# Population Protocols over Ordered Agents

Michael Blondin[1], Michaël Cadilhac[2], **Benjamin Courchesne**[1],
Lucie Guillou[3], Corto Mascle[3], Isa Vialard[4]

[1] Département d'informatique, Université de Sherbrooke, Canada

[2] DePaul University, Chicago, USA

[3] MPI for Software Systems, Kaiserslautern, Germany

[4] MPI for Software Systems, Saarbrücken, Germany

# Population protocols

# Population protocols

Distributed computing consisting of an unbounded number of passively mobile, finite-state agents

**Angluin & al. (PODC'04)**

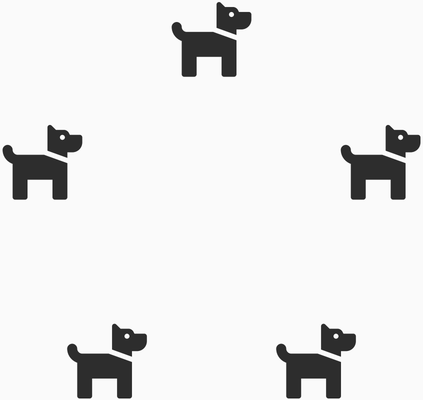
- Anonymous
- two-by-two interactions

# Population protocols

Distributed computing consisting of an unbounded number of passively mobile, finite-state ~~agents~~ dogs

**Angluin & al. (PODC’04)**

- Anonymous
- two-by-two interactions

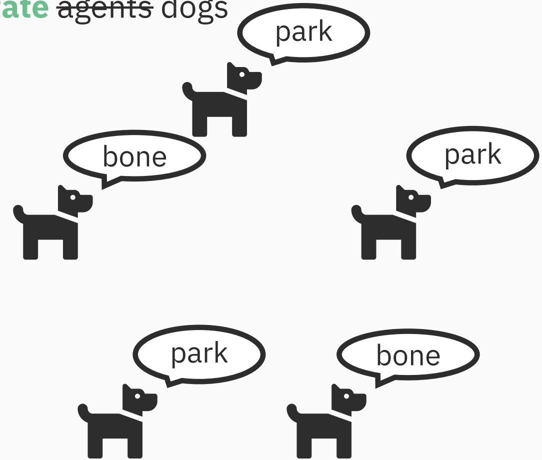


# Population protocols

Distributed computing consisting of an unbounded number of passively mobile, **finite-state** ~~agents~~ dogs

**Angluin & al. (PODC'04)**

- Anonymous
- two-by-two interactions



# Population protocols

Distributed computing consisting of an unbounded number of passively mobile, finite-state ~~agents~~ dogs

**Angluin & al. (PODC'04)**

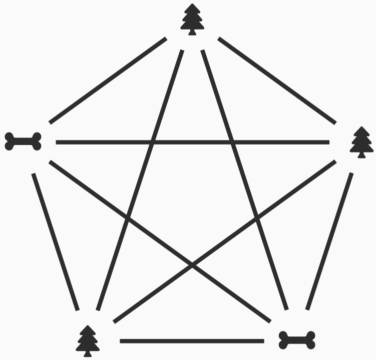
- **Anonymous**
- two-by-two interactions

# Population protocols

Distributed computing consisting of an unbounded number of passively mobile, finite-state ~~agents~~ dogs

**Angluin & al. (PODC'04)**

- Anonymous
- **two-by-two interactions**

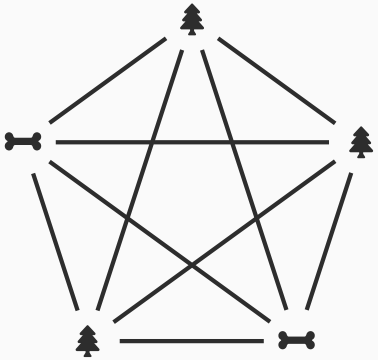


# Population protocols

Distributed computing consisting of an unbounded number of passively mobile, finite-state ~~agents~~ dogs

**Angluin & al. (PODC'04)**

- Anonymous
- **two-by-two interactions**
  (e.g. 🌲 🦴 → 🌲 🦴)

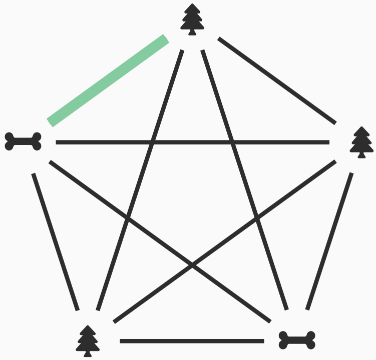


# Population protocols

Distributed computing consisting of an unbounded number of passively mobile, finite-state ~~agents~~ dogs

**Angluin & al. (PODC'04)**

- Anonymous
- two-by-two interactions
  (e.g. → )

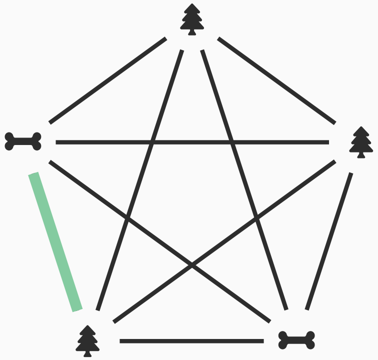


# Population protocols

Distributed computing consisting of an unbounded number of passively mobile, finite-state ~~agents~~ dogs

**Angluin & al. (PODC'04)**

- Anonymous
- two-by-two interactions (e.g. )

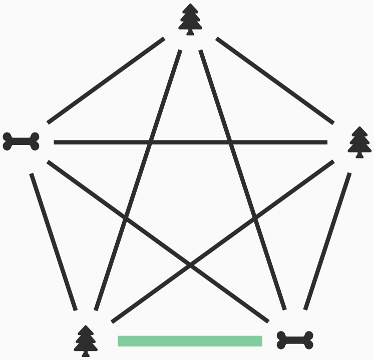


# Population protocols

Distributed computing consisting of an unbounded number of passively mobile, finite-state ~~agents~~ dogs

**Angluin & al. (PODC'04)**

- Anonymous
- two-by-two interactions
  (e.g. )

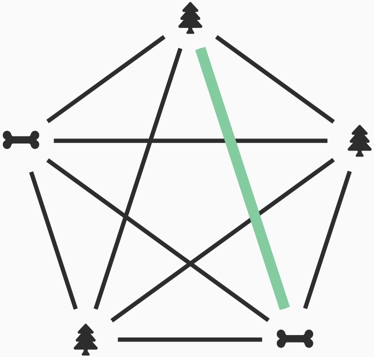


# Population protocols

Distributed computing consisting of an unbounded number of passively mobile, finite-state ~~agents~~ dogs

Angluin & al. (PODC'04)

- Anonymous
- two-by-two interactions
  (e.g. )

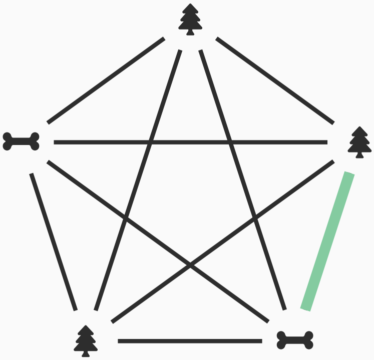


# Population protocols

Distributed computing consisting of an unbounded number of passively mobile, finite-state ~~agents~~ dogs

