Recent Algorithmic Advances in Population Protocols

Rati Gelashvili

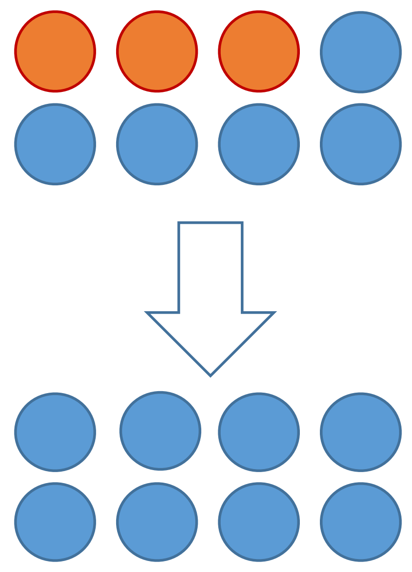
Population Protocols

[Angluin,Aspnes,Diamadi,Fischer,Peralta'04]

- n **Nodes**: ***simple, identical agents***
  - Each node is ***the same*** finite state automaton
  - For example: a molecule

- **Interactions** are ***pairwise***
  - According to a scheduler, e.g. random, weakly or globally fair
  - Among the edges of an underlying <u>communication graph</u>
  - Nodes update their state following interactions

- **Computation** is performed ***collectively***
  - Global ***configuration:*** *#nodes in each state*
  - No "fixed" decision time



In This Talk

Focus on a clique as an underlying graph

- Can be generalized to other communication graphs: [Draief,Vojnovic'12], [Sudo,Ooshita,Kakugawa,Masuzawa'12], [Mertzios,Nikoletseas,Raptopoulos,Spirakis'14]

Overview the model

- Number of interesting and studied settings and tasks

Essential Techniques for Protocol Design & Application Examples

- Phase Clocks
- Synthetic Coins
- Population Splitting

Computation

Stabilization:

Given an execution sequence up to a configuration, **configuration & all reachable configurations (must) satisfy a given predicate P**

- strongest possible requirement

Convergence:

From a configuration in a given interaction sequence, **configuration & all reachable configurations satisfy a given predicate P**

- good enough for many practical applications
- allows bypassing strong lower bounds for stabilization [Doty,Soloveichik'15,…]

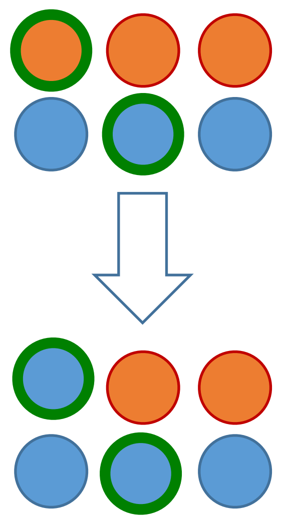
Always correct vs with high probability correct

Complexity Measures: Time

Meaningful requirements for scheduler

- weakly fair: *nodes interacting*
- globally fair: *reachable configurations reached*
- probabilistic: *most commonly, uniform random*

Stabilization (parallel) Time: *E[# interactions until stabilization] / n*
Convergence (parallel) Time: *E[# interactions until convergence] / n*



Parallel time: interpreted as interactions per node, or number of rounds

Complexity Measures: Space

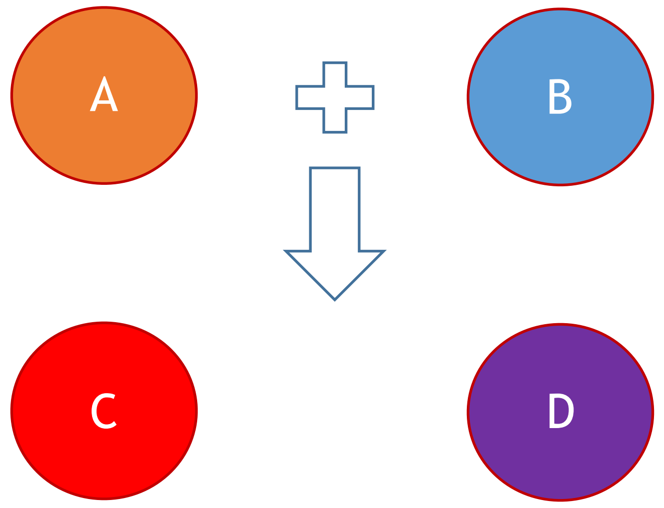
State Complexity: # ***distinct states*** *per node*

Most important measure

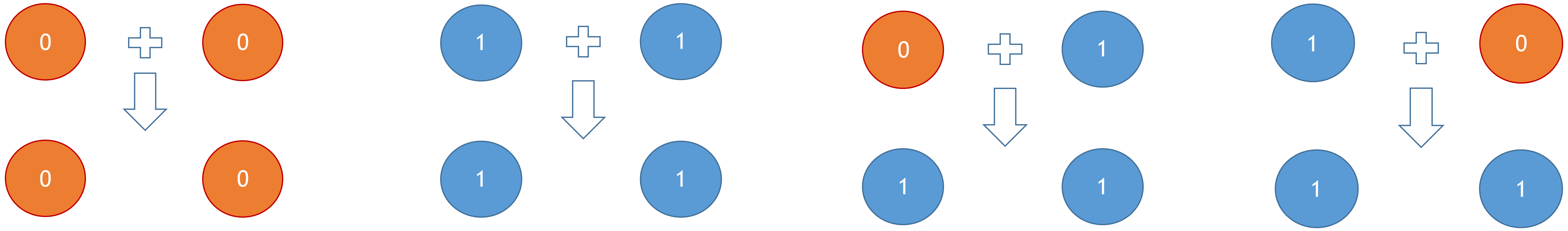
- Critical to be as small as possible
- Can be super-constant

What Can We Compute?

