The Uniqueness of Eigen-distribution under Non-directional Algorithms

Weiguang Peng, Shohei Okisaka, Wenjuan Li and Kazuyuki Tanaka

***Abstract*—Liu and Tanaka (2007) investigated the eigen-distribution, which achieves the distributional complexity, for uniform binary trees. In the present work, we extend their studies to balanced multi-branching trees. We show that an eigen-distibution is equivalent to E^i-distribution with respect to the closed set of all alpha-beta pruning algorithms. The proof is quite different from the uniform binary case given by Suzuki and Nakamura (2012). We also show that for any multi-branching tree, E^i-distribution is the unique eigen-distribution with respect to the set of all alpha-beta pruning algorithms.**

***Index Terms*—randomized complexity, alpha-beta pruning algorithms, balanced trees, uniform trees, AND-OR trees.**

I. Introduction

THIS study is a continuation of Liu and Tanaka [2] which investigated uniform binary AND-OR trees. We extend the study to a multi-branching case. By *balanced multi-branching*, we mean that all the nonterminal nodes at the same level have the same number of children and all paths from the root to the leaves are of the same length. It should be noted that the balancedness makes no restriction on the number of children for nodes at different levels. In this paper, we concentrate on $\mathcal{T}_n^h$, an n-branching tree with height h. We here notice that the argument for the uniform binary trees $\mathcal{T}_2^h$ cannot be generalized to $\mathcal{T}_n^h$ ($n > 2$) directly, since $\mathcal{T}_n^h$ inevitably corresponds to a non-uniform binary tree.

We quickly review the basics of game trees. An AND-OR tree (OR-AND tree, respectively) is a tree whose root is labeled AND (OR), and sequentially the internal nodes are level-by-level labeled by OR-node and AND-node (AND-node and OR-node) alternatively. Each probed leaf is assigned with Boolean value 0 or 1, via an assignment. By evaluating a tree, we are trying to compute the Boolean value of the root. We start from probing the leaves. Each leaf returns its value. The computation stops when we get enough information to evaluate the root value of the tree. The cost of computation is the number of the leaves that are queried during this computation, regardless of the remaining unqueried leaves.

An algorithm tells us how to proceed to evaluate a tree. The performance of algorithms makes a significant effect on the cost of computation. Among all these algorithms, alpha-beta pruning algorithm is known as one of the classical and effective algorithms [8] [7]. Knuth and Moore [4] conducted a detailed study on the alpha-beta pruning algorithm, which we briefly explain as follows. While evaluating an AND-node, if some child returns value 0, then the value of the AND-node is regarded as 0 without searching other children of this AND-node (which is known as an alpha-cut). On the other hand, when evaluating an OR-node, if some child returns value 1, then the value of the OR-node is recognized as 1 without searching other children of this OR-node (which is known as a beta-cut).

A randomized algorithm is a distribution over a family of deterministic algorithms. For a randomized algorithm, cost is computed as the expected cost over the corresponding family of deterministic algorithms. Yao's principle [1] indicates the relation between randomized complexity and distributional complexity as follows,

$$\underbrace{\min_{\mathbb{A}_R} \max_{\omega} cost(\mathbb{A}_R, \omega)}_{\text{Randomized complexity}} = \underbrace{\max_{d} \min_{\mathbb{A}_D} cost(\mathbb{A}_D, d)}_{\text{Distributional complexity}}.$$

where $\mathbb{A}_R$ ranges over randomized algorithms, ω ranges over assignments for leaves, d ranges over distributions on assignments and $\mathbb{A}_D$ ranges over deterministic algorithms. This result provides a new perspective to analyze randomized algorithms. Saks and Wigderson [9] showed that for any n-branching tree, the randomized complexity is $\Theta((\frac{n-1+\sqrt{n^2+14n+1}}{4})^h)$, where h is the height of tree.

Recently, several works have been done for uniform binary trees. Based on Saks and Wigderson [9], Liu and Tanaka [2] proposed the concept of eigen-distribution on assignments. They claimed that an eigen-distribution among the independent distributions (ID) is actually independently and identically distributed (IID). Suzuki and Niida [11] proved a stronger result by fixing the probability of root.

Liu and Tanaka [2] also introduced a reverse assigning technique to formulate sets of assignments for $\mathcal{T}_2^h$, namely 1-set and 0-set, in the case that assignments to leaves are correlated distributed (CD). They showed that E^1-distribution (a distribution on 1-set such that all deterministic algorithms have the same cost) is a unique eigen-distribution (the Liu-Tanaka Theorem). Suzuki and Nakamura [10] furthermore studied certain subsets of alpha-beta pruning algorithms on $\mathcal{T}_2^h$ and proved that the eigen-distribution with respect to a "closed" subset of alpha-beta pruning algorithms is unique, but for a set of directional algorithms, the uniqueness does not hold.

In this study, we proceed to balanced multi-branching trees. The remainder of this paper is organized as follows. In Section III, at first we give some technical lemmas to prove that the average cost on 1-set is larger than that on other closed sets. Based on these results, we show the relation between eigen-distribution and E^1-distribution for balanced multi-branching trees. In Section IV, we mainly show the uniqueness of eigen-distribution for the set of all alpha-beta pruning algorithms. The current paper is an extension of our conference talk [12].

Manuscript received July 5, 2016. This work was supported in part by the JSPS KAKENHI Grant Number JP26540001 and JP15H03634.

W. Peng (e-mail: pwgmath@gmail.com), S. Okisaka (e-mail: shohei.okisaka@gmail.com), W. Li (e-mail: sb2m701@math.tohoku.ac.jp) and K. Tanaka (e-mail: tanaka@math.tohoku.ac.jp) are with the Mathematical Institute, Tohoku University, Japan.

(Advance online publication: 27 August 2016)

II. Preliminary

For simplicity, we just consider n-branching trees, but all our results also hold for general balanced multi-branching trees.

In this study, we restrict ourselves to alpha-beta pruning algorithms. It should be noted that such a algorithm is both depth-first and deterministic. Depth-first means that when the algorithm evaluates the value of a certain node, it would not stop querying the leaves under this node until it knows the value of the node. An algorithm is directional if it queries the leaves in a fixed order, independent from the query history [6]. A typical directional algorithm SOLVE evaluates a tree from left to right [6]. If an algorithm proceeds depending on its query history, then we say it is non-directional. In this study, we denote $\mathcal{A}_D$ the set of all alpha-beta pruning algorithms, and $\mathcal{A}_{dir}$ the set of all directional algorithms.

First, we define a node-code for $\mathcal{T}_n^h$ as follows.

Definition 1 (Node-code)**.** Given a tree $\mathcal{T}_n^h$, a node-code is a finite sequence over $\{0, 1, \cdots, n-1\}$. An example of the node-code for $\mathcal{T}_3^2$ is illustrated in Fig. 1.

- The node-code of root is the empty sequence ε.
- For a non-terminal node with node-code v, the node-code for its n children are in the form of $v0, v1, \cdots, v(n-1)$ from left to right.

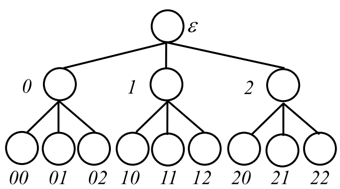

Fig. 1. Node-code for $\mathcal{T}_3^2$

We often identity "node" with "node-code".

Then the assignment for $\mathcal{T}_n^h$ is a function $\omega : \{0, 1, \cdots, n-1\}^h \rightarrow \{0, 1\}$. The set of assignments is denoted as $\Omega(\mathcal{T}_n^h)$. If $\mathcal{T}_n^h$ is clear from the context, then we just denote it as Ω.

Let $C(\mathbb{A}, \omega)$ denote the cost of an algorithm $\mathbb{A}$ under an assignment ω. Given a set of assignments Ω, d a distribution on Ω and $\mathbb{A} \in \mathcal{A}_D$, then the expected cost by $\mathbb{A}$ with respect to d is defined by $C(\mathbb{A}, d) = \sum_{\omega \in \Omega} d(\omega) \cdot C(\mathbb{A}, \omega)$. In fact, $C(\mathbb{A}, d)$ is the average cost if d is the uniform distribution on assignments.

