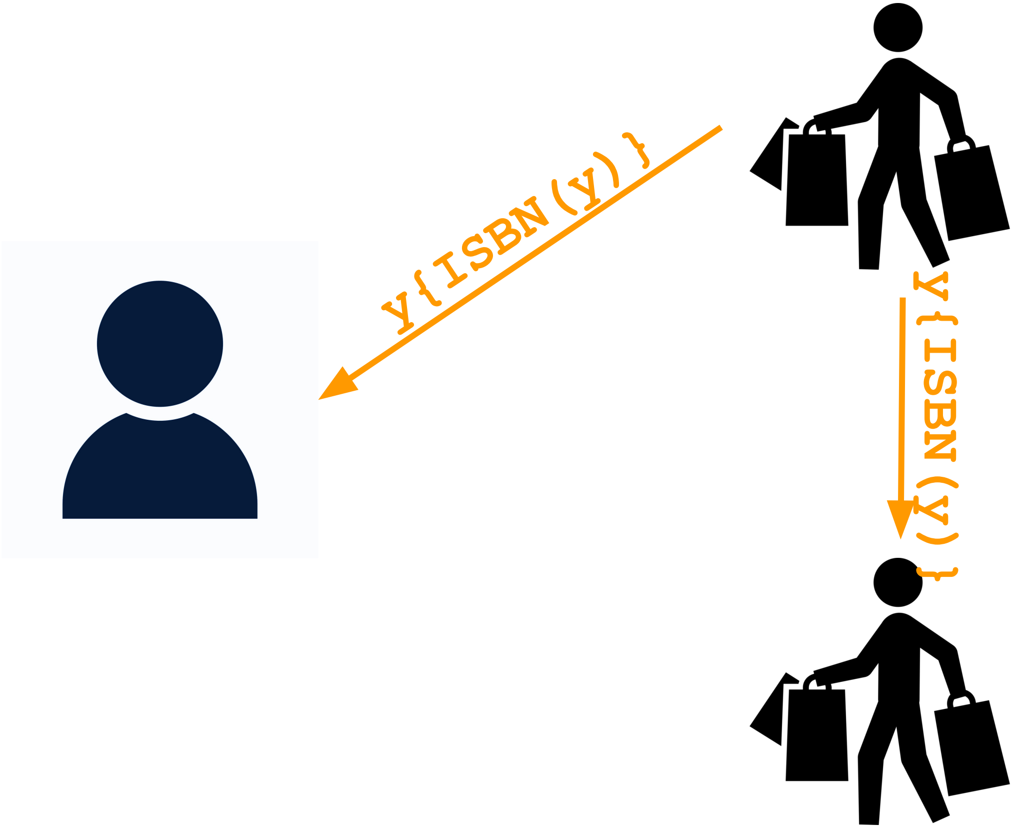
Characterizing Implementability of Global Protocols with Infinite States and Data

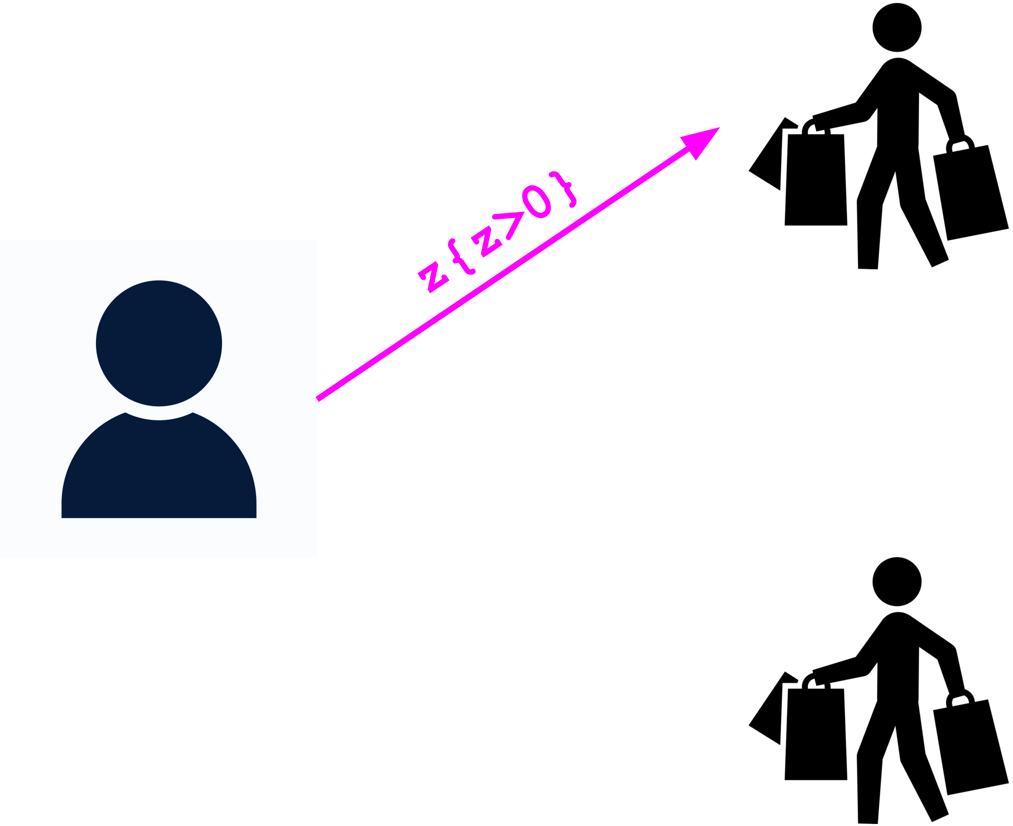
Elaine Li   Felix Stutz   Thomas Wies   Damien Zufferey

