最大格差最小 k -カットアルゴリズムによる最小 k -カット問題の近似解法

非会員 戴 陽 (神戸商大) 非会員 岩野 和生 (日本アイ・ビー・エム (株) 東京基礎研究所)
非会員 加藤 直樹 (神戸商大)

A New Approximate Algorithm for the Minimum k -Cut Problem by Using Minimum Range k -Cut Algorithm

Yang Dai, Nonmember (Kobe University of Commerce)

Kazuo Iwano, Nonmember (Tokyo Research Ceneter, IBM Japan)

Naoki Katoh, Nonmember (Kobe University of Commerce)

Given a connected undirected multigraph $G = (V, E)$ with positive edge capacities and a positive integer k with $k \geq 2$, the minimum k -cut problem is to find a set $S \subseteq E$ of minimum capacity whose removal leaves k connected components. It is known that this problem is NP-hard. In this paper we propose a new approximate algorithm for solving the minimum k -cut problem by using efficient algorithm for the minimum range k -cut problem. Our method is based on the association of the range value of a k -cut and its cut value when each edge weight is chosen uniformly randomly between 0 and 1. Though the closeness of our approximate solution has not yet been theoretically evaluated, we demonstrate its usefulness by performing computational experiments and comparing with another approximate algorithm proposed by Saran and Vazirani. Our computational experiments show that the new method produces almost the same solutions or even better solutions for small k like $k = 3, 4$, with running time much less than that for the algorithm of Saran and Vazirani.

キーワード：最小カット, 最小 k -カット, 最大格差最小 k -カット, 回路分割

1. まえがき

$G = (V, E)$ を節点集合 V , 枝集合 E の連結な単純無向グラフとし, G の各枝には正の容量が付されているものとする。以下, $n = |V|$, $m = |E|$ とする。 G の節点集合 V の k -分割 ($k \geq 2$) $V_1, V_2, \dots, V_k$ (すなわち, $\bigcup_{i=1}^k V_i = V$ かつ $V_i \cap V_j = \emptyset (i \neq j)$) に対して定まる枝集合

$$C = \{(u, v) \mid u \in V_i, v \in V_j, i \neq j, 1 \leq i, j \leq k\}$$

を k -カットと呼ぶ。すべての k -カットのなかでその枝の容量の総和が最小の k -カットを最小 k -カットと呼ぶ。最小 k -カットを求める問題を最小 k -カット問題と呼ぶ。とくに $k = 2$ のときは最小カット問題と呼ばれ, 従来から良く研究されているが ([3], [9], [12]), 最近, 永持, 茂木の両氏によって $O(mn + n^2 \log n)$ の効率のよいアルゴリズムが提案された [10]。

最小 k -カット問題は k を入力データの一部と見ると NP 完全であることが知られている [5]。ただし, n に関しては多項式時間アルゴリズムが知られている (しかし, $k = 3, 4$ でも n の高次のオーダーの多項式なので実用的ではない)。この問題は, 大規模集積回路 (VLSI) のチップの設計 [8] や並列計算, 分散計算の通信時間の最小化などに現れ [14], 実用上重要な応用例を持つ問題である。

最近 Saran と Vazirani [15] によって最悪比 $(2 - 2/k)$ の最小 k -カット問題の近似解を求める二つのアルゴリズムが提案された。一つは, すべての点対間の最小カットを表現する Gomory-Hu cut tree [6] にもとづくもので, もう一つは, 最小カットアルゴリズムを再帰的に適用するものである。

本論文では最小 k -カットを求める新しい近似アルゴリズムを提案し, 計算機実験による性能評価について報告する。提案するアルゴリズムは最大格差最小 k -カットを求めるアルゴリズムにもとづいている。

各枝 e に重み $w(e)$ が与えられている連結な無向グラフ (多重辺を持つことを許す) $G = (V, E)$ において, G の k -カットのなかでその k -カットに属する枝の重みの最大値と最小値の差 (それを最大格差と呼ぶ) を最小にする k -カットが最大格差最小 k -カットである。本論文でははじめにこの問題を解く $O(m + n \log n)$ 計算量のアルゴリズムを提案する。このアルゴリズムは筆者のうちの二人加藤, 岩野による最大格差最小 k -カットを求めるアルゴリズム [7] を拡張したものであり, とくに $k = 2$ の場合, [7] のアルゴリズムに一致する。

