On the Exact Round Complexity of Self-Composable Two-Party Computation

Sanjam Garg — Berkeley, UNIVERSITY OF CALIFORNIA

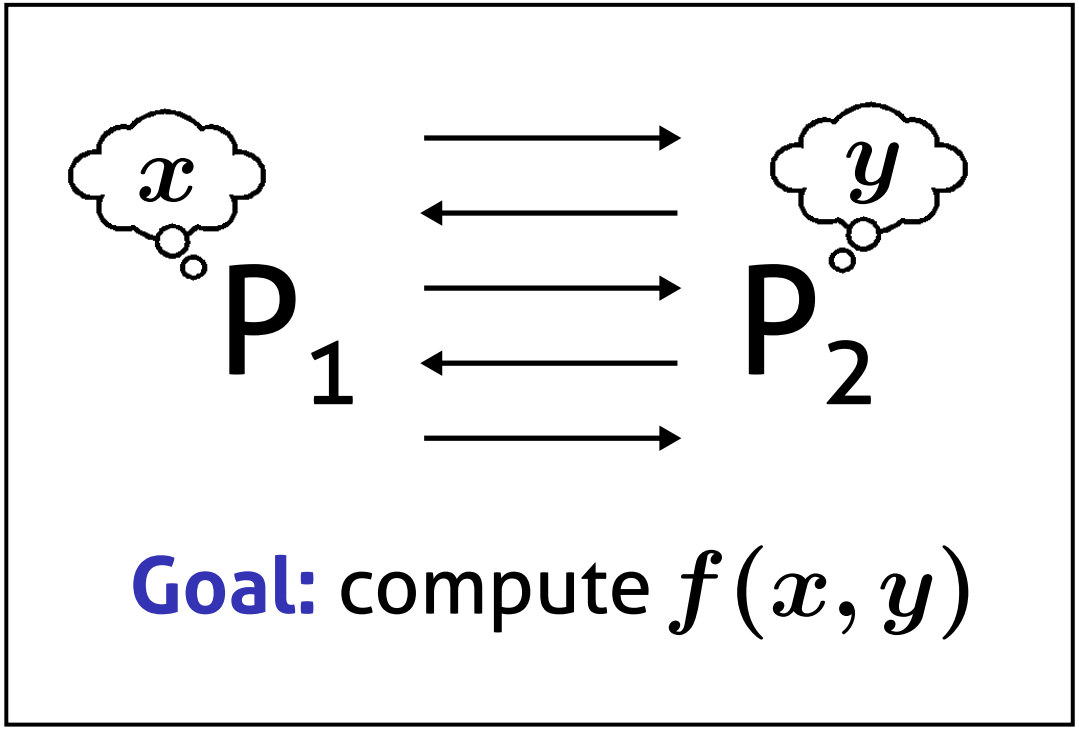
Susumu Kiyoshima — NTT

Omkant Pandey — Stony Brook University

Outline

Secure Two-Party Computation (2PC)

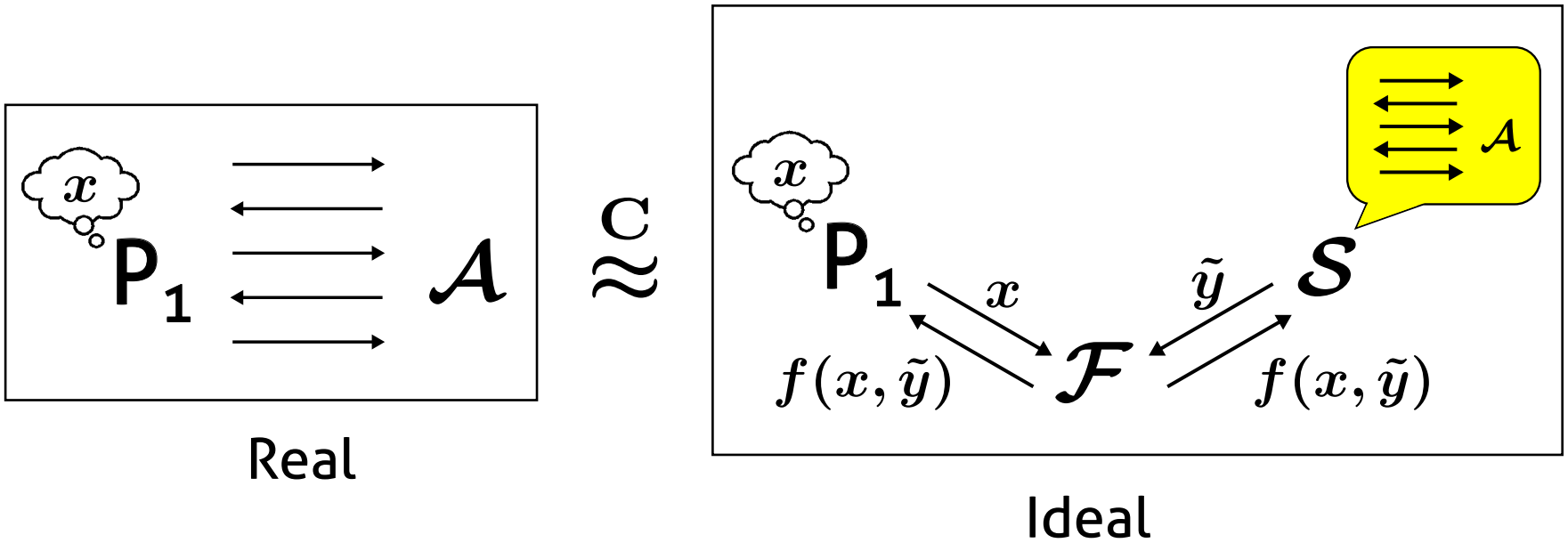
Goal: Two parties jointly compute *arbitrary* function



Security: Correctness, Privacy, Input independence, ...

