PKC 2025

Lattice-based Zero-knowledge Proofs for Blockchain Confidential Transactions

Shang GAO, Tianyu ZHENG, Yu GUO, Zhe PENG, Bin XIAO

2025/5/15

Bitcoin Transactions

- Public ledger for verification

Alice

Accounts:
$A_1 = 3$
$A_2 = 5$

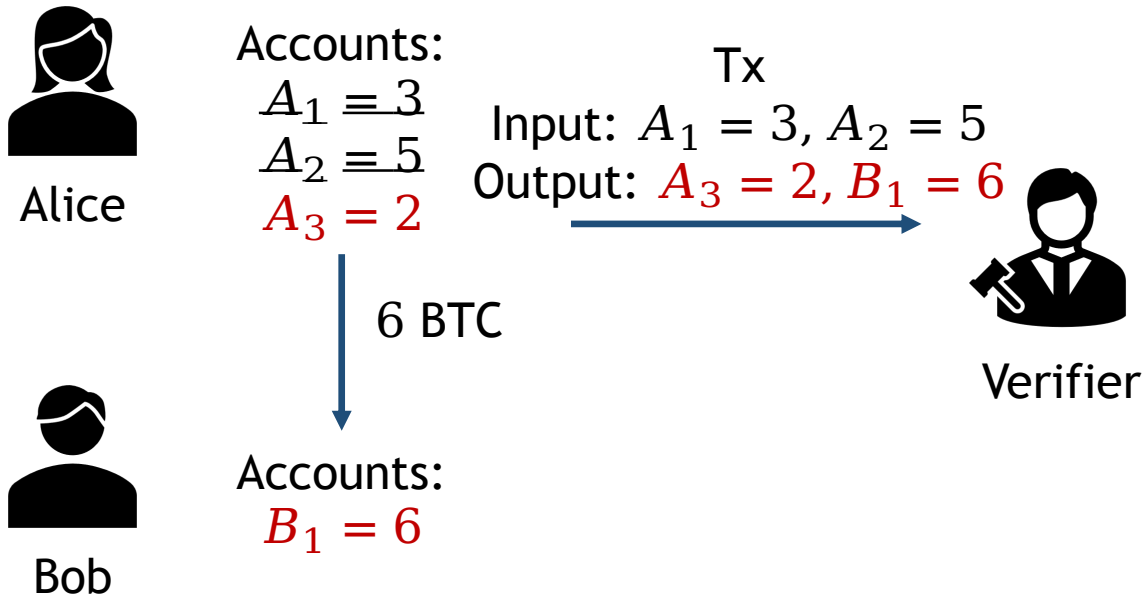
6 BTC

Bob

Verifier

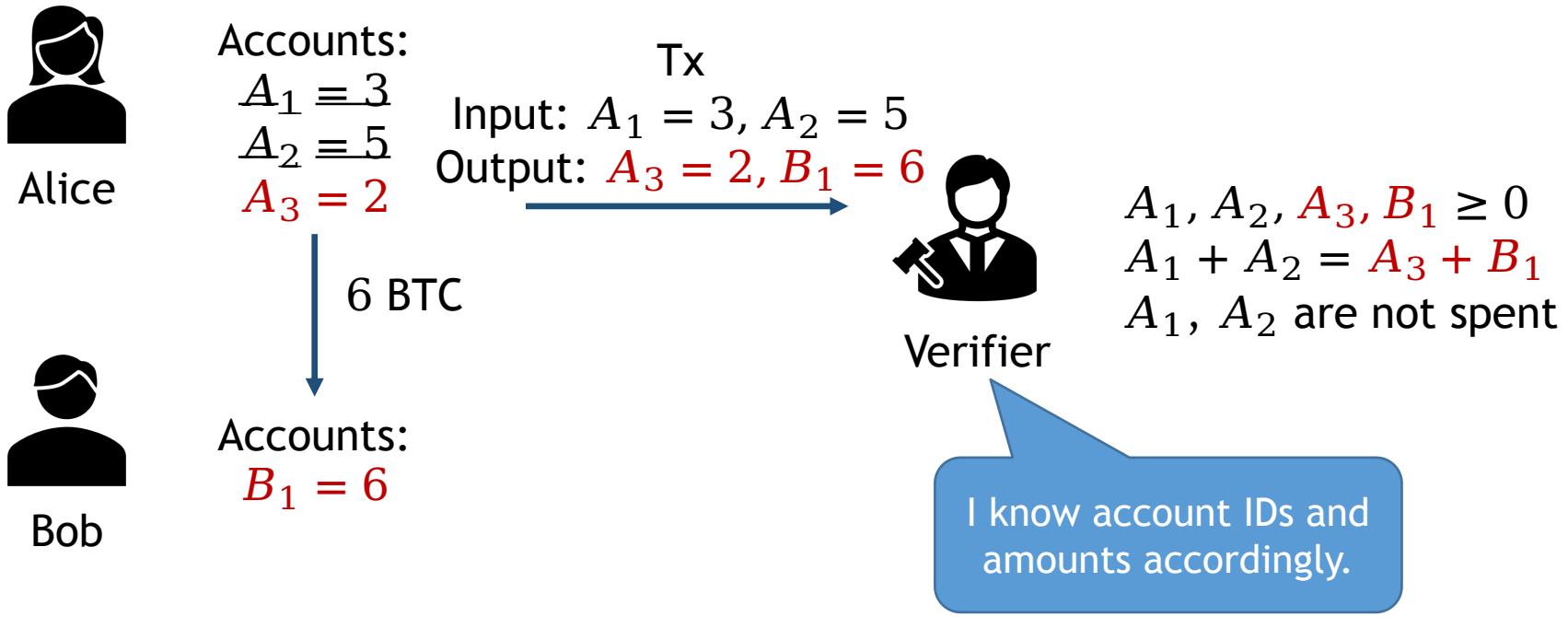
Bitcoin Transactions

- Public ledger for verification



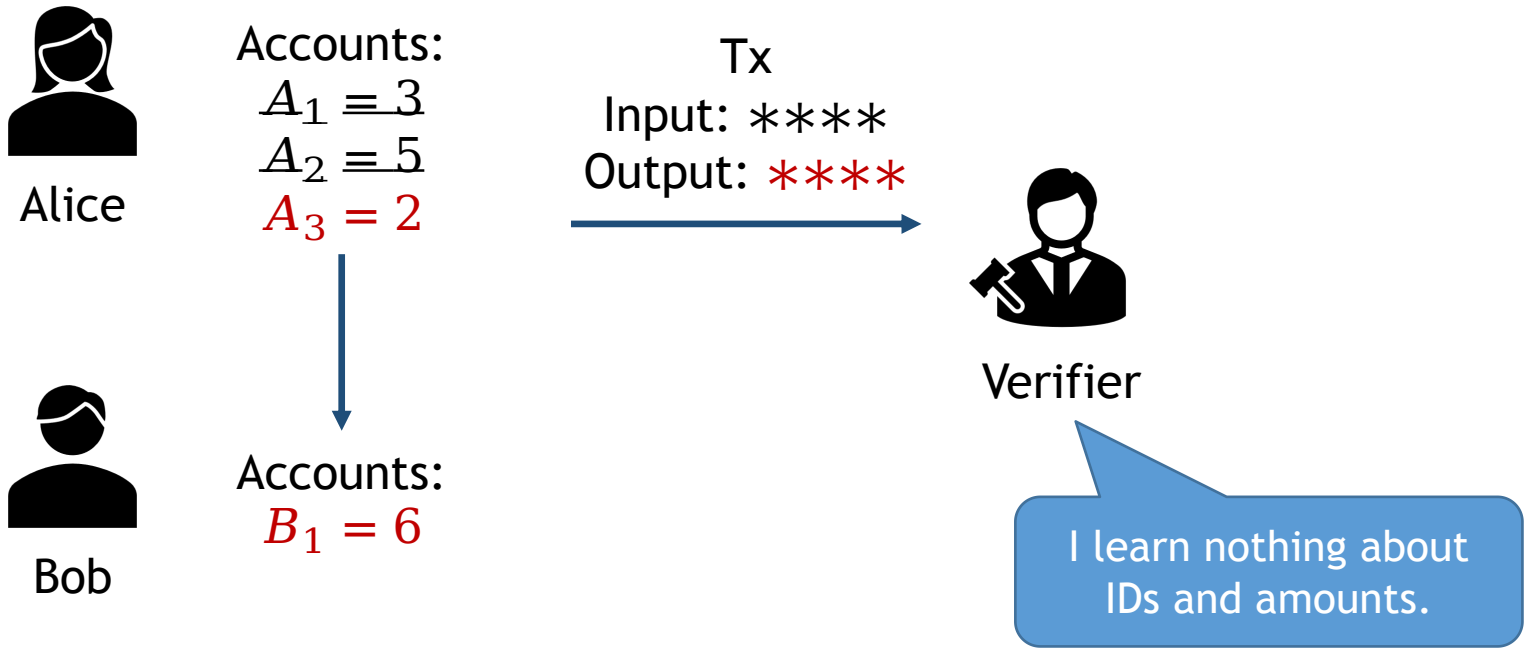
Bitcoin Transactions

- Public ledger for verification



