Vol. 30 No. 2 情報処理学会論文誌 Feb. 1989

タイル平面に基づく最小曲がり径路探索アルゴリズム†

佐藤政生†† 坂中二郎††† 大附辰夫†††

迷路法と線分探索法が持つ長所を継承し，短所を克服する新しい配線手法であるグリッドレス・ルータに関する研究が数年前よりなされるようになった．しかし，最小曲がり径路探索手法としては，効率の良くないものや，複雑なデータ構造によるものしか知られていない．そこで本論文では，最小曲がり径路探索問題に焦点をあて，タイル平面と呼ばれる単純なデータ構造に基づいた，高速に径路を求めるグリッドレス・ルータを提案する．このルータは，$O(n \log n)$ の手間の前処理後，$O(k)$ の手間で径路を，$O(k \log k)$ の手間で最小曲がり径路を求めるとともに，$O(k)$ の手間で図形データの更新を行う．ここで，n は図形の頂点数，k (通常，$k \ll n$) は径路を求める際に探索した図形の頂点数である．このときの空間複雑度は $O(n)$ である．また，提案した手法をインプリメントした結果を報告するとともに，従来の線分探索法との比較を行う．

1. まえがき

電子回路のレイアウト設計における配線手法としては，迷路法と線分探索法が広く普及しているが，それぞれ長所と短所を持つ．迷路法は，径路が存在すれば必ずそのうちの一つを求めるが，格子構造によってレイアウトを表現しているので，膨大な記憶容量を必要とし高速に径路を求めることができない．一方，線分探索法は，単純な径路は少ない記憶領域内で高速に求めることができるが，場合によっては迷路法よりも処理時間を費やす[1]か，径路が存在しても発見する保証がない[2]．

これらの配線手法に代わるものとして，1980年初頭より，レイアウト・データを図形そのものとしてとらえ，計算幾何学[3]の分野における高度なアルゴリズムやデータ構造を導入することによって，少ない記憶容量で高速に径路を求める配線手法が提案されるようになってきた[4]~[12]．これらの手法をグリッドレス・ルータと呼ぶ．

配線可能な場合には，レイアウト・データの増大を防ぐ，断線を起こしやすい配線の折れ曲がりを減らす，等の理由のために，いくつか存在する径路のうち最も単純な，つまり曲がり数が最も少ない径路を求めることが多い．最小曲がり径路を求めるグリッドレス・ルータとしては，$O(n \log^2 n)$ の手間と $O(n)$ の記憶容量[5),12)]，もしくは $O(n \log n)$ の手間と記憶容量[5),6),13)]で径路を求めるものが知られているが，使用するデータ構造が複雑なものであるので，単純な算法のものよりも多くの記憶容量を必要としていた．ここで，n は図形の頂点数である．また，1本の径路を求めた後には次の配線径路を求めるために，得られた径路を禁止領域とみなすように図形情報を更新する必要があるが，その際にも複雑な更新操作を行わなければならなかった．

そこで本文論では，単純なデータ構造に基づき，かつ，高速に径路を求める手法を提案する．この手法は，一層配線モデルにおいて，$O(n \log n)$ の手間の前処理後，$O(k \log k)$ の手間でネットごとの最小曲がり径路を探索し，径路が存在する場合には $O(k)$ の手間で図形データの更新を行う．ここで，k (通常，$k \ll n$) は径路を求める際に探索した図形の頂点数である．このとき空間複雑度は $O(n)$ である．したがって，繰返し径路探索が必要となる配線設計では非常に有効な手法であると考えられる．(単に，径路を求める場合は $O(k)$ の手間の径路探索で十分である．) また，提案した手法をインプリメントした結果を報告するとともに，従来の線分探索法との比較を行う．

以下では，議論を簡単にするために，配線領域は垂直および水平な線分で囲まれた領域(複合長方形領域)とし，その領域は縮退していないものとする．また，一層で2点 S, T 間の配線を行うものとし，このとき求める配線径路は斜め線を含まないものとする．

2. 前処理

次段の径路探索を効率良く行うために，以下のような前処理を施す．

まず，配線領域 (図 1) を配線幅，配線間隔を考慮することにより図 2 のように縮小する．このとき既配

† A Minimum Bend Path Algorithm Based on a Tile Plane by MASAO SATO (Department of Information Engineering, Faculty of Engineering, Takushoku University), JIRO SAKANAKA and TATSUO OHTSUKI (Department of Electronics and Communication Engineering, School of Science and Engineering, Waseda University).

