解　説

シストリック計算幾何アルゴリズムに関する最近の研究†

梅尾博司†† 浅野哲夫††

1. はじめに

計算幾何学 (computational geometry) とは，**幾何学** (geometry) に**計算量** (computational complexity) の理論を導入して，幾何学的な問題を計算機で効率よく処理するアルゴリズムを開発するとともに，その限界を究明する計算機科学の一分野である[1),2),18),22),23)]．近年，地理情報処理，VLSI の CAD，ロボティクス，パターン認識，コンピュータ・グラフィックスなどの分野では，多量の幾何学的情報を高速に処理する必要性が増大しつつある．計算幾何学はこれらの各分野で重要な役割を果たすものと期待されている．

過去十年間における計算幾何学の研究はノイマン型計算機を対象としてなされ，多数の効率の良い逐次型アルゴリズム (sequential algorithm) が開発されている[1),2),17),22),23)]．一方，巨大なデータの高速処理必要性と VLSI 技術の発展とが相まって，さまざまなアーキテクチャのパラレル・プロセサが百花撩乱のごとく試作・建造されている．安定した並列計算モデル上での計算幾何アルゴリズムの開発は，産業面からのニーズも著しい上記の各分野で今後必要不可欠のものとなろう．

本稿では，最近 2〜3 年の間に急速に研究が進み始めたパラレル・プロセサによる計算幾何学[7)~11),13),14)] (parallel computational geometry，なお従来の計算幾何学と今回対象とする**並列計算幾何学**を区別するために，前者を逐次型計算幾何学 (sequential computational geometry) と呼ぶ) のうち，シストリック・アレイによるもの[7)~11)] (**systolic computational geometry**) に焦点をしぼり，この分野における研究結果を整理する．

パラレル・プロセサは，ノイマン型計算機と異なりさまざまなアーキテクチャを持つ．アーキテクチャが違えば，当然アルゴリズムも異なり相互の互換性はきわめて少ない．現在のところ，どのアーキテクチャが有力であるという決め手はなく模索の段階である．したがって，この分野ではすべての研究者に共通する統一的計算モデルが存在しない．その点については，ノイマン型計算機において RAM，RASP という汎用性の高い共通モデルが存在した点と大いにちがう．一方 Kung[4),5)] の提案以来数多くの研究がなされているシストリック・アレイは，近年のプログラマブル・シストリック・チップの試作・応用，ノイマン型計算機との親和性の良さ，VISI 化にともなう数々のメリットなどから，比較的実用性が高いとみなされている[12)]．

本稿は，**シストリック計算幾何アルゴリズム**について，基本的なことがらを簡潔に説明し，これまでに提案されているアルゴリズムを整理することを目的とする．2 章では，まずシストリック・アレイ上での幾何学的データの表現法を説明し，計算幾何学問題を解く上で基本的と考えられる幾何学的演算を導入する．これらの演算は，ひとつのあるいは $O(1)$ 個のシストリック・セルが $O(1)$ 時間で実行可能なものである．次に最近点対問題ならびに交差線分対列挙問題を取り上げ，シストリック・アレイ上でいかにしてこれらの問題が $O(n)$ 時間で解決されうるかを説明する．これらは，シストリック計算幾何アルゴリズムの典型例の一つと考えられる．

3 章では，計算幾何学問題を凸包問題，重なり問題，幾何的探索問題，近接点問題，幾何的最適化問題の５つのカテゴリに分類し，それらのなかのどの問題がどのようなタイプのシストリック・アレイ上でどのくらいのセル数と時間で解決されうるかを整理する．内容はこれまでに発表された文献などで明らかになっているもの，本稿で新たに発見したものを含む．さらに未解決問題についても触れる．考察した問題は文献 1)，2)，ならびに 17) で取りあげられたものである．4 章では残された問題などについて議論する．

† Recent Studies on Systolic Computational Geometry by Hiroshi UMEO and Tetsuo ASANO (Osaka Electro-Communication Univ.).

†† 大阪電気通信大学

2. 準　備

本章では，まず $O(1)$ 時間で実行可能な幾何学的演算について説明する．次に SA 1，SA 2 と呼ばれる二つのタイプのシストリック・アレイを導入し，典型的なシストリック計算幾何アルゴリズムを説明する．SA 1 ならびに SA 2 は，線形計算[4),5)]に使用されたもので，計算幾何問題でも有用なことが明らかになろう．

2.1 シストリック・アレイ

シストリック・アレイは，通常ホスト計算機と多数の同一シストリック・セルから構成される並列処理システムである（図-1 参照）．アレイへのデータの入出力は直列に，アレイ上での処理は並列に行われ，しかもこれらの動作はパイプライン化され重畳的に進められるという特徴を持つ．詳細は文献 3)～5) を参照されたい．

2.2 幾何学的データの表現法

計算幾何学においては，次のような幾何的要素およびそれらの集合を基本データとする（図-2 参照）．

- **点** (point)
- **線分** (segment)
- **三角形** (triangle)
- **長方形** (rectangle)
- **単純多角形** (simple polygon)
- **複合長方形領域** (rectilinear region)
- **多角形領域** (polygonal region)

これらのデータは，頂点の xy 座標値（実数）の集合あるいは列として表現される．たとえば 図-2(a) に示す単純多角形は点 $P_1, P_2, \cdots, P_9$ の列：$(x_1, y_1), (x_2, y_2), \cdots, (x_9, y_9)$ として表現される．正確には，最小の y 座標を持つ点（複数個ある場合は，それらの中で最小の x 座標を持つ点）を起点とし，図形の内部を左手に見る方向（反時計方向）に境界をたどるときに出合う頂点を，出合った順序に並べる．図-2(b)，(c)における穴の境界に対しては，時計方向にたどるものとする．このような多角形あるいは領域の表現法は**標準形**と呼ばれ，一意的に定まる．また複数個の線分，面分，多角形などを区別するために，それぞれにラベルを付与する場合がある．上述した幾何図形の符号化法は，点を主体としたものであるが，辺をベースとした方が便利な場合もある．たとえば，辺の列 $e_1, e_2, \cdots, e_9$ として，図-2(a) を表現することができる．ここで $e_i = ((x_i, y_i), (x_{i+1}, y_{i+1}))$，ただし $x_{10} = x_1, y_{10} = y_1$ である．また時には，点と辺の両方のデータを併用する場合もある．

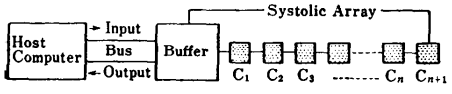
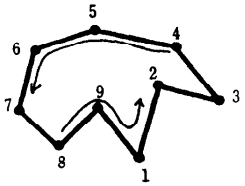
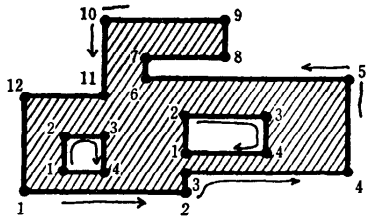


