Computationally Efficient Asynchronous MPC with Linear Communication and Low Additive Overhead

Akhil Bandarupalli
Purdue University

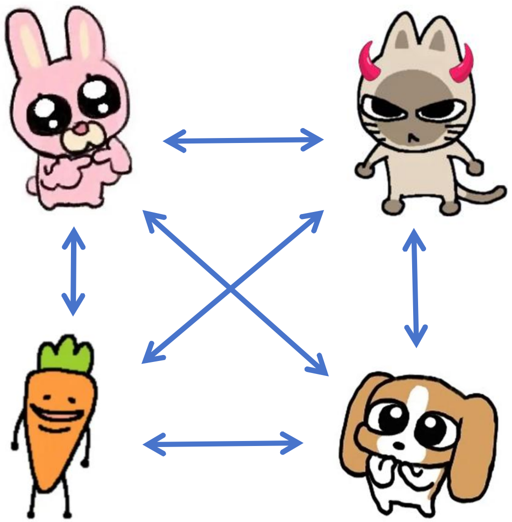
Xiaoyu Ji
Tsinghua University

Aniket Kate
Purdue University & Supra Research

Chen-Da Liu-Zhang
Lucerne U. of Applied Sciences and Arts
& Web3 Foundation

Yifan Song
Tsinghua University
& Shanghai Qi Zhi Institute

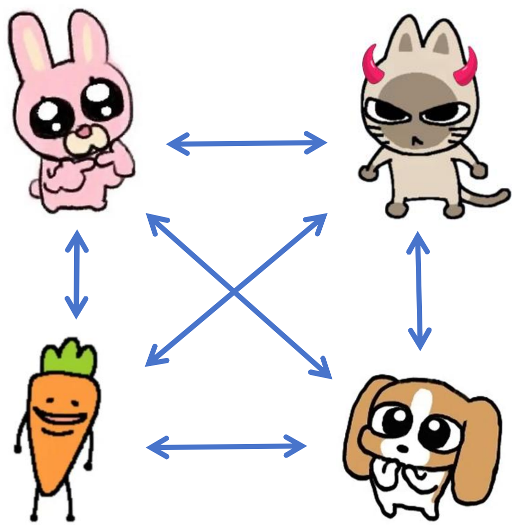
Multiparty Computation



Setting

- n parties, t of them are corrupted
- Malicious Adversary
- Asynchronous Network
- Complete network of bilateral channels

Multiparty Computation



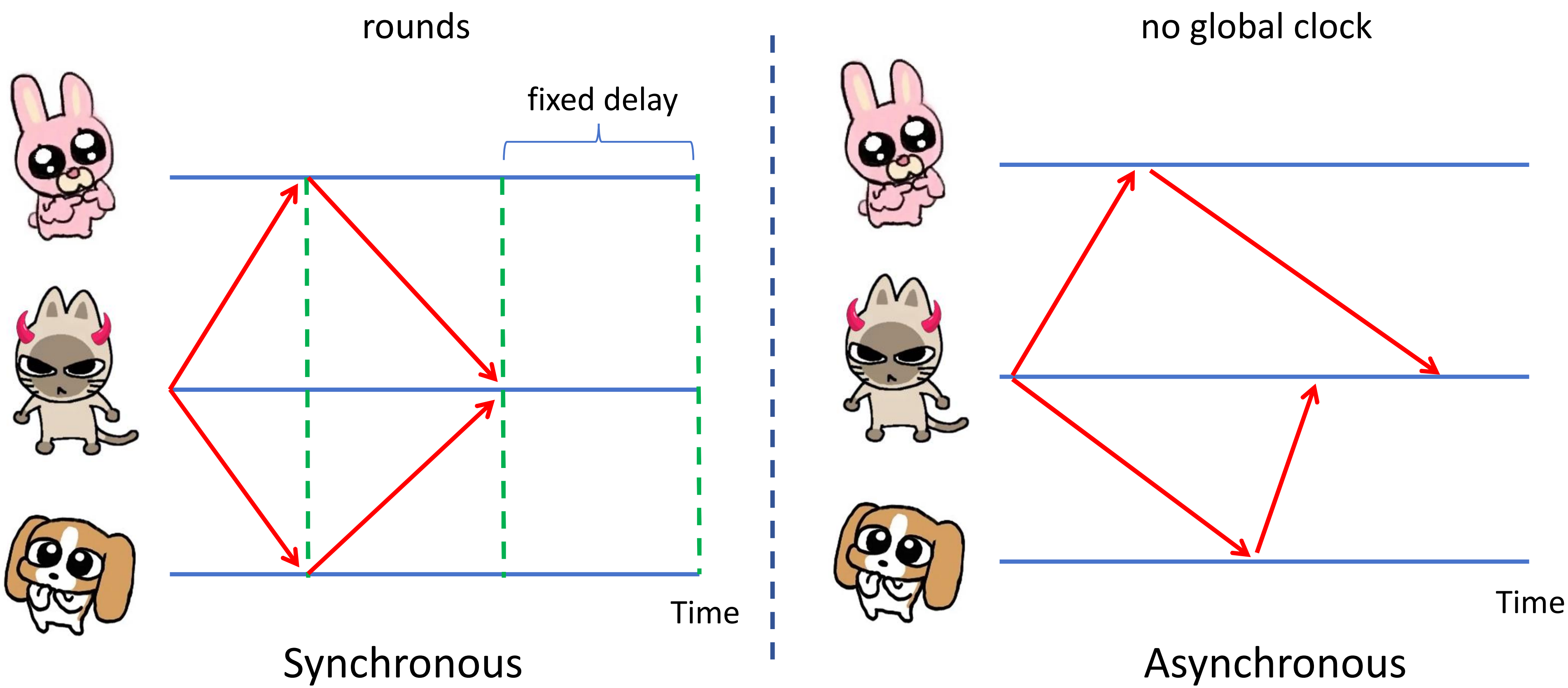
Setting

- n parties, t of them are corrupted
- Malicious Adversary
- Asynchronous Network
- Complete network of bilateral channels

Goal

- Correctness: All honest parties finally obtain correct output (GOD)
- Privacy: Adversary does not learn anything beyond the computed output

Network Setting



Landscape of Asynchronous MPC protocols

	Communication	Computation	Assumption
[CP23]	$O(Cn^4 + n^6)$	——	Secure Channels
[GLS24]	$O(Cn + Dn^2 + n^{14})$	——	Secure Channels

The communication complexity of information-theoretically secure AMPC protocols is too high to be practical.

[CP23] Ashish Choudhury and Arpita Patra. On the communication efficiency of statistically secure asynchronous mpc with optimal resilience.
[GLS24] Vipul Goyal, Chen-Da Liu-Zhang, and Yifan Song. Towards achieving asynchronous MPC with linear communication and optimal resilience.