**Angluin & al. (PODC'04)**

- Anonymous
- two-by-two interactions (e.g. )

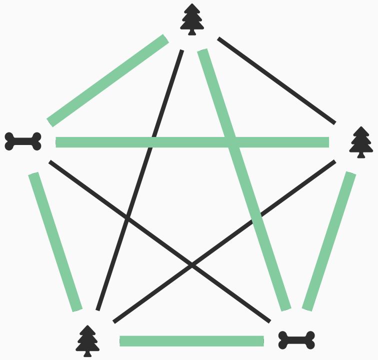


# Population protocols

Distributed computing consisting of an unbounded number of passively mobile, finite-state ~~agents~~ dogs

Angluin & al. (PODC'04)

- Anonymous
- two-by-two interactions (e.g. → )

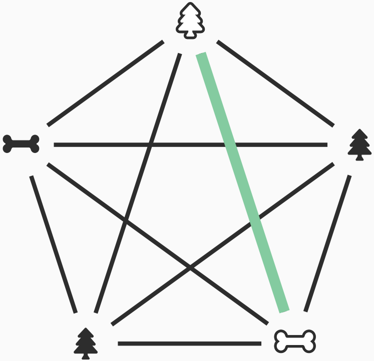


# Population protocols

Distributed computing consisting of an unbounded number of passively mobile, finite-state ~~agents~~ dogs

Angluin & al. (PODC'04)

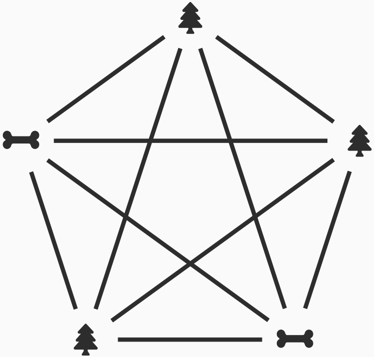
- Anonymous
- two-by-two interactions
  (e.g. )



# Example: Majority

# Example: the Majority protocol

This protocol computes whether initially there are more  or 

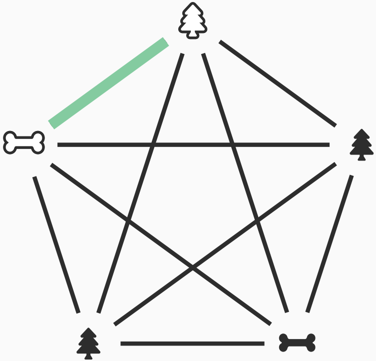


### Intuitions

- Opposite active agents become passive
- Active agent convince passive agent

# Example: the Majority protocol

This protocol computes whether initially there are more or

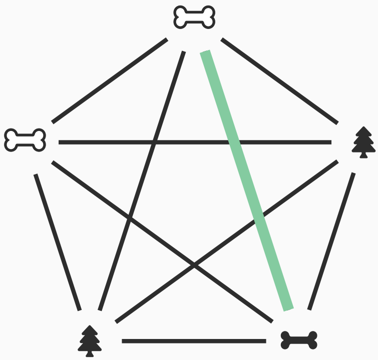


### Intuitions

- **Opposite active agents become passive**
- Active agent convince passive agent

# Example: the Majority protocol

This protocol computes whether initially there are more  or 

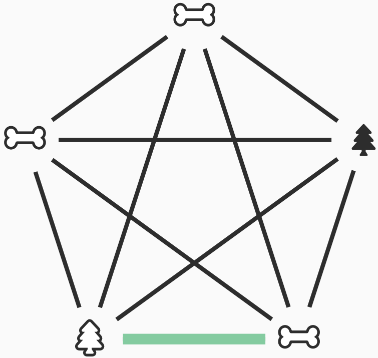


## Intuitions

- Opposite active agents become passive
- **Active agent convince passive agent**

# Example: the Majority protocol

This protocol computes whether initially there are more  or 

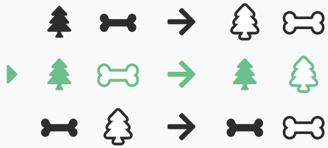


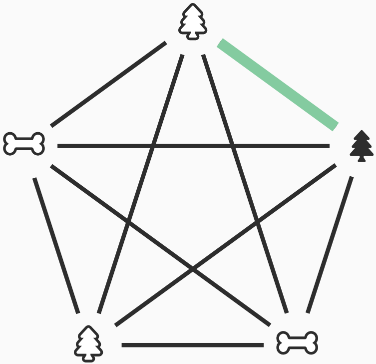
## Intuitions

- **Opposite active agents become passive**
- Active agent convince passive agent

# Example: the Majority protocol

This protocol computes whether initially there are more or



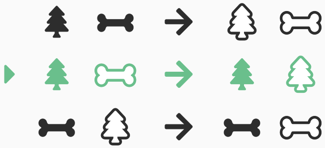


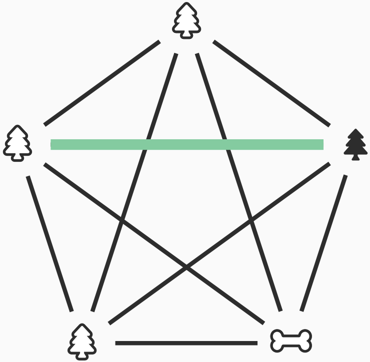
## Intuitions

- Opposite active agents become passive
- **Active agent convince passive agent**

# Example: the Majority protocol

This protocol computes whether initially there are more or



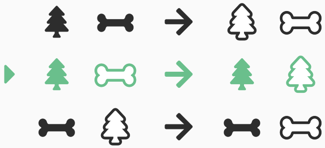


## Intuitions

- Opposite active agents become passive
- **Active agent convince passive agent**

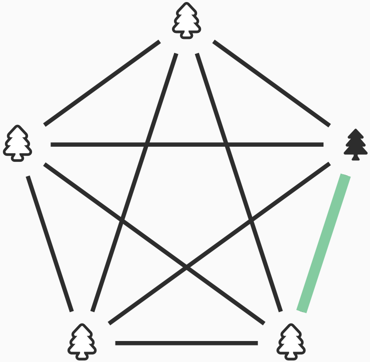
# Example: the Majority protocol

This protocol computes whether initially there are more or



## Intuitions

- Opposite active agents become passive
- **Active agent convince passive agent**



# Population protocols: formal model

| | |
|---|---|
| States: | finite set $Q$ |
| Input mapping: | $\iota\colon \Sigma \to Q$ with finite $\Sigma$ |
| Output mapping: | $O\colon Q \to \{0,1\}$ |
| Transitions: | $\to \subseteq Q^2 \times Q^2$ |