The concept of "transposition" has been introduced to investigate $\mathcal{T}_2^h$ in [10]. We extend this notion to n-branching trees. To start with, we introduce the transposition of node.

Definition 2 (Transposition of node, an extension of Definition 4 in [10])**.** For $\mathcal{T}_n^h$, suppose u is an internal node. For $i < n$, by $\mathrm{tr}_i^u(v)$, we denote the i-th u-transposition of a node v in $\mathcal{T}_n^h$ (Fig. 2), which is defined as follows

- The 0-th u-transposition of v is itself, that is, $\mathrm{tr}_0^u(v) = v$.
- For $i \in \{1, \cdots, n-1\}$, $\mathrm{tr}_i^u(v)$ is defined by

$$\mathrm{tr}_i^u(v) = \begin{cases} u(i-1)s & \text{if } v = uis, \\ uis & \text{if } v = u(i-1)s, \\ v & \text{otherwise,} \end{cases}$$

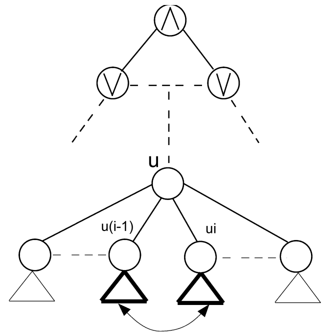

Fig. 2. Transposition of nodes under node u

where s is a finite sequence over $\{0, 1, \cdots, n-1\}$.

Definition 3 (Transposition of assignment)**.** For $\mathcal{T}_n^h$, suppose that u is an internal node, and ω an assignment. The i-th u-transposition of ω, denote $\mathrm{tr}_i^u(\omega)$, is defined by $\mathrm{tr}_i^u(\omega)(v) = \omega(\mathrm{tr}_i^u(v))$, where v is a leaf of $\mathcal{T}_n^h$.

Example 1. Fig. 3 shows an example of $\mathcal{T}_3^2$ with assignment $\omega = 000100111$. For transposition of node, if $u = 0$ and

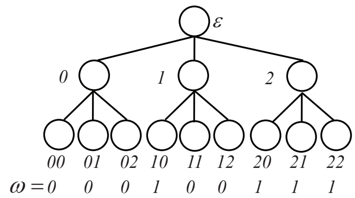

Fig. 3. An example of $\mathcal{T}_3^2$

$i = 1$, then $\mathrm{tr}_1^0(00) = 01$, $\mathrm{tr}_1^0(01) = 00$, and for other v, $\mathrm{tr}_1^0(v) = v$. For transposition of assignment, $\mathrm{tr}_2^{\varepsilon}(\omega) = 000111100$, and $\mathrm{tr}_1^1(\omega) = 000010111$.

Definition 4 (Transposition of algorithm)**.** For $\mathcal{T}_n^h$, suppose that u is an internal node, and $\mathbb{A}$ an algorithm in $\mathcal{A}_D$. For each assignment ω and the query history $(\alpha^1, \cdots, \alpha^m)$ of $(\mathbb{A}, \mathrm{tr}_i^u(\omega))$, the i-th u-transposition of $\mathbb{A}$, denote $\mathrm{tr}_i^u(\mathbb{A})$, has the query history $(\beta^1, \cdots, \beta^m)$ such that $\beta^j = \mathrm{tr}_i^u(\alpha^j)$ for each $j \leq m$.

Note that $C(\mathbb{A}, \mathrm{tr}_i^u(\omega)) = C(\mathrm{tr}_i^u(\mathbb{A}), \omega)$.

Definition 5 (Equivalent assignment class, closeness, connectness)**.** For $\mathcal{T}_n^h$, any assignments ω, ω', we denote $\omega \approx \omega'$ if $\omega' = \mathrm{tr}_i^u(\omega)$ for some u and i. An assignment ω is equivalent to ω' if there exists a sequence of assignments $\langle \omega_i \rangle_{i=1,\cdots,s}$ such that $\omega \approx \omega_1 \approx \cdots \approx \omega_s \approx \omega'$ for some $s \in \mathbb{N}$. Then we denote $[\![\omega]\!]$ as the equivalent assignment class of ω.

- A set Ω of assignments is closed if $\Omega = \bigcup_{\omega \in \Omega} [\![\omega]\!]$.
- A set Ω of assignments is connected if for any assignments $\omega, \omega' \in \Omega$, there exists a sequence of assignments $\langle \omega_i \rangle_{i=1,\cdots,s}$ in Ω such that $\omega \approx \omega_1 \approx \cdots \approx \omega_s \approx \omega'$.
- Given $\mathcal{A} \subseteq \mathcal{A}_D$, $\mathcal{A}$ is closed (under transposition) if for any $\mathbb{A} \in \mathcal{A}$, each internal node u and $i < n$, $\mathrm{tr}_i^u(\mathbb{A}) \in \mathcal{A}$.

Definition 6 (i-set for n-branching trees, adapted from [2])**.** Given $\mathcal{T}_n^h$, $i \in \{0, 1\}$, i-set consists of assignments such that

- the root has value i,
- if an AND-node has value 0 (or OR-node has value 1), just

one of its children has value 0 (1), and all the other $n-1$ children have 1 (0).

Note that i-set is closed and connected for $i \in \{0,1\}$.

Definition 7 (i'-set)**.** Given $\mathcal{T}_n^h$, $i \in \{0,1\}$, A closed set Ω of assignments is called an i'-set if it is not i-set and for any $\omega \in \Omega$, the root of the tree has value i with ω, which is denoted by $\omega(\varepsilon) = i$.

Definition 8 (E^i-distribution and eigen-distribution from [2])**.** Suppose $\mathcal{A}$ is a subset of $\mathcal{A}_D$ and Ω a set of assignments.
• A distribution d on i-set is called an E^i-distribution *w.r.t.* $\mathcal{A}$ if there exists $c \in \mathbb{R}$ such that for any $\mathbb{A} \in \mathcal{A}$, $C(\mathbb{A}, d) = c$.
• A distribution d on Ω is called an eigen-distribution *w.r.t.* $\mathcal{A}$ if for any distribution d' on Ω, $\min_{\mathbb{A}\in\mathcal{A}} C(\mathbb{A}, d) = \max_{d'} \min_{\mathbb{A}\in\mathcal{A}} C(\mathbb{A}, d')$ holds.

III. The Equivalence of Eigen-distribution and E^i-distribution for Balanced Multi-branching Trees

In this section, at first we show that any alpha-beta pruning algorithm has the same cost under the uniform distribution on a closed set of assignments, then give some technical lemmas to show that the average cost on 1-set is larger than the average cost on any i'-set. Based on these results, we investigate the equivalence of eigen-distribution and E^i-distribution for multi-branching trees. In the following sections, we denote $\mathcal{A}$ as a nonempty closed subset of $\mathcal{A}_D$.

Definition 9 (Definition 6 in [10])**.** Suppose that $p_1, \cdots, p_m$ are non-negative real numbers such that their sum is 1, $\Omega_1, \cdots, \Omega_m$ are disjoint non-empty subsets of assignments. We say that d is a distribution on $p_1\Omega_1 + \cdots + p_m\Omega_m$ if for each $1 \le j \le m$, there exists a distribution d_j on Ω_j such that $d = p_1 d_1 + \cdots + p_m d_m$.

For $\mathcal{T}_2^h$, Suzuki and Nakamura [10] applied a version of no-free-lunch theorem from [5] to study the equivalence of eigen-distribution and E^1-distribution. We can easily see that this theorem also works in the case of balanced multibranching trees as we state below.

Lemma 1. *Suppose $p_1, \cdots, p_m$ and $\Omega_1, \cdots, \Omega_m$ as in Definition 9. Assume that each Ω_j is connected. Then there exists $c \in \mathbb{R}$ such that for each distribution d on $p_1\Omega_1 + \cdots + p_m\Omega_m$, $\sum_{\mathbb{A}\in\mathcal{A}} C(\mathbb{A}, d) = c$ holds.*

Proof: See Lemma 1 in [10]. ■

Following is a technical lemma to show that for any closed subset of assignments with uniform distribution, all alpha-beta pruning algorithms have the same cost.

