A Simple Parallel Algorithm for Constructing Independent Spanning Trees on Twisted Cubes*

Ting-Jyun Yang[1] Jinn-Shyong Yang[2] Jou-Ming Chang[1,†] An-Hang Chen[1]

[1] Institute of Information and Decision Sciences,
National Taipei College of Business, Taipei, Taiwan, ROC

[2] Department of Information Management,
National Taipei College of Business, Taipei, Taiwan, ROC

Abstract

In 1989, Zehavi and Itai [46] proposed the following conjecture: a k-connected graph G must possess k independent spanning trees (ISTs for short) with an arbitrary node as the common root. An n-dimensional twisted cube, denoted by TQ_n, is a variation of hypercubes with connectivity n to achieving some improvements of structure properties. Recently, Yang [42] proposed an algorithm for constructing n edge-disjoint spanning trees in TQ_n for any odd integer $n \geqslant 3$. Moreover, he showed that half of them are ISTs. At a later stage, Wang et al. [32] confirm the ISTs conjecture by providing an $\mathcal{O}(N \log N)$ algorithm to construct n ISTs rooted at an arbitrary node on TQ_n, where $N = 2^n$ is the number of nodes in TQ_n. However, this algorithm is executed in a recursive fashion and thus are hard to be parallelized. In this paper, we present a non-recursive and fully parallelized approach to construct n ISTs rooted at an arbitrary node of TQ_n in $\mathcal{O}(\log N)$ time using N processors. In particular, the constructing rule of spanning trees is simple and the proof of independency is easier than ever before.

Keyword: independent spanning trees; interconnection networks; twisted cubes;

1 Introduction

A set of spanning trees in a graph G is said to be *independent* (ISTs for short) if all the trees are rooted at the same node r such that, for any other node $v(\neq r)$ in G, the paths from v to r in any two trees are internally node-disjoint (i.e., there exists no common node in the two paths except the two end nodes v and r). Constructing multiple spanning trees in networks have been studied from not only the theoretical point of view but also some practical applications such as fault-tolerant broadcasting [2, 19] and secure message distribution [2, 29, 35].

For a graph G, its vertex set and edge set are denoted by $V(G)$ and $E(G)$, respectively. If F is a subset of $V(G)$, we denote $G - F$ as the graph obtained from G by removing F. A graph G is k-*connected* if $|V(G)| > k$ and $G - F$ is connected for every subset $F \subseteq V(G)$ with $|F| < k$. Zehavi and Itai [46] proposed the following conjecture: If r is an arbitrary node of a k-connected graph G, then G possess k ISTs rooted at r. Till now, this conjecture has been shown to be true for k-connected graphs with $k \leqslant 4$ (see [19], [9, 46] and [10] for $k = 2, 3, 4$, respectively), but it is still open for $k \geqslant 5$. In particular, this conjecture has been confirmed for several restricted classes of graphs, e.g., graphs related to planarity [17, 18, 26, 27], graphs defined by Cartesian product [4, 28, 30, 31, 34, 37, 41], variations of hypercubes [5–8, 25, 32, 33, 35], special Cayley graphs [21, 22, 29, 36, 39, 40], and others [20, 38].

The family of twisted cubes was first introduced by Hilbers et al. [15] as a variation of hypercubes. Although Abraham and Padmanabhan [1] pointed out asymmetry of twisted cubes, it has been shown that twisted cubes possess some improvements of structure properties in contrast to hypercubes. For instance, Chang et al. [3] showed that the diameter, wide diameter, and faulty diameter of n-dimensional twisted cube, denoted by TQ_n, are about half of those of the n-dimensional hypercube. More research results on TQ_n can be found in the literature, e.g., the studies of Hamiltonian properties [16, 45], path embedding [11, 13], cycle embedding [12], mesh and torus embedding [23, 24], and

*This research was partially supported by National Science Council under the Grants NSC102-2221-E-141-002 and NSC102-2221-E-141-001-MY3.

†Corresponding author. Email: spade@mail.ntcb.edu.tw

fault-tolerant embedding [13,14,43,44]. In particular, Yang [42] proposed an algorithm for constructing n edge-disjoint spanning trees in TQ_n for any odd integer $n \geqslant 3$ and showed that half of them are ISTs. Since it is a fact stated in [3] that TQ_n has the connectivity n, Wang et al. [32] proposed an algorithm to construct n ISTs rooted at an arbitrary node in $\mathcal{O}(N \log N)$ time for TQ_n, where $N = 2^n$ is the number of nodes in TQ_n. However, this algorithm is executed in a recursive fashion and thus are hard to be parallelized. In this paper, we present a non-recursive and fully parallelized approach for constructing n ISTs rooted at an arbitrary node in TQ_n.

The rest of this paper is organized as follows. Section 2 formally gives the definition of twisted cubes and provides some useful terminologies and notations. Section 3 presents our algorithm for constructing n ISTs in TQ_n. Section 4 proves the correctness of the algorithm. The final section contains our concluding remarks.

2 Preliminary

Let $\mathbb{Z}_n = \{0, 1, \ldots, n-1\}$. For a binary string $x = x_{n-1}x_{n-2}\cdots x_0$ and $i \in \mathbb{Z}_n$, we define $\dddot{\oplus}(x, i) = x_i \oplus x_{i-1} \oplus \cdots \oplus x_0$, where $\oplus$ is the exclusive operation. The n-dimensional twisted cube, denoted by TQ_n, is a variant of the n-dimensional hypercube with 2^n node in which each node is labeled by a unique binary string of length n. Originally, it can be recursively defined as follow.

Definition 1. [15] The 1-dimensional twisted cube TQ_1 is defined to be the complete graph with two nodes labeled by 0 and 1. For an odd integer $n \geqslant 3$, TQ_n consists of four subcubes TQ^{00}_{n-2}, TQ^{01}_{n-2}, TQ^{10}_{n-2}, and TQ^{11}_{n-2}, where TQ^{ab}_{n-2} for $a, b \in \mathbb{Z}_2$ is isomorphic to TQ_{n-2} such that $V(TQ^{ab}_{n-2}) = \{abx \colon x \in V(TQ_{n-2})\}$ (i.e., adding two preceding bits a and b in the front of a node labeled by x) and $E(TQ^{ab}_{n-2}) = \{(abx, aby) \colon (x, y) \in E(TQ_{n-2})\}$. That is, $V(TQ_n) = \cup_{ab\in\mathbb{Z}_2} V(TQ^{ab}_{n-2})$. Define $E(TQ_n) = \cup_{ab\in\mathbb{Z}_2} E(TQ^{ab}_{n-2}) \cup E'$, where an edge $(u, v) \in E'$ if and only if the two nodes $u = u_{n-1}u_{n-2}\cdots u_0$ and $v = v_{n-1}v_{n-2}\cdots v_0$ satisfy one of the following conditions:

(1) $u = \bar{v}_{n-1}v_{n-2}\cdots v_0$;
(2) $u = \bar{v}_{n-1}\bar{v}_{n-2}v_{n-3}\cdots v_0$ for $\dddot{\oplus}(u, n-3) = 0$;
(3) $u = v_{n-1}\bar{v}_{n-2}v_{n-3}\cdots v_0$ for $\dddot{\oplus}(u, n-3) = 1$.