# Population protocols: formal model

**States:** **finite set $Q$**

Input mapping: $\iota\colon \Sigma \to Q$ with finite $\Sigma$

Output mapping: $O\colon Q \to \{0, 1\}$

Transitions: $\to \subseteq Q^2 \times Q^2$

# Population protocols: formal model

States: finite set $Q$

**Input mapping:** $\iota\colon \Sigma \to Q$ **with finite** $\Sigma$

Output mapping: $O\colon Q \to \{0,1\}$

Transitions: $\to \subseteq Q^2 \times Q^2$

$\iota(\text{“}park\text{”}) =$ 

$\iota(\text{“}bone\text{”}) =$ 

# Population protocols: formal model

States: finite set $Q$

Input mapping: $\iota\colon \Sigma \to Q$ with finite $\Sigma$

**Output mapping:** $O\colon Q \to \{0, 1\}$

Transitions: $\to\, \subseteq Q^2 \times Q^2$

$$
\begin{aligned}
O(\text{🌲}) &= O(\text{🌲}) &= 1\\
O(\text{🦴}) &= O(\text{🦴}) &= 0
\end{aligned}
$$

# Population protocols: formal model

States: finite set $Q$

Input mapping: $\iota\colon \Sigma \to Q$ with finite $\Sigma$

Output mapping: $O\colon Q \to \{0, 1\}$

**Transitions:** $\to\, \subseteq Q^2 \times Q^2$

# Population protocols: formal model

| | |
|---|---|
| States: | finite set $Q$ |
| Input mapping: | $\iota\colon \Sigma \to Q$ with finite $\Sigma$ |
| Output mapping: | $O\colon Q \to \{0, 1\}$ |
| Transitions: | $\to \subseteq Q^2 \times Q^2$ |

**Theorem** — Angluin & al. (PODC'06)

$L$ is computable by a population protocol iff it is semilinear.

# Population protocols: formal model

| | |
|---|---|
| States: | finite set $Q$ |
| Input mapping: | $\iota\colon \Sigma \to Q$ with finite $\Sigma$ |
| Output mapping: | $O\colon Q \to \{0, 1\}$ |
| Transitions: | $\to \subseteq Q^2 \times Q^2$ |

**Theorem** — Angluin & al. (PODC'06)

$L$ is computable by a population protocol iff it is semilinear.

- Protocol computes a predicate $\varphi\colon \mathbb{N}^\Sigma \to \{0, 1\}$ if, with probability 1, the agents eventually reach a unanimous and permanent agreement.

# Population protocols: formal model

A protocol is immediate-observation (IO-PP) if a single agent can change in a interaction, i.e. transitions are of this form :

$$p \quad q \quad \rightarrow \quad p' \quad q$$

$$p \quad q \quad \rightarrow \quad p \quad q'$$

# Population protocols: formal model

A protocol is immediate-observation (IO-PP) if a single agent can change in a interaction, i.e. transitions are of this form :

$$p \quad q \rightarrow p' \quad q$$
$$p \quad q \rightarrow p \quad q'$$

**Majority transitions** :

# Population protocols: formal model

A protocol is immediate-observation (IO-PP) if a single agent can change in a interaction, i.e. transitions are of this form :

$$p \quad q \quad \rightarrow \quad p' \quad q$$
$$p \quad q \quad \rightarrow \quad p \quad q'$$

**Majority transitions** :

non-immediate-observation

immediate-observation

Is not immediate-observation because the first transition changes both

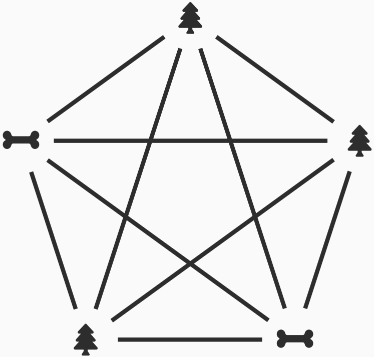
# Population protocols over ordered agents (PP[<])

**What happens if agents have unique identifiers ?**

# Population protocols over ordered agents

Distributed computing consisting of an unbounded number of passively mobile, finite-state ~~agents~~ dogs

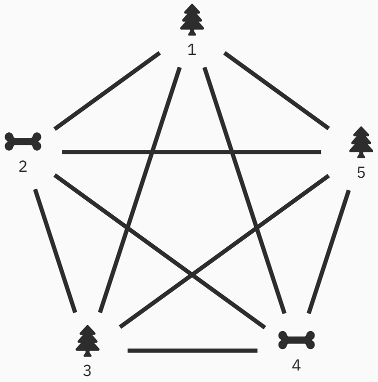
- Anonymous
- two-by-two interactions



# Population protocols over ordered agents

Distributed computing consisting of an unbounded number of passively mobile, finite-state ~~agents~~ dogs

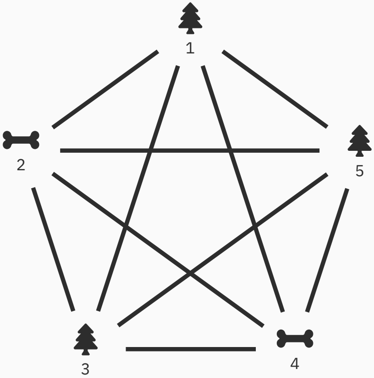
- ~~Anonymous~~ **With ordered IDs**
- two-by-two interactions



# Population protocols over ordered agents

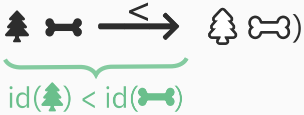
Distributed computing consisting of an unbounded number of passively mobile, finite-state ~~agents~~ dogs

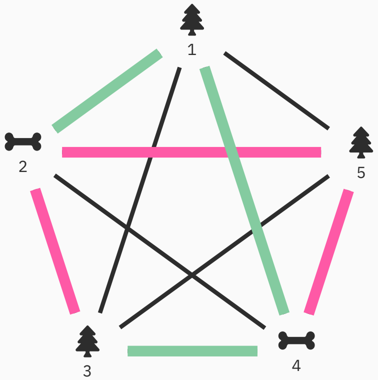
- ~~Anonymous~~ With ordered IDs
- **Ordered** two-by-two interactions (e.g. )



# Population protocols over ordered agents

Distributed computing consisting of an unbounded number of passively mobile, finite-state ~~agents~~ dogs

- ~~Anonymous~~ With ordered IDs
- Ordered two-by-two interactions
  (e.g. )


**: Possible** **: Impossible**

# Example: the growling language

# Example: a growling protocol

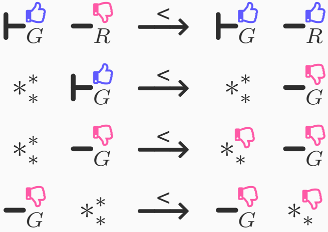
This protocol decides wether $w \in GR^*$

- $Q = \{\vdash^{\text{👍}}_{G}, -^{\text{👍}}_{R}, \vdash^{\text{👎}}_{G}, -^{\text{👎}}_{G}, -^{\text{👎}}_{R}\}$
- $\iota(G) = \vdash^{\text{👍}}_{G}, \quad \iota(R) = -^{\text{👎}}_{R}$
- $O(*^{\text{👍}}_{*}) = 1, \quad O(*^{\text{👎}}_{*}) = 0$
- Transitions :

