On Directed Edge-Disjoint Spanning Trees in Product Networks, An Algorithmic Approach

A.R. Touzene* and K. Day

Department of Computer Science, College of Science, Sultan Qaboos University, P.O. Box 36, Postal Code 123, Al-Khodh, Muscat, Sultanate of Oman.

Received 11 December 2013; accepted 15 September 2014

Abstract: In (Ku et al. 2003), the authors have proposed a construction of edge-disjoint spanning trees EDSTs in undirected product networks. Their construction method focuses more on showing the existence of a maximum number \((n_1+n_2-1)\) of EDSTs in product network of two graphs, where factor graphs have respectively \(n_1\) and \(n_2\) EDSTs. In this paper, we propose a new systematic and algorithmic approach to construct \((n_1+n_2)\) directed routed EDST in the product networks. The direction of an edge is added to support bidirectional links in interconnection networks. Our EDSTs can be used straightforward to develop efficient collective communication algorithms for both models store-and-forward and wormhole.

Keywords: Product networks, Directed edge-disjoint spanning trees, Interconnection networks.

*Corresponding author’s e-mail: touzene@squ.edu.om
1. Introduction

There has been increasing interest over the last two decades in product networks (Day, and Al-Ayyoub 1997; Ku et al. 2003; X and Yang 2007; Imrich et al. 2008; Klavari and Špacapan 2008; Jánicek et al. 2010; Hammack et al. 2011; Chen et al. 2011; Ma et al. 2011; Cheng et al. 2013; Erveš and Žerovnik 2013; Govorč et al. 2011; Kuškrovski 2014). The Cartesian product is a well-known graph operation. When applied to interconnection networks, the Cartesian product operation combines factor networks into a product network. Graph product is an important method to construct bigger graphs, including enhancing interconnection network fault-tolerance and developing efficient product networks is proposed. In Section 4, we conclude this paper.

2. Notations and Preliminaries

The Cartesian product $G = G_1 \times G_2$ of two undirected graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ is the undirected graph $G = (V, E)$, where $V$ and $E$ are given by: $V = \{<x_1, x_2> | x_1 \in V_1 \text{ and } x_2 \in V_2\}$, and for any $u = <x_j, y_j>$ and $v = <y_j, y_2>$ in $V$, $(u, v)$ is an edge in $E$ if, and only if, either $(x_j, y_j)$ is an edge in $E_1$ and $x_2 = y_2$, or $(x_j, y_2)$ is an edge in $E_2$ and $x_1 = y_1$. The edge $(u, v)$ is called a $G_1$-edge if $(x_j, y_j)$ is an edge in $G_1$, and it is called a $G_2$-edge if $(x_2, y_j)$ is an edge in $E_2$. $x_1$ is called the $G_1$-component of $u$ and $x_2$ is called the $G_2$-component. In all what follows we consider directed edges in the sense that the edge $(u, v)$ is different from the edge $(v, u)$.

