Information Processing Letters 125 (2017) 41–45

Contents lists available at ScienceDirect

Information Processing Letters

www.elsevier.com/locate/ipl

Optimal depth-first algorithms and equilibria of independent distributions on multi-branching trees

Weiguang Peng [a], NingNing Peng [b,*], KengMeng Ng [c], Kazuyuki Tanaka [a], Yue Yang [d]

[a] Mathematical Institute, Tohoku University, Japan
[b] Department of Mathematics, Wuhan University of Technology, China
[c] School of Physical and Mathematical Sciences, Nanyang Technological University, Singapore
[d] Department of Mathematics, National University of Singapore, Singapore

ARTICLE INFO

Article history:
Received 9 June 2016
Received in revised form 10 April 2017
Accepted 11 May 2017
Available online 17 May 2017
Communicated by R. Uehara

Keywords:
Multi-branching trees
Depth-first algorithms
Independent distribution
Computational complexity
Analysis of algorithms

ABSTRACT

The main purpose of this paper is to answer two questions about the distributional complexity of multi-branching trees. We first show that for any independent distribution d on assignments for a multi-branching tree, a certain directional algorithm DIR_d is optimal among all the depth-first algorithms (including non-directional ones) with respect to d. We next generalize Suzuki–Niida's result on binary trees to the case of multi-branching trees. By means of this result and our optimal algorithm, we show that for any balanced multi-branching AND–OR tree, the optimal distributional complexity among all the independent distributions (ID) is (under an assumption that the probability of the root having value 0 is neither 0 nor 1) actually achieved by an independent and identical distribution (IID).

1. Introduction

In this paper, we investigate the optimal depth-first algorithms and equilibria of independent distributions on multi-branching trees, in which every node may have different numbers of children. The height of a tree is the length of the longest path from the root to its leaves. Here, a balanced multi-branching tree means a tree such that all the non-terminal nodes at the same level have the same number of children and all paths from the root to the leaves are of the same length.

We first review some basic notions and results on game trees. An AND–OR tree (OR–AND tree, respectively) is a multi-branching tree such that the root is labeled AND (OR), and sequentially the internal nodes are level-by-level labeled by OR and AND (AND and OR) alternatively. An assignment for a tree is a mapping from the set of leaves to Boolean values $\{0, 1\}$. By evaluating a tree, we mean to compute the Boolean value of the root. For a given assignment, the cost of computation is defined to be the number of leaves that are queried to evaluate a tree. When we consider probability distributions on the set of assignments, the cost of computation is the expected cost under the given distribution.

An algorithm tells us a priority of searching leaves. An algorithm is called *alpha–beta pruning* if it checks only sufficiently many nodes to determine the value of the current subtree [1]. We assume that all the algorithms discussed here are deterministic alpha–beta pruning algorithms. A *directional* algorithm is one that queries the leaves on a given tree in a fixed order. *SOLVE* is a directional algorithm which evaluates a tree from left to right [3]. A *depth-first algorithm* is one that never jumps to another subtree until it completes evaluating the current one. For a given probabil-

* Corresponding author.
E-mail address: pengn@whut.edu.cn (N. Peng).

http://dx.doi.org/10.1016/j.ipl.2017.05.002
0020-0190/

ity distribution on the assignments, we are seeking for the optimal algorithms that can minimize the expected cost of computation under the given distribution.

The *deterministic complexity* is defined to be the minimum cost to compute the worst assignment for a tree, i.e., $\min_{A_D} \max_{\omega} C(A_D, \omega)$, where $C(A_D, \omega)$ is the cost of an algorithm A_D on an assignment ω, A_D ranges over all the deterministic algorithms and ω ranges over all the assignments. A *randomized algorithm* is a distribution over a family of deterministic algorithms. Then the *randomized complexity* to evaluate a tree is defined as $\min_{A_R} \max_{\omega} C(A_R, \omega)$, where $C(A_R, \omega)$ is the expected cost over the corresponding family of deterministic algorithms, A_R ranges over all the randomized algorithms and ω ranges over all the assignments. Obviously, the randomized complexity is not larger than the deterministic complexity.

