アルゴリズム 77−11
(2001. 3. 12)

スタイナーネットワーク問題に対する近似アルゴリズムの実験的性能評価

八巻 満隆　　浅野孝夫
中央大学大学院理工学研究科情報工学専攻

あらまし　スタイナーネットワーク問題はネットワーク設計やネットワークルーティングと密接に関係する問題で, 実用上しばしば起こる問題である. 本論文では, スタイナーネットワーク問題の特殊版であるスタイナー木問題に対して, 最近提案された精度保証付きの近似アルゴリズムを実装し, 実装上の工夫を施したアルゴリズムとともに計算機実験を行い, それらの実験的性能評価を行う.

Practical Evaluation of Approximation Algorithms for the Steiner Network Problem

YAMAKI Mitsutaka　　ASANO Takao
Department of Information and System Engineering, Chuo University

Abstract　The steiner network problem is closely related to network design and network routing in real world applications. In this paper, we implement several recently proposed approximation algorithms for the steiner tree problem on undirected graphs and directed graphs including our improvements, and try to evaluate the practical performances of these algorithms.

1　はじめに

スタイナーネットワーク問題は以下のようにいくつかの種類がある. それぞれその問題の条件を満たすような部分グラフで最小のコストのものを求めるという問題である.

無向グラフのスタイナー木問題

無向グラフ $G=(V,E)$ とコスト関数 $c: E \to R_+$ が与えられ, その頂点集合 V がターミナルの集合 $X \subseteq V$ とスタイナー点の集合 $V-X$ に分けられている. このときすべてのターミナルを被覆する木 (スタイナー木) で最小のコストのものを求める問題.

有向グラフのスタイナー木問題

有向グラフ $G=(V,A)$, コスト関数 $c: A \to R_+$, 根 $r \in V$ とターミナルの集合 $X \subseteq V(|X|=k)$ が与えられる. このとき根 r から全てのターミナル X へパスを持つような木で最小のコストのものを求める問題.

スタイナーネットワーク問題

グラフ G の各辺 e のコスト $c(e) \geq 0$ と p 個のノードのペア $\{(s_1,t_1),...,(s_p,t_p)\}$ が与えられたときに, どの i $(i=1,2,\cdots,p)$ に対しても s_i から t_i へのパス (G が無向グラフのときは s_i と t_i を結ぶパス) を含むような G の最小のコストの部分グラフ H を求める問題.

これらの問題は計算量が理論的にはどれも多項式時間では解くことができないと信じられている NP 困難のクラスに属する問題である. NP 困難の問題に対しては, 解の最適性を譲る代わりに短い時間でなるべく最適解に近い解を出す方法で解決するというのが一般的である. 特に実際に求められた解が最適解からどの程度悪くなる可能性があるのかという理論的な精度の保証を持ったものを精度保証のある近似アルゴリズムという. 本論文では, 無向グラフと有向グラフにおけるスタイナー木問題に対して, 上記のように理論的な観点から研究された近似アルゴリズムの実験的性能評価を行う.

2 無向グラフのスタイナー木問題に対する実験的性能評価

この問題に対する代表的なアルゴリズムの性能を以下の表 1 に示す．この中で，Prömel and Steger [6] は確率的アルゴリズムで，他は全て決定的アルゴリズムである．また近似比率 1.734 は Borchers, Du [2] の結果を適用したものである．

表 1: スタイナー木問題の代表的アルゴリズム

発表年	近似比率	著者
1980	2.000	Takahashi, Matsuyama [8]
1993	1.834	Zelikovsky [9]
1994	1.734	Berman, Ramaiyer [1]
1995	1.694	Zelikovsky [10]
1997	1.667	Prömel, Steger [6]
1997	1.644	Karpinski, Zelikovsky [5]
1998	1.598	Hougardy, Prömel [4]
2000	1.55	Robins, Zelikovsky [7]

本論文では特に 1993 年の近似比率 1.834 のアルゴリズム [9], 1995 年の近似比率 1.694 のアルゴリズム [10] , 2000 年の近似比率 1.55 のアルゴリズム [7] について実験的性能評価を行った．

