数理解析研究所講究録
第1832巻 2013年 158-176

完全二分 AND-OR 木上の最適な乱択アルゴリズムについて

仁井田哲尚　　小川孝典 *

1 序

AND-OR 木はブール関数の一種であり, ゲーム木でもある. 具体的には, 根に AND($\wedge$) がラベル付けされており, その子節には OR($\vee$), 更にその子節には AND($\wedge$) というように, AND($\wedge$) と OR($\vee$) が交互にラベル付けされている木のことである.

AND-OR 木を次のように計算する. 各葉に 0 か 1 を割り当てておく. AND-OR 木を計算する決定性アルゴリズムによって, 各葉へ問い合わせすることで葉に割り当てられた値が判明する. そうして, 判明した値に対して, 親節のラベルを演算子として, 子節の値を演算し, 親節に割り当てていく. 最終的に根の値が分かるまで行う.

そのとき決定性アルゴリズム A に対する AND-OR 木の計算コストを, 「A に対して, 根の値が判明するまでにかかった葉への問い合わせ回数」と定義する.

AND-OR 木の計算複雑さに関しては, 今まで様々な研究が行われてきた. 度々研究の対象となるのは計算複雑さの指標の一種である randomized complexity と distributional complexity の二つである.

Randomized complexity とは, 決定性アルゴリズムの集合上の確率分布について, その確率分布が真理値割り当て全体に対して, どれだけコスト期待値を下げることが出来るかという指標であり, distributional complexity とは, 真理値割り当ての集合上の確率分布を考え, その確率分布が, 決定性アルゴリズム全体に対して, どれだけコスト期待値を上げられるかという指標である.

この二つの値は一致することが [5] で示されている.

* 首都大学東京 理工学研究科 数理情報科学専攻

1986 年に発表された [3] では, 高さ h の完全二分 AND-OR 木の randomized complexity が $\Theta\left(\left(\frac{1+\sqrt{33}}{4}\right)^h\right)$ だと示されている.

また 2007 年に [2] では, 0-set と 1-set と呼ばれる真理値割り当ての集合を考え, distributional complexity を実現する確率分布が, どの決定性アルゴリズムに対してもコスト期待値が等しくなる E^1-分布と同値であることが示されている.

その結果に対し, [4] では, 真理値割り当て及びアルゴリズムの転置という概念を用いて, [2] での議論を細密化し, 転置に関して閉であるアルゴリズムの集合上でも [2] の結果と同様なことがいえることを示されている. その中で, 向きが固定された決定性アルゴリズムに対して E^1-分布は一意には定まらないことが示されている.

[2] や [4] では, 真理値割り当ての集合上の確率分布に焦点を当てており, AND-OR 木上の乱択アルゴリズムについては掘り下げられていない. そこで, [2], [4] の議論が乱択アルゴリズムにおいても成り立つのではないかという疑問を持った. 我々は [4] と同様な手法, すなわち, 真理値割り当て及びアルゴリズムの転置という概念を用いて, [4] の結果との双対となる randomized complexity を実現する乱択アルゴリズムの特徴付けを考えていく.

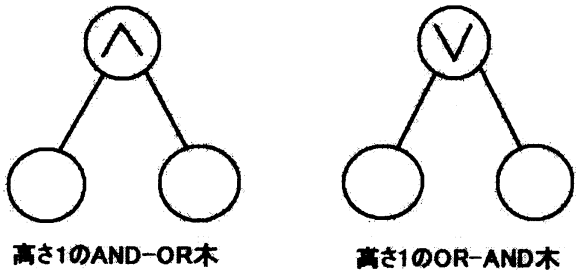
まず, randomized complexity を実現する乱択アルゴリズムを最適な乱択アルゴリズム, 1-set に対してコスト期待値が一定になる乱択アルゴリズムを E^1-乱択アルゴリズムとする. このとき, E^1-乱択アルゴリズムであることと最適な乱択アルゴリズムであることが同値であると予想した. 実際, [4] の補題 1, 補題 2 に関しては, 乱択アルゴリズムにおいてもほぼ同様な結果が得られた. 更に, 最適な乱択アルゴリズムは, E^1-乱択アルゴリズムであることを示した. しかし, その逆は成り立たないことが証明できたので, 我々の予想は完全には正しくないことがわかった. 加えて, 転置に関して連結かつ閉であるアルゴリズムの集合上の一様分布は最適な乱択アルゴリズムであることが証明できた.

以下, 本稿の構成を述べる.

第 2 節では AND-OR 木に関する基本的な事項及び、真理値割り当てとアルゴリズムの転置について述べる. 第 3 節では, [2], [4] で示された結果を具体的に述べる. つまり, distributional complexity を実現する真理値割り当ての集合上の確率分布の特徴付けはどのようなものであるかを述べる. 第 4 節では, [4] で示された結果の乱択アルゴリズムバージョンを考え, 最適な乱択アルゴリズムがどのような性質を持っているものか言及していく.

2 定義

高さ 1 の**完全二分 AND-OR 木 (perfect binary AND-OR tree)** とは, 「かつ」を表す論理記号 $\wedge$ の下から, 二本の枝が伸びている木構造のことである. 同様に, 高さ 1 の**完全二分 OR-AND 木 (perfect binary OR-AND tree)** とは, 「または」を表す論理記号 $\vee$ の下から, 二本の枝が伸びている木構造のことである. 木構造の頂点を**根 (root)**, 根から伸びた二本の枝の先を**葉 (leaf)** とよぶ.



高さ 1 の完全二分 AND-OR 木と完全二分 OR-AND 木を, それぞれ $T_2^{1,\wedge}, T_2^{1,\vee}$ とかく.