Saks and Wigderson [5] calculated the randomized complexity for any balanced n-branching tree (each non-terminal node has n children) with height h to be $\Theta((\frac{n-1+\sqrt{n^2+14n+1}}{4})^h)$. Yao's principle [9] indicates that the randomized complexity is equivalent to the *distributional complexity*, $\max_{d} \min_{A_D} C(A_D, d)$ with A_D ranging over the deterministic algorithms and d over the distributions on assignments. Yao's principle provides a profound perspective to analyze randomized algorithms.

Liu and Tanaka [2], subsequent to the study of Saks and Wigderson, investigated the uniform binary trees from the viewpoint of distributional complexity. A distribution δ is said to achieve the distributional complexity (or equilibrium) if and only if $\min_{A_D} C(A_D, \delta) = \max_{d} \min_{A_D} C(A_D, d)$. They assert that for any uniform binary AND–OR tree, if the equilibrium is achieved by an independent distribution (ID), then it is, in fact, an independent and identical distribution (IID). However, [2] does not include a proof of the assertion. Recently, Suzuki and Niida [7] gave a proof for the case where the probability of the root is constrained for uniform binary trees and showed Liu–Tanaka's assertion.

We treat probability distributions on multi-branching trees. In Section 2, for any ID d, we define a directional algorithm DIR_d, and show it is optimal among all the depth-first algorithms with respect to d for any multi-branching tree. Recall Tarsi's theorem [8] that SOLVE is optimal for IID. Our result is on ID among all the depth-first algorithms (with certain conditions) while Tarsi's theorem is on IID among all the algorithms not necessarily depth-first. In Section 3, we first extend the fundamental relationships between the minimum expected cost and the probability of the root of tree being 0 in [7] to balanced multi-branching trees. Based on this, we show that, for any ID d, there exists an IID d' such that the expected cost with d is not larger than that with d' following DIR_d. Then we establish Liu–Tanaka's assertion for any balanced multi-branching AND–OR tree (under an assumption that the probability of the root having value 0 is neither 0 nor 1).

2. DIR_d is optimal among all the depth-first algorithms

Let Ω be the set of assignments for a given tree. We say $d : \Omega \to [0, 1]$ is an *independent distribution* (denote $d \in \mathrm{ID}$) if there exist p_i's (the probability of the i-th leaf being 0) such that for any $\omega \in \Omega$, $d(\omega) = \prod_{\{i:\ \omega(i)=0\}} p_i \prod_{\{i:\ \omega(i)=1\}} (1 - p_i)$. We say $d \in \mathrm{IID}$ if d is an ID satisfying $p_1 = p_2 = \cdots = p_n$. By $C(A, \omega)$, we denote the number of leaves checked by an algorithm A with an assignment ω.

Given $d \in \mathrm{ID}$ and an algorithm A, for each node σ on T, we define $C_\sigma(A, d)$ and $p_\sigma(d)$ to be the evaluation cost of σ and the probability of σ being 0. Remark that if σ is a leaf, then $C_\sigma(A, d) = 1$ and $p_\sigma(d) = p_i$. If σ is a non-terminal node and A is an algorithm on T_σ, $C_\sigma(A, d)$ is the expected cost of computing the value of σ following A, and $p_\sigma(d)$ is the probability of σ being 0.

For any non-terminal node σ in T, T_σ denotes the subtree of T rooted from σ. For a node σ with n children, $T_{\sigma * i}$ $(1 \le i \le n)$ denotes the i-th subtree under σ from left to right, and particularly for the root λ, its subtree is simplified as T_i. For simplicity, we denote $C(A, d) = C_\lambda(A, d)$ at root λ, and $q_\sigma = p_\sigma(d)$ at any node σ. By q_i, we denote the probability of the root of T_i being 0 with respect to d.

