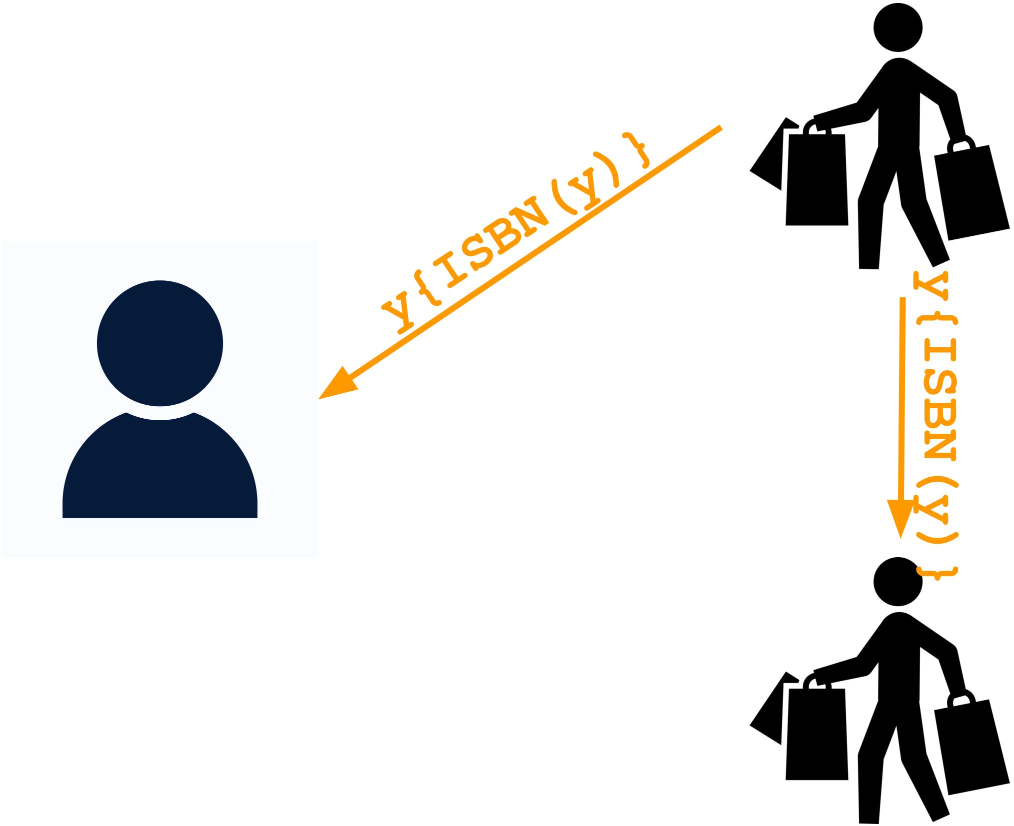
Certified Implementability of Global Multiparty Protocols