Landscape of Asynchronous MPC protocols

	Communication	Computation	Assumption
[CP23]	$O(Cn^4 + n^6)$	——	Secure Channels
[GLS24]	$O(Cn + Dn^2 + n^{14})$	——	Secure Channels
[CP15]	$O(Cn + Dn^2 + n^4)$	$\Omega(Cn)$ SHE	SHE
[SLL+24]	$O(Cn^2 + n^3 log\ n)$	$O(Cn^2)$ DLE	DLog + q-SDH

SHE: somewhat Homomorphic Encryption; DLE: Discrete-log exponentiation; q-SDH: q-Strong Diffie Hellman hardness assumptions.

The computational complexity of the existed AMPC protocols is too high to be practical.

[CP15] Ashish Choudhury and Arpita Patra. 2015. Optimally resilient asynchronous MPC with linear communication complexity.
[SLL+24] Yuan Su, Yuan Lu, Jiliang Li, Yuyi Wang, Chengyi Dong, and Qiang Tang. Dumbo-mpc: Efficient fully asynchronous mpc with optimal resilience.

The space in between: Lightweight Cryptography

Term coined by [SS24]

[SS24] Victor Shoup and Nigel P. Smart. Lightweight asynchronous verifiable secret sharing with optimal resilience.

Lightweight Cryptography

- Symmetric key cryptographic operations, Pseudorandom Functions, Hash computations
- **Computational Efficiency**: 100-1000x faster than heavyweight cryptographic operations

Operation	Computation Time
Discrete Log Exponentiation	70 micro seconds
Bilinear Pairings	600 micro seconds
Hash Computation	0.5 micro seconds
Hardware Accelerated Hash	0.04 micro seconds

Landscape of Asynchronous MPC protocols

	Communication	Computation	Assumption
[CP23]	$O(Cn^4 + n^6)$	——	Secure Channels
[GLS24]	$O(Cn + Dn^2 + n^{14})$	——	Secure Channels
[CP15]	$O(Cn + Dn^2 + n^4)$	$\Omega(Cn)$ SHE	SHE
[SLL+24]	$O(Cn^2 + n^3 log\, n)$	$O(Cn^2)$ DLE	DLog + q-SDH
[Mom24]	$O(Cn^2 + n^6)$	$O(Cn^2)$ Hash	ROM

By building based on ROM, [Mom24] balances the communication and computation, but is still not efficient.

[Mom24] Atsuki Momose. Practical asynchronous mpc from lightweight cryptography.

Landscape of Asynchronous MPC protocols

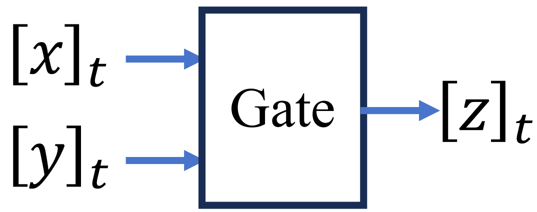
	Communication	Computation	Assumption
[CP23]	$O(Cn^4 + n^6)$	——	Secure Channels
[GLS24]	$O(Cn + Dn^2 + n^{14})$	——	Secure Channels
[CP15]	$O(Cn + Dn^2 + n^4)$	$\Omega(Cn)$ SHE	SHE
[SLL+24]	$O(Cn^2 + n^3 log\, n)$	$O(Cn^2)$ DLE	DLog + q-SDH
[Mom24]	$O(Cn^2 + n^6)$	$O(Cn^2)$ Hash	ROM
This work	$O(Cn + Dn^2 + n^4)$	$O(Cn)$ Hash	ROM

General Approach

Input: Each party secretly shares his input

$$x \longrightarrow [x]_t$$

Computation: All parties jointly compute a secret sharing for every wire value

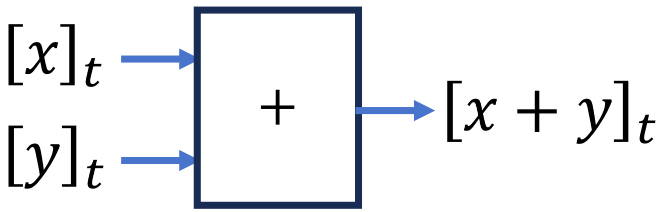


Output: All parties reconstruct the sharings for output wires

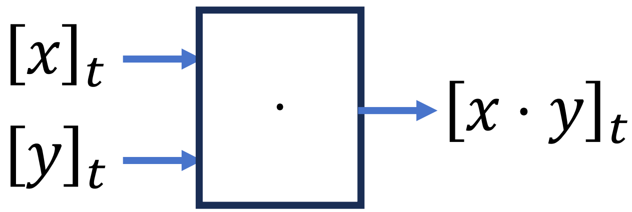
$$[z]_t \longrightarrow z$$

General Approach

Computation: All parties jointly compute a secret sharing for every wire value



Linear Homomorphism



Beaver Triples

Shamir Secret Sharing

Use Asynchronous Complete Secret Sharing (ACSS)

- Allow a dealer to share degree-t Shamir sharings such that:

If an honest party accepts his share, all honest parties eventually obtain valid shares

Lie on a valid degree-t polynomial.

Shamir Secret Sharing

Use Asynchronous Complete Secret Sharing (ACSS)

Lie on a valid degree-t polynomial.

- Allow a dealer to share degree-t Shamir sharings such that:

If an honest party accepts his share, all honest parties eventually obtain valid shares

Benefit: Computation phase can be achieved with $O(Cn + Dn^2)$ communication.

Shamir Secret Sharing

Use Asynchronous Complete Secret Sharing (ACSS)

- Allow a dealer to share degree-t Shamir sharings such that:

  If an honest party accepts his share, all honest parties eventually obtain valid shares

Lie on a valid degree-t polynomial.

Benefit: Computation phase can be achieved with $O(Cn + Dn^2)$ communication.

Best Prior Works:

IT-Secure: $O(n)$ per sharing **plus** $O(n^{12}\kappa)$ additive overheads [JLS24]

Assume RO: $O(n^2)$ per sharing **plus** $\widetilde{O}(n^3)$ additive overheads [SS24]

[JLS24] Xiaoyu Ji, Junru Li, and Yifan Song. Linear-communication asynchronous complete secret sharing with optimal resilience.

Shamir Secret Sharing

Use Asynchronous Complete Secret Sharing with Identifiable Abort (ACSS-Id)

- Weaker than ACSS, but still guarantees the reconstruction of the dealer's secret if terminated.

  If an honest party accepts his share, all honest parties eventually obtain valid shares or a proof;

Shamir Secret Sharing

Use Asynchronous Complete Secret Sharing with Identifiable Abort (ACSS-Id)

- Weaker than ACSS, but still guarantees the reconstruction of the dealer's secret if terminated.