We can perform interactions of the type



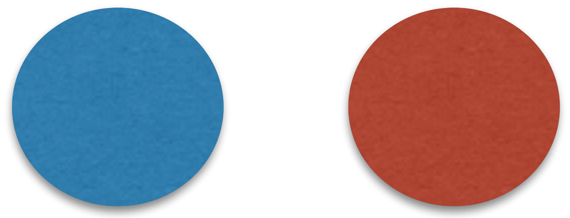
Computing **OR**



rumor (epidemy) spreading: takes O(log n) parallel time w.h.p.

Tasks: Majority

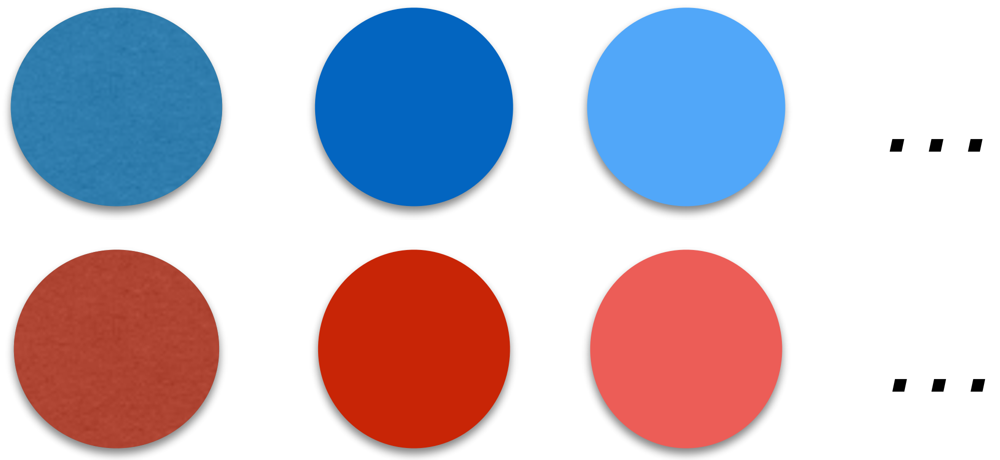
Two initial states**: A, B**



Output:

A if **#A** > **#B** initially.

B, otherwise.

