アルゴリズム 80−10
（2001．9．25）

最適二分探索木を与える領域と回転操作及び三角形分割

大西 建輔，星 守

電気通信大学 大学院 情報システム学研究科
〒 182-8585 東京都調布市調布ヶ丘 1-5-1
E-mail: {onishi, hoshi}@hol.is.uec.ac.jp

あらまし: 確率分布を与えた場合に路長の期待値を最適とする二分探索木, 最適木と呼ぶ, が二分木の内点数の二乗で計算できることは, Kunth によりすでに証明がなされている. 前回の発表では、確率分布のパラメタ空間をそれぞれの最適木に対応する領域に分割する手法について述べた.

本稿では、どのような二分探索木でも, 最適木となるりうるのか? これらの領域がどのような場合に接するのか?という問題に対して, 二分探索木の回転操作を用いた条件を与える.

Regions of optimal binary search tree, roation and triangulation

Kensuke Onishi, Mamoru Hoshi

Graduate School of Information Systems, University of Electro-Communications,
1-5-1 Chofugaoka, Chofu-shi, Tokyo, 182-8585, Japan
E-mail: {onishi, hoshi}@hol.is.uec.ac.jp

Abstract: For a given probabilistic distribution, a binary search tree with optimal path length among all trees is called *optimal tree*. Knuth showed that such a tree is computed in the time of square of the number of nodes of binary search tree. In our previous paper, we showed algorithms that the parametric space of discrete probabilistic distribution is divided into regions corresponding with binary search tree.

In this paper, we investigate that any binary search tree can become optimal tree and which two optimal regions are adjacent. Some conditions related with single rotation is showed.

1 始めに

二分探索木は, 語とその語の検索確率が存在する辞書のモデルやデータベースでのオブジェクトの索引の生成のモデルにも適用可能なデータ構造である. その研究は, 長年行われている. 例えば、内点までの路長の総和を最小化した二分木 (以下、最適木と呼ぶ) を計算するアルゴリズムが Knuth [2, pp.436–442] に紹介されている. このアルゴリズムは、二分木探索木の内点数を n とすると $O(n^2)$ 時間で最適木を計算することが可能である.

n 内点からなる二分探索木を考える. 内点の集合 $A = \{a_1, a_2, \ldots, a_n\}$ には, 全順序があり, その順に添字が付けられているとする. それぞれの内点には, 検索確率 p_i が付加されていると考える. このとき, 二分探索木 T の路長の期待値は, 根から a_i までの路長 $l(a_i)$ とその検索確率 p_i を用いて,

$$E(T) := \sum_{i=1}^{n} l(a_i) \cdot p_i$$

と表現される (以下, 単に路長和と呼ぶ). この値は, 二分探索木 T と確率 $\{p_i\}$ により決定される値であり, 各二分探索木に対し, 計算が可能である. また, 内点数を固定した二分木の個数は, 有限値 (Cataran 数) であるため, 少なくとも 1 つの最適値が存在し, その最適値を与える最適二分探索木 (以下, 最適木と呼ぶ) が存在する.

路長和 $E(T)$ は, 根から各内点までの路長 $\boldsymbol{l} = (l(a_1), l(a_2), \ldots, l(a_n))$ と各内点の検索確率 $\boldsymbol{p} = (p_1, p_2, \ldots, p_n)$ の関数とみなすことができる.

$$f(\boldsymbol{l}, \boldsymbol{p}) := E(T)$$

と定義すると, $2n$ 個の変数を持つ関数の最適化問題であることがわかる. Knuth [2] は, 与えられた $\boldsymbol{p}$ に対して, 最適となる二分木 T を計算するアルゴリズム, すなわち,

$$\boldsymbol{l}_{opt} = \arg \min_{\forall T,\ \boldsymbol{p}:\mathrm{fix}} \{f(\boldsymbol{l}, \boldsymbol{p})\}$$

の計算を $O(n^2)$ 時間で計算するためのアルゴリズムを提案した.

