Proceedings of the International MultiConference of Engineers and Computer Scientists 2016 Vol I,
IMECS 2016, March 16 - 18, 2016, Hong Kong

# The Eigen-distribution for Multi-branching Trees

Weiguang Peng, Shohei Okisaka, Wenjuan Li and Kazuyuki Tanaka

***Abstract*—In the present work, we extend the studies on eigen-distribution for uniform binary trees to balanced multi-branching trees. We show that for such general trees, an eigen-distribution is still equivalent to $E^i$-distribution with respect to alpha-beta pruning algorithms and the uniqueness of eigen-distribution holds, although the uniqueness fails if we are restricted to directional 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 root to leaf 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. Because of the page limit of this paper, we mostly 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 except for leaves. Each 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. The cost of computation is the number of leaves that are queried during the computation, regardless of the remaining unqueried leaves.

An algorithm tells 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 [1] [5]. In this paper, we only consider alpha-beta pruning algorithms.

A randomized algorithm is a distribution over a family of deterministic algorithms. For a randomized algorithm, cost is computed as the average cost over the corresponding family of deterministic algorithms. Yao's principle [10] 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, $\omega$ 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 [6] 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 [6], 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 [8] 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 [7] furthermore studied certain subsets of deterministic algorithms on $\mathcal{T}_2^h$ and proved that the eigen-distribution *w.r.t.* a "closed" subset of alpha-beta pruning algorithms is unique, but for a set of directional algorithms, it is not unique.

In this study, we proceed to balanced multi-branching case. In Section III, we investigate the relation between eigen-distribution and $E^i$-distribution for multi-branching trees. In Section IV, we mainly show that the uniqueness of eigen-distribution holds for the set of alpha-beta pruning algorithms, although the uniqueness does not hold for the set of directional algorithms.

## II. Preliminary

For simplicity, we just consider $n$-branching trees, but most of 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 [4]. A typical directional algorithm SOLVE evaluates a tree from left to right [4]. 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.

Manuscript received December 7, 2015; revised January 14, 2016. This work was supported in part by the Grants-in-Aid for Science Research (Japan): No. 2654001 and No. 15H03634.

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.

ISBN: 978-988-19253-8-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

**Definition 1** (Node-code)**.** Given a tree $\mathcal{T}_n^h$, a node-code is a finite sequence over $\{0, 1, \cdots, n-1\}$.

- The node-code of root is the empty sequence $\varepsilon$.
- For a nonterminal 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.

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

Then the assignment for $\mathcal{T}_n^h$ is a function $\omega : \{0, 1, \cdots, n-1\}^h \to \{0, 1\}$. 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 $\Omega$.

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

The concept of "transposition" has been introduced to investigate $\mathcal{T}_2^h$ in [7]. We extended 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 [7])**.** 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 $\mathcal{T}_n^h$, 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

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

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, $\omega$ is an assignment. The $i$-th $u$-transposition of $\omega$, 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. 1 shows an example of $\mathcal{T}_3^2$ with assignment $\omega = 000100111$. For transposition of node, if $u = 0$ and

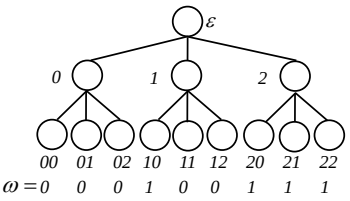


Fig. 1. An example of $\mathcal{T}_3^2$

$i = 1$, 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 $\omega$ 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 $j \le m$.

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 $\omega$, $\omega'$, we denote $\omega \approx \omega'$ if $\omega' = \mathrm{tr}_i^u(\omega)$ for some $u$, $i$. An assignment $\omega$ is equivalent to $\omega'$ 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 $\omega$.
• A set $\Omega$ of assignments is closed if $\Omega = \bigcup_{\omega \in \Omega} [\![\omega]\!]$.
• A set $\Omega$ 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 $\Omega$ 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, $i'$-set)**.** Given $\mathcal{T}_n^h$, $i \in \{0, 1\}$,
• $i^*$-set is the set of all assignments $\omega$ such that $\omega(\varepsilon) = i$ and $\omega \notin i$-set.
• A closed set $\Omega$ of assignments is called an $i'$-set if it is not $i$-set and for any $\omega \in \Omega$, $\omega(\varepsilon) = i$.

