UNIVERSITY OF GOTHENBURG

CHALMERS

Reconfigurable Interaction for MAS Modelling

Nir Piterman

Joint work with Yehia abd Alrahaman* and Giuseppe Perelli

D-SynMA
No.772459

European Research Council
Established by the European Commission

*who prepared many of these slides!

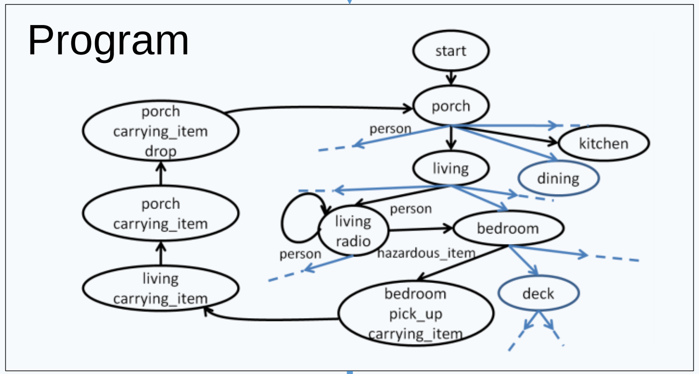
Single-Agent Reactive Synthesis

- Automatically produce reactive programs from high-level descriptions of desired behaviour.

- Walk around the aisles.
- Avoid obstacles.
- If spot missing merchandise notify manager.

- Walk around the aisles.
- Avoid obstacles.
- If spot missing merchandise notify manager.

Synthesis



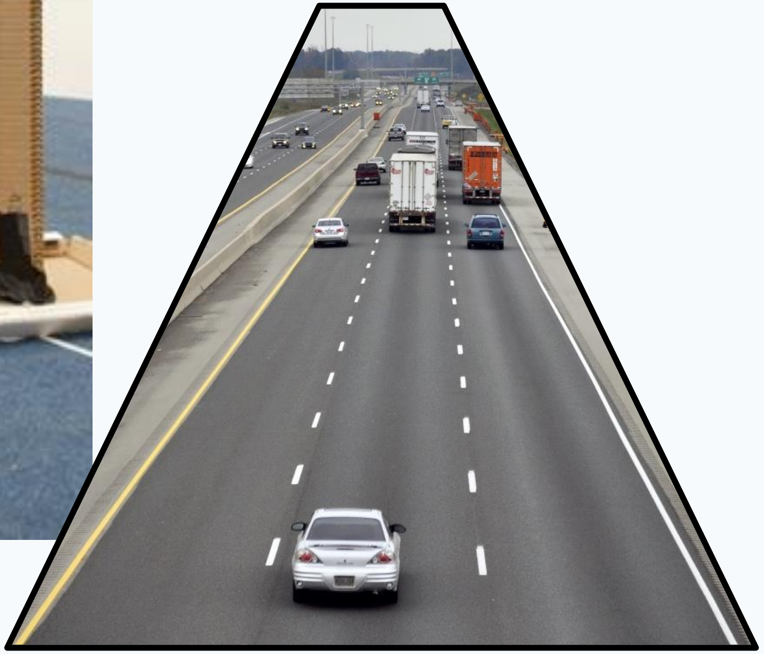
Single-agent Synthesis in Practice

- Robotics:
  - Hadas Kress-Gazit (Cornell)
  - Richard Murray (Caltech)
  - Ufuk Topcu (U Texas)
  - …

- Model-driven development, design of adaptive systems, industrial automation …

Autonomous Reactive Programs

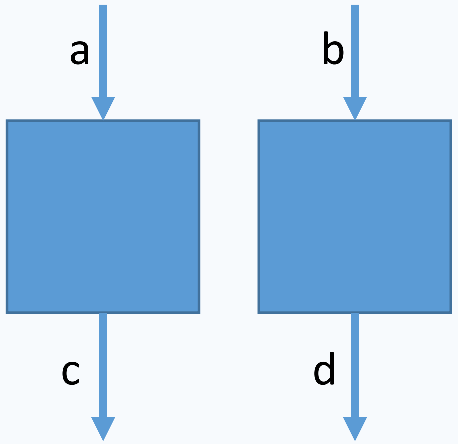
Autonomous Reactive Programs



Autonomous Reactive Programs

Autonomous Reactive Programs

Distributed Synthesis



- Shared Variables:
  - Distributed synthesis is undecidable [PR90].
  - Restricted architectures w very high complexity.
  - Bounded Synthesis …
- Message Passing:
  - Synchronization!
  - Limited interaction modes.
- Zielonka Automata:
  - More architectures are decidable [GGMW13,MW14].
  - Borderline of undecidability still unclear.
  - A transition combines the states of all participants.

Towards Multi-agent Synthesis?

- What are the minimal features of a cooperative reactive program?
  - Synchronization
  - Communication of data
  - Well defined interfaces
- Combine:
  - Shared variables
  - Message communication
  - Full synchronization
- Back to Modelling and Model Checking:
  - MCMAS [LQR17]
  - Reactive Module Games [GHW17]

ReCiPe Interaction Formalism

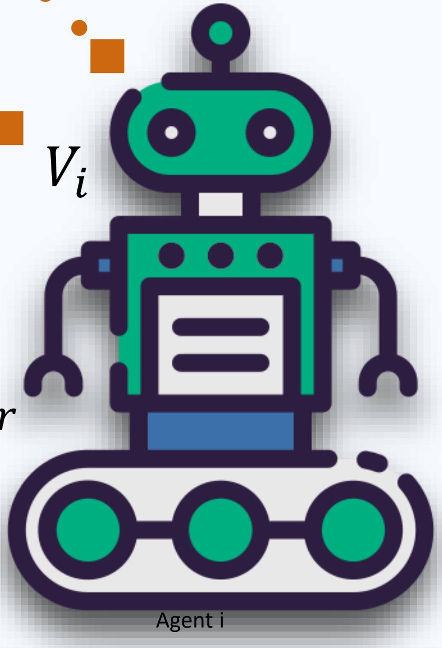
- Support realistic modelling of reconfigurable Multi-Agent Systems
- Support reconfigurable interaction interfaces – parameterized to the evolving state of agents.
- Modelling convenience for high-level interaction feature of MAS (e.g., coalition formation, collaboration, self-organization, etc.) that are currently hard to encode.