NYU   uni.lu   NYU   NVIDIA

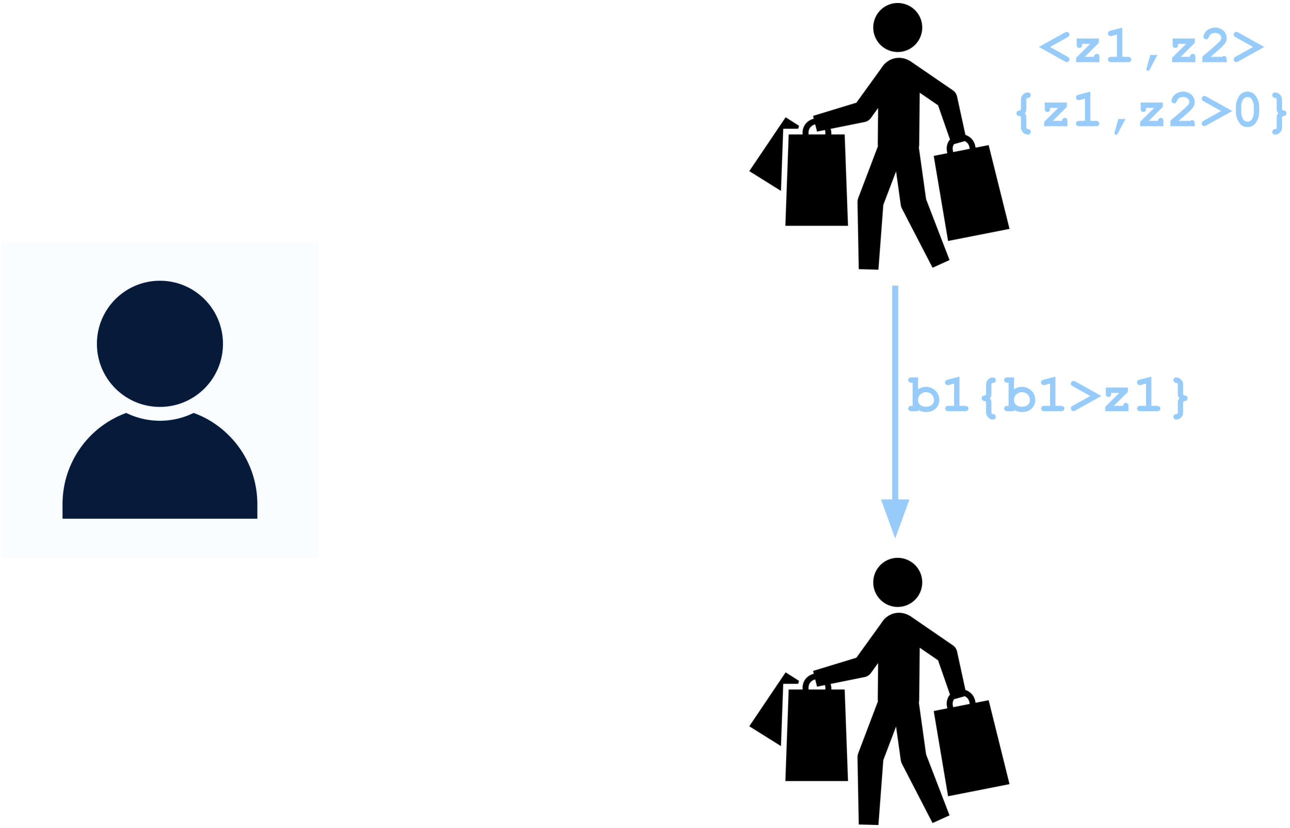
Global protocols: two-bidder protocol



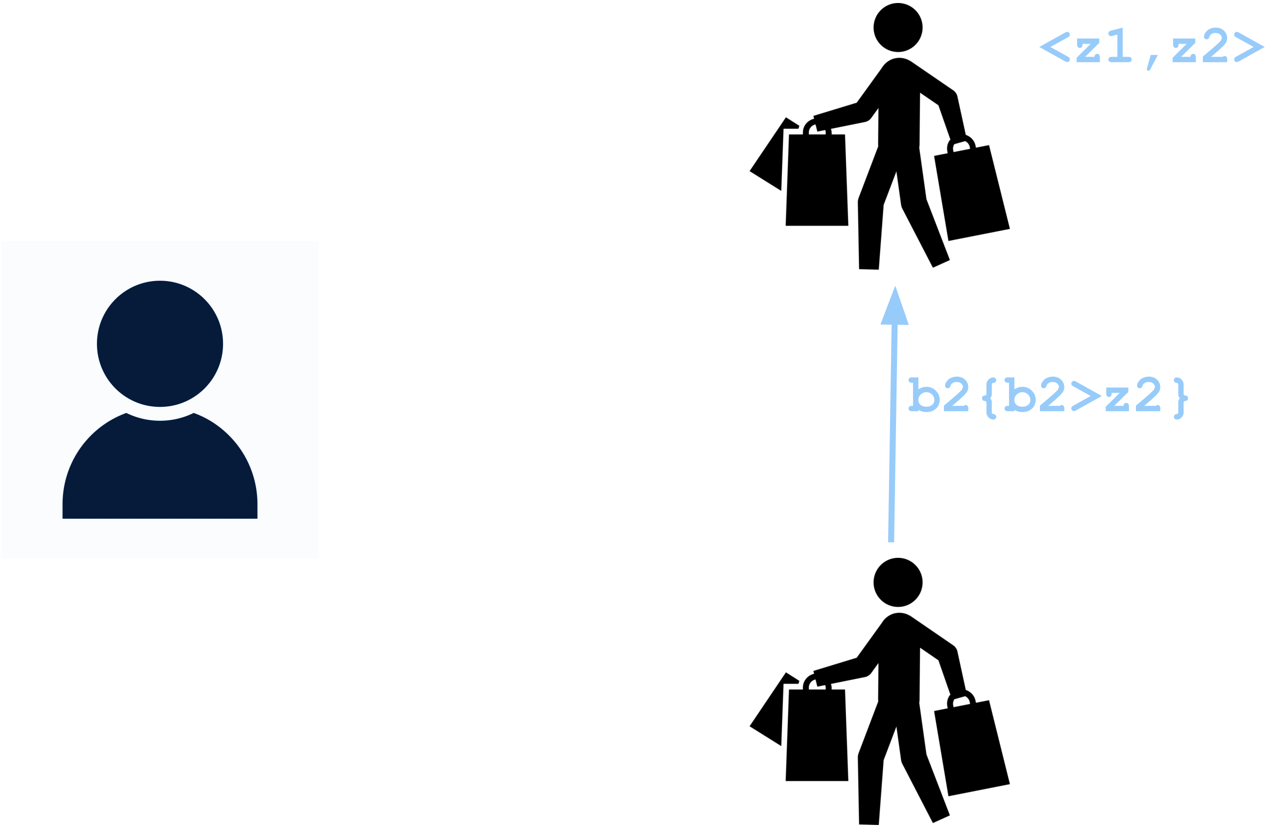
Global protocols: two-bidder protocol



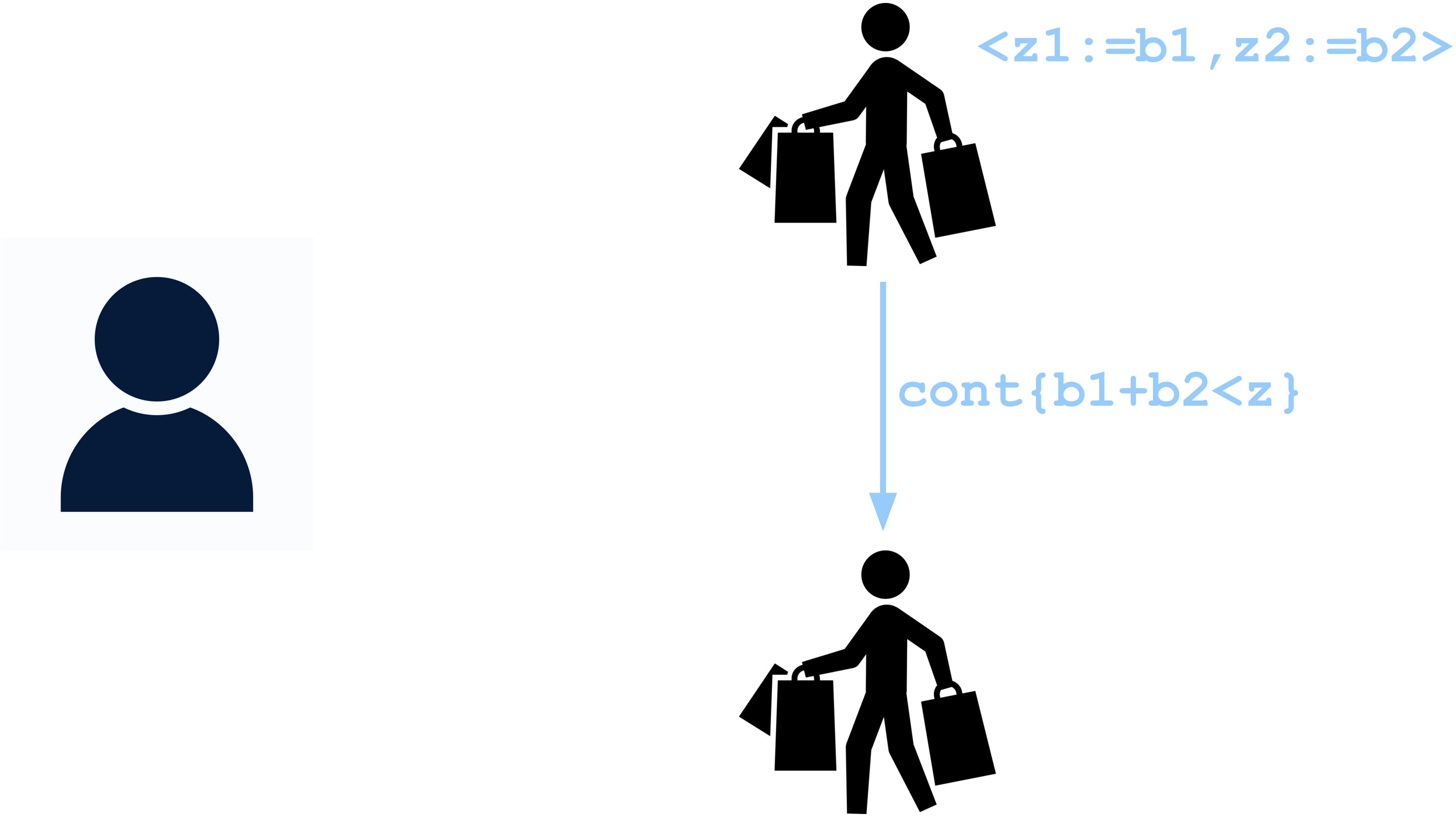
Global protocols: two-bidder protocol



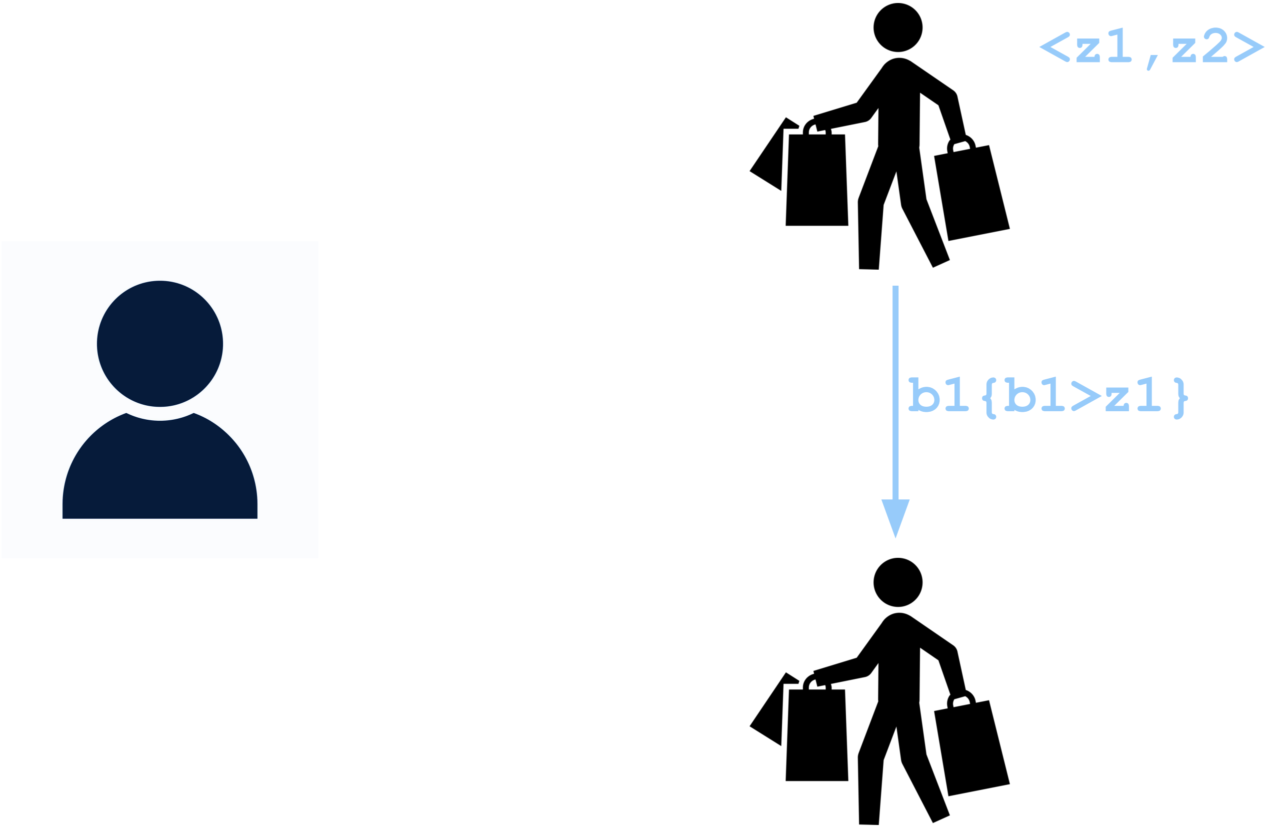
Global protocols: two-bidder protocol



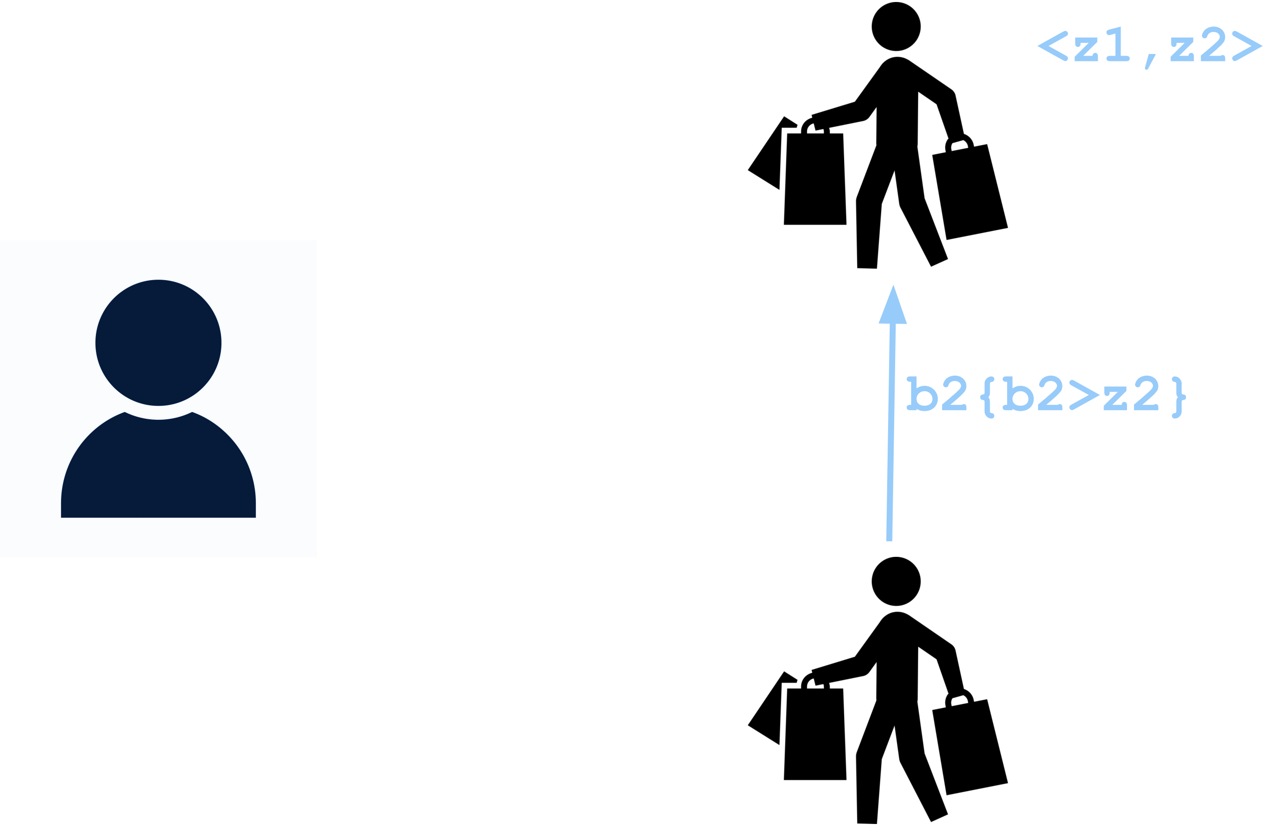
Global protocols: two-bidder protocol



