

分割統治法を利用した線形時間・並列画像処理アルゴリズムの設計

A Design of Full-Parallel Linear-Time Image Processing Algorithms
Based on Divide-and-Conquer Programming Technique

梅 尾 博 司

Hiroshi UMEO

大阪電気通信大学・工学部

Osaka Electro-Communication Univ.

Faculty of Engineering

Abstract In this paper we develop an $O(n)$ linear-time connected component labelling algorithm for any $n \times n$ binary images on full-parallel image processors. An application of the well-known divide-and-conquer programming technique to the design of image processing algorithms on full-parallel processors led us to the linear-time algorithm, which was thought to have an area-time complexity.

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持つ並列画像専用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], [13]。本稿では、従来から高速アルゴリズム(ノイマン型)設計法のひとつとして知られている分割統治法(Divide-and-Conquer Method)[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、線形時間を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、連結性の判定などにも数多くの研究[5], [10], [11], [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。

2.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.では、まず完全並列型イメージプロセッサ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 $O(n)$ 線形時間で実行する完全並列型イメージプロセッサアルゴリズムを示す。4.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 $\{a_{ij}\}$ ($a_{ij} \in \{\text{黒}, \text{白}\}$)を入力画像と考える。但し、 $0 \leq i, j \leq n-1$, $n = 2^m$, m は任意の自然数。以下では“黒”を“1”で、“白”を“0”で表す。各画素 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(図1参照)。“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。

本稿では連結性に関しては、4-連結性(以

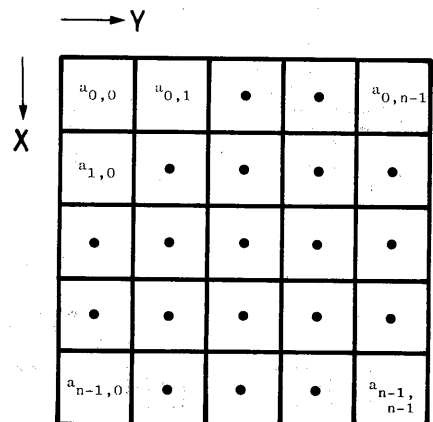


図1. サイズ $n \times n$ の画像

下では単に連結性と呼ぶ)を仮定する。すなわち、 (i,j) および (i',j') をそれぞれ“1”の画素を持つ任意のすす目の座標とする。 $|i-i'|+|j-j'| \leq 1$ を満足する時、上記ふたつのすす目は隣接していると呼ぶ。隣接関係の推移的閉包により、ふたつのすす目間の連結関係を定義する。“1”のすす目からなる集合Pを考える。P上の任意のふたつのすす目が連結している時、Pを連結要素と呼ぶ。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ったラベルを割りあてる操作で、1つの連結要素を構成するすべてのす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(図2参照)を考える。

次の孤立、非孤立連結要素の概念は、3.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の部分画像とする。R上の任意の連結要素をIとする。I上のすす目がRの外のP上の“1”のすす目と連結している時、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3参照。

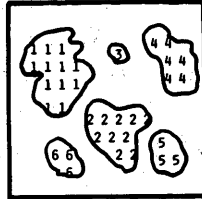
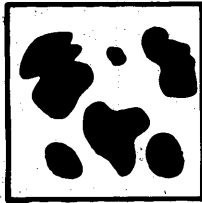


図2. 連結要素ラベリング(通し番号付け)

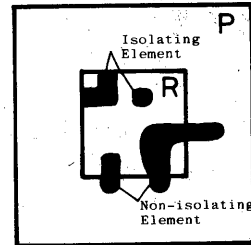


図3. R上の孤立、非孤立要素

2.2 完全並列型・イメージプロセサ

図4に示す $n \times n$ のプロセサ(各プロセサを $PE_{i,j}$, $0 \leq i, j \leq n-1$ で示す)からなる完全並列型・イメージプロセサMを考える。Mのすべてのプロセサは全く同一のもので、 $t=0$ 時から同期して動作する。

各プロセサは、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 $O(m^2)$ ビットの並列メモリを持つ(図5)。

