# *Progression-Free Sets and Sublinear Pairing-Based Non-Interactive Zero-Knowledge Arguments*

Helger Lipmaa

Cybernetica AS

Tallinn University

Estonian Theory Days, Nelijärve 2001





## Outline I

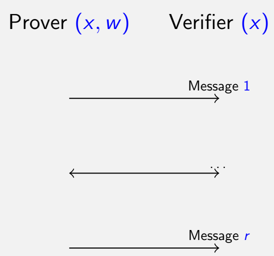






## Zero-Knowledge Arguments

- **Inputs:**
  - NP-language $L$ and a relation $R_L$ such that $\forall x$: $x \in L$ iff $\exists w$ such that $(x, w) \in R_L$
  - Common input $x$, Prover has private input $w$
- Prover wants to convince Verifier that $x \in L$ without revealing anything else
- **Efficiency requirements:** non-interactivity, small computation/communication?

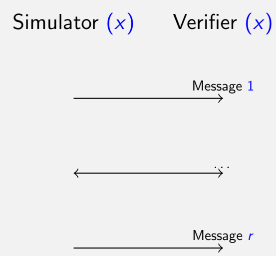






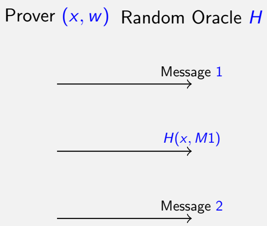

## Zero-Knowledge Arguments

- **Perfect Completeness:** If $(x, w) \in R_L$ then Verifier outputs 1
- **Computational Soundness:** If $x \notin L$ then for any PPT adversary Prover, the probability that Verifier outputs 1 is negligible
- **Perfect Zero-Knowledge:** Exists a simulator $S$ that can *perfectly* simulate the transcript between Prover and Verifier *without knowing* $w$



## Non-Interactive Zero-Knowledge

- Usually, ZK arguments are multi-round
- Inconvenient in applications: it would be good to create the argument once, and then let many different verifiers to verify it independently
- Well-known: no NIZK in plain model
- **Fiat-Shamir heuristic:** substitute the verifier's messages with the output of random oracle. Result is NIZK
  - Good: often very efficient
  - Bad: random oracles do not exist

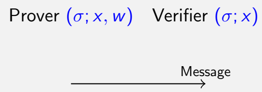







## NIZK in Common Reference String Model

- CRS model — a weaker setup assumption
- All parties are given a trusted CRS that is generated according to some nice probability distribution
- The simulator generates CRS together with a trapdoor that is only used in the proof








## Our Results: Quick Overview

- NIZK argument in the CRS model for circuit satisfiability

| CRS | Comm | P.comp | V.comp |
|---|---|---|---|
| [Groth 2010] | | | |
| $O(\lvert C\rvert^2)$ | 42 | $O(\lvert C\rvert^2)E + \Theta(\lvert C\rvert^2)M$ | $\Theta(\lvert C\rvert)$ |
| $O(\lvert C\rvert^{2/3+\varepsilon})$ | $\Theta(\lvert C\rvert^{2/3})$ | $O(\lvert C\rvert^{4/3})E + \Theta(\lvert C\rvert^{4/3})M$ | $\Theta(\lvert C\rvert)$ |
| This paper | | | |
| $O(\lvert C\rvert^{1+\varepsilon})$ | 32 | $O(\lvert C\rvert^{1+\varepsilon})E + \Theta(\lvert C\rvert^2)M$ | $\Theta(1)$ |
| $O(\lvert C\rvert^{1/2+\varepsilon})$ | $\Theta(\lvert C\rvert^{1/2})$ | $O(\lvert C\rvert^{1+\varepsilon})E + \Theta(\lvert C\rvert^{3/2})M$ | $\Theta(\lvert C\rvert^{1/2})$ |

- Zap (2-message witness-indistinguishable public-coin argument): verifier sends CRS, prover sends argument
  - Communication: $O(|C|^{1/2+\varepsilon})$ group elements
- Also: weaker security assumption
  - $q$-power (symmetric) DL instead of $q$-power CDH