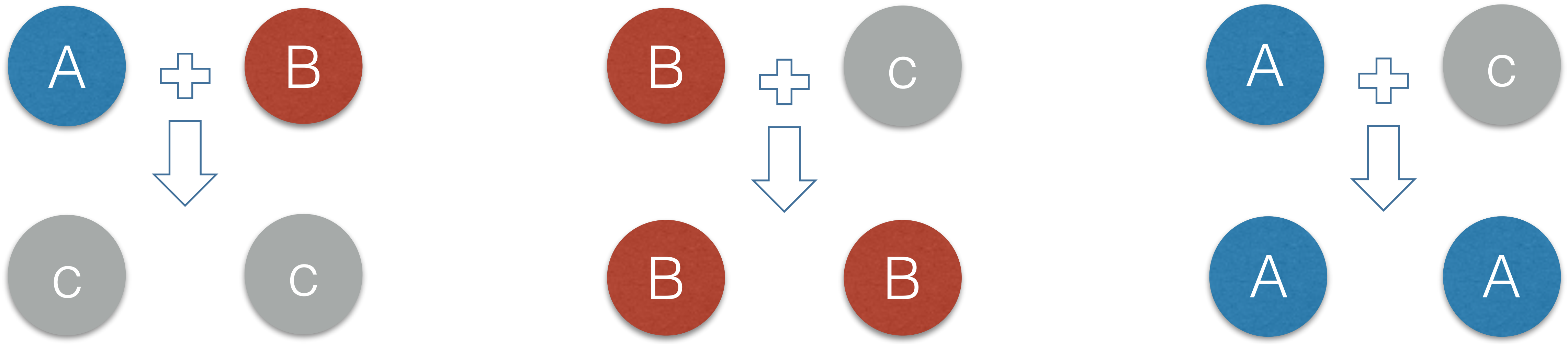


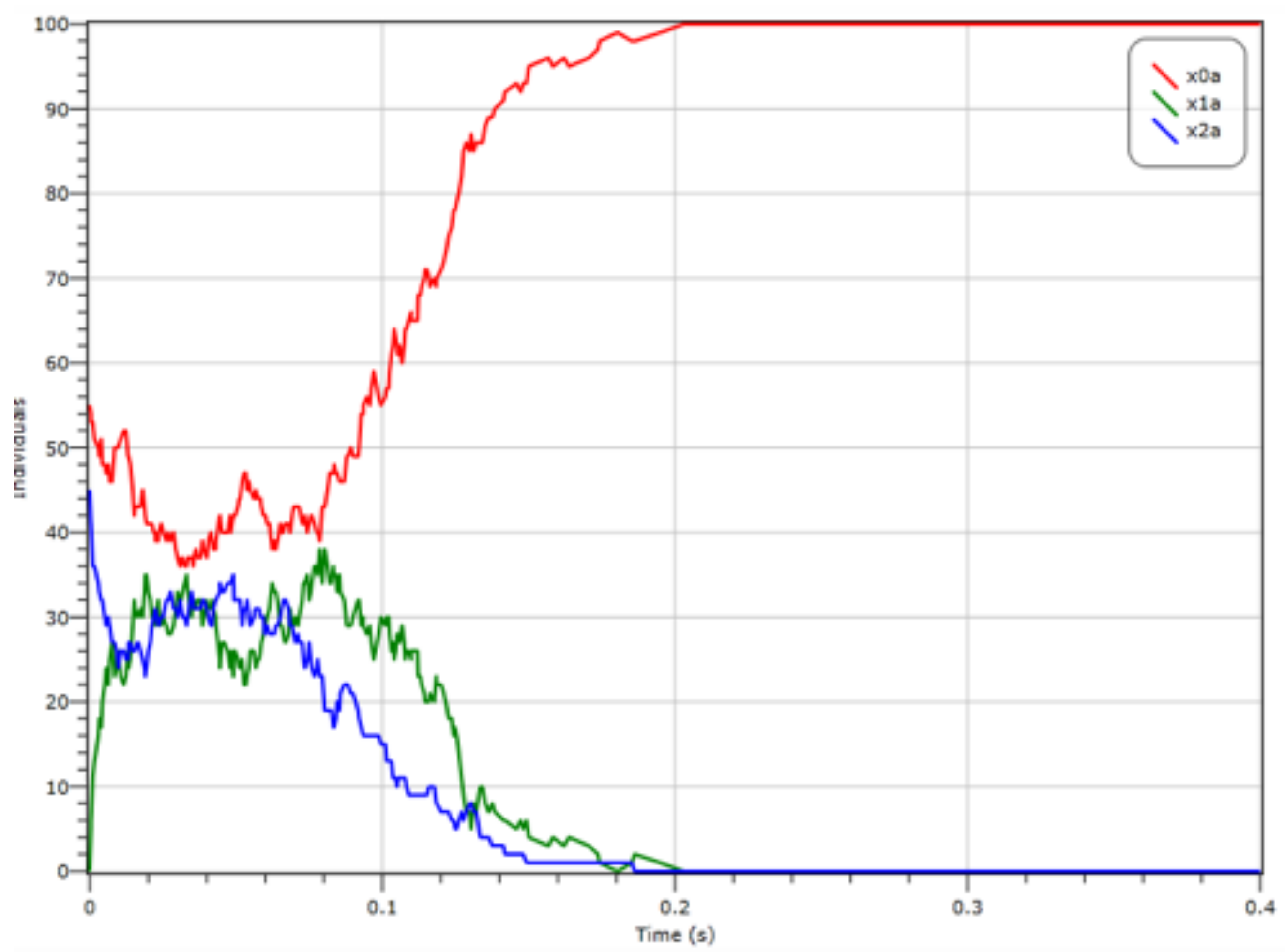
- The cell cycle switch implements approximate majority [Cardelli,Csikasz-Nagy'12]
- Implementation in DNA: [Chen,Dalchau,Srnivas,Philipps,Cardelli,Soloveichik,Seelig'13, Nature Nanotechnology]

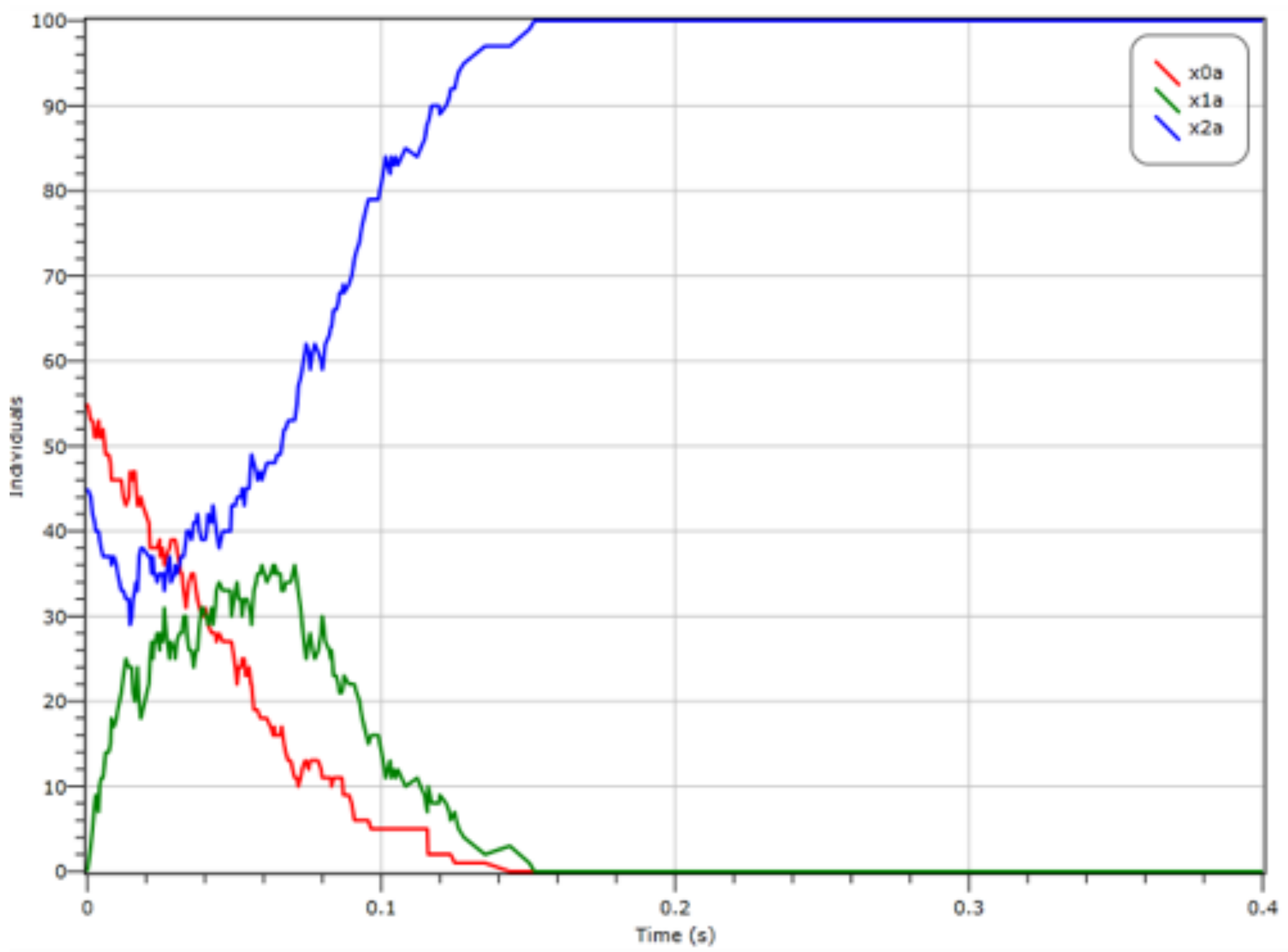
Example 3-State Protocol for Approximate Majority

[Angluin,Aspnes,Eisenstat'08, Draief,Vojnovic'12]

States: 

State Transition Rules: 





Initial Discrepancy
ε = |#A - #B| / n ≥ 1 / n.