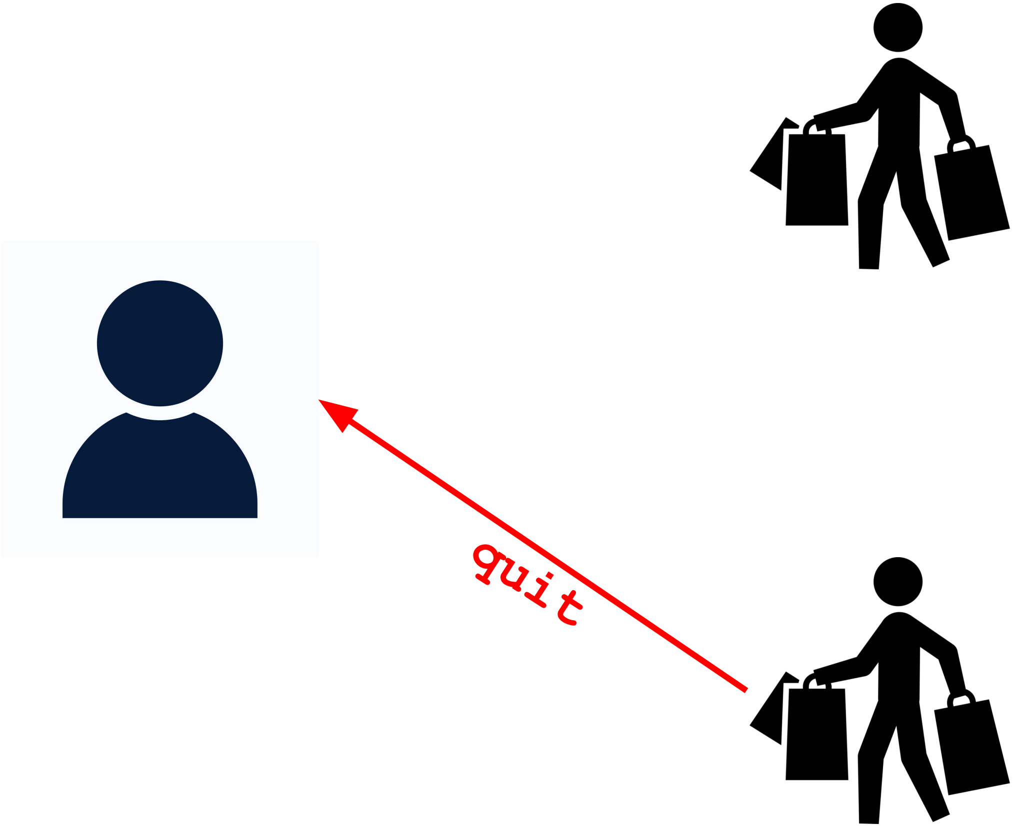
Global protocols: two-bidder protocol



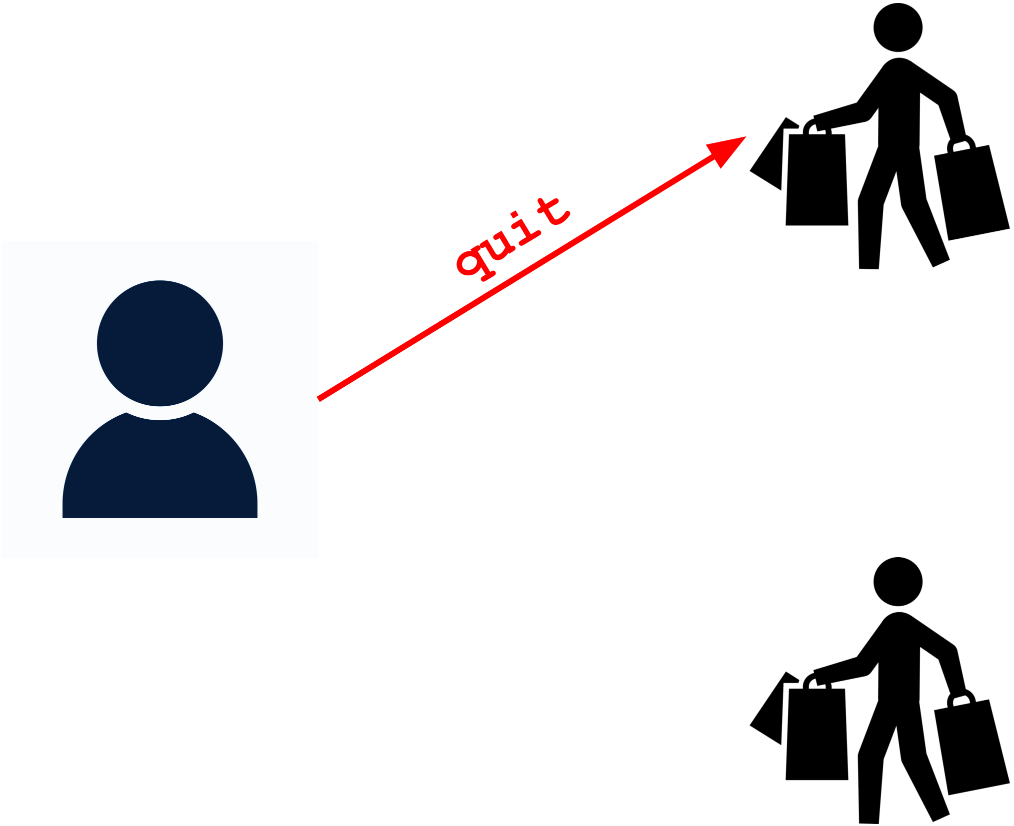
Global protocols: two-bidder protocol



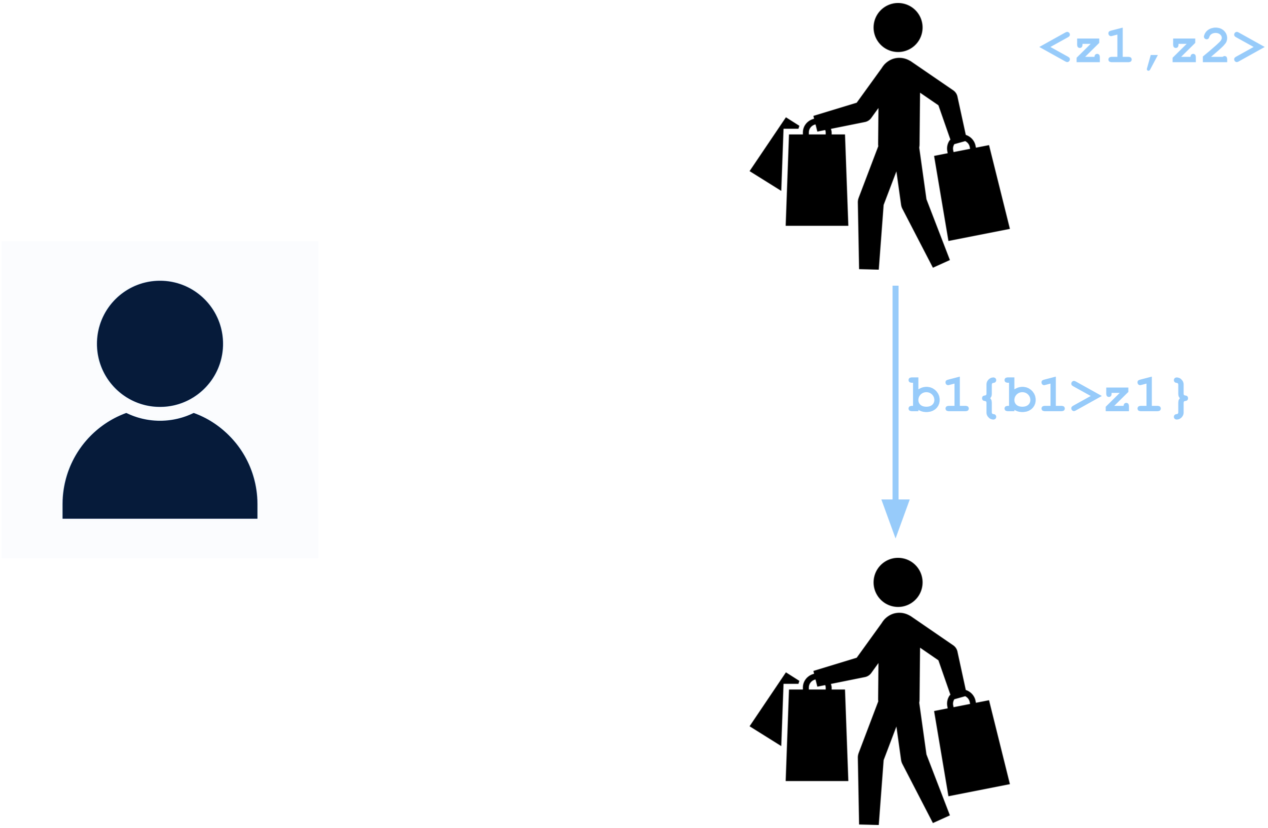
Global protocols: two-bidder protocol



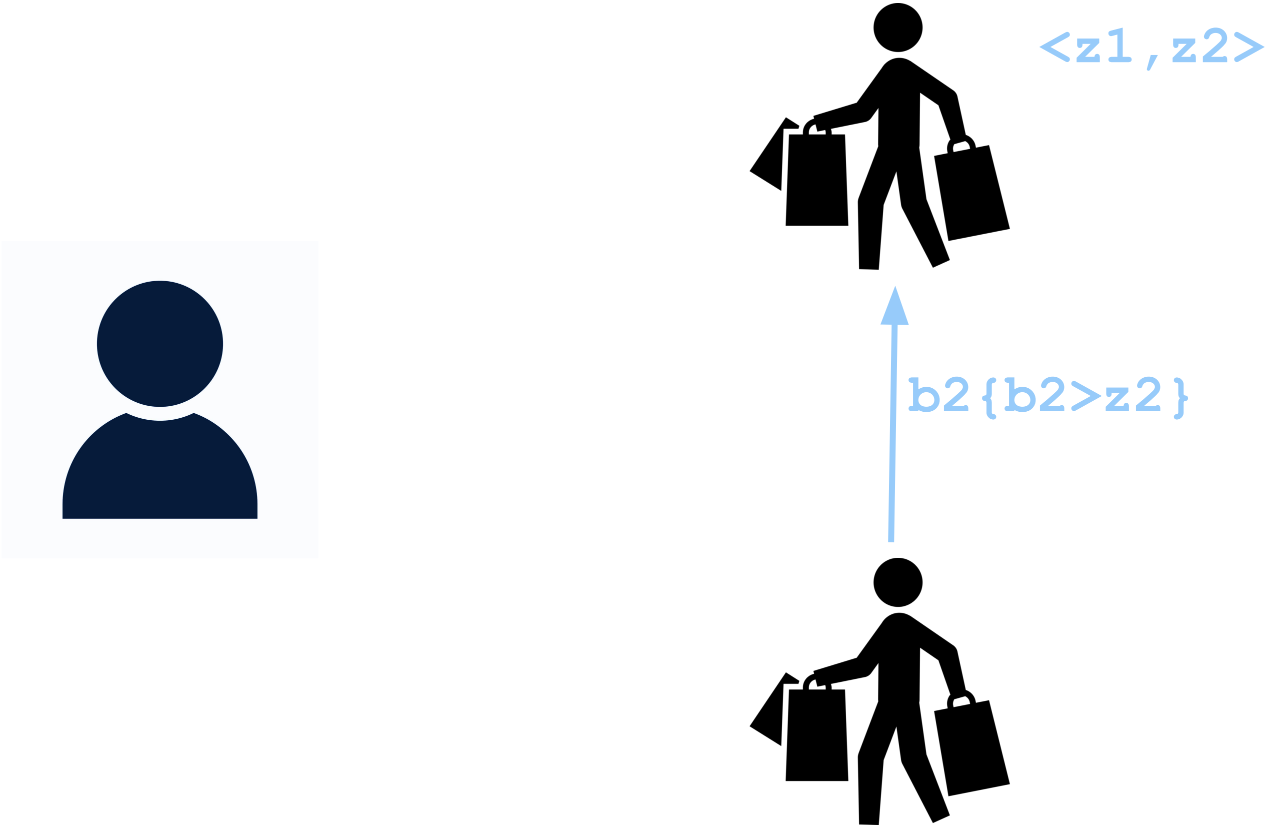
Global protocols: two-bidder protocol



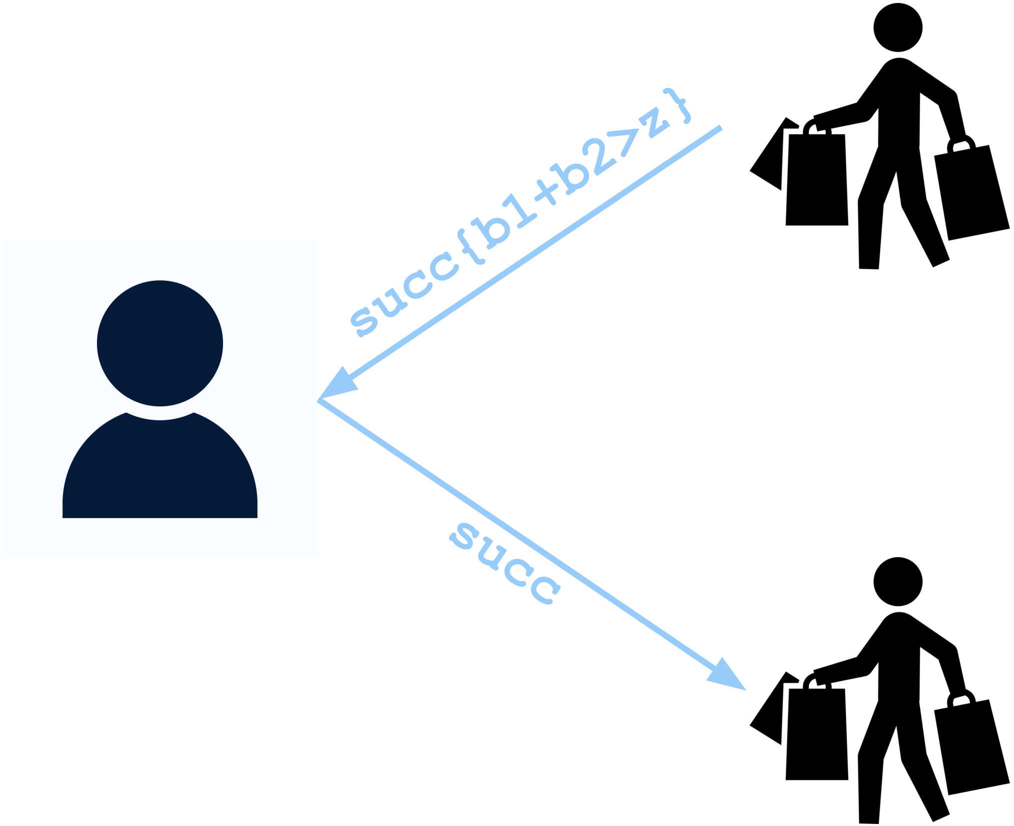
Global protocols: two-bidder protocol



