連想プロセッサを用いた多層化改良線分展開法

5L－3

中村恵介　桑原泰雄　久保田和人　佐藤政生　大附辰夫
早稲田大学理工学部

1．まえがき

ＶＬＳＩのレイアウト設計において，迷路法のような格子構造を用いない配線手法（グリッドレスルータ）の一つに改良線分展開法[1]がある．我々は，連想メモリを用いたハードウェアエンジンで改良線分展開法を高速化する手法について報告を行ってきた[2]．近年，改良線分展開法を３層以上の多層配線ができるように拡張した手法[3]（以後，多層化線分展開法と呼ぶ）が提案されている．この手法は多層の配線問題を扱うため単層の場合と比べて処理が複雑であり，大きな処理時間を要している．本稿ではこの多層化線分展開法を，連想メモリを用いたハードウェアエンジンで高速化する手法およびその評価について述べる．

ここで用いるハードウェアエンジンは，新たに開発したＣＡＭチップを用いて製作したものであり，[2]で用いたものと比べてより高速で効率の良い図形処理が可能となっている．以下では，多層化線分展開法のアルゴリズムを示し，連想メモリとそれを用いたハードウェアエンジン，および連想メモリによる図形探索処理について述べる．さらに多層化線分展開法をハードウェアエンジン上に実装し，ソフトウェアとの処理速度の比較実験を行い，その評価を行う．

2．多層化線分展開法のアルゴリズム[3]

多層化線分展開法はダイクストラ法に基づいており領域を逐次探索（展開）することにより径路を求めるものである．展開する図形はＡＬ(Active line)と呼ばれる線分とＡＲ(Active rectangle)と呼ばれる長方形である．それぞれ生成時にコストがつけられ，同一のＰＱ(Priority queue)に格納される．このＰＱからコスト最小のＡＬまたはＡＲが取り出され展開が行われる．ＡＬが取り出された場合，ＡＬを同層に関してそれと垂直な方向に障害物に当たるまで掃引した後，得られた長方形領域にＡＲを発生させ，その周囲にＡＬを発生させる（ＡＬの展開：図１(a)）．ＡＲが取り出された場合，隣接層に関してそれと重なる領域内における障害物の探索を行い，ビアの中心がおける矩形領域を抽出した後，得られた領域を長方形分割してＡＲを発生させ，さらにその周囲にＡＬを発生させる（ＡＲの展開：図１(b)）．ＡＲ，ＡＬはそれぞれバックトレースができるように発生元のＡＲへのポインタを持たせておく．

与えられた２点（Ｓ，Ｔ）間を配線するにあたり，まずＳから最初のＡＬを発生させ，コストをつけてＰＱへ格納する（図１(c)）．ＰＱからコスト最小の要素を取り出し，順次展開する（図１(d)）．このときＡＬの展開において展開方向の障害物にＴが含まれていれば探索処理を終了し，

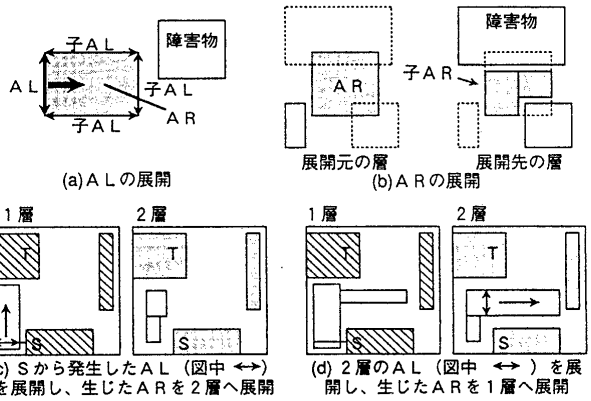

図１　多層化改良線分展開法の径路探索

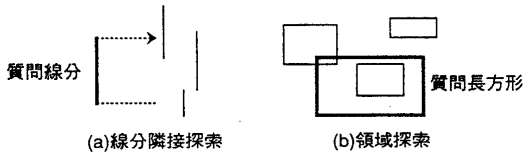

図２　基本的な2次元図形探索

バックトレースを行って径路を確定する．

本アルゴリズムでは，基本的な２次元図形探索処理として，質問線分に関して指定方向に隣接する線分を報告する線分隣接探索（図２(a)）と質問領域に関して交差するすべての図形を報告する領域探索（図２(b)）が用いられる．特に後者は単層の改良線分展開法では用いられなかった処理である．

3．連想メモリを用いたハードウェアエンジン

連想メモリは通常のＲＡＭの機能に加えて，外部から与えられたキーデータの指定部分が一致するワードをワード数によらない一定時間で検索する機能（一致検索）を持っている．各ワードは１ビットの応答レジスタを持ち，キーデータに対してマスクデータによりマスクされない部分が一致するすべてのワードの応答レジスタを１にする（図３）．一致検索を用いることにより，キーデータの指定部分に対して大きい（小さい）ワードを検索するしきい値検索や，与えられた指定部分が最も大きい（小さい）ワードを検索する極値検索もワード数によらない一定時間

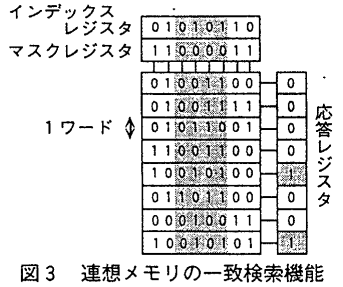

図３　連想メモリの一致検索機能

Multi-layer Improved Line-expansion Algorithm based on Content Addressable Memory
keisuke NAKAMURA, Yasuo KUWAHARA, Kazuto KUBOTA, Masao SATO, and Tatsuo OHTSUKI
School of Science and Engineering, Waseda University

