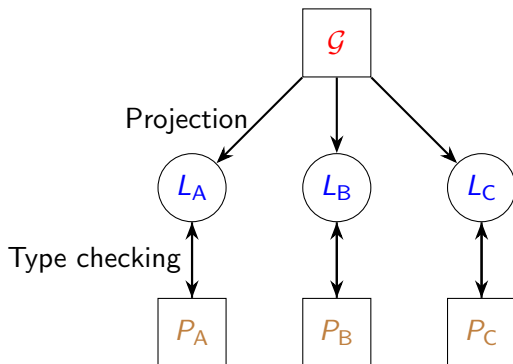


Multiparty Session Types

- ▶ Honda, K., Yoshida, N., and Carbone, M. (2008)
- ▶ Verification and design of *communication protocols*
- ▶ Avoid *deadlocks*, ensure *progress*, etc...



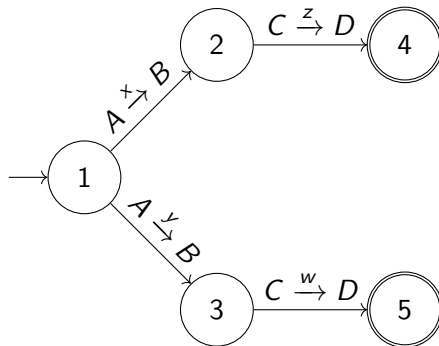
1. Global type

2. Local type

3. Processes

Global Types

- ▶ Description of a **global** behavior of a system.
- ▶ Defined as automata



Local type

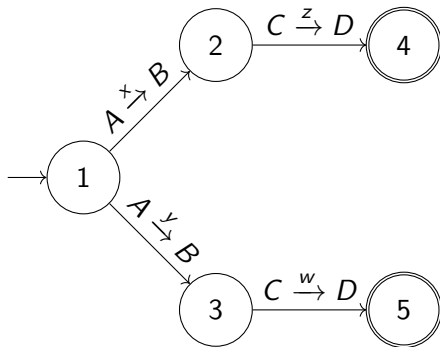
- ▶ Point of view of a participant
- ▶ Obtained via *projection* operation
- ▶ Behavior can differ for different communication semantics (p2p, sync)

Realisability problem: Does the implementation of a system **respects** the behavior described?

$$L(G) = L(\text{proj}(G))$$

The example is not realisable

This example is **not** realisable because C doesn't know what A sent.



The trace $A \xrightarrow{x} B; C \xrightarrow{w} D$ does not appear in $L(G)$.

Reduction to sync [1]


A global type G is deadlock-free realisable in **p2p** iff:

1. $L_{p2p}(proj(G))$ is sync;
2. $proj(G)$ is orphan-free in p2p;
3. $L_{p2p}(proj(G))$ is deadlock-free
4. G is weak realisable in sync
5. G is deadlock-free in sync

[1] Di Giusto, Cinzia, Etienne Lozes, and Pascal Urso. "Realisability and Complementability of Multiparty Session Types." (2025).

Reduction to sync [1]

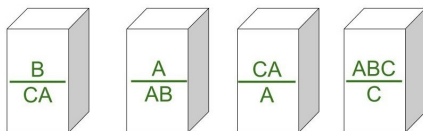
A global type G is deadlock-free realisable in **p2p** iff:

1. $L_{p2p}(proj(G))$ is sync;
2. $proj(G)$ is orphan-free in p2p;
3. $L_{p2p}(proj(G))$ is deadlock-free
4. G is weak realisable in sync 
5. G is deadlock-free in sync

[1] Di Giusto, Cinzia, Etienne Lozes, and Pascal Urso. "Realisability and Complementability of Multiparty Session Types." (2025).

First contribution


- ▶ Realisability for sync global types is **undecidable**.
Proof: by reduction to the PCP problem.
- ▶ PCP: given a set of tiles, find an ordering such that the strings formed by the top and bottom halves are equal.
- ▶ Proof adapted from Alur et al. [2]



[2] Alur, Rajeev, Kousha Etessami, and Mihalis Yannakakis. "Realizability and verification of MSC graphs." Theoretical Computer Science 331.1 (2005): 97-114.

Reduction to sync [1]

A global type G is deadlock-free realisable in **p2p** iff:

1. $L_{p2p}(proj(G))$ is sync;
2. $proj(G)$ is orphan-free in p2p;
3. $L_{p2p}(proj(G))$ is deadlock-free
4. G is weak realisable in sync
5. G is deadlock-free in sync 

[1] Di Giusto, Cinzia, Etienne Lozes, and Pascal Urso. “Realisability and Complementability of Multiparty Session Types.” (2025).