Note that Definition 1 can only be applied for odd integer n. Figure 1 depicts twisted cubes TQ_3 and TQ_5, respectively. Recently, Wang et al. [32] showed that Definition 1 can be further extended to any integer $n \geqslant 1$ by considering two types of TQ_n for even integer n as follows:

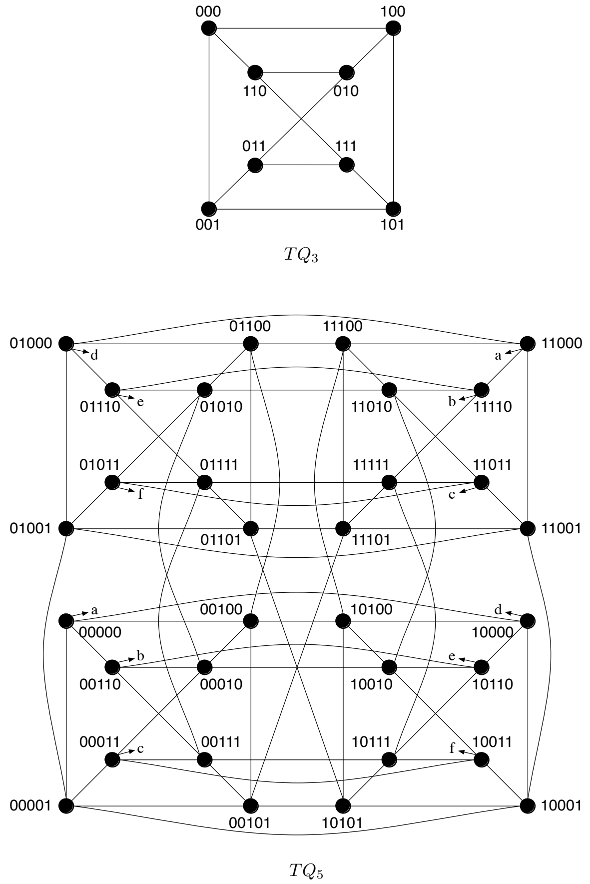

Figure 1: Twisted cubes TQ_3 and TQ_5.

Definition 2. [32] For an even integer $n \geqslant 2$, the n-dimensional twisted cube TQ_n is divided into two types: 0-type TQ_n and 1-type TQ_n, where the former is denoted by TQ^0_n and the latter is denoted by TQ^1_n. For any integer $b \in \mathbb{Z}_2$, $V(TQ^b_n) = \{ibx \colon i \in \mathbb{Z}_2 \text{ and } x \in V(TQ_{n-1})\}$ and $E(TQ^b_n) = \cup_{i\in\mathbb{Z}_2}\{(ibx, iby) \colon (x, y) \in E(TQ_{n-1})\} \cup E'$, where an edge $(u, v) \in E'$ if and only if the two nodes $u = u_{n-1}u_{n-2}\cdots u_0$ and $v = v_{n-1}v_{n-2}\cdots v_0$ satisfy $u = \bar{v}_{n-1}v_{n-2}\cdots v_0$.

Figure 2 illustrates the two types of twisted cubes TQ^0_4 and TQ^1_4, respectively. In the rest of this paper, we say TQ_n to mean either 0-type TQ_n or 1-type TQ_n if n is even and there is no ambiguity. Also, for notational convenience, a node $x \in V(TQ_n)$ is denoted by $x = (x_n)x_{n-1}x_{n-2}\cdots x_0$, where the first bit x_n enclosed by a pair of round brackets indicates that we can omit it if n is odd. According to Definitions 1 and 2, twisted cubes can be equivalently defined by the following non-recursive fashion:

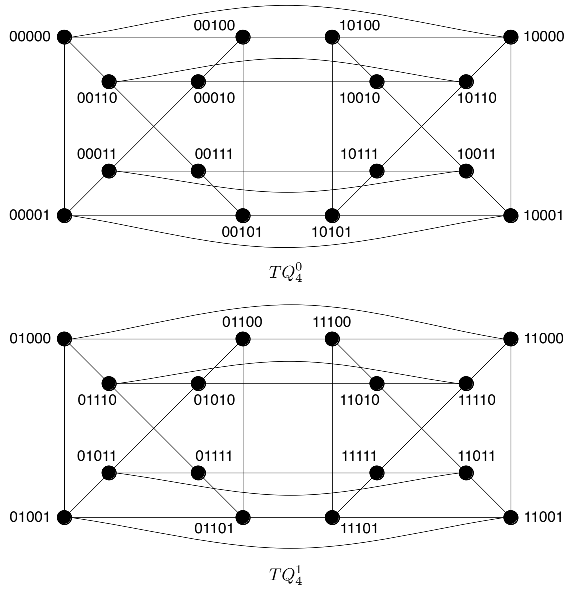

Figure 2: Two types of twisted cubes TQ_4^0 and TQ_4^1.

Definition 3. Let $n \geqslant 1$ be any integer. For $i \in \mathbb{Z}_n$ and a node $x \in V(TQ_n)$, define $N_i(x)$ as the ith dimensional adjacent vertex (or the i-neighbor) of x in TQ_n as follows:

(1) For even i,
$N_i(x) = (x_n)x_{n-1}x_{n-2}\cdots x_{i+1}\bar{x}_i x_{i-1}\cdots x_0$.

(2) For odd i,

(a) if n is even and $i = n-1$, then
$N_i(x) = \bar{x}_n x_{n-1}x_{n-2}\cdots x_0$;

(b) if n is odd or $n(\neq i+1)$ is even, then

(i) if $\dddot{\oplus}(x, i-1) = 0$ then $N_i(x) = (x_n)x_{n-1}\cdots x_{i+2}\bar{x}_{i+1}\bar{x}_i x_{i-1}\cdots x_0$;

(ii) if $\dddot{\oplus}(x, i-1) = 1$ then $N_i(x) = (x_n)x_{n-1}\cdots x_{i+1}\bar{x}_i x_{i-1}\cdots x_0$.

