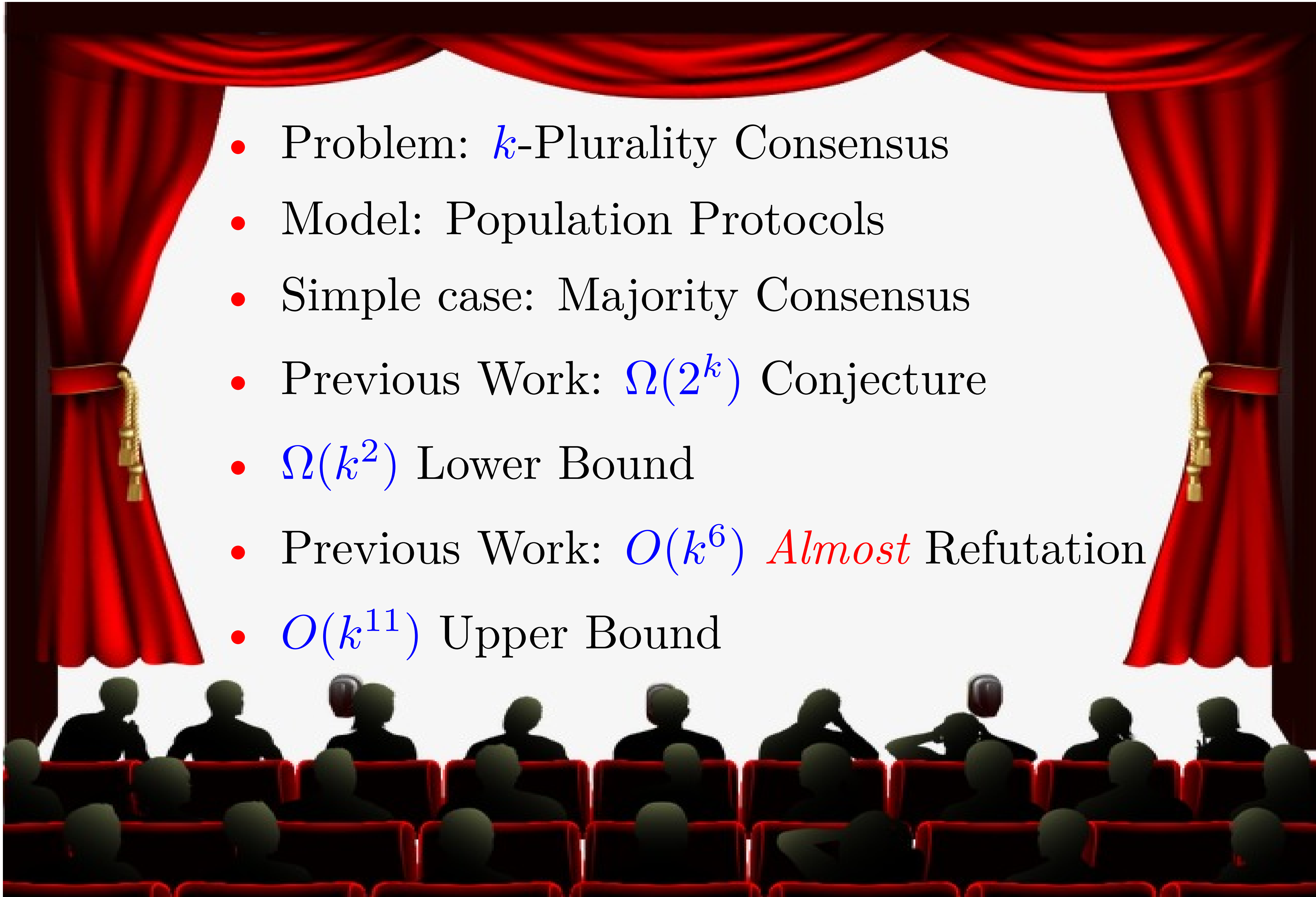
Natural Distributed Algorithms

- Lecture 6 -
Necessary Memory for Majority in Population Protocols

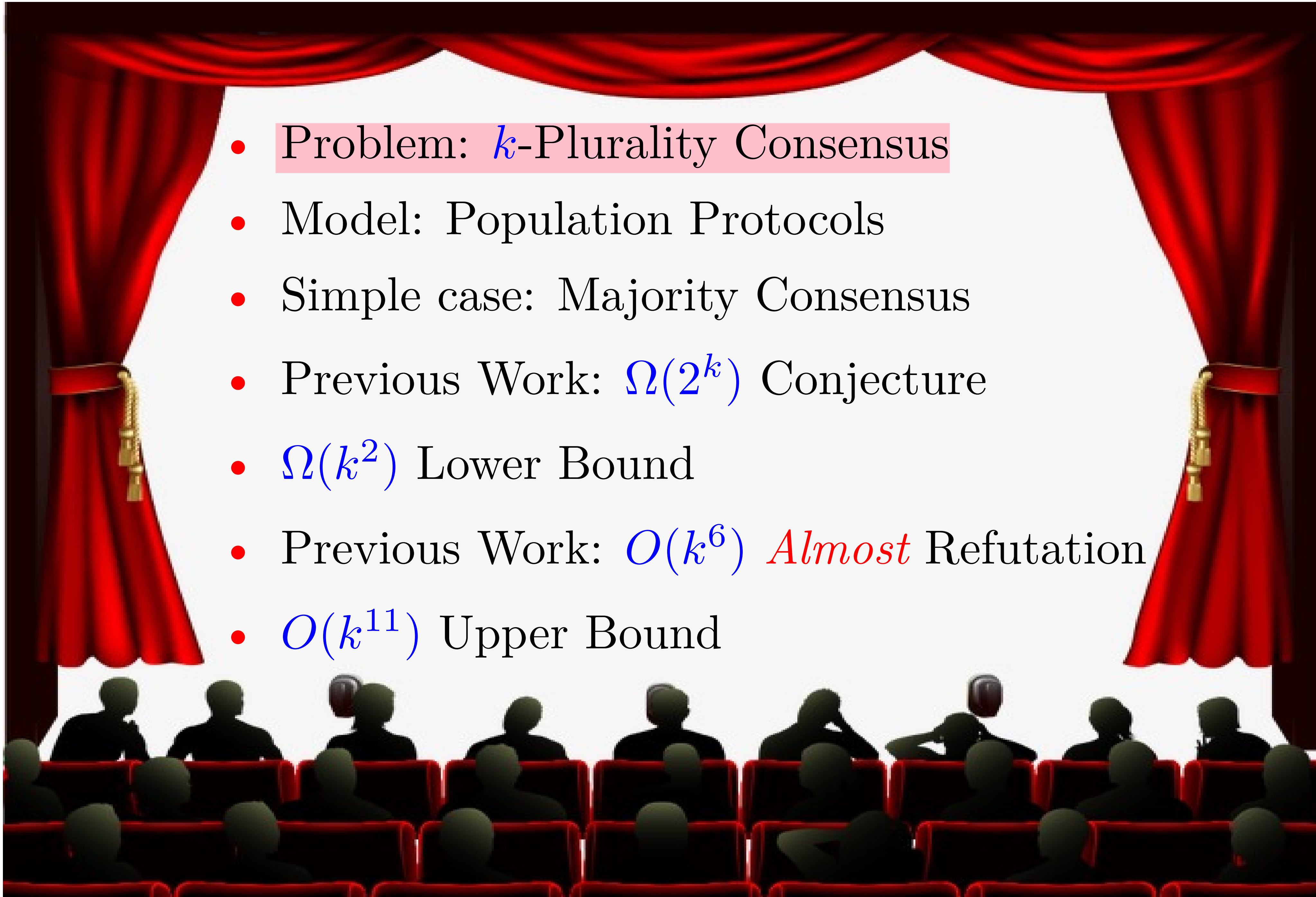
Emanuele Natale
CNRS - UCA
CdL in Informatica
Università degli Studi di Roma
"Tor Vergata"

Outline



- Problem: k-Plurality Consensus
- Model: Population Protocols
- Simple case: Majority Consensus
- Previous Work: $\Omega(2^k)$ Conjecture
- $\Omega(k^2)$ Lower Bound
- Previous Work: $O(k^6)$ *Almost* Refutation
- $O(k^{11})$ Upper Bound

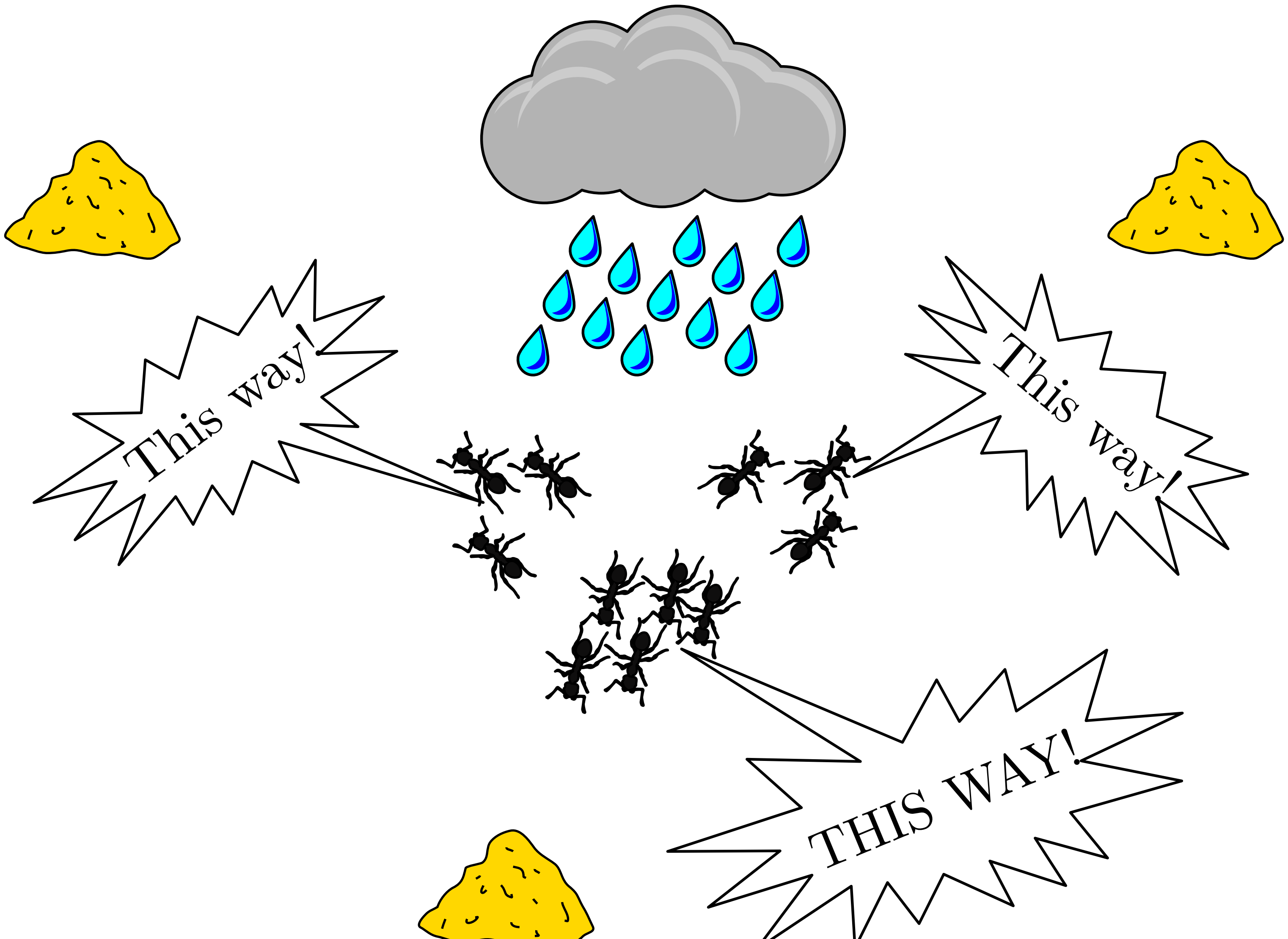
Outline



- Problem: k-Plurality Consensus
- Model: Population Protocols
- Simple case: Majority Consensus
- Previous Work: $\Omega(2^k)$ Conjecture
- $\Omega(k^2)$ Lower Bound
- Previous Work: $O(k^6)$ *Almost* Refutation
- $O(k^{11})$ Upper Bound

