Time-Space Trade-offs in Population Protocols

Alexander Nemetz-Fiedler

2017
Overview

• What are population protocols?
• What is leader election?
• What is the majority problem?
• Overview of the Paper
• Lower Bound for majority and leader election
• New algorithms for majority and leader election
  • Lottery Election (detailed)
  • Split-Join Majority (overview)
Population Protocols

- Set of $n \geq 2$ agents
- Each executing deterministic state machine
- State from a finite set $\Lambda_n$, might depend on $n$
- Transition function $\delta_n : \Lambda_n \times \Lambda_n \rightarrow \Lambda_n \times \Lambda_n$
- Output function $\gamma_n : \Lambda_n \rightarrow 0$
- Pairs of agents are chosen uniformly at random
- Each agent updates state according to transition function $\delta_n$
- Result can be checked with output function
Leader Election

- All agents start in the same initial state
- Output set $O$ is $\{Win, Lose\}$
- Goal: One agent has Output $Win$, the rest has $Lose$
Majority Problem

- Two initial states $A_n, B_n$
- Output set is $\{Win_a, Win_b\}$
- Goal: Output of every agent should correspond to majority of initial state
- If $A_n, B_n$ are split 50/50, output is arbitrary
The Paper

• Title: Time-Space Trade-Offs in Population Protocols (2017)
• Authors:
  • Alistarh (ETH Zürich)
  • Aspnes (Yale)
  • Eisenstat (Google)
  • Gelashvili (MIT)
  • Rivest (MIT)
• Trade-off between number of states and running time
• New and improved algorithms for majority and leader election
### The Paper

<table>
<thead>
<tr>
<th>Problem</th>
<th>Type</th>
<th>Expected Time Bound</th>
<th>Number of States</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Exact Majority</strong> $\epsilon = 1/n$</td>
<td>Algorithm</td>
<td>$O(n \log n)$</td>
<td>4</td>
<td>[DV12, MNRS14]</td>
</tr>
<tr>
<td></td>
<td>Algorithm</td>
<td>$O(\log^2 n)$</td>
<td>$\Theta(n)$</td>
<td>[AGV15]</td>
</tr>
<tr>
<td></td>
<td>Lower Bound</td>
<td>$\Omega(n)$</td>
<td>$\leq 4$</td>
<td>[AGV15]</td>
</tr>
<tr>
<td></td>
<td>Lower Bound</td>
<td>$\Omega(\log n)$</td>
<td>any</td>
<td>[AGV15]</td>
</tr>
<tr>
<td><strong>Leader Election</strong></td>
<td>Algorithm</td>
<td>$O(\log^3 n)$</td>
<td>$O(\log^3 n)$</td>
<td>[AG15]</td>
</tr>
<tr>
<td></td>
<td>Lower Bound</td>
<td>$\Omega(n)$</td>
<td>$O(1)$</td>
<td>[DS15]</td>
</tr>
<tr>
<td><strong>Exact Majority</strong> Leader Election</td>
<td>Lower Bound</td>
<td>$\Omega(n / \text{polylog} n)$</td>
<td>$&lt; 1/2 \log \log n$</td>
<td>This paper</td>
</tr>
<tr>
<td><strong>Exact Majority</strong></td>
<td>Algorithm</td>
<td>$O(\log^3 n)$</td>
<td>$O(\log^2 n)$</td>
<td>This paper</td>
</tr>
<tr>
<td><strong>Leader Election</strong></td>
<td>Algorithm</td>
<td>$O(\log^{5.3} n \log \log n)$</td>
<td>$O(\log^2 n)$</td>
<td>This paper</td>
</tr>
</tbody>
</table>

Figure 1: Summary of results and relation to previous work.
Overview

• What are population protocols?
• What is leader election?
• What is the majority problem?
• Overview of the Paper
• Lower Bound for majority and leader election
• New algorithms for majority and leader election
  • Lottery Election (detailed)
  • Split-Join Majority (overview)
Lower Bound (idea)

• Two-Step argument
• First: Hypothetical algorithm converges faster than allowed by the lower bound, set of low count states can be “erased”
• Second: Engineer examples to contradict the correctness of that algorithm
• Leader Election: Remaining low count states are set of all potential leaders
• Majority: Remaining low count states could sway the outcome of majority
Corollary 3.1. Any monotonic population protocol with $|\Lambda_n| \leq 1/2 \log \log n$ states for all sufficiently large number of agents $n$ that stably elects at least one and at most $\ell(n)$ leaders, must take $\Omega \left( \frac{n}{144 |\Lambda_n| \cdot |\Lambda_n|^6 \cdot \ell(n)^2} \right)$ expected parallel time to convergence.

- Monotonic: Number of states cannot decrease with increasing node count
Lower Bound (Majority)

**Corollary 3.2.** Any monotonic population protocol with $|\Lambda_n| \leq 1/2 \log \log n$ states for all sufficiently large number of agents $n$ that stably computes correct majority decision for initial configurations with majority advantage $\epsilon n$, must take $\Omega \left( \frac{n}{36 |\Lambda_n| \cdot |\Lambda_n|^6 \cdot \max(2|\Lambda_n|, \epsilon n)^2} \right)$ expected parallel time to convergence.