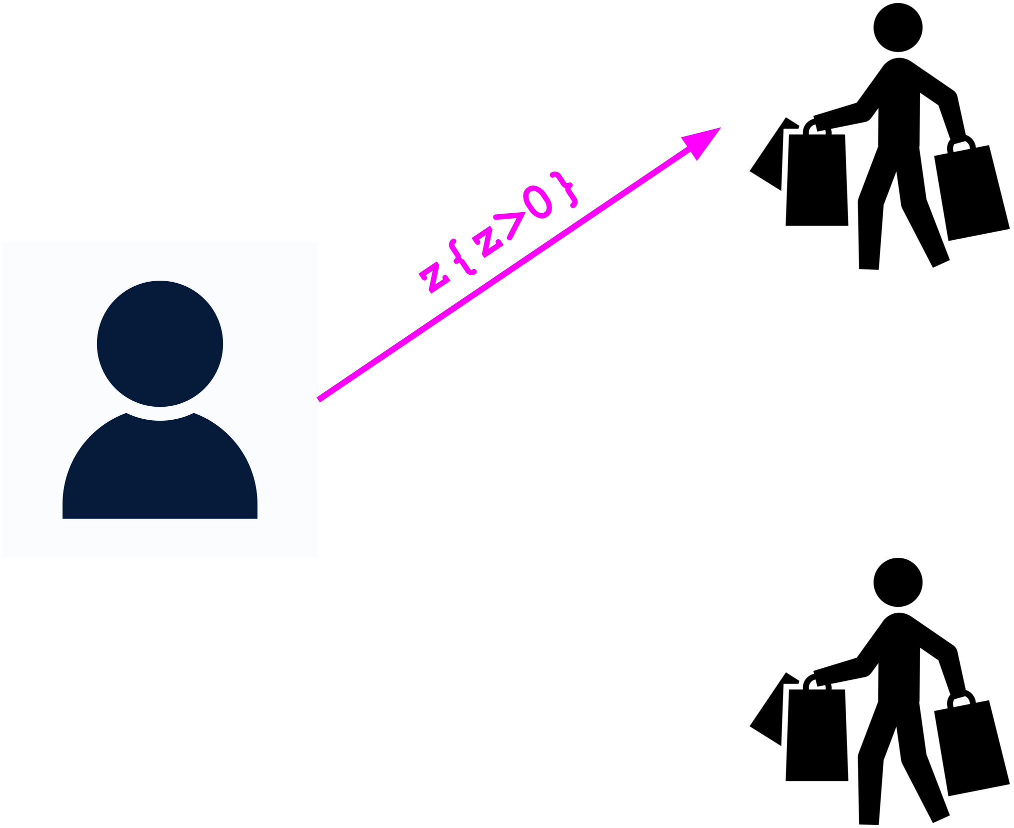
Elaine Li Thomas Wies

NYU

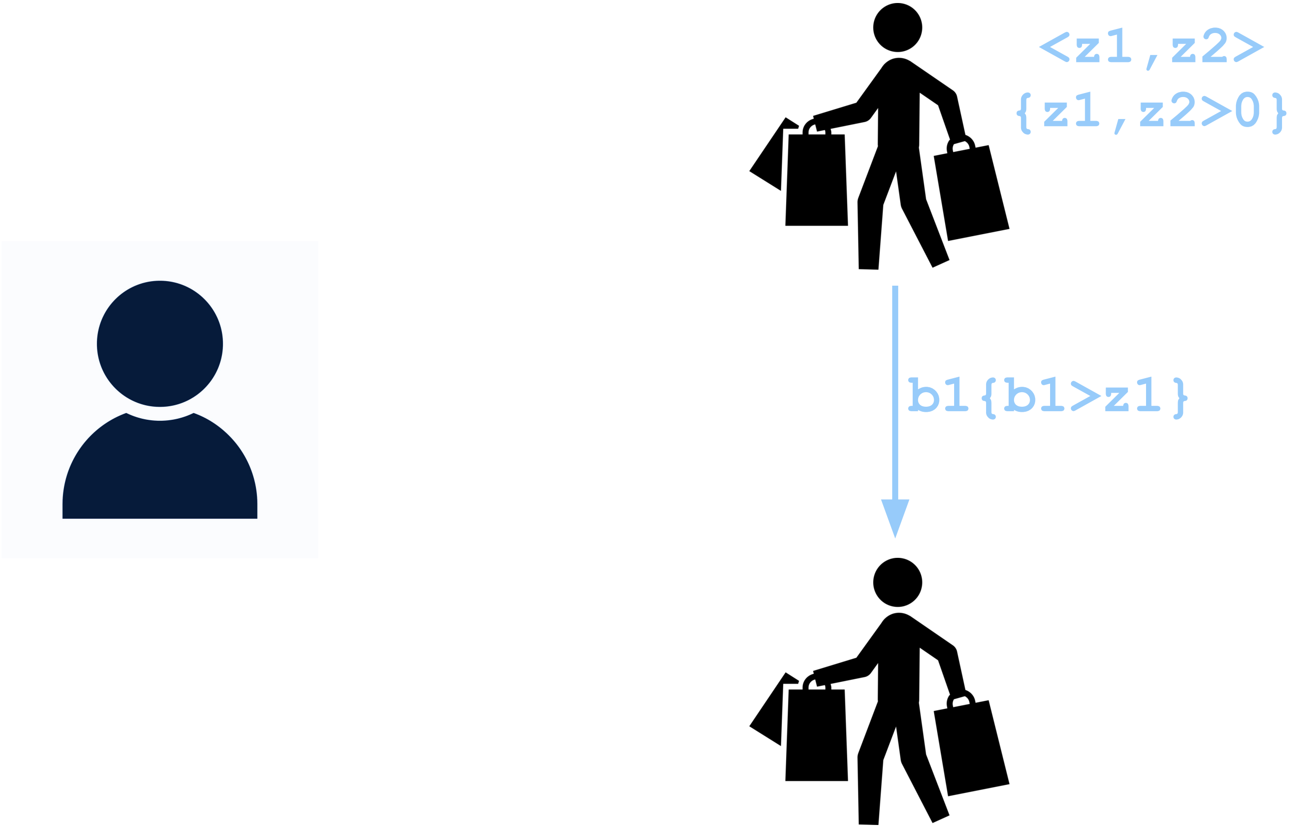
Global protocols: two-bidder protocol



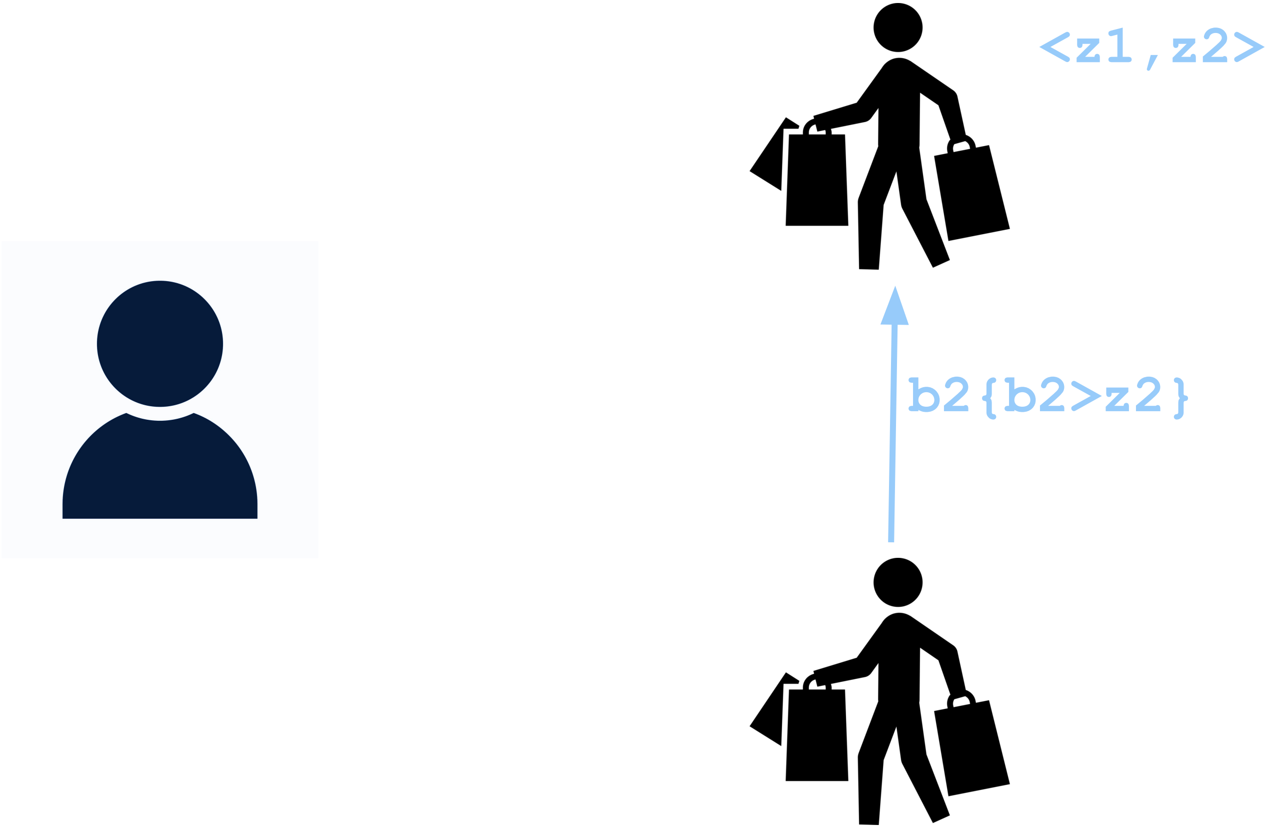
Global protocols: two-bidder protocol



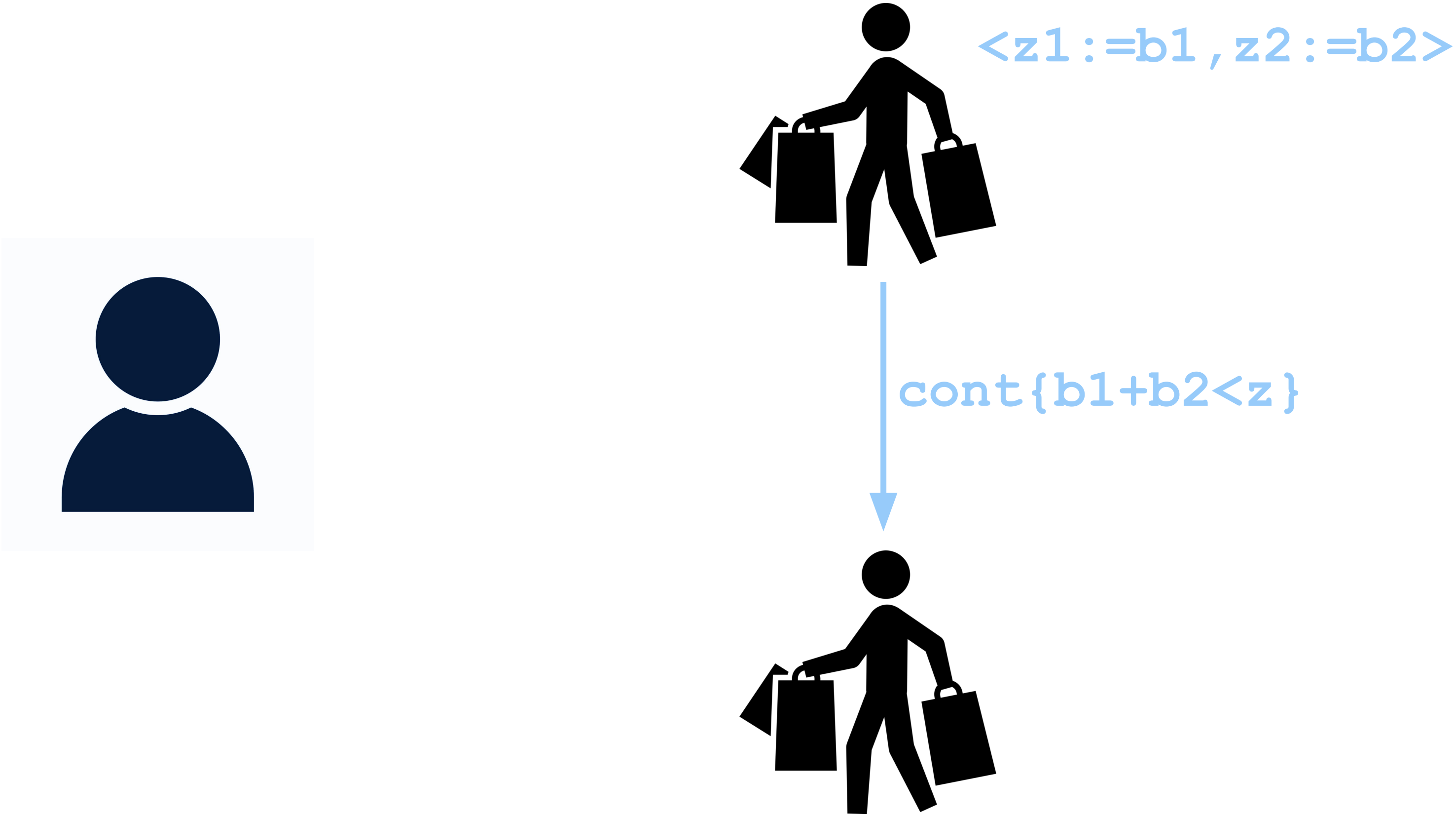
Global protocols: two-bidder protocol



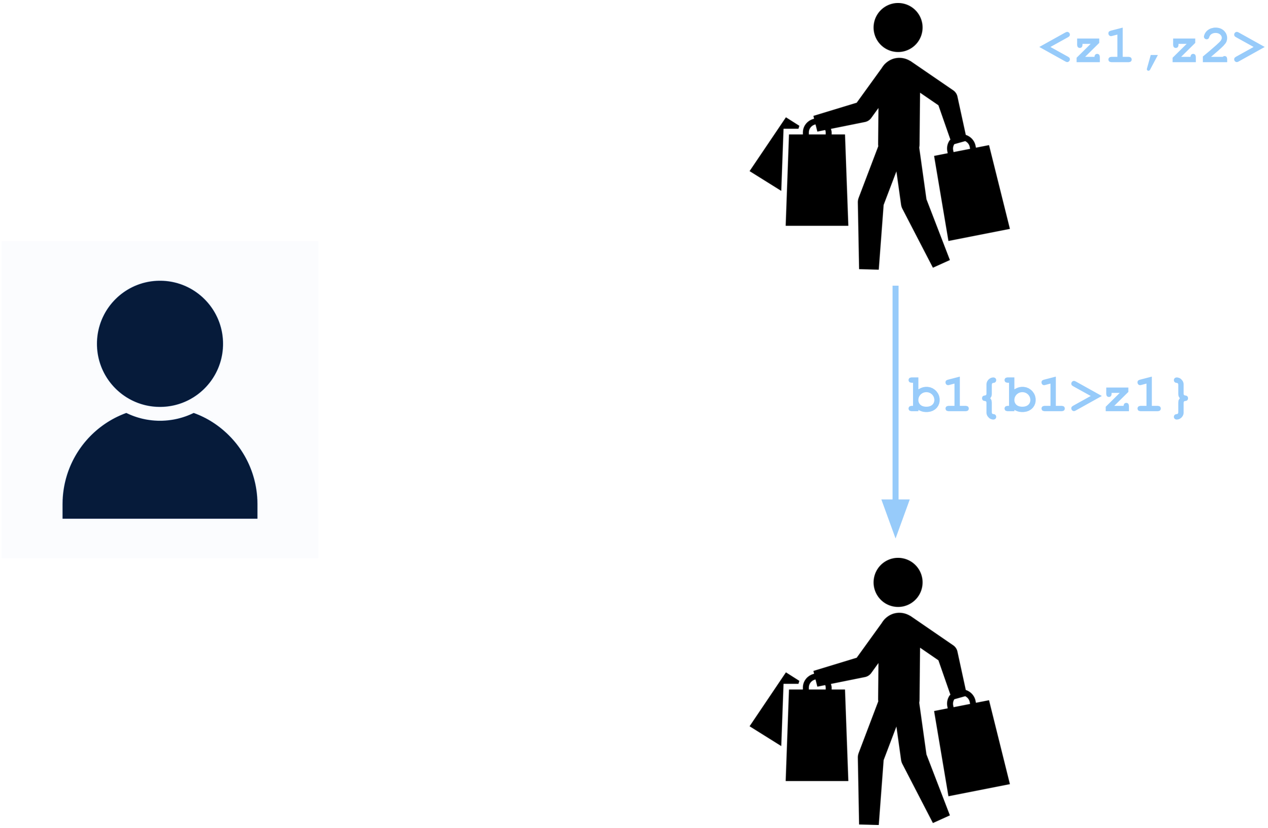
Global protocols: two-bidder protocol



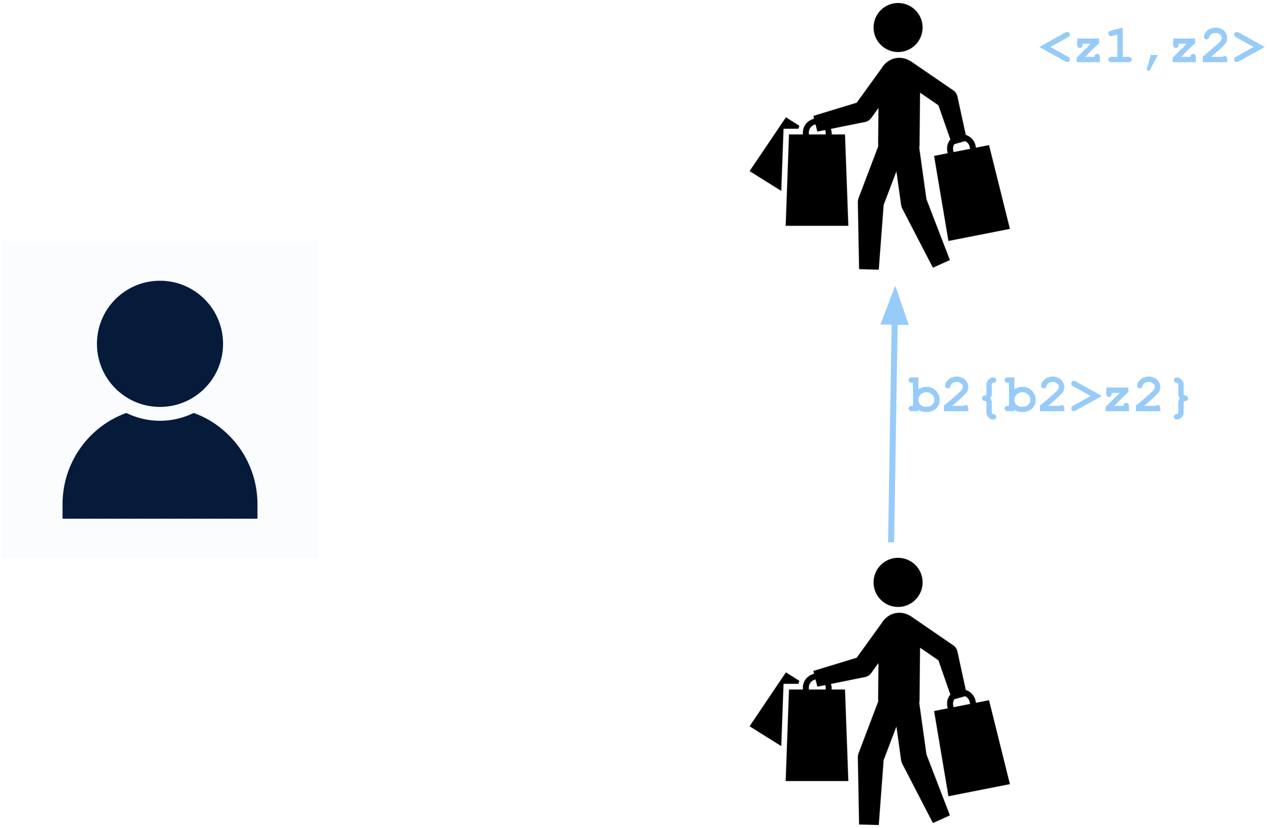
Global protocols: two-bidder protocol



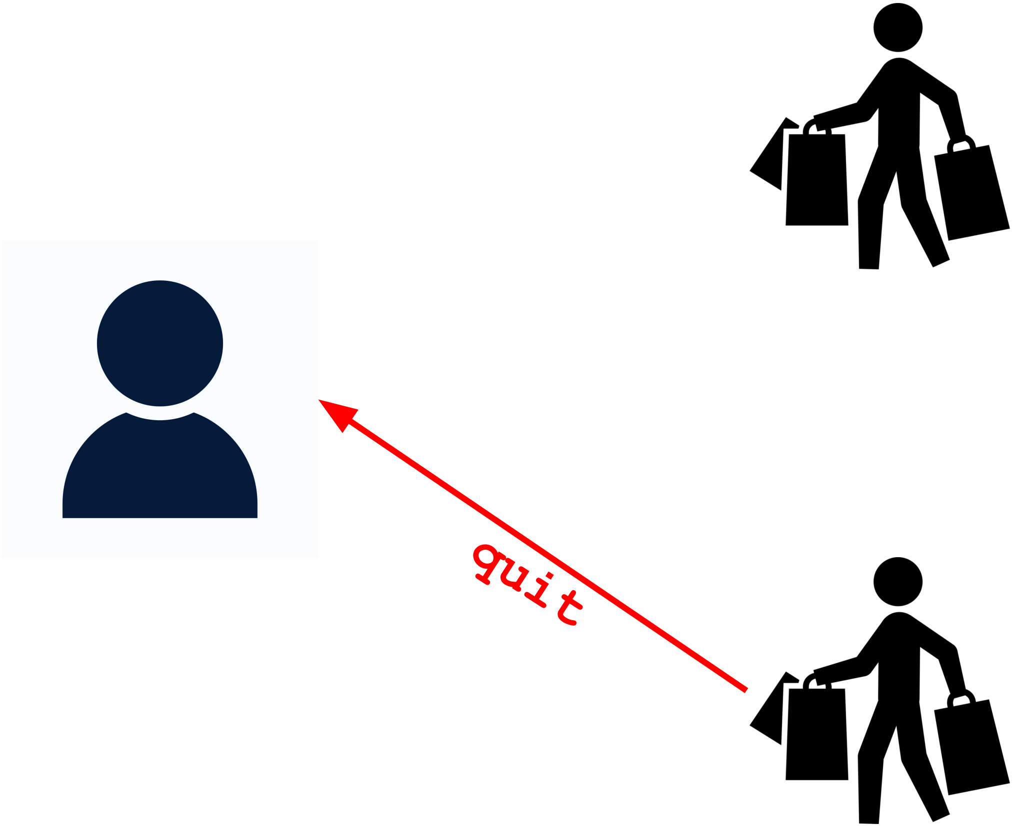
Global protocols: two-bidder protocol