Security Definition of 2PC

- Secure $\Leftrightarrow \forall$ malicious adv $\mathcal{A}$, $\exists$ simulator $\mathcal{S}$ s.t.



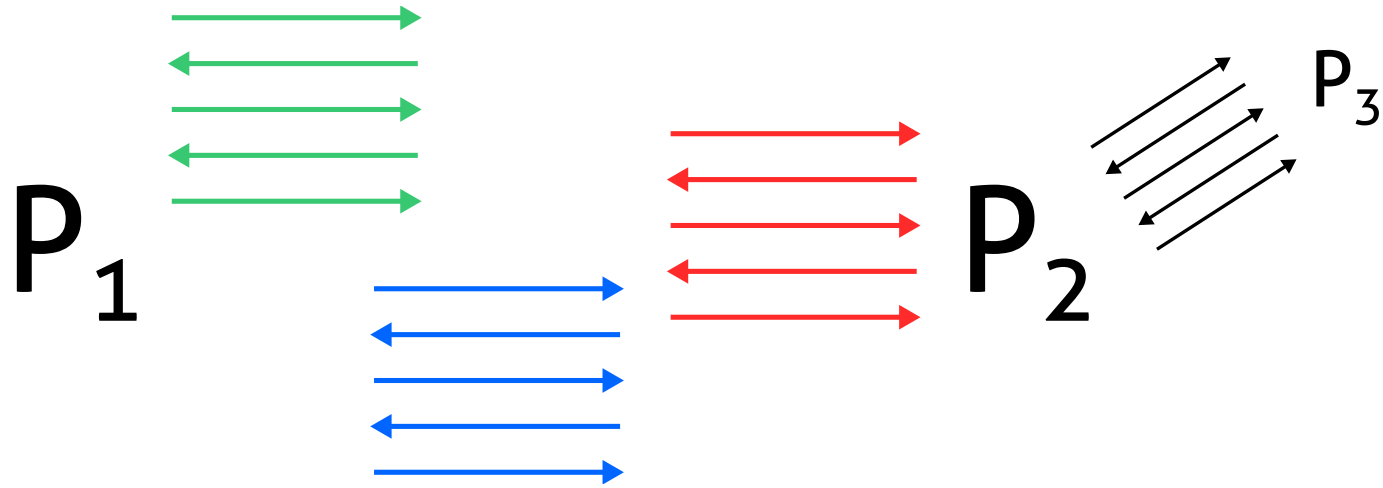
Real

Ideal

Guarantee: Real is *as secure as* Ideal

Concurrently Secure 2PC

- Two parties might join many sessions concurrently (possibly with other parties)



- Concurrent setting is more general, realistic, …

How to achieve Concurrent 2PC

Difficulty:

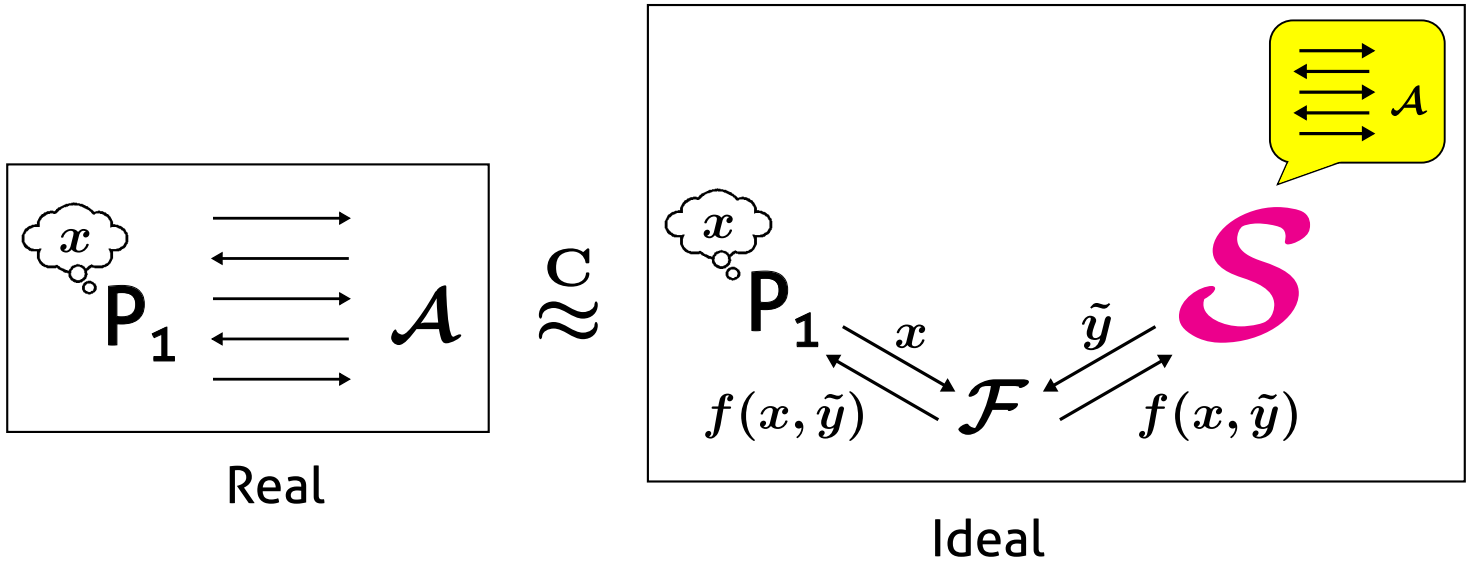
- impossible to achieve in plain model [CKL03, Lin04]