Global protocols: two-bidder protocol



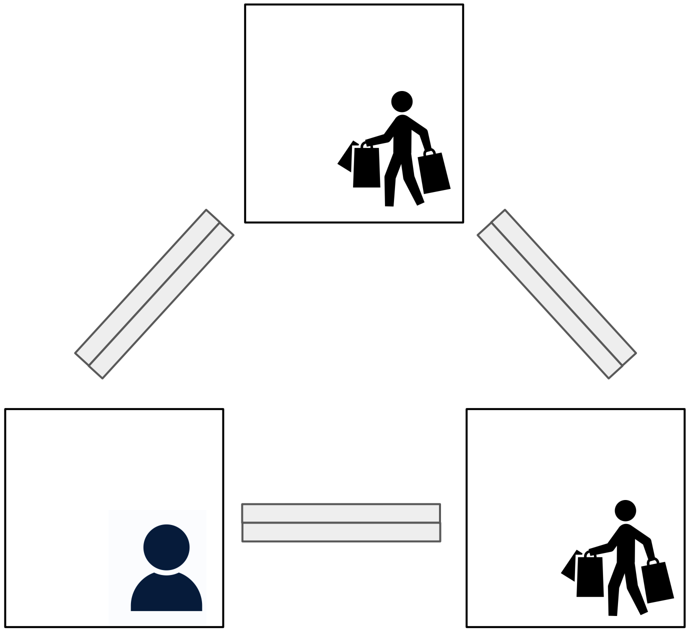
Global protocols: two-bidder protocol



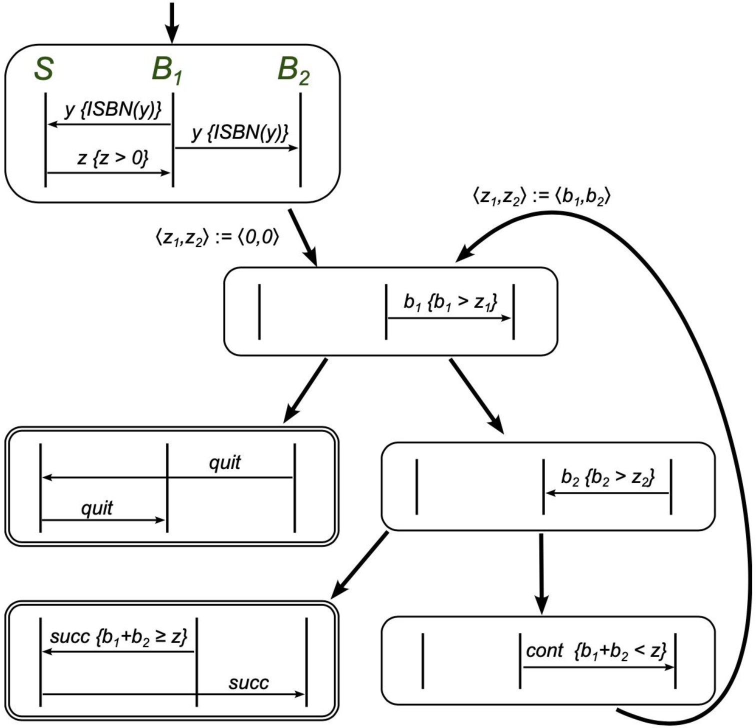
Overview

Asynchronous, message-passing programs are challenging to implement individually

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



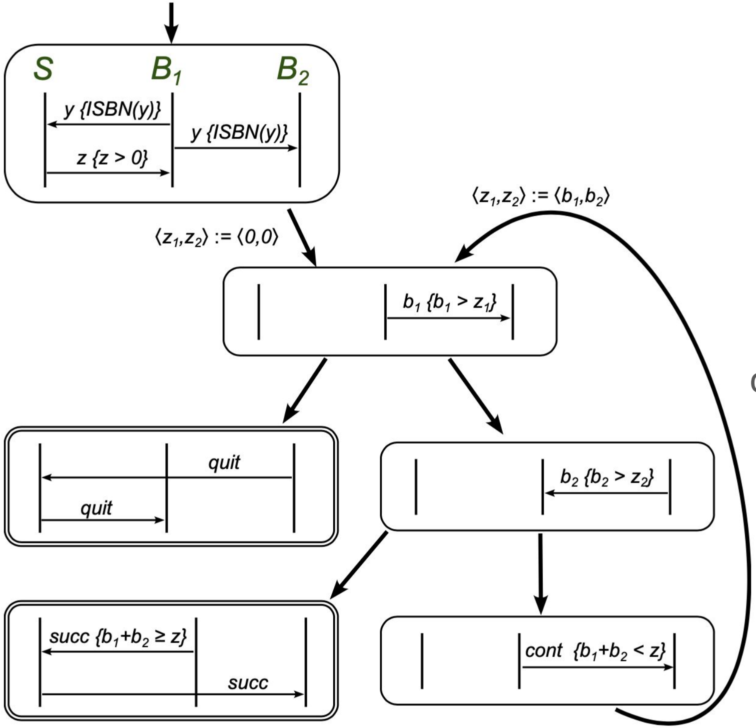
Overview



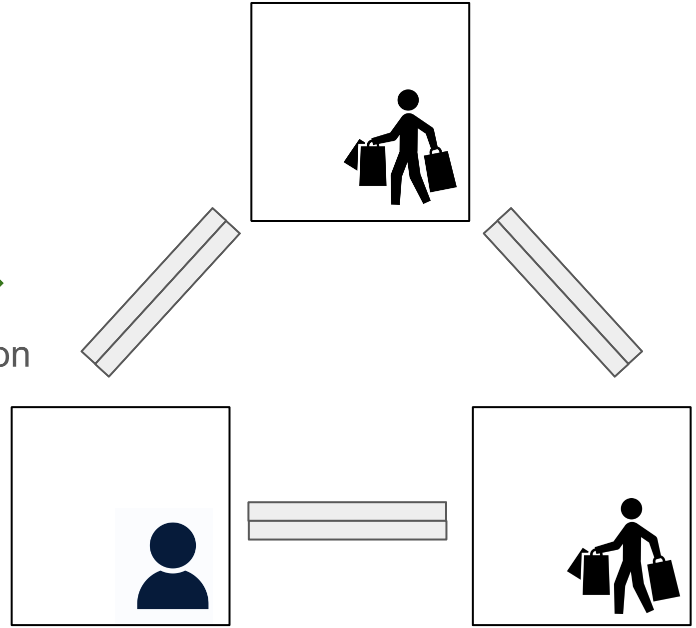
