Joint Conference of 1996 International Computer Symposium
December 19~21, Kaohsiung, Taiwan, R.O.C.

An Efficient Parallel Algorithm for Constructing a Spanning Tree on Trapezoid Graphs

Hon-Chan Chen and Yue-Li Wang

Department of Information Management
National Taiwan Institute of Technology
Taipei, Taiwan, Republic of China.

Abstract

Let $G = (V, E)$ be a connected graph of n vertices and m edges. The problem of constructing a spanning tree is to find an acyclic subgraph of G with $n-1$ edges. In this paper, we propose an efficient parallel algorithm for solving this problem on trapezoid graphs. Our algorithm runs in $O(\log n)$ time with $O(n / \log n)$ processors on the EREW PRAM model.

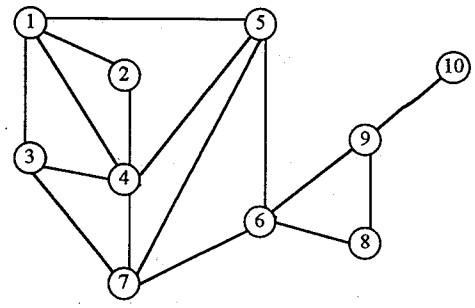
1. Introduction

Let $G = (V, E)$ be a graph with vertex set V and edge set E. A *spanning tree* of G is a spanning subgraph of G that is a tree [5]. Typically, there are many different spanning trees in a connected graph. Some properties of a tree are described as follows: On a graph T with $|V| = n$ and $|E| = m$, the following statements are equivalent:

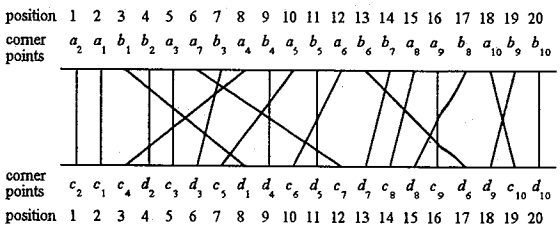
(1) T is a tree;
(2) T is connected and $m = n - 1$;
(3) Every pair of distinct vertices of T is joined by a unique path;
(4) T is acyclic and $m = n - 1$.

In this paper, we will construct a spanning tree on trapezoid graphs. A trapezoid i is defined by four corner points $[a_i, b_i, c_i, d_i]$ such that a_i and b_i are on the top channel and c_i and d_i are on the bottom channel of the trapezoid diagram. A graph $G = (V, E)$ is a *trapezoid graph* if it can be represented by a trapezoid diagram such that each trapezoid corresponds to a vertex in V and $(i, j) \in E$ if and only if trapezoids i and j intersect in the trapezoid diagram [4]. Figure 1 presents a trapezoid graph with its trapezoid diagram. In the diagram, there are 10 trapezoids, and the four corner points of trapezoid i are a_i, b_i, c_i and d_i, $i = 1, 2, ..., 10$. The class of trapezoid graphs includes two well-known classes of intersection graphs: the permutation graphs and the interval graphs. The permutation graphs are obtained in the case where $a_i = b_i$ and $c_i = d_i$ for all i, and the interval graphs are obtained in the case where $a_i = c_i$ and $b_i = d_i$ for all i.

It is easy to show that a trapezoid diagram can be reconstructed into another trapezoid diagram corresponding to the same trapezoid graph such that each trapezoid has four distinct corner points and all corner points are distinct. Therefore, we assume that the corner pointers on our trapezoid diagram are all distinct, and each corner point is at a specific position. We also assume that trapezoids are labeled in increasing order of their b corner points; i.e. $i < j$ if $b_i < b_j$. For example, in Figure 1 (b), trapezoid 6 is before trapezoid 7 since b_6 is at position 13 and b_7 is at position 14 on the top channel.



(a)



(b)

Figure 1. (a) A trapezoid graph
(b) The corresponding trapezoid diagram.