For example, if we consider the node $x = 01100$ in TQ_4^1 (respectively, in TQ_5), then $N_0(x) = 01101$, $N_1(x) = 01010$, $N_2(x) = 01000$, $N_3(x) = 11100$ (respectively, $N_3(x) = 00100$ and $N_4(x) = 11100$).

Throughout this paper, we also use the following notation. Two paths P and Q joining two distinct nodes x and y are *internally node-disjoint*, denoted by $P||Q$, if $V(P) \cap V(Q) = \{x, y\}$. Let T be a spanning tree rooted at node r of TQ_n. The parent of a node $x(\neq r)$ in T is denoted by $\textsc{parent}(T, x)$. For $x, y \in V(T)$, the unique path from x to y is denoted by $T[x, y]$. Hence, two spanning trees T and T' with the same root r are ISTs if and only if $T[x, r] \,||\, T'[x, r]$ for every node $x \in V(T) \setminus \{r\}$.

3 Parallel construction of ISTs on twisted cubes

Due to the fact that TQ_n is n-connected, we would like to construct n ISTs, which implies that the root in each spanning tree must have a unique child. We choose a node $r \in V(TQ_n)$ as the root arbitrarily. For $i \in \mathbb{Z}_n$, we denote T_i as a tree such that r takes its i-neighbor as the unique child. Let $N_i(r) = (c_n)c_{n-1}c_{n-2}\cdots c_0$. For each node $x = (x_n)x_{n-1}x_{n-2}\cdots x_0 \in V(TQ_n) \setminus \{r\}$ and $i \in \mathbb{Z}_n$, we define $I_i(x) = \{j \in \mathbb{Z}_n \colon x_j \neq c_j\}$. We say that x is *replaceable* in T_i if the following conditions are fulfilled:

$$i(\neq 0) \text{ is even, } x_{i-1} \neq c_{i-1} \text{ and } \dddot{\oplus}(r, i-2) = 0.$$

Otherwise, x is *irreplaceable*. Moreover, if x is replaceable, we let $H_i(x) = I_i(x) \oplus \{i\}$; otherwise, let $H_i(x) = I_i(x)$. Also, we define the following function:

$$\textsc{next}(i, x) = \begin{cases} i & \text{if } H_i(x) = \emptyset; \\ \min H_i(x) & \text{if } H_i(x) \neq \emptyset \text{ and } i > \max H_i(x); \\ \min\{j \in H_i(x) \colon j \geqslant i\} & \text{otherwise.} \end{cases}$$

That is, we regard $H_i(x)$ as a cyclic ordered set in increasing order. If $H_i(x) = \emptyset$ or $i \in H_i(x)$, the function outputs i; otherwise, the function outputs the next element in the cyclic order of $H_i(x)$ with respect to i.

It is clear that, for each node $x \in V(TQ_n) \setminus \{r\}$, finding $I_i(x)$, $H_i(x)$, $\textsc{next}(i, x)$ and determining whether x is placeable or not can be done in $\mathcal{O}(n)$ time provided i is given. In Figure 3, we present a fully parallelized algorithm for constructing n spanning trees with an arbitrary node $r = (r_n)r_{n-1}r_{n-2}\cdots r_0$ as their common root in TQ_n. For each node $x = (x_n)x_{n-1}x_{n-2}\cdots x_0 \in V(TQ_n) \setminus \{r\}$, the construction can be carried out by describing the parent of x in each spanning tree T_i.

Example 1. We describe how the algorithm constructs T_i in TQ_5 for $i = 2$. Suppose that we choose $r = 10110_2 = 22$ as the common root in the spanning trees. Clearly, the 2-neighbor of r is $N_2(22) = 10010_2 = 18$ and $\dddot{\oplus}(r, 2-2) = 0$. We first consider a node $x = 11000_2 = 24$. Clearly, $I_2(x) = \{1, 3\}$. Since $x_1 \neq c_1$, x is replaceable, and thus $H_2(x) = \{1, 2, 3\}$ and $j = \textsc{next}(2, x) = 2$. It follows that $\textsc{parent}(T_2, x) = N_2(x) = 11100_2 = 28$. Let $y = 28$. Clearly, $I_2(y) = \{1, 2, 3\}$. Since $y_1 \neq c_1$, y is replaceable, and thus $H_2(y) = \{1, 3\}$ and $j = \textsc{next}(2, y) = 3$. It follows that $\textsc{parent}(T_2, y) = N_3(y) = 10100_2 = 20$. Let

Algorithm CONSTRUCTING-ISTs
Input: All nodes of TQ_n and the common root $r = (r_n)r_{n-1}r_{n-2}\cdots r_0$.
Output: n ISTs $T_0, T_1, \ldots, T_{n-1}$ root at r.

for $i = 0$ to $n-1$ **do in parallel** /* construct T_i simultaneously */
  for each node x in TQ_n **do in parallel**
   /* generate parent of each node x simultaneously */
   $j = \text{NEXT}(i, x)$;
   if n is even and $j = n-1$, **then**
     $\text{PARENT}(T_i, x) = x + (-1)^{x_n} \times 2^n$;
   else if j is odd and $\ddot{\oplus}(r, j-1) = 0$, **then**
     $\text{PARENT}(T_i, x) = x + (-1)^{x_{j+1}} \times 2^{j+1} + (-1)^{x_j} \times 2^j$;
    else
     $\text{PARENT}(T_i, x) = x + (-1)^{x_j} \times 2^j$;

Figure 3: Algorithm for constructing n spanning trees in TQ_n.

$z = 20$. Clearly, $I_2(z) = \{1, 2\}$. Since $z_1 \neq c_1$, z is replaceable, and thus $H_2(z) = \{1\}$ and $j = \text{NEXT}(2, z) = 1$. It follows that $\text{PARENT}(T_2, y) = N_1(z) = 10010_2 = 18$. Let $c = 18$. Recall that c is the 2-neighbor of r. In this case, we have $I_2(c) = H_2(c) = \emptyset$ and $j = \text{NEXT}(2, c) = 2$. Thus, $\text{PARENT}(T_2, c) = N_2(c) = 10110_2 = r$.

For TQ_5, we provide all constructing results in Figure 4 and only summarize details of the construction of T_2 in Table 1. For convenience, we adopt the notation $x \xrightarrow{i} y$ to mean that $y = N_i(x)$ in TQ_n. For instance, we have $T_2[24, 22] : 24 \xrightarrow{2} 28 \xrightarrow{3} 20 \xrightarrow{1} 18 \xrightarrow{2} 22$ in Figure 4.

