数理解析研究所講究録 1205 巻 2001 年 37-42

A parallel algorithm for finding all hinge vertices of a trapezoid graph

本間宏利 Hirotoshi Honma† 増山繁 Shigeru Masuyama‡

† 釧路高専 情報工学科
Department of Information Engineering,
Kushiro National College of Technology

‡ 豊橋技術科学大学 知識情報工学系
Department of Knowledge-Based Information Engineering,
Toyohashi University of Technology

Keywords: Parallel algorithms; Trapezoid graphs; Hinge vertices; Shortest paths

1 Introduction

Given a simple undirected graph $G = (V, E)$ with vertex set V and edge set E, let $G - u$ be a subgraph induced by the vertex set $V - u$. We define the distance $d_G(x, y)$ as the length of the shortest path between vertices x and y in G. Chang et al.[1] defined that $u \in V$ is a *hinge vertex* if there exist two vertices $x, y \in V - \{u\}$ such that $d_{G-u}(x, y) > d_G(x, y)$. A graph without hinge vertices is called a *self-repairing graph*[2]. Farley et al. presented a constructive characterization related to the class of self-repairing graph. An *articulation vertex* is a special case of a hinge vertex in the sense that removal of an articulation vertex u changes the finite distance of some nonadjacent vertices x, y to infinity. For the design and analysis of distributed networks, the computation of topological properties is a very important research topic. The overall communication cost in a network is increased if a computer corresponding to a hinge vertex stalls. Therefore, finding the set of hinge vertices in a graph can be used to identify critical nodes in an actual network. A number of studies concerning hinge vertices have been made in recent years. There exists a trivial $O(n^3)$ sequential algorithm for finding all hinge vertices of a simple graph by a result in Ref. [1], e.g., Theorem 1 in this letter. In general, it is known that more efficient sequential or parallel algorithms can be developed by restricting classes of graphs. For instance, Chang et al. presented an $O(n + m)$ time algorithm for finding all hinge vertices of a strongly chordal graph[1]. Ho et al. presented a linear time algorithm for finding all hinge vertices of a permutation graph[3]. Recently, we presented a parallel algorithm which runs in $O(\log n)$ time with $O(n)$ processors, for finding all hinge vertices of an interval graph[6]. In this paper, we shall propose a parallel algorithm which runs in $O(\log n)$ time with $O(n)$ processors on CREW PRAM (Concurrent-Read Exclusive-Write Parallel Random Access Machine) for finding all hinge vertices of a *trapezoid graph*[4].

2 Definition

We first illustrate the *trapezoid diagram* before defining the trapezoid graph. There are two horizontal lines, called the *top channel* and the *bottom channel*, respectively. Each channel is labeled with consecutive integer values 1, 2, ..., $2n$ (where n is the number of trapezoids). A *trapezoid* T_i is defined by four corner points $[a_i, b_i, c_i, d_i]$ where a_i, b_i ($a_i < b_i$) lie on the top channel and c_i, d_i ($c_i < d_i$) lie on the bottom channel, respectively. Without loss of generality, we assume that each trapezoid has four corner points and all corner points are distinct[9]. The geometric representation described above is called a trapezoid diagram D[9]. We assume that trapezoids are labeled in increasing order of their corner points b_i's, i.e., $T_i < T_j$ if $b_i < b_j$.

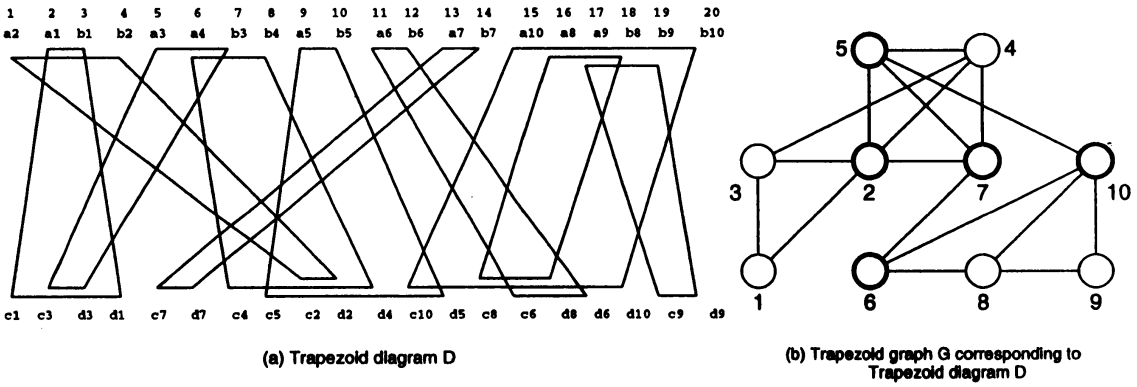

図 1: Trapezoid diagram D and graph G

An undirected graph $G = (V, E)$ is called a trapezoid graph if there exists a trapezoid diagram D satisfying

V={ i | vertex i corresponds to trapezoid T_i },

E={ (i, j) | trapezoids T_i and T_j intersect in trapezoid diagram D }[9].

