分割統治法に基づいた線形時間・画像連結要素ラベリング・アルゴリズム†

梅尾 博司††

近年のハードウェア価格の低下とともに，種々のアーキテクチャをもつ並列画像処理装置が開発され，それらの上で動作する数多くの並列画像処理アルゴリズムが提案されている．本稿では，従来から高速アルゴリズム設計法の一つとして知られている分割統治法を並列画像処理アルゴリズム設計に適用し，完全並列型プロセッサの一種と考えられるメモリ付きセルラオートマトン上で，線形時間で動作する連結要素ラベリング・アルゴリズムを提案する．連結要素のラベリングは，本来逐次的な性質ゆえにその並列化が困難なものと考えられてきた．本演算に関しては，自明な面積時間アルゴリズムが知られているだけで，線形時間アルゴリズムの存在については，理論的にも興味ある未解決問題として残されていたものである．時間計算量の減少とともに，プロセッサ当りのハードウェア量が増大し，実用化という観点からはさらに改良を重ねる必要がある．

1. まえがき

近年のハードウェア価格の低下とともに，種々のアーキテクチャをもつ並列画像処理装置が開発され，それらの上で動作する数多くの並列画像処理アルゴリズムが提案されている[1)-3),7),9),13)]．本稿では，従来から高速アルゴリズム設計法の一つとして知られている分割統治法[4)]を並列画像処理アルゴリズム設計に適用し，完全並列型プロセッサ[1)]の一種と考えられるメモリ付きセルラオートマトン[7)]上で，線形時間で動作する連結要素ラベリング（通し番号付け）・アルゴリズムを提案する．連結要素のラベリングに関しては，連結性の判定などとともに数多くの研究[5),8),10)-12)]がなされているが，本来逐次的な性質ゆえにその並列化が困難なものと考えられてきた．本演算に関しては，自明な面積時間アルゴリズムが知られているだけで，線形時間アルゴリズムの存在については理論的にも興味ある未解決問題として残されていたものである．時間計算量の減少とともに，プロセッサあたりのハードウェア量が増大し，実用化という観点からはさらに改良を重ねる必要がある．

2章では，連結要素ラベリング問題ならびに完全並列型・イメージ・プロセッサに関する諸定義を与える．3章では，まず完全並列型・イメージ・プロセッサ上における分割統治法の制御アルゴリズムを示す．次に，$n\times n$ 画像の連結要素ラベリングを $O(n)$ の時間で実行する並列アルゴリズムを提案する．4章では，分割統治法を利用した他の画像処理アルゴリズム設計について考察し，さらに今後に残された問題等について議論する．

2. 準　備

2.1 連結要素ラベリング

$n\times n$ の2値画像 $\{a_{ij}\}(a_{ij}\in\{0,1\})$ を入力画像と考える．ただし，$0\leqq i,j\leqq n-1$，$n=2^m$，m は任意の自然数．各要素 a_{ij} は，通常の方法により導入された x-y 平面の格子点 (i,j) に位置するます目上に記入されているものとする．“0” が記入されているます目を画像の“背景”，“1” が記入されているます目を考察対象画像と考える．

本稿では，“1” のます目に対して 4-隣接性（以下ではたんに隣接性と呼ぶ）を仮定する*．すなわち，(i,j) および (i',j') をそれぞれ “1” の画像をもつ任意のます目の座標とする．$|i-i'|+|j-j'|\leqq 1$ を満足するときに限り，上記二つのます目は互いに隣接しているという．二つの “1” のます目 (i,j)，(i',j') に対して，“1” のます目 $(i_0,j_0),(i_1,j_1),\cdots,(i_n,j_n)$；ただし，$(i,j)=(i_0,j_0)$，$(i',j')=(i_n,j_n)$，が存在し，かつ任意の m，$0\leqq m\leqq n-1$，について (i_m,j_m) と (i_{m+1},j_{m+1}) が隣接しているときに限り，(i,j) および (i',j') は互いに 4-連結（以下ではたんに連結と呼ぶ）であるという．入力画像を S とする．S の任意の “1” のます目を (i,j) とする．(i,j) に連結しているすべて

† An $O(n)$ Connected-Component Labelling Algorithm for Two-Dimensional $n\times n$ Images Based on the Divide-and-Conquer Programming Technique by Hiroshi Umeo (Department of Applied Electronic Engineering, Faculty of Engineering Osaka Electro-Communication University).

†† 大阪電気通信大学工学部応用電子工学科

* したがって，“0” のます目に対しては 8-隣接性，すなわち，上，下，左，右のます目だけでなく斜方向のます目も考慮に入れた隣接関係を仮定している．背景の連結性，連結成分に関しても同様である．

の“1”のます目の集合を S の連結成分と呼ぶ.

隣接性，連結性，連結要素に関する詳細な定義は文献12)等を参照されたい.