本論文で提案する最小 k -カット問題の近似アルゴリズムのアイデアは次の通りである。簡単のため, 枝の容量はすべて 1 の場合を考える。すなわち, 枝の本数最小の k -カット

を求める問題を考える。まず与えられたグラフ G の各枝の重みを区間 $[0, 1]$ の一様乱数によって定める。そして最大格差最小 k -カットを求める。そのカットに属する枝の本数を数える。このプロセスをある回数繰り返してそのなかで位数最小のカットを最小 k -カットの近似解として出力する。このアルゴリズムの妥当性は、各枝の重みを区間 $[0, 1]$ の一様乱数で定めたとき、枝の本数の少ない k -カットはその最大格差の期待値も小さいという観察にもとづいている。同様のアイデアが最小 k -カットを近似的に求めるアルゴリズムで用いられ、良い結果が報告されている [2]。枝の容量が正の実数の場合も少し修正を加えるだけで、この方法が適用できる。

本論文では、この近似アルゴリズムの性能を評価するために、計算実験を行った。Saran と Vazirani [15] の二つの近似アルゴリズムのうち、最小 k -カットアルゴリズムを再帰的に適用するという方法とわれわれの方法とを計算時間と近似解の良さの両面から比較した ([15] のもう一つの方法は、本論文で採用した [15] の方法より明らかに計算時間が大きくなることと、最悪比が変わらないという理由で今回の実験から省いた)。計算実験で用いたネットワークは点数 150、疎密度 10% ~ 50% とし、[11] と同様の方法で生成した。また $k = 3, 4, 6, 8$ の場合について計算実験を行った。その結果、特に $k = 3, 4, p = 0.01$ 以下の場合 (p はクラスター間密度と呼ばれ、4 節で正確に定義する)、提案したアルゴリズムは、枝の重みを乱数で与えて最大格差最小 k -カットを求めるという操作を 5 ~ 10 回繰り返すこと ([15] の $1/3 \sim 1/4$ の時間) で、[15] と同じ容量のカット、もしくは容量の小さいカットを得ている。 $k = 6, 8$ の場合は [15] の方法が優れていることが分かった。しかし、VLSI の設計に現れる回路分割の階層的分割では、比較的少数の分割 (例えば $k = 2$) を再帰的に行ない、最終的な複数分割を求めている [8]。したがって、このような場合、 $k = 2, 3, 4$ に対する本アルゴリズムを再帰的に用いることは有用であると考えられる。

2. 最大格差最小 k -カット問題

前節でも述べたように、われわれの最小 k -カット近似アルゴリズムは最大格差最小 k -カット問題のアルゴリズムにもとづいている。まず、最大格差最小 k -カット問題を解く効率のよいアルゴリズムを示す。簡単のために、 E の枝の重みはすべて異なるものと仮定する。また、 $T_{\min}$ と $T_{\max}$ をそれぞれ、 G の重み最小の木と最大の木とする。次の補題は明らかである [2]。

補題 1 任意の k -カット S について、 S の重み最大の枝は $T_{\max}$ に属し、 S の重み最小の枝は $T_{\min}$ に属する。

補題 1 より、最大格差最小 k -カットを求めるのに $T_{\max} \cup T_{\min}$ について調べれば十分である。従って以下では $E = T_{\max} \cup T_{\min}$ とする。また、 $T_{\min}$ の枝を重みの小さい順に並べたものを $e_1, e_2, \dots, e_{n-1}$ とする。すなわち、 $w(e_1) < w(e_2) < \dots < w(e_{n-1})$ である。($w(e_i)$ は、すべて異なると仮定

