Weakly balanced multi-branching AND-OR trees: Reconstruction of the omitted part of Saks-Wigderson (1986)

Ryoya Kurita, Taira Shimizu, Toshio Suzuki[*]
Tokyo Metropolitan University

July 1, 2022

Abstract

We investigate variants of the Nash equilibrium for query complexity of Boolean functions. We reconstruct some omitted proofs and definitions in the paper of Saks and Wigderson (1986). In particular, by extending observation by Arimoto (2020), we introduce concepts of "weakly balanced multi-branching tree" as modified versions of "nearly balanced tree" of Saks and Wigderson, and we show recurrence formulas of randomized complexity for weakly balanced multi-branching trees.

Keywords: AND OR trees, randomized complexity, multi-branching tree.

1 Introduction

Given a NAND tree, we investigate query complexity, in other words the number of Boolean variables probed during computation. In a NAND tree, we can move NOT gates to the position just above the leaves by means of de Morgan's law. Therefore the study of query complexity of NAND trees is the same as those of AND-OR trees or OR-AND trees. In the following we mainly discuss AND-OR trees. The idea of basic setting is as follows. A truth assignment wants bigger query complexity, while an algorithm wants smaller query complexity.

Among classical works, we are interested in the work of Saks and Wigderson [7]. Before [7], the researchers in this area mainly studied the case where a probability distribution on the truth assignments to the leaves is an identically and independent distribution (i.i.d. for short, Baudet [2], Pearl [5, 6], Tarsi 1983[9]). In [7], they study variants of Nash equilibriums under correlated distributions. To be more precise, *the randomized complexity* is

$$R := \min_{A_R} \max_{x} \mathrm{cost}(A_R, x), \tag{1.1}$$

where a tree is fixed and A_R runs over *randomized algorithms* (probability distributions on the deterministic algorithms for the tree) and x runs over truth assignments on the leaves of the tree. A dual notion, *the distributional complexity* is

$$P := \max_{\delta} \min_{A} \mathrm{cost}(A, \delta), \tag{1.2}$$

where a tree is fixed and δ runs over probability distributions on the truth assignments and A runs over deterministic algorithms.

[*]The corresponding author. This work was partially supported by Japan Society for the Promotion of Science (JSPS) KAKENHI (C) 21K03340 and the Research Institute for Mathematical Sciences, an International Joint Usage/Research Center located in Kyoto University.

In order to give an upper bound of R, they introduced a variant of R where algorithms are limited to a specific type. A randomized directional algorithm in the sense of Saks and Wigderson (in this paper, *r.d.a.* for short) is defined in a recursive manner. Given a tree T, let n be the number of child nodes of the root, $T_1, \dots,$ and T_n the subtrees just under the root, $\mathcal{S}_n$ the set of all permutations on $\{1, \dots, n\}$. A randomized algorithm A is an r.d.a. on T if there is a probability distribution π on $\mathcal{S}_n$ and r.d.a $A_1, \dots, A_n$ on $T_1, \dots, T_n$ respectively, and A runs $A_{\sigma(1)}, \dots, A_{\sigma(n)}$ in this order, where σ is the permutation on $\{1, \dots, n\}$ chosen by π. A variant of R is defined as follows.

$$d := \min_{A_R} \max_{x} \operatorname{cost}(A_R, x), \tag{1.3}$$

where a tree is fixed and A_R runs over r.d.a and x runs over truth assignments on the leaves of the tree. By the definition, we have $R \leq d$.

By the way, Yao's principle, a variant of von Neumann's minimax theorem, holds: $R = P$. In [7], they gave a detailed discussion of lower bonds for P in the case of uniform binary trees. Their method has two key ideas. First, they shrank a given tree by merging plural nodes. Second, by means of the shrinking method, they restricted the domain where δ of (1.2) runs to a specific type of truth assignments, that is, *reluctant inputs* in their terminology. Then the counterpart to the right-hand side of (1.2) is smaller or equal to the original left-hand side. By Yao's principle [10], we get a lower bound for R.

They showed that if a binary tree satisfies a certain hypotheses (the hypotheses in [7, Lemma 5.1]), the above-mentioned upper bound coincide with the above-mentioned lower bound. They defined *nearly balanced trees*, and asserted that each nearly balanced tree satisfies the above-mentioned hypotheses. In particular, letting $R(h)$ denote R of the uniform binary AND-OR tree of height h, they got a recurrence formula of $R(h+1)$ and $R(h)$. By means of the recurrence formula, they showed:

$$R(h) = O(\ (\frac{1+\sqrt{33}}{4})^h\ ) = O(\ D^{0.753}\ ), \tag{1.4}$$

where D is defined in a manner similar to R by restricting algorithms to deterministic ones. This result shows that randomized algorithm behaves better than deterministic ones in this setting.

Let $R(h, k)$ denote a counterpart to $R(h)$ in the uniform k-branching tree of height h. Without a detailed discussion, they showed:

$$R(h,k) = O(\ (\frac{k-1+\sqrt{k^2+14k+1}}{4})^h\ ) \tag{1.5}$$

Later, Liu and Tanaka [4] reconstructed a proof of (1.5) by means of their "reverse assignment technique". In another paper of the same year [3], Liu and Tanaka extended some results of [7]. After [4] and [3], subsequent works followed (see [8]).

The time interval between [7] and [3] is more than twenty years, which is not short. Here, "not short" has a double meaning: Why the old paper [7] is worth picking up again?: Why it took so long? To the former question, our answer is "Because the explicit recurrence formulas of expected costs in [7] are significant. Their proofs are sources of ideas." To the latter question, our answer is "Because [7] is not easy to read." In [7], there are no recurrence formulas and no proof for the multi-branching case. In addition, there are no proof of the assertion that each nearly balanced tree satisfies the hypotheses of [7, Lemma 5.1]. As far as we can see, the definition of nearly balanced trees would need a revision. Indeed the reconstruction of (1.5) by Liu and Tanaka [4] is interesting, we would like to read the omitted proof in [7] in a faithful style to the original proof of binary trees. The goal of this paper is to reconstruct such a proof for multi-branching trees under a modified version of "nearly balanced" concept.

In section 2, we will introduce some technical terms and will state our goal more clearly. Arimoto [1] proposed an alternative for the concept of nearly balanced tree in the case of uniform binary trees. He showed that the alternative concept implies the hypotheses in [7, Lemma 5.1]. In section 2, we introduce some concepts that extend the hypotheses in [7, Lemma 5.1] and the above-mentioned concept of Arimoto.

In sections 3 and 4, under an assumption similar to the hypotheses in [7, Lemma 5.1], we show recurrence formulas of randomized complexity for multi-branching trees. In section 3 we discuss the costed trees of height 1 as the base case. In section 4, we show recurrence formulas of randomized complexity in a faithful style to the original proof of binary trees. In particular, we shrink a given tree by merging plural nodes. Sections 3 and 4 are written by Kurita and Shimizu.

Section 5 is written by the third author. We apply results in sections 3 and 4 to multi-branching trees, and show a multi-branching counterpart to Arimoto's observation [1]. For the general background, [8] is a concise survey paper.

2 Notation and the detailed description of our goal

In this section, we are going to introduce our terminology on "weakly balanced trees". By means of the terminology, we will state our goal clearly.

For each $t \in \{0,1\}$, we define R_t in the same way as (1.1) by restricting ourselves to truth assignments x for which the root node has value t. In the same way, we define d_t and P_t. We are going to define weakly balanced trees. In the following definition, for each child node x_i of the root, we let $R_t(x_i)$ denote R_t of the subtree whose root is x_i.

Definition 2.1. Suppose that T is an AND-OR tree or an OR-AND tree. For each internal node v, let $N(v)$ be the degree of v, that is, the number of child nodes of v.

1. T is *weakly balanced in the first sense* if for each internal node v and for each child nodes x_i and x_j of v, we have $N(v) \geq N(x_i)$ and the following hold.

   (a) If v is an AND-node then we have:

   $$(N(v)-1)R_0(x_i) \leq N(v)R_0(x_j) \tag{2.1}$$

   (b) If v is an OR-node then we have:

   $$(N(v)-1)R_1(x_i) \leq N(v)R_1(x_j) \tag{2.2}$$

2. T is *weakly balanced in the second sense* if for each internal node v and its child nodes x_i and x_j we have the following.

   (a) If v is an AND-node then we have:

   $$R_0(x_i) \leq R_1(x_i) + R_0(x_j) \tag{2.3}$$

   (b) If v is an OR-node then we have:

   $$R_1(x_i) \leq R_0(x_i) + R_1(x_j) \tag{2.4}$$

If we let a_i (b_i, respectively) denote $R_0(x_i)$ ($R_1(x_i)$, respectively) then the above four equations (2.1),(2.2),(2.3) and (2.4) are the following, respectively. Here, N denotes $N(v)$.

$$(N-1)a_i \leq Na_j,\ (N-1)b_i \leq Nb_j,\ a_i \leq b_i + a_j,\ b_i \leq a_i + b_j \tag{2.5}$$

In the analysis of binary trees in [7], the following function played an important role.

$$\Psi(x_1,x_2,y_1,y_2) = \frac{x_1y_1 + x_2y_2 + y_1y_2}{y_1+y_2} \tag{2.6}$$

The first and the second authors generalized the above function (2.6) to a $2n$-ary function.

$$\Psi_n(x_1, \dots, x_n, y_1, \dots, y_n) = \frac{\sum_{i=1}^n x_i y_i + \sum_{1 \le i < j \le n} y_i y_j}{\sum_{i=1}^n y_i} \tag{2.7}$$

We believe that Saks and Wigderson used (2.7) in their unpublished draft. The function (2.7) was essentially introduced by Liu and Tanaka [4, Theorem 3] but their formula has a typo of forgetting to write the condition $i < j$ in the second term of the numerator of (2.7). Roughly speaking, the following concept says that (2.7) gives recurrence formulas for R_t.

