Equilibriums of independent distributions on uniform AND-OR trees

NingNing Peng
(KengMeng NG and Kazuyuki Tanaka, Yang Yue)

Nanyang Technological University, Singapore
nnpeng@ntu.edu.sg

Computability Theory and Foundations of Mathematics,
Sep. 7 - 11, 2015

Outline

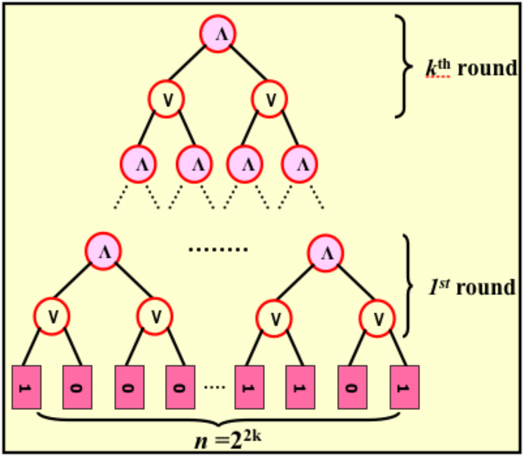
Abstract

In 2007, Liu and Tanaka showed that for any uniform binary AND-OR tree on the assignments that are independently distributed (ID), the distributional complexity is achieved only if the assignments are also identically distributed (IID).

We generalize Liu-Tanaka's result to uniform level-by-level k-branching AND-OR tree. The proof technique is different from available ones. One ingredient of our proof is a generalization of Suzuki-Niida's "fundamental relationships between costs and probabilities". Another ingredient of our proof is a careful analysis of the algorithms involved.

I. Background

Game tree: T_2^k



- **Round:**
  - **One level AND ($\wedge$) node followed by one level OR ($\vee$) node.**
- **Internal nodes:**
  - **$\wedge$ or $\vee$ labeled.**
- **External nodes (leaves):**
  - **1 or 0 labeled .**
  - **The number of leaves: 2^{2k}**

Game tree

We are interested in the class of Boolean functions represented by game trees.

- A game tree is a rooted tree in which each leaf (external node) has a distinct input variable, the internal nodes are labeled by AND / OR.
- A game tree is uniform if the internal nodes have same number of children and the root-leaf paths are of same length.

Boolean function

- Boolean variables $x_1, x_2, \cdots, x_n$ with unknown values
- Given Boolean function $f : \{0,1\}^n \to \{0,1\}$
- Goal: evaluate $f(x_1, \cdots, x_n)$.

Definition

The Boolean Decision tree model as a deterministic algorithm to compute a Boolean function.

Example

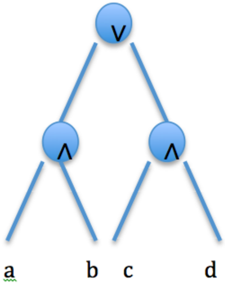
Give a Boolean function $f(a, b, c, d) = (a \wedge b) \vee (c \wedge d)$.

One of decision tree (algorithm) computing f.

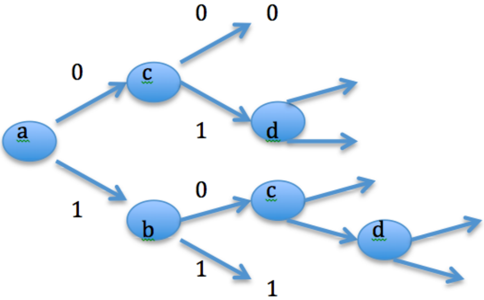
Example

Give a Boolean function $f(a, b, c, d) = (a \wedge b) \vee (c \wedge d)$.

One of decision tree (algorithm) computing f.



Game tree: Functions



Decision tree: Algorithm

Deterministic Complexity: $D(f)$

- The deterministic complexity of function f:
  $D(f) = \min_A \max_\omega C(A, \omega)$.

Deterministic Complexity: $D(f)$

- The deterministic complexity of function f:
  $D(f) = \min_A \max_\omega C(A, \omega)$.