大西らは, [5] において, 検索確率の空間 (n 次元離散パラメタ空間) $\mathcal{R} = [0,1]^n$ をそれぞれの最適木に対応する領域に分割することを考え, そのアルゴリズムを提案した. 言い替えると、最適化関数 $f(\boldsymbol{l}, \boldsymbol{p})$ の変数 $\boldsymbol{p}$ を動かし, それぞれの二分木が最適となる領域の計算を行うアルゴリズムを提案した. このアルゴリズムは, 1. すべての二分木を生成する; 2. 二分木に対し, 各内点の路長を決定する; 3. 最適領域分割の生成を行う. という 3 段階がある.

本稿では, まず, 二分木を生成すると同時に, その二分木の路長を決定していくアルゴリズムを示す (2 節). また, 最適領域の性質として, 次の 2 つを示す.

- 任意の二分探索木に対し, その木を最適木とする検索確率が存在する (4.1 節).
- 回転操作により隣接してる二分木は, 対応する領域は $\mathcal{R}$ 内で隣接している (4.2 節).

2 二分木の生成と各内点の路長を同時に計算するアルゴリズム

本節では, 各二分木を生成すると同時にその二分木の各内点までの路長を決定するアルゴリズムについて述べる. [5] で提案した 2 つのアルゴリズム (二分木の生成, 各内点までの路長の決定) は, 基本的に 2 進数列の 0,1 により, 分岐しているだけなので, 同時に計算を行うことが可能である. ただし, 部分木の共有化は不可能である. 次のような反例が存在する. 今, 010100111 と 001100111 という 2 つの 2 進数列を考える. 先頭の 3 ビットは, 010, 001 となり, 0, 1 の数は同じであり, 残りの部分の 100111 は共通である. このような部分をアルゴリズム (n 内点の二分木の生成木を構築) では, 共有化を行った. しかし, 二分木 010100111, 001100111 の各内点までの路長を考えると,

$$\boldsymbol{l}(010100111) = (0,1,3,2)$$

$$\boldsymbol{l}(001100111) = (1,0,2,1)$$

となり, それぞれの内点までの路長はすべて異なる. これは, 2 進数列の生成という意味では, 接尾文字が等しければ, 接頭文字の 0,1 の順序の違いだけで, 生成が可能であった. しかし, それぞれの内点に路長を割り振る作業を考えると, どの 0 ノードがどの a_i(Index=i) を割り振られるのかという部分に違いが存在する. 例えば, 010 ならば, 先頭の 0 が a_1(Index=1) となり, 路長 0 が割り振られる. また, 001 ならば, 2 番目の 0 が a_2(Index=2) となり, 路長 0 が割り振られることになる.

以上のことを考慮にいれ、次のアルゴリズム (n 内点の生成木に対し, 各内点のラベル付けと路長の計算, 図 1) を得る.

[注意]

- 路長は, 必ず 0 から割り振られる. なぜならば, 最初にポップされるのは, 根であるため. 同様に次にポップされる内点に割り振られるのは, 1 である (前順探索をおこなっているため).
- 索引 (Index) の変化により, 現在の路長をどの内点へ割り振るかが決定される.
- 路長自身は, スタックに幾つ 0 ノードが積まれているか, と何度スタックが空になったかで, 決定されるため, 変数として, 保持する必要はない.
- 1 をポップした時点で 1 つインデックスが決定する. つまり, 1 の辺を辿る度に, Index と Level が決定されていく. もし, 各内点までの路長を与えて, それを満たす木が存在するかどうかを調べるならば, この時点で与えられた路長と Level との比較を行えばよい.

3 回転操作と路長和の変化

本節では, 1 度の回転操作を行うことにより, 路長和がどのように変化するのかを考察する. 以下で回転操作という場合には, 辺により接続されている内点同士の回転操作だけを考える. つまり, 1 度だけの回転操作を考え, 2 度以上の連続した回転操作は扱わない.

まず, 内点 a_i, a_j の上下関係を変化させる回転操作を考える (図 2 参照). つまり, 根 a_j の左の子が a_i, 右の部分木が T_3, a_i の左部分木が, T_1, 右部分木が T_2 とする. この木全体を T_j と呼ぶとする. この木 T_j から, 木 T_i(根が a_i, その左部分木が T_1, 右の子が a_j, a_j の左部分木が T_2, 右部分木が T_3 となる) への回転操作に着目する.

このとき, 次の補題が成立する.

