Independent Spanning Trees on Multidimensional Torus Networks

Shyue-Ming Tang, Jinn-Shyong Yang, Yue-Li Wang, and Jou-Ming Chang

Abstract—Two spanning trees rooted at vertex \( r \) in a graph \( G \) are called independent spanning trees (ISTs) if for each vertex \( v \) in \( G, v \neq r \), the paths from vertex \( v \) to vertex \( r \) in these two trees are internally distinct. If the connectivity of \( G \) is \( k \), the IST problem is to construct \( k \) ISTs rooted at each vertex. The IST problem has found applications in fault-tolerant broadcasting, but it is still open for general graphs with connectivity greater than four. In this paper, we shall propose a very simple algorithm for solving the IST problem on multidimensional torus networks. In our algorithm, every vertex can determine its parent for a specific independent spanning tree only depending on its own label. Thus, our algorithm can also be implemented in parallel systems or distributed systems very easily.

Index Terms—Independent spanning trees, internally disjoint paths, multidimensional torus, parallel algorithms, fault-tolerant broadcasting.

1 INTRODUCTION

In [3], Bhuyan and Agrawal proposed a class of generalized nearest neighbor mesh hypercubes, denoted by \( R(m_d, m_{d-1}, \ldots, m_1) \) (or \( R_d \) for short), where \( m_i > 1 \), for \( 1 \leq i \leq d \). Generalized nearest neighbor mesh hypercubes are also called \( d \)-dimensional torus networks in [2], [4], [9]. An \( R_d \) has \( N = \Pi_{i=0}^{d} m_i \) number of vertices. Then, each vertex between 0 to \( N - 1 \) can be expressed as a \( d \)-tuple \((x_d, x_{d-1}, \ldots, x_1)\) for \( 0 \leq x_i \leq m_i - 1 \). Each vertex \((x_d, x_{d-1}, x_1, x_{d-1}, \ldots, x_1)\) is connected to all of those vertices labeled by \((x_d, x_{d-1}, x_1 \pm 1, x_{d-1}, \ldots, x_1)\), where \( 1 \leq i \leq d \) and \( x_1 \pm 1 \) is taken modulo \( m_1 \). The index \( i \) of \( x_i \) will be referred to as the \( i \)th dimension of \( R_d \), while \( m_i \) is the size of dimension \( i \). For example, \( R(3,4) \) is shown in Fig. 1. The sizes of dimensions 1 and 2 in \( R(3,4) \) are 4 and 3, respectively. Note that each vertex is connected in each dimension to the vertices with labels \(+1\) and \(-1\) which will be called the \(+1\) and \(-1\) directions, respectively, relative to the current node.

The regularity and connectivity of a \( d \)-dimensional torus are related to \( d \). We denote by \( \delta \) the degree of vertices in \( R(m_d, m_{d-1}, \ldots, m_1) \). Then, \( \delta = 2\delta_1 + \delta_2 \), where \( \delta_1 \) is the number of \( m_1 \geq 2 \) and \( \delta_2 \) is the number of \( m_i = 2 \). For example, \( R(2,3,2,4) \) is a 6-regular torus since \( \delta = 2\delta_1 + \delta_2 = 2 \cdot 2 + 2 = 6 \).

A set of different paths connecting two vertices in a graph is said to be internally disjoint if and only if any two paths in the set have no common vertex except the two end vertices. Considering graph \( G = (V, E) \), a spanning tree of \( G \) is a subgraph of \( G \) that is a tree and contains all vertices in \( V \). Two spanning trees of \( G \) are said to be independent if they are rooted at the same vertex, say \( r \), and for every vertex \( v \in V \setminus \{r\} \), the two paths from \( r \) to \( v \), one path in each tree, are internally disjoint. A set of spanning trees on a graph is said to be independent if they are pairwise independent. For example, four independent spanning trees (ISTs) rooted at vertex \((0,0)\) on \( R(3,4) \) are shown in Fig. 2, where every edge is represented by an arc toward the root.

The study of the IST problem has contributed to our understanding of reliable communication protocols [5], [17]. For example, a rooted spanning tree on an underlying graph can be viewed as a broadcasting channel for data communication. Further, the fault tolerance can be achieved by sending \( k \) copies of the message along the \( k \) ISTs rooted at the source node. If the source node is faultless, this scheme can tolerate up to \( k - 1 \) faulty nodes and complete the broadcasting process.

From the first, the IST problem has received much attention. However, it is still unsolved for general graphs. Zehavi and Itai [28] conjectured that for any \( k \)-connected graph \( G \), there exist \( k \) ISTs rooted at an arbitrary vertex in \( G \). The conjecture has been confirmed only for \( k \)-connected graphs with \( k \leq 4 \) in [7], [8], [17], [28], and it is still open for arbitrary \( k \)-connected graphs when \( k \geq 5 \). By providing the construction schemes, the conjecture has been proved to hold for several restricted classes of graphs or digraphs, such as planar graphs [16], [21], [22], product graphs [23], chordal rings [18], [25], de Bruijn and Kautz digraphs [13], [14], hypercubes [24], [27], recursive circulant graphs [26], and so forth.

Note that the development of algorithms for constructing ISTs tends to pursue two research goals, one is to design efficient construction algorithms (e.g., see [18], [21], [25] for...
linear-time algorithms) and the other is to reduce the height of ISTs [14], [24], [25].

Here, we want to point out that tori are also a kind of product graphs. Hence, the algorithm in [23] can also produce enough ISTs for a torus. However, their algorithm does not consider the special topology of tori, i.e., two adjacent vertices only have different values in one dimension. Suppose that graph $G$ is a product graph of graphs $G_1$ and $G_2$. By using the algorithm proposed in [23], the ISTs of $G$ are not constructed by directly combining ISTs of $G_1$ and $G_2$. Instead, lots of transformations are needed for constructing the resulting ISTs. Since a torus $R_d$ needs $d - 1$ product operations, by using their algorithm to construct ISTs of $R_d$, one has to repeat the above process $d - 1$ times. Thus, constructing ISTs by using their algorithm becomes very complicated. Especially, from the engineering viewpoint, their algorithm is impractical. We also want to address here that the algorithm proposed in this paper is a generalization of the algorithm in [27], and the set of ISTs on a hypercube generated by the algorithm in this paper is the same as that generated by the algorithm in [27].

A related problem of the IST problem is the edge-disjoint spanning tree problem. However, a set of edge-disjoint spanning trees might not be a set of ISTs. In [11], Fragopoulou proposed an algorithm for finding directed edge-disjoint spanning trees on a $k$-regular star graph. For the edge-disjoint spanning tree problem on $(n, k)$-arrangement graphs which are a class of generalized star graphs, Chen et al. [6] gave a way to embed $2(n - k)$ edge-disjoint spanning trees on this graph. And later, in [20], Lin proposed an algorithm to embed $k(n - k)$ edge-disjoint spanning trees on $(n, k)$-arrangement graphs which is the maximum number of edge-disjoint spanning trees in an $(n, k)$-arrangement graph. As for cube-connected-cycles networks, the reader is referred to [12] and [15] for the related results.

In this paper, we shall propose a simple algorithm for solving the IST problem on tori. In particular, we shall show that in our algorithm, every vertex can determine its parent for a specific independent spanning tree only depending on its own label. Thus, our algorithm can also be implemented in parallel systems or distributed systems very easily.

