Black Ninjas in the Dark: Formal Analysis of Population Protocols

Javier Esparza

Joint work with Michael Blondin, Pierre Ganty, Stefan Jaax, Antonín Kučera, Jérôme Leroux, Rupak Majumdar, Philipp J. Meyer, and Chana Weil-Kennedy

Technical University of Munich

European Research Council
Established by the European Commission

Deaf Black Ninjas in the Dark

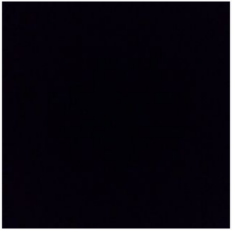
- Deaf Black Ninjas meet at a Zen garden in the dark

Deaf Black Ninjas in the Dark

- Deaf Black Ninjas meet at a Zen garden in the dark
- They must decide by majority to attack or not (no attack if tie)

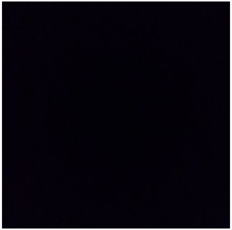
Deaf Black Ninjas in the Dark

- Deaf Black Ninjas meet at a Zen garden in the dark
- They must decide by majority to attack or not (no attack if tie)



Deaf Black Ninjas in the Dark

- Deaf Black Ninjas meet at a Zen garden in the dark
- They must decide **by majority** to attack or not (no attack if tie)
- **How can they conduct the vote?**



Deaf Black Ninjas in the Dark

- Ninjas wander randomly, interacting when they bump into each other.

Deaf Black Ninjas in the Dark

- Ninjas wander randomly, interacting when they bump into each other.
- Ninjas store their current estimation of the final outcome: **attack** or **don't attack**.

Deaf Black Ninjas in the Dark

- Ninjas wander randomly, interacting when they bump into each other.
- Ninjas store their current estimation of the final outcome: **attack** or **don't attack**.
- Additionally, they are active or passive .

attack
active

don't attack
active

attack
passive

don't attack
passive

Deaf Black Ninjas in the Dark

- Ninjas wander randomly, interacting when they bump into each other.
- Ninjas store their current estimation of the final outcome: **attack** or **don't attack**.
- Additionally, they are active or passive .

attack
active

don't attack
active

attack
passive

don't attack
passive

- Initially: all ninjas active, estimation = own vote.

Goal of voting protocol:

- eventually all ninjas reach the same estimation, and
- this estimation corresponds to the majority.

Deaf Black Ninjas in the Dark

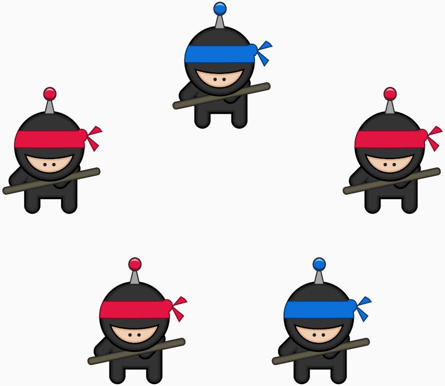
Goal of voting protocol:

- eventually all ninjas reach the same estimation, and
- this estimation corresponds to the majority.

Graphically:

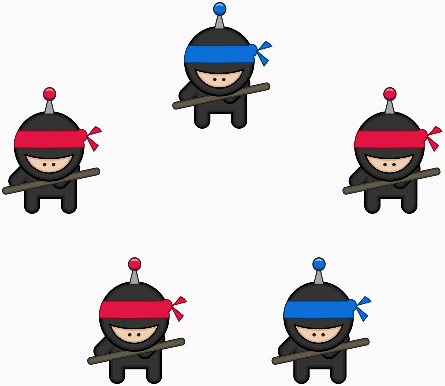
- Initially more red ninjas $\Longrightarrow$ eventually all ninjas red.
- Initially more blue ninjas or tie $\Longrightarrow$ eventually all ninjas blue.

Majority protocol: Are there more red ninjas than blue ninjas?



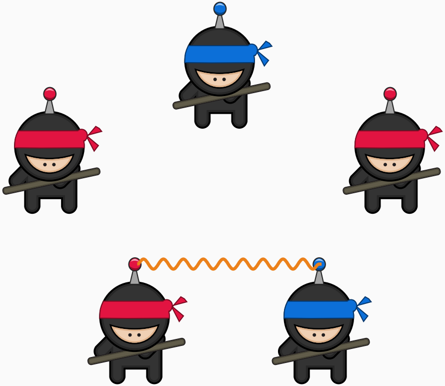
Majority protocol: Are there more red ninjas than blue ninjas?

- Active ninjas of opposite colors become passive and blue



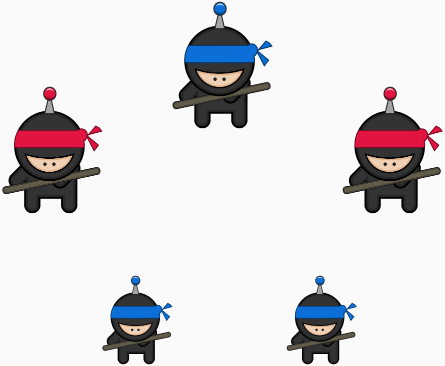
Majority protocol: Are there more red ninjas than blue ninjas?

- Active ninjas of opposite colors become passive and blue



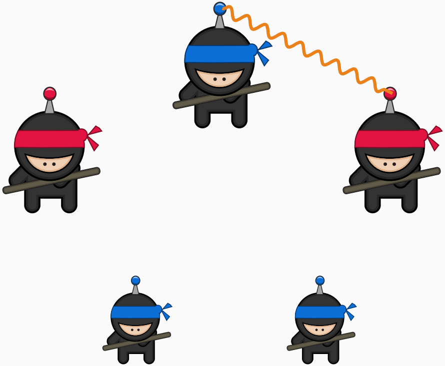
Majority protocol: Are there more red ninjas than blue ninjas?

- Active ninjas of opposite colors become passive and blue



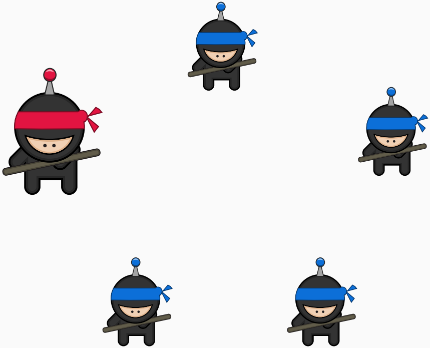
Majority protocol: Are there more red ninjas than blue ninjas?

- Active ninjas of opposite colors become passive and blue



Majority protocol: Are there more red ninjas than blue ninjas?

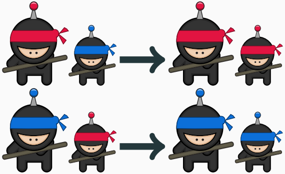
- Active ninjas of opposite colors become passive and blue

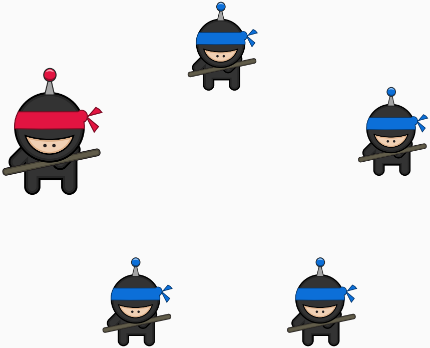


Majority protocol: Are there more red ninjas than blue ninjas?

- Active ninjas of opposite colors become passive and blue

- Active ninjas convert passive ninjas to their color

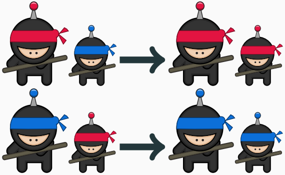


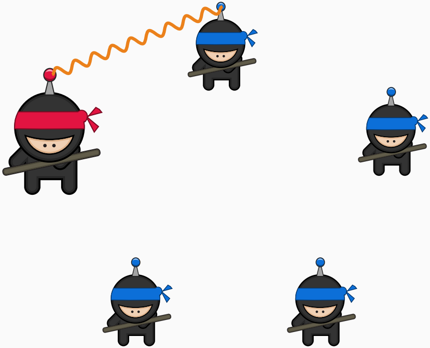


Majority protocol: Are there more red ninjas than blue ninjas?

- Active ninjas of opposite colors become passive and blue

- Active ninjas convert passive ninjas to their color

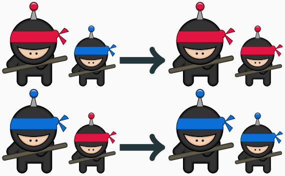


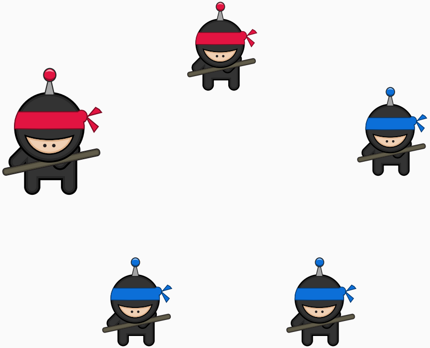


Majority protocol: Are there more red ninjas than blue ninjas?

- Active ninjas of opposite colors become passive and blue

- Active ninjas convert passive ninjas to their color

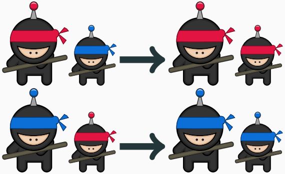


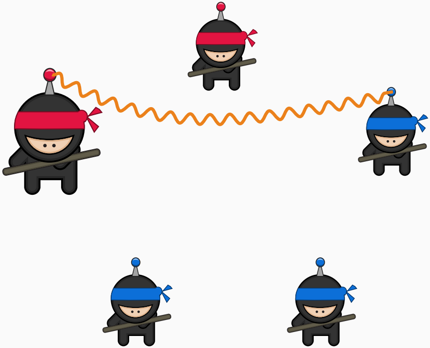


Majority protocol: Are there more red ninjas than blue ninjas?

- Active ninjas of opposite colors become passive and blue

- Active ninjas convert passive ninjas to their color

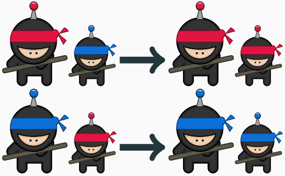


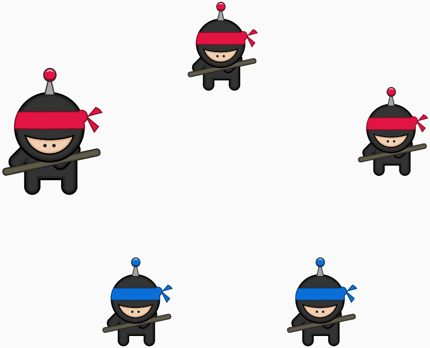


Majority protocol: Are there more red ninjas than blue ninjas?

- Active ninjas of opposite colors become passive and blue

- Active ninjas convert passive ninjas to their color

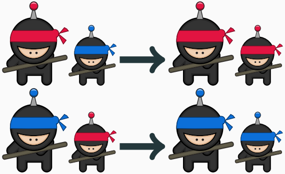


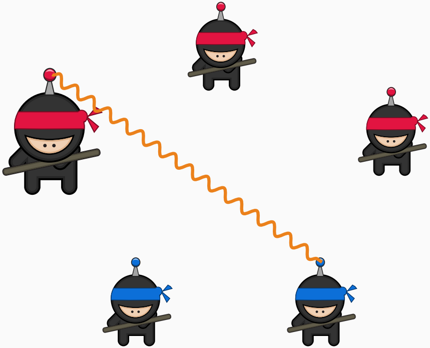


Majority protocol: Are there more red ninjas than blue ninjas?