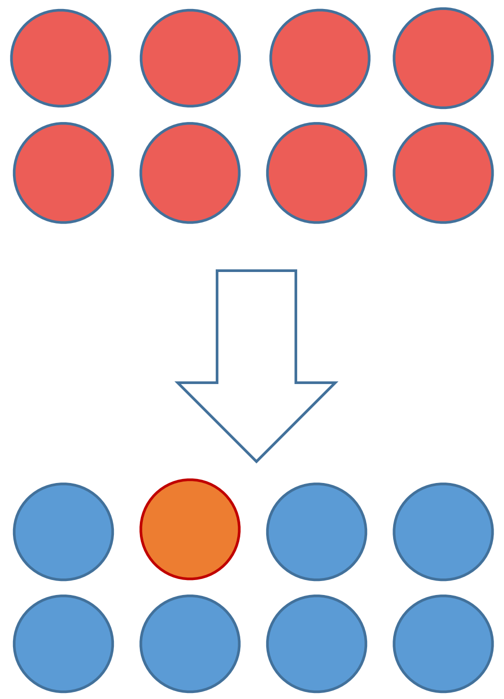
Given n nodes and discrepancy $\varepsilon > \log n / \sqrt{n}$, the running time is O(polylog n)

Error probability can be as high as ***constant*** for lower discrepancy.

Tasks: Leader Election

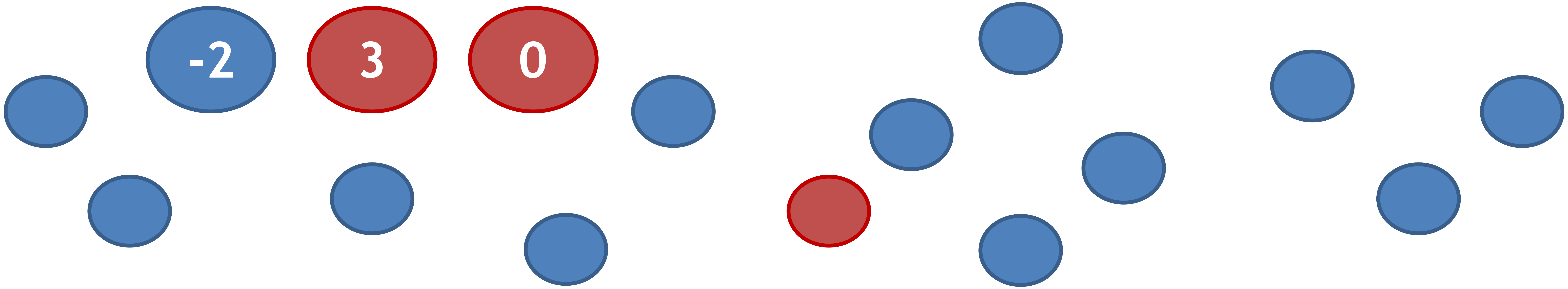
Input:

- All nodes start in the same initial state



Output:

- Exactly one node is in a "leader" state, remains leader forever

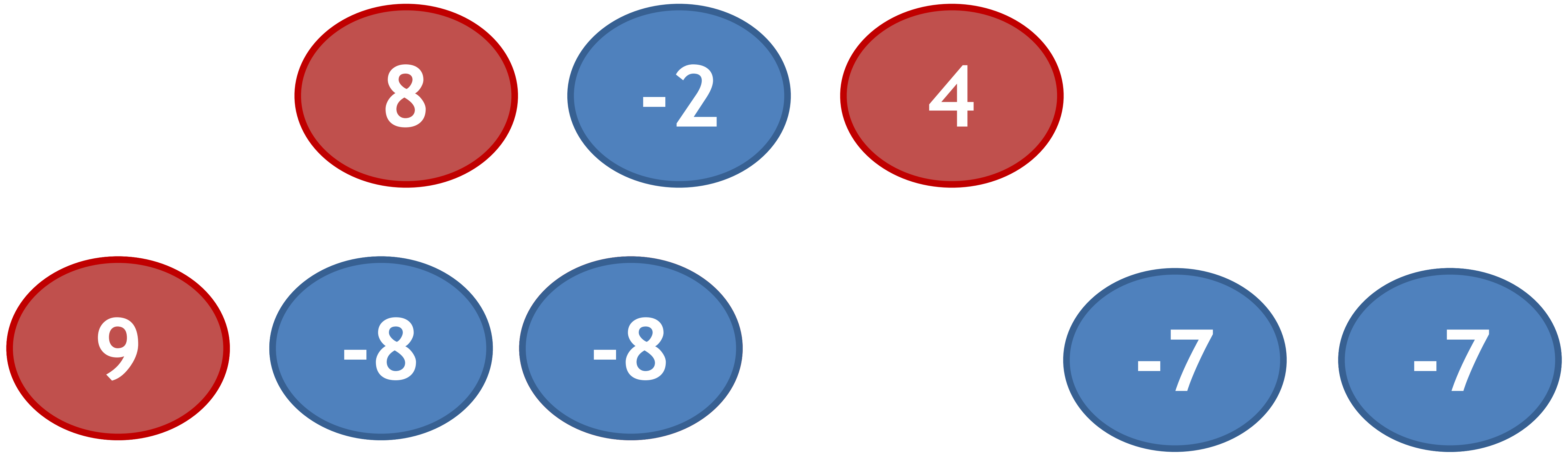


Correct, but slow

Example: Leader-Minion Algorithm

[Alistarh,Gelashvili'15]

