1997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

ネットワークの辺連結度増加問題を解くアルゴリズムの計算機実験

京都大学 *片山 茂樹 KATAYAMA Shigeki
01403794 京都大学 永持 仁 NAGAMOCHI Hiroshi
01001374 京都大学 茨木 俊秀 IBARAKI Toshihide

1 はじめに

容量付き無向グラフ $G=(V,E,c_G)$ のいくつかの辺の容量を, 増加容量の和が最小になるように増加させることにより, G の辺連結度を指定された値 k に増大させる問題を考える. ただし, V を頂点集合, E を辺集合, c_G を容量 $E\to\mathbf{R}^+$ とし, また $n=|V|, m=|E|$ と記す. G の辺連結度を k に上げるときに必要な容量の合計の最小値を $\Lambda_G(k)$ と書く. この問題は, グラフの辺連結度増加問題と呼ばれている.

Nagamochi と Ibaraki[1] は, すべての目標値 k に対する $\Lambda_G(k)$, およびそれを実現する最適解の全体を $O(mn+n^2\log n)$ 時間で出力するアルゴリズムを提案している. このアルゴリズムは, 全ての目標値 k に対する最適解をグラフ上の容量付きサイクルの集合として表現して出力する. この表現に必要なサイクルの数はアルゴリズムの理論的な解析により, 高々 $6n+4n\log_2 n$ であることが示されている.

以上のアルゴリズムは, 理論的な時間量の評価では他のアルゴリズムより高速であるが, かなり複雑な部分があるので, 実用性については検証が必要とされていた. そこで本研究では, 実際にこのアルゴリズムを実装し, 具体的なグラフに対して計算機上で走らせ, その性能を調べた.

2 定義

無向グラフ $G=(V,E,c_G)$ において, 互いに素な頂点の部分集合 $X,Y\subset V$ に対し, 端点がそれぞれ X と Y に属するような辺の集合を $E_G(X,Y)$ と書き, その容量を $d_G(X,Y)=\sum_{e\in E_G(X,Y)}c_G(e)$ と定義する. カットは $X\neq\emptyset$ かつ $X\subset V$ なる頂点の部分集合 X で定義され, その大きさは $d_G(X,V-X)$ である. これは単に $d_G(X)$ と表すこともある.

$X\subseteq V$ に対し内部辺連結度を $\lambda_G(X)=\min\{d_G(X')\mid\emptyset\neq X'\subset X\}$ で定義し, 特に $\lambda_G(V)$ (つまり, G の最小カットの大きさ) を G の辺連結度と呼ぶ. 例えば, 図 1のグラフ G の辺連結度は 7 であり, $X=\{v_3,v_4\}$ は G の 1 つの最小カットである.

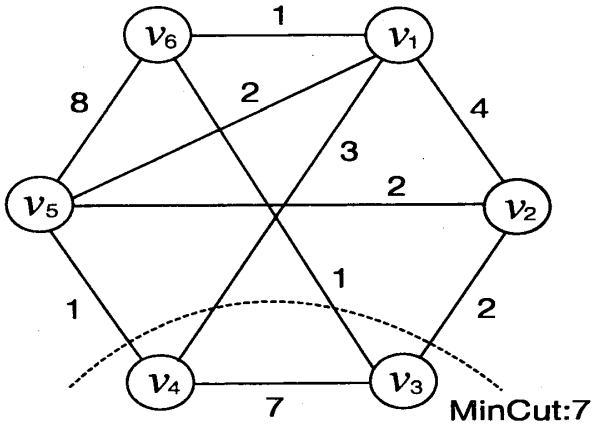

図 1: グラフの例 G

3 辺連結度増加問題を解くアルゴリズム

定理 [1]