## Basic Idea of SAT Argument

- Assume the circuit has only NAND gates
- Circuit size is $n$, thus $2n + 1$ wires $a_i$
- Prover multicommits to $2n + 1$ wires by one group element
- He proves the wires are consistent and that the last wire is equal to 1, by using a few "parallel" operations
  - All wires are Boolean: $a_i = a_i \cdot a_i$ for all $i$
  - Output wires of same gate have same value: define suitable permutation $\xi$ on all wires, show that $a_i = a_{\xi(i)}$ for all $i$
  - The NAND gates are respected
  - ...
- In total 7 permutation and product arguments
- Efficiency and security inherited from basic arguments

# Basic Idea: Prod/Perm Arguments

- Select random $x, \alpha, \beta$, let $\Lambda = (\lambda_1, \dots, \lambda_n)$
- $com^t(\sigma; \vec{a}; r) := (g_t^{f_1(x)}, g_t^{\alpha f_1(x)}, g_t^{\beta f_1(x)})$ for $f_1(x) = r + \sum a_i x^{\lambda_i}$.
- $\log \left( e(g_1^{f_1(x)}, g_2^{f_2(x)}) / e(g_1^{f_3(x)}, g_2^{f_4(x)}) \right)$

  $= f_1(x) f_2(x) - f_3(x) f_4(x) = \sum_{i \in \Lambda_1} \delta_i x^i + \sum_{i \in \Lambda_2} \gamma_i x^i$
- $f_3/f_4$ are chosen so that if the prover is honest, then $\delta_i = 0$
- $\Lambda_1 = \Lambda_1(\Lambda)$ and $\Lambda_2 = \Lambda_2(\Lambda)$ are such that $\Lambda_1 \cap \Lambda_2 = \emptyset$
  - $\Lambda$ is "progression-free" set of odd integers, $\lambda_n = O(n^{1+\varepsilon})$
- $(g_2^{x^i}, g_2^{\alpha x^i})$ belongs to CRS $\sigma$ iff $i \in \Lambda_2$ — $|\sigma| = O(n^{1+\varepsilon})$
- Security assumption: if $A(\sigma)$ can output $(X, \hat{X})$ such that $X_2 = X_1^{\alpha}$, then $A$ "knows" a representation $\log X_1 = \sum_{i \in \Lambda_2} \gamma_i x^i$





# Knowledge Commitment Scheme

- Let $par = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e) \leftarrow GBP(1^\kappa)$, and let $g_j$ be a generator of $\mathbb{G}_j$. Let $x, \alpha, \beta \leftarrow \mathbb{Z}_p$
- Fix subset $\Lambda = (\lambda_1, \dots, \lambda_n) \subseteq [q]$ with $0 < \lambda_i < \lambda_{i+1}$
- Prover commits to $\vec{a} = (a_1, \dots, a_n) \in \mathbb{Z}_p^n$, $n \leq q$ in $\mathbb{G}_t$
- The CRS is $\sigma = (par; (g_t^{x^i}, g_t^{\alpha x^i}, g_t^{\beta x^i})_{i \in \{0, \dots, q\}})$
- For $t \in \{1, 2\}$ and random $r \leftarrow \mathbb{Z}_p$,

$$com^t(\sigma, \vec{a}; r) = (g_t^{f(x)}, g_t^{\alpha f(x)}, g_t^{\beta f(x)}) \in \mathbb{G}_t^3$$

  for $f(x) = r + \sum_{i=1}^{n} a_i x^{\lambda_i}$.
- By security assumption, Prover knows $(\vec{a}, r)$





# Progression-Free Sets

- $\Lambda \in [n]$ is progression-free if it does not contain arithmetic progression of length 3
- That is: for $\lambda_i, \lambda_j, \lambda_k \in \Lambda$, $\lambda_k - \lambda_j = \lambda_j - \lambda_i$ iff $i = j = k$
- Let $r_3(n)$ be the cardinality of the largest progression-free subset of $[n]$
- [Elkin 2010]:

$$r_3(n) = \Omega\left( \frac{n \cdot (\log_2^{1/4} n)^{1/4}}{2^{2\sqrt{2 \log_2 n}}} \right) = \Omega(n^{1-\varepsilon})$$

  for any $\varepsilon > 0$
- [Sanders 2010]: $r_3(n) = O(n / \log^{1-o(1)} n)$

000 001 002 010 011 012 020 021 022 100 101 102 110 111 112 120 121 122 200 201 202 210 211 212 220 221 222