Definition 2.2. 1. T is *nice* if for each internal node v and its child nodes x_i and x_j we have the following.

(a) If v is an AND-node then $R_1(v) = \sum_i R_1(x_i)$ and
$R_0(v) = \Psi_{N(v)}(R_0(x_1), \dots, R_0(x_{N(v)}), R_1(x_1), \dots, R_1(x_{N(v)})))$.

(b) If v is an OR-node then $R_0(v) = \sum_i R_0(x_i)$ and
$R_1(v) = \Psi_{N(v)}(R_1(x_1), \dots, R_1(x_{N(v)}), R_0(x_1), \dots, R_0(x_{N(v)})))$.

Assume that T is a binary AND-OR tree for a while. Thus $N(v) = 2$ for any internal node v. Temporarily, assume in addition that $R_t(v) = a_t(v)$ for each $t = 0, 1$, where a_t in the right-hand side is not a_i in (2.5) but the quantity a_t recursively defined in Saks-Wigderson [7]. Under this assumption, "weakly balanced in the first sense" is exactly same as "weakly balanced" in Ariomoto [1]. Under the same assumption, "weakly balanced in the second sense" is exactly same as the hypotheses in [7, Lemma 5.1]. Arimoto showed the following.

Proposition 2.1. *(Arimoto [1]) Suppose that T is a binary AND-OR tree.*

1. *If T is weakly balanced in the sense of Arimoto then T is weakly balanced in the second sense.*
2. *If T is weakly balanced in the second sense then T is nice (A reconstruction of [7, Lemma 5.1]).*

Therefore in the case of uniform binary tree we have $R_t = O(((1 + \sqrt{32})/4)^h)$ ([7, Example 1.1]), thus (1.4) holds. Now, we turn to the multi-branching case. In sections 3 and 4, we will prove the following. This extends Proposition 2.1 (2) to the multi-branching case.

Theorem 2.1. *Suppose that T is a multi-branching AND-OR tree that is weakly balanced in the second sense. Then T is nice.*

Thus in the case of uniform k-ary trees we have the following.

Corollary 2.2. *([7, Theorem 5.4], [4]) Suppose that $k \ge 2$ and that T is a uniform k-ary AND-OR tree. Then its randomized complexity is given as follows.*

$$R = \Theta(((k - 1 + \sqrt{k^2 + 14k + 1})/4)^h) \tag{2.8}$$

Proof. (of Corollary from Theorem 2.1) For each $t = 0, 1$ and for a positive integer h, let $R^h_{\wedge,t}$ ($R^h_{\vee,t}$, respectively) denote the randomized complexity R_t with constraint that the root has value t, where an associated tree is a uniform k-ary AND-OR tree (OR-AND tree, respectively) of height h. By Theorem 2.1, we have the following for each positive integer h.

$$\begin{pmatrix} R^{h+2}_{\wedge,0} \\ R^{h+2}_{\wedge,1} \end{pmatrix} = \begin{pmatrix} \Psi_k(R^{h+1}_{\vee,0}, \dots, R^{h+1}_{\vee,1}, \dots) \\ k \times R^{h+1}_{\vee,1} \end{pmatrix} = \begin{pmatrix} R^{h+1}_{\vee,0} + \frac{k-1}{2} R^{h+1}_{\vee,1} \\ k \times R^{h+1}_{\vee,1} \end{pmatrix} = \begin{pmatrix} 1 & \frac{k-1}{2} \\ 0 & k \end{pmatrix} \begin{pmatrix} R^{h+1}_{\vee,0} \\ R^{h+1}_{\vee,1} \end{pmatrix} \tag{2.9}$$

In the same way, we have:

$$\begin{pmatrix} R^{h+1}_{\vee,0} \\ R^{h+1}_{\vee,1} \end{pmatrix} = \begin{pmatrix} k & 0 \\ \frac{k-1}{2} & 1 \end{pmatrix} \begin{pmatrix} R^{h}_{\wedge,0} \\ R^{h}_{\wedge,1} \end{pmatrix} \tag{2.10}$$

The product of the 2×2 matrix in (2.9) and that in (2.10) is the square of $A = \begin{pmatrix} \frac{k-1}{2} & 1 \\ k & 0 \end{pmatrix}$. The eigenvalues of A are $(k-1 \pm \sqrt{k^2+14k+1})/4$. Therefore for each $t = 0, 1$ we have $R_t^{h+2} = \Theta(((k-1+\sqrt{k^2+14k+1})/4)^{h+2})$. Hence equation (2.8) holds. $\square$

Our goal is to extend Proposition 2.1 (1) to the multi-branching case. In section 5 we will show that a slightly stronger property than "weakly balanced in the first sense" implies "the second sense".

Theorem 2.3. *Suppose that T is a multi-branching AND-OR tree that is weakly balanced in the first sense and both* (2.1) *and* (2.2) *hold at every internal node v. Then T is weakly balanced in the second sense.*

Thus, by Theorem 2.1, T is nice. In other words, the following holds.

Corollary 2.4. *Suppose that T is a multi-branching AND-OR tree that is weakly balanced in the first sense and both* (2.1) *and* (2.2) *hold at every internal node v. Then T is nice.*

3 Randomized complexity of height 1 costed trees

In the next section, we will show a slightly stronger version of Theorem 2.1. Our proof is induction on the height of a tree.

Lemma 3.1. *Suppose that T is a multi-branching AND-OR tree that is weakly balanced in the first sense. Then T is nice, and we have $R_t = d_t$ for each $t \in \{0, 1\}$.*

Conventions throughout sections 3 and 4 T^n denotes a tree such that the root has n child nodes. The subtrees just below the root are denoted by $T_1^n, \dots T_n^n$. We let $\mathcal{A}$ denotes the set of all r.d.a. for T^n. For each $i \in \{1, \dots, n\}$, we let a_i (b_i, respectively) denote $d_0(T_i^n)$ ($d_1(T_i^n)$, respectively). In our proof, it will be obvious that it equals $R_0(T_i^n)$ ($R_1(T_i^n)$, respectively) by the induction hypothesis. Unless specified, "a weakly balanced tree" means a weakly balanced tree in the second sense.

The concept of a *costed tree* is introduced in [7] as a tool of induction. Each leaf x has a positive real number $c(x)$ called the *cost* of x (or the *weight* of x); when an algorithm probes x then the cost is $c(x)$. The usual tree is the case where $c(x) = 1$ for all x. In this section, as the base case of induction, we consider costed trees of height 1. We only prove the case of AND-trees. Note that in height 1 case, every randomized algorithm is an r.d.a. hence we have $R_i(T^n) = d_i(T^n)$.

The case where root value is 1 When the root value is 1, any deterministic algorithm must probe all the child nodes. Thus we have:

$$R_1(T^n) = d_1(T^n) = \sum_{i=1}^{n} b_i \tag{3.1}$$

The case where root value is 0 In the remainder of this section, we look at the case where the root value is 0. We are going to prove $d_0(T^n) = \Psi_n$. We avoid combinatorial explosion in the multi-branching case by investigating lower and upper bounds of $d_0(T^n)$. Fortunately, we will see that the both lower and upper bounds are given by the same quantity Ψ_n. We begin with the upper bound.

Temporarily, let x_i be the truth assignment that assigns 1 only to the i-th leaf from the left and 0 to all the others. Given an algorithm $A \in \mathcal{A}$, we let $W_i^n(A)$ denote the expected cost of A on the input x_i. There are $n!$ deterministic algorithms for T^n. We denote them by $A_1, ..., A_m$ ($m = n!$). We let p_k denote $A(A_k)$, in other words, the probability that A selects A_k.

Clealy, we have the following.

$$W_i^n(A) = \sum_{k=1}^{m} p_k \,\mathrm{cost}(A_k,\ x_i) \tag{3.2}$$

We define a set V of randomized algorithms on T^n as follows. An algorithm $A \in \mathcal{A}$ belongs to V if for any deterministic algorithm B of positive probability (that is, $A(B) > 0$), either B probes the n-th leaf first or last.

By induction on the number of branchs, we show the following two assertions. First, we have $d_0(T^n) = \Psi_n$ in V: Second, there exists an $A \in V$ such that for all i we have $W_i^n(A) = \Psi_n$. What we really need is the first assertion. The second assertion will make our induction smooth.

Given an element of V, let p be the probability that the n-th leaf from the left is probed last. Recall that in such an algorithm, the n-th leaf from the left is probed first with probability $1-p$.

The case of $k = 2$

Is is shown in [7] that $d_0(T^2) = \Psi_2$ and there exists an A such that $W_1^2(A) = W_2^2(A) = \Psi_2$.