Bypass: Relaxed security definitions

- Super-polynomial-time simulation (SPS) [Pas03,PS04,BS05, ...]
- Angel-based UC [PS04,MMY06,CLP10, ...]
- Input indistinguishability [MPR06,GGJS10]
- Multiple ideal-query [GJO10,GJ13,CGJ15]

SPS security of Concurrent 2PC

- Simulator can run in super-poly time



Real

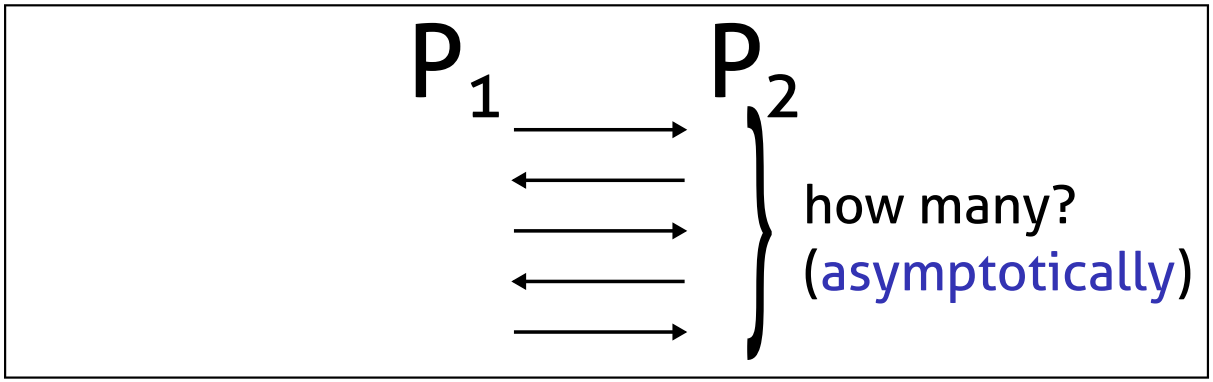
Ideal

Guarantee: Any attack can be simulated in Ideal in super-poly time

⇒ OK if Ideal is secure against super-poly adv

What is Known about Concurrent SPS 2PC

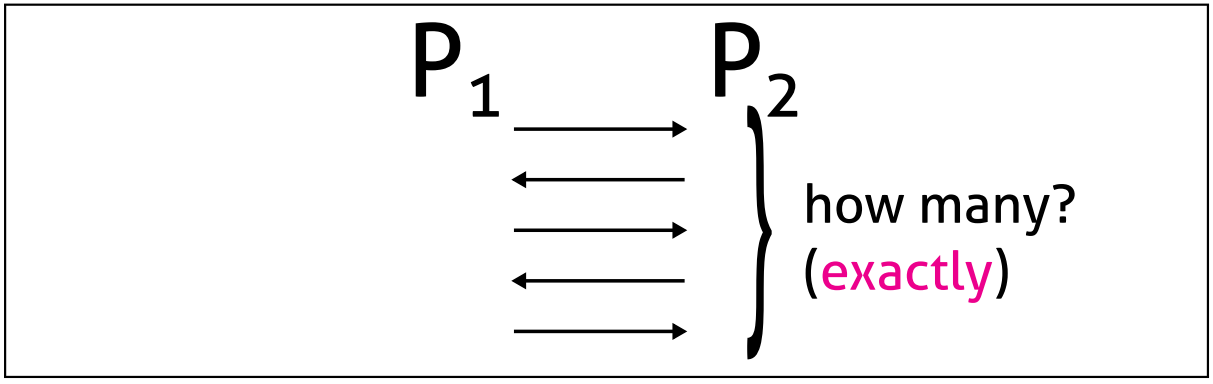
Asymptotic round complexity is well studied



- We have constant-round concurrent SPS 2PC under standard assumptions [GGJS12]
  (trapdoor permutations & collision-resistant hash)

What is Unknown about Concurrent SPS 2PC

Exact round complexity is not well studied



- In concurrent SPS, large constant (≥ 20) [GGJS12]
- In stand-alone, only 5 (optimal)! [KO04, ORS15]

The Problem We Studied

Innovative R&D by NTT

Can we get concurrently secure SPS 2PC with good exact round complexity?

Outline

Our Result

5-round concurrently secure SPS 2PC

(i.e., same round complexity as standalone case [KO04])

Assumption:

- 3-round non-malleable commitment w/ extractability property

  + standard crypo primitives (TDP and lossy encryption)

Note: Such non-malleable commitment exists under quasi-poly OWP [GRP16]

Remarks on Our Result

- ✔ Round complexity can be decreased to 4
  if only one party gets output
- ✔ Assumptions can be weakened to poly-hard ones
  if round complexity is increased to 7
- ✘ We don't know whether 5 is optimal