2.1 準備

入力グラフの辺のコストは三角不等式を満たし，グラフ G は完全グラフであると見なすことができる．また G_X はターミナルの集合 X の完全グラフを表すとし，$MST(G_X)$ は G_X の MST (最小スパンニング木) である．またターミナルスパニング木はスタイナー点を含まないスタイナー木を表す．ターミナルの部分集合 $X' \subseteq X$ 上のスタイナー木で X' のどのターミナルも葉になっているものを X' の full component という．ターミナルスパニング木 T のコストと full component K と T の辺から得られたスタイナー木のコストの差を $gain_T(K)$ と表す．つまり，$T[K]$ を $K \cup T$ の最小コストのスタイナー木とすると

$$gain_T(K) = cost(T) - cost(T[K])$$

である．これはターミナルのスパニング木 T から full component K と T の辺によって新たにターミナルのスパニング木を作ったときに節約されるコストを示している (図 1)．このとき T から取り除かれた辺を縮約してできたターミナルのスパンニング木を T' と表す．言い換えれば T' は T で X' の部分を結んでいた辺のコストを 0 としたものと考えてよい．

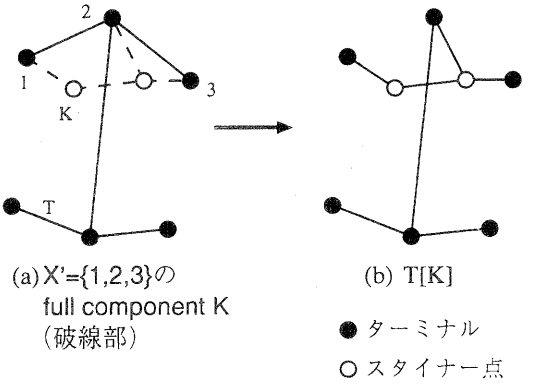

図 1: $gain_T(K)$ の説明図

2.2 アルゴリズム

アルゴリズム (近似比率 1.834)

1. $T = MST(G_X)$; $H = G_X$
2. Repeat forever
   (a) $|X'| = 3$ となる $X' \subseteq X$ のうちで最大の $gain_T(K) = cost(T) - cost(T[K])$ を実現する X' の full component K を求める．
   (b) もし $gain_T(K) \leq 0$ なら exit repeat
   (c) $H = H \cup K$; $T = T'$
3. Output $MST(H)$

近似比率 1.694 と近似比率 1.55 のアルゴリズムはそれぞれ $gain$ の定義の変更と，最大の $gain$ を求める際に探すターミナルの組合せを 4 つ以上のターミナルの組に対しても行うとい

う変更をしている. 以下にそれぞれの $gain$ の定義の変更を述べる.

近似比率 1.694 のアルゴリズムの $gain$

$$gain_T(K) = cost(T)/cost(T[K])$$

これよりアルゴリズムのステップ 2 の終了条件が $gain_T(K) \leq 1$ となる.

近似比率 1.55 のアルゴリズムの $gain$

ここでは新たに $loss$ というものを用い $gain$ を定義する. $loss(K)$ とは full component K のスタイナー点のスパニング森で各連結成分には少なくとも 1 つのターミナルが含まれている最小コストのものと定義し, そのときのコストも $loss(K)$ で表す (図 2). これはある full component に含まれるスタイナー点を解に含むと決めたとき, アルゴリズムでは貪欲に full component を選んでいるので, その full component のスタイナー点を解に含むとしたことで最適解が得られないということが起こることもあるが, そのとき最適解から離れてしまう値の上界になる. これは [7] で示されている. この $loss(K)$ を使い,

$$gain_T(K) = (cost(T) - cost(T[K]))/loss(K)$$

と定義している.

さらに $loss(K)$ の各連結成分を縮約した K のターミナルのスパンニング木を $C[K]$ と表し (すなわち K から $loss(K)$ の各辺のコストを 0 として得られるものが $C[K]$ である), アルゴリズムのステップ 2 の (c) の $T = T'$ は $T = MST(T \cup C[K])$ に変更されている.

