Overview on Constrained Multiparty Synchronisation in Team Automata

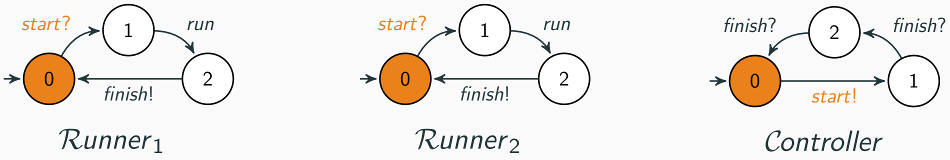
José Proença (CISTER & University of Porto, Portugal)

October, 19

FACS 2023 – Anniversary Track

This Talk

Overview on **Constrained** **Multiparty** Synchronisation in **Team Automata**



Team Automata (TA)

[FM'03,21,23] [CSCW'03]
[ICTAC'20,23]
[COORDINATION'17,20]

Multiparty synchronisation

Ctr $\rightarrow \{R1, R2\}$: start

Constrained synchronisation

start: $1 \rightarrow 2$
finish: $1 \rightarrow 1$

This Talk

Overview on Constrained Multiparty Synchronisation in Team Automata

Runners with orchestrators

- Reo
- BIP

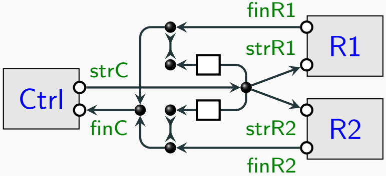
Runners with choreographies

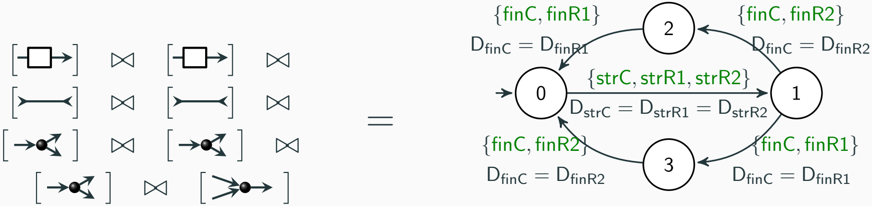
- Choreography Automata (CA)
- Multiparty Session Types (MPST)

Realisability for Team Automata

- challenges
- ongoing work

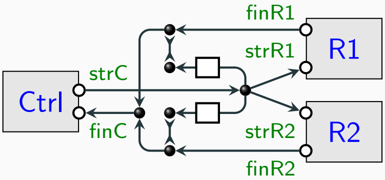
Orchestrators – Reo



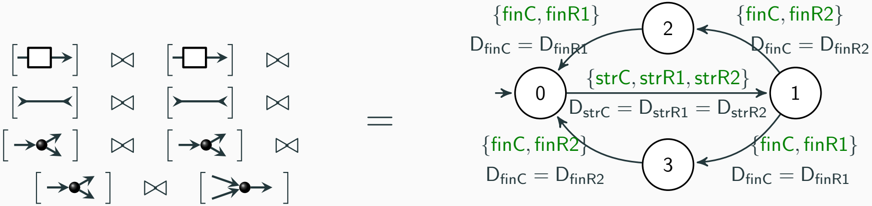


(Semantics with Constraint Automata)

Orchestrators – Reo

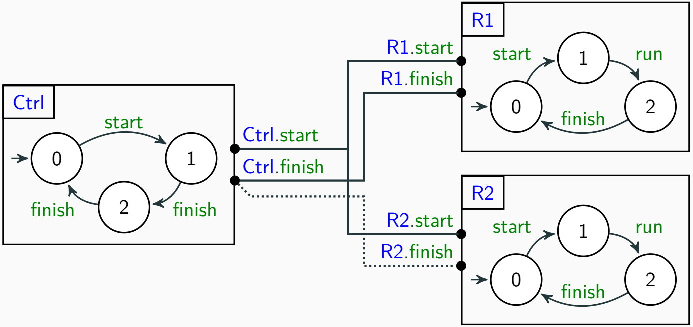


- Focus on connectors (not components)
- Connectors built compositionally
- Components should be flexible/compatible



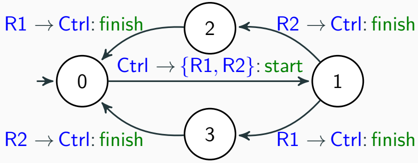
(Semantics with Constraint Automata)