The remaining part of this paper is organized as follows: Section 2 presents our algorithm that can construct $d$ ISTs on torus with degree $d$. In Section 3, we show the correctness of our algorithm. For avoiding tedious verification, we arrange
some lemmas in the Appendix for the interested readers. The last section contains our concluding remarks.

2 Constructing Independent Spanning Trees on $R^d$

Before presenting our algorithm, we have to give some notation and properties which are necessary for illustrating our algorithm. Throughout the paper, we use the term "torus" to mean a multidimensional torus.

To explicitly represent the adjacency of vertices in a torus, we use the notation $x \rightarrow y$ to mean that vertex $y$ is adjacent to vertex $x$ and $j$ is a jump from $x$ to $y$. Recall that the label of vertex $x$ is a $d$-tuple $(x_d, \ldots, x_{i+1}, x_i, x_{i-1}, \ldots, x_1)$ and $x$ is adjacent to $(x_d, \ldots, x_{i+1}, x_i + 1, x_{i-1}, \ldots, x_1)$ and $(x_d, \ldots, x_{i+1}, x_i - 1, x_{i-1}, \ldots, x_1)$. Jump $j$ occurs along a particular dimension $i$ and in a particular direction in that dimension. Due to our labeling scheme, the direction is either $+1$ or $-1$ relative to the label $x_i$. For example, jump $j = -3$ means that the dimension $d$ and direction of jump $j$ is $3$ and $-1$, respectively. We use $\text{dim}(j)$ and $\text{dir}(j)$ to stand for the dimension and direction, respectively, of jump $j$. Then, $x \rightarrow y$ means that $y = \text{dim}(j) = (x_d + \text{dir}(j)) \mod m_{\text{dim}(j)}$ and $x_i = y_i$ for $i \neq \text{dim}(j)$. For simplicity, in the remaining part of this paper, we use $j$ to denote $\text{dim}(j)$ when the dimension of jump $j$ is used as an index or a subscript and if the context is clear. Thus, the above formula can be rewritten as $y_i = (x_i + \text{dir}(j)) \mod m_i$. Further, the opposite jump of $j$, denoted by $\overline{j}$, is a jump with the same dimension and opposite direction of jump $j$. Thus, in this representation, $\text{dim}(j) = \text{dim}(\overline{j})$, $x_i = x_{\overline{j}}$, and $m_j = m_{\overline{j}}$. Let $J$ denote the set of all jumps in a torus, i.e., $J = \{\pm 1, \ldots, \pm d \}$. Note that in the special case of $m_j = 2$, a vertex through jumps $j$ or $\overline{j}$ directly to the same vertex and we only choose the jump $j$ with the $-1$ direction in this dimension. Then, the cardinality of $J$ is the degree of the torus. If a jump $j \in J$ occurs successively to connect $x$ and $y$, then we use the notation $x \xrightarrow{J} y$ to mean $y_j = (x_j + \text{dir}(j) \times \alpha) \mod m_j$ and $x_i = y_i$ for $i \neq \text{dim}(j)$, where $\alpha$ is the number of repetitions of jump $j$. Using torus $R(3, 2, 4)$ as an example, we have $J = \{1, -1, -2, +3, -3 \}$ which means that every vertex in dimensions 1 and 3 has two ways to go out, while a vertex in dimension 2 has only one way to go out. Accordingly, $(2, 1, 3) \xrightarrow{1} (2, 1, 0)$ (respectively, $(2, 1, 3) \xrightarrow{-1} (2, 0, 3)$) means that vertex $(2, 1, 3)$ takes only one step to reach $(2, 1, 0)$ (respectively, $(2, 0, 3)$) through the $+1$ (respectively, $-1$) direction of dimension 1 (respectively, dimension 2). And $(2, 1, 3) \xrightarrow{1} (2, 1, 1)$ (respectively, $(2, 1, 3) \xrightarrow{-1} (0, 1, 3)$) means that vertex $(2, 1, 3)$ takes two steps to reach vertex $(2, 1, 1)$ (respectively, $(0, 1, 3)$) through the $-1$ direction of dimension 1 (respectively, dimension 3).

Since a torus is vertex-symmetric, without loss of generality, we may consider the vertex $(0, 0, \ldots, 0)$ (call it vertex 0) as the root of $\delta$ ISTs on a torus. Suppose that $T$ is a set of $\delta$ ISTs rooted at vertex 0 for a torus. Based on the definition of the IST problem, it is obvious that the root owns a unique child in every spanning tree in $T$. If $j$ is the jump taken from the child to the root, we denote by $T_j$ for the specific spanning tree. For example, the four ISTs of $R(3, 4)$ are denoted by $T_{-1}, T_{+1}, T_{-2}, T_{+2}$ (see Figs. 2a, 2b, 2c, and 2d, respectively).

For each nonroot vertex $x = (x_d, x_{d-1}, \ldots, x_1)$ in $R^d$, the construction rule of a shortest path from $x$ to 0 (the root) is to reduce every $x_i$ to 0 ($i = 1, \ldots, d$) by taking a direction that is closer to 0. Based on a shortest path to 0, the jump set $J$ of $x$ is partitioned into three subsets: $J_p$ contains those jumps that appear in a shortest path from $x$ to 0, $J_q$ contains jumps whose opposite jumps belong to $J_p$, and $J_r = J \setminus (J_p \cup J_q)$ contains all the remaining jumps. Due to the rules of finding a shortest path from $x$ to 0, jump $j$ is in $J_p$ (respectively, $J_q$) if and only if $0 < x_i \leq |m_i|/2$ (respectively, $|m_i|/2 < x(i) \leq m_i - 1$), for $i = 1, 2, \ldots, d$. After $J_p$ is determined, $J_q$ is the set of jumps $J$ for all $j \in J_p$. Note that if $x_i = 0$, then both $+i$ and $-i$ belong to $J_r$. When sets $J_p, J_q, \text{and } J_r$ are clear from the context, we will omit the subscript and use $J_p, J_q, \text{and } J_r$, respectively, to replace them. For example, sets $J_p, J_q, \text{and } J_r$ of vertex $(1, 0, 3)$ in $R(3, 2, 4)$ are $\{1, -3\}, \{-1, +3\}, \text{and } \{-2\}$, respectively. Hereafter, we assume that $J = \{j_0, j_1, \ldots, j_{l-1}\}$ with $\text{dim}(j_0) < \text{dim}(j_1) < \cdots < \text{dim}(j_{l-1})$. In addition, jump $j_{i-1}$ in $J_p$ is called the predecessor of $j_i$ (denoted by pred$(J_p, j_i)$), while jump $j_{i+1}$ is called the successor of $j_i$ (denoted by succ$(J_p, j_i)$). Note that the indices $i - 1$ and $i + 1$ are taken modulo $t$ and the indices of the entries in $J_p$ are taken modulo $|J_p|$. Table 1 lists sets $J_p, J_q, \text{and } J_r$ for all nonroot vertices in $R(3, 4)$.

