6J-02

シュタイナー森問題に対する近似アルゴリズムの実際的性能評価 [1]

菅原 惇平
中央大学大学院 理工学研究科 情報工学専攻 [2]

鮭川 矩義　　浅野 孝夫
中央大学 理工学部 情報工学科 [3]

1 はじめに

シュタイナー森問題は，辺 $e \in E$ にコスト $c_e \geq 0$ の付随する無向グラフ $G = (V,E)$ と k 組のターミナル点対 $s_i, t_i \in V$ $(s_i \neq t_i, i = 1,2,\ldots,k)$ が与えられたときに，どのターミナル点対 s_i, t_i も連結となるような辺部分集合 $F \subseteq E$ で誘導される部分グラフのなかで最小コストのものを求める問題であり，NP-困難であることが知られている．この問題に対する最初の性能保証アルゴリズムは，1991 年に Agrawal, Klein and Ravi により提案された [1]．それはシュタイナー森問題の IP 定式化の LP 緩和に基づく主双対アルゴリズムであり，性能保証は 2 である．一方，LP 緩和を用いない最初の性能保証アルゴリズム（大食アルゴリズム）が，2015 年に Gupta and Kumar により提案された [2]．本稿では，Agrawal, Klein and Ravi と Gupta and Kumar のアルゴリズムを実装して様々な入力に対する実験を行い，性能の比較・評価を与える．

2 大食アルゴリズム

Gupta and Kumar [2] で提案された**大食アルゴリズム** (gluttonous algorithm) は，シュタイナー森問題に対する LP 緩和を用いない最初の性能保証アルゴリズムであり，貪欲法に基づいている．性能保証は高々96 であることが示されているが，実際的性能には不明な点が多い．以下は，大食アルゴリズムの概要である．

大食アルゴリズムは，ターミナル点の集合を，互いに素なターミナル点の部分集合族で管理する．このとき現れる部分集合をスーパーノードと呼ぶ（各スーパーノードに含まれるすべてのターミナル点は同一の点と見なされることになる）．したがって，アルゴリズムのどの時点でも，その時点でのスーパーノードのすべての集合を $\mathscr{C}$ とすると，$\cup_{S \in \mathscr{C}} S = \cup_{i=1}^{k} \{s_i, t_i\}$ である．アルゴリズムの開始時点で，各ターミナル点がスーパーノードを形成する（すなわち，$\mathscr{C} = \{\{s_1\},\{t_1\},\{s_2\},\{t_2\},\ldots,\{s_k\},\{t_k\}\}$ である）．アルゴリズムは，2 つのスーパーノードを選択して，併合することを繰り返す．そこで，ある $i = 1,2,\ldots,k$ が存在して，$s_i \in S$ かつ $t_i \notin S$ となるスーパーノード S は**活性** (active) であると呼ぶことにする．図 1 は活性なスーパーノードと（活性でない）不活性なスーパーノードの例である．

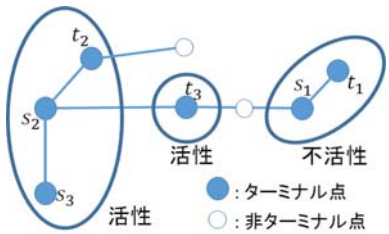

図 1　活性および不活性なスーパーノード

アルゴリズムは 2 つの活性なスーパーノードの併合（対応する 2 つの部分集合の和集合をとること）を，活性なスーパーノードがなくなるまで反復する．各反復で併合を行うと，併合されたスーパーノードに属するターミナル点間のコストを 0 とするので，更新されたスーパーノード間のコストは変化する．各反復の開始時の $\mathscr{C}$ と各スーパーノードに含まれる 2 点を結ぶ辺のコストを 0 としたグラフを $G_{\mathscr{C}}$ とする．$G_{\mathscr{C}}$ における（コストを長さと見なした）u,v 間の最短パスの長さを $d_{G_{\mathscr{C}}}(u,v)$ とする．その反復における 2 つのスーパーノード S_1, S_2 間の最短パスの長さは

$$d_{G_{\mathscr{C}}}(S_1,S_2) = \min_{u \in S_1, v \in S_2} d_{G_{\mathscr{C}}}(u,v)$$