Lemma 2. *For any balanced multibranching tree $\mathcal{T}$, suppose $p_1, \cdots, p_m$ and $\Omega_1, \cdots, \Omega_m$ as in Definition 9 and moreover each Ω_j is closed. Let $d_{unif}(p_1\Omega_1 + \cdots + p_m\Omega_m)$ denote the distribution $p_1 d_1 + \cdots + p_m d_m$, where each d_j is the uniform distribution on Ω_j. Then there exists $c \in \mathbb{R}$ such that for any algorithm $\mathbb{A} \in \mathcal{A}_D$, $C(\mathbb{A}, d_{unif}(p_1\Omega_1 + \cdots + p_m\Omega_m)) = c$.*

Proof: To begin with, we handle the case $m = 1$. We prove by induction on height h.
• For case $h = 1$, let $\mathbb{A}$ be a directional algorithm. Then for any $i \in \{0, \cdots, n-1\}$, we have $C(tr_i^\varepsilon(\mathbb{A}), \omega) = C(\mathbb{A}, tr_i^\varepsilon(\omega))$. Since Ω_1 is closed, $\sum_{\omega\in\Omega_1} C(\mathrm{tr}_i^\varepsilon(\mathbb{A}), \omega) = \sum_{\omega\in\Omega_1} C(\mathbb{A}, \omega)$. Then $C(\mathbb{A}, d_{\text{unif}}(\Omega_1)) = C(\mathrm{tr}_i^\varepsilon(\mathbb{A}), d_{\text{unif}}(\Omega_1))$.
• For the induction step, we show the case $h+1$ by induction on the number n of children under the root of tree $\mathcal{T}$.

(1) For $n = 1$, the case for height $h+1$ can be reduced to the case of height h.

(2) For induction step, $\mathcal{T}$ is divided into $\mathcal{T}_0$ and $\mathcal{T}'$ as shown in Fig. 4, where $\mathcal{T}_0$ is the left-most subtree under the root, and $\mathcal{T}'$ denotes the rest part.

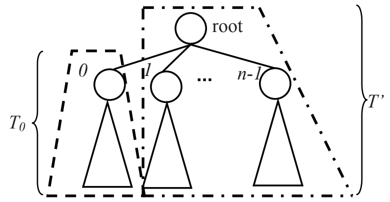

Fig. 4. An illustration of division of $\mathcal{T}$

Then Ω_1 can be represented by $\Omega_1 = \bigsqcup_{\omega_0\in W}\{\omega_0\} \times \Omega'_{\omega_0}$, where ω_0 is an assignment for the left-most subtree $\mathcal{T}_0$, W is a closed set of assignments for $\mathcal{T}_0$ and $\Omega'_{\omega_0} = \{\omega' : \omega_0\omega' \in \Omega\}$ is a closed set of assignments for $\mathcal{T}'$.

To compute $C(\mathbb{A}, d_{\text{unif}}(\Omega_1))$, we may assume that $\mathbb{A}$ evaluates $\mathcal{T}_0$ first since Ω_1 is closed. Then $C(\mathbb{A}, d_{\text{unif}}(\Omega_1))$ can be represented by

$$\frac{1}{|\Omega_1|} \cdot \sum_{\omega_0\in W} \sum_{\omega'\in\Omega'_{\omega_0}} \left[C(\mathbb{A}_0, \omega_0) + C(\mathbb{A}'_{\omega_0}, \omega')\right],$$

where $\mathbb{A}_0$ is an algorithm for $\mathcal{T}_0$, $\mathbb{A}'_{\omega_0}$ is an algorithm for $\mathcal{T}'$ which is applied after $\mathbb{A}$ evaluates the subtree $\mathcal{T}_0$ under the assignment ω_0. If the algorithm stops before $\mathbb{A}'_{\omega_0}$ starts, we set $C(\mathbb{A}'_{\omega_0}, \omega') = 0$ for each $\omega' \in \Omega'_{\omega_0}$.

Thus, $C(\mathbb{A}, d_{\text{unif}}(\Omega_1))$ can be computed as

$$\frac{1}{|\Omega_1|} \sum_{\omega_0\in W} \left[|\Omega'_{\omega_0}| C(\mathbb{A}_0, \omega_0) + \sum_{\omega'\in\Omega'_{\omega_0}} C(\mathbb{A}'_{\omega_0}, \omega') \right]. \quad (**)$$

It is observed that W can be partitioned as $W = W_1 \bigsqcup \cdots \bigsqcup W_k$ such that each W_j is closed and connected. Then for any ω, $\omega' \in W_j$, $\Omega'_\omega = \Omega'_{\omega'}$. So we let $a_j = |\Omega'_\omega|$ for $\omega \in W_j$.

Also by induction hypothesis in (2), we know that for any $\omega_0 \in W_j$, the value of $\sum_{\omega'\in\Omega'_{\omega_0}} C(\mathbb{A}'_{\omega_0}, \omega')$ is a constant or 0 and then we denote it by b_j. Thus, (**) can be replaced by

$$\begin{aligned} & \frac{1}{|\Omega_1|} \sum_{j=1}^{k} \sum_{\omega_0\in W_j} \left[a_j \cdot C(\mathbb{A}_0, \omega_0) + b_j\right] \\ = & \frac{1}{|\Omega_1|} \sum_{j=1}^{k} \left[a_j \cdot \sum_{\omega_0\in W_j} C(\mathbb{A}_0, \omega_0) + b_j |W_j| \right]. \end{aligned}$$

By induction hypothesis, $\sum_{\omega_0\in W_j} C(\mathbb{A}_0, \omega_0)$ is a constant, say e_j, when we fix some j. Therefore

$$C(\mathbb{A}, d_{\text{unif}}(\Omega_1)) = \frac{1}{|\Omega_1|} \sum_{j=1}^{k} \left[a_j \cdot e_j + b_j \cdot |W_j|\right].$$

This completes the proof for the case $m = 1$.

For the case $m > 1$, there exists c_i such that $C(\mathbb{A}, d_{\text{unif}}(\Omega_i)) = c_i$ for $1 \le i \le m$. It follows that $C(\mathbb{A}, d_{\text{unif}}(p_1\Omega_1 + \cdots + p_m\Omega_m)) = p_1c_1 + \cdots + p_mc_m$. ■

By our Lemma 2 and analogy to Lemma 2 in [10], we have

Lemma 3. *For any balanced multibranching tree $\mathcal{T}$, suppose that $p_1, \cdots, p_m$ and $\Omega_1, \cdots, \Omega_m$ as in Definition 9 and each Ω_j is closed and connected, d is a distribution on $p_1\Omega_1 + \cdots + p_m\Omega_m$. Then the following (i), (ii) and (iii) are equivalent:*

(i) $\min_{\mathbb{A}\in\mathcal{A}} C(\mathbb{A}, d) = \max_{d'} \min_{\mathbb{A}\in\mathcal{A}} C(\mathbb{A}, d')$, *where d' is a distribution on $p_1\Omega_1 + \cdots + p_m\Omega_m$.*

(ii) *There exists $c \in \mathbb{R}$ such that for any $\mathbb{A} \in \mathcal{A}$, $C(\mathbb{A}, d) = c$ holds.*

(iii) $\min_{\mathbb{A}\in\mathcal{A}} C(\mathbb{A}, d) = \sum_{1\le j\le m} p_j C(\mathbb{A}, d_{unif}(\Omega_j))$.

