2000年度日本オペレーションズ・リサーチ学会
秋季研究発表会

台形グラフにおける全関節節点を求める並列アルゴリズム

01506161 釧路高専　本間 宏利　HONMA Hirotoshi
01603863 豊橋技術科学大学　増山 繁 MASUYAMA Shigeru

1. はじめに

$G=(V,E)$ を単純無向グラフとし，$G-u$ を節点集合 $V-u$ から誘導される G の部分グラフとする．また，$d_G(x,y)$ を G 上の2節点 x,y 間の最短経路の長さとする．$d_{G-u}(x,y) > d_G(x,y)$ を満たすような二つの節点 $x,y \in V-u$ が存在する時，節点 u を関節節点 (*hinge vertex*) と定義する [1]．2000年度OR学会春季研究発表会では区間グラフにおける全関節節点を求める並列アルゴリズム [4] を紹介した．本研究ではその時に課題として挙げていた区間グラフの拡張概念を持つ台形グラフ [3] に対して，全関節節点を求めるプロセッサ数 $O(n)$，計算時間 $O(\log n)$ で実行可能な効率的な並列アルゴリズムを紹介する．

2. 定義

上下にそれぞれ top channel，bottom channel と呼ばれる2本の平行な水平線分が存在し，それぞれの線分を上底，下底として構成される四角形を台形と定義する．各 channel は左側から昇順に $1,2,...,2n$ なる整数値が割り付けられ，各台形 $T_i(1 \le i \le n)$ は4つの角点にそれぞれ角点番号 a_i,b_i,c_i,d_i を所持する．ここで，$a_i,b_i(a_i < b_i)$ は top channel 上の，c_i,d_i $(c_i < d_i)$ は bottom channel 上の点である．また，各台形は角点を共有することはない．このような台形の集合によって表現される幾何学的表現を台形ダイアグラム D と定義する (図1)．台形ダイアグラム上の各台形には，2つの台形 T_i,T_j に対して，$b_i < b_j$，かつ，その時に限り $i<j$ となるように台形番号 i, $1 \le i \le n$ が付される．

あるグラフ $G=(V,E)$ が台形グラフである必要十分条件は，台形ダイアグラム D 上の各台形が T の節点集合 V の要素に一対一対応し，かつ，2つの台形 T_i,T_j が台形ダイアグラム上で交差を持つとき，かつその時に限り，各台形の対応する節点間に辺が存在するような台形ダイアグラム D が存在することである．すなわち，

$G=(V,E)$,
$V=\{i \mid$ 節点 i は台形 T_i に対応する $\}$,
$E=\{(i,j) \mid$ 台形 T_i と T_j は台形ダイアグラム D で交差する $\}$.

また，台形グラフ G の各節点には，台形ダイアグラム D 上の対応する各台形の台形番号がそのまま節点番号として割り付けられる (図2)．

$TR(v), STR(v)(BR(v), SBR(v))$ の定義:
台形ダイアグラム上で節点 v に対応する台形 T_v と交差を持ち，角点番号 $b(d)$ が $b_v(d_v)$ よりも大きな台形の中で，最大の角点番号 $b(d)$ を持つ台形に対応する節点を $TR(v)(BR(v))$，同様に2番目に大きな節点を $STR(v)(SBR(v))$ と定義する．ただし，そのような節点が存在しない場合はそれぞれ $TR(v)=v$, $STR(v)=v(BR(v)=v, SBR(v)=v)$ とする．
$TL(v), STL(v)(BL(v), SBL(v))$ の定義:
台形 T_v と交差を持ち，角点番号 $a(c)$ が $a_v(c_v)$ よりも小さな台形の中で，最小の角点番号 $a(c)$ を持つ台形に対応する節点を $TL(v)(BL(v))$，同様に2番目に小さな節点を $STL(v)(SBL(v))$ と定義する．ただし，そのような節点が存在しない場合はそれぞれ $TL(v)=v$, $STL(v)=v(BL(v)=v, SBL(v)=v)$ とする．
判別集合 $D_{TR}(v), D_{TL}(v), D_{BR}(v)$ の定義:
台形ダイアグラム上の台形 T_v に対して，$D_{TR}(v)=\{i \mid b_{STR(v)} < i < b_{TR(v)}\}$, $D_{TL}(v)=\{i \mid a_{TL(v)} < i < a_{STL(v)}\}$, $D_{BR}(v)=\{i \mid d_{SBR(v)} < i < d_{BR(v)}\}$ をそれぞれ判別集合 D_{TR}, D_{TL}, D_{BR} と定義する (表1)．