と定義される．アルゴリズムは以下のように動作する．

(1) $G_{\mathscr{C}}$ における 2 つの異なる活性なスーパーノード間で，最短パスの長さが最も短い 2 つの異なるスーパーノード S_1, S_2 を求める．

(2) $G_{\mathscr{C}}$ における S_1, S_2 間の最短パス上の辺 e で，スーパーノード間にあるものをすべて E' に加える．

(3) $\mathscr{C} \leftarrow (\mathscr{C} \setminus \{S_1, S_2\}) \cup \{S_1 \cup S_2\}$ とする．

(1)〜(3) を活性なスーパーノードが存在する限り繰り返し，最後に E' から閉路となる辺を取り除いた辺部分集合 $F \subseteq E' \subseteq E$ を出力する．

3 主双対アルゴリズム

シュタイナー森問題に対して，1991 年に Agrawal, Klein and Ravi により提案されたアルゴリズムは，主双対アルゴリズムであり，シュタイナー森問題に対する最初の性能保証アルゴリズムである [1]．そこで，はじめにシュタイナー森問題の IP 定式化を与える．

[1] Evaluation of Practical Performances of Approximation Algorithms for the Steiner Forest Problem

[2] Jumpei Sugawara, Information and System Engineering Course, Graduate School of Science and Engineering, Chuo University

[3] Noriyoshi Sukegawa and Takao Asano, Department of Information and System Engineering, Faculty of Science and Engineering, Chuo University

一方の端点が部分集合 S に属し，他方の端点が S に属さないすべての辺の集合を $\delta(S)$ とし，s_i と t_i を分離する部分集合からなる集合を $\mathscr{S}_i$ とする．すなわち，$\mathscr{S}_i = \{S \subseteq V : |\, S \cap \{s_i, t_i\}\,| = 1\}$ とする．すると，シュタイナー森問題の IP 定式化は以下のように書ける．

$$\begin{aligned} \text{minimize} \quad & \sum_{e \in E} c_e x_e \\ \text{subject to} \quad & \sum_{e \in \delta(S)} x_e \geq 1, \ \forall S \subseteq V : \ \exists i, S \in \mathscr{S}_i, \\ & x_e \in \{0,1\}, \qquad e \in E. \end{aligned}$$

この IP の LP 緩和の双対問題は次のように書ける．

$$\begin{aligned} \text{maximize} \quad & \sum_{S \subseteq V : \exists i, S \in \mathscr{S}_i} y_S \\ \text{subject to} \quad & \sum_{S : e \in \delta(S)} y_S \leq c_e, \quad \forall e \in E, \\ & y_S \geq 0, \quad \forall S \subseteq V : \exists i, S \in \mathscr{S}_i. \end{aligned}$$

アルゴリズムは以下のように書ける．

最初すべての $S \subseteq V$ で y_S は 0 であるとし，$F = \emptyset$ とする．(V,F) で連結になっていない対 s_i, t_i が存在しなくなるまで以下の (1) を繰り返す．

(1) $|\, C \cap \{s_i, t_i\}\,| = 1$ を満たす (V,F) の連結成分 C のすべての集合を $\mathscr{C}$ とし，$C' \in \mathscr{C}$ かつ $e \in \delta(C')$ を満たす e のうちのどれかで最初に $\sum_{S : e \in \delta(S)} y_S = c_e$ となるまですべての $C \in \mathscr{C}$ の双対変数 y_C を一様に増加する．そしてそのような e を F に加える．

最後に，F から取り除いてもすべての対 s_i, t_i で連結性が保たれる辺をすべて取り除いて出力する．

4 計算機実験

本稿では，大食アルゴリズムと主双対アルゴリズムに加えて，大食アルゴリズムの (2) において E' に加える辺，すなわち，各反復で $G_{\mathscr{C}}$ における S_1, S_2 間の最短パス上の辺 e で，スーパーノード間にあるすべての辺，を縮約（両端点を同一視）して改良したアルゴリズムも加えて，計算機実験を行った．

実験に用いたパソコン NEC LaVie S PC-LS350TSR の仕様は以下の通りである．プロセッサーは intel(R) core(TM) i3-4100M CPU @ 2.50GHz，OS は Windows 8.1，メモリは 4.00GB である．用いたデータセットは，DIMACS のシュタイナー木問題の lin（lin01〜lin37）である [3].

