POSTER: Optimizing Zero-Knowledge Proofs for Verifiable Decision Trees

Hanwei Zhu
School of Computing
Australian National Univerisity
Canberra, Australia
henry.zhu@anu.edu.au

Sid Chi-Kin Chau
Data61
CSIRO
Sydney, Australia
sid.chau@data61.csiro.au

***Abstract*—In the evolving landscape of machine learning security, the integrity and confidentiality of decision tree models are paramount. This paper presents a groundbreaking approach in the realm of zero-knowledge proofs (ZKPs) for decision trees, addressing core challenges in privacy, verifiability, and efficiency. We introduce a technique that replaces traditional hash function commitments with constraint-based polynomial commitments, significantly reducing the number of multiplication gates and enhancing computational efficiency. A pivotal feature of our work is the decoupling of decision tree parameters from the tree structure itself. This modular design not only facilitates dynamic and seamless updates to the model parameters without overhauling the entire tree but also maintains the model's integrity and security.**

***Index Terms*—ML integrity, zk-snarks, decision trees**

I. Introduction

As machine learning models increasingly play a crucial role across a wide range of sectors, from healthcare and finance to autonomous driving and beyond, the imperative to safeguard their integrity and accuracy, while simultaneously ensuring the protection of the intellectual property rights inherent in these sophisticated algorithms, is becoming more pressing than ever before. This paper contributes a novel perspective to the domain by presenting a verification framework distinct from papers in the literature [1], [2], which predominantly rely on computationally intensive Merkle trees and hash functions within zero-knowledge proofs such as zk-SNARKs. Our approach significantly reduces the number of multiplication gates required for proof generation, enhancing computational efficiency. Contrary to other studies that bind decision trees to their parameters, our method introduces the flexibility of an "empty tree" structure with updatable parameters. This adaptability facilitates ongoing iterations and model refinements without compromising the verification process.

II. Our Approach

Our method encapsulates a three-step process: converting a decision tree inference problem into a constraint system, incorporating tree parameters and input data, and encoding the entire framework into a Polynomial Interactive Oracle Proof (IOP), as illustrated in Fig 1.

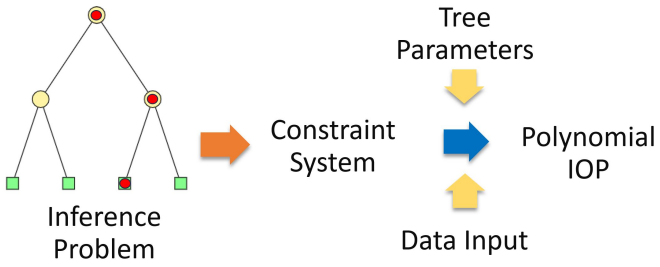

Fig. 1: Our Zero-knowledge Decision Tree Construction

A. Decision Tree Inference

Each internal node in a decision tree signifies a decision based on an attribute, with branches indicating outcomes and leaf nodes yielding the final prediction. Inference in decision trees proceeds by navigating from the root to an appropriate leaf, making sequential decisions at each node based on input attributes, until a final prediction is reached. In the paper, we assume the decision trees are binary and balanced, with a node's condition deemed satisfied (i.e., moving to its right child) only if the input exceeds a set threshold.

B. Constraint System Formulation

Following the inference procedure, we formalize the tree's logic into a constraint system. This transformation encapsulates the decision criteria and outcomes of the binary and balanced decision tree into a series of equations.

We first introduce 2 constraint gadgets in our design.
Positivity: A constraint system is satisfiable if a value $w > 0$ and a binary indicator $a = 1$, or $w \leq 0$ and $a = 0$. For simplicity, we write positivity satisfaction as $Pos(\cdot) = 1$.
Zeroality: The constraint system holds if $w = 0$ and $a = 1$, or $w \neq 0$ and $a = 0$. This can test equality by setting w as the difference of two values. Denote $Zero(\cdot) = 1$ as zeroality satisfaction.

Next, we elucidate the inference constraints of our decision tree. The system is segmented into input-threshold mapping, node evaluation, and path traversal. Given a balanced decision tree $\mathtt{DT}_{[h]}$ with height h, it includes a root, $2^h - 2$ internal nodes, and 2^h leaf nodes. A user provides data input $X = [x_1, \ldots, x_n]$, while the tree owner specifies an

attribute selector $S = [s_1, \ldots, s_{2^h-1}]$ and a threshold array $T = [t_1, \ldots, t_{2^h-1}]$, which guide comparisons at non-leaf nodes. The owner defines a label array $L = [l_{2^h}, \ldots, l_{2^{h+1}-1}]$ representing outcomes at leaf nodes, culminating in an output label y.