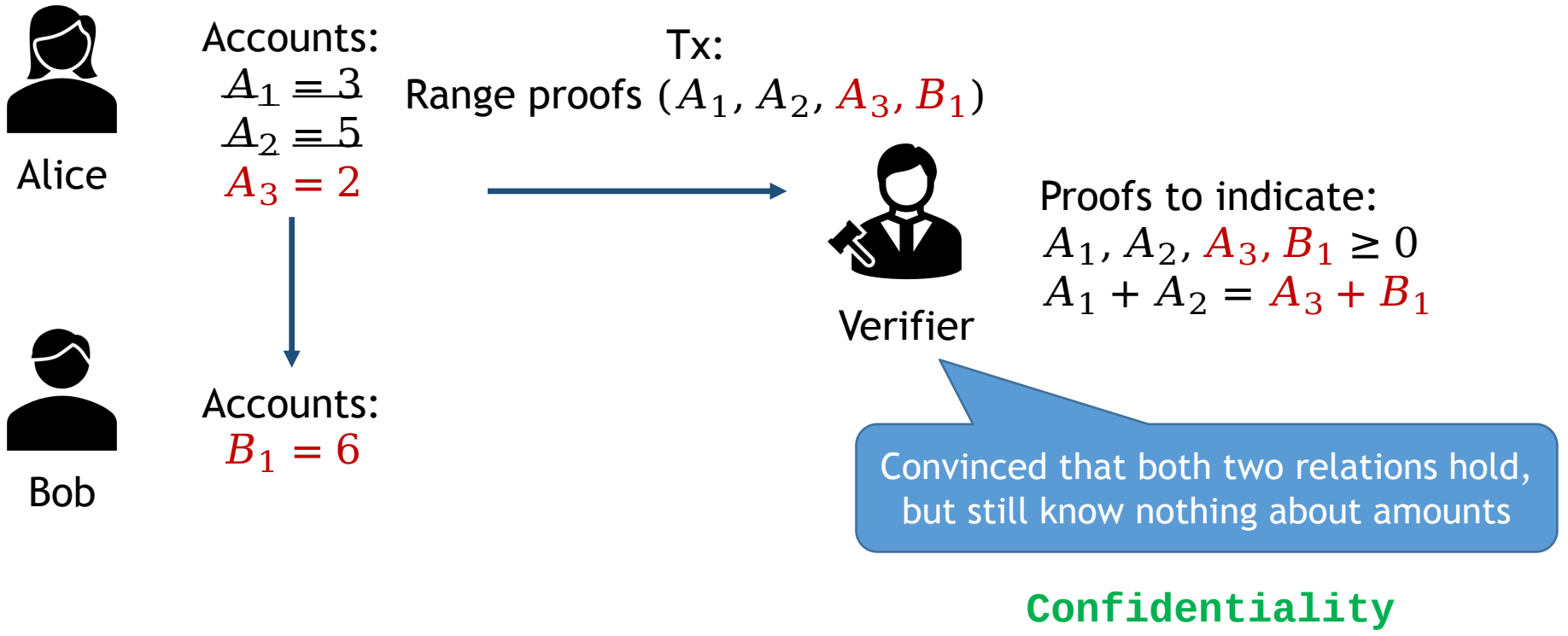
Anonymous Cryptocurrency

- ID and amounts should be private



Confidential Transactions

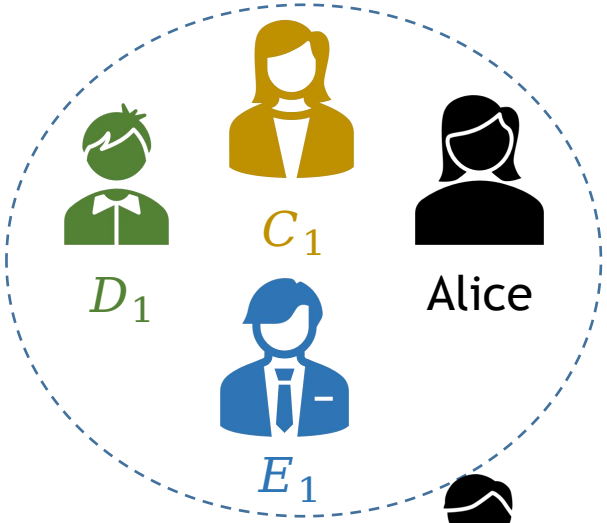
- Range proofs to hide the amount and allow verification at the same time



Confidentiality

Ring Confidential Transactions

- *Ring signature* to hide the identity.



Accounts:
$A_1 = 3$
$A_2 = 5$
$A_3 = 2$

Accounts:
$B_1 = 6$

Tx:
Ring signatures
$(A_1, A_2, C_1, D_1, E_1; A_3, B_1)$

Verifier

The “real” inputs are
in $(A_1, A_2, C_1, D_1, E_1)$ and not spent

Anonymity

Problems and Challenges

- Most of existing anonymous cryptocurrencies are not quantum secure.
  - Factoring and discrete logarithm are *easy* (polynomial time) on a quantum computer.
  - Both Ethereum community and Zcash team have put quantum-secure upgrades on their schedules.
- Additional requirements under lattice-based (post-quantum) solutions.
  - Messages must be *short* under short integer solution (SIS) problems.
  - Must convert a large m into a small one before using hashed-message commitment (HMC).

