How Many Cooks Spoil the Soup?

Othon Michail Paul G. Spirakis

Department of Computer Science, University of Liverpool, UK
Computer Technology Institute & Press "Diophantus" (CTI)

23rd International Colloquium on Structural Information and Communication Complexity (SIROCCO)
July 19-21, 2016
Helsinki, Finland

Leader Election

- A task of outstanding importance for distributed algorithms
- Typical approach to solve a higher-level task A:
  1. Devise an algorithm that elects a leader
  2. Devise an algorithm for A that assumes a pre-elected leader
  3. Compose the two algorithms
- Steps 1 and 3 usually enclose the full difficulty of task A

Question: *Can we solve A without ever electing a leader?*

Two Examples

1. Population Protocols [AADFP06]
   - Compute the semilinear predicates
   - The generic protocol elects a unique leader in every execution
   - All known generic constructions

     "fundamentally rely on the election of a single leader node, which coordinates phases of computation" [AG15]
2. Worst-case Dynamic Networks [KLO10]
   - k-token dissemination in $O(nk)$ rounds with $O(\log n)$ bits/message
   - The algorithm elects a leader in every execution
   - No algorithm is known to avoid this

Why?

- Curiosity: Is it really necessary?
- Fault-tolerance: A unique leader's crash can be fatal
- Parallelism: Symmetry-breaking and "centralized" coordination usually cost in time

Generalized Question: *Can we solve A without ever having fewer than k processes in a given "role"?*

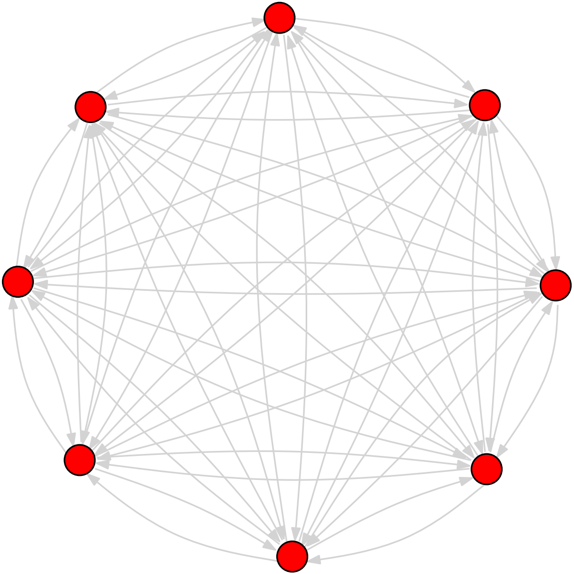
What is a "Role"?

- Meaningful definitions heavily depend on the model/application
- A leader role is typically the value of a local *leader* variable
- Could be defined as the complete local history of a process
- Or in terms of the external interface of a process
- In population protocols can be simply defined as the local state
  - u, v have the same role at a given time t iff, at that time, their local states are the same
  - makes them a good candidate to start this study

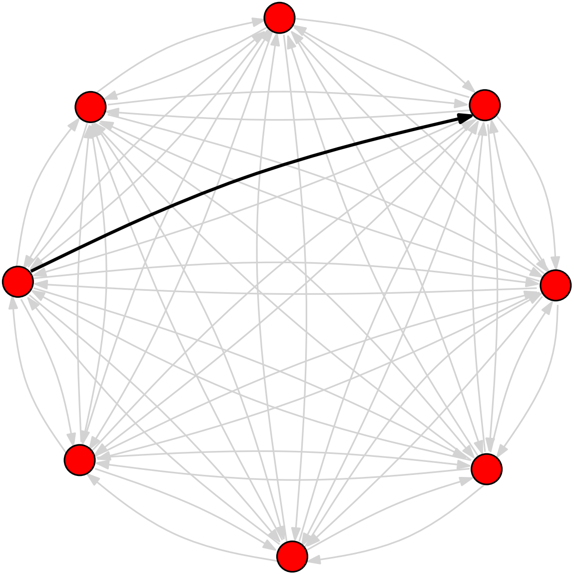
Difficulties:

- There are events controlled by the scheduler
  - even if the protocol has no inherent mechanism of breaking symmetry, the scheduler can always force it
  - we want to isolate the symmetry that is only due to the protocol
  - inherent symmetry vs. observed symmetry
- The sequential scheduler is problematic
  - $(r, r) \to (g, g)$, even rs initially
  - If a single interaction occurs, the new configuration has only 2 gs; symmetry breaking $= n - 2$
  - On the other hand, a perfect matching converts all rs to gs in one step; symmetry breaking $= 0$

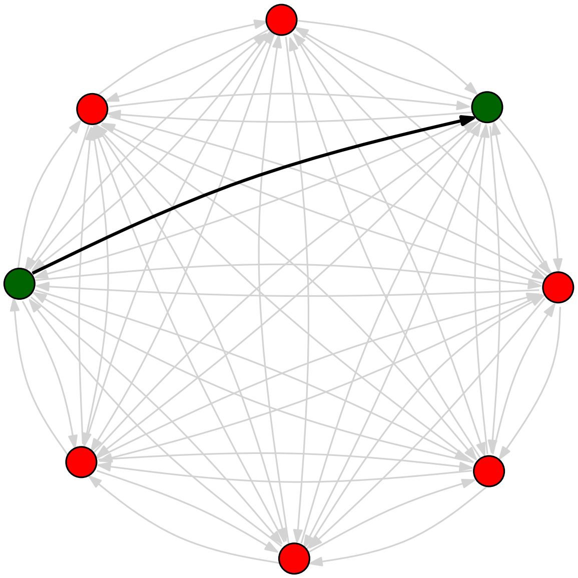
Sequential Scheduler



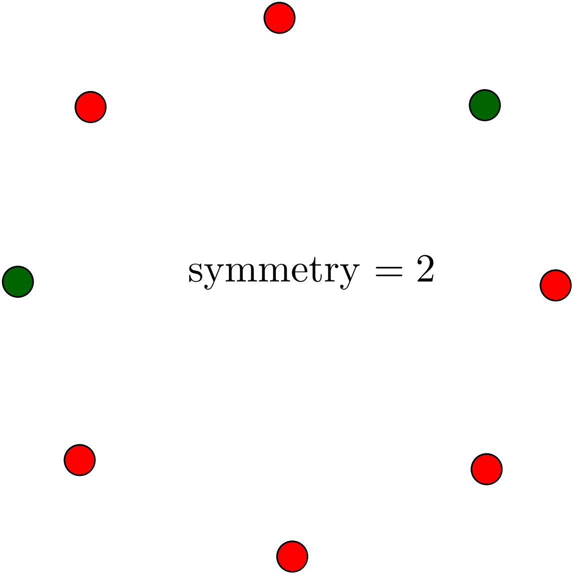
Sequential Scheduler

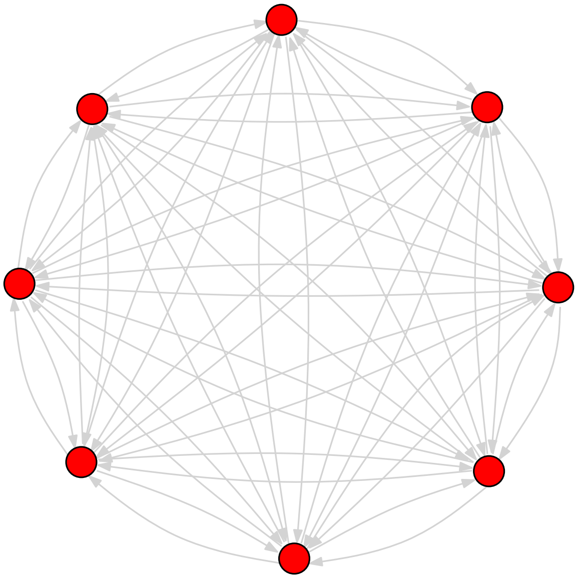


Sequential Scheduler

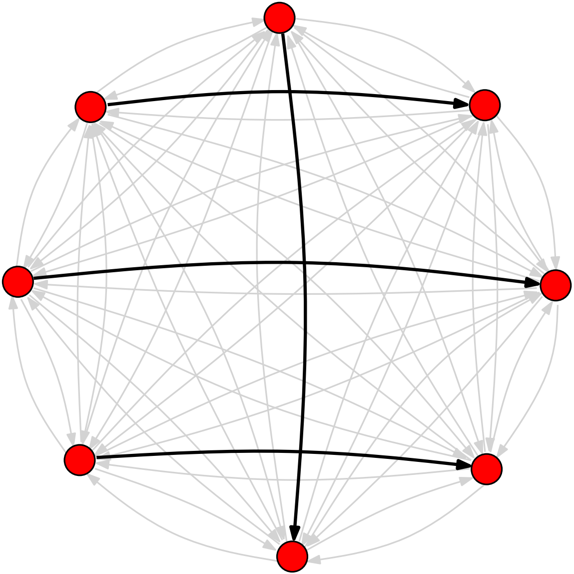


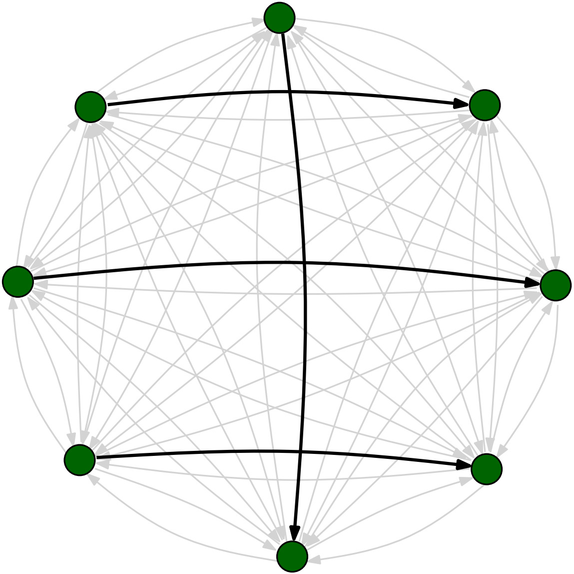
Sequential Scheduler

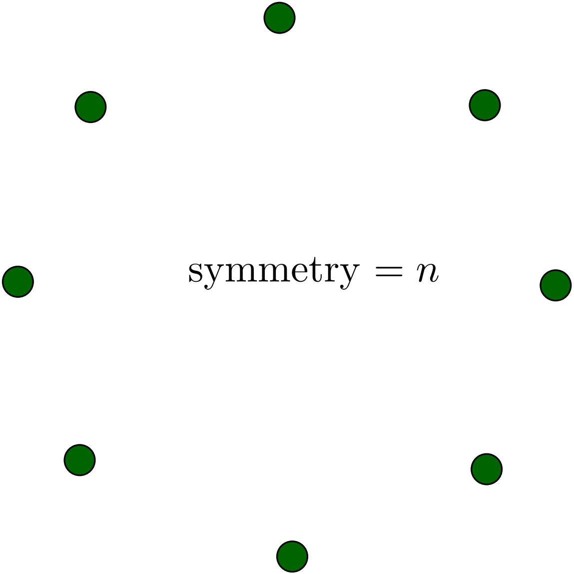




Parallel Scheduler






$\text{symmetry} = n$

Our Approach

Goal: Define a measure of the inherent symmetry of a population protocol

- Schedulers that can be maximally parallel
  - May select from a single interaction to a maximum matching per step
- To isolate the inherent symmetry
  - Focus on schedules that maximize symmetry for the given protocol
  - Introduce as much symmetry as possible to observe the maximum symmetry breaking that the protocol has to perform
  - Does not affect correctness: still under all fair schedules

Overview

- A new measure of the inherent symmetry of a population protocol
- Main positive result (partial characterization):
  - A wide subclass of semilinear predicates can be computed with symmetry $\Theta(N_{min})$, which is asymptotically optimal
  - N_{min}: minimum multiplicity of a state in the initial configuration
    - i.e., the initial symmetry
- Strong negative result:
  - The symmetry of any protocol that stably computes parity, is upper bounded by an integer depending only on the size of the protocol