†† 拓殖大学工学部情報工学科
††† 早稲田大学理工学部電子通信学科

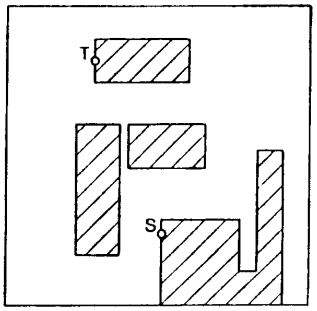

図 1 複合長方形配線領域
Fig. 1 A rectilinear routable region.

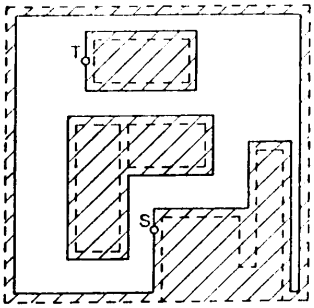

図 2 配線領域の縮小
Fig. 2 Virtual shrinking of the routable region in Fig. 1.

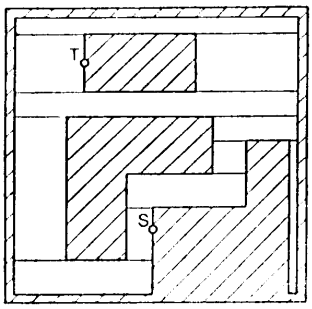

図 3 タイル平面：配線領域の長方形分割
Fig. 3 A tile plane: Partitioning a routable region into rectangles.

線はブロックと同様に禁止領域とみなされる．縮小後は，S, T 間の径路をその中心線で表すことができる．次に，縮小された配線領域を凹な頂点（内角が 270° の頂点）を通る水平な分割線で図 3 のように長方形（**タイル**）に分割する．生成されたタイルとその隣接関係を表現するデータ構造（**タイル平面**：tile plane[14]）を作成する．

前処理は以上で終了する．縮小操作および領域分割操作はそれぞれ $O(n \log n)$ の手間と $O(n)$ の記憶容量で行うことができる[15),16)]ので，前処理に必要な手間は $O(n \log n)$ で抑えられる．

単に 2 点間の径路を求めるのであれば，こうしてできたタイル平面上で S, T が属するタイル間の道を求め，さらに，この道を基に実際の径路を定めればよい．これは高々 n 節点の平面グラフ上で道を求める操作と同等であるから $O(k)$ の手間で行えることは明らかである．ここで，k ($k<n$) は探索した分割線の数である．

また，S, T が属するタイル間の道を求める際に，タイルの縦方向の幅をコストとして考え，ヒープ[17)]を用いたダイクストラ法を適用すれば，S, T 間の縦方向最短径路を求めることも可能である．このときの手間は $O(k \log k)$ となる．

次章では，以上のような径路探索法を若干凝ることにより，同程度の手間で曲がり数が最小の径路が求まることを示す．

3. 最小曲がり径路探索

本章では，前節の前処理によって作成された分割線を利用して，径路が存在するときにはその中でも曲がり数が最小のものを求める算法を提案する．

3.1 分割線とコスト

平面上の 1 点と始点 S を結ぶために最低必要な線分の数（すなわち，最小曲がり数＋1）をその点の**コスト**と呼ぶことにする．分割線上におけるコストの分布の一例を**図 4** に示す．次の定理が成り立つことは明らかであろう．

［定理 1］ おのおのの水平な分割線上の点のコストの最大値と最小値の差は高々 1 である．

この定理は提案する手法の中で重要な働きをするものであり，手法の正当性や効率の良さを保証するための基本的な性質である．

次に，始点 S から水平方向に配線すると限定した場合と垂直方向に配線すると限定した場合の 2 通りの場

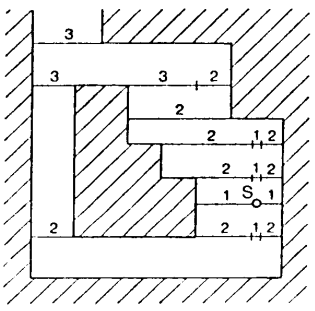

図 4 分割線とコスト
Fig. 4 Costs on slicing line segments.

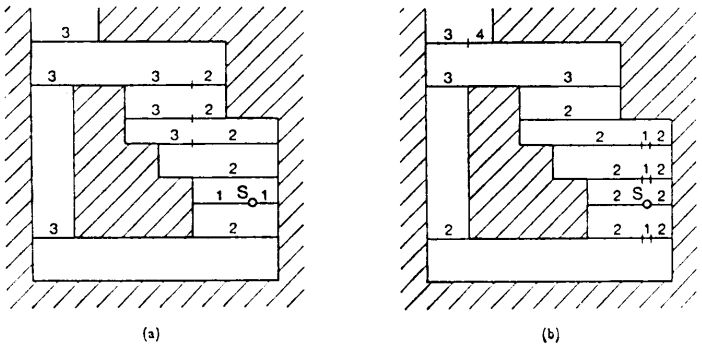

図 5 始点Sから水平方向に配線すると限定した場合のコスト(a)と垂直方向に配線すると限定した場合のコスト(b)

Fig. 5 Two cases of distribution of costs on slicing line segments: Every wiring path starts horizontally (a) and vertically (b) from a source point S.