We shall construct ISTS of a torus by using the concept of incidence coloring that will help the correctness proof of our algorithm. An incidence of a graph $G = (V, E)$ is a vertex and an edge pair $(x, e)$, where $x \in V, e \in E$, and $x$ is incident with $e$. An edge can also be identified by a jump $j$ if its incident vertex $x$ is already known. Thus, we also use $(x, j)$ to stand for an incidence in a torus and call $(x, j)$ the jump representation of incidence $(x, e)$. Notice that if $(x, e)$ and $(y, e)$ are two distinct incidences sharing the same edge $e$, then in the jump representation, their corresponding incidences are $(x, j)$ and $(y, \overline{j})$, respectively. In the following, for brevity, we also call $(x, j)$ an incidence. For example, the four incidences of vertex $x = (1, 1)$ in Fig. 1 are $(x, -1), (x, +1), (x, -2), \text{and } (x, +2)$, where $(x, -1)$ represents the incidence with vertex $(1, 1)$ and the edge between $(1, 1)$ and $(1, 0)$, $(x, +1)$

<table>
<thead>
<tr>
<th>x</th>
<th>$J_p$</th>
<th>$J_q$</th>
<th>$J_r$</th>
</tr>
</thead>
<tbody>
<tr>
<td>(0, 1)</td>
<td>{−1}</td>
<td>{+1}</td>
<td>{−2, +2}</td>
</tr>
<tr>
<td>(0, 2)</td>
<td>{−1}</td>
<td>{+1}</td>
<td>{−2, +2}</td>
</tr>
<tr>
<td>(0, 3)</td>
<td>{+1}</td>
<td>{−1}</td>
<td>{−2, +2}</td>
</tr>
<tr>
<td>(1, 0)</td>
<td>{−2}</td>
<td>{−2}</td>
<td>{−1, +1}</td>
</tr>
<tr>
<td>(1, 1)</td>
<td>{−1, +2}</td>
<td>{−1, +2}</td>
<td>{−1, +2}</td>
</tr>
<tr>
<td>(1, 2)</td>
<td>{−1, +2}</td>
<td>{−1, +2}</td>
<td>{−1, +2}</td>
</tr>
<tr>
<td>(1, 3)</td>
<td>{−1, +2}</td>
<td>{−1, +2}</td>
<td>{−1, +2}</td>
</tr>
<tr>
<td>(2, 0)</td>
<td>{−2}</td>
<td>{−2}</td>
<td>{−1, +1}</td>
</tr>
<tr>
<td>(2, 1)</td>
<td>{−1, +2}</td>
<td>{−1, +2}</td>
<td>{−1, +2}</td>
</tr>
<tr>
<td>(2, 2)</td>
<td>{−1, +2}</td>
<td>{−1, +2}</td>
<td>{−1, +2}</td>
</tr>
<tr>
<td>(2, 3)</td>
<td>{−1, +2}</td>
<td>{−1, +2}</td>
<td>{−1, +2}</td>
</tr>
</tbody>
</table>
represents the incidence with vertex (1, 1) and the edge between (1, 1) and (1, 2), etc. To construct a set $T$ of ISTs rooted at vertex 0 in a torus, we assign every incidence a color so that a set of incidences with the same color forms a spanning tree. Meanwhile, any two spanning trees constructed in this way are independent. When a color is assigned to an incidence, we call it the \textit{incidence color} of that incidence. Since a torus $R_d$ is $d$-regular, $d$ colors are needed to identify $d$ ISTs. In accord with the subscript labels of trees in $T$, we use $c_1$, $c_2$, $c_3$, $c_4$, $c_5$, $c_6$, ..., $c_{d+1}$ to denote these $d$ colors. Note that if $m_i = 2$, then $c_{i+1}$ will not appear in the color set. Fig. 3 demonstrates an incidence coloring of $(3, 4)$ that can be used to construct the ISTs of Fig. 2. For brevity, we omit wrapped around edges in Fig. 3.

Now, we are at a position to describe our incidence coloring algorithm for every vertex $x$ in $R_d$.

**Algorithm.** INCIDENCE-COLORING

\begin{itemize}
  \item \textbf{Input:} A vertex $x$ in $R_d$. \item \textbf{Output:} All colors of incidences with $x$. \item \textbf{begin} \item Step 1. Determine $J_p$, $J_q$, and $J_r$ with respect to vertex $x$. \item Step 2. For all $j \in J_p$, assign color $c_j$ to incidence $(x, k)$, where $k = \text{succ} (J_p, j)$. \item Step 3. For all $j \in J_q$, assign color $c_j$ to incidence $(x, j)$. \item Step 4. For all $j \in J_r$, assign color $c_j$ to incidence $(x, j)$. \item \textbf{end INCIDENCE-COLORING} \end{itemize}

Note that, in Step 2, the color assigned to incidence $(x, j)$ for $j \in J_p$ is $c_{k}$, where $k = \text{pred} (J_p, j)$. In Step 4, the color assigned to incidence $(x, j)$ for $j \in J_r$ is $c_j$ since both jumps $j$ and $\overline{j}$ are in $J_r$. We use an example to illustrate Algorithm INCIDENCE-COLORING. Let us see how to obtain all incidence colors of vertices (1, 1) and (0, 2) in torus $R(3, 4)$ (see Fig. 3). Sets $J_p = \{+1, -2\}$, $J_q = \{-1, +2\}$, and $J_r = \emptyset$ with respect to $x = (1, 3)$ are computed in Step 1. In Step 2, the colors of incidences $(x, +1)$ and $(x, -2)$ are $c_{-1}$ and $c_{+1}$, respectively, since jump $-2$ is the successor of jump $-1$ in $J_p$ and vice versa. In Step 3, the colors of incidences $(x, -1)$ and $(x, +2)$ are $c_{-1}$ and $c_{+2}$, respectively, and the four incidence colors of vertex $(1, 3)$ are found. For computing the incidence colors of vertex $x = (0, 2)$, sets $J_p = \{-1\}$, $J_q = \{+1\}$, and $J_r = \{-2, +2\}$ are computed in Step 1. Since succ$(J_p, -1) = -1$ itself, the color of incidence $(x, -1)$ is $c_{-1}$, which is computed in Step 2. In Step 3, the color $c_{+1}$ is assigned to incidence $(x, +1)$. In Step 4, colors $c_{+1}$ and $c_{-2}$ are assigned to incidences $(x, -2)$ and $(x, +2)$, respectively.

By using Algorithm INCIDENCE-COLORING, we can construct a set $T$ of ISTs rooted at vertex 0 in torus $R_d$ as given below.

**Algorithm.** GEN-ISTS

\begin{itemize}
  \item \textbf{Input:} A torus $R_d$. \item \textbf{Output:} A set $T$ of $d$ ISTs, where $d$ is the degree of vertices in $R_d$. \item \textbf{begin} \item Step 1. Use Algorithm INCIDENCE-COLORING to color every incidence in $R_d$. \item Step 2. For all $j \in J$, let $T_j$ contain vertex 0 together with all of the incidences with color $c_j$. \item Step 3. Output the set of $T_j$ for every $j \in J$ as $T$. \item \textbf{end GEN-ISTS} \end{itemize}

As an example, Fig. 2 shows the resulting four trees after applying Algorithm GEN-ISTS on $(3, 4)$.

The main idea of our algorithm is based on the following propositions.

**Proposition 2.1.** For nonroot vertices $x$ and $y$, if $x \xrightarrow{d} y$ and jump $j \in J_p$, then either $J_p = J_p$ or $J_p = J_p \setminus \{j\}$.