- Active ninjas of opposite colors become passive and blue

- Active ninjas convert passive ninjas to their color

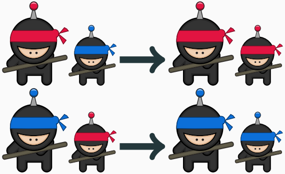


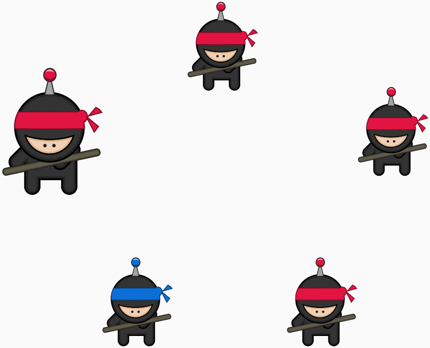


Majority protocol: Are there more red ninjas than blue ninjas?

- Active ninjas of opposite colors become passive and blue

- Active ninjas convert passive ninjas to their color

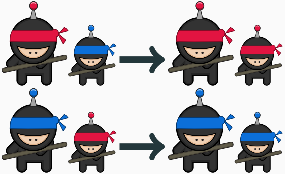


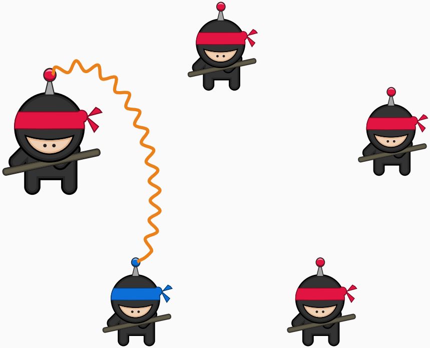


Majority protocol: Are there more red ninjas than blue ninjas?

- Active ninjas of opposite colors become passive and blue

- Active ninjas convert passive ninjas to their color

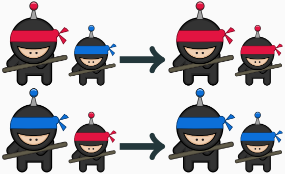


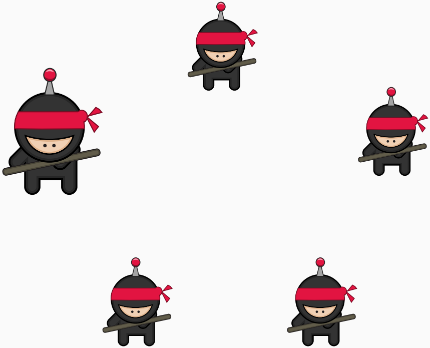


Majority protocol: Are there more red ninjas than blue ninjas?

- Active ninjas of opposite colors become passive and blue

- Active ninjas convert passive ninjas to their color

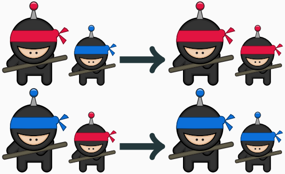




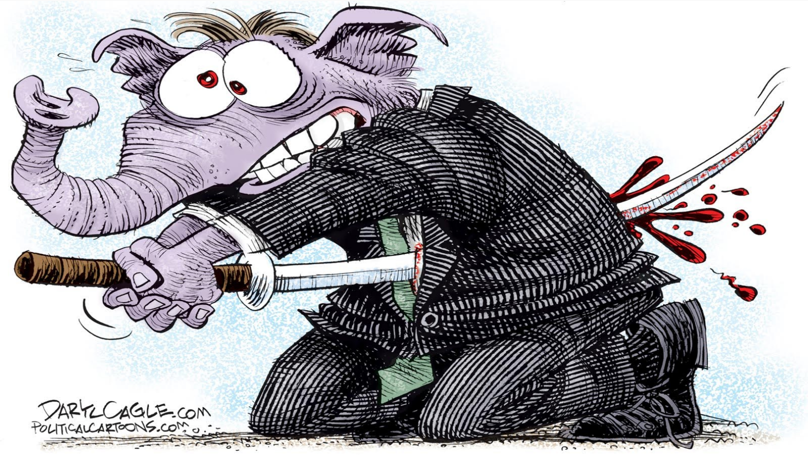
Majority protocol: Are there more red ninjas than blue ninjas?

- Active ninjas of opposite colors become passive and blue

- Active ninjas convert passive ninjas to their color



Sad story ...



Sensei II

Majority protocol: Why?

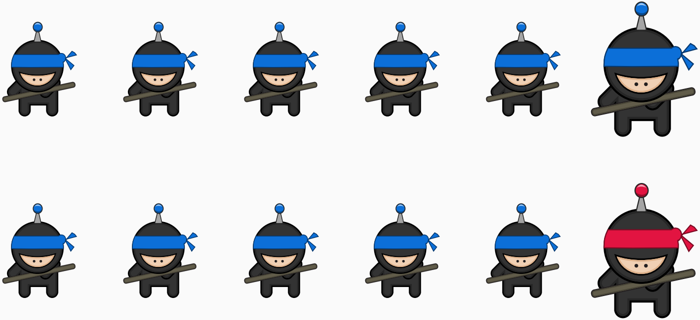
- The first rule has no priority over the other two.

Majority protocol: Why?

- The first rule has no priority over the other two.

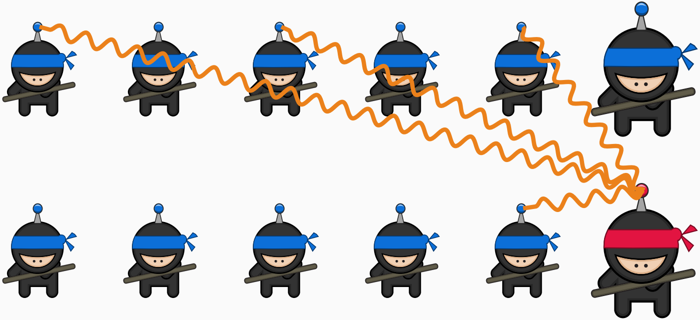
Majority protocol: Why?

- The first rule has no priority over the other two.



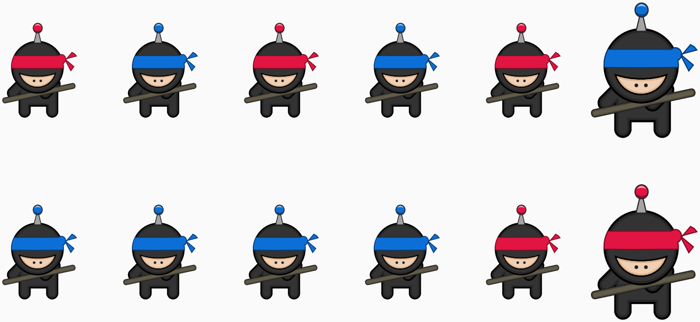
Majority protocol: Why?

- The first rule has no priority over the other two.



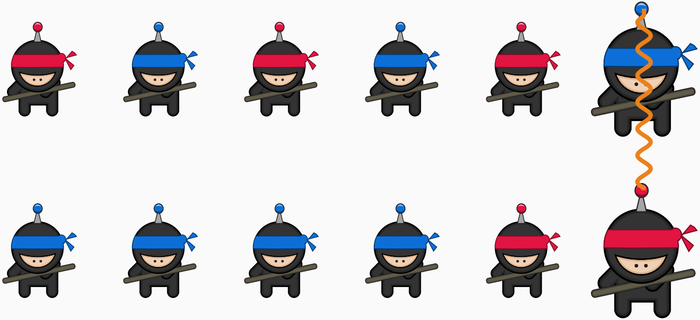
Majority protocol: Why?

- The first rule has no priority over the other two.



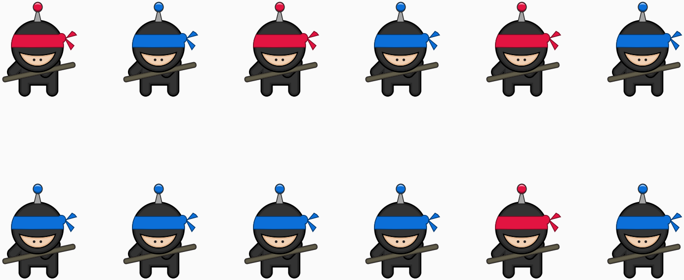
Majority protocol: Why?

- The first rule has no priority over the other two.



Majority protocol: Why?

- The first rule has no priority over the other two.



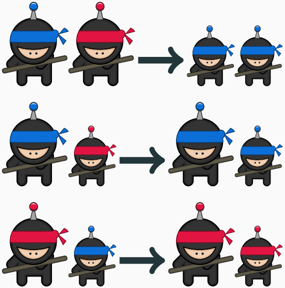
Majority protocol: Why?

- The first rule has no priority over the other two.

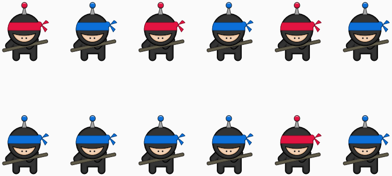
NO CONSENSUS!

Sensei II's protocol: Are there more red ninjas than blue ninjas?

Interaction rules:

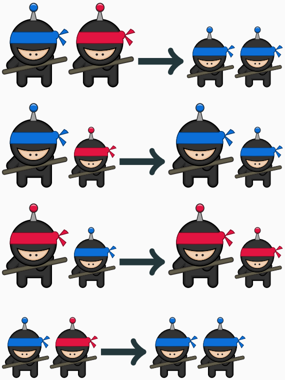


Sensei II



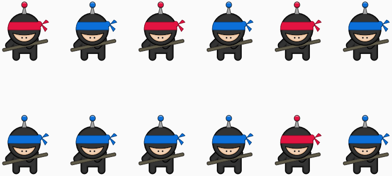
Sensei II's protocol: Are there more red ninjas than blue ninjas?

Interaction rules:



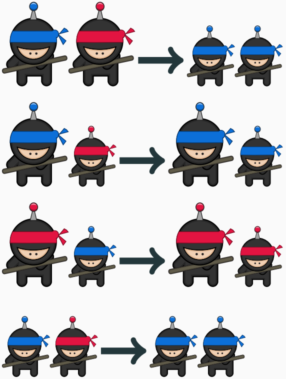
Passive blue ninjas convert passive red ninjas to their color

Sensei II



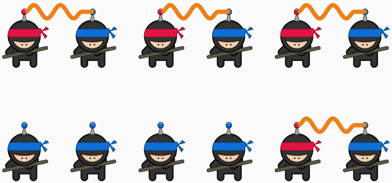
Sensei II's protocol: Are there more red ninjas than blue ninjas?

Interaction rules:



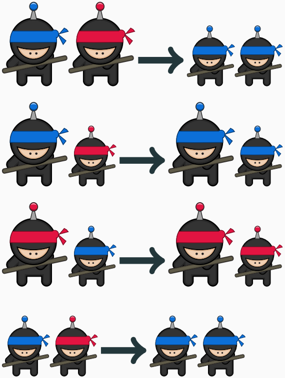
Passive blue ninjas convert passive red ninjas to their color

Sensei II



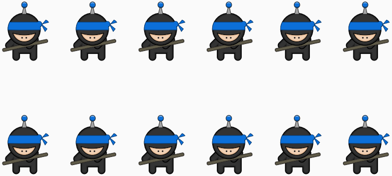
Sensei II's protocol: Are there more red ninjas than blue ninjas?

Interaction rules:



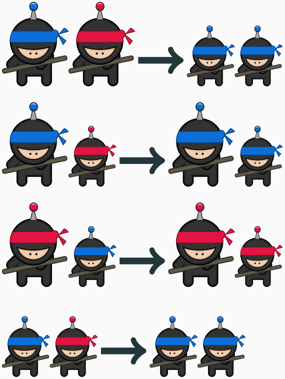
Passive blue ninjas convert passive red ninjas to their color

Sensei II



Sensei II's protocol: Are there more red ninjas than blue ninjas?