1. 最適解（G を目標の辺連結度 k にする為に最低限必要な増加容量の振り分け方）は，容量付きサイクルの積み重ねにより与えられる．

2. 最適解の表現に必要な $6n+4n\log n$ 個未満のサイクルを見つける $O(mn+n^2\log n)$ 時間のアルゴリズムが存在する．

図 2 は 図 1 のグラフ G の辺連結度を k に上げるときに必要な容量の合計の最小値 $\Lambda_G(k)$ である．図 3 では G に 1 辺の容量 0.5 のサイクル $v_4 \to v_5 \to v_4$ と，1 辺の容量 1 のサイクル $v_2 \to v_4 \to v_5 \to v_2$ を積み重ねることによって辺連結度を最適に 10 に増加している．

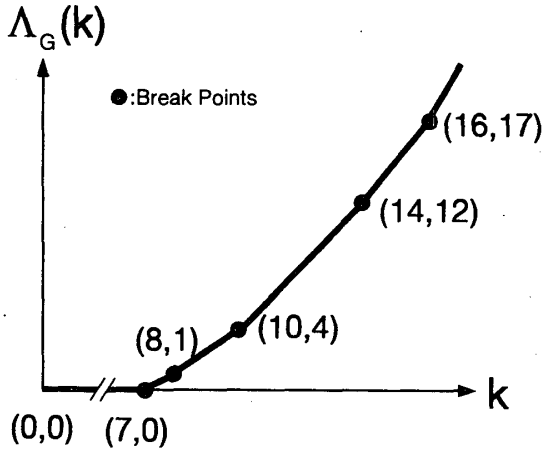

図 2: G の辺連結度を k に上げるときに必要な容量の合計の最小値 $\Lambda_G(k)$

4 計算機実験

4.1 目的

実際のグラフでは解の表現に必要なサイクルの数がどの程度の大きさになるのかを調べる．また，理論上の計算時間 $O(mn+n^2\log n)$ を実現するために [1] のアルゴリズム中で用いられるフィボナッチヒープ（複雑な動作をする）は実際のグラフに対してどの程度有効なのか，単純なヒープ（理論上の計算時間は遅くなるが実現が簡単）との比較により明らかにする．そして [1] のアルゴリズムはどの程度に規模のグラフまで実用的に扱いうるのかを明らかにする．

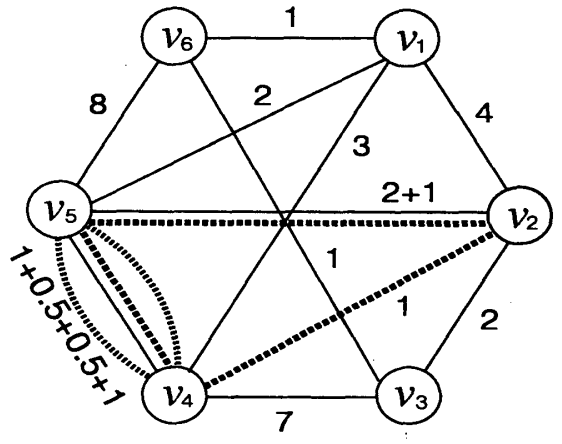

図 3: G の辺連結度を最適に 10 に増加した例

4.2 結果

最適解を与えるためのサイクルの数は疎なグラフ（$m = O(n)$），密なグラフ（$m = O(n^2)$）とも理論の上限よりも大幅に少なく，$2n$ にも満たなかった．計算時間は疎なグラフでは $n = 10000$，密なグラフでは $n = 1000$ 程度の大きさで数分から数 10 分であり（Sun Ultla 1 の場合），この程度の大きさまでなら実用的に扱いうる事が明らかになった．フィボナッチヒープについてであるが，ヒープを使用した場合と比較して，疎なグラフではフィボナッチヒープを使用した方が計算時間が速く，逆に密なグラフではフィボナッチヒープを使用した方が計算時間が遅くなった．ただ，どちらの場合も差はほとんどなく，複雑なフィボナッチヒープを用いる効果はあまりみられなかった．これらの理論的な解析が今後の課題である．

参考文献

[1] H.Nagamochi and T.Ibaraki, *Augment edge-connectivity over the entire range in $\tilde{O}(nm)$ Time,* Technical Report ♯96011, Kyoto University, Department of Applied Mathematics and Physics, 1996.