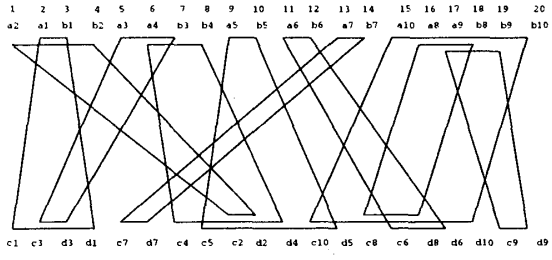

図 1: 台形ダイアグラム D

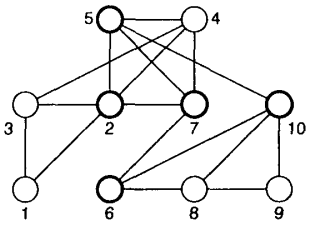

図 2: 台形グラフ G

3. 台形グラフにおける関節節点の性質

Lemma 1　節点 u が台形グラフ T の関節節点である必要十分条件は以下の条件のうちいずれかを満たすような2つの節点 $x,y(x<y)$ が存在することである．

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

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

(3) $u=BR(x)$, $c_y \in D_{BR}(x)$. *If* $BR(x)=TR(x)$ *then* $b_{STR(x)} < a_y$ *else* $b_{TR(x)} < a_y$. □

Lemma 2　グラフ $G=(V,E)$ を台形グラフとする．代表節点集合に対して以下の性質が成り立つ．

(1) $v \in P(TR)$ なる v に対して，$TR(v)=TR(w)$ なる節点 w が存在するなら，$D_{TR}(v) \supseteq D_{TR}(w)$.

(2) $v \in P(TL)$ なる v に対して，$TL(v)=TL(w)$ なる節点 w が存在するなら，$D_{TL}(v) \supseteq D_{TL}(w)$.

(3) $v \in P(BR)$ なる v に対して，$BR(v)=BR(w)$ なる節点 w が存在するなら，$D_{BR}(v) \supseteq D_{BR}(w)$. □

Lemma 3　グラフ $G=(V,E)$ を台形グラフとする．代表節点集合に対して以下の性質が成り立つ．

(1) $\sum_{v \in P(TR)} \mid D_{TR}(v) \mid \le 4n$.

(2) $\sum_{v \in P(TL)} \mid D_{TL}(v) \mid \le 4n$.

(3) $\sum_{v \in P(BR)} \mid D_{BR}(v) \mid \le 4n$. □

表 1: 各台形 T_i の角点番号

i	1	2	3	4	5	6	7	8	9	10
a_i	2	1	5	6	9	11	13	16	17	15
b_i	3	4	7	8	10	12	14	18	19	20
c_i	1	9	2	7	8	15	5	14	19	12
d_i	4	10	3	11	13	17	6	16	20	18
TR	3	7	4	7	10	10	7	10	10	10
STR	2	5	3	5	7	8	7	9	9	10
TL	2	2	2	2	2	6	2	6	10	5
STL	1	2	1	3	4	6	4	10	8	6
BR	2	5	4	5	10	10	6	9	9	9
SBR	1	4	2	4	5	6	5	10	9	10
BL	1	1	1	3	7	7	7	10	10	5
SBL	1	3	3	7	4	10	7	8	8	10
D_{TR}	5,6	11,12,13	ϕ	11,12,13	15,16,17,18,19	19	ϕ	ϕ	ϕ	ϕ
D_{TL}	ϕ	ϕ	ϕ	2,3,4	2,3,4,5	ϕ	2,3,4,5	12,13,14	ϕ	10
D_{BR}	5,6,7,8,9	12	ϕ	12	14,15,16,17	ϕ	14,15,16	19	ϕ	19

