設計自動化 44－9
（1988. 10. 28）

グリッドレス・ルータの連想メモリプロセッサへの実装

久保田和人・石和 信政・鈴木 敬・大附 辰夫

早稲田大学 理工学部

レイアウト設計の配線処理では、格子に基づかない配線手法「グリッドレス・ルータ」が提案されている。この手法は配線領域を図形として扱い、図形処理により径路を求める。従来ソフトウェアでは計算複雑度を抑えるために複雑なデータ構造を必要としていた。我々は連想メモリを用いて図形処理を高速に実行するハードウェア・エンジン（CHARGE）を提案し、試作を行ってきた。本稿ではグリッドレス・ルータの一手法である「改良線分探索法」をこのハードウェア上に実装し汎用計算機上でのソフトウェアとの比較を行った結果を示す。

An Implementation of a Gridless Router on CAM Based Hardware Engine

Kazuto Kubota, Nobumasa Ishiwa, Kei Suzuki
and Tatsuo Ohtsuki
School of Science and Engineering, Waseda University
3-4-1, Ohkubo, Shinjuku, Tokyo, 160 Japan

Several gridless routing algorithms, in which the area to be used for wiring is represented as a polygonal pattern, have been presented. These algorithms need complicated data structures to reduce computational complexity. We have proposed a CAM based hardware engine which can solve some basic geometrical search problems in constant time. In this paper, we present a hardware gridless routing algorithm and demonstrate its performance data measured on a prototype machine.

1．はじめに

ＶＬＳＩやＰＣＢの配線処理に格子を用いない配線手法「グリッドレス･ルータ（gridless router）」が提案されている[2-12]。格子構造を用いる配線手法として代表的な迷路法[1]は、径路が存在すれば必ず最短径路を発見できるという利点がある反面、扱う配線領域の面積に比例したメモリを必要とし、大きい配線領域では多大な処理時間を必要とする。また、複雑な設計規則に対応しづらいという欠点をもつ。これに対しグリッドレス・ルータは、配線領域を図形として扱い、図形操作により径路を発見する手法である。グリッドレス・ルータは次のような特徴を持つ。

① 図形データを直接扱うので、複雑な設計規則への対応が容易である。
② 迷路法のように格子を用いないので使用メモリが少ない。
③ 径路が存在すれば必ずこれを発見できる。
④ 配線結果のマスクパターンへの変換が容易である。

グリッドレス・ルータは、迷路法に比べれば少ないメモリで高速に処理を行うことができる[11]が、計算複雑度を抑えるために部分問題毎に異なった複雑なデータ構造を必要とし、プログラムの規模は大きくなる。

我々は、グリッドレス・ルータのアルゴリズムを提案するとともに、より高速で複雑なデータ構造を必要としない専用ハードウェアの研究も進めてきた[9][16-19]。このハードウェアは連想メモリ（Content adressable memory 以下ＣＡＭと略す)[14-15]を用いるもので､グリッドレス・ルータに限らず一般的な図形処理問題を解くことができる。ＣＡＭを用いることによりソフトウェアでは、データ数ｎに対して$O(\log n)$や$O(\log^2 n)$を必要とする問題をデータ数によらず一定時間で解くことができる。さらに、ソフトウェアでは問題毎にデータ構造を用意する必要があったがＣＡＭを用いると同一フォーマットのデータに対して、種々の問題を解くことができる。我々は、このハードウェアをＣＨＡＲＧＥ（ CAM based hardware engine for geometical problems ）と名付け実際に試作し、基本的な図形処理問題に関して評価を行ってきた[18]。本稿では「改良線分探索法」[12]をグリッドレス・ルータの一例として取り上げ、このアルゴリズムをＣＨＡＲＧＥ上に実装し評価を行った結果を示す。

2．連想メモリの機能とＣＨＡＲＧＥ

2．1　連想メモリの機能

ＣＡＭは、機能メモリの一種であり、ＲＡＭがアドレスによりデータをアクセスするのに対し、ＣＡＭでは、データの一部分が一致するデータを直接アクセスすることができる（図１）。この動作を一致検索と呼ぶ。

