4Q–09　配線可能性検証のためのフロー計算アルゴリズム

磯 直行　　平田 富夫

名古屋大学 大学院 工学研究科

1　はじめに

配線可能性検証とは，概略配線が詳細配線へ変換できるかどうかを判定することである．平面配線の場合，いくつかの検証アルゴリズムが知られているが，これらは皆，概略配線を与えたとき，クリティカルカット（端子点または障害物の頂点同士を結ぶ線分）についてそれと交差する概略配線数（フロー）が，そこを通過することのできる詳細配線の数（容量）を越えるか否かを調べることで判定を行っている．筆者らは[1, 2]で配線可能性検証のための二つのグラフ，“容量判定グラフ $\mathbf{G}_c$” と “フロー導出グラフ $\mathbf{G}_f$” を提案した．$\mathbf{G}_c$はフローと容量の比較を行わなければならないクリティカルカットを辺とするグラフである．また，$\mathbf{G}_f$は$\mathbf{G}_c$から一部の辺を除いてできるグラフで，このグラフの辺についてフローを知っていれば，$\mathbf{G}_c$の他の辺についてもフローが計算で求めることができる．概略配線処理によって$\mathbf{G}_f$の各辺についてフローを定め，次に，それを用いて$\mathbf{G}_c$のすべての辺のフローを計算する．最後に$\mathbf{G}_c$の各辺ごとにフローと容量の比較をして配線可能性を判定する．

本論文では，本配線可能性検証アルゴリズムの実行時に使用するメモリ量（記憶量）を削減する．つまりメモリ上に保持しておくデータは$\mathbf{G}_f$のみとし，$\mathbf{G}_c$は保持しない．$\mathbf{G}_f$に含まれない辺のフローが必要となったときには随時$\mathbf{G}_f$から計算して求めるが，そのためのアルゴリズムを与える．

2　記憶量の削減とフロー計算

改良された配線可能性検証アルゴリズムは，クリティカルカットを見つける毎にその容量とフローを比較する．クリティカルカットの発見は$\mathbf{G}_c$を列挙するアルゴリズム[1, 2]を利用する．発見されたクリティカルカットの両端点をa, bとすると，線分(a, b)は$\mathbf{G}_f$の辺と一致または$\mathbf{G}_f$の辺と交差する．$\mathbf{G}_f$の辺と一致する場合は何もする必要はない．一方，$\mathbf{G}_f$の辺と交差する場合には，交差する$\mathbf{G}_f$の辺を対角辺とする凸四辺形に次々にフリップ操作を適用しクリティカルカットのフローを計算する（[2]の図7）．

クリティカルカットabと交差する辺をaから近い順に$E_{ab} = \{e_1, e_2, \cdots, e_m\}$とする．$a, b$を通る直線により分けられる二つの半平面を$R, L$とする．$E_{ab}$内の辺の端点のうち$L$内にあるものを$E_{ab}$のインデックス順に並べ$s_1, s_2, \cdots, s_{\alpha-1}$とする．$R$内にあるものも同様に$t_1, t_2, \cdots, t_{\beta-1}$とする．ただし，$s_0 = t_0 = a, s_\alpha = t_\beta = b$である（図1）．

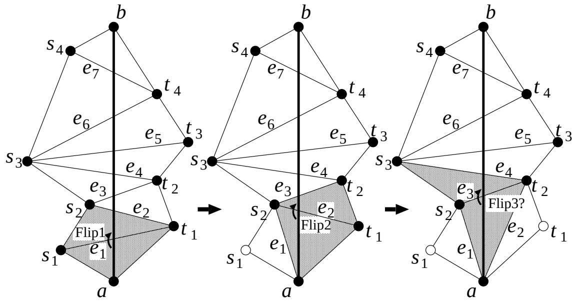

図 1: フリップ操作を連続してできない例

フリップすべき四辺形を選択する一つの方法は，E_{ab}のインデックス順にその辺を対角辺とする四辺形を調べ，それがフリップ操作可能な場合にフリップを行うことである．このとき四辺形が凸四辺形でなかったり，フリップをしても新しい対角辺がまたクリティカルカットと交差してしまい交差辺の数が減らない場合（図1のe_3）にはその辺に関するフリップ操作は行わず次のインデックスの辺から再び適用できるものを探す．このフリップの結果，今までフリップせずに飛ばしてきた辺に対する四辺形がフリップ可能になるのであればそれらについてもフリップ操作を適用する．このようにしてクリティカルカットのフローを計算する方法をNearestFirstとよぶことにする．しかし，この方法は交差辺の数の2乗に比例した時間を必要とする場合がある．

以下では，最悪の場合にはこれと同程度の手間を必要とするが，実用的にはより高速なアルゴリズムを与える．

E_{ab}を保存するためのスタック$\mathcal{S}$を用意する．アルゴリズムはまず前処理として次の手続き1を行う．

手続き 1

```
スタック $\mathcal{S}$ を空にする;
for $i := 1$ to $m$ do begin
    スタック $\mathcal{S}$ に $e_i$ を $push$ する;
    repeat
        while (1. $\mathcal{S}$ の先頭の辺 $e$ を対角辺とする
        四角形 $\mathcal{T}$ が凸で，かつ，$\mathcal{T}$ の点が $R$ 内に 3
        点または $L$ 内に 3 点が存在する) do begin
```

Flow Calculation for Planar Routability Checking
Naoyuki ISO and Tomio HIRATA
Graduate School of Engineering, Nagoya University
Furo-cho, Chikusa-ku, Nagoya-shi, 464-8603, Japan