- The role of symmetry in static anonymous systems has been deeply investigated [An80,YK96,Kr97,FMS98]
- This is not true for static systems with UIDs and dynamic systems
- Homonyms: restricted type of symmetry in systems with IDs [DFGKRT11]
- Systems not suffering from a necessity for global symmetry breaking:
  - Shared Memory with Atomic Snapshots, Quorums, LOCAL model
- In population protocols, avoiding to ever elect a unique leader has not been followed before
  - Only the common question of dropping a pre-elected leader

1. X and Y: finite input and output alphabets
2. Q: finite set of states
3. $I\colon X \to Q$: input function
4. $O\colon Q \to Y$: output function
5. $\delta\colon Q \times Q \to Q \times Q$: transition function

If $\delta(p,q) = (p',q')$, we write $(p,q) \to (p',q')$

Definition

A predicate $p\colon X^* \to \{0,1\}$ is stably computable if there exists a protocol s.t. for all $x \in X^*$, any fair execution beginning from $c_0 = I(x)$ reaches an output stable configuration c_s in which each node outputs $p(x)$.

Symmetry Formally

- symmetry of configuration c: $s(c) = \min_{q \in Q\,:\,c[q] \geq 1}\{c[q]\}$
  - e.g. if $c = (0, 4, 12, 0, 52)$ then $s(c) = 4$
- $\Gamma(c_0)$: all fair executions of $\mathcal{A}$ that begin from c_0, up to stability
- symmetry of $\mathcal{A}$ on $\alpha \in \Gamma(c_0)$: $s(\mathcal{A}, \alpha) = \min_{c \in \alpha}\{s(c)\}$

Definition

Define the symmetry of $\mathcal{A}$ on c_0 as $s(\mathcal{A}, c_0) = \max_{\alpha \in \Gamma(c_0)}\{s(\mathcal{A}, \alpha)\}$.

Remark

To estimate the inherent symmetry of $\mathcal{A}$ on a c_0, execute $\mathcal{A}$ against an imaginary symmetry maximizing scheduler.

Symmetry Formally

- $\mathcal{C}(N_{min})$: all initial configurations c_0 s.t. $s(c_0) = N_{min}$

Definition

Define $\mathcal{A}$'s symmetry on $\mathcal{C}(N_{min})$ as $s(\mathcal{A}, N_{min}) = \min_{c_0 \in \mathcal{C}(N_{min})}\{s(\mathcal{A}, c_0)\}$.

- min-max-min problem
- symmetry breaking $b(\mathcal{A}, N_{min}) = N_{min} - s(\mathcal{A}, N_{min})$
- To show that $\mathcal{A}$ is $\geq g(N_{min})$ symmetric asymptotically
  - $\forall c_0 \in \mathcal{C}(N_{min})$ $\exists$ an execution on c_0 that drops the initial symmetry by at most $N_{min} - g(N_{min})$
  - or at all, if $g(N_{min}) = N_{min}$
- To show that $\mathcal{A}$ is $\leq g(N_{min})$ symmetric
  - a symmetry breaking $\geq N_{min} - g(N_{min})$ on infinitely many N_{min}

Interpretation

- If we establish that a predicate p is $\geq g(N_{min})$ symmetric
  - $\exists$ protocol $\mathcal{A}$ stably computing p without an inherent mechanism of dropping symmetry more than $N_{min} - g(N_{min})$
  - e.g. if $N_{min} = n$ and $g(N_{min}) = \log n$, $\mathcal{A}$ does not inherently try to break symmetry more than $n - \log n$
- If we establish that a predicate p is $\leq g(N_{min})$ symmetric
  - Any protocol $\mathcal{A}$ that stably computes p has to drop symmetry by at least $N_{min} - g(N_{min})$ in every execution
  - e.g. if $g(N_{min}) = 1$, $\mathcal{A}$ elects a unique leader in every execution
- This definition leads to very strong impossibility results
  - upper bounds are also upper bounds on the observed symmetry
  - hold under any scheduler

An Example: Count-to-x

- $X = \{0, 1\}$, $Q = \{q_0, q_1, q_2 \ldots, q_x\}$,
- $I(0) = q_0$ and $I(1) = q_1$,
- $O(q_x) = 1$ and $O(q) = 0$, for $q \in Q \backslash \{q_x\}$, and
- δ:

$$\begin{aligned} (q_i, q_j) &\to (q_{i+j}, q_0), \text{ if } i + j < x \\ &\to (q_x, q_x), \text{ otherwise} \end{aligned}$$

Proposition

The symmetry of Protocol Count-to-x, for any $x = O(1)$, is at least $(2/3)\lfloor N_{min}/x \rfloor - (x-1)/3$, when $x \geq 2$, and N_{min}, when $x = 1$; i.e., it is $\Theta(N_{min})$ for any $x = O(1)$.