The case of $k = n-1$

Assume that $d_0(T^{n-1}) = \Psi_{n-1}$ and there exists an A such that the following holds.

$$W_1^{n-1}(A) = \cdots = W_{n-1}^{n-1}(A) = \Psi_{n-1} \tag{3.3}$$

We investigate W_n^i on V while in our mind we break T^n down into two parts, the n-th leaf from the left and "an AND-tree of height 1" consisting of the other $n-1$ leaves. By induction hypothesis, there exists an algorithm A' that achieves $d_0(T^{n-1}) = \Psi_{n-1}$. Therefore we have the following algorithm $A \in V$ for T^n.

The n-th leaf from the left the left is probed at either the beginning or the last, "The other" part is probed by A'.

In this case, the expected cost W_i^n with respect to A is as follows.

$$\begin{aligned} W_1^n(A) = \cdots = W_{n-1}^n(A) &= pd_0(T^{n-1}) + (1-p)(b_n + d_0(T^{n-1})) \\ &= p\Psi_{n-1} + (1-p)(b_n + \Psi_{n-1}) \\ &= (1-p)b_n + \Psi_{n-1}, \qquad (3.4) \\ W_n^n(A) &= p(b_1 + \cdots + b_{n-1} + a_n) + (1-p)a_n \\ &= p(b_1 + \cdots + b_{n-1}) + a_n \qquad (3.5) \end{aligned}$$

Each W_i^n is a linear function of p. Therefore, we know that $d_0(T^n) = \min_{0 \le p \le 1} \max\{(1-p)b_n + \Psi_{n-1},\ p(b_1 + \cdots + b_{n-1}) + a_n\}$ on V.

We investigate when it happens that $W_i^n(A)$ are the same for all i. As a representative, we investigate the condition $W_1^n(A) = W_n^n(A)$.

$$\begin{aligned} W_1^n(A) = W_n^n(A) &\iff (1-p)b_n + \Psi_{n-1} = p(b_1 + \cdots + b_{n-1}) + a_n \\ &\iff p(b_1 + \cdots + b_n) = \Psi_{n-1} - a_n + b_n \\ &\iff p = \frac{\Psi_{n-1} - a_n + b_n}{b_1 + \cdots + b_n} \quad (=: p_0) \qquad (3.6) \end{aligned}$$

We would like to show that this is possible. It is sufficient to show $0 \le p \le 1$, in other words:

$$\Psi_{n-1} \le b_1 + \cdots + b_{n-1} + a_n \tag{3.7}$$

$$a_n \le \Psi_{n-1} + b_n \tag{3.8}$$

By the induction hypothesis, there exists an A_0 such that all the expected costs $W_1^{n-1}(A_0)$, $\cdots$, and $W_{n-1}^{n-1}(A_0)$ in T^{n-1} are equal to Ψ_{n-1}.

$$\begin{aligned} \Psi_{n-1} \le \max_{B \in \mathcal{A}} W_{n-1}^{n-1}(B) &\iff \Psi_{n-1} \le b_1 + \cdots + b_{n-2} + a_{n-1} \qquad (3.9) \\ &\quad \text{[The maximum cost is achieved by "value 0 leaf later" strategy.]} \\ &\implies \Psi_{n-1} \le b_1 + \cdots + b_{n-1} + a_n \qquad (3.10) \\ &\quad [a_{n-1} \le b_{n-1} + a_n \text{ by the assumption of weakly balanced.}] \end{aligned}$$

Therefore, $\Psi_{n-1} \leq b_1 + \cdots + b_{n-1} + a_n$ holds. In the same way, we have the following.

$$\min_{B\in\mathcal{A}} W_{n-1}^{n-1}(B) \leq \Psi_{n-1} \iff a_{n-1} \leq \Psi_{n-1} \tag{3.11}$$

[The minimum cost is achieved by "value 0 leaf first" strategy.]

$$\iff a_{n-1} + b_n \leq \Psi_{n-1} + b_n$$

$$\Longrightarrow a_n \leq \Psi_{n-1} + b_n \tag{3.12}$$

[$a_n \leq a_{n-1} + b_n$ by the assumption of weakly balanced.]

Therefore, $a_n \leq \Psi_{n-1} + b_n$ holds, and we have shown $0 \leq p \leq 1$. By substituting p_0 for p, we have:

$$\begin{aligned}
W_n^n(A) =& p(b_1 + \cdots + b_{n-1}) + a_n \\
=& \frac{\Psi_{n-1} - a_n + b_n}{b_1 + \cdots + b_n}(b_1 + \cdots + b_{n-1}) + a_n \\
=& \frac{1}{b_1 + \cdots + b_n}\left\{\Psi_{n-1}(b_1 + \cdots + b_{n-1}) + (-a_n + b_n)(b_1 + \cdots + b_{n-1}) + a_n \sum_{i=1}^{n} b_i\right\} \\
=& \frac{1}{b_1 + \cdots + b_n}\left\{\sum_{i=1}^{n-1} a_i b_i + \sum_{1\leq i<j\leq n-1} b_i b_j + (-a_n + b_n)(b_1 + \cdots + b_{n-1}) + a_n \sum_{i=1}^{n} b_i\right\} \\
=& \frac{1}{b_1 + \cdots + b_n}\left\{\sum_{i=1}^{n-1} a_i b_i + \sum_{1\leq i<j\leq n-1} b_i b_j + a_n b_n + b_n(b_1 + \cdots + b_{n-1})\right\} \\
=& \frac{1}{b_1 + \cdots + b_n}\left\{\sum_{i=1}^{n} a_i b_i + \sum_{1\leq i<j\leq n} b_i b_j\right\} \\
=& \Psi_n
\end{aligned} \tag{3.13}$$

$$\begin{aligned}
W_1^n(A) =& (1-p)b_n + \Psi_{n-1} \\
=& \left(1 - \frac{\Psi_{n-1} - a_n + b_n}{b_1 + \cdots + b_n}\right) b_n + \Psi_{n-1} \\
=& \left\{\frac{-\Psi_{n-1} + a_n + (b_1 + \cdots + b_{n-1})}{b_1 + \cdots + b_n}\right\} b_n + \Psi_{n-1} \\
=& \frac{1}{b_1 + \cdots + b_n}\{-\Psi_{n-1} b_n + a_n b_n + b_n(b_1 + \cdots + b_{n-1}) + \Psi_{n-1}(b_1 + \cdots + b_n)\} \\
=& \frac{1}{b_1 + \cdots + b_n}\{\Psi_{n-1}(b_1 + \cdots + b_{n-1}) + a_n b_n + b_n(b_1 + \cdots + b_{n-1})\} \\
=& \frac{1}{b_1 + \cdots + b_n}\left\{\sum_{i=1}^{n} a_i b_i + \sum_{1\leq i<j\leq n}^{n} b_i b_j\right\} \\
=& \Psi_n
\end{aligned} \tag{3.14}$$

Two linear functions, one is of positive slope and the other is of negative slope, intersects at point A. Thus minmax of the two linear functions is achived at A. In other words, we have $\min_{0\leq p\leq 1} \max\{(1-p)b_n + \Psi_{n-1},\ p(b_1 + \cdots + b_{n-1}) + a_n\} = \Psi_n$, Thus, we have shown that $d_0(T^n)$ in V equals Ψ_n. Thus $d_0(T^n) \leq \Psi_n$ in $\mathcal{A}$.

The "root= i" counterparts to P and R are denoted by P_i and R_i. By means of Yao's principle ($P_i = R_i$) we are going to show $d_0(T^n) \geq \Psi_n$. We want to find a particular distribution δ_1 (root is i) and find $\min_{A_{\text{det}}} \text{cost}(A_{\text{det}}, \delta_1)$ (=:L). Then we have $L \leq \max_{\substack{\delta \\ \text{root}:0}} \min_{A_{\text{det}}} = P_0 = R_0 \leq d_0$, and we will get a lower bound for R_0 and d_0. Our δ_1 is as follows.

For each i, consider an input in which the i-th leaf from the left has a value of 0 and the other leaves have a value of 1. For this input, we give a probability $b_i / \sum_{j=1}^{n} b_j$. For all the other inputs, give probability 0. We want to find $\min_{A_{\text{det}}} \text{cost}(A_{\text{det}}, \delta_1)$. For example, let A_{left} be an algorithm that searches leaves from the left to right.

$$\begin{aligned}
\text{cost}(A_{\text{left}}, \delta_1) =& \frac{b_1}{\sum_{j=1}^{n} b_j} a_1 + \frac{b_2}{\sum_{j=1}^{n} b_j}(b_1 + a_2) + \cdots + \frac{b_n}{\sum_{j=1}^{n} b_j}(b_1 + b_2 + \cdots + b_{n-1} + a_n) \\
=& \frac{1}{\sum_{j=1}^{n} b_j}(a_1 b_1 + (a_2 b_2 + b_1 b_2) + \cdots + (a_n b_n + b_1 b_n + \cdots + b_{n-1} b_n)) \\
=& \frac{\sum_{i=1}^{n} a_i b_i \; + \sum_{1 \le i < j \le n}^{n} b_i b_j}{\sum_{i=1}^{n} b_i} \\
=& \Psi_n
\end{aligned} \tag{3.15}$$

Since the tree is symmetric, we obtain the same result for the other deterministic algorithms. Therefore, $\min_{A_{\text{det}}} \text{cost}(A_{\text{det}}, \delta_1) = \Psi_n$. Hence $R_0(T^n) \ge \Psi_n$ holds. Now we have shown $R_0(T^n) = d_0(T^n) = \Psi_n$ for a tree of height 1. Thus we have shown Lemma 3.1 for a costed tree of height 1.