Global protocols are **synchronous** specifications of **all** participants' behaviors

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

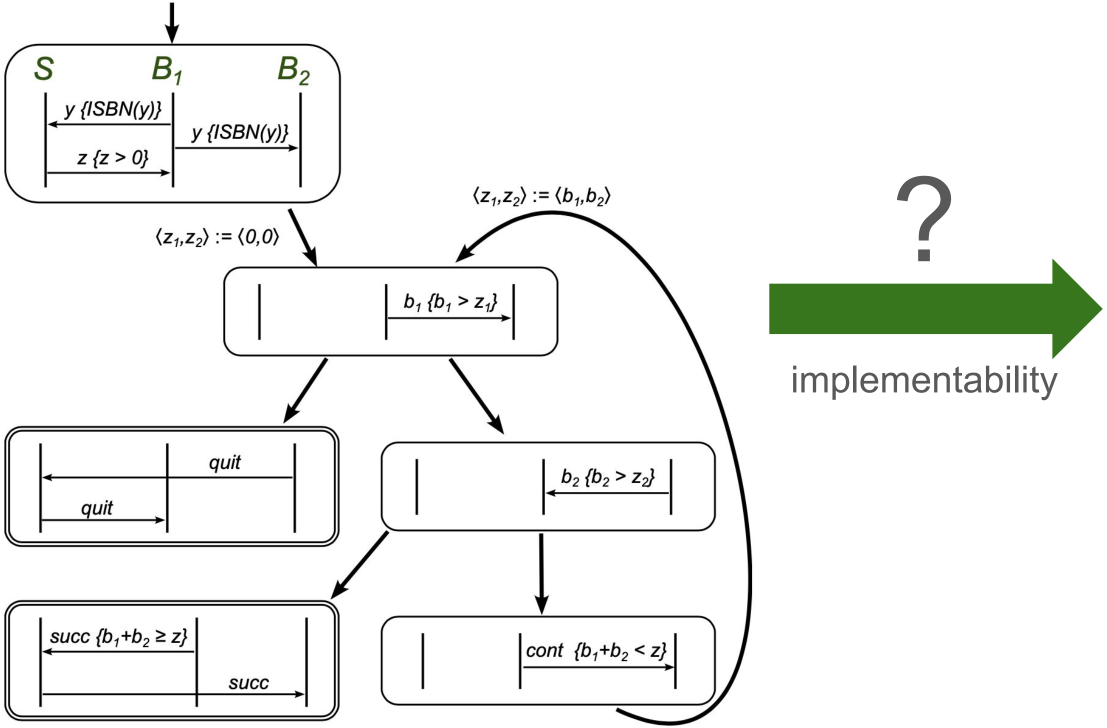
Overview



correct-by-construction synthesis

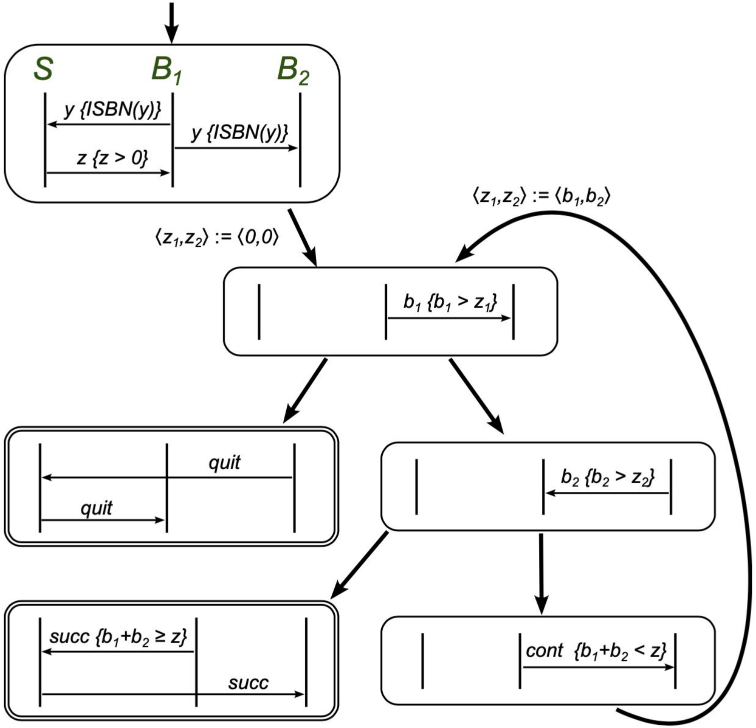


Overview



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}
?
implementability

Overview



implementability

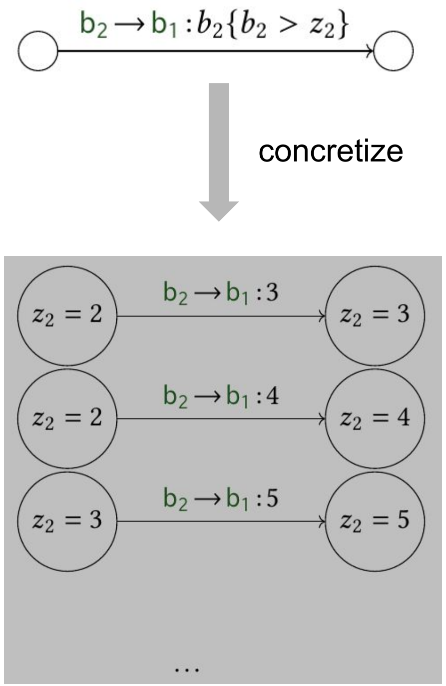
≈

$L_1 = L_2$?