4. アルゴリズム

Algorithm PHT

Input: 角点配列 $a[1..n], b[1..n], c[1..n], d[1..n]$

Output: 関節節点集合 S

Step 1 $TR[1:n]$, $TL[1..n]$, $BR[1..n]$, $BL[1..n]$ の構築

(1) 配列 $MTR[1..n]$ を作成する

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

$MTR[i] := min\{a[i], a[i+1], ..., a[n]\}$

(2) 配列 $TR_1[1..n]$ を作成する

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

$TR_1[i] := i$

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

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

else $b[i] < MTR[j]$ を満たす最小の $j(> i)$ に対して, $TR_1[i] := j - 1$

(3) 配列 $MTR[1..n]$ の再計算

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

$MTR[i] := min\{c[i], c[i+1], ..., c[n]\}$

(4) 配列 $TR_2[1..n]$ を作成する

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

$TR_2[i] := i$

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

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

else $d[i] < MTR[j]$ を満たす最小の $j(> i)$ に対して, $TR_2[i] := j - 1$

(5) 配列 $TR[1..n]$ を作成する

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

$TR[i] := max\{TR_1[i], TR_2[i]\}$

同様の方法で, $TL[1..n]$, $BR[1..n]$, $BL[1..n]$ を計算する

Step 2 $STR[1:n]$, $TL[1..n]$, $BR[1..n]$, $BL[1..n]$ の構築

Step 1 と同様の方法を用いる

Step 3 代表節点集合 $P(TR)$, $P(TL)$, $P(BR)$ の構築

同じ TR の内容を持つ節点グループから STR の内容が最小の節点を選択し, 代表節点集合 $P(TR)$ の要素とする. この操作は配列 $TR[1..n]$, $STR[1..n]$ に対して, $TR[1..n]$ の要素を優先的に辞書式順序に並列ソートを行うことによって実現可能である. 同様の方法で, $P(TL)$, $P(BR)$ を計算する.

Step 4 判別集合 $D_{TR}[1:n]$, $D_{TL}[1:n]$, $D_{BR}[1:n]$ の構築

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

$D_{TR}[i] = \{k \mid b[STR[i]] < k < b[TR[i]], k \in \mathbf{N}\}$

同様の方法で, $D_{TL}[1:n]$, $D_{BR}[1:n]$ を計算する

Step 5 関節節点の導出

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

$a_y \in D_{TR}[i]$ なる節点 y に対して, $BR[i] == TR[i]$ ならば $d_{SBR[i]} < c_y$ が, そうでなければ $d_{BR[i]} < c_y$ が成立するならば, $TR[i]$ を S に加える.

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

$b_x \in D_{TL}[i]$ なる節点 x に対して, $BL[i] == TL[i]$ ならば $d_x < c_{SBL[i]}$ が, そうでなければ $d_x < c_{BL[i]}$ が成立するならば, $TL[i]$ を S に加える.

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

$c_y \in D_{BR}[i]$ なる節点 y に対して, $BR[i] == TR[i]$ ならば $b_{STR[i]} < a_y$ が, そうでなければ $b_{TR[i]} < a_y$ が成立するならば, $BR[i]$ を S に加える.

End of Algorithm PHT

アルゴリズムの解説と計算量の解析を行う. Step 1,2 は $TR[i]$ の導出に配列 $MTR[1:n]$ を用いているが, これは並列プレフィックス演算により $O(n/log\ n)$ 個のプロセッサを用いて $O(log\ n)$ 時間で計算可能である. $MTR[i]$ の計算では $b[i] < MTR[j]$ になるような最小の j を必要としているが, 配列 $MTR[1:n]$ が昇順にソートされている特性を利用して 2 分探索を用いることで $O(n)$ 個のプロセッサを用いて $O(log\ n)$ 時間で $TR[1:n]$ は計算可能である. Step 3 では並列ソートを 2 回実行すればよいので, $O(n)$ 個のプロセッサを用いて $O(log\ n)$ 時間で計算可能である. Step 4 では Brent のスケジューリング原理を適用すると $O(n/log\ n)$ 個のプロセッサを用いて $O(log\ n)$ 時間で計算可能である. Step 5 では配列 $a[1:n]$ の要素をソートすることによって判別集合 $D_{TR}[i]$ の各要素に対して並列に 2 分探索が適用可能である. これらの並列技法は代表的な並列計算機モデルである CREW PRAM 計算機モデル上で実行可能である.

Theorem 1 台形グラフ $G = (V, E)$ において, 全関節節点を求める $O(n)$ 個のプロセッサによる $O(log\ n)$ 時間の並列アルゴリズムを *CREW PRAM* 計算機上で構築することが可能である. □

参考文献

[1] J.M. Chang, An efficient algorithm for finding the set of all hinge vertices in strongly chordal graphs, *Proc, Internat. Comp. Symp.* (1994) 277-282.

[2] Ting-Yem Ho, A linear time algorithm for finding all hinge vertices of a permutation graph, *Inf. Proc. Let.* **59** (1996) 103-107.

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

[4] 本間, 増山, 区間グラフにおける全関節節点を求める並列アルゴリズム, 日本 OR 学会春季研究発表会 (2000) 92-93.