画像処理における組合せ最適化問題

加藤　直樹

1. はじめに

画像データを局所的な特徴（明るさ，色）が一様な部分画像に分割する手法を，領域分割(region segmentation, image segmentation)とよぶ。領域分割は画像処理における重要かつ難しい問題であり，たとえば航空写真から特定の建物などの対象物の抽出，文字などのパターン認識などに応用される [4, 5]（図1参照)。領域分割問題はかなり古くから研究されており多くの手法が開発されているが [4, 5]，それを最適化問題と考えるとき，最適化問題として定式化して手法の構築を試みているものもあまりみられない。しかし，最近の計算機技術の発達にともなって，計算機の高速化や大容量の記憶装置が手軽に利用できるようになったことにともない，かなり大きな画像データを高速で計算機上で扱えるようになったのにともなって，画像処理に関するアルゴリズムの精密化，高速化がますます重要な課題となっている。

本論文では白黒のディジタル画像に限定し，与えられた画像を二つの領域に分割したい，つまり画像から背景部分を取り除いて対象物を抽出する問題に限定した領域分割問題を組み合わせ最適化問題に定式化し，それを解く手法を大阪電気通信大学浅野哲夫氏，日本IBM徳山豪氏と最近，共同開発したので，その方法について解説をおこなう[3]。ここでは，抽出したい対象物は1つの連結領域からなる場合を扱う。

この手法の共同開発者の浅野哲夫氏のグループによる計算機実験によるとわれわれの開発した手法はかなり良好であるという結果が得られている。領域分割問題を組合せ最適化問題という視点から定式化を試みている研究はわれわれの知る限りほとんどなく，OR技術（最適化手法）の応用可能性の広さ，普遍性をあらためて再認識している。

本論文の構成は以下の通りである。2節では本論文で扱う領域分割問題の定義をおこない，組合せ最適化問題としての定式化を与える。抽出したい領域に対する制約として，“領域が連結である”というだけでは一般にNP困難であることが分かっているので [3] 抽出される領域にさらに制限を加え，抽出領域が一本の x–単調な曲線で区切られている場合（第3節），さらに二本の x–単調な曲線で区切られている場合について考察し（第4節)，これらの場合に対する多項式時間アルゴリズムについて述べる。

2. 定義と問題の定式化

ディジタル画像とは，もとのアナログ的な画像情報を2次元格子（配列）上の各点にその近傍の画像情報を代表化させ，その点にその情報を与えたものである。アナログ画像からディジタル画像を作る方法については成書を見られたい [4, 5]。ディジタル画像の各点はピクセル(画素，pixel)とよばれる。本稿では白黒画像のみを対象にしているので，ピクセルに付与した情報は“明るさ” (brightness level)のみである。ディジタル画像では，明るさも離散データであり，たとえば8ビット程度で表現される。ディジタル画像の大きさは，それを表現する2次元格子（配列）のサイズで表される。本稿では簡単のため正方格子のみを考えることとし，$N\times N$のサイズを持つものとする。入力サイズはしたがって$N^2\times$(各点の明るさの情報量)となるが，各点の明るさが定数オーダのビット長で表現されているものとしてよいので，入力サイズは$O(N^2)$と考えてよい。簡単のため入力サイズを表す記号として$n = N^2$を用いる。すると，入力サイズは$O(n)$である。通常，最低でも400×400程度のディジタル画像は扱う必要があるので，nは相当大きいと考えてよい。いま，Gを$N\times N$の格子平面とする。つまり，

