Probabilistic Termination and Composability of Cryptographic Protocols*

Juan A. Garay
Texas A&M University
garay@cse.tamu.edu

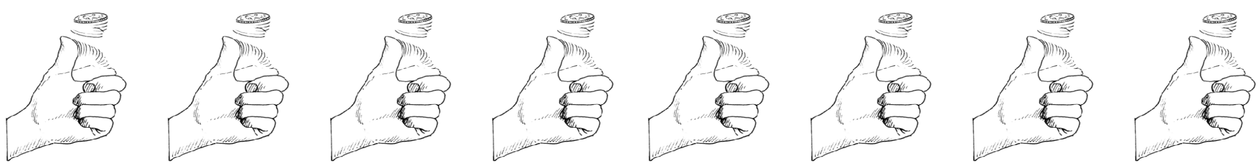
Joint work with Ran Cohen (MIT & NEU), Sandro Coretti (IOHK) and Vassilis Zikas (U. of Edinburgh & IOHK)

* Slides by Ran Cohen

Motivating Example

Coin flipping

- Stand-alone: Pr(*heads*) = ½
  Expected no. of coin tosses for *heads* outcome?
  2
- Flipping in parallel *n* coins:



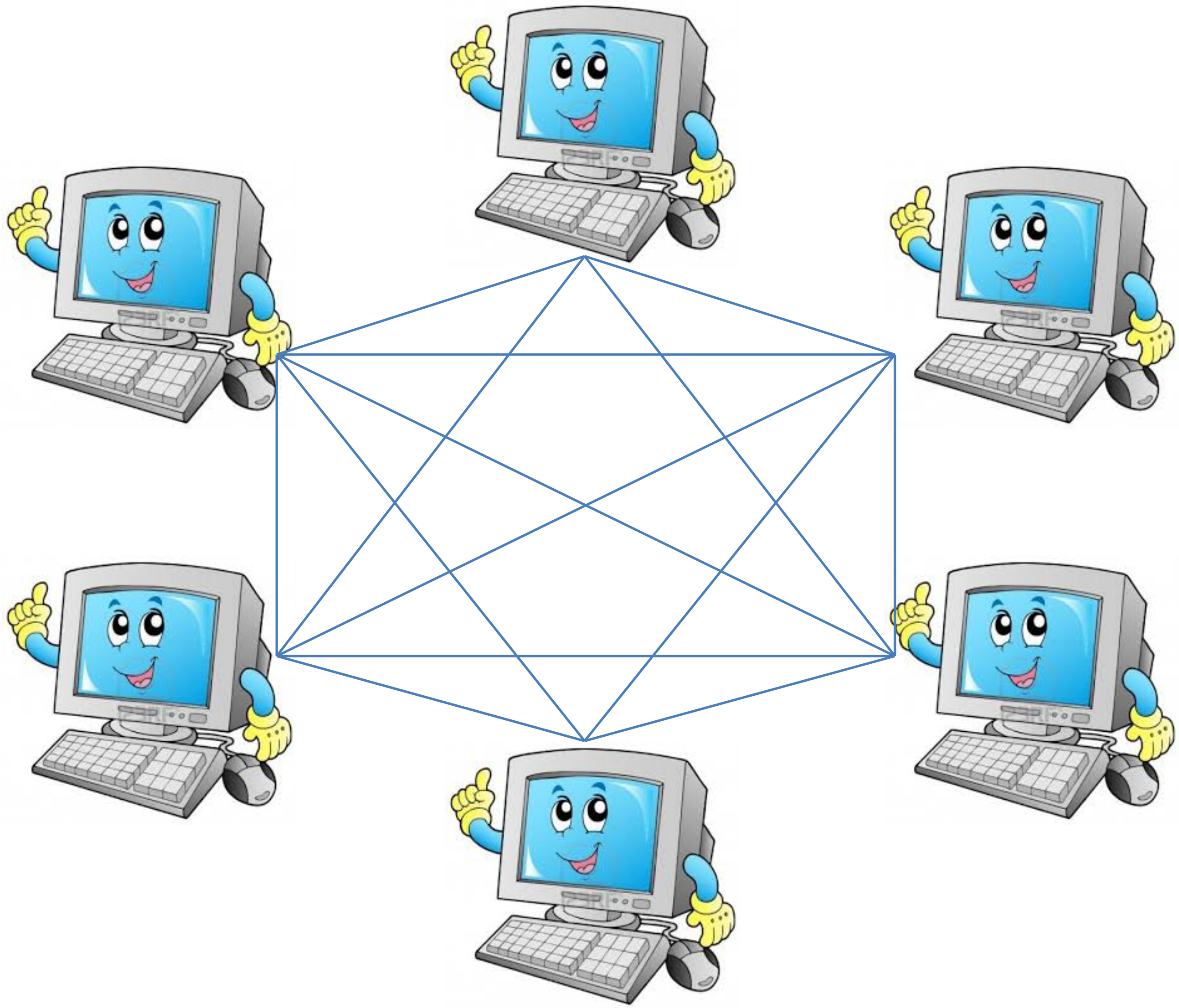
Expected no. of (parallel) coin tosses until all *heads*?

$O(\log n)$ $(\Theta(\log n))$

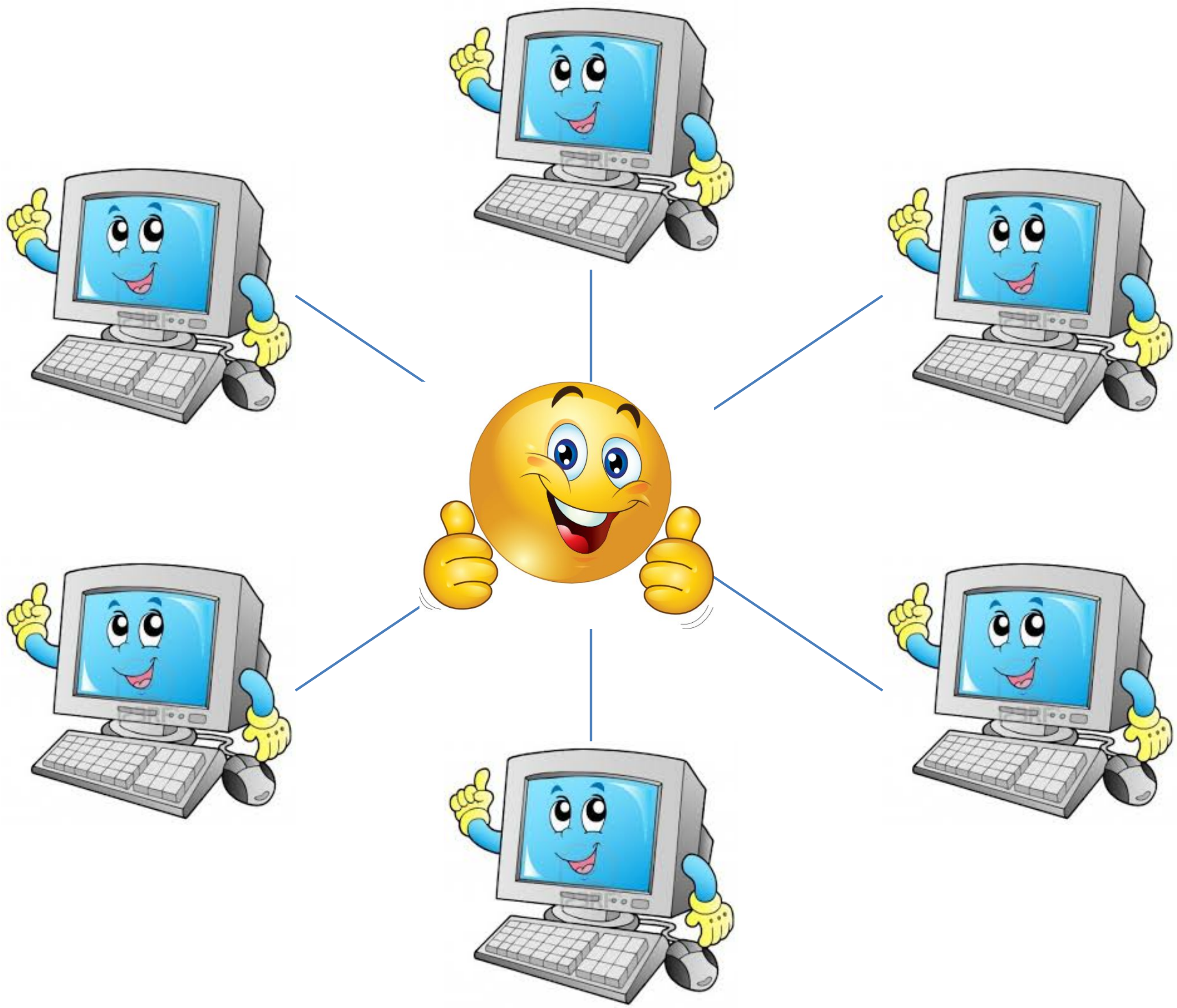
Motivation Example (2)

Fact: The mathematical expectation of the maximum of n random variables does not necessarily equal the maximum of their expectations [BE’03,Eis’08]

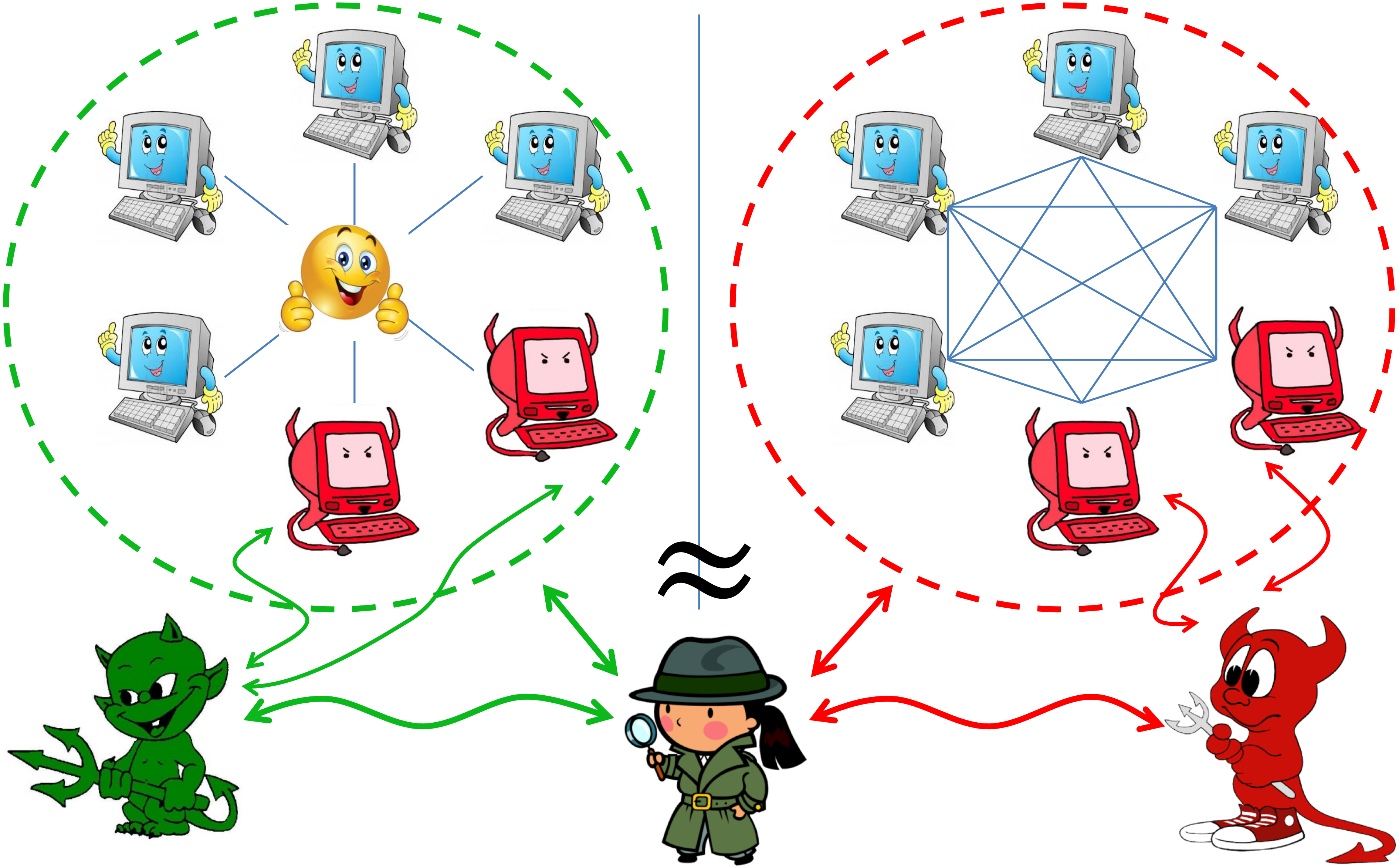
Secure Multiparty Computation (MPC)



