1995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

外平面グラフ上の s-t 最長路を求める並列アルゴリズム

02401223 豊橋技術科学大学 *中山慎一 NAKAYAMA Shin-ichi
01603863 豊橋技術科学大学 増山 繁 MASUYAMA Shigeru

1 まえがき

$G=(V,E)$ は節点の集合 V, 辺の集合 E からなる連結無向グラフであり, 相異なる 2 節点 s, t がそれぞれ始点, 終点としてあらかじめ指定されている. なお, s, t は任意に選択することが可能である. 全ての辺 $e \in E$ は, 距離関数 $l(e) \in Z^+$(Z^+ は非負整数の集合) で非負整数の距離をもつ. 指定された始点 s から終点 t への距離の総和が最長な単純な路 (即ち, 同じ節点を 2 度以上通らない路) を s-t 最長路という. s-t 最長路を求める問題は, 一般のグラフにおいてNP困難であることが既に知られている [1] が, グラフに何らかの制限をつけた場合, 多項式時間で s-t 最長路を求める逐次アルゴリズムがいくつか知られている [4]. また, Ellis らにより, 外平面グラフにおいて, 任意に与えられた相異なる 2 節点間の最長路を求める逐次アルゴリズム [1] が与えられた. 外平面グラフとは, 全ての節点を 1 つの面の境界におくように平面埋め込み可能なグラフである.

本論文では, 外平面グラフ上の任意に与えられた相異なる 2 節点間の最長路を求める並列アルゴリズムを提案し, この問題がクラス NC[2] に属することを示す. 外平面グラフは直並列グラフの部分クラスなので, 最長路の始点, 終点の組が st- 直並列グラフとしての端点 s,t にそれぞれ一致するなら, st- 直並列グラフに対する最長路を求める並列アルゴリズム [3] を利用して, 最長路を求めることができるが, 一致しない場合には求めることができない. それに対し, 我々の並列アルゴリズムは, 始点, 終点の組を st- 直並列グラフとしての両端点に限定せず任意に選択して解くことができ, 文献 [3] の並列アルゴリズムを利用する場合よりも適用範囲が広いという利点がある.

2 準備

簡単のため, 以下では, 外平面グラフ G は 2 連結グラフ, すなわち, G より $v \in V$ を取り除くと非連結になるようなカット節点 v が存在しないグラフであると仮定する. 2 連結とは限らない一般の外平面グラフの場合についても, 2 連結成分を並列に求め [4], 各 2 連結成分に対し本並列アルゴリズムを適用することにより容易に拡張できる. 2 連結な外平面グラフ G は, 全ての節点を 1 度だけ通る閉路 H を無限面 (グラフを平面に埋め込んだ時, グラフの外側の無限に広がった領域) との境界として埋め込み, 閉路 H を構成しない残りの辺を閉路 H の内側に埋め込むような平面埋め込みを持つ. 無限面との境界をなす辺のことを周辺と呼び, 他の辺を対角辺と呼ぶ. 以下の用語は文献 [3] に従う. 2 連結な外平面グラフ G 上で, 周辺はハミルトン閉路を構成している. これを $H(G)$ で表す. $H(G)$ 上で s から t へ点素な路が 2 つ存在するが, それらをそれぞれ A, D で表す. また, 路 A(, D) 上の異なる 2 つの節点 u, v 間の路を $A(u,v)$(, $D(u,v)$) で表す. 対角辺については次の 2 種類に分類できる. 対角辺 a,b の節点 a が路 A 上, 節点 b が路 D 上にある場合, 交差弦と呼び, 節点 a, b ともに A 上または D 上にある場合, 片側弦と呼ぶ. 路 $A(u,v)$ または $D(u,v)$ 上の節点の並びで, 交差弦が隣接していない節点の並びを自由列と呼ぶ. 交差弦は s から t に平面に埋め込まれている順に順序付けられる. これらを $T_1, T_2, \cdots, T_l$ と表す. 路 A 上の節点 u, v の組に対して, 路 A 上の節点のみを通る u から v への最長路の長さを $len_A(u,v)$, 路 D 上の節点 u, v の組に対して, 路 D 上の節点のみを通る u から v への最長路の長さを $len_D(u,v)$ で示す.