  If an honest party accepts his share, all honest parties eventually obtain valid shares or a proof;

Best Prior Work:

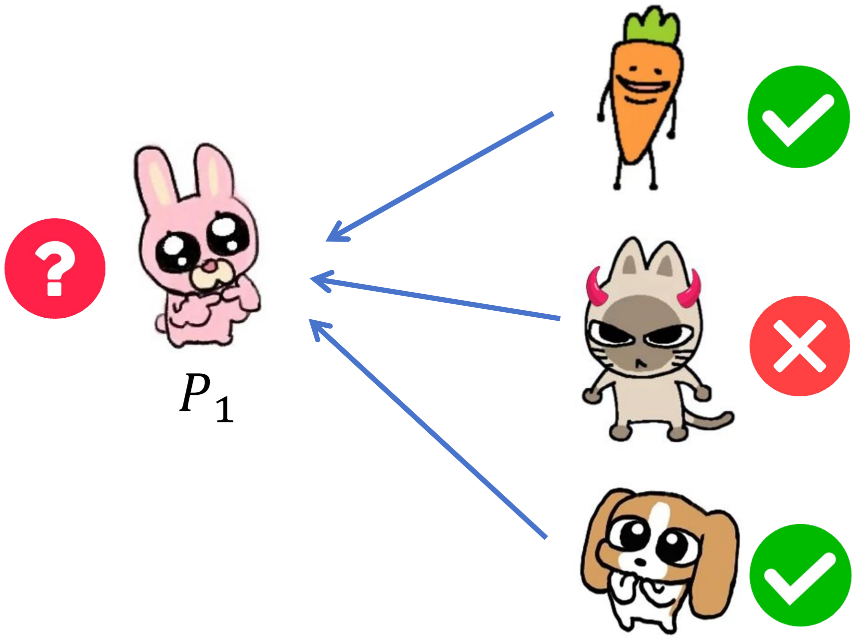
Assume RO: $O(n)$ per sharing **plus** $\widetilde{O}(n^2)$ additive overheads [SS24]

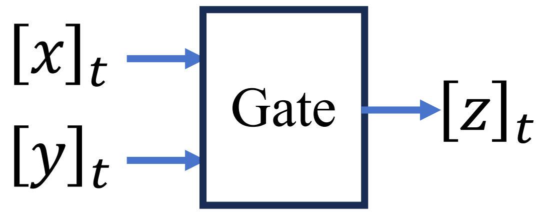
Can we aggressively use ACSS-Id to prepare degree-t Shamir sharings?

Problem

Online Phase:

All parties compute a multiplication gate with input $[x]_t, [y]_t$ and a triple $([a]_t, [b]_t, [c]_t)$



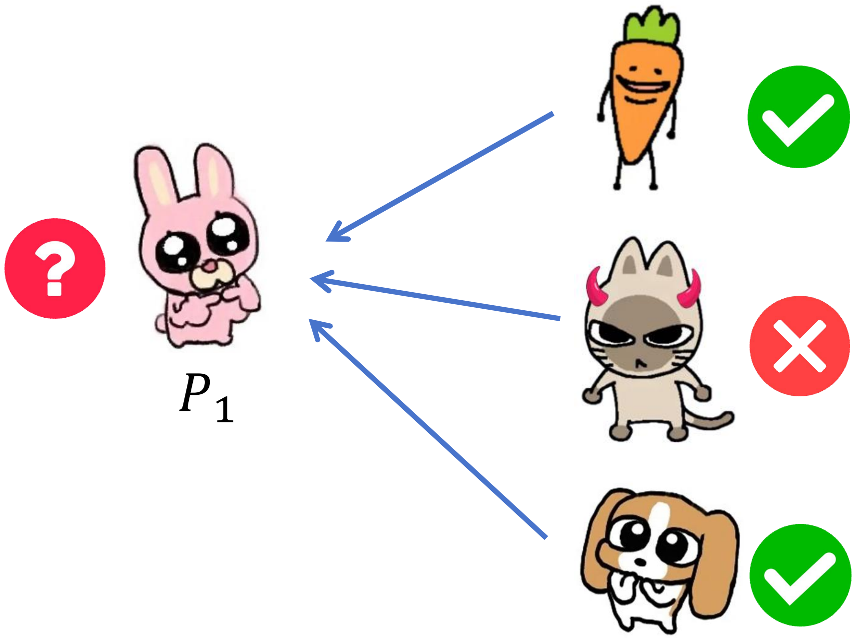


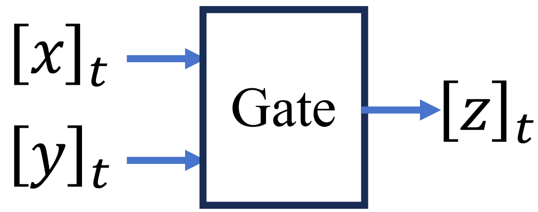
Reconstruct $[x + a]_t, [y + b]_t$

Problem

Online Phase:

All parties compute a multiplication gate with input $[x]_t, [y]_t$ and a triple $([a]_t, [b]_t, [c]_t)$





Reconstruct $[x + a]_t, [y + b]_t$

New Issue: Public Reconstruction cannot be done just by Error Correction!

Our Idea

Solution: Party Elimination based Public Reconstruction

For each degree-t Shamir secret sharing $[s]_t$, we can decompose it into:

$$[s]_t = \sum_{i=1}^{n} [s_i]_t$$

where each $[s_i]_t$ is distributed by party P_i through ACSS-Id

Our Idea

Solution: Party Elimination based Public Reconstruction

For each degree-t Shamir secret sharing $[s]_t$, we can decompose it into:

$$[s]_t = \sum_{i=1}^{n} [s_i]_t$$

where each $[s_i]_t$ is distributed by party P_i through ACSS-Id

Observation: If a party cannot compute his share of $[s]_t$, he can use the proof to accuse a corrupted party.

Our Idea

Solution: Party Elimination based Public Reconstruction

$$[s]_t = [s_1]_t + \sum_{i=2}^{n} [s_i]_t \quad \Longleftrightarrow \quad [s']_t = s_1 + \sum_{i=2}^{n} [s_i]_t$$

Note that $s = s'$

Observation: If a party cannot compute his share of $[s]_t$, he can use the proof to accuse a corrupted party.

Our Idea

Solution: Party Elimination based Public Reconstruction

Only reconstruct s_1!

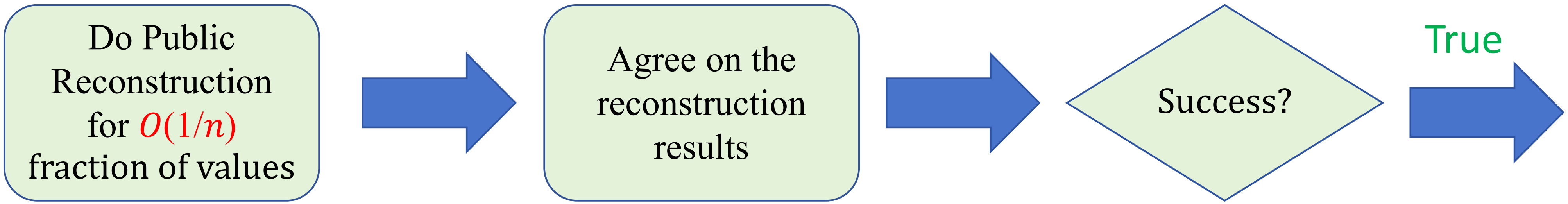
$$[s]_t = [s_1]_t + \sum_{i=2}^{n} [s_i]_t \iff [s']_t = \boxed{s_1} + \sum_{i=2}^{n} [s_i]_t$$