合を考える．それぞれの場合のコスト分布を**図5**に示す．ただし図5では，後述の径路探索アルゴリズムの記述を容易にするために，始点Sから水平（垂直）方向に配線する場合にはSを通る垂直（水平）線上には終点Tが存在しないものとし，その直線から微小距離だけ離れた所のコストを示している．このようにしても，図4のコスト分布が図5の二つのコスト分布より得られることは明らかであろう．いま，始点Sから水平方向に配線すると仮定すると，Sに関する分割線のコストは1であり，次にその分割線上から曲がって垂直線分で到達できる範囲のコストは2となる．このように線分から水平および垂直方向に探索する操作を交互に繰り返すことにより，分割線上で水平方向に進む径路を選択すると最小曲がり径路が得られる範囲には奇数，垂直方向の場合には偶数のコストを与えることとなる．この性質を定理にしてまとめる．

［定理2］ 始点Sから水平（垂直）方向に配線すると限定した場合，分割線上で水平方向に進む径路を選択すると最小曲がり径路が得られる範囲は奇数（偶数），垂直方向に進む径路を選択すると最小曲がり径路が得られる範囲は偶数（奇数）のコストを持つ．

この定理より，次の性質が得られる．

［定理3］ 始点Sから水平（垂直）方向に配線すると限定した場合には，分割線上には次の3種類のコストの分布しか起こらない．

① 偶数で一定である，

② 奇数で一定である，

③ ある偶数（奇数）とそれより1大きい奇数（偶数）が交互に分布する．

3.2 ラベル付け（コスト情報の伝達）

提案する径路探索手法は，基本的にはダイクストラ法であり，まずSに関する分割線に図5のようなラベル（コスト情報）を付ける．そして，この分割線が属するタイルと同じ周囲上に位置する分割線から順にコスト情報を伝達してゆく．そして，終点Tに付いたラベル値を更新するような，いかなる径路も存在しないことが明らかになったときラベル付けは終了し，逆追跡によって径路を求める．実際には，このラベル付けは始点Sから横方向に配線する場合と縦方向の場合を同時に扱うが，以下では説明と理解をしやすくするために，始点Sから横方向に配線する場合についてのみ説明する（横方向および縦方向に関する操作は基本的に等しい）．また，ある分割線からラベル付けすべき分割線は，上方のものと下方のものの2通りがあるが，操作は本質的に等しいので上方のものを中心に述べることにする．

Step 1. ヒープを設ける．最初，ヒープは空である．この木は通常のダイクストラ法で用いられるものと同様の働きをし，分割線をそのラベル値に基づいて保持する（このとき，分割線が次に進むべき方向も保持している）．ただし，ラベル値は常に一定値であるとは限らないので，次のような優先順序で分割線を保持するようにする．

1. 一定のコスト $2t$ を持つ分割線．
2. $2t$ と $(2t+1)$ のコストを持つ分割線．ただし，同じコストを持つ分割線が複数存在する場合には，進むべき方向が上（下）方のものは y 座標の小さい（大きい）順にする．
3. 一定のコスト $(2t+1)$ を持つ分割線．

ここで，t は整数である（ただし，コストは1以上である）．つまり，コスト一定の分割線は通常どおりに扱い，2種類のコスト a, b $(a<b)$ を持つ分割線は，一定コスト a の分割線よりは低く，一定コスト b の分割線よりは高い優先順位を持つものとして扱う．同じ2種類のコストを持つ分割線を y 座標順に順位付けするのは，一つの分割線が探索される回数を定数回に抑えるためである．