# Hadamard Product Argument

- Prover wants to convince Verifier that for given commitments $A \in \mathbb{G}_1, B \in \mathbb{G}_2, C \in \mathbb{G}_1$, she knows how to open them as $\vec{a}, \vec{b}, \vec{c}$, such that $c_j = a_j \cdot b_j$ for every $j \in [n]$

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $a_0$ | $a_1$ | $a_2$ | $a_3$ | $a_4$ | $a_5$ | $a_6$ | $a_7$ | $a_8$ | $a_9$ | $a_{10}$ | $a_{11}$ | $a_{12}$ | $a_{13}$ | $a_{14}$ | $a_{15}$ |
| $\cdot$ | $\cdot$ | $\cdot$ | $\cdot$ | $\cdot$ | $\cdot$ | $\cdot$ | $\cdot$ | $\cdot$ | $\cdot$ | $\cdot$ | $\cdot$ | $\cdot$ | $\cdot$ | $\cdot$ | $\cdot$ |
| $b_0$ | $b_1$ | $b_2$ | $b_3$ | $b_4$ | $b_5$ | $b_6$ | $b_7$ | $b_8$ | $b_9$ | $b_{10}$ | $b_{11}$ | $b_{12}$ | $b_{13}$ | $b_{14}$ | $b_{15}$ |
| = | = | = | = | = | = | = | = | = | = | = | = | = | = | = | = |
| $c_0$ | $c_1$ | $c_2$ | $c_3$ | $c_4$ | $c_5$ | $c_6$ | $c_7$ | $c_8$ | $c_9$ | $c_{10}$ | $c_{11}$ | $c_{12}$ | $c_{13}$ | $c_{14}$ | $c_{15}$ |

- Goal: to do verification in parallel

## Hadamard Product Argument: Idea

- Let $X_1 \leftarrow e(A, B)$, $X_2 \leftarrow e(C, \prod_{j=1}^{n} g_2^{x^{\lambda_j}})$, $h \leftarrow e(g_1, g_2)$
- $A = g_1^{r_1 + \sum_{j=1}^{n} a_j x^{\lambda_j}}$, thus $\log A = r_1 + \sum_{j=1}^{n} a_j x^{\lambda_j}$
- For fixed $\Lambda$, let $\Lambda_2 := \{0\} \cup \{\lambda_i\} \cup \{\lambda_i + \lambda_j\}_{i \neq j}$
- For some integers $\gamma_i$,

$$\log(X_1/X_2) = (r_1 + \sum_i a_i x^{\lambda_i}) \cdot (r_2 + \sum_i b_i x^{\lambda_i}) - (r_3 + \sum_i c_i x^{\lambda_i})(\sum_i x^{\lambda_i}) = \sum_{i=1}^{n} (a_i b_i - c_i) x^{2\lambda_i} + \sum_{i \in \Lambda_2} \gamma_i x^i$$

- If prover is honest then this is $0$:





## Hadamard Product Argument: Idea

- $\Lambda_2 := \{0\} \cup \{\lambda_i\} \cup \{\lambda_i + \lambda_j\}_{i \neq j}$
- For some integers $\gamma_i$,

$$\log(X_1/X_2) = (r_1 + \sum_i a_i x^{\lambda_i}) \cdot (r_2 + \sum_i b_i x^{\lambda_i}) - (r_3 + \sum_i c_i x^{\lambda_i})(\sum_i x^{\lambda_i}) = \sum_i (a_i b_i - c_i) x^{2\lambda_i} + \sum_{i \in \Lambda_2} \gamma_i x^i$$