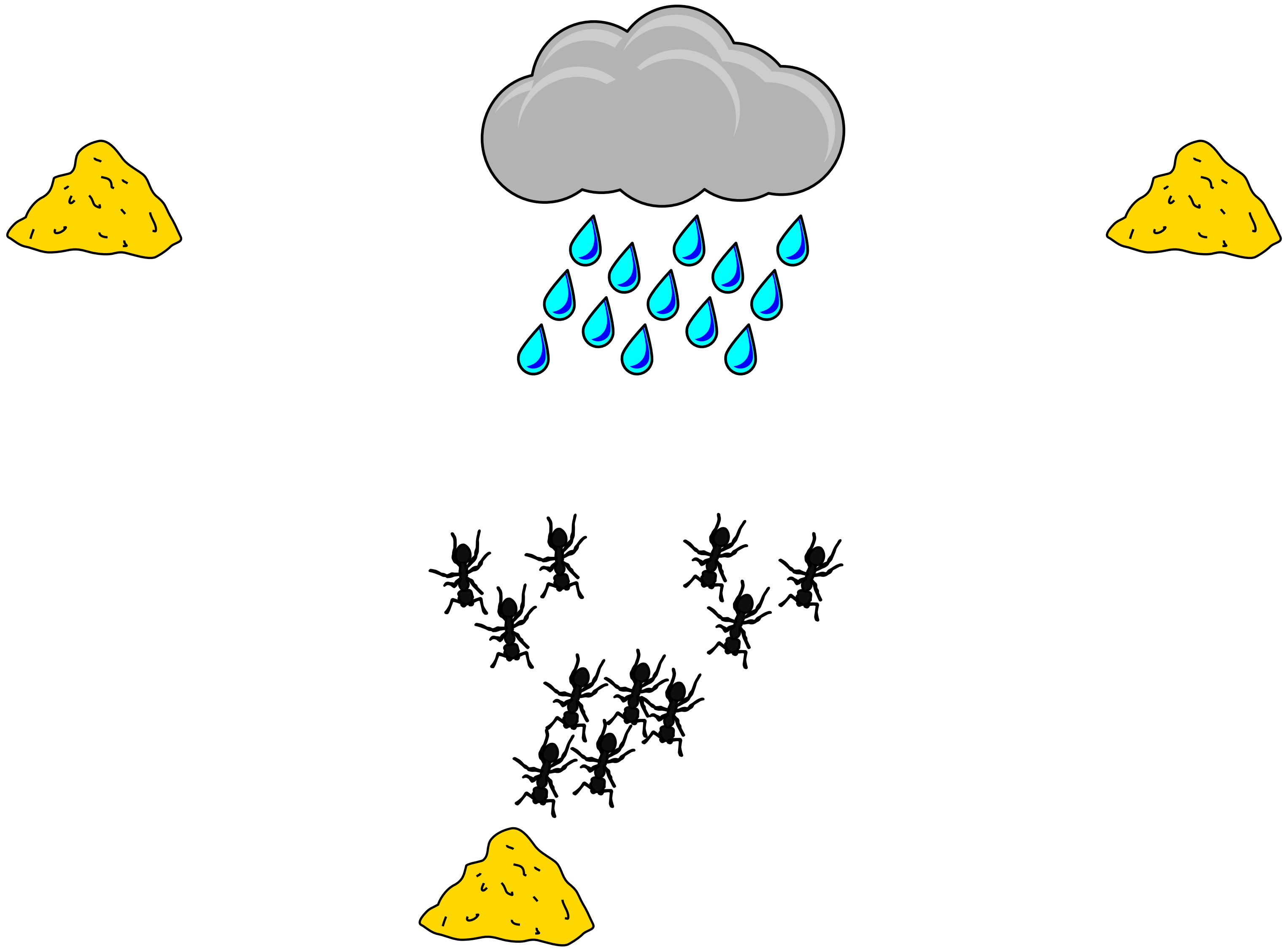
Recall: k-Plurality Consensus

Each agent supports one out of k opinions

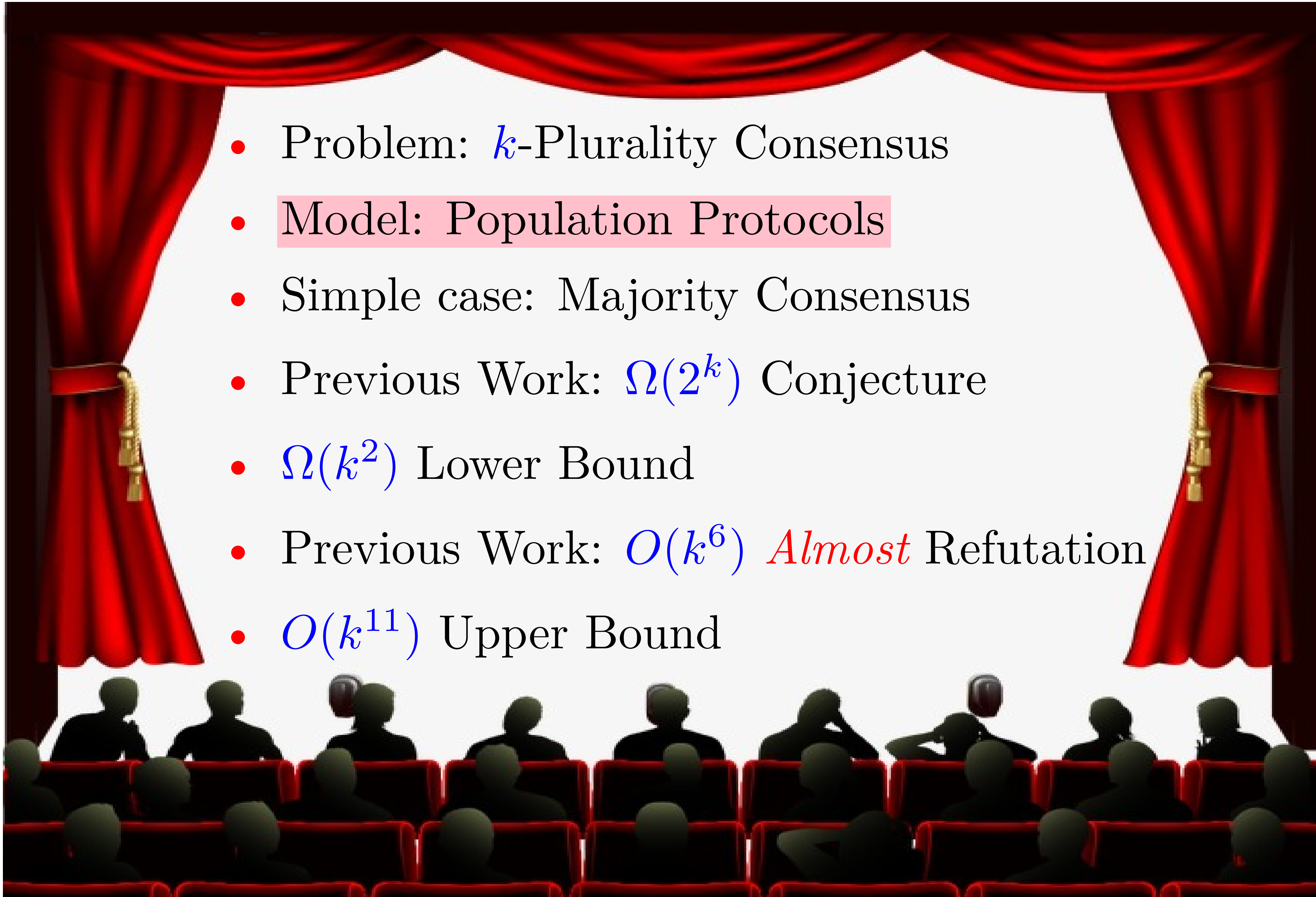


Recall: k-Plurality Consensus

All agents eventually support the same opinion



Outline



- Problem: k-Plurality Consensus
- Model: Population Protocols
- Simple case: Majority Consensus
- Previous Work: $\Omega(2^k)$ Conjecture
- $\Omega(k^2)$ Lower Bound
- Previous Work: $O(k^6)$ *Almost* Refutation
- $O(k^{11})$ Upper Bound

Population Protocols

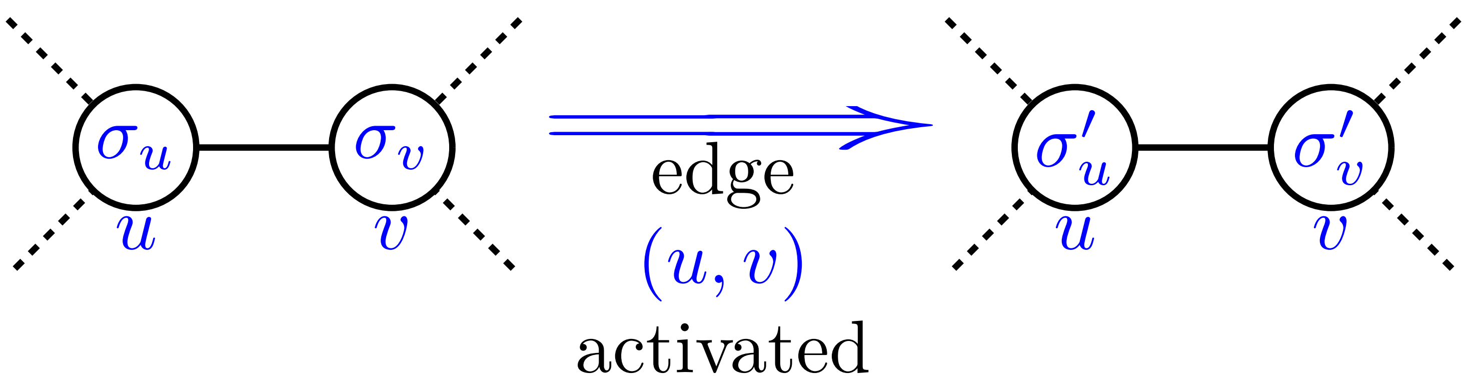
AKA chemical reaction networks, poisson clock models, etc.

Population Protocols

AKA chemical reaction networks, poisson clock models, etc.

- (Directed) graph G,
- set of nodes' states $\Sigma = (\sigma_u)_{u \in V}$,
- edges activated by a *scheduler*,
- function $\gamma : \Sigma \times \Sigma \to \Sigma \times \Sigma$ s.t. if edge (u, v) with states (σ_u, σ_v) activated, new states are

$$\gamma(\sigma_u, \sigma_v) = (\sigma'_u, \sigma'_v)$$



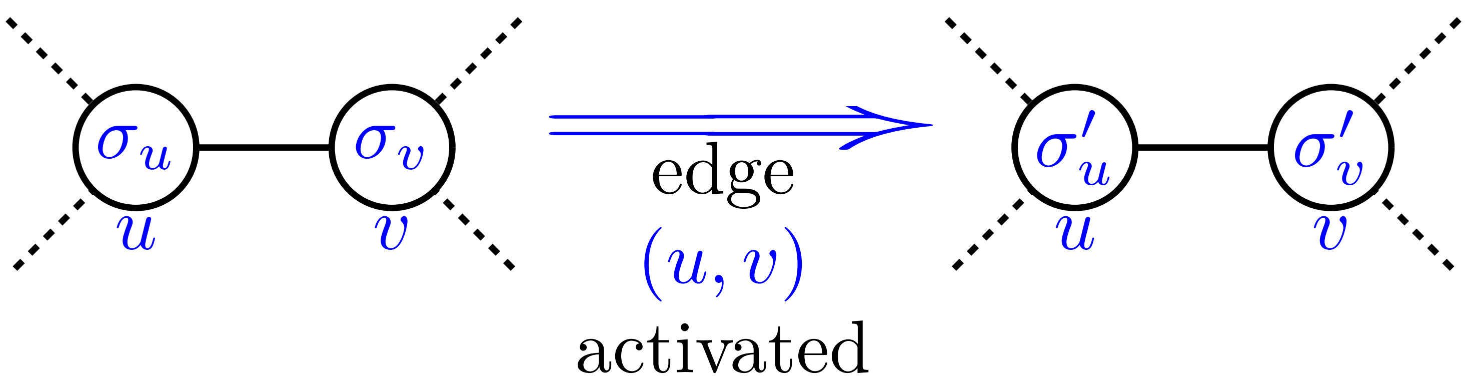
Population Protocols

AKA chemical reaction networks, poisson clock models, etc.

- (Directed) graph G,
- set of nodes' states $\Sigma = (\sigma_u)_{u \in V}$,
- edges activated by a *scheduler*,
- function $\gamma : \Sigma \times \Sigma \to \Sigma \times \Sigma$ s.t. if edge (u, v) with states (σ_u, σ_v) activated, new states are

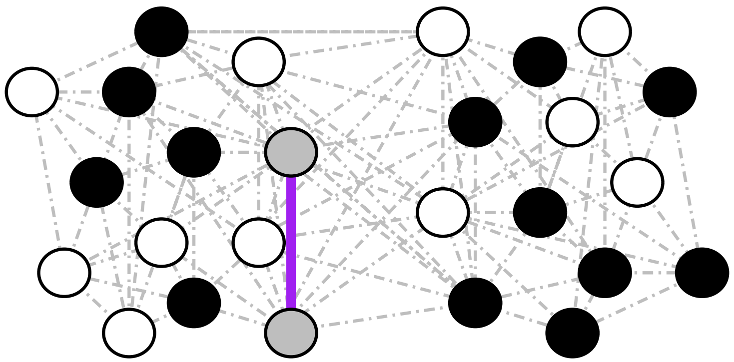
$$\gamma(\sigma_u, \sigma_v) = (\sigma'_u, \sigma'_v)$$

protocol's memory



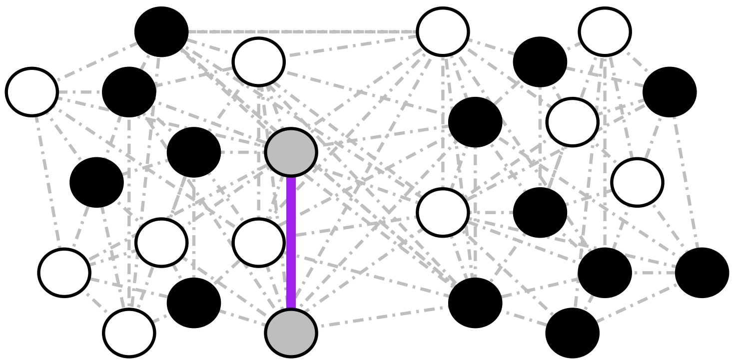
Population Protocols: Schedulers

Probabilistic scheduler: activate an edge chosen at random



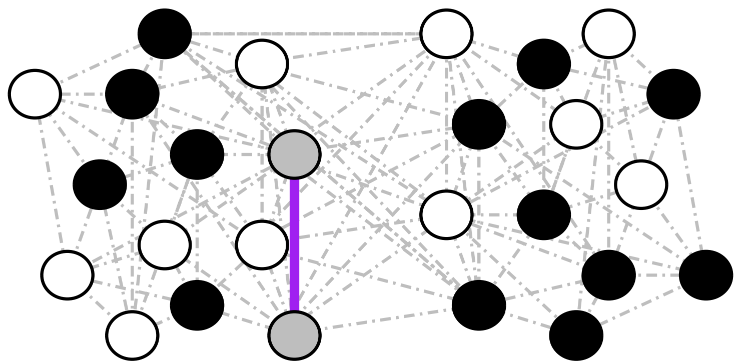
Population Protocols: Schedulers

Probabilistic scheduler:
activate an edge chosen
at random



What if a protocol P should never fail?

Population Protocols: Schedulers



Probabilistic scheduler: activate an edge chosen at random

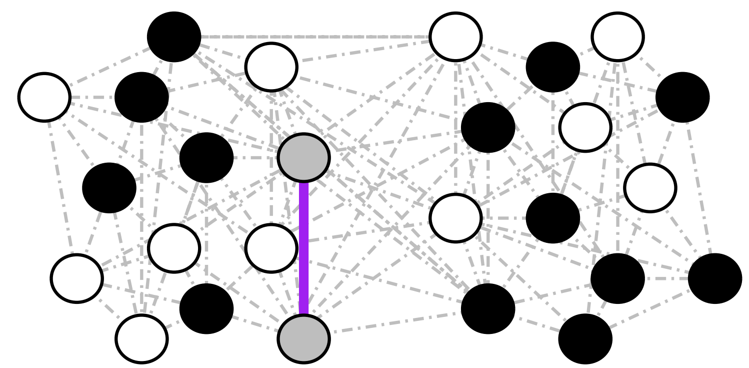
What if a protocol P should never fail?

A *configuration* is the state of all nodes $S = (\sigma_1, ..., \sigma_n)$.

S' *reachable* from S if it is possible to activate edges such that S becomes S'.

Population Protocols: Schedulers

Probabilistic scheduler: activate an edge chosen at random



What if a protocol P should never fail?

A *configuration* is the state of all nodes $S = (\sigma_1, ..., \sigma_n)$.