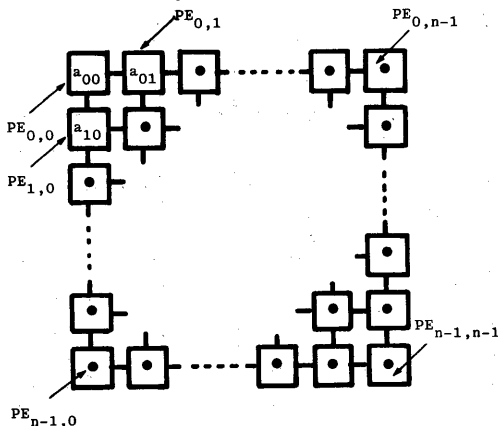


図4. 完全並列型イメージプロセサ

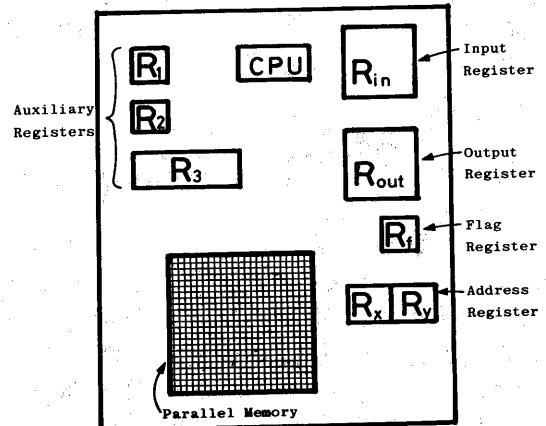


図5. プロセサの内部構成

時刻 t における PE_{ij} 内の R_{in} の内容を $R_{in}^t(i,j)$ で表わす。他のレジスタについても同様である。入力画像 $\{a_{ij}\}$ は、 $R_{in}^0(i,j)=a_{ij}$ として、1画像/1プロセサの割合で $t=0$ 時に予め各プロセサに与えられているものとする。また R_x, R_y は各プロセサの x, y 2方向の位置情報を記憶するのに使われ、 $R_x^0(i,j)=i$ の2進表現、 $R_y^0(i,j)=j$ 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 が停止した時、各プロセサの R_{out} に結果が出力される。各プロセサは、境界および隅のプロセサを除き、上、下、左、右の4つのプロセサと接続されている。

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

3. $O(n)$ 連結要素ラベリングアルゴリズム

本節では、 $n \times n$ 画像の連結要素ラベリングを $O(n)$ の線形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、各部分の解を見出した後、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(シングルプロセサ)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(プロセサ)の割り当てが可能となり、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はかられる。

次に完全並列型プロセ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、各プロセサのアドレス情報ならびに2次元セル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”(状態のひとつ)、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

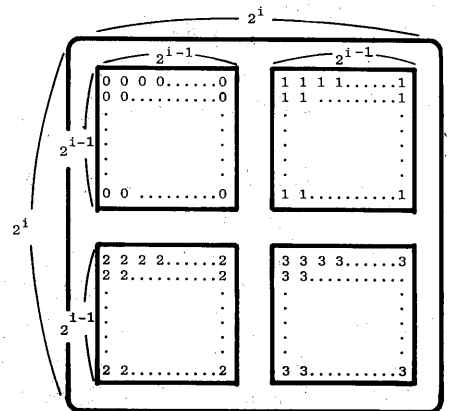


図6. (4つの)結合領域のアドレスレジスタの内容。

その後一斉射撃をするセルラー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合わせることにより、任意の自然数 n (≥ 2)に対して n ステップ後一斉射撃をするセルラー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】 n (≥ 2)を任意の自然数とする。 $t=0$ 時に $C_{0,0}$ に將軍状態を持つ $n \times n$ のセル空間に対し、 n ステップ後一斉射撃をするセルラー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1PE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持っているため、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。従って、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4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列に行なわれるため $(n/4)^2$ 個の部分解が同時に得られる。さらに隣接する4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我々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 $d_x^i, d_y^i, i=0, 1, 2, \dots, \log_2 n$ 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で表わす。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4つの隣接する領域上で行なわれる。ここで、4つの統合領域とは (d_x^{i-1}, d_y^{i-1}) の値が相対的に図6に示す位置関係にある領域である。 $n=16$ の場合における4回の各統合領域を図7に示す。