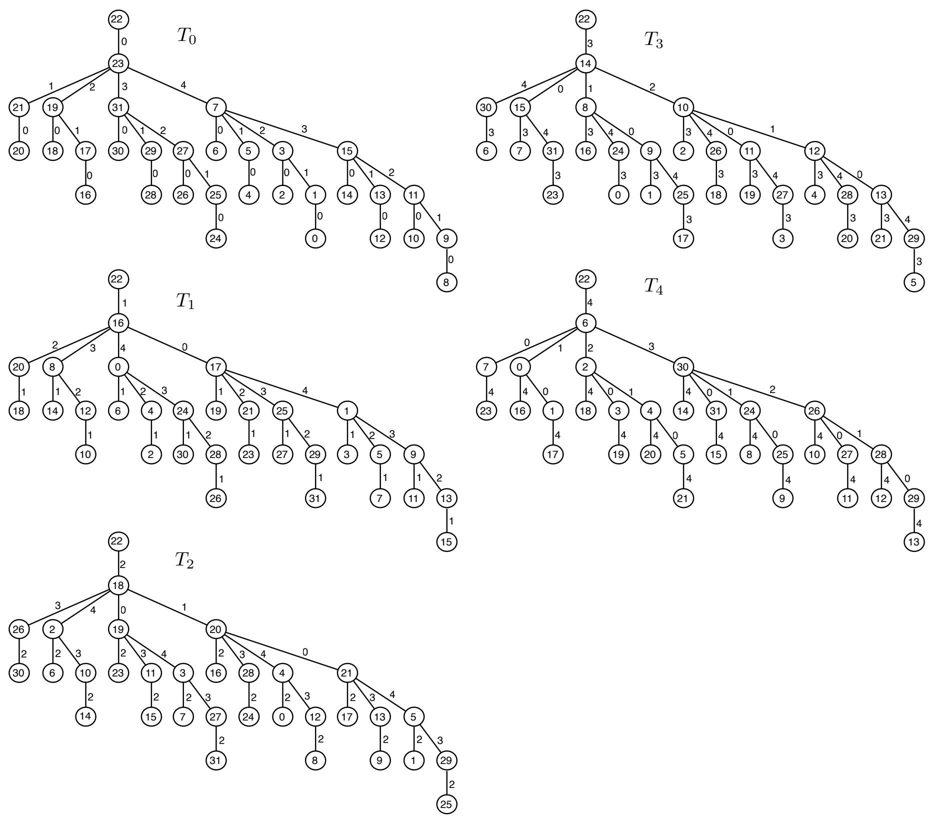

Figure 4: Five ISTs of TQ_5.

Table 1: The parent of nodes $x \in V(TQ_5) \setminus \{22\}$ in T_2 with root $r = 10110_2 = 22$.

$i = 2$, $N_2(22) = 10010_2 = 18$

x	binary string	$I_2(x)$	replaceable	$H_2(x)$	$j =$ $\text{NEXT}(2, x)$	$\ddot{\oplus}(x, j-1)$ when j is odd	$\text{PARENT}(T_2, x)$
0	00000	$\{1, 4\}$	yes	$\{1, 2, 4\}$	2	-	$= 0 + 2^2 = 4$
1	00001	$\{0, 1, 4\}$	yes	$\{0, 1, 2, 4\}$	2	-	$= 1 + 2^2 = 5$
2	00010	$\{4\}$	no	$\{4\}$	4	-	$= 2 + 2^4 = 18$
3	00011	$\{0, 4\}$	no	$\{0, 4\}$	4	-	$= 3 + 2^4 = 19$
4	00100	$\{1, 2, 4\}$	yes	$\{1, 4\}$	4	-	$= 4 + 2^4 = 20$
5	00101	$\{0, 1, 2, 4\}$	yes	$\{0, 1, 4\}$	4	-	$= 5 + 2^4 = 21$
6	00110	$\{2, 4\}$	no	$\{2, 4\}$	2	-	$= 6 - 2^2 = 2$
7	00111	$\{0, 2, 4\}$	no	$\{0, 2, 4\}$	2	-	$= 7 - 2^2 = 3$
8	01000	$\{1, 3, 4\}$	yes	$\{1, 2, 3, 4\}$	2	-	$= 8 + 2^2 = 12$
9	01001	$\{0, 1, 3, 4\}$	yes	$\{0, 1, 2, 3, 4\}$	2	-	$= 9 + 2^2 = 13$
10	01010	$\{3, 4\}$	no	$\{3, 4\}$	3	1	$= 10 - 2^3 = 2$
11	01011	$\{0, 3, 4\}$	no	$\{0, 3, 4\}$	3	0	$= 11 + 2^4 - 2^3 = 19$
12	01100	$\{1, 2, 3, 4\}$	yes	$\{1, 3, 4\}$	3	1	$= 12 - 2^3 = 4$
13	01101	$\{0, 1, 2, 3, 4\}$	yes	$\{0, 1, 3, 4\}$	3	0	$= 13 + 2^4 - 2^3 = 21$
14	01110	$\{2, 3, 4\}$	no	$\{2, 3, 4\}$	2	-	$= 14 - 2^2 = 10$
15	01111	$\{0, 2, 3, 4\}$	no	$\{0, 2, 3, 4\}$	2	-	$= 15 - 2^2 = 11$
16	10000	$\{1\}$	yes	$\{1, 2\}$	2	-	$= 16 + 2^2 = 20$
17	10001	$\{0, 1\}$	yes	$\{0, 1, 2\}$	2	-	$= 17 + 2^2 = 21$
18	10010	$\emptyset$	no	$\emptyset$	2	-	$= 18 + 2^2 = 22$
19	10011	$\{0\}$	no	$\{0\}$	0	-	$= 19 - 2^0 = 18$
20	10100	$\{1, 2\}$	yes	$\{1\}$	1	0	$= 20 - 2^2 + 2^1 = 18$
21	10101	$\{0, 1, 2\}$	yes	$\{0, 1\}$	0	-	$= 21 - 2^0 = 20$
22	10110	-	-	-	-	-	(root)
23	10111	$\{0, 2\}$	no	$\{0, 2\}$	2	-	$= 23 - 2^2 = 19$
24	11000	$\{1, 3\}$	yes	$\{1, 2, 3\}$	2	-	$= 24 + 2^2 = 28$
25	11001	$\{0, 1, 3\}$	yes	$\{0, 1, 2, 3\}$	2	-	$= 25 + 2^2 = 29$
26	11010	$\{3\}$	no	$\{3\}$	3	1	$= 26 - 2^3 = 18$
27	11011	$\{0, 3\}$	no	$\{0, 3\}$	3	0	$= 27 - 2^4 - 2^3 = 3$
28	11100	$\{1, 2, 3\}$	yes	$\{1, 3\}$	3	1	$= 28 - 2^3 = 20$
29	11101	$\{0, 1, 2, 3\}$	yes	$\{0, 1, 3\}$	3	0	$= 29 - 2^4 - 2^3 = 5$
30	11110	$\{2, 3\}$	no	$\{2, 3\}$	2	-	$= 30 - 2^2 = 26$
31	11111	$\{0, 2, 3\}$	no	$\{0, 2, 3\}$	2	-	$= 31 - 2^2 = 27$