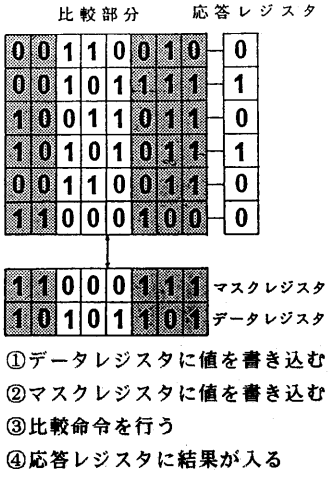

図１　連想メモリの動作

一回の一致検索に要する時間は、記憶アレイ中のデータ数によらず一定である。さらに、一致検索機能をもとにして、記憶アレイ中のデータの最大値、最小値、あるいはある値より大きいデータ（図２）、ある値より小さいデータを検索することができる。これらの操作を関係検索と呼ぶ。この関係検索もデータ数によらず一定時間で行うことができる。

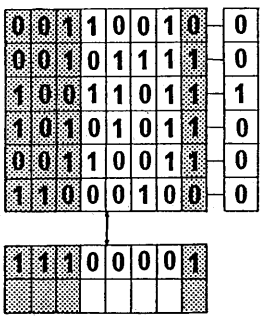

図２　関係検索（最大値）

また検索された複数のデータの特定のビットに対して、同時に書き込みを行える機能も有している。この機能を用いると、ＣＡＭの各ワードに対して並列に加・減算処理を行うことが可能であり、連想メモリを一次元のＳＩＭＤ型プロセッサとして利用することができる。

2．2　ＣＨＡＲＧＥの構成

ＣＨＡＲＧＥではＣＡＭチップに、ＮＴＴで開発されたチップ[14]［4k bit/chip:32bit/word × 128 word ］を用いている。このチップは最高１４０[ns]周期のクロックで実行させることが可能である。このような高速で

動作させるためには、直接ホストコンピュータがＣＡＭに命令を送る方式では効率が悪いため、専用のコマンドシーケンサを設ける必要がある。

図３にＣＨＡＲＧＥのブロック図を示す。本システムは大きく分けて、シーケンサ部、ＡＬＵ部、ＣＡＭ部、ＲＡＭ部の４つのモジュールから成り立っている。各部の制御はシーケンサにより駆動されるマイクロコードにより行われる。このマイクロコードの開発のため専用のアセンブラおよびハードウェアエミュレータを開発し、開発環境を整備した[19]。

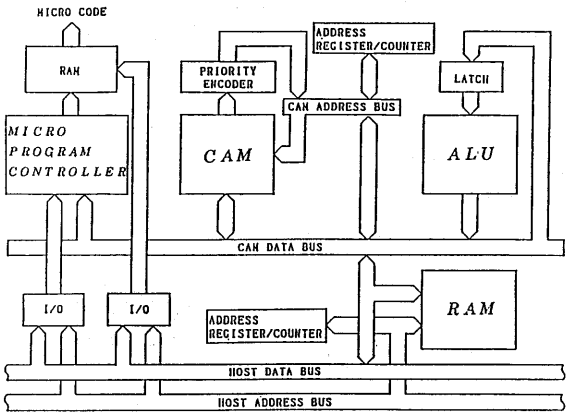

図３　ＣＨＡＲＧＥのハードウェア構成

３．連想メモリによる図形処理

ＣＡＭを用いることによりいろいろな図形処理問題を解くことができるが、ここではグリッドレス・ルータで扱われているいくつかの基本的な問題に関してＣＡＭを用いた解法を示す。

