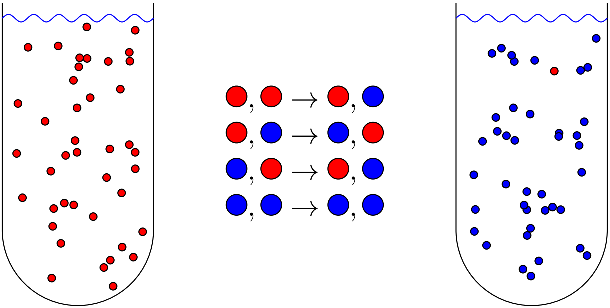
Clocked Population Protocols

James Aspnes

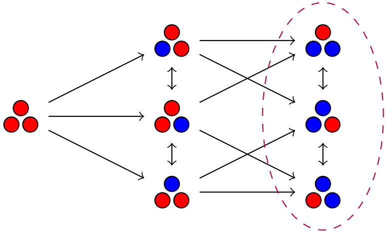
PODC 2017

Population protocols (Angluin et al., PODC 2004)



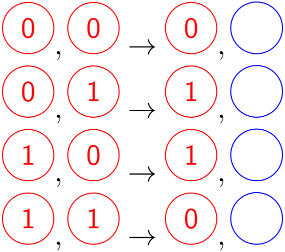
- **Interaction** updates state of both agents.
- Interactions happen one at a time.
- Who chooses which interaction happens next?
- The **adversary**, subject to a **fairness condition**.

Fairness



- If $C \to C'$, and C occurs ∞ often, so does C'.
- Equivalent: If C' is **enabled** ∞ often, C' occurs ∞ often.
- $\Rightarrow$ Any continuously reachable state is eventually reached.
- $\Rightarrow$ Any execution **converges** to some **terminal SCC**.
- Ideal case: unique terminal SCC with **stable output**.

Computation with population protocols



- Represent input as counts of agents in each state.
- Coalesce values along with leader election.
  - Gets parity, mod 3, etc.
  - Cancellation gets $<$, $=$.
- Run protocols in parallel for $f \wedge g$, $f \vee g$, etc.
- Result: Can **stably compute** all **semilinear** predicates, those definable in **first-order Presburger arithmetic**.

What population protocols can't do

- Anything that requires nested iteration:
  - Multiplication (by non-constants).
  - Division.
  - Many other operations.
  - Anything *not* definable in first-order Presburger arithmetic (Angluin et al., PODC 2006).
- Why not? Because we can't detect convergence.
- To avoid this result, we need to change the model.

Randomized pop. protocols (Angluin et al., DISC 2007)

- Assume **random** scheduling instead of adversarial scheduling.
- Still fair (with probability 1).
- But now we can predict how long operations take:
  - Use epidemics to propagate information.
  - Use **phase clock** to measure passage of time.
  - Use a leader agent to act as controller.
- Simulates **register machine** (whp) with polylog overhead.

$\Rightarrow$ Can compute any predicate in **RL**.

Absence detectors (Michail and Spirakis, JPDC 2015)

- Detect when no agents in a particular state exist.
- Implemented using **cover time oracle** that signals when an agent has encountered every other agent.
- Also simulates register machine.

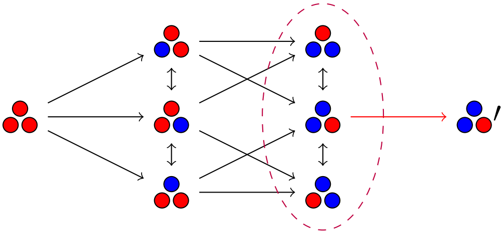
$\Rightarrow$ Can compute any predicate in **NL**.

Putting a clock in the model

- **Phase clock** signals when a register operation converges.
- **Cover time oracle** signals when absence detector converges.

Why not just put in an oracle that signals convergence?

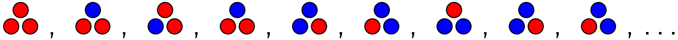
Clocked population protocol



In a **clocked population protocol**, each agent has a **tick** bit that signals convergence to a terminal SCC.

- **Clock transition** sets tick bit on one or more agents.
- Enabled in terminal SCC.
- Equivalently: Enabled in **limit configuration** of computation.

Limit configurations

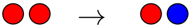


- If we wait long enough, we reach terminal SCC.
- How about waiting forever?
- Terminal SCC configurations = limit configurations.
- But who chooses which limit configuration?
- The **adversary**, subject to a **fairness condition**.

Measuring time with transfinite ordinals

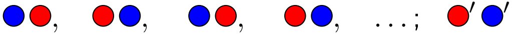
$$0, 1, 2, \ldots ; \omega, \omega + 1, \omega + 2, \ldots ; \omega \cdot 2, \omega \cdot 2 + 1, \ldots ; \omega^2, \omega^2 + 1, \ldots$$

Successor ordinals represent standard transitions.



$t \qquad\qquad t + 1$

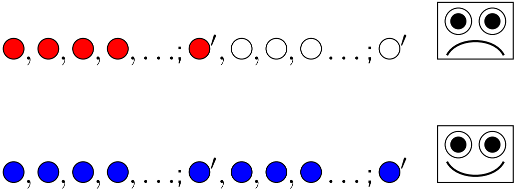
Limit ordinals represent clock transitions.