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

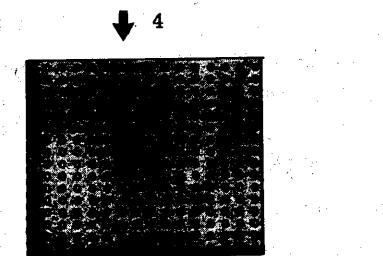
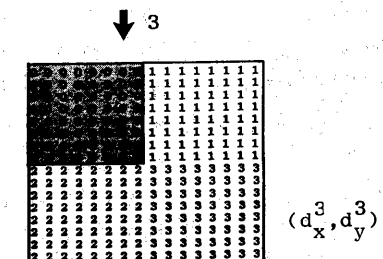
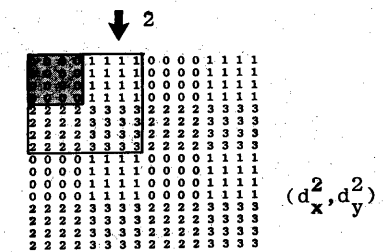
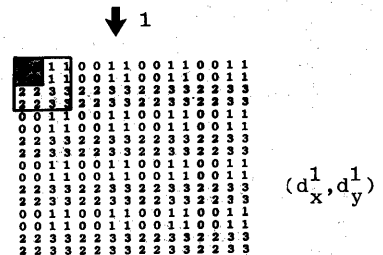
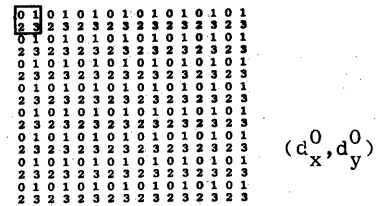


図7. $n=16$ の場合における統合過程。

部分解を得るのに要する時間(ステップ数)を $T(2^i)$ で、4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。この時、 $T(2^i) = T(2^{i-1}) + g(2^i)$ 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]。以下で考察するすべての画像処理アルゴリズムでは、 $g(2^i)$ が 2^i の1次関数であることが後ほど示される。従って、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、 $g(2^i)$ ステップごとにおこなう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サ、 $g(n) = kn$ とする。但し、 $k(\geq 2)$ は自然数。この時、次式で定まる時刻 $t = t_\ell$ ($0 \leq \ell \leq \log_2 n$)時のみ M 上で一斉射撃をするアルゴリズムがある。但し、 $t_0 = 2n$, $t_1 = t_0 + g(2^1) + 1$, $t_2 = t_1 + g(2^2) + 1$, ..., $t_\ell = t_{\ell-1} + g(2^\ell) + 1$, ..., $t_{\log_2 n} = t_{\log_2 n - 1} + g(n) + 1$ 。

(証明) 補題1に示した一斉射撃を利用する。 $R_f^0(0,0)$ ="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 k に $(d_x^0, d_y^0) = (0,0)$ なるプロセサの R_f 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。この時すべての 2×2 領域上で一斉に射撃が行われる。次の時刻 k には、 $(d_x^1, d_y^1) = (0,0)$ かつ $(d_x^0, d_y^0) = (0,0)$ なるプロセサの R_f を"将軍"状態にし、これらの将軍は $g(2^2)$ ステップ後に $2^2 \times 2^2$ の各領域を一斉に射撃状態にする。一般に $\ell+1$ 回目の一斉射撃は、 $0 \leq s \leq \ell-1$ なるすべての s に対して $(d_x^s, d_y^s) = (0,0)$ なるプロセサの R_f 上將軍を作り、これらの将軍は $2^\ell \times 2^\ell$ 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 $(d_{x, \log_2 n - 1}^s, d_{y, \log_2 n - 1}^s)$ 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サ、隣接する4つの $2^{i-1} \times 2^{i-1}$ の領域上の部分解を統合するのに $g(2^i) = k \cdot 2^i$ ステップを要するものとする。但し、 k はある定まった自然定数、 $i=1, 2, \dots, \log_2 n$ 。この時、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なう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、 $T(2^{\log_2 n}) = T(2^{\log_2 n - 1}) + k \cdot 2^{\log_2 n} = k(2^{\log_2 n} + 2^{\log_2 n - 1} + \dots + 2^2) + T(2^2) = k(\frac{1}{2}2^{\log_2 n} - 1) + T(2^2) = O(n)$ 。

