1997年度日本オペレーションズ・リサーチ学会
春季研究発表会

台形グラフにおける全域林構築のための $O(\log n)$ 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

全域林 (*spanning forest*) 構築問題とは，与えられたグラフ G の各連結成分に対して，全域木を構築する問題である．本研究ではグラフの形状を台形グラフ (*trapezoid graph*)[1][3] に制限し，$O(\log n)$ 時間，$O(n)$ プロセッサ数で実行される EREW PRAM 計算機モデル上での全域林構築のための効率的並列アルゴリズムを開発した．

台形グラフの定義について述べる．上部チャネル，下部チャネルと呼ばれる 2 本の水平な平行線分上に左から右に連続する整数値で番号付けされている点が存在すると仮定し，台形 T_i を上部チャネル，下部チャネル上の 4 つの点 $[a_i, b_i, c_i, d_i]$ を角点として持つ図形と定義する．このように定義される台形の幾何学的表現を台形ダイアグラム (*trapezoid diagram*)T といい，図 1に 17 個の台形から構成される台形ダイアグラムの例を示す．この台形ダイアグラムに基づいて，$b_i < b_j$ ならば $T_i < T_j$の関係を持つように，各台形は $T_1, T_2, T_3, ..., T_n$と番号付けられる．グラフ $G = (V, E)$ が台形グラフである必要十分条件は，次のような台形ダイアグラム Tが存在することである．
$V = \{\ i \mid$ 節点 i は台形 T_iに対応する $\}$
$E = \{\ (i, j) \mid$ 台形 T_i と T_j が台形ダイアグラム T で交差している $\}$[1]

2　並列アルゴリズム

台形グラフにおける全域林構築のための並列アルゴリズムの概略を示す．この並列アルゴリズムは入力として，台形ダイアグラム上の各台形における角点および，角点に割り付けられた整数値（角点番号）の配列を与え，出力として，その台形ダイアグラムに対応する台形グラフの全域林 F^*を返す．

最初に，Step 1 では与えられた入力から，台形番号の昇順に対応するような各角点 (a, b, c, d) の角点番号の配列を生成する．Step 2 では，以降の手続きを効率的に処理するための前処理計算を行う．具体的には Step 1 で生成した配列に対して，並列プレフィクス演算 [2] を行う．Step 3,4 では Step 2 で求めた計算結果を参照し，並列ポインタジャンピング [2] を適用して，台形ダイアグラムの上部チャネル上および下部チャネル上で交差している隣接する台形を求め，これらの台形間に対応する辺を台形グラフの全域林の辺として F^*に加える．Step 4 の終了後，F^*はいくつかの木の集合によって構成されている．最後に Step 5 で，これらの木を並列に併合することによって台形グラフの全域林を構築する．

以下に全域林構築のための並列アルゴリズム CSF(Construction of Spanning Forest) の詳細を記述する．

Algorithm CSF
Input : 上部チャネル上の角点および角点番号の配列 $T_T[1:2n]$, $P_T[1:2n]$. 下部チャネル上の角点および角点番号の配列 $T_B[1:2n]$, $P_B[1:2n]$.
Output : 台形グラフ G の全域林 F^*, ただし，初期値として F^*は n 個の節点だけをもつ空グラフとする．

(Step 1)
(1)　整数 i,$1 \le i \le 2n$ に対して並列に，$T_T[i]$ が角点 a_jであれば，a_jの角点番号 $P_T[i]$ を $P_a[j]$ へ格納する．$T_T[i]$ が角点 b_jであれば，b_jの角点番号 $P_T[i]$ を $P_b[j]$ へ格納する．
(2)　整数 i,$1 \le i \le 2n$ に対して並列に，$T_B[i]$ が角点 c_jであれば，c_jの角点番号 $P_B[i]$ を $P_c[j]$ へ格納する．$T_B[i]$ が角点 d_jであれば，d_jの角点番号 $P_B[i]$ を $P_d[j]$ へ格納する．
(Step 2)
(1) 節点番号 i,$1 \le i \le n$ に対して並列に，$\{P_a[n], P_a[n-1], ..., P_a[i]\}$ の要素の最小値を $L_a[i]$ に格納する．
(2) 節点番号 i,$1 \le i \le n$ に対して並列に，$\{P_c[n], P_c[n-1], ..., P_c[i]\}$ の要素の最小値を $L_c[i]$ に格納する．
(3) 節点番号 i,$1 \le i \le n$ に対して並列に，$\{P_d[1], P_d[2], ..., P_d[i]\}$ の要素の最大値を $R_d[i]$ に格納する．
(Step 3)
配列 $C[1:n]$ を用意する．ただし，$C[i]$,$1 \le i \le n$ の初期値は '0' とする．
(1) 節点番号 i,$1 \le i \le n$ に対して並列に，$P_a[i] = L_a[i]$ であれば i へのポインタ（自己ループ）を，$P_a[i] \ne L_a[i]$ であれば $i+1$ へのポインタを $S_a[i]$ に割り当てる．次に，$S_a[i]$,$1 \le i \le n$ に対して並列に，ポインタジャンピング操作を適用する．ポインタジャンピング操作とは自身の現在のポインタをそれが指し示す先のポインタに更新する操作であり，$\log n$ 回以内の操作によって，自身に最も近い自己ループを有する要素を指すポインタに更新される．
(2) 節点番号 i,$1 \le i \le n-1$ に対して並列に，$P_b[i] > L_a[i+1]$ であれば，$C[i]$ に $S_a[i+1]$ を格納し，F^*に