（１）線分と交差する線分の探索問題

ｎ本の水平線分の集合Ｈに対し、質問線分Ｖが与えられたとき、Ｖと交差するＨの要素を列挙する問題（図４）。

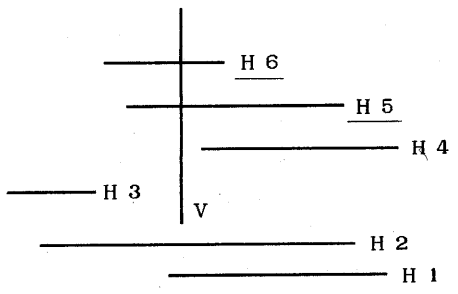

図４　線分と交差する線分の探索問題

（２）線分に隣接するの線分の探索問題

ｎ本の水平線分の集合Ｈに対し、質問線分Ｒが与えられたとき、Ｒから上（下）方向に長方形領域を広げ、この領域内にあるＨの要素のうち最も下（上）にある線分を見つける問題（図５）。

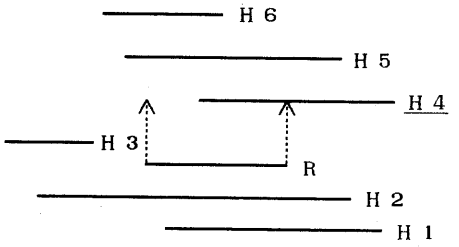

図５　線分に隣接する線分の探索問題

（３）長方形と交差する線分の探索問題

ｎ本の水平線分の集合Ｈに対し、質問長方形Ｒが与えられたとき、Ｒと交差する（または含まれる）Ｈの要素を列挙する問題（図６）。

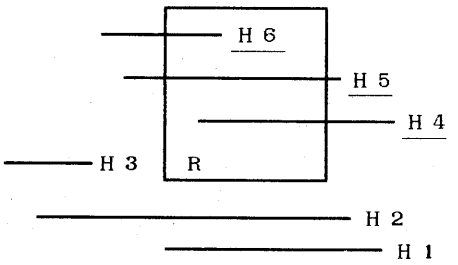

図６　長方形と交差する線分の探索問題

ここでは、上記の問題のうち、（２）の線分に隣接する線分の探索問題が、ＣＡＭを用いることにより、線分数ｎに依らず、一定時間で解けることを示す。なお、他の（１）（３）の問題や、被質問線分を点に置き換えた問題も同様な操作を行うことにより、一定時間で解くことができる。（２）の問題において、被質問線分は図７に示すようなフォーマットでＣＡＭ内に格納されている。質問水平線分を（XR_L，XR_R，YR）（図５ではＲ）とすると、

ｓｔｅｐ１：　被質問水平線分（X_L，X_R，Ｙ）に対し、$X_L \leqq XR_R$を満たす線分を選ぶ。

ｓｔｅｐ２：　ｓｔｅｐ１を満たし、かつ$X_R \geqq XR_L$を満たす線分を選ぶ。

ｓｔｅｐ３：　ｓｔｅｐ２を満たし、かつＹ＞YRを満たす線分を選ぶ。

ｓｔｅｐ４：　ｓｔｅｐ３を満たす線分の中で最小のＹ座標を持つものを選ぶ。

以上のように４回の関係検索を行うことによって、解を求めることができる。したがって、（３）の問題はＣＡＭを用いることにより、時間複雑度Ｏ（１）、空間複雑度Ｏ（ｎ）で解くことができる。

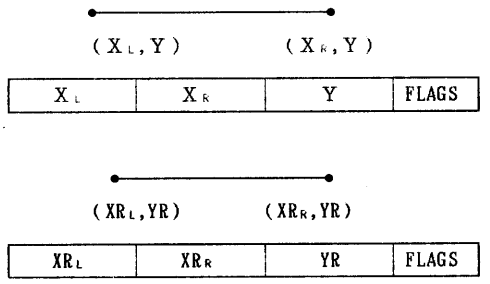

図７　線分探索のためのデータ

４．改良線分探索法の実装

４．１　ＣＡＭを用いた改良線分探索法のアルゴリズム

ＣＨＡＲＧＥ上に実装を行った改良線分探索法は、大きく分けて３つの処理から成る。前処理では与えられた禁止領域、既配線の情報から、配線が通過可能な水平・垂直線分（エスケープライン　以下Ｅ．Ｌ．と略す）を発生させる。径路探索では、径路を発見し、図形更新では発見された径路を禁止領域として、禁止領域及び配線領域の更新を行う。