Proof: We first show that (i) is equivalent to the assertion

$$\min_{\mathbb{A}\in\mathcal{A}} C(\mathbb{A}, d) = \frac{1}{|\mathcal{A}|} \sum_{\mathbb{A}\in\mathcal{A}} C(\mathbb{A}, d). \tag{$\clubsuit$}$$

If we assume that $\min_{\mathbb{A}\in\mathcal{A}} C(\mathbb{A}, d) = \max_{d'} \min_{\mathbb{A}\in\mathcal{A}} C(\mathbb{A}, d')$, then $\min_{\mathbb{A}\in\mathcal{A}} C(\mathbb{A}, d) \ge \min_{\mathbb{A}\in\mathcal{A}} C(\mathbb{A}, d_{\text{unif}}) = C(\mathbb{A}, d_{\text{unif}})$. By Lemma 2 and Lemma 1, we have $C(\mathbb{A}, d_{\text{unif}}) = \frac{1}{|\mathcal{A}|} \sum_{\mathbb{A}\in\mathcal{A}} C(\mathbb{A}, d_{\text{unif}}) = \frac{1}{|\mathcal{A}|} \sum_{\mathbb{A}\in\mathcal{A}} C(\mathbb{A}, d)$.

Hence,

$$\min_{\mathbb{A}\in\mathcal{A}} C(\mathbb{A}, d) \ge \frac{1}{|\mathcal{A}|} \sum_{\mathbb{A}\in\mathcal{A}} C(\mathbb{A}, d).$$

Clearly, we have $\min_{\mathbb{A}\in\mathcal{A}} C(\mathbb{A}, d) \le \frac{1}{|\mathcal{A}|} \sum_{\mathbb{A}\in\mathcal{A}} C(\mathbb{A}, d)$, which implies $\min_{\mathbb{A}\in\mathcal{A}} C(\mathbb{A}, d) = \frac{1}{|\mathcal{A}|} \sum_{\mathbb{A}\in\mathcal{A}} C(\mathbb{A}, d)$.

For the other direction, by Lemma 1, for any distribution d', $\sum_{\mathbb{A}\in\mathcal{A}} C(\mathbb{A}, d) = \sum_{\mathbb{A}\in\mathcal{A}} C(\mathbb{A}, d')$. Thus, $\sum_{\mathbb{A}\in\mathcal{A}} C(\mathbb{A}, d') \le \frac{1}{|\mathcal{A}|} \sum_{\mathbb{A}\in\mathcal{A}} C(\mathbb{A}, d)$ holds. Since $\min_{\mathbb{A}\in\mathcal{A}} C(\mathbb{A}, d) = \frac{1}{|\mathcal{A}|} \sum_{\mathbb{A}\in\mathcal{A}} C(\mathbb{A}, d)$, we have $\min_{\mathbb{A}\in\mathcal{A}} C(\mathbb{A}, d') \le \min_{\mathbb{A}\in\mathcal{A}} C(\mathbb{A}, d)$, i.e., $\min_{\mathbb{A}\in\mathcal{A}} C(\mathbb{A}, d) = \max_{d'} \min_{\mathbb{A}\in\mathcal{A}} C(\mathbb{A}, d)$.
Since (ii) is equivalent to ($\clubsuit$), the assertion (i) and (ii) are equivalent.

Next, we investigate the equivalence of assertion (iii) and ($\clubsuit$).
Since $\sum_{\mathbb{A}\in\mathcal{A}} C(\mathbb{A}, d) = \sum_{\mathbb{A}\in\mathcal{A}} C(\mathbb{A}, d_{\text{unif}}) = |\mathcal{A}|\, C(\mathbb{A}, d_{\text{unif}})$, we have

$$\min_{\mathbb{A}\in\mathcal{A}} C(\mathbb{A}, d_{\text{unif}}) = \frac{1}{|\mathcal{A}|} \sum_{\mathbb{A}\in\mathcal{A}} C(\mathbb{A}, d).$$

Thus, ($\clubsuit$) is equivalent to the following: $\min_{\mathbb{A}\in\mathcal{A}} C(\mathbb{A}, d) = \min_{\mathbb{A}\in\mathcal{A}} C(\mathbb{A}, d_{\text{unif}}) = \sum_{1\le j\le m} p_j C(\mathbb{A}, d_{\text{unif}}(\Omega_j))$. ■

Our goal of this section is to investigate the relation of eigen-distribution and E^1- distribution *w.r.t.* $\mathcal{A}$. To show this, we first need to consider the relation between average cost over 1-set and average cost over any other closed sets. We start with the base case of height 2, and then extend to general height h.

Part I: The case for height 2

In this part, we will investigate the relation between average cost on i-set for $i \in \{0, 1\}$ and the average cost on any i'-set for AND-OR trees $\mathcal{T}_n^2$. Since by Lemma 2, the average cost does not depend on an algorithm, we may only consider SOLVE. For simplicity, we denote $C(\omega) = C(\text{SOLVE}, \omega)$ and $C(\Omega) = C(\mathbb{A}, d_{\text{unif}}(\Omega))$, where $\mathbb{A} \in \mathcal{A}_D$ and Ω is closed.

Recall that the costs over 0-set and 1-set have been studied in [3].

Theorem 1 (Theorem 7 in [3]). *For any tree $\mathcal{T}_n^2$,*

$$C(0\text{-}set) = \frac{n^2 + 4n - 1}{4} \text{ and } C(1\text{-}set) = \frac{n(n+1)}{2}.$$

Then we show that

Lemma 4. *Given an AND-OR tree $\mathcal{T}_n^2$, for any connected $1'$-set Ω, $C(\Omega) < C(1\text{-}set)$.*

Proof: We can find an assignment in Ω in the form of $\omega = 0^{a_0}1^{b_0} \cdots 0^{a_{n-1}}1^{b_{n-1}}$ where for each $i < n$, $a_i + b_i = n$.

Let $M = \max\{C(\omega) : \omega \in \Omega\}$. Since Ω is closed and connected, we can show $M = n + \sum_{i=0}^{n-1} a_i$. We claim that

$$C(\Omega) \le \frac{M + n}{2}. \tag{$\star$}$$

The inequality ($\star$) implies that $C(\Omega) < C(1\text{-set})$ because $M < n^2$ and $C(1\text{-set}) = \frac{n^2+n}{2}$ (by Theorem 1).

To show ($\star$), we denote the reverse order of an assignment ω by ω^R. For example, if $\omega = 100110011$, $\omega^R = 110011001$. Since Ω is closed,

$$\text{the map } \omega \mapsto \omega^R \text{ is a bijection on } \Omega. \tag{$\ddagger$}$$

Moreover it is easy to show $C(\omega) + C(\omega^R) \le M + n$ for any $\omega \in \Omega$.

By ($\ddagger$), we have $C(\Omega) = \frac{\sum_{\omega\in\Omega} C(\omega)}{|\Omega|} = \frac{\sum_{\omega\in\Omega} C(\omega) + \sum_{\omega\in\Omega} C(\omega^R)}{2|\Omega|}$.

Thus, $C(\Omega) \le \frac{(M+n)|\Omega|}{2|\Omega|} = \frac{M+n}{2}$. ■

Lemma 5. *If $\Omega = \Omega_1 \sqcup \cdots \sqcup \Omega_k$, where each Ω_i is closed and pairwise disjoint, then $C(\Omega) = \sum_{i=1}^{k} \frac{|\Omega_i|}{|\Omega|} C(\Omega_i)$.*

Proof: Since each Ω_i is closed and pairwise disjoint, we have

$$C(\Omega) = \frac{\sum_{\omega\in\Omega} C(\omega)}{|\Omega|} = \frac{1}{|\Omega|}\left(\sum_{i=1}^{k} \sum_{\omega\in\Omega_i} C(\omega)\right)$$

$$= \left(\sum_{i=1}^{k} \frac{|\Omega_i|}{|\Omega|} \frac{\sum_{\omega\in\Omega_i} C(\omega)}{|\Omega_i|}\right)$$

$$= \sum_{i=1}^{k} \frac{|\Omega_i|}{|\Omega|} C(\Omega_i).$$

■

Since any closed set Ω can be represented as $\Omega = \Omega_1 \sqcup \cdots \sqcup \Omega_k$ where each Ω_i for $i \in \{1, \cdots, k\}$ is closed, connected and pairwise disjoint. By Lemma 4 and 5, we get the following theorem.