Problems and Challenges

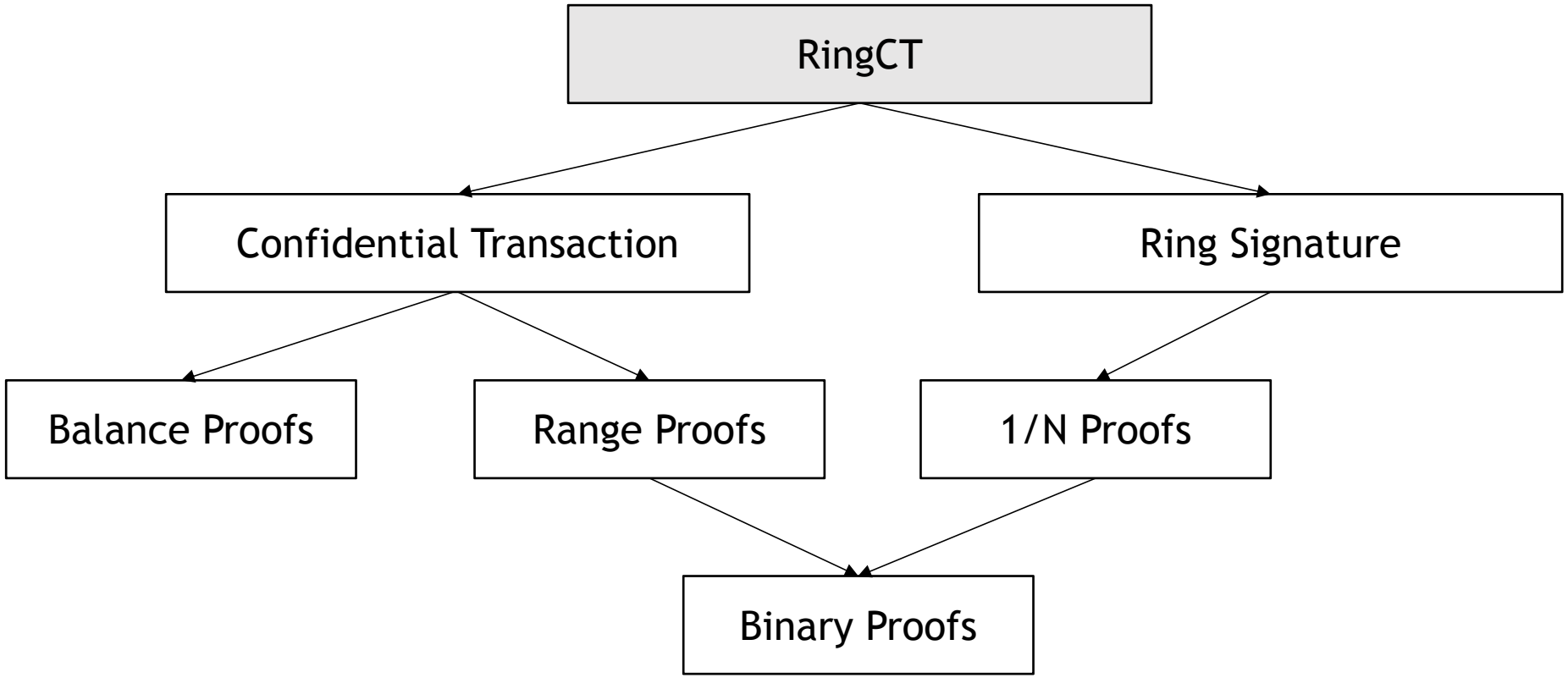
- Some lattice-based solutions are proposed, but not efficient in RingCT protocols [EZS+19, ESZ20].
  - Proof size is about 5~40x larger than traditional solutions.
  - Proving/verification time is about 2-4x slower.

  **Some techniques are not well-designed in lattice settings!*

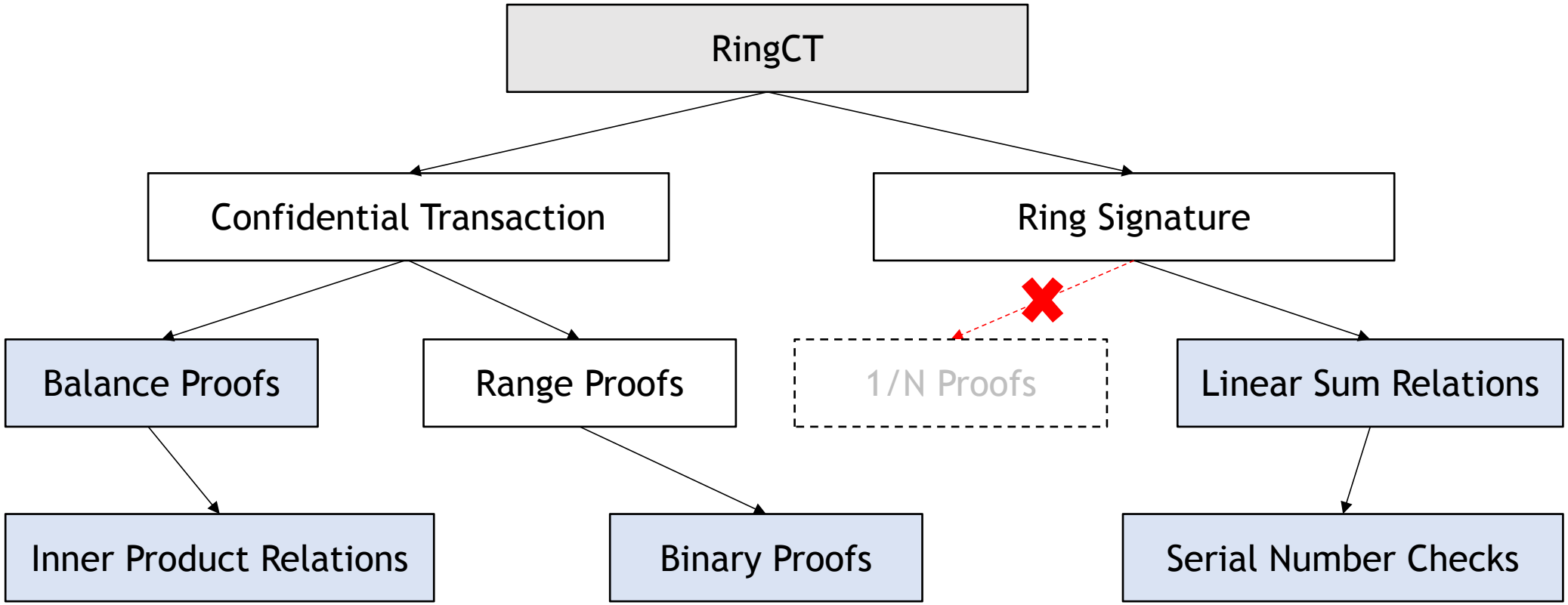
Objective

- To propose efficient and post-quantum RingCT protocols for anonymous cryptocurrencies.
  - New balance proofs under HMC.
  - New relations for linkable ring signatures.
  - New post-quantum RingCT protocol.
- Beneficiaries
  - Anonymous cryptocurrencies.
  - Privacy-preserving applications (e.g., anonymous e-voting).
  - Zk-Rollups.