4 Recurrence formulas for randomized complexity

In this section, we are going to show Lemma 3.1 for (not costed) trees of height ≥ 2.

4.1 Upper bounds for randomized complexity

In this subsection, by means of our observation for costed trees of height 1, we are going to show an upper bound for R with respect to a tree T of height h. First, we show that $d_0(T^n) \le \Psi_n$.

When algorithm A for T_n achieves both $d_0(T^n)$ and $d_1(T^n)$, we may assume the following two without loss of generality.

1. A_i $(i = 1, \ldots, n)$ achieves both $d_0(T_i^n)$ and $d_1(T_i^n)$ on the subtree T_i^n .
2. If x is the worst input (a minimizer of $\text{cost}(A, x)$) for A, then the input x_i of each subtree is the worst input for A_i.

Therefore we may regard T^n as a costed tree of height 1. Thus our method in section 3 applies to T^n.

Now that we know that $d_0(T^n)$ in set V equals Ψ_n, the real Ψ_n (for all r.d.a.) is at most Ψ_n. Since $R_0(T^n) \le d_0(T^n)$ we have:

$$R_0(T^n) \le \; \Psi_n \tag{4.1}$$

The method in the case where root value is 1 applies to $R_1(T_n)$ as well. Therefore we have:

$$R_1(T^n) \le d_1(T^n) \; = \; \sum_{i=1}^{n} b_i \tag{4.2}$$

4.2 Lower bounds for randomized complexity

By Yao's principle, the following holds.

$$\min_{A_R} \max_{x\in\{0,1\}^n} \mathrm{cost}(A_R, x) \;=\; \max_{d} \min_{A} \mathrm{cost}(A, d) \tag{4.3}$$

In this section, we discuss the right-hand side.

Proofs of the lower bounds for a uniform binary trees are given in Saks and Wigderson[7] (see also Arimoto[1]). We extend them to the multi-branching case. We distinguish two cases to prove lower bounds. Case 1 proceeds in the same way as Arimoto[1].Our new method for multi-branching case will appear in Case 2. We state a few preliminary concepts before the proof.

We define the functions $\ell_0(T^n), \ell_1(T^n)$ as follows.

$$\text{If } |T^n| = 1, \quad \ell_0(T^n) \;=\; \ell_1(T^n) \;=\; 1 \tag{4.4}$$

If the root is labeled with $\wedge$,

$$\begin{cases} \ell_1(T^n) \;=\; \ell_1(T_1^n) + \ell_1(T_2^n) + \cdots + \ell_1(T_n^n) \\ \ell_0(T^n) \;=\; \Psi_n(\ell_0(T_1^n), \ldots, \ell_0(T_n^n), \ell_1(T_1^n), \ldots, \ell_1(T_n^n)) \end{cases} \tag{4.5}$$

If the root is labeled with $\vee$

$$\begin{cases} \ell_0(T^n) \;=\; \ell_0(T_1^n) + \ell_0(T_2^n) + \cdots + \ell_0(T_n^n) \\ \ell_1(T^n) \;=\; \Psi_n(\ell_1(T_1^n), \ldots, \ell_1(T_n^n), \ell_0(T_1^n), \ldots, \ell_0(T_n^n)) \end{cases} \tag{4.6}$$

We are going to show the following theorem.

Theorem 4.1. *Assume that T^n satisfies the following conditions for all $i,\ j\ (i \neq j,\ 1 \leq i, j \leq n)$: If a node is labeled with $\wedge$,*

$$\ell_0(T_i^n) \leq \ell_1(T_i^n) + \ell_0(T_j^n);$$

if a node is labeled with $\vee$,

$$\ell_1(T_i^n) \leq \ell_0(T_i^n) + \ell_1(T_j^n).$$

Then it holds that $R_0(f) \geq \ell_0(T^n)$ and $R_1(f) \geq \ell_1(T^n)$.

To show Theorem 4.1, we define some symbols and functions.

c_i	A non-negative real-valued function whose domain is the set of all leaves
$X_{(T^n)}$	A set of all nodes whose children are leaves
v	A node whose children are leaves
$y_1, y_2, \ldots, y_n$	Child nodes of v
$R_i(f : c_0, c_1)$	The counterpart of $R_i(T^n)$ when the cost of probing a leaf is given by c_i
$(T^n)'$	A tree obtained by replacing $y_1, y_2, \ldots, y_n$ of T^n by y_1'. We abbreviate it to $T^{n'}$.

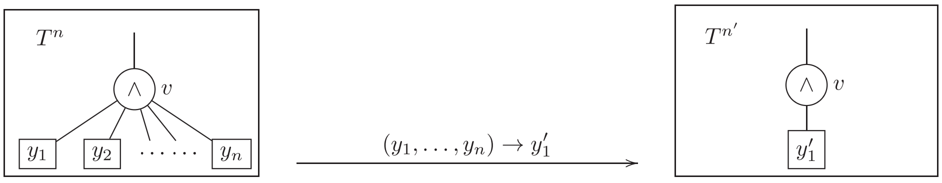

Figure 1: relationship between T^n and $T^{n'}$.

Note : In $T^{n'}$, v exceptionally has only one child. (Basically, there must be two or more children.) In $T^{n'}$, we may think of v as a leaf .

c'_i is a function on the leaves of $T^{n'}$ and is defined as follows.

$$\text{If } w \neq v, \begin{cases} c'_0(w) &= c_0(w) \\ c'_1(w) &= c_1(w) \end{cases} \tag{4.7}$$

If v is labeled with $\wedge$

$$\begin{cases} c'_1(y'_1) &= c_1(y_1) + c_1(y_2) + \cdots + c_1(y_n) \\ c'_0(y'_1) &= \Psi_n(c_0(y_1), \ldots, c_0(y_n), c_1(y_1), \ldots, c_1(y_n)) \end{cases} \tag{4.8}$$

If v is labeled with $\vee$

$$\begin{cases} c'_0(y'_1) &= c_0(y_1) + c_0(y_2) + \cdots + c_0(y_n) \\ c'_1(y'_1) &= \Psi_n(c_1(y_1), \ldots, c_1(y_n), c_0(y_1), \ldots, c_0(y_n)) \end{cases} \tag{4.9}$$

First, we would like to show the following lemma.

Lemma 4.2. *(The binary-case counterpart is shown in Saks-Wigderson[7])*

$$R_i(f : c_0 : c_1) \geq R_i(f : c'_0 : c'_1)$$

Let k be the number of internal nodes of T^n. By applying Lemma 4.2 k times, the following equations and inequalities will be established.

$$R_i(T^n) = R_i(f : c_0, c_1) \geq R_i(f : c'_0, c'_1) \geq R_i(f : c''_0, c''_1) \geq \cdots \geq R_i(f : c^k_0 : c^k_1) \ = \ \ell_i(T^n) \tag{4.10}$$

Therefore we will get Theorem 4.1. Here, c_i is a function that takes the value 1 at all leaves.

Here we define more symbols.

A	A deterministic Algorithm that evaluates T^n. Only alpha beta prunning algorithms are considered.
w	The first leaf that A probes
S'	A distribution defined on the input of $T^{n'}$
S	Abbreviation of distribution $S'[(y_1, \ldots, y_n; c)/(v; c')]$ on the input of T^n defined using S'. We will define it later.

$$\text{(} S : \text{A distribution on } T^n \text{)} \xleftarrow{S'[(y_1, \ldots, y_n; c)/(y'_1; c')] \hookleftarrow S'} \text{(} S' \text{ : A distribution on } T^{n'} \text{)}$$

$S'(x')$	The probability assigned to an input x' in S'
$S(x)$	The probability assigned to an input x in S
p	The probability that $v = 0$ on $T^{n'}$
p_{y_1}	$\dfrac{c_1(y_1)}{c_1(y_1) + \cdots + c_1(y_n)} p$
p_{y_2}	$\dfrac{c_1(y_2)}{c_1(y_1) + \cdots + c_1(y_n)} p$
$\vdots$	$\vdots$
p_{y_n}	$\dfrac{c_1(y_n)}{c_1(y_1) + \cdots + c_1(y_n)} p$

Now, recall that a truth assignment is a function on the leaves to $\{0, 1\}$. For each truth assignment x of T^n, we define truth assignment $x'(0)$ and $x'(1)$ of $T^{n'}$ as follows.

$x'(i)$ Remove $y_1, \dots, y_n$ from the domain of x and put y'_1 into the domain. The value of y'_1 is i and the values of the others are as before.