連結要素ラベリングとは，与えられた画像の各連結要素に異なったラベルを割りあてる操作で，一つの連結要素を構成するすべてのます目に同一のラベルを割りつける．本稿では，ラベルとして自然数を考え，連結要素の通し番号付けを考える.

次の孤立，非孤立連結要素の概念は，3章におけるラベリング・アルゴリズムの設計に重要な役割を果たす.

[定義] R を画像 P 上の部分領域とする．R 上の任意の連結要素を I とする．I 上のます目が R の外の P 上の“1”のます目と連結しているとき，I を R の非孤立要素と呼ぶ．非孤立でない R 上の連結要素を孤立要素と呼ぶ(**図1**参照).

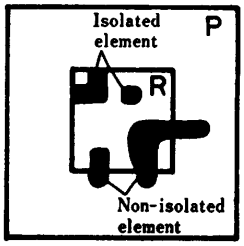

図1 R 上の孤立，非孤立要素
Fig. 1 Isolated and non-isolated components on R.

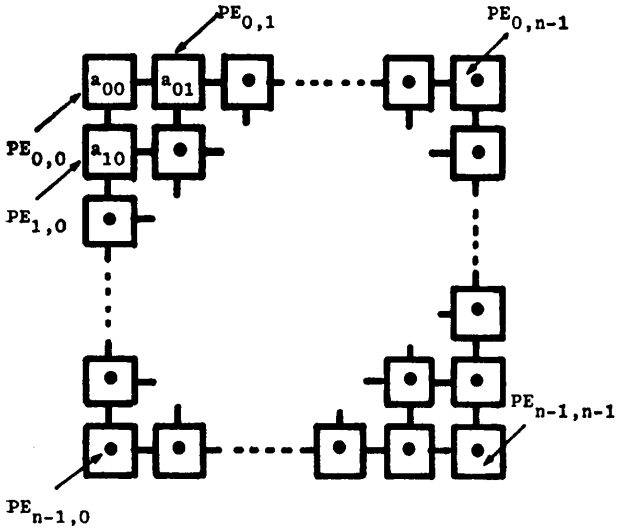

図2 完全並列型・イメージプロセッサ
Fig. 2 An illustration of full-parallel image processor.

2.2 完全並列型・イメージプロセッサ

図2に示す $n\times n$ のプロセッサ（各プロセッサを PE_{ij} で示す．ただし，$0\leqq i, j\leqq n-1$, $n=2^m$, m は自然数）からなる完全並列型・イメージプロセッサ M を考える．M のすべてのプロセッサはまったく同一のもので，$t=0$ 時から同期して動作する.

各プロセッサは，それぞれ1個の入力レジスタ R_{in}，出力レジスタ R_{out}，アドレス・レジスタ R_x, R_y，フラグ・レジスタ R_f ならびに $O(n^2)$ ビットの並列メモリをもつ．ほかに補助記憶レジスタとして R_1, R_2, R_3 をもつ(**図3**)．時刻 t における PE_{ij} 内の R_{in} の内容を $R^t_{in}(i,j)$ で表す．他のレジスタについても同様である．入力画像 $\{a_{ij}\}$ は，$R^0_{in}(i,j)=a_{ij}$ として，1画素/1PE の割合で $t=0$ 時にあらかじめ各プロセッサに与えられているものとする．また R_x, R_y は各プロセッサの x, y 方向の位置情報を記憶するのに使われ，$R^0_x(i,j)=$“i の2進表現”，$R^0_y(i,j)=$“j の2進表現”とセットされているものとする．R_f は M 上で分割統治法（後述）を実行する際，各領域間の同期をとるために使用される．M が停止したとき，各プロセッサの R_{out} にラベリング結果が出力される．各プロセッサは，境界および隅のプロセッサを除き，上，下，左，右の四つのプロセッサと接続されている.

時刻 $t+1$ における PE_{ij} の状態（R_{in}, R_{out}, R_x, R_y, R_f, R_1, R_2, R_3 および並列メモリの内容）s^{t+1}_{ij} は，$s^{t+1}_{ij}=\delta(s^t_{ij}, s^t_{i-1\,j}, s^t_{i+1\,j}, s^t_{i\,j-1}, s^t_{i\,j+1})$ により，上，下，左，右の隣接4プロセッサの局所的な情報のみに依存して決定される．ここに δ は，プロセッサの動作を記述する関数で，すべてのプロセッサに共通である.

3. $O(n)$ 連結要素ラベリングアルゴリズム

本章では，$n\times n$ 画像の連結要素ラベリングを $O(n)$

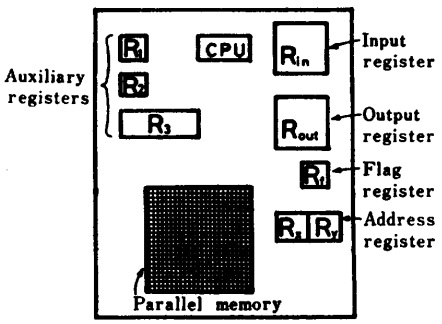

図3 プロセッサの内部構成
Fig. 3 A processing element.