Definition 1. For any uniform binary tree T and $d \in \mathrm{ID}$ on T, the depth-first directional algorithm DIR_d is defined inductively as follows. The basic case is trivial. For the induction case, let $\sigma * i$ $(i = 1, 2)$ be a child of non-terminal node σ, and assume $\mathrm{DIR}_{d_{\sigma * i}}$ has been defined for each subtree $T_{\sigma * i}$.

(1) In the case that σ is labeled $\wedge$, DIR_{d_σ} is the concatenation of $\mathrm{DIR}_{d_{\sigma*1}}$ and $\mathrm{DIR}_{d_{\sigma*2}}$ (denote $\mathrm{DIR}_{d_\sigma} := \mathrm{DIR}_{d_{\sigma*1}} \cdot \mathrm{DIR}_{d_{\sigma*2}}$) if $\frac{C_{\sigma*1}(\mathrm{DIR}_{d_{\sigma*1}},\ d_{\sigma*1})}{q_{\sigma*1}} \le \frac{C_{\sigma*2}(\mathrm{DIR}_{d_{\sigma*2}},\ d_{\sigma*2})}{q_{\sigma*2}}$, otherwise $\mathrm{DIR}_{d_\sigma} := \mathrm{DIR}_{d_{\sigma*2}} \cdot \mathrm{DIR}_{d_{\sigma*1}}$.

(2) In the case that σ is labeled $\vee$, $\mathrm{DIR}_{d_\sigma} := \mathrm{DIR}_{d_{\sigma*1}} \cdot \mathrm{DIR}_{d_{\sigma*2}}$ if $\frac{C_{\sigma*1}(\mathrm{DIR}_{d_{\sigma*1}},\ d_{\sigma*1})}{1-q_{\sigma*1}} \le \frac{C_{\sigma*2}(\mathrm{DIR}_{d_{\sigma*2}},\ d_{\sigma*2})}{1-q_{\sigma*2}}$, otherwise $\mathrm{DIR}_{d_\sigma} := \mathrm{DIR}_{d_{\sigma*2}} \cdot \mathrm{DIR}_{d_{\sigma*1}}$.

Theorem 1. *For any uniform binary tree T and $d \in \mathrm{ID}$, if A is any depth-first algorithm, then $C(A, d) \ge C(\mathrm{DIR}_d, d)$, i.e., DIR_d is optimal among all the depth-first algorithms.*

Proof. We prove this by induction on height h. The base case is trivial. For the induction step, let T be a uniform binary tree with height $h + 1$, where the root λ is labeled $\wedge$. The other case can be shown similarly.

Suppose that DIR_{d_i} is optimal for each subtree T_i with height h. Let Ω_{h+1} be the set of assignments for T, Ω_h and Ω'_h the set of assignments for T_1 and T_2. For any $d \in \mathrm{ID}$ on T, there exist d_i for $T_i (i = 1, 2)$ such that $d(\omega) = d_1(\omega_1) \times d_2(\omega_2)$, where $\omega = \omega_1\omega_2$, $\omega_1 \in \Omega_h$ and $\omega_2 \in \Omega'_h$. For any depth-first algorithm A and $d \in \mathrm{ID}$, if A evaluates the subtree T_1 first, then $C(A, d) = \sum_{\omega \in \Omega_{h+1}} C(A, \omega) \cdot d(\omega) = \sum_{\omega \in \Omega^0} C(A, \omega) \cdot d(\omega) + \sum_{\omega \in \Omega^1} C(A, \omega) \cdot d(\omega)$, where $\Omega^i := \{\omega \in \Omega_{h+1} \mid$ the root of T_1 has value i with $\omega\}$.

Assume A is a depth-first non-directional algorithm. By A_1, we denote the algorithm of A for T_1, and by A_{ω_1}