Ideal World

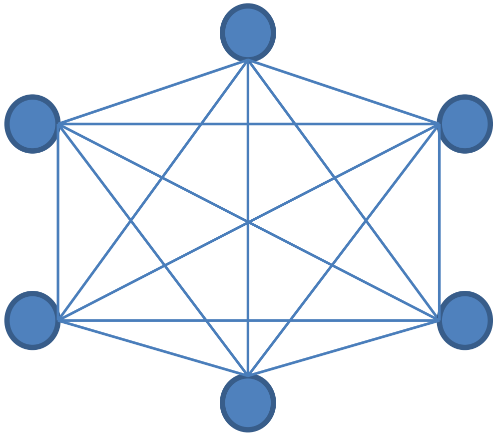


Simulation-based Security



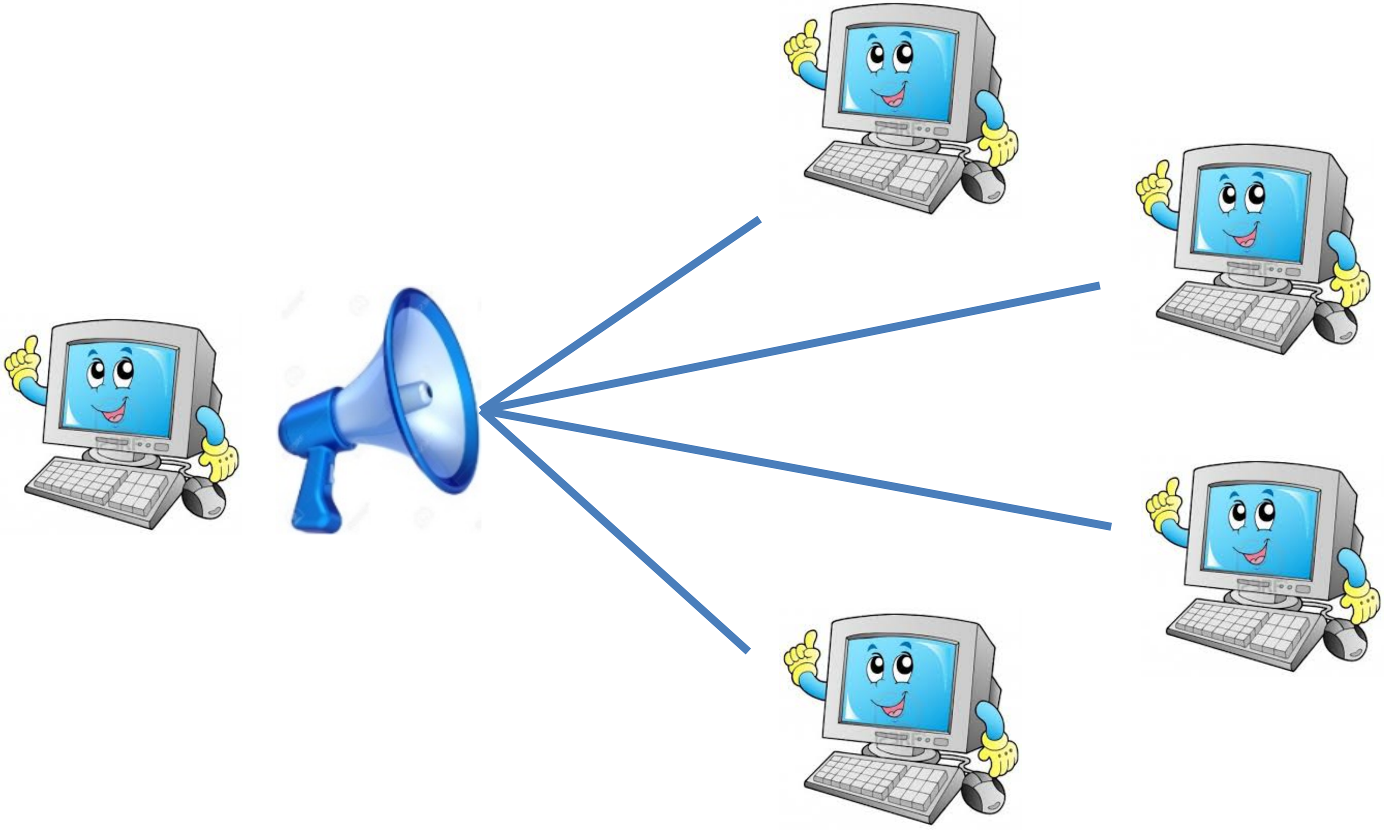
Communication Model

- Point-to-point model
  - Secure (private) channels between the parties (*Secure Message Transmission*)



- Broadcast model
  - Additional *broadcast channel*

Broadcast



Instantiating Broadcast Channel

Broadcast

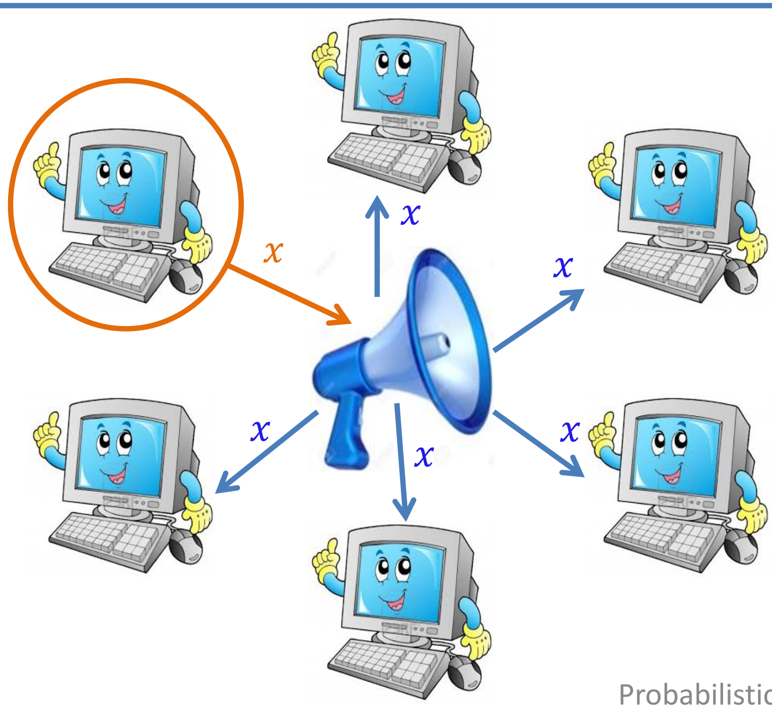
Sender with input x

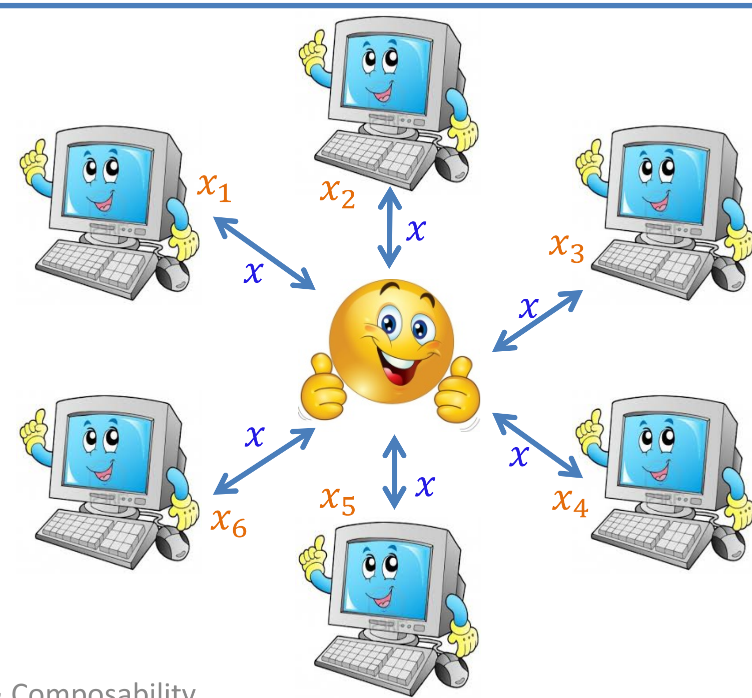
- **Agreement**: all honest parties output the same value
- **Validity**: if the sender is honest, the common output is x

Byzantine agreement

Each P_i has input x_i

- **Agreement**: all honest parties output the same value
- **Validity**: if all honest parties have the same input x, the common output is x





Instantiating Broadcast Channel

Broadcast

Sender with input x

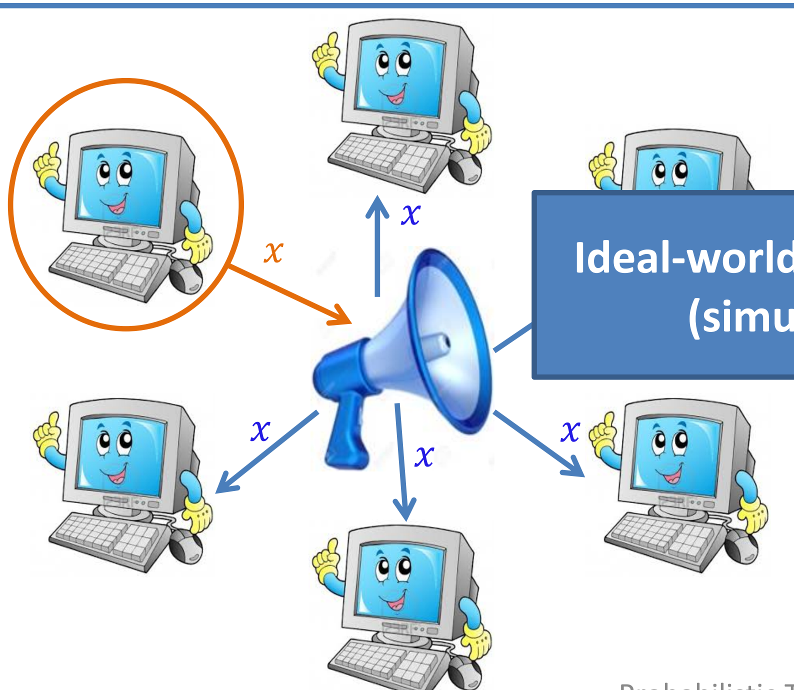
- **Agreement**: all hor output the same va
- **Validity**: if the sender is honest, the common output is x

Real-world security definition (property-based)

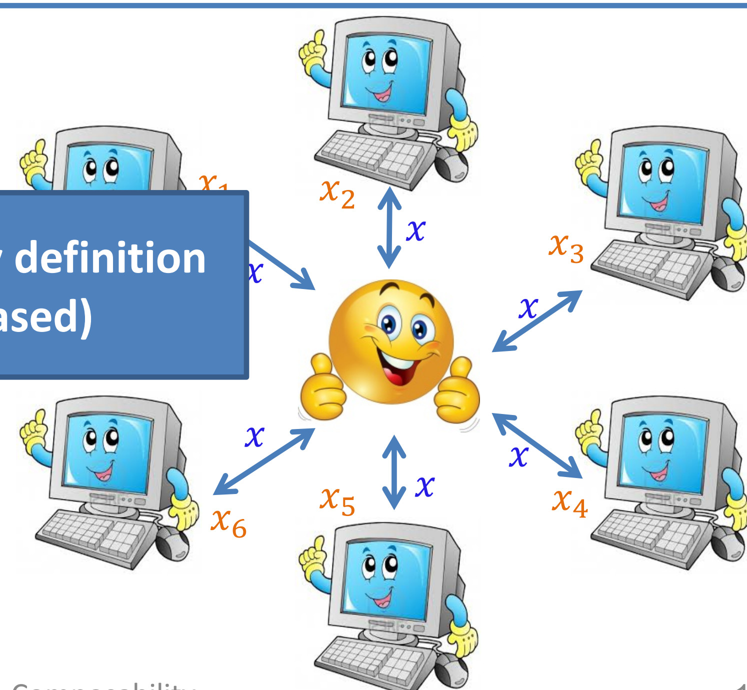
Byzantine agreement

input x_i

- nt: all honest parties e same value
- **Validity**: if all honest parties have the same input x, the common output is x

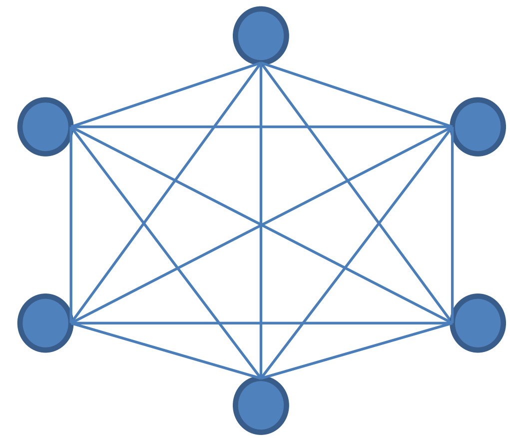


Ideal-world security definition (simulation-based)



Communication Model

- Point-to-point model
  - Secure (private) channels between the parties (*Secure Message Transmission*)



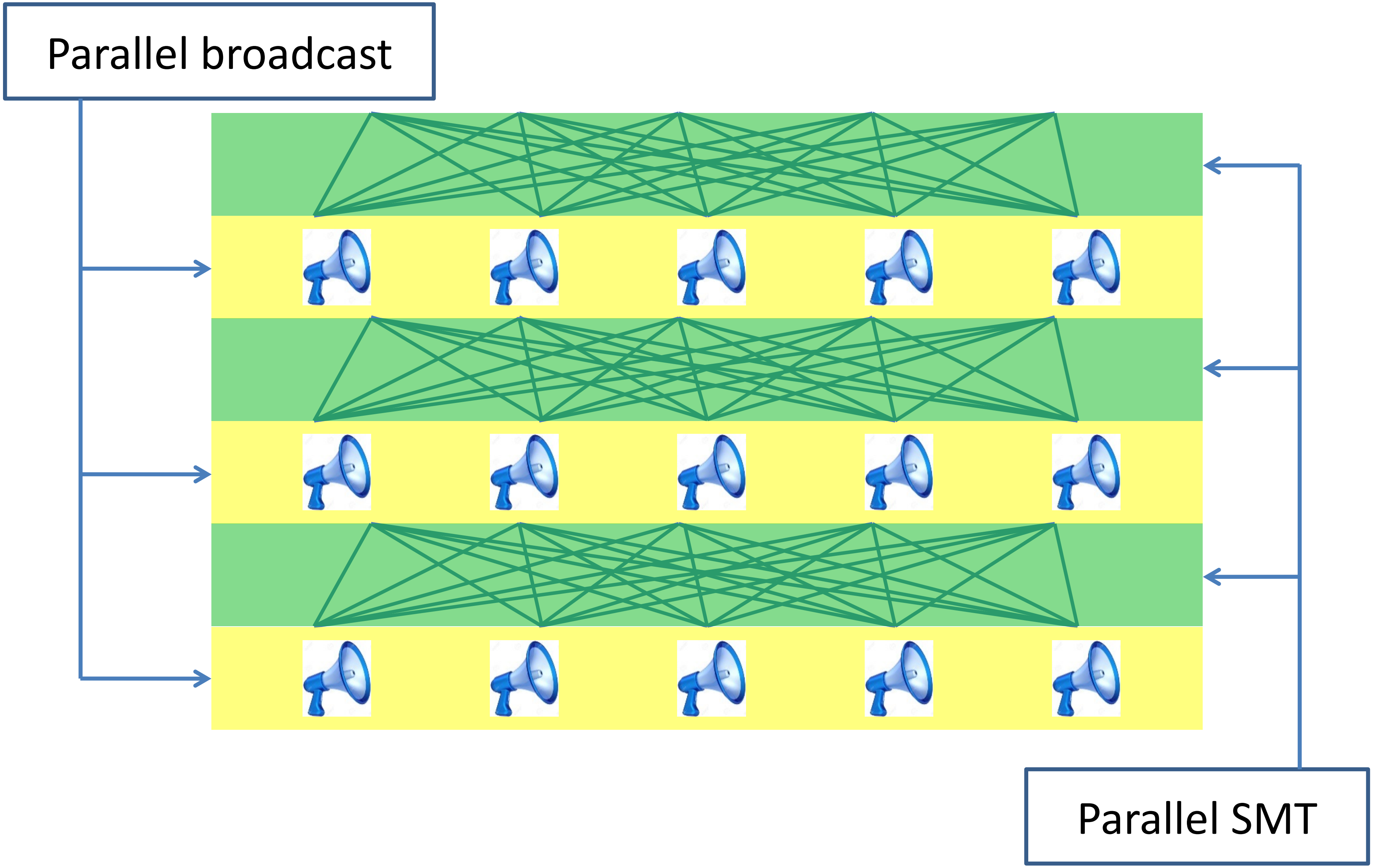
- Broadcast model
  - Additional *broadcast channel*