3. Construction of EDSTs in a Product Network

Consider two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ having the following properties: the graph $G_1$ contains $n_1$ EDST all rooted at $x$ denoted: $X_1(x), X_2(x), \ldots, X_{n_1}(x)$. Each $X_i(x)$ tree is assumed to be formed of an edge $(x_i, x_i)$, where $x_i$ is the $i$th neighbor of $x$, and a sub-tree denoted $X_i(x)/x$ rooted at $x_i$ that spans all the $G_1$ nodes other than $x$ (Fig. 1.a). The graph $G_2$ contains $n_2$ EDST all rooted at $y$ denoted: $Y_1(y), Y_2(y), \ldots, Y_{n_2}(y)$. Each $Y_i(y)$ tree is assumed to be formed of an edge $(y_j, y_j)$, where $y_j$ is the $j$th neighbor of $y$, and a sub-tree denoted $Y_i(y)/y$ rooted at $y_j$ that spans all the $G_2$ nodes other than $y$ (figure 1.b). In Fig. 1 (a, b) straight lines correspond to $G_1$-edges and dashed lines correspond to $G_2$-edges.
In what follows, we fix a specific node \( <x_0, y_0> \) in \( G \) as a desired root for the EDST to be constructed. We denote by \( <x_i, y_i>, \ i = 1, \ldots, n_1 \), the \( n_1 \) neighbors of \( <x_0, y_0> \) in \( G \) reached from \( <x_0, y_0> \) via \( G_1 \)-edges, and by \( <x_0, y_j>, \ j = 1, \ldots, n_2 \), the \( n_2 \) neighbors of \( <x_0, y_0> \) reached from \( <x_0, y_0> \) via \( G_2 \)-edges. For a given node \( x \) in \( G_1 \) and a given tree \( Y \) in \( G_2 \), we denote by \( <x, Y> \) the tree in \( G_1 \times G_2 \) obtained by fixing the \( G_1 \)-component to \( x \) and following the edges of \( Y \) in \( G_2 \). Similarly, \( <X, Y> \) denotes the tree in \( G_1 \times G_2 \) obtained by following the edges of \( X \) in \( G_1 \) while the \( G_2 \)-component is fixed to node \( y \).

### 3.1 The \( ST_1 \) and \( ST_2 \) EDST for \( G \)

We present a construction algorithm of \( n_1 + n_2 - 2 \) EDST (without using non-tree edges) for the product graph \( G \). \( n_1 \)-1 EDST for \( G \) denoted \( ST_1(i), \ i = 2, \ldots, n_1 \) and \( n_2 \)-1EDST for \( G \) denoted \( ST_2(j), \ j = 2, \ldots, n_2 \).

### 3.2 Construction of \( ST_1(i) \), for any \( i \) \( 2 \leq i \leq n_1 \)

1. Connect \( <x_0, y_0> \) to its neighbor \( <x_i, y_i> \) (see edge labeled 1 in Fig. 2(a)).
2. Attach to \( <x_i, y_i> \) the sub-tree \( <x_i(x_0)/x_0, y_i(y_0)> \) (see sub-tree labeled 2 in Fig. 2(a)).
3. Connect \( <x_i, y_i> \) to its neighbor \( <x_j, y_j> \) (see edge labeled 3 in Fig. 2(a)).
4. To \( <x_i, y_i> \) attach the sub-tree \( <x_i, Y_i(y_i)/y_i> \) (see sub-tree labeled 4 in Fig. 2(a)).
5. To each node \( <x_i, y_i> \) in the sub-tree \( <x_i, Y_i(y_i)/y_i> \) (including its root \( <x_i, y_i> \) ) attach the tree \( <X_i(x_0)/x_0, y_i> \) (see sub-tree labeled 5 in Fig. 2(a)).

6. Connect each node \( <x_i, y_i> \) in the sub-tree \( <x_i, Y_i(y_i)/y_i> \) (including its root \( <x_i, y_i> \) ) to its neighbor \( <x_0, y_j> \) (see edge labeled 6 in Fig. 2(a)).

### 3.3 Construction of the tree \( ST_2(j) \), \( j = 2, \ldots, n_2 \)

1. Connect \( <x_0, y_0> \) to its neighbor \( <x_0, y_j> \) (see edge labeled 1 in Fig. 2(b)).
2. Attach to \( <x_0, y_j> \) the sub-tree \( <x_0, Y_j(y_j)/y_j> \) (see sub-tree labeled 2 in Fig. 2(b)).
3. Connect \( <x_0, y_j> \) to its neighbor \( <x_i, y_j> \) (see edge labeled 3 in Fig. 2(b)).
4. To \( <x_i, y_j> \) attach the sub-tree \( <X_i(x_0)/x_0, y_j> \) (see labeled 4 in Fig. 2(b)).
5. To each node \( <x_i, y_j> \) in the sub-tree \( <X_i(x_0)/x_0, y_j> \) (including its root \( <x_i, y_j> \) ) attach the tree \( <X_i(x_0)/x_0, y_j> \) (see sub-tree labeled 5 in Fig. 2(b)).
6. Connect each node \( <x_i, y_j> \) in the sub-tree \( <X_i(x_0)/x_0, y_j> \) (including its root \( <x_i, y_j> \) ) to its neighbor \( <x_1, y_0> \) (see edge labeled 6 in Fig. 2(b)). In figure 2(a, b), straight lines are \( G_1 \)-edges and dashed lines are to \( G_2 \)-edges.