Then, we define $S'[(y_1, \dots, y_n; c)/(v; c')]$ (abbreviated to S) as follows.

$$S(x) := \begin{cases} \dfrac{c_1(y_1)}{c_1(y_1) + \cdots + c_1(y_n)} S'(x'(0)) & \text{if} \quad (y_1, \dots, y_n) = (0, 1, \dots, 1) \\ \quad \vdots & \quad \vdots \\ \dfrac{c_1(y_n)}{c_1(y_1) + \cdots + c_1(y_n)} S'(x'(0)) & \text{if} \quad (y_1, \dots, y_n) = (1, \dots, 1, 0) \\ S'(x'(1)) & \text{if} \quad (y_1, \dots, y_n) = (1, 1, \dots, 1) \\ 0 & \qquad \text{otherwise} \end{cases} \tag{4.11}$$

Lemma 4.3. *Assume that $v \in X_{(T^n)}$.For any pair (S', A) of a distribution S' on $T^{n'}$ and an algorithm A that evaluates T^n, there is an algorithm A' such that the followoing inequality holds.*

$$\text{cost}(A,\ S) \geq \text{cost}(A',\ S')$$

Here, S means $S'[(y_1, \dots, y_n; c)/(v; c')]$.

Proof. (of Theorem 4.1 from Lemma 4.3, similar to Arimoto [1])

We include a proof to keep the paper self-contained. Take a distribution $S'_{\max}$ on the inputs of $T^{n'}$ so that it maximizes $\min_{B'} \text{cost}(B',\ S')$. $S'_{\max}$ determines the distribution $S = S'[(y_1, \dots, y_n; c)/(v; c')]$ on the inputs of T^n. Fix an algorithm A that is optimal for S. Let A' be the algorithm in Lemma 4.3 with respect to the above-stated pair $(S'_{\max},\ A)$. Furthermore, let $S_{\max}$ be the distribution on the inputs of T^n that maximizes $\min_B \text{cost}(B,\ S)$. Then we have the following inequalities.

$$\min_{B'} \text{cost}(B',\ S'_{\max})\ \leq\ \text{cost}(A',\ S'_{\max})\ \leq\ \text{cost}(A,\ S)\ \leq\ \min_{B} \text{cost}(B,\ S_{\max}) \tag{4.12}$$

Therefore Lemma 4.2 holds. As we stated before, Theorem 4.1 follows. □

We are going to show Lemma 4.3.

Proof. (of Lemma 4.3)

We distinguish two cases depending on whether w is a child node of v or not.

Case 1 (Case 1 is similar to Arimoto [1])

The case where w is not a child of v. We include a proof to keep the paper self-contained. Assume the following for a contradiction.

There exist (as a counterexample) T^n, v, S', A such that no A' satisfy $\text{cost}(A, S) \geq \text{cost}(A', S')$.
(Note that S means $S'[(y_1, \dots, y_n; c)/(v; c')]$.)

Assume that T^n is a minimum (in the number of its nodes) counterexample. In the following v, S', A are those for this T^n. We define the symbols as follows:.

A_i When A probes x_w at the beginning and gets value i (i is 0 or 1) then the move of A after that is A_i.

p_w $p_w = \text{prob}_S[x_w = 0]$ ($= \text{prob}_{S'}[x_w = 0]$). Remark : S (S', respectively) is on T^n ($T^{n'}$, respectively)

$T^{n'}[i/w]$ The tree obtained by substituting i for w on $T^{n'}$

$T^n[i/w]$ The tree obtained by substituting i for w on T^n

$S_i,\ S'_i$ Distributions on $T^n[i/w]$, $T^{n'}[i/w]$, respectively (The definition will be described later.)

First, we define S_i using S. Suppose that $x_{\{i\}}$ is a truth assignment on $T^n[i/w]$. Recall that $x_{\{i\}} \cup \{(w,i)\}$ is an assignment on T^n such that w, a leaf not in $x_{\{i\}}$, receives value i. In S_i probability of $x_{\{i\}}$ is as follows.

$$\begin{aligned} S_0(x_{\{0\}}) : &= \frac{1}{p_w} S(x_{\{0\}} \cup \{(w,0)\}) && (4.13) \\ S_1(x_{\{1\}}) : &= \frac{1}{1-p_w} S(x_{\{1\}} \cup \{(w,1)\}) && (4.14) \end{aligned}$$

We define S'_i in the same way. Suppose that $x'_{\{i\}}$ is a truth assignment on $T^{n'}[i/w]$. Recall that $x'_{\{i\}} \cup \{(w,i)\}$ is an assignment on $T^{n'}$ such that w, a leaf not in $x'_{\{i\}}$, receives value i. In S'_i probability of $x'_{\{i\}}$ is as follows.

$$\begin{aligned} S'_0(x'_{\{0\}}) : &= \frac{1}{p_w} S'(x'_{\{0\}} \cup \{(w,0)\}) && (4.15) \\ S'_1(x'_{\{1\}}) : &= \frac{1}{1-p_w} S'(x'_{\{1\}} \cup \{(w,1)\}) && (4.16) \end{aligned}$$

$T^n[i/w]$ is smaller than T^n. By the minimality of T^n, Lemma 4.3 holds for $T^n[i/w]$ and the following inequality holds for some A'_i.

$$\mathrm{cost}(A_i,\ S_i) \geq \mathrm{cost}(A'_i,\ S'_i) \tag{4.17}$$

Then we can define the following algorithm using A'_0 and A'_1.

A'_2 A deterministic algorithm evaluating $T^{n'}$ that probes x_w at the beginning : Depending on the value of x_w, say i (i is 0 or 1), A'_2 behaves the same as A'_i.

By the assumption for a contradiction, the following inequality holds.

$$\mathrm{cost}(A,\ S) < \mathrm{cost}(A'_2,\ S') \tag{4.18}$$

Since $v \neq w$, we have $c_i(w) = c'_i(w)$. Therefore, by (4.17), the following inequality holds.

$$\begin{aligned} \mathrm{cost}(A,\ S) &= p_w(c_0(w) + \mathrm{cost}(A_0,\ S_0)) + (1-p_w)(c_1(w) + \mathrm{cost}(A_1,\ S_1)) \\ &\geq p_w(c'_0(w) + \mathrm{cost}(A'_0, S'_0)) + (1-p_w)(c'_1(w) + \mathrm{cost}(A'_1,\ S'_1)) \\ &= \mathrm{cost}(A'_2,\ S') \end{aligned} \tag{4.19}$$

This inequality contradicts (4.18).

Case 2

The case where w is a child of v. We can assume $w = y_1$ without loss of generality. We prove the case by induction on the number of v's children .

(i) Base case: v has two children. In the same way as Saks-Wigderson [7], we can show that there is an algorithm A' satisfies $\mathrm{cost}(A,\ S) \geq \mathrm{cost}(A',\ S')$.

(ii) Induction step: Assume that Lemma 4.3 holds when v has $n-1$ children. We are going to show it holds when v has n children. Instead of comparing $\mathrm{cost}(A,\ S)$ with $\mathrm{cost}(A',\ S')$ directly, we introduce a new tree T^{n^*} as an intermediate step. We are going to show Lemma 4.3 by comparing each of the above two with an intermediate cost.

$(T^n)^*$ A tree obtained by replacing $y_2, \dots, y_n$ of T^n by y_2^*. We abbreviate it to T^{n^*}.

The cost of y_2^* is defined as follows.

$$c_0^*(y_2^*) = \Psi_{n-1}(c_0(y_2),\dots,c_0(y_n),c_1(y_2),\dots,c_1(y_n))\ ,\ c_1^*(y_2^*) = \sum_{i=2}^{n} c_1(y_i)$$

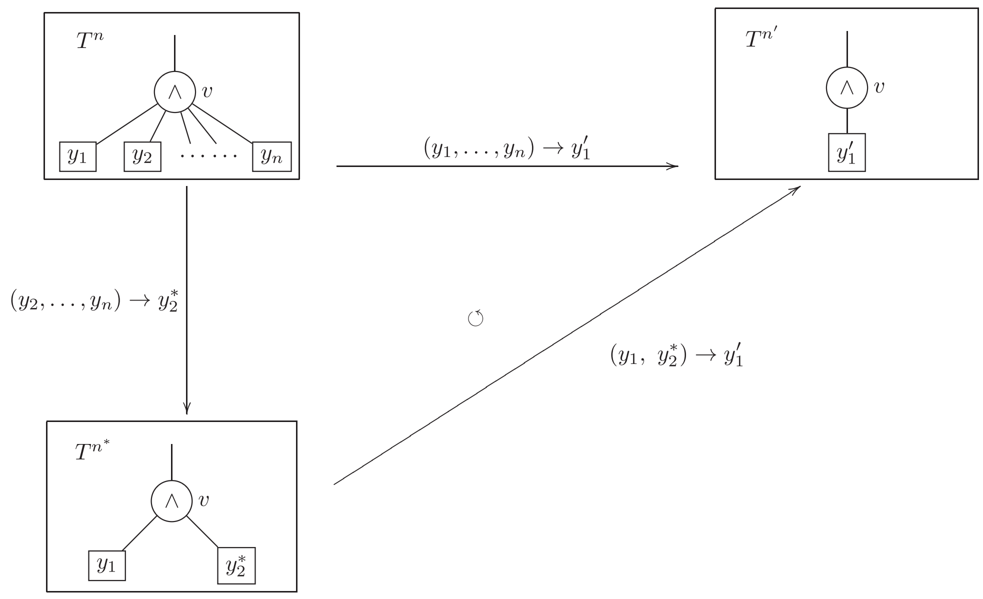

Figure 2: the relationship between T^n, $T^{n'}$ and T^{n^*}.