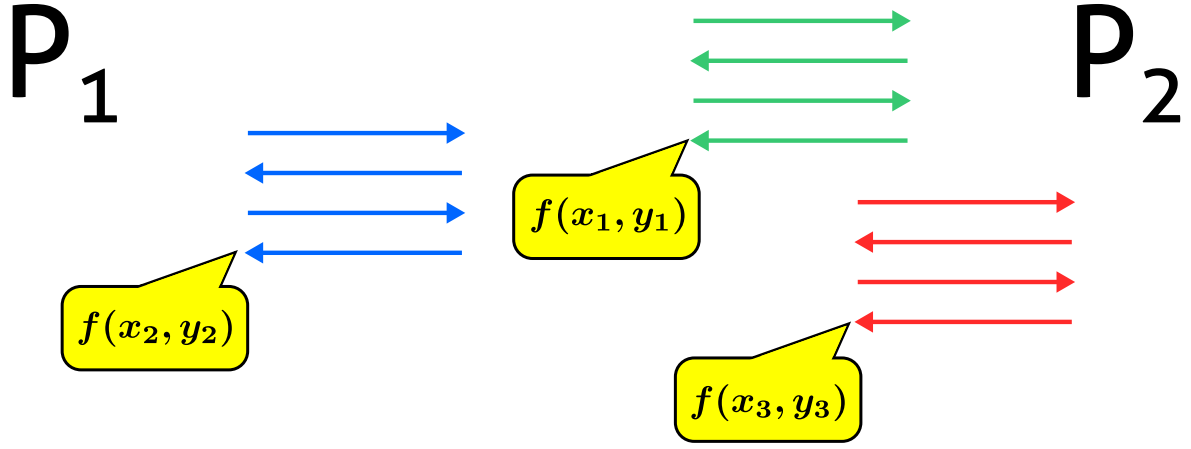
Outline

Bad News ☹: Our 2PC is Quite Complex

- **We carefully combine following primitives:**
  - garbled circuit
  - trapdoor permutation
  - 4-round ZK argument by Feige & Shamir [FS90]
  - ZAP
  - lossy encryption
  - symmetric-key encryption
  - MAC
  - non-interactive commitment
  - 3-round extractable commitment
  - 3-round non-malleable commitment
  - equivocal commitment by Katz & Ostrovsky [KO04]

So, Let's Focus on Simple Setting

- In this talk, we focus on the following setting
  - **Only one party gets output**
    - Add 1 round if both parties get outputs
  - **Each party has fixed role**
    - Add non-malleable com if roles are interchangeable



Overall Approach

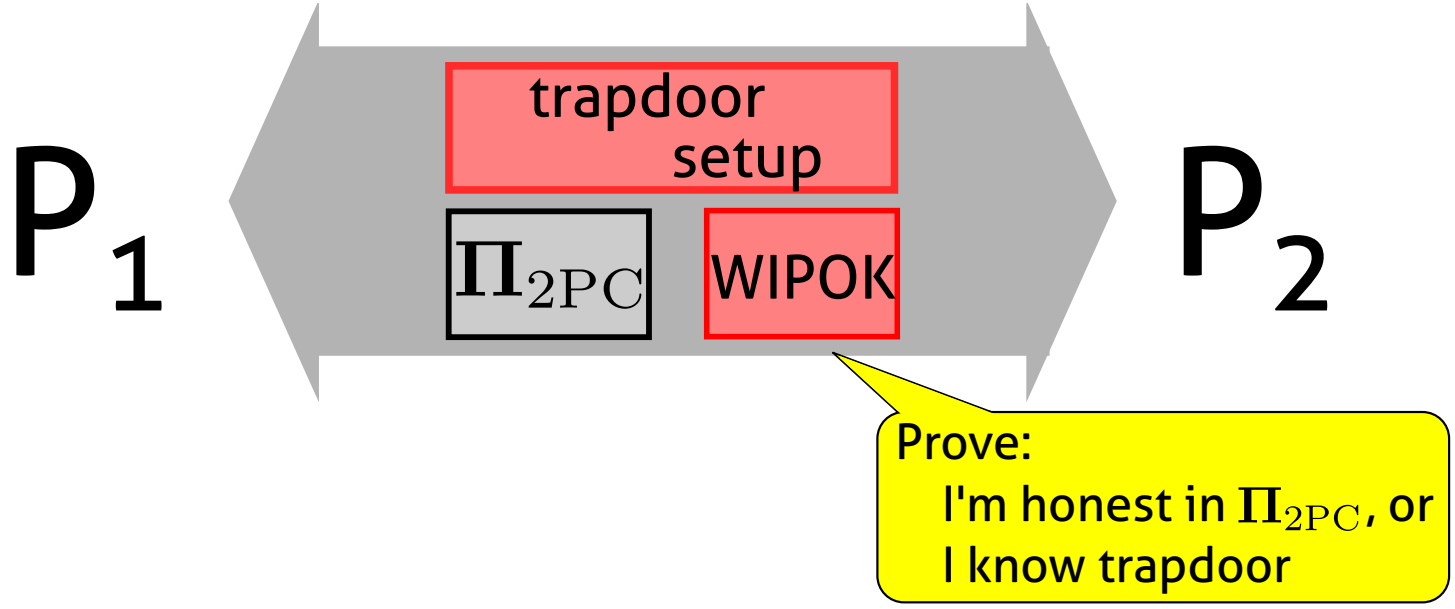
- **We already have:**
  1. 4-round 2PC protocol in stand-alone setting [KO04]
  2. compiler from stand-alone 2PC to concurrent SPS 2PC [GGJS12]

Let's combine them!

Concurrent SPS Compiler of [GGJS12] (1/3)

Compiler & simulator are simple:

Compiler: Add trapdoor setup phase & WI proofs



Simulator: Extract trapdoor by brute force & use it in WI proof

Concurrent SPS Compiler of [GGJS12] (2/3)

Showing indistinguishability is hard:

Real:

IND?

× Naive reduction run in super-poly time when emulating simulator internally

Ideal: Simulator obtain trapdoor in super-poly time

Concurrent SPS Compiler of [GGJS12] (3/3)

Key idea by [GGJS12]: Let's consider poly-time hybrid!

Real:

Hybrid: Simulator obtain trapdoor in poly time via rewinding extraction

Ideal: Simulator obtain trapdoor in super-poly time

Concurrent SPS Compiler of [GGJS12] (3/3)

Key idea by [GGJS12]: Let's consider poly-time hybrid!