（１）前処理

①　禁止領域・既配線・端子（以下この３つを障害物と呼ぶ）のデータを図８の形式でＣＡＭに入力する。

XL	XR	Y	FLAGS	00

横線分のデータ形式

X	YD	YU	FLAGS	01

縦線分のデータ形式

X	Y	FLAGS	10

点のデータ形式

図８　ＣＡＭ内での図形のデータ形式

②　ＣＡＭから障害物のデータを逐次取り出し、その周囲にコーナーポイント（以下Ｃ．Ｐ．と略す）の候補を発生させる。Ｃ．Ｐ．の候補のデータもＣＡＭに入力する。コーナーポイントは、配線幅と配線間隔を考慮して配線領域を縮小してできる図形の凹点に相当する。

③　各Ｃ．Ｐ．の候補について、その点を中心とする一辺の長さが２Ｄ（Ｄは配線幅と配線間隔を考慮した長さ）の正方形領域Ｒに関して「長方形と交差する線分の探索問題　３-（２）」を解く。もし障害物を表す線分が探索されなければその点をＣ．Ｐ．とする。探索されたならばＣ．Ｐ．とせず削除する。

④　確定したＣ．Ｐ．を中点とする長さ２Ｄの水平・垂直線分H_{CP}，V_{CP}を考える。V_{CP}関して「線分に隣接する線分の探索問題　３-（１）」を解き、最も近い障害物を求める。そこからＤだけ離れた位置を端点とする水平Ｅ．Ｌ．を作成する（図９）。H_{CP}に関しても同様な処理を行い垂直Ｅ．Ｌ．を作成する。これらのＥ．Ｌ．にはラベル値“０”を付加し逐次ＣＡＭに格納する。ラベル値は０～３まであり、０は未探索で探索されたものは１～３の値を持つ。

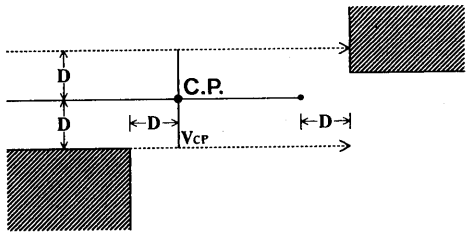

図９　Ｅ．Ｌ．の作成

（２）径路探索

与えられた端子対Ｓ（source）とＴ（target）間の最小曲がり径路を、以下の処理により求める。

①　他の配線を行う際には障害物として扱った端子の周囲のＣ．Ｐ．およびそのＣ．Ｐ．から延びるＥ．Ｌ．をＣＡＭから削除する。

②　端子Ｔから水平・垂直のＥ．Ｌ．（target line と呼ぶ）を発生させ、これをＣＡＭに格納する。さらに、端子Ｓからも水平・垂直のＥ．Ｌ．を発生させ、これらに“１”というラベルを付けＣＡＭに格納する。

③　探索されるＥ．Ｌ．がある限り、以下の処理を繰り返す。ｉの初期値は１である。ラベルｉのＥ．Ｌ．を順次取り出し、この線分に関して「線分と交差する線分の探索問題　３-（１）」を解く。探索された線分の中に、ラベル値が“０”のＥ．Ｌ．があれば、そのＥ．Ｌ．のラベル値をｉ＋１（ｉが３の時は１）とする。この中にtarget line があれば逆追跡（⑤以下の処理）を行い、さらに（３）の図形更新を行う。

④　もし、あるラベル値ｉについて交差するＥ．Ｌ．が存

在しない場合には径路は存在しない。一つでも交差するＥ.Ｌ.があればラベル値ｉ＋１について③を繰り返す。

⑤　２つの作業領域ＳＥＧとＰＯＩＮＴを用意する。初期値としてＳＥＧに、ラベル値の付いている target line を表す線分データ、ＰＯＩＮＴに端子Ｔ(target)を表す点のデータを入れる。

