plookup: speeding up SNARKs on non-friendly functions with lookup tables

Ariel Gabizon Zachary J. Williamson

Aztec

SNARKs are easy prey in a world full of nasty binary functions

$\mathbf{a}, \mathbf{b}, \mathbf{c} \in \mathbb{F}$

Want to show $\mathbf{c} = \mathbf{a}\ \ \mathbf{xor}\ \ \mathbf{b}$ as 8-bit strings

SNARKs are easy prey in a world full of nasty binary functions

$\mathbf{a}, \mathbf{b}, \mathbf{c} \in \mathbb{F}$

Want to show $\mathbf{c} = \mathbf{a} \;\; \boldsymbol{xor} \;\; \mathbf{b}$ as 8-bit strings

Standard way requires 25-32 constraints: Give bit decomposition of $\mathbf{a}, \mathbf{b}, \mathbf{c}$, check bitwise xor.

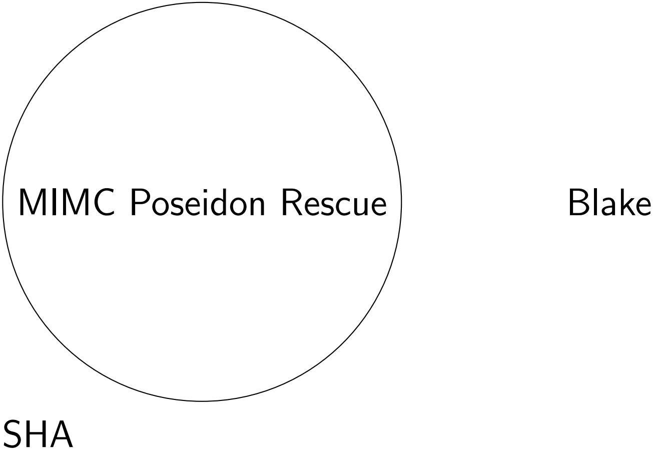
$\mathbf{a}, \mathbf{b}, \mathbf{c} \in \mathbb{F}$

Want to show $\mathbf{c} = \mathbf{a}\ \mathbf{xor}\ \mathbf{b}$ as 8-bit strings.

Standard way requires 25-32 constraints: Give bit decomposition of $\mathbf{a}, \mathbf{b}, \mathbf{c}$, check bitwise xor.

This is a *multiplicative* factor you pay on each small operation while computing SHA/BLAKE

Approach 1: Keep SNARKs in friendly neighborhoods



MIMC Poseidon Rescue

Blake

SHA

Our Approach: lookup tables (see also: Arya[Bootle, Cerulli, Groth, Jakobsen, Maller])

Precompute table $\mathbf{T}$ of all triplets $(\mathbf{a}, \mathbf{b}, \mathbf{c})$ s.t. $\mathbf{c} = \mathbf{a} \ \mathbf{xor} \ \mathbf{b}$.

Our Approach: lookup tables (see also: Arya[Bootle, Cerulli, Groth, Jakobsen, Maller])

Precompute table $\mathbf{T}$ of all triplets $(\mathbf{a}, \mathbf{b}, \mathbf{c})$ s.t. $\mathbf{c} = \mathbf{a} \ \mathbf{xor} \ \mathbf{b}$.

Instead of representing $\mathbf{xor}$ logic, check that $(\mathbf{a}, \mathbf{b}, \mathbf{c}) \in \mathbf{T}$

Our Approach: lookup tables (see also: Arya[Bootle, Cerulli, Groth, Jakobsen, Maller])

Precompute table $\mathbf{T}$ of all triplets $(\mathbf{a}, \mathbf{b}, \mathbf{c})$ s.t. $\mathbf{c} = \mathbf{a}\ \mathbf{xor}\ \mathbf{b}$.

Instead of representing $\mathbf{xor}$ logic, check that $(\mathbf{a}, \mathbf{b}, \mathbf{c}) \in \mathbf{T}$

After enough lookups, has amortized cost of ~ 1 constraint per $\mathbf{xor}$.

The plookup protocol in a nutshell

Witness $\mathbf{f} = \{\mathbf{f_i}\}_{\mathbf{i}\in[\mathbf{n}]}$ Table $\mathbf{t} = \{\mathbf{t_i}\}_{\mathbf{i}\in[\mathbf{d}]}$
Want to prove $\mathbf{f} \subset \mathbf{t}$. (using randomness we have reduced tuples to single elements).

First thing that comes to mind Some divisibility check between

$$F = \prod_{i \in [n]} (X - f_i), T = \prod_{i \in [d]} (X - t_i)$$