Fig. 1 illustrates a trapezoid diagram D and a trapezoid graph G corresponding to D. All hinge vertices of this trapezoid graph are vertices 2, 5, 6, 7 and 10.

The class of trapezoid graphs includes two well-known classes, the class of interval graphs[4], and that of *permutation graphs*[4]. The former is obtained by setting $a_i = c_i$ and $b_i = d_i$ for all$i$, and the latter is obtained by setting $a_i = b_i$ and $c_i = d_i$ for all i, respectively. When w has a larger vertex (resp., trapezoid) number than that of vertex (resp., trapezoid) v, then we say that w is larger than v when no confusion may arise.

In the following, terminologies used in this paper are introduced.

Let v be a vertex in a trapezoid graph G. Let $N(v)$ be $\{w \mid v, w \in V, (v, w) \in E\}$.

$TR(v)$ (resp., $BR(v)$) is the largest vertex $w \in V$ satisfying $b_w > b_v$ (resp., $d_w > d_v$), where T_w intersects with T_v. Similary, $STR(v)$ (resp., $SBR(v)$) is the second-largest vertex w under the same condition as $TR(v)$ (resp., $BR(v)$). $TL(v)$ (resp., $BL(v)$) is the smallest vertex $w \in V$ satisfying

$a_w < a_v$ (resp., $c_w < c_v$), where T_w intersects with T_v. Similary, $STL(v)$ (resp., $SBL(v)$) is the second-smallest vertex w under the same condition as $TL(v)$ (resp., $BL(v)$). When such a vertex does not exist for some definition, let e.g., $TR(v) = v$, $STR(v) = v$,..., $SBL(v) = v$. We define *detect vertex sets* as $D_{TR}(v) = \{i \mid b_{STR(v)} < i < b_{TR(v)}\}$, $D_{TL}(v) = \{i \mid a_{TL(v)} < i < a_{STL(v)}\}$ and $D_{BR}(v) = \{i \mid d_{SBR(v)} < i < d_{BR(v)}\}$, respectively. In addition, we define *represent vertex sets* $P(TR)$, $P(TL)$, $P(BR)$ as follows. First, let $x_1 < x_2 < ... < x_m$ be different values among $TR(v)$'s, $v \in V$ and we divide V into vertex sets $V_1, V_2, ..., V_m$, where $V_j = \{v \mid TR(v) = x_j\}$. Next, we find v_j for each V_j such that $|D_{TR(v_j)}| \geq |D_{TR(v)}|, \forall v \in V_j$. Here, we do not select v_j such that $D_{TR(v_j)} = \emptyset$. We define $P(TR)$ as a set consisting of all vertices v_j, $j = 1, 2, ..., m$. $P(TL)$, $P(BR)$ is defined similarly. Represent vertex sets of G in Fig. 1(b) are $P(TR) = \{1, 2, 5\}$, $P(TL) = \{7, 8, 10\}$, $P(BR) = \{1, 4, 5, 7, 8\}$.

3 Properties of the hinge vertices in a trapezoid graph

Theorem 1 due to Chang et al. [1] characterizes the hinge vertices of a simple graph. We apply this theorem for finding efficiently the hinge vertices of a trapezoid graph.

Theorem 1 *For a graph $G = (V, E)$, a vertex $u \in V$ is a hinge vertex of G if and only if there exist two nonadjacent vertices $x, y \in N(u)$ such that u is the only vertex adjacent with both x and y in G.* □

When the condition in this theorem is satisfied, we say that u is the hinge vertex of x and y. In the following, we shall describe lemmas characterizing the hinge vertices on a trapezoid graph.

Lemma 1 *Let u be a hinge vertex of a trapezoid graph G. Assume that $x, y (x < y) \in N(u)$ are two vertices satisfying $d_{G-u}(x, y) > d_G(x, y)$. Then, the following three conditions hold.*

(1) If $a_y < b_u$, then $b_{STR(x)} < a_y < b_{TR(x)}$, i.e., $a_y \in D_{TR}(x)$.

(2) If $b_u < a_y$ and $a_u < b_x$, then $a_{TL(y)} < b_x < a_{STL(y)}$, i.e., $b_x \in D_{TL}(y)$.

(3) If $b_x < a_u$ and $b_u < a_y$, then $d_{SBR(x)} < c_y < d_{BR(x)}$, i.e., $c_y \in D_{BR}(x)$.