Input 点数 n;

Output $2n+1$ 点からなる 2 進数列で, 二分木に対応する列の集合 B_n

各二分木に対して, 現在訪れている内点の根からの路長 $C[l]$ $(l=1,\ldots,n)$

1 $T(n,j,k)=\emptyset$ for $j=1,\cdots,n,\ k=1,\ldots,j$;

2 $B[i]=0$ $(i=0,\ldots,2n)$; /* $2n+1$ の 2 進数列を保存する */

3 $C[l]=0$ $(l=1,\ldots,n)$; /* 各内点の路長 (Level) を保持する */

4 $B_n=\emptyset$;

5 MakeBinarySequences$(n,0,0,0,0)$;

```
MakeBinarySeqences(n,j,k, Index, Level) {
    if (j == n) then
        for (i = j + k to 2n) do
            B[i] = 1;
            C[Index] = Level;
            Index := Index +1; Level := Level +1;
        endfor;
        B_n := B_n ∪ {B[0]B[1]···B[2n]};
        Print C[1]···C[n];
        End;
    endif
    if (j < k) then End; endif
    B[j + k] = 0;
    MakeBinarySeqences(n, j + 1, k, Index, Level+1);
    B[j + k] = 1;
    C[Index + 1] = Level-1;
    if (j == k) then Level := Level + 1; endif
    MakeBinarySeqences(n, j, k + 1, Index+1, Level-1);
}
```

図 1: アルゴリズム (n 内点の生成木に対し, 各内点のラベル付けと路長の計算)

補題 1 上の木 T_j から T_i への回転操作による路長和の変化 $R(i,j):=E(T_j)-E(T_i)$ は,

$$R(i,j)=-\sum_{k=s}^{i}p_k+\sum_{k=j}^{t}p_k \qquad (3.1)$$

となる. ただし, s は T_1 に含まれる最小内点のインデックス, t は T_3 に含まれる最大内点のインデックスとし, $p(a_i)$ を各内点 a_i の検索確率とする. また, 期待値を $p(a_i)$ の関数とみなした場合の係数の変化の形式で書くと,

$$(0,\cdots,0,-1,\cdots,-1,0,\cdots,0,1,\cdots,1,0,\cdots,0)$$

となる。

証明: ここで扱う回転操作は, 局所的な構造であり, 内点 a_j 以下の部分にしか関係しない. この a_j 以下の部分に含まれる内点は, T_1 に含まれる

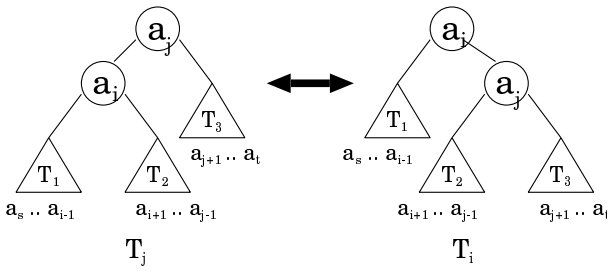

図 2: 二分木 T_j から T_i への回転操作

最小内点の a_s から T_3 に含まれる最大内点の a_t までである. これ以外の部分は, 変化がないので、a_k のうち, $k = 0, \cdots, s-1$ と $k = t+1, \cdots n$ まで部分は, 路長和を変化させない. つまり, $R(i,j)$ の $p_k(k = 0, \cdots, s-1, k = t+1, \cdots, n)$ の係数は, 0 なる.

次に, a_i の部分木である T_1, T_2 に関して考える. T_2 は, 回転操作が行われると, a_j の左部分木となり, そのどの内点も根からの路長は変わらない. また, ここに含まれる内点は, $a_{i+1}, \cdots, a_{j-1}$ である. つまり, これらの $R(i,j)$ の $p_k(k = i+1, \cdots, j-1)$ の係数も 0 となる. T_1 は, 全ての内点が 1 つだけ上にあがることになるので, ここでの利得が全ての内点の和の分だけあることになる. この部分木に含まれる内点は, $a_s, \cdots, a_{i-1}$ であり, a_i 自身の検索確率も利得となるので,

$$-\sum_{k=s}^{i} p_k$$

だけ路長和が変化する.