The ReCiPe Interaction Formalism

The basic building block is an Agent

Agent has a local state, V_i

Send and Receive transition relations $\mathcal{T}_i^s, \mathcal{T}_i^r$

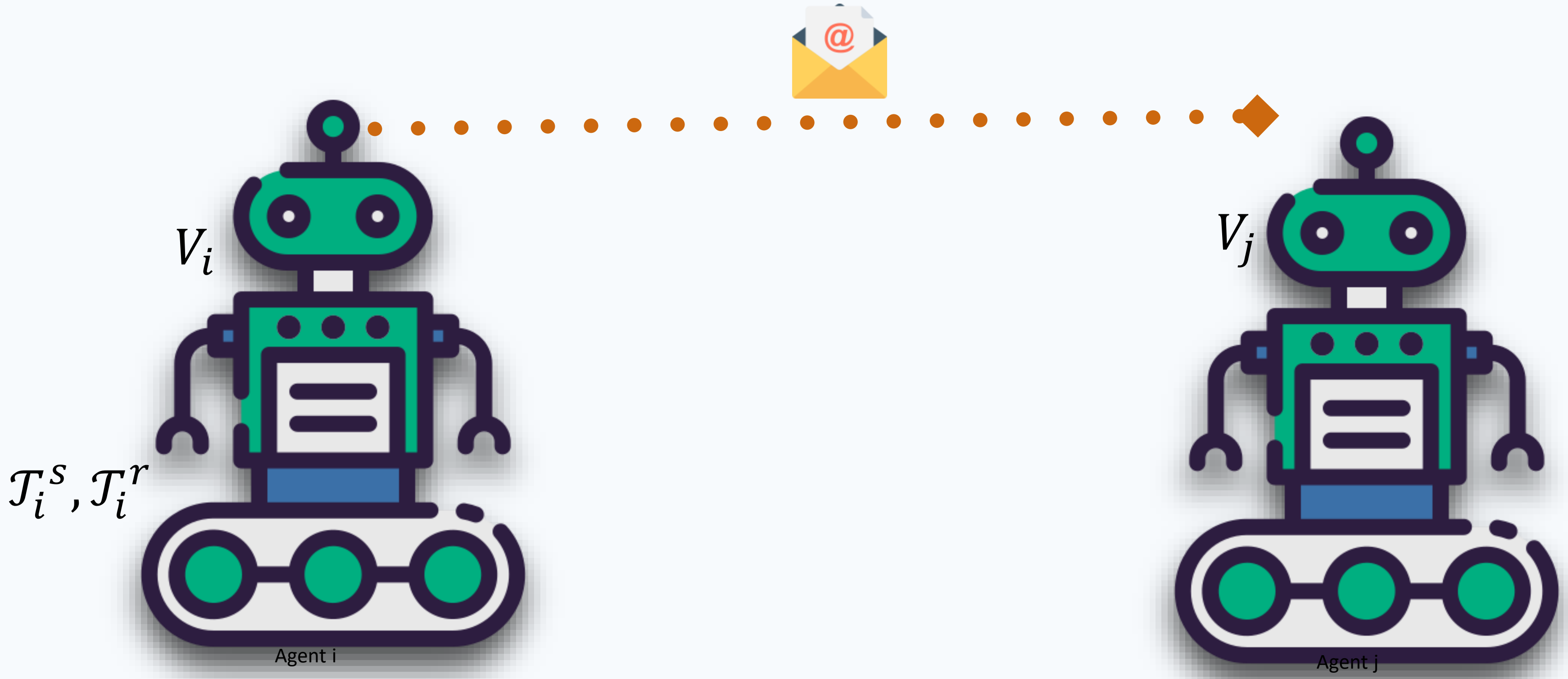


The ReCiPe Interaction Formalism

A system is a collection of agents

Agents interact based on multicast links c or a broadcast ★

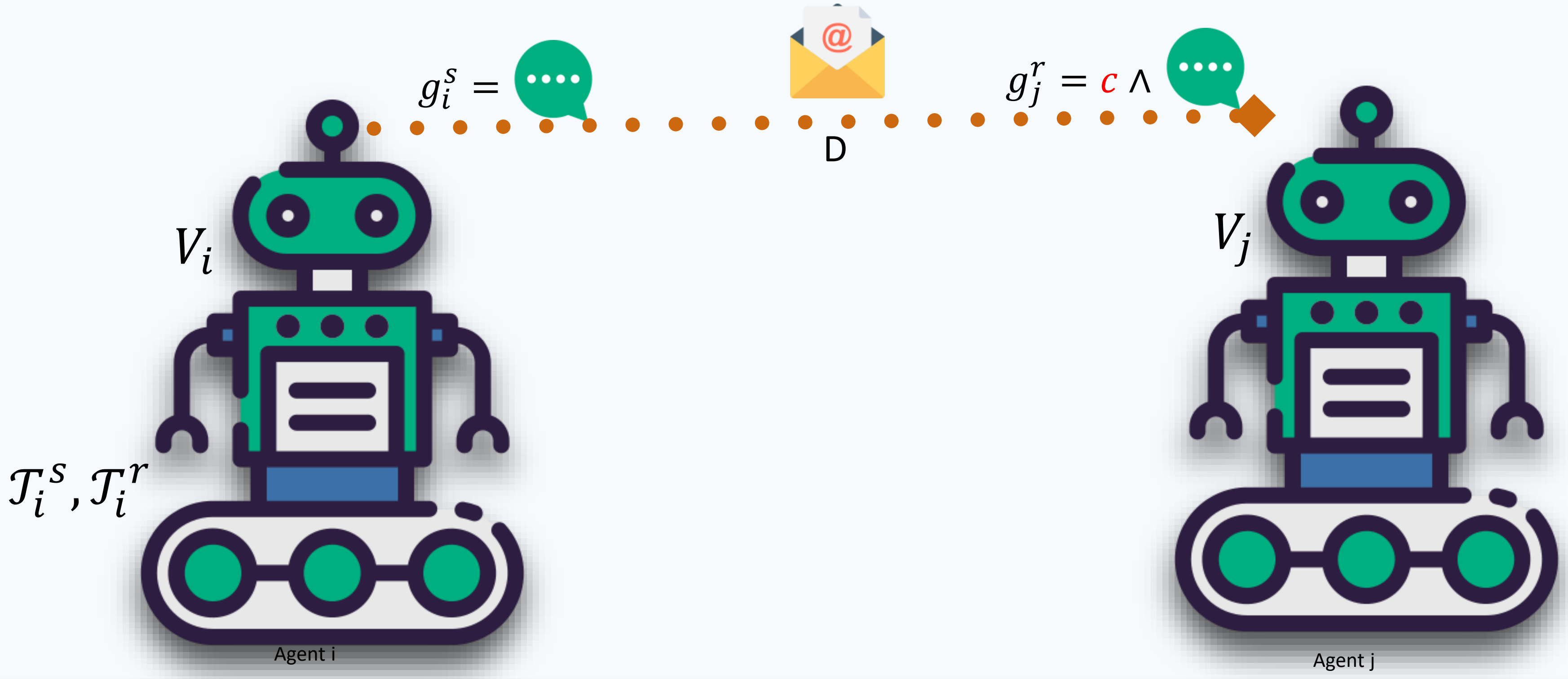
Messages transfer data (could include channel names)