**Theorem 1:** The set \( \{ST_1(i), 2 \leq i \leq n_1\} \cup \{ST_2(j), 2 \leq j \leq n_2\} \) is a family of \( (n_1 + n_2 - 2) \) edge-disjoint spanning trees in \( G \).

**Proof:** We show that all the nodes \( <x, y> \) of the product graph are reached in the \( (n_1 + n_2 - 2) \) edge-disjoint spanning tree using different edges.
• Case 1: nodes \(<x_0, y>\) are reached by different \(G_1\)-edges \(\langle x_0, y \rangle, \langle x_0, y \rangle, i = 2, \ldots, n_1\) in the different trees \(ST_1(i)\) (edges labeled 6 in Figure 2(a)). In trees \(ST_2(j), j = 2, \ldots, n_2\), these nodes are covered by \(G_2\)-edges of the sub-trees \(\langle x_0, y_{i_1} \rangle \) (edges labeled 5 in Figure 2(b)).

• Case 2: nodes \(<x, y>\), similar proof as in case 1 (symmetrical).

• Case 3: nodes \(<x, y>, i = 2, \ldots, n_1\) are covered in four different ways:
  1. In sub-trees \(\langle x_i, y_{i_1} \rangle \), \(i = 2, \ldots, n_1\) of trees \(ST_1(i)\) using \(Y_1\) tree edges (labeled 4 in Figure 2(a)).
  2. In sub-tree \(\langle x, y \rangle, j = 2, \ldots, n_2\) of the trees \(ST_2(j)\). These nodes are covered using \(Y_1\) tree edges \(j > 1\), (labeled 5 in Figure 2(b)).
  3. In sub-trees \(\langle x_i, y_{i_1} \rangle \), \(i = 2, \ldots, n_1\), of trees \(ST_1(i)\) using \(X_1\) tree edges (labeled 5 in Figure 2(a)).
  4. In sub-tree \(\langle x_i, y_{i_1} \rangle \), \(j = 2, \ldots, n_2\) of the trees \(ST_2(j)\) using \(X_1\) tree edges (labeled 4 in Figure 2(b)).

• Case 4: nodes \(<x, y>\), similar proof as in case 3 (symmetrical).

• Case 5: nodes \(<x, y>, x \neq x_i, y \neq y_{i_1}\) are covered using different \(G_1\)-edges in sub-trees \(\langle x_i, y_{i_1} \rangle \), \(i = 2, \ldots, n_1\) of trees \(ST_1(i)\) (sub-tree labeled 5 in Figure 2(a)). These nodes are covered using \(G_2\)-edges in the sub-trees \(\langle x_i, y_{i_1} \rangle \), \(j = 2, \ldots, n_2\) in the trees \(ST_2(j)\) (labeled 5 in Figure 2(b)).

3.4 The Special \(T_1\) and \(T_2\) EDSTs for \(G\)

We present a construction algorithm for the directed EDSTs in the product graph \(G\) denoted \(T_1\) and \(T_2\).

3.5 Construction of \(T_1\)

1. Connect \(<x_0, y>\) to its neighbor \(<x_1, y_0>\) (see edge labeled 1 in Figure 3(a)).

2. Attach to \(<x_1, y_0>\) the sub-tree \(<x_i, y_{i_1}>\) (see sub-tree labeled 2 in Figure 3(a)).

3. Connect \(<x_1, y_0>\) to its neighbor \(<x_1, y>\) (see edge labeled 3 in Figure 3(a)).

4. To each node \(<x_1, y>\), \(j = 1, \ldots, n_1\) in the sub-tree \(<x_i, y_{i_1}>\) (including its root \(<x_1, y_1>\) attach the tree \(<x_i, y_{i_1}>\) (see sub-tree labeled 4 in Figure 3(a)).