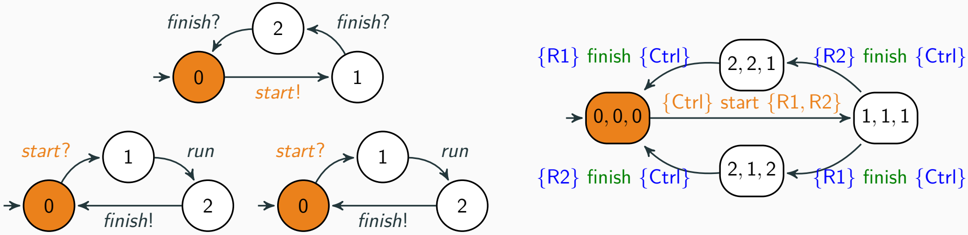
Orchestrators – BIP



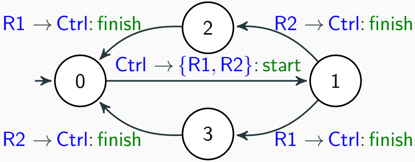
- components expose ports
- interactions restrict which ports can communicate
- constructors using unicast (●) and broadcast (▲) can be used to restrict interactions
- dataflow can be added

Choreographies – Choreography Automata (CA)

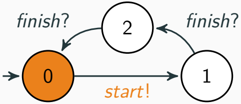


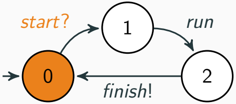


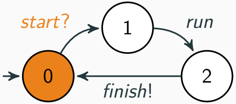
Choreographies – Choreography Automata (CA)

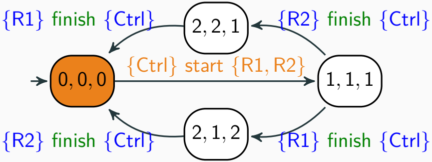


- Many results over the language of CA
- Projections of the language of CA
- *[Can be produced by other choreography languages]*









Choreographies – Multiparty Session Types (MPST)

$$\lambda X \cdot \mathsf{Ctrl} \rightarrow \{\mathsf{R1}, \mathsf{R2}\} : \left\{ \mathsf{start}.\begin{pmatrix} \mathsf{R1} \rightarrow \mathsf{Ctrl} : \mathsf{finish}. \\ \mathsf{R2} \rightarrow \mathsf{Ctrl} : \mathsf{finish}.X \end{pmatrix}, \mathsf{start}.\begin{pmatrix} \mathsf{R2} \rightarrow \mathsf{Ctrl} : \mathsf{finish}. \\ \mathsf{R1} \rightarrow \mathsf{Ctrl} : \mathsf{finish}.X \end{pmatrix} \right\}$$

Choreographies – Multiparty Session Types (MPST)

$$\mathcal{G} \quad \lambda X \cdot \mathsf{Ctrl} \rightarrow \{\mathsf{R1}, \mathsf{R2}\} : \left\{ \mathsf{start}. \begin{pmatrix} \mathsf{R1} \rightarrow \mathsf{Ctrl} : \mathsf{finish}. \\ \mathsf{R2} \rightarrow \mathsf{Ctrl} : \mathsf{finish}.X \end{pmatrix}, \mathsf{start}. \begin{pmatrix} \mathsf{R2} \rightarrow \mathsf{Ctrl} : \mathsf{finish}. \\ \mathsf{R1} \rightarrow \mathsf{Ctrl} : \mathsf{finish}.X \end{pmatrix} \right\}$$

$$\mathcal{L} \quad \mathsf{R1} \triangleright \lambda X \cdot \begin{pmatrix} \mathsf{Ctrl?start} . \\ \mathsf{Ctrl!finish} . X \end{pmatrix} \qquad \mathsf{R2} \triangleright \lambda X \cdot \begin{pmatrix} \mathsf{Ctrl?start} . \\ \mathsf{Ctrl!finish} . X \end{pmatrix} \qquad \mathsf{Ctrl} \triangleright \lambda X \cdot \begin{pmatrix} \cdots \\ \cdots \end{pmatrix}$$

Choreographies – Multiparty Session Types (MPST)

$$\mathcal{G} \quad \lambda X \cdot \mathsf{Ctrl} \to \{\mathsf{R1}, \mathsf{R2}\} : \left\{ \mathsf{start}. \begin{pmatrix} \mathsf{R1} \to \mathsf{Ctrl} : \mathsf{finish}. \\ \mathsf{R2} \to \mathsf{Ctrl} : \mathsf{finish}.X \end{pmatrix}, \mathsf{start}. \begin{pmatrix} \mathsf{R2} \to \mathsf{Ctrl} : \mathsf{finish}. \\ \mathsf{R1} \to \mathsf{Ctrl} : \mathsf{finish}.X \end{pmatrix} \right\}$$