Theorem 2. *Given an AND-OR tree $\mathcal{T}_n^2$, for any $1'$-set Ω, $C(1\text{-}set) > C(\Omega)$.*

Given sets of assignments $\langle\Omega_i\rangle_{0\le i\le n-1}$, we define $\Omega_0 \times \cdots \times \Omega_{n-1} = \{\omega_0 \cdots \omega_{n-1} : \omega_i \in \Omega_i \text{ for } i < n\}$. For any assignment ω of any $0'$-set, we represent $\omega = \omega_0 \cdots \omega_{n-1}$, where each ω_i is the assignment of i-th subtree. We denote ω_ℓ as the first ω_i such that $\omega_i = 0^n$ and ω_L as the last ω_i such that $\omega_i = 0^n$ in ω.

Thus for $\omega \in \Omega_0 \times \cdots \times \Omega_{n-1}$ such that $\omega(\varepsilon) = 0$, we have $C(\text{SOLVE}, \omega) = \sum_{i=0}^{\ell} C(\text{SOLVE}, \omega_i)$. That is, the problem of computing $C(\text{SOLVE}, \omega)$ turns into searching for the first 0^n-segment that appears in ω.

Lemma 6. *Given an AND-OR tree $\mathcal{T}_n^2$, for any connected $0'$-set Ω, $C(\Omega) < C(0\text{-set})$.*

Proof: For $\omega \in \Omega$, let $\omega = \omega_0 \cdots \omega_{n-1}$. First, if ω_i is in the form of $\omega_i = 0^{a_i} 1 u_i$, the reverse order of ω_i can be denoted as $\omega_i^R = 0^{b_i} 1 v_i$ where $a_i + b_i \leq n-1$, u_i and v_i sequence over $\{0, 1\}$. Otherwise, $\omega_i^R = \omega_i = 0^n$.

We denote $\omega' = {\omega_0}^R \cdots {\omega_{n-1}}^R$, $\omega'' = (\omega')^R$ and $\omega''' = \omega^R$. Since the tree is an AND-OR tree of height 2 and $\omega(\varepsilon) = 0$, the computation for ω will stop immediately after it finds the first 0^n-segment in ω. Then, we have

$$C(\omega) = \sum_{i<\ell} a_i + \ell + n,$$

where the first 0^n-segment appears in ω_ℓ, $\sum_{i<\ell} a_i$ counts the number of 0's that has been searched in the form of $0^{a_i} 1 u_i$ before ω_ℓ, ℓ counts the number of 1's that has been searched in the form of $0^{a_i} 1 u_i$ before ω_ℓ and n is the cost of ω_ℓ.

Through the same approach, we can compute

$$C(\omega') = \textstyle\sum_{i<\ell} b_i + \ell + n,$$

$$C(\omega'') = \textstyle\sum_{i>L} a_i + (n - L - 1) + n$$

$$C(\omega''') = \textstyle\sum_{i>L} b_i + (n - L - 1) + n.$$

Here denote $\widetilde{C}(\omega) = C(\omega) + C(\omega') + C(\omega'') + C(\omega''')$.

Then, $\widetilde{C}(\omega) = \sum\limits_{i \notin [\ell, L]} (a_i + b_i) + 2[n - (L - \ell) - 1] + 4n$.

Since $a_i + b_i \leq n - 1$ for each i, we have

$$\sum_{i \notin [\ell, L]} (a_i + b_i) \leq (n-1)[n - (L - \ell) - 1]. \tag{1}$$

Thus,

$$\widetilde{C}(\omega) \leq (n - (L - \ell) - 1) \cdot (n + 1) + 4n \leq n^2 + 4n - 1. \tag{2}$$

Since Ω is an $0'$-set, either (1) or (2) is strict.

Then $C(\Omega) = \frac{1}{4|\Omega|} \sum\limits_{\omega \in \Omega} \widetilde{C}(\omega) < \frac{n^2 + 4n - 1}{4} = C(0\text{-set})$. ∎

By Lemma 5 and 6, we obtain the relation between average cost on the 0-set and any $0'$-set.

Theorem 3. *Given an AND-OR tree $\mathcal{T}_n^2$, for any $0'$-set Ω, $C(0\text{-set}) > C(\Omega)$.*

Using similar proof idea in Lemma 6, we can also obtain that for an OR-AND tree $\mathcal{T}_n^2$, for any $1'$-set Ω, $C(1\text{-set}) > C(\Omega)$ holds. Also, the proof idea in Lemma 4 can be applied to show that for an OR-AND tree $\mathcal{T}_n^2$, for any $0'$-set Ω, $C(0\text{-set}) > C(\Omega)$ holds. Hence, we can get a more general statement as below.

Theorem 4. *Given $\mathcal{T}_n^2$ which can be either AND-OR tree or OR-AND tree, for any i'-set Ω, $C(i\text{-set}) > C(\Omega)$.*

Part II: The general case for height h

In this part, we extend the study to height $h \geq 2$. To simplify the notation, throughout the rest part, we denote $C(i\text{-set})$ by $C_i^{\wedge,h}$ ($C_i^{\vee,h}$, respectively) for AND-OR tree (OR-AND tree, respectively) of height h. For any i'-set Ω, we denote $C(\Omega)$ by $C_\Omega^{\wedge,h}$ ($C_\Omega^{\vee,h}$, respectively) for AND-OR tree (OR-AND tree, respectively) of height h. Let $i\text{-set}(\wedge, h)$ denote i-set for AND-OR tree $\mathcal{T}_n^h$ and $i\text{-set}(\vee, h)$ denote the i-set for OR-AND tree $\mathcal{T}_n^h$. The following lemma will be used in the proof of next lemma.

Lemma 7. *Given $\Gamma = \{0, \cdots, N-1\}$, N, $n \in \mathbb{N}$, let $\Psi(k)$ be the total number of k that appears in all elements of Γ^n. Then $\Psi(k) = n \cdot N^{n-1}$*

Proof: For any two different elements k, j of Γ, we have $\Psi(k) = \Psi(j)$. Then $\sum_{k=0}^{N-1} \Psi(k) = N \cdot \Psi(k)$. Moreover, we know that $\sum_{k=0}^{N-1} \Psi(k) = n \cdot \mid \Gamma^n \mid = n \cdot N^n$. As a result, $\Psi(k) = n \cdot N^{n-1}$. ∎

Note that for any AND-OR (OR-AND) tree $\mathcal{T}_n^{h+1}$, we can easily get n OR-AND (AND-OR) subtrees $\mathcal{T}_n^h$ under the root of $\mathcal{T}_n^{h+1}$. The following lemma shows the relation of cost between them.

Lemma 8. $C_1^{\wedge,h+1} = nC_1^{\vee,h}$, $C_0^{\wedge,h+1} = C_0^{\vee,h} + \frac{n-1}{2} C_1^{\vee,h}$, $C_1^{\vee,h+1} = C_1^{\wedge,h} + \frac{n-1}{2} C_0^{\wedge,h}$, *and* $C_0^{\vee,h+1} = nC_0^{\wedge,h}$.