The ReCiPe Interaction Formalism

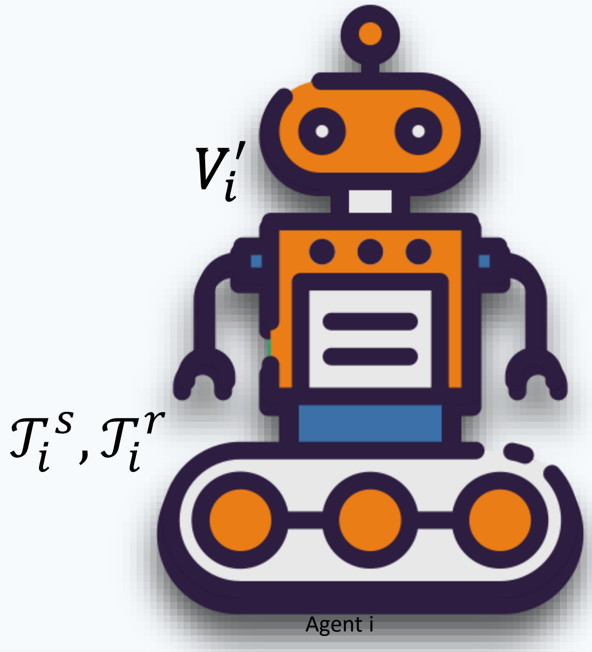
Senders can select the target using send guards g_i^s