また, 同様にして, T_3 では全ての内点が一つ下がるため,

$$\sum_{k=j}^{t} p_k$$

だけの増加がある. これらの和をとると, 式 (3.1) になる. また, p_k の関数とみなした場合の係数の変化を見ると, 上記のように記述できる. □

4 最適木と検索確率

4.1 与えられた二分木を最適木とする検索確率

本節では, 与えられた二分木を最適木とする検索確率について考える. 各最適領域は連結であり, かつ凸領域でもあるため, 最適木とする検索確率の周辺も, 同様に最適木を与える検索確率となる.

まず, 各内点に路長に応じた確率を付加することを考える. つまり, 内点 a_i の根からの路長を $l(a_i)$ とすると, その重みを

$$w_i = 1/K^{l(a_i)}$$

とする. この重み付けを**重み付け (その 1)** と呼ぶ.

[**注意**] 上記の $\boldsymbol{w} = (w_1, w_2, \ldots, w_n)$ は, $\sum_i w_i \neq 1$ なので, 確率としての条件を満たしていないが, これは $\sum_i w_i$ で正規化を行えばよく, 異なる二分木に対して, 路長和の比較を行うだけならば, 正規化後の検索確率 p_i, 重み w_i のどちらを用いて, 計算を行っても, 路長和は定数倍の違いしかなく同じ結果を得ることが可能である.

まず, 補題 1 を考えると, 回転操作を行ったとしても, T_j が最適のままで居るための条件として, T_1 にはすべての内点が存在し, T_3 は空集合という場合が考えられる. そのため, 次のような関係が成立しないと, 最適値であることを保てない.

$$(i \text{ の重み}) + (T_1 \text{の重みの和}) < (j \text{ の重み}) \tag{4.1}$$

つまり,

$$\frac{1}{K} + \frac{1}{K^2} + \frac{2}{K^3} + \cdots + \frac{2^m}{K^{m+2}} + \cdots < 1$$

$$\frac{K-1}{K(K-2)} < 1.$$

$K > 2, m \to \infty$ という制約下で K に関して解くと

$$K^2 - 3K + 1 > 0$$

$$K > \frac{3+\sqrt{5}}{2} \sim 2.618034$$

という K に対する制約が存在することになる.

つまり, $K > \frac{3+\sqrt{5}}{2}$ となる実数を取ることにより, 一つの重みを割り当てることができる. 例えば, $K = 3$ とすると, 4 内点からなる二分木 (図 3 参照) には,

$$\left(1, \frac{1}{3}, \frac{1}{9}, \frac{1}{27}\right) \sim \left(\frac{27}{40}, \frac{9}{40}, \frac{3}{40}, \frac{1}{40}\right)$$

という検索確率を割り当てれば良いことになる.

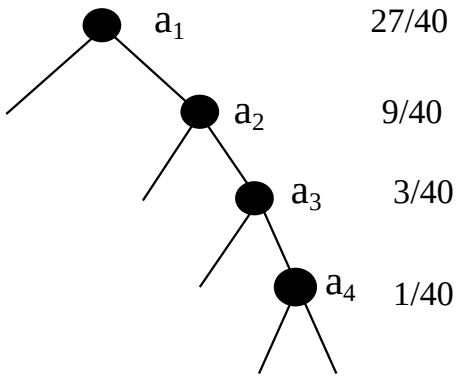

図 3: 4 点からなる二分木とそれを最適木とする検索確率

定理 1 ある二分探索木 T と任意の $K > \frac{3+\sqrt{5}}{2}$ を決める. すべての内点 a_i に,

$$w_i = 1/K^{l(a_i)}$$

という重みを付加する. ただし、$l(a_i)$ は, T における根から a_i までの路長とする, さらに, 検索確率を $p_i = w_i/\sum_i w_i$ とする.

このとき $\boldsymbol{p} = (p_1, p_2, \ldots, p_n)$ は, T のみを最適木とする検索確率である.

証明: まず, T の根に割り振られている内点 (以下 a_j とする) に関して考える. この内点 a_j が根以外にある二分木を, T' とする. T' 中の a_j の位置を根に近づけていく回転操作を考える. a_j の根からの路長を l をすると, ちょうど l 度の回転操作を行うことで, a_j を根まで持ち上げることが可能である. このときの回転操作で生成される木の列を

$$T' \to T'^1 \to T'^2 \to \cdots \to T'^l$$

とする.