Note that $s = s'$

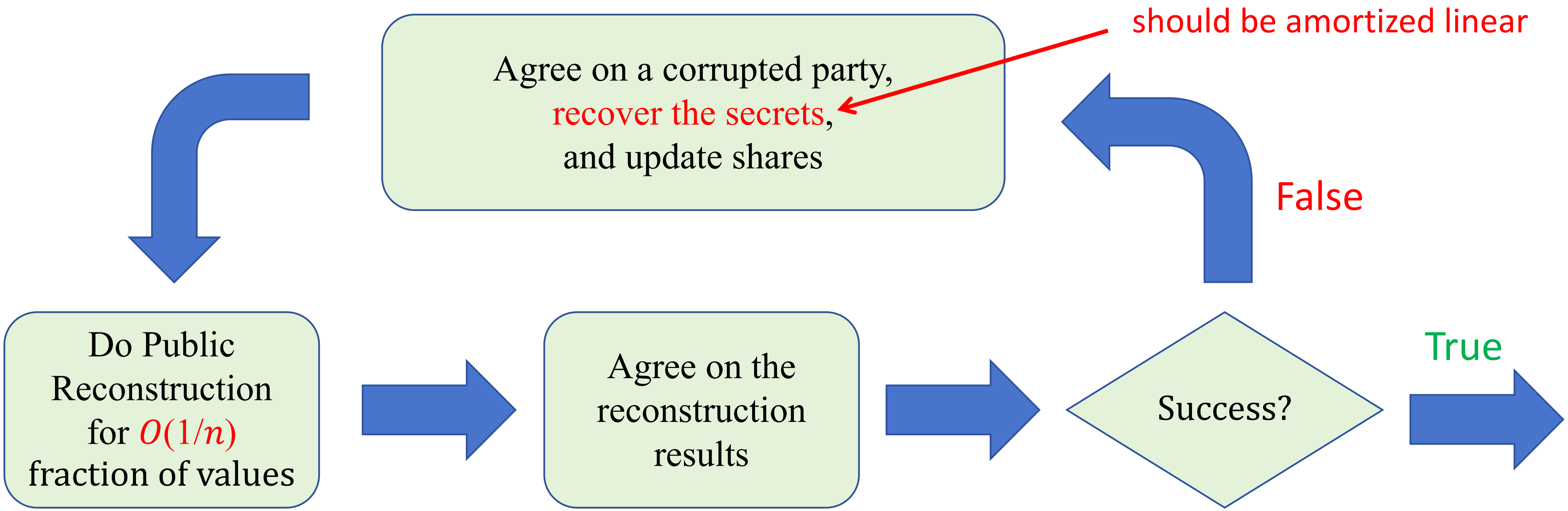
Observation: If a party cannot compute his share of $[s]_t$, he can use the proof to accuse a corrupted party.

Once all parties agree on a corrupted party, they reconstruct his secrets and update their shares. The public reconstruction will not fail due to this corrupted party next time!

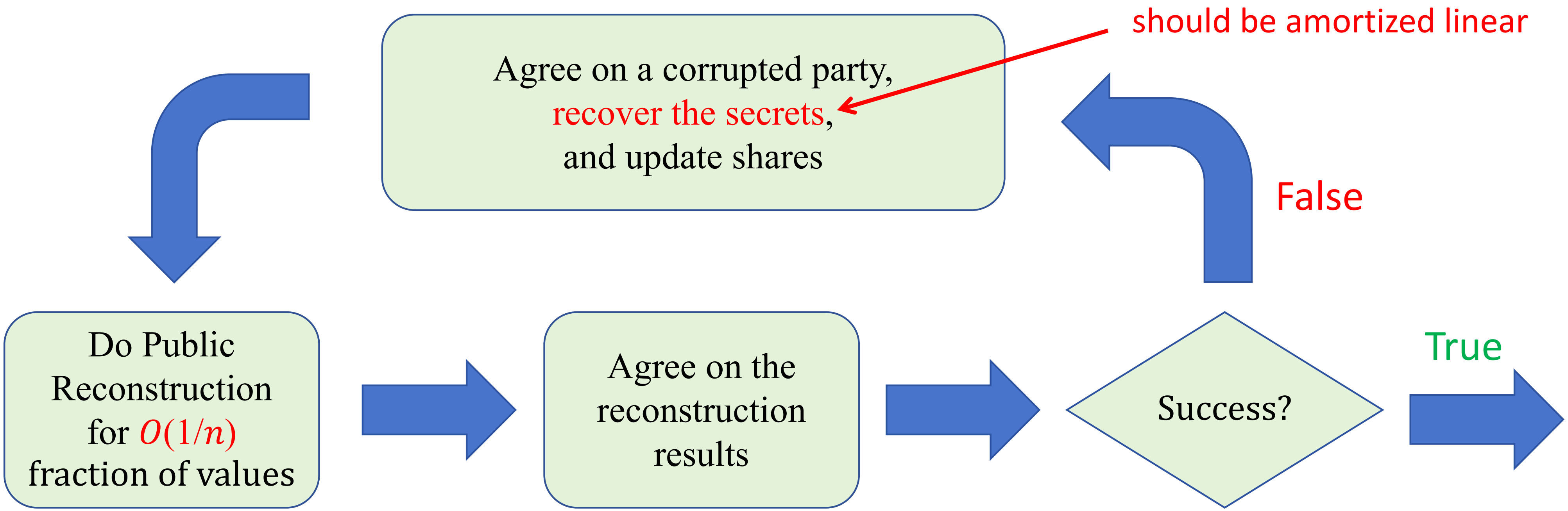
Summary of the Online Phase



Summary of the Online Phase



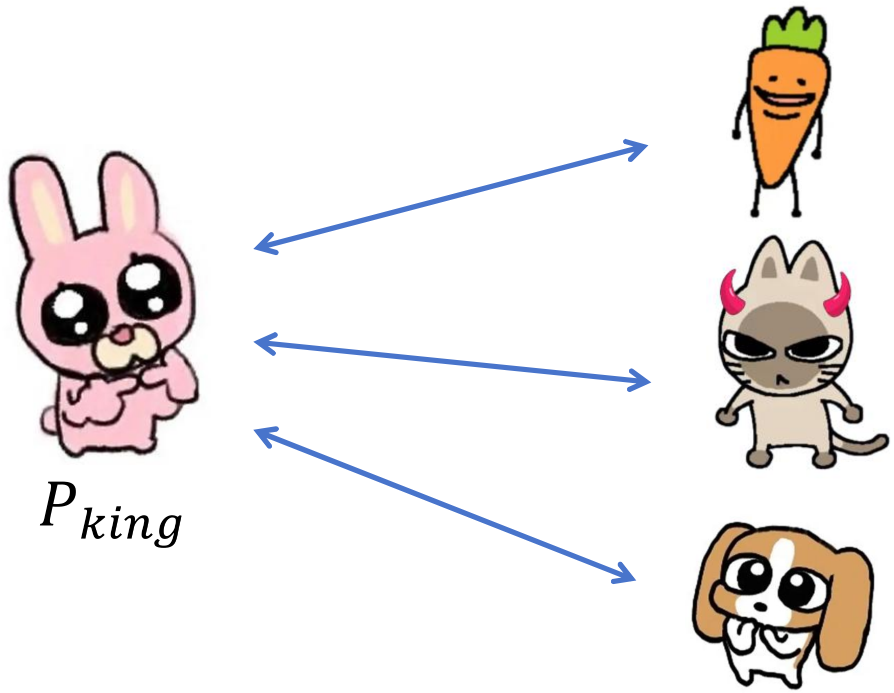
Summary of the Online Phase



Communication complexity: $O(Cn + Dn^2 + n^4)$ field elements.