かとう　なおき　神戸商科大学商経学部
〒651-21 神戸市西区学園西町 8-2-1

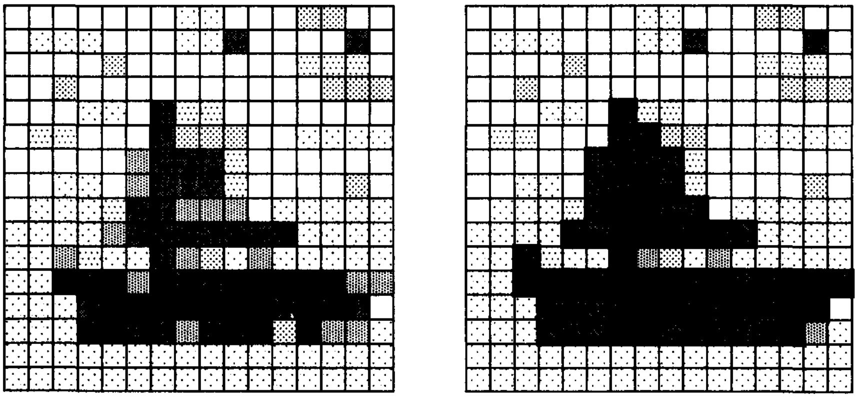

図 1: ディジタル画像とその領域分割

$$G = \{(i,j) \mid i = 1,2,\ldots,N,\ j = 1,2,\ldots,N\}$$

である。各ピクセル (i,j) に付与された“明るさ”を g_{ij} とする。これは非負の整数とする。

領域分割問題は与えられた画像からある意味のある対象物に対応する領域を抽出することであり、“イメージ切り出し”とも呼ばれる。領域分割の結果は人間の目によってのみ評価されるものだが，実験にもとづく経験から，判別分析で用いられる判別関数を最適化する分割が，多くの場合良い分割を与えることが知られている。したがって，取り出す領域に関する制約を何も与えなかったら問題は簡単で，あるしきい値以上（または以下）のピクセルをすべて求めておけばそれが最適解となる。しかし，取り出す領域が“連結”であるという制約を加えると上の最適化基準のもとで連結な一つの領域を背景から取り出す問題は難しくなり，一般に NP 困難であることが示される [3]。したがって，次のようなより単純な場合，つまり取り出したい領域の境界が一本または二本の x–単調な折れ線で定められる場合を考察する。ここで折れ線が x–単調とは、折れ線に沿って移動した時、x 軸方向に関して逆戻りしないことを意味する。言い換えれば，そのような領域は垂直線を任意に引いたとき，その直線と領域の共通部分が連結であるということである。一本または二本の x–単調な折れ線で囲まれた領域は必ずしも連結とは限らないが、本稿では連結領域のみを対象としているので、以下では、一本または二本の x–単調な折れ線で囲まれた連結な領域を“許容領域”と呼び、抽出したい領域を S_0, 背景 (S_0 の補集合) を S_1 と表すことにする。

さて，われわれの用いる目的関数について説明しよう。目的関数は，判別分析で用いられるクラス間分散 (interclass variance) で，その最大化を考える。$\mu = (\sum_{(i,j)\in G} g_{ij})/n$ を画像全体の明るさの平均値とする。n_i, μ_i $(i = 0,1)$ をそれぞれ集合 S_i の大きさ，集合 S_i の明るさの平均値とする。すなわち，$n_0 = |S_0|$, $n_1 = |S_1|$, $\mu_0 = \frac{1}{n_0}\sum_{(i,j)\in S_0} g_{ij}$, $\mu_1 = \frac{1}{n_1}\sum_{(i,j)\in S_1} g_{ij}$ である。このとき，S_0, S_1 の分割によるクラス間分散は

$$V(S_0,S_1) = n_0(\mu-\mu_0)^2 + n_1(\mu-\mu_1)^2$$

で定義される。$V(S_0,S_1)$ の最大化は以下に定義されるクラス内分散 (intraclass variance) $U(S_0,S_1)$ の最小化と等価であることが知られている。

$$U(S_0,S_1) = \sum_{(i,j)\in S_0}(g_{ij}-\mu_0)^2 + \sum_{(i,j)\in S_1}(g_{ij}-\mu_1)^2$$

われわれは実際の応用場面を想定して重み付きクラス間分散 $V^\alpha(S_0,S_1) = n_0^\alpha(\mu-\mu_0)^2 + n_1^\alpha(\mu-\mu_1)^2$ 最大化という目的関数も考えた。この目的関数の特徴は，α がチューニングパラメータとして働き，α を調整することによって抽出したい領域 S_0 のサイズをある程度コントロールできることであり，実用場面で望ましい特徴といえる。しかし本稿では，この目的関数最大化のアルゴリズムの説明は省略する。

