| = |P_1| + 1$. Since $CQ_n(p_{i-2}(z_i), P_{i-2}(w_1))$ is isomorphic to $CQ_2$, using the same arguments as in Case 1.2, we construct $P_n$ as given by (5). Since $CQ_n(P_{i-2}(z_i), P_{i-2}(w_1))$ and $CQ_n(P_2(w_1))$ are adjacent subgraphs, $CQ_n(P_{i-2}(z_i), P_{i-2}(w_1))$ is isomorphic to $CQ_n$. If $z'$ is also a neighbor of $z_i$ (as shown in Fig. 6a), we construct

$P_1 = \langle P_i, (z_1, w_1) \rangle \cup (z_1, z', v)\),

where $|P_1| = |P_1| + 1$. Otherwise, there exists a common neighbor of $z'$ and $z'$, denoted by $z''$ in $CQ_n(p_{i-2}(z_i))$ (as shown in Fig. 6b) and we construct

$P_1 = \langle P_i, (z_1, w_1) \rangle \cup (z_1, z', v, w),$

where $|P_1| = |P_1| + 1$.

Next, suppose some of $w_1, w_0, w_{i-1}, v_0$ are exactly $u$, say, $w_1 = w_0 = v$, $d_{CQ_n}(w_{i-1}, v) = 1$, and $d_{CQ_n}(w_0, v) = 2$. We simply let $P_1 = P_1$ and construct $P_{i-1}$ and $P_n$ as above. In particular, we have $v' = z_0$ in $P_n$. Since $CQ_n(p_{i-2}(w_2))$ is isomorphic to $CQ_2$, there exists a neighbor $w'$ of $v$ in $CQ_n(p_{i-2}(w_2))$ satisfying $w' \neq w_0$. Since $CQ_n(p_{i-2}(z_1), p_{i-2}(w_2))$ and $CQ_n(p_{i-2}(z_1), p_{i-2}(w_0))$ are isomorphic to $CQ_2$, there exists a neighbor $w'$ of $w'$ in $CQ_n(p_{i-2}(w_2))$ with $w'' \neq z_0$ and $z_1$ in $CQ_n(p_{i-2}(w_2))$ with $w' \neq z_1$ (as illustrated in Fig. 6c). We construct

$P_3 = \langle P_2, (z_2, z_0, v) \rangle \cup (z_2, u_0, w', w', v),$

where $|P_3| = |P_2| + 2$. The other cases of relative positions of $w_1, w_0, w_{i-1}, v_0$, and $v$ can be similarly treated.

Thus, we have obtained $n$ $(u, v)$-paths satisfying Conditions 1, 2, and 3. Let $x$ be any internal vertex of $P_1$ with $3 \leq i \leq n - 2$ and $y$ be any internal vertex of $P_1, P_2, P_{i-1}$, and $P_n$. Clearly, $p_{i-2}(x) \neq p_{i-2}(y)$. Moreover, any internal vertex of $P_1$ differs from those of $P_{i-1}$ at the last two bits and differs from those of $P_i, P_n$ at the first $n - 2$ bits. Similarly, any internal vertex of $P_2$ differs from those of $P_n$ at the last two bits. Thus, $P_1, P_2, \ldots, P_n$ are disjoint paths and satisfy Conditions 1, 2, and 3. Thus, the proof is completed.

APPENDIX B

Proof of Lemma 5. Let $x = x_{n-1} \cdots x_0 x_0$ and $y = y_{n-1} \cdots y_0 y_0$ be two vertices in $CQ_n$ where $(x, y) \in \chi[u, v]$. Let

$P_{y}(x, y) = (x - x_0^0, x_1^1, \ldots, x_{n-1} - y)$

and

$P_{y}(x, y) = (p_{i-2}(x) - x_0^0, x_1^1, \ldots, x_{n-2}(y) = y)$.

We can assume without loss of generality that $P_{y}(x, y)$ is the shortest among all of $P_{y}(x, y)$, where each $(x, y)$ is in the same equivalence class with $(x, y)$, i.e. $(x', y') \in \chi[u, v]$ and $(x, y) \equiv (x', y')$. It follows that $p_{i-2}(x) = 0$, since otherwise we can obtain another pair of vertices in this equivalence class having shorter path. Since $(u, v)$ is in $P_{y}(x, y)$ and is also a circ-d, $d \geq 2$, edge of $CQ_2$, it follows that

$(p_{i-2}(u), p_{i-2}(v)) = [x_0^0, x_{i-1}^1]$.
or $(p_{i-2}(v), p_{i-2}(u)) = [z_i^i, z_{i+1}^i]$ for some $i \geq 0$. We first assume that $(p_{i-2}(u), p_{i-2}(v)) = [z_i^i, z_{i+1}^i]$ for $i \geq 0$. In the following discussion, we abbreviate "$z_i^i z_{i+1}^i \sim y_i y_{i+1}^i$ in $(x', y')$" by simply "$z_i^i z_{i+1}^i \sim y_i y_{i+1}^i$" without ambiguity from the context. Let $y' = p_{i-2}(u) y_i y_{i+1}^i$, where $y_i y_{i+1}^i \notin \chi[u, v]$; and thus, $x_0 x_1 \neq y_0 y_1^i$.

