On the height of independent spanning trees of a *k*-connected *k*-regular graph

對於 *k*-連結, *k*-規則圖形獨立擴張樹高度之研究

Shyue-Ming Tang, Yue-Li Wang, Jing-Xian Lee

唐學明　　王有禮　　李景賢

國立台灣科技大學資管系

台北市基隆路四段 43 號

e-mail: ylwang@cs.ntust.edu.tw

Abstract

Two spanning trees of a given graph $G=(V,E)$ are said to be *independent* if they are rooted at the same vertex, say r, and for each 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 of G is said to be independent if they are pairwise independent. Zehavi and Itai conjectured that any k-connected graph has k independent spanning trees rooted at an arbitrary vertex. This conjecture is still open for $k \geq 4$. In this paper, we shall give the upper bound and the lower bound of the height of k independent spanning trees if they exist on a k-connected k-regular graph. An algorithm is also proposed to reduce the height of k independent spanning trees.

Keywords: independent spanning trees, vertex-disjoint path, connectivity, regular graph, fault-tolerant broadcasting.

1. Introduction

A set of paths connecting two vertices in a graph is said to be *internally disjoint* if and only if any pair of paths in the set has no common vertex except the two end vertices. Considering a graph $G=(V,E)$, a tree T is called a *spanning tree* of G if T is a subgraph of G and T 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 each 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 of a graph is said to be independent if they are pairwise independent.

Broadcasting in a distributed system is the message dissemination from a source node to every other node in the system. We can design a fault-tolerant broadcasting scheme based on independent spanning trees [1] [4]. The fault tolerance can be achieved by sending k copies of the message along n independent spanning trees rooted at the source node.

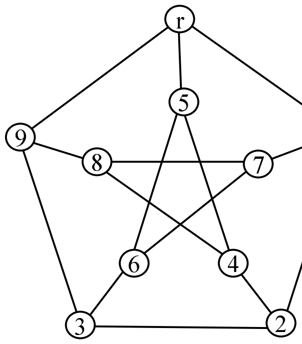
Itai and Rodeh [4] gave a linear time algorithm for finding two independent spanning trees in a biconnected graph. Cheriyan and Maheshwari [2] showed that, for any 3-connected graph G and for any vertex r of G, three independent spanning trees rooted at r can be found in $O(|V||E|)$ time. In [5] and [6], the authors conjectured that any k-connected graph has k independent spanning trees rooted at an arbitrary vertex r. Huck [3] has proved that the conjecture is true for planar graphs. The conjecture is still open for arbitrary k-connected graphs with $k \geq 4$.

A graph G is called k-regular if every vertex of G has degree k. A k-regular graph is not necessarily k-connected. This paper is concerned with k-regular graphs which is also k-connected. Most of interconnection networks are k-connected and k-regular, such as chordal rings, star graphs (Cayley graphs), hypercube, torus, and so on.

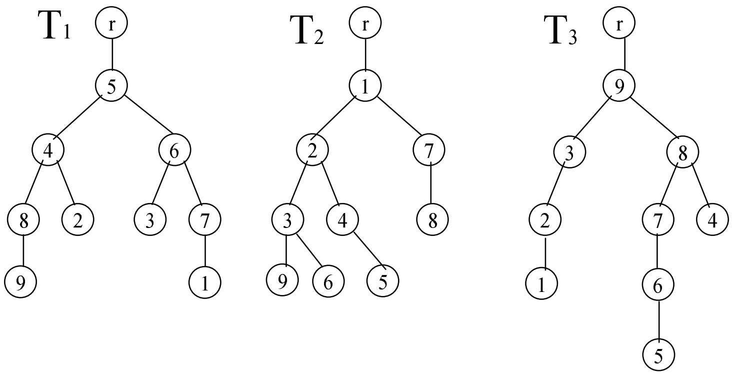
An example of 3-connected 3-regular graph is the Petersen graph, as shown in Figure 1(a). Using the algorithm proposed in [2], three independent spanning trees on the Petersen graph are constructed, as shown in Figure 1(b). Following the definition of independent spanning trees, for each vertex $v \in \{1,2,...,9\}$ the three paths from r to v, one path in each tree, are internally disjoint (or called vertex-disjoint).