A^* A deterministic algorithm that evaluates T^n. Only alpha beta prunning algorithms are considered.
S^* Abbreviation of distribution $S'[(y_1,y_2^*;c)/(y'_1;c')]$ on the input of T^{n^*} defined by using S'. We will define it later.

$$\begin{tikzcd}
(S : \text{A distribution on } T^n) & & (S' \ :\ \text{on } T^{n'}) \arrow[ll, "S'[(y_1,\dots,y_n;c)/(y'_1;c')] \mapsfrom S'"'] \arrow[dll, "S'[(y_1,y_2^*;c^*)/(y'_1;c')] \mapsfrom S'"] \\
& \circlearrowright & \\
(S^* \ :\ \text{on } T^{n^*}) \arrow[uu, "S^* \mapsto S^*[(y_2,\dots,y_n;c)/(y_2^*;c^*)]"] & &
\end{tikzcd}$$

Figure 3: the relationship between S, S' and S^*.

We define $S'[(y_1, y_2^*; c)/(y_1'; c')]$ (abbreviated to S^*) as follows.

$$S^*(x) := \begin{cases} 0 & \text{if} \quad (y_1, y_2^*) = (0,0) \\ \dfrac{c_1(y_1)}{c_1(y_1) + c_1^*(y_2^*)} S'(x'(0)) & \text{if} \quad (y_1, y_2^*) = (0,1) \\ \dfrac{c_1^*(y_2^*)}{c_1(y_1) + c_1^*(y_2^*)} S'(x'(0)) & \text{if} \quad (y_1, y_2^*) = (1,0) \\ S'(x'(1)) & \text{if} \quad (y_1, y_2^*) = (1,1) \end{cases} \tag{4.20}$$

For each truth assignment x on T^n, we define truth assignments $x^*(0)$ and $x^*(1)$ of T^{n^*} as follows.
$x^*(i)$ Remove $y_2, \ldots, y_n$ from the domain of x and put y_2^* into the domain. The value of y_2^* is i and the values of the others are as before.

Then we define S as follows. It is consistent with (4.11) which we defined immediately before Lemma 4.3.

$$S(x) := \begin{cases} \dfrac{c_1(y_2)}{c_1(y_2) + \cdots + c_1(y_n)} S^*(x^*(0)) & \text{if} \quad (y_2, \ldots, y_n) = (0, 1, \ldots, 1) \\ \vdots & \vdots \\ \dfrac{c_1(y_n)}{c_1(y_2) + \cdots + c_1(y_n)} S^*(x^*(0)) & \text{if} \quad (y_2, \ldots, y_n) = (1, \ldots, 1, 0) \\ S^*(x^*(1)) & \text{if} \quad (y_2, \ldots, y_n) = (1, 1, \ldots, 1) \\ 0 & \qquad \text{otherwise} \end{cases} \tag{4.21}$$

By using these symbols, we are going to show there is an algorithm A^* (A' , respectively) that satisfies $\text{cost}(A,\ S) \geq \text{cost}(A^*,\ S^*)$ ($\text{cost}(A^*,\ S^*) \geq \text{cost}(A',\ S')$, respectively) .

First, we are going to show there is an algorithm A^* that satisfies $\text{cost}(A,\ S) \geq \text{cost}(A^*,\ S^*)$. To show the existence, we define some symbols.
A_w^* We fix a deterministic algorithm A_w^* on T^{n^*} that probes x_w at the beginning.
We will define A^* by means of A_w^*.
A_i^* When A_w^* probes x_w at the beginning and get value i (i is 0 or 1) then the subsequent move of A_w^* is denoted by A_i^*.
p_{y_1} $p_w = \text{prob}_S[x_w = 0]$ ($= \text{prob}_{S^*}[x_w = 0]$). Remark : S (S^*, respectively) is on T^n (T^{n^*}, respectively)
$T^{n^*}[i/w]$ The tree obtained by substituting i for w on T^{n^*}.
$T^n[i/w]$ The tree obtained by substituting i for w on T^n.
S_i Distributions on $T^n[i/w]$ defined in the same way as (4.13) , (4.14).
S_i^* Distributions on $T^{n^*}[i/w]$. The definition is as follows.

We define S_i^* using S^*. Suppose that $x_{\{i\}}^*$ is a truth assignment on $T^{n^*}[i/w]$. Recall that $x_{\{i\}}^* \cup \{(w,i)\}$ is an assignment on T^{n^*} such that w, a leaf not in $x_{\{i\}}^*$, receives value i. In S_i^* probability of $x_{\{i\}}^*$ is as follows.

$$S_0^*(x^*_{\{0\}}) := \frac{1}{p_w} S^*(x^*_{\{0\}} \cup \{(w,0)\}) \tag{4.22}$$

$$S_1^*(x^*_{\{1\}}) := \frac{1}{1-p_w} S^*(x^*_{\{1\}} \cup \{(w,1)\}) \tag{4.23}$$

Then we can express $\mathrm{cost}(A,\ S)$ and $\mathrm{cost}(A_w^*,\ S^*)$ as follows.

$$\mathrm{cost}(A,\ S) = p_{y_1}\left(c_0(y_1) + \mathrm{cost}(A_0,\ S_0)\right) + (1-p_{y_1})\left(c_1(y_1) + \mathrm{cost}(A_1,\ S_1)\right) \tag{4.24}$$

$$\mathrm{cost}(A_w^*,\ S^*) = p_{y_1}\left(c_0(y_1) + \mathrm{cost}(A_0^*,\ S_0^*)\right) + (1-p_{y_1})\left(c_1(y_1) + \mathrm{cost}(A_1^*,\ S_1^*)\right) \tag{4.25}$$

If we can show the existence of A_0^* and A_1^* that satisfy $\mathrm{cost}(A_0,\ S_0) \geq \mathrm{cost}(A_0^*,\ S_0^*)$ and $\mathrm{cost}(A_1,\ S_1) \geq \mathrm{cost}(A_1^*,\ S_1^*)$, we obtain A_w^* that satisfies $\mathrm{cost}(A,\ S) \geq \mathrm{cost}(A_w^*,\ S^*)$.

1. Comparison of $\mathrm{cost}(A_0,\ S_0)$ and $\mathrm{cost}(A_0^*,\ S_0^*)$
   Since $x_{y_1} = 0$, v's value is determined as to be 0. Thus child nodes of v other than y_1 is not probed (either in A_0 or A_0^*). Therefore in the same way as Case 1, there exists an algorithm A_0^* with the following property.
   $$\mathrm{cost}(A_0,\ S_0) \geq \mathrm{cost}(A_0^*,\ S_0^*) \tag{4.26}$$
2. Comparison of $\mathrm{cost}(A_1,\ S_1)$ and $\mathrm{cost}(A_1^*,\ S_1^*)$
   In $T^n[1/w]$, v has $n-1$ children that A_1 has not probed yet. Therefore we can apply induction hypothesis to $T^n[1/w]$. Hence there is an algorithm A_1^* satisfying $\mathrm{cost}(A_1,\ S_1) \geq \mathrm{cost}(A_1^*,\ S_1^*)$.

Thus we have shown existence of A^* such that $\mathrm{cost}(A,\ S) \geq \mathrm{cost}(A_w^*,\ S^*)$.

Next, we are going to show there is an algorithm A' that satisfies $\mathrm{cost}(A^*,\ S^*) \geq \mathrm{cost}(A',\ S')$. It is not har to show $c_0(y_1) \leq c_0^*(y_2^*) + c_1(y_1)$ and $c_0^*(y_2^*) \leq c_0(y_1) + c_1^*(y_2^*)$ (see the next subsection for the detail). So we can apply induction hypothesis and show the exisitence of A' that satisfies $\mathrm{cost}(A^*,\ S^*) \geq \mathrm{cost}(A',\ S')$.

Thus we have shown that there are algorithms $A^*$and A' that satisfy $\mathrm{cost}(A,\ S) \geq \mathrm{cost}(A^*,\ S^*) \geq \mathrm{cost}(A',\ S')$: In particular we have $\mathrm{cost}(A,\ S) \geq \mathrm{cost}(A',\ S')$. Now we have shown Lemma 4.3. □

Thus we have proved Lemma 3.1. In particular, Theorem 2.1 is proved.

