On the Expressive Power of Tree-Structured Probabilistic Circuits

Lang Yin, Han Zhao

University of Illinois Urbana-Champaign

Motivation

Probabilistic circuits (PCs) have emerged as a powerful framework for efficient and exact probabilistic inference. Nevertheless, most PC learning algorithms can only output efficient circuits with a structure of a directed acyclic graph (DAG), but inefficient tree-structured circuits.

Question: Is there truly an exponential gap between DAGs and trees for the PC structure?

Our answer: Not quite. Even in the worst case, there is only a sub-exponential gap. But, with a depth restriction, a sub-exponential separation is inevitable.

Our Contributions

We provide results from two sides, inspired by previous works in circuit complexity theory.

- We proved, with a comprehensive algorithm, that for a network polynomial that can be computed efficiently with a DAG-structured PC, there always exists a sub-exponentially-sized tree-structured PC to represent it.
- Although conditional and not tight, we proved that, under a restriction on the depth of the trees, there exists a strictly sub-exponential separation between tree and DAG-structured PCs.

Key Definitions

Probabilistic Circuits (PCs):

- A **probabilistic circuit (PC)** is a rooted DAG whose leaves represent indicator variables, and internal nodes are sum and product nodes.
- Edges from sum nodes to their children must have positive weights.
- The value of a product node is the product of the values of its children. The value of a sum node is the weighted sum of the values of its children. The value of a PC is the value of its root.

Scope: The **scope** of a node is the set of variables whose indicators are descendants of the node

Decomposability: A PC is **decomposable** if and only if the children of every product node have disjoint scopes.

Smoothness: A PC is **smooth** if and only if the children of every sum node have the same scope.

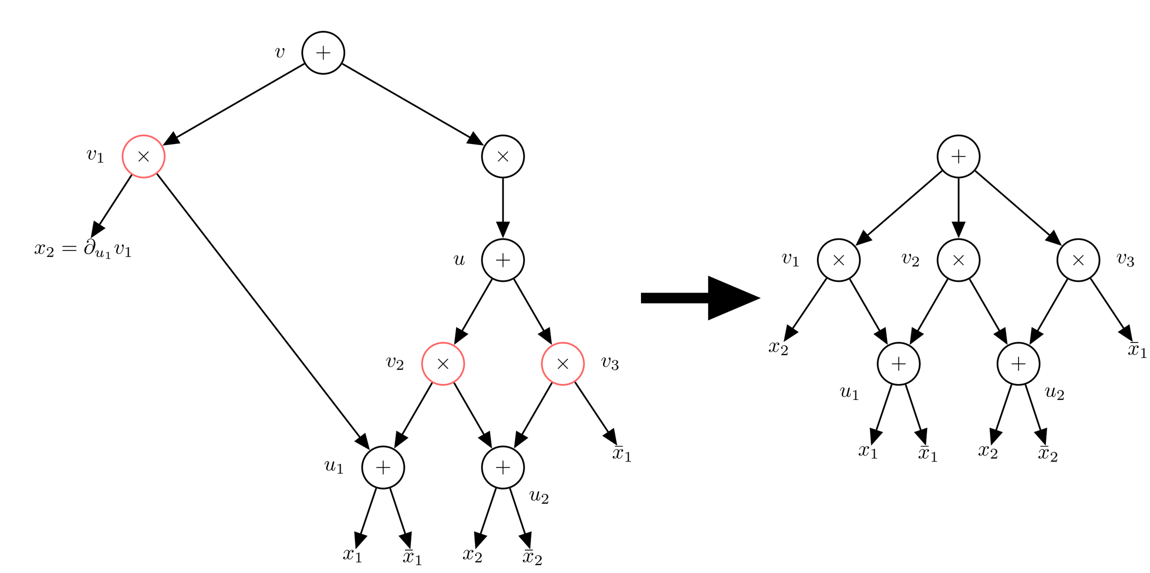
Validity: A PC is **valid** if and only if it is both decomposable and smooth.

Result One: A Universal Upper Bound

Theorem 1. For any DAG-structured PC with n variables and of size poly(n), there exists an equivalent, tree-structured PC of depth $O(\log n)$ and size $n^{O(\log n)}$.

The proof is constructive inspired by methods developed firstly by Valiant et al (1983) and later optimized by Raz and Yehudayoff (2008). The proof has two major steps.

- Transform the original DAG of size poly(n) to another DAG of depth $O(\log n)$ and size poly(n).
  - Reduce the depth from $O(\log^2 n)$ in the original algorithm to $O(\log n)$.
  - Must maintain decomposability and smoothness during the depth reduction.
- Use standard duplication to transform the new DAG to a tree of size $n^{O(\log n)}$.



The process of transforming an arbitrary DAG to a DAG with depth restriction. The red nodes are those that were critical in the transformation

Result Two: A Conditional Lower Bound

Theorem 2. There is a distribution so that any tree-structured PC of depth $o(\log n)$ computing that distribution must have size $n^{\omega(1)}$.

The proof conducts a reduction to a result in arithmetic trees by [Fournier et al. 2023].

- Given an eligible, minimum tree-structured PC T of depth $o(\log n)$.
- Removing negative indicators in T to make it an arithmetic tree T'
  - Must ensure that T' computes a polynomial specified in [Fournier et al. 2023].
  - Must utilize the structure derived from decomposability and smoothness to verify the distribution computed by T'.
- By Theorem 3 in [Fournier et al. 2023], the new tree T' must have size at least $n^{\omega(1)}$.
- The original tree T must be even larger.

Open Problems

- Tightness of the upper bound: is $n^{O(\log n)}$ the best one can achieve?
- Conditions on the lower bound: can we remove or at least relax the depth restriction $o(\log n)$?
- Refinements of the lower bound: with or without the depth restriction, may we obtain a more concrete exponent than $\omega(1)$?
- Ultimate goal: finding or proving the impossibility of a pair of identical upper and lower bounds

References

- Leslie G. Valiant, Sven Skyum, Stuart J. Berkowitz, and Charles Rackoff. Fast parallel computation of polynomials using few processors. *SIAM J. Comput.*, 12:641–644, 1983.
- Ran Raz and Amir Yehudayoff. Balancing syntactically multilinear arithmetic circuits. *Computational Complexity*, 17:515–535, 2008.
- Hervé Fournier, Nutan Limaye, Guillaume Malod, Srikanth Srinivasan, and Sébastien Tavenas. Towards optimal depth-reductions for algebraic formulas. In *Proceedings of the Conference on Proceedings of the 38th Computational Complexity Conference*, CCC '23.