Overview


RingCT
Confidential Transaction
Ring Signature
Balance Proofs
Range Proofs
1/N Proofs
Binary Proofs

Overview


RingCT
Confidential Transaction
Ring Signature
Balance Proofs
Range Proofs
1/N Proofs
Linear Sum Relations
Inner Product Relations
Binary Proofs
Serial Number Checks

Binary Proof

Batch Verification for Binary Proofs

- Binary Relations: $\boldsymbol{b}$ is binary -> $\boldsymbol{b} \circ (\boldsymbol{b} - \mathbf{1}) = \mathbf{0}$
- Batching technique for multiple binary proofs is not equipped in existing works like MatRiCT and MatRiCT+.
  - To mint S accounts the prover needs to run S binary proofs
  - opening vectors (size $|\boldsymbol{b}| \cdot \boldsymbol{S}$) needs to be included in the proof
- Can we adopt a similar amortized technique in [ACF21]?
  - Unfortunately, ACF21 only supports homomorphic relations
  - The binary relation is not!

Partially Amortized Binary Proofs

- Committed Binary Relations:

$$\boldsymbol{Com}(\boldsymbol{b}; \boldsymbol{r}) = B \quad (\mathbf{1})$$
$$\boldsymbol{b} \circ (\boldsymbol{b} - \mathbf{1}) = \mathbf{0} \quad (\mathbf{2})$$

- Idea: we can apply amortization on the (1) while leaving (2) unchanged
  - the prover sends the batched opening for $\sum_{i=0}^{S-1} c^i \, \boldsymbol{r}_i$
  - the prover sends the seperate openings for $\{\boldsymbol{b}_i\}_{i=0}^{S-1}$

Balance Proof

Range Proof and Balance Proof

- Confidentiality: input amounts v_1, v_2 and output amounts v_3, v_4 are hidden, but also allows verifications on:
  - $v_i > 0, \ \forall \ i \in \{1, \ 2, \ 3, 4\}$.
  - $v_1 + v_2 = v_3 + v_4$.

- As v_i can be any value, we need to convert it into a short message before using HMC.

Range Proof and Balance Proof

- To use HMC under lattice settings, MatRiCT [EZS+19] and MatRiCT+ [ESZ20] commits to the *bits* of v_i.
  - $v_3 = 10,\ \mathrm{Bits}(v_3) = (0,1,0,1) = \boldsymbol{b}_3.\ B_3 = Com(\boldsymbol{b}_3).$
- To ensure $v_3 > 0$, we only need to prove $\boldsymbol{b}_3$ is a *binary* vector. ← Binary proof
- But how to prove $v_1 + v_2 = v_3$, as $\mathrm{Bits}(v_1) + \mathrm{Bits}(v_2) \neq \mathrm{Bits}(v_3)$?
  - $\mathrm{Bits}(3) = (1,1,0,0)$
  - $\mathrm{Bits}(7) = (1,1,1,0)$
  - $\mathrm{Bits}(10) = (0,1,0,1)$
  - $\mathrm{Bits}(3) + \mathrm{Bits}(7) = (2,2,1,0) \neq (0,1,0,1) = \mathrm{Bits}(10)$

Balance Proof

- MatRiCT [EZS+19] and MatRiCT+ [ESZ20] use "*corrector values*", $\boldsymbol{c}$, to show $\mathrm{Bits}(v_1) + \mathrm{Bits}(v_2) + \mathrm{Corr}(\boldsymbol{c}) = \mathrm{Bits}(v_3)$.
  - $\boldsymbol{c} = (0,1,1,1,0)$.
  - $(2,2,1,0) + (0 - 2 \times 1, 1 - 2 \times 1, 1 - 2 \times 1, 1 - 2 \times 0) = (0,1,0,1)$.

- However, we further need to ensure each "corrector value" lies in a *proper range*, i.e., $c_i \in [-M + 1, S - 1]$ for M inputs and S outputs, which requires additional *range proofs*.

Balance Proof

- Though $\mathrm{Bits}(v_1) + \mathrm{Bits}(v_2) \neq \mathrm{Bits}(v_3)$, we have another relation:
  - $v_1 + v_2 = v_3 \iff \langle \mathrm{Bits}(v_1) + \mathrm{Bits}(v_2) - \mathrm{Bits}(v_3), \mathbf{2}^n \rangle = 0$
  - $\mathrm{Bits}(3) + \mathrm{Bits}(7) - \mathrm{Bits}(10) = (2,1,1,-1); 2 \times 2^0 + 1 \times 2^1 + 1 \times 2^2 - 1 \times 2^3 = 0$

- We can use this ***inner-product relation*** to prove the balance relation.
  - $\boldsymbol{c} = \sum \mathrm{Bits}(v_{in}) - \sum \mathrm{Bits}(v_{out}); \mathrm{Com}(\boldsymbol{c}) = \sum A_i - \sum B_i$
  - $c_i \in [-M, S]$ can be derived from $\mathrm{Com}(\boldsymbol{c}) = \sum A_i - \sum B_i$
  - We can use the "$\langle \boldsymbol{c}, \mathbf{2}^n \rangle = 0$" relation for balance proofs!

