Running-time Analysis of Broadcast Consensus Protocols

Philipp Czerner, Stefan Jaax
Fakultät für Informatik, TU München

March 23, 2021
Introduction
The Setting

▶ distributed computation
The Setting

- distributed computation
- population of *agents*
The Setting

- distributed computation
- population of agents
- agents are finite-state machines
The Setting

- distributed computation
- population of *agents*
- agents are finite-state machines
- random interactions
The Setting

- distributed computation
- population of agents
- agents are finite-state machines
- random interactions
The Setting

- distributed computation
- population of agents
- agents are finite-state machines
- random interactions
- want to reach consensus on whether the initial configuration satisfies a property
Population Protocols

- well-studied
Population Protocols

- well-studied
- finite set of states $Q$

compute exactly semi-linear (or Presburger) predicates
i.e. predicates expressible in first-order theory of integers with addition and the usual order

can compute majority: $x \geq y$

$\Omega(\frac{n^2}{\text{polylog}(n)})$ interactions to stabilise [Alistarh et al. 2017]

$O(n^{1+\epsilon})$ interactions to converge [Kosowski, Uznański 2018]
Population Protocols

- well-studied
- finite set of states $Q$
- pairwise transitions $T : Q^2 \rightarrow Q^2$
Population Protocols

- well-studied
- finite set of states $Q$
- pairwise transitions $T : Q^2 \rightarrow Q^2$
- compute exactly semi-linear (or Presburger) predicates

Moreover, we can compute the majority with

$$x \geq y$$

in $\Omega(n^2 / \text{polylog}(n))$ interactions to stabilise [Alistarh et al. 2017].

Convergence can be achieved with $O(n^{1+\varepsilon})$ interactions to converge [Kosowski, Uznański 2018].
Population Protocols

- well-studied
- finite set of states $Q$
- pairwise transitions $T : Q^2 \rightarrow Q^2$
- compute exactly semi-linear (or Presburger) predicates
  - i.e. predicates expressible in first-order theory of integers with addition and the usual order

\[ \Omega\left(\frac{n^2}{\text{polylog}(n)}\right) \text{ interactions to stabilise} \quad [\text{Alistarh et al. 2017}] \]
\[ O\left(n^{1+\varepsilon}\right) \text{ interactions to converge} \quad [\text{Kosowski, Uznański 2018}] \]
Population Protocols

- well-studied
- finite set of states $Q$
- pairwise transitions $T : Q^2 \rightarrow Q^2$
- compute exactly semi-linear (or Presburger) predicates
  - i.e. predicates expressible in first-order theory of integers with addition and the usual order
- can compute majority: $x \geq y$
Population Protocols

- well-studied
- finite set of states $Q$
- pairwise transitions $T : Q^2 \rightarrow Q^2$
- compute exactly semi-linear (or Presburger) predicates
  - i.e. predicates expressible in first-order theory of integers with addition and the usual order
- can compute majority: $x \geq y$
  - $\Omega(n^2 / \text{polylog}(n))$ interactions to stabilise [Alistarh et al. 2017]
Population Protocols

- well-studied
- finite set of states $Q$
- pairwise transitions $T : Q^2 \rightarrow Q^2$
- compute exactly semi-linear (or Presburger) predicates
  - i.e. predicates expressible in first-order theory of integers with addition and the usual order
- can compute majority: $x \geq y$
  - $\Omega(n^2 / \text{polylog}(n))$ interactions to stabilise [Alistarh et al. 2017]
  - $O(n^{1+\varepsilon})$ interactions to converge [Kosowski, Uznański 2018]
Broadcasts Consensus Protocols
What is a Broadcast Consensus Protocol (BCP)?

BCP = Population Protocol + Broadcasts
What is a Broadcast Consensus Protocol (BCP)?

\[ \text{BCP} \, = \, \text{Population Protocol} \, + \, \text{Broadcasts} \]

Why?
What is a Broadcast Consensus Protocol (BCP)?

BCP = Population Protocol + Broadcasts

Why?

1. Simple extension to population protocols for NL power
   [Blondin, Esparza, Jaax 2019]
What is a Broadcast Consensus Protocol (BCP)?

$\text{BCP} = \text{Population Protocol} + \text{Broadcasts}$

Why?

1. Simple extension to population protocols for NL power
   [Blondin, Esparza, Jaax 2019]
   - NL refers predicates decidable by log-space Turing machines, with input encoded as unary
What is a Broadcast Consensus Protocol (BCP)?