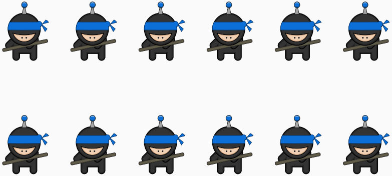
Interaction rules:



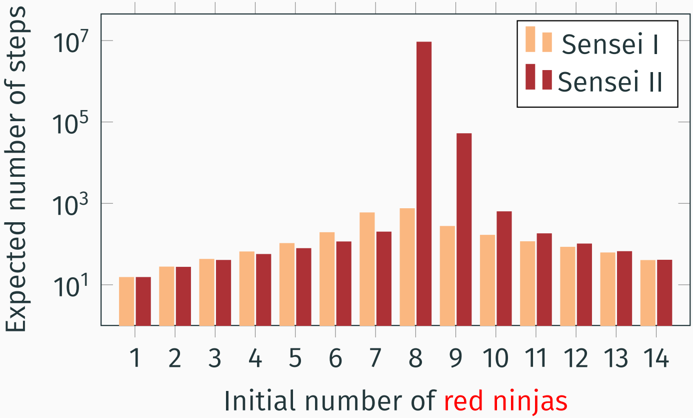
Passive blue ninjas convert passive red ninjas to their color

Sensei II

Go!



Sensei II's protocol: Are there more red ninjas than blue ninjas?



Expected number of steps to stable consensus for a population of 15 ninjas.

Very sad story ...

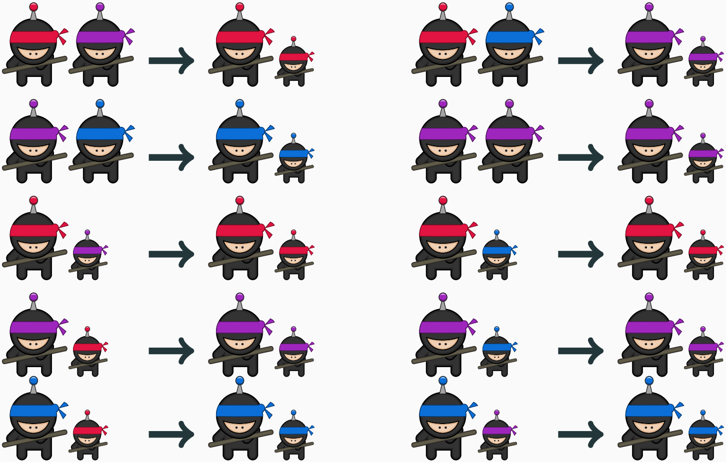
Sensei III

Sensei III's protocol

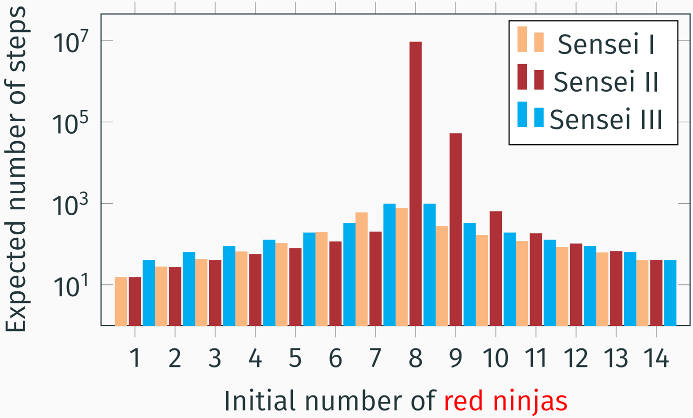
 = Attack majority  = Don't attack majority  = Tie

Interaction rules:

▸ Go!



Sensei III's protocol



Expected number of steps to stable consensus for a population of 15 ninjas.

Sensei III's questions

Formalization questions:

- What is a protocol?
- When is a protocol "correct"?
- When is a protocol "efficient"?

Sensei III's questions

Verification questions:

- *How do I check that my protocol is correct?*
- *How do I check that my protocol is efficient?*

Sensei III's questions

Expressivity questions:

- Are there protocols for other problems?
- How large is the smallest protocol for a problem?
- And the smallest efficient protocol?

Population protocols

Angluin, Aspnes *et al.* **PODC'04**

Formal model of distributed computation by collections of

identical, finite-state, and mobile agents

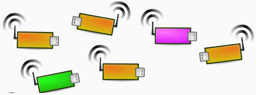
like

Population protocols

Formal model of distributed computation by collections of

identical, finite-state, and mobile agents

like



ad-hoc networks of mobile sensors

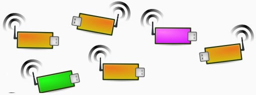
Population protocols

Angluin, Aspnes *et al.* PODC'04

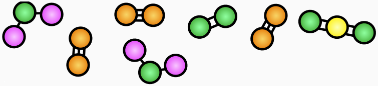
Formal model of distributed computation by collections of

identical, finite-state, and mobile agents

like



ad-hoc networks of mobile sensors



"soups" of molecules (Chemical Reaction Networks)

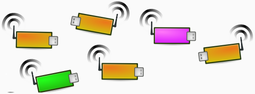
Population protocols

Angluin, Aspnes *et al.* PODC'04

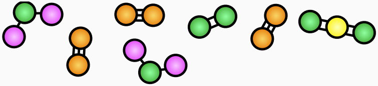
Formal model of distributed computation by collections of

identical, finite-state, and mobile agents

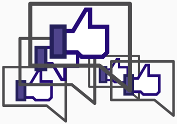
like



ad-hoc networks of mobile sensors



"soups" of molecules (Chemical Reaction Networks)



people in social networks

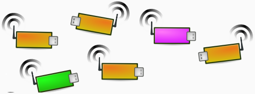
Population protocols

Angluin, Aspnes *et al.* **PODC'04**

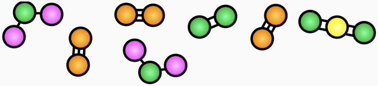
Formal model of distributed computation by collections of

identical, finite-state, and mobile agents

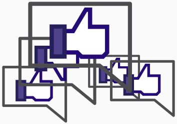
like



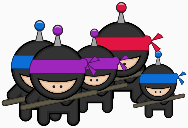
ad-hoc networks of mobile sensors



"soups" of molecules (Chemical Reaction Networks)



people in social networks



...and ninjas!

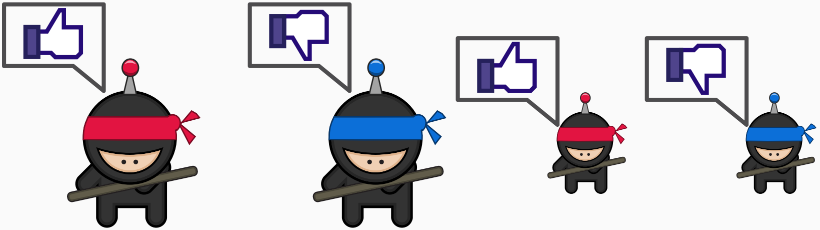
Population protocols: formal model

Angluin, Aspnes *et al.* PODC'04

- *States:* finite set Q
- *Opinions:* $O : Q \to \{0, 1\}$
- *Initial states:* $I \subseteq Q$
- *Transitions:* $T \subseteq Q^2 \times Q^2$

Population protocols: formal model

- *States*: finite set Q
- *Opinions*: $O : Q \to \{0, 1\}$
- *Initial states*: $I \subseteq Q$
- *Transitions*: $T \subseteq Q^2 \times Q^2$

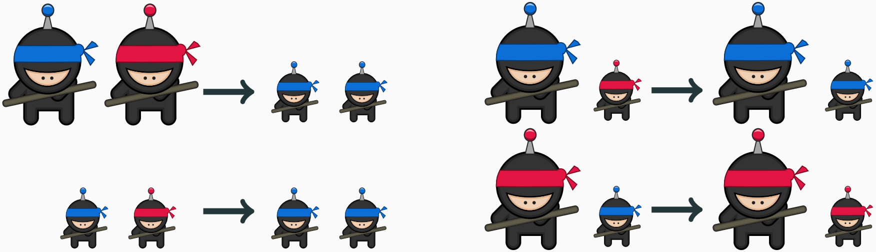


- *States*: finite set Q
- *Opinions*: $O : Q \to \{0, 1\}$
- *Initial states*: $I \subseteq Q$
- *Transitions*: $T \subseteq Q^2 \times Q^2$

Population protocols: formal model

Angluin, Aspnes *et al.* PODC'04

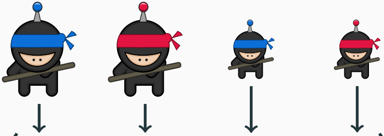
- *States*: finite set Q
- *Opinions*: $O : Q \to \{0, 1\}$
- *Initial states*: $I \subseteq Q$
- *Transitions*: $T \subseteq Q^2 \times Q^2$



Population protocols: formal model

Angluin, Aspnes *et al.* PODC'04

- *States*: finite set Q
- *Opinions*: $O : Q \to \{0, 1\}$
- *Initial states*: $I \subseteq Q$
- *Transitions*: $T \subseteq Q^2 \times Q^2$
- *Configurations*: $Q \to \mathbb{N}$

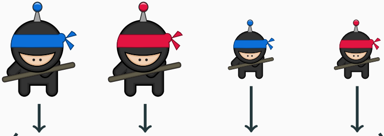


$(2 \quad 5 \quad 1 \quad 3)$

Population protocols: formal model

Angluin, Aspnes *et al.* PODC'04

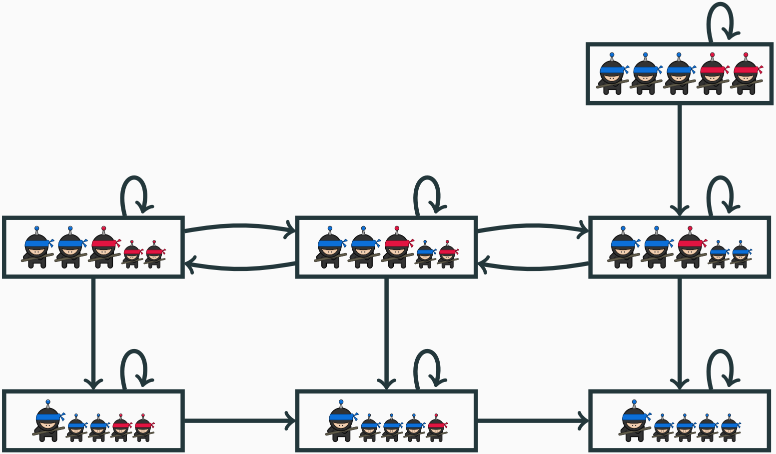
- *States:* finite set Q
- *Opinions:* $O : Q \to \{0, 1\}$
- *Initial states:* $I \subseteq Q$
- *Transitions:* $T \subseteq Q^2 \times Q^2$
- *Configurations:* $Q \to \mathbb{N}$
- *Initial configurations:* $I \to \mathbb{N}$



$(2 \quad 5 \quad 0 \quad 0)$

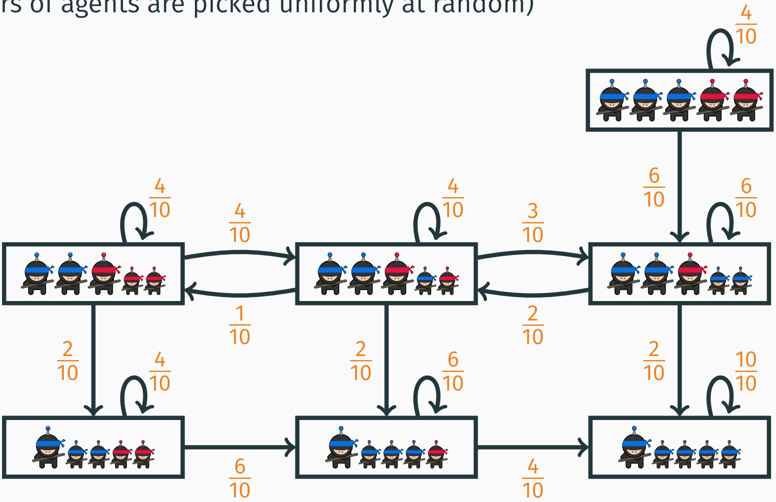
Population protocols: runs

Reachability graph for $(3, 2, 0, 0)$**:**



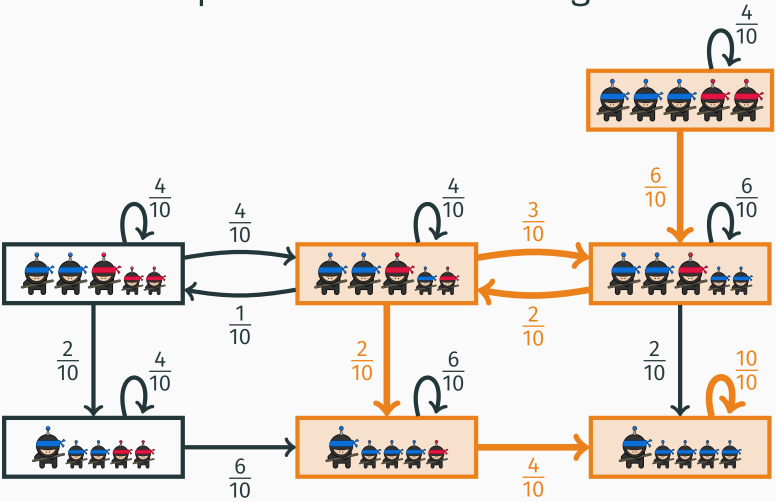
Underlying Markov chain:

(pairs of agents are picked uniformly at random)



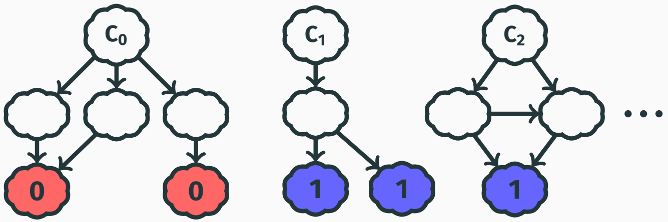
Population protocols: runs

Run : infinite path from initial configuration

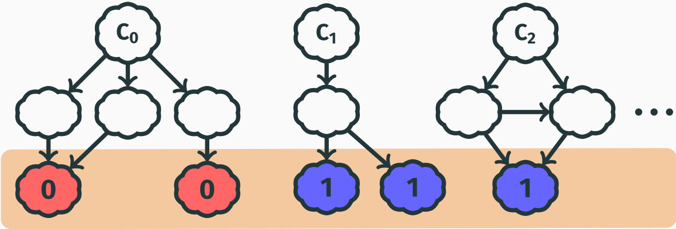


Population protocols: computing predicates

Protocol computes φ: InitC $\to \{0,1\}$:
for every $C \in$ InitC, the runs starting at C reach **stable consensus** $\varphi(C)$ with probability 1.



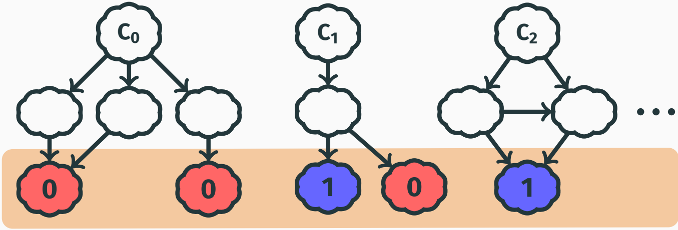
Protocol computes φ: InitC $\to \{0, 1\}$:
for every $C \in$ InitC, the runs starting at C
reach **stable consensus** $\varphi(C)$ with probability 1.



Protocol computes $\varphi(C_0) = 0, \varphi(C_1) = 1, \varphi(C_2) = 1, \ldots$

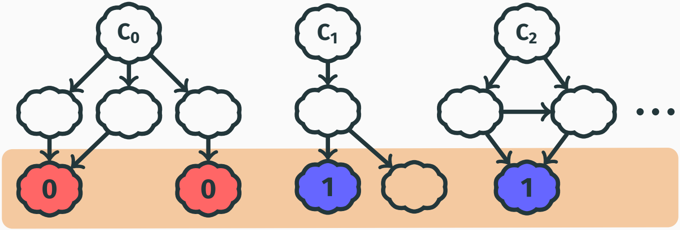
Population protocols: computing predicates

Protocol computes φ: InitC $\to \{0, 1\}$:
for every $C \in$ InitC, the runs starting at C
reach **stable consensus** $\varphi(C)$ with probability 1.



Protocol ill defined for C_1

Protocol computes φ: InitC $\to \{0, 1\}$:
for every $C \in$ InitC, the runs starting at C
reach **stable consensus** $\varphi(C)$ with probability 1.



Protocol ill defined for C_1 (Sensei I's problem)

A protocol is well specified if it computes some predicate

A protocol is well specified if it computes some predicate

A protocol for a predicate φ is correct if it computes φ (in particular, correct protocols are well specified)

Sensei III's questions

What predicates can we compute?

How fast can we compute them?

How succinctly can we compute them?

How can I check correctness?

How can I check efficiency?

To conclude ...

Expressive power

Angluin, Aspnes, Eisenstat Dist. Comp.'07

Population protocols compute precisely the predicates definable in Presburger arithmetic, *i.e.* $\mathrm{FO}(\mathbb{N}, +, <)$

Expressive power

Angluin, Aspnes, Eisenstat Dist. Comp.'07

Population protocols compute precisely the predicates definable in Presburger arithmetic, *i.e.* $\mathrm{FO}(\mathbb{N}, +, <)$

Proof: PPs compute all Presburger predicates

Since Presburger arithmetic has quantifier elimination, it suffices to:

Expressive power

Angluin, Aspnes, Eisenstat Dist. Comp.'07

Population protocols compute precisely the predicates definable in Presburger arithmetic, *i.e.* $\mathrm{FO}(\mathbb{N}, +, <)$

Proof: PPs compute all Presburger predicates

Since Presburger arithmetic has quantifier elimination, it suffices to:

- Exhibit PPs for threshold and modulo predicates

$$a_1x_1 + \cdots + a_nc_n \leq b \quad a_1x_1 + \cdots + a_nc_n \equiv b \pmod{c}$$

Expressive power

Angluin, Aspnes, Eisenstat Dist. Comp.'07

Population protocols compute precisely the predicates definable in Presburger arithmetic, *i.e.* $\text{FO}(\mathbb{N}, +, <)$

Proof: PPs compute all Presburger predicates

Since Presburger arithmetic has quantifier elimination, it suffices to:

- Exhibit PPs for threshold and modulo predicates

$$a_1 x_1 + \cdots + a_n c_n \leq b \quad a_1 x_1 + \cdots + a_n c_n \equiv b \pmod{c}$$

- Prove that computable predicates are closed under negation and conjunction

Expressive power

Angluin, Aspnes, Eisenstat Dist. Comp.'07

Population protocols compute precisely the predicates definable in Presburger arithmetic, *i.e.* $\mathrm{FO}(\mathbb{N}, +, <)$

Proof: PPs only compute Presburger predicates

- Much harder!

Expressive power

Angluin, Aspnes, Eisenstat Dist. Comp.'07

Population protocols compute precisely the predicates definable in Presburger arithmetic, *i.e.* $\mathrm{FO}(\mathbb{N}, +, <)$

Proof: PPs only compute Presburger predicates

- Much harder!
- Dist. Comp.'07 proof is "non-constructive"

Expressive power

Angluin, Aspnes, Eisenstat Dist. Comp.'07

Population protocols compute precisely the predicates definable in Presburger arithmetic, *i.e.* FO($\mathbb{N}$, $+$, $<$)

Proof: PPs only compute Presburger predicates

- Much harder!
- Dist. Comp.'07 proof is "non-constructive"
- "Constructive" proof by E., Ganty, Leroux, Majumdar Acta Inf.'17

Expressive power

Angluin, Aspnes, Eisenstat Dist. Comp.'07

Population protocols compute precisely the predicates definable in Presburger arithmetic, *i.e.* $\mathrm{FO}(\mathbb{N}, +, <)$

Other variants considered:

- Approximate protocols — *e.g.* Angluin, Aspnes, Eisenstat DISC'07
- Protocols with leaders — Angluin, Aspnes, Eisenstat Dist. Comput.'08
- Protocols with failures — Delporte-Gallet *et al.* DCOSS'06
- Trustful protocols — Bournez, Lefevre, Rabie DISC'13
- Mediated protocols, etc. — Michail, Chatzigiannakis, Spirakis TCS'11

Sensei III's questions

What predicates can we compute?

How fast can we compute them?

How succinctly can we compute them?

How can I check correctness?

How can I check efficiency?

To conclude ...

Efficiency

Efficiency measured by the expected number of interactions until stable consensus: $\mathit{Inter}(n)$

Efficiency

Efficiency measured by the expected number of interactions until stable consensus: $Inter(n)$

Depends on the population size n

Efficiency

Efficiency measured by the expected number of interactions until stable consensus: $Inter(n)$

Depends on the population size n

In a natural model: expected parallel time to consensus satisfies

$$Time(n) = Inter(n)/n$$

Efficiency

Angluin, Aspnes *et al.* **PODC'04**

Every Presburger predicate is computable in $\mathcal{O}(n \log n)$ time.

Efficiency

Angluin, Aspnes *et al.* **PODC'04**

Every Presburger predicate is computable in $\mathcal{O}(n \log n)$ time.

Angluin, Aspnes, Eisenstat Dist.Comp.'08

Every Presburger predicate is computable by protocols with leaders in $\log^{\mathcal{O}(1)} n$ time.

Efficiency

Angluin, Aspnes *et al.* **PODC'04**

Every Presburger predicate is computable in $\mathcal{O}(n \log n)$ time.

Angluin, Aspnes, Eisenstat Dist.Comp.'08

Every Presburger predicate is computable by protocols with leaders in $\log^{\mathcal{O}(1)} n$ time.

Alistarh, Aspnes, Eisenstat, Gelashvili, Rivest SODA'17

Every protocol computing majority takes $\Omega(n)$ time.
Majority is computable in $\log^{\mathcal{O}(1)} n$ time by leaderless protocols with $\mathcal{O}(\log^2 n)$ states.

Efficiency

Angluin, Aspnes *et al.* **PODC'04**

Every Presburger predicate is computable in $\mathcal{O}(n \log n)$ time.

Angluin, Aspnes, Eisenstat Dist.Comp.'08

Every Presburger predicate is computable by protocols with leaders in $\log^{\mathcal{O}(1)} n$ time.

Alistarh, Aspnes, Eisenstat, Gelashvili, Rivest SODA'17

Every protocol computing majority takes $\Omega(n)$ time.
Majority is computable in $\log^{\mathcal{O}(1)} n$ time by leaderless protocols with $\mathcal{O}(\log^2 n)$ states.

Open: Which predicates have $\log^{\mathcal{O}(1)} n$ leaderless protocols ?

Sensei III's questions

What predicates can we compute?

How fast can we compute them?

How succinctly can we compute them?

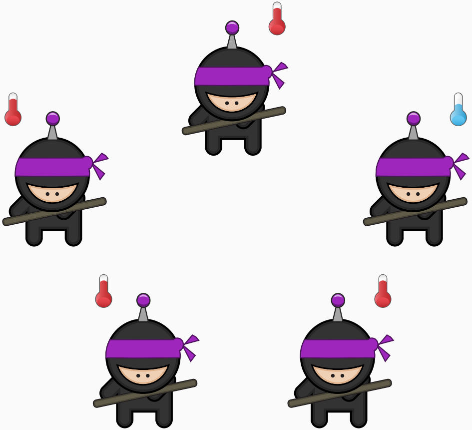
How can I check correctness?

How can I check efficiency?

To conclude ...

Succinctness–An Example

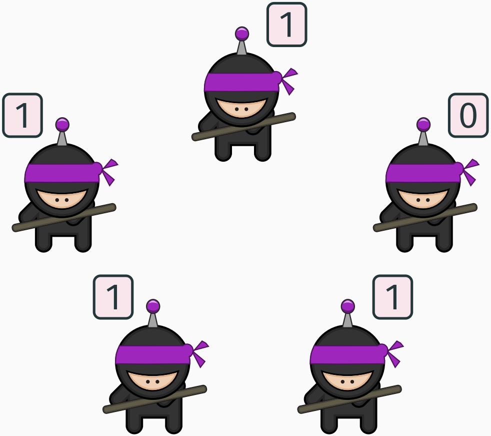
Protocol for: Are there at least 4 sick ninjas?