\textbf{Proof.} Recall that if jump $j \in J_p$, and dir$(j) = -1$ (respectively, +1), then $0 < x_j \leq |m_j|/2$ (respectively, $|m_j|/2 < x_j \leq m_j - 1$). Since $x \xrightarrow{d} y$, $y_j = x_j + \text{dir}(j)$. If $y_j \neq 0$, then either $0 < y_j \leq |m_j|/2$ or $|m_j|/2 < y_j \leq m_j - 1$ is in accord with the inequalities of $x_j$. By definition, jump $j$ is also in $J_p$. For the case where $y_j = 0$, jumps $j$ and $\overline{j}$ are in $J_r$. We conclude that if $x \xrightarrow{d} y$ and jump $j \in J_p$, then either $J_p = J_p$ or $J_p = J_p \setminus \{j\}$. \hfill $\square$

**Proposition 2.2.** If a jump, say $j$, of vertex $x$ belongs to $J_q$, and dir$(j) = -1$ (respectively, +1), then there exists a vertex $y$ and an integer $\alpha$ such that $x \xrightarrow{d} y$ and $y_j = \lfloor m_j/2 \rfloor$ (respectively, $y_j = \lfloor m_j/2 \rfloor + 1 - x_j$).

\textbf{Proof.} In the following, we only consider the case where dir$(j) = -1$. With a similar reasoning, we can handle the case where dir$(j) = +1$. If jump $j \in J_p$, then $y \in J_p$. By definition, $y \in J_p$, if and only if $|m_j|/2 < x_j \leq m_j - 1$. The value in dim$(j)$ of every vertex in the path $x \xrightarrow{d} y$ is decreasing, and therefore, there exists a vertex $y$ with $y_j = \lfloor m_j/2 \rfloor$ and $\alpha = x_j - \lfloor m_j/2 \rfloor$. \hfill $\square$

**Proposition 2.3.** If a jump, say $j$, of vertex $x$ belongs to $J_r$, and dir$(j) = -1$ (respectively, +1), then there exists a vertex $y$ such that $x \xrightarrow{d} y$ and $y_j = 1$ (respectively, $y_j = m_j - 1$).

\textbf{Proof.} A jump $j$ belonging to $J_r$ implies $x_j = 0$. Then, $x \xrightarrow{d} y$ implies that $y_j = x_j + 1 = 1$ if dir$(\overline{j}) = +1$ and $y_j = x_j - 1 = m_j - 1$ if dir$(\overline{j}) = -1$. \hfill $\square$

3 \ \textbf{THE CORRECTNESS OF ALGORITHM GEN-ISTS}

To show the correctness of Algorithm GEN-ISTS, we have to prove that all of the subgraphs generated by this algorithm

![Fig. 3. An incidence coloring of torus $R(3, 4)$.](image-url)
are spanning trees and paths from every nonroot vertex to the root in any two spanning trees are internally disjoint. Since the lemmas for supporting Lemmas 3.1 and 3.2 are quite tedious to go through, for easy readability, we put them in the Appendix for interested readers. In the rest of this paper, we assume that all incidence colors are obtained by Algorithm INCIDENCE-COLORING.

Lemma 3.1. Every $T_j$ generated by Algorithm GEN-ISTS is a spanning tree of $R_d$.

Proof. By Lemma 1.1, each vertex in $R_d$ has a unique incidence with color $c_j$ for any $j \in J$. By Lemma 1.2, no adjacent vertices $x$ and $y$ in $R_d$ have the same incidence color in incidences $(x,e)$ and $(y,e)$, where $e$ is an edge between them. By selecting all the incidences with color $c_j$ in the colored $R_d$, there are exactly $n - 1$ selected edges.

By Lemmas 1.7 to 1.9, there is no cycle in the selected edges of the same incidence color. Therefore, $T_j$ forms a spanning tree for each $j \in J$. Consequently, there are $\delta$ spanning trees generated by Algorithm GEN-ISTS. □

We now show the independency of ISTs in $T$. If we can prove that any two paths from vertex $x$ to vertex 0 in any two trees are internally disjoint, then the independency of ISTs in $T$ follows. We define three types of paths according to the set to which jump $j$ belongs. A path starting from vertex $x$ to vertex 0 is called a $p$-path of $x$ if it contains incidence $(x,j)$ for some $j \in J_p$, and all incidences in the path have the same color. Replacing $j \in J_p$ by $j \in J_q$ (respectively, $j \in J_r$) in the above definition, we call the corresponding path a $q$-path (respectively, an $r$-path). According to the classification of paths, each incidence of a vertex will belong to a $p$-path, $q$-path, or $r$-path of that vertex. If we can prove that the two paths in any one cross relation of these three paths for a vertex are internally disjoint, then it is equivalent to proving the independency of ISTs in $T$.

Lemma 3.2. The ISTs generated by Algorithm GEN-ISTS are mutually independent.

Proof. By Lemma 1.10 (respectively, Lemmas 1.13 and 1.15), let $P$ and $Q$, if they exist, be two distinct $p$-paths (respectively, $q$-paths and $r$-paths) of any vertex $x$ in $R_d$. Then, no common vertex occurs in two distinct $p$-paths (respectively, $q$-paths and $r$-paths) except vertices $x$ and 0. By Lemma 1.11 (respectively, Lemmas 1.12 and 1.14), for any vertex $x$ in $R_d$, the vertices in a $p$-path and a $q$-path (respectively, a $p$-path and an $r$-path, and a $q$-path and an $r$-path) of vertex $x$ are internally disjoint. Therefore, the ISTs generated by Algorithm GEN-ISTS are mutually independent. □

Clearly, Algorithm INCIDENCE-COLORING takes $O(\delta N)$ time sequentially. After all, incidence colors are determined, Algorithm GEN-ISTS can build, in parallel, an independent spanning tree in constant time if $N$ processors are used. By combining Lemmas 3.1 and 3.2, we give the following summarized theorems.

Theorem 1. Algorithm GEN-ISTS can be parallelized to construct a set of $\delta$ ISTs on torus $R_d$ in $O(\delta)$ time by using $O(N)$ processors under the PRAM EREW computational model, where $\delta$ is the degree of $R_d$ and $N$ is the number of vertices in $R_d$.

In addition, we discuss the height of each IST found by Algorithm GEN-ISTS. Let $H(T_j)$ denote the height of IST $T_j$ of $R_d$, where $j \in J$.

Theorem 2. If $T_j$ is an IST of $R_d$, then $H(T_j) = \lceil (m_j - 1)/2 \rceil + \sum_{i=1}^{d} |m_i/2|$.

Proof. Let $s$ be the son of vertex 0 in $T_j$, $t$ be the vertex with $t_i = |m_i/2|$, and $f$ be the vertex with $f_i = |m_i/2|$ for $1 \leq i \leq d$ except $f_j = m_j - 1$ (respectively, $f_j = 1$) if $\text{dir}(j) = -1$ (respectively, $\text{dir}(j) = +1$). By Propositions 2.1, 2.2, and 2.2, the distance between vertex 0 and vertex $f$ in $T_j$ is the height of $T_j$. Therefore, $H(T_j) = d_j(r,s) + d_j(s,t) + d_j(t,f)$, where $r$ is vertex 0 and $d_j(x,y)$ denotes the distance between vertices $x$ and $y$ in $T_j$.