Definition

The deterministic complexity $D(f)$ of a function $f(x_1, \cdots, x_n)$ is the minimum complexity of any deterministic decision tree algorithm that computes f.

Let A be a deterministic algorithm and ω an assignment to the leaves of tree T_2^k.

(A, ω) : the number of leaves queried by A computing T_2^k on ω.

$\mathcal{W}$: be the set of assignments.

p_ω^d : the probability of ω over $\mathcal{W}$ with respect to distribution d.

The average complexity $C(A, d)$ of a deterministic algorithm A on assignments with distribution d is defined by

$$C(A, d) = \sum_{\omega \in \mathcal{W}} p_\omega^d C(A, \omega).$$

Let $\mathcal{D}$ be the set of distributions, and $\mathcal{A}(T_2^k)$ the set of deterministic algorithms computing tree T_2^k. The **distributional complexity** $P(T_2^k)$ computing tree T_2^k is defined by

$$P(T_2^k) = \max_{d \in \mathcal{D}} \min_{A \in \mathcal{A}(T_2^k)} C(A, d).$$

Algorithms and distributions

- Directional Algorithms
- Depth first Algorithms
- independently distributed (ID)
- independently and identically distributed (IID)

Theorem (Tarsi, 1983)

For T_2^k, algorithm SOLVE is the optimal for solving tree.

Theorems

Theorem (Liu and Tanaka, 2007)

For T_2^k, $P_{ID}(T_2^k) = P_{IID}(T_2^k)$.

Theorems

Theorem (Liu and Tanaka, 2007)

For T_2^k, $P_{ID}(T_2^k) = P_{IID}(T_2^k)$.

Theorem (Suzuki and Niida, 2014)

Suppose that r is a real number such that $0 < r < 1$. Suppose that we restrict ourselves to distributions such that the probability of the root is r. Then, for T_2^k, $P_{ID}(T_2^k) = P_{IID}(T_2^k)$.

Suzuku-Niida change the problem into a Extremum Problem.

Theorems

Theorem (Liu and Tanaka, 2007)

For T_2^k, $P_{ID}(T_2^k) = P_{IID}(T_2^k)$.

Theorem (Suzuki and Niida, 2014)

Suppose that r is a real number such that $0 < r < 1$. Suppose that we restrict ourselves to distributions such that the probability of the root is r. Then, for T_2^k, $P_{ID}(T_2^k) = P_{IID}(T_2^k)$.

Suzuku-Niida change the problem into a Extremum Problem.

Theorem

For level-by-level uniform multi-branching tree T, $P_{ID}(T) = P_{IID}(T)$.

We also keep the probability same.

II. A generalization

Definition

We say that a finite branching tree T is a *level-by-level uniform multi-branching* if

(1) T is an AND-OR tree.

(2) For all σ and σ' on T, if $|\sigma| = |\sigma'|$ then σ has the same member of children as σ' does. Note that we do not require that nodes from different levels have the same number of children.

We now prove a technical lemma which is a generalization of Suzuki and Niida "fundamental relationships between costs and probabilities".

Lemma

Suppose that the distribution on T is IID with all leaves assigned probability x and we follow a depth-first algorithm A. Then

(1) $p_\sigma(x)$ *is a strictly increasing function of* x.

(2) $\frac{c_\sigma(x)}{p_\sigma(x)}$ *is strictly decreasing.*

(3) $\frac{c'_\sigma(x)}{p'_\sigma(x)}$ *is strictly decreasing if* σ *is not a leaf; and at leaves* $\frac{c'_\sigma(x)}{p'_\sigma(x)} = 0$ *is nonincreasing.*

Main Theorem

Theorem

For level-by-level uniform multi-branching tree T, $P_{ID}(T) = P_{IID}(T)$.