$h \geq 2$ に対して, $T_2^{h-1,\wedge}$ が定義されたとき, 次のようにして完全二分木 $T_2^{h,\wedge}$ をつくる:

- 葉につながっている論理記号が $\wedge$ のとき, $T_2^{h-1,\wedge}$ のすべての葉に, $T_2^{1,\vee}$ の根を貼り合わせる.
- 葉につながっている論理記号が $\vee$ のとき, $T_2^{h-1,\wedge}$ のすべての葉に, $T_2^{1,\wedge}$ の根を貼り合わせる.

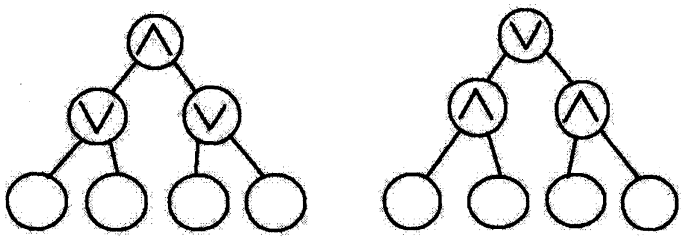
このようにして, 再帰的に一般の高さ h の完全二分 AND-OR 木 $T_2^{h,\wedge}$ が定義される.

同様に, $h \geq 2$ に対して, $T_2^{h-1,\vee}$ が定義されたとき, 次のようにして完全二分木 $T_2^{h,\vee}$ をつくる:

- 葉につながっている論理記号が $\wedge$ のとき, $T_2^{h-1,\vee}$ のすべての葉に, $T_2^{1,\vee}$ の根を貼り合わせる.
- 葉につながっている論理記号が $\vee$ のとき, $T_2^{h-1,\vee}$ のすべての葉に, $T_2^{1,\wedge}$ の根を貼

り合わせる.

このようにして, 再帰的に一般の高さ h の完全二分 OR-AND 木 $T_2^{h,\vee}$ が定義される.

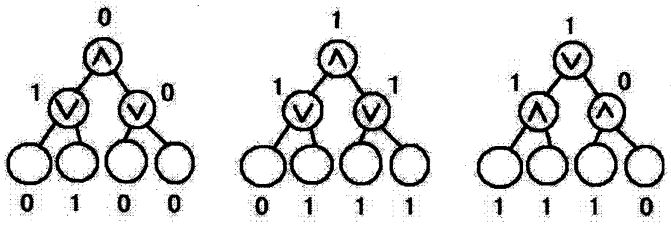


以下, 単に「木」という場合は, 完全二分 AND-OR 木と完全二分 OR-AND 木のいずれかをさすものとする. また, 以下では, 「完全二分」は省略する.

木構造の各点を**節 (node)** といい, 最上部の節を**根 (root)**, 最下部の各節を**葉 (leaf)** とよぶ. 根でも葉でもない節を**内部節 (internal node)** とよぶ.

さて, 各葉に 0(偽) または 1(真) を割り当てる. このとき, $\wedge$-節と $\vee$-節による論理計算により, 根には 0 または 1 が割り当てられることになる.

このルールによって, 木による計算が定義できる. 入力は葉への真理値割り当て, 出力は根に現れる真理値である.



記号. 高さ h の木の真理値割り当て全体を $\mathcal{W}^h$ で表すことにする. すなわち, $\mathcal{W}^h$ とは,

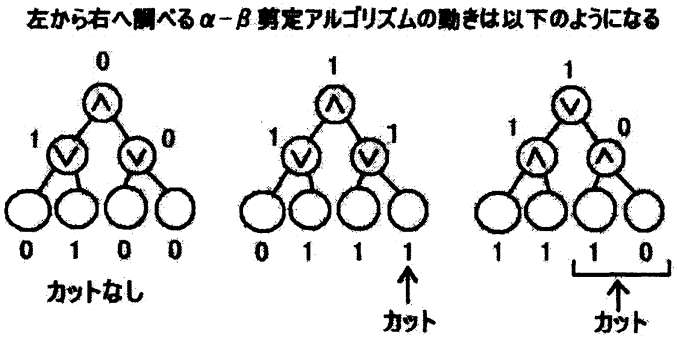
長さ 2^h のビット列の全体である. 木の高さを特に明示する必要のないときは, 単に $\mathcal{W}$ とかく.

木による計算が定義できたので, その複雑さを調べたい. そこで, 木による計算のコストを定義する.

真理値を木の葉に割り当て, その値を隠す. 次に, 決定性アルゴリズムに木の葉を探索させ, 隠された真理値をめくる.

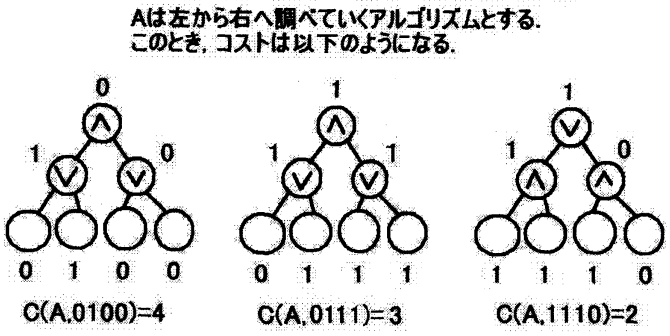
ここで, 木に対する**決定性アルゴリズム (deterministic algorithm)** とは, 「隠れている番号を知るために, どの葉を, どの順番でチェックするかの手順」のことである. 形式的には, 決定性アルゴリズムとは, **決定木 (decision tree)** のことである. 詳しくは, [1] を参照. また, 本稿で扱う決定性アルゴリズムはすべて α-β **剪定アルゴリズム** (α-β **pruning algorithm**) かつ**深さ優先アルゴリズム (depth first algorithm)** であるとする.