5. To each node \(<x_1, y>\), \(j = 1, \ldots, n_1\) in the sub-tree \(<x_i, y_{i_1}>\) (including its root \(<x_1, y_1>\) attach the tree \(<x_i, y_{i_1}>\) (see edge labeled 6 in Figure 3(a)).

6. Connect each node \(<x_1, y>\) in the sub-tree \(<x_i, y_{i_1}>\) (including its root \(<x_1, y_1>\) to its neighbor \(<x_0, y>\) (see edge labeled 5 in Figure 3(a)).
3.6 Construction of the Tree $T_2$

1. Connect node $x_i, y_j$ to its neighbor $x_0, y_i$ (see edge labeled 1 in Fig. 3(b)).
2. Attach to $x_0, y_i$ the sub-tree $x_0, Y_j(y_i)/y_i$ (see sub-tree labeled 2 in Fig. 3(b)).
3. Connect node $x_i, y_j$ to its neighbor $x_i, y_i$ (see edge labeled 3 in Fig. 3(b)).
4. To $x_i, y_i$ attach the sub-tree $X_i(x_i)/x_0, y_i$ (see labeled 4 in Fig. 3(b)).
5. To each node $x_i, y_i$, $i=1, \ldots, n$ in the sub-tree $X_i(x_i)/x_0, y_i$ (including its root $x_i, y_i$) attach the tree $x_i, Y_j(y_i)/y_i$ (see sub-tree labeled 5 in Fig. 3(b)).
6. Connect each node $x_i, y_j$ to its neighbor $x_0, y_i$ (including its root $x_i, y_j$) to its neighbor $x_0, y_i$ (see edge labeled 6 in Fig. 3(a)).
7. Connect each node $x_i, y_j$ to its neighbor $x_0, y_i$ (including its root $x_i, y_j$) to its neighbor $x_0, y_i$ (see edge labeled 6 in Fig. 3(a)).
8. Connect each node $x_i, y_j$ to its neighbor $x_i, y_i$ (including its root $x_i, y_j$) to its neighbor $x_i, y_i$ (including its root $x_i, y_i$) to its neighbor $x_0, y_i$ (see edge labeled 6 in Fig. 3(a)).

Figure 3. Construction of spanning trees $T_1$ and $T_2$. 

7. Connect each node $x_i, y_j$ in the sub-tree $X_i(x_i)/x_0, y_i$ (including its root $x_i, y_j$) to its neighbor $x_0, y_i$ (see edge labeled 6 in Fig. 3(a)).

8. Connect each node $x_i, y_j$ in the sub-tree $X_i(x_i)/x_0, y_i$ to the node $x_i, y_i$ (see edge labeled 6 in Fig. 3(a)).
Figure 4. Three EDSTs of the 3-cube and two EDSTs of the ring (3 nodes).
Figure 5 (a). Spanning Tree $ST_1(1)$. 

Figure 5 (b). Spanning Tree $ST_1(2)$. 

On Directed Edge-Disjoint Spanning Trees in Product Networks An Algorithmic Approach
Figure 5 (c). Spanning Tree $ST_1(3)$.

Figure 5 (d). Spanning Tree $ST_2(1)$. 
represent the dimension number relative to the 3-cube, see Figs. 4 and 5. The trees are directed from the root nodes to leave nodes.

4. Conclusions

In this paper, we presented a new systematic and algorithmic approach to construct $n_1+n_2$ (without using non-tree edges) directed rooted edges-disjoint spanning trees for product networks. The previous work on undirected EDSTs of the product networks (Ku et al. 2003) focuses more on the existence of $n_1+n_2-1$ but did not provide an explicit algorithmic way for their construction. Our $n_1+n_2$ EDSTs can be used straight-forward to develop efficient collective communication algorithms for both models store-and-forward and wormhole using bidirectional links.

References