Idea: use eliminated nodes as minions



If two contenders have values *c log n* apart, with *constant* probability, after *O(n log n)* interactions, one of them will not be a contender

For any two contenders, after *O(n log² n)* interactions, with *constant* probability, their values will be *c log n* apart

Additional Info

Bootstrapping protocols

- from with high probability to always correct

Other tasks

- Plurality, Counting, Naming

Other Settings

- self-stabilization: *possibly too hard for this model*
- loose Stabilization: *allow temporary divergence*
- robustness: *leaderless protocols, resilience to leaks*

Population Protocol Design Toolkit

1. Phase Clocks

[Angluin,Aspnes,Eisenstat,Ruppert'07]

Allows agents to have a common notion of time

- collectively count *phases* O(n log n) interactions
- original construction used constant states, required *a leader*

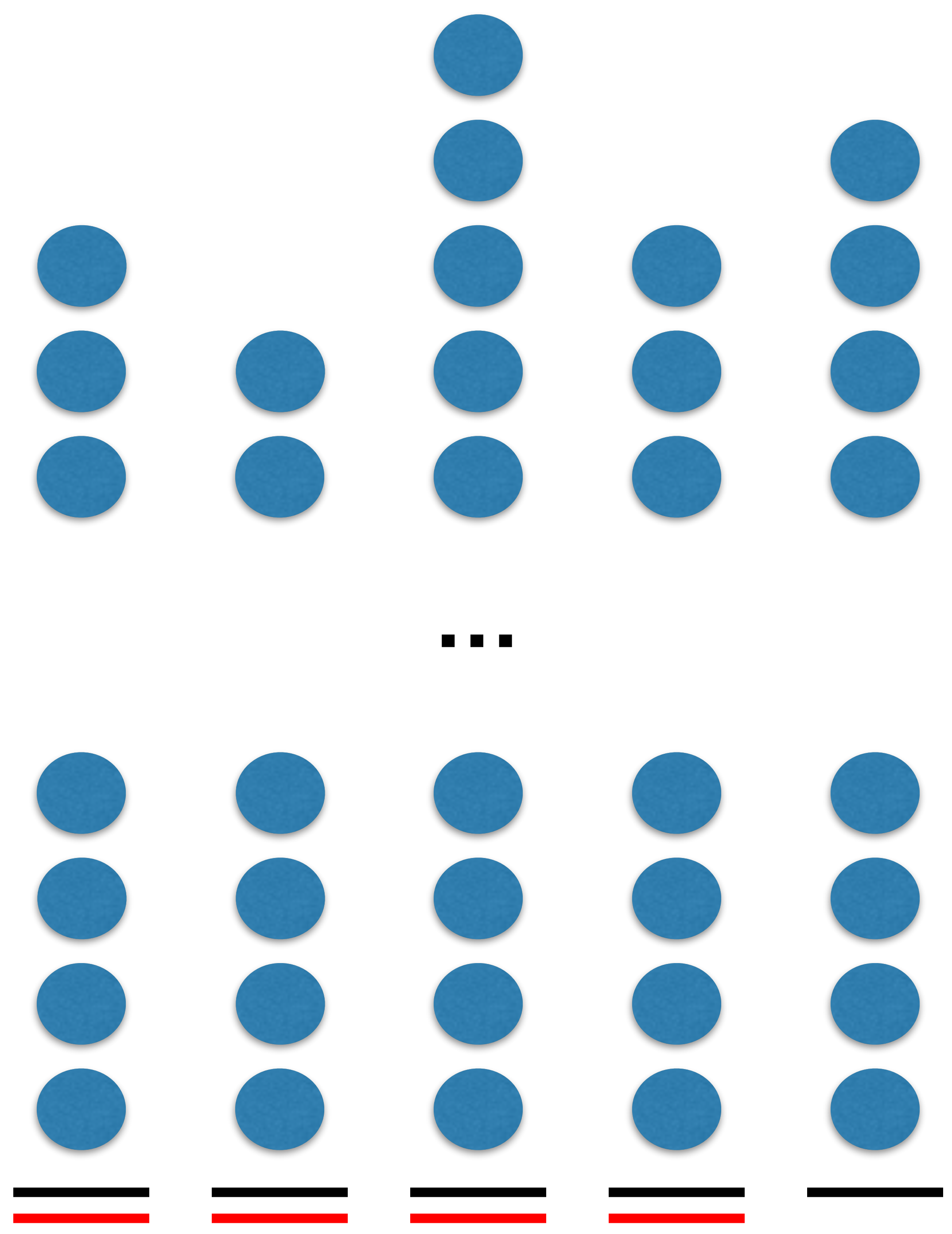
Limited use in algorithm design until lately:

- *Leaderless* phase clocks with O(log n) states [Alistarh,Aspnes,Gelashvili'17]
- *Junta-based* phase clocks with O(log log n) states [Gasieniec,Stachowiak'17]