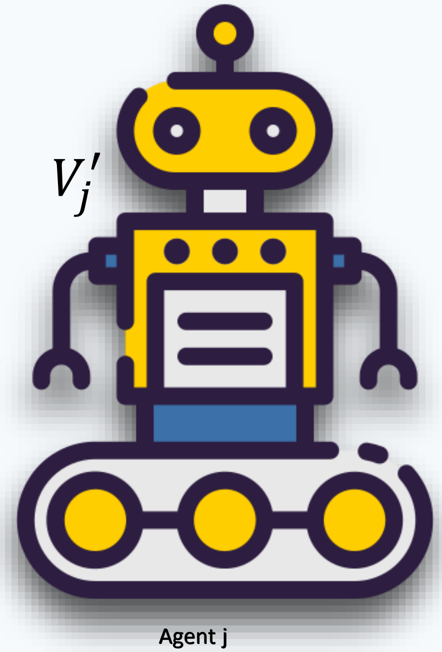
Receive guards g_j^r define connectivi



The ReCiPe Interaction Formalism

Side effects of interactions may incur reconfiguration changes

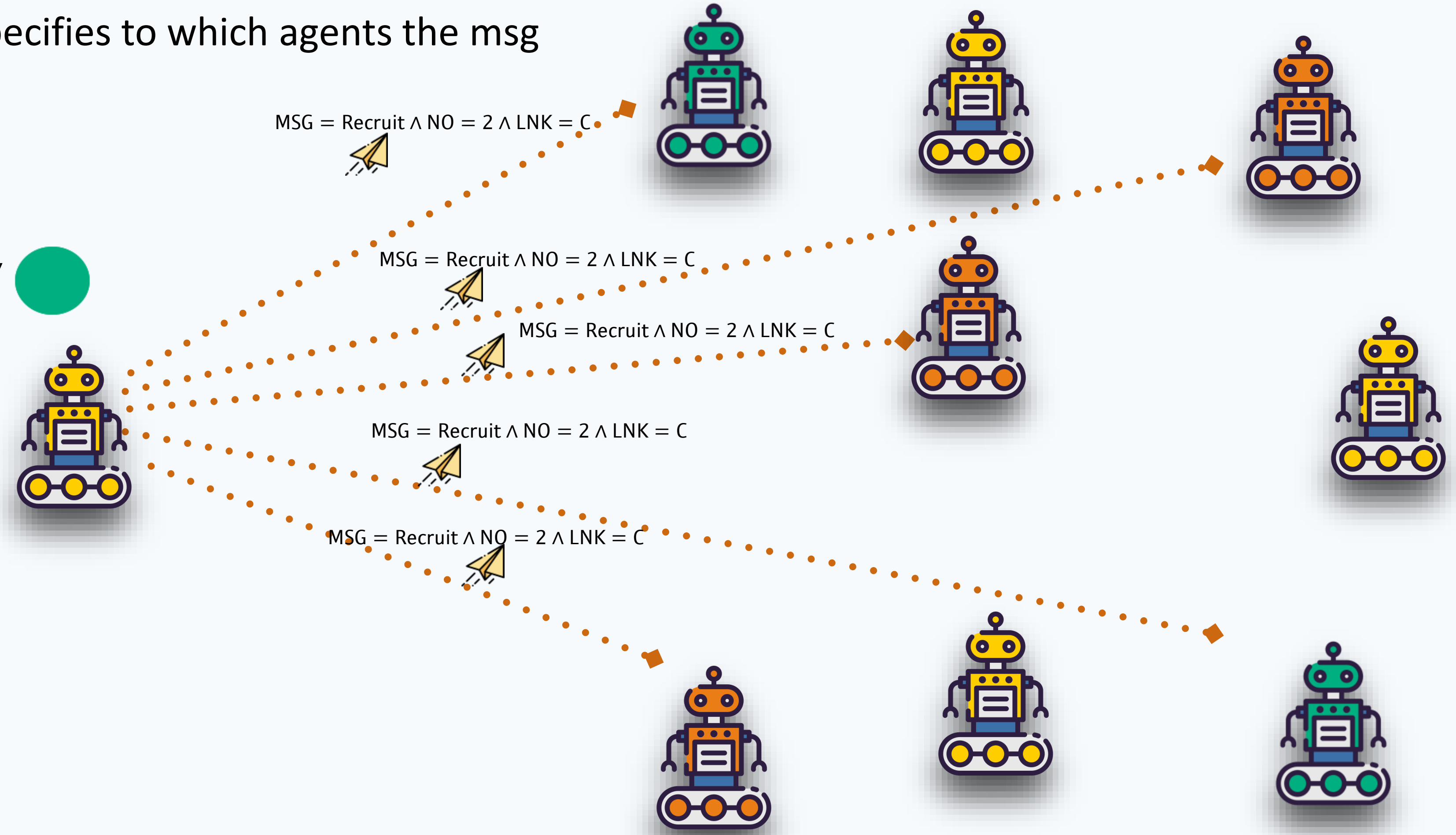




Non-Blocking Broadcast

Send guard specifies to which agents the msg is intended.

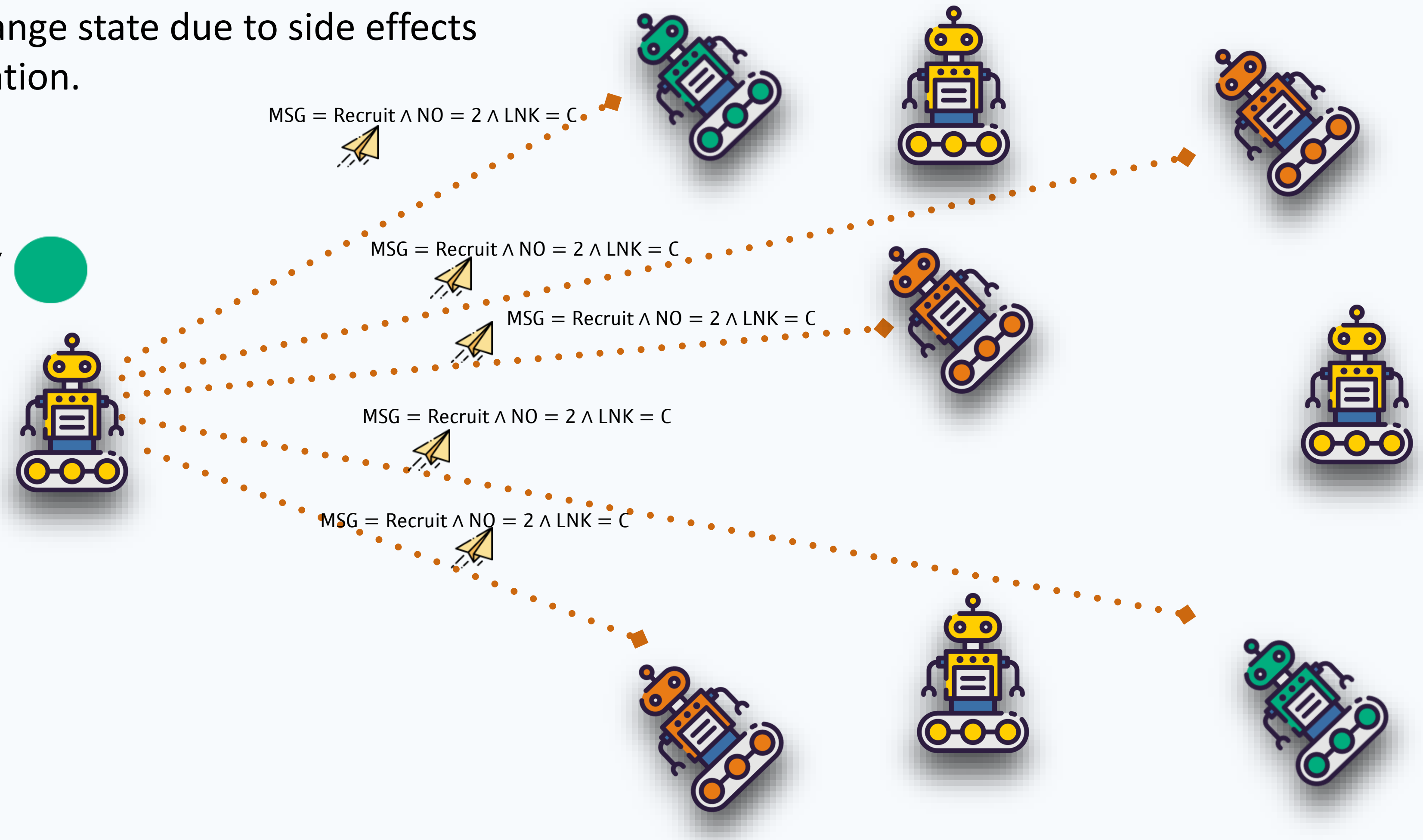
g^s = [orange] ∨ [green]