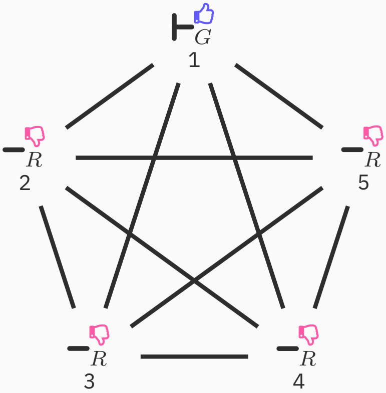
$$\vdash^{\text{👍}}_{G} \; -^{\text{👎}}_{R} \xrightarrow{<} \vdash^{\text{👍}}_{G} \; -^{\text{👍}}_{R} \;\Big|\; *^{*}_{*} \; \vdash^{\text{👍}}_{G} \xrightarrow{<} *^{*}_{*} \; -^{\text{👎}}_{G} \;\Big|\; \begin{array}{l} *^{*}_{*} \; -^{\text{👎}}_{G} \xrightarrow{<} *^{\text{👎}}_{*} \; -^{\text{👎}}_{G} \\ -^{\text{👎}}_{G} \; *^{*}_{*} \xrightarrow{<} -^{\text{👎}}_{G} \; *^{\text{👎}}_{*} \end{array}$$