Clearly, $d_j(r,s) = 1$ and

$$d_j(t,f) = \begin{cases} |m_j/2| - 1, & \text{if } \text{dir}(j) = +1, \\ m_j - |m_j/2| - 1, & \text{if } \text{dir}(j) = -1. \end{cases}$$

To compute $d_j(s,t)$, we know that if $\text{dir}(j) = -1$ (respectively, $\text{dir}(j) = +1$), then $s = (0, \ldots, s_j, 0, \ldots, 0)$ with $s_j = 1$ (respectively, $s_j = m_j - 1$). Thus,

$$d(s,t) = \begin{cases} \sum_{i=1}^{d} |m_i/2| + m_j - 1 - |m_j/2|, & \text{if } \text{dir}(j) = +1, \\ \sum_{i=1}^{d} |m_i/2| + |m_j/2| - 1, & \text{if } \text{dir}(j) = -1. \end{cases}$$

The above formula can be simplified as

$$d(s,t) = \begin{cases} \sum_{i=1}^{d} |m_i/2| + m_j - 1 - 2|m_j/2|, & \text{if } \text{dir}(j) = +1, \\ \sum_{i=1}^{d} |m_i/2| - 1, & \text{if } \text{dir}(j) = -1. \end{cases}$$

By combining the above terms for $\text{dir}(j) = +1$, we can yield

$$H(T_j) = d_j(r,s) + d_j(s,t) + d_j(t,f) = 1 + \sum_{i=1}^{d} |m_i/2| + m_j - 1 - 2|m_j/2| + |m_j/2| - 1$$

$$= 1 + \sum_{i=1}^{d} |m_i/2| + m_j - |m_j/2| - 2$$

$$= \sum_{i=1}^{d} |m_i/2| + |(m_j - 1)/2|.$$ 

For the case where $\text{dir}(j) = -1$, we can also derive

$$H(T_j) = \lceil (m_j - 1)/2 \rceil + \sum_{i=1}^{d} |m_i/2|.$$ Therefore, the theorem follows. □

4 Concluding Remarks

In this paper, we have presented a simple algorithm for solving the IST problem on torus networks which can be parallelized very easily. The proposed algorithm makes use of an incidence coloring approach to determine the parent of a vertex in a specific spanning tree.

Since the root has only one child in every IST, an IST is optimal if the path from every vertex to the child of the root is a shortest path [24]. In [27], Yang et al. have designed a parallel algorithm for generating $d$ optimal ISTs on a...
d-dimensional hypercube which can also be generated by
our algorithm. Finally, we shall point out that the
constructing scheme of ISTs on a torus is not unique. For
instance, in Step 2 of Algorithm INCIDENCE-COLORING, if
the operation $k = \text{suc}(J_p, j)$ is replaced by $k = \text{pred}(J_p, j)$,
then a different set of ISTs can be obtained.

**APPENDIX**

**Lemma 1.1.** Let $(x, i)$ and $(x, j), i, j \in J$, be any two distinct incidences of a nonroot vertex $x$ in $R_d$ and $c_k$ and $c_{k'}$, respectively, be their incidence colors. Then, $c_k$ and $c_{k'}$ are different colors.

**Proof.** If we can prove that $k \neq k'$, then $c_k$ is different from $c_{k'}$. Due to the fact that each of jumps $i$ and $j$ may belong to one of $J_p, J_{cp}$, and $J_r$, we consider the following six cases.

**Case 1.** $i, j \in J_p$. In this case, by Algorithm INCIDENCE-COLORING, $k = \text{pred}(J_p, i)$ and $k' = \text{pred}(J_p, j)$. It is clear that $k \neq k'$ for $i \neq j$.

**Case 2.** $i \in J_p$ and $j \in J_p$. If jump $j \in J_p$, then $k' = j$. By the definition of $J_p$, jump $j$ is not in $J_p$, and $k = \text{pred}(J_p, i) \neq k'$.

**Case 3.** $i \in J_p$ and $j \in J_r$. If jump $j \in J_r$, then $k' = j$. Since $J_r$ is a complement set of $J_p \cup J_r$, $k = \text{pred}(J_p, i) \neq k'$.

**Case 4.** $i, j \in J_p$. In this case, by Algorithm INCIDENCE-COLORING, $k = i$ and $k' = j$. Since jumps $i$ and $j$ are distinct, $k \neq k'$.

**Case 5.** $i \in J_p$ and $j \in J_r$. In this case, by Algorithm INCIDENCE-COLORING, $k = i$ and $k' = j$. With a similar reasoning as Case 3, $k \neq k'$.

**Case 6.** $i, j \in J_r$. By Algorithm INCIDENCE-COLORING, $k = i$ and $k' = j$. Therefore, $k \neq k'$ for $i \neq j$. 

**Lemma 1.2.** Let $(x, e)$ and $(y, e)$ be two distinct incidences of $R_d$, where $e$ is the common edge of nonroot vertices $x$ and $y$. If $c_k$ and $c_{k'}$ are colors assigned to incidences $(x, e)$ and $(y, e)$, respectively, then $c_k$ and $c_{k'}$ are two different colors.

**Proof.** Let incidences $(x, j)$ and $(y, j)$ be the jump representations of incidences $(x, e)$ and $(y, e)$, respectively. For the possible cases of jump $j$ on vertex $x$, we consider the following three cases.

**Case 1.** $j \in J_p$. By Proposition 2.1, either $J_p = J_{cp}$ or $J_p = J_p \setminus \{j\}$. If $J_p = J_{cp}$, then by the mutual exclusive property of $j$ and $J_p$, $j$ must be in $J_{cp}$ and $k' = j$. Thus, $k = \text{pred}(J_{cp}, j) \neq k'$. For the case where jump $j$ is in $J_{cp}$, $k' = j$. Furthermore, $y_j = 0$. If $k$ is also equal to $j$, i.e., $j = \text{pred}(J_{cp}, j)$, then this means that $j$ is the only jump in $J_{cp}$, and vertex $y$ is vertex 0. This is impossible for the assumption that vertex $y$ is a nonroot vertex. Therefore, $k \neq k'$.

**Case 2.** $j \in J_r$. In this case, jump $j$ may belong to one of $J_p, J_{cp}$, or $J_r$. If $J_p = J_{cp}$, then by the mutual exclusive property of $j$ and $J_p$ in $J_p, j \notin J_p$. Thus, in the subcase where $j \in J_{cp}$ and $J_r$, $k = j$ and $k' = \text{pred}(J_{cp}, j) \neq k$. For the subcase where $j \in J_{cp}$ and $J_r$, by Algorithm INCIDENCE-COLORING, $k = j$ and $k' = j$. It is clear that $k \neq k'$ in this subcase.

For the subcase where $j \in J_p$ and $J_r$, by Proposition 2.3, $y_j = 0$. Then, either $x_j = 1$ or $x_j = m_j - 1$. For both the possible values of $x_j$, jump $j$ must be in $J_{cp}$. It contradicts that $j \in J_{cp}$. Thus, this subcase is impossible.

**Case 3.** $j \in J_r$. In this case, we only need to consider $j \in J_{cp}$ and $\bar{j} \in J_{cp}$, since the other subcases, i.e., $\bar{j} \in J_p$ or $j \in J_{cp}$, can be handled by a similar argument as in Cases 1 and 2. By Algorithm INCIDENCE-COLORING, $k = \bar{j}$ and $k' = j$. Therefore, $k \neq k'$ unless $m_j = 2$. However, if $m_j = 2$, then $\bar{j}$ will be in $J_{cp}$ but not $J_{cp}$. This completes the proof.