⑥　ＳＥＧ（ラベル値ｉ）と交差し、かつ、ラベル値がｉ－１（ｉが１の時は３）であるＥ.Ｌ.を「線分と交差する線分の探索問題 3-(2)」を用いて求める。この中でＰＯＩＮＴに最も近いＥ.Ｌ.を求め、このＥ.Ｌ.とＳＥＧとの交点とＰＯＩＮＴ間に新規配線を作る（図１０）。このＥ.Ｌ.をＳＥＧに格納し、交点をＰＯＩＮＴに格納する。

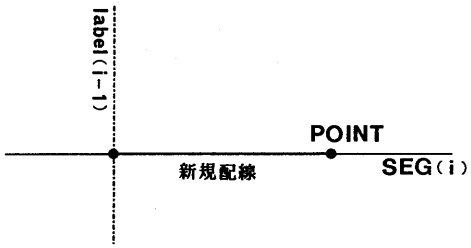

図１０　新規配線の作成

⑦　⑥の操作をＳＥＧと交差し、ラベルがｉ－１であるＥ.Ｌ.が存在しなくなるまで続ける。存在しなくなった場合は、ＰＯＩＮＴと端子Ｓ（source）間に新規配線を作る。

（３）図形更新

①　発見された径路を構成する線分について、これらの線分から配線幅と配線間隔を考慮した間隔Ｄ以内の領域Ｒと交差するＣ.Ｐ.及びＥ.Ｌ.を「長方形領域と交差する線分の探索問題 3-(3)」を用いて探索し、これらをＣＡＭから削除する。このとき、この領域外で、削除されるＥ.Ｌ.上にあるＣ.Ｐ.はＣ.Ｐ.の候補とする。

②　Ｃ.Ｐ.の候補からＥ.Ｌ.を発生させる。

③　（１）の前処理と同様な操作で発見された径路の周りに、Ｃ.Ｐ.およびＥ.Ｌ.を発生させる。

以上、前処理によって配線領域（図１１）にＣ.Ｐ.とＥ.Ｌ.が作成され（図１２）、径路探索によって最小曲がり径路が見つかり（図１３）、図形更新によって発見された径路の周りの図形が更新される（図１４）。