このとき,

$$E(T') > E(T'^1) > \cdots > E(T'^l)$$

が成立する. まず, 各内点に対する重み付け (その 1) を思い出そう. この重み付けは, 各内点が元々の根からの路長よりも大きくなればなるほど路長和が大きくなるように割り振られている. これは, 各内点が元々の根からの路長よりが小さくなればなるほど路長和が小さくなることと等価である. つまり, $T', T'^1, \ldots, T'^l$ という列に対して, $E(T)$ は単調減少することになり, 上の式が示される. 結局, a_j は, 根にある場合に最もよい路長和を与えることになる.

さて, a_j が根にある場合が最も路長和が小さくなることが分かったので, 次のレベルにある内点に付いて考える. このとき, a_j の左部分木 T_1 に含まれる内点は, $\{a_1, \ldots, a_{j-1}\}$ と a_j の右部分木 T_2 に含まれる内点は, $\{a_{j+1}, \ldots, a_n\}$ となり, これは, 二分探索木の性質から, 元々与えられている T, 上の列の T'^l のどちらの二分探索木でも, 同一の集合となる. そこで, それぞれの内点集合の対して, 先程と同様の議論を行えば, その集合内での路長和を最適にする内点は, T で a_j に接続している内点であることが分かった. 以下これを端点まで繰り返すことにより, T の最適性を示すことが証明された. □

よって, 直ちに次の系が得られる.

系 1 任意の二分探索木に対して, 最適となる検索確率が存在する.

4.2 2 つの二分木を最適二分探索木とする検索確率

本節では, 回転操作により接続されている 2 つの二分探索木 T_i, T_j の双方を最適木とする各内点の重みについて考察する. まず, 二分木 T_j から, T_i への回転操作を考える. T_j の構造は, T_j を根として持っている二分木で, その左の子は a_i, 右部分木は T_3(内点は $a_{j+1}, \cdots a_t$) であり, a_i の左部分木は T_1(内点は $a_s, \cdots a_{i-1}$), a_i の右部分木は T_2(内点は $a_{i+1}, \cdots a_{j-1}$) である. また, T_i の構造は, a_i を根として持ち, 左部分木が T_1, 右の子が a_j であり, a_j の左部分木は T_2 であり, 右部分木は T_3 である (図 2 参照). この回転操作による路長和の変化式 (3.1) より,

$$(T_1\text{の重み}) + (a_i\text{の重み}) = (T_3\text{の重み}) + (a_j\text{の重み})$$

という関係式が成立するならば, 内点 a_i と a_j の回転操作を行っても, T_i と T_j の路長和は変化しない.

つまり, 2 つの二分探索木の間で, 回転操作を行っても, 路長和が変化せず, この 2 つの二分探索木を最適木とする検索確率を構成することができる. そこで, 重み付け (その 1) と重み付けの補正値 W を用いた重み付け (その 2) を提案する.

[**重み付け (その 2)**]

1. 定数 K' を決め, 重み付け (その 1) を使い, すべての内点に重みを付ける.

2. 内点 a_i(根からの路長を $m+1$ とする) の重みを

$$w(a_i) = \frac{1}{K'^{m+1}} + W$$

とする. ただし, W は回転操作を行っても路長和が変化しないための補正値であり,

$$W := \sum_{k=j}^{t} w(a_k) - \sum_{k=s}^{i} w(a_k)$$

と定義する.

さて, 重み付け (その 2) を用いて, 重み付けをする. 前節同様に, 与えられた二分木に対して, 任意の K' を用いて最適木となる確率を計算できるわけではない. そこで, 補正値 W の評価やいくつかの状態 (図 4, 5) を考え, K' の範囲を与える.

まず, W の値の上限は、図 4 において, T_3 が完全二分木であり, T_1 が空集合の場合である. つまり, 次のように上限を計算することが可能である.

$$\begin{aligned} W &\leq \frac{1}{K'^{m}} + \left(\frac{1}{K'^{m+1}} + \frac{2}{K'^{m+2}} + \cdots \right) - \frac{1}{K'^{m+1}} \\ &\leq \frac{K'+2}{K'^{m+1}(K'-2)}. \end{aligned}$$

