

画像切り出しに対するアルゴリズムの提案

安齋 進也* 全 眞嬉† 葛西 亮生‡ コルマン マティアス§ 徳山 豪¶

1 はじめに

画像切り出しとは、与えられた画像からイメージを切り出す作業である。これは画像処理における重要な問題の 1 つであり、古くから注目を集めており、これまでに様々な手法が提案されてきた。この問題に対し、Asano ら [1] は組み合わせ最適化問題として定式化し、パラメトリック最適化の技法を利用して解決する枠組みを提案した。この手法によると、本論文で扱う最大重み領域問題を効率的に解くことが鍵となる。最大重み領域問題は前述の通り、画像処理分野において重要な問題であるが、他にもデータマイニング [5, 6] や放射線医療の最適化 [4] にも応用されている。

最大重み領域問題は以下のように定式化される。 $n \times n$ のピクセル平面 P と、ピクセル領域と呼ばれるピクセル集合の族 $\mathcal{F} \subseteq 2^P$ を考える。ピクセルとは、単位正方形 $p(i, j) = [i - 1, i] \times [j - 1, j]$ のことである。ここで、 $1 \leq i, j \leq n$ とする。ピクセル座標 (i, j) に位置するピクセルは実数値 $w(p) = w(i, j)$ を持ち、これを p の重みと呼ぶ。重みには負の値も存在する。従って、ピクセル平面 P に対する重みを表す行列 $W = (w_{i,j})$ は実数値行列で表される。便宜上、 $1 \leq i, j \leq n$ に対して、 $w(0, j) = w(n + 1, j) = w(i, 0) = w(i, n + 1) = 0$ とする。最大重み領域問題とは次のように表される。

最大重み領域問題

入力 $n \times n$ の重み行列 W

出力 $W(R) = \sum_{p \in R} w(p)$ を最大化する

領域 $R \in \mathcal{F}$

この問題の難しさはピクセル集合の領域族 \mathcal{F} に依存する。 $\mathcal{F} = 2^P$ の場合、この問題の解は自明であり、最適なピクセル領域は重みが正であるピクセルの集合として表される。一方で、 \mathcal{F} が P 内において、4 近傍連結であるような領域全体の場合、Asano ら [1] により NP 困難であることが示されている。このように、最大重み領域問題の難しさは領域族 \mathcal{F} に依存する。しかしながら、ピクセル平面 P のサイズ $n \times n = N$ に対して、多項式で解ける領域族も多く存在する。さらに、Chun らはより広い領域族を考え、それらの領域族に対する効率的なアルゴリズムを与えた [2]。具体的には、既にピクセル平面 P のサイズ N に対して多項式で解くことができる領域を基本図形とする。そして、その図形の非交差和領域を領域族として最大重み領域問題を N の多項式で解いている。

本論文では、Chun らにより多項式で解けることが示されている階段凸領域 [3] を基本図形とし、制限をつけた非交差和領域に対して最大重み領域問題を解くことを考える。階段凸領域とは、中心点 p を含む長方形の和集合領域で表される図形のことである。本文中では、表 1.1 に示す計算量を持つ FPT アルゴリズムについて述べる。FPT(Fixed Parameter Tractable) とは、計算量クラスの 1 つであり、計算量が $O(f(k)N^c)$ (c は定数) で表されるアルゴリズムを持つ問題のクラスである。