- α-β 剪定アルゴリズムとは, 以下のような制限を課したアルゴリズムである:
  $\wedge$-節の子節の片方が 0 であることがわかったとき, もう片方の節については調べない. 同様に, $\vee$-節の子節の片方が 1 であることがわかったとき, もう片方の節については調べない.
  これは, 必要以上に葉をチェックすることを防ぐ方法である.



- 深さ優先アルゴリズムとは, 以下のような制限を課したアルゴリズムである:
  節 u の値がわかったとき, u の親節 v の値がわかるまで v の子孫である葉以外は調べてはいけない.

このとき, 根の値がわかるまでにチェックする葉の枚数は, 真理値割り当てや決定性アルゴリズムによって変わる. そこで, 真理値割り当て ω に対して, 決定性アルゴリズム A によって探索するときの**コスト** $C(A,\omega)$ を「根の値がわかるまでチェックした葉の最小枚数」のこととする.



記号. 高さ h の木を探索する決定性アルゴリズムの全体を $\mathcal{A}_D^h$ で表すことにする. 木の高さを特に明示する必要のないときは, 単に $\mathcal{A}_D$ とかく.

次に, コストの期待値を定義する.

記号. 有限集合 X に対して, X 上の確率分布全体を $\mathcal{D}(X)$ と表すことにする. すなわち,

$$\mathcal{D}(X) := \left\{ d : X \to [0,1] : \sum_{x \in X} d(x) = 1 \right\}$$

である.

$\Omega \subset \mathcal{W}, \mathcal{A} \subset \mathcal{A}_D$ とする.
このとき, $A \in \mathcal{A}, d \in \mathcal{D}(\Omega)$ に対する**コスト期待値**を,

$$C(A,d) := \sum_{\omega \in \Omega} d(\omega) C(A,\omega)$$

とおく.
また, $\omega \in \Omega, A_R \in \mathcal{D}(\mathcal{A})$ に対する**コスト期待値**を,

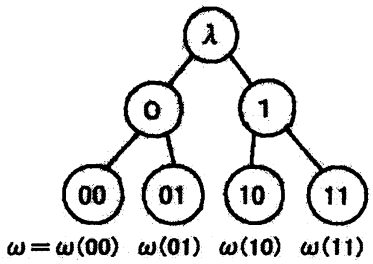
$$C(A_R,\omega) := \sum_{A \in \mathcal{A}} A_R(A) C(A,\omega)$$

とおく. アルゴリズムからなる集合上の確率分布 A_R を**乱択アルゴリズム (randomized algorithm)** とよぶ.

さて, コストの期待値を分析する際に, 真理値割り当ての全体や決定性アルゴリズム全体の集合を考えるよりも, その部分集合を考えた方がより精密な議論が展開できる可能性がある. そこで, 部分集合の性質を記述するために, 真理値割り当てやアルゴリズムの**転置 (transposition)** という概念を定義する.

各節は, ビット列によって辞書式に名前を付けておく. 内部節のコードを**内部節コード (internal node-code)**, 葉のコードを**葉コード (leaf-code)** という. 但し, 根のコードは空列を表す λ とする.

真理値割り当て ω は, 葉コードの集合から $\{0,1\}$ への関数とも思えるので, 葉コード l に対して, $\omega(l)$ で, ω の場所 l における値を表すことにする.



定義 2.1. [4]

- l は葉コードとし, u は内部節コードとする. このとき,

$$\mathrm{tp}_u(l) := \begin{cases} u(1-i)w & l = uiw\ (i \in \{0,1\},\ w \text{はビット列}) \text{となるとき} \\ l & \text{その他} \end{cases}$$

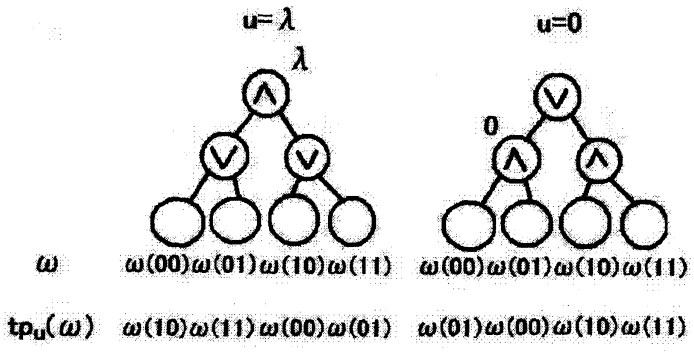
とおく. この $\mathrm{tp}_u(l)$ を, l の **u-転置 (u-transposition)** という.

- ω を真理値割り当て, u を内部節コードとする. また, 葉コードを左から並べた列を $\langle l_1, \cdots, l_n \rangle$ とする. このとき,

$$\mathrm{tp}_u(\omega) := \omega(\mathrm{tp}_u(l_1)) \cdots \omega(\mathrm{tp}_u(l_n))$$

とおく. これを, ω の **u-転置 (u-transposition)** という.