**Definition 8** ($E^i$-distribution from [2])**.** Suppose $\mathcal{A}$ is a subset of $\mathcal{A}_D$. 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 $\mathbb{A} \in \mathcal{A}$, $C(\mathbb{A}, d) = c$.

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

In this section, at first we show that any alpha-beta pruning algorithm on a closed set of assignments with uniform distribution (i.e., the same probability) has the same cost, 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 [7])**.** 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 $\Omega_j$ such that $d = p_1 d_1 + \cdots + p_m d_m$.

For $\mathcal{T}_2^h$, Suzuki and Nakamura [7] applied a version of no-free-lunch theorem from [9] to study the equivalence of eigen-distribution and $E^1$-distribution. We can easily see that this theorem also works in the case of $n$-branching trees as we state below.

**Lemma 1.** *For $\mathcal{T}_n^h$, suppose $p_1, \cdots, p_m$ and $\Omega_1, \cdots, \Omega_m$ as in Definition 9. Assume that each $\Omega_j$ 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 [7]. ■

ISBN: 978-988-19253-8-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

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 $\mathcal{T}_n^h$, suppose $p_1, \cdots, p_m$ and $\Omega_1, \cdots, \Omega_m$ as in Definition 9 and moreover each $\Omega_j$ 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 $\Omega_j$. 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 this case by induction on height $h$.

• For case $h = 1$. Since $\Omega_1$ 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 $\mathcal{T}$, which is obtained from $\mathcal{T}_n^h$ by cutting off some subtrees connecting the root.

(1) For $n = 1$, it is obvious.

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

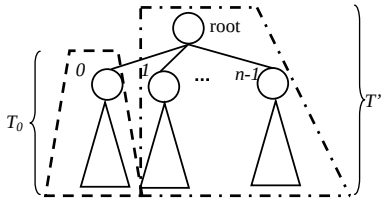


Fig. 2. An illustration of division of $\mathcal{T}$

Then $\Omega_1$ can be represented by $\Omega_1 = \bigsqcup_{\omega_0 \in W} \{\omega_0\} \times \Omega'_{\omega_0}$(disjoint union), where $\omega_0$ 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 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 $\Omega_1$ 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 $\omega_0$. 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 $\omega$, $\omega' \in W_j$, $\Omega'_{\omega} = \Omega'_{\omega'}$. So we let $a_j = |\ \Omega'_{\omega}\ |$ for $\omega \in W_j$. Also by induction hypothesis in (2), we know that for any $\omega_0 \in W_j$, $\sum_{\omega' \in \Omega'_{\omega_0}} C(\mathbb{A}'_{\omega_0}, \omega')$ is a constant or 0 and then we denote it by $b_j$. Thus, (**) can be replaced by

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

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

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

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 [7], we have

**Lemma 3.** *For any $\mathcal{T}_n^h$, suppose that $p_1, \cdots, p_m$ and $\Omega_1, \cdots, \Omega_m$ as in Definition 9 and each $\Omega_j$ 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)* $\min_{\mathbb{A} \in \mathcal{A}} C(\mathbb{A}, d) = \sum_{m \ge j \ge 1} p_j C(\mathbb{A},\ d_{unif}(\Omega_j))$

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 cost over any 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 only consider AND-OR trees $\mathcal{T}_n^2$.

**Definition 10** (The corresponding subset of $1^*$-set )**.** For an AND-OR tree $\mathcal{T}_n^2$ and any $\omega \in$ 1-set, $\omega$ can be represented in the form of

$$\overbrace{0\cdots0}^{n}\underbrace{1\overbrace{0\cdots0}^{n}}_{a_0\quad b_0}\cdots\underbrace{\overbrace{0\cdots0}^{n}1\overbrace{0\cdots0}^{n}}_{a_i\quad b_i}\cdots\underbrace{\overbrace{0\cdots0}^{n}1\overbrace{0\cdots0}^{n}}_{a_{n-1}\quad b_{n-1}}.$$

Then the corresponding subset of $1^*$-set for $\omega$ is defined by

$$\Lambda_\omega = \{0^{a_0}1u_0 0^{a_1}1u_1 \cdots 0^{a_{n-1}}1u_{n-1} : u_i \in \{0,1\}^{b_i}\} \backslash \{\omega\}$$

Since by Lemma 2, the average cost does not depend on an algorithm, we may only consider SOLVE. By the definition of SOLVE, we can directly get the following lemma.