Proof: Since all i-sets are closed, we can fix an algorithm as SOLVE. $0\text{-set}(\wedge, h+1)$ can be represent as $0\text{-set}(\wedge, h+1) = \bigsqcup\limits_{k=0}^{n-1} \Omega_k$, where $\Omega_k = (1\text{-set}(\vee, h))^k \times 0\text{-set}(\vee, h) \times (1\text{-set}(\vee, h))^{n-(k+1)}$, and $1\text{-set}(\wedge, h+1) = (1\text{-set}(\vee, h))^n$. Let $m_0 = |0\text{-set}(\vee, h)|$ and $m_1 = |1\text{-set}(\vee, h)|$.

$$C_0^{\wedge,h+1} = \frac{\sum\limits_{\omega \in 0\text{-set}(\wedge, h+1)} C(\omega)}{\mid 0\text{-set}(\wedge, h+1) \mid} = \frac{\sum\limits_{k=0}^{n-1} \sum\limits_{\omega \in \Omega_k} C(\omega)}{n \cdot m_0 \cdot (m_1)^{n-1}}$$

$$= \underbrace{\frac{\sum\limits_{k=0}^{n-1} \sum\limits_{\omega_0 \cdots \omega_{n-1} \in \Omega_k} \sum\limits_{i<k} C(\omega_i)}{n \cdot m_0 \cdot (m_1)^{n-1}}}_{(a)} + \underbrace{\frac{\sum\limits_{k=0}^{n-1} \sum\limits_{\omega_0 \cdots \omega_{n-1} \in \Omega_k} C(\omega_k)}{n \cdot m_0 \cdot (m_1)^{n-1}}}_{(b)}$$

Fix ω_k and any $\omega_{i \neq k} \in 1\text{-set}(\vee, h)$, the number of ω_k that appears in all assignments of Ω_k is $|1\text{-set}(\vee, h)|^{n-1}$. Thus,

$$(b) = \frac{\sum\limits_{k=0}^{n-1} \sum\limits_{\omega \in 0\text{-set}(\vee, h)} (m_1)^{n-1} \cdot C(\omega)}{n \cdot m_0 \cdot (m_1)^{n-1}} = \frac{\sum\limits_{\omega \in 0\text{-set}(\vee, h)} C(\omega)}{m_0}.$$

By Lemma 7, (a) can be calculated as

$$\frac{\sum\limits_{k=1}^{n-1} \sum\limits_{\omega \in 1\text{-set}(\vee, h)} m_0 \cdot (m_1)^{n-(k+1)} \cdot k \cdot (m_1)^{k-1} C(\omega)}{n \cdot m_0 \cdot (m_1)^{n-1}}$$

$$= \frac{m_0 \cdot (m_1)^{n-2} \cdot \sum\limits_{k=1}^{n-1} k \cdot \sum\limits_{\omega \in 1\text{-set}(\vee, h)} C(\omega)}{n \cdot m_0 \cdot (m_1)^{n-1}}$$

$$= \frac{n-1}{2m_1} \cdot \sum_{\omega \in 1\text{-set}(\vee, h)} C(\omega).$$

Thus,

$$C_0^{\wedge,h+1} = (a)+(b) = C_0^{\vee,h} + \frac{n-1}{2}C_1^{\vee,h}.$$

Next, we show $C_1^{\wedge,h+1} = nC_1^{\vee,h}$.

$$C_1^{\wedge,h+1} = \frac{1}{\mid \text{1-set}(\wedge,h+1)\mid}\sum_{\omega\in\text{1-set}(\wedge,h+1)} C(\omega)$$

$$= \frac{1}{\mid \text{1-set}(\wedge,h+1)\mid}\sum_{\omega_0\cdots\omega_{n-1}\in(\text{1-set}(\vee,h))^n}\sum_{i=0}^{n-1} C(\omega_i).$$

By Lemma 7, we have

$$C_1^{\wedge,h+1} = \frac{1}{{m_1}^n}\sum_{\omega\in\text{1-set}(\vee,h)} n\mid \text{1-set}(\vee,h)\mid^{n-1} C(\omega)$$

$$= \frac{\sum_{\omega\in\text{1-set}(\vee,h)} C(\omega)\cdot n}{m_1} = nC_1^{\vee,h}.$$

In the same way, we can calculate $C_1^{\vee,h+1} = C_1^{\wedge,h} + \frac{n-1}{2}C_0^{\wedge,h}$ and $C_0^{\vee,h+1} = nC_0^{\wedge,h}$. ■

Theorem 5. *For any i'-set Ω, $C_i^{\wedge,h} > C_\Omega^{\wedge,h}$ and $C_i^{\vee,h} > C_\Omega^{\vee,h}$.*

Proof: Since Ω is closed, we can fix an algorithm as SOLVE. We show this by induction on height h. By Theorem 4, the base case $h=2$ holds.

For the induction step, let $\Omega = \Omega_1 \sqcup \cdots \sqcup \Omega_k$, where for each $i \in \{1,\cdots,k\}$, $\Omega_i = [\![\omega_i^0]\!] \times \cdots \times [\![\omega_i^{n-1}]\!]$ and ω_i^j is an assignment of the j-th subtree under the root of $\mathcal{T}_n^h$.

- First, we show $C_1^{\wedge,h+1} > C_\Omega^{\wedge,h+1}$, where Ω is a $1'$-set.

$$C_\Omega^{\wedge,h+1} = \frac{1}{\mid\Omega\mid}\sum_{\omega\in\Omega} C(\omega) = \frac{1}{\mid\Omega\mid}\sum_{i=1}^{k}\sum_{\omega\in\Omega_i} C(\omega)$$

$$= \frac{1}{\mid\Omega\mid}\sum_{i=1}^{k}\sum_{v_0\in[\![\omega_i^{n-1}]\!]}\cdots\sum_{v_{\text{n-1}}\in[\![\omega_i^{n-1}]\!]}[C(v_0)+\cdots+C(v_{\text{n-1}})]$$

$$= \frac{1}{\mid\Omega\mid}\sum_{i=1}^{k}\left[\sum_{m=0}^{n-1}\left[\prod_{j\neq m}|[\![\omega_i^j]\!]|\sum_{v_m\in[\![\omega_i^m]\!]} C(v_m)\right]\right]$$

$$= \frac{1}{\mid\Omega\mid}\sum_{i=1}^{k}\sum_{m=0}^{n-1}\mid\Omega_i\mid C_{[\![\omega_i^m]\!]}^{\vee,h} = \sum_{i=1}^{k}\frac{\mid\Omega_i\mid}{\mid\Omega\mid}\sum_{m=0}^{n-1} C_{[\![\omega_i^m]\!]}^{\vee,h}.$$

By induction hypothesis, $C_{[\![\omega_i^m]\!]}^{\vee,h} < C_1^{\vee,h}$. Thus, $C_\Omega^{\wedge,h+1} < \sum_{i=1}^{k}\frac{|\Omega_i|}{|\Omega|}\sum_{m=0}^{n-1} C_1^{\vee,h} = nC_1^{\vee,h} = C_1^{\wedge,h+1}$.

By the same way, we show that $C_0^{\vee,h+1} > C_\Omega^{\vee,h+1}$.

- Next, we show $C_0^{\wedge,h+1} > C_\Omega^{\wedge,h+1}$, where Ω is a $0'$-set.

For $\omega = \omega_0\cdots\omega_{n-1}$ of $\mathcal{T}_n^{h+1}$, we denote $\widetilde{\omega} = \omega_{n-1}\cdots\omega_0$. Similar with Lemma 6, let ℓ (L, respectively) denotes the minimum (maximum) number such that ω_ℓ (ω_L) assigns 0 to all the leaves of ℓ-th (L-th) subtree under the root. Then $C_\Omega^{\wedge,h+1}$ can be computed by

$$\frac{1}{\mid\Omega\mid}\sum_{\omega\in\Omega} C(\omega) = \frac{1}{2\mid\Omega\mid}\sum_{\omega\in\Omega}[C(\omega)+C(\widetilde{\omega})]$$

$$= \frac{1}{2\mid\Omega\mid}\sum_{i=1}^{k}\sum_{\omega_i^0\in[\![\omega_i^0]\!]}\cdots\sum_{\omega_i^{n-1}\in[\![\omega_i^{n-1}]\!]}\Big[C(\omega_i^0)+\cdots+ C(\omega_i^\ell)+C(\omega_i^L)+\cdots+C(\omega_i^{n-1})\Big].$$

Since $\sum_{\omega_i^0\in[\![\omega_i^0]\!]}\cdots\sum_{\omega_i^{n-1}\in[\![\omega_i^{n-1}]\!]} C(\omega_i^j) = \mid\Omega_i\mid C_{[\![\omega_i^j]\!]}^{\vee,h}$, we can compute $C_\Omega^{\wedge,h+1}$ by $\frac{1}{2|\Omega|}\sum_{i=1}^{k}\mid\Omega_i\mid\Big[C_{[\![\omega_i^0]\!]}^{\vee,h}+\cdots+ C_{[\![\omega_i^\ell]\!]}^{\vee,h}+C_{[\![\omega_i^L]\!]}^{\vee,h}+\cdots+C_{[\![\omega_i^{n-1}]\!]}^{\vee,h}\Big]$.