以下では，ヒープに含まれる分割線のラベルを**仮ラベル**，ヒープから取り出された分割線のラベルを**永久ラベル**と呼ぶことにする．

Step 2. 結線すべき2点 S, T を通る分割線を作成する．始点Sに関する分割線に一定コスト1のラベルを付ける．そして，ヒープにこの分割線から上方に

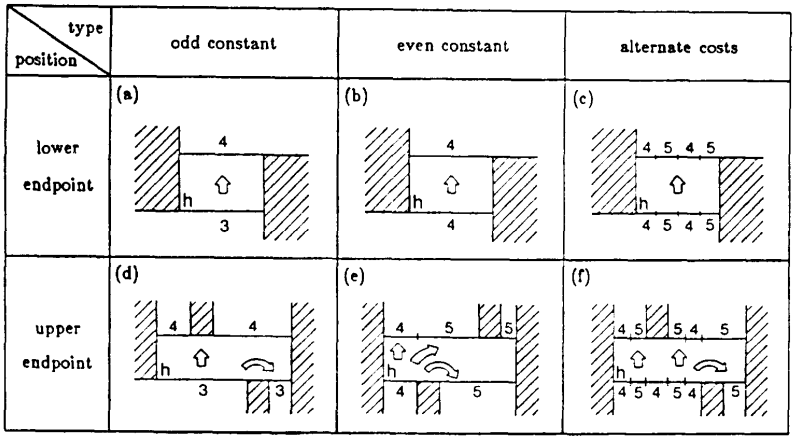

図 6 ラベル付け：コスト情報の伝達
Fig. 6 Labeling: Transmission of cost information.

探索してゆくものと，下方に探索してゆくものをそれぞれ挿入する．

Step 3. ヒープから分割線 h を取り出す（ヒープが空ならば径路は存在しない）．h が終点Tを含んでいるか，もしくは，h の持つラベル値がTに付いているコストより小さくなければ，Step 5 へ．そうでなければ Step 4 へ．

Step 4. h が持つラベルの種類，および，h の端点である図形の凹点が境界を構成する線分の上端点か下端点かによって，以下のいずれかの操作を行う（**図 6** 参照）．

（1） もし h が奇数一定のラベル c を持つならば，$c+1$ のコストを上方で隣接する分割線にラベルとして与える（図 6(a)）．このとき，もし h と同じ y 座標を持つ隣接分割線が存在するならば，コスト c をラベルとして与える（この状態は図 6(d)のように縮退しているときにのみ起こる）．

（2） もし h のラベルに偶数 c があるならば，上方の隣接分割線で h と同じ x 座標を被覆する部分に h と同じラベルを与える．そして，隣接分割線のその他の部分にはコスト $c+1$ をラベルとして与える（図 6(b)，(c)，(e)，(f)参照）．

このとき探索した分割線がすでにラベル付けされているときには，ラベル値の小さいほうを採用する（ラベル付けの競合に関しては次節参照）．h にラベルの種類とコスト（2種類のコストを持つ場合には値の大きいほうを採用し，仮ラベルが持つ2種類のコストの分布に関する情報は持たない），および h が探索された方向を永久ラベルとして残す．また，h によって初めて探索された分割線をヒープに挿入し，Step 3 へ．

Step 5. 逆追跡を行い径路を求める．

以上が最小曲がり径路探索法である．**図 5** のようなラベル付けが必要かつ十分な理由は，定理2を考慮すれば明らかであろう．この探索法で用いられるデータ構造の詳細，および，算法の時間複雑度，空間複雑度については，3.4 節で述べることにする．

3.3 ラベル付けの競合

径路探索時におけるラベル付けの競合について考察する．いまヒープから取り出された分割線を h，h からのラベル付けの対象となる分割線を g とする．h および g が持つラベル値がともに一定のコストの場合，もしくは，一方が一定のコストで他方が2種類のコストを持つ場合には明らかに，どちらの分割線を通過する径路を選択すべきかの判定を直ちに行うことができる．問題となるのは，同じ2種類のコストを持つラベルが競合する場合である．この場合には，h と g が進むべき方向によって以下の2通りがある．

① h も g も上方に進む場合．

この場合に問題となるのは，**図 7** のようなものだけである．つまり，h と g が持つ x 座標の区間に共通部分があり，g が h よりも上方に位置する場合である．このようなときには，同図のように g 上の共通部分に h が持つコスト情報を付加する．ここで注意すべきことは，3.2 節 Step 1 におけるヒープ内の分割線の優先順位の定義より，g のラベルは必ず仮ラベルであることであり，このことは g によって g より上方の分割線はまだ探索されていないことを示している．

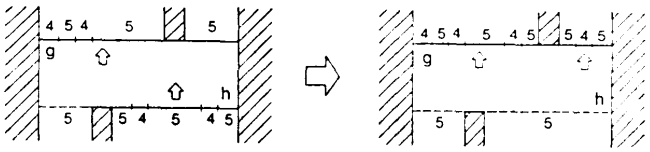

図 7 同方向に進む分割線のラベル付けの競合
（点線は永久ラベル，実線は仮ラベルを持つ分割線を表す）
Fig. 7 Labeling conflict caused by two slicing line segments with a same direction (slicing line segments indicated by broken lines have permanent labels and those indicated by solid lines have temporary labels).