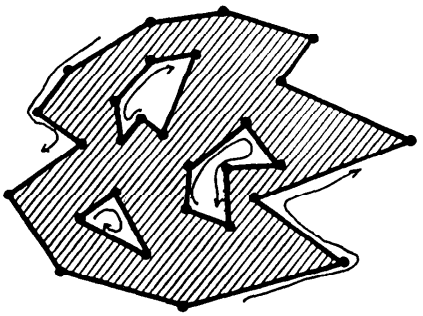
図-1　シストリック・アレイの概念図



(a) Simple polygon



(b) Rectilinear region



(c) Polygonal region

図-2　基本的な幾何データ

シストリック・アレイで計算幾何学問題を扱う場合，幾何的入力データは xy 座標値で上述の標準形に従って表現されているものとする．これらのデータは当初ホスト計算機（たとえばディスクなど）に貯えられており，必要に応じてバスを経由してシストリック・アレイに供給される．再びバスを経由して $O(n)$ 時間後に出力を回収する．出力をさらに別のシストリック・アレイに入力して処理する場合もある．

上記の幾何的図形の符号化法は，逐次型計算幾何学の場合と全く同一であるが，シストリック・アレイの場合もこの符号化法が有用と考えられる．別の符号化

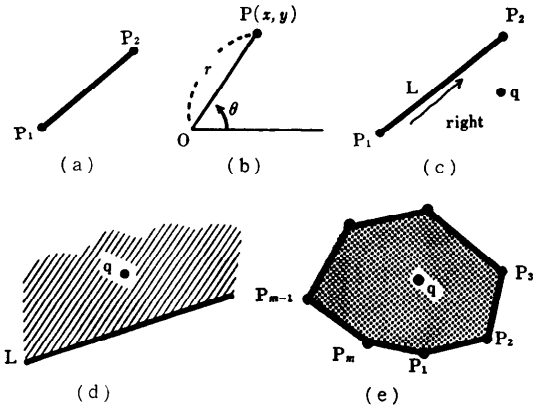


図-3 点に関する基本演算

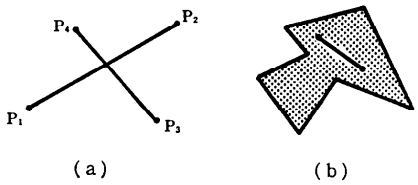


図-4 線分に関する基本演算

法として，幾何的図形を {0, 1}ᵈ 上の**2値図形**として表現する方法がある．これは主に，メモリ付き・セルオートマトンやピラミッド・マシンなどのビットマップ・アーキテクチャを持つ並列計算機上で幾何学問題を扱う場合に多く採用されている方法である．しかしながら，必要セル数の不経済さ，I/O 問題などからその有効性は明らかでない．

2.3 基本的な幾何学演算

シストリック・セルは通常の実数の四則演算に加え，次のような演算を $O(1)$ ステップで実行できる．

■点に関するもの（図-3 参照）

（a） 2点間の距離計算．

（b） 点Pの極座標計算．

（c） 方向付き線分Lと点qが与えられたとき，qがLの右（左）側に位置するか否かの判定．（f）の符号付き面積を計算することにより $O(1)$ 時間で可能．

（d） 直線 L と点qが与えられたとき，qが L の上（下）側にあるか否かの判定．線形不等式 $y>ax+b$ を評価することにより $O(1)$ 時間で可能．

（e） 点qが特定の領域 X の内部に位置するか否かの判定 (point location)．X として，三角形，長方形，円，単純凸 m 角形，単純 m 角形，複数個の穴を持つ m 角形（ただし，m は定数）などが考えられる．X が三角形，長方形，ならびに円の場合は，$O(1)$ 時間で可能である．m 角形の場合は，$O(m)$ 時間で可能と考えるのが妥当であろう．なぜなら，qを始点とする水平半直線と X 上の辺との線分交差判定（以下に述べる）により交差回数の奇偶性を調べることにより可能だからである．シストリック・アレイ上では X が単純でない場合でも $O(m)$ 時間で判定可能である．

（f） 三角形 ABC の符号付き面積の計算．△ABC の符号付き面積は次式で定義される．

$$\triangle ABC=\frac{1}{2}\begin{vmatrix} x_B-x_A, & y_B-y_A \\ x_C-x_A, & y_C-y_A \end{vmatrix}$$

この値は ABC が反時計回りの時は正，時計回りの時は負となる．

（g） 2点の偏角の大小比較．三角形の符号付き面積を計算することにより，与えられた2点 P_1, P_2 の偏角の大小比較を行うことができる．なぜなら，

$$\triangle OP_1P_2>0 \Longleftrightarrow \theta_2>\theta_1$$

が成立するからである．このとき，除算を必要としない．

■線分に関するもの（図-4 参照）

（a） 2個の線分の交差判定．三角形の符号付き面積を計算することにより判定可能．

（b） 線分Lが特定の領域 X の内部に含まれるか否かの判定．点の場合と同様に，X が m 角形の場合 $O(m)$ 時間で判定可能．両端点が X 内に含まれ，L がどの辺とも交差しない場合にかぎり L が X 内に含まれるからである．三角形，長方形，円の場合は，$O(1)$ ステップで判定可能．

■その他

それぞれ二つの三角形，長方形，円の交差判定も同様に $O(1)$ ステップで可能である．

2.4 二つの典型的なシストリック計算幾何アルゴリズム

計算幾何学問題をシストリック・アレイで解く場合，そのアルゴリズムは非常に似かよったものとなる．外見はずいぶんちがうように見えるアルゴリズムでも，アレイ上でのデータの流れなどその設計方針が

同一あるいは似かよっている場合が多い．本節では，シストリック計算幾何アルゴリズムの典型例を二つあげる．ひとつは，左端セルを I/O セルとするシストリック・アレイ **SA 1** によるものであり，もうひとつは，n 個のセルを出力セルとするシストリック・アレイ **SA 2** を利用する．

■シストリック最近点対アルゴリズム（SA 1 の利用）

次に述べる**最近点対問題** (all nearest neighbour pairs) を解くアルゴリズムは，シストリック計算幾何アルゴリズムの典型的なものと考えられる．最近点対問題とは，n 個の点 $P_1, P_2, \cdots, P_n$ が与えられたとき，すべての P_i $(1 \leqq i \leqq n)$ について，自身を除く $n-1$ 個の点の中で P_i に最も近い点 $P_{N(i)}$ を求める問題である．逐次型計算幾何学では，通常 Voronoi データ構造を利用する．n 点の Voronoi 図を作る前処理に $O(n \log n)$ 時間，n 点の最近点対を $O(n)$ 時間で見出せることが知られている[1]．