Component 1: Input-Threshold Mapping

This component aims to map input data to specific decision thresholds within the tree. It consists of the following equations:

$$\begin{cases} \sum_{p=1}^{n} x_{[p]} \cdot v_{[j,p]} = x_{[s_{[j]}]}, & \text{for } 1 \le j \le 2^h - 1 \\ v_{[j,p]} \cdot (1 - v_{[j,p]}) = 0, & \text{for } 1 \le j \le 2^h - 1, 1 \le p \le n \\ s_{[j]} - p = \omega_{[j,p]}, & \text{for } 1 \le j \le 2^h - 1, 1 \le p \le n \\ Zero(\omega_{[j,p]}) \cdot v_{[j,p]} = 0, & \text{for } 1 \le j \le 2^h - 1, 1 \le p \le n \\ (1 - Zero(\omega_{[j,p]})) \cdot & \\ \qquad (1 - v_{[j,p]}) = 0, & \text{for } 1 \le j \le 2^h - 1, 1 \le p \le n \end{cases}$$

The equation $\sum_{p=1}^{n} x_{[p]} \cdot v_{[j,p]} = x_{[s_{[j]}]}$ serves as a selector mechanism to identify the appropriate threshold $s_{[j]}$ for each input vector x. The binary variable $v_{[j,p]}$ is assigned a value of 1 when the index p of the input data x corresponds to the correct threshold index $s_{[j]}$. Additionally, $\omega_{[j,p]}$ ensures that the mapping operates correctly by representing the difference between the selector index $s_{[j]}$ and all potential indices p.

Component 2: Node Evaluation

This segment of the constraint system focuses on evaluating the conditions at each internal node by comparing input values to predetermined thresholds. The system is defined as follows:

$$\begin{cases} x_{[s_{[j]}]} - t_{[j]} = w_{[j]}, & \text{for } 1 \le j \le 2^h - 1 \\ c_{[j]} \cdot (1 - c_{[j]}) = 0, & \text{for } 1 \le j \le 2^h - 1 \\ Pos(w_{[j]}) \cdot c_{[j]} = 0, & \text{for } 1 \le j \le 2^h - 1 \\ (1 - Pos(w_{[j]})) \cdot (1 - c_{[j]}) = 0, & \text{for } 1 \le j \le 2^h - 1 \end{cases}$$

The equation $x_{[s_{[j]}]} - t_{[j]} = w_{[j]}$ is used to calculate the difference between the selected input value $x_{[s_{[j]}]}$ and the threshold value $t_{[j]}$ at each internal node j. The binary Boolean-like variable $c_{[i]}$ indicates whether the condition at a given node is satisfied, with a value of 1 meaning 'True' and 0 meaning 'False'.

Component 3: Path Traversal

This component is concerned with determining the appropriate leaf node based on the satisfaction of conditions along the path leading to it. The equations are formulated below.

The equation $\sum_{i=2^h}^{2^{h+1}-1} l_{[i]} \cdot z_{[i]} = y$ provides a method for identifying a single leaf node y, conditional on the satisfaction of all constraints $c_{[i]}$ along its path. Specifically, each term $l_{[i]}$ in the summation signifies a leaf node, and $c_{[i]}$ represents the conditions being met along the path to the corresponding leaf node i. The binary variable $\beta_{[i,a]}$ and its constraints ensure that the correct leaf node and its ancestral nodes are uniquely identified.

$$\begin{cases} \beta_{[i,a]} \cdot (1 - \beta_{[i,a]}) = 0, & \text{for } 2^h \le i \le 2^{h+1} - 1, \\ & \quad 1 \le a \le h + 1 \\ \sum_{a=1}^{h+1} \beta_{[i,a]} \cdot 2^{a-1} = i, & \text{for } 2^h \le i \le 2^{h+1} - 1 \\ \prod_{d=1}^{h} \Delta \cdot \Theta + & \\ \qquad (1 - \Delta) \cdot (1 - \Theta) = z_{[i]}, & \text{for } 2^h \le i \le 2^{h+1} - 1 \\ \sum_{i=2^h}^{2^{h+1}-1} l_{[i]} \cdot z_{[i]} = y, & \end{cases}$$

where $\Delta = c_{[\sum_{f=1}^{d} \beta_{[i,(h-f+2)]} \cdot 2^{d-f}]}$, and $\Theta = \beta_{[i,h-d+1]}$.