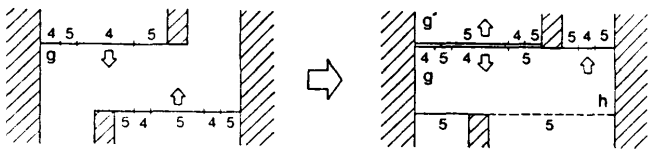

図 8 逆方向に進む分割線のラベル付けの競合
(図中では未探索の分割線は省略している)

Fig. 8 Labeling conflict caused by two slicing line segments with opposite directions each other (we omit unsearched slicing line segments in this figure).

② h と g の進むべき方向が逆の場合.

この場合にも問題となるのは，h と g が持つ x 座標の区間に共通部分があり，g 上で h から付けられるラベル値のほうが小さくなる区間が存在する場合のみである．このときには，共通部分に関する両方のコスト情報を見比べて，図 8 のように，新たに h からの情報を持った上向きに進む分割線 g' をヒープに挿入する．

3.4 データ構造と計算複雑度

本節では，3.2 節，3.3 節のアルゴリズムを効率よく実行するためのデータ構造を示し，アルゴリズムの時間および空間複雑度を考察する．以下では，配線領域を図形としてとらえた際の頂点数を n，径路探索において訪れた分割線の数を k とする ($k<n$).

コスト情報の伝達の際に，ラベル値が一定のコストからなるものを伝達することは明らかに $O(1)$ の手間で実行できる．問題となるのは図 6(c), (f)のように 2 種類のコストからなる情報を伝達する場合である．もし単純にこのようなラベル値を次の分割線のラベルとして複写すると，最悪の場合には，1 回の情報伝達に $O(k)$ の手間を必要とし，全体の径路探索としては $O(k^2)$ の手間（径路探索が配線領域全体におよぶ場合には $O(n^2)$ の手間となる）を要してしまう．そこで，このような場合には，分割線上で連続したコストの区間（この区間には自分のコストを持たせておく）の集合をあるリストにまとめて保持しておき，分割線はそのリストへのポインタだけを持つようにする．すると，図 6(c)のコスト情報の伝達を $O(1)$ の手間で行うことができる．同図(f)のようなコスト情報の伝達を行うためには，リストを任意の x 座標で分離することになる．この操作を効率良く行うためにリストには 2-3 木[17]を用いることにする．2-3 木によって分割線上で連続したコストの区間を保持していれば，図(f)の操作は $O(\log k)$ の手間で実行可能である．

ラベル付けが競合する際に，一定のコストからなるラベルどうし，もしくは，一定のコストからなるラベルと 2 種類のコストからなるラベルの競合は，明らかに $O(1)$ の手間で実行できる．問題となるのは図 7，図 8 のような場合である．図 7 において必要となる操作は二つのコスト情報の合併である．これは前述のようにコスト情報を 2-3 木で保持しているので，$O(\log k)$ の手間で実行することができる．次に，図 8 のように探索方向が異なる 2 種類のコストを持った分割線の競合について考察する．仮ラベルの分割線上において，一連のコストを持つ区間を**コスト区間**，隣接するコスト区間の境界を**接点**と呼ぶことにする．たとえば，図 6(c)の分割線 h 上には四つのコスト区間，三つの接点がある．分割線上に接点が存在するとき，その接点と同じ x 座標を持った凹な頂点が常に存在する．なぜならば，この接点は径路探索時に図 6(e), (f)のように凹な頂点を通る垂直な直線上に生ずるからである．探索時に一つの凹な頂点を通過したときに高々一つのコスト区間が増加すること，ならびに，分割線は探索されると永久ラベルになることを考慮すると，ある凹な頂点に対応する接点はその凹な頂点を通る垂直な直線と交差する分割線のうち高々一つの仮ラベルの分割線上にしか存在しない．したがって，接点の数は探索した凹な頂点数，つまり，探索した分割線の数で抑えられる．2 種類のコストを持つ分割線がヒープから取り出されると，探索順序としては，まずこの 2 種類のコストを持つ分割線がすべて探索され永久ラベルになるまで行われる．このとき，図 8 のように探索方向が異なる分割線の競合が起こると，最悪の場合には，分割線のコスト区間をすべて調べる必要があり $O(k')$ の手間を費やす．ここで，k' はコスト区間の数である．したがって上述の理由により探索した分割線の数に対応する．しかし，このようにしてもここで探索された分割線は二度と探索されないので，この競合に関する全体の手間は高々 $O(k)$ で抑えられる．