注意 2.2. $\mathrm{tp}_u(\omega)$ は, ω において, u の子孫となる葉の 2 つのグループを入れ替えたものである:



定義 2.3. [4]

- $A \in \mathcal{A}_D, \omega \in \mathcal{W}$ とする.
  割り当て ω のときに, A によってチェックされた葉コードをその順番に左から並べた列を, $\langle A, \omega \rangle$ の**質問履歴 (query history)** とよび, $\mathrm{qh}(A, \omega)$ とかく.
  質問履歴を $\mathrm{qh}(A, \omega) = \langle l_1, \cdots, l_k \rangle$ とするとき, $\mathrm{ah}(A, \omega) := \langle \omega(l_1), \cdots, \omega(l_k) \rangle$ を**回答履歴 (answer history)** という.
- u を内部節コードとし, $A \in \mathcal{A}_D$ とする. $B \in \mathcal{A}_D$ が A の **u-転置 (u-transposition)** であるとは, 以下を満たすときをいう:
  任意の $\omega \in \mathcal{W}$ に対して, $\mathrm{qh}(A, \mathrm{tp}_u(\omega)) = \langle v_1, \ldots, v_k \rangle$ とするとき,

$$\mathrm{qh}(B, \omega) = \langle \mathrm{tp}_u(v_1), \ldots, \mathrm{tp}_u(v_k) \rangle$$

  となる.
  このとき, B を $\mathrm{tp}_u(A)$ とかく.

注意 2.4. 任意の $\omega \in \mathcal{W}, A \in \mathcal{A}_D$ と内部節コード u に対して, 以下が成り立つ:

- $\mathrm{ah}(\mathrm{tp}_u(A), \omega) = \mathrm{ah}(A, \mathrm{tp}_u(\omega))$.
- $C(\mathrm{tp}_u(A), \omega) = C(A, \mathrm{tp}_u(\omega))$.

定義 2.5. [4] $\Omega \subset \mathcal{W}, \mathcal{A} \subset \mathcal{A}_D$ とする.

- Ω が**転置に関して閉**であるとは, 任意の $\omega \in \Omega$ と内部コード u に対して, $\mathrm{tp}_u(\omega) \in$

Ω となるときをいう.

- Ω が**転置に関して連結**であるとは, 任意の $\omega, \omega' \in \Omega$ に対して, ある $\omega_1, \ldots, \omega_n \in \Omega$ と, 内部節コード $u_1, \ldots, u_n$ が存在して,

$$\omega = \omega_1, \quad \mathrm{tp}_{u_j}(\omega_j) = \omega_{j+1}, \quad \omega_n = \omega'$$

が成り立つときをいう.

- $\mathcal{A}$ が**転置に関して閉**である, **転置に関して連結**である, も同様に定義する.

以下では,「転置に関して」を省略して, 単に「閉」,「連結」ということにする.

次に, コストの値が大きくならざるを得ないような真理値割り当ての集合を考える. これは, アルゴリズム側から見たときに「やっかい」な割り当てを集めたものである.

定義 2.6. [2] $i \in \{0,1\}, \Omega \subset \mathcal{W}$ とする.

Ω が **i-セット (i-set)** であるとは, Ω に属する任意の割り当てによる木の根の値は i であり, かつ, 以下を満たすような集合のうち最大のもののことである:

任意の Ω の元を木に割り当てたとき, $\wedge$-節の子節は, 0 と 1, 1 と 0, 1 と 1 のいずれか, $\vee$-節の子節は, 0 と 1, 1 と 0, 0 と 0 のいずれかとなる.

i-セットを表す集合を i-set とかく.

注意 2.7. i-セットをつくる方法として, **逆割り当て法 (reverse assigning technique : RAT, [2])** がある:

(1) 根に i を割り当てる.

(2) – 値が 1 の $\wedge$-節の子節には, 1 と 1 を割り当てる.
– 値が 0 の $\vee$-節の子節には, 0 と 0 を割り当てる.
– 値が 0 の $\wedge$-節の子節には, 一方に 0 を割り当て, もう一方に 1 を割り当てる.
– 値が 1 の $\vee$-節の子節には, 一方に 1 を割り当て, もう一方に 0 を割り当てる.

(3) i-セットとは, (1) と (2) によってつくられる真理値割り当てすべてからなる集合である.

注意 2.8.

- i-セットは連結な閉集合である.
- $\omega \in \mathcal{W}$ に対して,

$$\langle \omega \rangle := \left\{ \mathrm{tp}_{u_k} \circ \cdots \circ \mathrm{tp}_{u_1}(\omega) : \text{各 } u_j \text{ は内部節コード}, k \text{ は自然数} \right\}$$

は連結な閉集合となる. アルゴリズム A についても同様に $\langle A \rangle$ を定義すれば, $\langle A \rangle$

も連結な閉集合である.

- ある $\omega_1, \ldots, \omega_n \in \mathcal{W}$ が存在して,

$$\mathcal{W} = \langle \omega_1 \rangle \cup \cdots \cup \langle \omega_n \rangle \,,\ \langle \omega_i \rangle \cap \langle \omega_j \rangle = \varnothing \ (i \neq j)$$

とかける.

定義 2.9. 決定性アルゴリズムが**向き付き (directional)** であるとは, 葉を見る順番が, 真理値割り当てによらずに, あらかじめ決まっているときをいう. すなわち, $A \in \mathcal{A}_D$ が向き付きであるとは, 葉コードの順列 σ が存在して, 任意の $\omega \in \mathcal{W}$ に対して射影 p_ω が存在して,

$$\mathrm{qh}(A, \omega) = (p_\omega \circ \sigma)(l_1, \ldots, l_n)$$

が成り立つときをいう. 但し, $\langle l_1, \ldots, l_n \rangle$ は葉コードを左から並べた列とする.

向き付きアルゴリズムの全体を $\mathcal{A}_{\mathrm{dir}}$ とかく.

注意 2.10. $\mathcal{A}_{\mathrm{dir}}$ は連結な閉集合である.