$\omega \qquad \omega + 1 \qquad \omega + 2 \qquad \omega + 3 \qquad\qquad \omega \cdot 2$

- Configuration at limit α can be any configuration that is **cofinal** in α, with ticks added to any subset of the agents.
- **Cofinal** in α = occurs at unbounded times up to α.
- Cofinal generalizes **infinitely often**.

Fairness over transfinite intervals



- Old definition: If C is enabled ∞ often, C occurs ∞ often.
- New definition: If C is enabled cofinally in α, C occurs cofinally in α (for all limit ordinals α).
- Equivalent for standard transitions.
- Enforces delivering ticks eventually for clock transitions.

Is the model realistic?

Computation over infinite intervals with magical convergence detection seems pretty implausible!

- Not really infinite:
  - Replace $\omega, \omega \cdot 2, \ldots$ with $D, 2D, \ldots$, where D is some finite bound.
- Not really detecting convergence:
  - Any physically realizable system should converge whp in a fixed amount of time D.

So the clock mechanism can just be a clock.

Application: Counter machines

$$
\begin{array}{rcl}
q_{\mathrm{inc}}, 0 & \to & q_{\mathrm{next}}, 1 \\
q_{\mathrm{inc}}, 1 & \to & q_{\mathrm{inc}}, 1
\end{array}
\qquad
\begin{array}{rcl}
q_{\mathrm{dec}}, 0 & \to & q_{\mathrm{dec}}, 0 \\
q_{\mathrm{dec}}, 1 & \to & q_{\mathrm{success}}, 0 \\
q'_{\mathrm{dec}}, 0 & \to & q_{\mathrm{failure}}, 0
\end{array}
$$

- Supports operations INCREMENT and DECREMENT-IF-NONZERO.
- Represent counter values in unary.
- Special leader agent represents finite-state controller.
- Use clock ticks to detect zero during decrement.
- Equivalent to $O(\log n)$-bit Turing machine (Minsky 1967).

$\Rightarrow$ Clocked population protocols can simulate **NL** in $< \omega^2$ time.

Application: Tracking tick levels

$$0, 0, \ldots; 0', 1, 0, \ldots; 0', 1, 0, \ldots\ldots\ldots; 1', 2, 0, \ldots$$

$$\begin{array}{ccc@{\qquad}ccc} 0 & \to & 0 & 0' & \to & 1 \\ 1 & \to & 0 & 1' & \to & 2 \\ 2 & \to & 0 & 2' & \to & 3 \end{array}$$

- $0'$ can only occur at multiples of ω.
- $1'$ can only occur at multiples of ω^2
- In general, t' occurs at multiples of ω^{t+1}.

$\Rightarrow$ Model doesn't need to signal "higher-order" ticks.

Configuration graphs

$$
\begin{aligned}
G_0 &= \text{standard transitions.} \\
G_{k+1} &= G_k \text{ plus clock transitions leaving terminal SCCs in } G_k. \\
G_\omega &= \lim_{k\to\infty} G_k.
\end{aligned}
$$

- G_k represents all computations in time $< \omega^{k+1}$.
- For fixed population size, only finitely many configurations, so $G_\omega = G_i$ for some k.
- Can construct $G_0, G_1, \ldots, G_i$ in polynomial time.

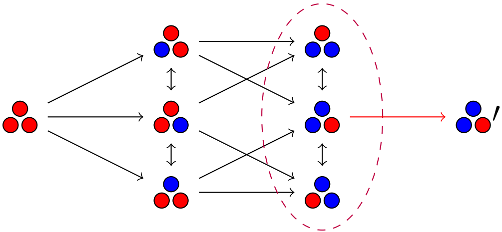
$\Rightarrow$ Clocked population protocols can be simulated in **P**.

$< \omega^k$ time in **NL**

- **L** can represent configurations of a clocked population protocol.
- **L** can compute standard transitions between configurations.
- **NL** can detect paths.
- **coNL** $=$ **NL** (Immerman-Szelepcsényi 1988) can detect no paths.
- Paths $+$ no paths $+$ $\mathbf{NL}^{\mathbf{NL}} = \mathbf{NL}$ means **NL** can recognize terminal SCCs.

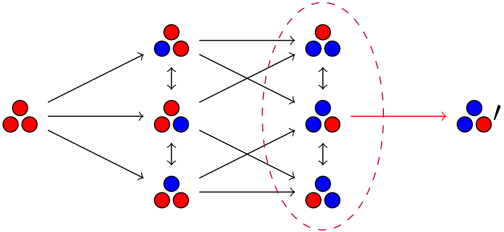
$\Rightarrow$ Can compute G_k for any fixed k in **NL**.
$\Rightarrow$ $< \omega^k$-time clocked population protocol in **NL**.
$\Rightarrow$ $< \omega^k$-time protocol simulated by $< \omega^2$-time protocol.

Summary



- **Clocked population protocols** add clocks for detecting convergence.
- Convergence as limits over transfinite intervals allows generalizing standard fairness.
- Allows composing and iterating population protocols.
- Can compute precisely **NL** in $< \omega^k$ time (and $< \omega^2$ is enough).
- Can be simulated by **P** even for unbounded time.

Open problem



Can an ω^{ω}-time clocked population protocol simulate **P**?

- No? Implies **NL** $\neq$ **P**.
- Yes? True if clocked pop. protocol can simulate **AL** $=$ **P**.