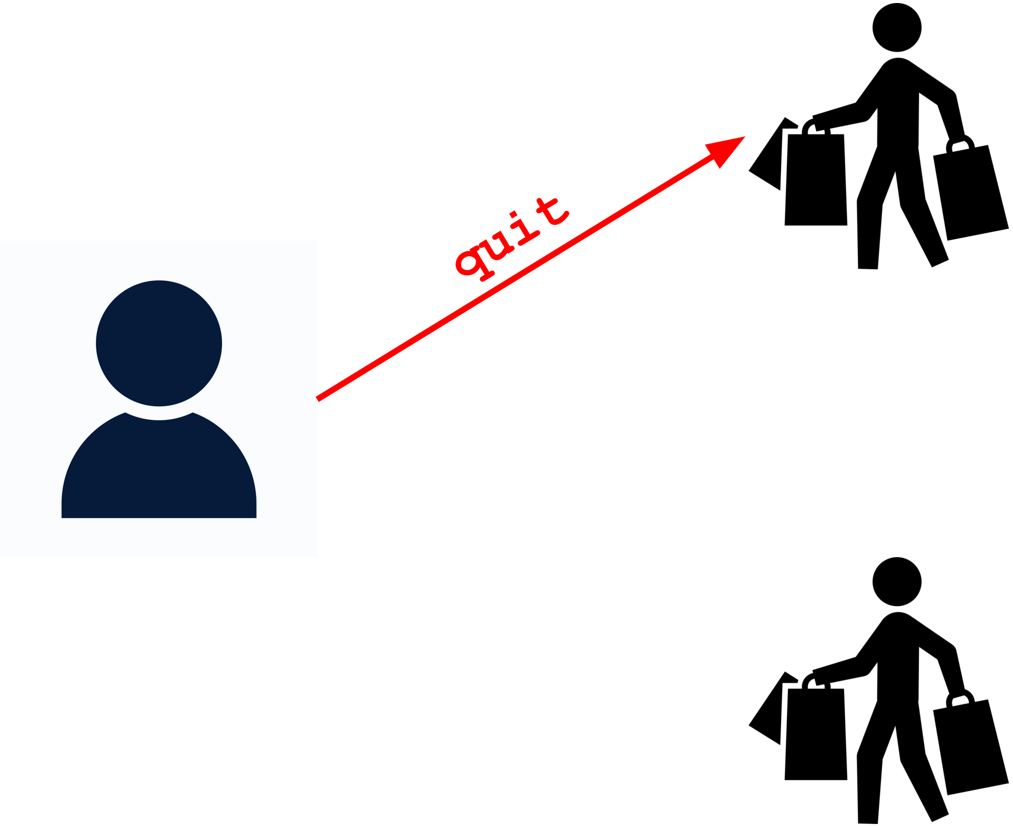
Global protocols: two-bidder protocol



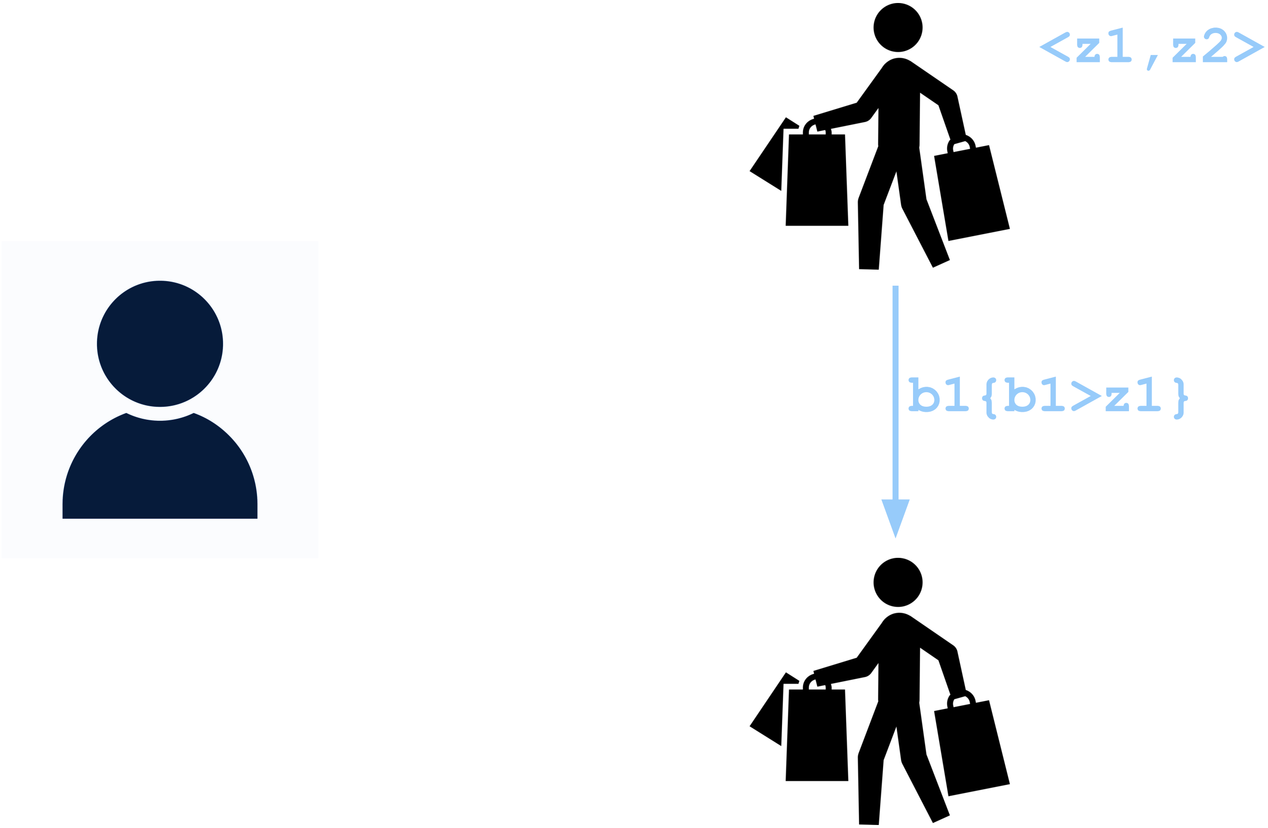
Global protocols: two-bidder protocol



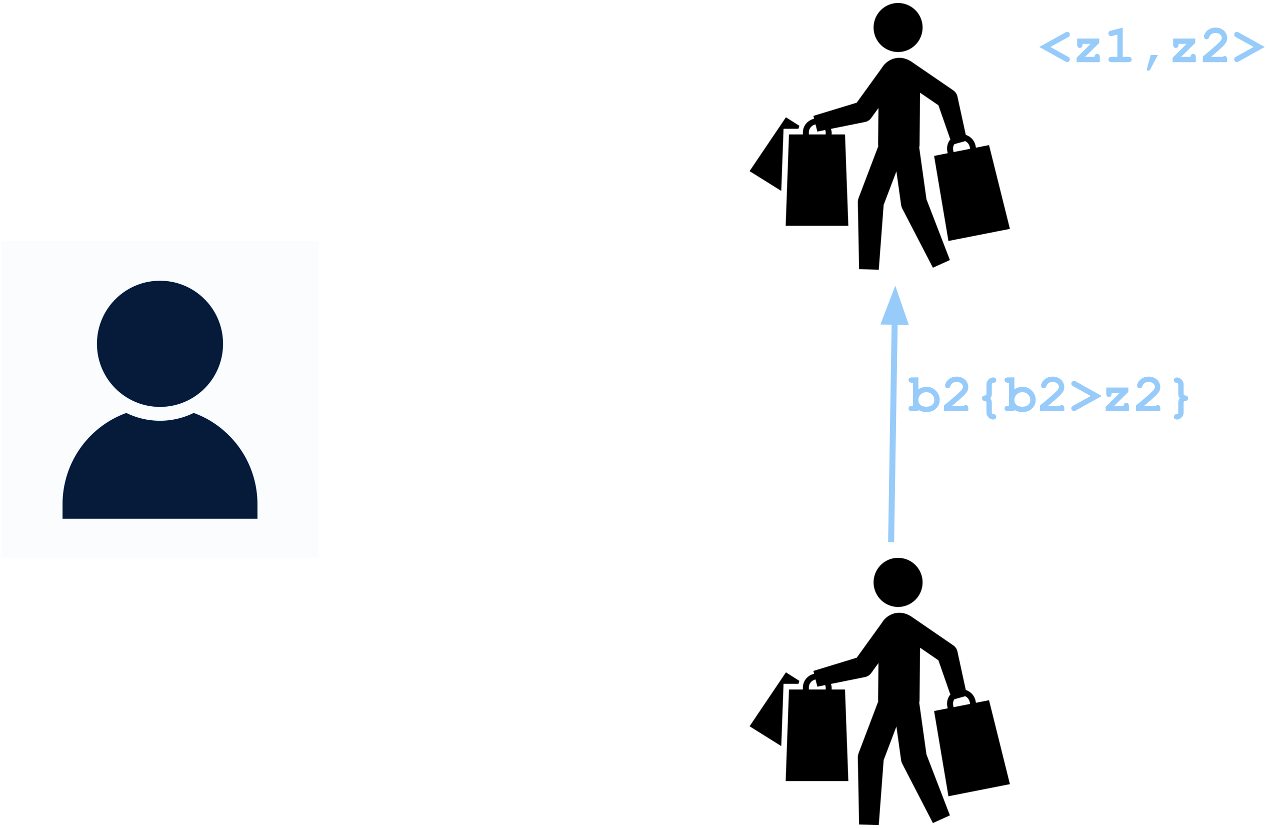
Global protocols: two-bidder protocol



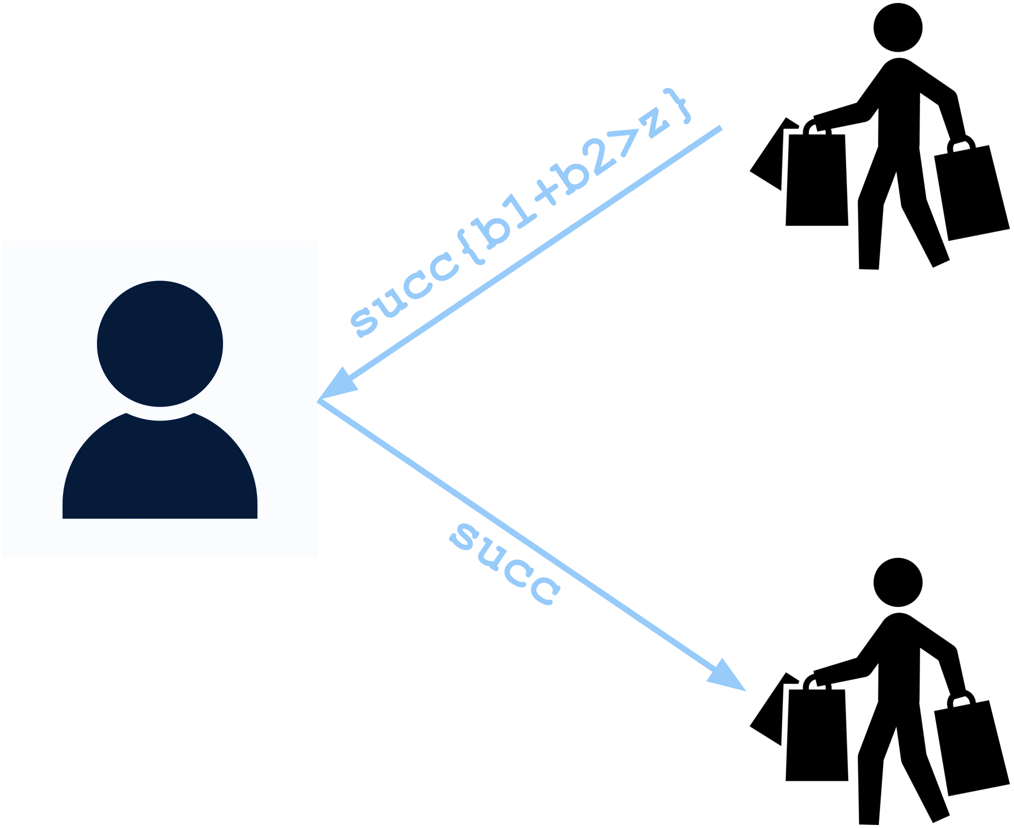
Global protocols: two-bidder protocol



Global protocols: two-bidder protocol



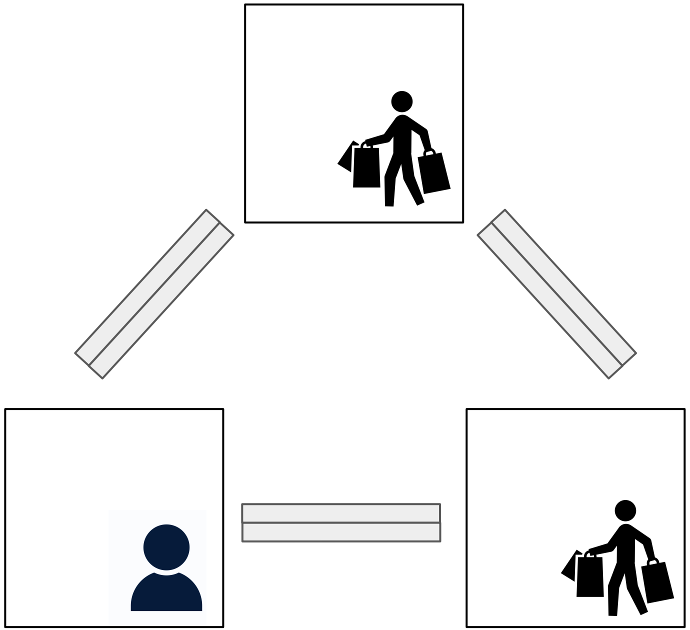
Global protocols: two-bidder protocol



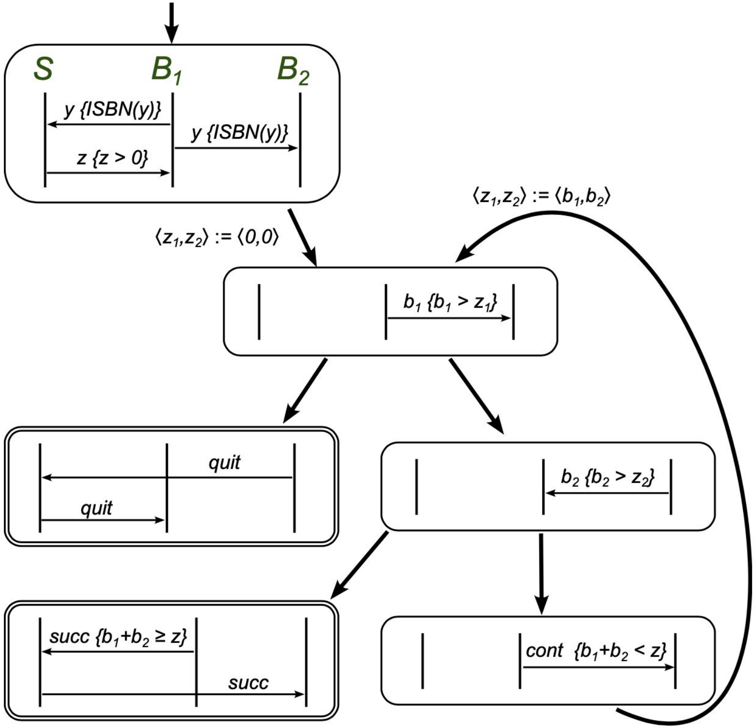
Overview

Asynchronous, message-passing programs are challenging to implement individually