2.3 計算機実験結果

3 つのアルゴリズムの性能比較

OR-Library のデータと乱数を用いて発生させたデータに対してそれぞれのアルゴリズムの実験を行った. 結果はどのアルゴリズムを用いても同じとなった. つまりどの $gain$ の定義に基づいても, 選ばれて来るスタイナー点は, 選ばれて来る順序は違っても, 最終的には同じものが選ばれて来ていた. この理由としてどの $gain$ も取り除かれる辺のコストが新たに加えられる辺のコストよりも大きいということに

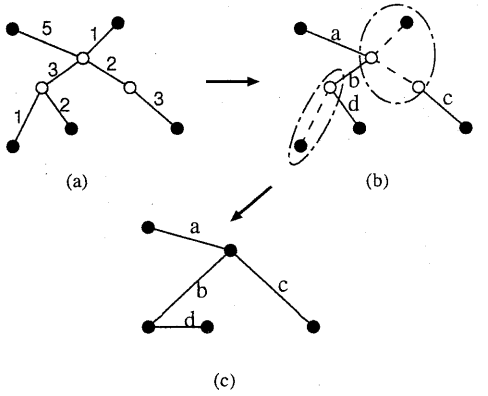

図 2: (a): full component K. (b): $loss(K)$ (破線部), (c): $C[K]$

依存しているためではないかと考えることができる. OR-Library のデータに対して得られた解は平均で最適解から約 1.59% 離れた解になった.

有向グラフのスタイナー木問題に対するアルゴリズムを用いた実験

ここでは有向グラフのスタイナー木問題に対するアルゴリズムを用い, 無向グラフのスタイナー木問題を解いた. 無向グラフの各辺を両方向に向かう辺を持つという形で有向グラフに置き換え, 有向グラフのスタイナー木問題の根を無向グラフのスタイナー木問題のどこかのターミナルであると見なすことで, 有向グラフのスタイナー木問題に対するアルゴリズムを無向グラフのスタイナー木問題に適用することができる. 実験で用いた以下のアルゴリズムの詳細は次章で述べる.

ここでの実験のデータは OR-Library のデータを用いた. 多くの場合は Charikar らのアルゴリズム [3] よりもそれを局所改善したものの方が良い解を出した (図 3). しかし全体的に見ると, Charikar らのアルゴリズムは最適解から大きく離れ, 局所改善したものでも, 最適解を得たものは少なく, Charikar らのアルゴリズムよりは改善されているもののやはり最適解からは離れている. Charikar らのアルゴリズムは平均で最適解から約 14.00% 離れた解になり,

局所改善したものは平均で最適解から約 2.85% 離れた解になった.

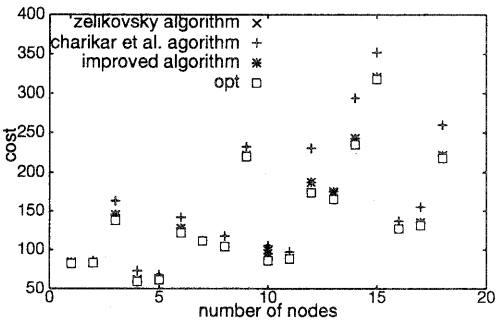

図 3: アルゴリズムの結果の比較 (表示が消えている部分は最適解と重なっている)

3 有向グラフのスタイナー木問題に対する実験的性能評価

有向グラフのスタイナー木問題に対しては素朴なアルゴリズムと Charikar らのアルゴリズム [3] が現在知られている. ここではこれらのアルゴリズムとこれらに局所改善を行ったアルゴリズムに対して計算機実験を行い, その性能評価を行った.

3.1 アルゴリズム

素朴なアルゴリズム

この問題に対して根から各ターミナルへの最短パスを計算し, それらを結びつける素朴なアルゴリズムが考えられる. このアルゴリズムに対して近似比率が $k = |X|$ になる例が以下のように構成できる. 根 r から 1 点 v へコスト C_{OPT} のパスがあり, v から各ターミナルへコスト 0 の辺があるようなグラフを考える. さらに, 根 r から各ターミナルへコスト $C_{OPT} - \epsilon$ の辺がある (図 4). 素朴なアルゴリズムではそれぞれの最短パスを拾い, v を通るような辺は使わないので, コスト $k(C_{OPT} - \epsilon)$ になってしまう. しかし最適解は, ϵ が十分小さい正数とすると, コスト C_{OPT} の辺 (r, v) を使い, そこから各ターミナルへコスト 0 の辺を使うというものになる. このコストは C_{OPT} になるため, アルゴリズムの近似比率は k となる.