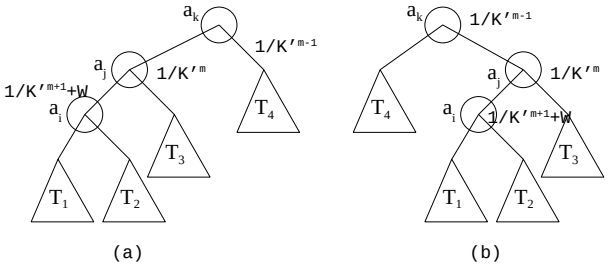

図 4: 重み付け (その 2) を行った場合に付加される重み (a) a_k の左の子が a_j の場合, (b) a_k の右の子が a_j の場合

さて, 重み付け (その 2) を用いて, 二分木に重み付けを行うことはできた. ここで重み付けに用いた定数 K' の満たすべき条件を考える. まず, 重み付け (その 1) の場合の K と同様の条件は, 満たさなければならない. さらに, a_i に重みを加えているので, それらを引き上げるような a_i とその親 a_k の回転操作を行うと, 路長和が大きくなるように制約を付ければよい. この場合分けは, 元々の二分木において, 上から a_k, a_j, a_i いう順になっているとすると, 左右どちらの子になるかを考えれば良い. 対称性を除くと, 次の 2 通りとなる.

- a_k を根とし, a_k の左の子が a_j, a_j の左の子が a_i で, 補正値 W が付加されている. その他は, 左から順に部分木 T_1, T_2, T_3, T_4 となっている (図 4 (a) の場合).
- a_k を根とし, a_k の右の子が a_j, a_j の左の子が a_i で, 補正値 W が付加されている. その他は, 左から順に部分木 T_4, T_1, T_2, T_3 となっている (図 4 (b) の場合).

まず, 図 4 (a) の場合を考える. このとき, 回転操作による変化量は,

$$\begin{aligned} &\{(a_k\text{の重み}) + (T_4\text{の重み})\} \\ &- \{(a_i, a_j\text{の重み}) + (T_1, T_2\text{の重み})\} \end{aligned}$$

となり, どのような T_1, T_2, T_4 に対しても, この変化量が正値でなければならない. 最悪の場合は, T_4 が空集合であり, T_1, T_2 が完全二分木である場合である. つまり,

$$\begin{aligned} &(a_k\text{の重み}) - \{(a_i\text{の重み}) \\ &+ (a_j\text{の重み}) + (T_1, T_2\text{の重み})\} > 0 \end{aligned}$$

となる必要がある. これを K' の式に直し, その範囲を計算する.

$$\begin{aligned} &\frac{1}{K'^{m-1}} - \left(\frac{1}{K'^{m}} + \frac{1}{K'^{m+1}} + W + \frac{1}{K'^{m+2}} + \cdots \right) \\ &= \frac{1}{K'^{m-1}} - \frac{1}{K'^{m}} - \frac{1}{K'^{m+1}} \left\{ 1 + \frac{2}{K'} + \cdots \right\} - W \\ &> \frac{1}{K'^{m-1}} - \frac{1}{K'^{m}} - \frac{1}{K'^{m}(K'-2)} - \frac{K'+2}{K'^{m+1}(K'-2)} \\ &= \frac{1}{K'^{m+1}(K'-2)} (K'^{3} - 3K'^{2} + 2) > 0 \end{aligned}$$

この式を $K' > 2$ という制約で解くと,

$$K' > 1 + \sqrt{3} \sim 2.7320508$$

となる.

次に, 図 4 (b) の場合を考える. この場合, 変化量は,

$$\begin{aligned} &\{(a_k\text{の重み}) + (T_4\text{の重み})\} \\ &- \{(a_j\text{の重み}) + (T_3\text{の重み})\} \end{aligned}$$

となるが, この部分は全く重み付け (その 1) と全く変わっておらず, その条件を満たしていればよい. よって,

$$K' > \frac{3+\sqrt{5}}{2}$$

となる.

さらに, a_i と a_j に回転操作を適用された後の状況を考える. この場合も, a_i と a_k の回転操作が行われても, 路長和が減少しないように K' の制限を行う必要がある.