# Example: a growling protocol

This protocol decides wether $w \in GR^*$

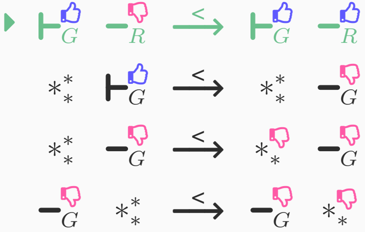


$\iota(GRRRR)$

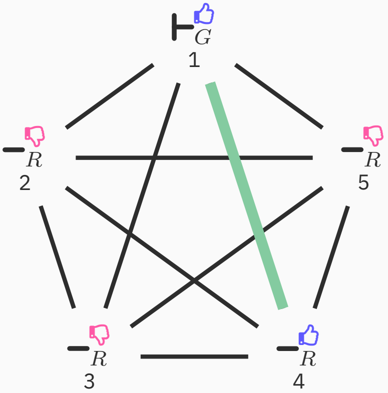


# Example: a growling protocol

This protocol decides wether $w \in GR^*$

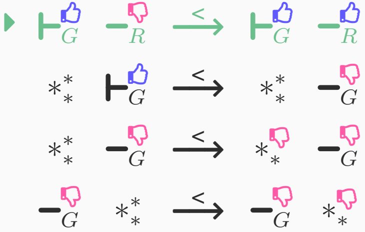



$\iota(GRRRR)$

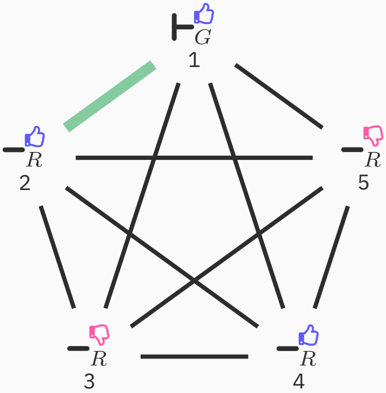


# Example: a growling protocol

This protocol decides wether $w \in GR^*$

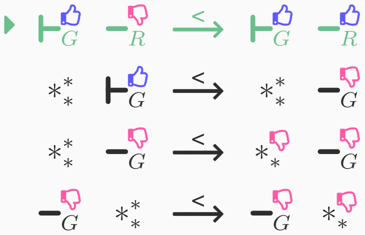


$\iota(GRRRR)$

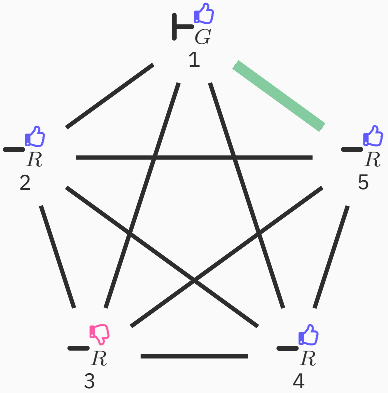


# Example: a growling protocol

This protocol decides wether $w \in GR^*$

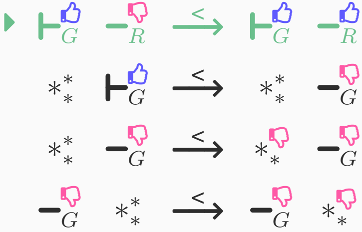



$\iota(GRRRR)$

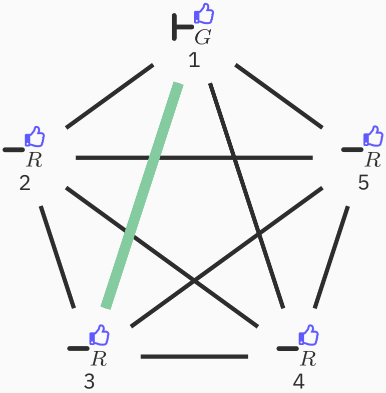


# Example: a growling protocol

This protocol decides wether $w \in GR^*$

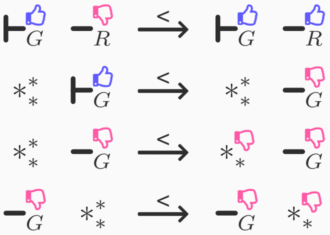



$\iota(GRRRR)$

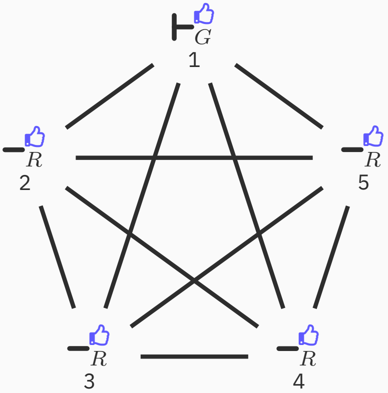


# Example: a growling protocol

This protocol decides wether $w \in GR^*$

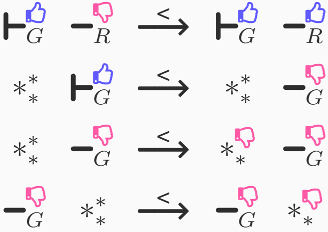


$\iota(GRRRR)$

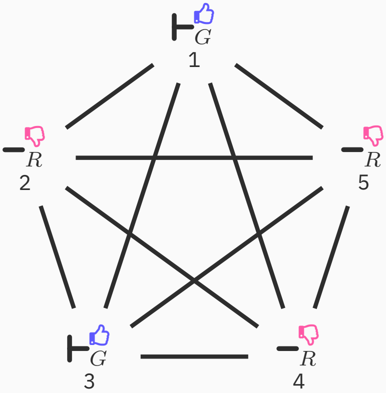


# Example: a growling protocol

This protocol decides wether $w \in GR^*$

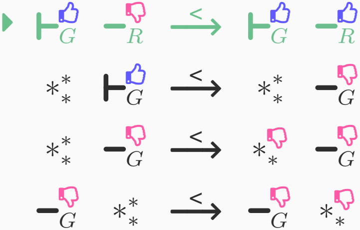



$\iota(GRGRR)$

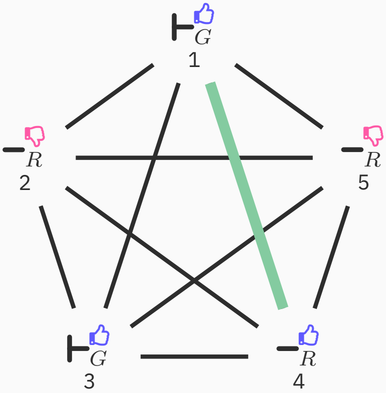


# Example: a growling protocol

This protocol decides wether $w \in GR^*$

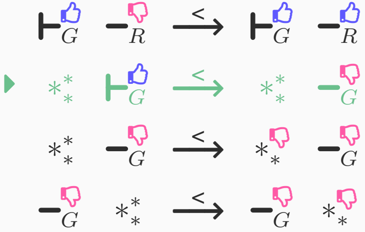


$\iota(GRGRR)$

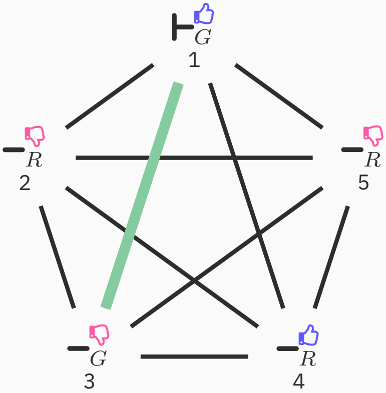


# Example: a growling protocol

This protocol decides wether $w \in GR^*$

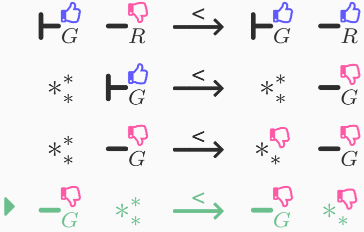


$\iota(GRGRR)$

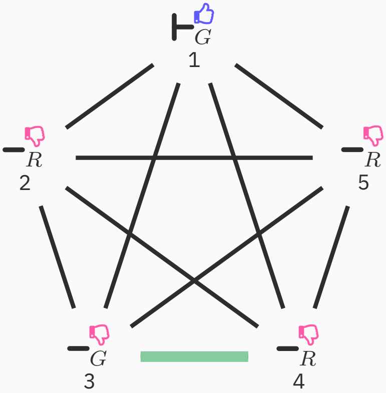


# Example: a growling protocol

This protocol decides wether $w \in GR^*$

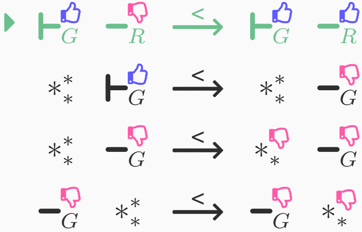


$\iota(GRGRR)$

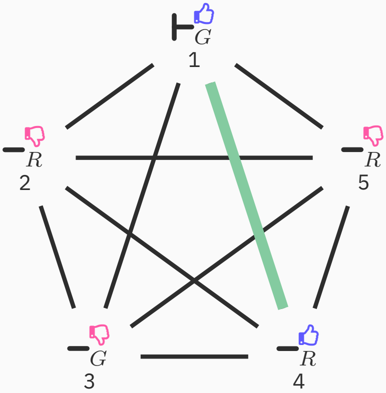


# Example: a growling protocol

This protocol decides wether $w \in GR^*$

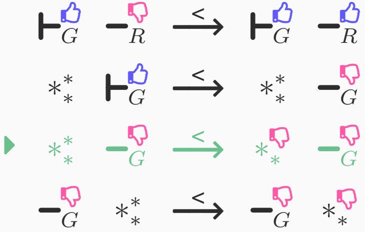



$\iota(GRGRR)$

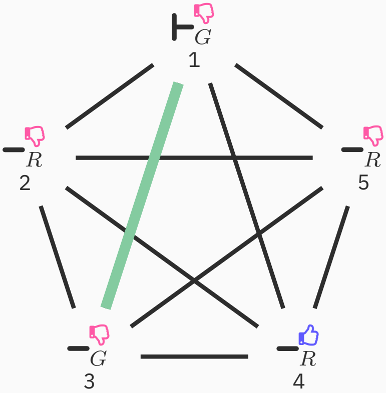


# Example: a growling protocol

This protocol decides wether $w \in GR^*$

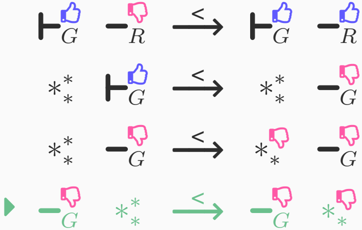



$\iota(GRGRR)$

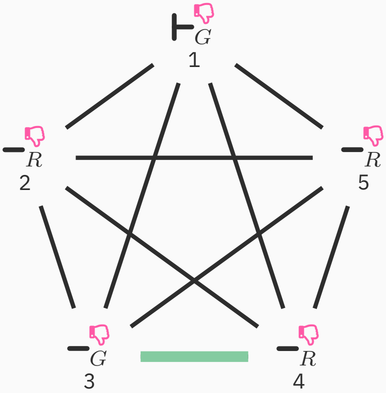


# Example: a growling protocol

This protocol decides wether $w \in GR^*$




$\iota(GRGRR)$



# Characterization of IO-PP[<]

# DA: unambiguous star-free languages

**Automata**

- partially ordered unambiguous automata
- partially ordered two-way deterministic automata

**DA**

**Algebra**

- languages recognized by monoids of the variety DA

**Logic**

- $\mathrm{LTL}[\mathbf{F}^{-1}, \mathbf{F}]$
- $\exists\forall\mathrm{FO}[<] \cap \forall\exists\mathrm{FO}[<]$
- $\mathrm{FO}^2[<]$

# DA: unambiguous star-free languages

**Automata**

- partially ordered unambiguous automata
- partially ordered two-way deterministic automata

**Algebra**

- languages recognized by monoids of the variety DA

**DA**

**Logic**

- LTL$[\mathbf{F}^{-1}, \mathbf{F}]$
- $\exists\forall\mathrm{FO}[<] \cap \forall\exists\mathrm{FO}[<]$
- FO$^2[<]$

**Distributed systems**

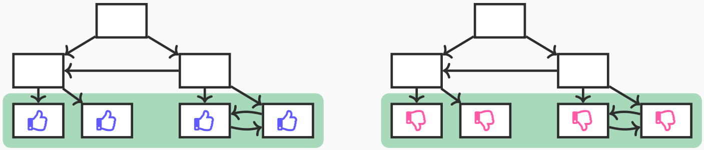
- IO-PP$[<]$

# Semi-decidability

- A protocol **decides** $L$ if, w.p. 1, the agents reach a permanent unanimous agreement

  $\boldsymbol{u} \in \boldsymbol{L} \;\Rightarrow\;$ eventually, all agents *accept* forever

  $\boldsymbol{u} \notin \boldsymbol{L} \;\Rightarrow\;$ eventually, all agents *reject* forever