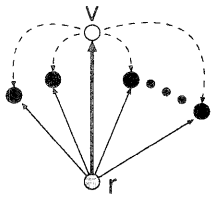

図 4: 近似比率が k になる例

Charikar らのアルゴリズム

Charikar らによって提案された近似比率 $i(i-1)k^{1/i}$ のアルゴリズムの説明をする. ここで i は任意の定数である.

一般性を失うことなく, 全ての点 u と v の間には G における u から v への最短パスと等しいコストを持つ辺 (u, v) が存在すると仮定できる. $c(e)$ を辺 e のコスト, $c(T)$ を木 T のコスト (T の辺のコストの和) とする. $k(T)$ を T に含まれるターミナルの数とする. $d(T)$ を T の密度とする. つまり T におけるターミナル 1 個当たりのコスト $d(T) = c(T)/k(T)$ である.

アルゴリズムは密度の小さい部分木すなわち根 r から中間の点 v へのパスと v からいくつかのターミナルへの最短パスからなる部分木を見つけることを繰り返す. このような構造を持つ小さい部分木を再帰アルゴリズム $DST_i(l, r, X)$ によって見つける. これは根 r から l 個のターミナルへのパスを含む部分木を見つけるアルゴリズムである. $T_i(l, r, X)$ を $DST_i(l, r, X)$ によって返される木とする. $DST_1(l, r, X)$ は根に最も近い l 個のターミナルをみつけ, それらを根からの最短パスを使って結ぶということを表している. $DST_i(l, r, X)$ は, この形の全ての木に含まれるような木 $T_{i-1}(l', v, X)$ を $DST_{i-1}(l', v, X)$ で求め, これらの木に $\{(r, v)\}$ を加えた木 $T_{i-1}(l', v, X) \cup \{(r, v)\}$ で密度が最小になる, 中間の点 v と l' を繰り返し見つける. ここで $1 \leq l' \leq l$ である. なお $T_i(l, r, X)$ では, 中間の点を高々 $i-1$ 個含み, 中間の点 v からは高々 l 個のターミナルへ最短パスが延びている (図 5).

アルゴリズム $DST_i(l, r, X)$

1. X の中に r から到達可能な l 個のターミナルが存在しないなら, return $\emptyset$.
2. $T = \emptyset$; $l'' = l$
3. $l'' > 0$ である限り以下の (a),(b),(c) を繰り返す
   (a) $T_{BEST} = \emptyset$; $d(T_{BEST}) = \infty$
   (b) 各点 $v \in V$ と各 l' $(1 \leq l' \leq l'')$ に対して以下の (i),(ii) を行う.
      (i) $T' = DST_{i-1}(l', v, X) \cup \{(r, v)\}$
      (ii) $d(T_{BEST}) > d(T')$ なら $T_{BEST} = T'$
   (c) $T = T \cup T_{BEST}$; $l'' = l'' - |X \cap V(T_{BEST})|$; $X = X - V(T_{BEST})$
4. return T

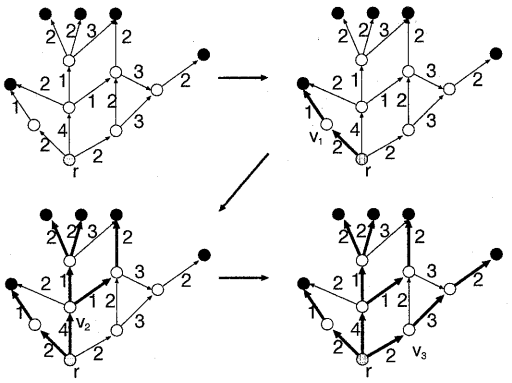

図 5: Charikar らのアルゴリズムの適用例 ($l = 5$ で v_1, v_2, v_3 は中間の点)

局所改善

素朴なアルゴリズムと Charikar らのアルゴリズムの局所改善を提案する. Charikar らのアルゴリズムでは, 密度の小さい部分木を見つけてその部分を解に加えるということを繰り返したが, ここでは一度見つけてきた部分木に対応するグラフの木のコストを 0 にし, そのグラフを用いて新たに密度の小さい部分木を求めてくるということを行う. これによって, 一度解に加えた部分木が再度選ばれ易くしている. つまり, アルゴリズムのステップ 4 で $c(G(T_{BEST})) = 0$ を行う (図 6). 同様に素朴なアルゴリズムでは, 根からあるターミナルへの最短パスを求め, そのパス上の辺のコストを 0 にし, そのグラフで次のターミナルへの最短パスを求めるということを繰り返す.