Proof Idea

- N_1: #nodes initially in q_1
- The scheduler forms $\lfloor N_1/x \rfloor$ groups of x q_1s each, and $r \leq x-1$ q_1s residue
- Sequential gathering to one of the nodes in each group
  - goes through states $q_1, q_2, \ldots, q_{x-1}$
  - in parallel to all groups, so cardinalities of states are always $\geq \lfloor N_1/x \rfloor$
- Cannot pick a perfect bipartite matching between q_1s and q_{x-1}s to obtain alarm states
  - could leave the symmetry-breaking residue of q_1s
- Instead, match in one step as many as possible so that, after the corresponding transitions, $N_x(t') \geq N_1(t')$

Proof Idea

- If we match approx. $1/3$ of the (q_1, q_{x-1}) pairs, then we will have as many q_x as we need in order to eliminate all q_1s in one step and all remaining q_{x-1}s in another step.
- The min symmetry in the whole course of this schedule is

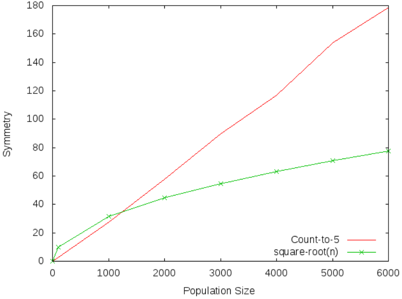
$$\begin{aligned} N_{x-1}(t') &= \lfloor N_1/x \rfloor - y = \lfloor N_1/x \rfloor - \frac{\lfloor N_1/x \rfloor + r}{3} \\ &= \frac{2}{3}\lfloor N_1/x \rfloor - \frac{r}{3} \geq \frac{2}{3}\lfloor N_1/x \rfloor - \frac{x-1}{3}. \end{aligned}$$

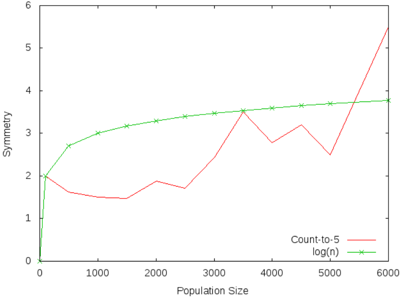
- Similar strategy if there are also q_0s initially
- In all cases, symmetry
  - $\geq (2/3)\lfloor N_{min}/x \rfloor + (x-1)/3 = \Theta(N_{min})$, for $x \geq 2$, and
  - $= N_{min}$, for $x = 1$

$\square$

Comparison to Observed Symmetry

- random parallel schedulers
  - e.g. in every step a maximum matching uniformly at random
- *"What is the average symmetry achieved by a protocol under such a scheduler?"*
- The expected observed symmetry of *Count-to-5*
  - if counted until q_5 becomes absolute majority, seems to grow faster than $\sqrt{n}$
  - if counted up to stability, seems to grow as fast as $\log n$





Theorem

Any predicate of the form $\sum_{i\in[k]} a_i N_i \geq c$, for integer constants $k \geq 1$, $a_i \geq 1$, and $c \geq 0$, can be computed with symmetry more than $\lfloor N_{min}/(c/\sum_{j\in L} a_j + 2)\rfloor - 2 = \Theta(N_{min})$.