S' *reachable* from S if it is possible to activate edges such that S becomes S'.

Fair scheduler: if S appears infinitely often, also any conf. reachable from S appears infinitely often:

S' reachable from S and $S_1, S_2, ..., S, ..., S, ..., S, ...$

$\implies S_1, S_2, ..., S', ..., S', ..., S', ...$

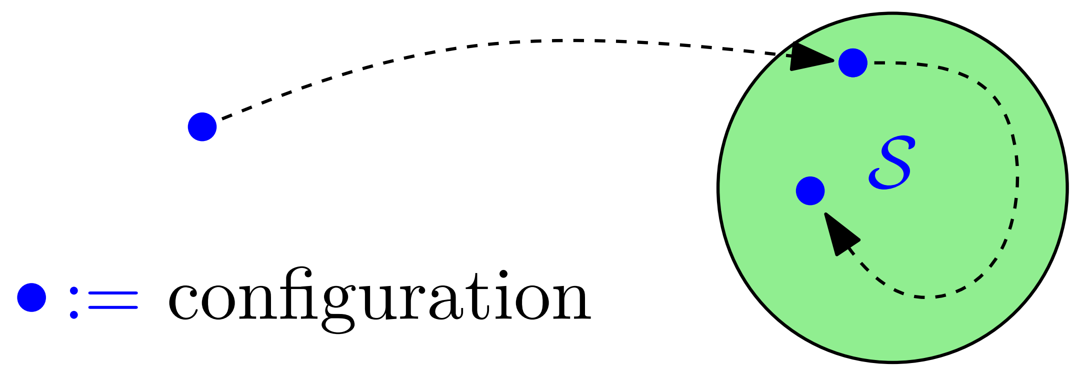
Self-Stabilization

n agents with states in Σ. Σ^n possible configurations.

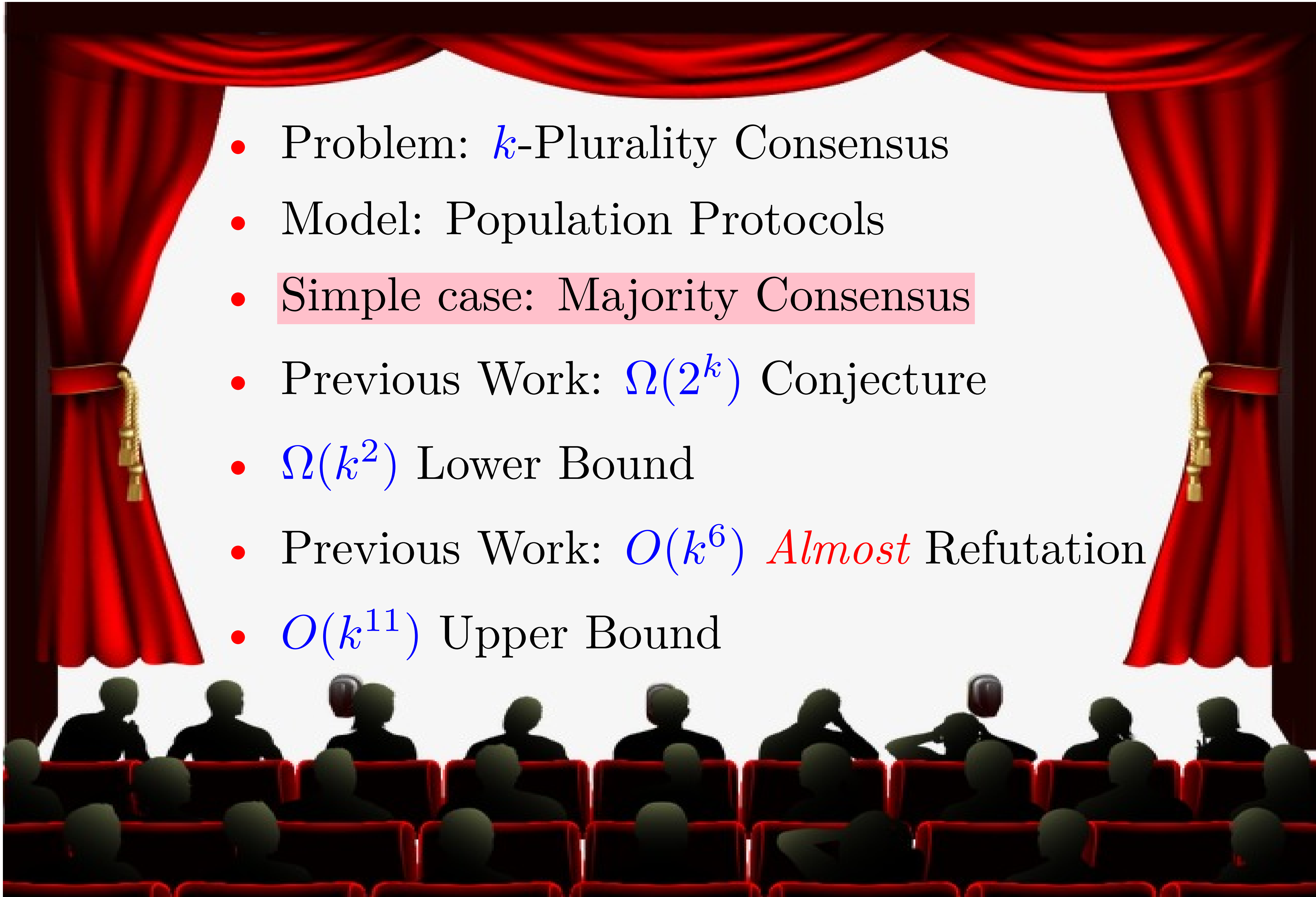
$\mathcal{S}$:= {"*correct* states of the system" }.

Convergence. Starting from any possible configuration, the system eventually reaches a configuration in $\mathcal{S}$.
Closure. If configuration in $\mathcal{S}$, it remains in $\mathcal{S}$.

A protocol is self-stabilizing iff guarantees convergence and closure w.r.t. $\mathcal{S}$.



Outline



- Problem: k-Plurality Consensus
- Model: Population Protocols
- Simple case: Majority Consensus
- Previous Work: $\Omega(2^k)$ Conjecture
- $\Omega(k^2)$ Lower Bound
- Previous Work: $O(k^6)$ *Almost* Refutation
- $O(k^{11})$ Upper Bound

Majority (2-Plurality) Consensus: 2-bit Protocol

[Mertzios et al. ICALP'16, Benezit et al. ICASSP'09]

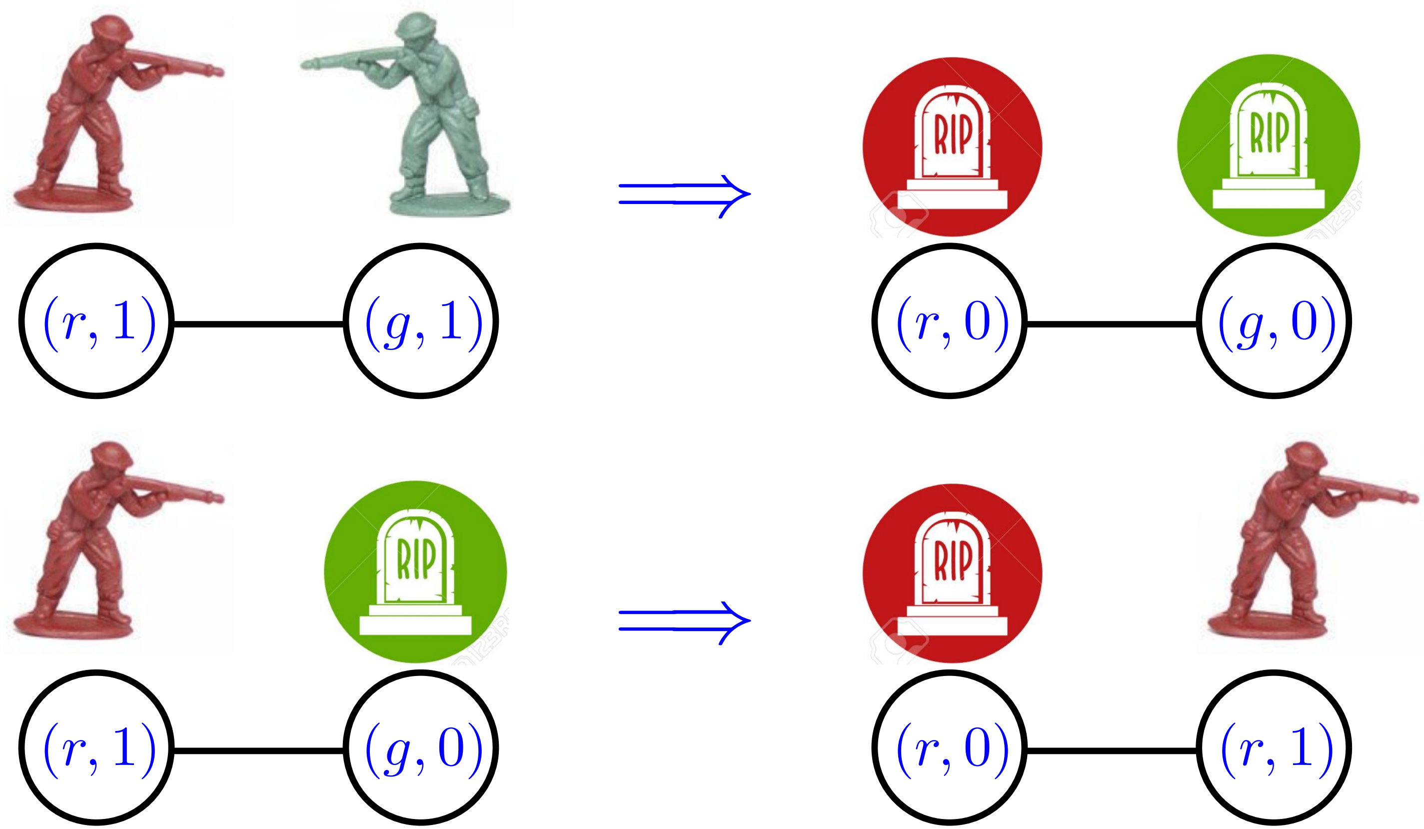
State: (green/red, defended or not)

Majority (2-Plurality) Consensus: 2-bit Protocol

[Mertzios et al. ICALP'16, Benezit et al. ICASSP'09]

State: (green/red, defended or not)

$u \setminus v$	$(g,0)$	$(g,1)$	$(r,0)$	$(r,1)$
$(g,0)$	–	$((g,1),(g,0))$	–	$((r,1),(r,0))$
$(g,1)$	$((g,0),(g,1))$	–	$((g,0),(g,1))$	$((g,0),(r,0))$
$(r,0)$	–	$((g,1),(g,0))$	–	$((r,1),(r,0))$
$(r,1)$	$((r,0),(r,1))$	$((r,0),(g,0))$	$((r,0),(r,1))$	–



Idea of Proof for 2-bit Protocol

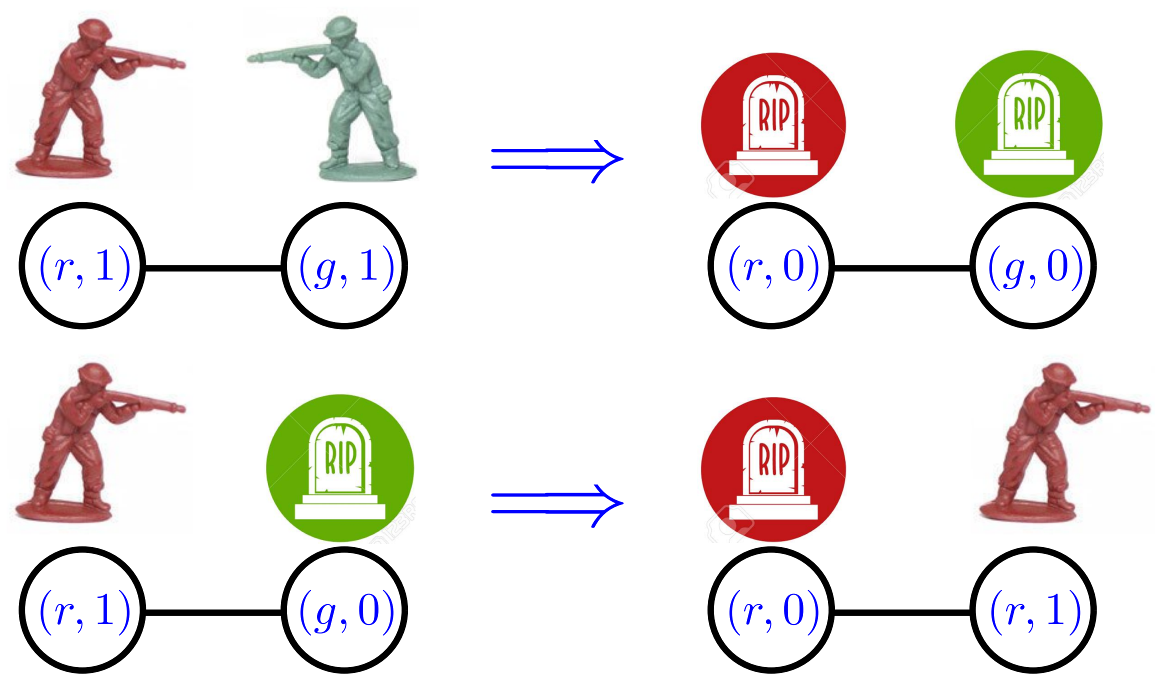
State: (green/red, defended or not)

[Mertzios et al. ICALP'16,
Benezit et al. ICASSP'09]

Idea of Proof for 2-bit Protocol

State: (green/red, defended or not)

[Mertzios et al. ICALP'16, Benezit et al. ICASSP'09]



- If there is a clear majority, at some point there is only one type (green or red) of "strong agent"
- At some point the strong agent visits all nodes

Majority Consensus: 2-bit Lower Bound

[Mertzios et al. ICALP'16]

Three possible states: $1, 0, \alpha$.

Majority Consensus: 2-bit Lower Bound

[Mertzios et al. ICALP'16]

Three possible states: $1, 0, \alpha$.