Leaderless Phase Clock: 2-Choice Load Balancing

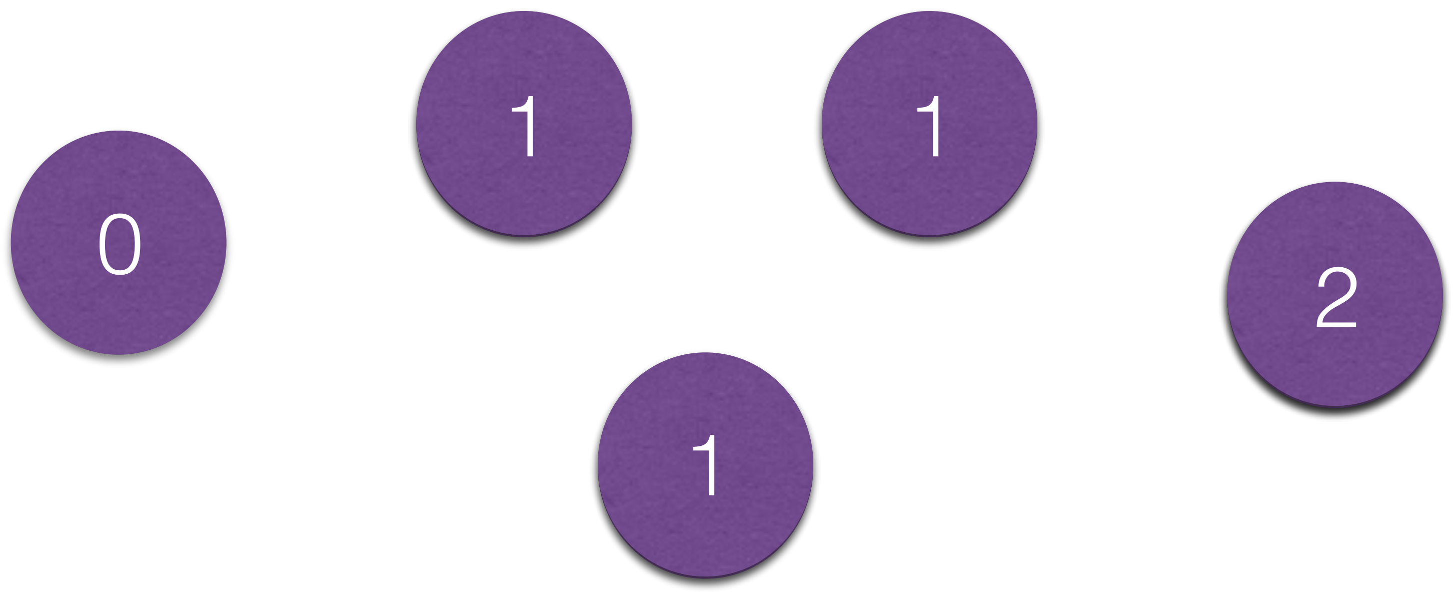
n empty bins, m >> n rounds, in each round

- choose two bins at random
- pick the bin with fewer balls, add a new ball

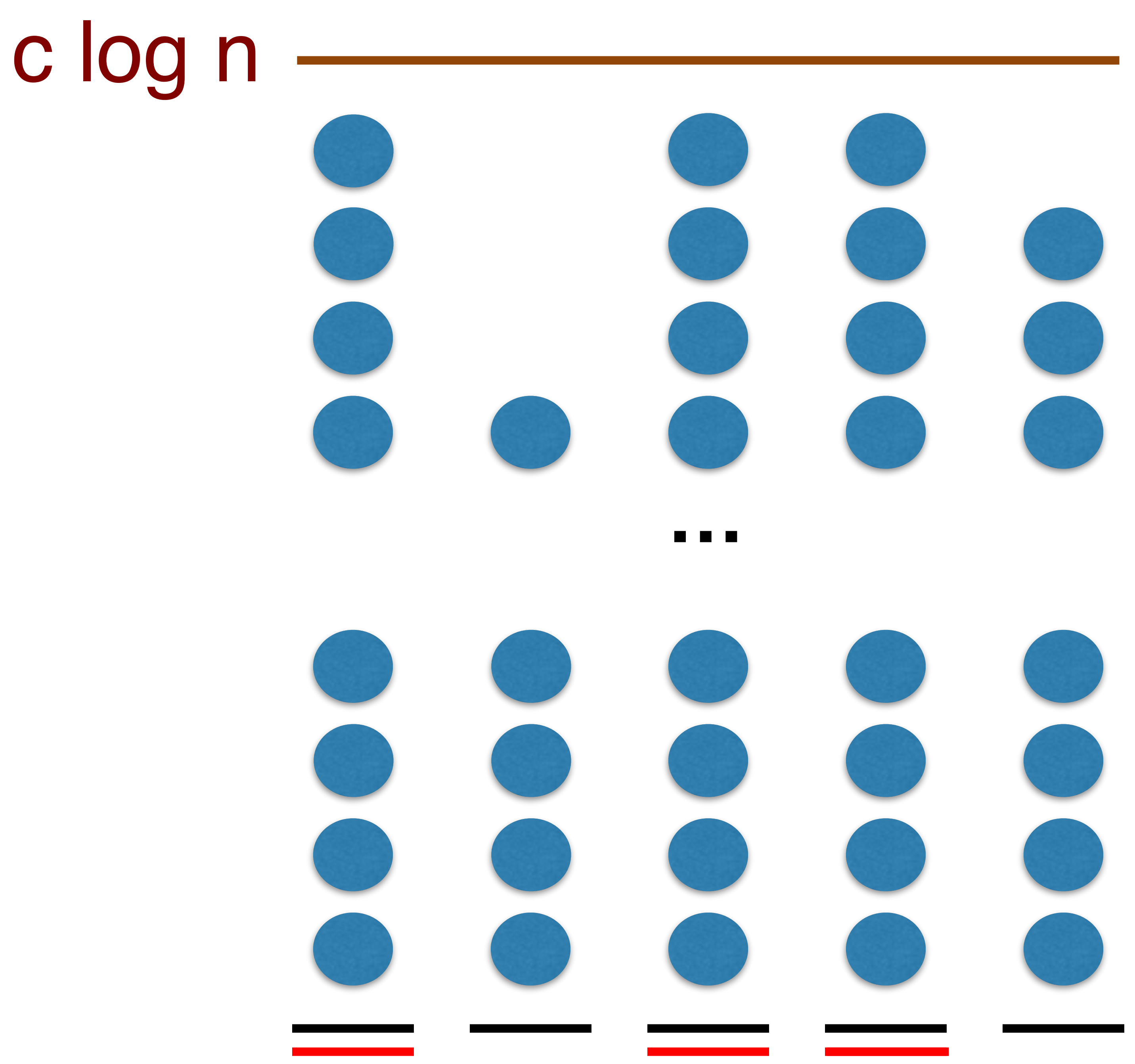


Theorem[Peres,Talwar,Wieder'15]**:** at any time, the difference between maximum and minimum number of balls in bins is at most O(log n), with high probability