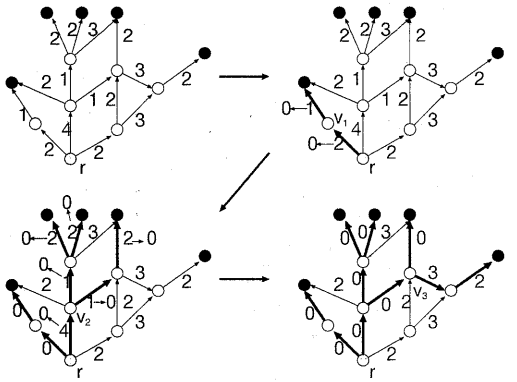

図 6: 図 5 の例に対する局所改善した Charikar らのアルゴリズムの適用例 ($l = 5$)

3.2 計算機実験結果

乱数を用いた実験

乱数を用い, 点を発生させ実験を行った. 点は $[0, 5 \times$ 点の数$]$ から座標を発生させた. つまり点の個数が 50 のときには $[0, 250]$ から座標を発生させている. 辺は乱数を用い 20% の確率で与えた. 辺のコストはユークリッド距離のものとマンハッタン距離のもので実験を行った.

発生させる点の数を 50 個から 80 個まで変化させ実験を行った. ここでターミナルの数は発生させた点の 10% としている. 結果はそれぞれ 10 回実験を行い平均をとったものである. 結果は図 7, 図 8 のようになった.

図 7, 図 8 より, 素朴なアルゴリズムに比べ, 素朴なアルゴリズムを局所改善したもの, Charikar らのアルゴリズム, それを局所改善したものはいずれも大幅な改善が見られた. また Charikar らのアルゴリズムを局所改善したものは, 改善前のものに比べいくらか良い解が得られている. 素朴なアルゴリズムと Charikar らのアルゴリズムの結果が同じになることもあ

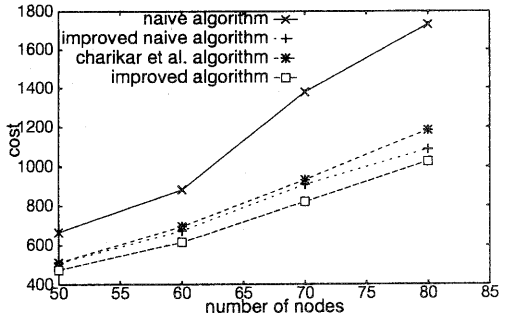

図 7: ユークリッド距離の場合の 4 つのアルゴリズムの比較

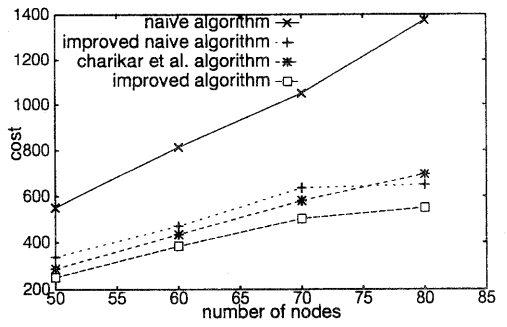

図 8: マンハッタン距離の場合の 4 つのアルゴリズムの比較

る. しかし素朴なアルゴリズムの方が Charikar らのアルゴリズムよりも良い解を得ることが無いことは明らかである. また Charikar らのアルゴリズムはそれを改善したものと結果が同じになることも, 良い解を得ることがある.

図 9 は乱数を用いて非対称な有向グラフを発生させ, 実験を行なったものである. 発生させる点の数を 50 個から 80 個まで変化させ実験を行った. ここでターミナルの数は発生させた点の 10% としている. 結果はそれぞれ 10 回実験を行い平均をとった. 辺は 20% の確率で与え, 辺のコストは乱数を用い $[0, 5 \times$ 点の数$]$ の範囲で与えた.