A number of algorithms for finding spanning trees are well known. In 1959, Moore [12] gave the Breadth-First Search algorithm to explore an unweighted graph and find its spanning tree. Its time complexity is $O(n + m)$, where n is the number of vertices and m the number of edges. For some restrict graphs, efficient sequential and parallel algorithms of finding spanning trees have been proposed [2, 6, 13, 14, 15]. Trapezoid graphs were first studied in [3, 4]. In [4], Dagan introduced a coloring algorithm for trapezoid graphs. Ma [11] presented an $O(n^2)$ time algorithm for recognizing trapezoid graphs. Recently, Daniel Liang [9, 10] gave some sequential algorithms for dominating and spanning tree problems on trapezoid graphs. There have been some parallel algorithms for permutation graphs [1, 6, 14]. Extending those algorithms from permutation graphs to trapezoid graphs is an interesting study.

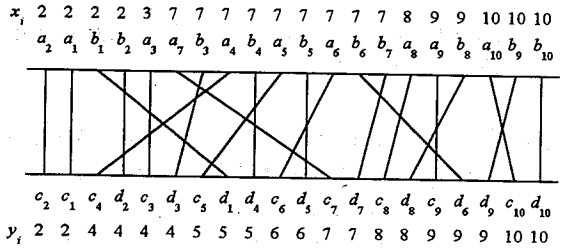
In this paper, we will propose an efficient parallel algorithm for constructing a spanning tree on connected trapezoid graphs. Our algorithm runs in $O(\log n)$ time with $O(n / \log n)$ processors on the EREW PRAM (Exclusive-Read-Exclusive-Write Parallel Random Access Machine) computational model. The remaining part of this paper is organized as follows. In Section 2, we introduce a parallel algorithm for constructing a spanning tree. The correctness of our algorithm is shown in Section 3. Finally, in Section 4, we give the conclusion of this paper.

2. An Algorithm for Constructing a Spanning Tree

Let G be a trapezoid graph of n vertices which are labelled from 1 to n, and let v be a vertex of G. Vertices adjacent to v are called the *neighbors* of v. The *maximum neighbor* of v is the neighbor which is greater than all other v's neighbors. If v's neighbors are all less than v, we call v an *absorbed vertex* and let v itself be its maximum neighbor; otherwise, v is a *normal vertex*. Our algorithm of constructing a spanning tree is presented as follows. In the algorithm, $pos(\cdot)$ stands for the position of some corner point, and (i, j) stands for the edge whose end vertices are i and j.

Algorithm A
Input: The trapezoid diagram of G with n trapezoids.
Output: A spanning tree T of G.
Method:
Step 1. Let T be a graph with n vertices and no edges.
Step 2. Scan corner points on the top channel from position 1 to position $2n$. Let x_i, $1 \le i \le 2n$, be the maximum trapezoid number among scanned trapezoids.
Step 3. Scan corner points on the bottom channel from position 1 to position $2n$. Let y_i, $1 \le i \le 2n$, be the maximum trapezoid number among scanned trapezoids.
Step 4. For each trapezoid i, $1 \le i \le n$, find its maximum neighbor. Suppose $pos(b_i) = s$ on the top channel and $pos(d_i) = t$ on the bottom channel. Let $z_i = \max \{x_s, y_t\}$. Then, the maximum neigh-bor of trapezoid i is z_i.
Step 5. For $i = 1, 2, ..., n-1$, let $T = T \cup (i, z_i)$ if $i \neq z_i$.
Step 6. If $i \neq z_i$ for all i, $1 \le i \le n-1$, then go to Step 8. Otherwise, let p_i, $1 \le i \le n-1$, be the maximum number in $\{1, 2, ..., i\}$ such that $z_{p_i} = \max \{z_1, z_2, ..., z_i\}$.
Step 7. For $i = 1, 2, ..., n-1$, let $T = T \cup (i, p_i)$ if $i = z_i$.
Step 8. The resulting T is a spanning tree.
End of Algorithm A

We use the graph of Figure 1 (b) as an example to illustrate Algorithm A. After the initialization in Step 1, we scan corner points on both channels from left to right. Figure 2 (a) shows the results of doing Steps 2 and 3. Step 4 finds the maximum neighbor for each trapezoid. For example, the maximum neighbor of trapezoid 3 is trapezoid 7 since $pos(b_3) = 7$ and $pos(d_3) = 6$ and the maximum of $x_7 = 7$ and $y_6 = 4$ is 7. In Step 5, we construct edges for normal vertices. Figure 2 (b) illustrates Steps 4 and 5. Since trapezoid 7 is an absorbed vertex, Steps 6 and 7 are needed. Since 6 is the maximum number in $\{1, 2, ..., 7\}$ such that $z_6 = \max \{z_1, z_2, ..., z_7\}$, $p_7 = 6$. Edges for absorbed vertices are constructed in Step 7. Steps 6 and 7 are shown in Figure 2 (c). Figure 2 (d) is the resulting spanning tree.