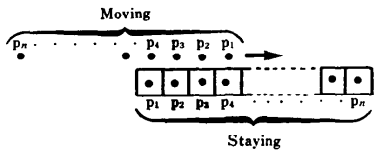
シストリック・アレイ上では次のような素朴な考え方をする．P_i の最近点 $P_{N(i)}$ を求めるには，すべての j $(1 \leqq j \leqq n,\ j \neq i)$ に対して，P_i と P_j の距離を計算しその中で最小距離を与える点 P_j を $P_{N(i)}$ とすればよい．したがって $P_{N(i)}$ を求めるのに $O(n)$ ステップを要す．すべての i について $P_{N(i)}$ を求めるために上記の計算をパイプライン化する．パイプライン・インタバルは1ステップでよい．したがって全体の計算も $O(n)$ 時間で済む．

まず n 個のセル C_i $(1 \leqq i \leqq n)$ を用意し，C_i にはあらかじめ P_i が入力されているとする．次に左端から $P_1, P_2, \cdots, P_n$ の順に1個／1ステップの割合で，すなわち，スピード1でポイント・データを流す．P_n の次にエンドマーク "♯" をつけておく．

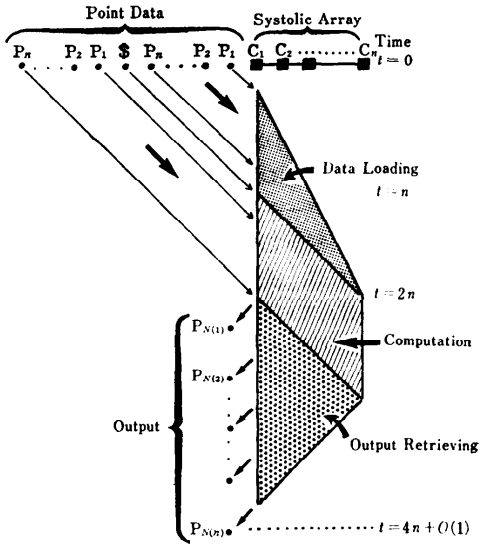
各 C_i は P_i と流れてくるポイント・データとの距離を1ステップで計算する．常に P_i との距離（これまでに流れてきた点との距離）が最小になるように，その距離とその距離を与える点データを更新する．このために $O(1)$ 個の有限レジスタを備えている．各 C_i はエンドマーク ♯ を読んだとき，各セル内に最近点 $P_{N(i)}$ を保持している．計算の様子を**図-5**(a) に示す．

右方向にスピード 1/2 で伝播する 1 bit の信号 s を利用することにより，C_i は P_i 同士との計算をしなくてすむ．s は動作開始と同時に C_i で生成される．

実際には，ポイント・データのコピーを二つ用意し，**図-5**(b) に示す時間－空間図式で計算させる．



（a） 最近点対問題を解くシストリック・アレイでのデータの動き



（b） シストリック最近点対アルゴリズムのための時間-空間図式

図-5

これによると $4n+O(1)$ ステップ後には，すべての計算結果がホスト計算機に回収される．読者は容易にセルを設計できると思われるのでこれ以上の詳細は省略する．

■シストリック交差線分対列挙アルゴリズム（SA 2 の利用）

計算幾何学の問題のなかには，入力データのサイズが $O(n)$，出力データのサイズが最悪の場合 $O(n^2)$ になる問題が多数ある．たとえば交差線分対列挙問題である．**交差線分対列挙問題** (segment intersection reporting) とは n 個の線分集合が与えられたとき，お互いに交差するすべての線分対を見つける問題である．逐次型では平面走査法を用い，$O((n+k)\cdot\log)n)$ 時間と $O(n+k)$ の空間を使用するアルゴリズムが知られている[1]．ここで k は交差線分対の総数を意味し，最悪の場合 $O(n^2)$ となる．

交差線分対列挙問題には，**図-6** に示すシストリック・アレイ SA 2 を利用する．SA 2 は，出力セルとし

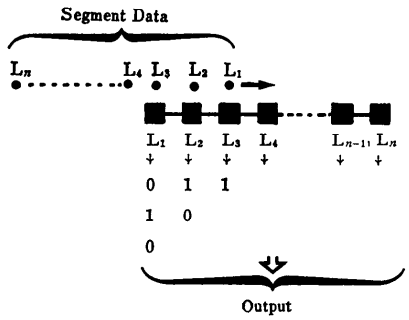


図-6 線分交差対問題を解くシストリック・アレイ SA 2

てすべての $C_i(1 \leqq i \leqq n)$ を使用する. SA2 は Kung[4),5)] らにより行列線形計算に利用されたものである.

あらかじめ線分データ L_i $(i=1,2,\cdots n)$ は各セルに1線分／1セルの割合で貯えられているものとする. 入力データの流れは, 先に示した SA 1 と同様である. C_i は自ら貯えている L_i と流れてくるデータ L_j との交差判定を $O(1)$ ステップで実行し, その判定結果に応じて 0/1 を出力する. 出力列より, どの線分がどの線分と交差するかを知るのは容易である. 全体で $3n+O(1)$ ステップで終了する.

n 個の出力ポートは, アレイへの入出力スピードとその上での計算スピードとをバランスさせ, セル間の内部通信量を $O(1)$ ビットに保つために必要である. この出力ポートのサイズは, 実現性という点で若干疑問が残る. しかしながら, 出力サイズが $O(n^2)$ で I/O ポートが $O(1)$ サイズであれば, 符号化などを考慮しないかぎり出力に $O(n^2)$ 時間を要す.

3. 最近の研究成果

計算幾何学における大部分の問題は, 次の5つのカテゴリに分類される. すなわち,

- **凸包問題** (convex hulls),
- **重なり問題** (intersection),
- **幾何的探索問題** (geometrical search),
- **近接点問題** (proximity),
- **幾何的最適化問題** (geometrical optimization).

各問題の詳細な説明, 応用分野, これまでに知られている最良 (最適) 逐次型アルゴリズムなどに関しては, 文献 1), 2), 17), 22), 23) を参照されたい.

計算幾何学における大部分の問題は問題の性質上, **静的** (static) な側面と**動的** (dynamic) な側面を持っている. 本稿では, まず静的な場合を扱うシストリック・アルゴリズムを解説し, 次に動的なそれについて述べる.

シストリック計算幾何学を扱った論文は, 筆者の知るかぎり, Chazelle[7)], Asano and Umeo[11)], Kane and Sahni[8)~10)], Savage[6)] の6編である.