- Monotonic: Number of states cannot decrease with increasing node count
Overview

• What are population protocols?
• What is leader election?
• What is the majority problem?
• Overview of the Paper
• Lower Bound for majority and leader election
• New algorithms for majority and leader election
  • Lottery Election (detailed)
  • Split-Join Majority (overview)
Synthetic Coin Flips

- Problem: We need a random coin parameter in our state, but states are deterministic (e.g. no randomness)
- Solution: Synthetic Coin Flips
- When x and y interact, they flip their own values
- E.g. $value' = 1 - value$
- Reminder: Interactions are chosen uniformly at random
- Randomness is extracted from the scheduler
- They prove that w.h.p. the distribution quickly becomes uniform
- Important: Happens independently from the algorithm
All nodes start in the same state, comprised of parameters:

- coin = \{0, 1\} (initially 0)
- mode = \{seeding, lottery, tournament, minion\} (initially seeding)
- payoff, level, counter, phase, ones

4 Modes:
- Seeding Mode, Lottery Mode, Tournament Mode, Minion Mode

Fix a parameter $m \geq (10 \log n)^2$

Protocol will use $O(m)$ states per node
Lottery Election - Seeding Mode

- Used to mix the coin parameter close to uniform random
- $payoff, level = 0$
- $counter = 4$
- In the first four interactions, decrease counter (and flip $coin$)
- When counter reaches 0, move on to Lottery Mode
Lottery Election - Lottery Mode

- Used to generate payoff values
- Higher values are less likely, finding a leader becomes easier
- Increment $payoff$ when partner has $coin = 1$
- When partner has $coin = 0$ or $payoff = \sqrt{m}$, move on to Tournament Mode
Lottery Election - Tournament Mode

- Forces agents to compete
- Generates additional tie-braking random values (level)
- Initialize $level = 0$
- $level$ is incremented if agent consecutively sees $\Theta(\log payoff)$
  coins set to 1
- This is implemented using phase and ones
- Level is capped at $\frac{\sqrt{m}}{\log m}$
Lottery Election - Tournament Mode

• When 2 agents $x$ and $y$ meet, compare:
  • $x$.payoff and $y$.payoff
  • If payoff equal: compare $x$.level and $y$.level
  • If level equal: compare $x$.coin and $y$.coin

• The smaller valued agent goes into minion mode
• It adopts level and payoff of opponent
Lottery Election - Minion Mode

• Keeps record of the maximum \((payoff, level)\) pair ever seen
• Propagates leaders with high \(payoff\)
• Helps eliminate other contenders
• Important: \(coin\) value not used as tie-breaker
Lottery Election - Complexity

- coin, counter, modes, ones are in $O(1)$
- payoff is limited to $O(\sqrt{m})$
- level is limited to $O\left(\frac{\sqrt{m}}{\log m}\right)$
- phase is limited to $O(\log m)$ since:
  - $\Theta(\log payoff) = \Theta(\log \sqrt{m}) = \Theta\left(\frac{1}{2} \log m\right) = \Theta(\log m)$
- State size: $O(1) \times O(\sqrt{m}) \times O\left(\frac{\sqrt{m}}{\log m}\right) \times O(\log m) = O(m)$
- $m$ was set to $(10 \log n)^2$ so state size is $O(\log^2 n)$
- They also prove that it takes $O(\log^{5.3} n \log \log n)$ parallel time
Overview

• What are population protocols?
• What is leader election?
• What is the majority problem?
• Overview of the Paper
• Lower Bound for majority and leader election
• New algorithms for majority and leader election
  • Lottery Election (detailed)
  • Split-Join Majority (overview)
Split-Join Majority

- **State:** A is positive, B is negative
- **To limit state space:**
  - State is $\langle x, y \rangle$ where $x, y \in \{0, 1, 2, 2^2, ..., 2^{\log n}\}$
  - $value(\langle x, y \rangle) = x - y$, so $x$ is “positive” and $y$ is “negative”
- **Opinion A starts as** $\langle 2^{\log n}, 0 \rangle$
- **Opinion B starts as** $\langle 0, 2^{\log n} \rangle$
- **Strong states:** non-zero
- **Weak states:** $\langle 0, 0 \rangle^+$ or $\langle 0, 0 \rangle^-$
When two agents interact, operations cancel, join, split are carried out:

### Cancel

\[
<2^{k+1}, 0> + <0, 2^k> = <2^k, 0> + <0, 0^+>
\]

### Join

\[
<2^k, 0> + <0, 0^+> = <2^k, 0> + <0, 0^+>
\]

### Split

\[
<2^k, 0> + <0, 0^+> = <2^{k-1}, 0> + <2^{k-1}, 0>
\]
• Since all operations preserve the sum
• And the initial sum is leaning to one side
• It is impossible for all agents to sway to the “wrong” side
• The Authors prove that the algorithm is guaranteed to converge in $O(\log^3 n)$ parallel time in expectation and w.h.p.
Questions?