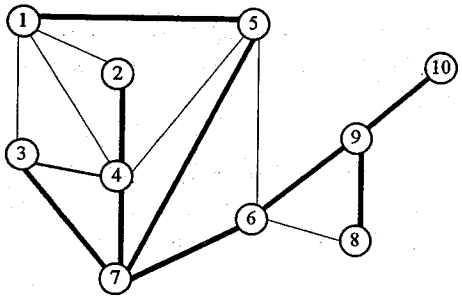
(a)

i	1	2	3	4	5	6	7	8	9
z_i	5	4	7	7	7	9	7	9	10

(b)

i	1	2	3	4	5	6	7	8	9
z_i	5	4	7	7	7	9	7	9	10
p_i	1	1	3	4	5	6	6	8	9

(c)



(d)

Figure 2. (a) Scanning all corner points
(b) Constructing edges for normal vertices
(c) Constructing edges for absorbed vertices
(d) The resulting spanning tree.

By applying parallel prefix computation technique [7, 8], Steps 2, 3, and 6 take $O(\log n)$ time and $O(n / \log n)$ processors on the EREW PRAM model. Other steps obviously can be done in $O(\log n)$ time using $O(n / \log n)$ processors. Thus, the time complexity of Algorithm A is $O(\log n)$ time with $O(n / \log n)$ processors on the EREW PRAM model.

3. The Correctness of Algorithm A

In this section, we will prove the correctness of our algorithm. We assume trapezoid graph G is connected. The following two properties are according to the definition of trapezoid graphs and the trapezoid diagrams.

Property 1. *Let i and j, $i < j$, be two vertices of G. If i is adjacent to j, then $pos(b_j) > pos(a_j)$ or $pos(d_j) > pos(c_j)$ in the trapezoid diagram.*

Property 2. *Let i, j, and k, $i < j < k$, be three vertices of G. If i is adjacent to k, then j is adjacent to i or k.*

Lemma 3. *Suppose $pos(b_i) = s$ and $pos(d_i) = t$ for some vertex i, $1 \le i \le n$, in the trapezoid diagram. Let $z_i = \max\{x_s, y_t\}$. Then, the maximum neighbor of i is z_i.*

Proof. Since x_s is the maximum trapezoid number among all scanned trapezoids at position s on the top channel, $i \le x_s$ and $pos(b_i) > pos(a_{x_s})$. Similarly, $i \le y_t$ and $pos(d_i) > pos(c_{y_t})$ on the bottom channel. Thus, the maximum neighbor of vertex i is z_i.

Q.E.D.

Lemma 4. *Let z_i be the maximum neighbor of i for each vertex i, $1 \le i \le n-1$. Then, the graph composed of edges (i, z_i) for all normal vertices i is acyclic.*

Proof. By Lemma 3, it is clear that (i, z_i) is an edge of G if i is a normal vertex. Since $z_i > i$ for all normal vertices i, there is no cycle in the graph composed of all (i, z_i).

Q.E.D.

Lemma 5. *Let i be an absorbed vertex, $1 \le i \le n-1$. Then, there must exist a vertex j, $j < i$, whose maximum neighbor z_j is greater than i and (i, j) is an edge of G.*

Proof. Assume to the contrary that $z_j \le i$ for every vertex j, $j < i$. Since no vertex less than i is adjacent to vertices greater than i and i is an absorbed vertex, G is disconnected. This is a contradition. Therefore, there must exist a vertex j such that $j < i < z_j$. By Property 2, i is adjacent to j or z_j. Since all of i's neighbors are less than i, i must be adjacent to j. Thus, (i, j) is an edge of G.