**Case 1:** $(p_{i-2}(u), p_{i-2}(v)) = [z_i^i, z_{i+1}^i]$. Since $P_B(x, y)$ is the shortest in the equivalence class, it follows that $x = u = z_i^i$.

Let $x' = z_i^i x_0$ or $z_i^i x_1 x_0$. It follows that

$$P_B(x', y) = P_B(x, y).$$

Therefore, $(u, v) \in P_B(x', y)$ and $\bar{P}_B(x, y) = \bar{P}_B(x', y)$, i.e., $(z_i x_1 x_0, y) \notin \chi(x, y)$ and $(z_i x_1 x_0, y) \notin \chi(x, y)$. Since $\rho_B(z_i x_1 x_0, y) = 1$ and the vertices $z_i x_1 x_0$ and $z_i$ are not adjacent, $P_B(z_i x_1 x_0, y)$ does not pass through $(u, v)$, i.e.,

$$(z_i x_1 x_0, y) \notin \chi[u, v].$$

Thus, $(z_i x_1 x_0, y) \notin \chi(x, y)$.

Since $\rho_B(x, y) = 0$ implies $\rho_B(x, y') = 1$, we consider the following three possibilities: First, let $x_1 x_0 \sim_{y_i} y_0 y_1^i$. Then $P_B(x, y')$ and $P_B(z_i x_1 x_0, y')$ will first change the zeroh double bit. It follows that $[x, \bar{v}] \notin P_B(x, y')$ and $[x, \bar{v}] \notin P_B(z_i x_1 x_0, y')$, i.e., $(x, y') \notin \chi[u, v]$ and $(z_i x_1 x_0, y') \notin \chi[u, v]$. Furthermore,

$$P_B(z_i x_1 x_0, y')$$

will not route through $(u, v)$ since $x_1 x_0 \neq x_0 x_1$. Therefore, $(z_i x_1 x_0, y') \notin \chi[u, v]$. Since $x_1 x_0 \sim_{y_i} y_0 y_1^i$ implies $x_1 x_0 \neq y_1 y_0^i$ and $z_i x_1 x_0 \neq y_1 y_0^i$, then $P_B(z_i x_1 x_0, y')$ will first route to a $d$ neighbor of $z_i x_1 x_0$ and then change the zeroth double bit. It follows that $(u, v) \notin P_B(z_i x_1 x_0, y')$, i.e., $(z_i x_1 x_0, y') \notin \chi[u, v]$. Thus when $x_1 x_0 \sim_{y_i} y_0 y_1^i$, none of $(x, y')$, $(z_i x_1 x_0, y')$, $(z_i x_1 x_0, y')$, and $(z_i x_1 x_0, y')$ is in $\chi[u, v]$. 

---

Fig. 6. Illustration in Appendix A for examples of relative positions of some specific vertices in $C_{Q_{BR}}$.
Next, let $x_1 \tilde{x}_0 \sim^P y_1 \tilde{y}_0$. Similarly, we can also show that none of $(x, y'), (\tilde{x}_2 z_0, y'), (\tilde{x}_2 z_0, y')$, and $(\tilde{x}_0 z_1, \tilde{y}_0, y)$ is in $\chi[u, v]$.

Finally, let $\tilde{x}_0 z_1 \sim^P \tilde{y}_1 \tilde{y}_0$. We can also show that

$$(\tilde{x}_0 z_1, \tilde{y}_1 \tilde{y}_0) \notin \chi[u, v],$$

$$(\tilde{x}_2 z_0, \tilde{y}_0) \notin \chi[u, v],$$

and $(\tilde{x}_0 z_1, \tilde{y}_0, y) \notin \chi[u, v]$. Since $\tilde{x}_0 z_1 \sim^P \tilde{y}_1 \tilde{y}_0$, then $P_B(x, y')$ will route through $(x, v)$ and then change the zeroth double bit. Let $L$ be a path in $CQ_{2-1}$ given by $L = \tilde{P}_B(u, y')$. To be specific,

$$P_B(x, y') = (x = u, v, p_k(x, y'), v_0, p_k(x, y'), v_0; L).$$

It follows that $(x, y') \notin \chi[u, v]$ and $P_B(x, y') = \tilde{P}_B(x, y')$. Therefore, $(x, y') \equiv (x, y)$ given $y'/y' \tilde{x}_2 z_0$. Thus, we obtain exactly four elements in the same equivalence class, i.e., $(x, y), (\tilde{x}_0 z_1, \tilde{y}_0, y), (\tilde{x}_2 z_0, y'), y' \tilde{y}_0$. We note that $(x, y)$-pairs discussed in Case 1 and then changing

$$y' \tilde{y}_0 \sim^P \tilde{x}_2 z_0.$$

Since $P_B(x, y')$ is the shortest in the equivalence class, it follows that $x' = u'$ for some $x'$. $P_B(x, u') = 0$ and $P_B(u', y') = 0$.