文献 7) は, 凸包問題, 点位置決定問題, 線分／半平面交差問題, 最近点対問題ならびに三角形分割問題の動的な側面に焦点をあて, これらの問題を $O(n)$ 時間で解決する SA 1 タイプのシストリック・アレイを与えた. しかしながらその記述は非常に不透明で理解するのが難しい. 文献 11) は, 先に説明した最近対点アルゴリズムのデータ・フローに着目し, 可視問題, 台形／三角形分割問題を扱った. 文献 8)～10) は, 集積回路の設計自動化における重要ないくつかの問題を扱い, 特にデザイン・ルールチェック, 複合長方形の論理 AND/OR 操作, ならびにネット抽出を行うシストリック・アレイを与えた. 文献 6) は, 連結グラフの全域木を計算するシストリック・アレイを与えた. 文献 7) は, これを利用して, ユークリッド平面上における n 点の最小全域木を計算するアルゴリズムを与えた.

3.1 凸包問題

凸包問題とソーティングは密接な関係がある. シストリック・アレイは, $O(n)$ の時間で n 個のデータをソートできる. これを利用して, 凸包問題も同様に $O(n)$ 時間で解決できる. たとえば逐次的ソーティングを利用して凸包を計算する Graham の方法[1)] (**3.3** 節, 時間計算量は $O(n \log n)$) があるが, シストリック・ソータを利用すれば, 簡単に $O(n)$ の凸包計算シストリック・アルゴリズムが得られる.

■静的凸包計算

Chazelle[7)] は, 次の補題を用いて $O(n)$ 時間のシストリック凸包アルゴリズムを得た. **図-7** に示すような斜線の領域 $(0°<\theta<180°)$ を, Mを中心とする**凸ウェッジ**と呼ぶ. このとき, 次の補題が成立する.

[補題] $S=\{M_1, M_2, \cdots, M_n\}$ を平面上の n 点の集合とする. このとき, 次の二つの命題(1), (2)は同値である.

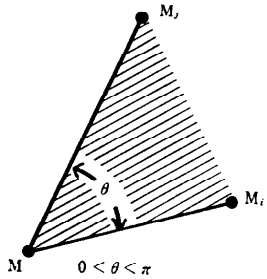


図-7 M を中心とする凸ウェッジ

(1) Sの点MがSの凸包上の点である.

(2) Sのすべての点を含みMを中心とする凸ウェッジが存在する.

最近点対問題と全く同様に解く. 各セル C_i は次々と送られてくるポイント・データをもとに M_i を中心とする最大角ウェッジを計算する. この計算はMを原点としたときの偏角を調べることにより容易に1ステップでできる. すべてのデータを読み終えたとき, 最大角ウェッジが凸であるような点のみ出力する. 出力データは高々 n 個である. 全体で $4n+O(1)$ ステップで終わる.

上の方法で出力される凸包上の頂点は, 意味のある順序を持っていない. n 点の重心を原点としたときの凸包上の頂点の偏角を計算し, それらをシストリック・ソータに入力後整形すれば, 反時計方向の順に頂点をならべることができる. 実際には, 前章で述べた符号付き面積を計算することにより, 2点の偏角の大小比較を行う. 時間計算量はそれでも $O(n)$ である.

■動的凸包計算

Chazelle[7] は, 次の命令集合を実行する動的シストリック・アレイ CH1, CH2 を提案した. アレイの構造は SA1 タイプである.

・CH1:

(1) 点Mを挿入/削除せよ,

(2) 凸包上のすべての辺を出力せよ,

(3) Mが凸包内に存在するか.

・CH2:

(1) 点Mを挿入せよ,

(2) 凸包上のすべての辺を半時計方向の順に出力せよ,

(3) Mが凸包内に存在するか.

両者の設計において(1)は $O(1)$ ステップのパイプライン・インタバルで実現されているが, (2), (3)は $O(n)$ ステップの応答時間を持つ.

CH1 は削除命令をサポートするかわりに, 凸包上の辺を意味のある順序では出力できない. CH2 はその逆である. それぞれ一長一短がある. 両者の欠点を補うものとして文献 7) では, CH1 の右側に CH2 を接続し, (2)の出力命令が CH1 にきたときにだけ CH1 上のデータを CH2 に順次送り CH2 に凸包上のすべての辺を半時計方向順に出力させることを考えている.

点データの挿入/削除命令をどのようにしてシストリック・アレイ上で実現するかについては, 以下で述べる幾何的動的探索問題を参照されたい.

3.2 重なり問題

重なり問題は, 多数の線分, 長方形, 半平面などの重なりの検出 (detection), 重なっている対の数え上げ (counting), 重なっている対の列挙 (reporting), 重なり領域の計算 (construction) などを扱う. 他に可視領域の計算問題 (visibility problem) がある.

●交差列の検出/数え上げ

n 個の線分が与えられているとする. この中で, 互いに交差するものがあるか否かを検出する問題を考える. 最近点対問題におけるポイント・データを線分データに置き換える. 各セルは, 前章に示した2線分の交差判定を実行する. 要求される答えは, yes/no あるいは1個の整数であるので, もしあるセル上で交差が検出されれば, それを左端のセルに伝えれば良い. 四則演算が $O(1)$ 時間で可能なセルを仮定しているので, 交差列の総数も同様に左端のセルに集めることができる. 全体で $4n$ ステップで終了する. 円, 長方形などの交差判定も同様に可能である.

●交差対の列挙

前章ですでに説明. SA2 を利用. 線分のかわりに三角形, 長方形, 円にかえても同様に議論が可能である.

●交差部分の計算

二つの多角形 P, Q (頂点数をそれぞれ m, n とする) あるいは n 個の長方形, 半平面が与えられたとき, 重なっている領域を計算する問題である.

(i) P, Q がともに凸多角形である場合: $\theta(m+n)$ 時間の逐次型アルゴリズムが存在する. したがってシストリック・アレイによる高速化は必要なし.

(ii) P, Q が単純な多角形である場合: 重なり部分は必ずしも一個の多角形にならず, nm 個の領域からなる場合がある. 逐次型では $O((m+n)\log(m+n)+k)$ (k は出力サイズを意味する) 時間で動作するアルゴリズムが知られているが, $O(m+n)$ 時間のシストリック・アルゴリズムは知られていない.

(iii) n 個の長方形の場合: 文献 10) は, n 個の長方形の AND/OR 操作を実行するシストリック・アルゴリズムを与えた.