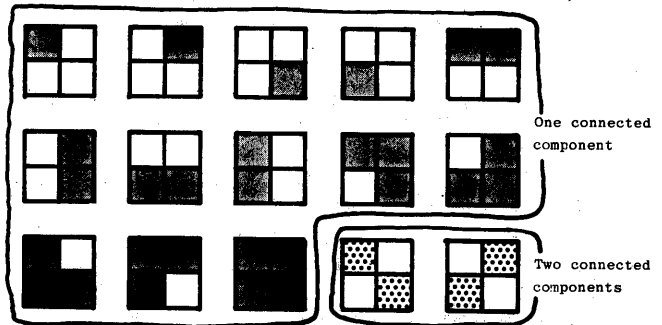


図8. 2×2 の領域上における連結要素

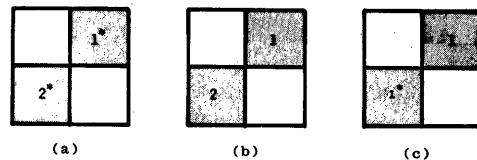


図9. 2×2 の領域上における孤立、非孤立要素に対するラベル付け

SIMD並列計算機等大局的通信網を持つ並列計算機では、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 をサイズ $n \times n$ のセラーオートマトン、 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時、 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(通し番号付け)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。

$n \times n$ ($n = 2^m$) 個のプロセサからなる完全並列型プロセサ 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8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*目を基準として時計方向回りに“1”、“2”のラベルを付ける。例えば、非孤立要素が2個の場合、図9(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、各プロセサは図8に示す15種類のパターンを記憶しているものとする。従って、 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サの $Root$ に入れられる。

以後 M は、以下に示す部分解の統合過程を $\log_2 n - 1$ 回繰り返して、全体解を得る。 $2^{i-1} \times 2^{i-1}$ ($i \geq 2$) 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

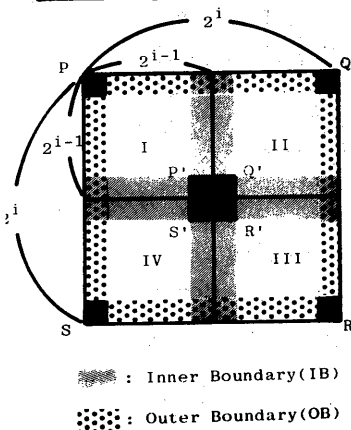


図10. 統合アルゴリズムの説明に使用される記号。

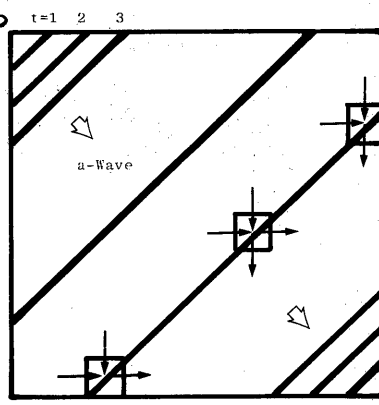


図11. I領域におけるa波の伝播。

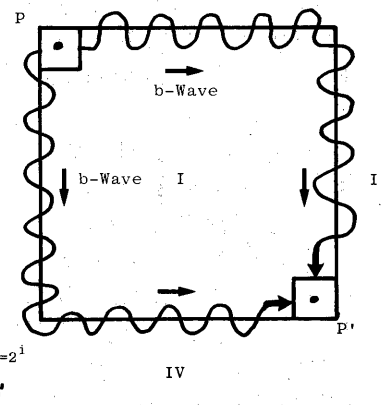


図12. I領域におけるb波の伝播。

るものとする。R上の孤立要素に対しては、 $1, 2, \dots, s$ の連続自然数が、非孤立要素に対しては、 $1^*, 2^*, \dots, t^*$ の連続自然数に*印のついたラベルが付与される。この形式は、図8, 9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。

分割統治法の統合過程に基づいて隣接する4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4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,
- (C) ラベルの更新計算,
- (D) 再ラベル付け.

以下の説明では、便宜上図10に示す記号を使用する。4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で表わす。Uの四隅に位置するプロセサをP, Q, R, S, Uの中心に位置するプロセサをP', Q', R', S'とする(図10)。更に、4つの各領域の周囲に位置するプロセサのうち、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) とそれぞれ呼ぶ(図10参照)。

ステージ(A): P, Q, R, Sの各プロセサから同時に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、進行方向が違った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。

a波: a波は右下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、いつも各プロセサの R_{out} の値を読みながら、これま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サ(実際は右および下方に隣接するプロセサ)の R_1 に伝える。 R_{out} の内容が非孤立要素であれば、単にそ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11参照)。

b波: b波はPにて2個生成される。それぞれOBおよびIBに沿って、図12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、その要素がUの領域外に連結要素を持つか否かを調べる。もし持つ場合は、今度はその要素をUにおける非孤立要素として新たに*印をひとつ追加した(合計2つの*印が付加された)ラベルを 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並みに連結関係を R_2, R_3 に次の形で貯える。 $C(R_2) := C(R_2) \cup \{s^*\}$, $C(R_3) := C(R_2) \cup \{(s^*, t^*)\}$, 但し、 s^* はI上のある連結要素のラベルで、ラベル t^* を持つIIあるいはIV上の連結要素を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、次のプロセサに R_2, R_3 の情報を伝達していく。

