Square Span Programs with Applications to Succinct NIZK Arguments

George Danezis, University College London

Cédric Fournet, Microsoft Research

Jens Groth, University College London

Markulf Kohlweiss, Microsoft Research

Non-interactive zero-knowledge argument

Common reference string

Statement: $x \in L$

$(x, w) \in R_L$

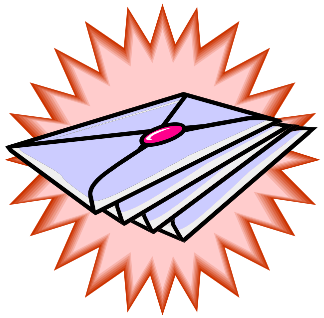
Proof: π $\longrightarrow$

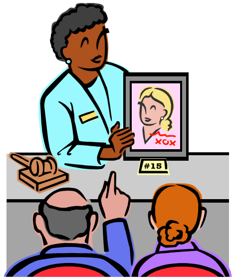
OK

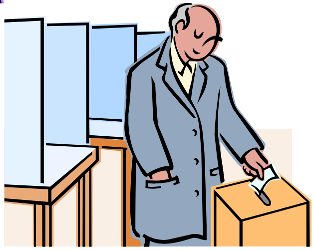
Zero-knowledge:
Nothing but truth revealed

Soundness:
Statement is true

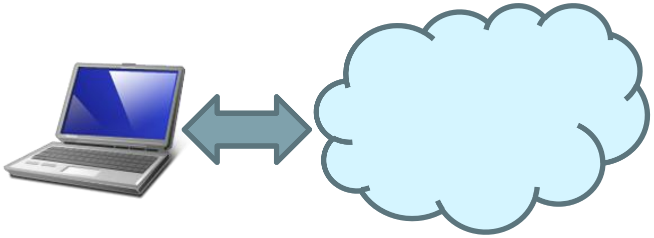
Applications







NIZK arguments guarantee honesty (soundness), yet also preserve privacy (zero-knowledge)

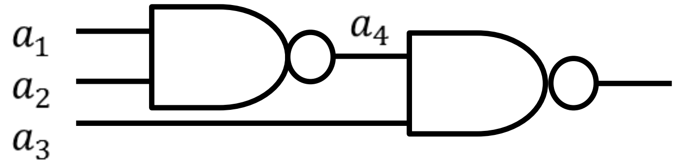


Our contribution

- NIZK argument
  - Perfect correctness
  - Perfect zero-knowledge
  - Computational soundness
    - Knowledge extractor assumptions using pairings
- Conceptually simple
  - New characterization of NP as Square Span Programs
- Small size
  - Four group element proofs

Technical path

- Relation R_L for NP-language L given by circuit C_R



- Characterize satisfiability of C_R as constraints

$$\vec{a}V \in \{0,2\}^d$$

- Rewrite constraints as square span program

$$t(x) \text{ divides } \left(\sum a_i v_i(x)\right)^2 - 1$$

Example

- Consider circuit with single XOR-gate $a_0 = a_1 \oplus a_2$
- Satisfiability corresponds to the constraints

$$a_1, a_2 \in \{0,1\} \qquad 1 + a_1 + a_2 \in \{0,2\}$$

- Which we can write

$$\vec{a}V = (1, a_1, a_2)\begin{pmatrix} 0 & 0 & 1 \\ 2 & 0 & 1 \\ 0 & 2 & 1 \end{pmatrix} \in \{0,2\}^3$$

- This is satisfied if and only if for all columns V_j

$$(\vec{a}V_j - 1)^2 = 1$$

Example continued

- Define

$$t(x) = (x+1)(x-1)x \qquad v_0(x) = -x^2$$
$$v_1(x) = 1 - x \qquad v_2(x) = 1 + x$$

- Chosen such that

$$\begin{pmatrix} v_0(-1) & v_0(1) & v_0(0) \\ v_1(-1) & v_1(1) & v_1(0) \\ v_2(-1) & v_2(1) & v_2(0) \end{pmatrix} = \begin{pmatrix} -1 & -1 & 0 \\ 2 & 0 & 1 \\ 0 & 2 & 1 \end{pmatrix}$$