Non-Blocking Broadcast

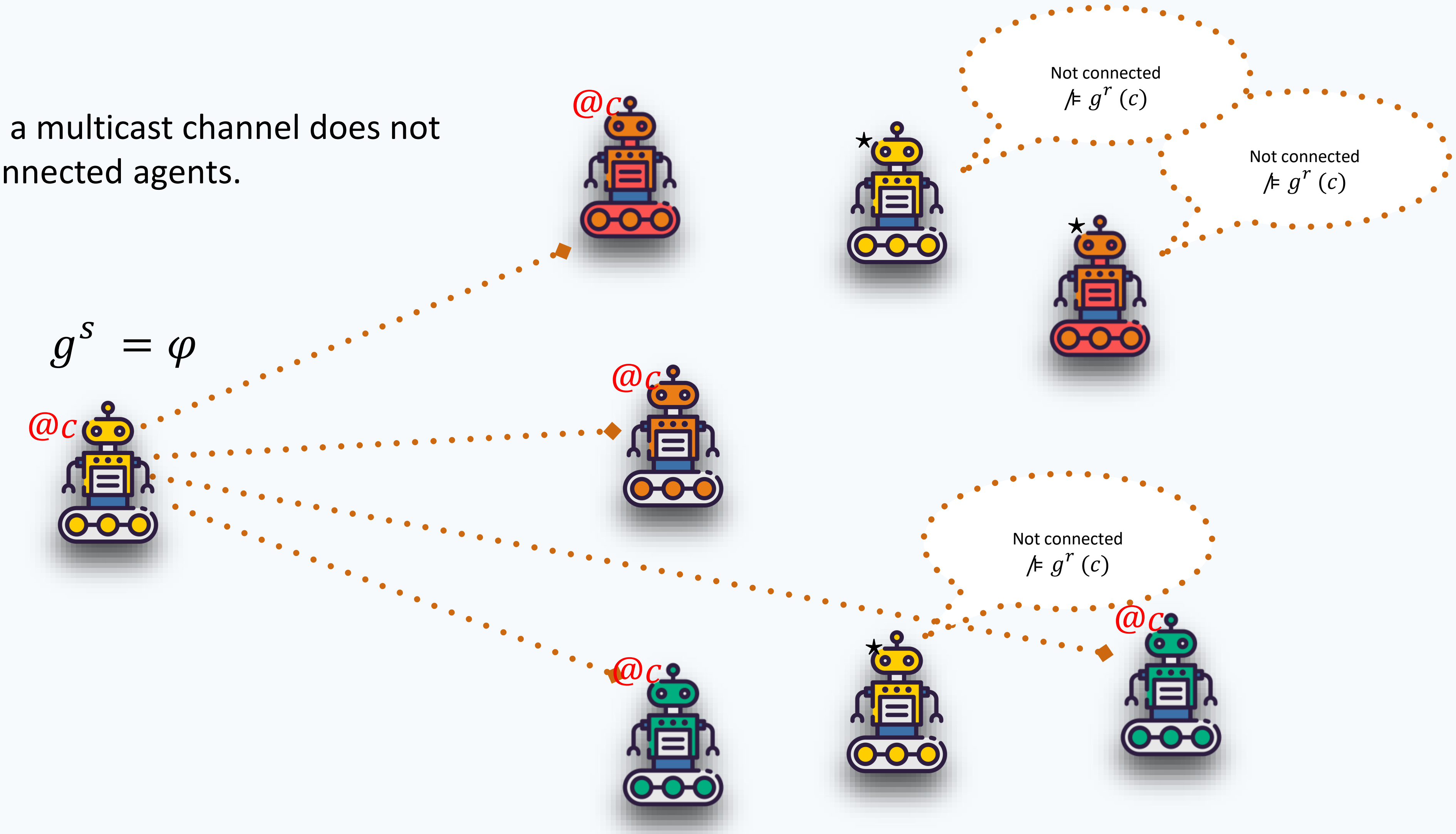
Recipients change state due to side effects of communication.