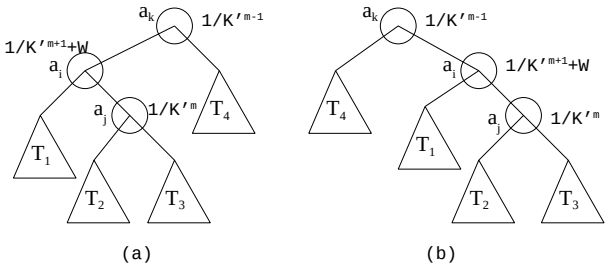

図 5: 図 4 において, a_i と a_j の回転操作を行ったあとの状態 (a) a_k の左の子が a_i の場合, (b) a_k の右の子が a_i の場合

a_i と a_k の回転操作を行っても, 最適値を保持することが可能な K' の範囲を考える. 図 5(a) の場合には, その変化量は,

$$\{(a_k\text{の重み}) + (T_4\text{の重み})\} - \{(a_i\text{の重み}) + (T_1\text{の重み})\}$$

となり, これがどのような T_1, T_4 に対しても必ず正値となるように K' を決めたい. 最悪の場合は, T_4 が空集合で, T_1 が完全二分木となる場合である. つまり,

$$\frac{1}{K'^{m-1}} - \frac{1}{K'^{m+1}} - W - \left(\frac{1}{K'^{m+2}} + \frac{2}{K'^{m+3}} + \cdots\right) > 0$$

W の評価式を用いて, 式の計算を行うと

$$\frac{1}{K'^{m+1}(K'-2)}\left(K'^3 - 2K'^2 - 2K' - 1\right) > 0$$

となり,

$$K' > \frac{1}{6}C + \frac{20}{3C} + \frac{2}{3} \sim 2.831177208$$

という制約が成立する. ただし, $C = (316 + 12\sqrt{249})^{1/3}$ である.

図 5(b) の場合には, その変化量は,

$$\{(a_k\text{の重み}) + (T_4\text{の重み})\} - \{(a_i, a_j\text{の重み}) + (T_2, T_3\text{の重み})\}$$

となる. この値がどのような場合にでも, 正値にならなければならない. 最悪の場合は, T_4 が空集合で, T_2, T_3 が完全二分木の場合である. つまり,

$$\frac{1}{K'^{m-1}} - \left(\frac{1}{K'^{m+1}} + W + \frac{1}{K'^{m+1}} + \frac{2}{K'^{m+2}} + \cdots\right) > 0$$

この式を解くと,

$$\frac{1}{K'^m(K'-2)}(K'^2 - 3K' - 2) > 0$$

となり,

$$K' > \frac{3+\sqrt{17}}{2} \sim 3.5615528$$

という範囲となる.

これら 5 つの K' の制約の交わり部分を考えると,

$$K' > \frac{3+\sqrt{17}}{2} \sim 3.5615528$$

という範囲に K' を取ることにより, すべての場合で回転操作により, 最適値を保持できるようになる.

よって、次の定理が成立する.

定理 2 回転操作で結ばれた 2 つの二分木だけを最適にする検索確率が存在する.

証明: $K'(> (3+\sqrt{17})/2)$ による重み付け (その 2) を用いて, 二分木 T, T' に重みをつける. このとき, $K' > (3+\sqrt{5})/2$ となるため, 定理 1 の証明と同様に, 上から順に回転操作にかかわらない部分の内点の重みは, T, T' が最適であるように決めることが可能である.

さて, 問題なのは, 回転操作にかかわる部分である. まず, a_i が a_j の子であるという仮定を置いても, 一般性は失わない (逆になる場合は, 二つの内点の役割を入れ換えて考えればよい). このように仮定を置くと, a_i が左の子か, 右の子かということで違いが計算に違いが生じる (図 4(a),(b) 参照). 上記のように K' を指定すれば, このどちらの場合も, 回転操作により, 路長和が大きくなる.

また, a_i と a_j に回転操作を適用した後 (図 5(a),(b) 参照) に, a_i と a_k の関係をそれぞれ考えると, この場合も K' を上記の条件を満たすようにとれば, 路長和が小さくなることはない.

よって, T と T' だけを最適にする検索確率が存在することを示せた. □

上の定理を言い替えると, 以下の補題を得ることができる.

系 **2** 回転操作で結ばれた 2 つの二分探索木を最適にする領域は接している.

ここで実際に図 6 の場合の二分探索木 T に重み付け (その 2) を適用する.