Triple Generation

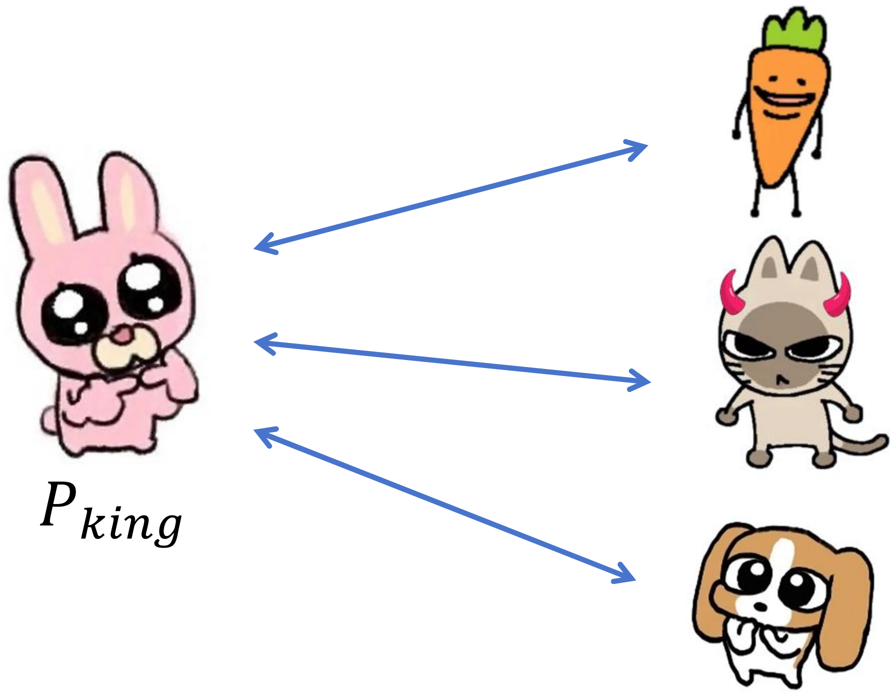
DN Technique [DN07]



Triple Generation

DN Technique:

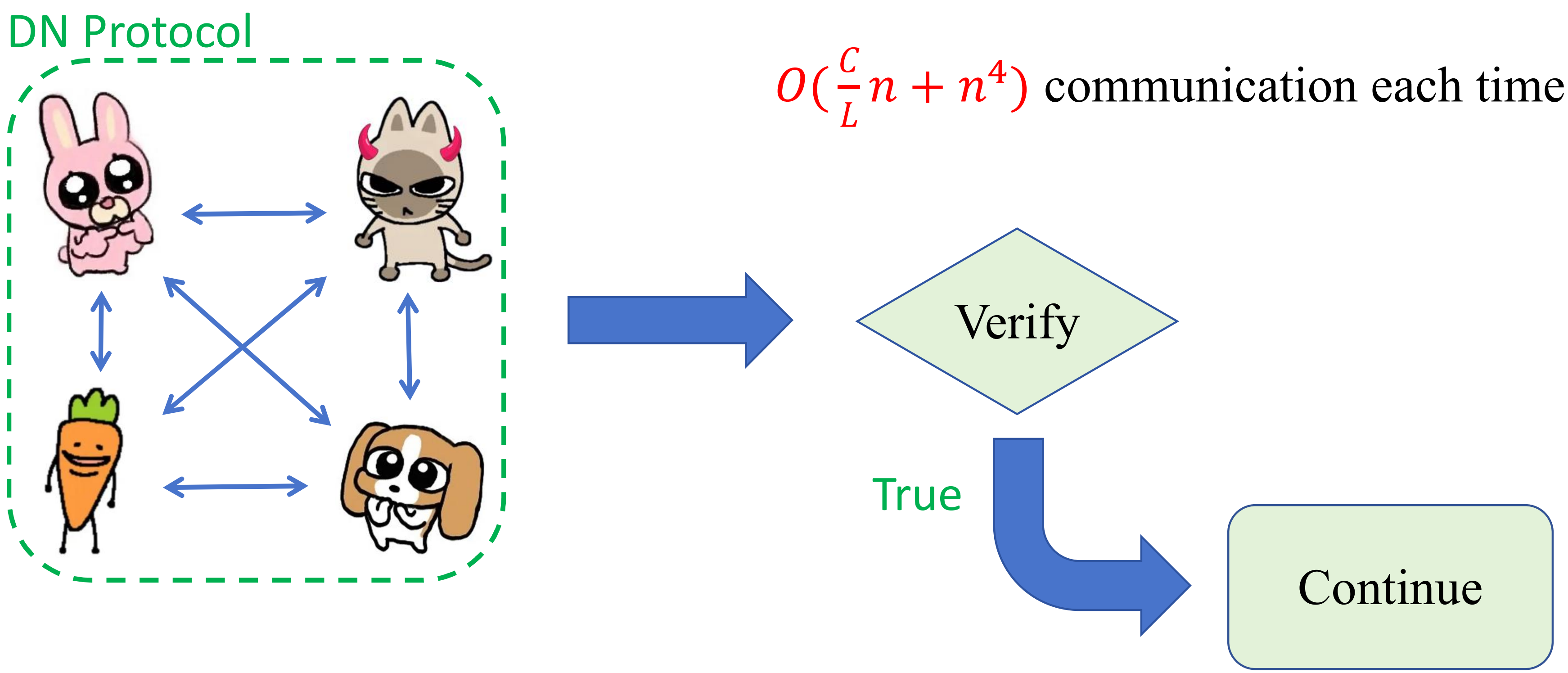
- Difficult in asynchronous setting: the king may not be online.