Global protocol semantics

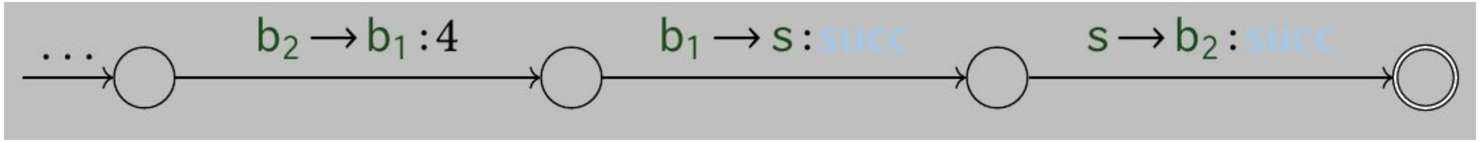
Symbolic transition

Concrete transition



Global protocol semantics

Concrete run



Synchronous

$\ldots b_2 \to b_1 : 4 \cdot b_1 \to s : succ \cdot s \to b_2 : succ$

Asynchronous

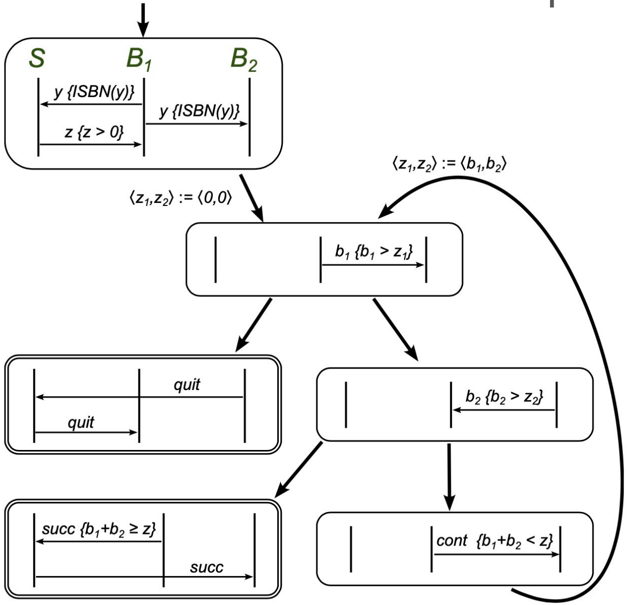
$\ldots b_2 \, ! \, b_1 : 4 \cdot b_1 \, ? \, b_2 : 4 \cdot b_2 \, ! \, s : succ \cdot s \, ? \, b_2 : succ \cdot s \, ! \, b_2 : succ \cdot b_2 \, ? \, s : succ$

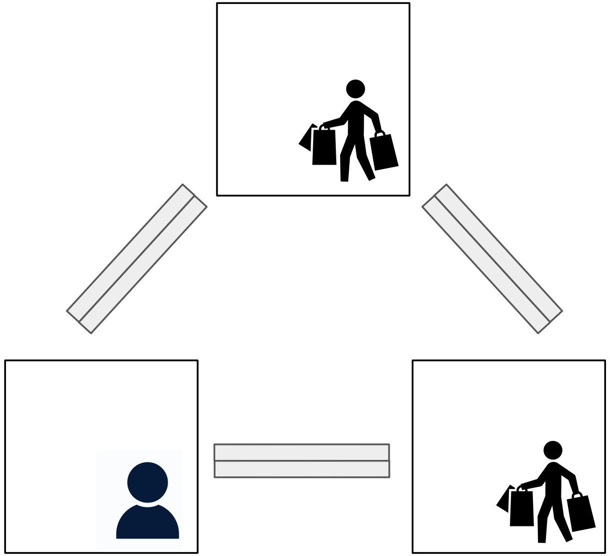
…closed under CLTS-equivalence

Implementability

Implementability = exists a CLTS satisfying

protocol fidelity + deadlock freedom?

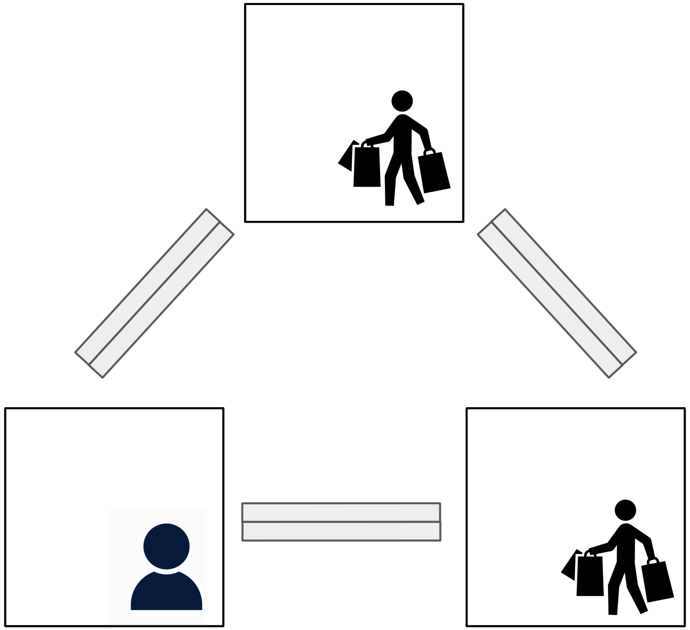




Implementation model

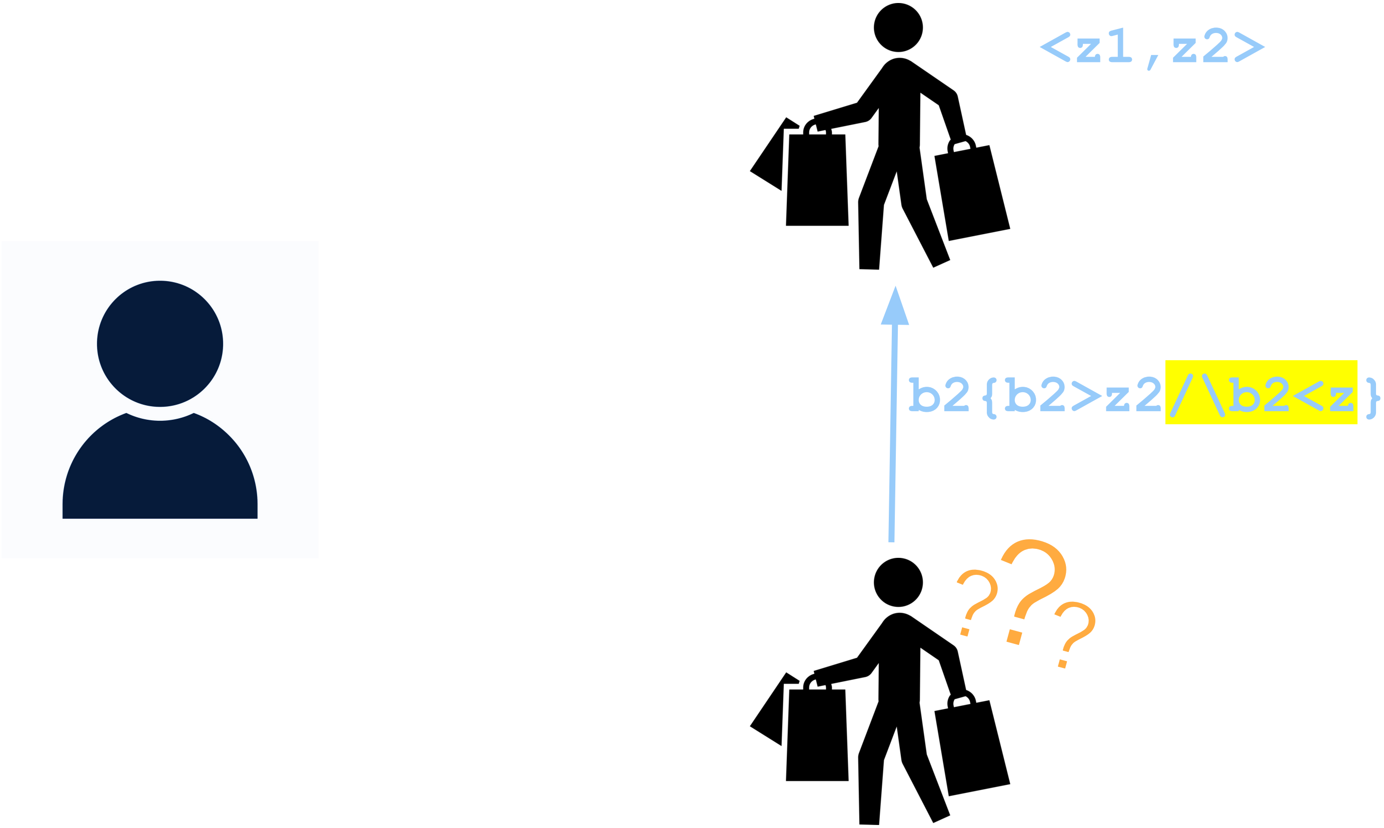
(controllable) (non-controllable)

