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 $\infty$ often, so does $C'$.
- Equivalent: If $C'$ is enabled $\infty$ often, $C'$ occurs $\infty$ 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 $f \land g, f \lor g$, etc.
- Result: Can stably compute all semilinear predicates, those definable in first-order Presburger arithmetic.
Any operation 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.

⇒ 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.

⇒ 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?
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, ...; ω, ω + 1, ω + 2, ...; ω · 2, ω · 2 + 1, ...; ω², ω² + 1, ...

**Successor ordinals** represent standard transitions.

\[
\text{configuration at limit } \alpha \text{ can be any configuration that is cofinal in } \alpha, \text{ with ticks added to any subset of the agents.}
\]

**Limit ordinals** represent clock transitions.

\[
\text{Configuration at limit } \alpha \text{ can be any configuration that is cofinal in } \alpha, \text{ with ticks added to any subset of the agents.}
\]

\[
\text{Cofinal in } \alpha = \text{ occurs at unbounded times up to } \alpha.
\]

\[
\text{Cofinal generalizes infinitely often.}
\]
Fairness over transfinite intervals

- Old definition: If $C$ is enabled $\infty$ often, $C$ occurs $\infty$ often.
- New definition: If $C$ is enabled cofinally in $\alpha$, $C$ occurs cofinally in $\alpha$ (for all limit ordinals $\alpha$).
- 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

\[
\begin{align*}
q_{\text{inc}}, 0 & \rightarrow q_{\text{next}}, 1 & q_{\text{dec}}, 0 & \rightarrow q_{\text{dec}}, 0 \\
q_{\text{inc}}, 1 & \rightarrow q_{\text{inc}}, 1 & q_{\text{dec}}, 1 & \rightarrow q_{\text{success}}, 0 \\
q'_{\text{dec}}, 0 & \rightarrow q_{\text{failure}}, 0
\end{align*}
\]

- 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 $\mathbf{NL}$ in $< \omega^2$ time.
Application: Tracking tick levels

\[0, 0, \ldots; 0', 1, 0, \ldots; 0', 1, 0, \ldots; 1', 2, 0, \ldots\]

\[
\begin{align*}
0 & \rightarrow 0 & 0' & \rightarrow 1 \\
1 & \rightarrow 0 & 1' & \rightarrow 2 \\
2 & \rightarrow 0 & 2' & \rightarrow 3
\end{align*}
\]

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

⇒ Model doesn't need to signal “higher-order” ticks.
Configuration graphs

\( G_0 = \) standard transitions.
\( G_{k+1} = G_k \) plus clock transitions leaving terminal SCCs in \( G_k \).
\( G_\omega = \lim_{k \to \infty} G_k \).

- \( G_k \) 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 + \( NL^{NL} = NL \) 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 $\text{NL}$ in $< \omega^k$ time (and $< \omega^2$ is enough).
- Can be simulated by $P$ even for unbounded time.
Can an $\omega^\omega$-time clocked population protocol simulate $P$?

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