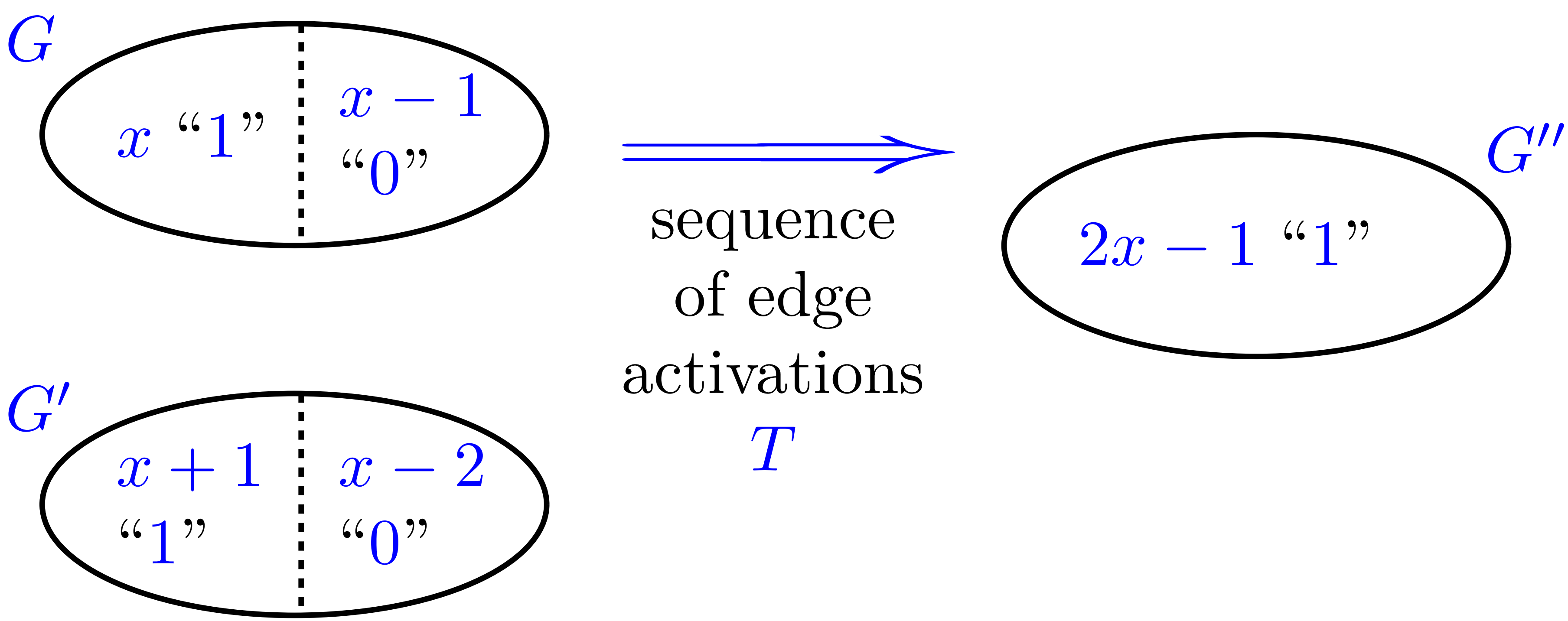
Observe: α counts either as "output 1" or "output 0".
Wlog assume α counts as "output 0".

Majority Consensus: 2-bit Lower Bound

[Mertzios et al. ICALP'16]

Three possible states: $1, 0, \alpha$.

Observe: α counts either as "output 1" or "output 0".
Wlog assume α counts as "output 0".

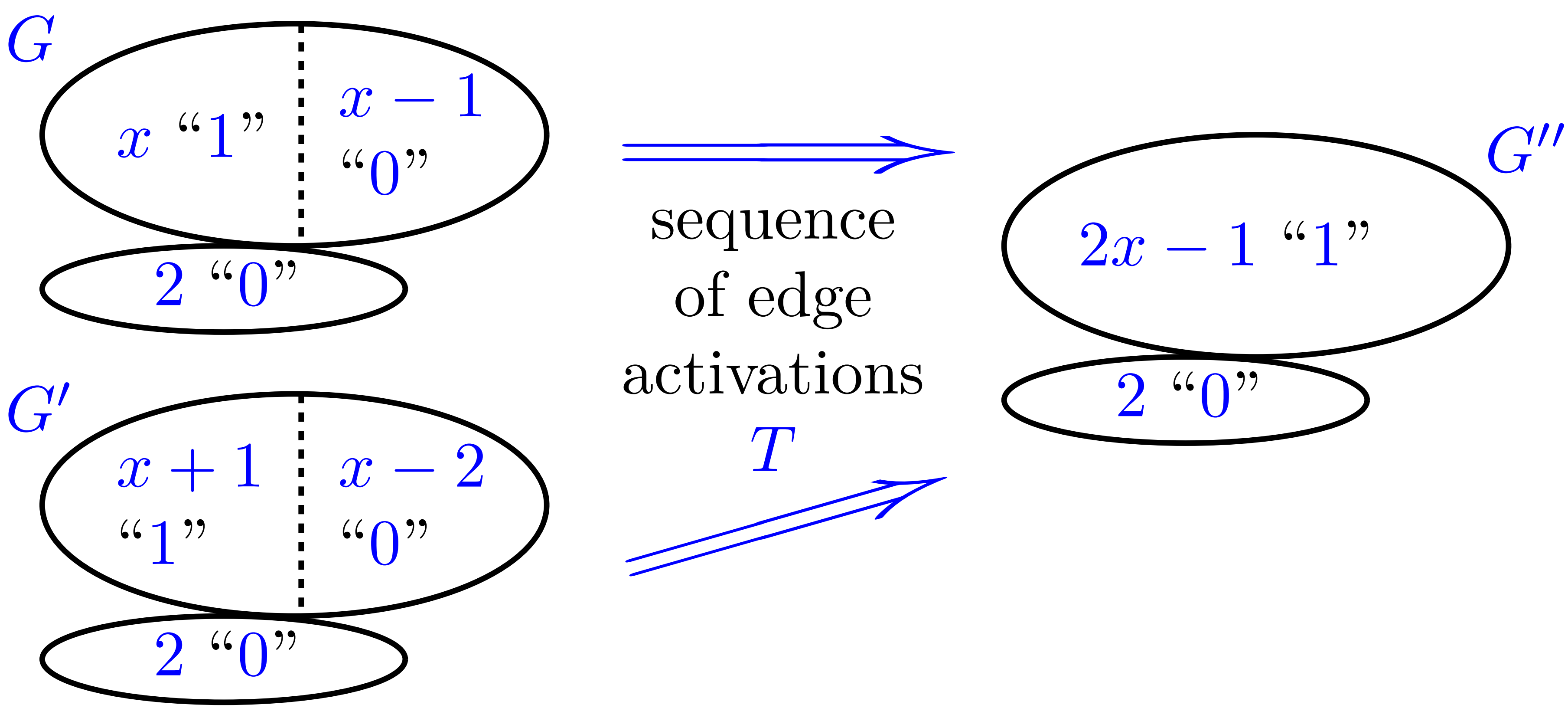


Majority Consensus: 2-bit Lower Bound

[Mertzios et al. ICALP'16]

Three possible states: $1, 0, \alpha$.

Observe: α counts either as "output 1" or "output 0".
Wlog assume α counts as "output 0".

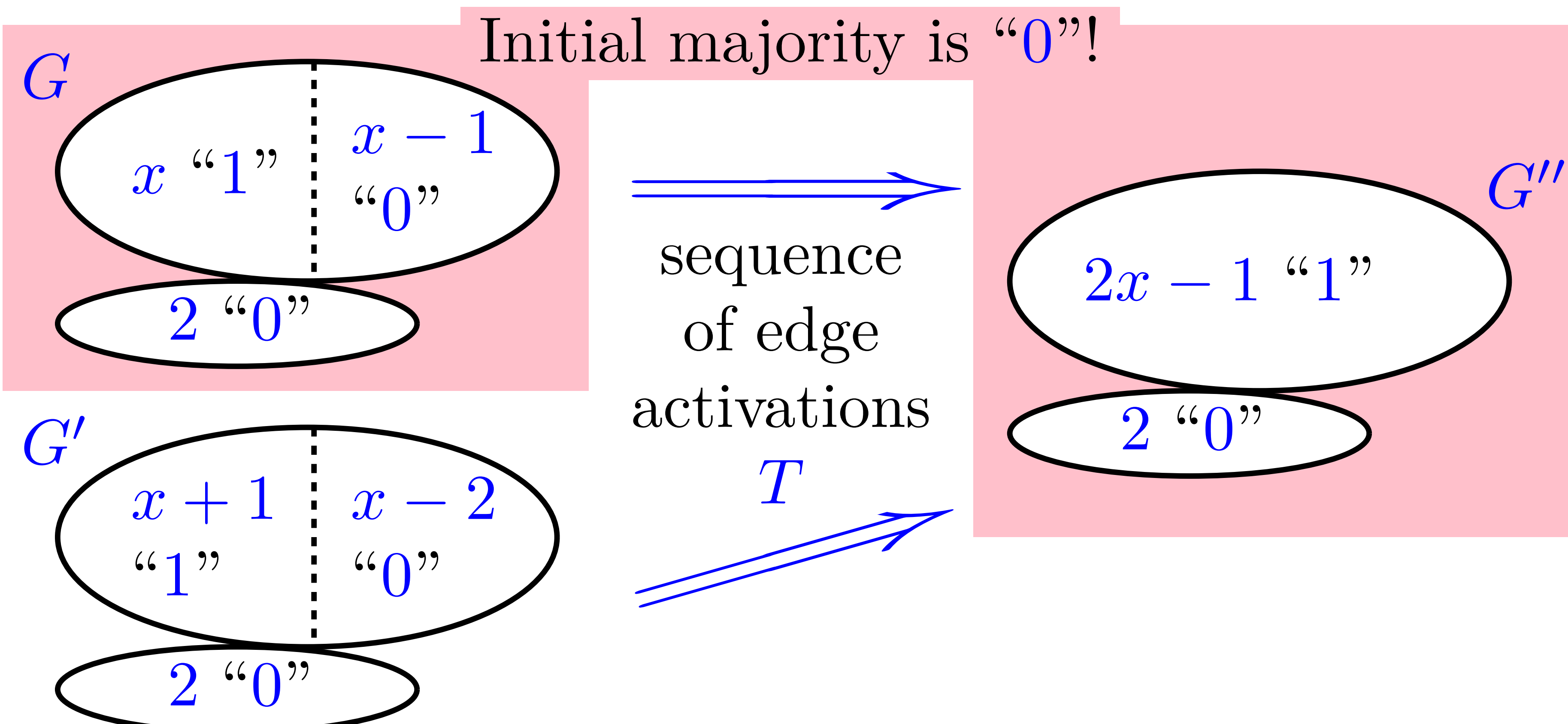


Majority Consensus: 2-bit Lower Bound

[Mertzios et al. ICALP'16]

Three possible states: $1, 0, \alpha$.

Observe: α counts either as "output 1" or "output 0".
Wlog assume α counts as "output 0".



Outline

- Problem: k-Plurality Consensus
- Model: Population Protocols
- Simple case: Majority Consensus
- Previous Work: $\Omega(2^k)$ Conjecture
- $\Omega(k^2)$ Lower Bound
- Previous Work: $O(k^6)$ *Almost* Refutation
- $O(k^{11})$ Upper Bound

Salehkaleybar et al.'s Conjecture [TSIPN'15]

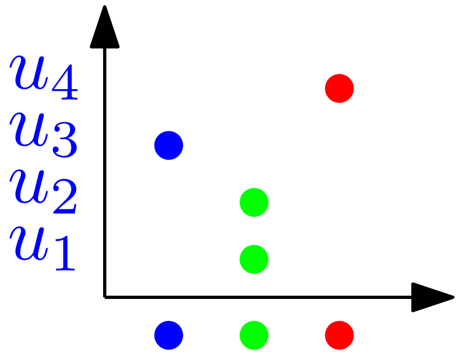
Problem. Plurality consensus in population protocols with fair scheduler.
Opinions can *only be tested for equality*.

Salehkaleybar et al.'s Conjecture [TSIPN'15]

Problem. Plurality consensus in population protocols with fair scheduler.
Opinions can *only be tested for equality*.

Protocol DMVR.

- Each node initially has a *coin* = its opinion

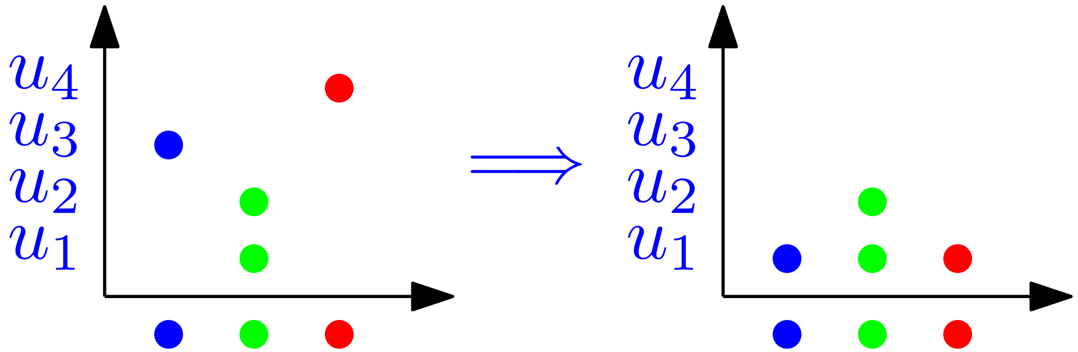


Salehkaleybar et al.'s Conjecture [TSIPN'15]

Problem. Plurality consensus in population protocols with fair scheduler.
Opinions can *only be tested for equality*.

Protocol DMVR.

- Each node initially has a *coin* = its opinion

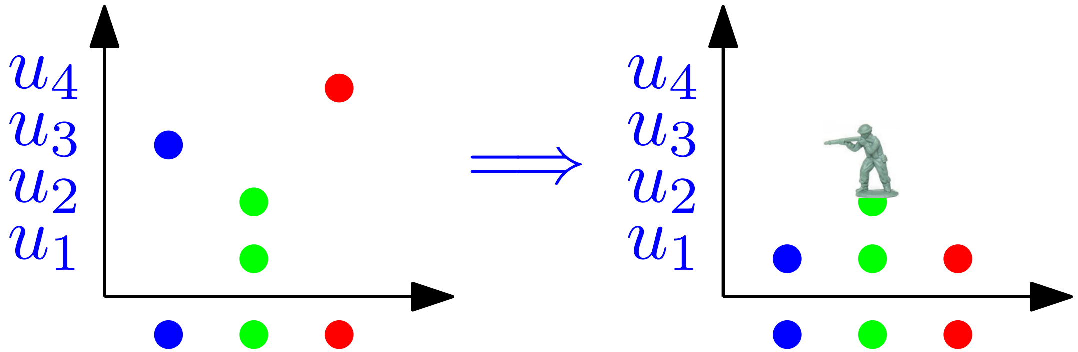


Salehkaleybar et al.'s Conjecture [TSIPN'15]

Problem. Plurality consensus in population protocols with fair scheduler.
Opinions can *only be tested for equality*.