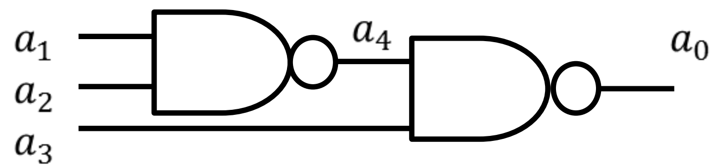
- Then $\left(\vec{a}V_j - 1\right)^2 = 1$ for all j if and only if

$$t(x) \text{ divides } \left(v_0(x) + a_1 v_1(x) + a_2 v_2(x)\right)^2 - 1$$

- Using techniques presented later we can use the

Characterizing circuit satisfiability as a set of integer constraints

- Consider a circuit with wires $a_0, \ldots, a_m$



- The circuit is satisfiable if all wires $a_i \in \{0,1\}$, and they respect all the gates, and the output is 1
- Write the wires as a vector $\vec{a} = (a_0, a_1, \ldots, a_m)$
  The condition $a_i \in \{0,1\}$ can be written $\vec{a}I \in \{0,1\}^{m+1}$
  - Could omit trivial checks, e.g., we know $a_0 = 1$ for satisfied circuit and we may know some inputs $a_1, \ldots, a_\ell \in \{0,1\}$

Linearization of the gates

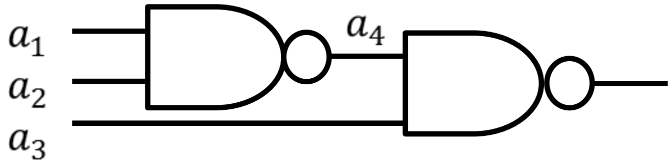
- Fan-in 2 gates can be linearized
  - For the XOR-gate we have for $a, b, c \in \{0,1\}$
    $a \oplus b = c$ if and only if $a + b + c \in \{0,2\}$
  - For the NAND-gate we have for $a, b, c \in \{0,1\}$
    $\neg(a \wedge b) = c$ if and only if $a + b + 2c - 2 \in \{0,1\}$
- In our paper we show that by multiplying some equations by 2, we can write all non-trivial fan-in 2 gates on the form $\alpha a + \beta b + \gamma c + \delta \in \{0,2\}$

Characterizing circuit satisfiability as constraints on an affine map

- We write the gate equations in a matrix G such that the wires respect the n gates if and only if
$$\vec{a}G \in \{0,2\}^n$$
- Combined with the constraint $\vec{a}I \in \{0,1\}^{m+1}$ we get the circuit is satisfiable if and only if
$$\vec{a}V = \vec{a}(2I|G) \in \{0,2\}^{m+n+1}$$
- The constants are small, so the equivalence with circuit satisfiability also holds over $\mathbf{Z}_p$ for $p > 8$

Technical path

- Relation R_L for NP-language L given by circuit C_R



- Characterize satisfiability of C_R as constraints

$$\vec{a}V \in \{0,2\}^d$$

- Rewrite constraints as square span program

$$t(x) \;\; \text{divides} \;\; \left(\sum a_i v_i(x)\right)^2 - 1$$

Square span program

- A squ ls
  $v_0(x),$ $x)) = d$
- We sa pts an
  input s
  $a_{\ell+1}, \ldots, m$

Quadratic span programs

Two sequences of polynomials:

$$t(x) \text{ divides } \sum a_i v_i(x) \cdot \sum b_i w_i(x)$$

Quadratic arithmetic programs

Three sequences of polynomials

$$t(x) \text{ divides } \sum a_i v_i(x) \cdot \sum b_i w_i(x) - \sum c_i y_i(x)$$

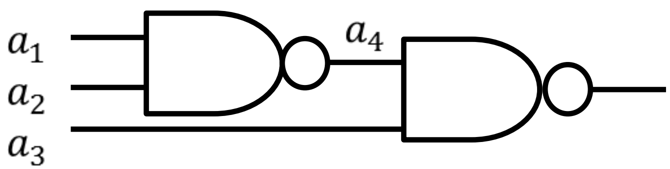
$$t(x) \quad \text{divides} \quad \left(\sum_{i=0}^{m} a_i v_i(x)\right)^2 - 1$$

using $a_0 = 1$

- We say a square span program accepts a language if it accepts the elements in the language

From affine map constraints to SSPs

- Earlier



$\Rightarrow$ $\vec{a}V \in \{0,2\}^d$ over $\boldsymbol{Z}_p$

- Pick distinct points $r_1, \ldots, r_d \in \boldsymbol{Z}_p$
- Define $t(x) = \prod(x - r_j)$
- Define $v_0'(x), v_1(x), \ldots, v_m(x)$ such that

$$\begin{pmatrix} {v_0}'(r_1) & \ldots & {v_0}'(r_d) \\ \vdots & \ddots & \vdots \\ v_m(r_1) & \ldots & v_m(r_d) \end{pmatrix} = V$$

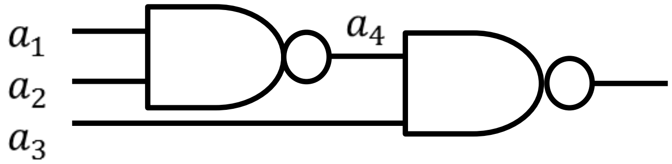
and $v_0(x) = v_0'(x) - 1$

Why does the square span program work?