の時間で実行する並列アルゴリズムを提案する．本アルゴリズムの高速性は，完全並列型プロセッサ上での分割統治法の利用による．

3.1 完全並列型プロセッサによる分割統治法の制御アルゴリズム

分割統治法とは，問題をよりサイズの小さな部分に分割し，各部分の解を見いだした後，その部分解を統合して全体の解を得る手法で，従来のノイマン型計算機を対象とした高速アルゴリズム設計に有用なことが知られている[4]．本稿では，完全並列型プロセッサ上で同手法を使用することにより，部分問題ごとに資源（プロセッサ）の割当てが可能となり，かつそれらの解は各部分で同時に得られるのでノイマン型と比較してより高速化が図られる．

次に完全並列型プロセッサ上で分割統治法を制御するアルゴリズムを示す．この制御には，各プロセッサのアドレス情報ならびに2次元セルラオートマトンの同期化手法として知られている一斉射撃アルゴリズムが重要な役割を果たす．一斉射撃問題とは，$n\times n$ のセル空間上で $t=0$ 時に $C_{0,0}$（$C_{0,0}$ とは $(0,0)$ に位置するセルを意味する）にひとりの“将軍”（状態の一つ），他はすべて静止状態と考えられる n^2-1 人の“兵士”からなる初期状況から出発して，ある時刻に初めてセル空間全体が一斉に“射撃状態”になるセルの遷移関数（n に依存しない）を設計する問題で，並列システムにおける処理上の大局的なタイミングと見なされ，従来から数多くの研究がなされている．Beyer[11]により，$2n-2$ ステップ最適時間射撃アルゴリズムが得られている．しかも，わずか8状態の有限オートマトンで実現可能なことが知られている．また，$t=0$ 時にセル空間の4隅，すなわち $C_{0,0}$, $C_{n-1,0}$, $C_{0,n-1}$, $C_{n-1,n-1}$ に位置する将軍により準備された一斉射撃は $n\times n$ の正方形を $n-1$ ステップで射撃可能なことが知られている[3]．遷移関数を少し工夫すれば，それぞれ $2n$，および n ステップ後に一斉射撃をするセルラオートマトンが得られる．また，$2n$ および n ステップのアルゴリズムを組み合わせることにより，任意の自然数 $k(\geq 2)$ に対して kn ステップ後に一斉射撃をするセルラオートマトンも容易に得られる．以上を次の補題にまとめる．

【補題 1】 $k(\geq 2)$ を任意の自然数とする．$t=0$ 時に $C_{0,0}$ に将軍状態をもつ $n\times n$ のセル空間に対し，kn ステップ後に一斉射撃をするセルラオートマトンが存在する．

本稿では，各 PE のフラグ・レジスタ上で一斉射撃を他の処理と並行して実行し，分割統治法により分割された各領域間の大局的な同期をとる目的で使用する．

一般に分割統治法は，データ（問題）の分割および部分解の統合の2過程から成立する．M 上では1データ/1 PE なるデータの割りつけがなされ，しかも各プロセッサはアドレスをもっているため，ある意味でデータはすでに分割されているものと考えることができる．したがって，分割統治法におけるデータの分割過程は以下に示すように不要となる．解の統合は次のようになされる．

$n\times n$ 画像に対し M は部分解を $\log_2 n$ 回統合する．最初 2×2 の画像領域に対して合計 $(n/2)^2$ 個の部分解を並列に得る．次に，隣接している四つの 2×2 領域上の解を統合して $2^2\times 2^2$ 領域上の解を得る．これらの解の統合が全プロセッサ上で並行して行われるため $(n/4)^2$ 個の部分解が同時に得られる．さらに隣接する四つの $2^2\times 2^2$ 領域上の部分解を $2^3\times 2^3$ の領域上に統合，…と以下同様に繰り返す．全体で $\log_2 n$ 回の解の統合過程を繰り返し，$n\times n$ 領域上の部分解が得られた時点で終了する．$n\times n$ 領域上の部分解が実際われわれが求めている解である．

どの領域の部分解をどの領域の部分解と統合するかは，アドレス・レジスタの内容により決められる．R_x, R_y の各桁を下位から順に d_x^i, d_y^i, $i=0,1,2,\cdots,\log_2 n-1$ とする．(d_x^i, d_y^i) として，$(0,0),(0,1),(1,0),(1,1)$ が考えられる．以後これらを順に 0, 1, 2, 3 で表す．第 $i(1\leq i\leq \log_2 n)$ 回目の部分解の統合は，サイズ $2^{i-1}\times 2^{i-1}$ の四つの隣接する領域上でなされる．ここで，四つの統合領域とは (d_x^{i-1}, d_y^{i-1}) の値が相対的に図4に示す位置関係にある領域である．