したことに注意。) ある k -カット S において、 $\max(S) = \max_{e \in S} w(e)$, $\min(S) = \min_{e \in S} w(e)$ とする。また、最大格差を $\text{range}(S) = \max(S) - \min(S)$ によって定義する。 $\min\text{-edge}(S)$ は k -カット S に属する重み $w(e) = \min(S)$ の枝 e を表す。 $\min\text{-edge}(S) = e$ なる k -カット S が e に対して upper-critical とは $\min\text{-edge}(S') = e$, $\max(S') < \max(S)$ となるような k -カット S' が存在しないことを意味する。このような k -カット S を $S(e)$ と表す。区間 $[\min(S(e)), \max(S(e))]$ は e に対する upper-critical 区間という。明らかに、 $S(e_j) (j > n-k+1)$ は $e_1, e_2, \dots, e_{j-1}$ を含まない。また、 $e_1, e_2, \dots, e_{j-1}$ 以外の枝を取り除いたあとのグラフは多くても $k-1$ 個の連結成分しか含まない。従って、 $S(e_j) (j > n-k+1)$ は k -カットではあり得ない。以上の議論と補題 1 より、最大格差最小 k -カットは $S(e_i)$, $1 \leq i \leq n-k+1$ を調べることによって求められる。

アルゴリズムは $S(e_1), S(e_2), \dots, S(e_{n-k+1})$ という順に計算するが、 $S(e_i)$ の枝集合を記憶するのではなく、これまで一番小さい最大格差の値を覚えておく。 $1 \leq i \leq n-k+1$ なる i について、 $e_1, \dots, e_{i-1}$ の枝が縮約された後のグラフを G_{i-1} とする。 T_{i-1} を G_{i-1} の重み最大の木とする。特に、 $G_0 = G$, $T_0 = T_{\max}$ である。 P を枝 e_i の両端点を結ぶ T_{i-1} 上の路とし、 $f_i (\in T_{\max})$ を P 上の重み最小の枝とする。ただし、 $e_i \in T_{i-1}$ のとき、 e_i のみからなる路を P とする。まず、 $f_i \in S(e_i)$ を示す。 P には一本の枝しかないならば (この場合は $f_i = e_i$ のときのみ起こる)、明らかに f_i が $S(e_i)$ に属さねばならない。以下、 P には 2 本以上の枝がある場合について考える。 $S(e_i)$ は k -カットであるから、 $S(e_i)$ に属する枝のうち、少なくとも一本は路 P にあることが分かる。仮に f_i が $S(e_i)$ に属していないとする。そうすると、ある枝 f' があり、 $f' \in P \cap S(e_i)$ 。 $S(e_i)$ が k -カットなので T_{i-1} の枝を丁度 $k-1$ 本含む。ここでこの $k-1$ 本の枝の集合を F とする。 T_{i-1} から枝集合 $F \cup \{f_i\} - \{f'\}$ を取り除くと、 k 個の連結成分が得られる。この k 個の連結成分をつなぐ枝の集合 S は k -カットである。 G_{i-1} は $e_1, \dots, e_{i-1}$ を縮約したグラフであり、 $e_1, \dots, e_{i-1}$ は $T_{\min}$ の重みの小さい方の $i-1$ 本の枝であることから、 $\min(S) = w(e_i)$ であることに注意しておく。次の二つのケースが考えられる。(1) f' が F の枝の中で重み最大の枝ではない。この場合、 $\text{range}(S) = \text{range}(S(e_i))$ 。カット S は e_i に対しても upper-critical である。新たに S を $S(e_i)$ にすることによって、 $f_i \in S(e_i)$ になる。(2) f' が F の枝の中で重み最大の枝である。 $w(f_i) < w(f')$ より $\text{range}(S) < \text{range}(S(e_i))$ となり、 $S(e_i)$ が e_i に対して upper-critical であることに矛盾する。従って、 f_i は $S(e_i)$ に属さねばならない。

$F_{i-1}(j)$ を T_{i-1} から重みの小さい順に選んだ j 本の枝の集合とする。 k -カット $S(e_i)$ は次のように構成できる。 $f_i \in F_{i-1}(k-1)$ の場合、 T_{i-1} から $F_{i-1}(k-1)$ (一方、 $f_i \notin F_{i-1}(k-1)$ の場合、 $F_{i-1}(k-2) \cup f_i$) を除いて得られる k 個の連結成分によって定まる k -カットを $S(e_i)$ とする。

補題 2 f_i を、 e_i に対して上記の様に定めた T_{i-1} の枝とする。

この時、 $f_i \in F_{i-1}(k-1)$ ならば、 $\max(S(e_i)) = \max\{w(f) \mid f \in F_{i-1}(k-1)\}$ 。そうでなければ、 $\max(S(e_i)) = w(f_i)$ である。