Protocol DMVR.

- Each node initially has a *coin* $=$ its opinion

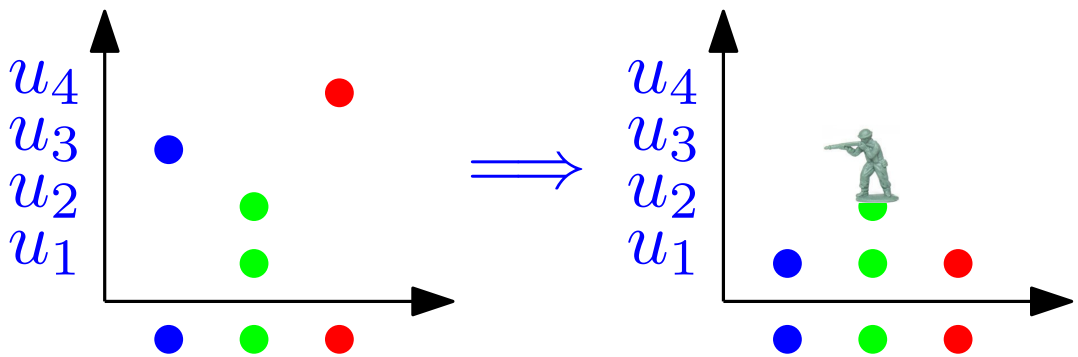


Salehkaleybar et al.'s Conjecture [TSIPN'15]

Problem. Plurality consensus in population protocols with fair scheduler.
Opinions can *only be tested for equality*.

Protocol DMVR.

- Each node initially has a *coin* = its opinion



Coins are accumulated on few nodes

- When (u, v) interact:

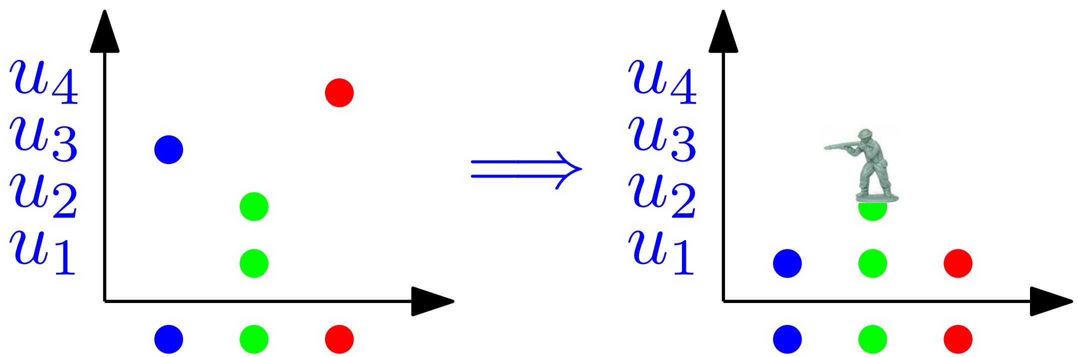
$$\boxed{\begin{array}{l} new\ coins(u) = coins(u) \cap coins(v) \\ new\ coins(v) = coins(u) \cup coins(v) \end{array}}$$

Salehkaleybar et al.'s Conjecture [TSIPN'15]

Problem. Plurality consensus in population protocols with fair scheduler.
Opinions can *only be tested for equality*.

Protocol DMVR.

- Each node initially has a *coin* = its opinion



Coins are accumulated on few nodes

- When (u, v) interact:

$$\boxed{\begin{aligned} \mathit{new\ coins}(u) &= \mathit{coins}(u) \cap \mathit{coins}(v) \\ \mathit{new\ coins}(v) &= \mathit{coins}(u) \cup \mathit{coins}(v) \end{aligned}}$$

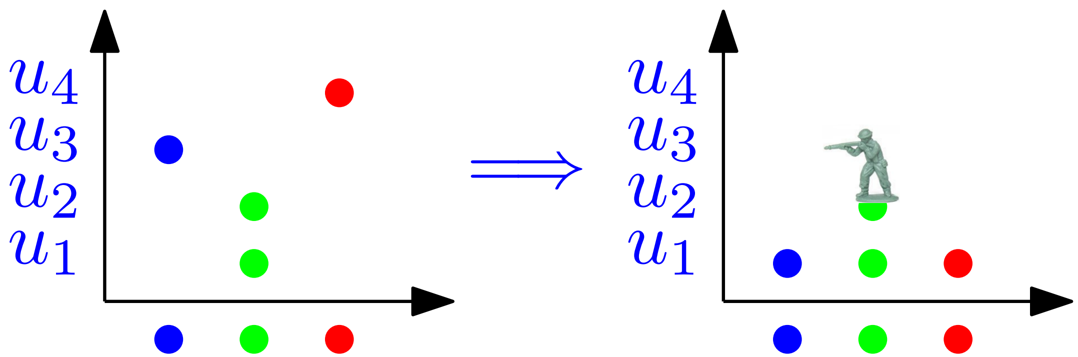
Potential function
$\sum_v |\text{coins}(v)|^2$

Salehkaleybar et al.'s Conjecture [TSIPN'15]

Problem. Plurality consensus in population protocols with fair scheduler.
Opinions can *only be tested for equality*.

Protocol DMVR.

- Each node initially has a *coin* $=$ its opinion



Coins are accumulated on few nodes

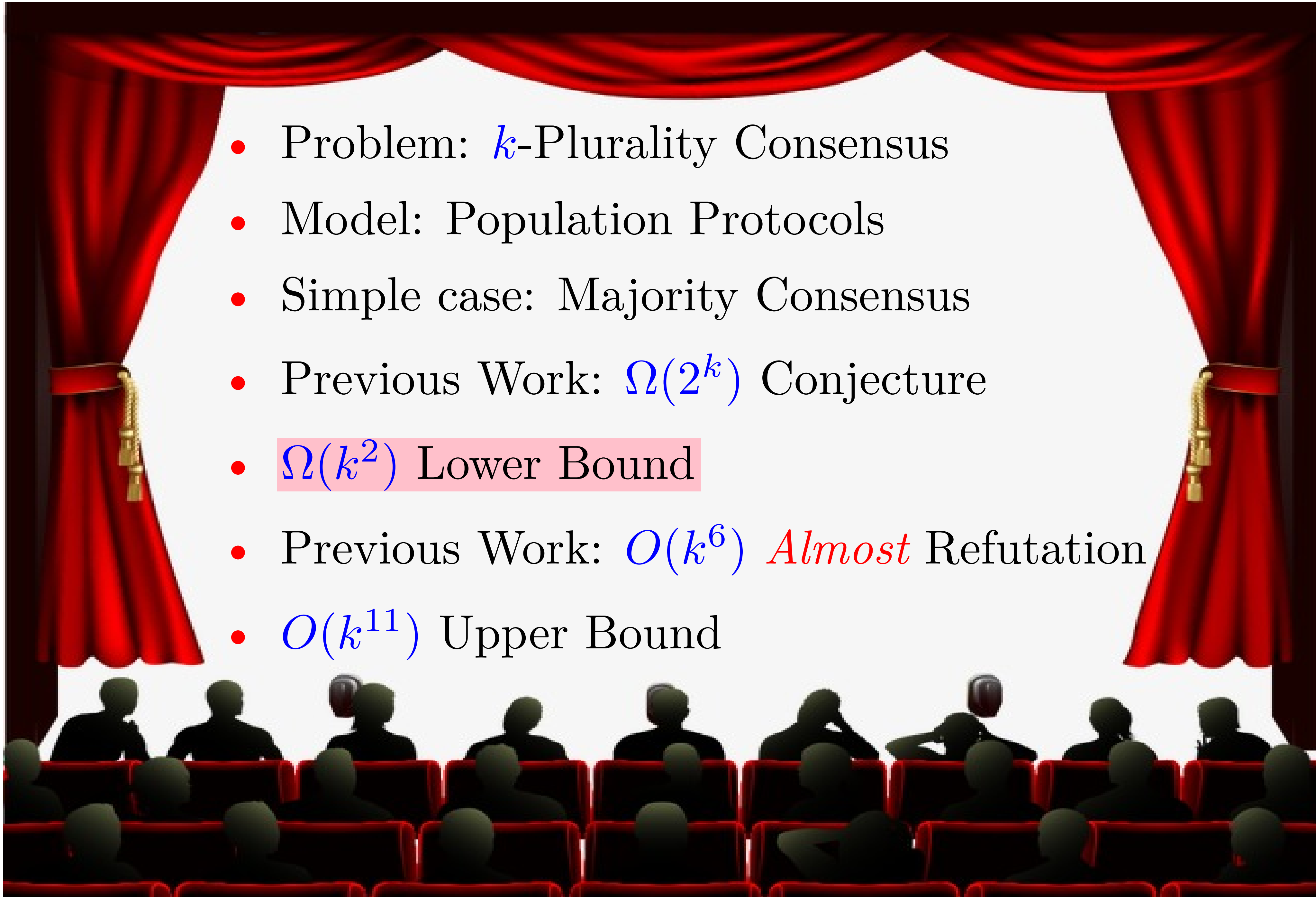
- When (u, v) interact:

$$\boxed{\begin{aligned} new\ coins(u) &= coins(u) \cap coins(v) \\ new\ coins(v) &= coins(u) \cup coins(v) \end{aligned}}$$

Potential function
$\sum_v |\text{coins}(v)|^2$

Conjecture. $O(2^k)$ states are necessary.

Outline



- Problem: k-Plurality Consensus
- Model: Population Protocols
- Simple case: Majority Consensus
- Previous Work: $\Omega(2^k)$ Conjecture
- $\Omega(k^2)$ Lower Bound
- Previous Work: $O(k^6)$ *Almost* Refutation
- $O(k^{11})$ Upper Bound

$\Omega(k^2)$ Lower Bound I

k colors, Σ states.
Protocol P eventually reaches plurality consensus.

$\Omega(k^2)$ Lower Bound I

k colors, Σ states.
Protocol P eventually reaches plurality consensus.

There is output function
$\Phi : \Sigma \to (\text{“}i \text{ is plurality”})_{i \in \{1,\ldots,k\}}$

$\Omega(k^2)$ Lower Bound I

k colors, Σ states.
Protocol P eventually reaches plurality consensus.

There is output function
$\Phi : \Sigma \to (\text{“}i \text{ is plurality”})_{i \in \{1,\dots,k\}}$
$\implies$ there is a color c^* s.t. $|\{\sigma : \Phi(\sigma) = c^*\}| \leq \Sigma/k$

$\Omega(k^2)$ Lower Bound I

k colors, Σ states.
Protocol P eventually reaches plurality consensus.

There is output function
$\Phi : \Sigma \to (\text{“}i \text{ is plurality”})_{i \in \{1,\ldots,k\}}$
$\implies$ there is a color c^* s.t. $|\{\sigma : \Phi(\sigma) = c^*\}| \leq \Sigma/k$

In at most $\approx \left(\frac{2e \cdot x}{\frac{|\Sigma|}{k} - 1}\right)^{\frac{|\Sigma|}{k} - 1}$ config.s all nodes output c^*.

$\Omega(k^2)$ Lower Bound I

k colors, Σ states.
Protocol P eventually reaches plurality consensus.

There is output function
$\Phi : \Sigma \to (\text{“}i \text{ is plurality”})_{i \in \{1,\dots,k\}}$
$\implies$ there is a color c^* s.t. $|\{\sigma : \Phi(\sigma) = c^*\}| \leq \Sigma/k$

In at most $\approx \left(\frac{2e \cdot x}{\frac{|\Sigma|}{k} - 1}\right)^{\frac{|\Sigma|}{k} - 1}$ config.s all nodes output c^*.

There are
$\approx \left(\frac{x-1}{2k-4}\right)^{k-2}$ initial
config.s of the form

x “c^*” | $(x-1)/2$ pairs of nodes with other opinions

$\Omega(k^2)$ Lower Bound I

k colors, Σ states.
Protocol P eventually reaches plurality consensus.

There is output function
$\Phi : \Sigma \to (\text{"}i \text{ is plurality"})_{i \in \{1,\ldots,k\}}$
$\implies$ there is a color c^* s.t. $|\{\sigma : \Phi(\sigma) = c^*\}| \leq \Sigma/k$

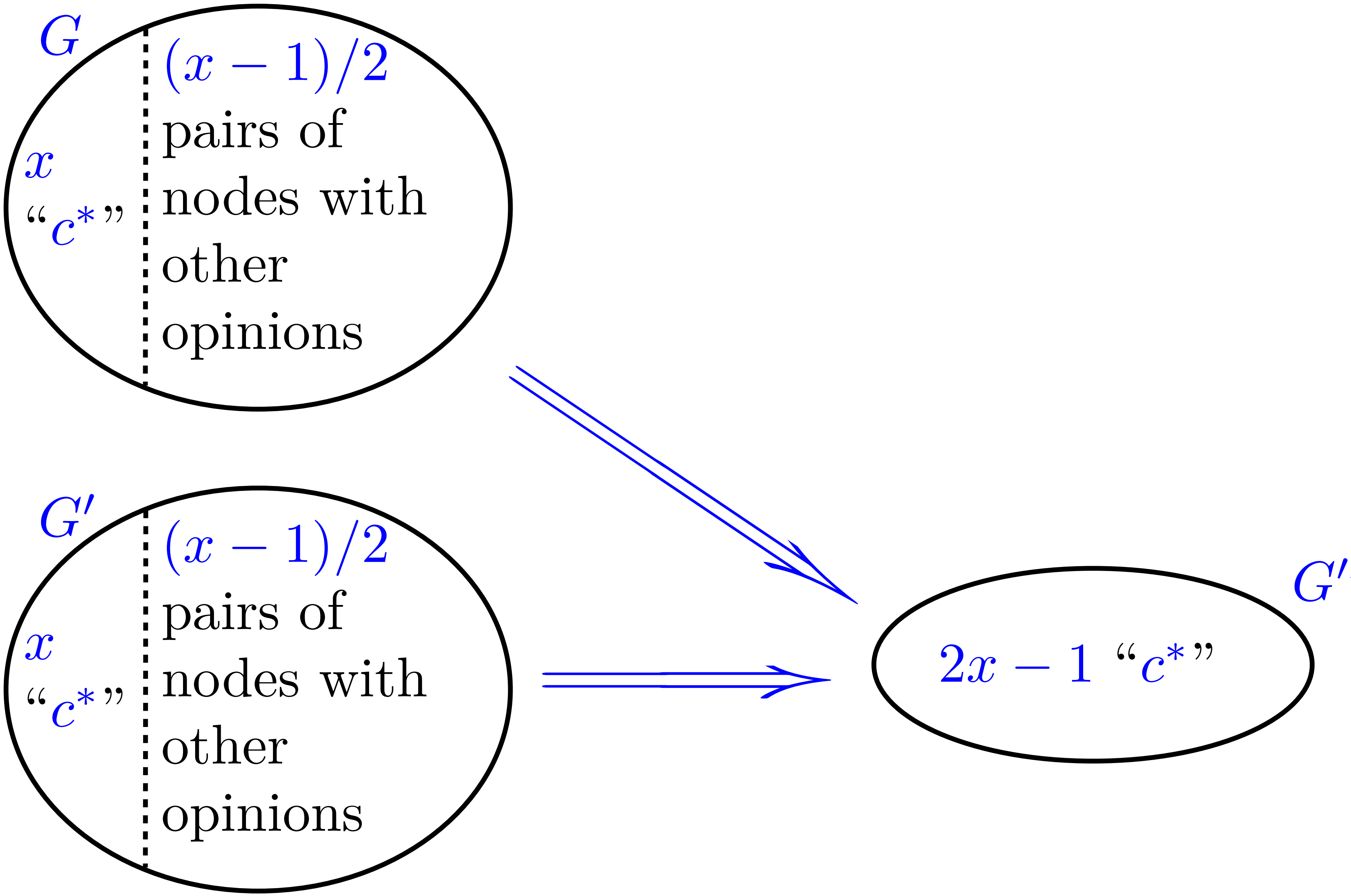
In at most $\approx \left(\frac{2e \cdot x}{\frac{|\Sigma|}{k} - 1}\right)^{\frac{|\Sigma|}{k} - 1}$ config.s all nodes output c^*.