Succinctness–An Example

Protocol for: Are there at least 4 sick ninjas?

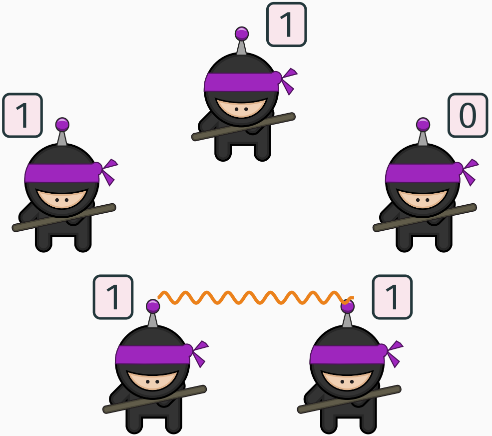
- Each ninja is in a state of $\{0, 1, 2, 3, 4\}$
- Initially, sick ninjas in state 1, healthy ninjas in state 0
- $(m, n) \mapsto (m + n, 0)$ if $m + n < 4$
- $(m, n) \mapsto (4, 4)$ if $m + n \geq 4$



Succinctness–An Example

Protocol for: Are there at least 4 sick ninjas?

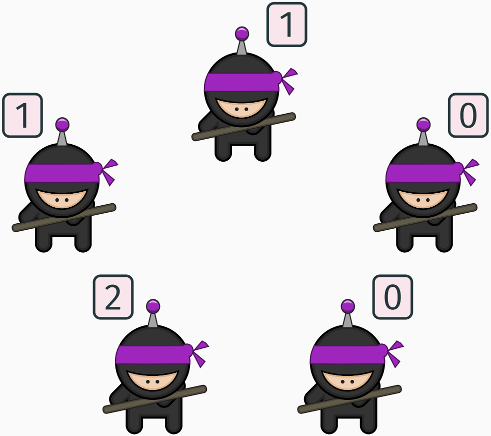
- Each ninja is in a state of $\{0, 1, 2, 3, 4\}$
- Initially, sick ninjas in state 1, healthy ninjas in state 0
- $(m, n) \mapsto (m + n, 0)$
  if $m + n < 4$
- $(m, n) \mapsto (4, 4)$
  if $m + n \geq 4$



Succinctness–An Example

Protocol for: Are there at least 4 sick ninjas?

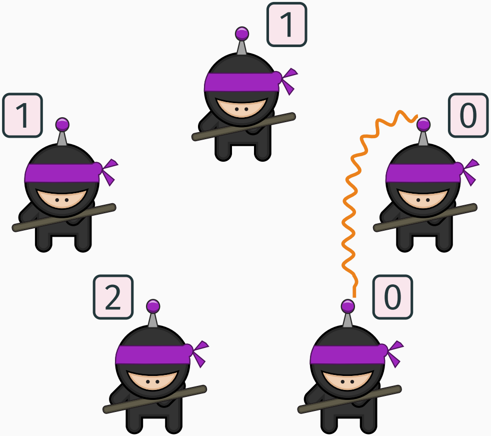
- Each ninja is in a state of $\{0, 1, 2, 3, 4\}$
- Initially, sick ninjas in state 1, healthy ninjas in state 0
- $(m, n) \mapsto (m + n, 0)$ if $m + n < 4$
- $(m, n) \mapsto (4, 4)$ if $m + n \geq 4$



Succinctness–An Example

Protocol for: Are there at least 4 sick ninjas?

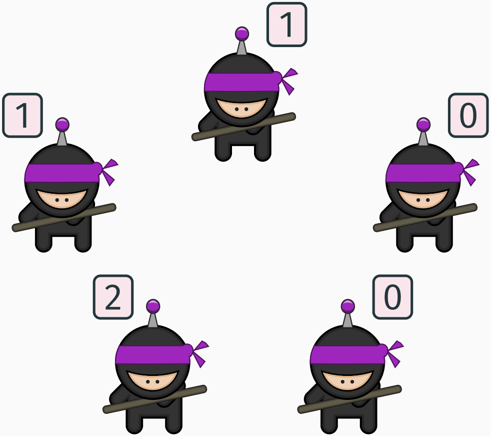
- Each ninja is in a state of $\{0, 1, 2, 3, 4\}$
- Initially, sick ninjas in state 1, healthy ninjas in state 0
- $(m, n) \mapsto (m + n, 0)$
  if $m + n < 4$
- $(m, n) \mapsto (4, 4)$
  if $m + n \geq 4$



Succinctness–An Example

Protocol for: Are there at least 4 sick ninjas?

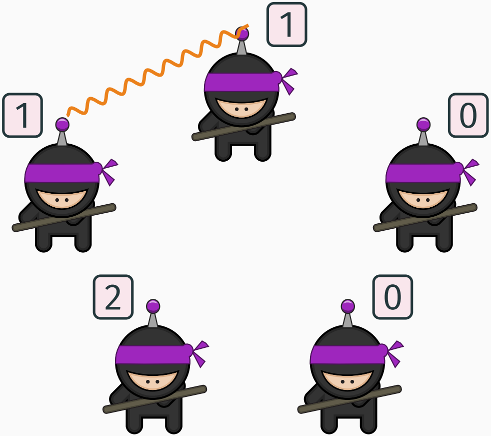
- Each ninja is in a state of $\{0, 1, 2, 3, 4\}$
- Initially, sick ninjas in state 1, healthy ninjas in state 0
- $(m, n) \mapsto (m + n, 0)$
  if $m + n < 4$
- $(m, n) \mapsto (4, 4)$
  if $m + n \geq 4$



Succinctness–An Example

Protocol for: Are there at least 4 sick ninjas?

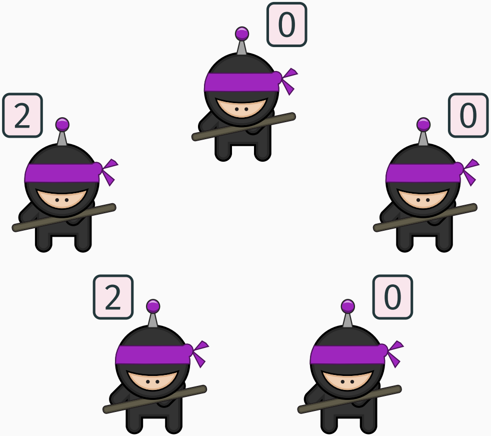
- Each ninja is in a state of $\{0, 1, 2, 3, 4\}$
- Initially, sick ninjas in state 1, healthy ninjas in state 0
- $(m, n) \mapsto (m + n, 0)$ if $m + n < 4$
- $(m, n) \mapsto (4, 4)$ if $m + n \geq 4$



Succinctness–An Example

Protocol for: Are there at least 4 sick ninjas?

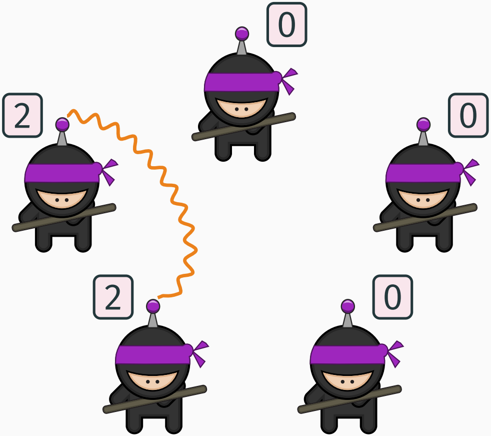
- Each ninja is in a state of $\{0, 1, 2, 3, 4\}$
- Initially, sick ninjas in state 1, healthy ninjas in state 0
- $(m, n) \mapsto (m + n, 0)$ if $m + n < 4$
- $(m, n) \mapsto (4, 4)$ if $m + n \geq 4$



Succinctness–An Example

Protocol for: Are there at least 4 sick ninjas?

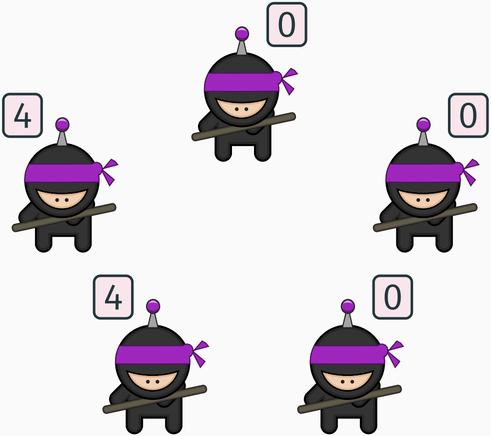
- Each ninja is in a state of $\{0, 1, 2, 3, 4\}$
- Initially, sick ninjas in state 1, healthy ninjas in state 0
- $(m, n) \mapsto (m + n, 0)$
  if $m + n < 4$
- $(m, n) \mapsto (4, 4)$
  if $m + n \geq 4$



Succinctness–An Example

Protocol for: Are there at least 4 sick ninjas?

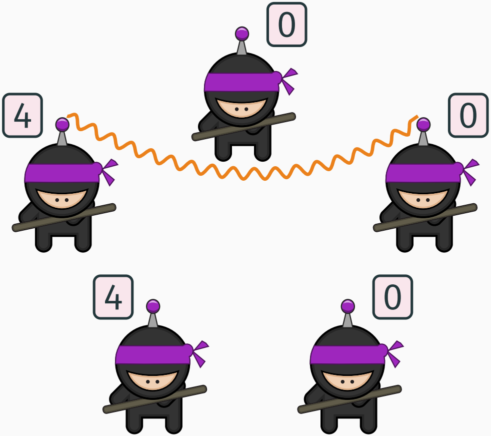
- Each ninja is in a state of $\{0, 1, 2, 3, 4\}$
- Initially, sick ninjas in state 1, healthy ninjas in state 0
- $(m, n) \mapsto (m + n, 0)$
  if $m + n < 4$
- $(m, n) \mapsto (4, 4)$
  if $m + n \geq 4$



Succinctness–An Example

Protocol for: Are there at least 4 sick ninjas?

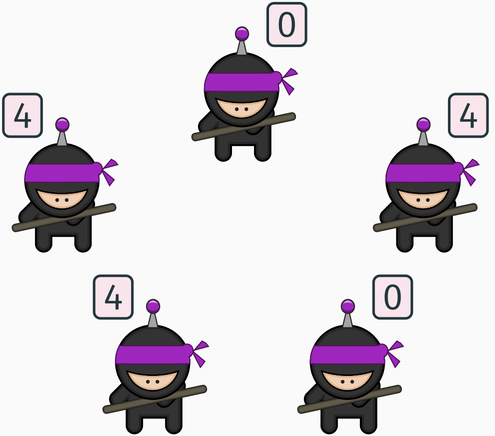
- Each ninja is in a state of $\{0, 1, 2, 3, 4\}$
- Initially, sick ninjas in state 1, healthy ninjas in state 0
- $(m, n) \mapsto (m + n, 0)$ if $m + n < 4$
- $(m, n) \mapsto (4, 4)$ if $m + n \geq 4$



Succinctness–An Example

Protocol for: Are there at least 4 sick ninjas?

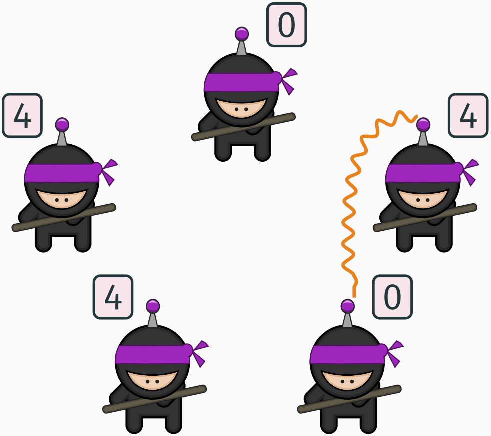
- Each ninja is in a state of $\{0, 1, 2, 3, 4\}$
- Initially, sick ninjas in state 1, healthy ninjas in state 0
- $(m, n) \mapsto (m + n, 0)$
  if $m + n < 4$
- $(m, n) \mapsto (4, 4)$
  if $m + n \geq 4$



Succinctness–An Example

Protocol for: Are there at least 4 sick ninjas?