Protocol Positive-Linear-Combination

$Q = \{q_0, q_1, q_2, \ldots, q_c\}$

$I(\sigma_i) = q_{a_i}$, for all $\sigma_i \in X$

$O(q_c) = 1$ and $O(q) = 0$, for all $q \in Q\backslash\{q_c\}$

δ:

$$\begin{aligned}(q_i, q_j) &\rightarrow (q_{i+j}, q_0), \text{ if } i + j < c\\ &\rightarrow (q_c, q_c), \text{ otherwise}\end{aligned}$$

Output-stable States

Theorem

Let $\mathcal{A}$ be a protocol with a reachable disseminating state q and let $\mathcal{C}_0^d$ be the subset of its initial configurations that may produce q. Then the symmetry of $\mathcal{A}$ on $\mathcal{C}_0^d$ is $\Theta(N_{min})$.

- i.e., disseminating states can be exploited for maximum symmetry
- immediately applies to single-signed linear combinations
  - passing a threshold results in the appearance of a disseminating state
- does not apply to linear combinations with mixed signs:

Proposition

Let p be a predicate of the form $\sum_{i\in[k]} a_i N_i \geq c$ such that at least two a_is have opposite signs. Then there is no protocol, having a reachable output-stable state, that stably computes p.

Harder Predicates

- Predicates that do not allow for output-stable states
  - mixed-signed linear combinations, like majority
  - modulo predicates, like parity
  - not captured by the previous characterization
- The majority predicate $N_a - N_b > 0$ can be computed with symmetry $\min\{N_{min}, |N_a - N_b|\}$
  - generalizes to any predicate with mixed signs
- For every constant $k \geq 1$, majority can be computed with symmetry k

Parity Cannot be Computed with High Symmetry

- Parity: all nodes start from q_1, true iff the number of nodes is odd

Theorem

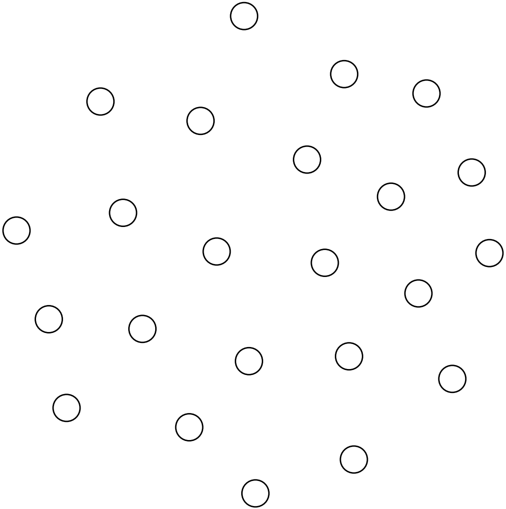
Let $\mathcal{A}$ be a protocol with set of states Q, that solves parity. Then the symmetry of $\mathcal{A}$ is less than $2^{|Q|-1}$.

Proof

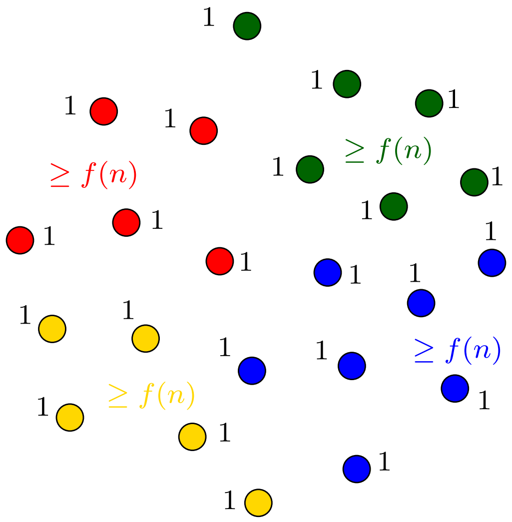
- *Assume $\mathcal{A}$ solves it with symmetry $f(n) \geq 2^{|Q|-1}$*
- *Take any initial C_n for any sufficiently large odd n*
- *$\exists$ execution α on C_n that reaches stability without ever dropping the minimum cardinality of an existing state below $f(n)$*
- *C_{stable}: the first output-stable configuration of α*
  - *all nodes give output 1 and output 0 cannot be produced*
  - *every $q \in Q$ that appears in C_{stable} has multiplicity $C_{stable}[q] \geq f(n)$*