4.3 Remarks

In this subsection, we abbreviate $\Psi_{n-1}(c_0(y_2), \ldots, c_0(y_n), c_1(y_2), \ldots, c_1(y_n))$ to Ψ_{n-1} and $\Psi_n(c_0(y_1), \ldots, c_0(y_n), c_1(y_1), \ldots, c_1(y_n))$ to Ψ_n.

$$\begin{aligned}
\left(\sum_{i=2}^{n} c_1(y_i)\right)\left(c_0(y_1) + \sum_{j=2}^{n} c_1(y_j)\right) &= c_0(y_1)\sum_{i=2}^{n} c_1(y_i) + \sum_{i=2}^{n} c_1(y_i)^2 + 2\sum_{2\leq i<j\leq n} c_1(y_i)c_1(y_j) \\
&= \sum_{i=2}^{n} \left(c_0(y_1) + c_1(y_i)\right) c_1(y_i) + 2\sum_{2\leq i<j\leq n} c_1(y_i)c_1(y_j) \\
&\geq \sum_{i=2}^{n} c_0(y_i)c_1(y_i) + 2\sum_{2\leq i<j\leq n} c_1(y_i)c_1(y_j) \\
&\qquad \text{[From the assumption, } c_0(y_i) \leq c_0(y_1) + c_1(y_i)\text{]} \\
&\geq \sum_{i=2}^{n} c_0(y_i)c_1(y_i) + \sum_{2\leq i<j\leq n} c_1(y_i)c_1(y_j) = \left(\sum_{i=2}^{n} c_1(y_i)\right)\Psi_{n-1}
\end{aligned} \tag{4.27}$$

By $\sum_{i=2}^{n} c_1(y_i) > 0$, we have $\Psi_{n-1} \leq c_0(y_1) + \sum_{i=2}^{n} c_1(y_i)$. Next, we are going to show $c_0(y_1) \leq \Psi_{n-1} + c_1(y_1)$. Note that $\Psi_2\left(\Psi_{n-1}, c_0(y_1), \sum_{i=2}^{n} c_1(y_i), c_1(y_1)\right) = \Psi_n$.

$$\begin{aligned}
& \Psi_{n-1} \leq c_0(y_1) + \sum_{i=2}^{n} c_1(y_i) \\
\Leftrightarrow\quad & c_1(y_1)\Psi_{n-1} \leq c_0(y_1)c_1(y_1) + c_1(y_1)\sum_{i=2}^{n} c_1(y_i) \\
\Leftrightarrow\quad & c_1(y_1)\Psi_{n-1} + \sum_{i=2}^{n} c_1(y_i)\Psi_{n-1} \leq \sum_{i=2}^{n} c_1(y_i)\Psi_{n-1} + c_0(y_1)c_1(y_1) + c_1(y_1)\sum_{i=2}^{n} c_1(y_i) \\
\Leftrightarrow\quad & \sum_{i=1}^{n} c_1(y_i)\Psi_{n-1} \leq \sum_{i=2}^{n} c_1(y_i)\Psi_{n-1} + c_0(y_1)c_1(y_1) + c_1(y_1)\sum_{i=2}^{n} c_1(y_i) \\
\Leftrightarrow\quad & \Psi_{n-1} \leq \left(\sum_{i=2}^{n} c_1(y_i)\Psi_{n-1} + c_0(y_1)c_1(y_1) + c_1(y_1)\sum_{i=2}^{n} c_1(y_i)\right) \Big/ \sum_{i=1}^{n} c_1(y_i)
\end{aligned} \tag{4.28}$$

The right-hand side of the last formula equals $\Psi_2\left(\Psi_{n-1}, c_0(y_1), \sum_{i=2}^{n} c_1(y_i), c_1(y_1)\right) = \Psi_n$. Hence $\Psi_{n-1} \leq \Psi_n$. Now, by means of $\Psi_{n-1} \leq \Psi_n$, $c_0(y_1) \leq c_0(y_2) + c_1(y_1)$, and $\Psi_1(c_0(y_2), c_1(y_2)) = c_0(y_2)$, we have:

$$c_0(y_1) \leq c_0(y_2) + c_1(y_1) \leq \Psi_2(c_0(y_2), c_0(y_3), c_1(y_2), c_1(y_3)) + c_1(y_1) \leq \cdots \leq \Psi_{n-1} + c_1(y_1) \tag{4.29}$$

Thus, we have shown $c_0(y_1) \leq \Psi_{n-1} + c_1(y_1)$.

5 More on the concept of weak balance

In this section we show Theorem 2.3.

Theorem 2.3 *Suppose that T (the height ≥ 1) is an AND-OR tree (or an OR-AND tree) that is weakly balanced in the first sense and both* (2.1) *and* (2.2) *hold at every internal node v. We assume that T is not costed: The cost of each leaf is* 1*. Then T is weakly balanced in the second sense. (Thus, by Theorem 2.1, T is nice.)*

By induction on the height of a tree, we are going to show Theorem 2.3 simultaneously with the following lemma.

Lemma 5.1. *Suppose that T satisfies the assumption of Theorem 2.3. Then for each internal node v, we have the following. Here, we let N denote $N(v)$ and $x_1, \ldots, x_N$ the child nodes of v (Each x_i is either a leaf or a gate whose label, $\wedge$ or $\vee$, is different from that of v).*

1. If v is an AND-node then the following hold.

$$\frac{N+1}{2N}\sum_{i=1}^{N} R_1(x_i) \le \Psi_N(R_0(x_1),\dots,R_0(x_N),R_1(x_1),\dots,R_1(x_N))) \tag{5.1}$$

$$(= \frac{\sum_{i=1}^{N} a_i b_i + \sum_{1\le i<j\le N} b_i b_j}{\sum_{i=1}^{N} b_i})$$

Here we let a_i (b_i, respectively) denote $R_0(x_i)$ ($R_1(x_i)$, respectively).

$$\Psi_N(R_0(x_1),\dots,R_0(x_N),R_1(x_1),\dots,R_1(x_N))) \le \sum_{i=1}^{N} R_1(x_i) \tag{5.2}$$

2. If v is an OR-node then the following hold.

$$\frac{N+1}{2N}\sum_{i=1}^{N} R_0(x_i) \le \Psi_N(R_1(x_1),\dots,R_1(x_N),R_0(x_1),\dots,R_0(x_N))) \tag{5.3}$$

$$(= \frac{\sum_{i=1}^{N} a_i b_i + \sum_{1\le i<j\le N} a_i a_j}{\sum_{i=1}^{N} a_i})$$

$$\Psi_N(R_1(x_1),\dots,R_1(x_N),R_0(x_1),\dots,R_0(x_N))) \le \sum_{i=1}^{N} R_0(x_i) \tag{5.4}$$

In the case where the height is 1 and the root is an AND-node, "weakly balanced in the second sense" $\Longleftrightarrow$ $1 \le 1{+}1$, which is apparently true. Thus Theorem 2.3 holds. We also have $\sum_{i=1}^{N} R_1(x_i) = N$ and $\Psi_N = (N+1)/2$, thus it is easy to verify Lemma 5.1. The case where the height is 1 and the root is an OR-node is similar. Thus the base case is finished.

Now, suppose that T is a tree satisfying the assumption of the theorem and that $h \ge 2$ is its height. As an induction hypothesis, we assume that the theorem and the lemma hold for all trees (satisfying the assumption of the theorem) of lower heights. In particular, each such tree of lower height is nice.

Proof. (Induction step for Lemma 5.1 (5.1)) We are going to show Lemma 5.1 for height h. The case of $N = 2$ is done by Arimoto [1, Lemma 2.2]. Suppose $N \ge 3$. We look at assertion 1 of Lemma 5.1. Assertion 2 may be shown in a similar way.

Claim 1. $\sum_i b_i^2 + \sum_{i<j} b_i b_j \ge \frac{N+1}{2N}(\sum_i b_i)^2$

Proof of Claim 1:

$$\begin{aligned}\sum_i b_i^2 + \sum_{i<j} b_i b_j - \frac{N+1}{2N}(\sum_i b_i)^2 &= \sum_i b_i^2 + \sum_{i<j} b_i b_j - \frac{N+1}{2N}(\sum_i b_i^2 + \sum_{i<j} 2b_i b_j)\\ &= \frac{N-1}{2N}\sum_i b_i^2 - \frac{1}{N}\sum_{i<j} b_i b_j\\ &= \frac{1}{2N}\sum_{i<j}(b_i-b_j)^2 \ge 0 \end{aligned} \tag{5.5}$$

To see the last equality, observe the following fact. For each k $(1 \le k \le N)$, the set $\{(j,k) | 1 \le j < k\} \cup \{(k,j) | k < j \le N\}$ has exactly $N-1 (= (k-1)+(N-k))$ elements. In other words, $(N-1)$-many b_k^2 appears in the expansion of $\sum_{i<j}(b_i - b_j)^2$. Q.E.D. (Claim 1)

Now, by induction hypothesis, Lemma 5.1 and Theorem 2.3 hold at lower levels. Thus we have:

$$\frac{N_i+1}{2N_i} a_i \le b_i \le a_i \tag{5.6}$$

By the second inequality of the above and by (5.5), we have $\frac{N+1}{2N}(\sum_i b_i)^2 \le \sum_i a_i b_i + \sum_{i<j} b_i b_j$. By dividing both sides by $\sum_i b_i$, we get $\frac{N+1}{2N}\sum_i b_i \le \Psi_N$, that is, (5.1) of Lemma 5.1. □

Proof. (Induction step for Lemma 5.1 (5.2) in the case where $N \ge 3$)

For each i, let $N_i = N(x_i)$ (In the case where x_i is a leaf, $N_i = 0$). Without loss of generality, we may assume b_1 is the maximum among $b_1, \dots, b_N$. In other words, the following holds for each i.

$$b_1 \ge b_i \tag{5.7}$$

By our assumption of weakly balanced in the first sense (with both (2.1) and (2.2) at every internal node), we have the following for each i.

$$N \ge N_i \tag{5.8}$$
$$b_i \ge \frac{N-1}{N} b_1 \tag{5.9}$$

Claim 2.

1. $\sum_{i=1}^N a_i b_i \le \frac{2N}{N+1}\sum_{i=1}^N b_i^2$
2. $\sum_{2\le i<j\le N} b_i b_j + \frac{2}{N+1}\sum_{i=2}^N b_i^2 - \frac{N-1}{N+1} b_1^2 \ge 0$

Proof of Claim 2:

By the first inequality of (5.6) and (5.8), we have $a_i \le \frac{2N}{N+1} b_i$. By means of this inequality, we get assertion 1 of Claim 2.