g^s = ● ∨ ●



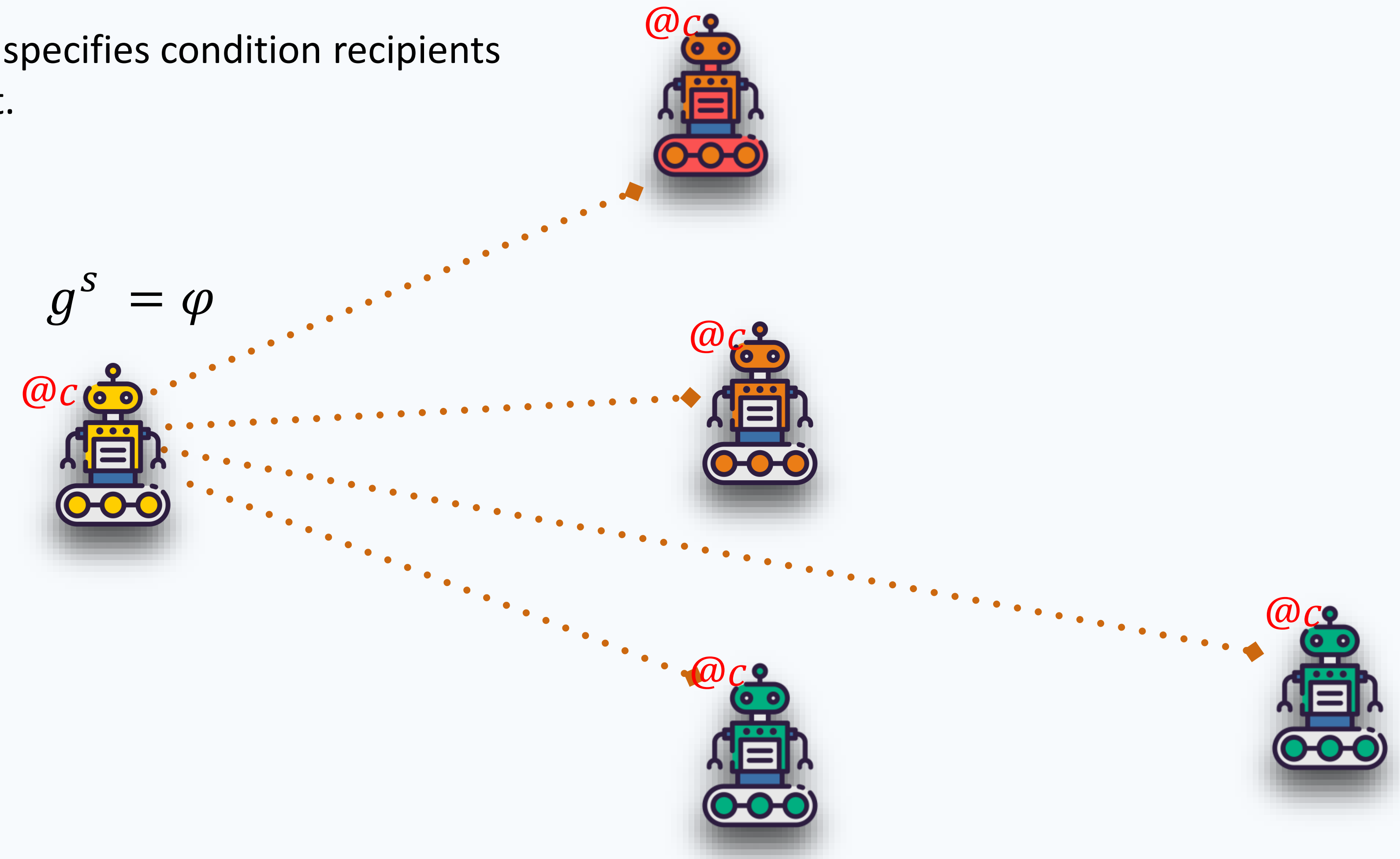
Blocking Multicast

Sending on a multicast channel does not affect unconnected agents.



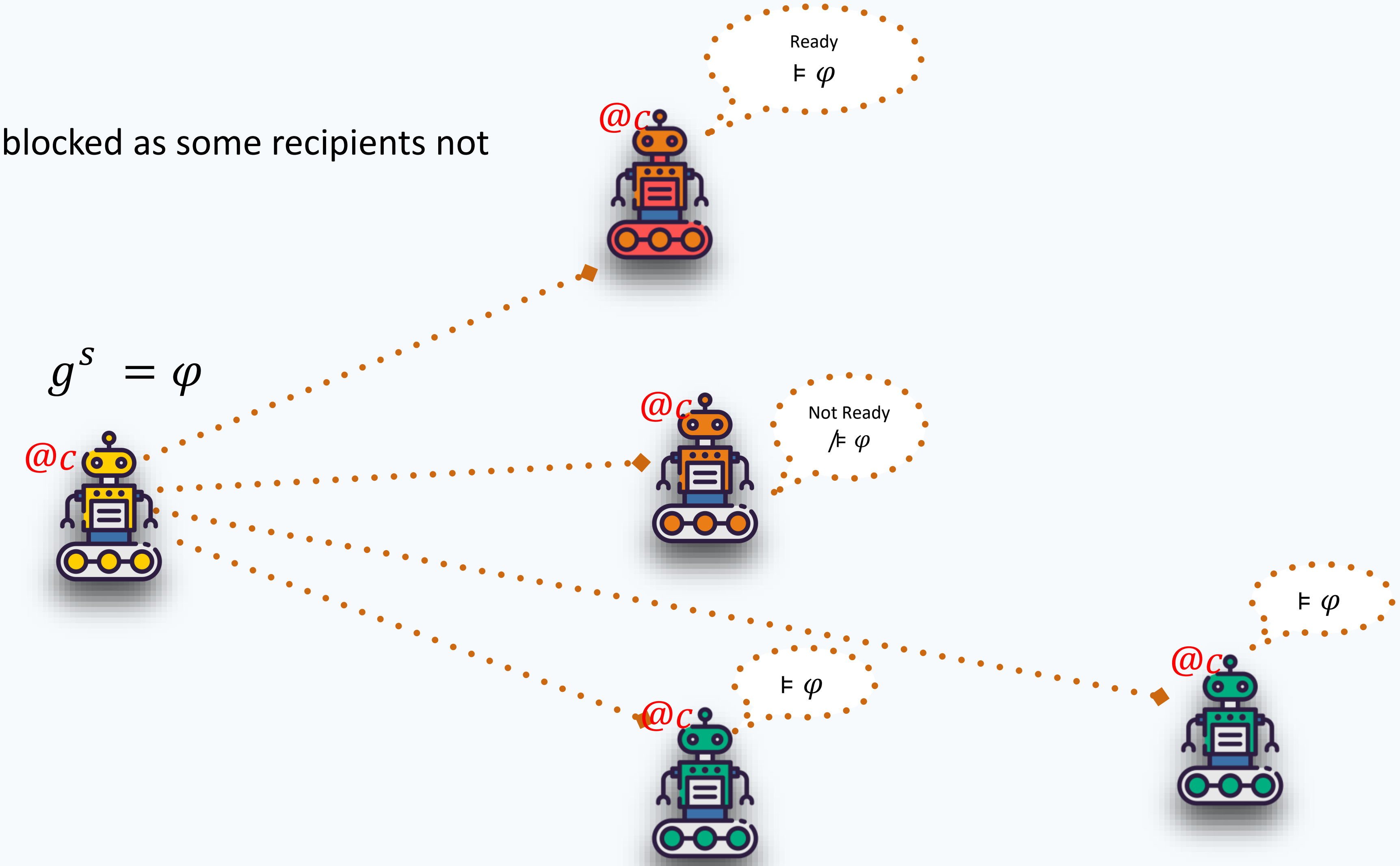
Blocking Multicast

Send guard specifies condition recipients shoud meet.