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

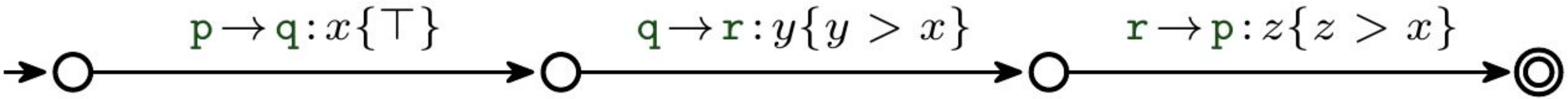


- Infinite-state generalization of CSMs [Brand and Zafiropulo 83]
- Message delay, reordering, ~~dropping, duplication~~

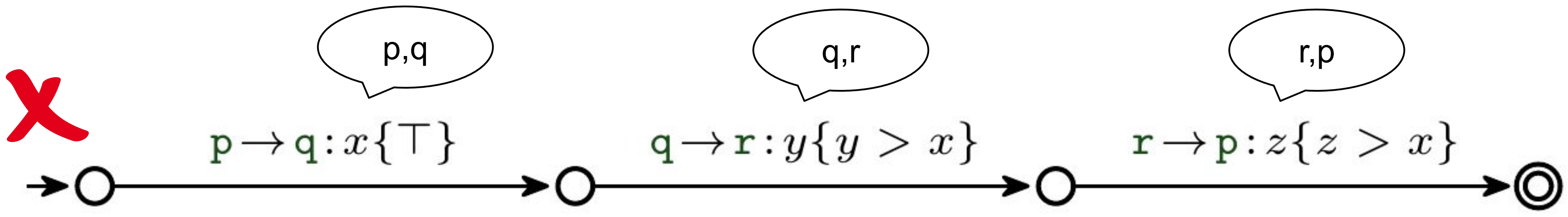
Non-implementability: two-bidder protocol



Non-implementability

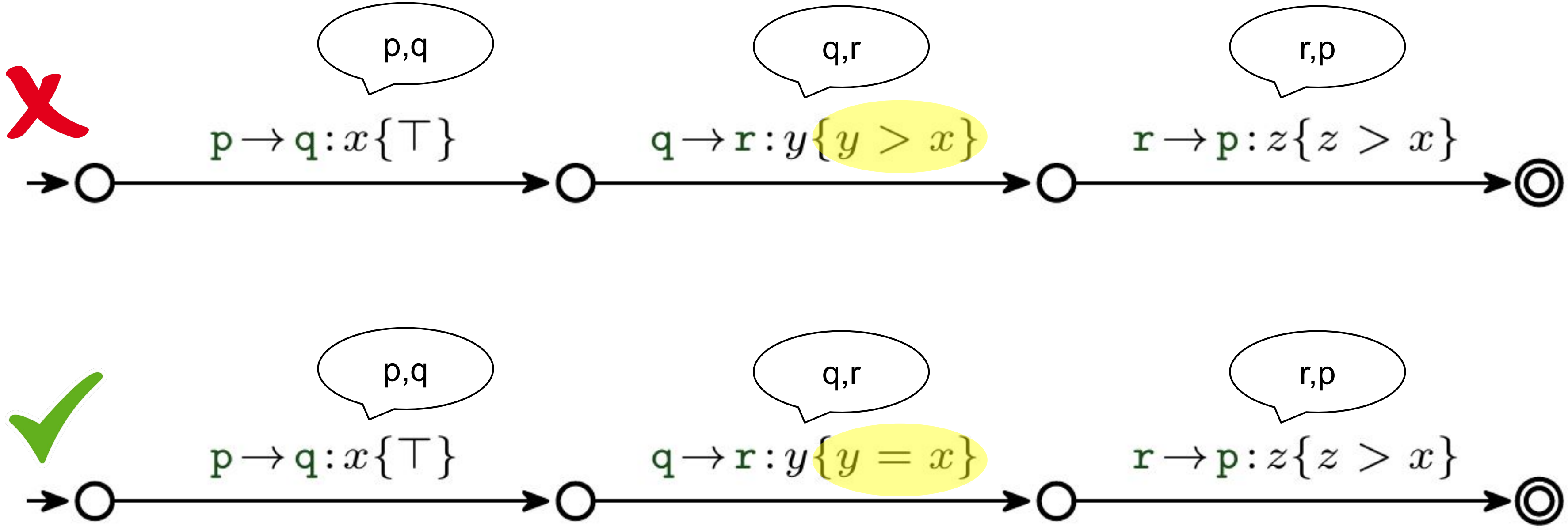


Non-implementability

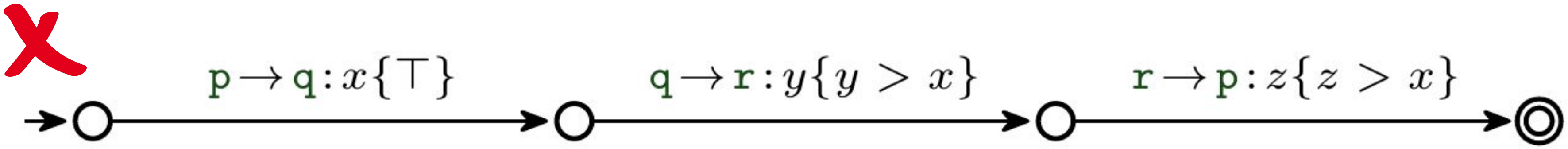


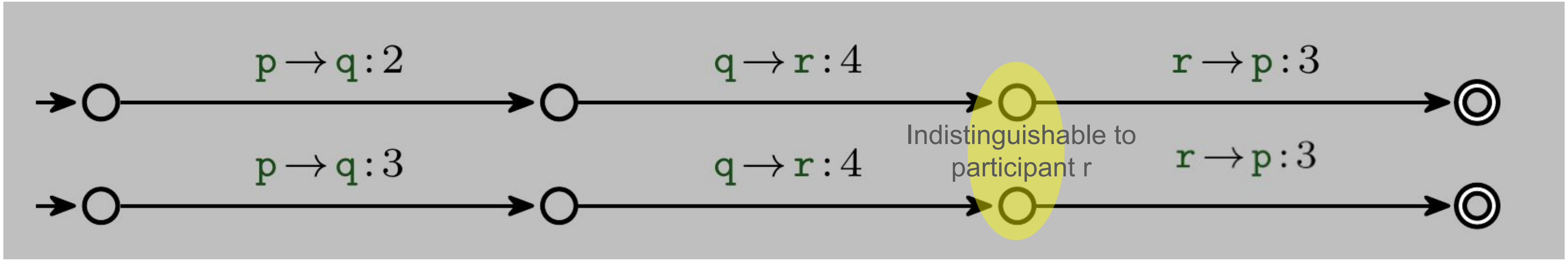
Syntactic classification of "known" and "unknown" variables [Zhou et al. 20]

Non-implementability



Non-implementability





Coherence Conditions

From two locally indistinguishable global states, a participant can either:

- Send a message permissible from both states (Send Coherence)
- Receive a message that distinguishes the two states (Receive Coherence)

but cannot choose between sending or receiving a message (No Mixed Choice)

Coherence Conditions

From two locally indistinguishable global states, a participant can either:

- Send a message permissible from both states (Send Coherence)
- Receive a message that distinguishes the two states (Receive Coherence)

but cannot choose between sending or receiving a message (No Mixed Choice)

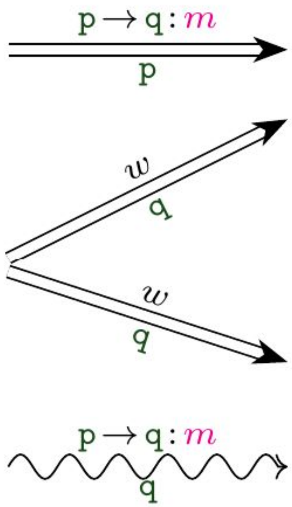
Theorem. A concrete protocol* is implementable if and only if it satisfies the Coherence Conditions.

Deciding implementability, part one

Implementability

(co-)reachability reduction

Coherence Conditions



Algorithm for finite protocols

Algorithm 1 Check CC for finite protocols

▷ *Let LTS* $\mathcal{S} = (S, \Gamma, T, s_0, F)$

▷ *Checking Send Coherence*

for $s_1 \xrightarrow{p \to q:m} s_2 \in T$ **do**

  for $s \neq s_1 \in S$ **do**

    if $\mathcal{L}(S, \Gamma_p \uplus \{\varepsilon\}, T_p, s_0, \{s\}) \cap \mathcal{L}(S, \Gamma_p \uplus \{\varepsilon\}, T_p, s_0, \{s_1\}) \neq \emptyset$ **then**

      $b \leftarrow \bot$

      for $s_3 \xrightarrow{p \to q:m} s_4 \in T$ **do** $b \leftarrow b \vee \left(s \overset{\varepsilon}{\underset{p}{\Rightarrow}}{}^{*} s_3 \right)$

      if $\neg b$ **then return** $\bot$

▷ *Checking Receive Coherence*

for $s_1 \xrightarrow{p \to q:m} s_2, s_3 \xrightarrow{r \to q:m} s_4 \in T, s_1 \neq s_2, p \neq r$ **do**

  if $\mathcal{L}(S, \Gamma_q \uplus \{\varepsilon\}, T_q, s_0, \{s_1\}) \cap \mathcal{L}(S, \Gamma_q \uplus \{\varepsilon\}, T_q, s_0, \{s_3\}) \neq \emptyset$ **then**

    if $\text{avail}_{p,q,\{q\}}(m, s_4)$ **then return** $\bot$

▷ *Checking No Mixed Choice*

for $s_1 \xrightarrow{p \to q:m} s_2, s_3 \xrightarrow{r \to p:m} s_4 \in T, s_1 \neq s_2$ **do**