BCP = Population Protocol + Broadcasts

Why?

1. Simple extension to population protocols for NL power [Blondin, Esparza, Jaax 2019]
   ▶ NL refers predicates decidable by log-space Turing machines, with input encoded as unary
   ▶ much bigger than just semi-linear predicates

2. Study broadcasts in the computation-by-consensus paradigm

3. Model global influences in e.g. biological systems (cf. [Bertrand et al. 2017])

4. Construct faster and more powerful protocols
What is a Broadcast Consensus Protocol (BCP)?

BCP = Population Protocol + Broadcasts

Why?

1. Simple extension to population protocols for NL power [Blondin, Esparza, Jaax 2019]
   - NL refers predicates decidable by log-space Turing machines, with input encoded as unary
   - much bigger than just semi-linear predicates
   - other extensions in the literature: clocks, cover-time service / absence-detection
What is a Broadcast Consensus Protocol (BCP)?

BCP = Population Protocol + Broadcasts

Why?

1. Simple extension to population protocols for NL power
   [Blondin, Esparza, Jaax 2019]
   - NL refers predicates decidable by log-space Turing machines,
     with input encoded as unary
   - much bigger than just semi-linear predicates
   - other extensions in the literature: clocks, cover-time service / absence-detection

2. Study broadcasts in the computation-by-consensus paradigm
What is a Broadcast Consensus Protocol (BCP)?

BCP = Population Protocol + Broadcasts

Why?

1. Simple extension to population protocols for NL power [Blondin, Esparza, Jaax 2019]
   ▶ NL refers predicates decidable by log-space Turing machines, with input encoded as unary
   ▶ much bigger than just semi-linear predicates
   ▶ other extensions in the literature: clocks, cover-time service / absence-detection

2. Study broadcasts in the computation-by-consensus paradigm

3. Model global influences in e.g. biological systems (cf. [Bertrand et al. 2017])
What is a Broadcast Consensus Protocol (BCP)?

BCP = Population Protocol + Broadcasts

Why?

1. Simple extension to population protocols for NL power
   [Blondin, Esparza, Jaax 2019]
   - NL refers predicates decidable by log-space Turing machines,
     with input encoded as unary
   - much bigger than just semi-linear predicates
   - other extensions in the literature: clocks, cover-time service / absence-detection

2. Study broadcasts in the computation-by-consensus paradigm

3. Model global influences in e.g. biological systems
   (cf. [Bertrand et al. 2017])

4. Construct faster and more powerful protocols
Results

Prior work:
- Blondin, Esparza and Jaax show that BCPs compute exactly NL
  - no bounds on running time
  - multiple stages of reduction $\rightarrow$ complicated protocols

\[^1\text{w.r.t. number of transitions}\]
Prior work:
- Blondin, Esparza and Jaax show that BCPs compute exactly NL
  - no bounds on running time
  - multiple stages of reduction $\rightarrow$ complicated protocols

Our results:
1. time-optimal\(^1\), simple protocols for semi-linear predicates
   - expected $O(n \log n)$ transitions

\(^1\)w.r.t. number of transitions
Results

Prior work:
- Blondin, Esparza and Jaax show that BCPs compute exactly NL
  - no bounds on running time
  - multiple stages of reduction → complicated protocols

Our results:
1. time-optimal\(^1\), simple protocols for semi-linear predicates
   - expected $\mathcal{O}(n \log n)$ transitions
2. poly-time BCPs are precisely ZLP
   - i.e. predicates decidable by zero-error, log-space, expected poly-time randomised Turing Machines

\(^1\)w.r.t. number of transitions
What is a Broadcast Consensus Protocol (BCP)?

BCP = Population Protocol + Broadcasts

Formally:

- finite set of states $Q$,
- transitions $B : Q \rightarrow Q \times Q^Q$
What is a Broadcast Consensus Protocol (BCP)?

BCP = Population Protocol + Broadcasts

Formally:

finite set of states $Q$, transitions $B : Q \rightarrow Q \times Q^Q$

- Pairwise interactions can be simulated
What is a Broadcast Consensus Protocol (BCP)?