Example 2. To demonstrate that our algorithm can also be applied on TQ_n for even integer n, we provide partial results of TQ_4. Table 2 shows the construction of T_3 in TQ_4^0 and Table 3 shows the construction of T_2 in TQ_4^1. For TQ_4^0, we let $r = 10011_2 = 19$ be the root and the 3-neighbor of r is $N_3(19) = 00011_2 = 3$. In this case, since $i = 3$ is odd, x is irreplaceable, and thus $H_i(x) = I_i(x)$ for every node $x \in V(TQ_4^0)$. For TQ_4^1, we let $r = 01110_2 = 14$ be the root and the 2-neighbor of r is $N_2(14) = 01010_2 = 10$. In this case, since $i(\neq 0)$ is even and $\ddot{\oplus}(r, 2-2) = 0$, if $x_1 \neq c_1$ for a node $x \in V(TQ_4^1)$ then $H_i(x) = I_i(x) \oplus \{i\}$; otherwise, $H_i(x) = I_i(x)$. As a result, according to the function $\text{NEXT}(i, x)$, we can determine the parent of x for every node $x \in V(TQ_4)$.

4 Correctness

In this section, we will show the validity of the algorithm. Firstly, we prove the reachability between every node $x(\neq r)$ and the root r in T_i, thereby proving the existence of a unique path from x to the root in the tree.

Lemma 1. *Let $r \in V(TQ_n)$ be an arbitrary node. The constructions of T_i for all $i \in \mathbb{Z}_n$ are spanning trees rooted at r.*

Proof. From CONSTRUCTING-ISTS, since every node $v \in V(TQ_n)$ must be contained in T_i, it follows that T_i is a spanning subgraph of TQ_n. Suppose that $r = (r_n)r_{n-1}r_{n-2}\cdots r_0$ and $N_i(r) = (c_n)c_{n-1}c_{n-2}\cdots c_0$. Let $x = (x_n)x_{n-1}x_{n-2}\cdots x_0$ be any node of TQ_n. In the following, we show that $T_i[x, r]$ is the unique path connecting x and r in T_i. We first consider $I_i(x) = \emptyset$. In this case, $x_j = c_j$ for $j \in \mathbb{Z}_n$. Thus, $x = c = N_i(r)$. In particular, if $i \neq 0$ then $x_{i-1} = c_{i-1}$. Thus, x is irreplaceable. It follows that $H_i(x) = I_i(x) = \emptyset$ and $\text{NEXT}(i, x) = i$. If n is even and $i = n-1$, by Line 5 of the algorithm, we have $\text{PARENT}(T_i, x) = x + (-1)^{x_n} \times 2^n = N_{n-1}(x) = r$. On the other hand (i.e., n is odd or $i \neq n-1$), by Line 7 and Line 9 of the algorithm, we have either $\text{PARENT}(T_i, x) = x + (-1)^{x_{i+1}} \times 2^{i+1} + (-1)^{x_i} \times 2^i = N_i(x) = r$ or $\text{PARENT}(T_i, x) = x + (-1)^{x_i} \times 2^i = N_i(x) = r$. This shows that $T_i[x, r] : x \xrightarrow{i} r$ is the desired path connecting x and r in T_i.

Next, we suppose that $I_i(x) = \{j_0, j_1, \ldots, j_{p-1}\}$ is nonempty and is treated as an ordered set such that $j_0 < j_1 < \cdots < j_{p-1}$. Clearly, $1 \leqslant p \leqslant n$. By

Table 2: The parent of nodes $x \in V(TQ_4^0) \setminus \{19\}$ in T_3 with root $r = 10011_2 = 19$.

$i = 3,\quad N_3(19) = 00011_2 = 3$

x	binary string	$I_3(x)$	replaceable	$H_3(x)$	$j =$ NEXT$(3, x)$	$\ddot{\oplus}(x, j-1)$ when j is odd	PARENT(T_3, x)
0	00000	$\{0,1\}$	no	$\{0,1\}$	0	-	$= 0 + 2^0 = 1$
1	00001	$\{1\}$	no	$\{1\}$	1	1	$= 1 + 2^1 = 3$
2	00010	$\{0\}$	no	$\{0\}$	0	-	$= 2 + 2^0 = 3$
3	00011	$\emptyset$	no	$\emptyset$	3	-	$= 3 + 2^4 = 19$
4	00100	$\{0,1,2\}$	no	$\{0,1,2\}$	0	-	$= 4 + 2^0 = 5$
5	00101	$\{1,2\}$	no	$\{1,2\}$	1	1	$= 5 + 2^1 = 7$
6	00110	$\{0,2\}$	no	$\{0,2\}$	0	-	$= 6 + 2^0 = 7$
7	00111	$\{2\}$	no	$\{2\}$	2	-	$= 7 - 2^2 = 3$
16	10000	$\{0,1,4\}$	no	$\{0,1,4\}$	4	-	$= 16 - 2^4 = 0$
17	10001	$\{1,4\}$	no	$\{1,4\}$	4	-	$= 17 - 2^4 = 1$
18	10010	$\{0,4\}$	no	$\{0,4\}$	4	-	$= 18 - 2^4 = 2$
19	10011	-	-	-	-	-	(root)
20	10100	$\{0,1,2,4\}$	no	$\{0,1,2,4\}$	4	-	$= 20 - 2^4 = 4$
21	10101	$\{1,2,4\}$	no	$\{1,2,4\}$	4	-	$= 21 - 2^4 = 5$
22	10110	$\{0,2,4\}$	no	$\{0,2,4\}$	4	-	$= 22 - 2^4 = 6$
23	10111	$\{2,4\}$	no	$\{2,4\}$	4	-	$= 23 - 2^4 = 7$

Table 3: The parent of nodes $x \in V(TQ_4^1) \setminus \{14\}$ in T_2 with root $r = 01110_2 = 14$.