3 並列アルゴリズム

並列アルゴリズムを次のように構成する. まず, 与えられた外平面グラフ G に対して, その s-t 最長路長が G の s-t 最長路長に等しくなるように非サイクル有向グラフ G^* を構成する. G の節点に接続する交差弦の数により, 最長距離になりうる単純な路の構成が異なるので, それぞれに場合分けをして G^* の有向辺, 距離を付けている. 非サイクル有向グラフの各節点間の最長路を求める並列アルゴリズム [4] は既に知られているので, これを用いて並列に G^* の s-t 最長路を求める.

以下, 非サイクル有向グラフ $G^*=(V^*,E^*)$ の構成を示す. G の交差弦 T_i, $i=1,\cdots,l$, の A 上の端点を u_i, D 上の端点を v_i とする. ある端点に複数の交差弦が接続している場合, 例えば A 上の端点 u に $T_j,\cdots,T_{j+k}$ が接続しているとすると, $u_j=\cdots=u_{j+k}=u$ になる.

G の始点 s, 終点 t, 交差弦 T_j, $j=1,\cdots,l$, の端点 u_j, v_j にそれぞれ対応して, 節点集合 $V^*=\{s^*\} \cup \{t^*\} \cup \{u_j^*\} \cup \{v_j^*\}$, $j=1,\cdots,l$, を作る.

G の交差弦 $T_{i+1}=\{u_{i+1},v_{i+1}\}$, $i=1,\cdots,l-1$, に対し, 1 つ前の交差弦 T_i との端点の関係によって場合分けし, 以下のように有向辺 $e^* \in E^*$, および全ての辺 $e^* \in E^*$ に距離 $l^*(e^*) \in Z^+$(ただし, Z^+ は正整数の集合) を付ける.

構成 [a]　T_{i+1} の端点 u_{i+1}, v_{i+1} がいずれも T_i の端点 $\{u_i, v_i\}$ と一致していない場合: 有向辺 (u_i^*, u_{i+1}^*), (v_i^*, v_{i+1}^*), (u_i^*, v_{i+1}^*), (v_i^*, u_{i+1}^*) を付け, それぞれの辺の距離を,
$l^*((u_i^*, u_{i+1}^*)) \leftarrow len_A(u_i, u_{i+1})$,
$l^*((v_i^*, v_{i+1}^*)) \leftarrow len_D(v_i, v_{i+1})$,
$l^*((u_i^*, v_{i+1}^*)) \leftarrow len_A(u_i, u_{i+1}) + l(\{u_{i+1}, v_{i+1}\})$,
$l^*((v_i^*, u_{i+1}^*)) \leftarrow len_D(v_i, v_{i+1}) + l(\{v_{i+1}, u_{i+1}\})$
とする.

構成 [b]　T_{i+1} の端点 u_{i+1}, v_{i+1} が T_i の端点 u_i, v_i と一致している場合: ここでは, A 上の端点 u に $T_i = \{y_i, v_i\}$, $T_{i+1} = \{u_{i+1}, v_{i+1}\}$ が接続しているとする. つまり, $u_i = u_{i+1} = u$ である. $T_j = \{u_j, v_j\}$ は, $u_j (\neq u)$ を端点としてもち, $j < i$ で, 最も大きい番号 j の交差弦とする.
有向辺 (v_i^*, v_{i+1}^*), (u_i^*, u_{i+1}^*), (u_i^*, v_{i+1}^*), (v_j^*, u_{i+1}^*) を付け, それぞれの辺の距離を,
$l^*((v_i^*, v_{i+1}^*)) \leftarrow len_D(v_i, v_{i+1})$,
$l^*((u_i^*, u_{i+1}^*)) \leftarrow 0$,
$l^*((v_j^*, u_{i+1}^*)) \leftarrow len_D(v_j, v_{i+1}) + l(\{v_{i+1}, u\})$,
$l^*((u_i^*, v_{i+1}^*)) \leftarrow l(\{u, v_{i+1}\})$ とする.
D 上の端点 v に $T_i = \{u_i, v_i\}, T_{i+1} = \{u_{i+1}, v_{i+1}\}$ が接続している場合 (つまり $v_i = v_{i+1} = v$): 有向辺, 距離は, 上の構成の
$u, u_i, u_{i+1}, v, v_i, v_{i+1}, v_j, u_i^*, u_{i+1}^*, v_i^*, v_{i+1}^*, v_j^*, len_D$ を
$v, v_i, v_{i+1}, u, u_i, u_{i+1}, u_j, v_i^*, v_{i+1}^*, u_i^*, u_{i+1}^*, u_j^*, len_A$ にそれぞれ置き換えた有向辺, 距離になる.