非対称なグラフにおいても対称なグラフと同様に, 素朴なアルゴリズムに比べ, 素朴なアルゴリズムを局所改善したもの, Charikar らのアルゴリズム, それを改善したものはいずれも大幅な改善が見られた. また Charikar らのアルゴリズムを改善したものは, 改善前のものに比べいくらか良い解が得られている.

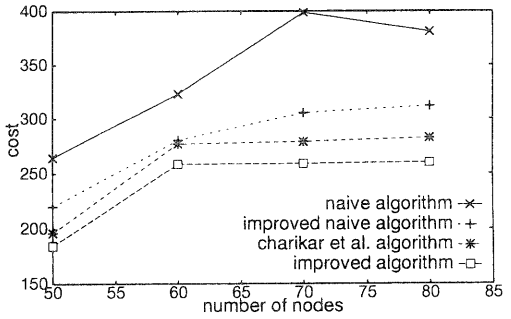

図 9: 非対称なグラフにおける 4 つのアルゴリズムの比較

OR-Library のデータを用いた実験

OR-Library の steinb の 18 個のデータを用い, 実験を行った. しかし, データは本来無向グラフのスタイナー木問題に対して使われる物だったため, 有向グラフのスタイナー木問題に対して使用するにはデータの変更をする必要があった. そこで, ここでは次のようにデータを変更した.

OR-Library のデータは,

$$\begin{array}{ccc} 1 & 3 & 4 \\ 2 & 5 & 3 \\ 2 & 6 & 7 \\ 3 & 5 & 2 \\ & \vdots & \end{array}$$

というような形になっている. これは本来, 点 1 と点 3 の間にコスト 4 の無向辺があるということを表しているが, ここでは点 1 から点 3 へコスト 4 の有向辺があると解釈することにする. つまり図 10 である.

これを全てのデータに対して行うと, 図 11 のような非連結のグラフができる. そこでこのグラフに対して新たに根を加え, 根から全ての

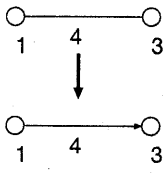

図 10: 無向辺を有向辺へ変更する

ターミナルへのパスが存在するように, 根から全ての点への辺を持たせる. この根から各点への辺のコストは, ルートを加える前のグラフにおける, 到達可能な 2 点で, 最も遠い点の間のコストを持たせる. これを行うと図 12 のようになる.

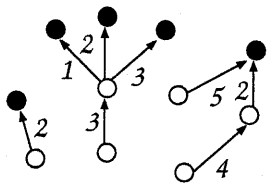

図 11: OR-Library のデータを有向グラフに適用する

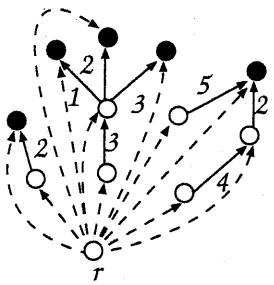

図 12: 根 r を加える (ここで根からターミナルへの辺のコストは 6 になる)

このデータを用い実験を行った結果は図 13 のようになる. ここでも素朴なアルゴリズムに比べ, 素朴なアルゴリズムを局所改善したもの, Charikar らのアルゴリズム, それを局所改善したものは大幅な改善が見られる. さらに図 14 は素朴なアルゴリズムを局所改善したもの, Charikar らのアルゴリズム, それを局所改善したもの, 下界を拡大したものである. これを見ると, 多くの場合, Charikar らのアルゴリズムを局所改善したものは局所改善していないものに比べある程度の改善が見られることがわかる.