Leaderless Phase Clock



Nodes simulate 2-choice process



..modulo c log n, with wraparound

..possible with high probability when c is large enough, such that the O(log n) gap is smaller than c log n

Phase Clocks

Junta-based clock [GS'17] works in two stages

- elect a junta of $n^{1-\varepsilon}$ nodes (uses O(log log n) states)
- implement and analyze a phase clock suggested by [AAER'07]

Follow up by [Berenbrink,Elsässer,Friedetzky,Kaaser,Kling,Radzik'18]:

- possible to reuse O(log log n) states after the first stage
- elegant and simplified exposition of [GS'17]

Hierarchy of phase clocks [Kosowski,Uznanski'18]

- count in phases of $O(n \log^k n)$ interactions for parameter k
- compute semi-linear predicates fast without a leader extending [Angluin,Aspnes,Eisenstat'08]

Population Protocol Design Toolkit

2. Synthetic coins

[Alistarh,Aspnes,Eisenstat,Gelashvili,Rivest'17]

The state transition function of population protocols is deterministic

- could randomization help in algorithm design?
- yes, e.g. loosely-stabilizing leader election ***if*** nodes have access to uniform random bits [Sudo,Ooshita,Kakugawa,Masuzawa'14]

But there is a source of randomness: the scheduler.

Extract *synthetic randomness*! (slight increase in state complexity)

[Cardelli,Kwiatkowska,Laurenti'16] introduced a similar construction, focusing on computability

Synthetic Coins

Simplest Algorithm:

- **the state:** a flip bit F, initially 0
- **initialization:** do four interactions, updating F = 1 – F'
- **simulated coin flip:** use F of the interaction partner

Analyzed as a random walk on a hypercube

- after constant parallel time, roughly half 0s and 1s

Major improvements by [Berenbrink,Kaaser,Kling,Otterbach'18]

- generate coins with a specific (non-zero) bias
- get a stronger concentration by extending the initialization stage

Faster construction of a spectrum of coins with different biases

- by [Gasieniec,Stachowiak,Uznanski'18], extending first stage of [Gasieniec,Stachowiak'17]

Population Protocol Design Toolkit

3. Population Splitting

[Ghaffari,Parter'16]

Reduces state complexity when there are mutually exclusive roles

- node's state does not need to encode all roles at once!

Idea used ad-hoc in some algorithms

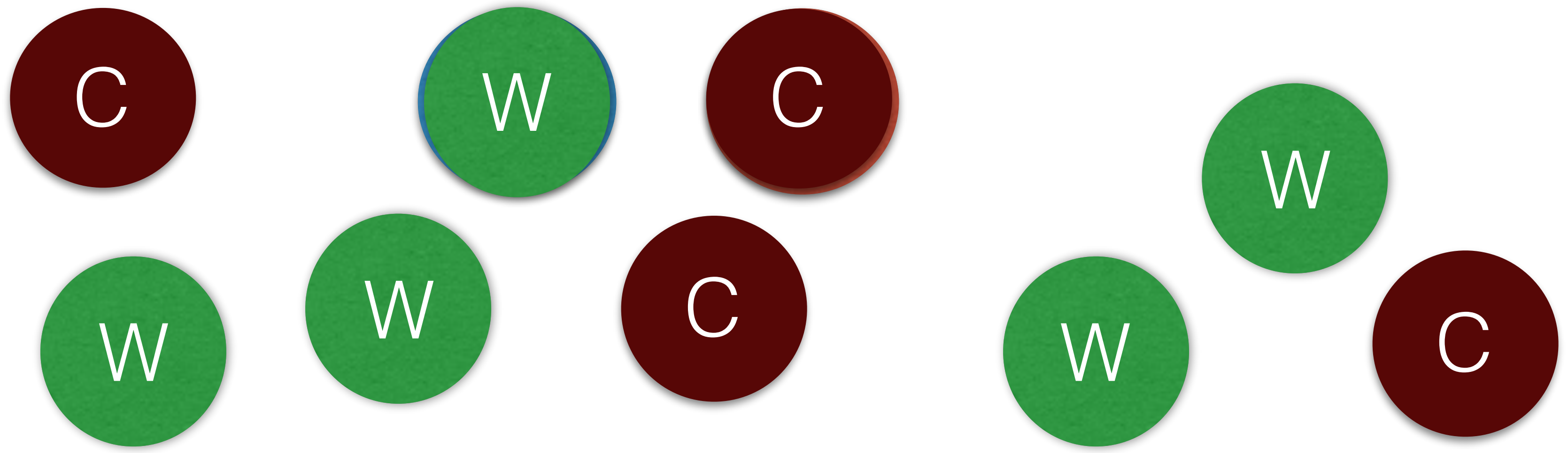
- Leader-Minion [AG'15], each node either a leader or a minion at any time
- indicator for stage of the protocol and role [AAEGR'17], rest of the state shared

can be thought of as some sort of task allocation [Cornejo,Dornhaus,Lynch,Nagpal'14]