In this paper, we focus our efforts on the height of the independent spanning trees of a k-connected k-regular graph. We shall give the upper bound and the lower bound of the height of independent spanning trees. It is obvious that the performance of the fault-tolerant broadcasting can be improved by reducing the height of independent spanning trees. An algorithm is also designed to achieve this purpose.

The remaining part of this paper is organized as follows. In Section 2, we introduce some properties of independent spanning trees in a k-connected k-regular graph. In Section 3, we shall propose an algorithm for reducing the height of independent spanning trees. Section 4 contains our concluding remarks.



(a)



(b)

Figure 1. (a) The Petersen graph (b) Three independent spanning trees on the Petersen graph.

2. The height of independent spanning trees

Let G be a k-connected k-regular graph. Suppose k independent spanning trees rooted at a vertex r in G exist, denoted by T_1, T_2, ..., T_k. We define *child*(v,i), *parent*(v,i) as the children set and the parent vertex of a vertex v in T_i, respectively. The *ancestor* of a vertex v in T_i, denoted by *ancestor*(v,i), is the vertex set of the path from r to v in T_i. The *descendant* of a vertex v in T_i, denoted by *descendent*(v,i), is the vertex set of the subtree rooted at v in T_i. The *neighborhood* of a vertex v is the set $N(v)$ consisting of all vertices which are adjacent with v in G. Then we have the following lemmas.

Lemma 1. *Let T_1, T_2, ..., T_k be k independent spanning trees rooted at a vertex r in G. For every vertex v in G, $v \neq r$,*

$parent(v,i) \cap parent(v,j) = \phi$, where $i \neq j$ and $i,j \in \{1,2,...,k\}$. Meanwhile, $\cup_{i=1,...,k}\ parent(v,i) = N(v)$.

Proof. This lemma holds directly from the definition of independent spanning trees. For every vertex $v \in V\backslash\{r\}$, if $parent(v,i) \cap parent(v,j) \neq \phi$, the two paths from r to v in T_i and T_j have a common vertex and are not internally disjoint, i.e., T_i and T_j are not independent. Therefore, every vertex $v \in V\backslash\{r\}$ must have k distinct parents in k independent spanning trees, and the k distinct parents are the neighborhood of v. Q.E.D.

Lemma 2. *Let T_1, T_2, ..., T_k be k independent spanning trees rooted at a vertex r in G. For every vertex v in G, $child(v,i) \cap child(v,j) = \phi$, where $i \neq j$ and $i,j \in \{1,2,...,k\}$. Meanwhile, if $v \in N(r)$, $\cup_{i=1,...,k}\ child(v,i) = N(v)\backslash\{r\}$, otherwise $\cup_{i=1,...,k}\ child(v,i) = N(v)$.*

Proof. Suppose there exist two independent spanning trees $T_i \neq T_j$ and $u \in child(v,i) \cap child(v,j)$, then for the vertex u, $parent(u,i) = parent(u,j)$. It is a violation against Lemma 1. Thus every vertex v in G must have k distinct children sets (may be empty) in k independent spanning trees.

For every vertex v, $v \notin N(r)$, if there exists a vertex $u \in N(v)$ but u is not in any of the children sets of v, then u must have a new parent in some T_i, it is a contradiction. Therefore, the union of the children sets of v must be the neighborhood of v. In case $v \in N(r)$, the children sets of v cannot include the root vertex r. Thus the union of $child(v,i)$ equals $N(v)\backslash\{r\}$. Q.E.D.

Corollary 3. *Let T_1, T_2, ..., T_k be k independent spanning trees rooted at a vertex r in G. The root vertex r of T_i, $i \in \{1,2,...,k\}$, must have one child.*

The proof is trivial.

Using vertex 4 in Figure 1 as an example, $parent(4,1) \cup parent(4,2) \cup parent(4,3) = \{5\}\cup\{2\}\cup\{8\} = \{5,2,8\} = N(4)$; $child(4,1) \cup child(4,2) \cup child(4,3) = \{8,2\}\cup\{5\} \cup\{\} = \{5,2,8\} = N(4)$. For every vertex in $N(r) = \{5, 1, 9\}$, the vertex is also the unique child of r in T_1, T_2, or T_3.