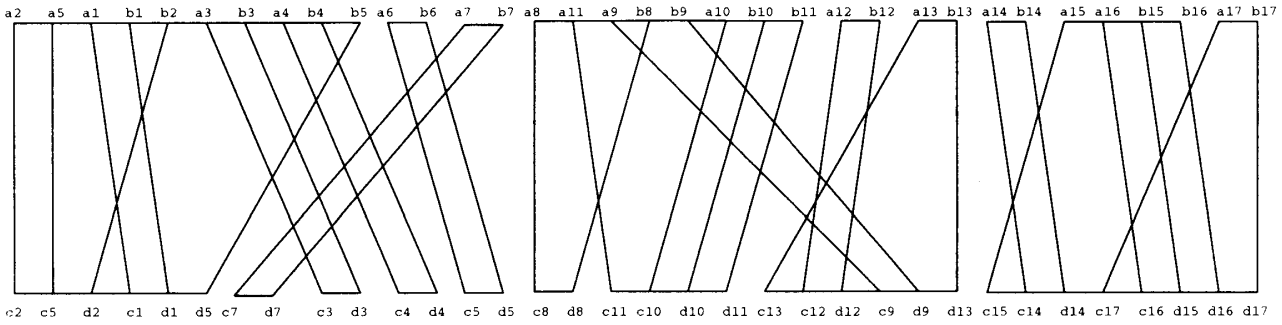

図 1: 台形ダイアグラム T

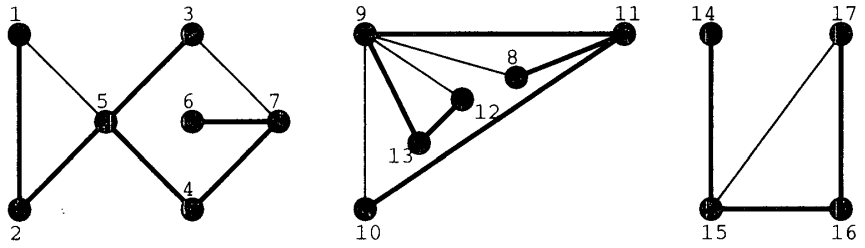

図 2: 台形グラフ G の全域林 F^*

辺 $(i, S_a[i+1])$ を追加する.

(Step 4)

(1) 節点番号 i,$1 \leq i \leq n$ に対して並列に, $P_c[i] = L_c[i]$ であれば i へのポインタを, $P_a[i] \neq L_a[i]$ であれば $i+1$ へのポインタを $S_c[i]$ に割り当てる. 次に, $S_c[i]$,$1 \leq i \leq n$ に対して並列に, ポインタジャンピング操作を適用する.

(2) 節点番号 i,$1 \leq i \leq n-1$ に対して並列に, $P_d[i] > L_c[i+1]$ かつ $C[i]$='0' であれば, $C[i]$ に $S_c[i+1]$ を格納し, F^* に辺 $(i, S_c[i+1])$ を追加する.

(Step 5)

(1) 節点番号 i,$1 \leq i \leq n$ に対して並列に, $P_d[i] = R_d[i]$ であれば i へのポインタを, $P_d[i] \neq R_d[i]$ であれば $i-1$ へのポインタを $S_d[i]$ に割り当てる. 次に, $S_d[i]$,$1 \leq i \leq n$ に対して並列に, ポインタジャンピング操作を適用する.

(2) 節点番号 i,$1 \leq i \leq n-1$ に対して並列に, $R_d[i] > L_c[i+1]$ かつ $C[i]$='0' であれば, $C[S_c[i+1]]$ に $S_d[i]$ を格納し, F^* に辺 $(S_c[i+1], S_d[i])$ を追加する.

並列アルゴリズム SCF によって構築された図 1の台形グラフ G の全域林 F^* の構築例を図 2に示す.

アルゴリズム CSF の計算量解析

Step 1-(1),(2) は Brent のスケジュール原理 [2] によって $O(n/log\ n)$ 個のプロセッサ, $O(log\ n)$ 時間で実行できる.

Step 2-(1),(2),(3) はプレフィクス演算により $O(n/log\ n)$ 個のプロセッサ, $O(log\ n)$ 時間で実行できる.

Step 3,Step 4,Step 5-(1) は並列ポインタジャンピング技法によって, $O(n)$ 個のプロセッサ, $O(log\ n)$ 時間で実行できる.

Step 3,Step 4,Step 5-(2) は Brent のスケジュール原理によって $O(n/log\ n)$ 個のプロセッサ, $O(log\ n)$ 時間で実行できる.

以上の並列計算は全て EREW PRAM 計算機モデル上で実行可能である. よって, 並列アルゴリズム CSF は EREW PRAM 計算機モデル上で $O(n)$ 個のプロセッサ, $O(log\ n)$ 時間で実行できる. ただし, n は台形グラフの節点数である.

[定理] アルゴリズム CSF は EREW PRAM 計算モデル上で, $O(n)$ 個のプロセッサを用いて, $O(log\ n)$ 時間で台形グラフの全域林を構築する.

参考文献

[1] I. Dagan, M. C. Golumbic and R. Y. Pinter, Trapezoid graphs and their coloring, *Discrete Applied Mathematics*, **21** (1988) 35-46.

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

[3] Y. D. Liang, Dominations in trapezoid graphs, *Information processing. letters.*, **52** (1994) 309-315.