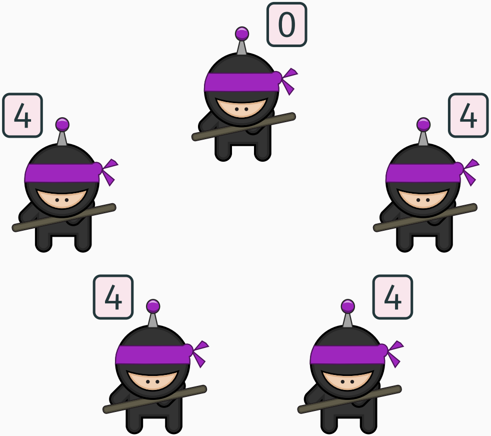
- Each ninja is in a state of $\{0, 1, 2, 3, 4\}$
- Initially, sick ninjas in state 1, healthy ninjas in state 0
- $(m, n) \mapsto (m + n, 0)$
  if $m + n < 4$
- $(m, n) \mapsto (4, 4)$
  if $m + n \geq 4$



Succinctness–An Example

Protocol for: Are there at least 4 sick ninjas?

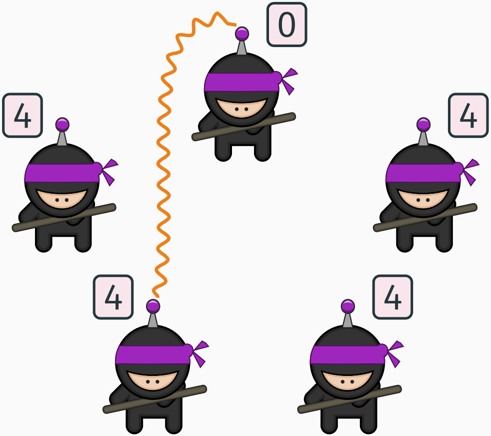
- Each ninja is in a state of $\{0, 1, 2, 3, 4\}$
- Initially, sick ninjas in state 1, healthy ninjas in state 0
- $(m, n) \mapsto (m + n, 0)$
  if $m + n < 4$
- $(m, n) \mapsto (4, 4)$
  if $m + n \geq 4$



Succinctness–An Example

Protocol for: Are there at least 4 sick ninjas?

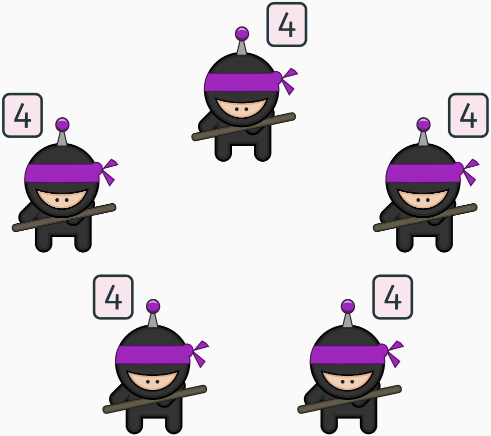
- Each ninja is in a state of $\{0, 1, 2, 3, 4\}$
- Initially, sick ninjas in state 1, healthy ninjas in state 0
- $(m, n) \mapsto (m + n, 0)$
  if $m + n < 4$
- $(m, n) \mapsto (4, 4)$
  if $m + n \geq 4$



Succinctness–An Example

Protocol for: Are there at least 4 sick ninjas?

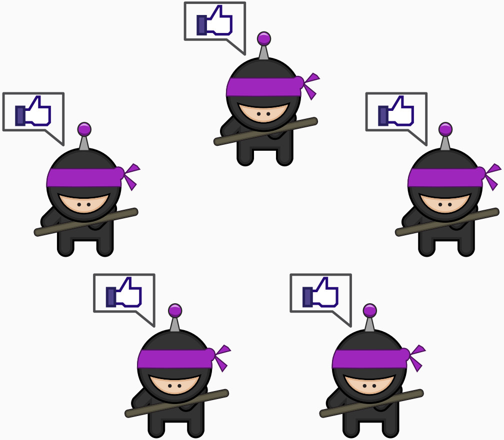
- Each ninja is in a state of $\{0, 1, 2, 3, 4\}$
- Initially, sick ninjas in state 1, healthy ninjas in state 0
- $(m, n) \mapsto (m + n, 0)$
  if $m + n < 4$
- $(m, n) \mapsto (4, 4)$
  if $m + n \geq 4$



Succinctness–An Example

Protocol for: Are there at least 4 sick ninjas?

- Each ninja is in a state of $\{0, 1, 2, 3, 4\}$
- Initially, sick ninjas in state 1, healthy ninjas in state 0
- $(m, n) \mapsto (m + n, 0)$ if $m + n < 4$
- $(m, n) \mapsto (4, 4)$ if $m + n \geq 4$



Sensei III's questions: Succinctness–An Example

Protocol for: Are there at least 2^ℓ sick ninjas?

- Each ninja is in a state of $\{0, 1, \ldots, 2^\ell - 1, 2^\ell\}$
- Initially, sick ninjas in state 1, healthy ninjas in state 0
- $(m, n) \mapsto (m + n, 0)$
  if $m + n < 2^\ell$
- $(m, n) \mapsto (2^\ell, 2^\ell)$
  if $m + n \geq 2^\ell$

Sensei III's questions: Succinctness–An Example

Protocol for: Are there at least 2^ℓ sick ninjas?

- Each ninja is in a state of $\{0, 1, \ldots, 2^\ell - 1, 2^\ell\}$
- Initially, sick ninjas in state 1, healthy ninjas in state 0
- $(m, n) \mapsto (m + n, 0)$
  if $m + n < 2^\ell$
- $(m, n) \mapsto (2^\ell, 2^\ell)$
  if $m + n \geq 2^\ell$

- Each ninja is in a state of $\{0, 2^0, \ldots, 2^{\ell-1}, 2^\ell\}$
- Initially, sick ninjas in state 2^0, healthy ninjas in state 0
- $(2^m, 2^m) \mapsto (2^{m+1}, 0)$
  if $m + 1 \leq \ell$
- $(2^\ell, n) \mapsto (2^\ell, 2^\ell)$

Sensei III's questions: Succinctness–An Example

Protocol for: Are there at least 2^ℓ sick ninjas?

- Each ninja is in a state of $\{0, 1, \ldots, 2^\ell - 1, 2^\ell\}$
- Initially, sick ninjas in state 1, healthy ninjas in state 0
- $(m, n) \mapsto (m + n, 0)$
  if $m + n < 2^\ell$
- $(m, n) \mapsto (2^\ell, 2^\ell)$
  if $m + n \geq 2^\ell$

- Each ninja is in a state of $\{0, 2^0, \ldots, 2^{\ell-1}, 2^\ell\}$
- Initially, sick ninjas in state 2^0, healthy ninjas in state 0
- $(2^m, 2^m) \mapsto (2^{m+1}, 0)$
  if $m + 1 \leq \ell$
- $(2^\ell, n) \mapsto (2^\ell, 2^\ell)$
- Can be generalized to non-powers of 2

Succinctness

Just gave a protocol for **X ≥ c** with $\mathcal{O}(\log c)$ states.

Succinctness

Just gave a protocol for **X ≥ c** with $\mathcal{O}(\log c)$ states.

Is $\mathcal{O}(\log \log c)$ possible?

Succinctness

Just gave a protocol for $\mathbf{X} \geq \mathbf{c}$ with $\mathcal{O}(\log c)$ states.

Is $\mathcal{O}(\log \log c)$ possible?

Not for every **c** ...

Blondin, E., Jaax STACS'18

There exist infinitely many **c** such that every protocol for $\mathbf{X} \geq \mathbf{c}$ has at least $(\log \mathbf{c})^{1/4}$ states

Succinctness

Just gave a protocol for **X ≥ c** with $\mathcal{O}(\log c)$ states.

Is $\mathcal{O}(\log \log c)$ possible?

Not for every **c** ...

Blondin, E., Jaax STACS'18

There exist infinitely many **c** such that every protocol for **X ≥ c** has at least $(\log \mathbf{c})^{1/4}$ states

...but for some **c**, if we allow leaders:

Blondin, E., Jaax STACS'18

For infinitely many **c** there is a protocol with two leaders and $\mathcal{O}(\log \log \mathbf{c})$ states that computes **X ≥ c**

Succinctness

Blondin, E., Jaax STACS'18

For infinitely many $\mathbf{c}$ there is a protocol with two leaders and $\mathcal{O}(\log\log \mathbf{c})$ states that computes $\mathbf{X} \geq \mathbf{c}$

Proof:

Succinctness

Blondin, E., Jaax STACS'18

For infinitely many **c** there is a protocol with two leaders and $\mathcal{O}(\log\log \mathbf{c})$ states that computes **X ≥ c**

Proof:

- **Mayr and Meyer '82**: For every n there is a commutative semigroup presentation and two elements s, t such that the shortest word α leading from s to t (i.e., $t = s\,\alpha$) has length $|\alpha| \geq 2^{2^n}$

Succinctness

Blondin, E., Jaax STACS'18

For infinitely many **c** there is a protocol with two leaders and $\mathcal{O}(\log\log \mathbf{c})$ states that computes $\mathbf{X} \geq \mathbf{c}$

Proof:

- **Mayr and Meyer '82**: For every n there is a commutative semigroup presentation and two elements s, t such that the shortest word α leading from s to t (i.e., $t = s\,\alpha$) has length $|\alpha| \geq 2^{2^n}$
- Construct a protocol that "simulates" derivations in the semigroup

Succinctness

$\mathcal{O}(\log \log c)$ without leaders?

Succinctness

$\mathcal{O}(\log \log c)$ without leaders? Open

Succinctness

$\mathcal{O}(\log\log c)$ without leaders? Open

And $\mathcal{O}(\log\log\log c)$ with leaders?

Succinctness

$\mathcal{O}(\log \log c)$ without leaders? Open

And $\mathcal{O}(\log \log \log c)$ with leaders? Open

Succinctness

$\mathcal{O}(\log\log c)$ without leaders? Open

And $\mathcal{O}(\log\log\log c)$ with leaders? Open

$\mathcal{O}(\log|\varphi|)$ states for all φ?

Succinctness

$\mathcal{O}(\log\log c)$ without leaders? Open

And $\mathcal{O}(\log\log\log c)$ with leaders? Open

$\mathcal{O}(\log|\varphi|)$ states for all φ? Open

Sensei III's questions

What predicates can we compute?

How fast can we compute them?

How succinctly can we compute them?

How can I check correctness?

How can I check efficiency?

To conclude ...