Lemma 4. *Let T_1, T_2, ..., T_k be k independent spanning trees rooted at a vertex r in G. For every vertex v in G, and $v \notin \{r\}\cup N(r)$, $N(v) \cap ancestor(v,i) = parent(v,i)$. For $v \in N(r)$, $N(v) \cap ancestor(v,i) = parent(v,i) \cup \{r\}$, where $i \in \{1,2,...,k\}$.*

Proof. Suppose a vertex u in T_i is both an ancestor and a neighbor of v, but not the parent of v. By Lemma 1, there must exist a T_j, $j \neq i$ and $parent(v,j) = u$, such that the two paths from r to v in T_i and T_j have a common vertex u and are not internally disjoint. Thus, T_i and T_j are not independent.

Obviously, root vertex r has no parent vertex in T_i. For $v \in N(r)$, there are two cases, $parent(v,j) = r$ or $parent(v,i) \neq r$. Whether r is the parent vertex of v or not, r is an ancestor of v in T_i. Therefore, $N(v) \cap ancestor(v,i) = parent(v,i) \cup \{r\}$. Q.E.D.

Similarly, we can prove the following lemma.

Lemma 5. *Let T_1, T_2, ..., T_k be k independent spanning trees rooted at a vertex r in G. For every vertex v in G and $v \neq r$, $N(v) \cap descendant(v,i) = child(v,i)$. In case $v = r$, $N(r) \cap descendant(r,i) = N(r)$, where $i \in \{1,2,...,k\}$.*

Proof. Suppose a vertex u in T_i is both a descendant and a neighbor of v, but not a child of v. By Lemma 2, there must exist a T_j, $j \neq i$ and $u \in child(v,j)$, such that the two paths from r to u in T_i and T_j have common vertex v and are not internally disjoint. Thus, T_i and T_j are not independent.

In case $v = r$, $N(r) \cap descendant(r,i) = N(r)$ because all vertices other than r are descendants of r in T_i. Q.E.D.

Corollary 6. *Let T_1, T_2, ..., T_k be k independent spanning trees rooted at a vertex r in G. The root of T_i, $i \in \{1,2,...,k\}$, has k-1 grandchildren.*

Proof. By Corollary 3, we know that the root vertex r of T_i has only one child. Let v be the unique child of r, all vertices other than r are descendants of v in T_i . Suppose v has k-2 children, there must exist one vertex u that is not a child of v, but u is both a descendant and a neighbor of v. This is a violation against Lemma 5. Q.E.D.

Corollary 7. *Let T_1, T_2, ..., T_k be k independent spanning trees rooted at a vertex r in G , the unique child of r in T_i has no child in T_j , for all $j \neq i$, i,j $\in$ {1,2,...,k}.*

Proof. By Corollary 6, we know that the root vertex r of T_i has k-1 grandchildren. Let v be the unique child of the root in some T_i, $v \in N(r)$. By Lemma 2, the union of k children sets of v in T_i has k-1 vertex. Therefore, for all $T_j \neq T_i$, v has an empty children set. Q.E.D.

We summarize Lemmas 4 and 5 as the following theorem.

Theorem 8. *Let T_1, T_2, ..., T_k be k independent spanning trees rooted at a vertex r in G. Let P be a path from r to v in T_i. Any vertex in P may have at most two neighbors that can be found in P.*

Proof. By Lemmas 4 and 5, for every vertex v in G, one of the three cases must be held: (i) For $v \notin \{r\} \cup N(r)$, a neighbor of v is either the parent or a child of v in T_i. (ii) For $v = r$, a neighbor of r is either the unique child of r or a leaf vertex (with no child) in T_i . (iii) For $v \in N(r)$, a neighbor of v is either the parent of v or the root vertex in T_i. Therefore, if P is a path from r to v in T_i , every vertex in P may have at most two neighbors also in P. Q.E.D.

Now, we are at a position to deduce the upper *bound* and the lower bound of the height of k independent spanning trees. Let $d_G(u,v)$ denote the *distance* between vertices u and v in G, the *height* of a spanning tree T rooted at r in G, denoted by *height*(T), is the maximum distance of the paths from r to any other vertices in T, i.e., $height(T) = \max\{ d_T(r,v) \}$, where $v \in V\backslash\{r\}$.