証明. 上の議論より、 $[w(e_i), \max(S(e_i))]$ によって決まる k -カットは e_i に対して upper-critical である。■

T_0 からはじめ、アルゴリズムは $T_1, T_2, \dots, T_{n-k+1}$ の順に計算する。そのとき、dynamic trees [13] を使い、 T_i を管理しておく。いま T_{i-1} を求めたとする。定義より、 f_i を計算し、 $F_{i-1}(k-1)$ の集合にあるかどうかを判定する。補題 2 より、 $S(e_i)$ が求められるが、そのためには各 i について $O(n)$ 時間必要なので、全体で $O(n^2)$ 時間となるので、 $S(e_i)$ の枝を列挙しない。その代わりに、各 $S(e_i)$ の最大格差を計算しておく。 e_i の upper-critical 区間が分かった後、枝 e_i を縮約し、 $F_i(k-1)$ を求め、 e_{i+1} の upper-critical 区間を求める手続きに進む。以上の計算は dynamic trees を使い、 T_i を管理すれば、 $O(\log n)$ 計算時間で実行できる。最小木 $T_{\min}$ と最大木 $T_{\max}$ は $O(m + n \log n)$ の計算時間 [4] で求められる。 $T_{\min}$ の $n-1$ 本の枝をソートするには $O(n \log n)$ 時間が必要である。従って、最大格差最小 k -カットの区間が $O(m + n \log n)$ の時間で計算することができる。以下、この区間を $[w(e_{i*}), w(f_{i*})]$ とする。

手続き Minimum-Range-k-Cut

入力: グラフ $G = (V, E)$ 。枝の重み $w: E \rightarrow R^+$ 。 ($w(e_i)$ はすべて異なるものと仮定する。)

出力: 最大格差最小 k -カット。

Step 1: 最大木 $T_{\max}$ 及び最小木 $T_{\min}$ を求め、 $E := T_{\max} \cup T_{\min}$ とおく。

Step 2: $T_{\min}$ の枝 $e_1, e_2, \dots, e_{n-1}$ を重みの増加順にソートしたものとする。すなわち、 $w(e_1) < w(e_2) < \dots < w(e_{n-1})$ 。

Step 3: 各 $e_i \in T_{\min}$, $1 \leq i \leq n-k+1$ に対して、upper-critical 区間を求め、最大格差最小の区間を $[w(e_{i*}), w(f_{i*})]$ とする。

Step 4: もし $f_{i*} \in F_{i-1}(k-1)$ ならば、 T_{i-1} から $F_{i-1}(k-1)$ を取り除く。もし $f_{i*} \notin F_{i-1}(k-1)$ ならば、 T_{i-1} から $f_{i*} \cup F_{i-1}(k-2)$ を取り除く。