Real:

Hybrid: Simulator obtain trapdoor in poly time via rewinding extraction

IND

✓ Only difference is extraction (brute-force v.s. rewinding)

Ideal: Simulator obtain trapdoor in super-poly time

Concurrent SPS Compiler of [GGJS12] (3/3)

Key idea by [GGJS12]: Let's consider poly-time hybrid!

Real:

IND

✓ Reduction works because both are poly-time

Hybrid: Simulator obtain trapdoor in poly time via rewinding extraction

IND

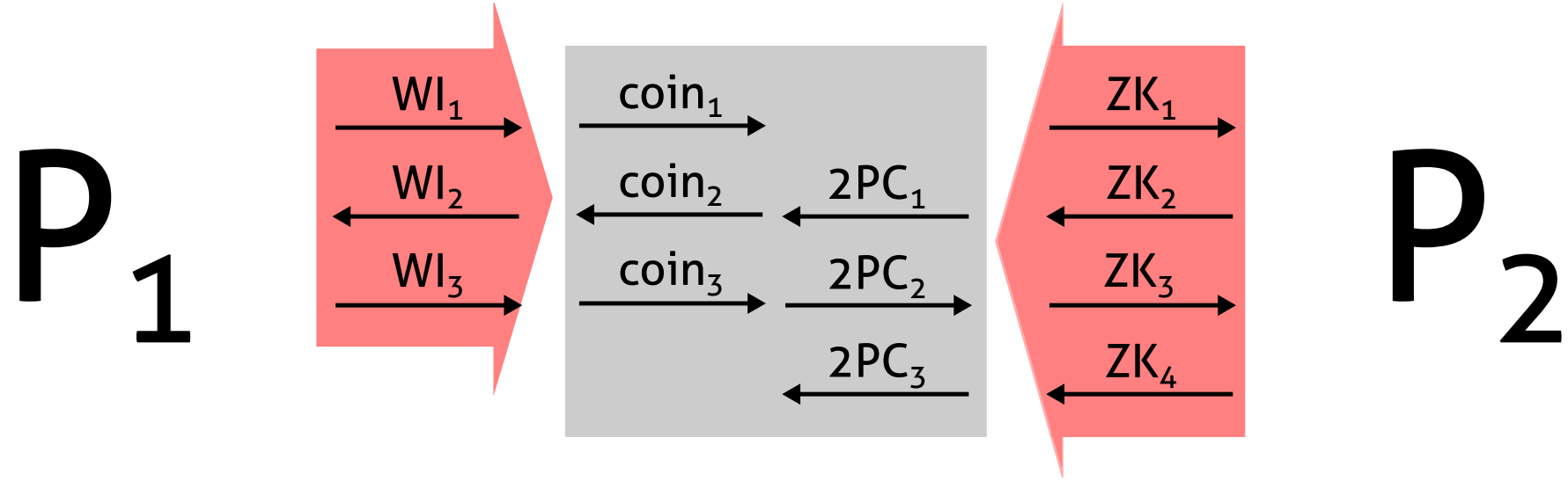
✓ Only difference is extraction (brute-force v.s. rewinding)

Ideal: Simulator obtain trapdoor in super-poly time

4-round 2PC by [KO04] + Compiler by [GGJS12]

Designing super-poly-time simulator is easy:

KO protocol: semi-honest 2PC + coin-tossing + WIPOK/ZKAOK



Simulator: extract witness from WIPOK/ZKAOK

4-round 2PC by [KO04] + Compiler by [GGJS12]

Showing indistinguishability is hard:

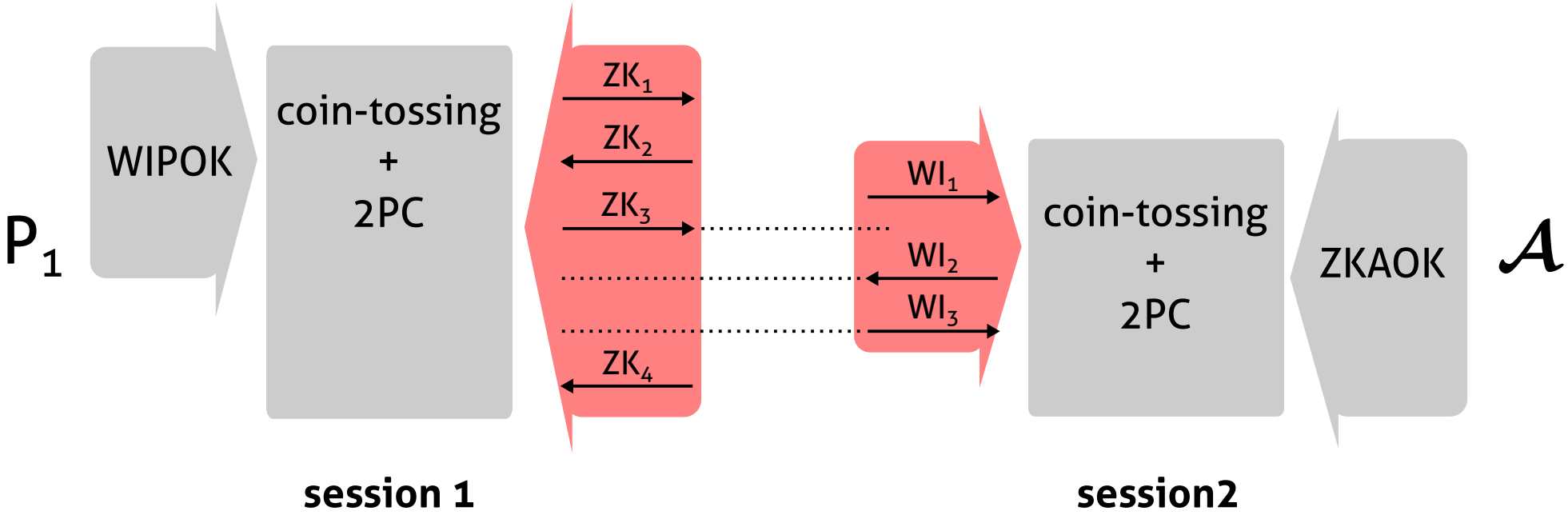
Real:

IND?