BCP = Population Protocol + Broadcasts

Formally:

finite set of states \( Q \),
transitions \( B : Q \rightarrow Q \times Q^Q \)

- Pairwise interactions can be simulated
- Non-determinism can be simulated
Transitions

Run on population of agents $C \in \mathbb{N}^Q$ (multiset of states).
Transitions

Run on population of agents $C \in \mathbb{N}^Q$ (multiset of states).

Execute: transition $q \mapsto r, f$, with $q, r \in Q$, $f : Q \to Q$
Transitions

Run on population of agents $C \in \mathbb{N}^Q$ (multiset of states).

Execute: transition $q \mapsto r$, with $q, r \in Q$, $f : Q \rightarrow Q$
Transitions

Run on population of agents $C \in \mathbb{N}^Q$ (multiset of states).

Execute: transition $q \mapsto r, f$, with $q, r \in Q, f : Q \rightarrow Q$
Computation

initial states $I \subseteq Q$
output mapping $O : Q \rightarrow \{0, 1\}$
predicate $\varphi : \mathbb{N}^I \rightarrow \{0, 1\}$

How do we compute $\varphi$?
Pick agents at random until everyone has (and retains) the same output.
Computation

initial states $I \subseteq Q$
output mapping $O : Q \rightarrow \{0, 1\}$
predicate $\varphi : \mathbb{N}^I \rightarrow \{0, 1\}$

How do we compute $\varphi$?
initial states \( I \subseteq Q \)

output mapping \( O : Q \rightarrow \{0, 1\} \)

predicate \( \varphi : \mathbb{N}^l \rightarrow \{0, 1\} \)

How do we compute \( \varphi \)?

Pick agents at random until everyone has (and retains) the same output.
Example

Majority \( \varphi(x, y) \iff x \geq y \)

\((x, y) = (2, 3)\)

input
Example

**Majority** $\varphi(x, y) \Leftrightarrow x \geq y$

$(x, y) = (2, 3)$

```
input  multiset
```

- $x$
- $y$
Example

**Majority** \( \varphi(x, y) \iff x \geq y \)

\[(x, y) = (2, 3)\]

- **input**
  - \(x\)
  - \(y\)

- **multiset**
  - \(x\)
  - \(y\)

- **population**
  - \(x\)
  - \(y\)
Compute \( \varphi(x, y) \iff x \geq y \)
Compute $\varphi(x, y) \iff x \geq y$

$y \mapsto 0, \{x \mapsto x', y \mapsto y', 0 \mapsto 0'\}$
Compute $\varphi(x, y) \iff x \geq y$ $x' \mapsto 0, \{x' \mapsto x, y' \mapsto y, 0' \mapsto 0\}$
Compute $\varphi(x, y) \iff x \geq y$

$y \mapsto 0, \{x \mapsto x', y \mapsto y', 0 \mapsto 0'\}$
Compute $\varphi(x, y) \iff x \geq y$

$x' \mapsto 0, \{ x' \mapsto x, y' \mapsto y, 0' \mapsto 0 \}$

output 1

disabled

output 0

"active"
Compute $\varphi(x, y) \iff x \geq y$

$y \mapsto 0, \{x \mapsto x', y \mapsto y', 0 \mapsto 0'\}$

- Output 1
- Output 0
- "active"
- "disabled"
Semi-linear predicates
Semi-linear predicates

▶ Example generalises to all semi-linear predicates
Semi-linear predicates

- Example generalises to all semi-linear predicates
- Shared global state
Semi-linear predicates

▶ Example generalises to all semi-linear predicates
▶ Shared global state

Steps:
1. Decompose semi-linear predicate into boolean combination of modulo and threshold predicates
2. Protocol for modulo predicates
3. Protocol for threshold predicates
4. Boolean combinations (simple)
Modulo predicates

\[ a_1 x_1 + \ldots + a_l x_l \equiv b \pmod{k} \]

Global state is \( \{0, \ldots, k - 1\} \), additions modulo \( k \)
Threshold predicates

\[ a_1 x_1 + \ldots + a_l x_l \geq k \]

Global state is large enough counter, take care not to overflow.
Standard coupon-collector analysis for $O(n \log n)$ transitions
- Standard coupon-collector analysis for $O(n \log n)$ transitions
- Simple matching lower bound (all agents have to act at least once)
- Standard coupon-collector analysis for $O(n \log n)$ transitions
- Simple matching lower bound (all agents have to act at least once)

Thus we get time-optimal BCPs for semi-linear predicates.
Thank you for your attention!