By induction hypothesis,

$$C_\Omega^{\wedge,h+1} < \frac{\sum_{i=1}^{k}\mid\Omega_i\mid\Big[\ell C_1^{\vee,h}+2C_0^{\vee,h}+(n-L-1)C_1^{\vee,h}\Big]}{2\mid\Omega\mid}$$

$$\leq \frac{1}{2\mid\Omega\mid}\sum_{i=1}^{k}\mid\Omega_i\mid\Big[(n-1)C_1^{\vee,h}+2C_0^{\vee,h}\Big]$$

$$= \frac{n-1}{2}C_1^{\vee,h}+C_0^{\vee,h} = C_0^{\wedge,h+1}.$$

In the same way, we can show that $C_1^{\vee,h+1} > C_\Omega^{\vee,h+1}$. ■

Theorem 6. *For any $\mathcal{T}_n^h$, $C_1^{\wedge,h} > C_0^{\wedge,h}$.*

Proof: We show that for $h \geq 1$,

$$C_1^{\wedge,h} = C_0^{\vee,h},\ C_1^{\vee,h} = C_0^{\wedge,h} \text{ and } \frac{n+1}{2}C_1^{\vee,h} > C_0^{\vee,h},\ (\spadesuit)$$

which implies $C_1^{\wedge,h+1} > C_0^{\wedge,h+1}$ by Lemma 8.

We prove ($\spadesuit$) by induction on height h. For $h=1$, $C_1^{\wedge,1} = C_0^{\vee,1} = n$, $C_1^{\vee,1} = C_0^{\wedge,1} = \frac{n}{2}$.

For the induction step, the first two equalities follows from Lemma 8 and $\frac{n+1}{2}C_1^{\vee,h+1} = \frac{n+1}{2}C_1^{\wedge,h} + \frac{n^2-1}{4}C_0^{\wedge,h} > \frac{(n+1)^2}{4}C_0^{\wedge,h} > nC_0^{\wedge,h} = C_0^{\vee,h+1}$. ■

By Theorem 5 and 6, we have the following theorem.

Theorem 7. *For an AND-OR tree $\mathcal{T}_n^h$, any closed but not 1-set Ω, $C(\text{1-set}) > C(\Omega)$.*

By Lemma 3 and Theorem 7, we can show that

Lemma 9. *For an AND-OR tree $\mathcal{T}_n^h$ and d an eigen-distribution* w.r.t. *$\mathcal{A}$, then d is a distribution on the 1-set.*

Proof: Suppose for an AND-OR tree $\mathcal{T}_n^h$, and d is an eigen-distribution *w.r.t.* $\mathcal{A}$. Let $\langle\Omega_j\rangle_{1\leq j\leq m}$ be a partition of all assignments such that each Ω_j is connected and closed.

Without loss of generality, let Ω_1 be the 1-set. For each j, p_j denotes the probability for Ω_j under the distribution of d. By Theorem 7, for each $j>1$, $C(\Omega_1) > C(\Omega_j)$. By Lemma 3,

$$\min_{\mathbb{A}\in\mathcal{A}} C(\mathbb{A},d) = p_1 C(\Omega_1) + \sum_{j=2}^{m} p_j C(\Omega_j).$$

Since d is an eigen-distribution *w.r.t.* $\mathcal{A}$, $p_1 = 1$ and for each $j \in \{2,\cdots,m\}$, $p_j = 0$. Thus, d is a distribution on the 1-set. ■

Theorem 8. *Assume an AND-OR tree $\mathcal{T}_n^h$, d is a probability distribution on the assignments, $\mathcal{A}$ is a closed subset of $\mathcal{A}_D$. Then the following two conditions are equivalent.*

a) *d is an eigen-distribution* w.r.t. *$\mathcal{A}$.*
b) *d is an E^1-distribution* w.r.t. *$\mathcal{A}$.*

Proof: By Lemma 9, d is an eigen-distribution on 1-set. Thus the equivalence holds by Lemma 3. ■

Remark 1. *(1) For the case OR-AND tree, eigen-distribution is equivalent to E^0-distribution* w.r.t. $\mathcal{A}$.
(2) The above remark and Theorem 8 also hold for balanced multi-branching trees.

IV. Eigen-distribution *w.r.t* $\mathcal{A}_D$ is unique

To start with, we investigate the relation of E^i-distribution and uniform distribution for n-branching trees. By Lemma 2 and Theorem 8, we can show the following

Corollary 1. *For any AND-OR tree $\mathcal{T}_n^h$ and closed subset $\mathcal{A} \subseteq \mathcal{A}_D$, the uniform distribution on 1-set is an eigen-distribution* w.r.t. $\mathcal{A}$.

From this section, we also consider non-directional algorithms, which play an important role to investigate the uniqueness of eigen-distribution. While a deterministic algorithm $\mathbb{A}$ works, the order of searching leaves may depend on the query history. If so, $\mathbb{A}$ is called a non-directional algorithm. We first provide an example of a such algorithm.

Example 2. Given a tree $\mathcal{T}_3^2$, where each leaf is labeled from left to right as shown in Fig.5. Let $\mathbb{A}$ be a directional algorithm on $\mathcal{T}_3^2$ denoted as $\mathbf{123456789}$, it means the algorithm evaluates the leaves from left to right. We can define a non-directional algorithm $\mathbb{A}'$ denoted as $\widehat{\mathbf{123}}\underline{\mathbf{456789}}$, where the order of searching leaves depends on the query history as follows

- if $\omega(00) = 1$, then the algorithm continues as the searching order $\mathbf{789456}$;
- otherwise, the algorithm continues from left to right as the searching order $\mathbf{456789}$.

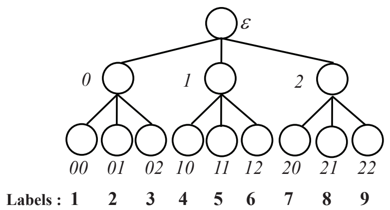

Fig. 5. $\mathcal{T}_3^2$ with label on leaves

Next we show the uniqueness of eigen-distribution *w.r.t* $\mathcal{A}_D$. We start with the base case of height 2.

Theorem 9. *For any AND-OR tree $\mathcal{T}_n^2$, E^1-distribution* w.r.t. *$\mathcal{A}_D$ is uniform.*

Proof: For simplicity, we also consider $\mathcal{T}_3^2$ as shown in Fig.5. Let d be an E^1-distribution for $\mathcal{T}_3^2$ such that the probability of d being an assignment ω of 1-set is $d(\omega)$. Suppose $\omega_1 = 001001001$, $\omega_2 = 001001010$ with probability $p_1 = d(\omega_1)$ and $p_2 = d(\omega_2)$. We start with showing that $p_1 = p_2$.

We consider a directional algorithm $\mathbb{A}$ denoted as $\mathbf{123456789}$, and a non-directional algorithm $\mathbb{A}'$ denoted as $\widehat{\mathbf{1234567}}\underline{\mathbf{89}}$, which probes the left-most two subtrees with label $\mathbf{123456}$, and then the algorithm proceeds as follows

- if the cost of evaluating the left-most two subtrees is 6, it exchanges the searching order of $\mathbf{8}$ and $\mathbf{9}$;
- otherwise, it continues as in $\mathbb{A}$.

Thus, if the assignment for $\mathcal{T}_3^2$ is in the form $001001\omega'$, where $\omega' \in \{001, 010, 100\}$, then the right-most subtree is searched as $\mathbf{798}$, otherwise $\mathbf{789}$.

Then we have

$$C(\mathbb{A}, d) = C(\mathbb{A}, \omega_1)p_1 + C(\mathbb{A}, \omega_2)p_2 + \cdots = 9p_1 + 8p_2 + \underbrace{\cdots}_{r_1}$$

$$C(\mathbb{A}', d) = C(\mathbb{A}', \omega_1)p_1 + C(\mathbb{A}', \omega_2)p_2 + \cdots = 8p_1 + 9p_2 + \underbrace{\cdots}_{r_2}$$

Using the two given algorithms $\mathbb{A}$ and $\mathbb{A}'$, the values of r_1 and r_2 are equal. Since d is an E^1-distribution, $C(\mathbb{A}, d) = C(\mathbb{A}', d)$. Thus $p_1 = p_2$. By the same argument, we can show that for any assignments ω and ω', $d(\omega) = d(\omega') = \frac{1}{27}$.