Theorem 9. *Let G be a k-connected k-regular graph with order n, and T_1, T_2, ..., T_k are k independent spanning trees of G, then* $\log_{k-1}(n-1)$ $<$ *height(T_i)* $< \dfrac{kn}{2(k-1)} - 1$, *for $i \in$ {1,2,...,k}.*

Proof. Let $P_i = <r, v_1, v_2, \ldots, v_h>$ be the longest path in T_i and let $height(T_i) = d_{Ti}(r,v_h) = h$. We prove the upper bound first. By Theorem 8, every vertex in P_i may have at least k-2 neighbors which are not in P_i . Thus, the total number of vertices outside P_i is at least $((h-1)(k-2)+2(k-1))/k$ if each vertex has k neighbors in P_i. We have the following inequation :

$$(h+1)(k-2) / k \;<\; n - (h+1) .$$

By reduction, we have

$$h \;<\; \frac{kn}{2(k-1)} - 1 .$$

The lower bound of h is found on such a flat tree that every internal vertex (except the root) has k-1 children, i.e.,

$$h \;>\; \log_{k-1}(n-1) .$$ Q.E.D.

3. An algorithm for reducing the height of independent spanning trees

In this section, we shall propose an algorithm to reduce the height of independent spanning trees. By analyzing and exchanging the positions of vertices in T_i, the height of T_i may be reduced.

In a k-connected k-regular graph G, every vertex v has k neighbors. By Lemma 1, for $v \neq r$, each neighbor of v is the parent of v in T_i ($i \in \{1,2,...,k\}$). If the position of v in T_i changes, its parent must be changed. Thus a change in T_i triggers another change in T_j ($j \neq i$). This phenomenon results in a k-permutation of the parents of v. We define a *possible exchange* of the positions of v in T_i as a permutation ($\pi_1\ \pi_2 \ldots \pi_k$). That is, the parent of v in T_i is changed from vertex *parent*(v,i) to vertex *parent*(v,π_i). For example, vertex 6 in Figure 1(b) has three parents, one in each tree. The possible exchanges are (321), (312), (231), (213), (132) and (123), where (π_1 π_2 π_3) denotes that the parent of vertex 6 in T_i is changed from vertex *parent*(6,*i*) to vertex *parent*(6, π_i), *i*=1,2,3. By the way, exchange (123) means no exchange.

Although there are *k*! possible exchanges for *k* distinct parents of *v* in *k* independent spanning trees, only some of them are feasible. A *feasible exchange* is defined as the possible exchange that follows properties of independent spanning trees. Particularly, a feasible exchange cannot violate the "2-neighbors" property in T_i mentioned in Theorem 8. The position of the unique child of the root in T_i cannot be changed because all exchanges are infeasible. Let *u* be the new parent of *v* in T_i , the exchange is feasible if and only if (i) $u \notin$ *descendant*(*v*,*i*), (ii) for all *w* in *descendant*(*v*,*i*)\{*v*}, *N*(*w*) $\cap$ *ancestor*(*u*,*i*) = ϕ or {*r*}, i.e., $\cup_{w\in descendant(v,i)} = \{u\}$ or {*u*,*r*}. The time complexity for identifying the feasibility of an exchange in *k* independent spanning trees is $O(k^2 n)$, where *n* is the number of vertices in *G*.

For example, vertex 6 in Figure 1(b), only (132) is feasible exchange because the position of vertex 6 in T_1 cannot be changed. Figure 2 is the result of the feasible exchange, where T_1*, T_2* and T_3* denote three independent spanning trees after the feasible exchange. Obviously, the height of T_3 is reduced from 5 to 4.

A feasible exchange may not be beneficial to the height of independent spanning trees. Let T_i* (i∈ {1,2,...,*k*}) denote *k* independent spanning trees. We define the *benefit* of a feasible exchange on vertex *v* in T_i by

$$benefit(v, i, x_i, y_i) = |descendant(v,i)|\ (x_i - y_i),$$

where x_i and y_i denote the distance from *r* to *v* in T_i and T_i* respectively. Note that an exchange affects not only the distance from *r* to *v* but also the distance from *r* to all descendants of *v*. Thus we multiple the distance change with the number of vertices in *descendant*(*v*,*i*).