以下, 本稿の全てで, Ω は $\mathcal{W}$ の部分集合, $\mathcal{A}$ は $\mathcal{A}_D$ の部分集合を表すものとする. さらに, T で, AND-OR 木または OR-AND 木を表すとする.

3 先行研究の紹介

定義 3.1.

- $$P(T, \Omega, \mathcal{A}) := \max_{d \in \mathcal{D}(\Omega)} \min_{A \in \mathcal{A}} C(A, d)$$

を, $\Omega, \mathcal{A}$ に対する T の **distributional complexity** とよぶ.

- $$R(T, \Omega, \mathcal{A}) := \min_{A_R \in \mathcal{D}(\mathcal{A})} \max_{\omega \in \Omega} C(A_R, \omega)$$

を, $\Omega, \mathcal{A}$ に対する T の **randomized complexity** とよぶ.

実は, distributional complexity と randomized complexity は等しい:

定理 3.2 (Yao 1977, [5]).

$$P(T, \Omega, \mathcal{A}) = R(T, \Omega, \mathcal{A}).$$

では, $P(T, \Omega, \mathcal{A})$ や $R(T, \Omega, \mathcal{A})$ を達成するような $d \in \mathcal{D}(\Omega)$ や $A_R \in \mathcal{D}(\mathcal{A})$ はどのような性質を持つのであろうか? distributional complexity に関してはこの問題について, [2] や [4] において考察されている.

定義 3.3. [2] d_e が $\mathcal{A}$ に対する Ω 上の**固有分布 (eigen distribution)** であるとは,

$$P(T,\Omega,\mathcal{A}) = \min_{A\in\mathcal{A}} C(A,d_e),$$

すなわち,

$$\max_{d\in\mathcal{D}(\Omega)} \min_{A\in\mathcal{A}} C(A,d) = \min_{A\in\mathcal{A}} C(A,d_e).$$

となるときをいう.

定義 3.4. [2] d が $\mathcal{A}$ に対する E^1**-分布** (E^1**-distribution**) であるとは,

- d は 1-set 上の分布であり,
- ある定数 c が存在して, 任意の $A\in\mathcal{A}$ に対して, $C(A,d)=c$ となる

ときをいう.

以下では, 木はすべて AND-OR 木であるとする.

定理 3.5 (Liu-田中 2007, [2]). 以下は同値である:

(1) d は $\mathcal{A}_D$ に対する $\mathcal{W}$ 上の 固有分布である.
(2) d は $\mathcal{A}_D$ に対する E^1-分布である.

[4] では, 定理 3.5 をさらに精密にした定理が証明されている:

定理 3.6 (鈴木-中村 2012, [4]). $\mathcal{A}$ は閉とする. このとき, 以下は同値である:

(1) d は $\mathcal{A}$ に対する $\mathcal{W}$ 上の固有分布である.
(2) d は $\mathcal{A}$ に対する E^1-分布である.

以下, 定理 3.6 を証明するために必要な補題, 定理を述べておく.

補題 3.7 (鈴木-中村 2012, [4]). Ω は連結, $\mathcal{A}$ は閉とする. このとき, ある定数 c が存在して, 任意の $d\in\mathcal{D}(\Omega)$ に対して,

$$\sum_{A\in\mathcal{A}} C(A,d) = c$$

が成り立つ.

定理 3.8 (鈴木-中村 2012, [4]). (1) $\Omega,\mathcal{A}$ は両方とも閉とする. また, d_u を Ω 上の一様分布とする. このとき, ある定数 c が存在して, 任意の $A\in\mathcal{A}$ に対して, $C(A,d_u)=c$ が成り立つ.

(2) Ω は閉かつ連結, $\mathcal{A}$ は閉とする. d_u を Ω 上の一様分布とする.
このとき, 任意の $d \in \mathcal{D}(\Omega)$ に対して, 以下は同値である:
(a) d は Ω 上の $\mathcal{A}$ に対する固有分布である.
(b) ある定数 c が存在して, 任意の $A \in \mathcal{A}$ に対して, $C(A,d) = c$ が成り立つ.
(c) ある $B \in \mathcal{A}$ が存在して, $\min_{A\in\mathcal{A}} C(A,d) = C(B,d_u)$ が成り立つ.
(d) $\min_{A\in\mathcal{A}} C(A,d) = \dfrac{1}{|\mathcal{A}|}\sum_{A\in\mathcal{A}} C(A,d)$.

1-set は閉かつ連結であることに注意すると, 定理 3.8 から以下がわかる.

系 3.9. (1) 任意の閉集合 $\mathcal{A}$ に対して, 1-set 上の一様分布は $\mathcal{A}$ に対する E^1-分布である.
(2) 任意の閉集合 $\mathcal{A}$ に対して, 以下は同値である:
(a) d は 1-set 上の $\mathcal{A}$ に対する固有分布である.
(b) d は $\mathcal{A}$ に対する E^1-分布である.

補題 3.10 (鈴木-中村 2012, [4]). $\mathcal{A}$ は閉とする. このとき, 以下は同値である:

(1) d は $\mathcal{W}$ 上の $\mathcal{A}$ に対する固有分布である.
(2) d は 1-set 上の $\mathcal{A}$ に対する固有分布である.

よって,

$$\max_{d\in\mathcal{D}(\mathcal{W})} \min_{A\in\mathcal{A}} C(A,d) = \max_{d\in\mathcal{D}(\text{1-set})} \min_{A\in\mathcal{A}} C(A,d)$$

が成り立つ.