- Communication errors e.g. orphan messages, unspecified receptions
- Deadlocks



Overview

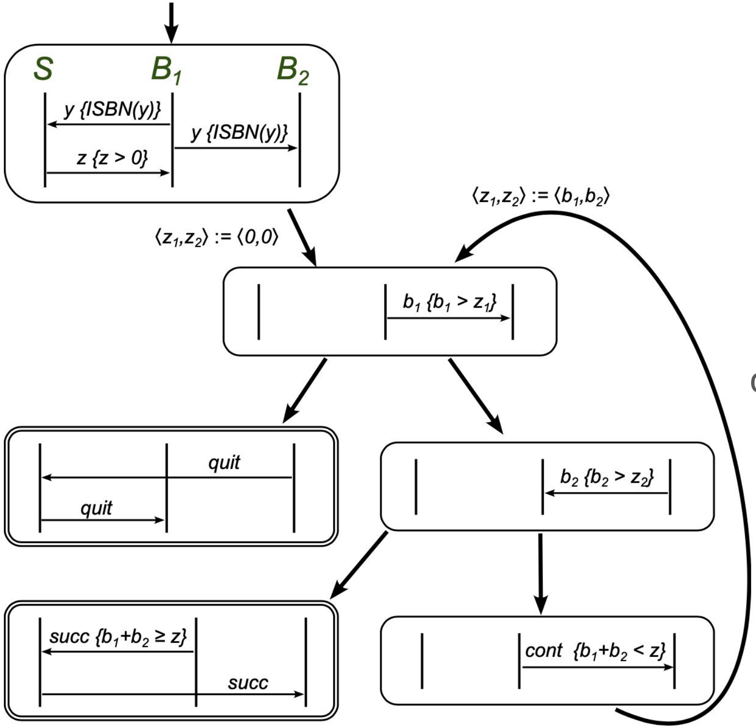


Global protocols are **synchronous** specifications of **all** participants' behaviors

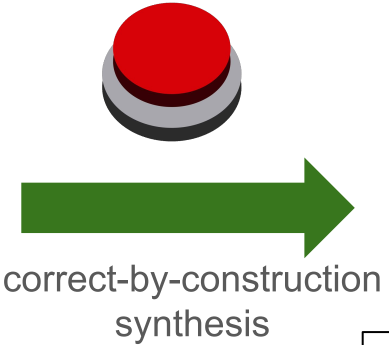
- High-level message sequence charts [Mauw and Reniers 97]
- Session types [Honda et al. 08]
- Choreographic programming [Carbone and Montesi 13]

Applications: ITU standard, UML, Web Service Choreography Description Language, cyber-physical systems etc.