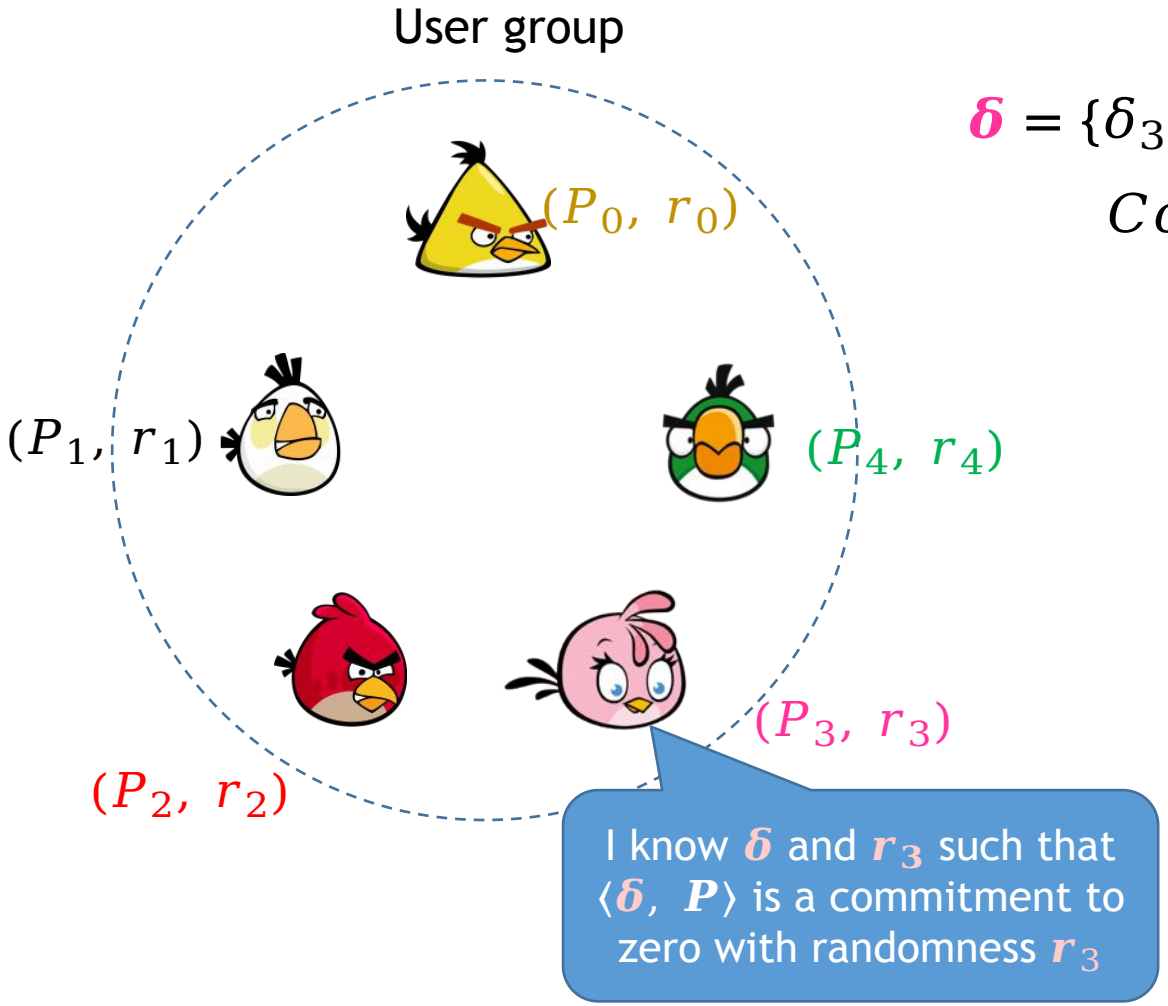
- Unfortunately, it is hard to ensure both $f_i = x \cdot c_i + d_i$ and $\sum f_i 2^i$ being short at the same time.

Balance Proof

- We address this issue by finding d_i's to ensure both f_i's and $\langle \boldsymbol{f}, \mathbf{2}^n \rangle$ are short.
  - Sample d'_i's for $1 \leq i \leq n-1$ and set $d'_0 = d'_n = 0$.
  - Set $d_i = d'_i - 2d'_{i+1}$.
  - $f_i = x \cdot c_i + d_i = x \cdot c_i + d'_i - 2d'_{i+1}$ which is still relatively short.
  - $\langle \boldsymbol{f}, \mathbf{2}^n \rangle = x\langle \boldsymbol{c}, \mathbf{2}^n \rangle + \langle \boldsymbol{d}, \mathbf{2}^n \rangle = \langle \boldsymbol{d}, \mathbf{2}^n \rangle = \sum_{i=0}^{n-1} d'_i - 2\sum_{i=0}^{n-1} d'_{i+1} = d'_0 - 2d'_n = 0$.

Ring Signature

One-out-of-many Proof



$$\boldsymbol{\delta} = \{\delta_{3,0}, \delta_{3,1}, \delta_{3,2}, \delta_{3,3}, \delta_{3,4}\} = \{0,\ 0,\ 0,\ 1,0\};$$

$$Com(0; r_3) = \sum_{i=0}^{4} \delta_{3,i} \cdot P_i = \langle \boldsymbol{\delta},\ \boldsymbol{P} \rangle.$$

$$\boldsymbol{P} = \{P_0,\ P_1,\ P_2,\ P_3,\ P_4\}$$

Verifier

User 3 needs to prove:

1. $\boldsymbol{\delta}$ is a binary vector; ← Binary proof
2. $\boldsymbol{\delta}$ only have one “1”;
3. $\langle \delta, P \rangle$ is a commitment to 0.