There are
$\approx \left(\frac{x-1}{2k-4}\right)^{k-2}$ initial
config.s of the form

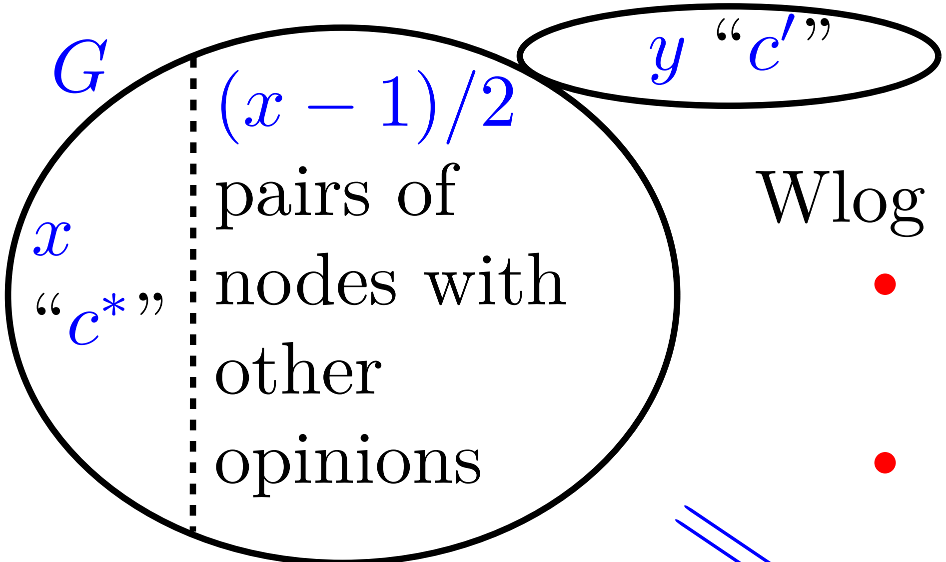
x "c^*" | $(x-1)/2$ pairs of nodes with other opinions

Pigeonhole: if $|\Sigma| < k^2 - k$, 2 config.s G and G' in
converge to identical configurations.

$\Omega(k^2)$ Lower Bound II

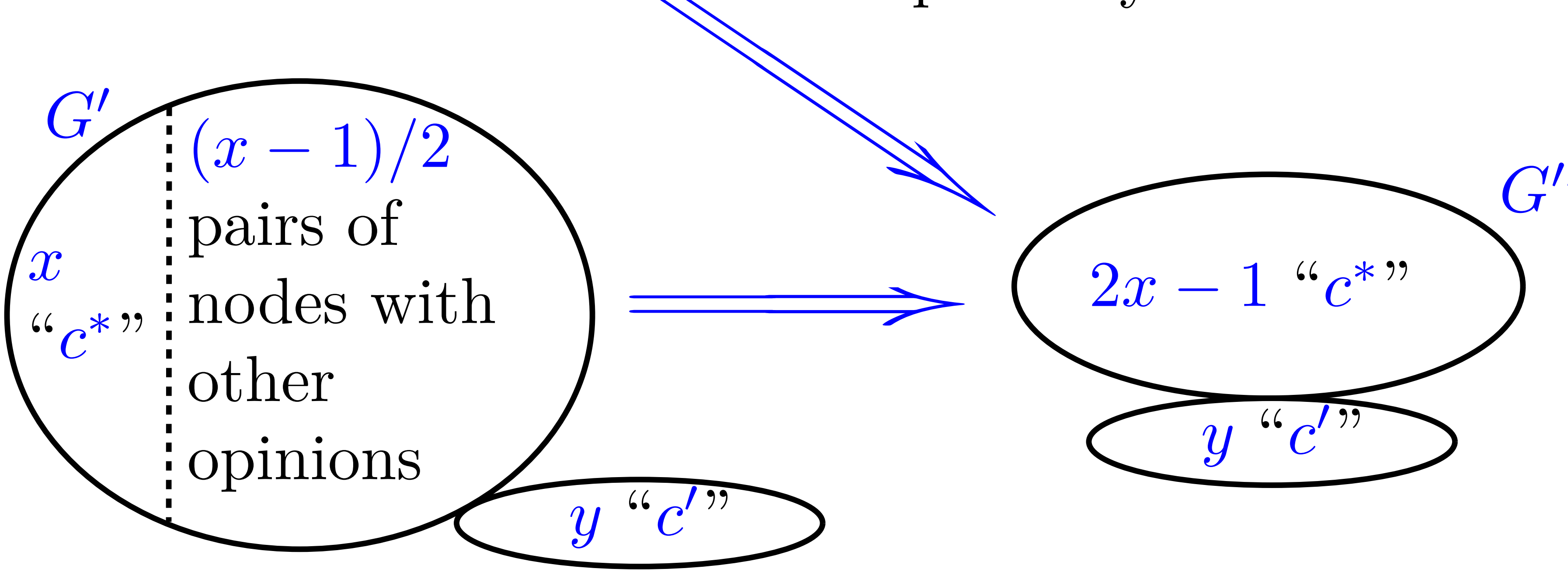


$\Omega(k^2)$ Lower Bound II

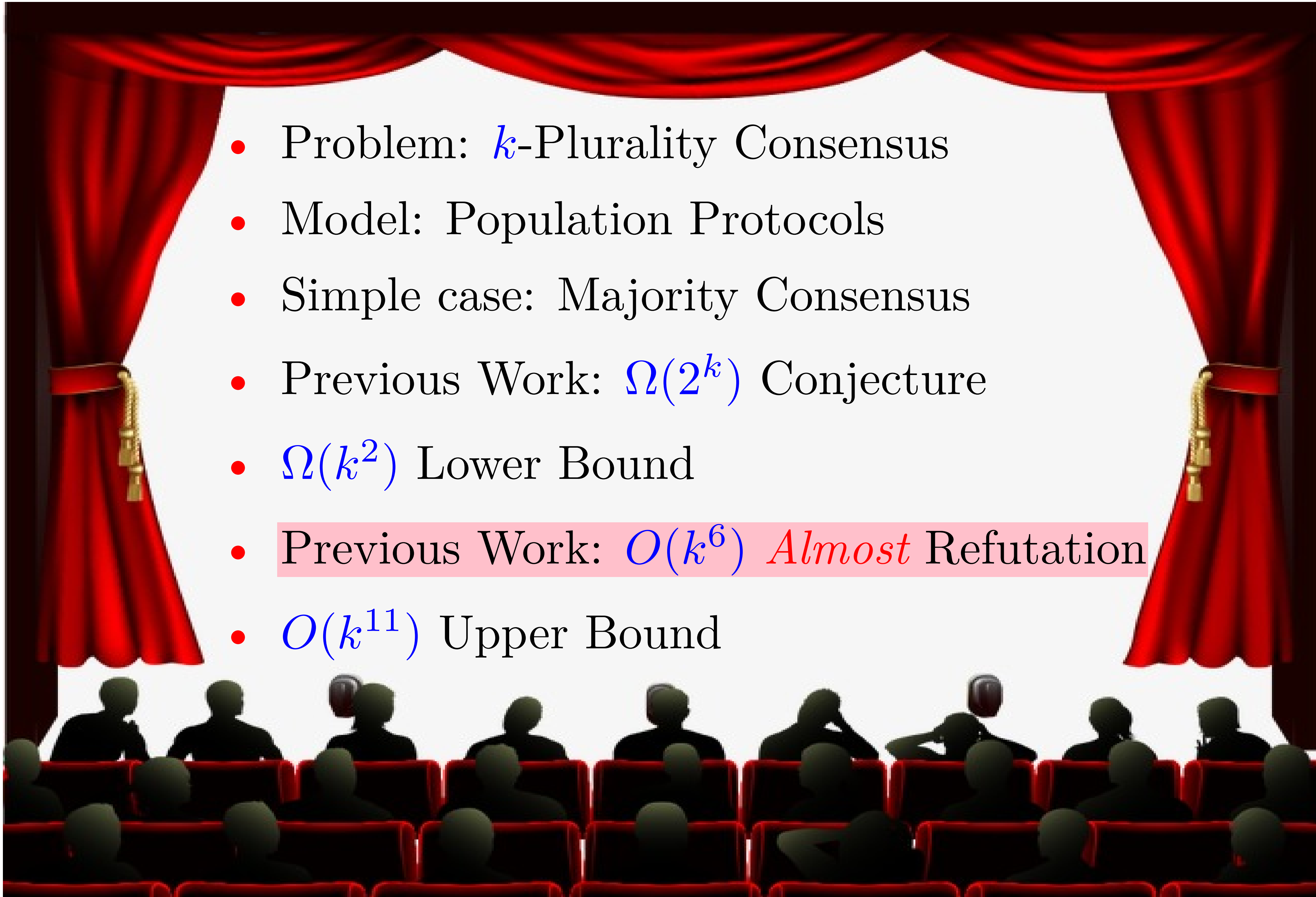


Wlog there is opinion c' and y s.t.

- adding y "c'" to G makes c' plurality in G
- adding y "c' to G' leaves c^* plurality in G'.



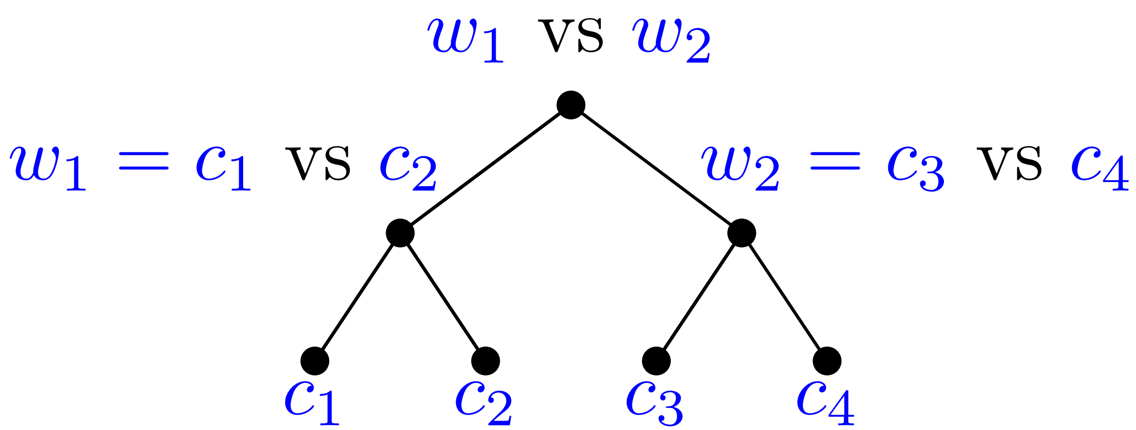
Outline



- Problem: k-Plurality Consensus
- Model: Population Protocols
- Simple case: Majority Consensus
- Previous Work: $\Omega(2^k)$ Conjecture
- $\Omega(k^2)$ Lower Bound
- Previous Work: $O(k^6)$ *Almost* Refutation
- $O(k^{11})$ Upper Bound

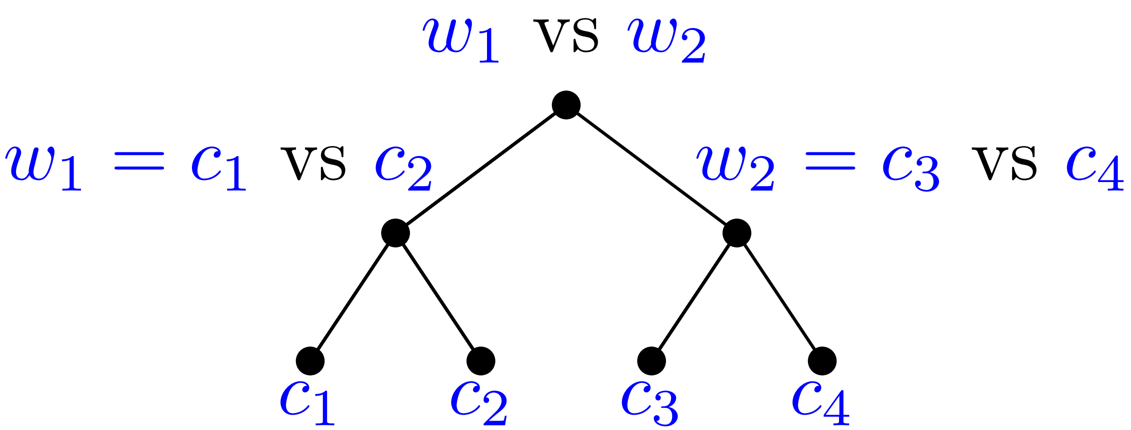
Plurality Consensus via Tournament Tree

Idea. Compute plurality by *majority* tournament.