**Lemma 4.** *For any $\omega$, $\omega' \in$1-set, if $C(SOLVE, \omega) = C(SOLVE, \omega')$, then $\sum_{i=0}^{n-1} b_i = \sum_{i=0}^{n-1} b'_i$ and $|\ \Lambda_\omega\ | = |\ \Lambda_{\omega'}\ |$.*

For any $\omega \in$ 1-set, if $C(\text{SOLVE}, \omega) = k$, then we have $|\ \Lambda_\omega\ | = 2^{n^2-k} - 1$. We can write $1^*$-set $= \bigsqcup_{n \le k \le n^2} \bigsqcup_{\substack{\omega \in \text{1-set}, \\ C(\text{SOLVE}, \omega) = k}} \Lambda_\omega$. Thus, for any $\mathbb{A} \in \mathcal{A}_D$, $|\ 1^*\text{-set}\ | = \sum_{n \le k \le n^2} |\ \{\omega \in \text{1-set} : C(\mathbb{A}, \omega) = k\}\ | \cdot (2^{n^2-k} - 1)$.

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 $\Omega$ is closed.

Following is a key technical lemma for the next theorem.

**Lemma 5.** *For any non-negative integers $a_1, \cdots, a_n$, $b_1, \cdots, b_n$ and $c_1, \cdots, c_n$, if $b_1 > b_2 > \cdots > b_n$ and $c_1 < c_2 < \cdots < c_n$, then $\frac{\sum_{k=1}^{n} a_k \cdot c_k}{\sum_{k=1}^{n} a_k} > \frac{\sum_{k=1}^{n} a_k \cdot b_k \cdot c_k}{\sum_{k=1}^{n} a_k \cdot b_k}$.*

*Proof:* It is enough to show that

$$(\sum_{k=1}^{n} a_k \cdot c_k)(\sum_{k=1}^{n} a_k \cdot b_k) - (\sum_{k=1}^{n} a_k \cdot b_k \cdot c_k)(\sum_{k=1}^{n} a_k) > 0 \quad (*)$$

ISBN: 978-988-19253-8-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

Left side of $(*) = \sum_{k,l=1}^{n} c_k \cdot a_k \cdot a_l \cdot b_l - \sum_{k,l=1}^{n} c_k \cdot a_k \cdot b_k \cdot a_l$
$= \sum_{1\le k<l\le n} c_k \cdot a_k \cdot a_l(b_l - b_k) - \sum_{1\le k<l\le n} c_l \cdot a_k \cdot a_l(b_l - b_k)$
$= \sum_{1\le k<l\le n} (c_k - c_l) \cdot a_k \cdot a_l \cdot (b_l - b_k)$
Since $c_k - c_l < 0$ and $b_l - b_k < 0$, $(*)$ holds. ■

**Theorem 1.** *$C(1^*\text{-set}) < C(1\text{-set})$.*

*Proof:* Since $1^*$-set and 1-set are closed, we fix the algorithm as SOLVE. By the construction of 1-set and $1^*$-set, for any assignment $\omega$ in 1-set expect $\hat{\omega} = \underbrace{\underbrace{0\cdots0}_{n-1}1\cdots\underbrace{0\cdots0}_{n-1}1}_{n\times n}$, there exists at least one assignment $\omega'$ of $1^*$-set such that $C(\omega) = C(\omega')$. By Lemma 4, for those assignments of 1-set that have the same cost *w.r.t.* SOLVE, their corresponding subsets of $1^*$-set are of the same cardinality. Thus, we compute the average cost on $1^*$-set by

$$C(1^*\text{-set}) = \frac{1}{|\ 1^*\text{-set}\ |}\sum_{\omega\in 1^*\text{-set}} C(\omega)$$
$$= \frac{\sum_{k=n}^{n^2} k\cdot |\ \{\omega \in \text{1-set} : C(\omega)=k\}\ |\cdot(2^{n^2-k}-1)}{\sum_{k=n}^{n^2} |\ \{\omega \in \text{1-set} : C(\omega)=k\}\ |\cdot(2^{n^2-k}-1)}.$$

The average cost on 1-set can be calculated by

$$C(\text{1-set}) = \frac{\sum_{k=n}^{n^2} k\cdot |\ \{\omega \in \text{1-set} : C(\omega)=k\}\ |}{\sum_{k=n}^{n^2} |\ \{\omega \in \text{1-set} : C(\omega)=k\}\ |}.$$

Thus by Lemma 5, we show that $C(1^*\text{-set}) < C(\text{1-set})$ ■

Furthermore, we provide a new method to show the relation between average cost on $i$-set for $i \in \{0,1\}$ and the average cost on any $i'$-set. Recall that the costs over 0-set and 1-set were studied in [3].