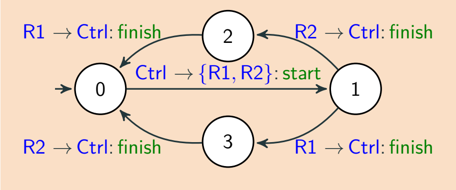
$$\mathcal{L} \quad \mathsf{R1} \triangleright \lambda X \cdot \begin{pmatrix} \mathsf{Ctrl?start}\,. \\ \mathsf{Ctrl!finish}\,.\,X \end{pmatrix} \qquad \mathsf{R2} \triangleright \lambda X \cdot \begin{pmatrix} \mathsf{Ctrl?start}\,. \\ \mathsf{Ctrl!finish}\,.\,X \end{pmatrix} \qquad \mathsf{Ctrl} \triangleright \lambda X \cdot \begin{pmatrix} \cdots \\ \cdots \end{pmatrix}$$

Process R1 | *Process R2* | *Process Ctr*

Realisability for Team Automata

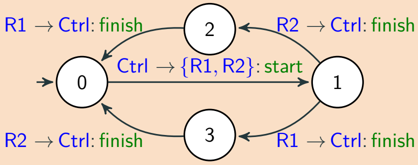
What is Realisability in TA?

Start with global $\mathcal{G}$



What is Realisability in TA?

Start with global $\mathcal{G}$



Synthesize a realisation $\mathcal{R}$

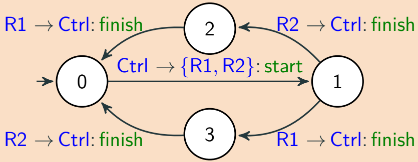
$$\mathcal{R} = \text{Ctr} \leftrightarrow \text{R1} \leftrightarrow \text{R2}$$

such that

$\mathcal{G}$ *"somehow" behaves* as $\mathcal{R}$

What is Realisability in TA?

Start with global $\mathcal{G}$



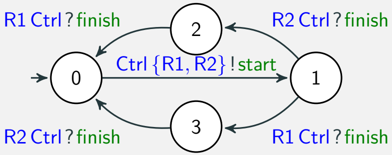
Synthesize a realisation $\mathcal{R}$

$$\mathcal{R} = \text{Ctr} \leftrightarrow \text{R1} \leftrightarrow \text{R2}$$

such that

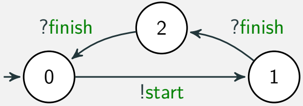
$\mathcal{G}$ *"somehow" behaves* as $\mathcal{R}$

Different agents and networks



(Ctrl with rich labels)

vs.



(Ctrl with poor labels)

How much do local agents know? Different network assumptions?

Properties and Behavioural Equivalence

Properties expressible in dynamic logic

- *No runner should finish before it has been started by the controller*
- *Any started runner should be able to finish its run*

Properties expressible with regular expressions

- *Runner 1 can finish immediately after Runner 2*
- *After starting and runner 1 finishing, it is not possible to start another race*

Properties and Behavioural Equivalence

Properties expressible in dynamic logic

- *No runner should finish before it has been started by the controller*
- *Any started runner should be able to finish its run*

Properties expressible with regular expressions

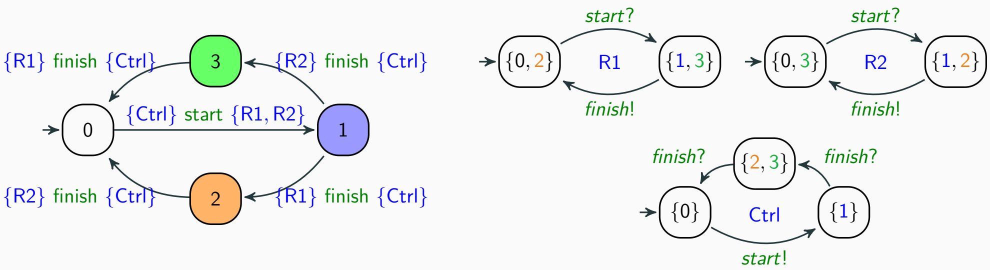
- *Runner 1 can finish immediately after Runner 2*
- *After starting and runner 1 finishing, it is not possible to start another race*

Properties of $\mathcal{G}$ should also hold for $\mathcal{R}$ (and vice-versa)