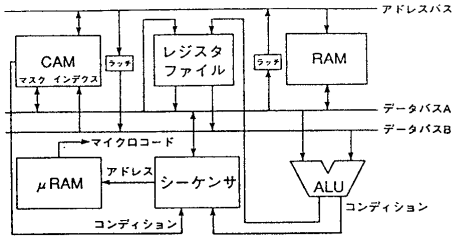

図4　連想メモリ（ＣＡＭ）を用いたハードウェアエンジン

で行うことができる．

新たに製作した図形処理専用ハードウェア[4]はＣＡＭ，ＡＬＵ，レジスタファイル，ＲＡＭ，シーケンサからなり，水平型のマイクロコードで駆動されるＶＬＩＷ型のアーキテクチャをとっている（図４）．本ハードウェアはホストコンピュータのバスに接続され，スレーブプロセッサとして利用される．

４．連想メモリによる図形処理

連想メモリにおいて一つの図形データは連続した４ワードに格納され，各ワードは４ワードを区別するためのタグ，図形を区別するための属性，層を区別する層番号，ワーク，および座標データの５フィールドに分けられる（図５）．

質問領域Ｒの座標を（qxl,qyb,qxr,qyt）とし、探索対象となる図形データを（xl,yb,xr,yt）とした場合，次の５ステップで領域探索を行うことができる．

(1) タグ0のワードを検索
(2) タグ0のワードに対しxl<qxrを検索
(3) (2)の条件を満たすタグ1のワードに対しqxl<xrを検索
(4) (3)の条件を満たすタグ2のワードに対しyb<qytを検索
(5) (4)の条件を満たすタグ3のワードに対しqyb<ytを検索

上記の操作はしきい値検索４回，一致検索４回を用いることにより探索図形数によらない一定時間で行うことができる．同様に，線分隣接探索の処理時間も探索図形数によらない．

５．計算機実験結果

多層化線分展開法は，Sparc Station2(28.5MIPS)上にＣ言語で実装されている．ＣＡＭを用いた図形処理専用ハードウェアで処理が高速化される部分は，図形探索処理の部分である．従って，改良線分展開法の図形処理部分に要する時間がどのくらい高速化されるか，また改良線分展開法の処理時間全体がどのくらい高速化されるかを計測した．

実験は100,000×100,000の３層からなる配線領域内に乱数によって長方形の障害物を生成し，同様に乱数により生

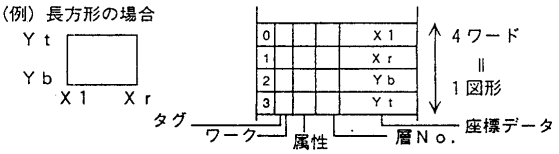

図５　連想メモリにおける図形のデータ構造

成された５組の端子対を配線するという処理を障害物の線分数をパラメータにして行った．ソフトウェアのデータ構造はバケット[5]を用いた．またＣＡＭには全ての図形が格納できるものとし，クロックは100nSとした．表１に示す実験結果では，５ネットの配線に要する全処理時間とその図形処理に要する時間，および図形処理のうち前述の２探索処理に要する時間を取り上げた．

基本的な図形探索処理１回にかかる時間は，ソフトウェアは探索線分数とともに増加しているのに対してＣＡＭは定数時間であり，線分隣接探索では16～48倍，領域探索では17～170倍の高速化が得られている．このＣＡＭによる高速化の効果は探索線分数が多いほど大きいことがわかる．また全図形探索処理に要する時間は23～88倍，改良線分展開法の処理時間全体では2.6～3.9倍の高速化が得られた．

６．むすび

本ハードウェアエンジンを用いた多層化改良線分展開法の高速化の効果が４倍程度にとどまったのは，図形処理以外のソフトウェアの部分がボトルネックになっていると考えられる．今後の課題として，図形処理以外でＣＡＭの検索機能が利用できる部分の高速化が挙げられる．

参考文献

[1]小島　他：“線分展開法の改良とその評価”，情処研報，設計自動化，Vol.48, No.6, pp.1-8(1989).

[2]M.Sato et al.: "A Hardware Implementation of Glidless Routing Based on Content Addressable Memory", Proc.27th DA Conf., pp.646-649 (1990).

[3]石川　他：“改良線分展開法の多層化”、信学技報、VLD91-85,pp.41-48(1991).

[4]桑原　他：“連想メモリを用いた図形処理用ハードウェアエンジン”，信学技報，CPSY92-17, pp.63-70(1992).

[5]T.Asano, M.Edahiro, et al.: "Practical Use of Bucketing Techniques in Computational Geometry", G.T.Toussaint, Ed., North-Holland(1985).

表１　実験結果

線分数	線分隣接探索					領域探索					全図形探索処理(sec)		全処理(sec)	
	探索	SPARC (msec)		CAM (msec)		探索	SPARC (msec)		CAM (msec)		SPARC	CAM	SPARC	SPARC
	回数	@	total	@	total	回数	@	total	@	total				+CAM
512	2187	0.37	820	0.018	39	1988	0.20	390	0.009	19	1.6	0.068	11.7	4.2
1024	3062	0.39	1190	0.018	54	2177	0.34	740	0.009	20	2.6	0.089	15.9	5.6
2048	4830	0.62	2990	0.018	85	3363	0.59	1990	0.009	31	6.5	0.140	37.6	13.5
4096	6769	0.76	5120	0.018	119	4773	0.99	4710	0.009	45	12.9	0.197	61.4	23.2
8192	17151	0.87	14990	0.018	303	11995	1.55	18610	0.009	114	43.0	0.489	127.8	33.1

注）@は探索１回あたりの処理時間を示す．