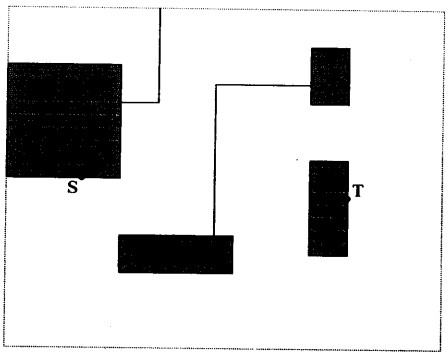

図１１　配線領域

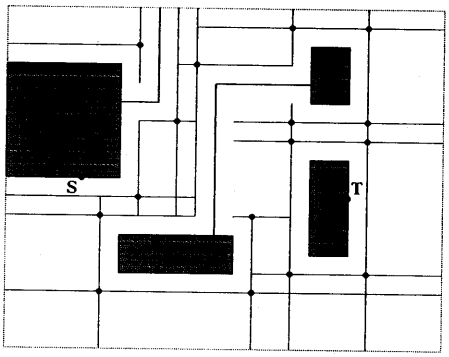

図１２　Ｃ.Ｐ.とＥ.Ｌ.の発生

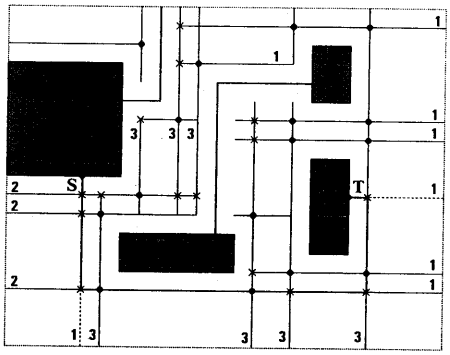

図１３　径路の探索

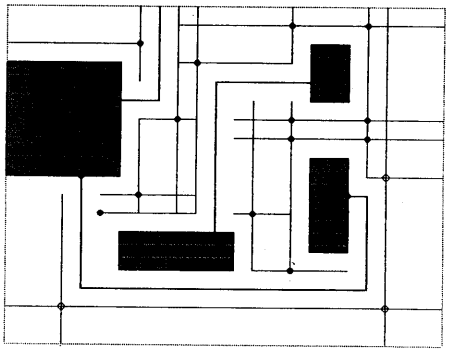

図１４　更新された配線領域

4．2　計算複雑度の評価

次に前節で述べた改良線分探索法について、その計算複雑度の評価を行う。

（１）前処理

ある障害物からは、定数時間でＣ.Ｐ.の候補を作ることができるので、ｎ個の障害物からはＯ（ｎ）の時間でＣ.Ｐ.の候補を作成できる。Ｃ.Ｐ.の候補はその周囲の障害物の有無を調べることにより、Ｃ.Ｐ.となるか削除される。障害物の有無は、「長方形領域と交差する線分の探索問題 3-(3)」を解くことによりわかる。この問題は第３章で示したように、一つの質問長方形について一定時間で解くことができる。Ｃ.Ｐ.はＯ（ｎ）個存在するので、この計算時間はＯ（ｎ）で抑えられる。１つのＣ.Ｐ.から延びるＥ.Ｌ.の端点の座標を決定するには、「線分に隣接する線分の探索問題 3-(2)」を４回解けばよく、この処理もまた一つのＣ.Ｐ.に関して一定時間で行うことができる。従って、前処理全体の計算複雑度は、Ｏ（ｎ）となる。

（２）径路探索

径路発見のために行うラベル付けでは「線分と交差する線分の探索問題 3-(1)」を、探索されたＥ.Ｌ.について順次行うことになる。この処理の手間は、探索されたＥ.Ｌ.の本数ｓに比例する。このアルゴリズムでは、一度質問線分となったＥ.Ｌ.は二度と探索の対象とならないので、最悪の場合でも処理の手間は、Ｅ.Ｌ.の本数Ｏ（ｎ）を超えることはない。逆追跡の処理は、見つかった径路の曲がり数をｂとするとＯ（ｂ）本の線分に対して「線分と交差する線分の探索問題 3-(1)」解くことになるので、処理時間はＯ（ｂ）となる。よって径路探索全体としての計算複雑度はＯ（ｓ）となる。（ただし、ｎ≧ｓ≧ｂ）。

（３）図形更新

更新されるＥ.Ｌ.と削除されるＣ.Ｐ.は、Ｏ（ｂ）本の線分（発見した径路）について、「長方形領域と交差する線分の探索問題 3-(3)」を解くことにより求められる。発見されたＣ.Ｐ.の削除は１命令で行うことができる。更新の対象となるＥ.Ｌ.（これをｕ本とする）は、１本あたり一定時間で更新できるので、処理時間はＯ（ｕ）となる。また、Ｏ（ｂ）本の新規の配線から、Ｃ.Ｐ.とＥ.Ｌ.を作成する処理は前処理と同様にＯ(b)である。よって図形更新の計算複雑度はＯ（ｕ）となる（ただし、ｎ≧ｓ≧ｕ≧ｂ）。

以上（１）（２）（３）より改良線分探索法はＣＡＭを用いることにより時間複雑度がＯ（ｎ）で実装できる。空間複雑度はｎ頂点の配線領域の図形データに対し、この図形データ、Ｃ.Ｐ.、Ｅ.Ｌ.の数はそれぞれＯ（ｎ）個に抑えられることからＯ（ｎ）である。ソフトウェアで点や線分を保持するためにヒープ探索木（priority search tree ）[13]を用いて改良線分探索法を実装した場合の時間複雑度とＣＡＭを用いた場合との比較を表１に示す。なおヒープ探索木を用いた場合、空間複雑度はＯ（ｎ）である。

表１　改良線分探索法の時間複雑度

	ＣＡＭ	ソフトウェア Priority Search Tree
前処理	Ｏ（ｎ）	Ｏ（ｎ$\log^2$ｎ）
径路探索	Ｏ（ｓ）	Ｏ（ｓ$\log^2$ｎ）
図形更新	Ｏ（ｕ）	Ｏ（ｕ$\log^2$ｎ）