Checking correctness

Protocols can become complex, even for B ≥ R:

Fast and Exact Majority in Population Protocols

Dan Alistarh
Microsoft Research

Rati Gelashvili*
MIT

Milan Vojnović
Microsoft Research

$weight(x) = \begin{cases} |x| & \text{if } x \in StrongStates \text{ or } x \in WeakStates; \\ 1 & \text{if } x \in IntermediateStates. \end{cases}$

$sgn(x) = \begin{cases} 1 & \text{if } x \in \{+0, 1_d, \ldots, 1_1, 3, 5, \ldots, m\}; \\ -1 & \text{otherwise.} \end{cases}$

$value(x) = sgn(x) \cdot weight(x)$

```
/* Functions for rounding state interactions */
```

$\phi(x) = -1_1$ if $x = -1; 1_1$ if $x = 1; x$, otherwise

$R_\downarrow(k) = \phi(k$ if k odd integer, $k - 1$ if k even)

$R_\uparrow(k) = \phi(k$ if k odd integer, $k + 1$ if k even)

$Shift\text{-}to\text{-}Zero(x) = \begin{cases} -1_{j+1} & \text{if } x = -1_j \text{ for some index } j < d \\ 1_{j+1} & \text{if } x = 1_j \text{ for some index } j < d \\ x & \text{otherwise.} \end{cases}$

$Sign\text{-}to\text{-}Zero(x) = \begin{cases} +0 & \text{if } sgn(x) > 0 \\ -0 & \text{oherwise.} \end{cases}$

procedure update$\langle x, y \rangle$
  if $(weight(x) > 0$ *and* $weight(y) > 1)$ ***or*** $(weight(y) > 0$ *and* $weight(x) > 1)$ **then**
    $x' \leftarrow R_\downarrow\left(\frac{value(x)+value(y)}{2}\right)$ and $y' \leftarrow R_\uparrow\left(\frac{value(x)+value(y)}{2}\right)$
  else if $weight(x) \cdot weight(y) = 0$ *and* $value(x) + value(y) > 0$ **then**
    if $weight(x) \neq 0$ **then** $x' \leftarrow Shift\text{-}to\text{-}Zero(x)$ and $y' \leftarrow Sign\text{-}to\text{-}Zero(x)$
    else $y' \leftarrow Shift\text{-}to\text{-}Zero(y)$ and $x' \leftarrow Sign\text{-}to\text{-}Zero(y)$
  else if $(x \in \{-1_d, +1_d\}$ *and* $weight(y) = 1$ *and* $sgn(x) \neq sgn(y))$ ***or***
      $(y \in \{-1_d, +1_d\}$ *and* $weight(x) = 1$ *and* $sgn(y) \neq sgn(x))$ **then**
    $x' \leftarrow -0$ and $y' \leftarrow +0$
  else
    $x' \leftarrow Shift\text{-}to\text{-}Zero(x)$ and $y' \leftarrow Shift\text{-}to\text{-}Zero(y)$

Checking correctness

Protocols can become complex, even for B ≥ R:

Fast and Exact Majority in Population Protocols

Dan Alistarh
Microsoft Research

Rati Gelashvili*
MIT

Milan Vojnović
Microsoft Research

$$weight(x) = \begin{cases} |x| & \text{if } x \in StrongStates \text{ or } x \in WeakStates; \\ 1 & \text{if } x \in IntermediateStates. \end{cases}$$

$$sgn(x) = \begin{cases} 1 & \text{if } x \in \{+0, 1_d, \ldots, 1_1, 3, 5, \ldots, m\}; \\ -1 & \text{otherwise.} \end{cases}$$

$value(x) = sgn(x) \cdot weight(x)$

/* Functions for rounding state interactions */

$\phi(x) = -1_1$ if $x = -1$; 1_1 if $x = 1$; x, otherwise

$R_\downarrow(k) = \phi(k$ if k odd integer, $k-1$ if k even)

$R_\uparrow(k) = \phi(k$ if k odd integer, $k+1$ if k even)

$$Shift\text{-}to\text{-}Zero(x) = \begin{cases} -1_{j+1} & \text{if } x = -1_j \text{ for some index } j < d \\ 1_{j+1} & \text{if } x = 1_j \text{ for some index } j < d \\ x & \text{otherwise.} \end{cases}$$

$$Sign\text{-}to\text{-}Zero(x) = \begin{cases} +0 & \text{if } sgn(x) > 0 \\ -0 & \text{oherwise.} \end{cases}$$

procedure update$\langle x, y \rangle$
  if $(weight(x) > 0$ *and* $weight(y) > 1)$ ***or*** $(weight(y) > 0$ *and* $weight(x) > 1)$ **then**
    $x' \leftarrow R_\downarrow\left(\frac{value(x)+value(y)}{2}\right)$ and $y' \leftarrow R_\uparrow\left(\frac{value(x)+value(y)}{2}\right)$
  else if $weight(x) \cdot weight(y) = 0$ *and* $value(x) + value(y) > 0$ **then**
    if $weight(x) \neq 0$ **then** $x' \leftarrow Shift\text{-}to\text{-}Zero(x)$ and $y' \leftarrow Sign\text{-}to\text{-}Zero(x)$
    else $y' \leftarrow Shift\text{-}to\text{-}Zero(y)$ and $x' \leftarrow Sign\text{-}to\text{-}Zero(y)$
  else if $(x \in \{-1_d, +1_d\}$ *and* $weight(y) = 1$ *and* $sgn(x) \neq sgn(y))$ ***or***
      $(y \in \{-1_d, +1_d\}$ *and* $weight(x) = 1$ *and* $sgn(y) \neq sgn(x))$ **then**
    $x' \leftarrow -0$ and $y' \leftarrow +0$
  else
    $x' \leftarrow Shift\text{-}to\text{-}Zero(x)$ and $y' \leftarrow Shift\text{-}to\text{-}Zero(y)$

How can we verify correctness automatically?

Checking correctness—Early days

Model checkers:

- `PAT`: model checker with global fairness

  (Sun, Liu, Song Dong and Pang CAV'09)

- `bp-ver`: graph exploration

  (Chatzigiannakis, Michail and Spirakis SSS'10)

- Conversion to counter machines + `PRISM/Spin`

  (Clément, Delporte-Gallet, Fauconnier and Sighireanu ICDCS'11)

Checking correctness—Early days

Model checkers:

- `PAT`: model checker with global fairness

  (Sun, Liu, Song Dong and Pang CAV'09)

- `bp-ver`: graph exploration

  (Chatzigiannakis, Michail and Spirakis SSS'10)

- Conversion to counter machines + `PRISM`/`Spin`

  (Clément, Delporte-Gallet, Fauconnier and Sighireanu ICDCS'11)

Only for populations of fixed size!

Checking correctness—Early days

Theorem provers:

- Verification with the interactive theorem prover **Coq**

(Deng and Monin TASE'09)

Checking correctness—Early days

Theorem provers:

- Verification with the interactive theorem prover **Coq**

(Deng and Monin TASE'09)

Not automatic!

Checking correctness—Early days

Theorem provers:

- Verification with the interactive theorem prover **Coq**

(Deng and Monin TASE'09)

Challenge: verifying automatically
<u>all</u> sizes

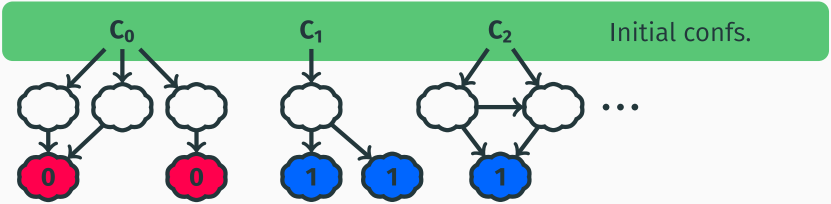
Checking correctness—Decidability

E., Ganty, Leroux, Majumdar Acta Inf.'17

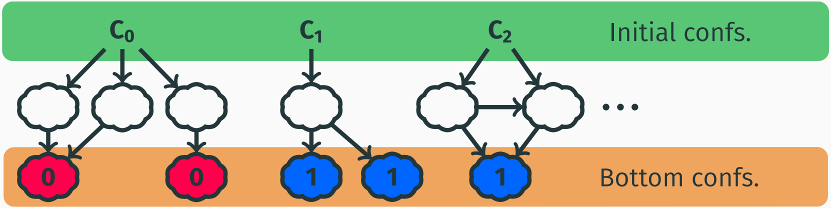
It is decidable if a population protocol is well specified (i.e., if it computes some predicate).

Checking correctness—Decidability

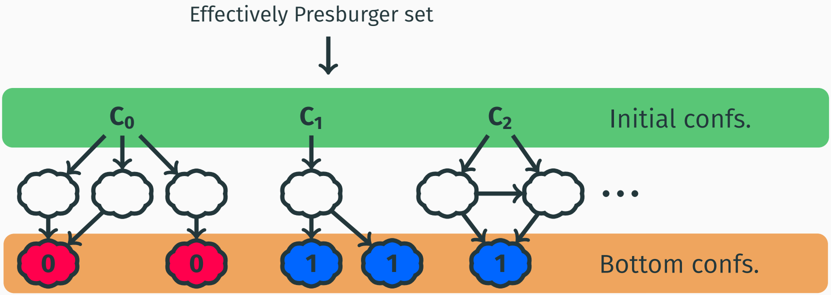
Acta Inf.'17



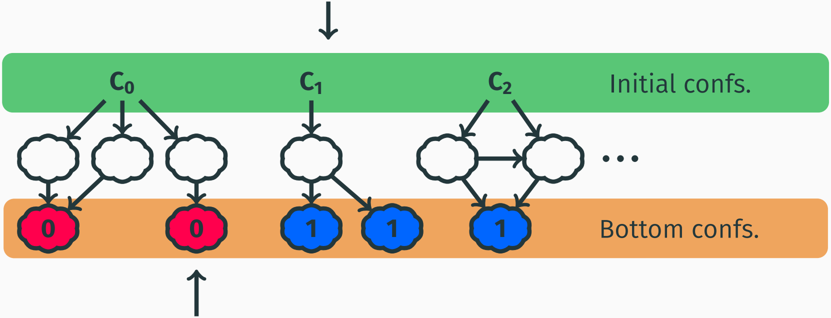
Checking correctness—Decidability



Checking correctness—Decidability



Effectively Presburger set

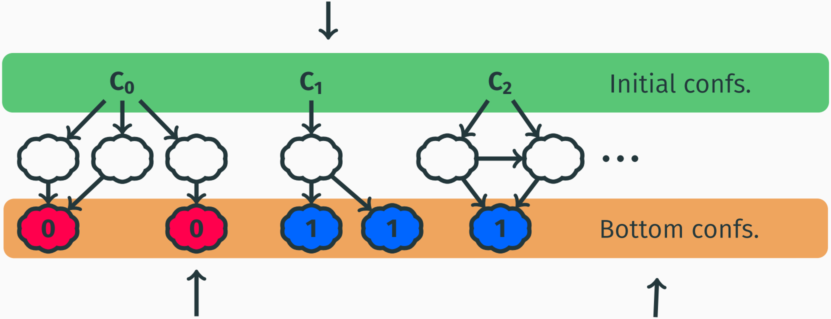


Eilenberg and Schützenberger '69:
Semilinear set
→ Presburger

Checking correctness—Decidability

Acta Inf.'17

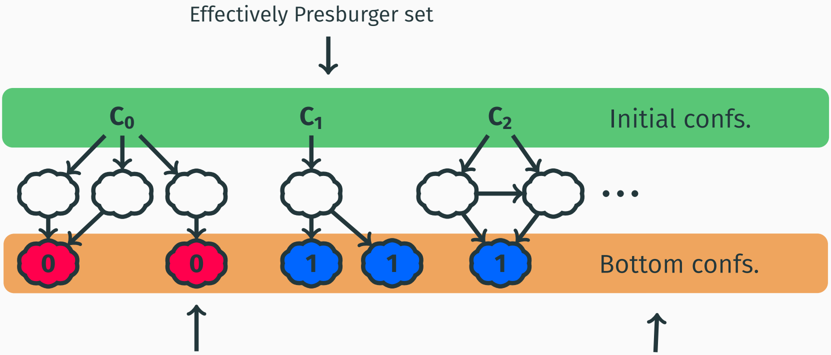
Effectively Presburger set



Eilenberg and Schützenberger '69:
Semilinear set
$\rightarrow$ Presburger

Leroux '11:
Effectively semilinear
$\rightarrow$ effectively Presburger

Checking correctness—Decidability



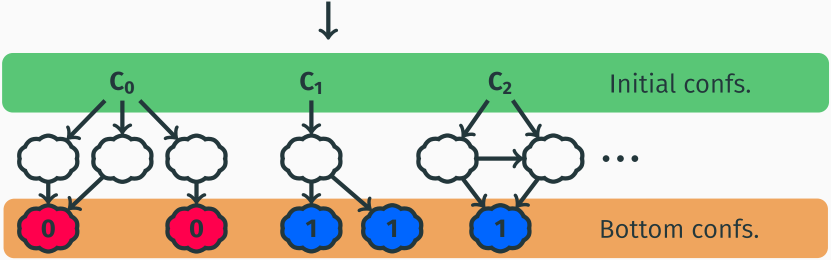
Eilenberg and Schützenberger '69:
Semilinear set
→ Presburger

Leroux '11:
Effectively semilinear
→ effectively Presburger

Reduction to the VAS reachability problem between Presburger sets

Checking correctness—Decidability

Effectively Presburger set



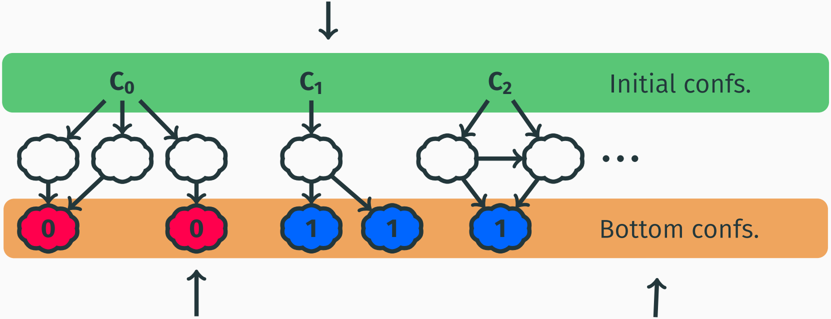
Eilenberg and Schützenberger '69:
Semilinear set
$\rightarrow$ Presburger

Leroux '11:
Effectively semilinear
$\rightarrow$ effectively Presburger

Reduction to the VAS reachability problem between Presburger sets
$\Rightarrow$ Reduction to the VAS reachability problem (VAS engineering)

Checking correctness—Decidability

Effectively Presburger set



Eilenberg and Schützenberger '69:
Semilinear set
$\to$ Presburger

Leroux '11:
Effectively semilinear
$\to$ effectively Presburger

Reduction to the VAS reachability problem between Presburger sets
$\Rightarrow$ Reduction to the VAS reachability problem (VAS engineering)
$\Rightarrow$ Decidable (**Mayr '81, Kosaraju '83**).