Step 5: Step 4 で得られた k 個の成分をつなぐ G の枝集合を出力し、終了する。

定理 3 上の手続きは最大格差最小 k -カットを $O(m + n \log n)$ 時間で正しく求める。

3. 最小 k -カット問題の近似アルゴリズム

本論文で提案する最小 k -カットを求める新しい近似アルゴリズムの詳細は以下の通りである。

手続き Approximate-Minimum-k-Cut

入力: グラフ $G = (V, E)$ 。

出力: 最小 k -カットの近似解 $\hat{S}$ 。

Step 1: $l :=$ 繰り返し回数, $\nu^* := \infty$, $k := 1$ とする。

Step 2: G の各枝 e に区間 $[0, 1]$ の一様乱数によって重み $w(e)$ を付加する。

Step 3: G の最大格差最小 k -カット S を手続き Minimum-Range-k-Cut によって求める。

Step 4: $|S| < \nu^*$ ならば、 $\hat{S} := S$; $\nu^* := |S|$ とおく。

Step 5: $k := k + 1$ 。 $k \leq l$ なら Step 2 へ戻り、そうでなければ Step 6 へ。

Step 6: $\hat{S}$ を出力し、終了する。

定理 4 手続き Approximate-Minimum-k-Cut の計算量は $O(lm + ln \log n)$ である。

上の手続きは容量付きのネットワークに拡張できる。容量が正の整数の場合、各枝に容量分の多重枝を導入し、多重グラフとみなして上の手続きで計算できる。この方法では一回の最大格差最小 k -カットを求めるのに $O(mC + n \log n)$ の時間かかる。(但し、 C は容量の最大値である。) これは擬多項式オーダーであり多項式時間アルゴリズムではないが、以下の方法によりより実用的な多項式時間のアルゴリズムを得ることができる。もとの容量付きネットワークの枝の整数容量を $c(e)$ とする。上の議論より、最大木と最小木の枝が分かれば、最大格差最小 k -カットの計算ができる。従って、 $c(e)$ 本の枝に一樣にランダムの変数を付すことは必要ではない。代わりに、 $c(e)$ 個の一樣ランダム変数のうち、重みの最大値と最小値 (それぞれ y' と y で表す) を求めれば十分である。これは以下のように求められる。 x と x' を区間 $[0, 1]$ の独立一様分布によって生成された二つのランダムの変数とする。簡単のため、 $x < x'$ とする。そうすると、 y と y' は以下の式より計算できる。

$$x = 1 - (1 - y)^{c(e)/2}, \quad x' = (y')^{c(e)/2}. \quad (1)$$

詳しくは [2] を参照されたい。 y と y' を計算した後、もとのネットワークの各枝に y と y' を重みとする二本の枝を導入し、多重グラフを構成する。この多重グラフに対して上の手続きを施すと、もとのグラフの最大格差最小 k -カットが $O(m + n \log n)$ の時間で求められる。容量が実数であるようなネットワークについては、まず最小容量を 1 であるようにスケーリングした後、上の手続きを適用することができる [2]。

4. アルゴリズムの実験結果

本節では前節で提案したアルゴリズムの計算機実験の結果について述べる。実験は SUN SPARC IPX 上の Pascal

で行った。実験対象のネットワークは [11] の方法によって次のように生成した。

ネットワークは次のようなパラメータによって決められている。1) グラフの点数 n ; 2) 枝の疎密度 $d(\%)$; 3) 分割数 c ; 4) 容量を決めるパラメータ p (今後クラスター間密度と呼ぶ), $0 \leq p \leq 1$ 。ネットワークを生成するとき、まず n 個の節点と $n-1$ 本の枝から連結グラフ (鎖状グラフ) を作る。次に分割数によってランダムに c 個の節点部分集合を決める。枝の容量は次のように決める。両端点異なる節点集合にある枝に対して区間 $[1, 10^5 p)$ の一様乱数で容量を与え、両端点同じ節点集合にある枝に対して $[1, 10^5)$ の一様乱数で容量を決める。

今回の実験では、グラフの点数を $n = 150$ 、分割数を $c = 8$ と固定した。 p の値は 0.001, 0.01, 0.1, 1 とし、各 p の値に対して、疎密度 d を 10% ~ 50% まで 10% ずつ変化させ、対 (p, d) 毎に 10 個のネットワークを生成した。この生成法によって、実際の VLSI 設計などにでてくる複数のクラスターがある密度で結合されているようなネットワークが構築できることになる。

実験結果は 10 個のネットワークの結果の平均をとった。以下、簡単のために新しい近似アルゴリズムは AM と略し、比較の対象となった Saran と Vazirani [15] のアルゴリズムは SV と略す。SV における最小カットを求めるとき、[10] の高速アルゴリズムを採用した。また、アルゴリズム SV の所要した CPU 時間内にアルゴリズム AM のできる繰り返しの回数を l^* とする。(l^* がそれぞれのケースにおいて異なることに注意。)

表 1 はアルゴリズム AM が l^* 回の繰り返しで求めた解をアルゴリズム SV と比較した結果を示している。表 1 の #1, #2, #3 は SV と比較してそれぞれ良い解, 同じ解, 良くない解を見つけたケースの数である。表 1 によると, $k = 3, 4$ の場合, SV と同じカットもしくはよりよいカットを求めた割合は約 80.5% である。うち, SV より良いカットを求めた割合は約 2.75% である。

図 1 ~ 図 4 は, AM が初めて SV よりも良い解もしくは同じ解をレポートした時の繰り返しまでの CPU 時間を示したものである。図に示すように, アルゴリズム SV と比較して, アルゴリズム AM の方が SV の約 $1/3 \sim 1/4$ の時間で済むことが分かる。