$i = 2,\quad N_2(14) = 01010_2 = 10$

x	binary string	$I_2(x)$	replaceable	$H_2(x)$	$j =$ NEXT$(2, x)$	$\ddot{\oplus}(x, j-1)$ when j is odd	PARENT(T_2, x)
8	01000	$\{1\}$	yes	$\{1,2\}$	2	-	$= 8 + 2^2 = 12$
9	01001	$\{0,1\}$	yes	$\{0,1,2\}$	2	-	$= 9 + 2^2 = 13$
10	01010	$\emptyset$	no	$\emptyset$	2	-	$= 10 + 2^2 = 14$
11	01011	$\{0\}$	no	$\{0\}$	0	-	$= 11 - 2^0 = 10$
12	01100	$\{1,2\}$	yes	$\{1\}$	1	0	$= 12 - 2^2 + 2^1 = 10$
13	01101	$\{0,1,2\}$	yes	$\{0,1\}$	0	-	$= 13 - 2^0 = 12$
14	01110	-	-	-	-	-	(root)
15	01111	$\{0,2\}$	no	$\{0,2\}$	2	-	$= 15 - 2^2 = 11$
24	11000	$\{1,4\}$	yes	$\{1,2,4\}$	2	-	$= 24 + 2^2 = 28$
25	11001	$\{0,1,4\}$	yes	$\{0,1,2,4\}$	2	-	$= 25 + 2^2 = 29$
26	11010	$\{4\}$	no	$\{4\}$	4	-	$= 26 - 2^4 = 10$
27	11011	$\{0,4\}$	no	$\{0,4\}$	4	-	$= 27 - 2^4 = 11$
28	11100	$\{1,2,4\}$	yes	$\{1,4\}$	4	-	$= 28 - 2^4 = 12$
29	11101	$\{0,1,2,4\}$	yes	$\{0,1,4\}$	4	-	$= 29 - 2^4 = 13$
30	11110	$\{2,4\}$	no	$\{2,4\}$	2	-	$= 30 - 2^2 = 26$
31	11111	$\{0,2,4\}$	no	$\{0,2,4\}$	2	-	$= 31 - 2^2 = 27$

definition, $x_j \neq c_j$ for $j \in I_i(x)$ and $x_j = c_j$ for $j \in \mathbb{Z}_n \setminus I_i(x)$. There are two scenarios as follows:

Case 1: $i \in I_i(x)$ and x is replaceable (see, e.g. node $x = 12$ in Table 1) or $i \notin I_i(x)$ and x is irreplaceable (see, e.g. node $x = 11$ in Table 1). Clearly, $H_i(x) = I_i(x) \setminus \{i\}$ for the former, and $H_i(x) = I_i(x)$ for the latter. Let $j_k =$ NEXT(i, x) where $0 \leqslant k \leqslant p-1$. Clearly, $j_k \neq i$. Assume that $y(\neq r) = (y_n)y_{n-1}y_{n-2}\cdots y_0$ is the parent of x in T_i. That is, $y =$ PARENT$(T_i, x) = N_{j_k}(x)$. Consider the following two subcases:

Case 1.1: j_k is odd and $\ddot{\oplus}(r, j_k - 1) = 0$. By Line 7 of the algorithm, we have $y = x + (-1)^{x_{j_k+1}} \times 2^{j_k+1} + (-1)^{x_{j_k}} \times 2^{j_k}$ (i.e., $y_{j_k+1} = \bar{x}_{j_k+1}$ and $y_{j_k} = \bar{x}_{j_k} = c_{j_k}$). In this case, if $y_{j_k+1} = c_{j_k+1}$, then $I_i(y) = I_i(x) \setminus \{j_k, j_k+1\}$ (see, e.g. node $x = 11$ and $y = 19$ in Table 1); otherwise, $I_i(y) = (I_i(x) \cup \{j_k+1\}) \setminus \{j_k\}$ (see, e.g. node $x = 27$ and $y = 3$ in Table 1).

Case 1.2: j_k is even or $\ddot{\oplus}(r, j_k - 1) = 1$. By Line 9 of the algorithm, we have $y = x + (-1)^{x_{j_k}} \times 2^{j_k}$ (i.e., $y_{j_k} = \bar{x}_{j_k} = c_{j_k}$). In this case, we have $I_i(y) = I_i(x) \setminus \{j_k\}$ (see, e.g. node $x = 12$ and $y = 4$ in Table 1).

From above, we can determine $I_i(y)$. In particular, we show that $j_k \notin I_i(y)$. Since $j_k \neq i$ and only the elements of $I_i(y)$ and i can be included in $H_i(y)$, it implies that $j_k \notin H_i(y)$. By a similar argument, if $I_i(y) \neq \emptyset$, let $z =$ PARENT$(T_i, y) = N_{j_\ell}(y)$ be the parent of y in T_i, where $j_\ell =$ NEXT(i, y). Again, we can determine $I_i(z)$ and show that $j_k, j_\ell \notin H_i(z)$. By this way, we find a sequence of nodes $y, z, \ldots, c$ in T_i such that $I_i(c) = \emptyset$, and thus $c = N_i(r)$. Recall that we have already constructed $T_i[c, r] = c \xrightarrow{i} r$ for connecting $N_i(r)$ and r in T_i. Therefore, we obtain the following unique path that connects x and r in T_i:

$$T_i[x,r] : x \xrightarrow{j_k} y \xrightarrow{j_\ell} z \xrightarrow{j_m} \cdots \xrightarrow{j_q} c \xrightarrow{i} r.$$

Case 2: $i \in I_i(x)$ and x is irreplaceable (see, e.g. node $x = 7$ in Table 1) or $i \notin I_i(x)$ and x is replaceable (see, e.g. node $x = 8$ in Table 1). In this case, we have $\text{NEXT}(i,x) = i$. Let $y = \text{PARENT}(T_i, x) = N_i(x)$. If i is odd and $\ddot{\oplus}(r, i-1) = 0$, by Line 7 of the algorithm, we have $y = x + (-1)^{x_{i+1}} \times 2^{i+1} + (-1)^{x_i} \times 2^i$ (i.e., $y_{i+1} = \bar{x}_{i+1}$ and $y_i = \bar{x}_i = c_i$). Moreover, if $y_{i+1} = c_{i+1}$, then $I_i(y) = I_i(x) \setminus \{i, i+1\}$; otherwise, $I_i(y) = (I_i(x) \cup \{i+1\}) \setminus \{i\}$. On the other hand (i.e., i is even or $\ddot{\oplus}(r, i-1) = 1$), by Line 9 of the algorithm, we have $y = x + (-1)^{x_i} \times 2^i$ (i.e., $y_i = \bar{x}_i = c_i$). Thus, $I_i(y) = I_i(x) \setminus \{i\}$. This shows that the current status of y is in the situation of Case 1. Let $P = T_i[y, r]$ be the path connecting y and r in T_i. Therefore, we obtain the unique path $T_i[x, r]$ by concatenating $x \xrightarrow{i} y$ and P. □

According to the proof of Lemma 1, we have the following properties.