上記のようにコスト情報の伝達の 1 回当りの手間は高々 $O(\log k)$ であることがわかる．そして，伝達の回数は一つの線分が上方に高々 1 回，下方に高々 1 回の探索を行うので，全体としては $O(k)$ 回である．したがって，径路探索全体の時間複雑度は $O(k \log k)$ となる．最後に，空間複雑度について考察する．分割線上のコスト情報を正確に保持しているのは仮ラベルのときだけであり，前述のように仮ラベルの分割線上

のコスト区間の数は探索した分割線の数で抑えられるので，仮ラベルの情報量は総和は常に $O(k)$ となる．また，それ以外の分割線は $O(1)$ の情報しか保持していないので，提案した径路探索に必要な記憶容量は明らかに $O(k)$ である．

4. 図形データの更新

径路決定後，その径路が通るタイルに対して図形データの更新がなされる．この操作は，次の径路探索において，この径路が禁止領域として扱われるようにするために，長方形分割を局所的に行うものである．これは文献 18)のアルゴリズムを適用することにより $O(k)$ の手間で実行可能である．また，径路探索時についたラベルはつぎの探索のときには不必要なので空の状態に戻す．この操作も $O(k)$ の手間で行えることは明らかであろう．したがって，図形データの更新に必要な手間は $O(k)$ となる．

5. 計算機実験結果

最小曲がり径路を求めるタイル探索法をC言語を用いてインプリメントし，計算機実験を行った．使用した計算機は VAX11/785，OS は UNIX 4.2 BSD である．配線禁止領域，配線要求等は乱数によって発生させた．

まず，禁止領域の頂点数 n の増加に対する各操作の実行時間の変化を図 9 に示す．ここで，前処理以外の実行時間は 10 本の配線を行ったときの平均値である．図 9 を見ると，前処理：preprocessing（タイル平面の作成），配線準備：preparation for wire routing（ラベルのクリアと始点・終点を通る分割線の生成），ラベル付け：labeling（径路探索）に関する直線の傾きはほぼ 1 である．理論的考察では，前処理に要する手間は $O(n \log n)$ であったので，ほぼ理論通りの結果となっている．配線準備とラベル付けの計算複雑度はそれぞれ $O(k)$，$O(k \log k)$ であったが，本実験では，配線径路が配線領域全体に広がるように端子を指定しているので，k は n とほぼ同程度となっている．そのために，直線は前処理と同じような傾きになったものと考えられる．

逆追跡：backtracking と領域更新：updating は直線の傾きがそれぞれ 0.3，0.5 と，n に対してはかなり小さい増加率を示す．これは，この二つの操作が配線領域全体にわたらず，配線径路の周囲のみで行われているためである．

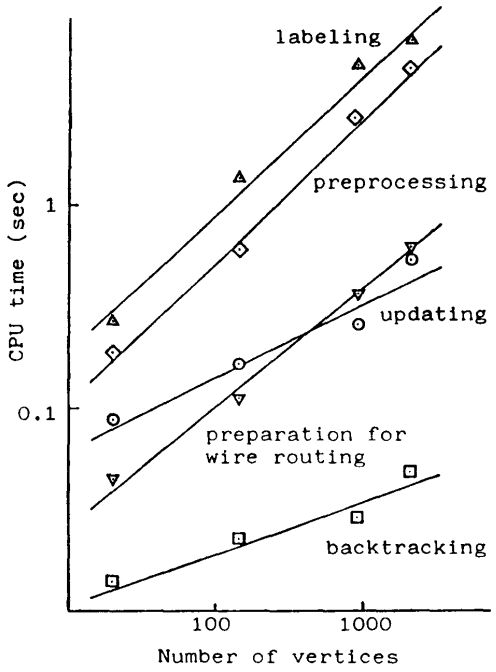

図 9 配線領域の頂点数とタイル探索法の各処理の実行時間
Fig. 9 Performance data of the tile-search method.

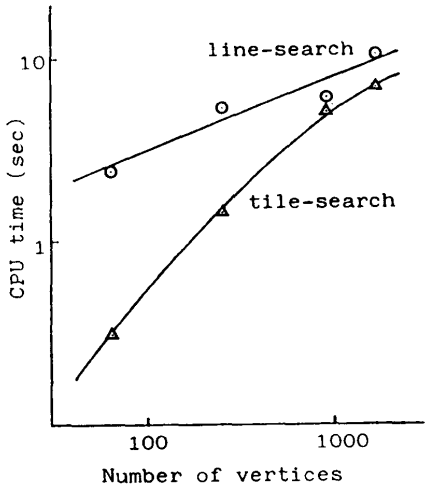

図 10 配線領域の頂点数によるタイル探索法と線分探索法の比較
Fig. 10 Comparison between the tile-search method and the line-search method concerning the number of vertices of routing region.