図 5 は $k = 3, p = 0.01$ の場合に AM において, 第 i 番目の繰り返しまでに得られた解を SV の解と比較したものである。縦軸の値はアルゴリズム AM の求めた解の値とアルゴリズム SV のそれとの比である (AV の求めた解の値/SV の求めた解の値)。この図に見られるように, AM はこのパラメーターの場合には非常に早い時期に良い解を出すことが分かる。同じような結果はほかの小さい p, k についても得られた。

図 6 ~ 図 9 は異なる p に対してアルゴリズム AM が l^* 回の繰り返しまでに求めた解とアルゴリズム SV の求めた解との相対精度を表す。縦軸の値はアルゴリズム AM の求めた

表 1. AM と SV の結果の比較

Table 1. The comparison of results of AM and SV

p		$k = 3$	$k = 4$	$k = 6$	$k = 8$
0.001	#1	0	1	0	12
	#2	50	49	49	32
	#3	0	0	0	6
	l^*	23	33	48	56
0.01	#1	4	5	4	2
	#2	46	45	15	7
	#3	0	0	31	41
	l^*	22	33	53	62
0.1	#1	0	1	0	0
	#2	45	22	1	0
	#3	5	27	49	50
	l^*	21	32	54	74
1	#1	0	0	0	0
	#2	47	8	0	0
	#3	3	42	50	50
	l^*	21	32	52	68

解の値とアルゴリズム SV のそれとの比である (AV の求めた解の値/SV の求めた解の値)。(図 6 において $k = 3, 4, 6$ の直線がほぼ重なっていることに注意する。)図からもわかるように, すべての解は SV の解の 1.5 倍以内に収まっている。また, $k = 3, 4$ の場合, アルゴリズム AM がネットワークの疎密度と関係なく安定していることが認められる。

5. むすび

本研究は最小 k -カット問題に対して新しい近似アルゴリズムを提案し, 数値実験によってその評価を行った。その結果, $k = 3, 4$, クラスター間密度 p が 0.01 より小さい場合は [15] の近似アルゴリズムと比較して, 平均的に 5 ~ 10 回の繰り返し (SV の約 $1/3 \sim 1/4$ の時間) で SV と同等もしくはより良い解が得られた。

回路分割などの実用の場面においては, 再帰的に回路を分割していく手法が用いられる [8] が, この際に $k = 2, 3, 4$ などの分割数を用いることは自然である。その際にアルゴリズム AM は有用であると考えられる。

さらに, 本アルゴリズムは, [2] で提案した $s-t$ カットの近似解法にもとづき, 目標点をもつ k 個の成分に分割する問題 [1] を求めるアルゴリズムにも同じ計算複雑度で簡単に拡張できる (Saran と Vazirani の方法はこの問題には適用できないことに注意)。また, [2] で議論されているように, 等分割 k カットアルゴリズムにも拡張可能である。

分割数, クラスター間密度, k の値について, より広範囲な実験を行なうこと, さらに実用規模 ($n \geq 10,000$) のグラフについて実験を行なうこと, $k = 3, 4$ の場合を再帰的に用いて複数分割を行なう場合の性能評価を行なうこと等が今後の課題である。

図 1. AM と SV の CPU 時間
Figure 1. CPU time of the two algorithms

図 2. AM と SV の CPU 時間
Figure 2. CPU time of the two algorithms

図 3. AM と SV の CPU 時間
Figure 3. CPU time of the two algorithms

図 4. AM と SV の CPU 時間
Figure 4. CPU time of the two algorithms

図 5. 繰り返し回数に対する AM のふるまい
Figure 5. Behavior of approximate minimum k-cut algorithm with respect to number of iterations

図 6. AM と SV が求めた解の相対精度
Figure 6. ratio of solutions found by two algorithms

図 7. AM と SV が求めた解の相対精度
Figure 7. ratio of solutions found by two algorithms

図 8. AM と SV が求めた解の相対精度
Figure 8. ratio of solutions found by two algorithms

図 9. AM と SV が求めた解の相対精度
Figure 9. ratio of solutions found by two algorithms

(平成5年8月19日受付)