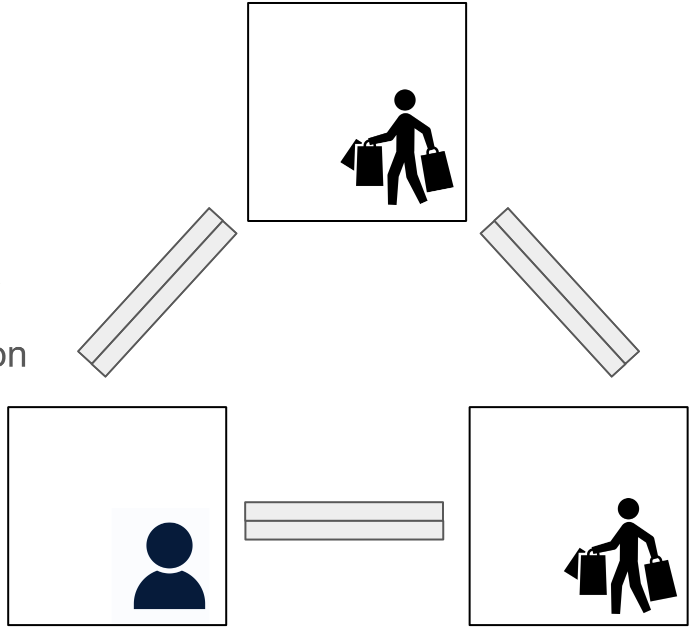
Overview



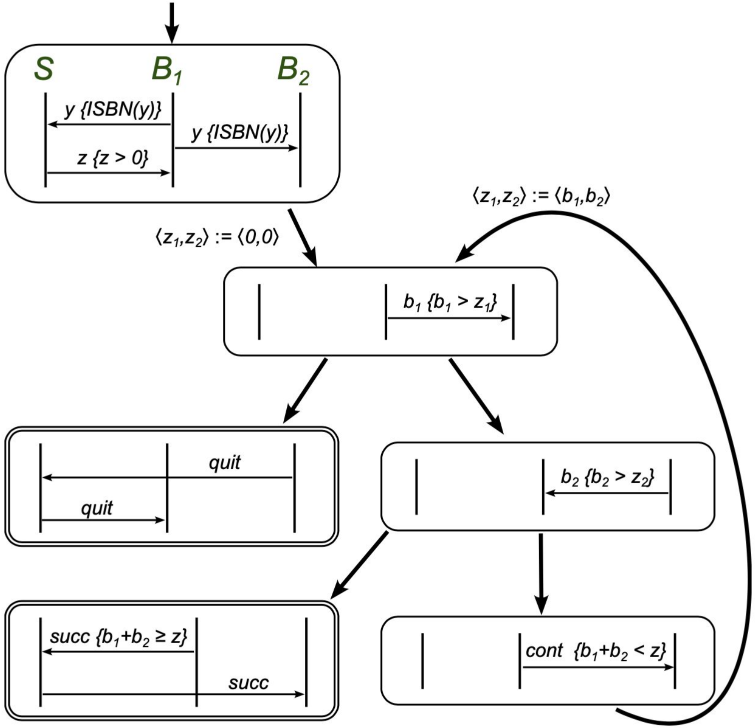
S
B1
B2
y {ISBN(y)}
y {ISBN(y)}
z {z > 0}
⟨z1,z2⟩ := ⟨0,0⟩
⟨z1,z2⟩ := ⟨b1,b2⟩
b1 {b1 > z1}
quit
quit
b2 {b2 > z2}
succ {b1+b2 ≥ z}
succ
cont {b1+b2 < z}



correct-by-construction
synthesis



Overview



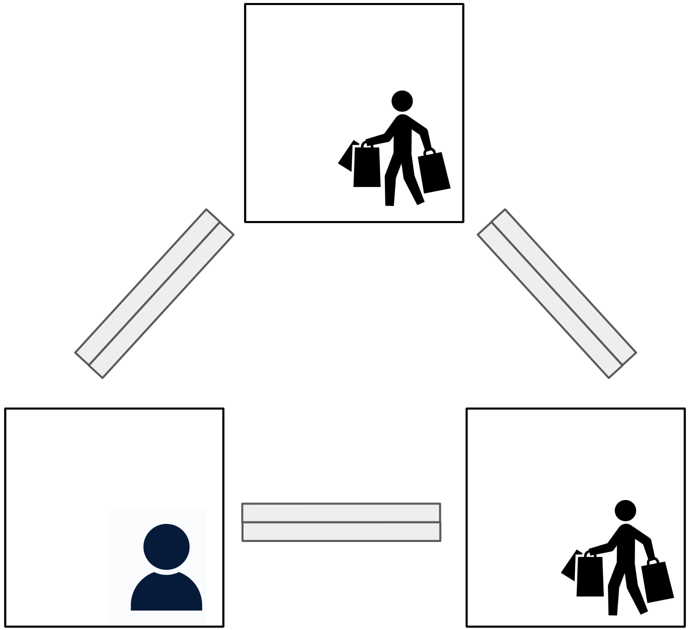
S
B_1
B_2
y {ISBN(y)}
y {ISBN(y)}
z {z > 0}
⟨z_1,z_2⟩ := ⟨0,0⟩
⟨z_1,z_2⟩ := ⟨b_1,b_2⟩
b_1 {b_1 > z_1}
quit
quit
b_2 {b_2 > z_2}
succ {b_1+b_2 ≥ z}
succ
cont {b_1+b_2 < z}

?
implementability

Implementation model

(controllable) **(non-controllable)**

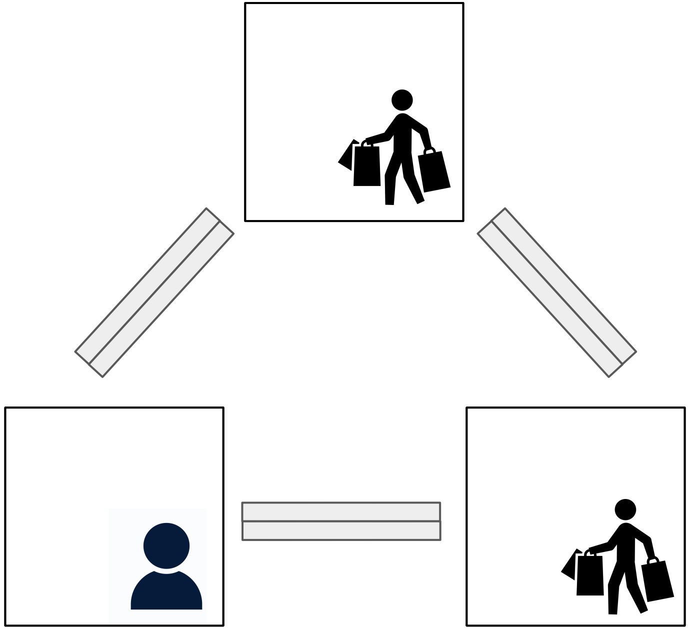
Communicating Labeled Transition System (CLTS) = per participant LTS + peer-to-peer FIFO channels



- Communicating state machines [Brand and Zafiropulo 83]
- Ubiquitous in theory and verification

Implementation model

Communicating Labeled Transition System (CLTS) = **per participant LTS** + peer-to-peer FIFO channels

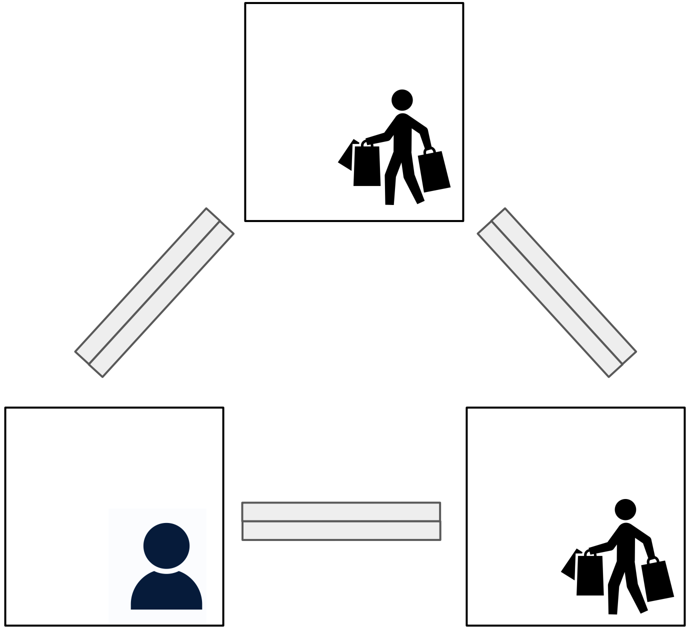


$$b_1 ! s : y \cdot b_1 ! b_2 : y \cdot s ? b_1 : y \cdot s ! b_1 : z \cdot b_2 ? b_1 : y \quad ✓$$

$$b_1 ! s : y \cdot b_1 ! b_2 : y \cdot s ! b_1 : z \cdot s ? b_1 : y \cdot b_2 ? b_1 : y \quad ✗$$

Implementation model

Communicating Labeled Transition System (CLTS) = per participant LTS + **peer-to-peer FIFO channels**



$$b_1 ! s : y \cdot b_1 ! b_2 : y \cdot s ? b_1 : y \cdot b_2 ? b_1 : y \cdot s ! b_1 : z$$

$$b_1 ! s : y \cdot b_1 ! b_2 : y \cdot b_2 ? b_1 : y \cdot s ? b_1 : y \cdot s ! b_1 : z$$

$$b_1 ! s : y \cdot b_1 ! b_2 : y \cdot s ? b_1 : y \cdot s ! b_1 : z \cdot b_2 ? b_1 : y$$

CLTS indistinguishability

A word is **executable** if it is a trace of some* CLTS

Two words are **indistinguishable** if any CLTS that executes one must execute the other

$$b_1 ! s : y \cdot b_1 ! b_2 : y \cdot s ? b_1 : y \cdot b_2 ? b_1 : y \cdot s ! b_1 : z \quad ✓$$

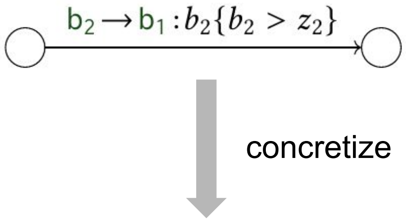
$$b_1 ! s : y \cdot b_1 ! b_2 : y \cdot b_2 ? b_1 : y \cdot s ? b_1 : y \cdot s ! b_1 : z \quad ✓$$

Global protocol semantics must be indistinguishability-closed w.r.t. its target implementation model

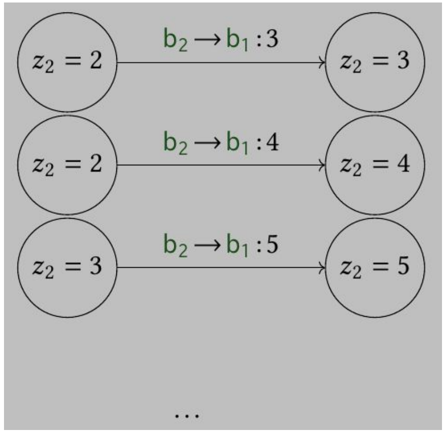
*non-deadlocking

Global protocol semantics

Symbolic transition

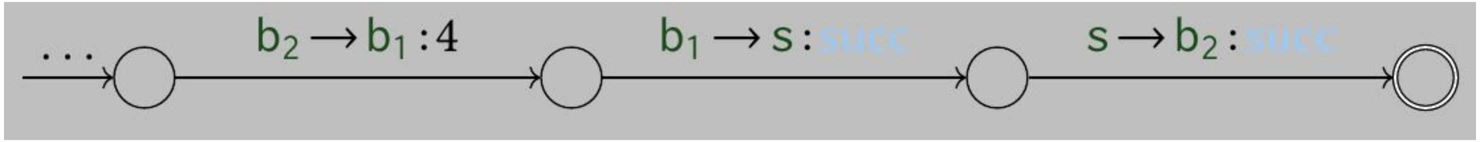


Concrete transition



Global protocol semantics

Concrete run



Synchronous

$$\ldots b_2 \to b_1 : 4 \cdot b_1 \to s : \mathrm{succ} \cdot s \to b_2 : \mathrm{succ}$$