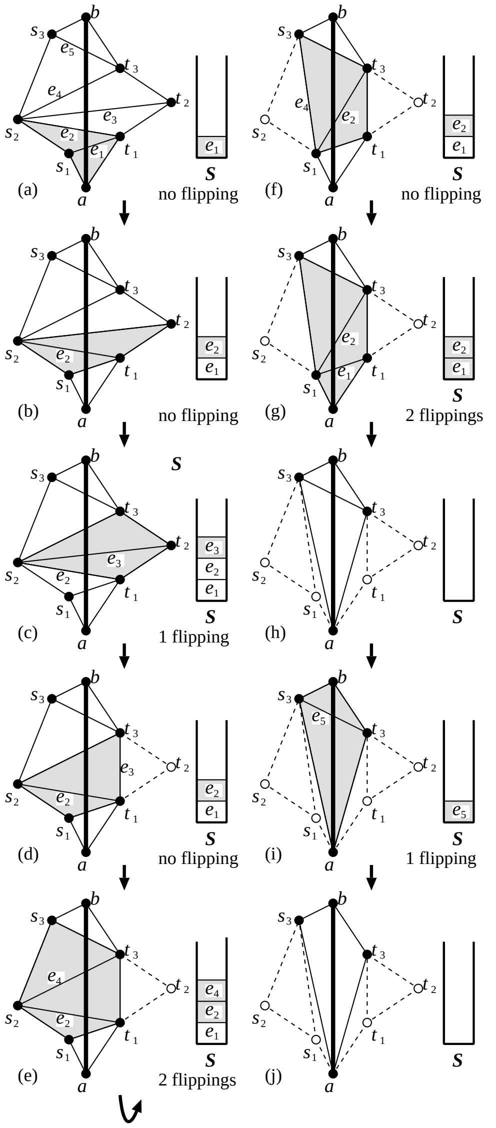

図 2: フリップ操作の適用順序

```
        その3点を $u_i, u_{i+1}, u_{i+2}$ とする．$\mathcal{T}$にフリップを適用し，新たな $(u_i, u_{i+2})$ のフローを計算する（図 2(c)）．スタックから $e$ を pop する．
    end;
    while （2. $\mathcal{S}$の先頭の辺 $e'$ と2番目の辺 $e''$ を対角辺とする五角形 $\mathcal{P}$が凸である） do
    begin
        このとき，$L$ または $R$に $\mathcal{P}$の輪郭線に沿って連続した 3 点 $u_i, u_{i+1}, u_{i+2}$が存在し，$u_{i+1}$は $e'$と $e''$の共通の端点である（図 2(e)）．$e''$と $e'$についてこの順にフリップを適用し，辺 $(u_i, u_{i+2})$ のフローを計算する（図 2(f)）．スタックから $e'$ を pop する．
    end;
until （1, 2 のいずれにも該当しない）
end;
```

表 1: 使用メモリ量と実行時間

部品数	ネット数	M_1[byte]	M_2[byte]	T_1[s]	T_2[s]
68	828	161k	52k	0.96	15.72
124	1891	380k	98k	1.08	82.66
134	2744	408k	106k	1.16	94.47

（計算機: SPARCstation 10）

手続き 1が終了すると，L, Rそれぞれに凸頂点を一つだけ残した図形を得る（図 2(j)）．この凸四辺形の対角辺がクリティカルカット ab である．このアルゴリズムは 1 回または 2 回のフリップ操作で交差辺を 1 本消去するので，大抵の場合，クリティカルカットのフローは $O(n)$ 時間で計算できる．しかし，特殊な点配置の場合には手続き 1の終了後にスタック $\mathcal{S}$に交差辺が残ってしまう場合がある．このときは，その得られた結果に対して NearestFirst を適用しフリップを繰り返す．

3 プリント配線板データへの適用

本節では，上で述べた配線可能性検証アルゴリズムを計算機上に実装し，3 種類の実際のプリント配線板部品配置データに適用した結果を示す．表 1に，改良前および改良後のアルゴリズムのメモリ量をそれぞれ M_1, M_2で示す．本改良により 26 %から 32 %のメモリしか使用しないことがわかる．

T_1, T_2は改良前および改良後のアルゴリズムを実行したときの時間である．フローの再計算を行っているため実行時間が増えているが，実用上十分な時間で配線可能性検証が実現できていることがわかる．

4 おわりに

概略配線を実際に平面上に配線できるかどうかを判定する配線可能性アルゴリズムを改良し，実行時に必要とする記憶量を削減した．

謝辞　本研究に関し，実験用プリント配線板設計データを提供いただきました株式会社日立製作所オフィスシステム事業部 の皆様に感謝致します．本研究の一部は財団法人 堀情報科学振興財団の援助を受けた．記して感謝します．

参考文献

[1] Naoyuki Iso, Yasushi Kawaguchi, Tomio Hirata, “Efficient Routability Checking for Global Wires in Planar Layouts,” IEICE Trans. Fundamentals, Vol. E80–A, No.10, pp. 1878–1882, 1997.

[2] 川口 泰，磯 直行，平田富夫，“配線可能性検証のための容量判定グラフとフロー導出グラフ,” 信学論，A, Vol. J80-A, No.1, pp. 135–142, 1997.

[3] C.Leiserson and F.Maley : “Algorithms for routing and testing routability of planar VLSI layouts,” Proc. of 17th ACM Symp. on Theory of Computing, pp.69–78, 1985.