*Agreement 1 with probability 1*

*Agreement 0 with probability 1*

# Semi-decidability

- A protocol decides $L$ if, w.p. 1, the agents reach a permanent unanimous agreement

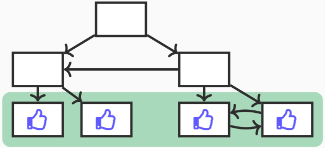
  $\boldsymbol{u} \in \boldsymbol{L} \;\Rightarrow\;$ eventually, all agents *accept* forever

  $\boldsymbol{u} \notin \boldsymbol{L} \;\Rightarrow\;$ eventually, all agents *reject* forever

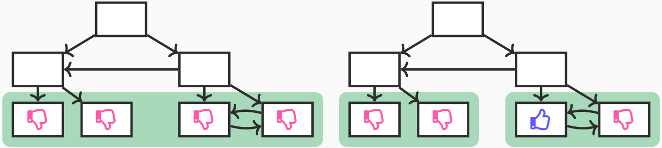
- A protocol **semi-decides** $L$ if, w.p. 1

  $\boldsymbol{u} \in \boldsymbol{L} \;\Rightarrow\;$ eventually, all agents *accept* forever

  $\boldsymbol{u} \notin \boldsymbol{L} \;\Rightarrow\;$ **never** do all agents accept forever



*Agreement 1 with probability 1*



*Agreement 0 with probability 1* *Agreement 0* *No stable agreement*

# Semi-decidability

- A protocol decides $L$ if, w.p. 1, the agents reach a permanent unanimous agreement
  $\boldsymbol{u} \in \boldsymbol{L} \;\Rightarrow\;$ eventually, all agents *accept* forever
  $\boldsymbol{u} \notin \boldsymbol{L} \;\Rightarrow\;$ eventually, all agents *reject* forever

- A protocol semi-decides $L$ if, w.p. 1
  $\boldsymbol{u} \in \boldsymbol{L} \;\Rightarrow\;$ eventually, all agents *accept* forever
  $\boldsymbol{u} \notin \boldsymbol{L} \;\Rightarrow\;$ never do all agents accept forever

**Lemma** **Our work (ICALP'26)**

If protocol and its complement are semi-deciders for $\boldsymbol{L}$ and $\Sigma^+ \setminus \boldsymbol{L}$, then $\boldsymbol{L}$ recognized by a protocol.

# IO-PP[<] $\supseteq$ DA

- $DA = \exists\forall\,\mathrm{FO}[<] \cap \forall\exists\,\mathrm{FO}[<]$ **Pin & Weil (ICALP'95)**

# IO-PP[<] $\supseteq$ DA

- $DA = \exists\forall\,\mathrm{FO}[<] \cap \forall\exists\,\mathrm{FO}[<]$

**Pin & Weil (ICALP'95)**

- $\exists\forall\mathrm{FO}[<] = \bigcup A_0^*\, a_1\, A_2^* \ldots a_{m-1}\, A_m^*$

# IO-PP[<] $\supseteq$ DA

- $DA = \exists\forall\,\mathrm{FO}[<] \cap \forall\exists\,\mathrm{FO}[<]$ **Pin & Weil (ICALP'95)**

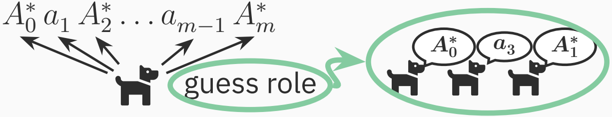
- $\exists\forall\mathrm{FO}[<] = \bigcup A_0^*\, a_1\, A_2^* \ldots a_{m-1}\, A_m^*$

guess role

# IO-PP[<] $\supseteq$ DA

- $DA = \exists\forall\,\mathrm{FO}[<] \cap \forall\exists\,\mathrm{FO}[<]$ **Pin & Weil (ICALP'95)**

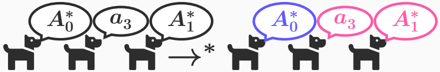
- $\exists\forall\mathrm{FO}[<] = \bigcup A_0^*\, a_1\, A_2^* \ldots a_{m-1}\, A_m^*$



# IO-PP[<] $\supseteq$ DA

- $DA = \exists\forall\, \mathrm{FO}[<] \cap \forall\exists\, \mathrm{FO}[<]$ **Pin & Weil (ICALP'95)**

- $\exists\forall\mathrm{FO}[<] = \bigcup A_0^*\, a_1\, A_2^* \ldots a_{m-1}\, A_m^*$

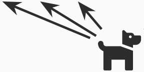



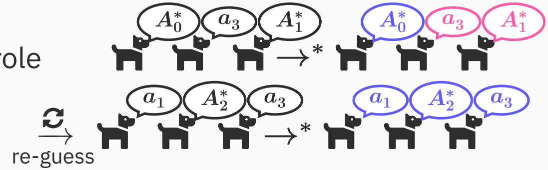
# IO-PP[<] $\supseteq$ DA

- $DA = \exists\forall\,\mathrm{FO}[<] \cap \forall\exists\,\mathrm{FO}[<]$

**Pin & Weil (ICALP'95)**

- $\exists\forall\mathrm{FO}[<] = \bigcup A_0^*\, a_1\, A_2^* \ldots a_{m-1}\, A_m^*$

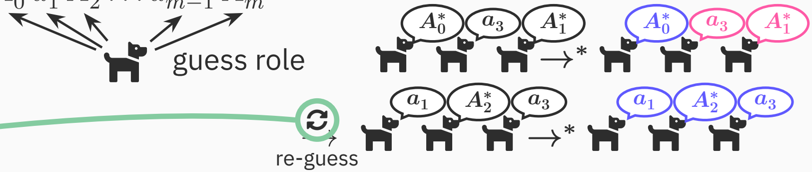


# IO-PP[<] $\supseteq$ DA

- $DA = \exists\forall\,\mathrm{FO}[<] \cap \forall\exists\,\mathrm{FO}[<]$ **Pin & Weil (ICALP'95)**

- $\exists\forall\mathrm{FO}[<] = \bigcup A_0^* \, a_1 \, A_2^* \ldots a_{m-1} \, A_m^*$



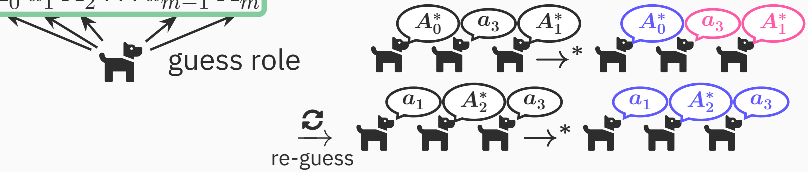

- **stabilizing input**: agents can change their mind about their input.

# IO-PP[<] ⊇ DA

- $DA = \exists\forall\,\mathrm{FO}[<] \cap \forall\exists\,\mathrm{FO}[<]$

**Pin & Weil (ICALP'95)**

- $\exists\forall\mathrm{FO}[<] = \bigcup \overbrace{\boxed{A_0^*\, a_1\, A_2^* \ldots a_{m-1}\, A_m^*}}^{\text{is IO-PP[<]-semi-decidable}}$



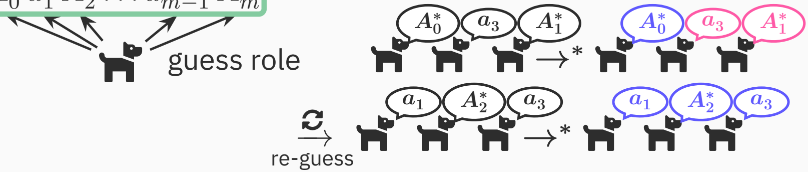

- stabilizing input: agents can change their mind about their input.