Asynchronous

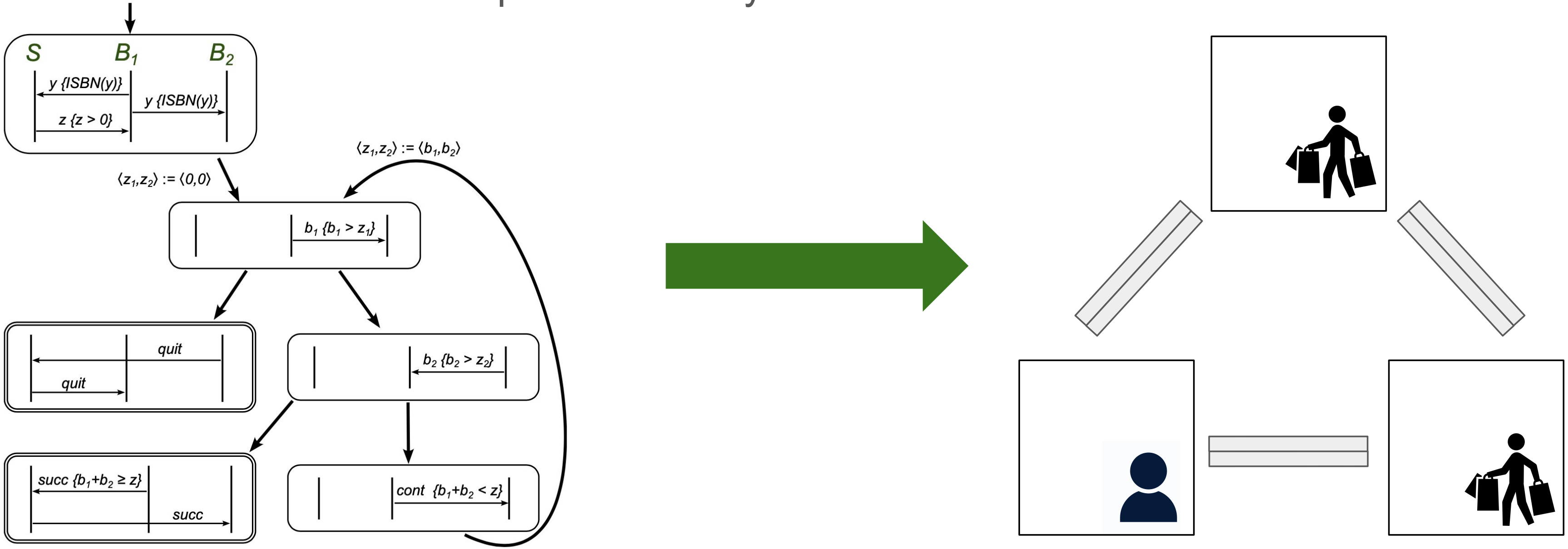
$$\ldots b_2 \,!\, b_1 : 4 \cdot b_1 \,?\, b_2 : 4 \cdot b_2 \,!\, s : \mathrm{succ} \cdot s \,?\, b_2 : \mathrm{succ} \cdot s \,!\, b_2 : \mathrm{succ} \cdot b_2 \,?\, s : \mathrm{succ}$$

…and all words indistinguishable

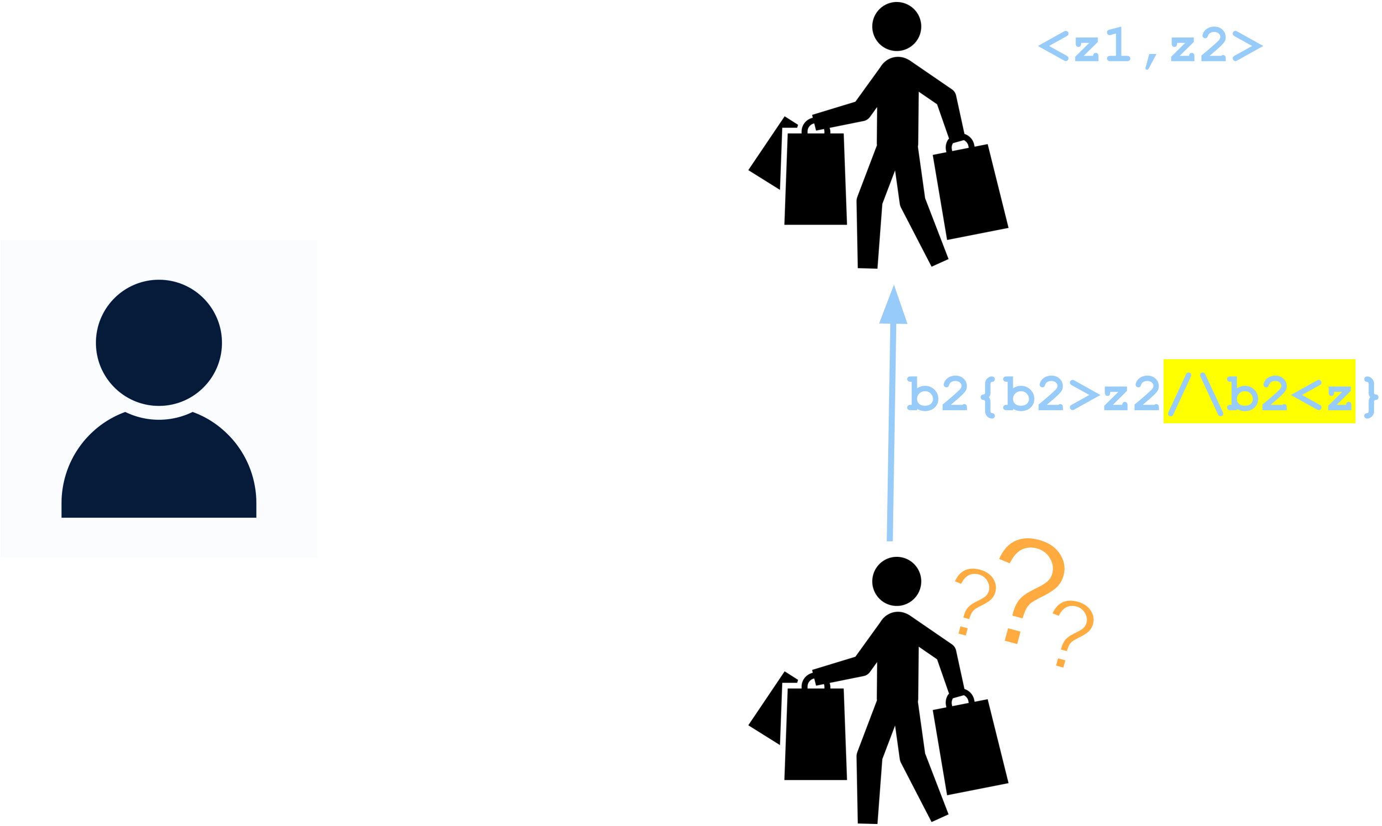
Implementability

Implementability = exists a CLTS satisfying

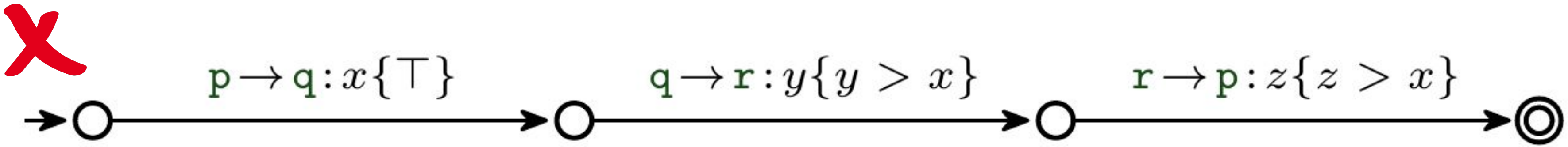
protocol fidelity + deadlock freedom?

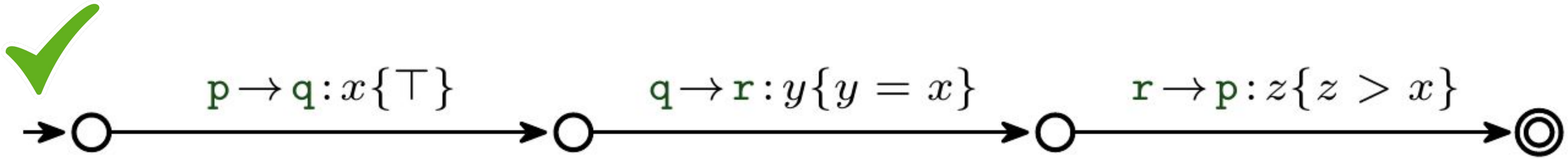


Non-implementability: two-bidder protocol

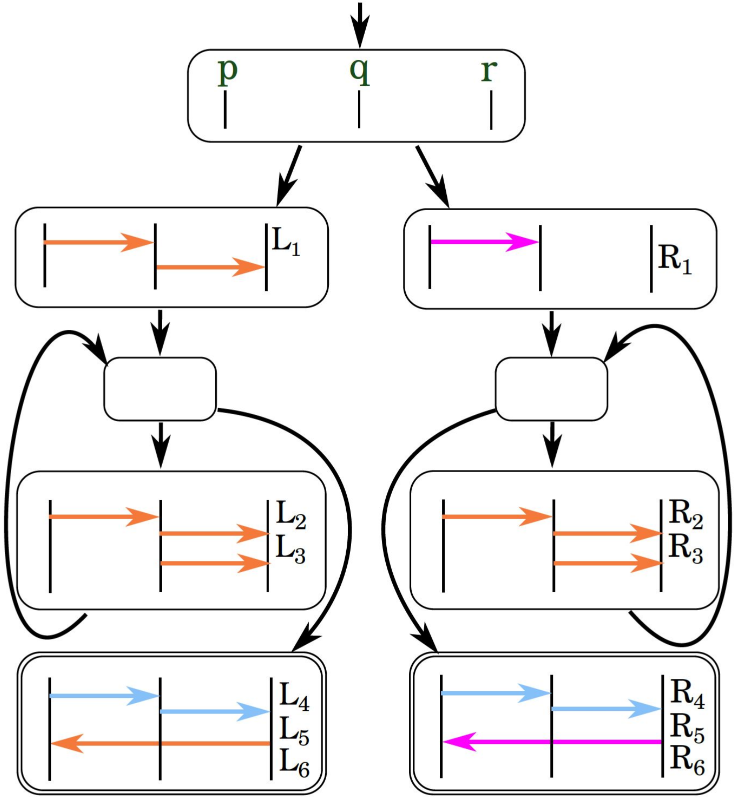


Non-implementability: protocol variables

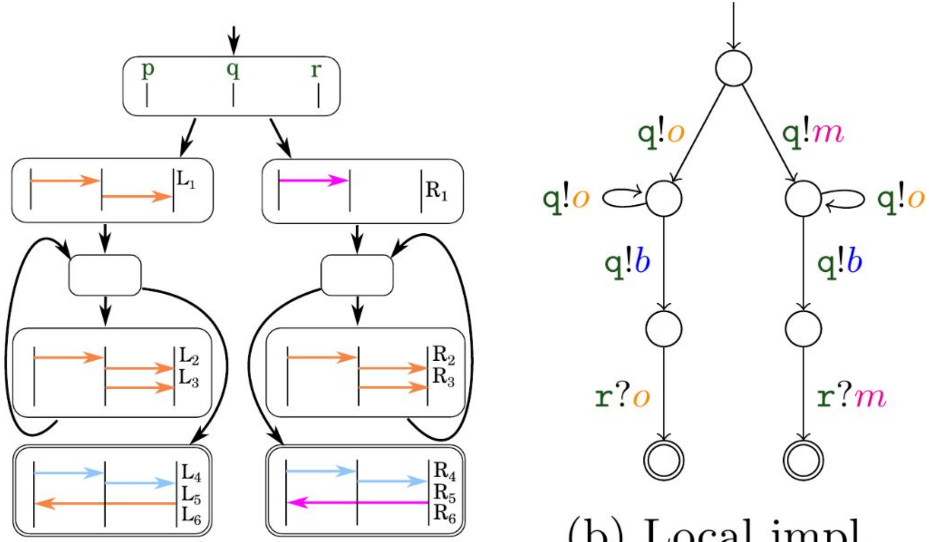




Non-implementability?