*東北大学大学院情報科学研究科

†第 1 著者に同じ

‡第 1 著者に同じ

§ブリュッセル自由大学

¶第 1 著者に同じ

表 1.1. アルゴリズムの計算量

制限	計算量
4 分の 1	$O(k!N^2)$
2 分の 1	$O(k^{O(k^2)}N^{1.5})$

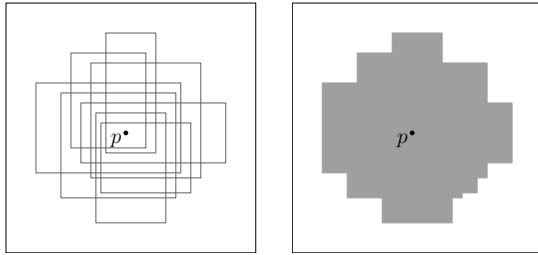


図 2.1. 階段凸領域

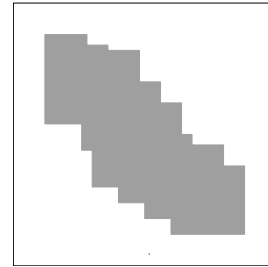


図 2.2. 直交凸領域

2 基本図形

本章では、アルゴリズムを提案する領域族における基本図形について述べる。

2.1 階段凸領域

本節では、階段凸領域について述べる。

階段凸領域 (図 2.1 参照) とは、中心点となる格子点 p が与えられたとき、次のように定義される領域のことである。

定義 2.1. 格子点 p を中心とする階段凸領域とは、 p を内部に含む長方形の和集合として表すことができる領域のことである。

1 つの階段凸領域に対し、最適領域を求めるのに $O(N)$ 時間かかることが Chun らにより示されている [3]。この手法はまず、格子点 p を通るように水平線と垂直線を引き、階段凸領域を 4 つの部分領域に分割する。その後、それぞれの部分領域に対し、動的計画法を用いて最適領域を構成することで、全体の最適領域を求めている。

2.2 直交凸領域

本節では、直交凸領域について述べる。今回、基本図形として用いられていないが、アルゴリズムを述べる上で重要となる図形である。

直交凸領域 (図 2.2 参照) とは、次のように定義される領域のことである。

定義 2.2. 直交凸領域とは、 x 単調かつ y 単調である領域のことである。

直交凸領域の性質を考えると、次の 4 つの基本となる図形が考えられる。

- 直交凸領域の上側が単調増加で、下側が単調減少 (タイプ W)
- 直交凸領域の上側も下側も単調増加 (タイプ U)
- 直交凸領域の上側も下側も単調減少 (タイプ D)
- 直交凸領域の上側が単調減少で、下側が単調増加 (タイプ N)

直交凸領域は、これらの特定の組み合わせにより表すことができる。1 つの直交凸領域に対する最適領域を求めるには、Yoda らにより動的計画法を用いて $O(N^{1.5})$ 時間かかることが既に示されている [7]。

3 k 個の中心点を持つ階段凸領域

本章では、前章で基本図形として述べた階段凸領域の中心点を 1 個ではなく、複数個に拡張した場合について述べる。

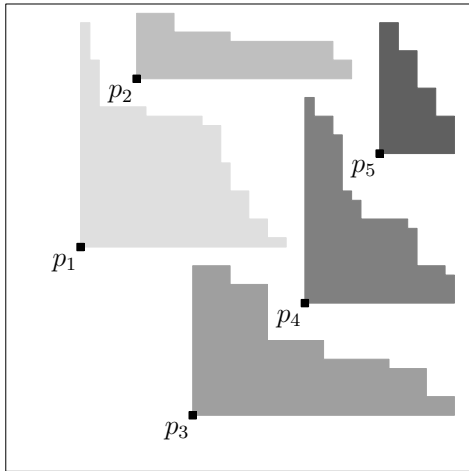


図 3.1. 階段凸領域を 4 分の 1 に制限した問題

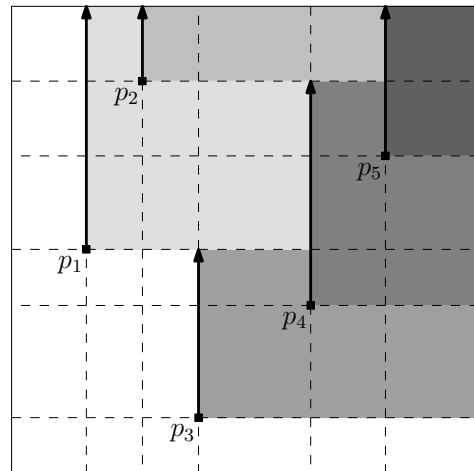


図 3.2. ピクセル平面の分割

3.1 領域を 4 分の 1 に制限した場合

本節では、階段凸領域を 4 分の 1 に制限した場合の問題について考える。すなわち、図 3.1 のような中心点からの領域が右上のみの階段凸領域について問題を考える。