Lemmas 1.1 and 1.2 reveal that there are $N - 1$ incidences having the same incidence color. In the following, we shall show that for every vertex $x$ in $R_d$, there exist $\delta$ different paths from vertex $x$ to the root and all incidences in each path contain the same color. By using this result, we can prove that every tree obtained by Algorithm GEN-ISTS is a spanning tree. The proofs are given below after a few definitions and results. In the remaining part of this section, we shall denote that an incidence of some vertex, say $x$, is in a path, we mean the incidence which contains vertex $x$ and the edge which is closer to vertex 0 in the path. If all incidences in a path have the same incidence color, say $c_k$, then we say that the incidence color of this path is $c_k$. By Lemma 1.1, for any vertex $x$, there exists incidence $(x, j)$ for any jump $j \in J$. Recall that a path starting from vertex $x$ to vertex 0 is called a $p$-path of vertex $x$ if it contains incidence $(x, j)$ for some $j \in J_p$, and all incidences in the path have the same color. Replacing $j \in J_p$, by $j \in J_{cp}$ (respectively, $j \in J_r$) in the above definition, we call the corresponding path a $q$-path (respectively, an $r$-path).

For example, see Fig. 3 again. The path $(0, 2) \rightarrow (0, 1) \rightarrow (0, 0)$ is a $p$-path of vertex $(0, 2)$ since jump $1$ is in $J_p$ with respect to vertex $(0, 2)$ and the incidence color of this path is $c_1$. Path $(0, 2) \rightarrow (0, 3) \rightarrow (0, 0)$ is a $q$-path of vertex $(0, 2)$ in which the color assigned to every incidence is $c_1$ and jump $+1$ is in $J_q$ with respect to vertex $(0, 2)$. Paths $(0, 2) \rightarrow (2, 1) \rightarrow (1, 0) \rightarrow (0, 0)$ and $(0, 2) \rightarrow (2, 1) \rightarrow (0, 0) \rightarrow (2, 0) \rightarrow (0, 0)$ are $r$-paths of vertex $(0, 2)$ in which the former path has incidence color $c_2$ and the latter has color $c_2$ and both jumps $-2$ and $+2$ are in $J_r$ with respect to vertex $(0, 2)$.

We can find some properties on a $p$-path. Assume that $J_p = \{j_0, j_1, \ldots, j_t\}$ with $\dim(j_0) < \dim(j_1) < \cdots < \dim(j_t)$ and $k = j_t$. For some $0 \leq t \leq t - 1$. A vertex is called a $t$-jump vertex if $|J_p| = t$. For the case where $t = 1$, vertex $x$ is called a 1-jump vertex. If $t = 2$, then vertex $x$ is called a 2-jump vertex, etc. For a vertex $x$ with $|J_p| > 1$, we also call it a multijump vertex. A vertex $x$ is said to be a boundary $t$-jump in dimension $j$ if $j \in J_p$ and $x_j$ is equal to 1 or $m_j - 1$. For example, see Fig. 3 again. Vertices $(0, 1), (0, 2), (0, 3), (1, 0)$, and $(2, 0)$ are 1-jump vertices in which vertices $(0, 1)$ and $(0, 3)$ are boundary 1-jump vertices in dimension 1, and vertices $(1, 0)$ and $(2, 0)$ are boundary
1-jump vertices in dimension 2. The other vertices are 2-jump (or multijump) vertices where all vertices are boundary 2-jump vertices in dimension 2, while vertices (1, 1), (1, 3), (2, 1), and (2, 3) are boundary 2-jump vertices in dimension 1. Propositions 1.3-1.6 describe some properties of t-jump vertices.

**Proposition 1.3.** If vertex \( x \) is a 1-jump vertex and jump \( j \) is the only jump in \( J_p \), then there exists only one p-path from vertex \( x \) to the root and all incidences in the path are with color \( c_j \).

**Proposition 1.4.** Let vertex \( x \) be a boundary 2-jump vertex in dimension \( j \). Then, there exists a 1-jump vertex \( y \) such that either \( x \xrightarrow{j} y \) or \( x \xrightarrow{2-j} y \) must hold. Moreover, the incidence color of incidence \( (x, j) \) is \( c_b \), where \( b \in J_p \), which is also the color of all incidences in the path from vertex \( x \) to the root.

**Proposition 1.5.** Let vertex \( x \) be a boundary t-jump vertex in dimension \( j \) and \( |J_p| = t \geq 2 \). Then, there exists an \( s \)-jump vertex with \( s = t - 1 \) such that either \( x \xrightarrow{j} y \) or \( x \xrightarrow{2-j} y \) must hold.

**Proposition 1.6.** If vertex \( x_1 \) is a multijump vertex, then there exists a path \( P = x_1x_2 \ldots x_t \) such that \( x_i \xrightarrow{j} x_{i+1} \) and \( J_{p_{i-1}} = J_{p_{i-2}} \), for \( i = 1, 2, \ldots, t - 1 \) and \( x_t \) is a boundary multijump vertex in dimension \( j \). Moreover, every incidence in path \( P \) has color \( c_{b_t} \), where \( b_t = \text{pred}(x_t) \).

**Lemma 1.7.** For every jump \( j \in J_p \), there exists exactly one p-path of vertex \( x \) in \( R_d \).

**Proof.** Let \( k = \text{pred}(x) \). There are two cases to consider which are \( k = j \) and \( k \neq j \). First, we consider the case where \( k = j \). By Proposition 1.3, there is exactly one p-path from vertex \( x \) to the root with incidence color \( c_j \) and the lemma holds trivially. Now, we consider \( k \neq j \). By Propositions 1.5 and 1.6, there is exactly one path from vertex \( x \) to an \( s \)-jump vertex \( y \) with a specific incidence color \( c_s \) such that \( s = |J_p| = |J_p| - 1 \). Notice that in this path, every vertex passes through jump \( j \) to its next vertex and with incidence color \( c_s \). Similarly, starting from vertex \( y \) and applying the above process repeatedly, a unique boundary 2-jump vertex will be reached. Then, by Proposition 1.4, a unique 1-jump vertex will be reached in turn. The path from this 1-jump vertex to the root is already considered in the beginning of this proof. By combining all of the above subpaths, a unique p-path of vertex \( x \) corresponding to jump \( j \) is obtained and all incidences in the path have color \( c_j \).

For example, in Fig. 3, vertex (1, 2) has two p-paths. One of them is the path \( (1, 2) \xrightarrow{2-j} (0, 2) \xrightarrow{1-j} (0, 1) \xrightarrow{1-j} (0, 0) \) in which vertex (1, 2) is a boundary 2-jump vertex in dimension 2. By Proposition 1.4, through jump \( -2 \) with incidence color \( c_{-2} \), the newly reached vertex is vertex \( (0, 2) \). Since vertex \( (0, 2) \) is a 1-jump vertex, by Proposition 1.3, there is exactly one p-path from vertex \( (0, 2) \) to the root with incidence color \( c_{-1} \). The other p-path of vertex \( (1, 2) \) is path \( (1, 2) \xrightarrow{-1} (1, 1) \xrightarrow{-1} (1, 0) \xrightarrow{2-j} (0, 0) \) and all incidences in this path have color \( c_{-2} \).