C. Encoding into Polynomial IOP

Finally, building upon the constraint system, we encode the decision tree's structured logic into a Polynomial Interactive Oracle Proof (IOP). In this paper, we utilize Sonic zk-SNARK [3]. This system translates the constraint system into three polynomial equations, as illustrated below.

$$r[X,Y] \triangleq \sum_{i=1}^{N} \mathbf{a}_i (XY)^i + \sum_{i=1}^{N} \mathbf{b}_i (XY)^{-i} + \sum_{i=1}^{N} \mathbf{c}_i (XY)^{i-N}$$

$$s[X,Y] \triangleq \sum_{i=1}^{N} \hat{\mathbf{u}}_i [Y] X^{-i} + \sum_{i=1}^{N} \hat{\mathbf{v}}_i [Y] X^{i} + \sum_{i=1}^{N} \hat{\mathbf{w}}_i [Y] X^{i+N}$$

$$t[X,Y] \triangleq r[X,1](r[X,Y] + s[X,Y]) - \hat{k}[Y]$$

Specifically, we map the equations of the constraint system to vectors $\mathbf{a}$, $\mathbf{b}$, and $\mathbf{c}$ to represent multiplication constraints, and vectors $\hat{\mathbf{u}}$, $\hat{\mathbf{v}}$, $\hat{\mathbf{w}}$, and $\hat{k}$ to represent linear constraints. Unlike the approaches in [1], [2], our framework uniquely generates the polynomials $r[X,Y]$, $s[X,Y]$, and $t[X,Y]$ by leveraging the constraint system independently from the secret tree parameters and data inputs. Then, to prove the validity of the system, a prover needs to commit to $t[X,Y]$ using a polynomial commitment scheme. The verifier's task is to confirm whether $t[X,Y]$ possesses a zero constant term. Since the constant term of $t[X,Y]$ reflects the constraint system's satisfaction, verifying its nullity is tantamount to affirming the system's fulfillment. This verification step upholds the integrity of the decision tree's inference process while preserving the zero-knowledge property.

REFERENCES

[1] H. Wang, Y. Deng, and X. Xie, "Public verifiable private decision tree prediction," in *Information Security and Cryptology: 16th International Conference, Inscrypt 2020, Guangzhou, China, December 11–14, 2020, Revised Selected Papers*. Berlin, Heidelberg: Springer-Verlag, 2020, p. 247–256.

[2] J. Zhang, Z. Fang, Y. Zhang, and D. Song, "Zero knowledge proofs for decision tree predictions and accuracy," in *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 2039–2053.

[3] M. Maller, S. Bowe, M. Kohlweiss, and S. Meiklejohn, "Sonic: Zero-knowledge snarks from linear-size universal and updateable structured reference strings," Cryptology ePrint Archive, Paper 2019/099, 2019.

Optimizing Zero-Knowledge Proofs for Verifiable Decision Trees

Hanwei Zhu Sid Chi-Kin Chau
Australian National University Data61, CSIRO

Introduction

Context and Importance

- **Pervasiveness of Machine Learning:** Essential in healthcare, finance, and autonomous driving with a rapidly growing role in decision-making.
- **Necessity for Integrity and Security:** Integrity and accuracy are critical as they impact human lives and financial stability; reliability is a societal imperative.

Problem Statement

- **Verification Challenges:** Verifying ML model is difficult due to restricted access to proprietary and sensitive information; existing methods lack scalability and are computationally intensive [1, 2].
- **Intellectual Property Concerns:** Protecting intellectual property in algorithms is crucial for maintaining competitive advantages and promoting innovation.

Research Gap

- **Limitations of Existing Solutions:** Current methods bind models to parameters, reducing flexibility and requiring expensive hash computations.

Significance

- **Enhanced Verification Processes:** Research improves verification, ensuring safer, more reliable ML applications.
- **Strengthened Stakeholder Confidence:** Ensures trust in model integrity while safeguarding intellectual property.
- **Support for Sustainable AI Deployment:** Promotes secure, adaptable AI technology deployment.
- **Blockchain Readiness:** The reduced computational complexity makes the system suitable for blockchain applications, enhancing transparency and security.