多数の解の統合は，各領域で同期して全プロセッサ上で一斉に開始される．各領域間の同期はさきに説明した一斉射撃を利用する．サイズ $2^i\times 2^i$ $(i\geq 1)$ の部分解を得るのに要する時間（ステップ数）を $T(2^i)$ で，四つの $2^{i-1}\times 2^{i-1}$ の領域上の解を統合するのに要する時間を $g(2^i)$ で表す．このとき，$T(2^i)=T(2^{i-1})+g(2^i)$ が成立．$g(2^i)$ が 2^i の1次関数であるかぎり，$T(2^i)$ も 2^i の1次関数であることが知られている[4]．連結要素ラベリングでは，$g(2^i)$ が 2^i の1次関数であることが後ほど示される．したがって，ここでは，一般性を失うことなく，$g(2^i)=k\cdot 2^i$（k はある自然数）と仮定できる．解の統合は，$g(2^i)$ ステップご

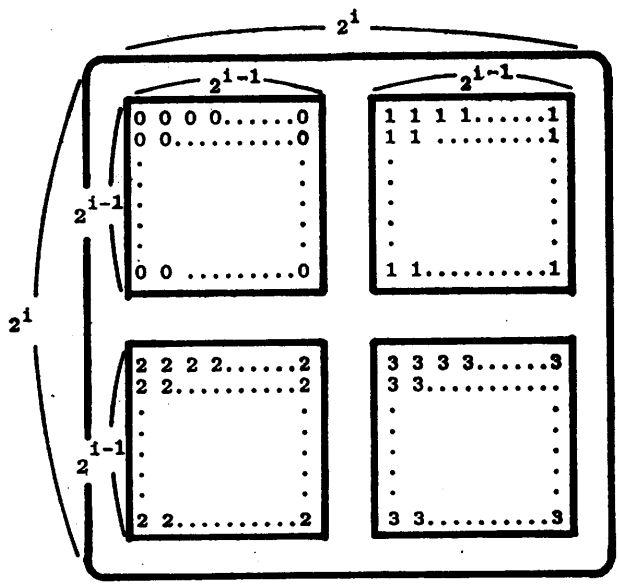

図 4 (四つの) 統合領域のアドレス・レジスタの内容
Fig. 4 Four combined processor regions and the contents of their address registers.

とに発生する一斉射撃を合図として開始される．次に補題としてまとめる．

【補題 2】 M を完全並列型プロセッサ，$g(n)=kn$ とする．ただし，$k(\geqq 2)$ は自然数．このとき，次式で定まる時刻 $t=t_l$ (l は $0\leqq l\leqq \log_2 n$ のすべての整数) 時のみ M 上で一斉射撃をするアルゴリズムがある．ただし，$t_0=2n$, $t_1=t_0+g(2^1)+1$, $t_2=t_1+g(2^2)+1,\cdots$, $t_l=t_{l-1}+g(2^l)+1,\cdots,t_{\log_2 n}=t_{\log_2 n-1}+g(n)+1$.

(証明) 補題1に示した一斉射撃を利用する．$R_t^0(0,0)$="将軍" 状態とする．$t=2n$ 時に，M は最初の一斉射撃をする．この一斉射撃は，次の時刻に $(d_x^0, d_y^0)=(0,0)$ なるプロセッサの R_t を"将軍"状態にセットする．これらの将軍は，2×2 の領域を $g(2^1)$ ステップ後に射撃状態にする．このときすべての 2×2 領域上で一斉射撃が行われる．次の時刻には，$(d_x^1, d_y^1)=(0,0)$ かつ $(d_x^0, d_y^0)=(0,0)$ なるプロセッサの R_t を"将軍"状態にし，これらの将軍は $g(2^2)$ ステップ後に $2^2\times 2^2$ の各領域を一斉射撃状態にする．一般に $l+1$ 回目の一斉射撃は，$0\leqq s\leqq l-1$ なるすべての s に対して $(d_x^s, d_y^s)=(0,0)$ なるプロセッサの R_t 上に将軍を作り，これらの将軍は $2^l\times 2^l$ の領域を一斉に射撃状態にする．アドレスレジスタの最上位桁，すなわち $(d_x^{\log_2 n-1}, d_y^{\log_2 n-1})$ の値が参照されれば，次の一斉射撃の準備とともに最後の一斉射撃であることを全プロセッサに知らせる． ∎

補題1，2より次の定理を得る．

【定理 1】 M を完全並列型イメージ・プロセッサ，隣接する四つの $2^{i-1}\times 2^{i-1}$ の領域上の部分解を統合するのに $k\cdot 2^i$ ステップを要するものとする．ただし $k(\geqq 2)$ はある定まった自然定数，$i=1, 2, \cdots, \log_2 n$. このとき，分割統治法により $O(n)$ の時間で $n\times n$ の解を得る M 上のアルゴリズムがある．