(iv) n 個の半平面: 逐次型アルゴリズムでは, $\Omega(n\log n)$ 時間を要することが知られている. 文献 7) は, 次の基本命令集合を実行し動的にこの問題を解決するシストリック・アレイを提案した. 基本命令集合:(1)一つの半平面の挿入/削除命令, (2)重なり

部分を構成する辺の列挙命令，(3)点位置決定命令．

●多角形の包含判定問題

平面上に与えられた二つの単純多角形 P, Q に対し，平行移動と回転を許したとき，P が Q を含むか否かを判定する問題は，逐次型で $O(m^3n^3(m+n)\times\log(m+n))$ 時間で動作するアルゴリズム[2]が知られている．このような問題こそシストリック・アレイによる並列化が重要と考えられるが，いまだその線形時間アルゴリズムは知られていない．

3.3 可視問題

可視問題とは，幾何学的要素 P (たとえば，線分集合，多角形（の集合)) と一つの光源 q が与えられたとき，光があたる P の部分領域を計算する問題である．コンピュータ・グラフィックスの分野で，**隠線／隠面消去問題**として古くから知られている．

光源の種類として

- **点光源**：visibility problem from a point,
- **平行光線**（無限遠点にある点光源）：parallel view,
- **有限長の線分上に置かれた無数の点光源**：visibility problem from an edge,

などが考えられている．P が単純多角形である場合，上記の光源に対して $\theta(n)$ で動作する逐次型アルゴリズムが知られている．しかしながら P が多数の多角形，線分集合から成る場合は，可視問題はソーティング問題と密接に関係し $\Omega(n\log n)$ 時間を必要とすることが知られている．

文献 11) は，P が多角形領域，q が点および平行光線の場合の可視領域を求めるシストリック・アルゴリズムを与えた．P が n 個の頂点からなる場合，SA 1 タイプのシストリック・アレイ上で $O(n)$ 時間で解くことができる．P が多数の線分からなる場合も同じ考え方で容易に解くことができる．

n 個の垂直線分が与えられたとき，互いに見ることができる線分対を見いだす問題 (parallel view の一種と見なせる) は，Lodi and Pagli[14] らにより，直交木アレイ (orthogonal mesh of tree machines) 上で $O(\log n)$ で解けることが示された．SA 1 タイプのシストリック・アレイ上では，本稿で示してきたアルゴリズムを少し工夫することにより $O(n)$ 時間で解くことが可能である．

表-1 に重なり問題における主なアルゴリズムを整理する．表中＊マークは，本稿で新たに得られた結果である．それらの証明などの詳細は紙面の都合上省略する．？マークはシストリック・アルゴリズムが知られていない問題を示す．

表-1 重なり問題を解くシストリック・アルゴリズム

Intersection Problem	Systolic Computational Geometry Algorithm Time complexity and array-type
Segment (circle, rectangle) intersection detection	$O(n)$ [*] SA 1
Two-polygon intersection (inclusion) detection	$O(n)$ [Chazelle] SA 1
Segment (circle, rectangle) intersection counting	$O(n)$ [*] SA 1
Segment (circle, rectangle) intersection reporting	$O(n)$ [*] SA 2
Intersection construction of two (star-shaped) simple polygons	?
n Half-plane intersection construction	$O(n)$ [Chazelle] SA 1
n-Rectangle AND/OR intersection construction	[Kane and Sahni] SA 1
d-Range intersection counting ($d\geqq 1$)	$O(n)$ [*] SA 1
Isothetic rectangle containment reporting	$O(n)$ [*] SA 2
Polygon containment detection with rotation and translation	?
Visibility polygon from a point	$O(n)$ [Asano and Umeo] SA 1
Parallel view of visibility polygon	$O(n)$ [Asano and Umeo] SA 1
Visibility of vertical n segments	$O(n)$ [*] SA 1
Maintenance of visibility segments (dynamic)	$O(n)$ [*] SA 1

3.4 幾何的探索問題

幾何的探索問題とは，平面上の与えられた幾何的対象（点，線分，多角形など）の集合 S に関する質問 Q が与えられたとき，Q に対してある条件（交わる，含む，右にある，など）を満たす S の要素を検索する問題である[19]．S を**台集合**と呼ぶ．S に対し，通常複数回の問い合わせがなされる．

問い合わせの条件に応じて，探索問題は，

（1） **条件を満たす S の要素があるか否かを検出する問題**

（2） **条件を満足する要素がいくつあるかという数え上げ問題**

（3） **条件を満足する要素をすべて列挙する問題**

に分類される．

検出あるいは数え上げ問題の場合，ひとつの問い合わせに対する応答は，それぞれ yes/no あるいは1個の整数である．列挙問題では最悪の場合 $O(n)$ 個の要素が出力されることに注意されたい．

複数回の問い合わせがなされるとき，問い合わせの間に台集合のデータが更新されうるか否かにより，探索問題は動的あるいは静的であると言われる．実用上，**動的複数回問い合わせ可能なデータ構造**が重要である．

静的探索問題に対しては数多くの逐次型高速アルゴリズムが発見されている[1),2),17),22),23)]．しかしながら，動的な場合の同様な高速アルゴリズムの開発はまだあまりなされていないのが現状である．シストリック・アレイ上では，動的な問題が比較的高速に（シストリック・アーキテクチャであるかぎり最適時間で），しかも単純なセルの組み合わせで実現できることが以下で明らかになろう．

シストリック探索問題では，アルゴリズムの性能を測るものとして，次の測度が重要である．すなわち，

・必要なセル数 $S(n)$,

・1回の問い合わせ後，それに対する応答が得られるまでの応答時間 $Q(n)$,

・多数の問い合わせをするときのパイプライン・インタバル $P(n)$.

動的探索の場合問い合わせ命令のほかに，台集合に対するデータの挿入／削除命令が加わる．上記3種類の命令を考慮した場合のパイプライン・インタバルは，実用上からも重要である．また場合によっては，すでに S 内に存在するデータと同一データを冗長に挿入したり，存在しないデータを削除したりするときがありうるが，本稿で与えるシストリック・アレイは上記のようなケースにも矛盾なく対処可能である．このような命令をそれぞれ**冗長挿入命令**，**冗長削除命令**と呼ぶ．

■探索問題を応答時間 $O(n)$，パイプライン・インタバル $O(1)$ で解くシストリック・アレイ

幾何的探索問題の代表として，線分交差探索問題，点位置決定問題，領域探索問題および，点包囲問題の4つがあげられる[19)]．ここでは，動的線分交差探索問題を解くシストリック・アレイを設計しよう．他の問題も SA 1，あるいは SA 2 で同様に解くことができる．線分交差探索問題とは，n 本の線分の集合 S と質問線分 L が与えられたとき，L と交わる S の線分すべてを列挙する問題である．

シストリック・アレイは SA 1 あるいは SA 2 タイプを利用する．SA 2 は列挙的問い合わせ命令を $O(1)$ のパイプライン・インタバルでサポートする．質問 Q はホスト計算機から命令列 $I_1, I_2, \cdots$，として与えられる．ここで Q は次の命令セットのなかの任意の命令である．