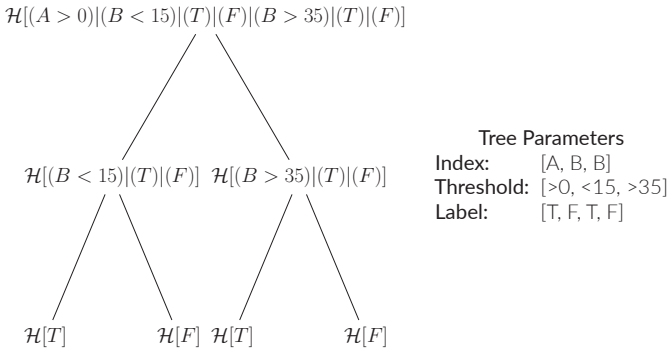

Figure 1. An Example of Existing Merkle Tree-based Approach

Research Objectives

- **Contribution of This Research:**
  - Develops a novel framework that reduces multiplication gates in proof generation, enhancing computational efficiency.
  - Utilizes an "empty tree" structure for dynamic parameter updates, maintaining high security and verifiability without re-verification.
  - Offers major enhancements over traditional methods, allowing models to adapt swiftly to new data or changes without full re-verification.

Our Approach

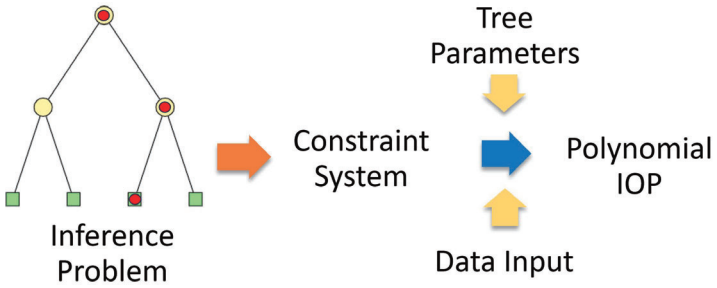

Figure 2. Our Zero-knowledge Decision Tree Construction

Decision Tree Inference Problem

- **Node Significance:** Each internal node in a decision tree represents a decision based on an attribute, leading to various outcomes.
- **Branches and Outcomes:** Branches from each node indicate possible outcomes, with leaf nodes providing the final prediction.
- **Inference Process:** Inference involves navigating from the root to an appropriate leaf, making sequential decisions at each node based on the input attributes.
- **Tree Characteristics:** We assume decision trees are binary and balanced.
- **Decision Criteria:** A node's condition is considered satisfied (leading to the right child) only if the input exceeds a set threshold.

Constraint System Formulation

- **Input-Threshold Mapping:** Maps data inputs to decision thresholds for node comparisons.
- **Node Evaluation:** Assesses conditions at internal nodes against thresholds.
- **Path Traversal:** Identifies the correct leaf node based on path conditions.

Polynomial IOP Implementation

- **Framework Utilization:** Uses Sonic zk-SNARK [3] to transform decision tree logic into Polynomial IOP with separate tree parameter and user data input.
- **Equation Formulation:** Translates constraints into three main polynomial equations:
  - $r[X, Y]$ - Handles multiplication constraints.
  - $s[X, Y]$ - Manages additive constraints.
  - $t[X, Y]$ - Final polynomial for verification.
- **Efficiency and Innovation:** Reduces computational complexity by optimizing polynomial generation separate from tree parameters and data inputs.
- **Verification Process:** Implements a zero-knowledge proof by verifying if $t[X, Y]$ has a zero constant term, ensuring system integrity without compromising privacy.

References

[1] H. Wang, Y. Deng, and X. Xie, "Public verifiable private decision tree prediction," in *Information Security and Cryptology: 16th International Conference, Inscrypt 2020, Guangzhou, China, December 11–14, 2020, Revised Selected Papers*. Berlin, Heidelberg: Springer-Verlag, 2020, p. 247–256.

[2] J. Zhang, Z. Fang, Y. Zhang, and D. Song, "Zero knowledge proofs for decision tree predictions and accuracy," in *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 2039–2053.

[3] M. Maller, S. Bowe, M. Kohlweiss, and S. Meiklejohn, "Sonic: Zero-knowledge snarks from linear-size universal and updateable structured reference strings," Cryptology ePrint Archive, Paper 2019/099, 2019.