- Synchronous communication
  - Bounded delay
  - Global clock
  - Protocol proceeds in rounds
  - Guaranteed termination

Broadcast and MPC: Love-Hate Relationship

The Love

Protocols with Broadcast

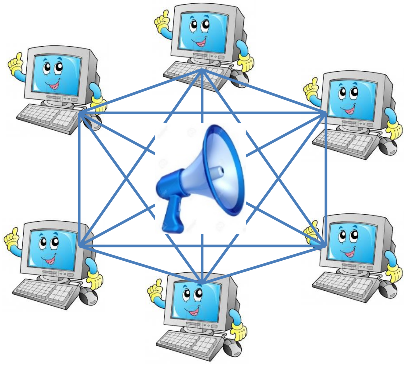


Broadcast is Good for MPC

Everything computable can be ***securely*** computed

Every function can be securely computed with guaranteed output delivery assuming honest majority

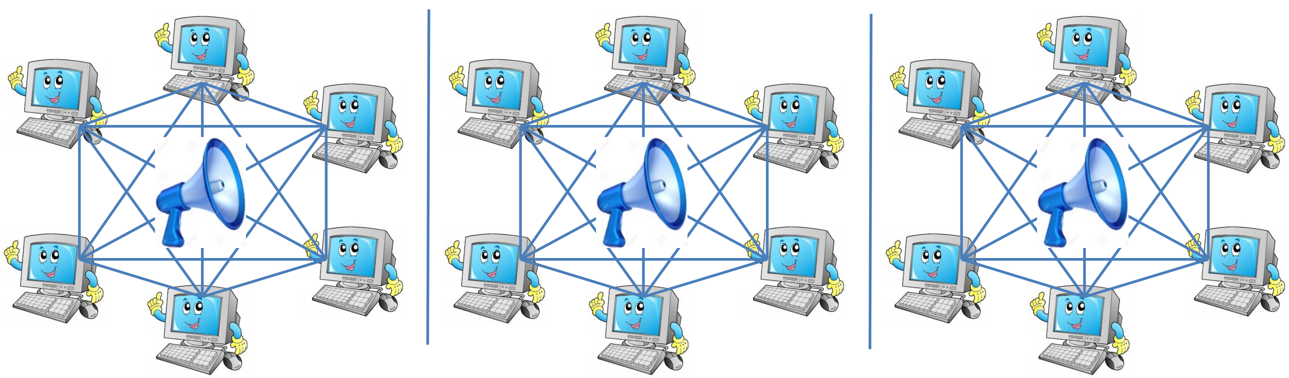
- $O(\text{depth})$ rounds (info-theoretic)
  [BenOr-Goldwasser-Wigderson'88, Chaum-Crépeau-Damgård'88, Rabin-BenOr'89]
- $O(1)$ rounds (OWF)
  [Beaver-Micali-Rogaway'90, Damgård-Ishai'05]
- 2 rounds (iO, Threshold-PKI, FHE, NIZK)
  [Garg-Polychroniadu'15, Gordon-Liu-Shi'15, Cohen-shelat-Wichs'18, Benhamouda-Lin'19, Garg-Srinivasan'19, …]



Broadcast is *Very* Good for MPC

If r-round π is secure under parallel composition

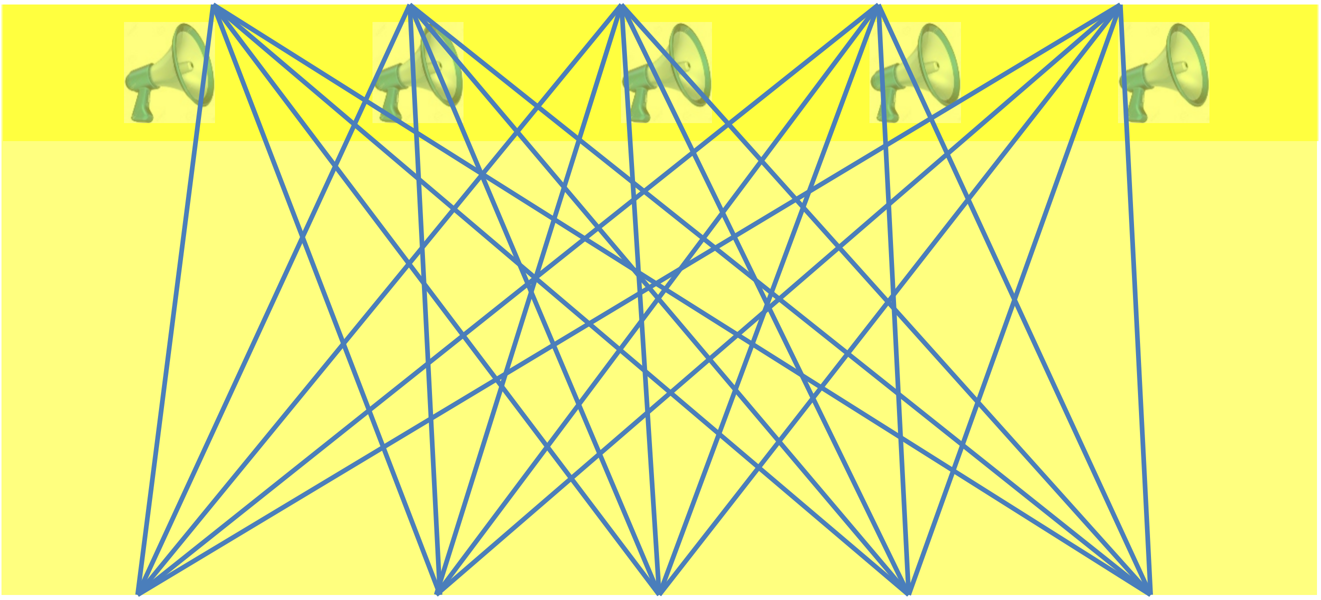
$\Rightarrow$ poly-many parallel executions of π in r rounds



The Hate:
What if Broadcast Doesn't Exist?

MPC Protocols w/o Broadcast

MPC Protocols w/o Broadcast



Protocols Implementing Broadcast

- Broadcast protocol $\Leftrightarrow t < \frac{n}{3}$ [Pease-Shostak-Lamport'80]
- Trusted setup (PKI) required for $t \geq \frac{n}{3}$ [Borcherdin'96]
- Some functions can be computed without setup [Cohen-Lindell'14, Cohen-Haitner-Omri-Rotem'16]

Round Complexity of Broadcast Protocols

- **LB1**: Deterministic protocols require $\Omega(n)\ (t+1)$ rounds
  [Fischer-Lynch'82, Dolev-Strong'83, Dolev-Reischuk-Strong'90]

Deterministic Broadcast Protocols

The [DS82] broadcast protocol: Assumes PKI, tolerates arbitrary number of corruptions ($t < n$)

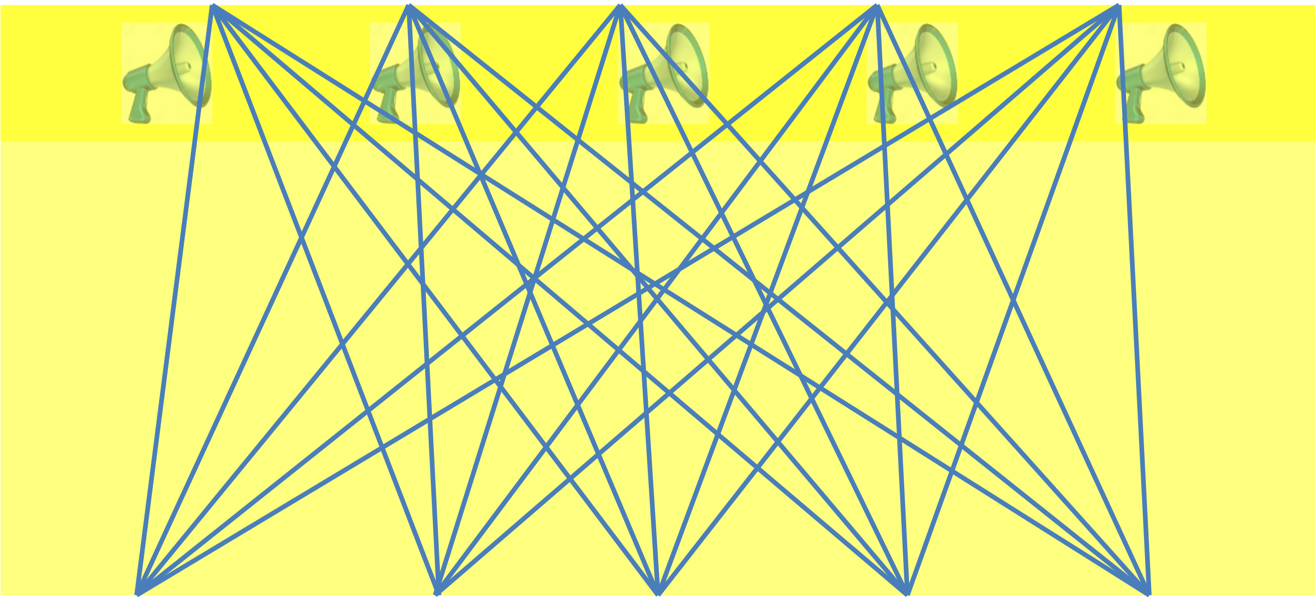
- Source signs its input value and sends to all parties
- r = 1,…,t+1:
  - If any value $v_i \in V = \{0,1\}$ has been *newly* added to a set of accepted values, sign it and send value and signatures to everybody
  - If a value/signatures message is received by any party containing valid signatures by at least r distinct players including the sender, then accept the value and update signatures
- If only one accepted value, then the party outputs that value; otherwise a default value

Deterministic Broadcast Protocols (2)

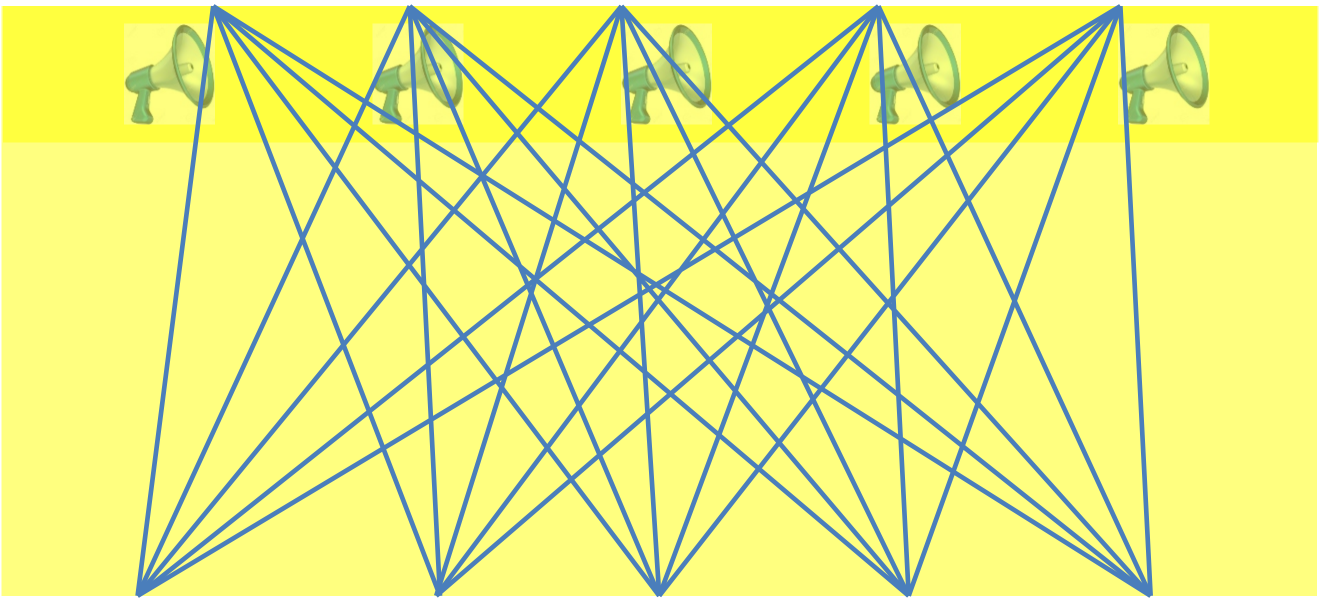
- Perfect and adaptive security for $t < n/3$ [BGP'89, GM'93, HZ'10]
- Deterministic Termination (DT) – single output round
- Compose nicely
- Require $O(n)$ rounds – this is inherent [FL'82, DS'82]

Deterministic Broadcast Protocols (2)

- Perfect and adaptive security for $t < n/3$ [BGP'89, GM'93, HZ'10]
- Deterministic Termination (DT) – single output round
- Compose nicely
- Require $O(n)$ rounds – this is inherent [FL'82, DS'82]



MPC Protocols w/o Broadcast



Round Complexity of Broadcast Protocols

- **LB1**: Deterministic protocols require $\Omega(n)$ $(t+1)$ rounds
  [Fischer-Lynch'82, Dolev-Strong'83, Dolev-Reischuk-Strong'90]
- **LB2**: Randomized r-round protocols fail w.p. $\frac{1}{c \cdot r^r}$
  [Chor-Merritt-Shmoys'85, Karlin-Yao'86]
- **UB:** Expected-constant rounds (guaranteed w/ polylog rounds)
  [BenOr'83, Rabin'83, Feldman-Micali'88, Fitzi-Garay'03, Katz-Koo'06]
  [Micali'17, Micali-Vaikuntanathan'17, Abraham-Devadas-Dolev-Nayak-Ren'18]
  [Abraham-Chan-Dolev-Nayak-Pass-Ren-Shi'19]