Checking correctness—Decidability

E., Ganty, Leroux, Majumdar   Acta Inf.'17

It is decidable if a population protocol computes a given predicate (Presburger formula).

Checking correctness—Decidability

E., Ganty, Leroux, Majumdar Acta Inf.'17

It is decidable if a population protocol computes a given predicate (Presburger formula).

There is an algorithm that returns the predicate computed by a well-specified protocol.

Checking correctness—Decidability

E., Ganty, Leroux, Majumdar Acta Inf.'17

VAS reachability is reducible to the well-specification problem for population protocols.

Checking correctness—Decidability

Acta Inf.’17

E., Ganty, Leroux, Majumdar   Acta Inf.’17

VAS reachability is reducible to the well-specification problem for population protocols.

Czerwinski, Lasota, Lazic, Leroux, Mazowiecki   arXiv’18

VAS reachability is non-elementary.

Checking correctness—Decidability

E., Ganty, Leroux, Majumdar Acta Inf.'17

VAS reachability is reducible to the well-specification problem for population protocols.

Czerwinski, Lasota, Lazic, Leroux, Mazowiecki arXiv'18

VAS reachability is non-elementary.
$\Rightarrow$ Well specification is non-elementary.

A class $\mathcal{P}$ of protocols is complete if for every Presburger predicate φ some protocol in $\mathcal{P}$ computes φ

A class $\mathcal{P}$ of protocols is complete if for every Presburger predicate φ some protocol in $\mathcal{P}$ computes φ

Goal: Find a complete class of protocols verifiable in reasonable time

Checking correctness—Feasibility

Blondin, E., Jaax, Meyer , PODC'17

The class of strongly silent protocols is complete, and its verification problem is in DP.

Checking correctness—Feasibility

Intel Core i7-4810MQ CPU and 16 GB of RAM.

Protocol	Predicate	$\lvert Q\rvert$	$\lvert T\rvert$	Time[s]
Majority[1]	$x \geq y$	4	4	0.1
Approx. Majority[2]	Not well-specified	3	4	0.1
Broadcast[3]	$x_1 \vee \ldots \vee x_n$	2	1	0.1
Threshold[4]	$\Sigma_i \alpha_i x_i < c$	76	2148	2375.9
Remainder[5]	$\Sigma_i \alpha_i x_i \bmod 70 = 1$	72	2555	3176.5
Sick ninjas[6]	$x \geq 50$	51	1275	181.6
Sick ninjas[7]	$x \geq 325$	326	649	3470.8
Poly-log sick ninjas	$x \geq 8 \cdot 10^4$	66	244	12.79

[1] Draief et al., 2012 [2] Angluin et al., 2007 [3] Clément et al., 2011
[4][5] Angluin et al., 2006 [6] Chatzigiannakis et al., 2010 [7] Clément et al., 2011

Checking correctness—Feasibility

Blondin, E., Jaax, Meyer , PODC'17

The class of strongly silent protocols is complete, and its verification problem is in DP.

Mission accomplished?

Checking correctness—Feasibility

Blondin, E., Jaax, Meyer , PODC'17

The class of strongly silent protocols is complete, and its verification problem is in DP.

Mission accomplished?

Not yet. For some predicates no strongly silent *succinct* protocols are known.

Checking correctness—Feasibility

A class $\mathcal{P}$ of protocols is complete and succinct if for every Presburger predicate φ some protocol in $\mathcal{P}$ with $\log(|\varphi|)$ states computes φ

A class $\mathcal{P}$ of protocols is complete and efficient if for every Presburger predicate φ some protocol in $\mathcal{P}$ computes φ in $\mathcal{O}(n^2 \log n)$ time.

Checking correctness—Feasibility

Are strongly silent protocols complete and succinct?

Checking correctness—Feasibility

Are strongly silent protocols complete and succinct?

Are strongly silent protocols complete and efficient?

Checking correctness—Feasibility

Are strongly silent protocols complete and succinct?

Are strongly silent protocols complete and efficient?

What is the lowest expected time for a complete class of protocols?

Checking correctness—Feasibility

Are strongly silent protocols complete and succinct?

Are strongly silent protocols complete and efficient?

What is the lowest expected time for a complete class of protocols?

...and for a complete and succinct class?

Checking correctness—Feasibility

Are strongly silent protocols complete and succinct?

Are strongly silent protocols complete and efficient?

What is the lowest expected time for a complete class of protocols?

...and for a complete and succinct class?

...and for a complete and efficient class?

Are strongly silent protocols complete and succinct? **Open**

Are strongly silent protocols complete and efficient? **Open**

What is the lowest expected time for a complete class of protocols? **Open**

...and for a complete and succinct class? **Open**

...and for a complete and efficient class? **Open**

Sensei III's questions

What predicates can we compute?

How fast can we compute them?

How succinctly can we compute them?

How can I check correctness?

How can I check efficiency?

To conclude ...

Our approach:

- Most protocols are naturally designed in stages
- Construct these stages automatically
- Derive upper bounds on $Inter(n)$ from stages structure

Checking expected termination time

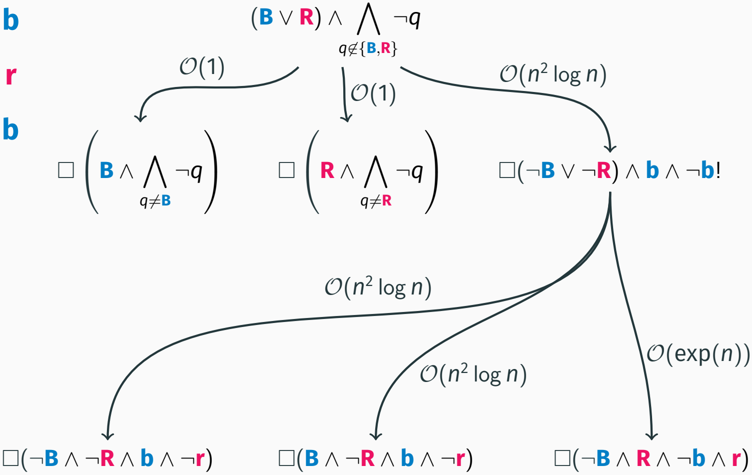
Blondin, E., Kucera CONCUR'18

$\mathbf{B}, \mathbf{R} \mapsto \mathbf{b}, \mathbf{b}$

$\mathbf{B}, \mathbf{r} \mapsto \mathbf{B}, \mathbf{b}$

$\mathbf{R}, \mathbf{b} \mapsto \mathbf{R}, \mathbf{r}$

$\mathbf{b}, \mathbf{r} \mapsto \mathbf{b}, \mathbf{b}$



Checking expected termination time

Blondin, E., Kucera CONCUR'18

- Prototype implemented in python™ + Microsoft Z3

Checking expected termination time

Blondin, E., Kucera CONCUR'18

- Prototype implemented in python™ + Microsoft Z3

- Can report: $\mathcal{O}(1), \mathcal{O}(n), \mathcal{O}(n \log n), \mathcal{O}(n^2), \mathcal{O}(\text{poly}(n))$ or $\mathcal{O}(\exp(n))$ time

Checking expected termination time

Blondin, E., Kucera CONCUR'18

- Prototype implemented in python™ + Microsoft Z3
- Can report: $\mathcal{O}(1), \mathcal{O}(n), \mathcal{O}(n \log n), \mathcal{O}(n^2), \mathcal{O}(\text{poly}(n))$ or $\mathcal{O}(\exp(n))$ time
- Decidability of checking $\mathit{Time}(n) \geq f(n)$?

  Open

Sensei III's questions

What predicates can we compute?

How fast can we compute them?

How succinctly can we compute them?

How can I check correctness?

How can I check efficiency?

To conclude ...

Peregrine: a tool for population protocols Blondin, E., Jaax CAV'18

`Peregrine`: Haskell + Microsoft Z3 + JavaScript

`peregrine.model.in.tum.de`

- Design of protocols
- Manual and automatic simulation
- Statistics of properties such as termination time
- Automatic verification of correctness
- More to come!

Conclusion

Population protocols are a great model to study fundamental questions of distributed computation:

- Power of anonymous computation
- Network-independent algorithms
- Role of leaders
- Emergent behaviour and its limits

Conclusion

…and of formal verification:

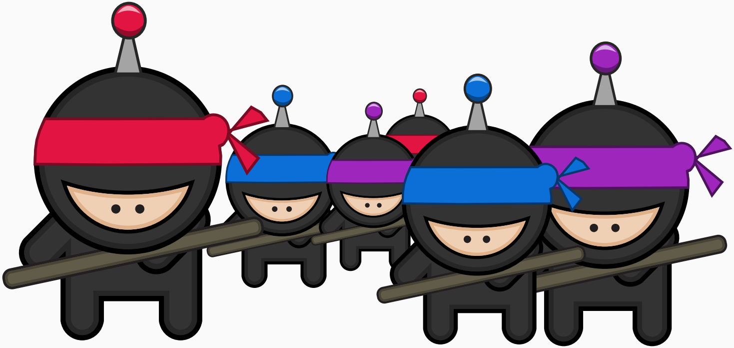
- Verification of stochastic parameterized systems (parameterization, liveness under fairness)
- Automatic synthesis of parameterized systems

Join the team!

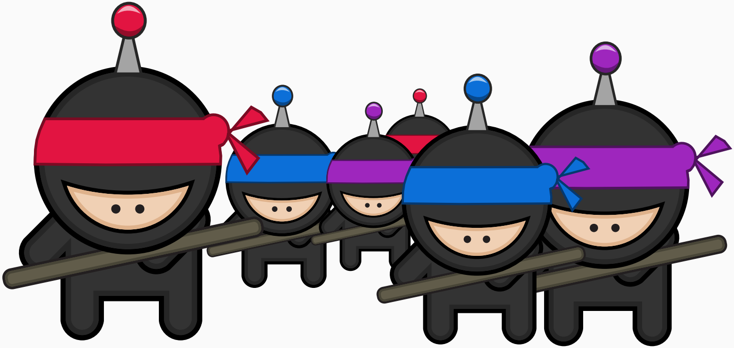
ERC Advanced Grant —

PaVeS: Parameterized Verification and Synthesis

- Goal: Develop proof and synthesis techniques for distributed algorithms working correctly for an arbitrary number of processes
- Start of the project: Sept. 1, 2018
- Start of employment for PhD students and post-docs: flexible, until about Sept. 1, 2019



THANK YOU!



THANK YOU!

Go!