定理 3.6 の証明.

$$\begin{aligned} & d \text{ は } \mathcal{W} \text{ 上の } \mathcal{A} \text{ に対する固有分布である.} \\ \overset{\text{補題 3.10}}{\Leftrightarrow}\ & d \text{ は 1-set 上の } \mathcal{A} \text{ に対する固有分布である.} \\ \overset{\text{系 3.9(2)}}{\Leftrightarrow}\ & d \text{ は } \mathcal{A} \text{ に対する } E^1\text{-分布である.} \end{aligned}$$

□

4 結果

我々は ramdomized complexity について考察をしていく. 具体的には,

$$\max_{\omega\in\Omega} C(A_R,\omega) = \min_{A'_R\in\mathcal{D}(\mathcal{A})} \max_{\omega\in\Omega} C(A'_R,\omega) \qquad (*)$$

を満たす乱択アルゴリズム A_R はどのようなものになるかを調べていく.

定義 4.1. $(*)$ を満たす乱択アルゴリズム $A_R \in \mathcal{D}(\mathcal{A})$ を, Ω に対する $\mathcal{A}$ 上の**最適 (optimal)** な乱択アルゴリズムと呼ぶ.

我々は, 前節で紹介した [2] や [4] における固有分布に関する結果と類似した, 最適アルゴリズムに関する定理群が得られるのではないか, と予想した. 結果としては, 補題 3.7, 定理 3.8 の乱択アルゴリズムバージョンとでもいうべき定理が得られた.

以下では, 木はすべて AND-OR 木であるとする.

定理 4.2. $\mathcal{A}$ は連結, Ω は閉であるとする. このとき, ある定数 c が存在して, 任意の $A_R \in \mathcal{D}(\mathcal{A})$ に対して,

$$\sum_{\omega\in\Omega} C(A_R,\omega) = c$$

が成り立つ.

証明. $T_2^{\wedge,h}$ は完全二分木なので, 任意の $A \in \mathcal{A}$, $\omega \in \Omega$, 内部節コード u に関して $C(\mathrm{tp}_u(A),\omega) = C(A,\mathrm{tp}_u(\omega))$ が成り立つ. このことを用いて, 任意の $A \in \mathcal{A}$ と内部節コード u について,

$$\begin{aligned}\sum_{\omega\in\Omega} C(\mathrm{tp}_u(A),\omega) &= \sum_{\omega\in\Omega} C(A,\mathrm{tp}_u(\omega)) \\ &= \sum_{\omega\in\Omega} C(A,\omega) \quad (\Omega \text{ は閉より})\end{aligned}$$

がいえる. よって, $\mathcal{A}$ は連結なので, 任意の $A \in \mathcal{A}$ に対して

$$\sum_{\omega\in\Omega} C(A,\omega) = c \quad (constant)$$

がいえる. そして, このことより,

$$\begin{aligned}\sum_{\omega\in\Omega} C(A_R,\omega) &= \sum_{\omega\in\Omega}\sum_{A\in\mathcal{A}} A_R(A)C(A,\omega)\\ &= \sum_{A\in\mathcal{A}} A_R(A)\sum_{\omega\in\Omega} C(A,\omega)\\ &= c \quad (constant)\end{aligned}$$

がいえた. □

次に, 定理 3.8 の乱択アルゴリズム版を考えたい. ただ, 完全な類似は成り立たないことが以下の例からわかる.

例 4.3. 高さ 1 の AND-OR 木を考える. この木を探索する決定性アルゴリズムは, 左の葉から探索するか, 右の葉から探索するかの 2 通りしかない. そこで, それらのアルゴリズムをそれぞれ $\rightarrow, \leftarrow$ によって表すとする. また, $\mathcal{A}_D = \{\rightarrow, \leftarrow\}$ 上の一様分布を A_R^u とする. このとき,

$$C(A_R^u, 00) = A_R^u(\rightarrow)C(\rightarrow, 00) + A_R^u(\leftarrow)C(\leftarrow, 00) = \frac{1}{2}(1+1) = 1,$$
$$C(A_R^u, 01) = A_R^u(\rightarrow)C(\rightarrow, 01) + A_R^u(\leftarrow)C(\leftarrow, 01) = \frac{1}{2}(1+2) = \frac{3}{2}$$

より, $C(A_R^u, 00) \neq C(A_R^u, 01)$ である.

そこで, 真理値割り当ての集合に連結性を仮定することにより, 以下の定理を得た.

定理 4.4. (1) $\mathcal{A}$ は閉, Ω は連結とする. A_R^u を $\mathcal{A}$ 上の一様分布とすると, ある定数 c が存在して, 任意の $\omega \in \Omega$ に対して, $C(A_R^u, \omega) = c$ が成り立つ.

(2) $\mathcal{A}$, Ω は閉かつ連結とし, A_R を $\mathcal{A}$ 上の乱択アルゴリズムとするとき, 以下の四つは同値である.

- (a) A_R は Ω に対して最適である.
- (b) ある定数 c が存在して, 任意の $\omega \in \Omega$ に対して, $C(A_R, \omega) = c$ が成り立つ.
- (c) ある $\alpha \in \Omega$ が存在して, $\max_{\omega\in\Omega} C(A_R, \omega) = C(A_R^u, \alpha)$ が成り立つ.
- (d) $\max_{\omega\in\Omega} C(A_R, \omega) = \frac{1}{|\Omega|}\sum_{\omega\in\Omega} C(A_R, \omega)$

証明. (1) $\mathrm{tp}_u(\omega) \in \Omega$ となる $\omega \in \Omega$ について,

$$\begin{aligned} C(A_R^u, \mathrm{tp}_u(\omega)) &= \frac{1}{|\Omega|} \sum_{A \in \mathcal{A}} C(A, \mathrm{tp}_u(\omega)) \\ &= \frac{1}{|\Omega|} \sum_{A \in \mathcal{A}} C(\mathrm{tp}_u(A), \omega) \\ &= \frac{1}{|\Omega|} \sum_{A \in \mathcal{A}} C(A, \omega) \quad (\mathcal{A} \text{ は閉より}) \\ &= C(A_R^u, \omega) \end{aligned}$$

となる. Ω は連結なので, 任意の $\omega \in \Omega$ に対して,

$$C(A_R^u, \omega) = c \qquad (constant).$$

(2) $(b) \Leftrightarrow (d)$, $(c) \Leftrightarrow (d)$ は補題 4.2, 補題 4.4(1) より自明である.