These protocols have ***probabilistic termination***

- Termination round not *a priori* known
- Non-simultaneous termination

Randomized Broadcast Protocols

Randomization can help [Ben-Or'83, Rabin'83]

Binary BA protocol [Feldman-Micali'88]

- Proceeds in phases until termination
- In each phase each party has an input bit
  - If all honest parties start the phase with the same bit, they terminate at the end of the phase
  - Otherwise, with probability $p > 0$ all honest parties agree on the same bit at the end of the phase (and terminate in the next phase)
  - With probability $1 - p$
    - No agreement at the end of the phase, or
    - the adversary makes some of the honest parties terminate; the remaining parties will terminate in the next phase

Randomized Broadcast Protocols (2)

- [FM’88] has *Probabilistic Termination* (PT):
  - Expected $O(1)$ rounds
  - No guaranteed termination: Statistical security (for PPT parties)
  - No simultaneous termination: Honest parties might terminate at different rounds [DRS’90]
  - All honest parties terminate in a constant window
- Extends to multi-valued BA [Turpin, Coan’84]
  - Two additional rounds
- Perfect security [Goldreich-Petrank’90]
  - Best of both worlds
- Variant for parallel broadcast [Ben-Or-El-Yaniv’03]

Composition of PT Protocols

Sequential composition

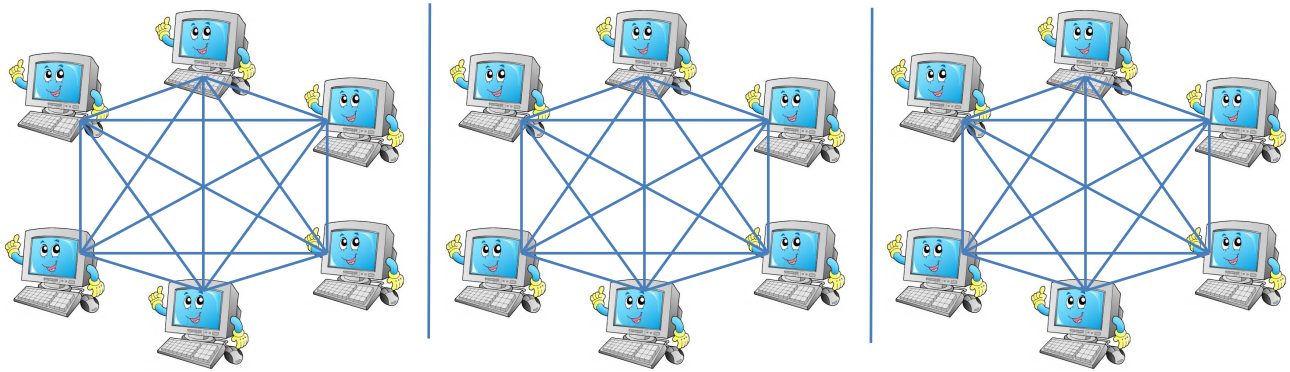
Simultaneous start

Non-simultaneous termination

⇒ **Non-simultaneous start**

Parallel composition

Naïve parallel composition is ***not* round preserving**



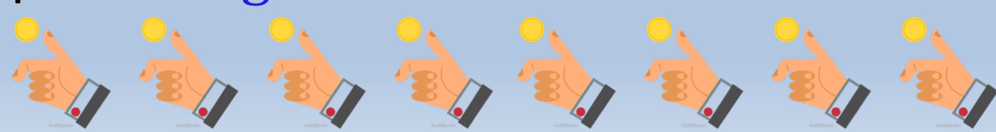
Naïve Parallel Composition

Protocol with *expected* $O(1)$ rounds (geometric distribution)
$\Rightarrow n$ parallel instances take *expected* $\Theta(\log n)$ rounds

<u>Example</u>: Coin flipping

- Stand-alone coin flip: $\Pr(heads) = 1/2$
  $\Rightarrow$ output is $heads$ in expected 2 rounds

- Flipping in parallel n coins, each coin until $heads$
  $\Rightarrow$ expected $\log n$ rounds



Broadcast Composition: Prior Work

- Sequential composition of m BA protocols in expected $O(m)$ rounds
  [Lindell-Lysyanskaya-Rabin'02]
- Parallel composition of m BA protocol in expected $O(1)$ rounds
  [BenOr-ElYaniv'03, Fitzi-Garay'03, Katz-Koo'06]
- All prior work use **property-based** definitions
- Security under **composition**?

What's missing?

Main challenge: How to simulate ***probabilistic termination***

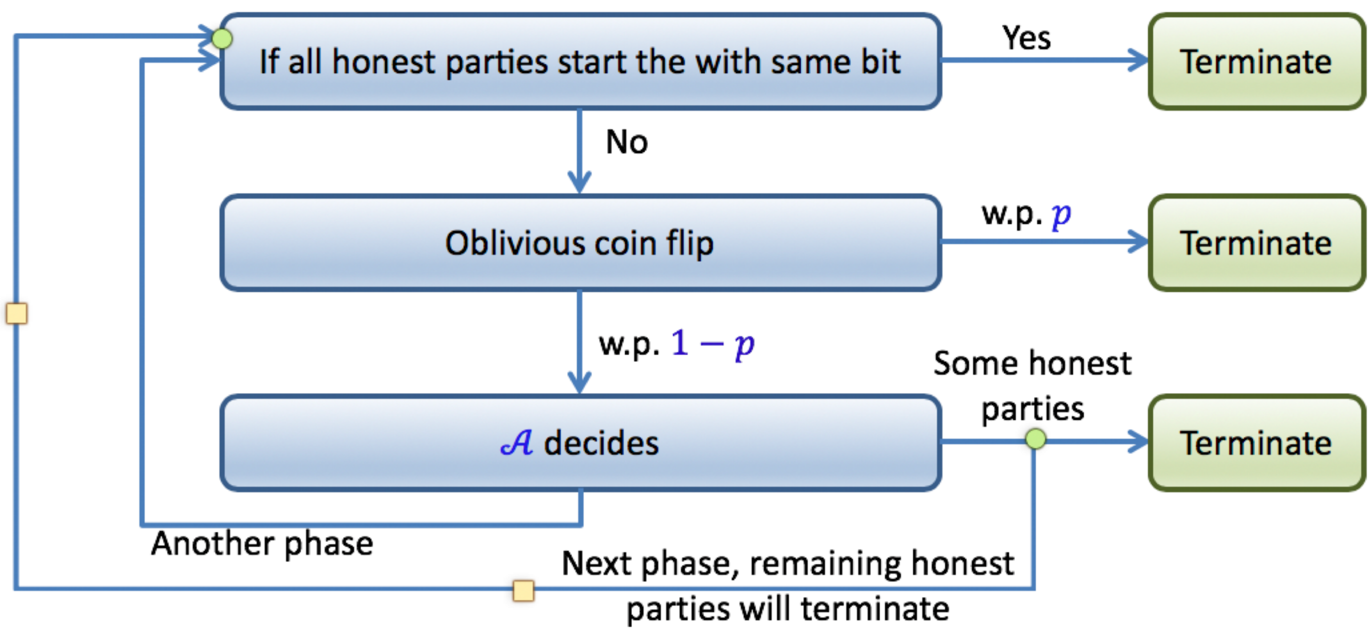
The Setting

- Secure channels (SMT = Secure Message Transmission)
- Synchronous communication [Katz-Maurer-Tackmann-Zikas'13]
- [KMTZ'13] model sync. **Deterministic-Termination** (DT) protocols in UC
  - Environment observes in which round the protocol terminates
  - Ideal functionality is parameterized by number of rounds
  - Parties continuously request output — receive it at last round
- PT protocols are very delicate — many subtle issues not captured by [KMTZ'13]

Randomized BA/Broadcast Protocol

[Feldman-Micali'88]

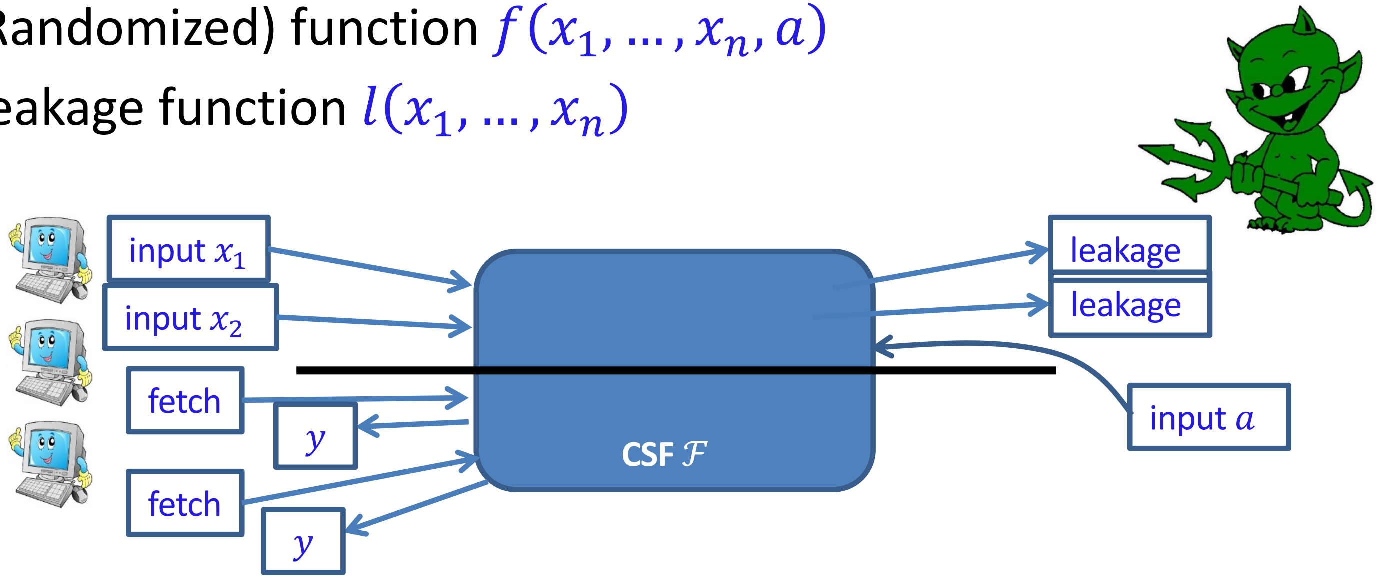
- Proceeds in phases until termination
- In each phase each party has an input bit



The Framework
Part I: Modeling Probabilistic Termination

Canonical Synchronous Functionality

- Separate the function from the round structure
- A CSF consists of input round and output round
- Parameterized by
  - (Randomized) function $f(x_1, \dots, x_n, a)$
  - Leakage function $l(x_1, \dots, x_n)$



CSF Examples

SFE: parties compute a function g

- $f(x_1, \dots, x_n, a) = g(x_1, \dots, x_n)$
- $l(x_1, \dots, x_n) = (|x_1|, \dots, |x_n|)$

Broadcast: P_i broadcasts x_i

- $f(x_1, \dots, x_n, a) = (x_i, \dots, x_i)$
- $l(x_1, \dots, x_n) = |x_i|$

parallel version

Byzantine Agreement:

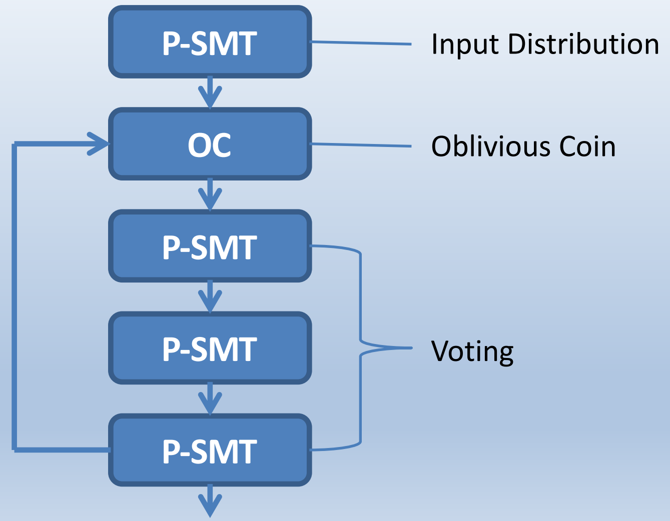
- $f(x_1, \dots, x_n, a) = \begin{cases} y \text{ if at least } n - t \text{ inputs are } y \\ a \text{ otherwise} \end{cases}$
- $l(x_1, \dots, x_n) = (x_1, \dots, x_n)$

Synchronous Normal Form (SNF)

SNF protocol:

- In each round exactly one ideal functionality is called ("stand-alone composition")
- All hybrids are (2-round) CSFs

Example: Protocol π_{RBA} (based on [FM'88])

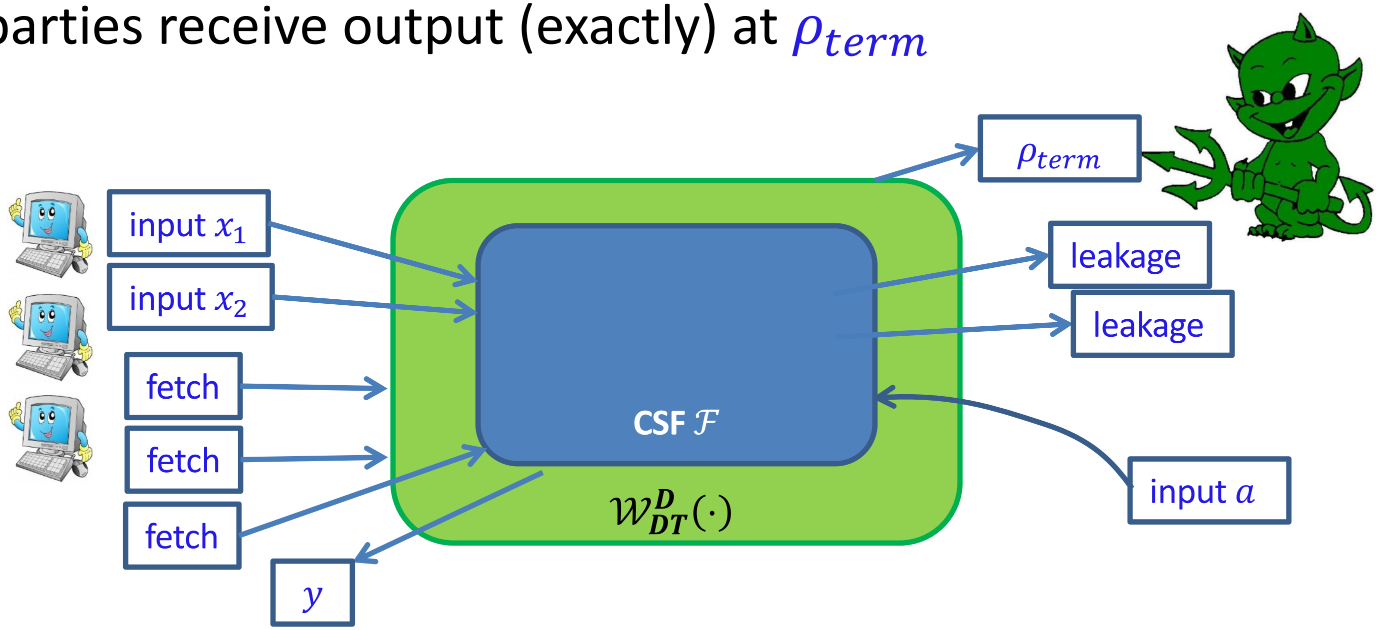


P-SMT = parallel SMT

Extending Rounds (Deterministic Termination)