次に，従来の線分探索法[1]と本手法との実行速度の比較結果を報告する．禁止領域の頂点数に対する二つの手法の径路探索時間（10 本の配線を行ったときの平均値）を図 10 に示す．頂点数が少ない内は，タイル探索法のほうがはるかに速く径路を求め，頂点数の

増加にともなって両者の差が小さくなっている．これは，頂点数の増加につれて，配線領域が狭くなり，そのために線分探索法が発生するエスケープ・ラインの数の増加率が減少するためである．しかし，配線領域が狭くなると，タイル探索法におけるラベル付けの競合がかえって起こりにくくなってくるので，この手法による実行時間の増加率も減少する．このために，線分探索法のほうがタイル探索法よりも速く径路を求めるということは起こりにくい．

得られた径路を曲がり数によって分類し，その径路を求めるために要した時間を平均した結果を**図 11** に示す．線分探索法では，曲がり数が2までの径路は非常に高速に求まるが，3回曲がり以上の径路となると，エスケープ・ラインが大量に発生し，急激に所要時間が増大することがわかる．これに対して，タイル探索法では，すでに存在する分割線にコスト情報を伝達するだけで，新たに分割線を発生することはないので，ほぼ曲がり数に比例した時間で径路を求めている．

以上の結果と考察より，タイル探索法は線分探索法よりも短時間に径路を求める有効な配線手法であることがわかる．なお，詳細なインプリメント方法，実験データについては文献 19)を参照していただきたい．

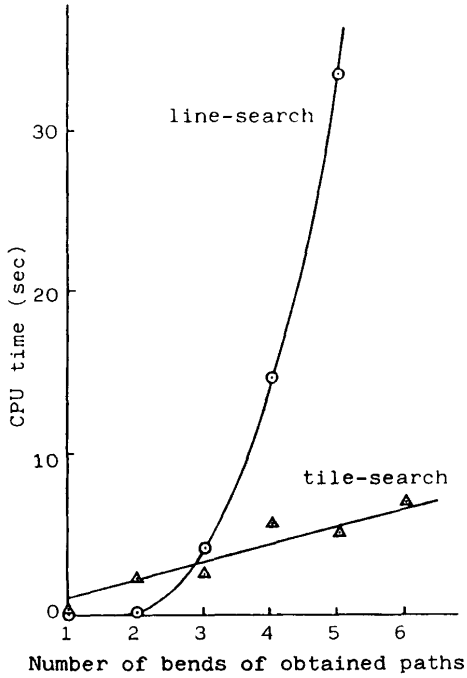

図 11 径路の曲がり数によるタイル探索法と線分探索法の比較

Fig. 11 Comparison between the tile-search method and the line-search method concerning the number of bends of obtained paths.

6. むすび

本論文では一層配線問題に焦点をあて，$O(n \log n)$ の手間の前処理を1回行えば，$O(n)$ の記憶容量の下で，1ネット当り $O(k)$ の手間で径路を，$O(k \log k)$ の手間で最小曲がり径路を求め，図形情報の更新も行う径路探索手法を提案した．ここで，n は配線領域を図形としてとらえた際の頂点数，k は径路探索時に訪れた頂点の数である．また，計算機実験を通じてその有効性を確認した．提案した手法は最小曲がり径路の保証はなくなるが多点間ネットを処理するように拡張できることは明らかであろう．

類似した配線問題として，最短径路を求める問題がある．現在のところ $O(n)$ の記憶容量を用いて $O(n \log^2 n)$ の手間 ($O(n \log n)$ の記憶容量では $O(n \log n)$ の手間) で径路探索を行うアルゴリズム[7]が最良のものであるが，複雑なデータ構造に基づいている．そこで今後は，単純なデータ構造に基づくと同時に，逐次配線を行うことを考慮した効率の良い配線手法の構築が望まれる．

参考文献

1) Mikami, K. and Tabuchi, K.: A Computer Program for Optimal Routing of Printed Circuit Conductors, *IFIP Congress 68*, pp. 1475-1478 (1968).
2) Hightower, D. W.: A Solution to Line-Routing Problem on the Continuous Plane, *Proc. DA Workshop*, pp. 1-24 (1969).
3) Lee, D. T. and Preparata, F. P.: Computational Geometry—A Survey, *IEEE Trans. Comput.*, Vol. C-33, No. 12, pp. 1072-1101 (1984).
4) Lipski, W. and Preparata, F. P.: Segments, Rectangles, Contours, *J. Algorithms*, Vol. 2, pp. 63-76 (1981).
5) Lipski, W.: Finding a Manhattan Path and Related Problems, *Networks*, Vol, 13, pp. 399-409 (1983).
6) Lipski, W.: An $O(n \log n)$ Manhattan Path Algorithm, *Inf. Process. Lett.*, Vol. 19, No. 2, pp. 99-102 (1984).
7) 小島郁太郎，佐藤政生，大附辰夫：多角形領域上の最短径路アルゴリズム，信学技報，CAS 83-205, pp. 45-50 (1984).
8) 佐藤政生，大附辰夫：グリッドレス・ルーター格子を用いない二層配線径路探索手法，信学論 (D), Vol. J 69-D, No. 5, pp. 802-809 (1986).
9) Wu, Y.-F., Widmayer, P., Schlag, M. D. F.