Q.E.D.

Theorem 6. *Algorithm A constructs a spanning tree on trapezoid graphs in $O(\log n)$ time using $O(n / \log n)$ processors on the EREW PRAM model.*

Proof. The time complexity of Algorithm A is described in Section 2. All we need to show is that the graph constructed by our algorithm is acyclic with $n-1$ edges. Let z_i be the maximum neighbor of i for each vertex i, $1 \le i \le n-1$. By Lemma 4, the graph T composed of edges (i, z_i) for all normal vertices i is acyclic. Let j be an absorbed vertex. By Lemma 5, there must exist a vertex k, $k < j$, such that $z_k > j$ and (j, k) is an edge. We then insert edge (j, k) into T. Since we add a new vertex with a new edge into T, T is still acyclic. After all absorbed vertices (except vertex n) with their incident edges are inserted into T, T remains acyclic. Therefore, T is acyclic with $n-1$ edges. This completes the proof.

Q.E.D.

4. Conclusion

Extending algorithms from permutation graphs to trapezoid graphs is an interesting study. In this paper, we use prefix computation technique to design a parallel algorithm for constructing a spanning tree on trapezoid graphs. Our algorithm runs in $O(\log n)$ time with $O(n / \log n)$ processors on the EREW PRAM model.

Reference

[1] K. Arvind, V. Kamakoti, and C. P. Rangan, Efficient Parallel Algorithms for Permutation Graphs, *Journal of Parallel and Distributed Computing*, Vol. 26, 1995, pp. 116-124.

[2] L. Chen, Solving the Shortest-Paths Problem on Bipartite Permutation Graphs Efficiently, *Information Processing Letters*, Vol. 55, 1995, pp. 259-264.

[3] D. G. Corneil and P. A. Kamula, Extensions of Permutation and Interval Graphs, *Congressus Numerantium*, Vol. 58, 1987, pp. 267-275.

[4] I. Dagan, M. C. Golumbic, and R. Y. Pinter, Trapezoid Graphs and Their Coloring, *Discrete Applied Mathematics*, Vol. 21, 1988, pp. 35-46.

[5] R. Gould, *Graph Theory*, Benjamin Cummings., 1988.

[6] O. H. Ibarra and Q. Zheng, An Optimal Shortest Path Parallel Algorithm for Permutation Graphs, *Journal of Parallel and Distributed Computing*, Vol. 24, 1995, pp. 94-99.

[7] J. JáJá, *Introduction to Parallel Algorithms*, Addison-Wesley, 1992.

[8] C. P. Kruskal, L. Rudolph and M. Snir, The Power of Parallel Prefix, *IEEE Transactions on Computers*, Vol. 34, 1985, pp. 965-968.

[9] Y. D. Liang, Dominations in Trapezoid Graphs, *Information Processing Letters*, Vol. 52, 1994, pp. 309-315.

[10] Y. D. Liang, Steiner Set and Connected Domination in Trapezoid Graphs, *Information Processing Letters*, Vol. 56, 1995, pp. 101-108.

[11] T. H. Ma, On the 2-Chain Subgraph Cover and Related Problems, *Journal of Algorithms*, Vol. 17, 1994, pp. 251-268.

[12] E. F. Moore, The Shortest Path Through a Maze. *Proceeding of International Symposium on Switching Theory*, 1957, Part II, Harvard University Press, Cambridge, 1959, pp. 285-292.

[13] C. Rhee, Y. D. Liang, S. K. Dhall and S. Lakshmivarahan, Efficient Algorithms for Finding Depth-First and Breadth-First Search Trees in Permutation Graphs, *Information Processing Letters*, Vol. 49, 1994, pp. 45-50.

[14] Y. L. Wang, H. C. Chen, and C. Y. Lee, An $O(\log n)$ Parallel Algorithm for Constructing a Spanning Tree on Permutation Graphs, *Information Processing Letters*, Vol. 56, 1995, pp. 83-87.

[15] Y. L. Wang, K. C. Chiang, and M. S. Yu, Optimal Algorithms for Interval Graphs, *1994 International Computer Symposium Conference Proceedings*, 1994, pp. 19-24.