How Previous Approach Works

Construction from [Mom24]:

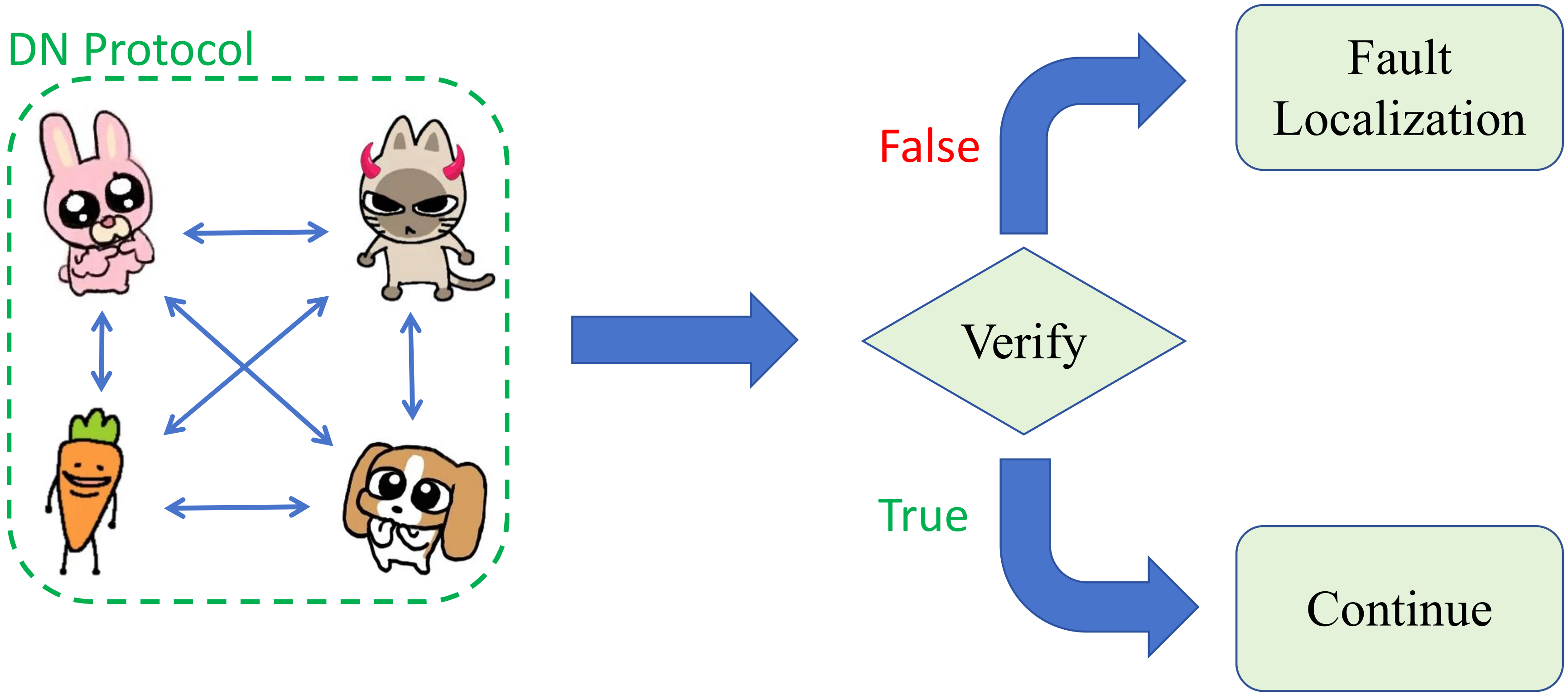
- Use DN + Party Elimination framework: divide the generation of triples into L segments



How Previous Approach Works

Construction from [Mom24]:

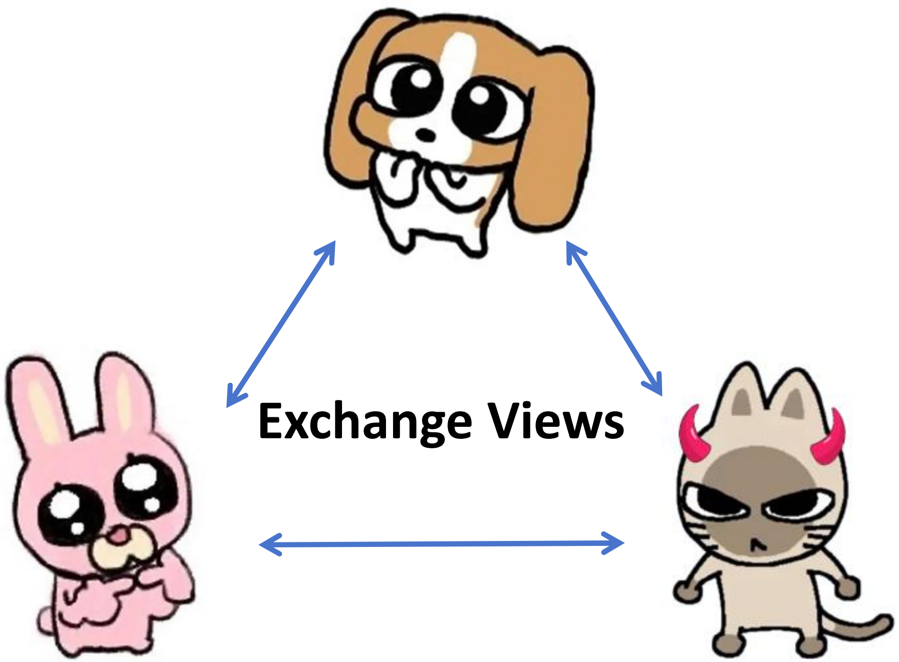
- Use DN + Party Elimination framework: divide the generation of triples into L segments



How Previous Approach Works

Construction from [Mom24]:

- Do Fault Localization if the verification fails



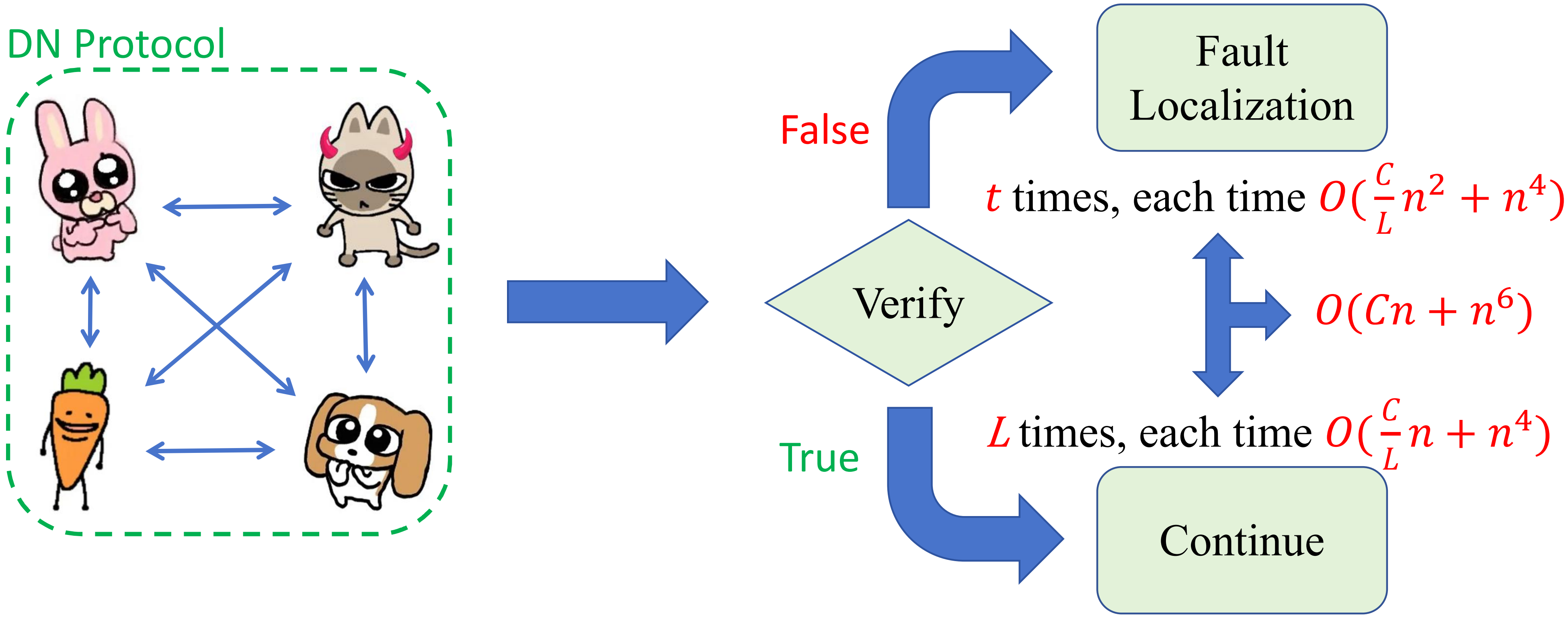
$O(\frac{C}{L}n^2 + n^4)$ communication each time