アルゴリズムは以下で述べるものである。前処理として、各中心点から水平線と垂直線を引き、グリッドをセルに分割する。このとき、グリッドは高々 $(k+1)^2$ 個のセルに分割される。分割したセルに対し、下から第 1 行目、第 2 行目、 \dots 、第 $k+1$ 行目とし、左から第 1 列目、第 2 列目、 \dots 、第 $k+1$ 列目とする。また、第 i 行、第 j 列目にあるセルを $c_{i,j}$ とする。問題の許容解とセルの関係に注目すると以下の補題が成り立つ。

補題 3.1. あるセル c には、 c よりも左下にある中心点からの領域のうち、1 つしか入り込めない。

以上のことから、各セルに対して、どの中心点からの領域が入り込むか割り当てて考える。

ステップ 1 (セルへの番号の割り当て)

各中心点からの領域がどの高さ h_{p_i} まで入り込むかを割り当てて。次に、一番右にある点から順に、番号をセルに割り当てていく。中心点 p_i のすぐ上にあるセルの行を l_1 とし、すぐ右にあるセルの列を l_2 とす

る。セル $c_{i,j} (l_1 \leq i \leq h_{p_i}, l_2 \leq j \leq k+1)$ に中心点 p_i の番号を割り当てる。ただし、すでに番号が割り当てられている場合は、割り当てないものとする。

ステップ 2 (ピクセル平面の分割)

同じ番号が割り当てられたセルの集合を 1 つの領域とし、図 3.2 のようにピクセル平面を分割する。このような割り当ての総数は $O(k!)$ である。

ステップ 3 (最適領域の計算)

他の中心点からの領域が入り込むことがないため、分割された領域ごとに最適解を求めることで、全体の最適解を求めることができる。

従って、1 つの割り当てに対し、 $O(N)$ で求めることができるため、全体の計算時間は $O(k!N)$ となる。ゆえに、この問題は FPT であることがわかる。

3.2 領域を 2 分の 1 に制限した場合

本節では、階段凸領域を 2 分の 1 に制限した場合の問題について考える。すなわち、図 3.3 のような中心点の右側のみの階段凸領域について問題を考える。

前節と同様に、グリッドをセルに分割し、各中心点ごとに最適解を構成する手法では、セルに 2 つの中心点からの領域が入り込むことがあるため、領域が

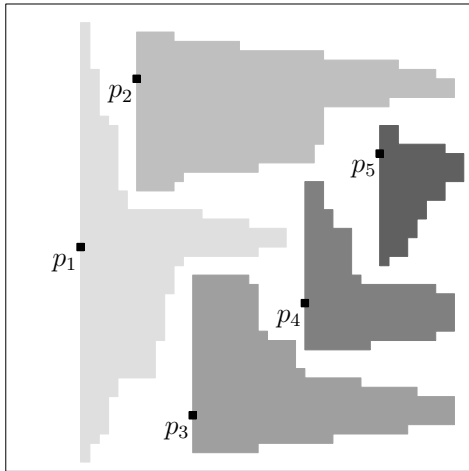


図 3.3. 階段凸領域を 2 分の 1 に制限した問題

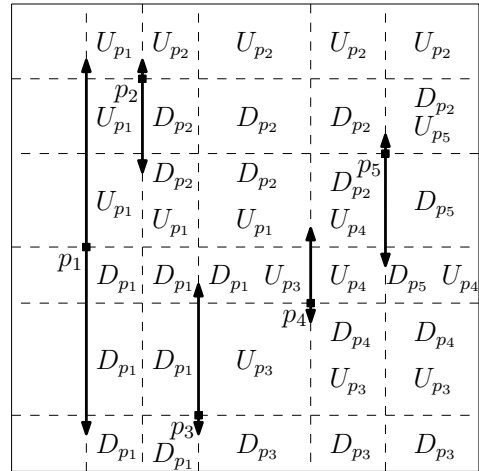


図 3.4. セルへのラベル付け

交差する場合がある。この手法のままでは、出力領域が問題の定義に反する場合がある。そこで、最大重み領域を直接求めるのではなく、その補集合領域を用いて求めることを考える。つまり、ピクセル平面の重みの符号を反転させたピクセル平面に対して、補集合領域に関する最大重み領域問題を解く。

アルゴリズムは以下で述べるものである。前処理として、前節と同様にグリッドをセルに分割する。

ステップ 1 (セルへの番号の割り当て)