Corollary 2. *For $i \in \mathbb{Z}_n$, let $T_i[x,r] : v_0(= x) \xrightarrow{j_1} v_1 \xrightarrow{j_2} \cdots \xrightarrow{j_k} v_k \xrightarrow{i} r$ be a path constructed from Lemma 1. Then, the following statements hold:*

(2) *For $1 \leqslant \ell < m \leqslant k$, $j_\ell \notin H_i(v_m)$ (i.e., $j_\ell \neq j_m$).*

(3) *For $2 \leqslant \ell \leqslant k$, $j_\ell \neq i$. In particular, it is possible $j_1 = i$.*

For instance, if we consider the path $T_2[25, 22] : 25 \xrightarrow{2} 29 \xrightarrow{3} 5 \xrightarrow{4} 21 \xrightarrow{0} 20 \xrightarrow{1} 18 \xrightarrow{2} 22$ in Figure 4, we can verify from Table 1 as follows: $H_2(25) = \{0,1,2,3\}$, $H_2(29) = \{0,1,3\}$, $H_2(5) = \{0,1,4\}$, $H_2(21) = \{0,1\}$, $H_2(20) = \{1\}$ and $H_2(18) = \emptyset$. Let $\text{HEIGHT}(T)$ denote the height of a tree T. Since $|I_i(x)| \leqslant n$ for every node $x \in V(MQ_n)$, the following result can be obtained from Corollary 2 directly.

Corollary 3. *For $i \in \mathbb{Z}_n$, $\text{HEIGHT}(T_i) \leqslant n+1$.*

Lemma 4. *The spanning trees constructed from* Constructing-ISTs *are independent.*

Proof. We prove the lemma by contradiction. Suppose that the lemma is false. That is, there exist two integers $i, j \in \mathbb{Z}_n$ and a node $x \in V(TQ_n) \setminus \{r\}$ such that the following two paths constructed in Lemma 1 satisfy $\{x, r\} \subsetneq P \cap Q$:

$$P = T_i[x,r] :$$
$$u_0(= x) \xrightarrow{j_0} u_1 \xrightarrow{j_1} u_2 \xrightarrow{j_2} \cdots \xrightarrow{j_{k-1}} u_k \xrightarrow{i} r$$

and

$$Q = T_j[x,r] :$$
$$v_0(= x) \xrightarrow{\ell_0} v_1 \xrightarrow{\ell_1} v_2 \xrightarrow{\ell_2} \cdots \xrightarrow{\ell_{m-1}} v_m \xrightarrow{j} r.$$

Suppose that $u_p = v_q$ for $1 \leqslant p < k$ and $1 \leqslant q < m$. Let $A = \{j_p, j_{p+1}, \ldots, j_{k-1}, i\}$ and $B = \{\ell_q, \ell_{q+1}, \ldots, \ell_{m-1}, j\}$. Since $i \neq j$, by Corollary 2 we have $A \neq B$. Let $d = \max((A \cup B) \setminus (A \cap B))$. This implies that the dth bit of u_p is different from that of v_q, which leads to a contradiction. □

According to Lemmas 1 and 4, we have the following theorem.

Theorem 5. *Let $N = 2^n$ and $r \in V(TQ_n)$ be an arbitrary node. Algorithm* Constructing-ISTs *can correctly construct n ISTs rooted at r in $\mathcal{O}(N \log N)$ time. In particular, the algorithm can be parallelized on TQ_n by using N processors to run in $\mathcal{O}(\log N)$ time.*

5 Concluding remarks

In this paper, we provide a non-recursive and fully parallelized approach for constructing n ISTs rooted at an arbitrary node of TQ_n in $\mathcal{O}(\log N)$ time, where $N = 2^n$ is the number of nodes. Indeed, all ISTs constructed in here are isomorphic to those in [32] and have height $n+1$. There are also other variants of hypercubes without node-symmetry, e.g., Möbius cubes, crossed cubes and locally twisted cube. Although some algorithms in [5, 6, 25] can simultaneously construct multiple ISTs for these variants, none of them can be fully parallelized for the construction of each spanning tree. To the best of our knowledge, for class of networks without node-symmetry, the present paper is the first to employ the fully parallelized approach for constructing ISTs.

References

[1] S. Abraham and K. Padmanabhan, The twisted cube topology for multiprocessors: a study in network asymmetry, *J. Parallel Distrib. Comput.*, 13 (1991) 104–110.

[2] F. Bao, Y. Funyu, Y. Hamada and Y. Igarashi, Reliable broadcasting and secure distributing in channel networks, in: *Proc. of 3rd International Symposium on Parallel Architectures, Algorithms and Networks*, ISPAN'97, Taipei, December 1997, pp. 472–478.

[3] C.-P. Chang, J.-N. Wang, L.-H. Hsu, Topological properties of twisted cubes, *Inform. Sci.*, 113 (1999) 147–167.

[4] X.-B. Chen, Parallel construction of optimal independent spanning trees on Cartesian product of complete graphs, *Inform. Process. Lett.*, 111 (2011) 235–238.

[5] B. Cheng, J. Fan, X. Jia, and J. Jia, Parallel construction of independent spanning trees

and an application in diagnosis on Möbius cubes, *J. Supercomput.*, 65 (2013) 1279–1301.

[6] B. Cheng, J. Fan, X. Jia, and J. Wang, Dimension-adjacent trees and parallel construction of independent spanning trees on crossed cubes, *J. Parallel Distrib. Comput.*, 73 (2013) 641–652.

[7] B. Cheng, J. Fan, X. Jia, and S. Zhang, Independent spanning trees in crossed cubes, *Inform. Sci.*, 233 (2013) 276–289.

[8] B. Cheng, J. Fan, X. Jia, S. Zhang, and B. Chen, Constructive algorithm of independent spanning trees on Möbius cubes, *Comput. J.*, 56 (2013) 1347–1362.

[9] J. Cheriyan and S.N. Maheshwari, Finding nonseparating induced cycles and independent spanning trees in 3-connected graphs, *Journal of Algorithms*, 9 (1988) 507–537.

[10] S. Curran, O. Lee and X. Yu, Finding four independent trees, *SIAM Journal on Computing*, 35 (2006) 1023–1058.

[11] J. Fan, X. Jia, X. Lin, Optimal embeddings of paths with various lengths in twisted cubes, *IEEE Trans. Parallel Distrib. Syst.*, 18 (2007) 511–521.

[12] J. Fan, X. Jia, X. Lin, Embedding of cycles in twisted cubes with edge-pancyclic, *Algorithmica*, 51(2008) 264–282.

[13] J. Fan, X. Lin, Y. Pan, X. Jia, Optimal fault-tolerant embedding of paths in twisted cubes, *IEEE Trans. Parallel Distrib. Syst.*, 67 (2007) 205–214.