$(d) \Rightarrow (a)$ を示す. (d) を仮定すると, 任意の $A'_R \in \mathcal{D}(\mathcal{A})$ に対して,

$$\begin{aligned} \max_{\omega \in \Omega} C(A_R, \omega) &= \frac{1}{|\Omega|} \sum_{\omega \in \Omega} C(A_R, \omega) \\ &= \frac{1}{|\Omega|} \sum_{\omega \in \Omega} C(A'_R, \omega) \quad (\text{定理 4.2 より}) \\ &\leq \max_{\omega \in \Omega} C(A'_R, \omega) \end{aligned}$$

となる. よって, (a) が成り立つ.

$(a) \Rightarrow (d)$ を示す. (a) を仮定すると,

$$\begin{aligned} \max_{\omega \in \Omega} C(A_R, \omega) &= \min_{A'_R \in \mathcal{D}(\mathcal{A})} \max_{\omega \in \Omega} C(A'_R, \omega) \\ &\leq \max_{\omega \in \Omega} C(A_R^u, \omega) \\ &= \frac{1}{|\Omega|} \sum_{\omega \in \Omega} C(A_R^u, \omega) \quad ((1) \text{ より}) \\ &= \frac{1}{|\Omega|} \sum_{\omega \in \Omega} C(A_R, \omega) \quad (\text{定理 4.2 より}) \end{aligned}$$

となる. よって, (d) が成り立つ. □

ここで, E^1-分布の定義を参考にした乱択アルゴリズムを考える.

定義 4.5. A_R が $\mathcal{A}$ 上の **E^1-乱択アルゴリズム (E^1-randomized algorithm)** であるとは,

- A_R は $\mathcal{A}$ 上の分布であり,
- ある定数 c が存在して, 任意の $\omega \in$1-set に対して, $C(A_R, \omega) = c$ となる

ときをいう.

E^1-分布はすべての決定性アルゴリズムに対してコスト一定であるのに対し, E^1-乱択アルゴリズムはすべての真理値割り当てに対してコストが一定であることは要求していないことに注意する.

E^1-乱択アルゴリズムの定義と定理 4.4 から次の系を得る.

系 4.6. (1) 任意の閉集合 $\mathcal{A}$ に対して, $\mathcal{A}$ 上の一様分布は E^1-乱択アルゴリズムである.

(2) 任意の連結閉集合 $\mathcal{A}$ に対して, 以下は同値である:

(a) A_R は $\mathcal{A}$ 上の 1-set に対する最適な乱択アルゴリズムである.

(b) A_R は $\mathcal{A}$ 上の E^1-乱択アルゴリズムである.

次に, 我々は定理 3.6 に類似した最適乱択アルゴリズムに関する性質を導くことを試みた. 結果的には, 最適であることの方が E^1 であることより強い条件であることがわかった

定理 4.7. $\mathcal{A}$ を連結閉集合とし, A_R を $\mathcal{A}$ 上の乱択アルゴリズムとする. このとき, (1) $\Rightarrow$ (2) が成り立つ:

(1) A_R は $\mathcal{W}$ に対して最適である.

(2) A_R は 1-set に対して最適である.

したがって, A_R が $\mathcal{W}$ に対して最適であるならば, A_R は E^1-乱択アルゴリズムである.

証明. A_R は $\mathcal{W}$ に対して最適であるとする. このとき,

$$\max_{\omega \in \text{1-set}} C(A_R, \omega) \geq \min_{A'_R \in \mathcal{D}(\mathcal{A})} \max_{\omega \in \text{1-set}} C(A'_R, \omega)$$

は直ちにわかる。一方、

$$\begin{aligned}\max_{\omega\in\text{1-set}} C(A_R,\omega) &\le \max_{\omega\in\mathcal{W}} C(A_R,\omega)\\ &= \min_{A'_R\in\mathcal{D}(\mathcal{A})}\max_{\omega\in\mathcal{W}} C(A'_R,\omega)\\ &= \max_{d\in\mathcal{D}(\mathcal{W})}\min_{A\in\mathcal{A}} C(A,d) \quad (\text{定理 3.2 より})\\ &= \max_{d\in\mathcal{D}(\text{1-set})}\min_{A\in\mathcal{A}} C(A,d) \quad (\text{補題 3.10 より})\\ &= \min_{A'_R\in\mathcal{D}(\mathcal{A})}\max_{\omega\in\text{1-set}} C(A'_R,\omega) \quad (\text{定理 3.2 より})\end{aligned}$$

となる. よって, A_R は 1-set に対する最適な乱択アルゴリズムである。 □

指導教官である鈴木登志雄氏の指摘により, 定理 4.7 の逆は成り立たないことがわかった:

例 4.8. 高さ $h \ge 2$ の AND-OR 木を左部分木から先に探索する向き付きアルゴリズム全体を $\mathcal{A}^h_{\mathrm{dir},L}$ とする. このとき, $\mathcal{A}^h_{\mathrm{dir},L}$ 上の一様分布が定理 4.7 の逆の反例となる.