各中心点からの領域がどの高さまで入り込むのか上下について割り当てる。ここで、中心点 p_i の上側にあり、入り込む可能性があるセルにラベル U_{p_i} を付け、下側についてはラベル D_{p_i} を付ける。また、1つのセルに2つの中心点からの領域が入り込む可能性があるため、ラベルが2つ付いても良い(図 3.4 参照)。

ステップ 2 (ピクセル平面の分割)

列ごとにセルをみていき、同じラベルを持つセルの集合を1つの領域 R_i とする。もし、2つのラベルが存在する場合は、どちらか一方が同じであれば同じ集合の要素とする。

ステップ 3 (有向森の作成)

セル $c_{i,j}$ が持つラベルと、セル $c_{i-1,j}$ が持つラベルが少なくとも1つ一致しているならば、セル $c_{i,j}$ を含む領域からセル $c_{i-1,j}$ を含む領域へ有向辺を付け

る。ただし、すでに点同士で向辺が存在する場合は、辺を新たに付け加えることはしないものとする(図 3.5 参照)。

ステップ 4 (最適領域の計算)

ピクセル平面上で右から左へと、動的計画法を用いて計算する。1つの領域 R_i に注目すると、中心点からの領域の入り込み方として、

1. 領域 R_i の上にある中心点 p_{l_1} からのみ入り込む場合
2. 領域 R_i の下にある中心点 p_{l_2} からのみ入り込む場合
3. 領域 R_i の上下2つの中心点から入り込む場合

の3通りある。領域 R_i に入り込むある中心点 p_{l_1} からの領域は右から左へ見ると、領域 R_i 内では単調減少になっている。同じように下にある中心点 p_{l_2} からの領域は、単調増加になっている。これらのことから、領域 R_i の補集合領域 R_i^c は直交凸領域のタイプ N になっていることがわかる。つまり、第 m 列において第 s 行から第 t 行までのピクセルを取るときの最適値 $f([s, t], m)$ の計算は直交凸領域で述べた漸化式で行うことができる。次に、有向辺が存在する領域間の計算について考える。領域間の関係として、

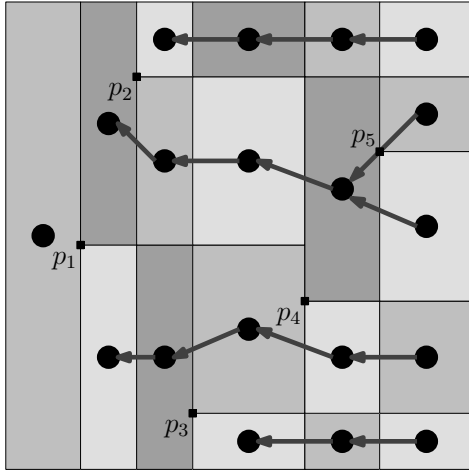


図 3.5. ラベルに対する有向森

1. 領域 R_i と領域 R_j の両方にラベル $D_{p_{i_1}}$ が存在
2. 領域 R_i と領域 R_j の両方にラベル $U_{p_{i_2}}$ が存在
3. 領域 R_i と領域 R_j の両方に 2 つのラベル $D_{p_{i_1}}$ と $U_{p_{i_2}}$ が存在

の 3 通りが考えられる。ここで、1. のようにラベル $D_{p_{i_1}}$ が一致しているとき、それまで計算してきた結果を $f_1([s, t], m)$ とおく。また、2. のようにラベル $U_{p_{i_2}}$ が一致しているときは $f_2([s, t], m)$ とおく。1. の場合 (図 3.6 左参照) は以下のような漸化式で表すことができる。

$$\begin{aligned}
 & f([s, t], m) \\
 &= \max_{t_1 \geq t} \{f_1([s_1, t_1], m - 1)\} + w([s, t], m) \\
 &= \max \left\{ \begin{array}{l} f([s, t + 1], m) - w(t + 1, m), \\ f_1([s_1, t], m - 1) + w([s, t], m) \end{array} \right\}
 \end{aligned}$$

2. の場合 (図 3.6 真ん中参照) は、

$$\begin{aligned}
 & f([s, t], m) \\
 &= \max_{s_2 \leq s} \{f_2([s_2, t_2], m - 1)\} + w([s, t], m) \\
 &= \max \left\{ \begin{array}{l} f([s - 1, t], m) - w(s - 1, t), \\ f_2([s, t_2], m - 1) + w([s, t], m) \end{array} \right\}
 \end{aligned}$$