- Many protocols require more than two rounds
- Wrap the CSFs with *round-extension wrappers*
  - Sample a termination round $\rho_{term} \leftarrow D$
  - All parties receive output (exactly) at ρ_{term}

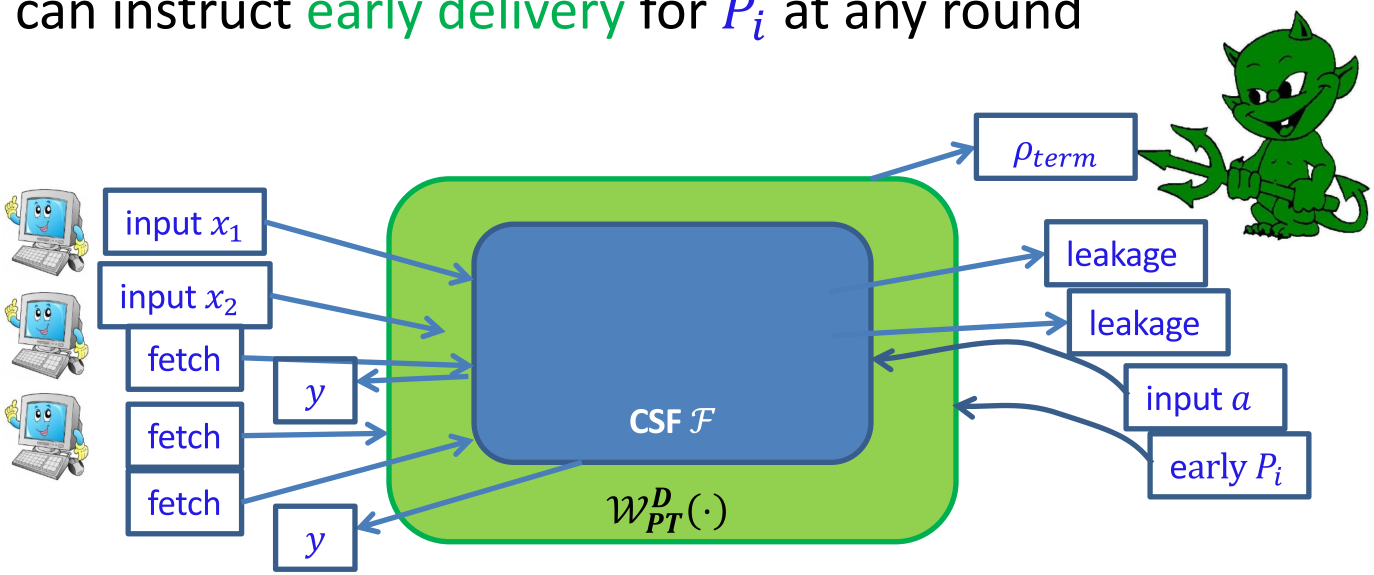
As in [KMTZ'13]



Extending Rounds (Probabilistic Termination)

Termination round is an upper bound

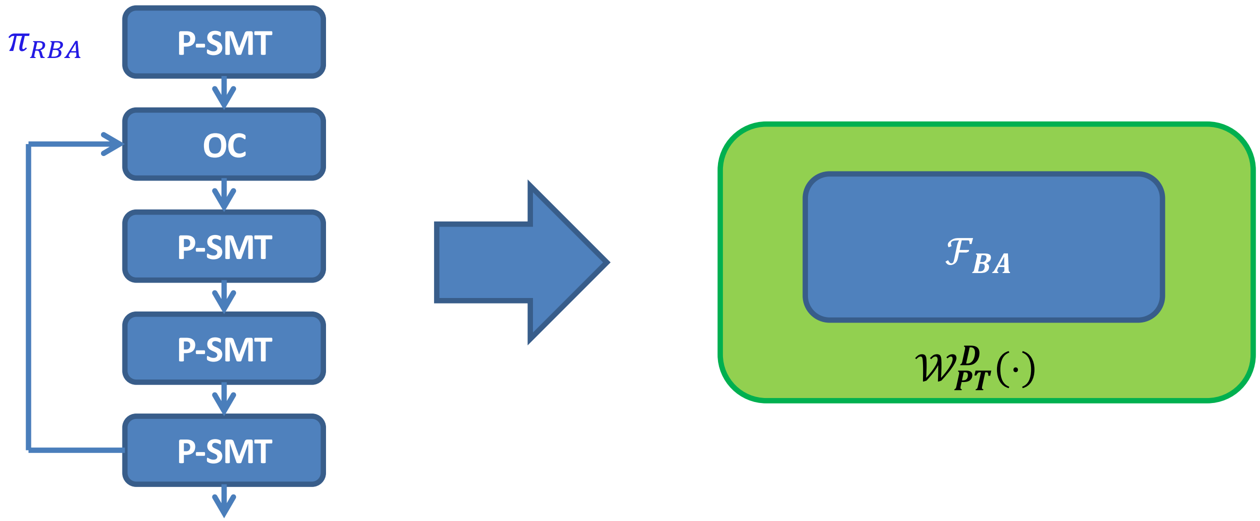
- Sample a termination round $\rho_{term} \leftarrow D$
- All parties receive output <u>by</u> ρ_{term}
- $\mathcal{A}$ can instruct early delivery for P_i at any round



Where Do We Stand?

Protocol π_{RBA} realizes $\mathcal{W}_{PT}^{D}(\mathcal{F}_{BA})$ in $(\mathcal{F}_{PSMT}, \mathcal{F}_{OC})$-hybrid model

assuming all parties start at the same round



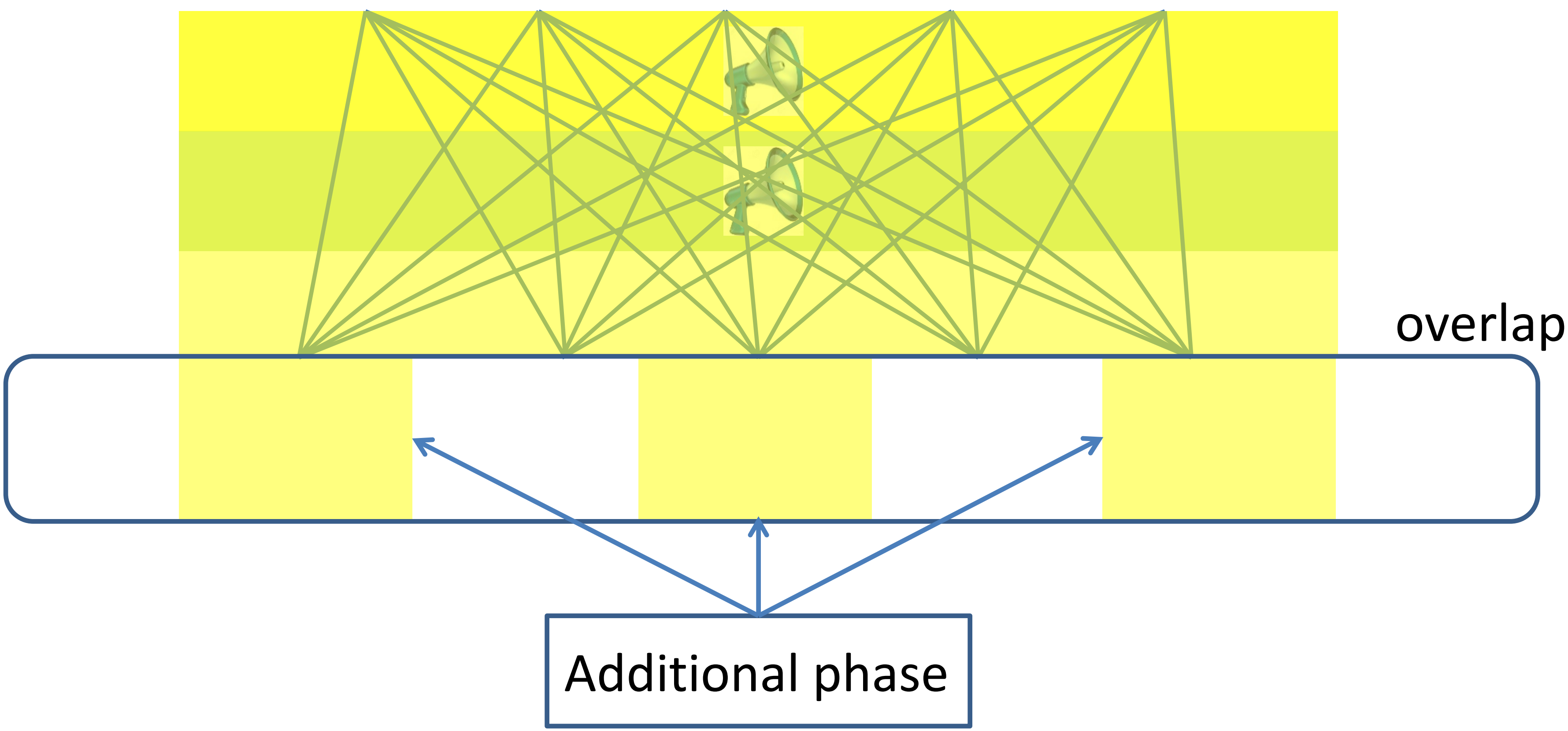
D: Geometric distribution with parameter *p* over the phases

The Framework
Part II: Non-Simultaneous Start
(Dealing with “slack”)

Problem: Sequential Composition

New execution starts **after all** parties finished previous one

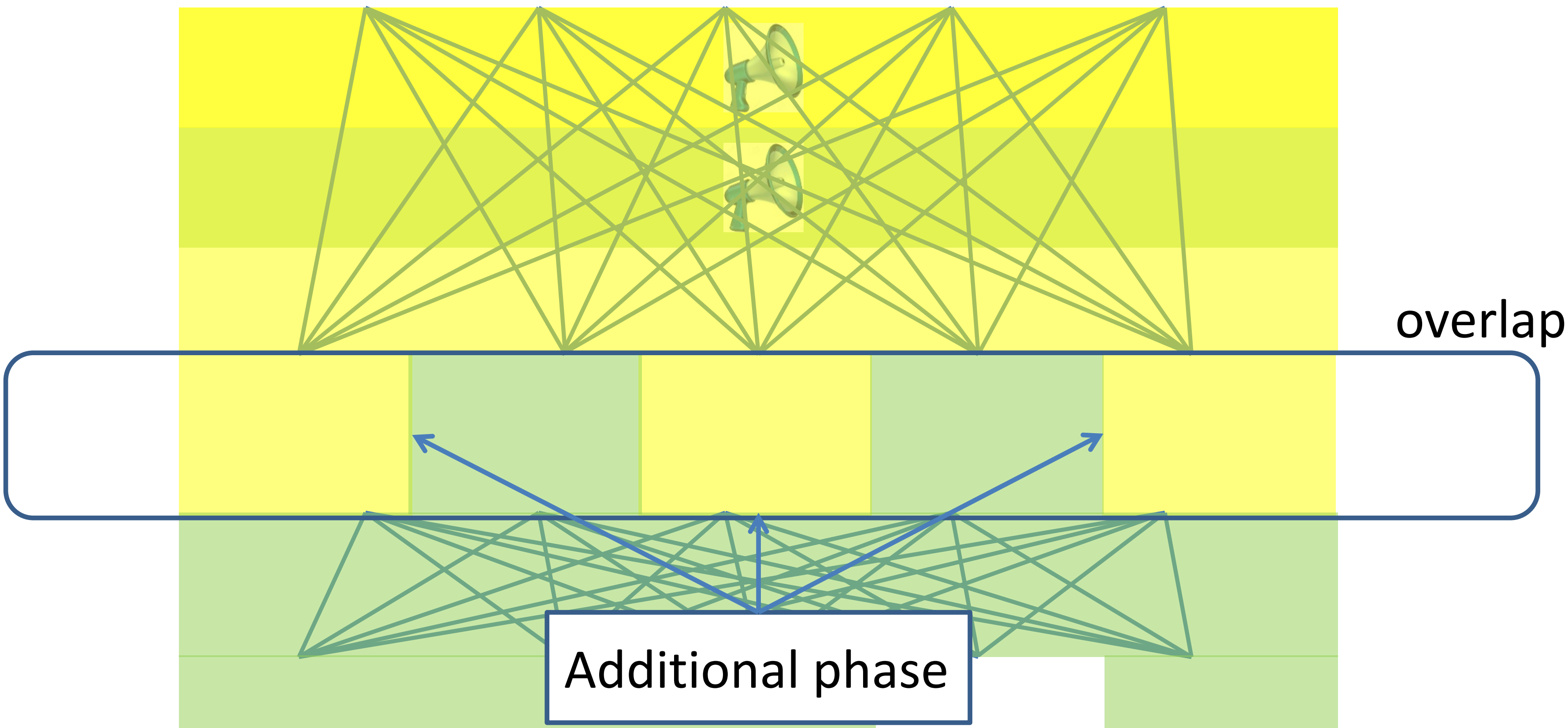
With PT protocols, fast parties start new execution **before** slow parties finished previous execution