3. 領域の境界が一本の x–単調な折れ線で定められる場合

本節で扱う場合というのは，つまり S_0 が画像の上半分もしくは下半分を占めている場合である。以下の

説明ではS_0が上半分を占める場合のみを考えることにする。まずはじめに目的関数$V(S_0,S_1)$はつぎのように書き換えられることに注意しておく。

$$V(S_0,S_1)=n_0n_1(\mu_0-\mu_1)^2/n$$

というのは，

$$nV(S_0,S_1)=n\{n_0(\mu-\mu_0)^2+n_1(\mu-\mu_1)^2\}$$
$$=n\{n_0\mu_0^2+n_1\mu_1^2-n\mu^2\}$$
$$=n_0^2\mu_0^2+n_1^2\mu_1^2+n_0n_1(\mu_0^2+\mu_1^2)-n^2\mu^2 \ (*)$$

より

$n^2\mu^2=(n\mu)^2=(n_0\mu_0+n_1\mu_1)^2$だから，$(*)=n_0n_1\{\mu_0^2+\mu_1^2-2\mu_0\mu_1\}=n_0n_1(\mu_0-\mu_1)^2$
となるからである。

したがって，問題はつぎのように書き換えられる。

$$\text{maximize } D(S_0,S_1)\equiv\sqrt{n_0n_1}|\mu_0-\mu_1|,$$

すなわち，

$$\text{maximize } \sqrt{\frac{n_0}{n_1}}\left|(1+\frac{n_1}{n_0})\sum_{(i,j)\in S_0}g_{ij}-H\right|,$$

ただし，$H=\sum_{(i,j)\in G}g_{i,j}$である。

$P(k)$を$n_0=k$という制約のもとで上の目的関数を最大化する部分問題とする。アルゴリズムは$k=1,2,\ldots,n$のすべてのkに対して部分問題$P(k)$を解き，そこで得られる解のなかで$D(S_0,S_1)$を最大にする解を出力する。各j $(j=1,2,\ldots,N)$に対して，

$$f_j(x_j)=\sum_{i=1}^{x_j}g_{ij},\quad x_j=0,1,\ldots,N$$

とする。$x_j=0$なら上の和は0と考えることにする。$t=n_1/n_0$とおくと，$P(k)$はつぎのように定式化される。

$$P(k):\qquad \text{maximize } \sqrt{1/t}|(1+t)\sum_{j=1}^{N}f_j(x_j)-H|$$
$$\text{subject to } \sum_{j=1}^{N}x_j=k.$$

ここで，x_jは格子平面Gのj列に関する決定変数で，j列では格子点$(1,j),(2,j),\ldots,(x_j,j)$が$S_0$に属し，$(x_j+1,j),(x_j+2,j),\ldots,(N,j)$が$S_1$に属していることを意味している。$P(k)$の最適解を$x=(x_1,x_2,\ldots,x_N)$とすると，$x$は必ずしも連結領域を表現していないことに注意されたい。つまり$j_1<j_2<j_3$に対し，$x_{j_1}>0,x_{j_2}=0,x_{j_3}>0$なら$j_2$列のところで非連結になっているからである。連結性を保証するためにはこのようなことが生じないようような制約条件を加えなくてはならないが，説明を簡単にするため，その点は無視して$P(k)$をどのように解くかについて述べる。(領域が連結であることを動的計画法の定式化でどのように取り扱うかは次節にあるので，そちらを見て頂きたい。)

明らかに$P(k)$を解くにはつぎの二つの問題$P_+(k)$と$P_-(k)$の最適解を求め，大きい方の目的関数値をとる解を選べばよい。

$$P_+(k):\qquad \text{maximize } \sqrt{1/t}((1+t)\sum_{j=1}^{N}f_j(x_j)-H)$$
$$\text{subject to } \sum_{j=1}^{N}x_j=k.$$

$$P_-(k):\qquad \text{maximize } \sqrt{1/t}(H-(1+t)\sum_{j=1}^{N}f_j(x_j))$$
$$\text{subject to } \sum_{j=1}^{N}x_j=k.$$