[14] J.-S. Fu, Fault-free Hamiltonian cycles in twisted cubes with conditional link faults, *Theoret. Comput. Sci.*, 407 (2008) 318–329.

[15] P.A.J. Hilbers, M.R.J. Koopman, J.L.A. van de Snepscheut, The twisted cube, in: *PARLE: Parallel Architectures and Languages Europe, Vol. 1: Parallel Architectures*, *Lecture Notes in Computer Science*, Vol. 258, 1987, pp. 152-159

[16] W.-T. Huang, J.-M. Tan, C.-N. Hung, L.-H. Hsu, Fault-tolerant Hamiltonicity of twisted cubes, *J. Parallel Distrib. Comput.*, 62 (2002) 591–604.

[17] A. Huck, Independent trees in graphs, *Graphs Combin.*, 10 (1994) 29–45.

[18] A. Huck, Independent trees in planar graphs, *Graphs Combin.*, 15 (1999) 29–77.

[19] A. Itai and M. Rodeh, The multi-tree approach to reliability in distributed networks, *Inform. Comput.*, 79 (1988) 43–59.

[20] Y. Iwasaki, Y. Kajiwara, K. Obokata, and Y. Igarashi, Independent spanning trees of chordal rings, *Inform. Process. Lett.*, 69 (1999) 155–160.

[21] J.-S. Kim, H.-O. Lee, E. Cheng, and L. Lipták, Optimal independent spanning trees on odd graphs, *J. Supercomputing*, 56 (2011) 212–225.

[22] J.-S. Kim, H.-O. Lee, E. Cheng, and L. Lipták, Independent spanning trees on even networks, *Inform. Sci.*, 181 (2011) 2892–2905.

[23] C.-J. Lai, C.-H. Tsai, Embedding a family of meshes into twisted cubes, *Inform. Process. Lett.*, 108 (2008) 326–330.

[24] P.-L. Lai, C.-H. Tsai, Embedding of tori and grids into twisted cubes, *Theoret. Comput. Sci.*, 411 (2010) 3763–3773.

[25] Y-J. Liu, J.K. Lan, W.Y. Chou, and C. Chen, Constructing independent spanning trees for locally twisted cubes, *Theoret. Comput. Sci.*, 412 (2011) 2237–2252.

[26] K. Miura, S. Nakano, T. Nishizeki, and D. Takahashi, A linear-time algorithm to find four independent spanning trees in four connected planar graphs, *Internat. J. Found. Comput. Sci.*, 10 (1999) 195–210.

[27] S. Nagai and S. Nakano, A linear-time algorithm to find independent spanning trees in maximal planar graphs, *IEICE Trans. Fund. Electron. Comm. Comput. Sci.*, E84-A (2001) 1102–1109.

[28] K. Obokata, Y. Iwasaki, F. Bao, and Y. Igarashi, Independent spanning trees of product graphs and their construction, *IEICE Trans. Fund. Electron. Comm. Comput. Sci.*, E79-A (1996) 1894–1903.

[29] A.A. Rescigno, Vertex-disjoint spanning trees of the star network with applications to fault-tolerance and security, *Inform. Sci.*, 137 (2001) 259–276.

[30] S.-M. Tang, Y.-L. Wang, and Y.-H. Leu, Optimal independent spanning trees on hypercubes, *J. Inform. Sci. Eng.*, 20 (2004) 143–155.

[31] S.-M. Tang, J.-S. Yang, Y.-L. Wang, and J.-M. Chang, Independent spanning trees on multidimensional torus networks, *IEEE Trans. Comput.*, 59 (2010) 93–102.

[32] Y. Wang, J. Fan, G. Zhou, and X. Jia, Independent spanning trees on twisted cubes, *J. Parallel Distrib. Comput.*, 72 (2012) 58–69.

[33] Y. Wang, J. Fan, X. Jia, and H. Huang, An algorithm to construct independent spanning trees on parity cubes, *Theoret. Comput. Sci.*, 465 (2012) 61–72.

[34] J. Werapun, S. Intakosum, and V. Boonjing, An efficient parallel construction of optimal independent spanning trees on hypercubes, *J. Parallel Distrib. Comput.*, 72 (2012) 1713–1724.

[35] J.-S. Yang, H.-C. Chan, and J.-M. Chang, Broadcasting secure messages via optimal independent spanning trees in folded hypercubes, *Discrete Appl. Math.*, 159 (2011) 1254–1263.

[36] J.-S. Yang and J.-M. Chang, Independent spanning trees on folded hyper-stars, *Networks*, 56 (2010) 272–281.

[37] J.-S. Yang and J.-M. Chang, Optimal independent spanning trees on Cartesian product of hybrid graphs, *Comput. J.*, 57 (2014) 93–99.

[38] J.-S. Yang, J.-M. Chang, S.-M. Tang, and Y.-L. Wang, Reducing the height of independent spanning trees in chordal rings, *IEEE Trans. Parallel Distrib. Syst.*, 18 (2007) 644–657.

[39] J.-S. Yang, J.-M. Chang, S.-M. Tang, and Y.-L. Wang, On the independent spanning trees of recursive circulant graphs $G(cd^m, d)$ with $d > 2$, *Theoret. Comput. Sci.*, 410 (2009) 2001–2010.

[40] J.-S. Yang, J.-M. Chang, S.-M. Tang, and Y.-L. Wang, Constructing multiple independent spanning trees on recursive circulant graphs $G(2^m, 2)$, *Int. J. Found. Comput. Sci.*, 21 (2010) 73–90.

[41] J.-S. Yang, S.-M. Tang, J.-M. Chang, and Y.-L. Wang, Parallel construction of optimal independent spanning trees on hypercubes, *Parallel Comput.*, 33 (2007) 73–79.

[42] M.-C. Yang, Constructing edge-disjoint spanning trees in twisted cubes *Inform. Sci.*, 180 (2010) 4075–4083.

[43] M.-C. Yang, T.-K. Li, J.-M. Tan, L.-H. Hsu, On embedding cycles into faulty twisted cubes, *Inform. Sci.*, 67 (2007) 205–214.

[44] M.-C. Yang, Edge-fault-tolerant node-pancyclicity of twisted cubes, *Inform. Process. Lett.*, 109 (2009) 1206–1210.

[45] X. Yang, Q. Dong, E. Yang, J. Cao, Hamiltonian properties of twisted hypercube-like networks with more faulty elements, *Theoret. Comput. Sci.*, 412 (2011) 2409–2417.

[46] A. Zehavi and A. Itai, Three tree-paths, *J. Graph Theory*, 13 (1989) 175–188.