Problem: Sequential Composition

New execution starts **after all** parties finished previous one

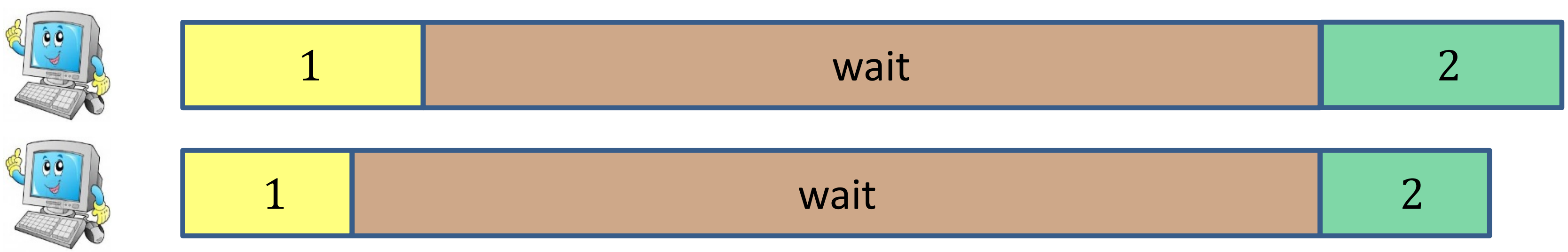
With PT protocols, fast parties start new execution **before** slow parties finished previous execution



Sequential Composition: Solutions

Goal: ℓ sequential executions of expected $O(1)$ rounds protocols in expected $O(\ell)$ rounds

- Naïve solution #1: wait until re-synchronized



- Naïve solution #2:
  Expand each round to $2c + 1$ rounds
  - Execution 1, start slack $c_1 = c$, expansion factor $2c_1 + 1$
  - Execution 2, slack $c_2 = c(2c_1 + 1)$, factor $2c_2 + 1$
  - Execution 3, slack $c_3 = c(2c_2 + 1)$, factor $2c_3 + 1$
  - After i executions, slack $c(2c_{i-1} + 1) = O\left(2^{i-1}c^i\right)$

Sequential Composition: Solutions (2)

Goal: ℓ sequential executions of expected $O(1)$ rounds protocols in expected $O(\ell)$ rounds

- [LLR'02] – add re-synchronization points
  - Statistical security (inherent)
  - Static corruptions
  - Property-based security
- [BE'03, KK'06]
  - Simpler solutions, partial proofs (no simulation)
- We introduce a generic compiler for PT protocols
  - Supports non-simultaneous start of the protocol
  - Reduces the slackness to 1
  - Simulation-based security – a composition theorem

Non-Simultaneous Start: Our Solution

Main idea: Add "dummy" rounds to make overlap meaningless

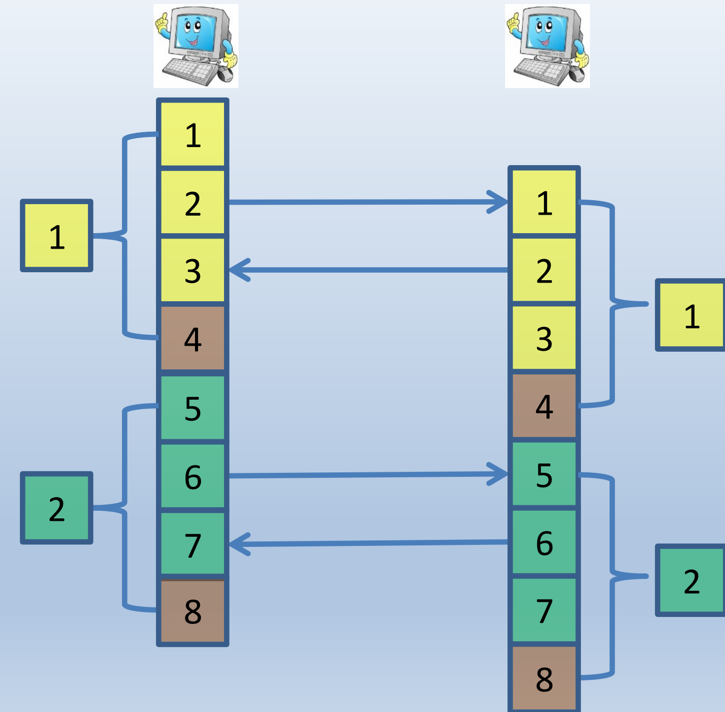
Extend each round to $3c+1$:

- $2c+1$ rounds: listen
  - Round $c+1$: listen & send
- c rounds: wait (without listening)

Concurrent Composition

- Each party proceeds in a locally sequential manner
- Round r messages
  after round $r-1$
  before round $r+1$

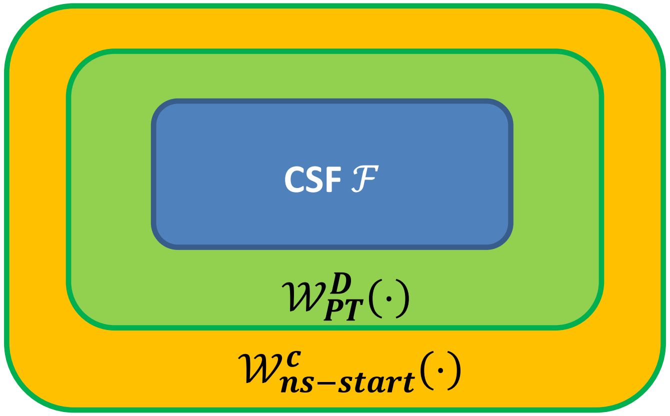
Example: PSMT ($c = 1$)



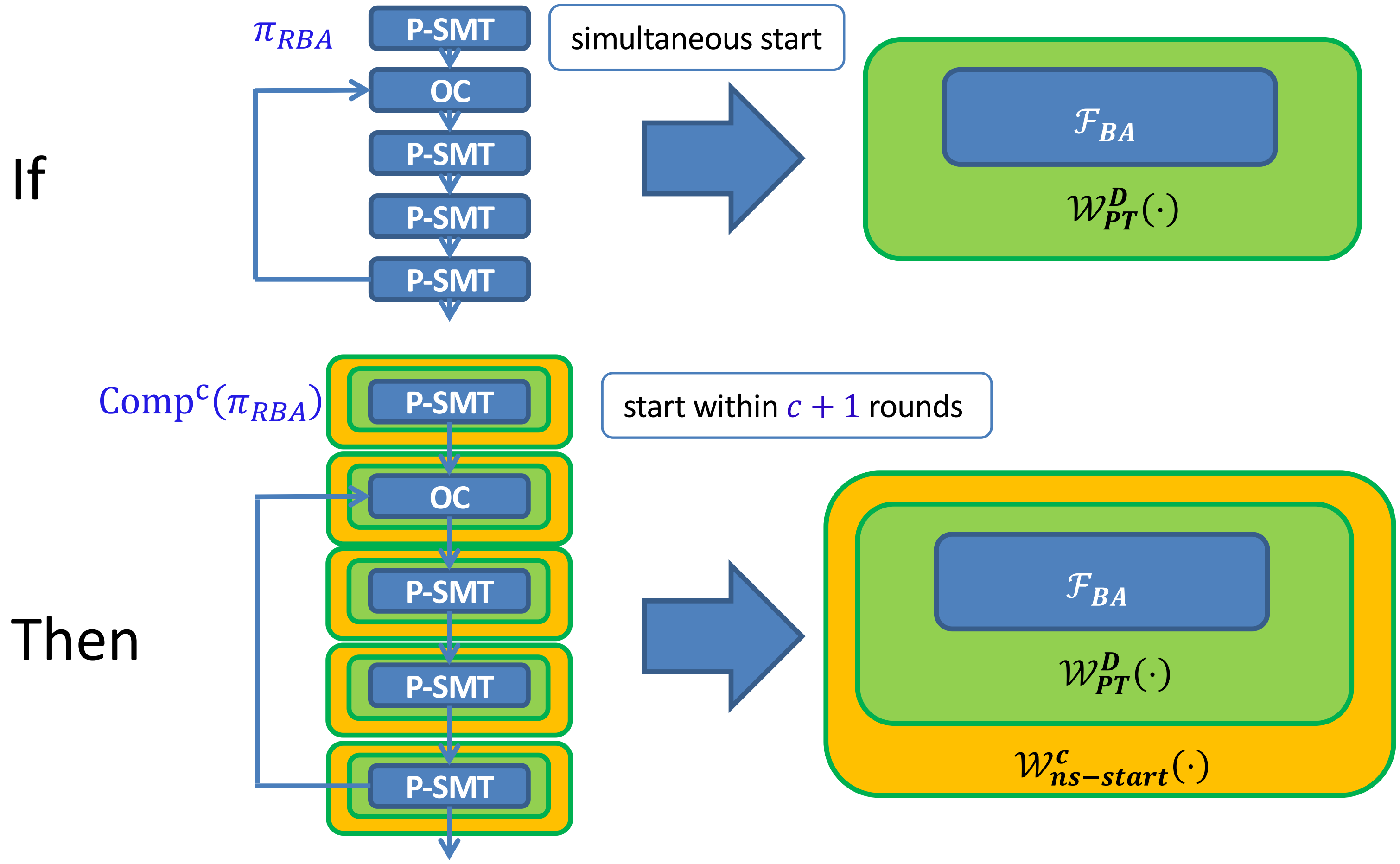
Controlling Round Blowup

- Sequential PT hybrids might cause exponential round blowup
- Use "asynchrony-reduction" techniques [Bracha'84]
  - Upon receiving output v, send (ok, v) to all the parties
  - Upon receiving $t + 1$ messages (ok, v), accepts v
  - Upon receiving $n - t$ messages (ok, v), terminates
- Reduces the asynchrony to **1 round**
- Applies to public-output functionalities

Captured by non-simultaneous start wrapper



Composition Theorem (Illustrated)



Applications

P-Broadcast can be realized over P2P channels in expected $O(1)$ rounds

Recipe for MPC:

1) Construct protocol assuming broadcast channel
2) Replace broadcast channel using PT parallel broadcast protocol

SFE can be realized over P2P channels

- Info-theoretic in expected $O(\text{depth})$ rounds
- Assuming OWF in expected $O(1)$ rounds

Composition of *Arbitrary* PT Protocols

Arbitrary PT Protocols

Problem:

Solutions for broadcast crucially rely on its **privacy-free** nature

The new MPC protocols have probabilistic termination

(Naïve parallel composition not round preserving)

Can parallel composition of *arbitrary* PT protocols be round-preserving?

Parallel Composition of Functions

Given n-party functions $f_1, f_2, \ldots, f_m$

denote by $f_1 \parallel f_2 \parallel \cdots \parallel f_m$ the following function:

- Each P_i has input $\boldsymbol{x}_i = \left(x_i^1, x_i^2, \ldots, x_i^m\right)$
- Output is $\boldsymbol{y} = (y_1, y_2, \ldots, y_m)$

$$f_1(x_1^1, x_2^1, \ldots, x_n^1)$$

$$f_2(x_1^2, x_2^2, \ldots, x_n^2)$$

$$f_m(x_1^m, x_2^m, \ldots, x_n^m)$$

Arbitrary PT Protocols (2)

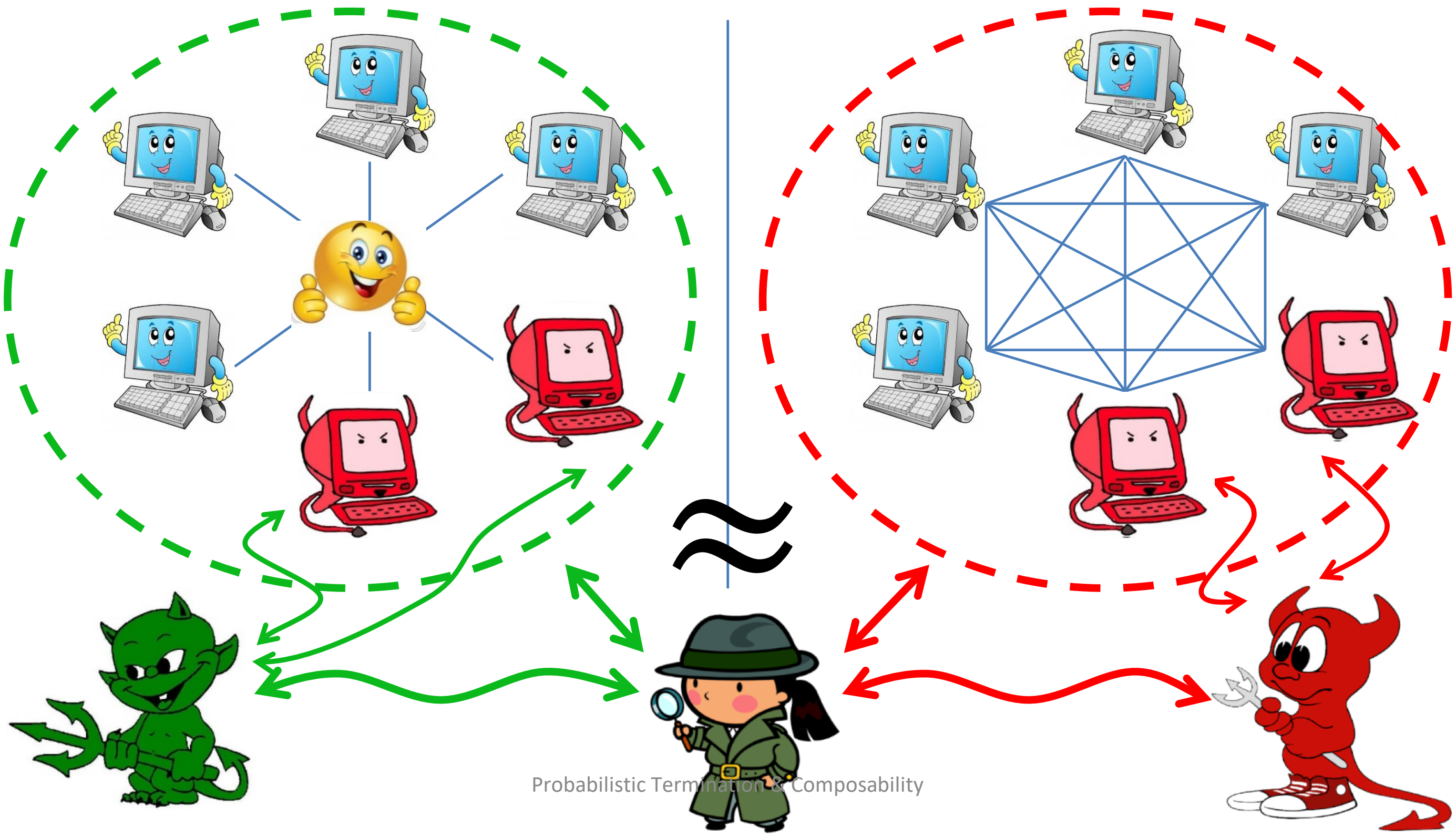
Can parallel composition of arbitrary PT protocols be round-preserving?
In a **black-box** way?