Next, we look at assertion 2 of Claim 2. By means of (5.9), the following is a lower bound of the left-hand side of assertion 2 of Claim 2.

$$\begin{aligned} &\binom{N-1}{2}\left(\frac{N-1}{N}b_1\right)^2 + \frac{2}{N+1}(N-1)\left(\frac{N-1}{N}b_1\right)^2 - \frac{N-1}{N+1}b_1^2 \\ =&\frac{(N-1)b_1^2}{2N^2(N+1)}\left((N-1)^2(N-2)(N+1) + 4(N-1)^2 - 2N^2\right) \end{aligned} \tag{5.10}$$

The rightmost factor of the last formula is positive for $N \ge 3$. Therefore we get assertion 2 of Claim 2. Q.E.D. (Claim 2)

Now we are going to show (5.2) of Lemma 5.1. By multiplying the both sides of (5.2) by $\sum_i b_i$, we know that (5.2) is equivalent to $\sum_i a_i b_i + \sum_{i<j} b_i b_j \le (\sum_i b_i)^2$. By assertion 1 of Claim 2, it is sufficient to show the following is nonnegative.

$$(\sum_i b_i)^2 - \sum_{i<j} b_i b_j - \frac{2N}{N+1}\sum_{i=1}^N b_i^2 = \sum_{i<j} b_i b_j - \frac{N-1}{N+1}\sum_i b_i^2 \tag{5.11}$$

Now, we have $\sum_{i<j} b_i b_j = \sum_{j\ge 2} b_1 b_j + \sum_{2\le i<j} b_i b_j$ and the following.

$$\frac{N-1}{N+1}\sum_i b_i^2 = \frac{N-1}{N+1}b_1^2 + \sum_{i=2}^N b_i^2 - \frac{2}{N+1}\sum_{i=2}^N b_i^2$$

Hence (5.11) equals the following.

$$\sum_{j=2}^N (b_1 - b_j) b_j + \sum_{2\le i<j\le N} b_i b_j + \frac{2}{N+1}\sum_{i=2}^N b_i^2 - \frac{N-1}{N+1} b_1^2$$

By (5.7) and Claim 2 (2), the above is nonnegative. Thus we have shown (5.2) of Lemma 5.1. □

Proof. (Induction step for Theorem 2.3) We are going to show the tree is weakly balanced in the second sense. We begin by notation and a review of our hypotheses. Same as before, we look at an AND-node v assuming that its child nodes are $x_1, \dots, x_N$. Fore each $i \leq N$, we let N_i denote the number of child nodes of x_i. Assume that $x_{i,\ell}$ ($1 \leq \ell \leq N_i$) are child nodes of x_i. For each $x_{i,\ell}$, we let $a_{i,\ell}$ ($b_{i,\ell}$, respectively) denote $R_0(x_{i,\ell})$ ($R_1(x_{i,\ell})$, respectively),

Suppose that i, j are natural numbers such that $1 \leq i \leq j \leq N$. Our goal is $a_i \leq b_i + a_j$. This equation surely holds in the case of $i = j$ and in the case where x_i is a leaf: In the latter case we have $a_i = b_i = 1$. Thus, throughout the rest of the proof, we assume that $i \neq j$ and x_i is not a leaf. Therefore x_i is an OR-node, and each of its child node is either an AND-node or a leaf.

By the induction hypothesis, Lemma 5.1 and Theorem 2.3 hold each $x_{i,\ell}$. Thus for each ℓ, by means of (5.8) (in the form of $N_{i,\ell} \leq N_i$), we have:

$$\frac{N_i + 1}{2N_i} b_{i,\ell} \leq a_{i,\ell} \leq b_{i,\ell} \tag{5.12}$$

Since the tree is assumed to be weakly balanced in the first sense, we have:

$$(N-1)a_i \leq N a_j \tag{5.13}$$

Without loss of generality, we may assume $a_{i,1}$ is the maximum among $a_{i,\ell}$ ($1 \leq \ell \leq N_i$). In other words, the following holds for each ℓ.

$$a_{i,1} \geq a_{i,\ell} \tag{5.14}$$

Claim 3.

1. $a_j \geq \sum_{m=2}^{N_i} a_{i,m}$
2. $\sum_{1 \leq \ell < m \leq N_i} a_{i,\ell} a_{i,m} \leq (\sum_{\ell=1}^{N_i} a_{i,\ell}) a_j$

Proof of Claim 3:

$$\begin{aligned}
a_j - \sum_{m \geq 2} a_{i,m} &\geq \frac{N-1}{N} a_i - \sum_{m \geq 2} a_{i,m} \text{ [By (5.13)]} \\
&= \frac{N-1}{N} \sum_{\ell=1}^{N_i} a_{i,\ell} - \sum_{m \geq 2} a_{i,m} \text{ [The subtree below } x_i \text{ is nice.]} \\
&= \frac{N-1}{N} a_{i,1} - \frac{1}{N} \sum_{m \geq 2} a_{i,m} \\
&\geq \frac{N_i - 1}{N} a_{i,1} - \frac{1}{N} \sum_{m \geq 2} a_{i,m} \text{ [By (5.8)]} \\
&= \frac{1}{N} \sum_{m \geq 2} (a_{i,1} - a_{i,m}) \text{ [It is a sum of } (N_i - 1) \text{ items.]} \\
&\geq 0 \text{ [By (5.14)]}
\end{aligned} \tag{5.15}$$

Thus assertion 1 of Claim 3 holds. In assertion 2, the right-hand side minus the left-hand side is $\sum_{\ell=1}^{N_i} a_{i,\ell}(a_j - \sum_{m > \ell} a_{i,m})$. Here, $a_j - \sum_{m > \ell} a_{i,m} \geq a_j - \sum_{m \geq 2} a_{i,m}$, which is nonnegative by assertion 1. Hence, assertion 2 of Claim 3 holds. Q.E.D.(Claim 3)

We are ready for $a_i \leq b_i + a_j$.

$$\begin{aligned}
(\sum_{\ell} a_{i,\ell})^2 &= \sum_{\ell} a_{i,\ell}^2 + 2 \sum_{\ell < m} a_{i,\ell} a_{i,m} \\
&\leq \sum_{\ell} a_{i,\ell}^2 + \sum_{\ell < m} a_{i,\ell} a_{i,m} + (\sum_{\ell} a_{i,\ell}) a_j \text{ [By Claim 3]} \\
&\leq \sum_{\ell} a_{i,\ell} b_{i,\ell} + \sum_{\ell < m} a_{i,\ell} a_{i,m} + (\sum_{\ell} a_{i,\ell}) a_j \text{ [By (5.12)]}
\end{aligned}$$

By dividing the leftmost side and the rightmost side by $\sum_\ell a_{i,\ell}$, we get:

$$\sum_\ell a_{i,\ell} \leq \Psi_N(b_{i,1}, \dots, a_{i,1}, \dots) + a_j \tag{5.16}$$

Since the subtree below x_i is nice, the left-hand side is a_i, and the right-hand side is $b_i + a_j$. Hence it holds that $a_i \leq b_i + a_j$ □

Acknowledgment

We would like to thank our colleague Fuki Ito for helpful discussions.

References

[1] Arimoto, S. A commentary on the work of Saks and Wigderson 1986: Correlated distributions on an unbalanced binary AND-OR tree. In: *Sūrikaisekikenkyūsho Kōkyūroku*, 2150, Research Institute for Mathematical Sciences, Kyoto University (2020) 1–9.

[2] Baudet, G. M. On the branching factor of the alpha-beta pruning algorithm. *Artif. Intell.*, 10 (1978) 173–199.

[3] Liu, C.G., Tanaka, K. Eigen-distribution on random assignments for game trees. *Inform. Process. Lett.*, 104 (2007) 73–77.

[4] Liu, C.G., Tanaka, K. The computational complexity of game trees by eigen-distribution. In: Dress, A., Xu, Y., Zhu, B. (eds.) COCOA 2007. LNCS 4616 (2007) 323–334, Springe, Heidelberg.

[5] Pearl, J. Asymptotic properties of minimax trees and game-searching procedures. *Artif. Intell.*, 14 (1980) 113–138.

[6] Pearl, J. The solution for the branching factor of the alpha-beta pruning algorithm and its optimality. *Communications of the ACM*, 25 (1982) 559–564.

[7] Saks, M., Wigderson,A. Probabilistic Boolean decision trees and the complexity of evaluating game trees. In: *Proc. 27th IEEE FOCS*, 1986, 29–38.

[8] Suzuki, T. Kazuyuki Tanaka's work on AND-OR trees and subsequent developments. *Annals of the Japan Association for Philosophy of Science*, 25 (2017) 79–88.

`http://www.iaeng.org/IJAM/issues_v42/issue_2/IJAM_42_2_07.pdf`

[9] Tarsi, M. Optimal search on some game trees. *J. ACM*, 30 (1983) 389–396.

[10] Yao, A. C-C. Probabilistic computations: toward a unified measure of complexity. In: *Proc. 18th IEEE FOCS*, 1977, 222–227.

Department of Mathematical Sciences
Tokyo Metropolitan University
Tokyo 192-0397
JAPAN
E-mail address: toshio-suzuki@tmu.ac.jp (the third author)