以下，$P_+(k)$の解法について述べる ($P_-(k)$も同様の方法で解ける)。t,k,Hは定数なので，$P_+(k)$はつぎの問題に帰着される。

$$\text{maximize } \sum_{j=1}^{N}f_j(x_j)\ \text{ subject to } \sum_{j=1}^{N}x_j=k \quad (1)$$

ここまでくると，多くの読者はお気付きのことと思うが，問題(1)は標準的な動的計画法を用いて解くことができる。そのための再帰方程式を作るため次式を定義しておく。

$$F(l,m)=\max[\sum_{j=1}^{m}f_j(x_j)|\sum_{j=1}^{m}x_j=l\ ].$$

この式はS_0が，Gの最初のm列のうちl個の格子点を用いたときの最適解の値であり，$F(k,N)$を与える解が問題(1)の解となっている。明らかに$F(l,m)$はつぎの再帰方程式を満たす。

$$F(l,m)=\max_{x_m=0,\ldots,N}[f_m(x_m)+F(l-x_m,m-1)]$$

この方程式から動的計画法による$F(k,N)$を計算するアルゴリズムが自然に得られる。$m\le N$, $x_m\le N$, and $k \leq n$ なのでこの動的計画法によるアルゴリズムの計算量は $O(N^2 n) = O(n^2)$, 記憶領域は $O(N^2) = O(n)$ である。

補題 1 $O(n^2)$ 時間, $O(n)$ 記憶領域で $F(1,N)$, $F(2,N), \ldots, F(n,N)$ が計算できる。

$D(S_0, S_1)$ を最大にする分割を求めるには, x_mの値を各 $F(l,m)$ の計算時に記憶しておく必要がある。このため $n \times N$の配列 $X(\cdot,\cdot)$ を用意し, $X(l,m)$ にその値を覚えておけばよい(これも動的計画法の標準的技法である)。そうすれば, この配列を用いて最適分割が簡単に得られる。したがってつぎの定理を得る。

定理 1 nピクセルからなる画像に対して, 一本のx−単調な折れ線による最適分割は $O(n^2)$ 時間, $O(n^{1.5})$ 記憶容量で計算できる。

図2には本手法と従来手法とを比較した実験例がある。従来手法より鮮明に領域分割が行われていることが見て取れる。

4. 領域の境界が二本のx−単調な折れ線で定められる場合

本節で扱うケースは定式化は複雑になるが, 基本的に動的計画法を用いて解くことができる。境界が二本の x−単調な折れ線で定められる連結な領域を許容領域と呼び, そのような領域は, 任意の垂直線との共通部分が連結であることで特徴づけられていることを思い出して頂きたい。目的関数は前節と同じであることに注意されたい。各 $j = 1,2,\ldots,N$と各区間 $I \subset [0,N]$ に対して,

$$f_j(I) = \sum_{i \in I} g_{ij}$$

を定義しておく。Iが空であることも許しておく。

$t = n_1/n_0$とおき, $P(k)$ を前節と同様の問題として定義する。本節では $P(k)$ はつぎのように書ける。

$$P(k): \quad \text{maximize } \sqrt{1/t}\,|(1+t)\sum_{j=1}^{N} f_j(I_j) - H|$$

subject to

(i) $\sum_{j=1}^{N} |I_j| = k$

(ii) $1 \leq \exists a < \exists b \leq N$ such that

$I_j = \emptyset$ if $j < a$ or $j > b$,

and $I_j \cap I_{j+1} \neq \emptyset$ if $a \leq j \leq b-1$.