Lemma 2 *A vertex u is a hinge vertex of G if and only if there exist two vertices x,y satisfying the following condition.*

(Case 1) $u = TR(x)$, $a_y \in D_{TR}(x)$: If $BR(x) = TR(x)$, then $d_{SBR(x)} < c_y$. Otherwise, $d_{BR(x)} < c_y$.

(Case 2) $u = TL(y)$, $b_x \in D_{TL}(y)$: If $BL(y) = TL(y)$, then $d_x < c_{SBL(y)}$. Otherwise, $d_x < c_{BL(y)}$.

(Case 3) $u = BR(x)$, $c_y \in D_{BR}(x)$: If $BR(x) = TR(x)$, then $b_{STR(x)} < a_y$. Otherwise, $b_{TR(x)} < a_y$.

Lemma 3 *Let $G = (V, E)$ be a trapezoid graph. For vertices $v, w \in V$, the following condition holds.*

(Case 1) $v \in P(TR)$, $w \in V$: If $TR(w) = TR(v)$, then $D_{TR}(w) \subseteq D_{TR}(v)$.

(Case 2) $v \in P(TL)$, $w \in V$: If $TL(w) = TL(v)$, then $D_{TL}(w) \subseteq D_{TL}(v)$.

(Case 3) $v \in P(BR)$, $w \in V$: If $BR(w) = BR(v)$, then $D_{BR}(w) \subseteq D_{BR}(v)$.

Lemma 4 *Let $G = (V, E)$ be a trapezoid graph with n vertices.*

(1) $\sum_{v \in P(TR)} \mid D_{TR}(v) \mid \leq 2n$.

(2) $\sum_{v \in P(TL)} \mid D_{TL}(v) \mid \leq 2n$.

(3) $\sum_{v \in P(BR)} \mid D_{BR}(v) \mid \leq 2n$.

Lemma 4-(1) implies that we need to search a_y, $a_y \in D_{TR}(x)$ for deciding whether a given vertex $u = TR(x)$ is a hinge vetex or not. However, it suffices to consider only $x \in P(TR)$, and the number of times for searching is at most $2n$.

Algorithm PHV

Input: The arrays of corner points $a[1..n], b[1..n], c[1..n], d[1..n]$ of trapezoids in the trapezoid diagram corresponding to the given trapezoid graph.

Output: The set of hinge vertices.

Step 1 (Construction of $TR[1:n]$,...,$SBR[1:n]$)

(1) Make array $TR[1:n]$

for all i, $1 \leq i \leq n$, **in parallel do** $MTR[i] := min\{a[i], a[i+1], ..., a[n]\}$

(2) Make array $TR_1[1..n]$

for all i, $1 \leq i \leq n$, **in parallel do** $TR_1[i] := i$

for all i, $1 \leq i \leq n-1$, **in parallel do**

if $b[i] > MTR[n]$ **then** $TR_1[i] := n$

else $TR_1[i] := j - 1$ for the smallest $j(> i)$ such that $b[i] < MTR[j]$

(3) Re-computation of array $MTR[1..n]$

for all i, $1 \leq i \leq n$, **in parallel do** $MTR[i] := min\{c[i], c[i+1], ..., c[n]\}$

(4) Make array $TR_2[1..n]$

for all i, $1 \leq i \leq n$, **in parallel do** $TR_2[i] := i$

for all i, $1 \leq i \leq n-1$, **in parallel do**

if $d[i] > MTR[n]$ **then** $TR_2[i] := n$

else $TR_2[i] := j - 1$ for the smallest $j(> i)$ such that $d[i] < MTR[j]$

(5) Make array $TR[1..n]$

for all i, $1 \leq i \leq n$, **in parallel do** $TR[i] := max\{TR_1[i], TR_2[i]\}$

Similarly, compute $TL[1..n]$, $BR[1..n]$, $BL[1..n]$, $STR[1:n]$, $STL[1..n]$, $SBR[1..n]$ and $SBL[1..n]$.

Step 2 (Construction of represent vertex sets $P(TR)$, $P(TL)$, $P(BR)$)

$P(TR)$ consists of elements index i such that $TR[i]$ has different values. Here, if there exists some $TR[i]$'s which have the same value, select i such that $STR[i]$'s value is the minimum.

This Step is realized by applying sorting in lexicographic order for $TR[1..n]$, $STR[1..n]$.

Similarly, compute $P(TL)$, $P(BR)$.

Step 3 (Construction of detect vertex set $D_{TR}[1:n]$, $D_{TL}[1:n]$, $D_{BR}[1:n]$)

for all i, $i \in P(TR)$, **in parallel do** $D_{TR}[i] = \{k \mid b[STR[i]] < k < b[TR[i]], k \in \mathbf{N}\}$

Similarly, compute $D_{TL}[1:n]$, $D_{BR}[1:n]$.