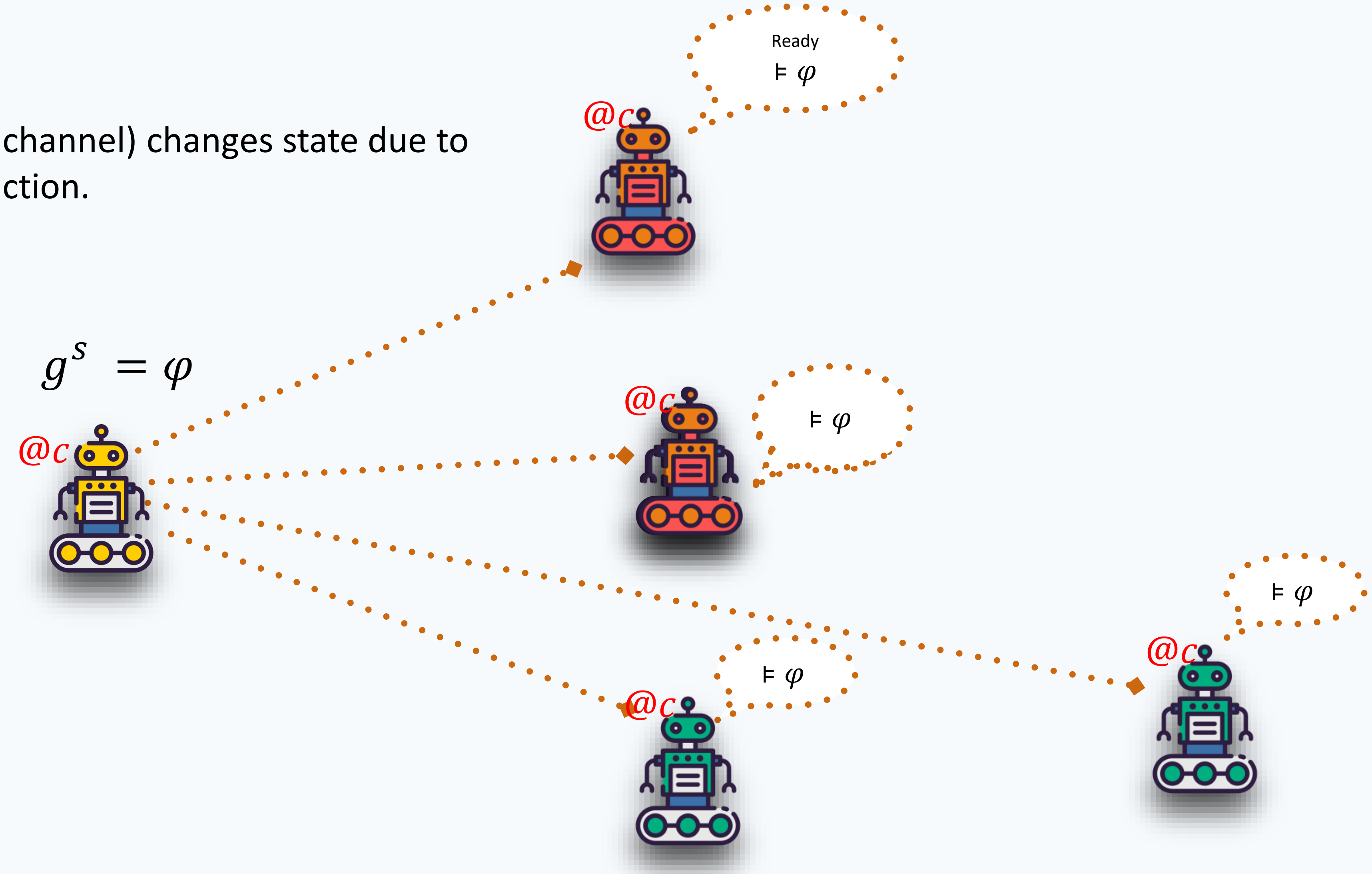
Blocking Multicast

Message is blocked as some recipients not ready.