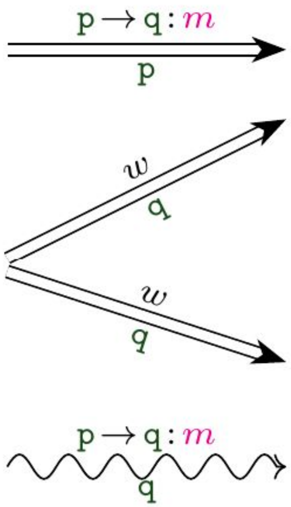
  if $\mathcal{L}(S, \Gamma_q \uplus \{\varepsilon\}, T_q, s_0, \{s_1\}) \cap \mathcal{L}(S, \Gamma_q \uplus \{\varepsilon\}, T_q, s_0, \{s_3\}) \neq \emptyset$ **then return** $\bot$

return $\top$

Deciding implementability, part two

Symbolic Coherence Conditions

Coherence Conditions



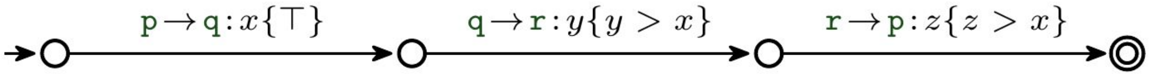
µCLP reduction

(first-order fixpoint logic modulo theories)

$$\mathsf{unreach}^{\varepsilon}_{p,q}(s,\boldsymbol{r},x) :=_\nu$$
$$\wedge \Big(\bigwedge_{(s,p\to q:y\{\varphi\},s')\in\Delta} \neg\varphi[x/y]\Big)$$
$$\wedge \Big(\bigwedge_{\substack{(s,q\to t:y\{\varphi\},s')\in\Delta \\ p\neq q\wedge p\neq t}} \forall y.\,\varphi \Rightarrow \mathsf{unreach}^{\varepsilon}_{p,q}(s',\boldsymbol{r}',x)\Big)\ .$$

$$\mathsf{avail}_{p,q,\mathcal{B}}(x_1,s,\boldsymbol{r}) :=_\mu$$
$$\bigvee_{\substack{(s,r\to t:x\{\varphi\},s')\in\Delta \\ r\in\mathcal{B} \\ r\neq p\vee t\neq q}} \exists x.\,\mathsf{avail}_{p,q,\mathcal{B}\cup\{t\}}(x_1,s',\boldsymbol{r}')\wedge\varphi$$
$$\bigvee_{\substack{(s,r\to t:x\{\varphi\},s')\in\Delta \\ r\notin\mathcal{B} \\ r\neq p\vee t\neq q}} \exists x.\,\mathsf{avail}_{p,q,\mathcal{B}}(x_1,s',\boldsymbol{r}')\wedge\varphi$$
$$\bigvee_{\substack{(s,p\to q:x\{\varphi\},s')\in\Delta \\ p\notin\mathcal{B}}} \varphi[x_1/x]\ .$$

$$\mathsf{prodreach}_{p}(s_1',\boldsymbol{r}_1',s_2',\boldsymbol{r}_2') :=_\mu$$
$$\vee\ (s_1'=s_0\wedge s_2'=s_0)$$
$$\vee \Big(\bigvee_{\substack{(s_1,r\to s:x_1\{\varphi_1\},s_1')\in\Delta_1 \\ (s_2,r\to s:x_2\{\varphi_2\},s_2')\in\Delta_2 \\ p=r\vee p=s}} \exists x_1x_2.\,\mathsf{prodreach}_{p}(s_1,\boldsymbol{r}_1,s_2,\boldsymbol{r}_2)\wedge\varphi_1\wedge\varphi_2\wedge x_1=x_2\Big)$$
$$\vee \Big(\bigvee_{\substack{(s_1,r\to s:x_1\{\varphi_1\},s_1')\in\Delta_1 \\ p\neq r\wedge p\neq s}} \exists x_1.\,\mathsf{prodreach}_{p}(s_1,\boldsymbol{r}_1,s_2',\boldsymbol{r}_2')\wedge\varphi_1\Big)$$
$$\vee \Big(\bigvee_{\substack{(s_2,r\to s:x_2\{\varphi_2\},s_2')\in\Delta_2 \\ p\neq r\wedge p\neq s}} \exists x_2.\,\mathsf{prodreach}_{p}(s_1',\boldsymbol{r}_1',s_2,\boldsymbol{r}_2)\wedge\varphi_2\Big)\ .$$



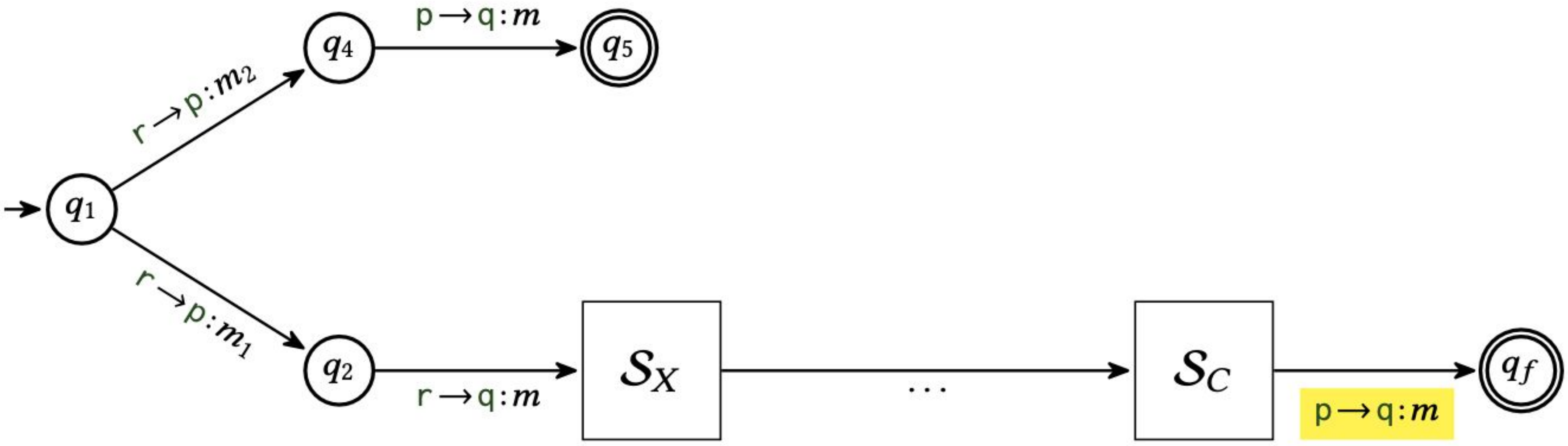
Complexity

Fragment	Complexity
Symbolic, boolean	PSPACE-complete
Non-symbolic	co-NP-complete
Multiparty session types	co-NP-complete

Complexity

3-SAT instance: set of variables X, set of clauses C

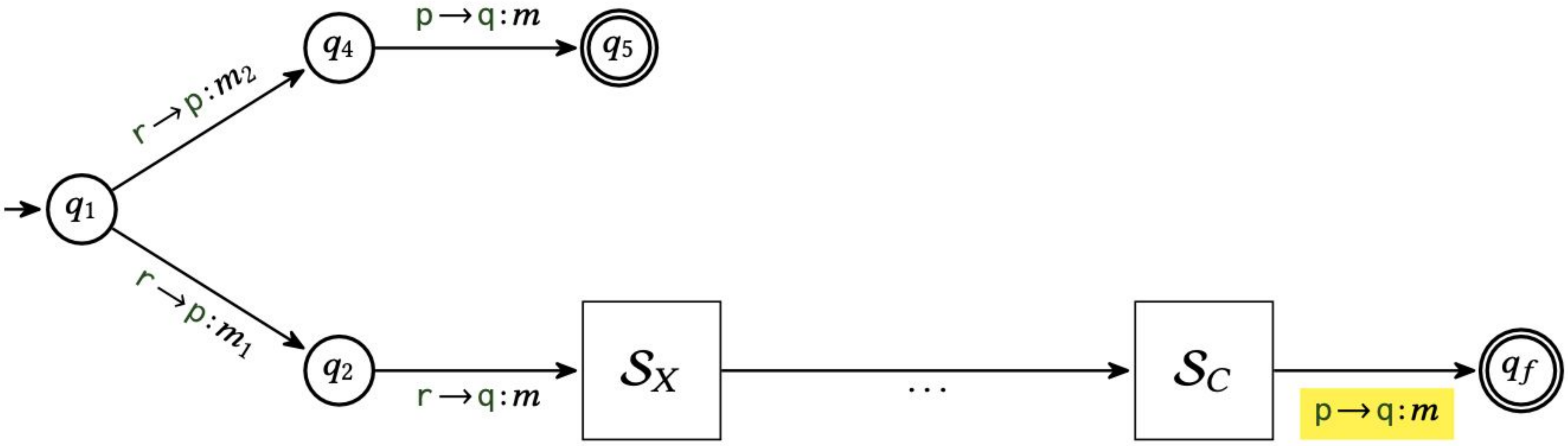
S is implementable iff 3-SAT instance is satisfiable



Complexity

3-SAT instance: set of variables X, set of clauses C

~~S is implementable~~ m is *receivable* iff 3-SAT instance is satisfiable



Synthesis

Theorem. If a concrete protocol is implementable, then the *canonical* implementation implements it.

- Synthesis is as hard as determinizing the specification fragment
- Off-the-shelf determinization algorithms:
  - All finite fragments (subset construction)
  - Symbolic finite automata (modified subset construction)
  - Classes of timed and register automata

Conclusion

- First sound and (relatively) complete algorithm for deciding implementability of symbolic global protocols
- Improves prior work in terms of **expressivity**, **completeness** and **complexity**

- Extras: Sprout implementation [Li et al. CAV'25], Rocq mechanization [Li and Wies ITP'25]

Conclusion

- First sound and (relatively) complete algorithm for deciding implementability of symbolic global protocols
- Improves prior work in terms of **expressivity**, **completeness** and **complexity**

- Extras: Sprout implementation [Li et al. CAV'25], Rocq mechanization [Li and Wies ITP'25]

Thank you!