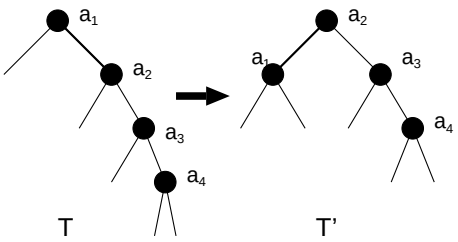

図 6: 回転操作により隣接した 2 つの二分探索木 T, T'

まず, $K' = 4$ と取ると, $\boldsymbol{w} = \left(1, \frac{1}{4}, \frac{1}{16}, \frac{1}{64}\right)$ が T に割り振られる. $W = 59/64$ となるので, $\boldsymbol{w} = \left(1, \frac{59}{64}, \frac{1}{16}, \frac{1}{64}\right)$ となる. これを正規化すると, 検索確率 $\boldsymbol{p} = \left(\frac{1}{2}, \frac{59}{128}, \frac{1}{32}, \frac{1}{128}\right)$ を得ることができる. このときの 14 個の二分探索木に対する路長和は, $\frac{314}{128}$, $\frac{309}{128}$, $\frac{195}{128}$, $\frac{249}{128}$, $\frac{194}{128}$, $\frac{\mathbf{70}}{\mathbf{128}}$, $\frac{73}{128}$, $\frac{124}{128}$, $\frac{131}{128}$, $\frac{186}{128}$, $\frac{188}{128}$, $\frac{183}{128}$, $\frac{73}{128}$, $\frac{\mathbf{70}}{\mathbf{128}}$ となり, 2 つの二分探索木 T, T' だけが最適木となっていることがわかる.

5 まとめ

本稿では, 二分探索木が最適木となるような領域の性質について述べた. まず、第 2 節では, 二分木を生成すると同時にその内点への路長を計算するアルゴリズムを提案した. このアルゴリズムは, 部分木の共有化こそできないものの, 各内点への路長を与えた場合にそれを満たす二分木があるかどうかを判定することが 1 つのアルゴリズムで可能となった.

次に, 与えられた二分探索木 T に対し, T を最適木とする検索確率を構成することで, 任意の二分探索木が最適木となることを示した. また, 回転操作で結ばれた 2 つの二分探索木に対して, 双方を最適木とする確率分布を構成した. これにより, 回転操作で結ばれた 2 つの二分探索木に対応する領域は, 隣接していることがわかった. 言い替えると, ある二分探索木 T と回転操作により移りあえる二分探索木の集合を $\mathcal{R}$ とし, T を最適木とする確率分布の領域に隣接している領域に対応する二分探索木の集合を $\mathcal{S}$ とした場合に,

$$\mathcal{R} \subseteq \mathcal{S}$$

が成立する. この包含関係は、二分探索木によっては, 等号であるが, 真に含まれる場合もあることが既にわかっている. どのような場合に, 真に含まれることが起こるのかを調べるのはこれからの課題である.

また, 二分木と凸多角形の三角形分割の関係は, Eular らにより示されている [3, pp.269-272]. さらに, [4] において, 二分木での回転操作と凸多角形の三角形分割での対角変形の等価性が示されている. 我々は, 二分探索木と凸多角形の重み付き三角形分割の関係が存在するのではないかと考え, 研究を進めている. 最適領域同士の隣接関係は, 回転操作や対角変形よりも, 隣接数の多い構造を持つため, その接続関係を与える必要十分条件に関しても研究を行っている.

参考文献

[1] H. Edelsbrunner : *Algorithms in Combinatorial Geometry*. Springer-Verlag, Berlin, 1987.

[2] D.E. Knuth: *The Art of Computer Programming*, Vol. 3, Second Edition, Addison-Wealey, 1998.

[3] R. Sedgewick, P. Flajolet: *An Introduction to the Analysis of Algorithms*, Addison-Wealey, 1996.

[4] D.D. Sleator, R.E. Tarjan, W.P. Thurston: Rotation Distance, Triangulations, and Hyperbolic Geometry. *Proceedings of the eighteenth annual ACM symposium on Theory of computing*, 1986, pp.122 - 135.

[5] 大西 建輔, 星　守: 検索確率をもつ二分探索木の探索長の最適期待値を与える領域分割の生成. 情報処理学会研究報告 2001-AL-76, 2001, pp.65-72.