**Theorem 2** (Theorem 7 in [3])**.**
*$C(0\text{-set}) = \frac{n^2+4n-1}{4}, C(1\text{-set}) = \frac{n(n+1)}{2}$.*

**Lemma 6.** *For any connected $1'$-set $\Omega$, $C(\Omega) < C(1\text{-set})$.*

*Proof:* We can find an assignment in $\Omega$ 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 $\Omega$ is closed and connected, we can show $M = n + \sum_{i=0}^{n-1} a_i$. We claim that

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

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

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

$$\omega \mapsto \omega^R \text{ is a bijection on } \Omega. \qquad (\ddagger)$$

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

By Lemma 6 and 7, we get the following theorem.

**Theorem 3.** *For any $1'$-set $\Omega$, $C(1\text{-set}) > C(\Omega)$.*

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 $\omega$ of any $0'$-set, we represent $\omega = \omega_0\cdots\omega_{n-1}$, where each $\omega_i$ is the assignment of $i$-th subtree. We denote $\omega_\ell$ as the first $\omega_i$ such that $\omega_i = 0^n$ and $\omega_L$ as the last $\omega_i$ such that $\omega_i = 0^n$ in $\omega$.

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 $\omega$.

**Lemma 8.** *For any connected $0'$-set $\Omega$, $C(\Omega) < C(0\text{-set})$.*

*Proof:* For $\omega \in \Omega$, let $\omega = \omega_0\cdots\omega_{n-1}$. First, if $\omega_i$ is in the form of $\omega_i = 0^{a_i}1u_i$, the reverse order of $\omega_i$ can be denoted as $\omega_i^R = 0^{b_i}1v_i$ where $a_i + b_i \le n-1$, $u_i$ 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 $\omega$ will stop immediately after it finds the first $0^n$-segment in $\omega$. Then, we have

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

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

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

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

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

Thus,

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

Since $\Omega$ is an $0'$-set, we have $\widetilde{C}(\omega) < n^2 + 4n - 1$.

Thus $C(\Omega) = \frac{1}{4|\Omega|}\sum_{\omega\in\Omega} \widetilde{C}(\omega) < \frac{n^2+4n-1}{4} = C(\text{0-set})$. ■

By Lemma 7 and 8, we obtain the relation between average cost on the 0-set and any $0'$-set.

**Theorem 4.** *For any $0'$-set $\Omega$, $C(0\text{-set}) > C(\Omega)$.*

Using similar proof ideas in Theorem 3 and 4, we can also show the relations for OR-AND trees. Hence, we can get a more general statement as follows.

**Theorem 5.** *Given $\mathcal{T}_n^2$ which can be either AND-OR tree or OR-AND tree, for any $i'$-set $\Omega$, $C(i\text{-set}) > C(\Omega)$.*

*Part II: The general case for height $h$*

In this part, we extend the study to height $h \ge 2$. 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 trees (OR-AND trees, respectively) of height $h$. For any $i'$-set $\Omega$,

ISBN: 978-988-19253-8-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

we denote $C(\Omega)$ by $C_\Omega^{\wedge,h}$ ($C_\Omega^{\vee,h}$, respectively) for AND-OR trees (OR-AND trees, respectively) of height $h$. Let $i$-set$(\wedge,h)$ denote $i$-set for AND-OR trees $\mathcal{T}_n^h$ and $i$-set$(\vee,h)$ denote the $i$-set for OR-AND trees $\mathcal{T}_n^h$.

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

*Proof:* We fix the algorithm as SOLVE. 0-set$(\wedge,h+1)$ can be represent as 0-set$(\wedge,h+1)= \bigsqcup_{k=0}^{n-1} \Omega_k$ such that $\Omega_k = (1\text{-set}(\vee,h))^k \times 0\text{-set}(\vee,h) \times (1\text{-set}(\vee,h))^{n-(k+1)}$. Let $m_0 = |0\text{-set}(\vee,h)|$ and $m_1 = |1\text{-set}(\vee,h)|$.