Second contribution

- ▶ RE_{SCU}: Model-checking TUI tool written in OCaml
- ▶ Added two verification if *sync* system:
 - ▶ *Deadlock*: a final state is always reachable
 - ▶ *Progress*: the system can always perform an action

RESCU Example - Dining Philosophers

Two Philosophers, two forks.

This system is RSC.

There are some sink states:

Sink: Id=11 Configuration={F0:4; F1:3; P1:2; P2:2}

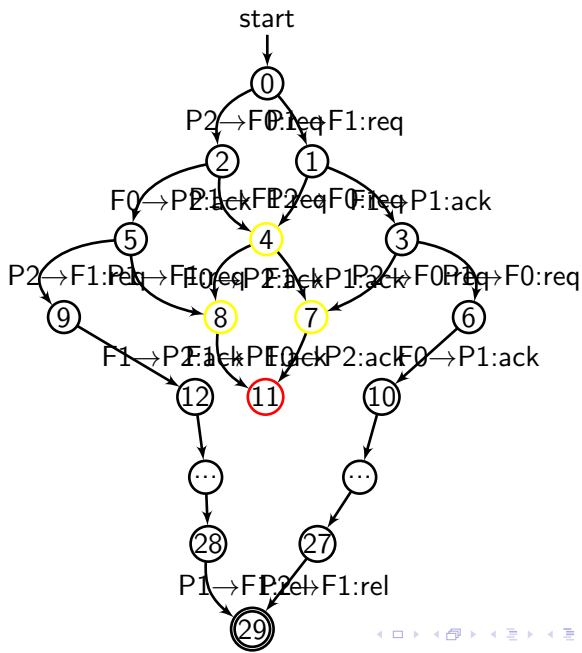
There are some deadlock states:

Deadlock: Id=4 Configuration={F0:2; F1:1; P1:1; P2:1}

Deadlock: Id=11 Configuration={F0:4; F1:3; P1:2; P2:2}

...

RESCU Example - Dining Philosophers



Conclusion

Summary of contributions:

- ▶ Proof of undecidability for weak realisability in sync
- ▶ Enriched the tool RESCU

Future work:

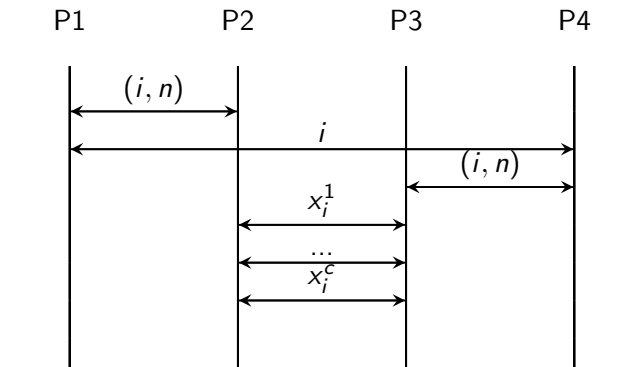
- ▶ Prove undecidability of deadlock-free realisability for sync global types
- ▶ Continue the development of RESCU

Thanks! Questions?

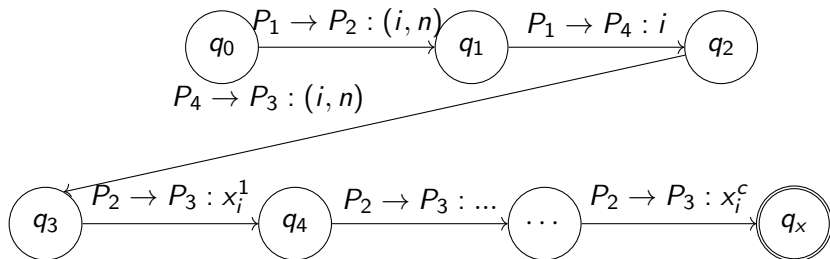
Weak and Safe realisability

- ▶ Weak realisability: the global type G is *weak* realisable in sync if there exist a CFSM system that can implement the global type.
- ▶ Weak realisability: the global type G is *safe* realisable in sync if it is *weak realisable* and the CFSM system is **deadlock free**.

The MSC M_i^n



The global type G_i^n



The global type L^*