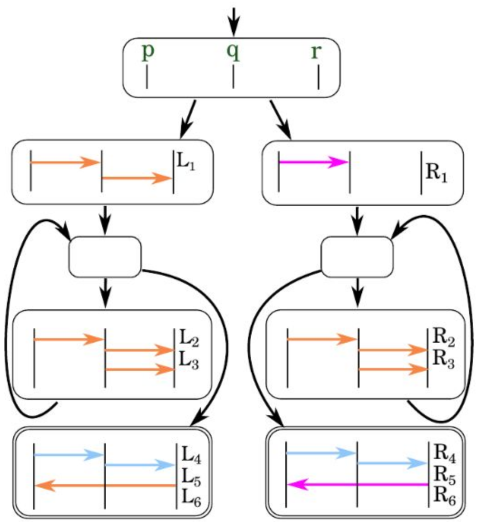
Odd-even example



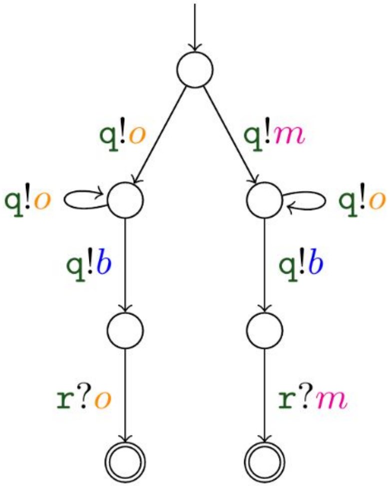
(a) Odd-even protocol

(b) Local impl. for role p

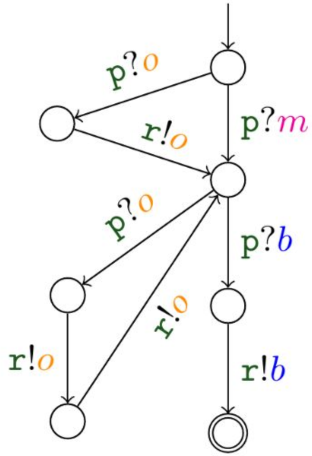
Odd-even example



(a) Odd-even protocol

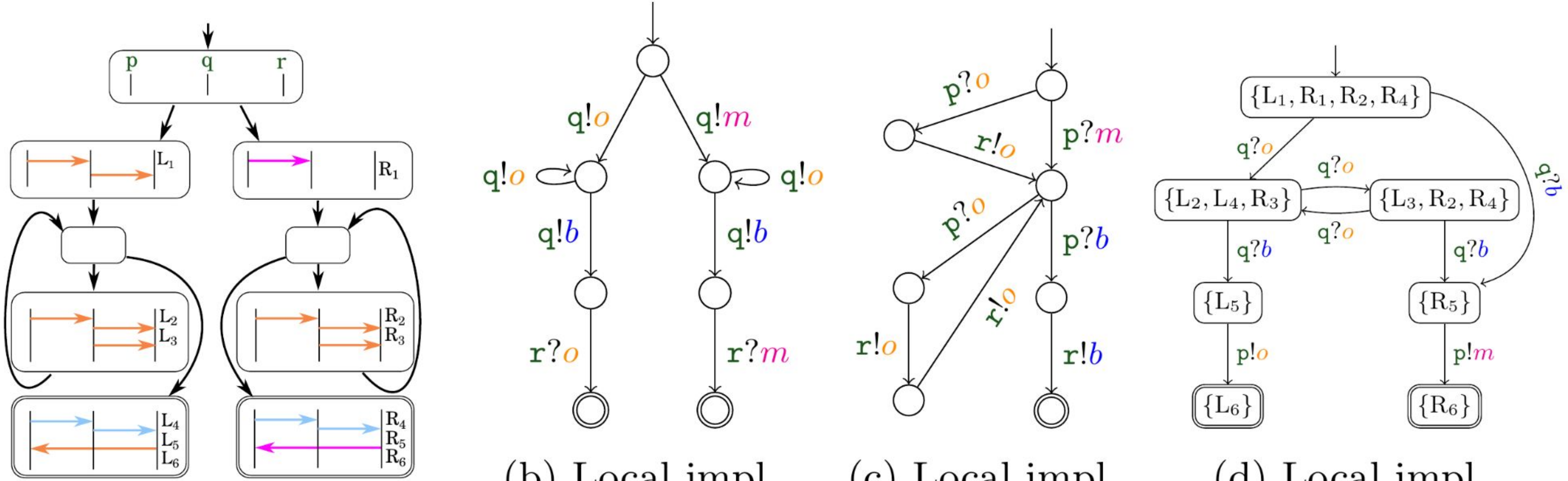


(b) Local impl. for role p



(c) Local impl. for role q

Odd-even example



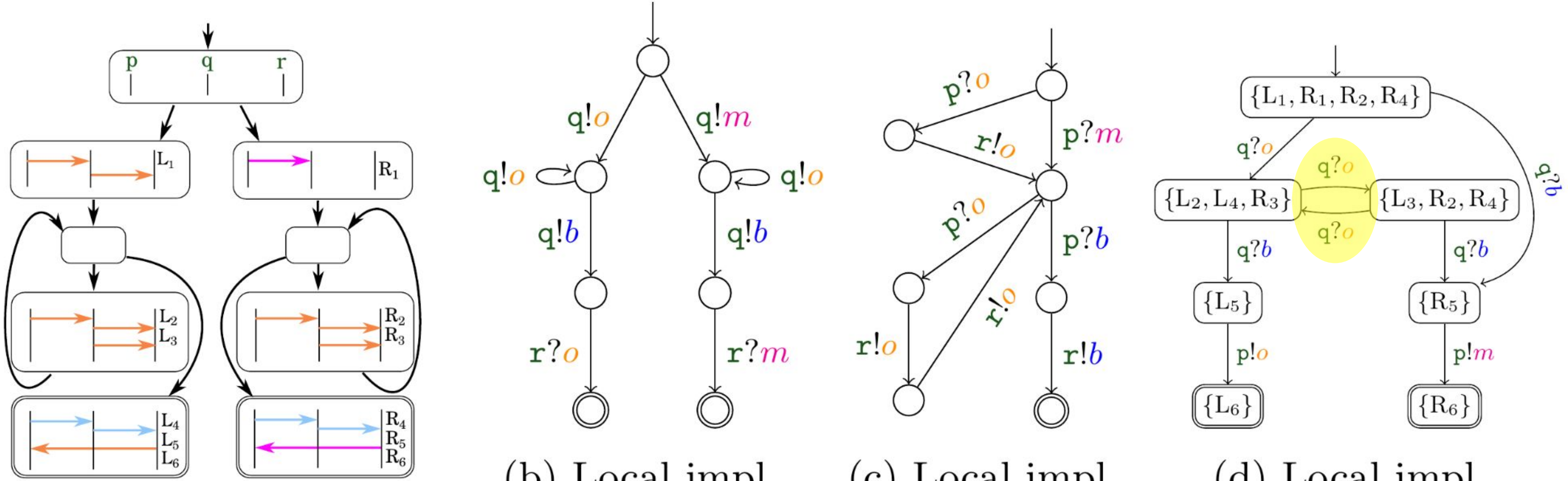
(a) Odd-even protocol

(b) Local impl. for role p

(c) Local impl. for role q

(d) Local impl. for role r

Odd-even example



(a) Odd-even protocol

(b) Local impl. for role p

(c) Local impl. for role q

(d) Local impl. for role r

Flawed results in the literature

FIFO asynchronous and when they are just rendez-vous synchronizations. This property was claimed to be decidable in several conference and journal papers [1, 4, 3, 2] for either mailboxes ($*$-1) or peer-to-peer (1-1) communications, thanks to a form of small model property. In this paper, we show that this small model property does not hold neither for mailbox communications, nor for peer-to-peer communications, therefore the decidability of synchronizability becomes an open question. We close this question for peer-to-peer communications, and we show that synchronizability is actually undecidable. We show that synchronizability is decidable if the topology

[Finkel and Lozes 17]

(b)	Caires and Pérez [2016]; Chen [2015]; Deniélou et al. [2012]; Deniélou and Yoshida [2012]; Toninho and Yoshida [2016]	$\geq$ full	no	flawed	(C2)

[Scalas and Yoshida 19]

Our reduction shows that deciding the $\mathsf{avail}_{\mathsf{p},\mathsf{q},\{\mathsf{q}\}}(m, s)$ predicate for global types is in co-NP, which refutes the polynomial time upper bound claimed in [55]. The proof of Lemma 5.9 can be

[Li et al. 25]

Rocq mechanization

```
Theorem preciseness :
  ∀ {State : Type}
    (S : @LTS SyncAlphabet State),
    GCLTS S →
    (∃ {LocalState : Type},
        @implementable State LocalState S) ↔
      @NMC State S ∧ @RCC State S ∧ @SCC State S.
```

Theorem [Li et al. 25]. A protocol is implementable if and only if it satisfies the Coherence Conditions.

In a nutshell, from two simultaneously reachable global states, a participant can:

- Send a message permissible from both states (SCC)
- Receive a message distinguishing the two states (RCC)

but cannot choose between sending or receiving a message (NMC).