Q：{L を挿入せよ (Insert (L)),
　　L を削除せよ (Delete (L)),
　　L と交差する S の線分があるか否か？,
　　L と交差する S の線分の総数は？,
　　L と交差する S の線分をすべて列挙せよ}

動的探索問題はシストリック・アレイA上で次のようにして解かれる．まず Insert (L) 命令をどのように実現するか説明しよう．台集合の要素（データ）L は左端のセルから順につめられる．各セルは3個のレジスタ R_1, R_2, R_3 を持つ．データは1データ／1レジスタの割合で貯えられる．アレイ上の R_i $(1 \leqq i \leqq 3)$ を第 i 層と呼び U_i で示す．データ L は**図-8(a)**に示すように U_2 上を右方向に進み，R_1 が空いている最初のセルを捜し，そこに落ちつく．したがって台集合のデータは U_1 上に貯えられる．U_1 上の最右端のデータには，エンド・マーク "#" をつけておく．Delete (L) も U_2 上を右方向に進む．各セルで R_1 の内容と比較しながら，R_1 にLを含んでいるセル（セルFとする）を見つけると，セルFの R_1 を空にする．セルFより右にデータを持ったセルがあれば，それらの R_1 のデータを順次左方向に1駒移動させる．これは，セルFより右方向に伝わる信号により制御される．Delete (L) 命令は，このセル上で消滅し以後右方向には伝わらない（**図-8(b)**参照）．Insert/Delete 命令が冗長である場合も上記の議論を少し変更するだけで同様に実現することができる．

Lと交差する線分の検出／交差総数探索命令も U_2 上を右に進みながら，各セルの R_1 との交差判定をくり返す．各セルで得られた部分解はタグとしてその命令に付与され，命令とともに移動する．命令が最右端に到達したとき，上記二つの命令に対する解を得るこ

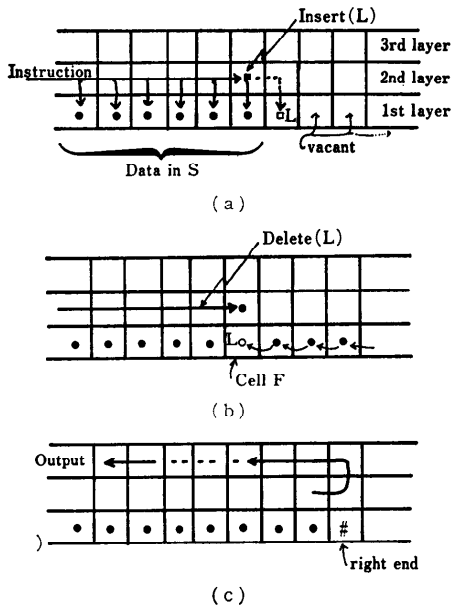


図-8 パイプライン・インターバル $O(1)$ でデータを挿入(a)，削除(b)，回収(c)するシストリック・アレイ

とができる．答は U_3 上を左方向に進みホスト計算機に回収される（図-8(c)参照）．SA 1 タイプのAは，上記4つの命令を $S(n)=O(n)$，$Q(n)=O(n)$，$P(n)=O(1)$ で実現できる．n は台集合 S のサイズを意味する．

一つの列挙命令に対して，出力サイズは通常 $O(n)$ である．したがって，SA 1 上では $P(n)=O(1)$ で列挙命令を実現することはできない．なぜなら I/O 幅が $O(1)$ であるから．列挙命令を含む探索問題には SA 2 タイプが適している．SA 2 タイプのシストリック・アレイは，上記5つの命令を，$S(n)=O(n)$，$Q(n)=O(n)$，$P(n)=O(1)$ で実現できる．

静的探索に関しては，次の問題に言及するにとどめる．他のいろんな問題も，同様に取り扱える．佐藤ら[25]は，複合長方形領域（台集合）の内点と外点を分類する $O((m+n)\log m)$ 時間逐次型アルゴリズムを与えた．ここで，m は複合長方形を構成する線分数，n は分類すべき点の総数を意味する．我々には n 点を $O(m+n)$ ステップで分類する SA 1 タイプのシストリック・アレイの構成は容易である．

幾何的探索問題に対して，次のことが言える．点位置決定問題，ならびに各種の検出／数え上げ問題は，一つの問い合わせに対して $O(1)$ の出力サイズを持つ．他方，一般的な領域探索問題，点包囲問題ならびに交差探索問題においては，一つの問い合わせに対して $O(n)$ サイズの出力がある．シストリック・アレイで上記の問題を解く場合，動的あるいは静的いずれの場合でも，前者の問題に対しては SA 1 タイプが，後者には SA 2 タイプが適している．いずれの場合も $S(n)=n$（あるいは $O(n)$），$Q(n)=n$（あるいは $O(n)$），$P(n)=1$（あるいは $O(1)$）である．

■動的なシストリック・幾何的探索アルゴリズムの特徴

1. 探索時間が $O(n)$ であり，前処理した逐次型アルゴリズム（探索時間は大部分が $O(\log n)$ である）よりは遅い．
2. 前処理が不要である．逐次型アルゴリズムで

表-2 シストリック・幾何的探索アルゴリズム

Geometric searching Problem	Systolic Computational Geometry Algorithm $S(n)$: number of cells $Q(n)$: response time for one-time query $P(n)$: pipeline interval Time complexity and array-type	
	Static	Dynamic
Range search counting/detecting	$S(n)=n^{[*]}$ $Q(n)=O(n)$ $P(n)=O(1)$ SA1	$S(n)=O(n)^{[*]}$ $Q(n)=O(n)$ $P(n)=O(1)$ SA1
Point location	[Chazelle] $S(n)=O(n)$ $Q(n)=O(n)$ $P(n)=O(1)$ SA1	$S(n)=O(n)^{[*]}$ $Q(n)=O(n)$ $P(n)=O(1)$ SA1
Range search reporting		$S(n)=O(n)^{[*]}$ $Q(n)=O(n)$ $P(n)=O(1)$ SA 2
k-Size-polygon-range retrieval		
Fixed/variable-disk retrieval		
Range search reporting in a set of line segments (window clipping)		
Orthogonal intersection searching • Point enclosure • Segment • Rectangle		
Polygon intersection searching		

は，通常 $O(n)$, $O(n \log n)$ あるいは $O(n^2)$ 時間の前処理を要する場合が多い．

3. パイプライン・インタバルは $O(1)$．多数の問い合わせをするときは，実質的に $O(1)$ の応答時間となり，逐次型と比較して高速である．逐次型のパイプライン・インタバルは $O(\log n)$ であることに注意されたい．したがって，1. の応答時間の遅さは，多数の問い合わせがなされる場合相殺される．