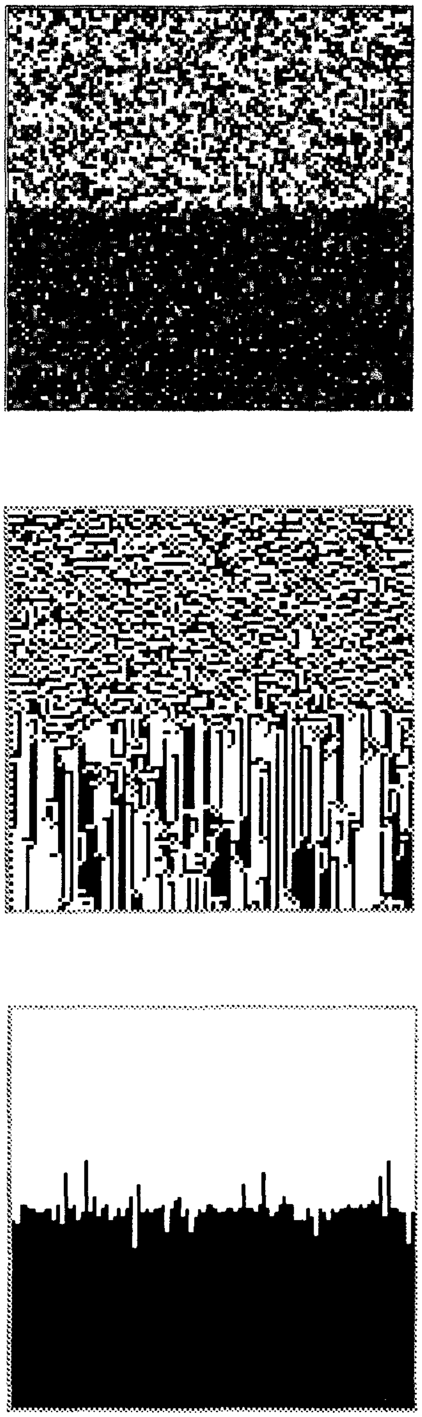

図2：われわれのアルゴリズムの実験例。(一番上の図は入力データのディジタル画像、真中の図は領域拡張法と呼ばれる方法による領域分割の結果、下の図は我々の手法の適用結果を示している。)

ここで，$I_j=[x_j,y_j]$ とすると，x_j,y_jは格子平面のj列に対応する決定変数で，j列における S_0の境界を指し，格子点 $(x_j,j),(x_j+1,j),\cdots,(y_j,j)$ が S_0に属し，その列の他の点は S_1に属していることを意味している。上の制約条件 (ii) は S_0が連結であることを保証している。

4.1 素朴なアルゴリズム

前節と同じように，$P_+(k),P_-(k)$の部分問題に分けて考えるが，ここでは $P_+(k)$をどのように解くかについてのみ説明する。解法は前節と同じように動的計画法である。まずはじめに素直に動的計画法を適用するとどうなるかについて説明する。

まず，1列から m 列の k個のピクセルからなり，m 列が区間 Iであるような許容領域のなかで，その明るさの総和の最大値を $F[k,I,m]$ とする。同様に，1列から m 列の k個のピクセルからなり，m 列の点 (t,m)を含むような許容領域のなかで，その明るさの総和の最大値を $F(k,t,m)$ とする。また，1列から m 列の k個のピクセルからなり，m 列のピクセルは一つも含まないような許容領域のなかで，その明るさの総和の最大値を $F(k,0,m)$ とする。このとき，$P_+(k)$ の最適解は

$$\max_{S_0 \text{ with } |S_0|=k} \sum_{(i,j)\in S_0} g_{ij}$$

の解であり，

$$\max_{S_0 \text{ with } |S_0|=k} \sum_{(i,j)\in S_0} g_{ij} = \max_{t=0,1,..,N} F(k,t,N)$$

pp だから，すべての k,t に対して $F(k,t,N)$ を計算しておけばよい。その計算において，初期条件として

$$F(0,t,m)=F[0,I,m]=0$$

としておく。$k>0$ に対して，$F(k,t,m)$ は $F[k,I,m]$からつぎの式によって計算されることに注意しておく。

$$F(k,t,m)=\max_{t\in I} F[k,I,m]. \qquad (2)$$

するとつぎの再帰方程式を得る。

$$F[k,I,m]=f_m(I)+\max_{l\in I} F(k-|I|,l,m-1) \quad (3)$$

if $I\neq\emptyset$,

$$F[k,\emptyset,m]=\max_{l=0,1,..,N} F(k,l,m-1).$$

ここでは便宜上，$0\in\emptyset$ と仮定している。これより動的計画法のアルゴリズムが自然に得られる。

$O(N^2)$ 個の異なる区間 Iがあり，k,t,m の組み合わせの数は $O(N^4)$ 個であるので (2), (3) を素直に用いると，すべての k,t に対して $F(k,t,N)$ を計算するのに $O(N^7)=O(n^{3.5})$ 時間かかる。「はじめに」でも触れたが，実際の応用では n が大きいので $O(n^{3.5})$ 時間では，いくら多項式オーダといっても時間が掛かりすぎる。以下ではこれを $O(n^{2.5})$ に改良する方法について述べる。