**Lemma 1.8.** For every jump \( j \in J_q \), there exists exactly one q-path of vertex \( x \) in \( R_d \).

**Proof.** By Algorithm INCIDENCE-COLORING, color \( c_j \) is assigned to incidence \((x, j)\). By Proposition 2.2, there is exactly one path \( x \xrightarrow{\alpha} y \) for jump \( j \), where \( \alpha = x_j - [m_j/2] \) and \( y_j = [m_j/2] \) if \( \text{dir}(j) = -1 \) or \( \alpha = [m_j/2] + 1 - x_j \) and \( y_j = [m_j/2] + 1 \) if \( \text{dir}(j) = +1 \). Since all nonendpoints in the way \( x \xrightarrow{\alpha} y \) have the same \( J_q \) set as \( J_q \), it can be seen easily that every incidence in the way \( x \xrightarrow{\alpha} y \) is with color \( c_j \). When vertex \( y \) is reached, we can see that, by definition, jump \( j \) will be in \( J_p \). Then, by Lemmas 1.2 and 1.7, there is exactly one p-path form vertex \( y \) to vertex 0, and all incidences in the path are with color \( c_j \). By combining \( x \xrightarrow{\alpha} y \) and the final p-path, we can obtain a unique q-path for jump \( j \) of vertex \( x \) in which every incidence has color \( c_j \).

We use vertex (1, 3) in Fig. 3 to illustrate the meaning of Lemma 1.8. There are two q-paths of vertex \( x = (1, 3) \) since \( J_q = \{-1, +2\} \). The q-path of vertex \( x \) with incidence color \( c_{-1} \) is \((1, 3) \xrightarrow{-1} (1, 2) \xrightarrow{-1} (0, 2) \xrightarrow{-1} (0, 1) \xrightarrow{-1} (0, 0) \). Note that the first step \((1, 3) \xrightarrow{-1} (1, 2) \) of this q-path can be viewed as \((1, 3) \xrightarrow{-1} (1, 2) \) with \( \alpha = 1 \). Further, jump \(-1 \) is in \( J_q \) of vertex \( (1, 2) \) since \( J_q = \{-1, -2\} \) with respect to vertex \( (1, 2) \). The other q-path of vertex \( x \) is \((1, 3) \xrightarrow{+2} (2, 3) \xrightarrow{+2} (2, 0) \xrightarrow{+2} (0, 0) \) and every incidence in this path has color \( c_{+2} \).

**Lemma 1.9.** For every jump \( j \in J_r \), there exists exactly one r-path of vertex \( x \) in \( R_d \).

**Proof.** By Algorithm INCIDENCE-COLORING, color \( c_j \) is assigned to incidence \((x, e)\) whose jump representation is \((x, j)\). Let \( y \) be the vertex with incidence \((y, e)\), namely, \((y, j)\) in jump representation. By Proposition 2.3, either \( y = 1 \) if \( \text{dir}(j) = -1 \) or \( m_j - 1 \) if \( \text{dir}(j) = +1 \). In both cases, jump \( j \) will be in \( J_p \). Then, by Lemma 1.7, there exists exactly one p-path from vertex \( y \) with incidence color \( c_j \). Therefore, combining \( x \xrightarrow{\alpha} y \) with the p-path with incidence color \( c_j \) of vertex \( y \), we can obtain a unique r-path for jump \( j \) of vertex \( x \).

We use vertex (0, 1) in Fig. 3 to illustrate the meaning of Lemma 1.9. There are two r-paths of vertex \( x = (0, 1) \) since \( J_r = \{-2, +2\} \). The r-path of vertex \( x \) with incidence color \( c_{-2} \) is \((0, 1) \xrightarrow{+2} (1, 1) \xrightarrow{+2} (1, 0) \xrightarrow{+2} (0, 0) \). The other r-path of vertex \( x \) is \((0, 1) \xrightarrow{-2} (2, 0) \xrightarrow{-2} (0, 0) \) and the incidence color in this path is \( c_{+2} \).

**Lemma 1.10.** Let \( P \) and \( Q \) be two distinct p-paths, if they exist, of any vertex \( x \notin \emptyset \) in \( R_d \). Then, no common vertex occurs in these two p-paths except vertices \( x \) and \( 0 \).

**Proof.** Recall that \( J_p = \{j_0, j_1, \ldots, j_t\} \) with \( \text{dim}(j_0) < \text{dim}(j_1) < \cdots < \text{dim}(j_t) \). Assume without loss of generality that the incidence colors of paths \( P \) and \( Q \) are \( c_{j_k} \) and \( c_{j_l} \), respectively, for some \( 0 < i < t - 1 \). (Note that if the incidence colors of paths \( P \) and \( Q \) are \( c_{j_k} \) and \( c_{j_k} \),
Fig. 4. An illustration for Lemma 1.10.

respectively, for some $k < i$, the proof can be handled similarly.) Thus, by Lemma 1.7, the jumps occurred in $P$ and $Q$ are in these orders $j_1, j_2, \ldots, j_i, j_{i+1}, \ldots, j_t, j_0$ and $j_{i+1}, j_{i+2}, \ldots, j_t, j_0, j_{i+1}, \ldots, j_i$. Let $P = P_{j_1}P_{j_2} \cdots P_{j_t}P_{j_0}$ and $Q = Q_{j_1}Q_{j_2} \cdots Q_{j_t}Q_{j_0}$, where $P_i, 0 \leq i \leq t - 1$, denotes a subpath of $P$, and in this subpath, every incidence has the same jump $j_i$. Similarly, $Q = Q_{j_1}Q_{j_2} \cdots Q_{j_t}Q_{j_0}$, Assume that $y \neq 0$ is a vertex in path $P$. For the possible positions of vertex $y$ in path $P$, we consider the following three subpaths of path $P$: subpath $P_{j_i}P_{j_{i+1}} \cdots P_{j_t}P_{j_0}$, subpath $P_{j_{i+1}}P_{j_{i+2}} \cdots P_{j_t}P_{j_0}$, and subpath $P_{j_t}P_{j_0}$ and denote them by $P_j$, $P_m$, and $P_r$, respectively (see Fig. 4a). Now, we consider three cases to prove that vertex $y$ does not occur in path $Q$.

Case 1. Vertex $y$ is in subpath $P_j$ (see Fig. 4b).

In this case, we separate path $Q$ into two subpaths: subpath $Q_{j_1}Q_{j_2} \cdots Q_{j_t}Q_{j_0}$ and subpath $Q_{j_1}Q_{j_2} \cdots Q_{j_t}$, and subpath $Q_{j_t}$ denote them by $Q_f$ and $Q_r$, respectively. We can see that $y_{j_i} = x_{j_i} + \text{dir}(j_i)$ at the first step of subpath $P_j$ and never goes back to $x_{j_i}$ in this subpath. However, in subpath $Q_r$, the value in dim($j_i$) of any vertex in this subpath is the same as $x_{j_i}$. Thus, vertex $y$ cannot appear in subpath $Q_r$. In the subpath $Q_f$, vertex $y$ can also not appear in it since $y_{j_0}$ is different from any vertex in this subpath $Q_f$ in which the value of any vertex $y$ in dim($j_0$) is 0.

Case 2. Vertex $y$ is in subpath $P_m$ (see Fig. 4c).