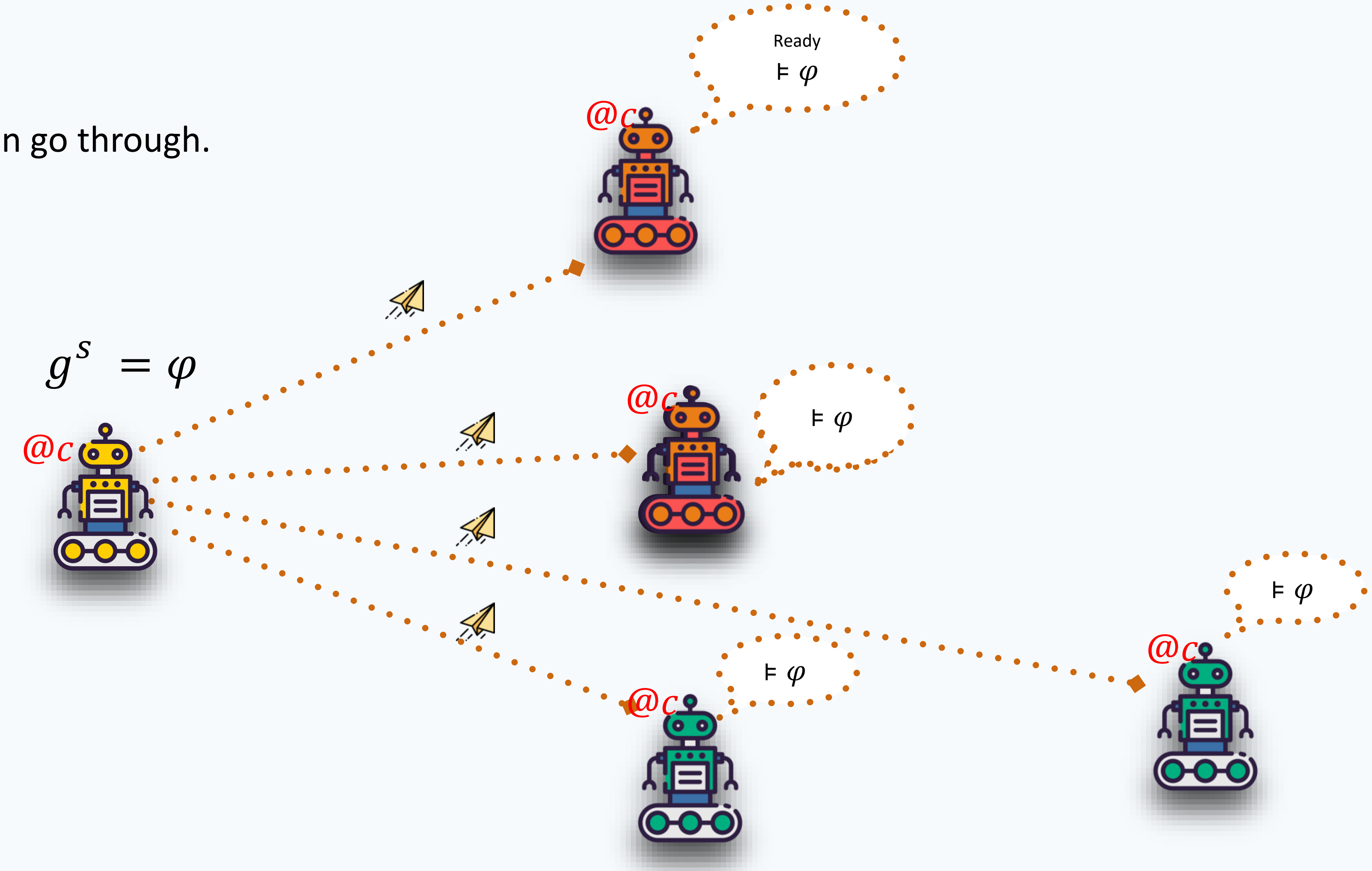
Blocking Multicast

Listener (to channel) changes state due to other interaction.



Blocking Multicast

Message can go through.



Agent Dynamics – Bottom Up

- We extend doubly-labeled transition systems.
- *Channeled Transition Systems (CTS):*

$$T = (C, \Sigma, \Upsilon^+, S, S_0, R, L, Ls)$$

  - C set of channels (including broadcast $\star$) and $Ls: S \to 2^C$ channel-listening function.
  - Σ state alphabet and $L: S \to \Sigma$ state labeling function.
  - $\Upsilon = \Upsilon^+ \times \{!, ?\} \times C$ transition alphabet and $R \subseteq S \times \Upsilon \times S$ transition relation.
- Parallel composition:
  - $L(s_1, s_2) = (L_1(s_1), L_2(s_2))$.
  - $\mathrm{Ls}(\mathrm{s}_1, \mathrm{s}_2) = \mathrm{Ls}_1(\mathrm{s}_1) \cup Ls_2(s_2)$.
  - R – synchronized send-receive, pass send/receive if not listening, pass broadcast if no option.
- Both single agent and system are same kind of TS.

Global Symbolic Dynamics

- The transition relation of the system is characterised as follows:

$$\rho : \exists ch\, \exists \mathrm{D} \bigvee_k \mathscr{T}_k^s(V_k, V_k', \mathrm{D}, ch) \wedge \bigwedge_{j \neq k} \exists \mathrm{CV}. f_j \wedge$$

$$\left(\begin{array}{lr} g_j^r(V_j, ch) \wedge \mathscr{T}_j^r(V_j, V_j', \mathrm{D}, ch) \wedge & g_k^s(V_k, ch, \mathrm{D}, \mathrm{CV}) \\ \vee & \neg g_j^r(V_j, ch) \wedge \mathsf{Id}_j \\ \vee & ch = \star \wedge \neg g_k^s(V_k, ch, \mathrm{D}, \mathrm{CV}) \wedge \mathsf{Id}_j \end{array} \right)$$

Notice the existential quantification.

Linear Time Reasoning about Transition Labels

- Information about messages hidden in transitions.
- Extend next operator(s) to refer to contents of messages.
- Inspired by Fluent LTL, branching-time logics.
- Refer to agent intentions in sending.
  - (How does this relate to knowledge?)

Extending LTL to reason about messages

Refer to the intention of an agent

$$\color{red}{O} =\cdot^{\exists} (\text{type} = \;) \wedge\cdot^{\exists} (\text{type} = \;)$$

The sender intends to interact with robots of types and

$$\varphi_1 := \langle \text{ch} = \text{c} \rangle \text{true}$$

Use channel c to send a msg

$$\varphi_2 := \langle \text{msg} = \text{recruit} \wedge \text{No} = 2 \wedge \color{red}{O} \rangle \text{true}$$

Recruit 2 robots from each type

$$\varphi_3(\text{k}) := \langle \text{msg} = \text{form} \wedge\cdot^{\exists} (\text{type} = \text{k}) \rangle \text{true}$$

Send a formation msg to a robot of type k

Refer to the interaction protocol

$\varphi_4 :=$ after φ_2 have exactly two $\varphi_3(\;)$ and two $\varphi_3(\;)$ until φ_1

After recruitment, 4 formation msgs are sent before robots can synchronise on their dedicated link c

Automata-Theoretic Model Checking

Satisfiability

Theorem

The satisfiability of LTOL *is PSPACE.*

Model Checking

Theorem

The model-checking problem of LTOL *is PSPACE.*

Conclusions and Future Work

Conclusions

- We proposed a formalism that support flexible and reconfigurable interaction interfaces for collaborative Multi-Agents Systems.
- To be able to reason about the unique interaction features of our framework, we extended LTL to consider messages and their constraints.
- We computed an PSPACE upper bound for the satisfiability and model-checking problems

Future works

- we want to provide tool support for ReCiPe and LTOL
- Consider distributed executions (M. traces?) and logics tailored for them
- Reformulate distributed synthesis in ReCiPe and LTOL settings

thank YOU so much

UNIVERSITY OF GOTHENBURG

CHALMERS UNIVERSITY OF TECHNOLOGY