ｎ…図形の頂点数
ｓ…径路探索で探索されたＥ.Ｌ.の数
ｕ…図形更新で探索されたＥ.Ｌ.の数
ｂ…径路の曲がり数
（ただし、ｎ≧ｓ≧ｕ≧ｂ）

5．実験

図形の頂点数ｎが５０と１００の配線データに対して、実際に配線実験を行いその結果を示す。また、同じ問題を汎用コンピュータ上のソフトウェアで解いた場合との比較を行う。配線データは、１２８×１２８の領域に一様乱数を用いて禁止領域を発生させたものである。図１５に、本実験で取り扱った配線データの一例を示す。

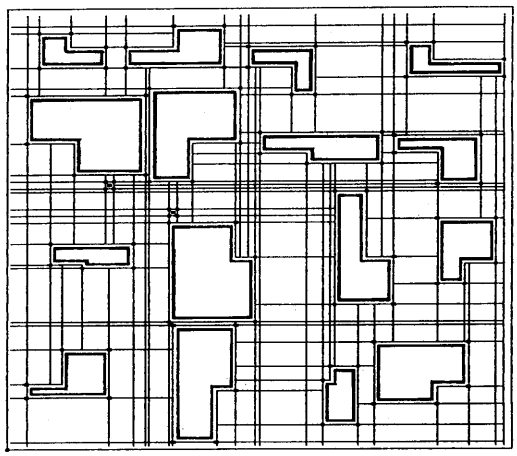

図１５　頂点数１００の例題

表２，３に各処理毎の処理時間を示す。なお、ソフトウェアはデータ構造Priority Search Treeを用いてＣ言語

でVAX11-785上に実装したものである。

表２　処理時間の比較（ｎ＝５０の場合）

	ＣＨＡＲＧＥ[s]	ソフトウェア[s]
前処理	1.7×10^{-2}	4.0×10^{-1}
径路探索	1.8×10^{-3}	0.8×10^{-1}
図形更新	0.7×10^{-2}	2.3×10^{-1}

表３　処理時間の比較（ｎ＝１００の場合）

	ＣＨＡＲＧＥ[s]	ソフトウェア[s]
前処理	3.3×10^{-2}	8.5×10^{-1}
径路探索	2.7×10^{-3}	0.6×10^{-1}
図形更新	1.2×10^{-2}	2.4×10^{-1}

前処理では、禁止領域を表す全ての線分について処理を行うので、データ数と処理時間はほぼ比例すると予想される。実際に、頂点数５０と１００のデータでは、処理時間は約２倍となっている。

径路探索と図形更新は、理論上の計算複雑度は最悪の場合Ｏ（ｎ）である。実験結果では、頂点数１００のときの処理時間は５０のときの約１．６倍程度である。これは、径路探索や図形更新の操作が、領域に存在するすべての線分を対象として処理を行うのではなく、新規配線の周りの局所的な部分についてのみ行われるためであると考えられる。理論的には、ＣＨＡＲＧＥの計算複雑度はソフトウェアよりも良く、部分問題に関しては実験結果を示した[18]。今回の実験ではＣＡＭの容量が足りず、大規模な例題を行えなかったが、大規模な問題ではさらにＣＡＭの効果が期待できる。

実際的な処理速度の面においては、ＣＨＡＲＧＥはソフトウェアと比較して約２０倍以上高速であることが実験結果からわかる。ソフトウェアを実装した VAX11-785 のサイクルタイムは 133[ns]であり、ＣＨＡＲＧＥのサイクルタイムは 200[ns]である。仮に、ＣＨＡＲＧＥが VAX11-785 と同じサイクルタイムで動作したとすれば、処理速度の差はさらに広がることになる。