One-out-of-many Proof

- MatRiCT [EZS+19] and MatRiCT+ [ESZ20] use this technique [GK15] directly in lattice-based RingCT protocols.

- Unfortunately, in lattice settings, the *binary proof* requires *larger* parameters than other parts, which results in a *larger* proof size.

Linear Sum Proof

- Can we remove the costly *binary proof* ?
  - This indicates a weaker relation: the “linear sum relation”.
- Is this weaker relation still secure for ring signatures?
  - ***Linear sum relation is sufficient for ring signatures!*** (see our paper for the detailed proofs).

The prover needs to prove:

1. ~~$\boldsymbol{b}$ is a binary vector;~~
2. Not all b_i’s are “0”;
3. $\langle \boldsymbol{b}, \boldsymbol{P} \rangle$ is a commitment to 0.

Linear Sum Proof

- Can we remove the costly *binary proof* [GZGX21]?
  - This indicates a weaker relation: the “linear sum relation”.
- Is this weaker relation still secure for ring signatures?
  - ***Linear sum relation is sufficient for ring signatures!*** (see our paper for the detailed proofs).

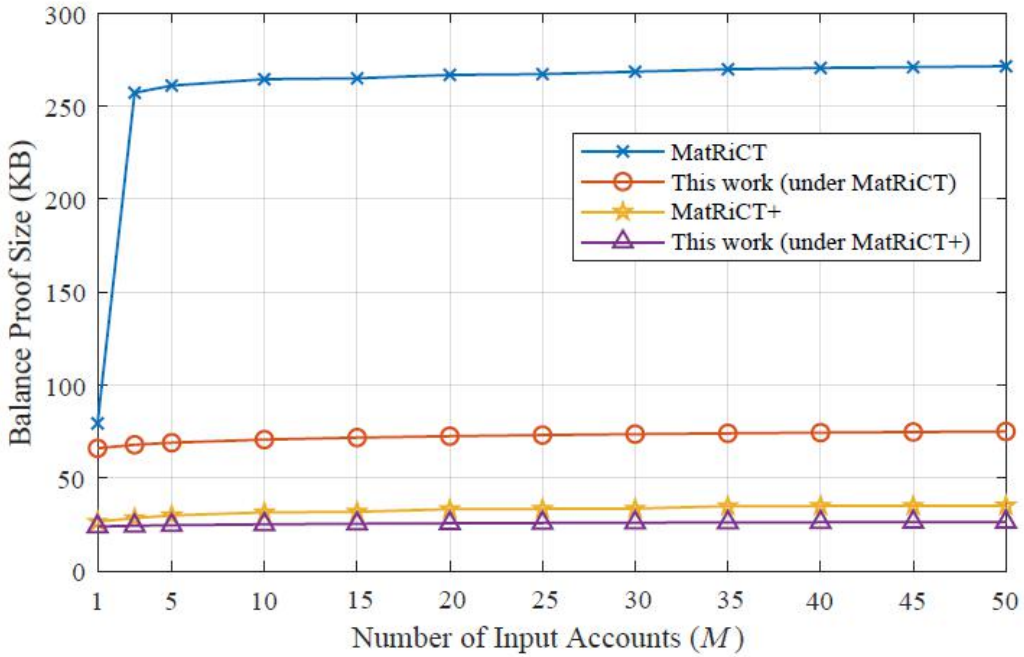
- Unfortunately, the linear sum proof cannot use some techniques in [GK15] to reduce the proof size.

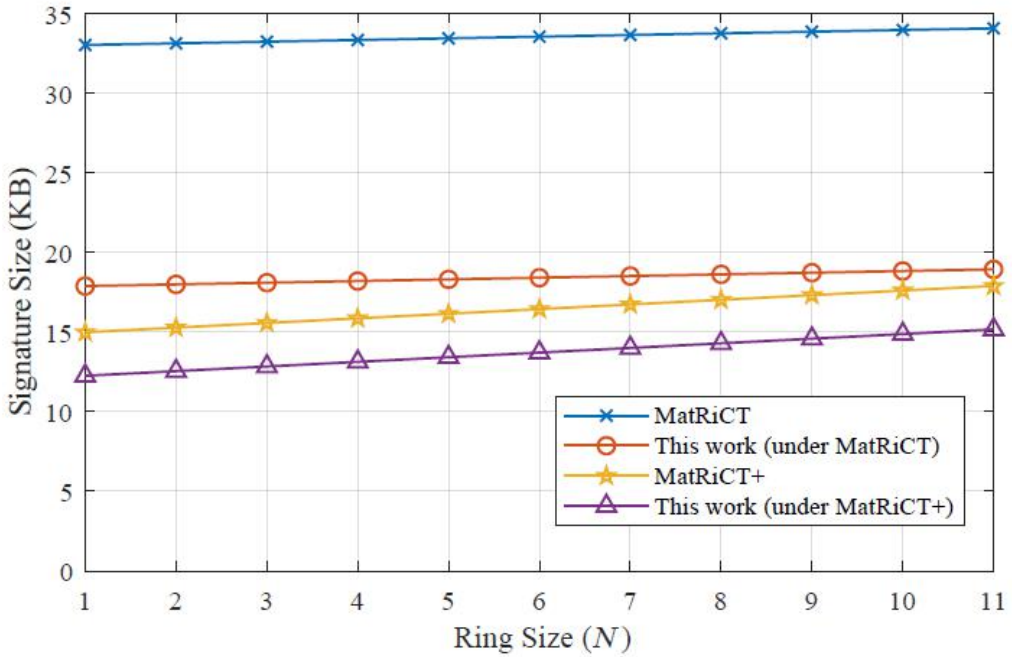
- But we may consider an unbalanced relation: the *prover* runs with a *stricter relation*, and the *verifier* checks with a *relaxed relation* [GZGX21].