- If $\Lambda$ is progression-free set of odd integers, then $2\lambda_i \notin \Lambda_2$
- Thus: $c_i = a_i b_i$ for all $i \in [n]$ iff $\log(X_1/X_2)$ can be represented as $\sum_{i \in \Lambda_2} \gamma_i x^i$
  - The iff part follows from security assumptions

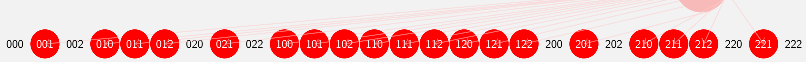




## Hadamard Product: CRS Generation

- Let $par = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e) \leftarrow GBP(1^\kappa)$
- Set $x, \alpha \leftarrow \mathbb{Z}_p$, and let $g_t$ be a generator of $\mathbb{G}_t$ for $t \in \{1, 2\}$
- Define CRS as

$$\sigma = (par; (g_1^{\alpha x^i}, g_1^{\beta x^i}, g_1^{\gamma x^i}, g_2^{\beta x^i})_{i \in \{0\} \cup \Lambda}, (g_2^{x^i}, g_2^{\alpha x^i})_{i \in \Lambda_2})$$




- Due to Elkin, $|\sigma|, |\Lambda_2| = O(n^{1+\varepsilon})$ for any $\varepsilon > 0$





## Hadamard Product: Argument

- Recall $\sigma = \left(par; \{\{g_t^{x^i}\}_{0 \leq i \leq 2\lambda_n}, \{g_t^{\alpha x^i}\}_{i \in \Lambda_2}\}_{t \in \{1,2\}}\right)$
- Let $A = com^1(\sigma; \vec{a}; r_1)$, $B = com^2(\sigma; \vec{b}; r_2)$, $C = com^1(\sigma; \vec{c}; r_3)$.
- Prover sets $\pi_1 \leftarrow \prod_{i \in \Lambda_2} \left(g_2^{x^i}\right)^{\gamma_i}$, $\pi_2 \leftarrow \prod_{i \in \Lambda_2} \left(g_2^{\alpha x^i}\right)^{\gamma_i}$
- Argument: $(\pi_1, \pi_2) \in \mathbb{G}_2^2$
- All $\gamma_i$ can be computed by doing $\Theta(n^2)$ multiplications in $\mathbb{Z}_p$
- Two $O(n^{1+\varepsilon})$-multi-exponentiations, $\Theta(n^2)$ multiplications in $\mathbb{Z}_p$

## Hadamard Product: Verification

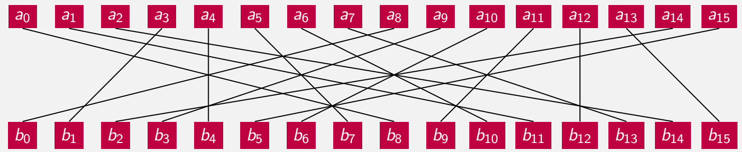
- Include $D \leftarrow \prod_{j=1}^{n} g_2^{x^{\lambda_j}}$ in CRS
- Verifier checks that
  - $e(A, B)/e(C, D) = e(g_1, \pi_1)$
  - $e(g_1^{\alpha}, \pi_1) = e(g_1, \pi_2)$
- $5$ pairings





## Permutation Argument

- Prover has committed to $\vec{a}, \vec{b}$ and wants to convince Verifier that for a public permutation $\varrho$, $a_{\varrho(j)} = b_j$.




- Similar idea: construct a formal polynomial $f(x)$, such that Prover is honest iff for a fixed set $\Lambda_2'$, $\exists \vec{\delta} : f(x) = \sum_{j \in \Lambda_2'} \delta_j x^j$.
- $\Lambda_2'$ is constructed so that from the progression-freeness of $\Lambda$ and security assumptions it follows that the whole permutation argument is secure
- Complexity: almost the same as for product argument