Rocq mechanization

```
Theorem preciseness :
  ∀ {State : Type}
    (S : @LTS SyncAlphabet State),
    GCLTS S →
    (∃ {LocalState : Type},
        @implementable State LocalState S) ↔
      @NMC State S ∧ @RCC State S ∧ @SCC State S.
```

Design choices:

- Unified protocol representation

```
Record LTS {A : Type} {State : Type} :=
  mkLTS {
      transition : State → A → State → ℙ;
      s0 : State;
      final : State → ℙ;
    }.
```

- Formal language-theoretic treatment

Rocq mechanization

```
Theorem preciseness :
  ∀ {State : Type}
    (S : @LTS SyncAlphabet State),
    GCLTS S →
    (∃ {LocalState : Type},
        @implementable State LocalState S) ↔
      @NMC State S ∧ @RCC State S ∧ @SCC State S.
```

Takeaways:

- Generalization to infinite participants
- Sink-finality GCLTS assumption can be conditionally removed
- Infinite word semantics bug

Rocq mechanization

```
Theorem preciseness :
  ∀ {State : Type}
    (S : @LTS SyncAlphabet State),
    GCLTS S →
    (∃ {LocalState : Type},
        @implementable State LocalState S) ↔
      @NMC State S ∧ @RCC State S ∧ @SCC State S.
```

Takeaways:

- Generalization to infinite participants
- Sink-finality GCLTS assumption can be conditionally removed
- **Infinite word semantics bug**

Global semantics must be indistinguishability-closed

Finite words:

$$\mathsf{b}_1\,!\,\mathsf{s}:\mathsf{y}\cdot\mathsf{b}_1\,!\,\mathsf{b}_2:\mathsf{y}\cdot\mathsf{s}\,?\,\mathsf{b}_1:\mathsf{y}\cdot\mathsf{b}_2\,?\,\mathsf{b}_1:\mathsf{y}\cdot\mathsf{s}\,!\,\mathsf{b}_1:\mathsf{z}\ \checkmark$$

$$\mathsf{b}_1\,!\,\mathsf{s}:\mathsf{y}\cdot\mathsf{b}_1\,!\,\mathsf{b}_2:\mathsf{y}\cdot\mathsf{b}_2\,?\,\mathsf{b}_1:\mathsf{y}\cdot\mathsf{s}\,?\,\mathsf{b}_1:\mathsf{y}\cdot\mathsf{s}\,!\,\mathsf{b}_1:\mathsf{z}\ \checkmark$$

Global semantics must be indistinguishability-closed

Infinite words:

$$\mathsf{p}\,!\,\mathsf{q}\!:\!m^n \cdot \mathsf{p}\,!\,\mathsf{q}\!:\!m' \cdot \mathsf{r}\,!\,\mathsf{s}\!:\!m' \qquad \text{for all } n \in \mathbb{N} \quad ✓$$

$$\mathsf{r}\,!\,\mathsf{s}\!:\!m' \cdot \mathsf{p}\,!\,\mathsf{q}\!:\!m^\omega \quad ✓$$

Global semantics must be indistinguishability-closed

Infinite words:

$$\mathtt{p}\,!\,\mathtt{q}:m^n \cdot \mathtt{p}\,!\,\mathtt{q}:m' \cdot \mathtt{r}\,!\,\mathtt{s}:m' \qquad \text{for all } n \in \mathbb{N} \quad ✓$$

$$\mathtt{r}\,!\,\mathtt{s}:m' \cdot \mathtt{p}\,!\,\mathtt{q}:m^\omega \quad ✓$$

"there exists a global run for every prefix of an infinite word" [Majumdar et al. 22]

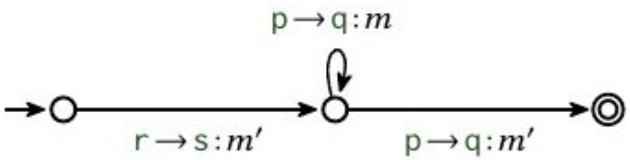
Global semantics must be indistinguishability-closed

Infinite words:

$\mathsf{p}\,!\,\mathsf{q}\!:\!m^n \cdot \mathsf{p}\,!\,\mathsf{q}\!:\!m' \cdot \mathsf{r}\,!\,\mathsf{s}\!:\!m'$ for all $n \in \mathbb{N}$ ✓

$\mathsf{r}\,!\,\mathsf{s}\!:\!m' \cdot \mathsf{p}\,!\,\mathsf{q}\!:\!m^\omega$ ✓

"there exists a global run for every prefix of an infinite word" [Majumdar et al. 22]



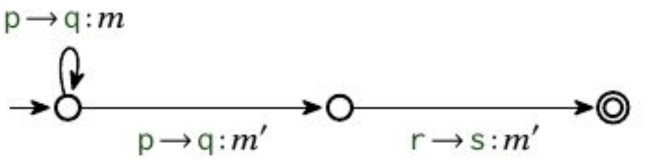
Global semantics must be indistinguishability-closed

Infinite words:

$$\mathsf{p}\,!\,\mathsf{q}:m^n \cdot \mathsf{p}\,!\,\mathsf{q}:m' \cdot \mathsf{r}\,!\,\mathsf{s}:m' \qquad \text{for all } n \in \mathbb{N}$$

$$\mathsf{r}\,!\,\mathsf{s}:m' \cdot \mathsf{p}\,!\,\mathsf{q}:m^\omega$$

"there exists a global run for every prefix of an infinite word" [Majumdar et al. 22]



Global semantics must be indistinguishability-closed

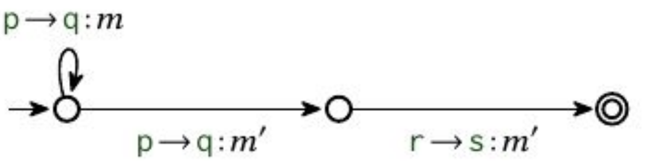
Infinite words:

$\mathsf{p}\,!\,\mathsf{q}:m^n \cdot \mathsf{p}\,!\,\mathsf{q}:m' \cdot \mathsf{r}\,!\,\mathsf{s}:m'$ for all $n \in \mathbb{N}$ ✓

$\mathsf{r}\,!\,\mathsf{s}:m' \cdot \mathsf{p}\,!\,\mathsf{q}:m^\omega$ ✓

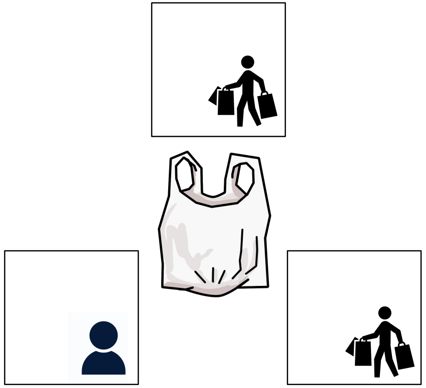
"there exists a global run for every prefix of an infinite word" [Majumdar et al. 22]

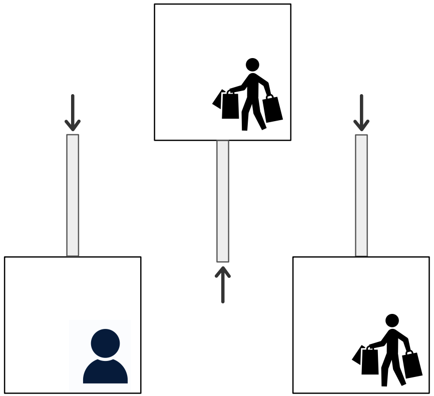
"for every prefix of an infinite word there exists a global run"

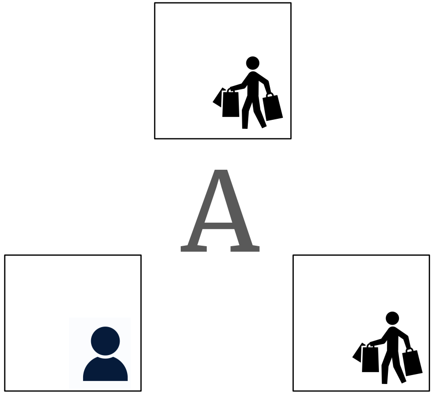


Future work and extensions

- Modeling and proving implementability of existing protocol formalisms [Castro-Perez et al. 21, Hirsch and Garg 22]
- Synthesis of certified implementations
- Investigating different network models

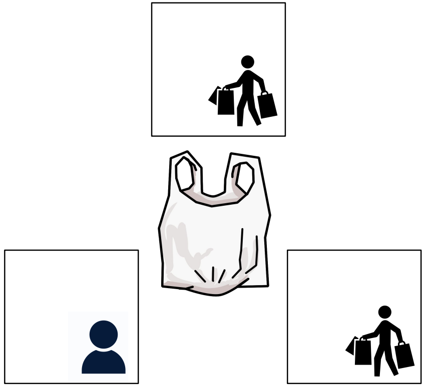


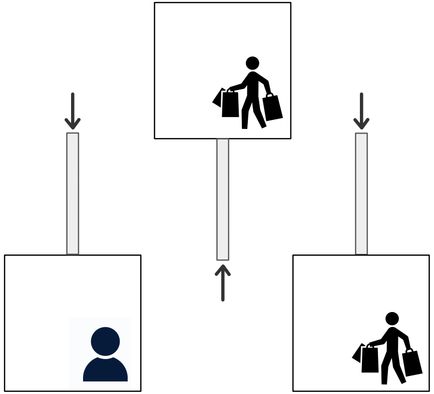


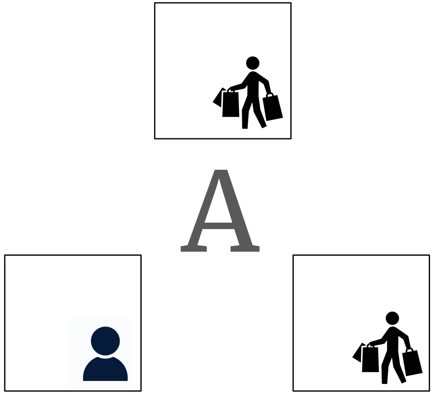


Future work and extensions

- Modeling and proving implementability of existing protocol formalisms [Castro-Perez et al. 21, Hirsch and Garg 22]
- Synthesis of certified implementations
- Investigating different network models

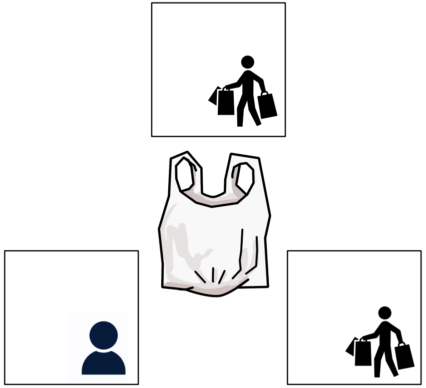


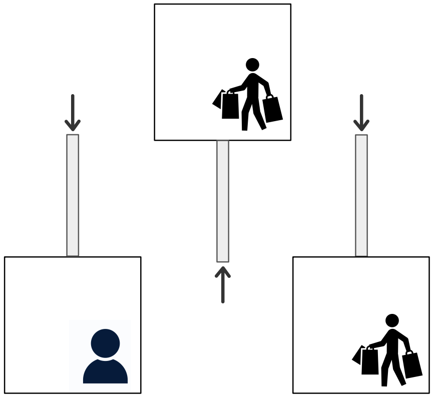


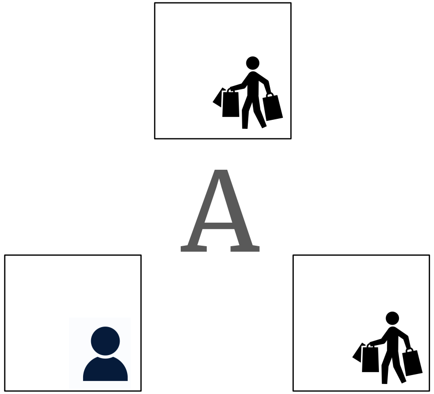


Future work and extensions

- Modeling and proving implementability of existing protocol formalisms [Castro-Perez et al. 21, Hirsch and Garg 22]
- Synthesis of certified implementations
- Investigating different network models

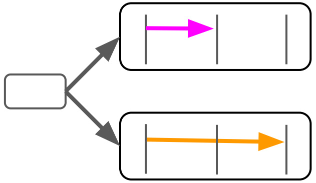




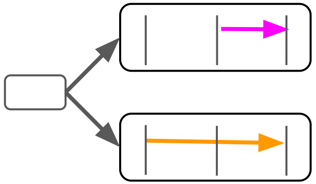


GCLTS assumptions

1) Deadlock-free
2) Deterministic
3) Sender-driven choice



decidability



4) Sink-finality: no final states with outgoing transitions

cf. "local acceptance" for Zielonka automata, only required for finite CLTS semantics