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

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

Since $\omega_k \in 0\text{-set}(\vee,h)$, $\omega_i \in 1\text{-set}(\vee,h)$, $i\neq k$,

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

Also (a) can be calculated as $\frac{n-1}{2m_1}\cdot \sum_{\omega\in 1\text{-set}(\vee,h)} C(\omega)$. Thus,

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

In the same way, we can get other equalities. ■

**Theorem 6.** *For $i'$-set $\Omega$, $C_i^{\wedge,h} > C_\Omega^{\wedge,h}$ and $C_i^{\vee,h} > C_\Omega^{\vee,h}$.*

*Proof:* Since $\Omega$ is closed, we can fix the algorithm as SOLVE. We show this by induction on height $h$. By Theorem 5, the base case $h=2$ holds.

For $h>2$, 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 $\omega_i^j$ 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 $\Omega$ is a $1'$-set.

$$C_\Omega^{\wedge,h+1} = \tfrac{1}{|\Omega|}\sum_{\omega\in\Omega} C(\omega) = \tfrac{1}{|\Omega|}\sum_{i=1}^{k}\sum_{\omega\in\Omega_i} C(\omega)$$
$$= \tfrac{1}{|\Omega|}\sum_{i=1}^{k}\left[\sum_{m=0}^{n-1}\prod_{j\neq m}|[\![\omega_i^j]\!]|\sum_{w_i^j\in[\![\omega_i^j]\!]} C(w_i^j)\right]$$
$$= \tfrac{1}{|\Omega|}\sum_{i=1}^{k}\sum_{m=0}^{n-1}|\ \Omega_i\ |\ C_{[\![\omega_i^m]\!]}^{\vee,h} = \sum_{i=1}^{k}\tfrac{|\Omega_i|}{|\Omega|}\sum_{m=0}^{n-1} C_{[\![\omega_i^m]\!]}^{\vee,h}.$$

By induction hypothesis, $C_{[\![\omega_i^m]\!]}^{\vee,h} < C_1^{\vee,h}$. Thus,

$$C_\Omega^{\wedge,h+1} < \sum_{i=1}^{k}\tfrac{|\Omega_i|}{|\Omega|}\sum_{m=0}^{n-1} C_1^{\vee,h} = nC_1^{\vee,h} = C_1^{\wedge,h+1}.$$

• Next, we show $C_0^{\wedge,h+1} > C_\Omega^{\wedge,h+1}$, where $\Omega$ 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 8, let $\ell$ ($L$, respectively) denotes the minimum (maximum) number such that $\omega_\ell$ ($\omega_L$, respectively) assigns 0 to all the leaves of $\ell$-th ($L$-th) subtree under the root. Then $C_\Omega^{\wedge,h+1}$ can be computed by

$$\tfrac{1}{|\Omega|}\sum_{\omega\in\Omega} C(\omega) = \tfrac{1}{2|\Omega|}\sum_{\omega\in\Omega}[C(\omega)+C(\widetilde{\omega})]$$
$$= \tfrac{1}{2|\Omega|}\sum_{i=1}^{k}|\ \Omega_i\ |\ \left[C_{[\![\omega_i^0]\!]}^{\vee,h}+\cdots+C_{[\![\omega_i^\ell]\!]}^{\vee,h}+C_{[\![\omega_i^L]\!]}^{\vee,h}+\cdots+C_{[\![\omega_i^{n-1}]\!]}^{\vee,h}\right].$$

By induction hypothesis,

$$C_\Omega^{\wedge,h+1} < \frac{\sum_{i=1}^{k}|\Omega_i|\left[\ell C_1^{\vee,h}+2C_0^{\vee,h}+(n-L-1)C_1^{\vee,h}\right]}{2|\Omega|}$$
$$\leq \tfrac{1}{2|\Omega|}\sum_{i=1}^{k}|\ \Omega_i\ |\ \left[(n-1)C_1^{\vee,h}+2C_0^{\vee,h}\right]$$
$$= \tfrac{n-1}{2}C_1^{\vee,h}+C_0^{\vee,h} = C_0^{\wedge,h+1}.$$ ■

**Theorem 7.** *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 9.

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 9 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 6 and 7, we have the following theorem.