# IO-PP[<] ⊇ DA

- $DA = \exists\forall\,\mathrm{FO}[<] \cap \forall\exists\,\mathrm{FO}[<]$ **Pin & Weil (ICALP'95)**

**is IO-PP[<]-semi-decidable**

- $\exists\forall\mathrm{FO}[<] = \boxed{\bigcup A_0^*\, a_1\, A_2^* \dots a_{m-1}\, A_m^*}$




- stabilizing input: agents can change their mind about their input.

**Lemma** **Our work (ICALP'26)**

Semi-deciders are closed under ∪ and ∩

# IO-PP[<] $\subseteq$ DA

**Language $L$ is in DA iff :**

- For all $\boldsymbol{W} \equiv_{\boldsymbol{L}} \boldsymbol{W}\boldsymbol{W}, \quad \boldsymbol{W} \equiv_{\boldsymbol{L}} \boldsymbol{W}\ h\boldsymbol{W}$ with $\boldsymbol{\alpha}(\boldsymbol{h}) \subseteq \boldsymbol{\alpha}(\boldsymbol{W})$

- $L$ is regular

# IO-PP[<] $\subseteq$ DA

**Language $L$ is in DA iff :**

- For all $W \equiv_L WW$, $W \equiv_L W\, hW$ with $\alpha(h) \subseteq \alpha(W)$

$$w_1 \equiv_L w_2 \Leftrightarrow \forall x, y \cdot (x\, w_1\, y \in L \Leftrightarrow x\, w_2\, y \in L)$$

- $L$ is regular

# IO-PP[<] $\subseteq$ DA

**Language $L$ is in DA iff :**

- For all $W \equiv_L WW$, $\quad W \equiv_L W\ hW$ with $\alpha(h) \subseteq \alpha(W)$

  $w_1 \equiv_L w_2 \Leftrightarrow \forall x, y \cdot \ (x\, w_1\, y \in L \Leftrightarrow x\, w_2\, y \in L)$

  If I duplicate a group of agents, each clone can replay the original's trajectory exactly

- $L$ is regular

# IO-PP[<] $\subseteq$ DA

**Language $L$ is in DA iff :**

- For all $\boldsymbol{W} \equiv_L \boldsymbol{W}\ \boldsymbol{W}$, $\quad \boldsymbol{W} \equiv_L \boldsymbol{W}\ h\boldsymbol{W}$ with $\boldsymbol{\alpha(h)} \subseteq \boldsymbol{\alpha(W)}$

  $\boldsymbol{w_1 \equiv_L w_2 \Leftrightarrow \forall x, y \cdot (x\, w_1\, y \in L \Leftrightarrow x\, w_2\, y \in L)}$

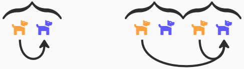
  If I duplicate a group of agents, each clone can **replay the original's trajectory exactly**

- $L$ is regular

# IO-PP[<] $\subseteq$ DA

**Language $L$ is in DA iff :**

- For all $\boldsymbol{W} \equiv_L \boldsymbol{W}\,\boldsymbol{W}$, $\quad \boldsymbol{W} \equiv_L \boldsymbol{W}\, h\boldsymbol{W}$ with $\boldsymbol{\alpha}(\boldsymbol{h}) \subseteq \boldsymbol{\alpha}(\boldsymbol{W})$



  $\boldsymbol{w_1 \equiv_L w_2 \Leftrightarrow \forall x, y \cdot (x\, w_1\, y \in L \Leftrightarrow x\, w_2\, y \in L)}$

  If I duplicate a group of agents, each clone can ~~replay~~ mimic the original's ~~trajectory exaclty~~ interaction

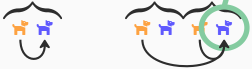
- $L$ is regular

# IO-PP[<] $\subseteq$ DA

only the observer changes, not the observed

**Language $L$ is in DA iff :**

- For all $\boldsymbol{W} \equiv_L \boldsymbol{W}\,\boldsymbol{W}$, $\boldsymbol{W} \equiv_L \boldsymbol{W}\,h\boldsymbol{W}$ with $\boldsymbol{\alpha}(\boldsymbol{h}) \subseteq \boldsymbol{\alpha}(\boldsymbol{W})$



  $\boldsymbol{w_1 \equiv_L w_2 \Leftrightarrow \forall x, y \cdot (x\,w_1\,y \in L \Leftrightarrow x\,w_2\,y \in L)}$

  If I duplicate a group of agents, each clone can ~~replay~~ mimic the original's ~~trajectory exaclty~~ interaction

- $L$ is regular

# IO-PP[<] $\subseteq$ DA

**Language $L$ is in DA iff :**

- For all $W \equiv_L WW$, $\quad \underbrace{W} \equiv_L \underbrace{W}\,\underbrace{h}\,\underbrace{W}$ with $\alpha(h) \subseteq \alpha(W)$

  $w_1 \equiv_L w_2 \Leftrightarrow \forall x, y \cdot (x\, w_1\, y \in L \Leftrightarrow x\, w_2\, y \in L)$

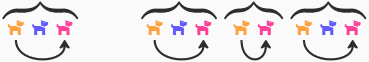
  If I insert new agents (of already existing types), they can mimic the behavior of the original group

- $L$ is regular

# IO-PP[<] $\subseteq$ DA

**Language $L$ is in DA iff :**

- For all $\boldsymbol{W} \equiv_L \boldsymbol{W}\boldsymbol{W}$, $\quad \boldsymbol{W} \;\equiv_L\; \boldsymbol{W} \;\; h \;\; \boldsymbol{W}$ with $\boldsymbol{\alpha(h)} \subseteq \boldsymbol{\alpha}(\boldsymbol{W})$



  $\boldsymbol{w_1 \equiv_L w_2} \Leftrightarrow \forall \boldsymbol{x, y} \cdot (\boldsymbol{x\, w_1\, y} \in \boldsymbol{L} \Leftrightarrow \boldsymbol{x\, w_2\, y} \in \boldsymbol{L})$

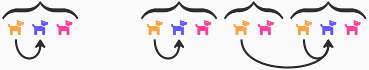
  If I insert new agents (of already existing types), they can mimic the behavior of the original group

- $L$ is regular

# IO-PP[$<$] $\subseteq$ DA

**Language $L$ is in DA iff :**

- For all $W \equiv_L WW$, $\quad W \equiv_L W \; h \; W$ with $\alpha(h) \subseteq \alpha(W)$



  $w_1 \equiv_L w_2 \Leftrightarrow \forall x, y \cdot (x\, w_1\, y \in L \Leftrightarrow x\, w_2\, y \in L)$

  If I insert new agents (of already existing types), they can mimic the behavior of the original group

- $L$ is regular

# IO-PP[<] $\subseteq$ DA

**Language $L$ is in DA iff :**

- For all $W \equiv_L WW$, $\quad \underbrace{W}\ \equiv_L\ \underbrace{W}\ \underbrace{h}\ \underbrace{W}$ with $\alpha(h) \subseteq \alpha(W)$

  $w_1 \equiv_L w_2 \Leftrightarrow \forall x, y \cdot (x\, w_1\, y \in L \Leftrightarrow x\, w_2\, y \in L)$

  If I insert new agents (of already existing types), they can mimic the behavior of the original group

- $L$ is regular
  - Every word long enough contains a pattern that can be pumped to produce a new word in the same equivalence class.
  - Finitely many equivalence classes $\Rightarrow L \in$ IO-PP[<] is regular.