BB w.r.t. **function**
[Rosulek'12, IKPSY'16]

BB w.r.t. **protocol**
[IKOS'07, IKPSY'16]

Synchronous MPC [KMTZ'13, CCGZ'16]

- Ideal world captures round complexity of π
- Trusted party samples $r_{term} \leftarrow D = D(\pi)$
- Parties continuously ask for output (receive by r_{term})
- $\mathcal{S}$ can instruct early delivery for specific parties



Protocol-BB Parallel Composition

Protocol-BB Parallel Composition

Theorem:

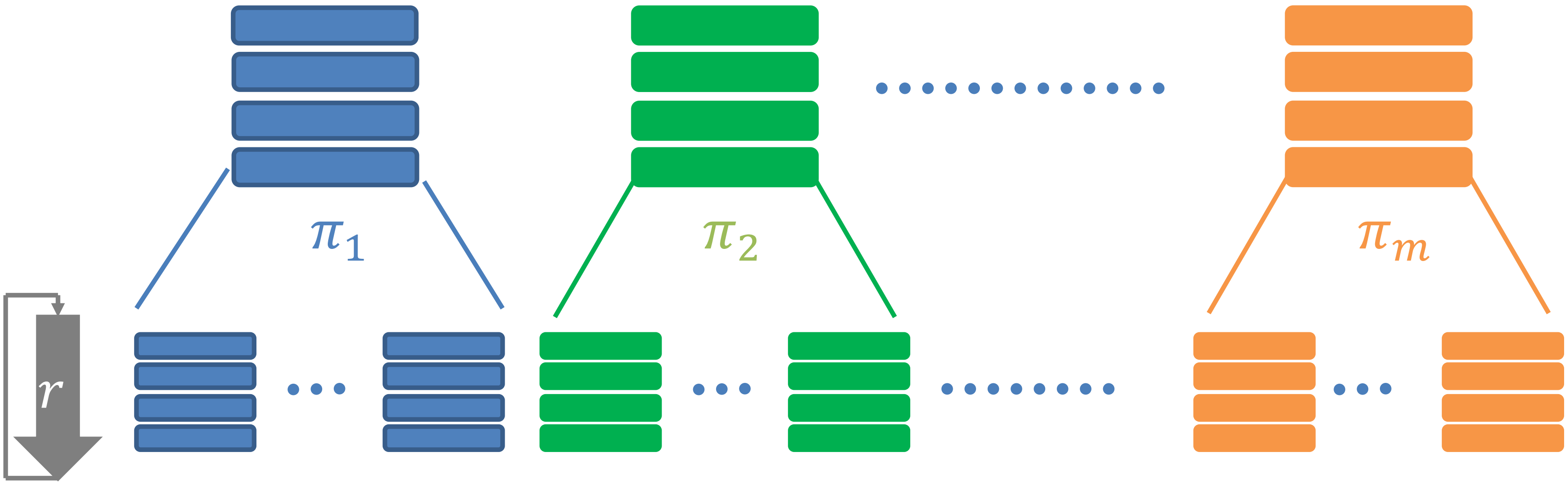
- Let $\pi_1, \dots, \pi_m$ be PT protocols realizing $f_1, \dots, f_m$
- Then $\pi = \mathrm{compiler}(\pi_1, \dots, \pi_m)$ realizes $f_1 \parallel \cdots \parallel f_m$ s.t.
  - Composition is round-preserving, i.e.

$$\mathbb{E}(\pi) = O\left(\max_i \mathbb{E}(\pi_i)\right)$$

  - Black-box w.r.t. protocols $\pi_1, \dots, \pi_m$

The compiler doesn't know the code of π_i

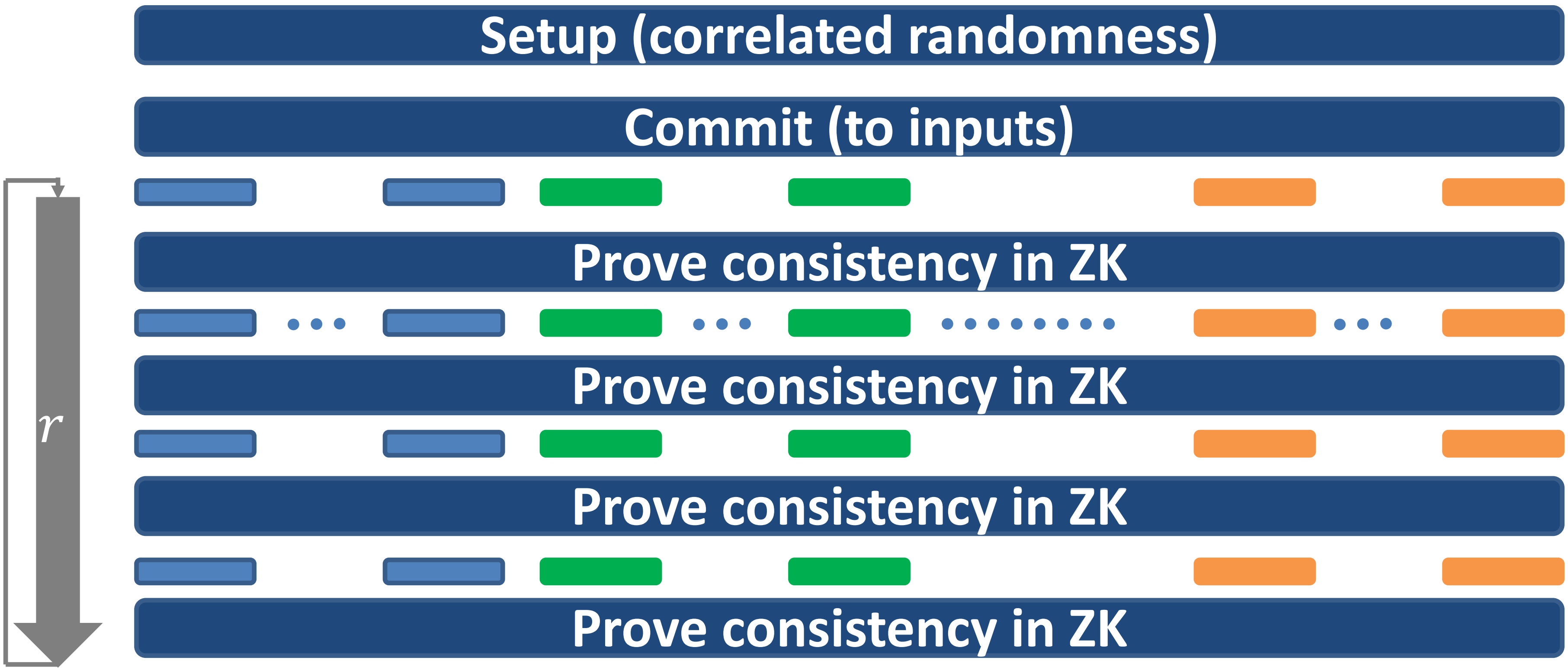
Protocol Compiler



Prevent Multiple Inputs

Use **Setup, Commit, then Prove** functionality with a tweak [Canetti-Lindell-Ostrovsky-Sahai'02] [Ishai-Ostrovsky-Zikas'14]

Prevent Multiple Inputs



Use **Setup, Commit, then Prove** functionality with a tweak [Canetti-Lindell-Ostrovsky-Sahai'02, Ishai-Ostrovsky-Zikas'14]

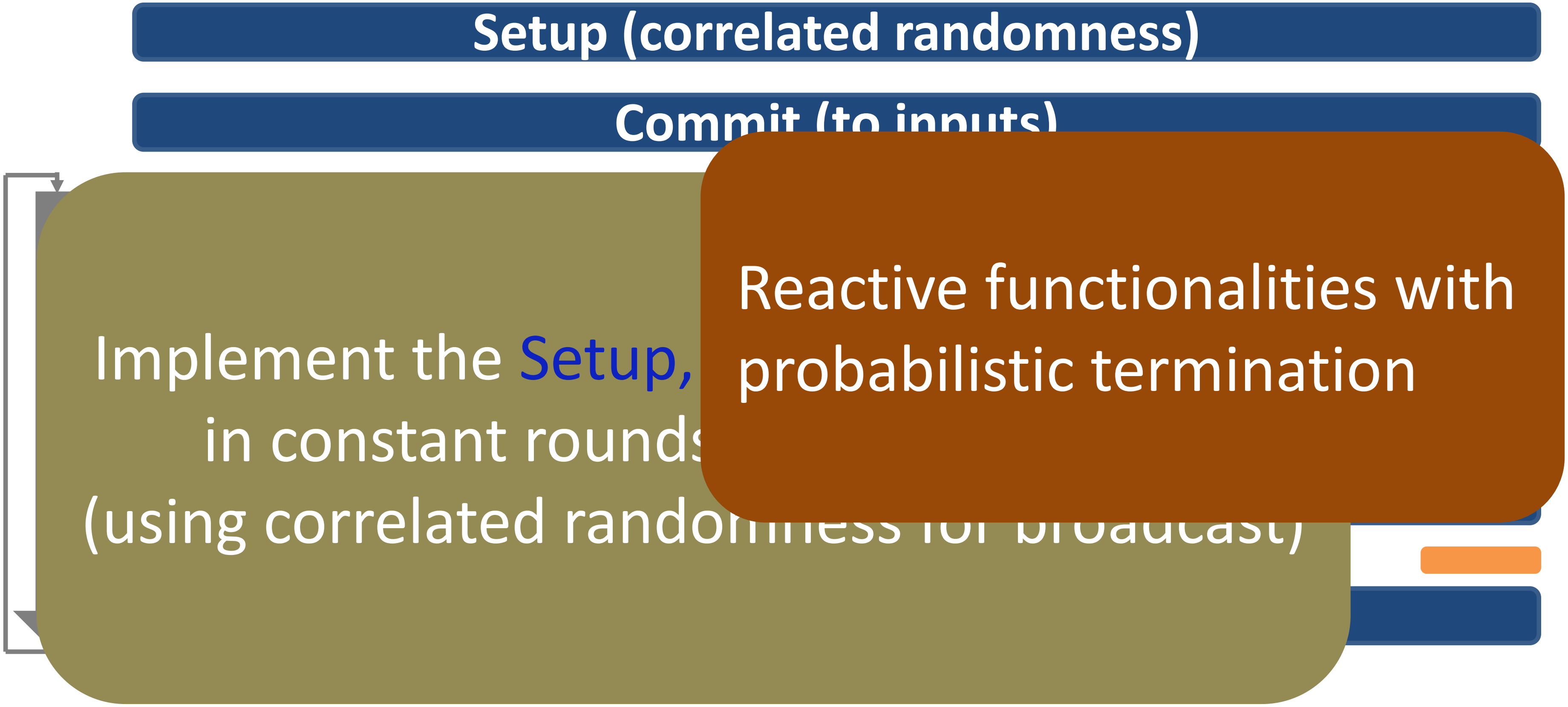
Technical Challenges

Setup (correlated randomness)

Commit (to inputs)

Implement the Setup, Commit functionality
in constant rounds & info. theoretic
(using correlated randomness for broadcast)

Technical Challenges



Technical Challenges

Setup (correlated randomness)

Commit (to inputs)

Implement the Setup,
in constant rounds
(using correlated randomness for broadcast)

Reactive functionalities with probabilistic termination

Extend the sequential-composition theorems from [CCGZ'16]

Technical Challenges

Setup (correlated randomness)

Commit (to inputs)

Implement the Setup,
in constant rounds

st)

Reactive functionalities with probabilistic termination

mposition

1-to-many information-theoretic ZK black-box in $\pi_1, \dots, \pi_m$ (honest majority) in constant rounds

Technical Challenges

Setup (correlated randomness)

Commit (to inputs)

Reactive functionalities with

Impleme

in co

1-to-many

black-box

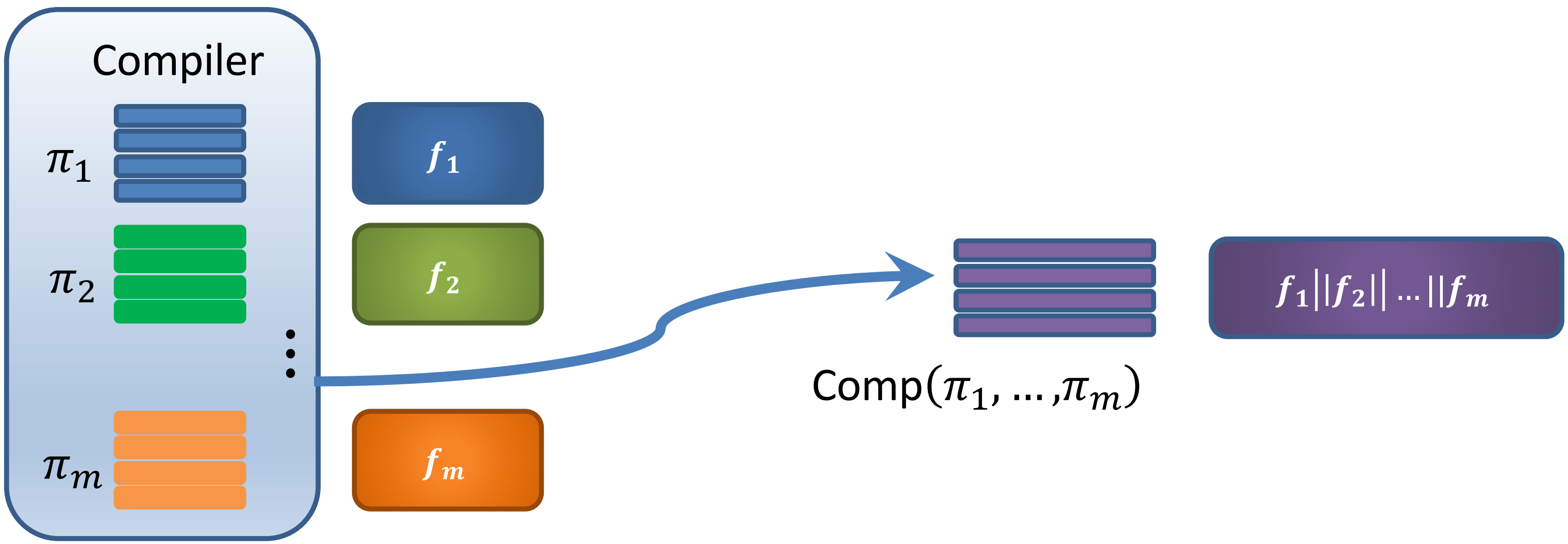
in constant

n

Recover from invalid ZK proofs without:

1) Breaching privacy ($\mathcal{A}$ might have learned output)
2) Blowing up round complexity

Round-preserving Protocol-BB Parallel Composition

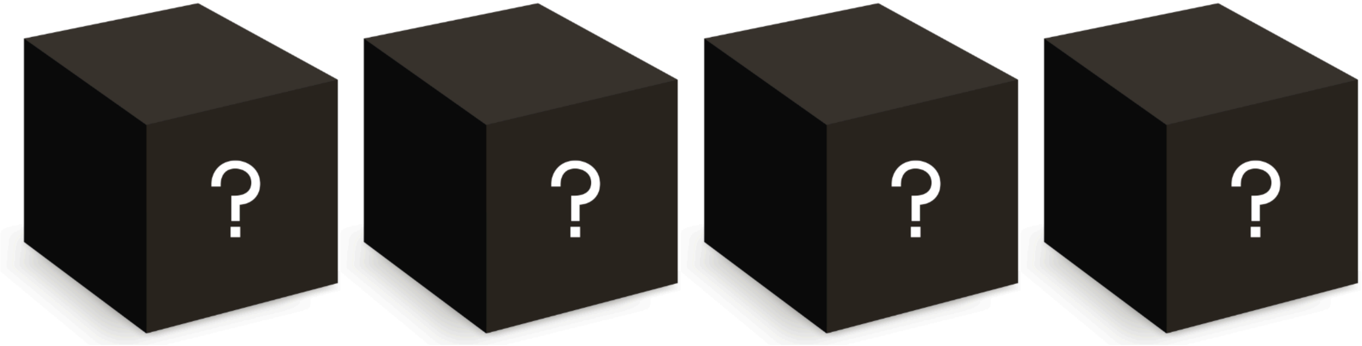


π_i has expected $O(1)$ rounds

$\text{Comp}(\pi_1, \ldots, \pi_m)$ has expected $O(1)$ rounds

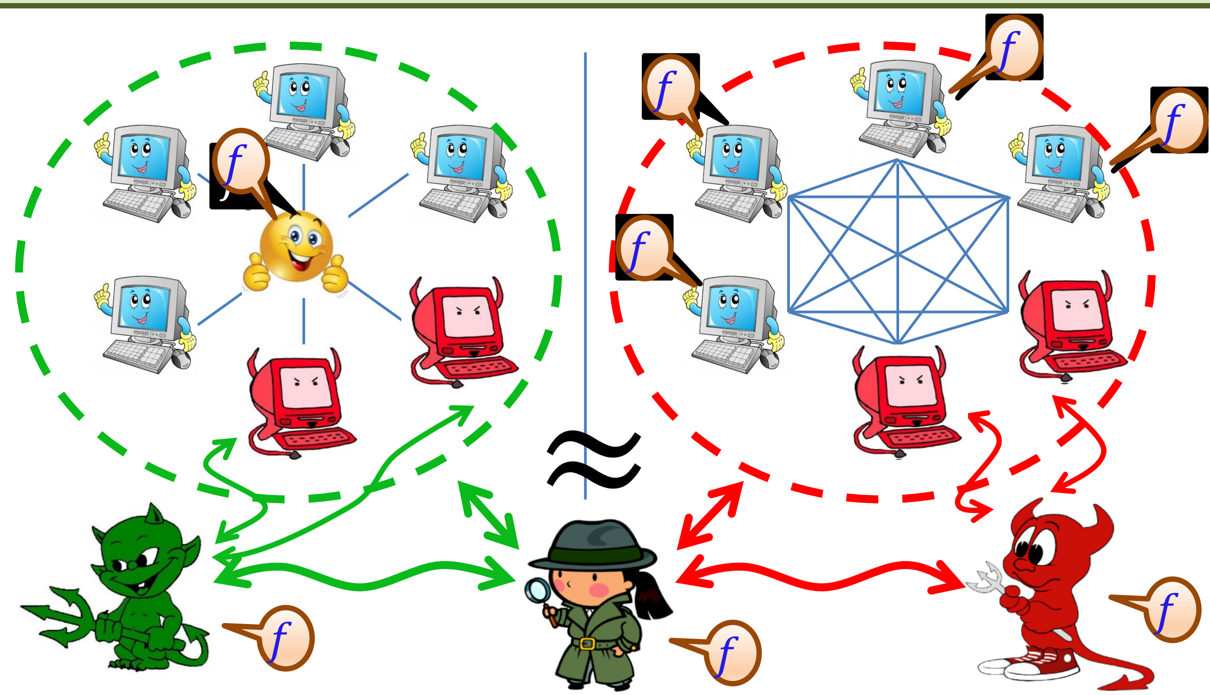
Black-box: Compiler doesn’t look at the code of π_i

FBB Parallel Composition



Functionally BB Protocols

Protocol π is **FBB protocol** for $\mathcal{F}$ if $\forall f \in \mathcal{F}$ protocol π^f securely computes f

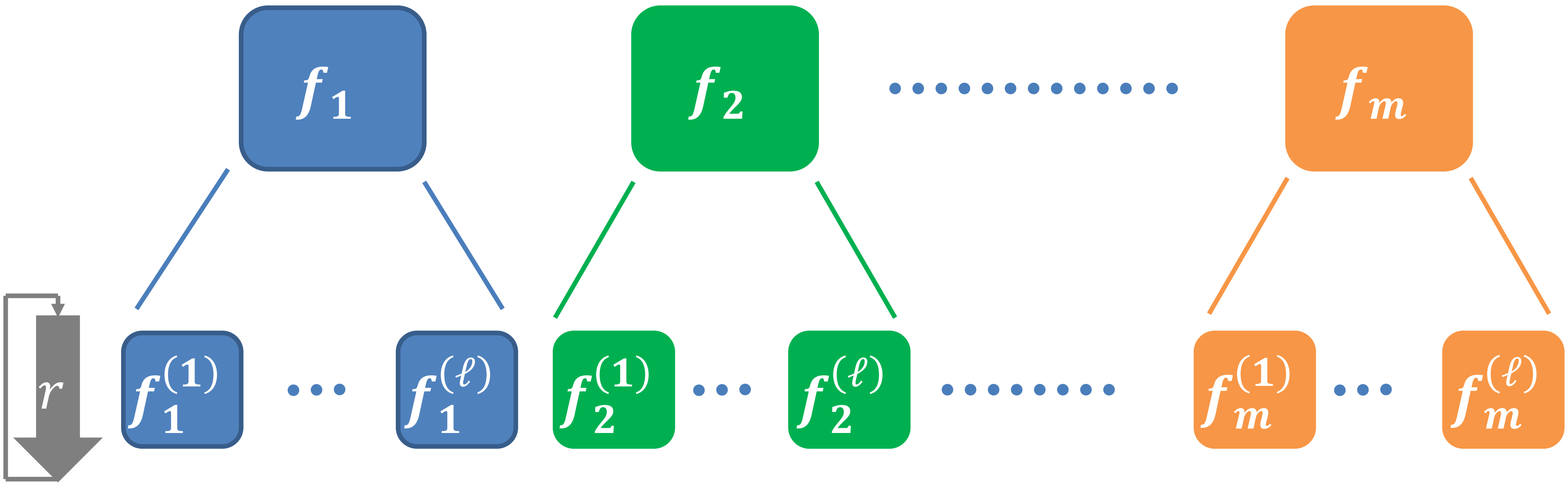


Semi-Honest FBB Protocol

Theorem:

- Let $\mathcal{F}_1, \ldots, \mathcal{F}_m$ be deterministic function classes
- Consider $(\mathcal{F}_1, \ldots, \mathcal{F}_m)$-hybrid model
  that $\forall j$ computes a function $f_j \in \mathcal{F}_j$
  with expected constant round complexity μ
- Then $\exists$ FBB protocol for $\mathcal{F}_1 \parallel \cdots \parallel \mathcal{F}_m$
  with expected constant round complexity

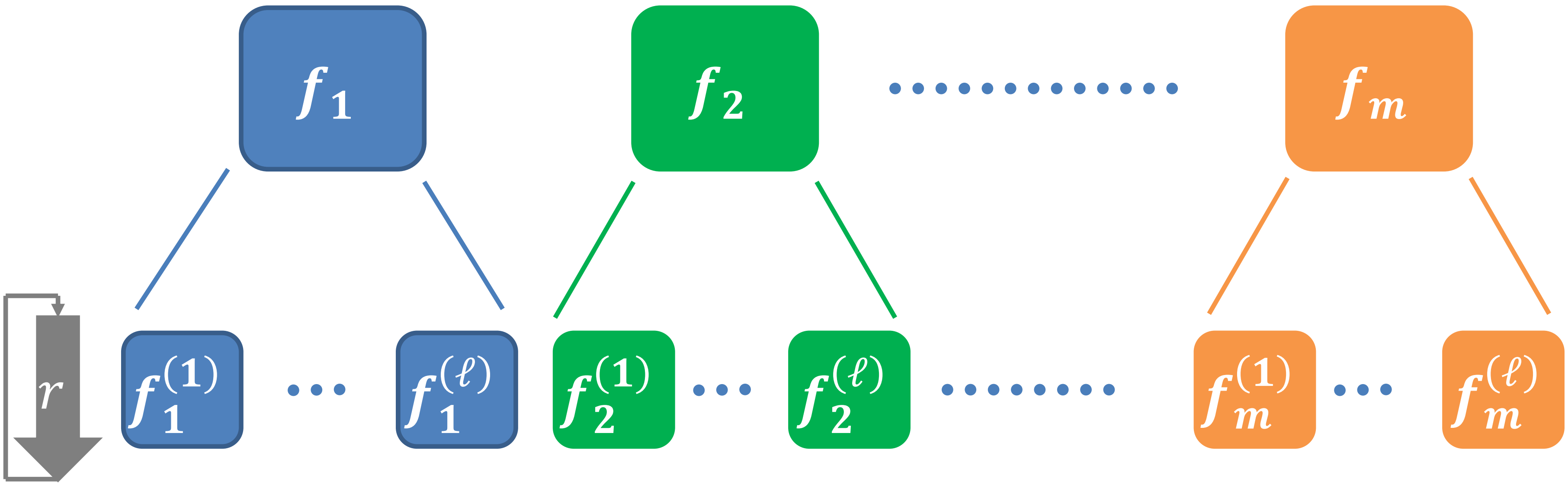
Semi-Honest FBB Protocol



1) Parties invoke ℓ instances of each f_j
2) Each P_i sends x_i^j to all instances of f_j and asks output for r rounds
3) If some P_i received output y_j for each f_j distribute $(y_1, \ldots, y_m)$ and halt, otherwise restart

parameters

Semi-Honest FBB Protocol



Proof intuition:

- ✓ Correctness
- ✓ Privacy: corrupt parties always use the same input values (semi-honest)
- ✓ Round complexity: for $\ell = \Omega(\log m)$ and constant $r > \mu$, the expected number of "restarts" is constant (Markov)

What About Malicious Adv.?

- The previous protocol is **not maliciously secure**
- The adversary can send different x_i^j and $\tilde{x}_i^j$ to f_j and learn multiple outputs
- This is inherent for **batched parallel-composition protocols**
  - For some f_k, all parties use original inputs $(x_1^k, \dots, x_n^k)$ in two calls to the trusted party
  - Possibly in different rounds ρ and ρ'
  - Possibly for computing different f_j and $f_{j'}$

Functionally BB Parallel Composition

There exist function classes $\mathcal{F}_1, \dots, \mathcal{F}_m$ s.t. for protocols computing $\mathcal{F}_1 \parallel \cdots \parallel \mathcal{F}_m$ in $(\mathcal{F}_1, \dots, \mathcal{F}_m)$-hybrid model, either:

- Correctness is lost
- Privacy is broken
- Round complexity blows-up

Using known techniques

Want: $\mathcal{F}_1 \parallel \cdots \parallel \mathcal{F}_m$

Have: $\mathcal{F}_1$ $\mathcal{F}_2$ … $\mathcal{F}_m$

Summary & Open Questions

- We considered composability of cryptographic protocols with *probabilistic termination*
- Framework for designing cryptographic protocols in stand-alone fashion and compiler to fast composition in the UC framework
  - P-Broadcast can be realized over P2P channels in expected $O(1)$ rounds
  - Recipe for MPC:
    1) Construct protocol assuming broadcast channel
    2) Replace broadcast channel using PT parallel broadcast protocol
  - MPC can be realized over P2P channels
    - Info-theoretic in expected $O(\text{depth})$ rounds
    - Assuming OWF in expected $O(1)$ rounds

Summary & Open Questions (2)

- Parallel composition of *arbitrary* PT protocols
  - **Black-box w.r.t. protocols:** Round-preserving compiler for parallel composition
  - **Functionally block-box (FBB) protocols:**
    - No round-preserving parallel composition (using known techniques)
    - Round-preserving parallel composition with semi-honest security

Open:

- Does there exist a round-preserving FBB protocol for parallel composition of PT protocols?
- Partially synchronous/asynchronous PT protocols
- Dishonest-majority PT protocols

References

- R. Cohen, S. Coretti, J. Garay and V. Zikas. "Probabilistic Termination and Composability of Cryptographic Protocols," J. Cryptology 21(3): 690-741 (2019). Preliminary version in *Crypto 2016.* (See also ePrint)
- R. Cohen, S. Coretti, J. Garay and V. Zikas. "Round-preserving Parallel Composition of Probabilistic-Termination Protocols," *ICALP 2017*: 37:1-37:15. (See also ePrint)

Thank You