Plurality Consensus via Tournament Tree

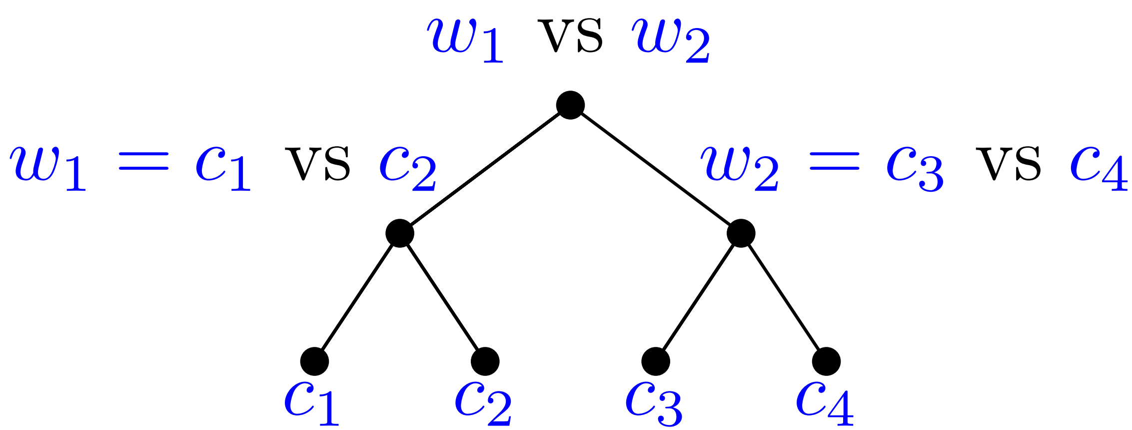
Idea. Compute plurality by *majority* tournament.



Requires agreement on the leaves/labels.

Plurality Consensus via Tournament Tree

Idea. Compute plurality by *majority* tournament.

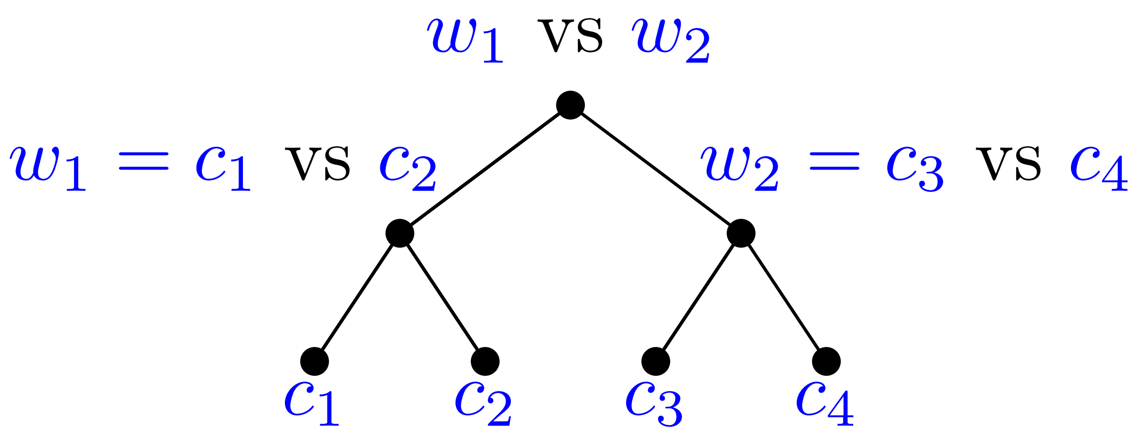


Requires agreement on the leaves/labels.

Problem. Not clear who should play at each match: winner of previous matches can change.

Plurality Consensus via Tournament Tree

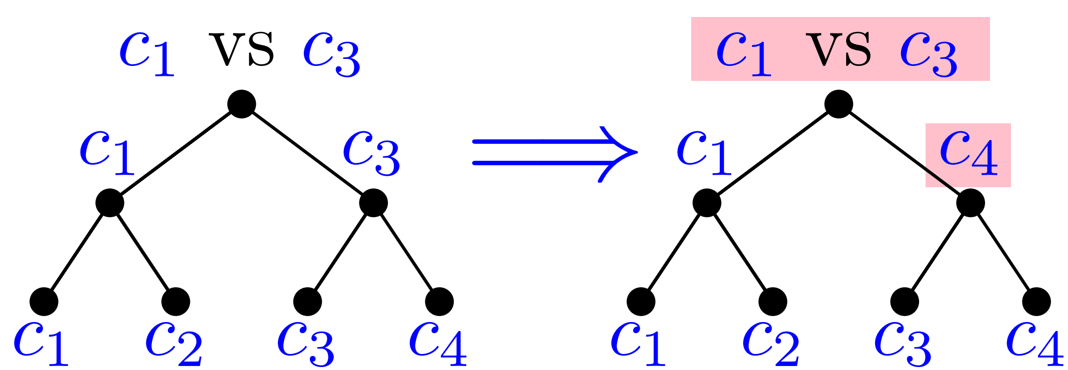
Idea. Compute plurality by *majority* tournament.



Requires agreement on the leaves/labels.

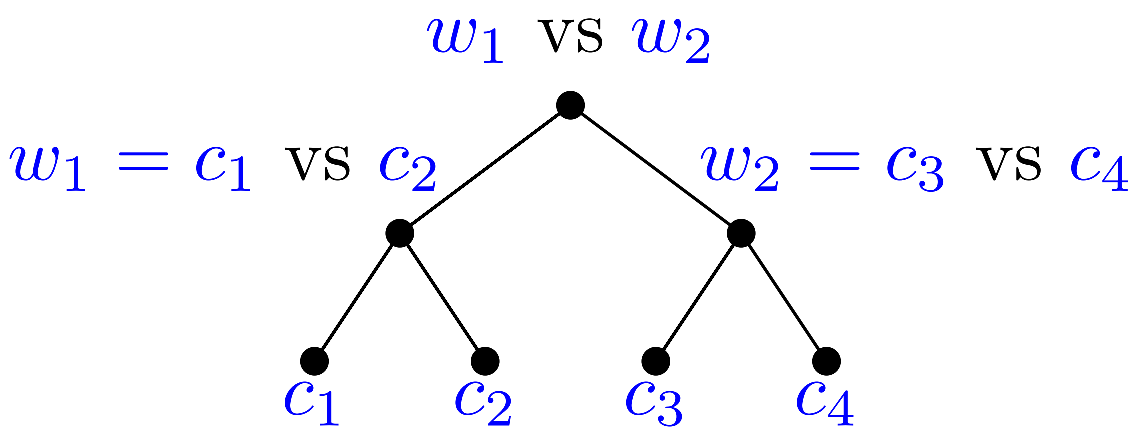
Problem. Not clear who should play at each match: winner of previous matches can change.

c_1 may already have been competing against c_4: it cannot simply start afresh



Plurality Consensus via Tournament Tree

Idea. Compute plurality by *majority* tournament.

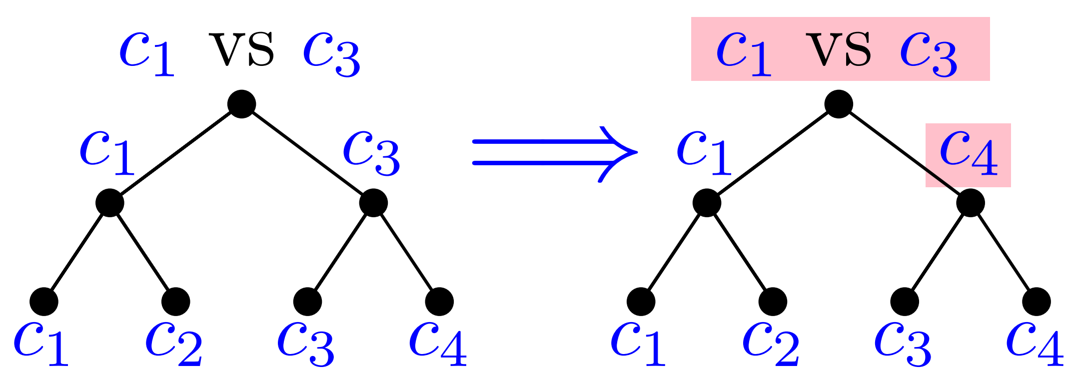


Requires agreement on the leaves/labels.

Problem. Not clear who should play at each match: winner of previous matches can change.

Solved if nodes can change *opinion*.

c_1 may already have been competing against c_4: it cannot simply start afresh



Dynamic Plurality Consensus

[Gasieniec et al. OPODIS'16]

Nodes can *change opinion* during execution.

States and weights

s	$w(s)$
$[-2]$	-2
$[-1]$	-1
$[0], \langle -1 \rangle, \langle 0 \rangle, \langle 1 \rangle$	0
$[1]$	1
$[2]$	2

Updating the state

s_a, $c_a = 1$ changes to $c'_a = -1$	s'_a
$[0], \langle -1 \rangle, \langle 0 \rangle, \langle 1 \rangle$	$[-2]$
$[1]$	$[-1]$
$[2]$	$[0]$
s_a, $c_a = -1$ changes to $c'_a = 1$	s'_a
$[0], \langle -1 \rangle, \langle 0 \rangle, \langle 1 \rangle$	$[2]$
$[-1]$	$[1]$
$[-2]$	$[0]$

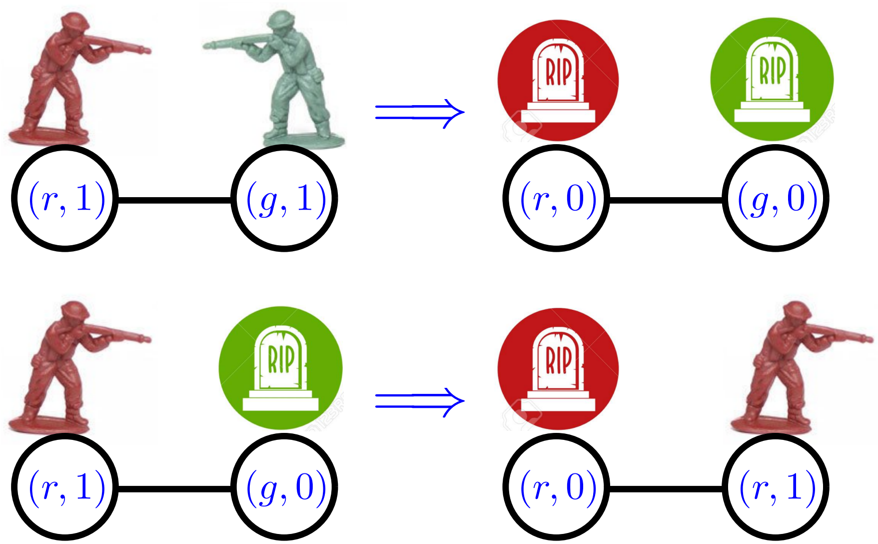
Transitions

$s_a \backslash s_b$	$[-2]$	$[-1]$	$[0]$	$[1]$	$[2]$
$[-2]$	$([-2], [-2])$	$([-2], [-1])$	$([-2], \langle -1 \rangle)$	$([-1], \langle -1 \rangle)$	$([0], [0])$
$[-1]$	$([-1], [-2])$	$([-1], [-1])$	$([-1], \langle -1 \rangle)$	$([0], [0])$	$(\langle 1 \rangle, [1])$
$[0]$	$(\langle -1 \rangle, [-2])$	$(\langle -1 \rangle, [-1])$	$([0], [0])$	$(\langle 1 \rangle, [1])$	$(\langle 1 \rangle, [2])$
$[1]$	$(\langle -1 \rangle, [-1])$	$([0], [0])$	$(\langle 1 \rangle, [1])$	$([1], [1])$	$([1], [2])$
$[2]$	$([0], [0])$	$([1], \langle 1 \rangle)$	$([2], \langle 1 \rangle)$	$([2], [1])$	$([2], [2])$
weak	$(\langle -1 \rangle, [-2])$	$(\langle -1 \rangle, [-1])$	$(\langle 0 \rangle, [0])$	$(\langle 1 \rangle, [1])$	$(\langle 1 \rangle, [2])$

Dynamic Plurality Consensus

[Gasieniec et al. OPODIS'16]

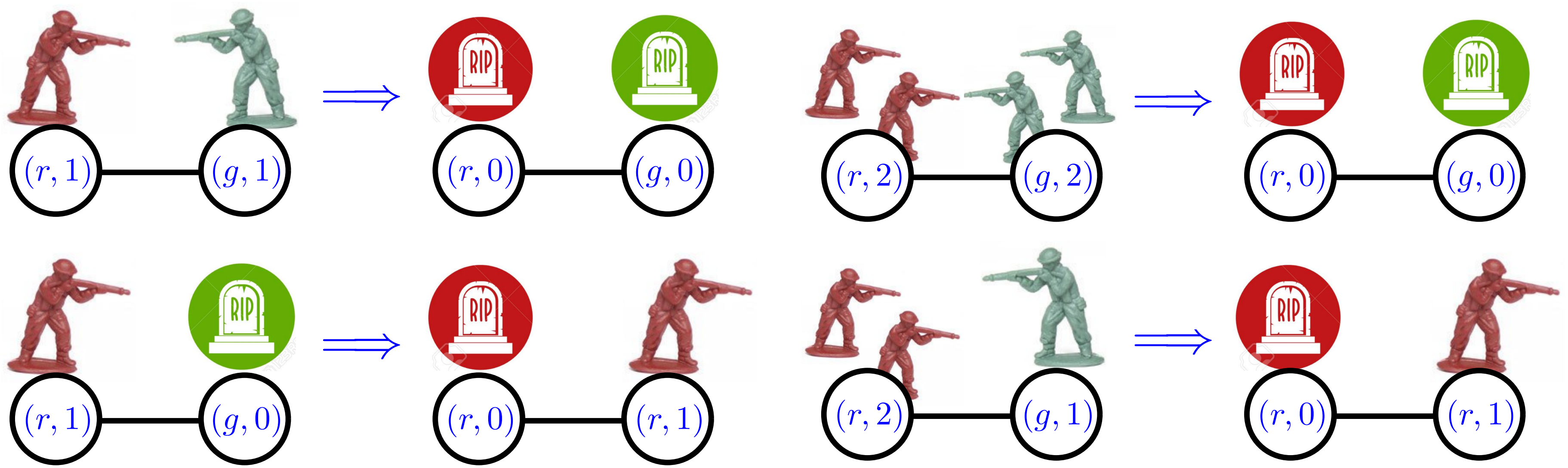
Nodes can *change opinion* during execution.