以上より、連想メモリを用いた本システムのアーキテクチャが、図形処理に有効であり、図形処理を用いる配線手法を高速に実行することが示された。

６．おわりに

本稿では連想メモリプロセッサのアプリケーションとして「改良線分探索法」の実装を行い、処理の手間が改善されることを示した。また、配線実験の結果から、本システム上に実装された配線手法は、汎用コンピュータ上に実装されたソフトウェアと比較して、約２０倍以上高速であることがわかった。現在、ＣＨＡＲＧＥシステムでは、ＣＡＭの大容量化を検討中である。これが実現すれば、より大規模で実際的な配線問題に対する処理の高速化が期待できる。

<謝辞>

本研究は、財団法人大川情報通信基金:助成番号63-13（昭和６３年度）「計算幾何学アルゴリズムのハードウェアに関する研究」の助成のもとに行われたものである。

<参考文献>

[1] C.Y.Lee: “An Algorithm for Path Connection and its Apllications”, IRE. Trans EC-10, pp.346-365 (1961).

[2] W.Lipski.Jr: “Finding a Manhattan Path and Related Problems”,NETWORKS, Vol.13, pp.399-409 (1983).

[3] W.Lipski.Jr: “An O(n log n) MANHATTAN PATH ALGORITHMS” Inf. Process. Lett., Vol. 19, No2, pp.99-102(1984).

[4] 小島，佐藤，大附：“多角形領域上の最短径路アルゴリズム”，信学技報，CAS83-205,pp.45-50(1984)

[5] 佐藤，大附：“グリッドレス・ルーター格子を用いない二層径路探索手法－”,信学論,Vol.J-69-D,No5, pp.802-809,(May 1986).

[6] Wu.Ying Fung,P.Widmayer,M.D.F.Schlag and C.K. Wong: “Rectilinear shortest paths and minimum spanning trees in the presence of rectilinear obstacles”,IEEE Trans. on Comput. Vol.C-36,No.3 pp.321-331(1987)

[7] K.L.Clarkson,S.Kapoor and P.M.Vaidya: “Rectilinear Shortest Paths through Polygonal obstacles in $O(n(\log n)^2)$ time”,Proc.3rd Annual Symp. on Computional Geometry,pp.251-257(1987).

[8] A.Margarino, A.Romano, A.DeGloria, F.Curatelli and P.Antogetti:“A Tile Expansion Router”,IEEE Trans.on CAD,Vol.CAD-6,No4(1987).

[9] K.Suzuki, T.Ohtsuki and M.Sato: “A Gridless Router:Software and Hardware Implementation”, VLSI’87, pp.121-131(1987).

[10] 坂中，佐藤，大附：“配線問題におけるタイル探索法の評価”信学技法，VLD87-115,pp.61-68(1987).

[11] 佐藤，：“迷路法とグリッドレス・ルータの比較評価”，昭和６３年度信学春全大，A-269,pp.1-271 (1988)

[12] 小島，鈴木，佐藤，大附：“ヒープ探索木を用いた改良線分探索法”，信学技報，VLD88-9,pp.65-72 (1988).

[13] E.M.McCreight:“Priority Search Trees”,SIAM J. Comput., Vol14, No.2, pp.257-276(1985).

[14] 小倉，山田，丹野，石川：“４ｋｂ　ＣＭＯＳ連想メモリＬＳＩ”，信学技報，SSD82-78,pp.45-53 (1983)

[15] 小倉，山田，山田，：“２０ｋｂ　ＣＭＯＳ連想メモリＬＳＩ”,昭和６１年度 信学総全大447，pp.2-235(1986)
[16] 鈴木，橘，佐藤，“連想メモリによる図形処理問題の解法”，信学技報，CAS84-117,pp.13-20(1984)
[17] 鈴木，橘，佐藤，大附：“連想メモリを用いた配線処理算法”，信学技報，CAS84-193,pp.25-32(1985)
[18] 井出，石和，小島，鈴木，大附：“連想メモリを用いた図形処理装置の試作”，信学技報，CAS87-106(1987)
[19] 石和，井出，鈴木，大附：“連想メモリを用いた図形処理装置とソフトウェア開発環境”，昭和６３年度信学春全大，A-284,pp.1-286(1988)