**Theorem 8.** *For an AND-OR tree $\mathcal{T}_n^h$, any closed but not 1-set $\Omega$, $C(1\text{-set}) > C(\Omega)$.*

By Lemma 3 and Theorem 8, we can easily show that

**Lemma 10.** *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.*

**Theorem 9.** *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 10, $d$ 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 9 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 Theorem 9, and an argument similar to Theorem 7 in [7], we can show the following

**Corollary 1.** *For any tree $\mathcal{T}_n^h$, there are uncountably many eigen-distributions* w.r.t. *$\mathcal{A}_{dir}$.*

Next we show the uniqueness of eigen-distribution *w.r.t* $\mathcal{A}_D$.

**Theorem 10.** *For any AND-OR tree $\mathcal{T}_n^2$, $E^1$-distribution* w.r.t. *$\mathcal{A}_D$ is unique.*

ISBN: 978-988-19253-8-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

*Proof:* For simplicity, we consider $\mathcal{T}_3^2$ as shown in Fig.3.

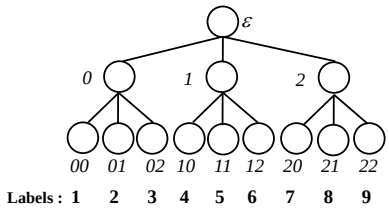


Fig. 3. $\mathcal{T}_3^2$ with label on leaves

Let $d$ be an $E^1$-distribution for $\mathcal{T}_3^2$. 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 $\overbrace{\mathbf{123456}}\,\underline{\mathbf{789}}$, 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 $\omega$ and $\omega'$, $d(\omega) = d(\omega') = \frac{1}{27}$.

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

**Remark 2.** *We can also show that for any OR-AND tree $\mathcal{T}_n^2$, $E^0$-distribution* w.r.t. *$\mathcal{A}_D$ is unique.*

**Theorem 11.** *For any AND-OR tree $\mathcal{T}_n^2$, $E^0$-distribution* w.r.t. *$\mathcal{A}_D$ is unique.*

*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 $\Omega_1 \bigsqcup \Omega_2 \bigsqcup \Omega_3$, where for $i \in \{1, 2, 3\}$, $\Omega_i$ 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 10, we can show that all the assignments in $\Omega_i$ 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 $\overbrace{\mathbf{123}}\,\underline{\mathbf{456789}}$, 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 $p_2 = p_3$.

By the same argument, we can show $p_1 = p_2$. ■

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

By induction, we can show that

**Theorem 12.** *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. We showed that for any $\mathcal{T}_n^h$ and a probability distribution $d$ on all assignments, the followings 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 [6] proved that for $\mathcal{T}_2^h$, the distributional complexity is equal to $\max_d \min_{\mathbb{A} \in \mathcal{A}_{dir}} C(\mathbb{A}, \omega)$. Suzuki and Nakamura [7] remarked that it is indeed equal to $\max_d \min_{\mathbb{A} \in \mathcal{A}} C(\mathbb{A}, \omega)$ for any closed set of 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.

## VI. Acknowledgment

The authors would like to express sincere appreciations to Prof. C.G. Liu (NPU, China), Prof. Y. Yang (NUS, Singapore), Prof. K.M. Ng (NTU, Singapore) and Prof. T. Suzuki (TMU, Japan) for their valuable discussions.

## References

[1] D. E. Knuth and R. W. Moore, "An analysis of alpha-beta pruning," *Artificial Intelligence*, vol. 6, no. 4, pp. 293-326, 1975.
[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] J. Pearl, "Asymptotic properties of minimax trees and game-searching procedures," *Artificial Intelligence*, vol. 14, no. 2, pp. 113-138, 1980.
[5] 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.
[6] 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.
[7] 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.
[8] 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.
[9] D. H. Wolpert and W. G. MacReady, "No-free-lunch theorems for search," *Technical Report SFI-TR-95-02-010*, Santa Fe Institute, 1995.
[10] A. C. C. Yao, "Probabilistic computations: toward a unified measure of complexity," *Proc. 18th Annual IEEE Symposium on Foundations of Computer Science*, pp. 222-227, 1977.