(略証) M は補題2で示した時間間隔で一斉射撃をする．一斉射撃とともに部分解の統合が一斉に開始され，次の一斉射撃までの間に統合過程が終了する．最後の一斉射撃がおこるまで統合過程は繰り返される．$n\times n$ の解を得るのに要する時間 $T(n)$ は次式により $O(n)$ となる．すなわち，

$$\begin{aligned} T(2^{\log_2 n}) &= T(2^{\log_2 n-1})+k\cdot 2^{\log_2 n} \\ &= k(2^{\log_2 n}+2^{\log_2 n-1}+\cdots+2^2)+T(2^2) \\ &= k(n/2-1)+T(2^2) \\ &= O(n) \end{aligned}$$ ∎

SIMD 型並列計算機等大局的通信網をもつ並列計算機では，一斉射撃に相当する信号をホスト計算機から全プロセッサにブロード・キャストすることが可能となり本稿で述べた一斉射撃アルゴリズムは不要である．次の補題は，以下のラベリング・アルゴリズムで使われる．

【補題 3】[6] M を $n\times n$ のセルからなるセルラオートマトン，G を n 個の節点からなる任意の無向グラフとする．M の初期計算状況として，G の隣接行列が与えられたとき，M は $O(n)$ ステップで G の推移的閉包を計算することができる．

3.2 線形時間・連結要素ラベリング・アルゴリズム

次の定理を証明する．

【定理 2】 $n\times n$ の2値画像に対し，$O(n)$ の時間で連結要素のラベリングをする完全並列型アルゴリズムが存在する．

(証明) 本アルゴリズムは定理1に示した分割統治法に基づいている．

$n\times n$ 個のプロセッサからなる完全並列プロセッサ M は，最初 2×2 の領域上の連結要素をラベル付けする．2×2 の領域上に存在する連結要素として，**図5**

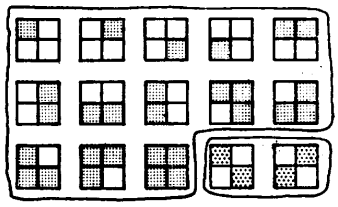

図 5 2×2 の領域上における連結要素
Fig. 5 Connected components in the 2×2 region.

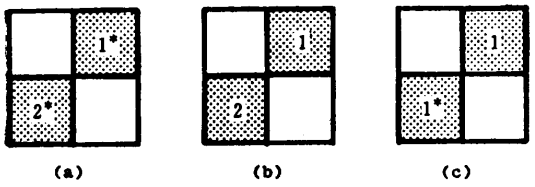

図 6 2×2 の領域上の孤立，非孤立要素に対するラベル付け
Fig. 6 Some typical labellings for isolated and non-isolated components in a 2×2 region.

に示す 15 種類が考えられる．これらの連結要素は，2×2 の領域において孤立要素あるいは非孤立要素のいずれかである．これらの要素に対するラベル付けは次のようになされる．まず，非孤立要素と孤立要素の区別は，＊印の有無によりなされる．またこれらの要素が 1 個の場合，"1" というラベルを付ける．2 個ある場合は，左上のます目を基準として時計方向回りに "1", "2" のラベルを付ける．たとえば，非孤立要素が 2 個の場合，図 6(a) のように，孤立要素が 2 個の場合 (b) のように，非孤立要素と孤立要素がそれぞれ 1 個の場合 (c) のようにラベル付けがなされる．

M は 2×2 領域上のラベル付けを特殊な場合として扱い，各プロセッサは図 5 に示す 15 種類のパターンを記憶しているものとする．したがって，2×2 の領域上のラベル付けは定数時間で完了する．ラベル値は各プロセッサの R_{out} に入れられる．

以後 M は，以下に示す部分解の統合過程を $\log_2 n - 1$ 回繰り返し，全体解を得る．

$2^{i-1}\times 2^{i-1}$ $(i\geqq 2)$ の各領域上の画像に対して，次のような形式でラベル付けがすでになされているものと仮定．

[ラベリング形式]* 領域 R 内に，s 個の孤立要素および t 個の非孤立要素があるものとする．R 上の孤立要素に対しては，"1", "2", …, "s" の連続自然数のラベルが，非孤立要素に対しては，"1*", "2*", …, "t*" の連続自然数に＊印のついたラベルが付与される．

分割統治法の統合過程に基づいて隣接する四つの $2^{i-1}\times 2^{i-1}$ 領域上のラベル付けを統合し，$2^i\times 2^i$ の領域上の上記の形式に基づいたラベル付けを得る方法を示す．本アルゴリズムは次の四つの段階に分けられる．