4.2 計算時間の改善

いま，mを固定し，すべての l,t に対して $F(l,t,m-1)$ が計算済みと仮定しよう。このとき，(3) を用いて $F[k,I,m]$ をより速く計算する方法について考えよう。まず，

$$w(k,I,m)=\max_{s\in I} F(k,s,m-1)$$

を定義しておく。すると，(3) 式は $F[k,I,m]=f_m(I)+w(k-|I|,I,m)$ と書ける。したがって，すべての l,q,m に対して $w(q,l,m)$ を計算すればよい。つぎの等式は明かであろう。

$$\begin{aligned} &w(q,[i,j+1],m) \\ &\quad =\max\{w(q,[i,j],m),F(q,j+1,m-1)\} \qquad (4) \end{aligned}$$

つぎに各区間をその区間が始まる添字でグループ分けしておく。i 番目のグループは $\{[i,i],\ [i,i+1],\dots,[i,N]\}$ である。各グループのなかで，(4) 式を用いて各 jに対して $w(q,[i,j],m)$ を jの小さい順に計算する。各 jあたりの計算量は (4) から明らかに定数オーダである。したがって，すべての $k,I(=[i,j]),m$ に対して (3) を計算するのに $O(N^5)=O(n^{2.5})$ で済む。

つぎに (2) 式の計算時間を改良しよう。そのためのキーポイントはこの計算を計算幾何学の問題とみなし，最近注目を集めている行列最大値探索問題に変換しその高速アルゴリズムを利用することである。まず kと m を固定し，各区間 Iに対して，y座標が $F[k,I,m]$, x 軸への射影が Iである水平線分を考える。すると，(2)式の解釈は，“x 座標の各値 $t=1,2,\dots,N$ に対して $y=t$ なる垂直線を引いたとき交わる線分で高さ最大のものを求めよ” という問題になる。そのために行列 M_hを各 $h=1,2,\dots,N$に対して定義する。その (i,j) 要素は $j\geq i\geq h$ なら $F[k,[h,j],m]$，そうでなければ 0 というふうにしておく。

$$
\begin{array}{cc}
 & \begin{array}{cccccc} 1 & & h & h+1 & & N \end{array} \\
M_h = \begin{array}{c} 1 \\ \vdots \\ h-1 \\ h \\ h+1 \\ \vdots \\ N \end{array} &
\begin{pmatrix}
0 & \dots & 0 & 0 & \dots & 0 \\
\vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\
0 & \dots & 0 & 0 & \dots & 0 \\
0 & \dots & F[k,[h,h],m] & F[k,[h,h+1],m] & \dots & F[k,[h,N],m] \\
0 & \dots & 0 & F[k,[h,h+1],m] & \dots & F[k,[h,N],m] \\
\vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\
0 & \dots & 0 & 0 & \dots & F[k,[h,N],m]
\end{pmatrix}
\end{array}
$$

図 3　行列 M_h

行列 M_h がどのような構造になっているかについては図3を参照されたい。このようにして得られた行列 M_h の t 行の最大値を $r_h(t)$ とする。(もしもある行で同じ値の成分が最大値を取る時は、もっとも大きいインデックスを用いる)。すると，

$$\max_{t\in I} F[k,I,m] = \max_{h=1,2,..,N} r_h(t)$$

が成り立つので，高速に $r_h(t)$ を求めることができれば $\max_{t\in I} F[k,I,m]$ は各 t あたり $O(N)$ で計算できる。$r_h(t)$ の高速計算法はつぎのようにして実現できる。

