Realisability of Global Types: Decidability and Verification

Presentata da: **Gabriele Genovese**

Relatore: Prof. Ivan Lanese
Correlatori: Prof. Cinzia Di Giusto
Chiar.mo Prof. Étienne Lozes
Controrelatore: Chiar.mo Prof. Luca Padovani

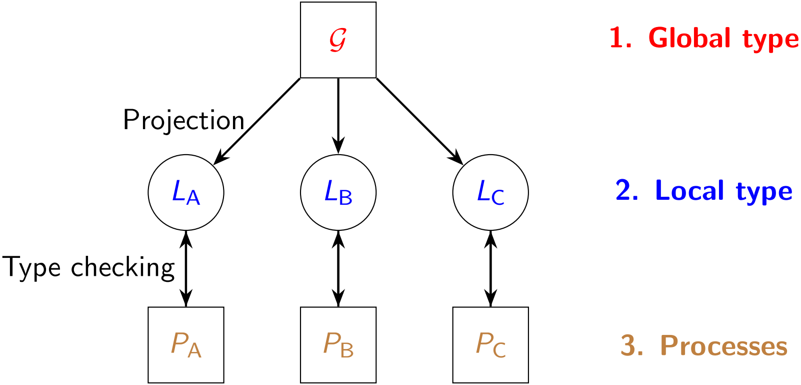
30 Ottobre 2025

i3S
sophia antipolis

ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

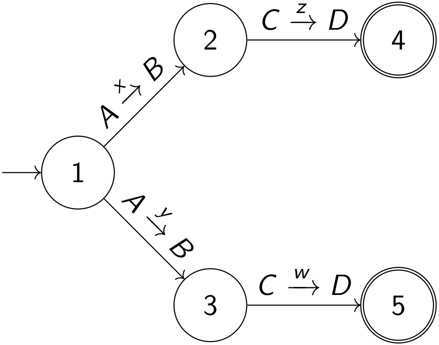
Multiparty Session Types

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



Global Types

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



Local type

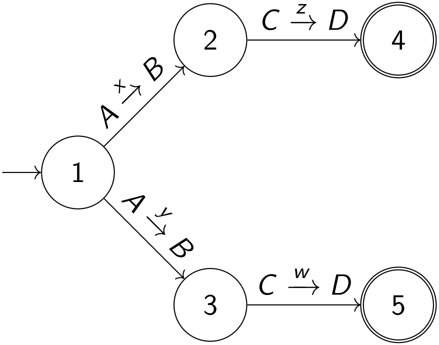
- Point of view of a participant
- Obtained via *projection* operation
- Behavior can be different under different communication semantics (p2p, sync)

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

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

The example is not realisable

This global type 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. $L_{\mathrm{p2p}}(\mathit{proj}(G))$ is sync;
2. $\mathit{proj}(G)$ is orphan-free in p2p;
3. $L_{\mathrm{p2p}}(\mathit{proj}(G))$ is deadlock-free
4. G is weak realisable in sync
5. G is deadlock-free in sync

[1] Di Giusto...

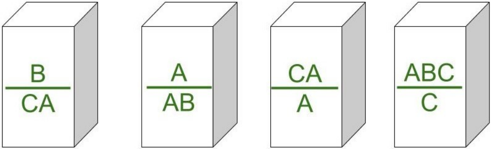
Reduction to sync

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

1. $L_{\text{p2p}}(\mathit{proj}(G))$ is sync;
2. $\mathit{proj}(G)$ is orphan-free in p2p;
3. $L_{\text{p2p}}(\mathit{proj}(G))$ is deadlock-free
4. G is deadlock-free realisable in sync $\longleftarrow$

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 adaptated from Alur et al. [2]



[2] Alur...

Reduction to sync [1]

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

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

[1] Di Giusto...

Second contribution

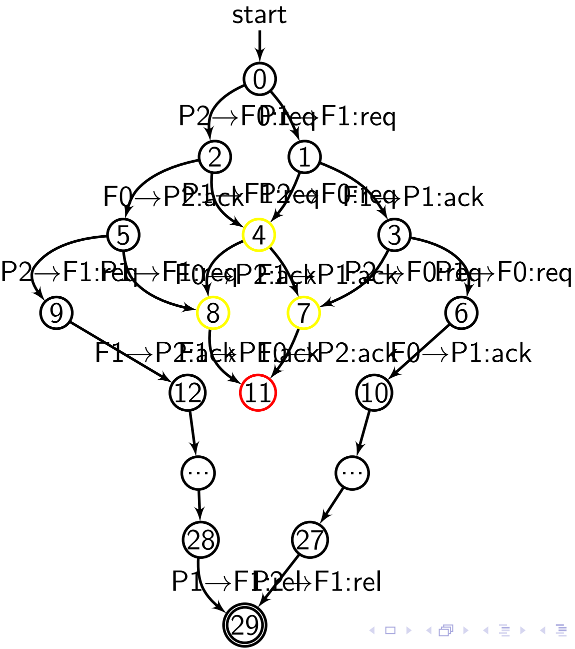
- ReSCu: Model-checking TUI tool written in OCaml
- Added two verification if *sync* system:
  - *Deadlock*: a final state is always reachable
  - *Progress*: the system can always perform an action

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 vs safe

dettagli prova