- We want for all columns j that $\vec{a}V_j \in \{0,2\}$ or equivalently that $\vec{a}V_j - 1 \in \{-1,1\}$ (over $\boldsymbol{Z}_p$)
- By the choice of polynomials $v_0(x), \ldots, v_m(x)$ we have $\sum a_i v_i(r_j) = \vec{a}V_j - a_0 1 = \vec{a}V_j - 1$
- This gives us the condition $\left(\sum a_i v_i(r_j)\right)^2 = 1$
- So $\vec{a}V \in \{0,2\}^d$ if and only if all r_j are roots in the polynomial $\left(\sum a_i v_i(x)\right)^2 - 1$
- I.e., $t(x) = \prod(x - r_j)$ divides $\left(\sum a_i v_i(x)\right)^2 - 1$

Technical path

- Relation R_L for NP-language L given by circuit C_R



- Characterize satisfiability of C_R as constraints

$$\vec{a}V \in \{0,2\}^d$$

- Rewrite constraints as square span program

$$t(x) \text{ divides } \left(\sum a_i v_i(x)\right)^2 - 1$$

Prime order bilinear groups

- $\mathrm{Gen}(1^k)$ generates $(p, \mathbb{G}, \hat{\mathbb{G}}, \mathbb{G}_T, e)$
- $\mathbb{G}, \hat{\mathbb{G}}, \mathbb{G}_T$ finite cyclic groups of prime order p
- Pairing $e: \mathbb{G} \times \hat{\mathbb{G}} \to \mathbb{G}_T$
  - $e(U^a, \hat{V}^b) = e(U, \hat{V})^{ab}$
  - $\mathbb{G} = \langle G \rangle,\ \hat{\mathbb{G}} = \langle H \rangle,\ \mathbb{G}_T = \langle e(G, \hat{G}) \rangle$
- Deciding group membership, group operations, and bilinear pairing efficiently computable

- Statement: $(a_1, \dots, a_\ell)$ accepted by SSP
- Common reference string: $\beta, s \leftarrow \boldsymbol{Z}_p$

  $(G, \hat{G}, \dots, G^{s^d}, \hat{G}^{s^d}, G^{\beta v_{\ell+1}(s)}, \dots, G^{\beta v_m(s)}, G^{\beta t(s)}, \tilde{G}, \tilde{G}^{\beta})$
- Argument: Pick $\delta \leftarrow \boldsymbol{Z}_p$ and compute $h(x)$ such that

$$h(x)t(x) = \left(\sum a_i v_{i(x)} + \delta t(x)\right)^2 - 1$$

  Compute $\pi = (H, V_w, B_w, \hat{V})$ as

$$B_w = G^{\beta(\sum_{i>\ell} a_i v_i(s) + \delta t(s))} \qquad H = G^{h(s)}$$

$$V_w = G^{\sum_{i>\ell} a_i v_i(s) + \delta t(s)} \qquad \hat{V} = \hat{G}^{\sum a_i v_i(s) + \delta t(s)}$$

- Verification: Compute $V = G^{\sum_{i\le\ell} a_i v_i(s)} V_w$ and check

$$e(V, \hat{G}) = e(G, \hat{V}) \qquad e(V_w, \tilde{G}^{\beta}) = e(B_w, \tilde{G})$$

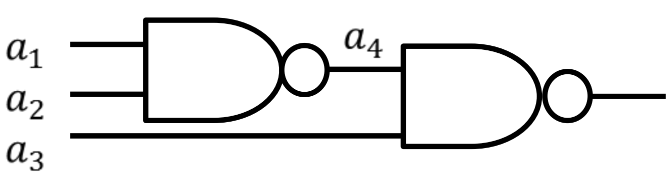
Succinct NIZK argument

- Perfect correctness
- Perfect zero-knowledge
- Computational soundness
  - Assuming d-PKE, d-PDH, and
- Succinct
  - 4 group elements ≈ 160 bytes
- Efficient
  - Prover: $O(d \log d)$ multiplications and 3 exponentiations
  - Verifier: ℓ exponentiations and 6 pairings ≈ 6ms

Pinocchio
Argument: 8 elements
Computation: Better when statements involve additions or multiplications instead of fan-in 2 gates since it is based on quadratic arithmetic programs

Summary

- Introduced square span programs
  - Conceptually simple type of quadratic span programs
- Showed they are NP-complete



$$\vec{a}V \in \{0,2\}^d$$

$$t(x) \text{ divides } \left(\sum a_i v_i(x)\right)^2 - 1$$

- Succinct NIZK argument