(A) 各領域上での孤立要素および非孤立要素のラベル情報の収集．

(B) 非孤立要素の連結関係の推移的閉包計算．

* この形式は，図 6 に示した 2×2 の領域上のラベリング方式と一致していることに注意されたい．

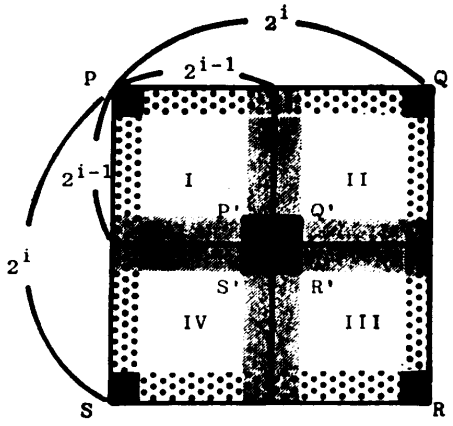

図 7 統合アルゴリズムの説明で使用される記号
Fig. 7 Illustrations of notations used in the proof of theorem 2.

(C) ラベル更新計算．

(D) 再ラベル付け．

以下の説明では，次の記号を使用する (図 7 参照)．四つの領域を左上から時計方向回りに I, II, III, IV 領域と呼ぶ．I, II, III, IV の合併領域を U で表す．U の四隅に位置するプロセッサを P, Q, R, S とし，U の中心に位置するプロセッサを P', Q', R', S' とする．さらに，四つの各領域の周囲に位置するプロセッサのうち，互いに接する部分を内部境界 (Inner Boundary, IB と略す)，接しない部分を外部境界 (Outer Boundary, OB と略す) と呼ぶ．

ステージ (A)：P, Q, R, S の各プロセッサから同時に 2 種類の波 (a 波，b 波とする) が生成される．a 波は各領域を対角線方向に，b 波は境界に沿って進む．a 波は各領域の孤立要素の最大ラベルを，b 波は各領域に存在するすべての非孤立要素の接続関係，すなわち OB および IB を経由して他領域上のどの非孤立要素とつながっているかを調べ，それらに関する情報を U の中心プロセッサ P', Q', R', S' に集める．各領域における a，b 波は，進行方向が違うだけで機能は同一であるので，I におけるそれらの動作を説明する．以下の説明中，$C(R)$ はレジスタ R の内容を意味する．

a 波：a 波は右下り対角線方向にスピード 1 で伝播する．a 波の伝播は各プロセッサの R_1 の状態変化として記述される．a 波の波頭 (wave front) では，いつも各プロセッサの R_{out} の値を読みながら，これま

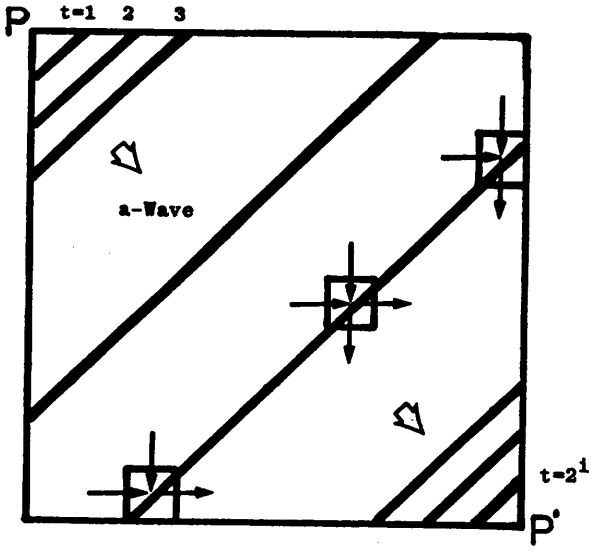

図 8 I 領域における a 波の伝播
Fig. 8 a-Wave propagation in the I region

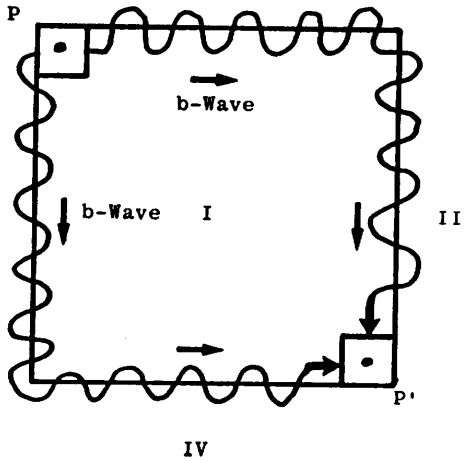

図 9 I 領域における b 波の伝播
Fig. 9 b-Wave propagation in the I region.