# DA: unambiguous star-free languages

**Automata**

- partially ordered unambiguous automata
- partially ordered two-way deterministic automata

**Algebra**

- languages recognized by monoids of the variety DA

**DA**

**Logic**

- LTL$[\mathbf{F}^{-1}, \mathbf{F}]$
- $\exists\forall\mathrm{FO}[<] \cap \forall\exists\mathrm{FO}[<]$
- FO$^2[<]$

**Distributed systems**

- IO-PP$[<]$

Our work (ICALP’26)

# Beyond IO-PP[<]

# PP[<] $\supseteq$ p.o. Parikh automata

**Theorem** — **Our work (ICALP'26)**

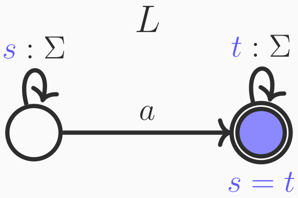
If $L$ and $\overline{L}$ are both recognized by p.o. Parikh automata, then $L$ is decided by a PP[<].

# PP[<] $\supseteq$ p.o. Parikh automata

**Theorem** — **Our work (ICALP'26)**

If $L$ and $\overline{L}$ are both recognized by p.o. Parikh automata, then $L$ is decided by a PP[<].

**E.g. Median language** : $L = \bigcup_{n \in \mathbb{N}} \Sigma^n a \Sigma^n$

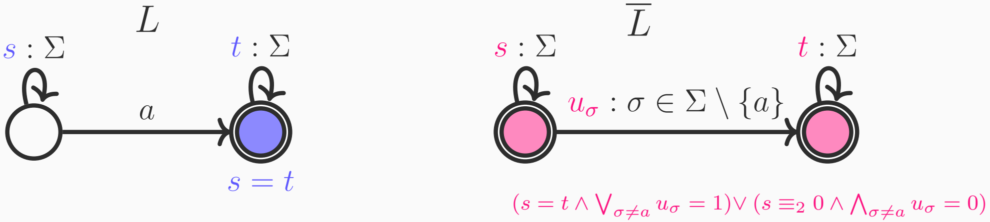


# PP[<] $\supseteq$ p.o. Parikh automata

**Theorem** — **Our work (ICALP'26)**

If $L$ and $\overline{L}$ are both recognized by p.o. Parikh automata, then $L$ is decided by a PP[<].

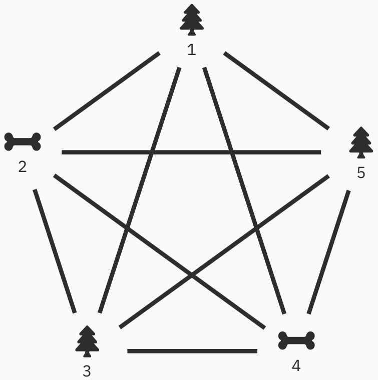
**E.g. Median language** : $L = \bigcup_{n\in\mathbb{N}} \Sigma^n a \Sigma^n$



# PP[+1] = NSPACE(n)

Distributed computing consisting of an unbounded number of passively mobile, finite-state ~~agents~~ dogs

- ~~Anonymous~~ With ordered IDs
- Ordered two-by-two interactions (e.g. $\xrightarrow{+1}$ )

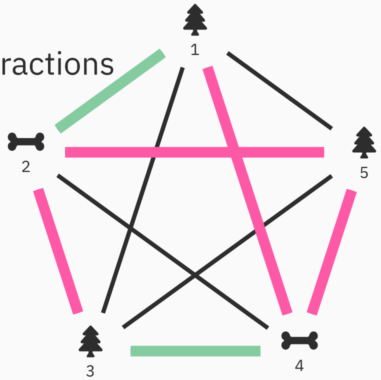


# PP[+1] = NSPACE(n)

Distributed computing consisting of an unbounded number of passively mobile, finite-state ~~agents~~ dogs

- ~~Anonymous~~ With ordered IDs
- **Neighbor** Ordered two-by-two interactions (e.g. 🌲 🦴 $\xrightarrow{+1}$ 🌲 🦴)

  id(🦴) = id(🌲) + 1

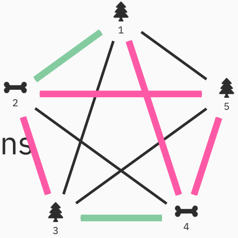



●: **Possible** ●: **Impossible**

# PP[+1] $=$ NSPACE(n)

Distributed computing consisting of an unbounded number of passively mobile, finite-state ~~agents~~ dogs

- ~~Anonymous~~ With ordered IDs
- Neighbor Ordered two-by-two interactions
  (e.g. $\xrightarrow{+1}$ )



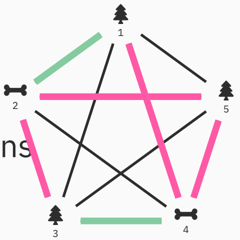

**Theorem** **Our work (ICALP'26)**

- Any $L \in$ NSPACE(n) is decidable by a IO-PP[+1]
- Any $L$ decided by a PP[{+1, <}] belongs to NSPACE(n)

# PP[+1] $=$ NSPACE(n)

Distributed computing consisting of an unbounded number of passively mobile, finite-state ~~agents~~ dogs

- ~~Anonymous~~ With ordered IDs
- Neighbor Ordered two-by-two interactions
  (e.g. $\xrightarrow{+1}$ )




**Theorem** — **Our work (ICALP'26)**

- Any $L \in$ NSPACE(n) is decidable by a IO-PP[+1]
- Any $L$ decided by a PP[{+1, <}] belongs to NSPACE(n)

PP[+1] $=$ IO-PP[+1] $=$ NSPACE(n)

# Conclusion

# Conclusion

## Summary

- Introduction to population protocols over ordered agents
- $\text{IO-PP}[<] = \text{DA}$
- $\text{IO-PP}[+1] = \text{PP}[+1] = \text{NSPACE}(n)$

## Future work

- Characterization of PP[<]

  Expressiveness ?

  State complexity ?

  Expected time complexity ?

# Conclusion

## Summary

- Introduction to population protocols over ordered agents
- IO-PP[<] = DA
- IO-PP[+1] = PP[+1] = NSPACE(n)

## Future work

- Characterization of PP[<]

  Expressiveness ?

  State complexity ?

  Expected time complexity ?

Are unambiguous p.o. Parikh automata closed under complementation ?

# Thank you !