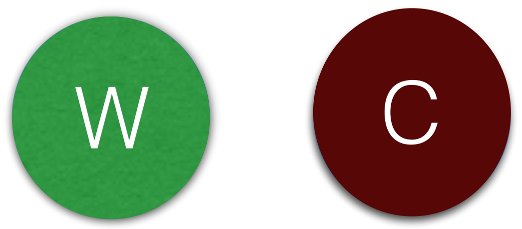
Population Splitting

Example explicit application from [Alistarh,Aspnes,Gelashvili'18]

- During first interactions, one node becomes *a worker*, another *a clock*



May need to use synthetic coins to break ties



Other examples:

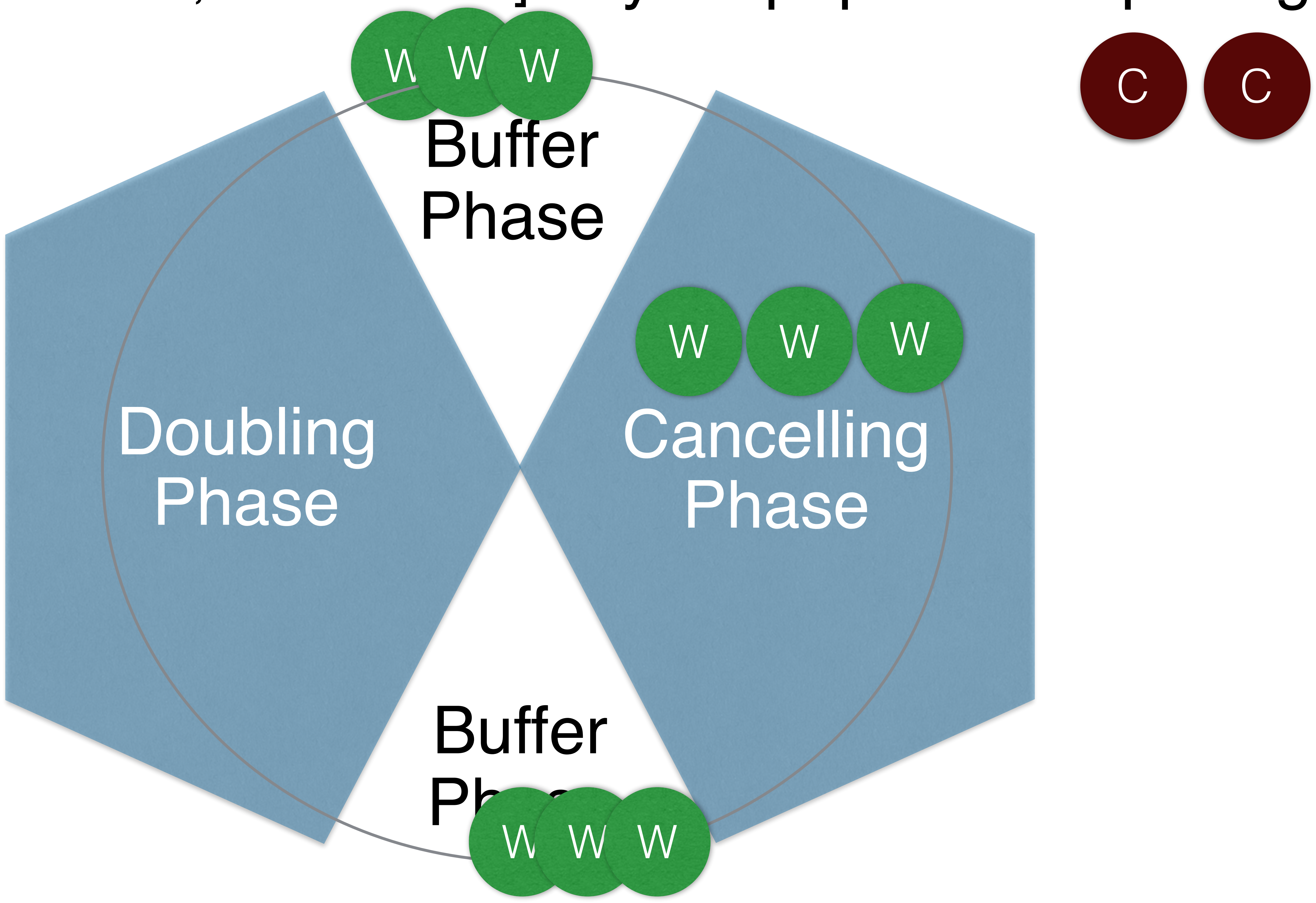
- [Gasieniec,Stachowiak,Uznanski'18]: most complex explicit splitting
- [Berenbrink,Elsässer,Friedetzky,Kaaser,Kling,Radzik'18]: most delicate explicit splitting

Applications: Majority

Requires Ω(log n) states to stabilize in polylog(n) time*

- Both state-optimal protocols [AAG'18,BEFKKR'18] rely on population splitting

- phases of O(n log n) interactions, w.h.p.
- use rumor spreading (phase updates, exceptions, etc)
- need backup protocols
- [BEFKKR'18] fuses phases together, splitting gets complicated



*under some combinatorial assumptions that all known protocols satisfy

Applications: Leader Election

Synthetic coin invented for leader election, still used in best protocols [Gasieniec,Stachowiak'17, Gasieniec,Stachowiak,Uznanski'18]. Original ideas:

- use coin outcome to decide to increase seeding or not
- lottery: decide whether to drop out based on random seeding

Powerful combination with phase clocks, e.g. in each phase

- flip an almost fair coin
- rumor spread existence of 1 to eliminate all 0s from contention

Conclusions

Population Protocols are a fertile ground for algorithmic research

- ..and lower bounds also based on nice combinatorial arguments

Interesting to explore directions

- Other graphs
- Other tasks
- Convergence vs stabilization vs loose stabilization
- Approximate Protocols
- Remove assumption in the majority lower bound

While staying simultaneously aware of motivations and open-minded

The contents of this talk will appear as a survey in SIGACT News.