証明は省略する. 証明内では, 以下に述べる定理 4.10 を用いる.

最後に, 連結閉集合上の一様分布の性質を述べる.

定理 4.9 (鈴木-中村 2012, [4]). $\mathcal{A}$ が閉集合のとき, 以下が成り立つ:

$$P(T,\mathcal{W},\mathcal{A}_D) = P(T,\mathcal{W},\mathcal{A}).$$

すなわち,

$$\max_{d\in\mathcal{D}(\mathcal{W})}\min_{A\in\mathcal{A}_D} C(A,d) = \max_{d\in\mathcal{D}(\mathcal{W})}\min_{A\in\mathcal{A}} C(A,d).$$

鈴木登志雄氏の指摘により, この定理を応用して以下のことがわかった.

定理 4.10. $\mathcal{A}$ は連結閉集合であるとする. このとき, $\mathcal{A}$ 上の一様分布は $\mathcal{A}_D$ 上の最適な乱択アルゴリズムである. すなわち, A^u_R を $\mathcal{A}$ 上の一様分布とするとき,

$$\min_{A_R\in\mathcal{D}(\mathcal{A}_D)}\max_{\omega\in\mathcal{W}} C(A_R,\omega) = \max_{\omega\in\mathcal{W}} C(A^u_R,\omega)$$

が成り立つ.

証明. 定理 4.9 と定理 3.2 より,

$$\min_{A_R\in\mathcal{D}(\mathcal{A}_D)}\max_{\omega\in\mathcal{W}} C(A_R,\omega) = \min_{A_R\in\mathcal{D}(\mathcal{A})}\max_{\omega\in\mathcal{W}} C(A_R,\omega)$$

が成り立つ. これより, A_R^u を $\mathcal{A}$ 上の一様分布とするとき, A_R^u が $\mathcal{A}$ 上の最適なアルゴリズムであることを示せば十分であることがわかる.

まず, $\mathcal{W}$ を閉かつ連結な割り当ての集合に分割する. つまり,

$$\mathcal{W} = \Omega_1 \cup \cdots \cup \Omega_n,\ \Omega_i \cap \Omega_j = \varnothing (i \neq j),\ \text{各}\ \Omega_i\ \text{は閉かつ連結}$$

とする. このとき, 定理 4.4 より, 任意の $A_R \in \mathcal{D}(\mathcal{A})$ に対して,

$$\max_{\omega \in \Omega_j} C(A_R^u, \omega) \leq \max_{\omega \in \Omega_j} C(A_R, \omega)\ (j = 1, \ldots, n)$$

となる. よって,

$$\max_{\omega \in \Omega} C(A_R^u, \omega) \leq \max_{\omega \in \Omega} C(A_R, \omega)$$

となる. したがって, A_R^u は $\mathcal{A}$ 上の最適なアルゴリズムである. □

系 4.11. $\mathcal{A}_{\mathrm{dir}}$ 上の一様分布は $\mathcal{A}_D$ 上の最適な乱択アルゴリズムである. 一方, 最適な乱択アルゴリズムは $\mathcal{A}_{\mathrm{dir}}$ 上の確率分布であるとは限らない.

証明. $\mathcal{A}_{\mathrm{dir}}$ は閉かつ連結なので, 定理 4.10 より, $\mathcal{A}_{\mathrm{dir}}$ 上の一様分布は $\mathcal{A}_D$ 上の最適な乱択アルゴリズムである.

また, $A \in \mathcal{A}_D \setminus \mathcal{A}_{\mathrm{dir}}$ の生成する集合 $\langle A \rangle$ は閉かつ連結なので, この集合上の一様分布も $\mathcal{A}_D$ 上の最適な乱択アルゴリズムである. □

以上により, AND-OR 木に関する最適な乱択アルゴリズムは, 向き付きアルゴリズムからなる集合上の分布以外にも存在することが証明できた.

さて, 今後の課題として次のようなものが挙げられる:

(*a*) [4] では, 固有分布の個数も分析されている. そこで, 最適な乱択アルゴリズムの個数に関する結果を得られるであろうか?

(*b*) 本稿の全ての事柄に関して, 木の形が今回取り扱った完全二分木のような形のものだけではなく, 一般の木の場合でも同様のことがいえるのか?

(*a*) に関する予想として, 以下が挙げられる:

予想. $\mathcal{A}$ を連結閉集合とするとき, $\mathcal{A}$ 上の i-set に対する最適な乱択アルゴリズムは無限個存在する.

(*b*) に関しては, 不完全な木, または二分でない木に関して転置という概念をうまく定義することが不可欠であると思われる.

参考文献

[1] S.Arora, B.Barak, *Computational Complexity : A Modern Approach*, Cambridge university press, New York, 2009.

[2] C.Liu, K.Tanaka, *Eigen-Distribution on Random Assignments for Game Trees*, Information Processing Letters, vol.104(2), pp.73-77, 2007.

[3] M.Saks, A.Wigderson, *Probabilistic Boolean decision trees and the complexity of evaluating game trees*, in: Proc. 27th Annual IEEE Symposium on Foundations of Computer Science (FOCS), pp.29-38, 1986.

[4] T.Suzuki, R.Nakamura, *The eigen distribution of an AND-OR tree under directional algorithms*, IAENG International Journal of Applied Mathematics, vol.42(2), pp.122-128, 2012.

[5] A.C.-C.Yao, *Probabilistic computations: towards a unified measure of complexity*, in: Proc. 18th annual IEEE symposium on foundations of computer science (FOCS), pp.222-227, 1977.