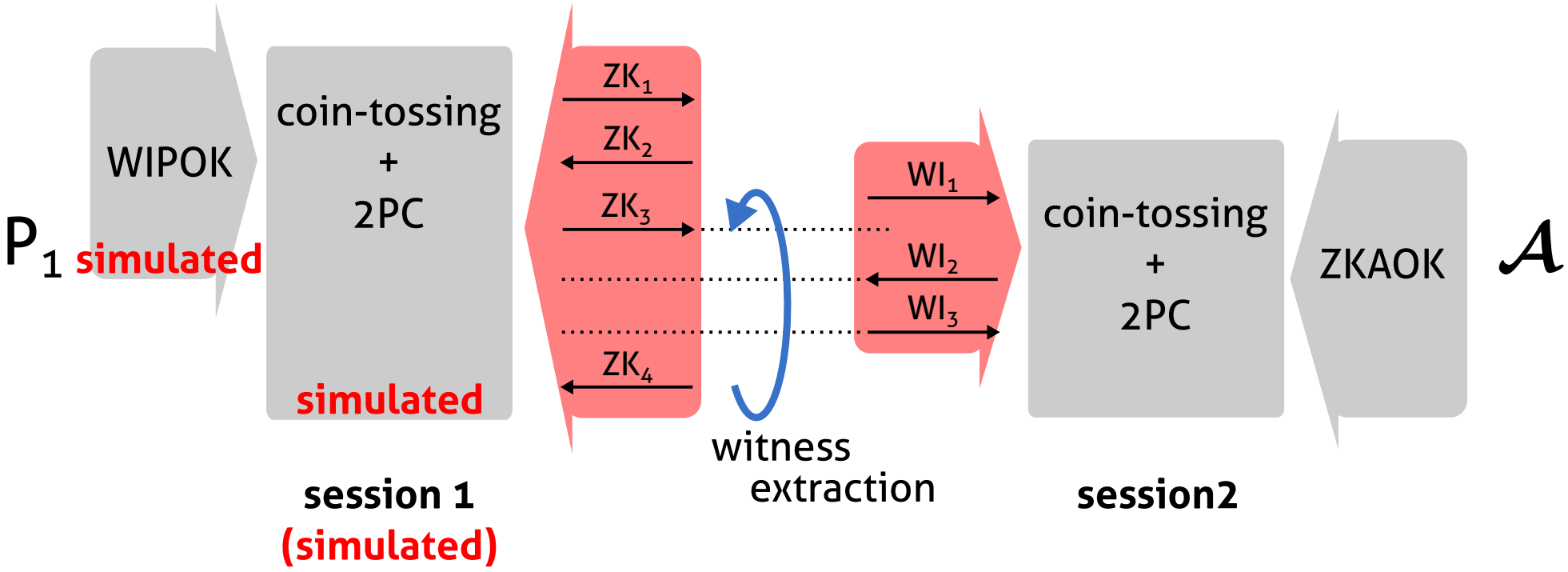
Hybrid: Simulator obtain trapdoor in poly time via rewinding extraction