Step 4 (Finding of hinge vertices)

for all i, $i \in P(TR)$, **in parallel do**

for vertices y, $a_y \in D_{TR}[i]$,

if ($BR[i] = TR[i]$ and $d_{SBR[i]} < c_y$) or ($BR[i] \neq TR[i]$ and $d_{BR[i]} < c_y$)

then $TR[i]$ is a hinge vertex.

for all i, $i \in P(TL)$, **in parallel do**

for vertices x, $b_x \in D_{TL}[i]$,

if ($BL[i] = TL[i]$ and $d_x < c_{SBL[i]}$) or ($BL[i] \neq TL[i]$ and $d_x < c_{BL[i]}$)

then $TL[i]$ is a hinge vertex.

for all i, $i \in P(BR)$, **in parallel do**

for vertices y, $c_y \in D_{BR}[i]$,

if ($BR[i] = TR[i]$ and $b_{STR[i]} < a_y$) or ($BR[i] \neq TR[i]$ and $b_{TR[i]} < a_y$)

then $TR[i]$ is a hinge vertex.

End of Algorithm PHV

We shall describe details of algorithm PHV and analyze the complexity. Algorithm PHV finds all hinge vertices of an interval graph based on the necessary and sufficient condition for a hinge vertex described in Lemma 3.

In Step 1, we obtain the largest trapezoid $TR[i]$ which intersects trapezoid T_i in parallel for i, $1 \leq i \leq n$. $MTR[1:n]$ is calculated in $O(\log n)$ time using $O(n/\log n)$ processors by applying parallel prefix computation[5][7]. $TR_1[1..n]$ and $TR_2[1..n]$ are calculated in $O(\log n)$ time using $O(n)$ processors by applying binary sort in parallel, respectively.

In Step 2, we obtain represent vertex sets. The sorting in lexicographic order is realized by executing the parallel sorting twice. The parallel sorting can be implemented in $O(\log n)$ time using $O(n)$ processors.

In Step 3, we obtain detect vertex sets. It can be implemented in $O(\log n)$ time with $O(n/\log n)$ processors by applying Brent's scheduling principle [5][7].

In Step 4, we decide whether $TR[i]$ is a hinge vertex or not by checking whether there exist some common elements in detect vertex set $D_{TR}[i]$ and array $a[1:n]$ for i selected by Step 4. Binary search can be applied in parallel by sorting $a[1:n]$ in advance. A parallel sorting computation takes $O(\log n)$ time with $O(n)$ processors [8].

EREW PRAM is sufficient for achieving the complexity in Steps 1-(1) and 3. However, CRCW PRAM is required for parallel binary search executed in Steps 1-(2) and 4, respectively. Therefore Algorithm PHV is implemented on CREW(Concurrent Read Exclusive Write) model. Hence we have the following theorem.

Theorem 2 *Given a trapezoid graph G, Algorithm PHV finds the set of all hinge vertices of G in $O(\log n)$ time using $O(n)$ processors on CREW PRAM.* □

4 Concluding remarks

In this paper, we proposed a parallel algorithm which runs in $O(\log n)$ time with $O(n)$ processors on CREW PRAM for finding all hinge vertices of a trapezoid graph. Extending the results to other graphs is left for future research.

参考文献

[1] J.M. Chang, C.C. Hsu, Y.L. Wang and T.Y. Ho, Finding the Set of All Hinge Vertices for Strongly Chordal Graphs in Linear Time, *Information Sciences* **99** (1997) 173-182.

[2] A.M. Farley and A. Proskurowski, Self-repairing networks, *Parallel Processing Lett.* **3** (1993) 381-391.

[3] T.-Y. Ho, Y.-L. Wang and M.-T. Juan, A linear time algorithm for finding all hinge vertices of a permutation graph, *Inf. Process. Lett.* **59** (1996) 103-107.

[4] M. C. Golumbic, *Algorithmic Graph Theory and Perfect Graphs*, Academic Press, New York (1988).

[5] A. Gibbons and W. Rytter, *Efficient parallel algorithms*, Cambridge University Press (1988).

[6] H. Honma and S. Masuyama, A parallel algorithm for finding all hinge vertices of an Interval graph, to appear in IEICE Trans. Information and Systems.

[7] J. JáJá, *An Introduction to parallel algorithms*, Addison-Wesley Publishing Company (1992).

[8] R. Cole, Parallel merge sort, *SIAM J. Computing* **17** (1988) 770-785.

[9] Y. D. Liang, Dominations in trapezoid graphs, *Inf. Process. Lett.* **52** (1994) 309-315