まず，行列 M_h の最初の $h-1$ 行を除く。このようにして得られる横長の行列 (M'_h と書くことにする) は完全単調 (totally monotone) と呼ばれるわれわれにとって実に都合のよい性質がある [1]。行列が 単調 (monotne) というのは，各行の最大値のインデックスは行のインデックスが増えると大きくなる (非減少) というものであり，完全単調というのはこの性質がすべての部分行列に対して成り立つというものである。このような行列のすべての行の最大値が $O(N)$ 時間で求められるということが知られている [1]。その方法について説明しよう。われわれの行列 M'_h が完全単調であるというのは簡単に分かる。簡単のため単調ということだけを示しておこう。M'_h の第1行は M_h の第 h 行であることに注意しておくと，M'_h の第1行目の最大インデックスが $h+s$ なら、第 $s+1$ 行目までの最大値のインデックスはずっとおなじものが続き、第 $s+2$ 行目では、この成分が対角線の左に消えてしまうので、最大インデックスが当然それより右にある、というようにジグザクに変わっていくことから明らかである。

単調という性質を使うと，すべての行の最大値を $O(N\log N)$ 時間で求めることはプライオリティキュー (priority queue) を用いると簡単にできる。はじめに M'_h の第1行 (つまり M_h の第 h 行) の非零要素をプライオリティキューに入れておく。M'_h のすべての非零要素が M'_h の第1行に現れていることに注意すると，上でも説明したように M'_h の第1行目の最大インデックスが $h+s$ なら、第 $s+1$ 行目までの最大値のインデックスはずっとおなじものが続くのでそのあいだ $F[k,[h,h],m],\dots,F[k,[h,h+s-1],m]$ を削除していけばよい。第 $h+s+1$ 行目ではプライオリティキューから $F[k,[h,h+s-1],m]$ を削除し，最大値を求めれば $h+s+1$ 行の最大値が求められる。以下，同じ作業を続けていけばよい。プライオリティキューから要素を削除したり最大値を求めたりするのは $O(\log N)$ でできるので全体の計算時間は $O(N\log N)$ となる。

完全単調という性質を使うと上の計算がさらに高速にできて，$O(N)$ で計算できる。紙面の都合上その方法については省略するが興味ある読者は [1] や [2] を参照されたい。これより前節の素朴なアルゴリズムが $O(n^{2.5})$ に改善できた。

定理 2 n ピクセルからなる画像に対して，二本の x–単調な折れ線による最適分割は $O(n^{2.5})$ 時間で計算できる。

5.　おわりに

本稿では画像の領域分割という問題に対して領域の境界を定める折れ線が x–単調という場合に限って多項式時間アルゴリズムがどのように構成できるかを見てきた。アルゴリズムで用いる解法は標準的なテクニックであり実現も容易である。このアルゴリズムがどの程度実用に供するものかをみるための予備実験を浅野氏のグループが企業と共同ですでに行っており，その結果は良好であるという報告を受けている。

また，本稿では触れることができなかったが，前節

のアルゴリズムはさらに $O(n^2)$ 時間、$O(n)$ 記憶領域まで改良が進み，$O(\epsilon^{-1} n \log H)$ 時間のϵ–近似アルゴリズムも得ている（ここでHは画像の明るさレベルの総和である）。

今後の課題としては取り出したい画像の境界がx–単調でない場合についてのよい近似アルゴリズムの開発を行うことである。

参考文献

1. A. Aggarwal, M.M. Klawe, S. Moran, P.W. Shor, and R. Wilber: "Geometric applications of a matrix-searching algorithm," *Algorithmica*, Vol.2, pp.195-208, 1987.

2. 浅野哲夫, 計算幾何学, 朝倉書店, 1990.

3. T. Asano, D. Chen, N. Katoh, and T. Tokuyama, Polynomial Solutions to Image Segmentation, 情報処理学会アルゴリズム研究会資料, 95-AL-45, 1995 年 5 月.

4. A. Rosenfeld and A.C. Kak, ディジタル画像処理, (長尾真 監訳), 近代科学社, 1978.

5. 高木幹雄, 下田陽久 監修, 画像解析ハンドブック, 東京大学出版会, 1991.