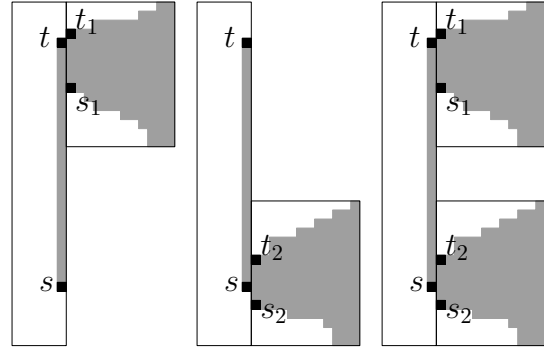


図 3.6. セルの結合部分に関する漸化式

と表すことができる。3. の場合 (図 3.6 右参照) は、

$$\begin{aligned}
 & f([s, t], m) \\
 &= \max_{t_1 \geq t} \{f_1([s_1, t_1], m - 1)\} \\
 & \quad + \max_{s_2 \leq s} \{f_2([s_2, t_2], m - 1)\} + w([s, t], m) \\
 &= \max \left\{ \begin{array}{l} f([s, t + 1], m) - w(t + 1, m), \\ f([s - 1, t], m) - w(s - 1, m), \\ f_1([s_1, t], m - 1) \\ + f_2([s, t_2], m - 1) + w([s, t], m) \end{array} \right\}
 \end{aligned}$$

となる。ここで、 s_1, t_2 は t_1, s_2 が決定したときに $f_1([s_1, t_1], m), f_2([s_2, t_2], m)$ を最大にする添字である。漸化式から、3 変数が決定したときにかかる計算時間は定数であることがわかる。つまり、領域間の計算時間は直交凸領域のタイプ N の計算時間と同様に $O(N^{1.5})$ となる。また、有向森を作る割り当ては $O(k^{O(k^2)})$ であるから、全体の計算時間は $O(k^{O(k^2)} N^{1.5})$ となる。

以上のことから、この問題も前節と同じように FPT であることがわかる。

4 結論と今後の課題

本論文では、 k 個の中心点が与えられたとき、制限付きの階段凸領域の非交差和領域に対する最大重み領域問題が FPT であることを示した。画像切り出しに関する研究において、 k 個の基本図形に分割する研

究はまだまだ発展途上である。そのため、本論文の結果は興味深いものであると言える。

応用面においては、画像の「形」に注目したデータマイニングツール (Shape Detector) の開発に用いることができる。これを、他のデータマイニングツールと組み合わせることで、高精度な画像データマイニングが可能となる。

制限のない階段凸領域の非交差和領域に対する最大重み領域問題を解くアルゴリズムは未解決であり、今後の課題である。

参考文献

- [1] T. Asano, D. Z. Chen, N. Katoh, and T. Tokuyama. Efficient algorithms for optimization-based image segmentation. *Int. J. Comput. Geometry Appl.*, 11(2):145–166, 2001.
- [2] J. Chun, R. Kasai, M. Korman, and T. Tokuyama. Algorithms for computing the maximum weight region decomposable into elementary shapes. In *ISAAC*, pages 1166–1174, 2009.
- [3] J. Chun, K. Sadakane, and T. Tokuyama. Efficient algorithms for constructing a pyramid from a terrain. In *JCDCC*, pages 108–117, 2002.
- [4] C. Engelbeen and S. Fiorini. Constrained decompositions of integer matrices and their applications to intensity modulated radiation therapy. In *CTW*, pages 177–180, 2008.
- [5] T. Fukuda, Y. Morimoto, S. Morishita, and T. Tokuyama. Data mining using two-dimensional optimized association rules: Scheme, algorithms, and visualization. In *SIGMOD Conference*, pages 13–23, 1996.
- [6] T. Fukuda, Y. Morimoto, S. Morishita, and T. Tokuyama. Data mining with optimized two-dimensional association rules. *ACM Trans. Database Syst.*, 26(2):179–213, 2001.
- [7] K. Yoda, T. Fukuda, Y. Morimoto, S. Morishita, and T. Tokuyama. Computing optimized rectilinear regions for association rules. In *KDD*, pages 96–103, 1997.