Let $x' = \tilde{x}_0 z_1$ or $\tilde{x}_0 z_1$. It follows that

$$P_B(x', y') = (x', y', P_B(x, y')).$$

Therefore, $[u, v] \notin P_B(x', y')$ and $P_B(x', y') = \tilde{P}_B(x, y')$, i.e., $(\tilde{x}_0 z_1, \tilde{y}_0) \equiv (x, y)$ and $(\tilde{x}_0 z_1, \tilde{y}_0) \equiv (x, y)$. Let

$$y' \tilde{y}_0 \sim^P \tilde{x}_2 z_0$$

and let $L$ be a subpath of $P_B(x, y')$ given by

$$L = (x_1, \ldots, x_m = y).$$

Then, $(x, x')$ is an edge, it follows that $(x_1 z_0, x_1 x_0) \in R$ and moreover, $(x_1 z_0, x_1 x_0) \in R$. Therefore, we obtain

$$P_B(x', y') = (x', \tilde{x}_0 z_1, x_1 x_0, x_1, L).$$

It follows that

$$(\tilde{x}_0 z_1, \tilde{y}_0) \notin \chi[u, v]$$

and $P_B(x, y') = P_B(x, y')$, i.e., $(\tilde{x}_0 z_1, \tilde{y}_0) \equiv (x, y)$. Therefore, $(\tilde{x}_0 z_1, \tilde{y}_0) \equiv (x, y)$ for all $x', y'$.

Since $P_B(x, y') = 0$ implies $P_B(x, y') = 1$, there are three possibilities: $\tilde{x}_0 z_0 \tilde{y}_1 \tilde{y}_0$, $x_0 \tilde{x}_0 \tilde{y}_1 \tilde{y}_0$, or $x_0 \tilde{x}_0 \tilde{y}_1 \tilde{y}_0$. For $\tilde{x}_0 z_0 \tilde{y}_1 \tilde{y}_0$ or $x_0 \tilde{x}_0 \tilde{y}_1 \tilde{y}_0$, the vertex $u$ is not on $P_B(x, y')$ since we can first change the zeroth double bit and then, $y_0 \tilde{y}_0$. For $x_0 \tilde{x}_0 \tilde{y}_1 \tilde{y}_0$, since $(x, x')$ is a $dist-p$ edge for $p > 2$, it follows that $x_0 \tilde{x}_0 \tilde{y}_1 \tilde{y}_0$. Then $P_B(x, y')$ is obtained by traversing from $z$ to $x'$ and then changing the zeroth double bit of $x'$. Therefore, $P_B(x, y')$ will not route through $[u, v]$ or $(x, y') \equiv (x, y)$ for all $y' \neq y$.

Since $P_B(x, u') = 1$ and $P_B(u', y') = 1$ for $x' \neq x$ and $y' \neq y$, it follows that $P_B(x, y')$ will not route through vertex $u$ and therefore, $(x, y') \notin \chi[u, v]$ for all $x' \neq x$ and $y' \neq y$. Hence, the equivalence class in $\chi[u, v]$ containing $(x, y)$ has exactly four elements $(\tilde{x}_0 z_0, y')$, where $x_1, y_0 = 0, 1$.

We note that $(x, y)$-pairs discussed in Case 1 are different from those in Case 2 since $P_B(x, y')$ in Case 1 is not equal to $P_B(x, y')$ in Case 2. Therefore, these two cases define different equivalence classes. We can show, using similar arguments, that $\chi[u, v]$ containing $(x, y)$ with $(p_k(x, y'), p_k(x, y')) = [\tilde{x}_1, \tilde{y}_1]$, for $i \geq 0$ defines different equivalence classes from others. Thus, each equivalence class in $\chi[u, v]$ contains exactly four elements, and the lemma is proven.

[1]

ACKNOWLEDGMENTS

This work was supported in part by the National Science Council of the Republic of China under contract NSC86-2213-E-003-026.

REFERENCES


Chien-Ping Chang received a BS degree in electrical engineering from Chung-Cheng Institute of Technology in 1986, and a PhD degree in computer and information science from National Chiao Tung University, Taiwan, Republic of China, in 1995. He is currently a senior engineer at Chung Shan Institute of Science and Technology, Taiwan, Republic of China. His research interests include interconnection networks, graph algorithms, and medical images.

Ting-Yi Sung received a BS degree in management science from National Chiao Tung University, Taiwan, Republic of China, in 1980, and a PhD degree in operations research from New York University in 1989. She is currently an associate research fellow at the Institute of Information Science, Academia Sinica, Taiwan, Republic of China. Her research interests include fault tolerance and architectures for interconnection networks, graph algorithms, and mathematical programming. She is a member of the IEEE.

Lih-Hsing Hsu received his BS degree in mathematics from Chung Yuan Christian University, Taiwan, Republic of China, in 1975, and his PhD degree in mathematics from the State University of New York at Stony Brook in 1981. He is currently a professor in the Department of Computer and Information Science, National Chiao Tung University, Taiwan, Republic of China. His research interests include interconnection networks, algorithms, graph theory, and VLSI layout.