Ideal: Simulator obtain trapdoor in super-poly time

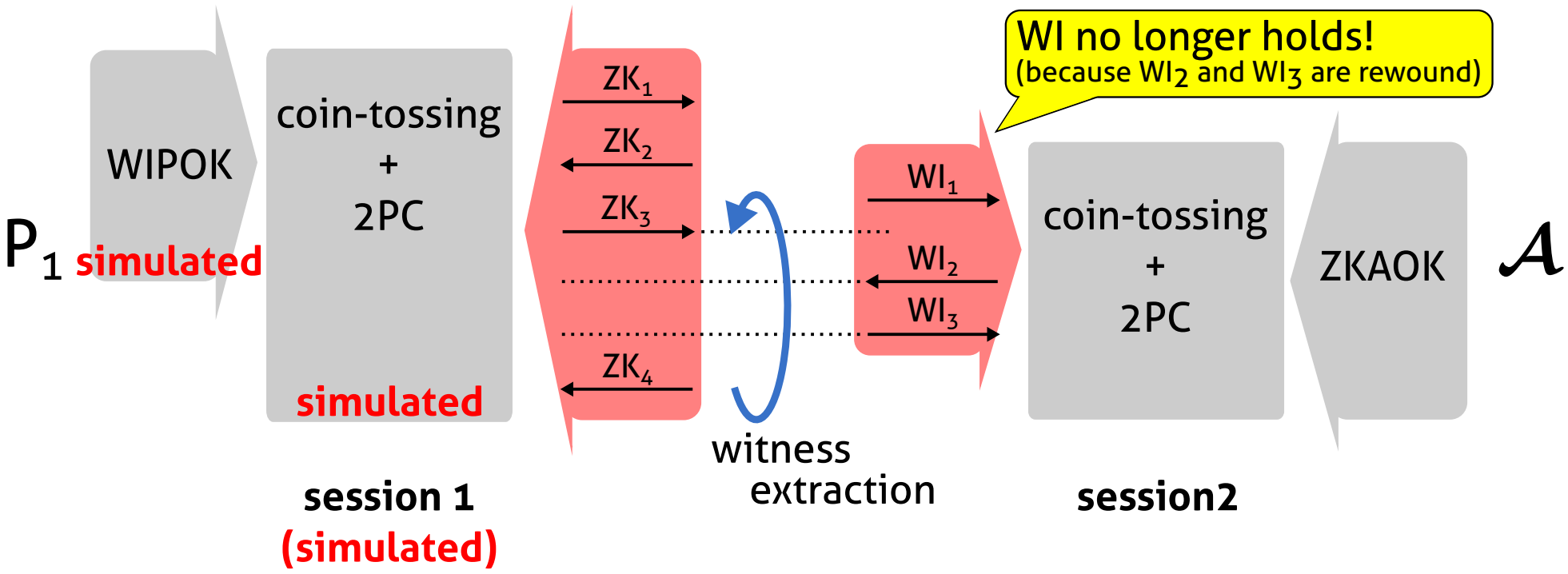
On IND between Real and Hybrid



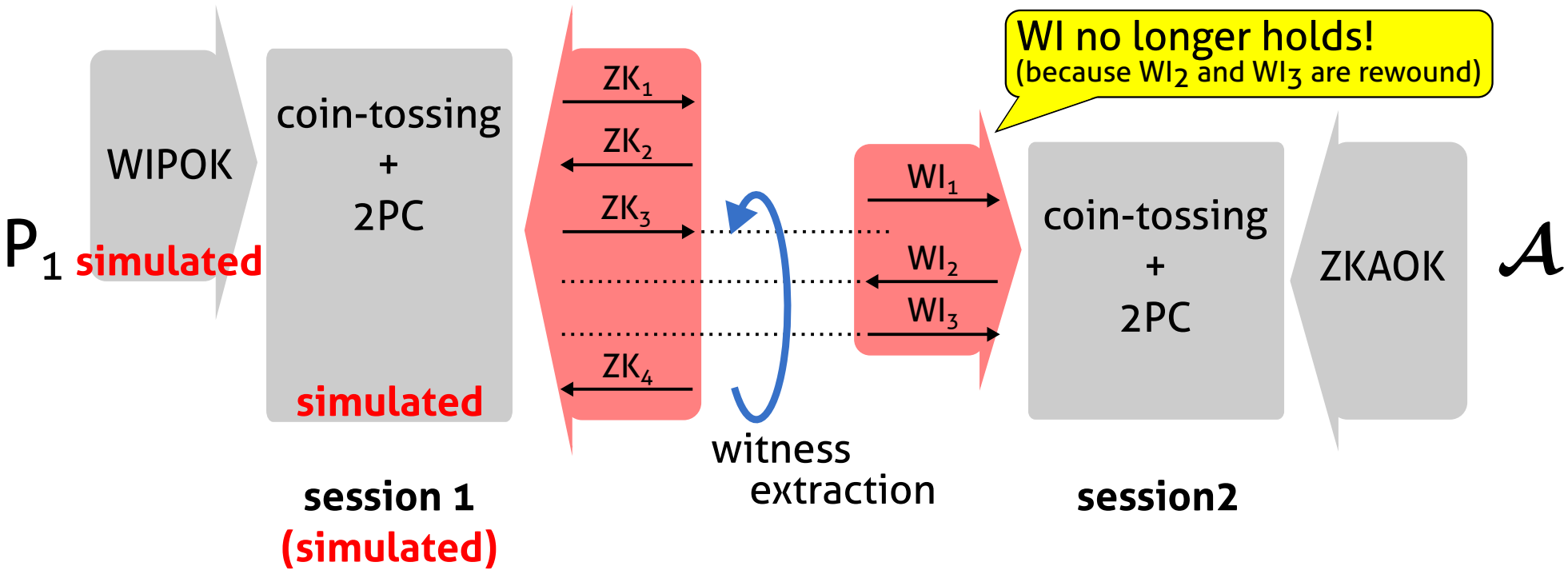
On IND between Real and Hybrid



On IND between Real and Hybrid



On IND between Real and Hybrid

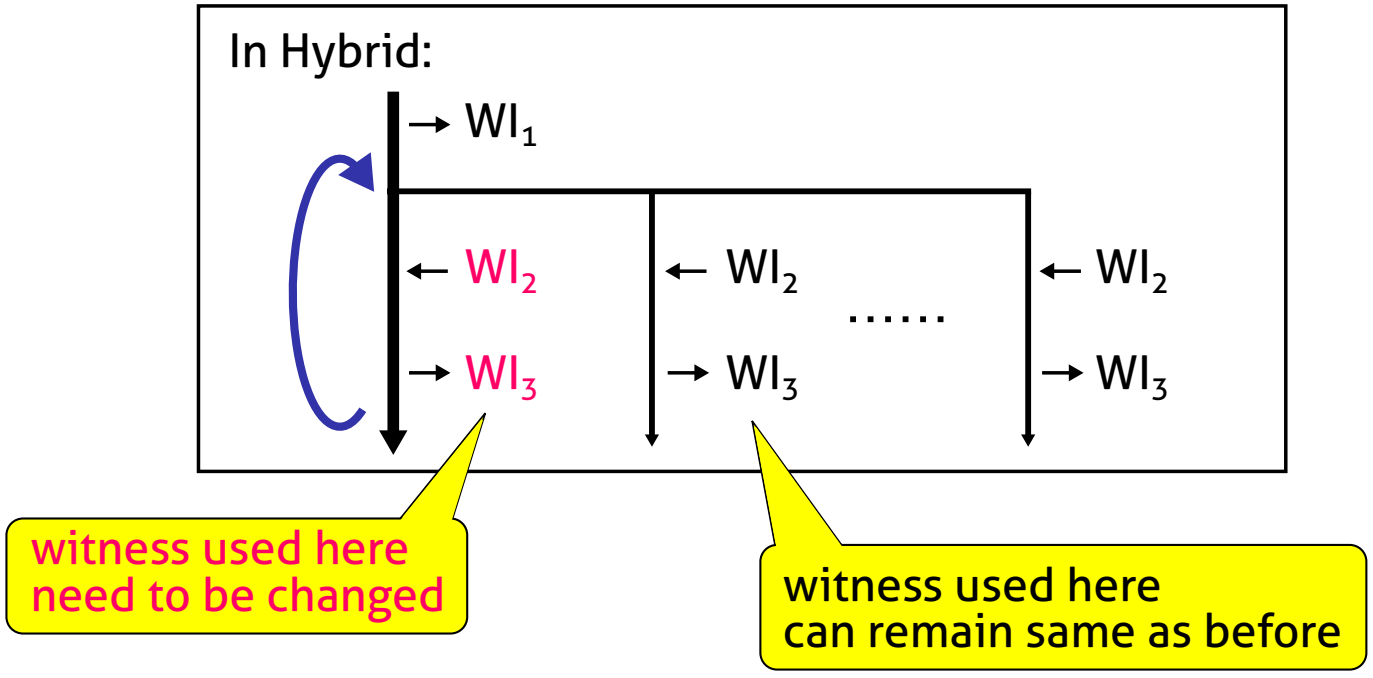


- **Wanted:** WIPOK that is "WI" under rewinding
  - Resettable WI is incompatible with POK...

Our Solution

Observation:

We need to change witness only on "main thread"!

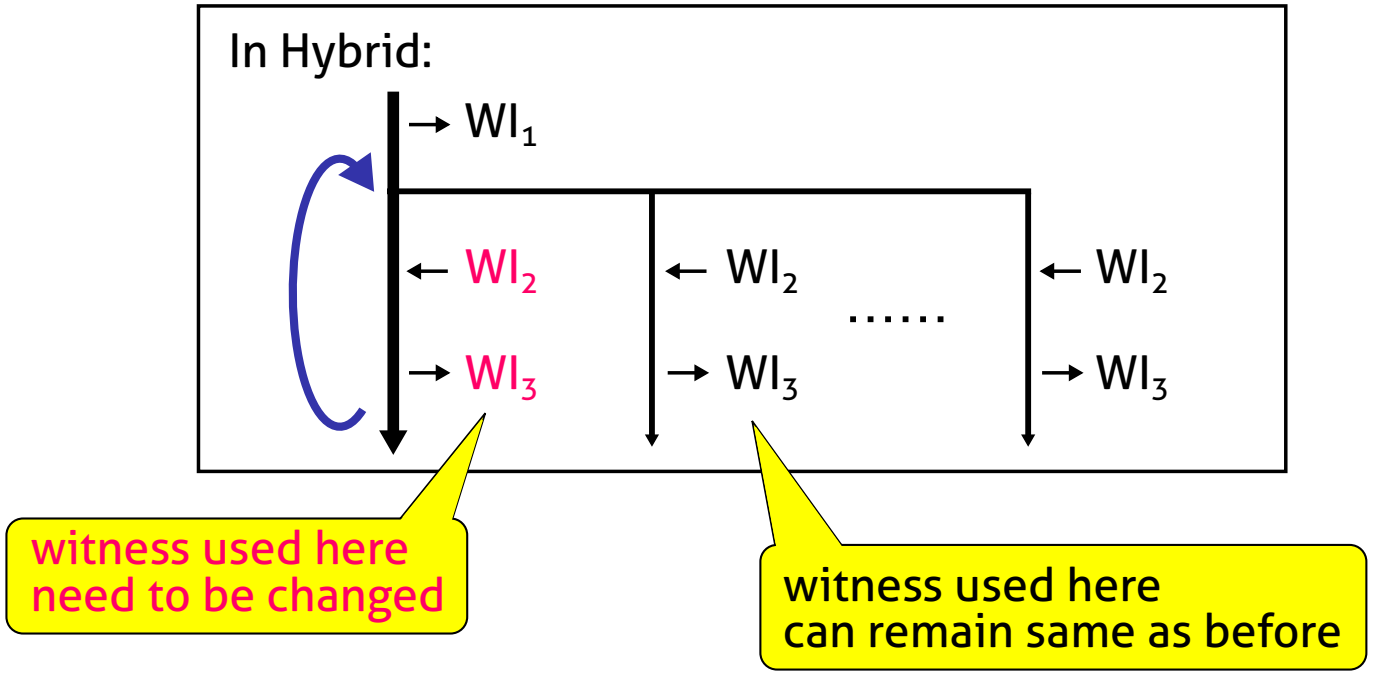


- This is because **Ideal** has only main thread
  - We use rewinding only in **Hybrid**

Our Solution

Observation:

We need to change witness only on "main thread"!



- By combining ZAP and extractable commitment, we obtain WIPOK that is WI in above setting

Other Technicalities

1. IND between Ideal and Hybrid:
   - Not trivial
     (Rewinding and brute-force can extract different values)
     $\Rightarrow$ We use lossy encryption to solve the problem
2. Interchangeable role
   - We use non-malleable commitment and statistically secure primitives in standard way [BPS06]

Summary

Our Result:

5-round concurrently secure SPS 2PC

(i.e., same round complexity as standalone case [KO04])

Assumption:

- 3-round non-malleable commitment w/ extractability property

  + standard crypo primitives (TDP and lossy encryption)

Note: Such non-malleable commitment exists under quasi-poly OWP [GRP16]

Appendix