最適解が判明している DIMACS のシュタイナー木問題（シュタイナー森問題の特殊ケース）の入力 [3] に対して，表 1 は，各アルゴリズムを実行して得られた結果からいくつかを選択したものである．表 2 は，（シュタイナー木問題とはならない）シュタイナー森問題の入力に対して各アルゴリズムを実行して得られた結果のなかからいくつかを選択したものである（k はターミナル点対数）．

表 1　アルゴリズムの近似率（k はターミナル点数)

	点数	辺数	k	最適解	近似率 [大食]	近似率 [大食改]	近似率 [主双対]
lin04	157	266	6	1239	1.0226	1.0226	1.0226
lin05	160	269	9	1703	1.0617	1.0317	1.0711
lin10	321	540	20	4132	1.0678	1.03	1.0649
lin13	822	1466	16	4609	1.0523	1.0067	1.044
lin15	840	1484	34	7145	1.0371	1.015	1.0237
lin19	2010	3662	41	13268	1.0439	1.019	1.0479
lin22	3692	6726	28	10519	1.0481	1.0241	1.042
lin23	3716	6750	52	17560	1.0435	1.0211	1.0498
lin24	7998	14734	16	15076	1.1	1.0423	1.1059
lin28	8062	14798	81	32584	1.0664	1.0306	1.0656

表 2　アルゴリズムの解と計算時間（秒）

	点数	辺数	k	解 [大食]	解 [大食改]	解 [主双対]	時間 [大食]	時間 [主双対]
lin04	157	266	6	1592	1530	1392	0.003	0.000
lin05	160	269	9	1734	1702	1734	0.003	0.001
lin10	321	540	20	5624	5508	5548	0.016	0.002
lin13	822	1466	16	8130	8004	8072	0.02	0.015
lin15	840	1484	34	11623	11329	11667	0.047	0.018
lin19	2010	3662	41	19372	18868	19600	0.083	0.116
lin22	3692	6726	28	22322	21396	21994	0.074	0.376
lin23	3716	6750	52	30272	29478	30354	0.175	0.413
lin24	7998	14734	16	24314	22990	24790	0.084	1.625
lin28	8062	14798	81	54394	52396	54430	0.681	2.003

5 比較・評価

大食アルゴリズムと主双対アルゴリズムを比較すると，表 1，2 からもわかるように入力によってその優劣は異なる．しかし，今回の入力において総合的にみれば，主双対アルゴリズムがやや優れていることがわかる．また，大食アルゴリズムを改良した場合，近似率はほとんどの入力で良くなった．今回の入力で改善幅の大きなものは 0.05 程度近似率が良くなっていた．最後に実行時間の視点で見ると，大食アルゴリズムと主双対アルゴリズムは入力により実行時間の優劣も変わることがわかる．しかし，点数 n と辺数 m が大きい入力では主双対アルゴリズムは遅く，最大で 20 倍程度大食アルゴリズムとの差が観察された．

著者の一人の浅野孝夫は，本稿の研究に対して中央大学特定課題研究費および文科省科研費（研究課題番号 15K11988）から一部援助を受けた．

参考文献

[1] A. Agrawal, P. Klein, and R. Ravi: “When trees collide: An approximation algorithm for the generalized Steiner problem on networks.” *SIAM Journal on Computing* 24.3 (1995), 440-456.

[2] A. Gupta and A. Kumar: “Greedy Algorithms for Steiner Forest.” *ACM STOC* (2015), 871-878.

[3] 11th DIMACS Implementation Challenge in Collaboration with ICERM: Steiner Tree Problems. 29 December 2015. http://dimacs11.cs.princeton.edu/downloads.html