The Technique Lemma

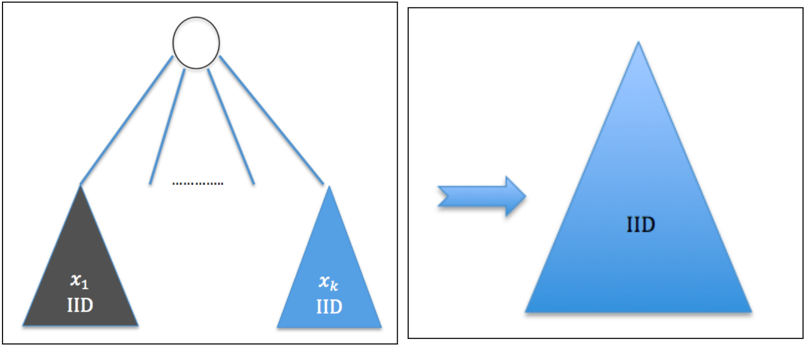
Lemma

Given a ID distribution d, we find an depth first algorithm A and an IID distribution d' such that $p_\sigma(d) = p_\sigma(d')$.

Corollary

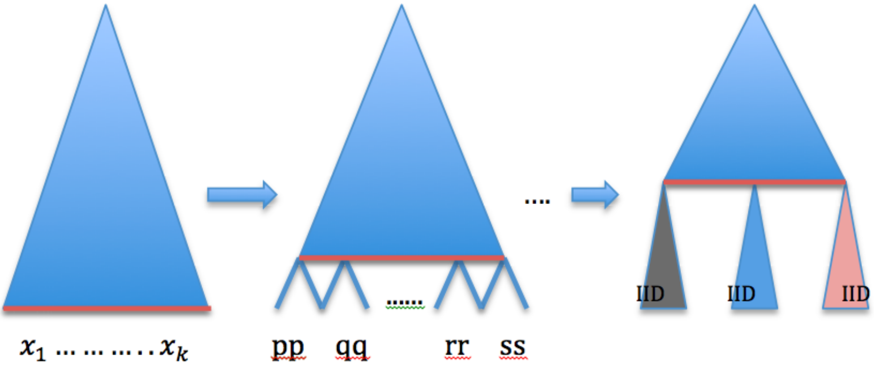
Given a node σ and a almost IID distribution $d = (x_1, x_2, \cdots, x_n)$, then we can find another $d^{'}$, such that $c(A, d') \geq c(A, d)$ while keep the probability $p_\sigma(d) = p_\sigma(d')$.

Proof Idea: the one Step (The Technique Lemma)



- One step change from almost IID to full IID (Lemma 2).
- $C(A, d) \leq C(A, d')$ and $p(A, d) = p(A, d')$.
- inequality imply non maximnm, in other word, maximnm imply equality.

The Induction Step (The Technique Lemma)



Proof.

We first show that $P_{ID} \le P_{IID}$. By the previous lemma, for any ID d, there exists IID d' such that

$$\min_{A_D} c(A_D, d) \le c(SOLVE, d') = P_{IID}$$

for all depth-first algorithms A_D (the last equality is due to Tarsi [**?**]).
When we allow A to be non depth-first, the minimal value cannot increase, thus we have

$$\min_{A} c(A, d) \le c(SOLVE, d') \le P_{IID}$$

hence

$$P_{ID}(T) = \max_{d} \min_{A} c(A, d) \le P_{IID}.$$

On the other hand, any IID is also ID, hence by logic

$$\max_{d:IID} \min_{A} c(A, d) \le \max_{d:ID} \min_{A} (A, d),$$

that is, $P_{IID}(T) \le P_{ID}$. We are done.

Open Questions

Question

What is the optimal algorithm for ID case?

References

Michael Tasi
Optimal Search on Some Game Trees.
Journal of the ACM, Vol.30 (3), PP. 389-396, 1983.

Liu C.G., and Tanaka K.:
Eigen-Distribution on Random Assignments for Game Trees.
Information Processing Letters, 104(2): 73-77, 2007.

Toshio Suzuki and Yoshinao Niida
Equilibrium Points of an AND-OR Tree: under Constraints on Probability.
arXiv:1401.8175.

Thank you very much!