でに読んだ孤立要素の最大ラベルを R_1 に記憶し，それを対角線方向のプロセッサ（実際は右および下方に隣接するプロセッサ）の R_1 に伝える．R_{out} の内容が非孤立要素であれば，たんに R_1 の内容を伝達するだけである．2^i ステップ（P, P' 間の距離）後に a 波は P' に到着し，P' の R_1 は I における孤立要素の最大ラベルを保持している（図 8 参照）．

b 波：b 波は P にて 2 個生成される．右および下方向に進む波である．それぞれ OB その後 IB に沿って，スピード 1 で図 9 に示すようにジグザグに進み，I に存在するすべての非孤立要素のラベルを R_2 に貯える．OB および IB 上を進む b 波は，他領域の b 波との区別のためのラベルがつけられているものとする．OB に沿って進む間，I 上の非孤立要素に出会うたびに，その要素（s^* とする）が U の領域外に連結要素をもつか否かを調べる．もつ場合に限り，その要素を U における非孤立要素として新たに＊印を一つ追加した（合計二つの＊印が付加された）ラベルを $C(R_2) := C(R_2) \cup \{s^{**}\}$ により R_2 に記憶する．

IB に沿って進む間，b 波は IB を経由して連結している連結要素のラベル集合ならびに連結関係を R_2, R_3 に次の形で貯える．$C(R_2) := C(R_2) \cup \{s^*\}$, $C(R_3) := C(R_3) \cup \{(s^*, t^*)\}$，ただし，$s^*$ は I 上のある連結要素のラベルで，ラベル t^* をもつ II あるいは IV 上の連結要素と連結しているものとする．b 波は境界上を進行するに従って，次のプロセッサに R_2, R_3 の情報を伝達していく．

二つの b 波は 2^{i+1} ステップ後に P' に到着する．次の 1 ステップで P' は二つの b 波が運んできた R_2 上の情報を合併し，自身の並列メモリ上でラベルの番号によりソートする．2 個以上の同一ラベル番号がある場合は 1 個にする．s^*, s^{**} が同時に含まれる場合は，s^{**} のみを残し，R_3 の内容で s^* に関する部分 s^{**} に書き換える．さらに R_3 に対しても，I，II 間および I，IV 間の連結関係を区別しながら，R_3 上の全要素 $\{(s^x, t^y)\}$，ただし，$x, y \in \{*, **\}$ を s の値の増大順にソートしておく．ソートすべき要素数はたかだか 2^i（すなわち，I 領域の周囲長の半分）個であるので，並列メモリ上では $O(2^i)$ の時間でソーティングは完了する*．

ステージ（B）：（A）が終わった時点で，Q', R', S' の R_2, R_3 上の情報を P' に集め，P' の R_3 の内容を以下に示す隣接行列 $\{m_{ij}\}$ の形で P' の並列メモリ上に設定する．その後 $\{m_{ij}\}$ の推移的閉包を計算する．並列メモリはたかだか $4 \cdot 2^i \times 4 \cdot 2^i$ のセルからなるセルラオートマトンで，全体の行および列は便宜上四つのブロックに分割されているものとする．各ブロック上のすべての行および列には，それぞれ I，II，III，IV における非孤立要素のラベルが行および列の名前として（A）でソートされた順に並べられる．各セル C_{ij} $(1 \leqq i, j \leqq 4 \cdot 2^i)$ には，$\{0, 1^*, 1^{**}\}$ の記号が記入される．$\{0, 1^*, 1^{**}\}$ は次の意味をもつ．いま i 行，j 列のラベルをそれぞれ $l(i)$, $l(j)$ とする．ラベルとして

* たとえば，文献 14)(p. 241) における odd-even transposition sort 等を使用する．

$l(i)$, $l(j)$ をもつ要素がともに U 上の孤立要素でかつ連結しているときに限り m_{ij}="1*", また少なくとも一方が U 上の非孤立要素で連結しているときは m_{ij}="1**" とする．それ以外は m_{ij}="0" とする．以上のような隣接行列の設定を，P' の R_2, R_3 を参照しながら，各行，各列が独立して並列に実行する．したがって，$O(2^i)$ の時間で終了する．

次に補題3の方法でこの隣接行列の推移的閉包を計算する．あるセル上に新しく "1**" 要素が生まれるたびに，その行上を左方向にパルスが伝わり，その行ラベルに "*" を一つ追加する．並列メモリの初期設定ならびに推移的閉包の計算に要する時間は $O(2^i)$ である．

ステージ(C)：Ⅰ，Ⅱ，Ⅲ，Ⅳにおける孤立要素数を p, q, r, s とする．これらの情報は(A)の段階でa波が中心プロセッサ P', Q', R', S' に到着したとき，R_1 に貯えられている．記号 $[i \to j_i]$ $(1 \leqq i \leqq p)$ は，すべての $i(1 \leqq i \leqq p)$ に対しラベル i をラベル j_i に変えることを意味する．

Ⅰ，Ⅱ，Ⅲ，Ⅳの孤立要素に対しては，次のラベルを割りあてる．

Ⅰ：$[i \to i]$ $(1 \leqq i \leqq p)$

Ⅱ：$[i \to i+p]$ $(1 \leqq i \leqq q)$

Ⅲ：$[i \to i+p+q]$ $(1 \leqq i \leqq r)$

Ⅳ：$[i \to i+p+q+r]$ $(1 \leqq i \leqq s)$

複数個の領域にわたる U 上の孤立要素に対しては次のラベルを割りあてる．ただし $t=p+q+r+s+1$ とする．すなわち推移的閉包の計算終了後，行ラベルを上から下に探索し，新しく見いだした孤立要素に対し，t, $t+1$, $t+2$, … というラベルを割りあてる．さらに U における非孤立要素に対しては，上から順に 1*, 2*, 3*, …というラベルを割りあてる．