- Dynamic logic: bisimilar (non-deterministic) systems obey the same formulas
- Regular expressions: language equivalent systems include the same expressions

Our Approach to Synthesize a Realisation

$\mathcal{G}$	$\mathcal{R}$



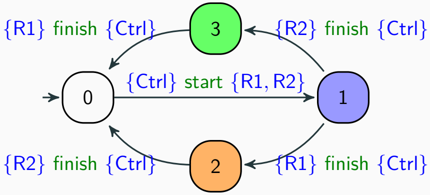
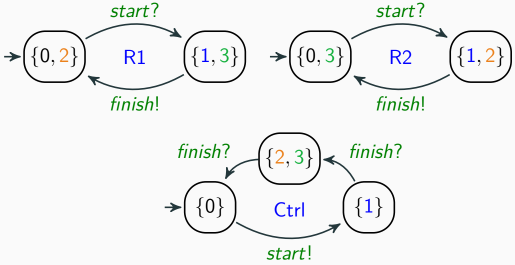
Group indistinguishable states

$\text{R1} : 0 \equiv_{R1} 2 \ ; \ 1 \equiv_{R1} 3$

$\text{R2} : 0 \equiv_{R2} 3 \ ; \ 1 \equiv_{R2} 2$

$\text{Ctrl} : 2 \equiv_{Ctrl} 3$

Our Approach to Synthesize a Realisation

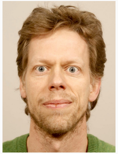
$\mathcal{G}$	$\mathcal{R}$
	

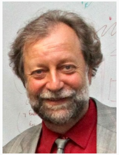
Group indistinguishable states

R1 : $0 \equiv_{R1} 2$; $1 \equiv_{R1} 3$

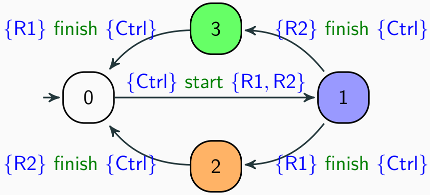
R2 : $0 \equiv_{R2} 3$; $1 \equiv_{R2} 2$

Ctrl : $2 \equiv_{Ctrl} 3$





Realisability Conditions: Which States are Indistinguishable?



Group indistinguishable states

R1 : $0 \equiv_{R1} 2$; $1 \equiv_{R1} 3$

R2 : $0 \equiv_{R2} 3$; $1 \equiv_{R2} 2$

Ctrl : $2 \equiv_{Ctrl} 3$

Sufficient condition to discover equivalences

1. collapse "τ" transitions
2. $\forall$ label γ,
   participant k,
   similar k-transition,
   shared ***indistinguishable*** source g
3. $\exists$ shared g' ***indistinguishable*** target s.t.

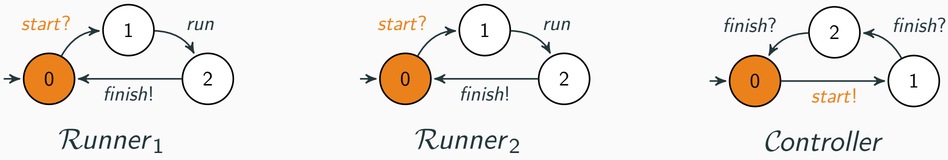
$$g \xrightarrow{\gamma} g'$$

Some Challenges

- compact representation of the global $\mathcal{G}$
  e.g., $\Big(\mathsf{Ctrl} \to \{\mathsf{R1}, \mathsf{R2}\} : \mathsf{start} \;\; ; \;\; (\mathsf{R1} \to \mathsf{Ctrl} : \mathsf{finish} \mid\mid \mathsf{R2} \to \mathsf{Ctrl} : \mathsf{finish})\Big)^*$
- alternatively: learn $\mathcal{G}$
- other network assumptions (e.g., asynchronous, causal channels, lossy, . . .)
- heterogeneous agents (different assumptions/realisations)
- variability: global representation for any number of runners
  (to match the flexibility of sync. types, e.g., start: [1] $\to$ [2..*])
- refine realisations: can we make the local behaviour *"more specific"*, such that its composition is weakly bisimilar to the global behaviour?

Wrap Up

Overview on **Constrained Multiparty** Synchronisation in **Team Automata**



Runners with orchestrators

- Reo
- BIP

Runners with choreographies

- Choreographic Automata (CA)
- Multiparty Session Types (MPST)

Realisability for Team Automata

- challenges
- ongoing work