The general case $\mathcal{T}_n^2$ can be treated similarly. ∎

Using the same approach in Theorem 9, we can show that

Corollary 2. *For any OR-AND tree $\mathcal{T}_n^2$, E^0-distribution* w.r.t. *$\mathcal{A}_D$ is uniform.*

Theorem 10. *For any AND-OR tree $\mathcal{T}_n^2$, E^0-distribution* w.r.t. *$\mathcal{A}_D$ is uniform.*

Proof: For simplicity, we consider $\mathcal{T}_3^2$ again. Let d be an E^0-distribution for $\mathcal{T}_3^2$. We partition 0-set as $\Omega_1 \sqcup \Omega_2 \sqcup \Omega_3$, where for $i \in \{1, 2, 3\}$, Ω_i is the collection of assignments such that 000 is assigned to the i-th subtree of $\mathcal{T}_3^2$ under the root. By the same method in Theorem 9, we can show that all the assignments in Ω_i have the same probability and we denote it as p_i for $i \in \{1, 2, 3\}$.

For any $\mathbb{A} \in \mathcal{A}_D$, $C(\mathbb{A}, d) = \sum_{i=1}^{3} \sum_{\omega \in \Omega_i} p_i \cdot C(A, \omega)$.

We consider a directional algorithm $\mathbb{A}$ denoted as $\mathbf{123456789}$ and a non-directional algorithm $\mathbb{A}'$ denoted as $\widehat{\mathbf{123}}\underline{\mathbf{456789}}$, which first evaluates the subtree with label $\mathbf{123}$, and then it proceeds as follows

- if the assignment for the subtree with label $\mathbf{123}$ is 000, it continues as in $\mathbb{A}$;
- otherwise, it exchanges the searching order of the subtrees with label $\mathbf{456}$ and $\mathbf{789}$.

Then, we have

$$C(\mathbb{A}, d) = \sum_{i=1}^{3} \sum_{\omega \in \Omega_i} p_i \cdot C(\mathbb{A}, \omega),$$

$$C(\mathbb{A}', d) = \sum_{i=1}^{3} \sum_{\omega \in \Omega_i} p_i \cdot C(\mathbb{A}', \omega).$$

By algorithms $\mathbb{A}$ and $\mathbb{A}'$, we can calculate that

$$\sum_{\omega \in \Omega_2} C(\mathbb{A}, \omega) = \sum_{\omega \in \Omega_3} C(\mathbb{A}', \omega),$$

$$\sum_{\omega \in \Omega_3} C(\mathbb{A}, \omega) = \sum_{\omega \in \Omega_2} C(\mathbb{A}', \omega),$$

$$\sum_{\omega \in \Omega_1} C(\mathbb{A}, \omega) = \sum_{\omega \in \Omega_1} C(\mathbb{A}', \omega),$$

$$45 = \sum_{\omega \in \Omega_2} C(\mathbb{A}, \omega) \neq \sum_{\omega \in \Omega_3} C(\mathbb{A}, \omega) = 63.$$

Therefore we have $p_2 = p_3$. When we consider $\mathbb{A}$ denoted as $\mathbf{789123456}$ and $\mathbb{A}'$ denoted as $\widehat{\mathbf{789}}\underline{\mathbf{123456}}$, if the right-most subtree is assigned 000, it is same as $\mathbb{A}$, otherwise we change the query order of $\mathbf{123}$ and $\mathbf{456}$. We can show that $p_1 = p_2$. So each assignment has the same probability.

In the same way, we can show that the E^0-distribution *w.r.t.* $\mathcal{A}_D$ is also uniform for general case. ∎

Using the same approach in Theorem 10, we have the following corollary.

Corollary 3. *For any OR-AND tree $\mathcal{T}_n^2$, E^1-distribution* w.r.t. *$\mathcal{A}_D$ is uniform.*

By induction on the height of the tree, we can show the following theorem

Theorem 11. *For any tree* $\mathcal{T}_n^h$, E^i*-distribution* w.r.t. $\mathcal{A}_D$ *is uniform. Thus eigen-distribution* w.r.t. $\mathcal{A}_D$ *is unique.*

V. Conclusion

This study extended the Liu-Tanaka Theorem to balanced multi-branching trees. Although, for convenience, we just treat n-branching trees, all the theorems in this paper also hold for balanced multi-branching trees. We showed that for any balanced multi-branching tree and a probability distribution d on all assignments, the following three conditions are equivalent: an eigen-distribution, an E^i-distribution and the uniform distribution on the i-set *w.r.t.* $\mathcal{A}_D$.

Saks and Wigderson [9] proved that for $\mathcal{T}_2^h$, the distributional complexity is equal to $\max\limits_{d} \min\limits_{\mathbb{A} \in \mathcal{A}_{dir}} C(\mathbb{A}, d)$. Suzuki and Nakamura [10] remarked that it is indeed equal to $\max\limits_{d} \min\limits_{\mathbb{A} \in \mathcal{A}} C(\mathbb{A}, d)$ for any closed set of all alpha-beta pruning algorithms on $\mathcal{T}_2^h$. Similarly, using our arguments in Part III, we can conclude that the equality still holds for any balanced multi-branching tree.

Acknowledgement

The authors would like to express sincere appreciations to Prof. Chenguang Liu (Northwestern Polytechnical University, China), Prof. Yue Yang (National University of Singapore, Singapore), Prof. KengMeng Ng (Nanyang Technological University, Singapore) and Prof. Toshio Suzuki (Tokyo Metropolitan University, Japan) for their valuable discussions.

References

[1] A.C. C. Yao, "Probabilistic computations: toward a unified measure of complexity," in: *Proc. 18th Annual IEEE Symposium on Foundations of Computer Science*, pp. 222-227, 1977.

[2] C.G. Liu and K. Tanaka, "Eigen-distribution on random assignments for game trees," *Information Processing Letters*, vol. 104, no. 2, pp. 73-77, 2007.

[3] C.G. Liu and K. Tanaka, "The computational complexity of game trees by eigen-distribution," *Combinatorial Optimization and Applications*, Springer Berlin Heidelberg, pp. 323-334, 2007.

[4] D.E. Knuth and R. W. Moore, "An analysis of alpha-beta pruning," *Artificial Intelligence*, vol. 6, no. 4, pp. 293-326, 1975.

[5] D.H. Wolpert and W. G. MacReady, "No-free-lunch theorems for search," *Technical Report SFI-TR-95-02-010*, Santa Fe Institute, 1995.

[6] J. Pearl, "Asymptotic properties of minimax trees and game-searching procedures," *Artificial Intelligence*, vol. 14, no. 2, pp. 113-138, 1980.

[7] J. Pearl, "The solution for the branching factor of the alpha-beta pruning algorithm and its optimality," *Communications of the ACM*, vol. 25, no. 8, pp. 559-564, 1982.

[8] M. Tarsi, "Optimal search on some game trees," *Journal of the ACM*, vol. 30, no. 3, pp. 389-396, 1983.

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

[10] T. Suzuki and R. Nakamura, "The eigen distribution of an AND-OR tree under directional algorithms," *IAENG International Journal of Applied Mathematics*, vol. 42, no. 2, pp. 122-128, 2012.

[11] T. Suzuki and Y. Niida, "Equilibrium points of an AND-OR tree: under constraints on probability," *Annals of Pure and Applied Logic*, vol. 166, no. 11, pp. 1150-1164, 2015.

[12] W. Peng, S. Okisaka, W. Li, and K. Tanaka, "The Eigen-distribution for multi-branching trees," *Lecture Notes in Engineering and Computer Science: Proceedings of The International MultiConference of Engineers and Computer Scientists 2016*, IMECS 2016, 16-18 March, 2016, Hong Kong, pp. 88-93.