To achieve linear communicaiton, $L = O(n^2)$

How Previous Approach Works

Construction from [Mom24]:

- Use DN + Party Elimination framework: divide the generation of triples into L segments



Our Idea

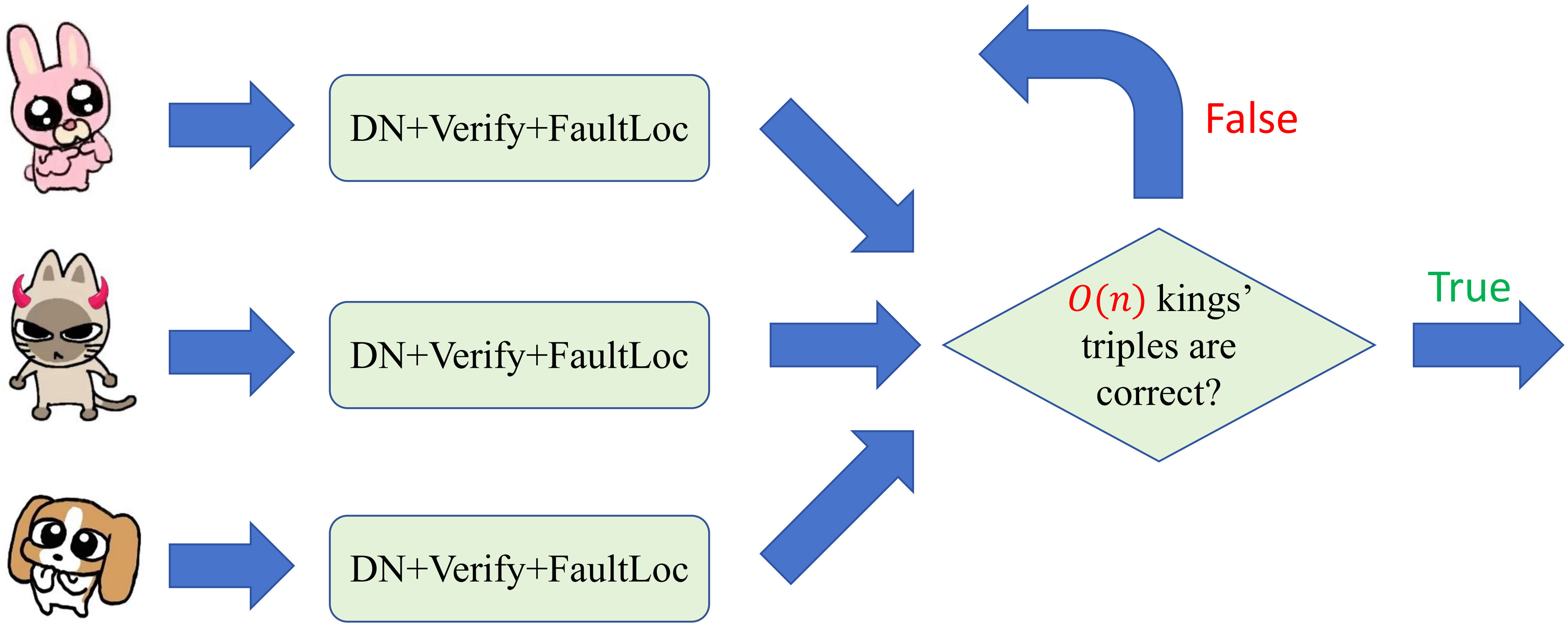
Reducing the additive overhead from $O(n^6)$ to $O(n^4)$:

- Reveal partial views to each party for Fault Localization

$$O(\frac{C}{L}n^2 + n^4) \Rightarrow O(\frac{C}{L}n + n^3)$$

Our Idea

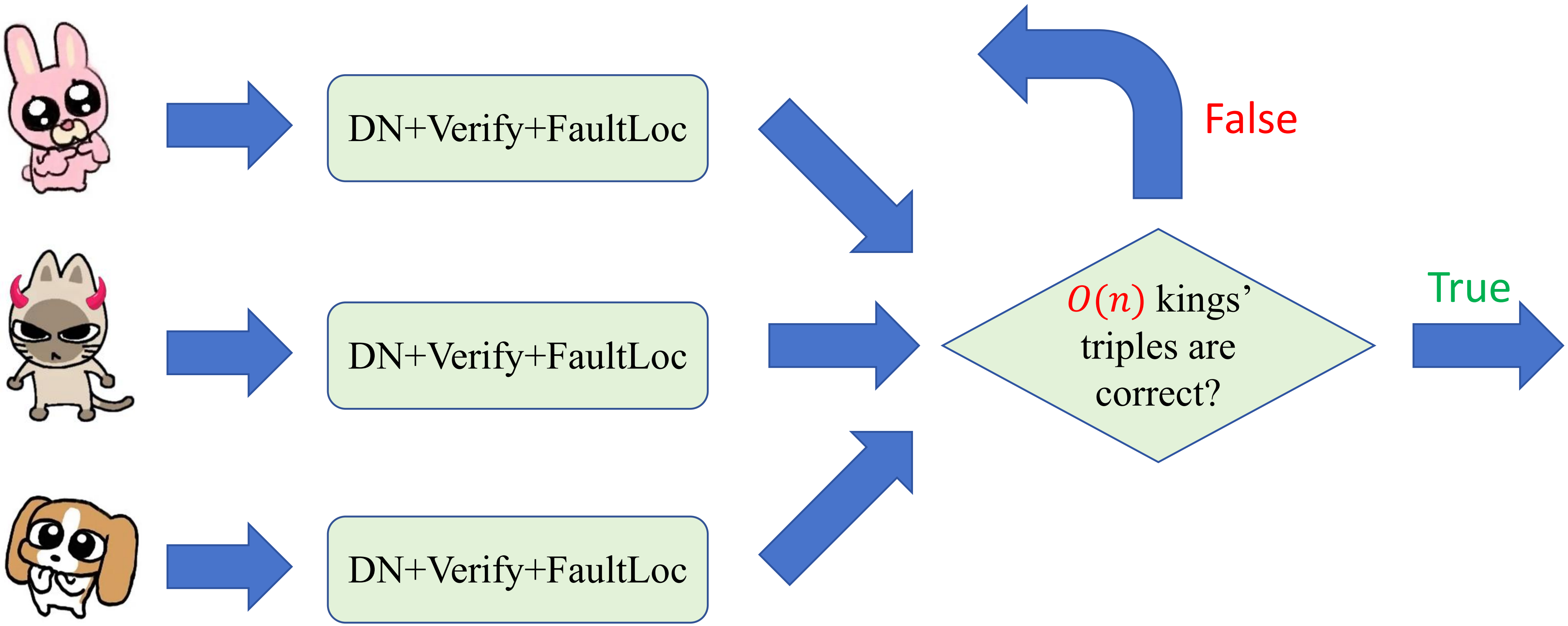
Divide the generation of triples into L segments, each king generates $O(1/(nL))$ fraction of triples in each segment