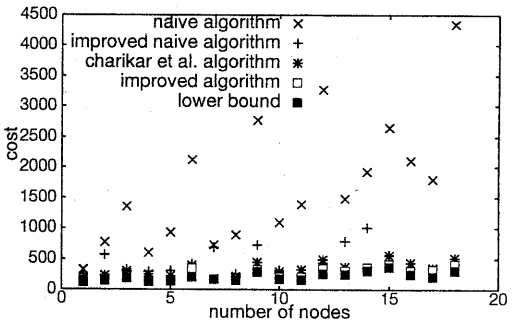

図 13: OR-Library のデータを使った実験結果 1

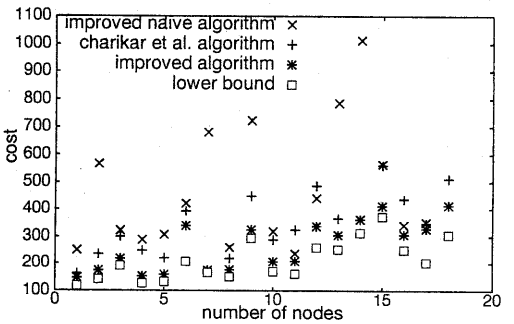

図 14: OR-Library のデータを使った実験結果 2

ここで下界は OR-Library のデータの最適解の値と新たに付け加えた根への 1 本の辺のコストの値の和である. つまり図 15 のようになる.

4 まとめ

無向グラフのスタイナー木問題に対する 3 つ近似アルゴリズムを調査し, その実験的性能評価を行った. それぞれ異なった $gain$ を用い

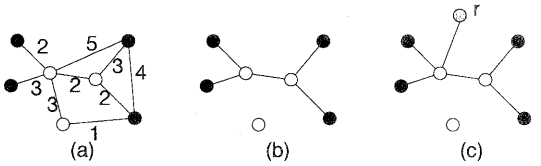

図 15: (a). 無向グラフのスタイナー木問題の入力グラフ, (b). 入力グラフの最適解, (c). 有向グラフのスタイナー木問題に対する下界

てアルゴリズムが作られていたが, 良い近似比率を得ている *gain* を用いているものほど良い解を出すというわけではないという結果になった. しかし多くの場合最適解かそれに限りなく近い解を出した.

また有向グラフのスタイナー木問題に対する近似アルゴリズムについてはアルゴリズムの局所改善案を示し, その実験的性能評価を行った. 実験的には局所改善はより良い解が得られた. また精度という面からも, 近似比率を考えるとなかなか良い精度の解が得られていると感じた.

しかしどちらの問題に対するアルゴリズムも時間的な問題が残っていると感じた. 大きなインスタンスに対してのこれらのアルゴリズムの適用は難しいと思われる. またさらなる近似比率の改善が見込まれる.

5 謝辞

本研究は, 一部, 中央大学理工学研究所, 同先端技術センターおよび文部省科学研究費補助金からの援助のもとで行われたものである.

参考文献

[1] P. Berman and V. Ramaiyer : Improved approximations for the Steiner tree probelm, *Journal of Algorithms 17*, 1994,pp.381-408.

[2] A. Borchers and D.-Z. Du : The k-Steiner ratio in graphs, *SIAM J. Computing 26*, 1997,pp.857-869.

[3] M. Charikar, C. Chekuri, T. Cheung, Z. Dai, A. Goel, S. Guha and M. Li : Approzimation Algorithms for Directed Steiner Problems, *Proc. Symposium on Discrete Algorithms*, 1998,pp.192-200.

[4] S. Hougardy and H. J. Prömel : A 1.598 Approximation Algorithm for the Steiner Tree Problem in Graphs, *Proc. Symposium on Discrete Algorithms*, 1999,pp.448-453.

[5] M. Karpinsky and A. Zelikovsky : New Approximation Algorithms for the Steiner Tree Problems, *Proc. Journal of Combinatorial Optimization, 1*, 1997,pp.47-65.

[6] H. J. Prömel and A. Steger : RNC-approximation algorithms for the Steiner problems, *Proc. Symposium on Tehoretical Aspects of Computer Science*, 1997,pp.559-570.

[7] G. Robins and A. Zelikovsky : Improved Steiner Tree Approximation in Graphs, *Proc. Symposium on Discrete Algorithms*, 2000,pp.770-779.

[8] H. Takahashi and A. Matsuyama : An approximate solution for the Steiner problem in graphs, *Math. Jap. 24*, 1980,pp.573-577.

[9] A. Zelikovsky : An 11/6-Approximation Algorithm for the Network Steiner Problem in Graphs, *Algorithmica 9*, 1993,pp.463-470.

[10] A. Zelikovsky : Better approximation bounds for the network and Euclidean Steiner tree problems, *Technical report CS-96-06, University of Virginia*,