and Wong, C. K.: Rectilinear Shortest Path and Minimum Spanning Trees in the Presence of Rectilinear Obstacles, *IEEE Trans. Comput.*, Vol. C-36, No. 3, pp. 321-331 (1987).
10) Clarkson, K. L., Kapoor, S. and Vaidya, P. M.: Rectilinear Shortest Paths through Polygonal Obstacles in $O(n(\log n)^2)$ time, *Proc. 3rd Annual Symp. on Computational Geometry*, pp. 251-257 (1987).
11) Margarino, A., Romano, A., De Gloria, A., Curatelli, F. and Antognetti, P.: A Tile-Expansion Router, *IEEE Trans. Comput.-Aided Des. Integrated Circuits & Syst.*, Vol. CAD-6, No. 4, pp. 507-517 (1987).
12) 小島直仁, 鈴木 敬, 佐藤政生, 大附辰夫: ヒープ探索木を用いた改良線分探索法, 信学技報, VLD 88-9, pp. 65-72 (1988).
13) Imai, H. and Asano, T.: Dynamic Segment Intersection Search with Applications, *Proc. 25th FOCS*, pp. 393-401 (1984).
14) Ousterhout, J. K.: Corner Stitching: A Data-Structuring Technique for VLSI Layout Tools, *IEEE Trans. Comput.-Aided Des. Integrated Circuits & Syst.*, Vol. CAD-3, No. 1, pp. 87-100 (1984).
15) 佐藤政生, 橘 昌良, 大附辰夫: 図形整形アルゴリズムとその LSI パターン設計への応用, 信学論 (C), Vol. J 66-C, No. 12, pp. 1132-1139 (1983).
16) 大附辰夫, 佐藤政生, 橘 昌良, 鳥居司郎: 複合長方形領域の最小分割, 情報処理学会論文誌, Vol. 24, No. 5, pp. 647-653 (1983).
17) Aho, A. V., Hopcroft, J. E. and Ullman, J. D.: *The Design and Analysis of Computer Algorithms*, Addison-Wesley, Reading, Mass. (1974).
18) Shand, M. A.: Algorithms for Corner Stitched Data-Structures, *Algorithmica*, Vol. 2, pp. 61-80 (1987).
19) 坂中二郎, 小島直仁, 佐藤政生, 大附辰夫: 配線問題におけるタイル探索法の評価, 信学技報, VLD 87-115, pp. 61-68 (1987).

(昭和 63 年 4 月 18 日受付)
(昭和 63 年 12 月 12 日採録)

佐藤 政生 (正会員)

昭和 56 年早稲田大学理工学部電子通信学科卒業. 昭和 61 年同大学院博士課程修了. 昭和 59 年早稲田大学情報科学研究教育センター助手, 昭和 61 年カリフォルニア大学バークレー校研究員を経て, 現在, 拓殖大学工学部情報工学科助教授. 工学博士. 電子回路設計の自動化, ノイズ解析, 計算幾何学, グラフ理論とその応用などに関する研究に従事. 昭和 62 年度丹羽記念賞受賞. 電子情報通信学会, プリント回路学会, 日本 OR 学会, IEEE, ACM 各会員.

坂中 二郎

昭和 61 年早稲田大学理工学部電子通信学科卒業. 昭和 63 年同大学院修士課程修了. 同年, 日本アイ・ビー・エム入社. LSI 設計の自動化, 計算幾何学等の研究に従事. 現在, 同社大和研究所勤務. 電子情報通信学会会員.

大附 辰夫 (正会員)

昭和 38 年早稲田大学理工学部電子通信学科卒業. 昭和 40 年同大学院修士課程修了. 同年日本電気(株)入社. 昭和 55 年早稲田大学理工学部電子通信学科教授, 現在に至る. 工学博士. 電子回路の CAD およびこれに関連した基礎研究に従事. 昭和 44 年度電子通信学会論文賞, 1974 年 IEEE CAS Society より Cuillmin-Cauer 賞受賞. 共著「VLSI の設計 I」(岩波書店), 編共著「Layout Design and Verification」(North-Holland). 電子情報通信学会, 電気学会, プリント回路学会各会員. IEEE Fellow.