Our Idea

Additive Overheads are $O(Ln^3)$

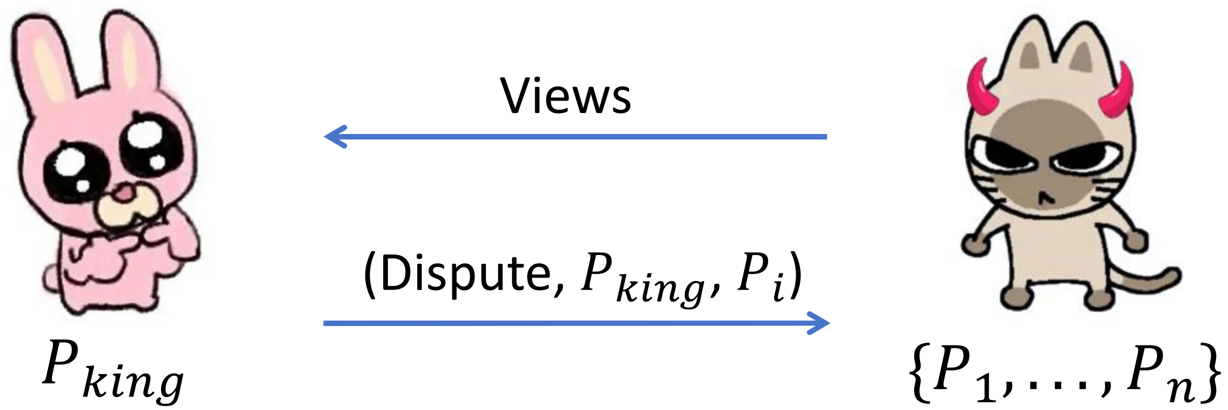
Divide the generation of triples into L segments, each king generates $O(1/(nL))$ fraction of triples in each segment



Our Idea

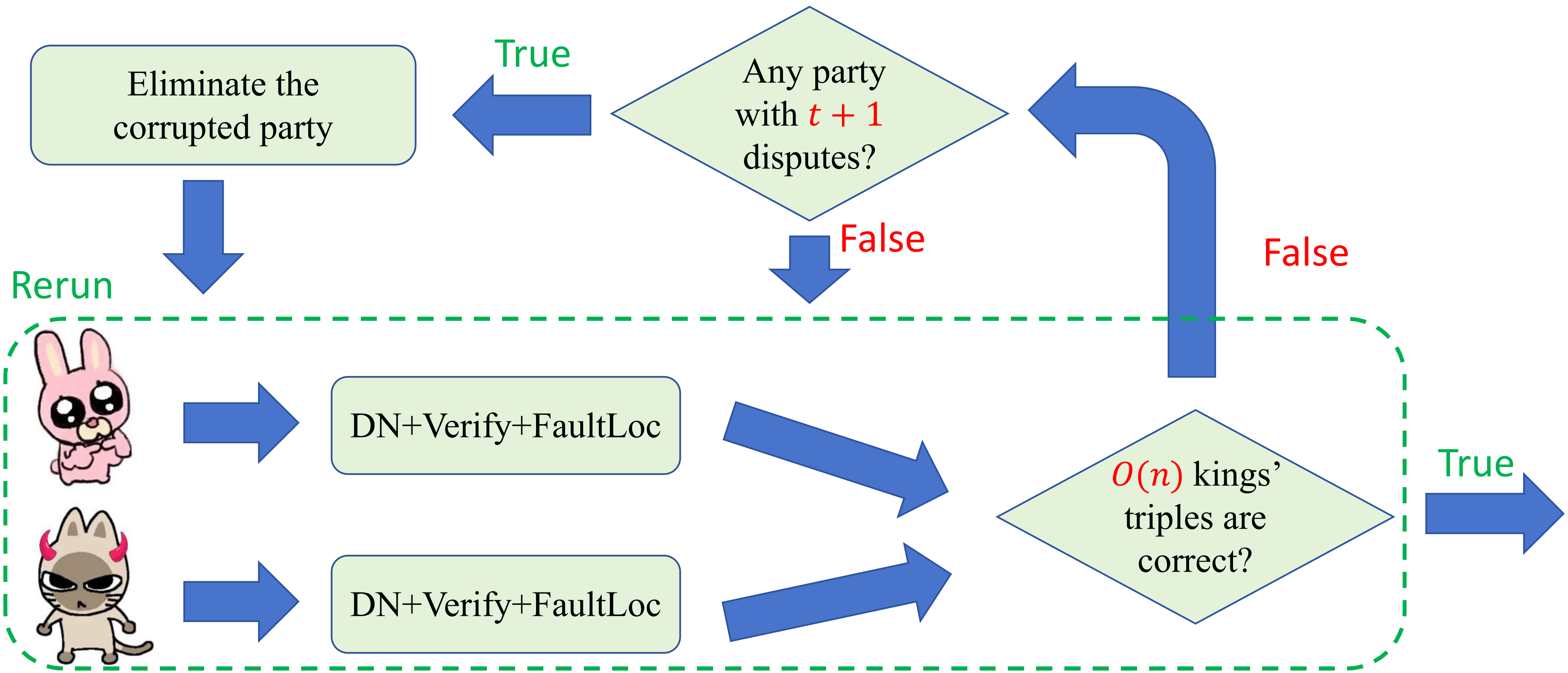
Goal: guarantee termination in $L = O(n)$ segments.

Dispute Control: only a corrupted party will conflict with more than t parties.



If all parties fail the generation in one segment, there are at least $O(n)$ new dispute pairs. All parites will fail the generation for at most $O(n^2/n) = O(n)$ segments.

Summary of Triple Generation



Communication complexity: $O(Cn+n^4)$ field elements.

Thank you!

Q & A

Link to Paper

https://eprint.iacr.org/2024/1666

Credit: Icons: https://www.flaticon.com/