新しいラベルが決定されると，その情報をもったパルスが行上を右方向に伝わる．パルスは 1* あるいは 1** を見いだすたびに2方向に分かれる．水平方向に進むものと垂直方向に進むものである．上端に達するとその列ラベルに新しいラベル情報を伝える．各列は，最初に到着したラベルを列ラベルとする．新しいラベル値の計算は並列メモリ上でパイプライン的になされるので $O(2^i)$ の時間で終了する．

ステージ(D)：(C)で計算した新しいラベルを列の各ブロック上で，$[x \to y]$ なる形の書換え規則を集め，次にそれらを P', Q', R', S' に分配する．最後に，これらの規則を記憶した波（c波とする）は，a波と反対方向に進み，各セルの R_{out} の内容をその規則に適合するように書き換える．c波は各プロセッサの R_3 上を伝播する．要する時間は $O(2^i)$ である．c波が P, Q, R, S に到着したとき，$2^i \times 2^i$ の領域上のラベリングは終了している．

(A)，(B)，(C)，(D)の4ステージからなる統合過程は，$O(2^i)$ ステップで終了する．したがって，定理1より $n \times n$ 画像のラベリングに要する時間は $O(n)$ である． ∎

4. むすび

完全並列型イメージ・プロセッサ上でサイズ $n \times n$ の2値画像の連結要素ラベリングを $O(n)$ の時間で実行する並列アルゴリズムを提案した．本アルゴリズムの高速性は，完全並列型イメージ・プロセッサ上での分割統治法の利用による．本アルゴリズムと同様な手法により，図形のヒストグラム計算，連結要素の面積計算，核パターンに連結する図形のラベリング等，比較的逐次的な性格をもつ画像処理アルゴリズムを線形時間で実行可能となる．今後は，より少ないハードウェア上で同様な高速アルゴリズムの実現が望まれる．

謝辞 日頃，ご討論いただく本学工学部応用電子工学科浅野哲夫助教授ならびに本研究テーマを再考察する動機を与えていただいた豊橋技科大・山下雅史博士に謝意を表す．

参考文献

1) 木戸出，坂上：パイプライン方式と完全並列処理方式が増えた最近の画像処理装置，日経エレクトロニクス，1982.7.19, pp. 179-212 (1982).
2) 鳥脇，横井：画像処理アルゴリズム，情報処理，Vol. 21, No. 6, pp. 613-619 (1980).
3) 梅尾，菅田：一斉射撃を用いた並列・データ・ルーティング・アルゴリズム，情報処理学会論文誌，Vol. 24, No. 1, pp. 1-7, (1983).
4) Aho, A., Hopcroft, J. E. and Ullman, J. D.: *Data Structures and Algorithms*, Addison-Wesley, Reading (1982).
5) Samet, H.: Connected Component Labelling Using Quadtrees, *JACM*, Vol. 28, No. 3, pp. 487-501 (1981).
6) Van Scoy, F. L.: The Parallel Recognition of Classes of Graphs, *IEEE Trans. Comput.*, Vol. C-29, No. 7, pp. 563-570 (1980).
7) Dyer, R. C.: Parallel Image Processing by Memory-Augmented Cellular Automata, *IEEE Trans. PAMI*, Vol. PAMI-3, No. 1, pp. 29-41 (1981).

8) Kosaraju, S. R.: Fast Parallel Processing Array Algorithms for Some Graph Problems, Proc. 11th ACM Symp. Theory of Computing, pp. 231-236 (1979).
9) Hwang, K. and King-sun, Fu.: Computer Architectures for Image Processing and Database Management, *IEEE Comput.*, Vol. 16, No. 1, pp. 51-60 (1983).
10) Levialdi, S.: On Shrinking Binary Picture Patterns, *Comm. ACM*, Vol. 15, No. 1, pp. 7-10 (1972).
11) Beyer, W. T.: Recognition of Topological Invariants by Iterative Arrays, MAC TR-66, MIT, Cambridge, Mass. (1969).
12) Rosenfeld, A. and Kak, C. A.: Digital *Picture Processing*, 2 nd Ed, Vol. 1, 2, Academic Press, New York (1982).
13) Pavlides, T: *Algorithms for Graphics and Image Processing*, Computer Science Press, Maryland (1982).
14) Knuth, D. E.: *Sorting and Searching*, *The Art of Computer Programming*, Vol. 3, pp. 723, Addison-Wesley, Reading (1973).

(昭和58年7月19日受付)
(昭和58年12月13日採録)