## Argument for Circuit Satisfiability

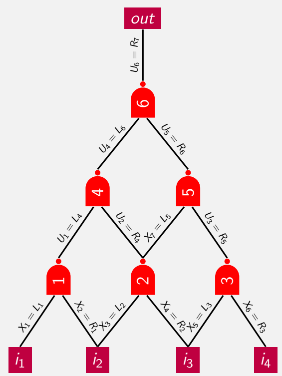
- Prover and Verifier share a circuit $C$. Prover wants to convince Verifier he knows a satisfying assignment
- Binary circuit, only NAND gates, $a \overline{\wedge} b = \neg(a \wedge b)$
- We describe the circuit by using its number of gates, and two permutations that show that the circuit is self-consistent





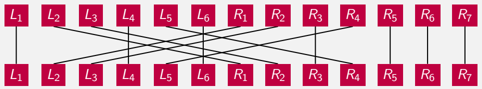
## Circuit Description

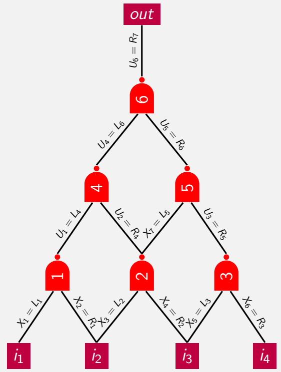
- Circuit has $n$ gates, every gate $i$ has inputs $L_i$ and $R_i$, and output $U_i$. $U_n$ is the output of the circuit
- There are $2n + 1$ wires. Every wire, except one we done by $R_{n+1}$, is equal to $L_i$ or $R_i$ for $i \in [n]$
- Every gate has at least one output wire $U_i$. There are $n + 1$ more wires $X_i$ that correspond to inputs to the circuit, and multiple outputs
- Denote $A = (L_1, \ldots, L_n, R_1, \ldots, R_n, R_{n+1})$, $B = (U_1, \ldots, U_n, X_1, \ldots, X_{n+1})$



## Circuit Consistency

- Circuit consistency will be given by two permutations $\xi$ and $\tau$
- Input consistency permutation $\xi : [2n+1] \to [2n+1]$
  - For every $(A_{i_1}, \dots, A_{i_t})$ that have to be equal, $\xi$ permutes $A_{i_1} \to \dots \to A_{i_t} \to A_{i_1}$
  - For other input nodes $t$, $\xi(t) = t$
  - Clearly, circuit is inconsistent if for some $j$, $A_{\xi(j)} \neq A_j$



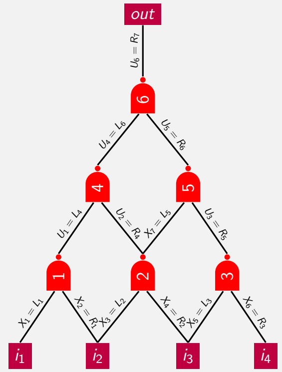








## Circuit Consistency

- Circuit consistency will be given by two permutations $\xi$ and $\tau$
- Throughput consistency permutation $\tau : [2n+1] \to [2n+1]$
  - Every wire is both an input wire (is equal to some $A_i$) and an output wirte (is equal to some $B_j$)
  - Define $\tau(i) = j$
  - Clearly circuit is inconsistent if for some $j$, $A_{\tau^{-1}(j)} \neq B_j$








## Full Argument: Idea

- Commit to $A$, $A' = (R_1, \dots, R_n, L_1, \dots, L_n, R_{n+1})$, $A'' = (R_1, \dots, R_n, 0, \dots, 0, R_{n+1}$, $B$ and $B' = (U_1, \dots, U_n, 0, \dots, 0)$
- Check all values are Boolean: $A \circ A = A$
- Check $A$ and $A'$ are consistent (permutation argument)
- Check $A'$ and $A''$ are consistent (product argument)
- Check $B$ and $B'$ are consistent (product argument)
- Check that NANDs are observed and $U_n = 1$: $A'' \circ A = (1_1, \dots, 1_{n-1}, 2_n, 1_{n+1}, \dots, 1_{2n+1}) - B'$
- Check that $\xi$ is observed (permutation argument with $A, A$)
- Check that $\tau$ is observed (permutation argument with $A, B$)

Done!





## Questions?