In this case, let $Q_f$ and $Q_r$ denote subpaths $Q_{j_1}Q_{j_2} \cdots Q_{j_i-1}Q_{j_i}Q_{j_{i+1}} \cdots Q_{j_t}$ and $Q_{j_t}$, respectively. We can see that vertex $y$ cannot appear in the subpath $Q_f$. The reason is that $y_{j_i}$ is changed, i.e., $y_{j_i} = x_{j_i} + \text{dir}(j_i)$ in the beginning of subpath $P_m$. However, the value in dim($j_i$) of any vertex in subpath $Q_f$ remains unchanged as $x_{j_i}$. By observing the value in dim($j_0$), we can find that the value in dim($j_0$) of any vertex in subpath $Q_f$ is 0. However, $y_{j_0} = x_{j_0} \neq 0$ in subpath $P_m$.

This is why vertex $y$ cannot appear in subpath $Q_f$ in this case.

Case 3. Vertex $y$ is in subpath $P_r$ (see Fig. 4d).

Clearly, in this case, $y_{j_i} = 0$ when vertex $y$ is in subpath $P_r$. However, no vertex in path $P$ has value 0 in dim($j_i$) unless it is vertex 0. Therefore, in this case, vertex $y$ does not appear in subpath $Q_r$. This completes the proof. \hfill \qed

Lemma 1.11. For any vertex $x$ in $R_n$, the vertices in a $p$-path and a $q$-path of vertex $x$ are internally disjoint.

Proof. Let $j \in J_p$ be the jump used by a $q$-path of vertex $x$. Consider the values in dim($j$) of the vertices in both paths. The $q$-path of vertex $x$ contains a subpath $x \xrightarrow{a} y$ and another subpath from vertex $y$ to vertex 0 which is a $p$-path of vertex $y$ with incidence color $c_j$. This means that jump $j$ will be in $J_p$. In the subpath $x \xrightarrow{a} y$, the value in dim($j$) of every vertex is increased one by one if $\text{dir}(j) = +1$, and decreased one by one otherwise. In the following subpath, i.e., the $p$-path of vertex $y$ with incidence color $c_j$, the value in dim($j$) of every vertex remains unchanged as $y_j$ until a $1$-jump vertex, say $z$, is reached. Since vertex $z$ is a $1$-jump vertex and by Proposition 1.3, the values in dim($j$) of the vertices in the $p$-path of vertex $z$ are increased one by one if $\text{dir}(j) = +1$, and decreased one by one otherwise. However, $j \in J_p$ implies that $j \notin J_p$. By Proposition 1.3 again, the values in dim($j$) of the vertices in the $p$-path of vertex $x$ are increased or decreased in opposite direction comparing to the corresponding values in the $q$-path. Thus, the lemma follows. \hfill \qed

Lemma 1.12. For any vertex $x$ in $R_n$, the vertices in a $p$-path and an $r$-path of vertex $x$ are internally disjoint.

Proof. Let $j \in J_r$ be the jump used by an $r$-path of vertex $x$. We focus on the values in dim($j$) of the vertices in both paths. By Proposition 2.3 and Lemma 1.9, the values in dim($j$) of the vertices in the $r$-path of vertex $x$ are
Lemma 1.13. Let $P$ and $Q$, if they exist, be two distinct q-paths of any vertex $x$ in $R_d$. Then, no common vertex occurs in these two q-paths except vertices $x$ and 0.

Proof. Let $(x, j)$ be the starting incidence of path $P$, where $j \in J_x$. By Lemma 1.8, path $P$ contains two subpaths: $x \xrightarrow{j} y$ and a p-path from vertex $y$ to vertex 0. Therefore, the values in $\text{dim}(j)$ of the vertices in subpath $x \xrightarrow{j} y$ are increased one by one $\alpha$ times if $\text{dir}(j) = +1$, and otherwise, decreased one by one $\alpha$ times. Then, in the latter subpath, the values in $\text{dim}(j)$ of the vertices in the p-path from vertex $y$ to vertex 0 remain unchanged until a 1-jump vertex, say vertex $z$, is reached. Note that jump $j$ is the only jump in $J_x$. By Proposition 1.3, the values in $\text{dim}(j)$ of the vertices in the p-path from vertex $z$ to vertex 0 are also increased or decreased one by one according to the direction of jump $j$. However, the values in $\text{dim}(j)$ of the vertices in path $Q$ either remain unchanged as $x_j$, or are changed in an opposite direction to the corresponding position of path $P$. Thus, no common vertex occurs in paths $P$ and $Q$ except vertices $x$ and 0.

Lemma 1.14. For any vertex $x$ in $R_d$, the vertices in a q-path and an r-path of vertex $x$ are internally disjoint.

Proof. By definition, if jump $j \in J_x$, then $x_j = 0$. Thus, by Lemma 1.8, the values in $\text{dim}(j)$ of the vertices in any q-path from vertex $x$ to vertex 0 are always zero. However, by Lemma 1.9 and in the first step $x \xrightarrow{} y$ of the r-path with color $c_j$ of vertex $x$, the value $y_j$ is changed to 1 or $m_j - 1$ depending on the direction of $j$. After vertex $y$ is reached, a p-path with color $c_j$ of vertex $y$ will be a subpath of the original q-path of vertex $x$. Moreover, the values in $\text{dim}(j)$ of the vertices in the mentioned p-path of vertex $y$ are not equal to zero unless vertex 0 is reached. Therefore, the vertices in a q-path and an r-path of vertex $x$ are internally disjoint.

Lemma 1.15. Let $P$ and $Q$, if they exist, be two distinct r-paths of any vertex $x$ in $R_d$. Then, no common vertex occurs in these two r-paths except vertices $x$ and 0.

Proof. With a similar argument as Lemma 1.14, we can find that the values in $\text{dim}(j), j \in J_x$, of the vertices in path $P$ are not zero except vertices $x$ and 0. However, the values in $\text{dim}(j)$ of the vertices in path $Q$ remain zero consistently in path $Q$. Therefore, there is no common vertex in these two r-paths except vertices $x$ and 0.

ACKNOWLEDGMENTS

This research is supported by the National Science Council of Taiwan under the Grants NSC95-2115-M-141-001-MY2 and NSC96-2221-E-260-014.

REFERENCES


Shyue-Ming Tang received the MS degree from the National Defense Management College in 1986, and the PhD degree in information management from the National Taiwan University of Science and Technology in 2002. He is currently an associate professor at the National Defense University. His current research interests include graph theory and algorithm analysis.

Jinn-Shyong Yang received the BS degree in electrical engineering from the National Taiwan University of Science and Technology in 1993, and the MS and PhD degrees in information management from the National Taiwan University of Science and Technology, in 2000 and 2007, respectively. Currently, he is an associate professor in the Department of Information Management of National Taipei College of Business. His current research interests include graph theory, parallel computing, and network technology.

Yue-Li Wang received the BS and MS degrees from the Information Engineering Department of Tam-Kang University in 1975 and 1979, respectively, and the PhD degree in information engineering from the National Tsing-Hua University in 1988. Now, he is a professor in the Department of Information Management of National Taiwan University of Science and Technology. His research interests include graph theory, algorithm analysis, and parallel computing.

Jou-Ming Chang received the BS degree in applied mathematics from the Chinese Culture University in June 1987, the MS degree in information management from the National Chiao Tung University in June 1992, and the PhD degree in computer science and information engineering from the National Central University in June 2001. Currently, he is a professor in the Department of Information Management of National Taipei College of Business (NTCB). His research interests include algorithm design and analysis, graph theory and combinatorics, and parallel and distributed computing.

For more information on this or any other computing topic, please visit our Digital Library at www.computer.org/publications/dlib.