The *total benefit* of a feasible exchange on vertex *v* is the summation of the benefit in T_i (i∈ {1,2,...,*k*}), i.e.,

$$total\_benefit(v) = \Sigma\ benefit(v, i, x_i, y_i),$$

where $i \in \{1,2,...,k\}$.

A feasible exchange is beneficial if and only if the total benefit is positive. The time complexity for computing the total benefit of an exchange is O(*n*), where *n* is the number of vertices in *G*.

For example, the total benefit of the feasible exchange (132) on vertex 6 in Figure 2 is *benefit*(6, 2, 4, 3) + *benefit*(6, 3, 4, 3) = 1 (4-3) + 2 (4-3) = 3.

For a small constant *k*, the following algorithm can reduce the height of independent spanning trees on a *k*-connected *k*-regular graph.

Algorithm Reduce_Height

Input: A *k*-connected *k*-regular graph *G*, a root vertex *r*, and *k* independent spanning trees T_i , $i\in \{1,2,...,k\}$.

Output: *k* new independent spanning trees with reduced height.

Step 1. For every vertex *v* in *G* except *r* do

Step 2. For every possible exchange π do

Step 3. Identify the feasibility of π in T_i , $i\in \{1,2,...,k\}$

Step 4. If π is not feasible then goto Step 2

Step 5. Compute *total_benefit*(*v*) for the feasible exchange π

Step 6. Determine the exchange of *v* with maximum *total_benefit*(*v*)

Step 7. Execute the exchange in T_i , $i\in \{1,2,...,k\}$

End of Algorithm Reduce_Height

Step 3 takes $O(k^2n)$ time to identify the feasibility of a possible exchange in *k* independent spanning trees. Step 5 takes O(*kn*) time to compute the total benefit of a feasible exchange. Thus the complexity of algorithm **Reduce_Height** is $O(n^2)$ for a small constant *k*.

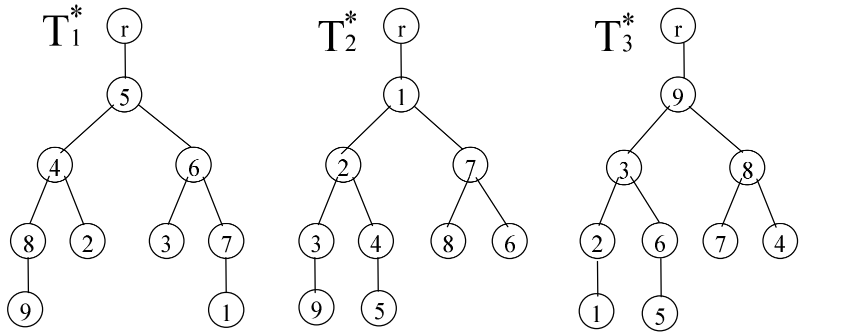


Figure 2. Independent spanning trees of the Petersen graph after a feasible exchange on vertex 6.

4. Concluding remarks

For a k-connected k-regular graph G, we give the upper bound and the lower bound of the height of k independent spanning trees. However, It remains unknown whether there is a polynomial time algorithm for reducing the height of k independent spanning trees to a minimum extent. It also remains unknown whether there is an efficient algorithm to construct k independent spanning trees rooted at arbitrary vertex in G directly. We are now working on these topics.

Acknowledgement

This work is supported by the National Science Council, Republic of China, under Contract 90-2213-E-011-067.

References

[1] F. Bao, Y. Igarashi, S.R. Öhring, "Reliable broadcasting in product networks ," *IEICE Technical Report COMP* 95(18), 1995, pp.57-66.

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

[3] A. Huck, "Independent trees in planar graphs," *Graphs and Combinatorics* 15, 1999, pp.29-77.

[4] A. Itai, M. Rodeh, "The multi-tree approach to reliability in distributed networks," *Information and Computation* 79, 1988, pp.43-59.

[5] Samir Khuller, Baruch Schieber, "On independent spanning trees", *Information Processing Letters* 42, 1992, pp.321-323.

[6] A. Zehavi, A. Itai, "Three tree-paths," *Journal of Graph Theory* 13, 1989, pp.175-188.