表-2 に SA 1，SA 2 で解くことのできる探索問題をまとめる．

3.5 近接点問題／幾何的最適化問題

最近接点対問題，ボロノイ図の構成，三角形，台形，長方形などへの基本図形分割が近接点問題の代表である．

最近接点対問題については前章で触れた．n 点が与えられたとき，**最も近接している2点を探す問題 (closest pair)**，**n 点のユークリッド最小極大木の構成問題**などは，SA 1 タイプのシストリック・アレイ上で $O(n)$ 時間で解けることが知られている[6),7)]．

n 点のボロノイ図を構成するシストリック・アレイは知られていない．シストリック計算幾何学においては重要な未解決問題の一つである．

多角形を三角形，台形，凸多角形などの基本図形にシストリック・アレイ上で分割する問題は，Chazelle[7)]，Asano and Umeo[11)] らにより考察されている．文献 7) は，三角形分割を動的に実行する SA 1 タイプのシストリック・アレイ TR 1 を提案した．TR 1 は n 点の凸包を $O(n)$ 時間で三角形に分割し，次の三種類の命令をサポートする．

（1） 点Mの挿入，
（2） 点Mを含む三角形の探索，
（3） すべての三角形の列挙．

文献 11) は複数個の穴を含む多角形領域を台形，ならびに三角形に $O(n)$ 時間で分割する SA 1 タイプのシストリック・アレイを示した．

複合長方形領域をいくつかのあるいは最小個数の長方形に分割する高速アルゴリズムは，LSI のアートワークなどで重要な役割を果たす．今井，浅野[23)]，大附ら[24)]は，本問題に対して $O(n^{3/2} \log n)$, $O(n \log n)$ 時間で動作する逐次型アルゴリズムを提案している．

文献 11) の分割アルゴリズムと同じ考え方で，複合長方形領域（いくつかの穴を含む）を長方形の集合に分割する SA 1 タイプのシストリック・アレイ（時間計算量は $O(n)$）を構成することは容易である．

表-3 近接点問題に対するシストリック・アルゴリズム

Proximity and Related Problem	Systolic Computational Geometry Algorithm Time complexity and array-type
Closest pair	$O(n)$ [Chazelle] SA 1
All nearest neighbours	$O(n)$ [Chazelle] SA 1
Euclidean minimum spanning tree	$O(n)$ [Chazelle], [Savage] SA 1
Triangulation of point set (dynamic)	$O(n)$ [Chazelle] SA 1
Nearest neighbour search (Post Office Problem)	$O(n)$ [Chazelle] SA 1
k-Nearest neighbours search	$O(n)$ [*] SA 1
The Voronoi diagram	?
All nearest neighbours for polygons	$O(n)$ [*] SA 1
Farthest pair or diameter problem	$O(n)$ [*] SA 1
Closest pair between sets	$O(n)$ [*] SA 1
Fixed-radius near neighbour	$O(n)$ [*] SA 1
Shortest path problem with obstacles	?
Triangulation of a polygonal region	$O(n)$ [Asano and Umeo] SA 1
Quadrilaterization of a rectilinear polygon	$O(n)$ [*] SA 1
Decomposition of rectilinear polygon with holes into rectangles	$O(n)$ [*] SA1

最大空円（正方形，長方形）問題，最小包含円（長方形）問題，円（長方形）配置問題などが，幾何的最適化問題の代表である．n 点が与えられたとき，それらをすべて含む最小長方形（向きは任意）を求める問題（最小包含長方形問題）は，これまでと同様な考え方で SA 1 タイプのシストリック・アレイ上で $O(n)$ 時間で解くことができる．他の問題は，たとえばアレ

イ・サイズを $O(n^2)$ に増加させるなどシストリック・アレイの能力を拡大すると $O(n)$ 時間アルゴリズムが得られるが，SA1/2 タイプのアレイ上では線形時間アルゴリズムは知られていない．今後の研究が期待される領域である．表-3 に近接点問題およびそれらに関する問題のシストリック・アルゴリズムの性能をまとめる．

4. おわりに

計算幾何学問題を並列計算機上で解く試みは最近始まったばかりである．1990 年代には，大部分の計算機はなんらかの並列アーキテクチャを採用する見通しである．計算幾何学問題を並列アーキテクチャ上でいかに扱うかということに関する知見は，VLSI の CAD，ロボティクス，コンピュータ・グラフィックスなどの分野でその必要性が高いにもかかわらず，ほとんど集積されていない．

本稿では，これまでに十二分に議論しつくされ，非常に単純な構造を持つ二つのタイプのシストリック・アレイ上でどのような計算幾何学の問題が線形時間で解けうるかを整理した．大部分の問題は，シストリック最近点対アルゴリズムと同じか，それを少し変更した制御フローで解ける反面，同じ考え方では解決が難しい問題，たとえばボロノイ図の構成問題などもあることが判明した．

シストリック計算幾何アルゴリズムは次の特徴を持つ．

1. セル間の制御アルゴリズムが比較的単純である．

2. 逐次型計算幾何アルゴリズムと比較して，前処理が不要な場合が多い．

3. プロセッサ数（セル数）・時間積という観点からみると，シストリック・アルゴリズムは最適でない．通常シストリック・アレイはサイズ n の問題に対し，$O(n)$ 個のセルを使用し，$O(n)$ 時間で結果を得る．プロセッサ数・時間積という観点からは，シストリック・アルゴリズムは $O(n^2)$ になり，最適アルゴリズムではない．なぜなら，大多数の計算幾何アルゴリズムは逐次型 RAM 上で $O(n \log n)$ の時間複雑度を持っているからである．

4. 大部分のシストリック計算幾何アルゴリズムは，理解しやすく正当性の証明も容易である．しかし，いくつかの問題は理解が非常に難しいものもある[6),10)]．

5. シストリック・アレイの入出力データの前処理／後処理として，他の用途のシストリック・アレイが使われることがある．特に，シストリック・ソータは有用である．逐次型計算幾何学におけるいくつかの問題の下界 $\Omega(n \log n)$ は，ソーティング問題が $\Omega(n \log n)$ であるという事実に由来している．ところが，シストリック・アレイでは，ソーティングが非常に簡単に $O(n)$ ステップで実現できる（たとえば，バブルソートを並列化すれば良い）ので，必要な時間計算量を増加させることなく，前／後処理フィルタとして使用できる．

次の特徴はシストリック・アレイ全般にあてはまる．

6. 時間計算量 $O(n)$ の係数が比較的小さく，かつどのくらいの大きさかをほぼ正確に評価できる．これは，他の並列計算モデル，たとえば concurrent read/exclusive write parallel RAM などで議論する場合[13)]と本質的に異なる点である．いろいろな用途のシストリック・アレイが実際に設計されている[12)]という点からも，シストリック計算幾何アルゴリズムは他の並列計算モデル上のものよりは比較的実用性が高いと考えられる．