Dynamic Plurality Consensus

[Gasieniec et al. OPODIS'16]

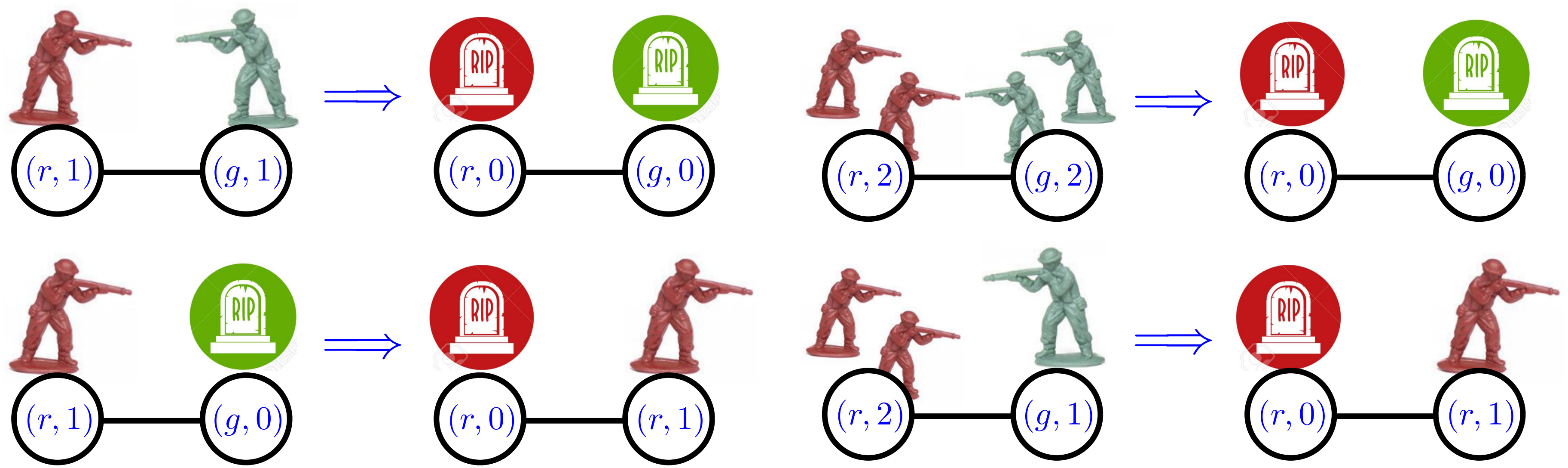
Nodes can *change opinion* during execution.



Dynamic Plurality Consensus

[Gasieniec et al. OPODIS'16]

Nodes can *change opinion* during execution.

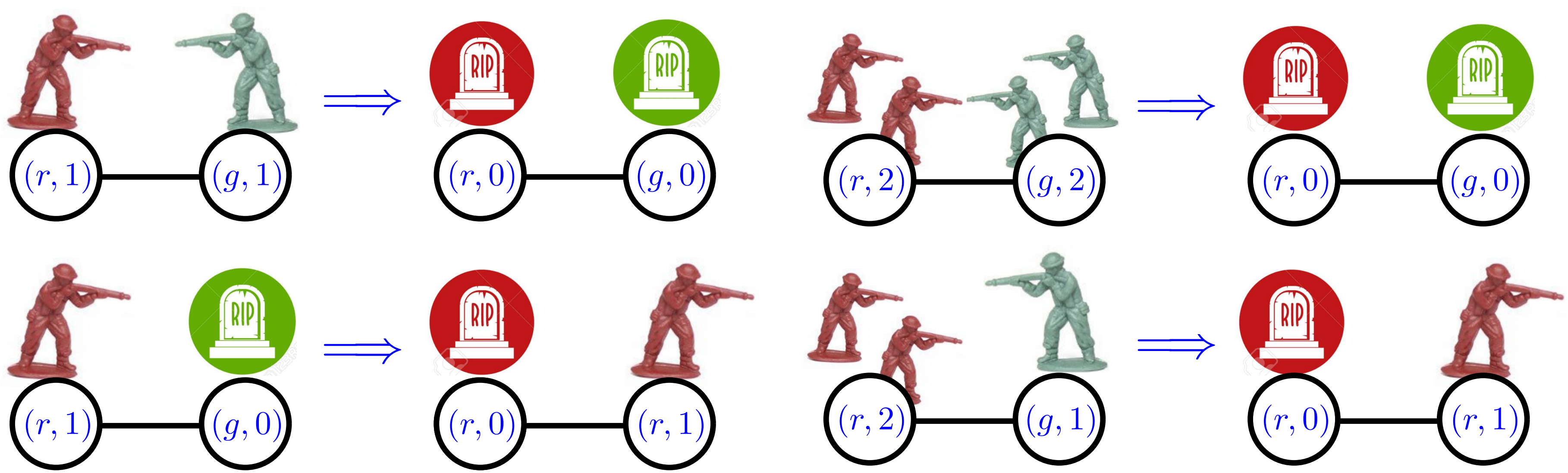


Nodes changing opinion generate *two* soldiers of the new opinion.

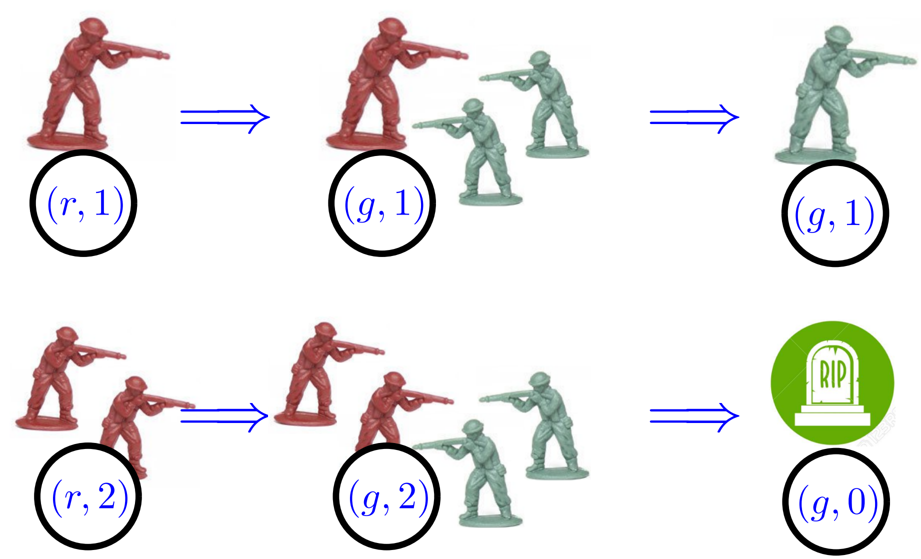
Dynamic Plurality Consensus

[Gasieniec et al. OPODIS'16]

Nodes can *change opinion* during execution.



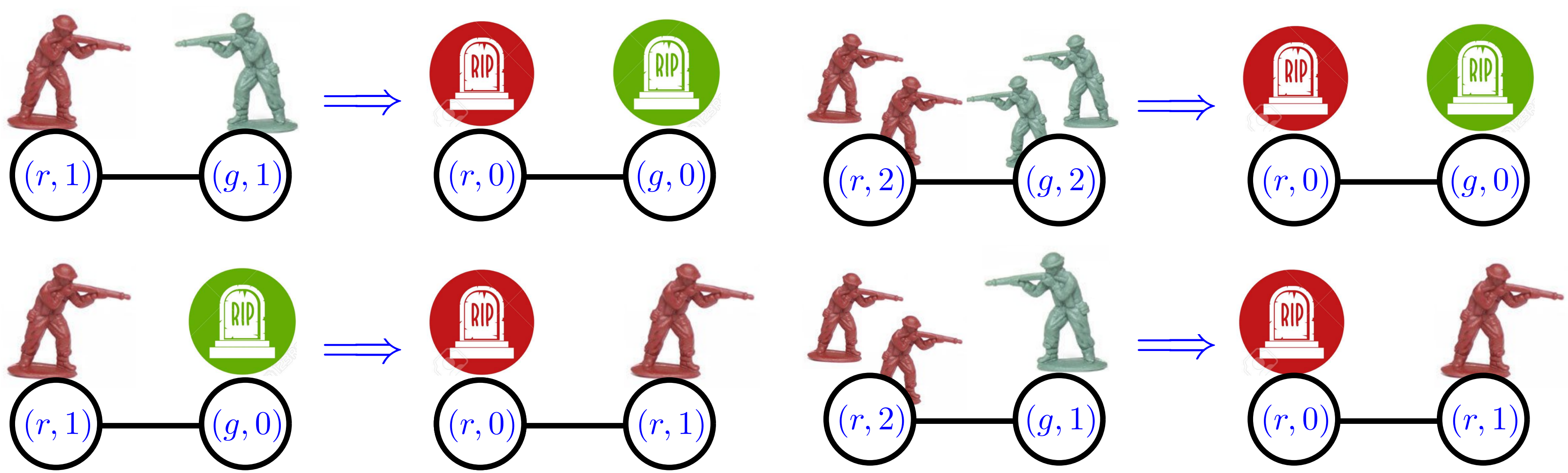
Nodes changing opinion generate *two* soldiers of the new opinion.



Dynamic Plurality Consensus

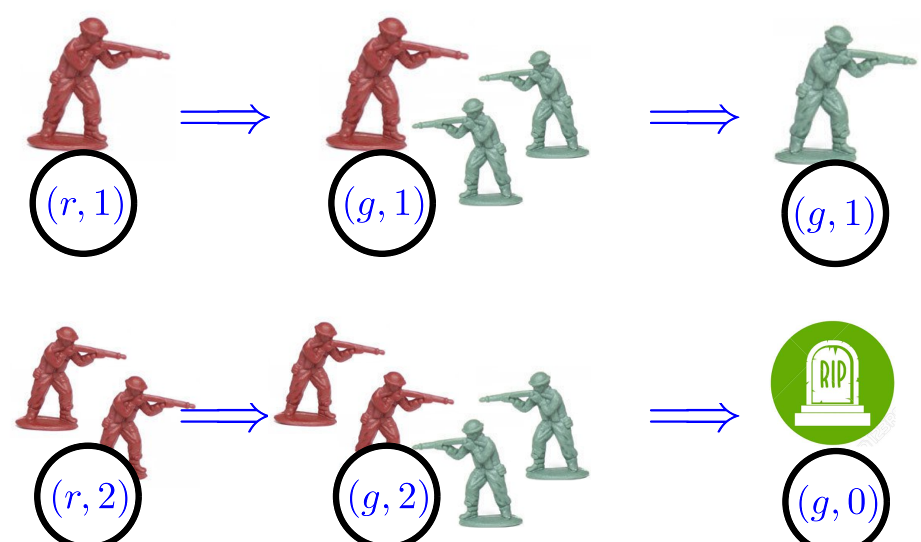
[Gasieniec et al. OPODIS'16]

Nodes can *change opinion* during execution.



Nodes changing opinion generate *two* soldiers of the new opinion.

Balance of opinions
equals
balance of soldiers



Outline

- Problem: k-Plurality Consensus
- Model: Population Protocols
- Simple case: Majority Consensus
- Previous Work: $\Omega(2^k)$ Conjecture
- $\Omega(k^2)$ Lower Bound
- Previous Work: $O(k^6)$ *Almost* Refutation
- $O(k^{11})$ Upper Bound

$O(k^{11})$ Upper Bound (Refunting Conjecture)

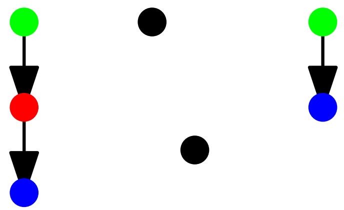
To refute Salehkeleybar's conjecture we provide a protocol that *creates a labeling* and can run *in parallel* with Gasieniec et al.'s.

$O(k^{11})$ Upper Bound (Refunting Conjecture)

To refute Salehkeleybar's conjecture we provide a protocol that *creates a labeling* and can run *in parallel* with Gasieniec et al.'s.

Idea. Have agents arrange opinions in a linked list.

Problem. Multiple lists can appears. How to delete/merge lists?

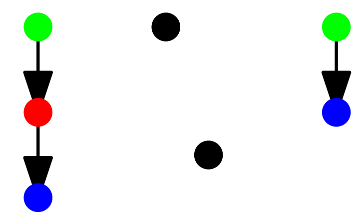


$O(k^{11})$ Upper Bound (Refunting Conjecture)

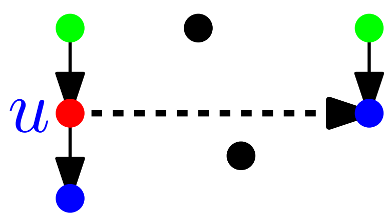
To refute Salehkeleybar's conjecture we provide a protocol that *creates a labeling* and can run *in parallel* with Gasieniec et al.'s.

Idea. Have agents arrange opinions in a linked list.

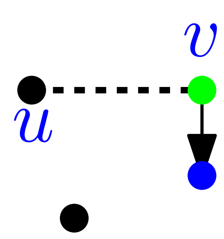
Problem. Multiple lists can appears. How to delete/merge lists?



Ideas. Start deleting from *roots* of lists and append elements by travelling from root to last item.



u will inform parent that list shall be deleted.



u starts by designating v as parent. Eventually u designates as parents v's child, and so on.

Conclusions & Open Problem

Non-ordered self-stabilizing plurality consensus in population protocols with fair scheduler can be solved using $O(k^{11})$ states per agent.

$\Omega(k^2)$ states per agent are necessary.

Conclusions & Open Problem

Non-ordered self-stabilizing plurality consensus in population protocols with fair scheduler can be solved using $O(k^{11})$ states per agent.

$\Omega(k^2)$ states per agent are necessary.

(Ordered) plurality consensus in population protocols with fair scheduler can be solved using $O(k^6)$ states per agent.

Conclusions & Open Problem

Non-ordered self-stabilizing plurality consensus in population protocols with fair scheduler can be solved using $O(k^{11})$ states per agent.

$\Omega(k^2)$ states per agent are necessary.

(Ordered) plurality consensus in population protocols with fair scheduler can be solved using $O(k^6)$ states per agent.

What is the space complexity of plurality consensus in population protocols with fair scheduler?

Project Idea

Simulate the DMVR and Dynamics Majority algorithms on Erdős-Rényi graphs with parameter p, varying the parameter.

- Salehkaleybar, S., A. Sharif-Nassab, and S.J. Golestani. 2015. "Distributed Voting/Ranking with Optimal Number of States per Node." IEEE Transactions on Signal and Information Processing over Networks PP (99): 1–1. https://doi.org/10.1109/TSIPN.2015.2477777.
- Gasieniec, Leszek, David Hamilton, Russell Martin, Paul G. Spirakis, and Grzegorz Stachowiak. 2017. "Deterministic Population Protocols for Exact Majority and Plurality." In 20th International Conference on Principles of Distributed Systems (OPODIS 2016), 70:14:1–14:14. Leibniz International Proceedings in Informatics (LIPIcs). https://doi.org/10.4230/LIPIcs.OPODIS.2016.14.

Simulations should be performed using open-source software with some effort to make them efficient (e.g. coded in Python using Numpy), and the source code should be made publicly available (e.g. on Gitlab) and GPL licensed.