構成 [c]　s と $T_1 = \{u_1, v_1\}$, $T_l = \{u_l, v_l\}$ と t について: 有向辺 (s^*, u_1^*), (s^*, v_1^*), (u_l^*, t^*), (v_l^*, t^*) を付け, それぞれの辺の距離を,
$l^*((s^*, u_1^*)) \leftarrow \max\{\ len_A(s, u_1),\ len_D(s, v_1) + l(\{v_1, u_1\})\ \}$,
$l^*((s^*, v_1^*)) \leftarrow \max\{\ len_D(s, v_1),\ len_A(s, u_1) + l(\{u_1, v_1\})\ \}$,
$l^*((u_l^*, t^*)) \leftarrow len_A(u_l, t)$, $l^*((v_l^*, t^*)) \leftarrow len_D(v_l, t)$
とする.

以上の G^* の構成法により, 次の補題が成り立つ.

[補題 1]　構成した G^* は, 非サイクル有向グラフで, s^* は出力辺のみ, t^* は入力辺のみをもつ. □

以下に並列アルゴリズム Procedure Longest Path の概略を示す.

Procedure Longest Path
begin
(Step 1) 外平面グラフ G の周辺を求める [5].
(Step 2) G の周辺を用いて, G のハミルトン有向閉路を構成し, このハミルトン有向閉路の s から t への路 $s(= x_0)$, x_1, $\cdots$, $t(= x_k)$ を A とし, t から s への路の辺の向きを全て逆にした路 $s(= y_0)$, y_1, $\cdots$, $t(= y_{k'})$ を D とする.
(Step 3) s を番号 1, t を番号 $|V|$, 残りの節点を路 A 上, D 上でそれぞれ s から t へ昇順になるように番号付けを行なう. ただし, A 上の節点の番号は D 上の s, t 以外の節点の番号より小さいとする.
(Step 4) 対角弦 T_i, $i = 1, \cdots, l$, を求める.
(Step 5) 各片側辺 $\{u, v\}$(周辺でも対角弦でもない辺) に対し, Step 3 で付けた番号に基づき, 番号の小さい端点を始点に, 大きい端点を終点にする有向辺に置き換える.
(Step 6) Step 2, Step 5 により向き付けされた辺のみからなるグラフ G^{**}, いわゆる, 外平面グラフ G より対角弦を取り除いた有向グラフは, Step 2, Step 5 の構成より非サイクル有向グラフなので, 路 A, D の節点のみを通る x から y への最長路の長さ $len_A(x, y)$, $len_D(x, y)$, $x, y \in V$ を, 非サイクル有向グラフの各節点間の最長距離を求める並列アルゴリズム [4] を利用して求める.
(Step 7) 非サイクル有向グラフ G^* を構成して, G^* の s-t 間の最長距離を非サイクル有向グラフの各節点間の最長距離を求める並列アルゴリズム [4] を用いて求める.
end. □

以上から次の定理が得られる. この定理より, 外平面グラフ上の s-t 最長路問題がクラス NC に属することがわかる.

[定理 1]　*Procedure Longest Path* により, *CREW PRAM* 上で全体として, $O(M(n))$ 個のプロセッサを用いて $O(\log^2 n)$ 時間で外平面グラフの s-t 最長路を求めることができる. ただし, n は G の節点の個数, $M(n)$ は 2 つの $n \times n$ 行列の積を $O(\log n)$ 時間で実行するのに必要なプロセッサ数 *[4]* である. □ *(証明略)*

参考文献

[1] J. A. Ellis, M. Mata and G. MacGillivray: "A linear time algorithm for longest (s,t)-paths in weighted outerplanar graphs", *Information Processing Letters*, 32, pp.199-204, 1989.

[2] A. Gibbons and W. Rytter: *Efficient Parallel Algorithms*, Cambridge University Press, 1988.

[3] X. He and Y. Yesha: "Binary tree algebraic computation and parallel algorithms for simple graphs", *J. Algorithms* 9, pp.92-113, 1988.

[4] J. van Leeuwen, eds.: *Handbook of Theoretical Computer Science*, Elsevier Science Publishers B.V., 1990.

[5] 中山慎一, 増山繁: "外平面グラフ上の *st*- 最短経路を求める並列アルゴリズム", 1994 年度日本 OR 学会秋季研究発表会アブストラクト集, pp.240-241.