最近，Tarjan and Van Wyk[15)] らは，n 点からなる単純多角形の三角形分割が逐次型計算機上で $O(n)$ 時間で可能なことを示した*．これにともない，いくつかの問題（たとえば，可視問題[16)]）の時間計算量が $O(n \log n)$ から $O(n)$ に改良されている．したがってシストリック・アレイあるいは，他のパラレル・マシンで解かなくても十分高速化されうる可能性のある問題が現れている．このような理由から並列計算幾何学の研究はもう少し時間的猶予が必要かも知れない．

最後に，今後研究が必要と考えられるテーマをあげておく．

・三次元物体を扱う効率の良いアルゴリズムの設計（逐次型，並列型の両方）．

・動的問題を効率的に扱う統一的手法，アーキテクチャの提案．たとえば，parallel geometry query machine のアーキテクチャなど．

・ピラミッド・マシン，セルラ・オートマトン（2次元）などビットマップ・アーキテクチャを持つ並列計算モデルが多数提案されている．2値図形として表現された幾何学問題をビットマップ・アーキテクチャ上で解

* 最近，彼らの $O(n)$ 時間アルゴリズムは誤っており，正しくは $O(n \log \log n)$ 時間アルゴリズムであると伝えられているが，詳細は不明である．

決する並列アルゴリズムの開発.

謝辞 論文を改善する上で貴重なご助言をいただいた査読者に謝意を表す. なお本研究の一部は文部省・科学研究費 (No. 61750346 ならびに No. 61750347) より補助をうけた.

参考文献

1) Preparata, F. P. and Shamos, M. I.: Computational Geometry, an Introduction, Springer-Verlag, New York, p. 390 (1985).
2) Lee, D. T. and Preparata, F. P.: Computational Geometry-A Survey, IEEE Trans. Comput, Vol. C-33, No. 12, pp. 1072-1101 (1984).
3) Leiserson, C. E.: Systolic Priority Queues, Proc. of the Caltech Conf. on VLSI (Deitz, C. E. ed.), pp. 199-214 (1979).
4) Kung, H. T.: Let's Design Algorithms for VLSI Systems, Proc. of the Caltech. Conf. on VLSI (Deitz, C. E. ed.), pp. 55-90 (1979).
5) Kung, H. T. and Leiserson, C. E.: Algorithms for VLSI Processor Arrays, in Chapter 8 of Mead, C. A. and Conway, L. A. "Introduction to VLSI Systems", Addison-Wesley, Reading, Massachusetts (1980).
6) Savage, C.: A Systolic Data Structure Chip for Connectivitity Problems, in "VLSI Systems and Computations" (Kung, H. T., Sproull, R. F. and Steele, G. L. Jr. eds.), pp. 296-300 (1981).
7) Chazelle, B.: Computational Geometry on a Systolic Chip, IEEE Trans. Comput., Vol. C-33, No. 9, pp. 774-785 (1984).
8) Kane, R. and Sahni, S.: A Systolic Design Rule Checker, Proc. of the IEEE 21st Design Automation Conf., pp. 243-250 (1984).
9) Kane, R. and Sahni, S.: Hardware Algorithm for Net Extraction, Proc. of the IEEE ISCAS 85, pp. 51-54 (1985).
10) Kane, R. and Sahni, S.: Systolic Algorithms for Rectilinear Polygons, Proc. of the IEEE Intern. Conf. on Computer Design: VLSI in Computers ICCD '84, pp. 831-836 (1984).
11) Asano, T. and Umeo, H.: Systolic Algorithms for Computing the Visibility Polygon and Triangulation of a Polygonal Region, Tech. Rep. of IECE of Japan, pp. 53-60 (1986).
12) Arnold, E., Kung, H. T., Menzilcioglu, O. and Sarocky, K.: A Systolic Array Computer, Proc. of the IEEE Int. Conf. on Acoustics, Speech and Signal Processing, pp. 232-235 (1985).
13) Aggarwal, A., Chazelle, B., Guibas, L., O' Dunlaing, C. and Yap, C.: Parallel Computational Geometry, Proc. of the IEEE Conf. on FOCS '85, pp. 468-477 (1985).
14) Lodi, E. and Pagli, L.: A VLSI Algorithm for a Visibility Problem, in VLSI: Algorithms and Architectures (Bertolazzi, P. and Luccio, F. (eds.)), pp. 125-134, Elsevier Science Publishers B. V., North Holland (1985).
15) Tarjan, R. E. and Van Wyk, C. J.: A Linear-time Algorithm for Triangulating Simple Polygons, Proc. of the Annual Symp. on Theory of Computing (1986).
16) Toussaint, G. T.: A Linear-time Algorithm for Solving the Strong Hidden-line Problem in a Simple Polygon, Tech. Tep. of Mcgill Univ., No. SOCS-86, 2 (1986).
17) Toussaint, G. T.: Pattern Recognition and Geometrical Complexity, Proc. of the 5th Int. Conf. on Pattern Recognition, pp. 1324-1347 (1980).
18) 浅野孝夫：計算幾何学とその応用，情報処理，Vol. 25, No. 3, pp. 208-221 (1984).
19) 浅野孝夫：幾何学的探索アルゴリズムとその応用，Proc. of the 6th Mathematical Programming Symposium, pp. 205-220 (1985).
20) 新實治野，富田眞治：グラフィックスと専用マシン，bit, Vol. 17, No. 10, pp. 1189-1211 (1985).
21) 佐藤政生，大附辰夫：VLSI 設計における計算幾何学の応用，bit, Vol. 17, No. 4, pp. 483-489 (1985).
22) 浅野哲夫，浅野孝夫：計算幾何学とは，「コンピュータと数学」5，コンピュータから生まれた新しい数学(野崎昭弘，廣瀬 健編)，数学セミナー別冊，pp. 184-192 (1986).
23) 今井 治，浅野孝夫，伊理正夫：計算幾何学 1, 2, 3, 4, 完，bit, Vol. 16, No. 12, pp. 52-61; Vol. 16, No. 13, pp. 51-61; Vol. 17, No. 1, pp. 108-116; Vol. 17, No. 2, pp. 98-107; Vol. 17, No. 3, pp. 92-100 (1985).
24) 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).
25) 佐藤政生，橘 昌良，鳥居司郎，大附辰夫：複合長方形領域の内点と外点を分類するためのアルゴリズム一実装設計への応用一，電子通信学会論文誌，Vol. J66-D, No. 2, pp. 214-219 (1983).

(昭和 61 年 6 月 26 日受付)