Lattice-based RingCT

Performance

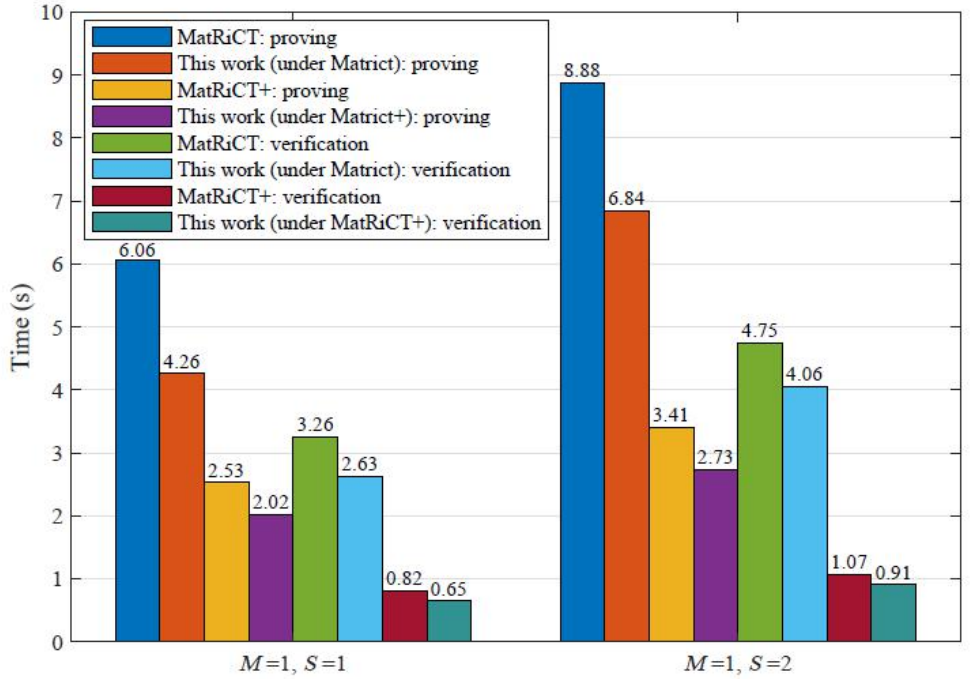
- Balance proof: reduce 90% size of MatRiCT and 30% of MatRiCT+.
- Ring signature: reduce 60% size of MatRiCT and 20% of MatRiCT+.

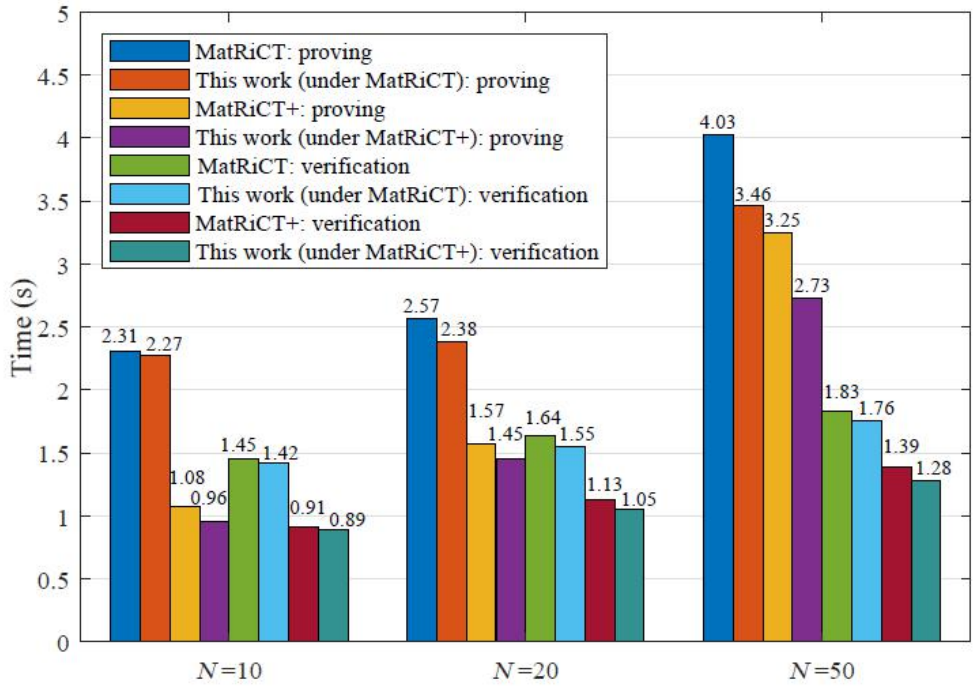




Performance

- Balance proof: reduce 70% size of MatRiCT and 20% of MatRiCT+.
- Ring signature: reduce 60% size of MatRiCT and 15% of MatRiCT+.





Conclusion

- Anonymous cryptocurrecy: need to hide amounts and indentities
- Binary proofs: partially amortized
- Balance proofs: corrector values -> inner-product relations
- Ring signatures: one-out-of-many proofs -> linear-sum relations
- RingCT: linkable ring signatures + serial number checks

Thanks!

Q&A