2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2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に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^{**} のみを残す。更に R_3 の内容で S^* に関する部分を S^{**} に書き換える。更に 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 $\{S^*, t^*\}$ 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高々 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 $O(2^i)$ の時間を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、 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し、その後 $\{m_{ij}\}$ の推移的閉包を計算する。並列メモリは高々 $4 \cdot 2^i \times 4 \cdot 2^i$ のセルからなるセルオートマトンで、全体が行および列は便宜上4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} ($1 \leq i, j \leq 4 \cdot 2^i$) には、 $\{0, 1^*, 1^{**}\}$ の記号が記入される。 $\{0, 1^*, 1^{**}\}$ は次の意味を持つ。今、 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 $l(i), l(j)$ を持つ要素がともに孤立要素でかつ連結している時に限り $m_{ij} = "1^*"$ 、また少なくとも一方が非孤立要素で連結している時は $m_{ij} = "1^{**}"$ 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。従って $O(2^i)$ の時間で終了する。

次に補題3に示した方法でこの隣接行列の推移的閉包を計算する。新しく $"1^{**}"$ 要素が生ずるたびに、その行上を左方向にパルスが伝わり、その行ラベルに $"x"$ をひとつ追加する。

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 に到着した時、 R_2 に貯えられている。記号 $[i \rightarrow j]$ ($1 \leq i \leq p$)は、ラベル i ($1 \leq i \leq p$)をラベル j 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: $[i \rightarrow i]$ ($1 \leq i \leq p$), II: $[i \rightarrow i+p]$ ($1 \leq i \leq q$),
III: $[i \rightarrow i+p+q]$ ($1 \leq i \leq r$), IV: $[i \rightarrow i+p+q+r]$ ($1 \leq i \leq s$).

複数個の領域がわたる孤立要素に対しては次のラベルを割りあてる。 $t = p + q + r + s + 1$ とする。推移的閉包の計算終了後、行ラベルを上から下に探索し、新しく見出した孤立要素に対し、 $t, t+1, t+2, \dots$ というラベルを割りあてる。更にUにおける非孤立要素に対しては、上から順に $1^*, 2^*, 3^*, \dots$ 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、その情報を持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^{**} を見出すたびに2方向に分かれる。水平方向に進むものと垂直上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。

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、 $[x \rightarrow y]$ 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Routeの内容をその規則に適合するように書換える。C波が P, Q, R, S に到着した時、 $2^i \times 2^i$ 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, $O(2^n)$ ステップで終了する。
従って, 定理 1 より $n \times n$ の画像のラベリングに要する時間は $O(n)$ である。 ■

4. おまけ

完全並列型イメージプロセッサ上で $n \times n$ 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持つ画像処理アルゴリズムを $O(n)$ の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。

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.
- [2] 島脇, 横井; "画像処理アルゴリズム", 情報処理学会会誌 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 24巻, 1号, (1983).
- [4] Aho, A., Hopcroft, J.E., and Ullman, J.D.; "Data Structures and Algorithms", Addison Wesley, (1982).
- [5] Samet, H.; "Connected component labelling using quadrees", JACM, Vol.28, No.3, (1981).
- [6] Van Scoy, F.L.; "The parallel recognition of classes of graphs", IEEE Trans. C-29, No.7, (1980).
- [7] Dyer, R.C.; "Parallel image processing by memory-augmented cellular automata", IEEE, Trans. PAMI-3, No.1, (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.; "Computer architectures for image processing", IEEE Computer, Vol.16, No.1, Jan., (1983).
- [10] Levialdi, S.; "On shrinking binary picture patterns", Comm. ACM, Vol.15, pp.7-10, (1972).
- [11] Beyer, W.T.; "Recognition of topological invariants by iterative arrays", MAC TR-66, MIT, Cambridge, Mass., Oct., (1969).
- [12] Rosenfeld, A and Kak, C.A.; "Digital picture processing", Second Edition, Vol.1, 2, Academic Press, (1982).
- [13] Pavlides, T.; "Algorithms for graphics and image processing", Computer Science Press, (1982).