文 献

- [1] Dahlhaus, E., D.S. Johnson, C.H. Papadimitriou, P.D. Seymour, and M. Yannakakis, The complexity of multiway cuts, *the Proc. ACM 24th Symp. Theoretical Computer Science*, (1992), pp. 241-251.
- [2] Dai, Y., H. Imai, K. Iwano, N. Katoh, K. Ohtsuka, and N. Yoshimura, A New Unified Approximate Approach to the Minimum Cut Problem and its Variants Using Minimum Range Cut Algorithms, IBM Research Report, RT0082, Nov. 1992.
- [3] Even, S. and R.E. Tarjan, Network flow and testing graph connectivity, *SIAM J. Computing*, Vol. 4 (1975), pp. 507-518.
- [4] Fredman, M.L. and R.E. Tarjan, Fibonacci heaps and their uses in improved network optimization algorithms, *Journal of the ACM*, Vol. 34, No. 3, (1987) pp. 596-615.
- [5] Goldschmidt, O. and D.S. Hochbaum, Polynomial algorithm for the k -cut problem, *Proc. 29th IEEE Symp. Found. Compt. Sci.*, (1988), pp. 444-451.
- [6] Gomory, R.E. and T.C. Hu, Multi-terminal network flows, *J. SIAM*, Vol 9, No.4 (1961), pp. 551-570.
- [7] Katoh, N. and K. Iwano, Efficient algorithms for minimum range cut problems, IBM Research Report, RT0057 (1991) (also appeared in *Proceedings of 2nd Workshop, WADS '91*, Lecture Notes in Computer Science 519, Springer-Verlag, (1991), pp. 80-91).
- [8] Lengauer, T., *Combinatorial Algorithms for Integrated Circuit Layout*, John Wiley & Sons, West Sussex, England, 1990.
- [9] Matula, D.W., Determining edge connectivity in $O(nm)$, *Proc. 28th IEEE Symp. Found. Compt. Sci.*, (1987), pp. 249-251.
- [10] Nagamochi, H. and T. Ibaraki, Computing edge-connectivity in multiple and capacitated graphs, *SIAM J. Discrete Math.*, 5 (1992), 54-66.
- [11] Nagamochi, H., T. Ono, and T. Ibaraki, An efficient implementation for minimum capacity cut algorithm, 情報処理学会アルゴリズム研究会研究報告, 93-AL-33-1, (1993), pp. 1-8.
- [12] Picard, J.C. and M. Queyranne, Selected applications of minimum cuts in networks, *INFOR*, Vol. 20 (1982), pp. 394-422.
- [13] Sleator, D.D., and R.E. Tarjan, A data structure for dynamic trees, *Journal of Computer and System Sciences*, 26 (1983), pp. 362-391.
- [14] Stone, H.S., Multiprocessor scheduling with the aid of network flow algorithms, *IEEE Trans. Software Engineering*, SE-3 (1977), pp. 85-93.
- [15] Saran, H. and V.V. Vazirani, Finding k -cuts within twice the optimal. *Proc. 32nd IEEE Symp. Found. Compt. Sci.*, (1991), pp. 743-751.

戴 陽

(非会員) 昭和59年上海海運学院管理学科卒業。平成元年筑波大学社会学部工学研究科博士課程前期課程修了。平成3年同研究より博士(経営工学)。平成3年より神戸商科大学管理科学科助手。数理計画法に関する研究に従事。

岩野 和生

(非会員) 昭和50年東京大学理学部数学科卒業。同年日本アイ・ビー・エム(株)入社。昭和57年より米国プリンストン大学工学及び応用科学部大学院。昭和59年同学部電気工学及びコンピュータサイエンス学科よりMSE及びMA, 昭和62年同学部コンピュータサイエンス学科よりPh.D取得。現在, 日本アイ・ビー・エム(株)東京基礎研究所先進情報科学担当。主な研究テーマはアルゴリズムの設計と応用, 組合せ最適化。IEEE, ACM, SIAM, Sigma Xi 各会員。

加藤 直樹

(非会員) 昭和48年京大・工・数理卒。昭和50年同大大学院修士課程修了。昭和56年工学博士。昭和53年より大阪府立成人病センター情報企画室勤務。昭和56年神戸商科大学管理科学科講師。昭和57年同助教授, 平成2年同教授。平成2年度日本オペレーションズ・リサーチ学会文献賞受賞。組合せ最適化アルゴリズム, データベースの一貫性制御, 計算幾何学に関する研究に従事。