Parity Cannot be Computed with High Symmetry

- *Consider C_{2n}, i.e., the unique initial configuration on 2n nodes*
- *even n, thus parity is false*
- *Partition C_{2n} into two parts of size n*
- *Finite prefix β of a fair execution on C_{2n}:*
  - *simulate α in each part, until it reaches C_{stable}*
  - *$2C_{stable}$: consists precisely of two copies of C_{stable}*
- *Any fair execution on $2C_{stable}$ must produce a state q_0 with output 0*
- *q_0 must also be reachable from a sub-configuration C_{small} of $2C_{stable}$ of size at most $2^{|Q|-1}$ (by a proposition)*
- *But C_{small} is also a sub-configuration of C_{stable}*
  - *So, q_0 can also be produced by C_{stable}*
  - *Contradicts the fact that C_{stable} is output-stable with output 1* $\square$

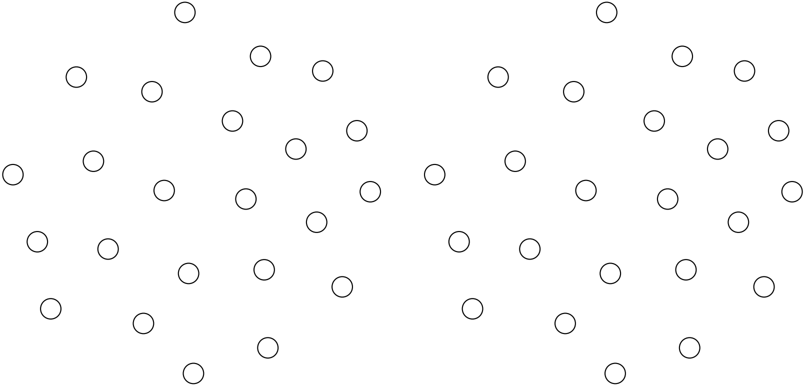


C_n: n odd



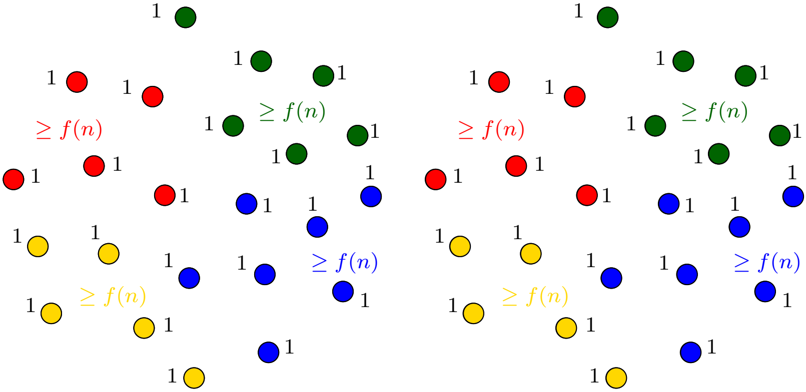
C_{stable}: all give output 1

Parity Cannot be Computed with High Symmetry



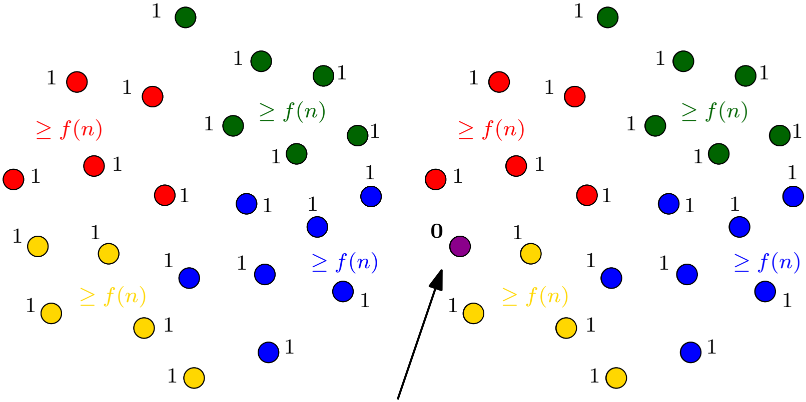
C_{2n}: $2n$ even

Parity Cannot be Computed with High Symmetry

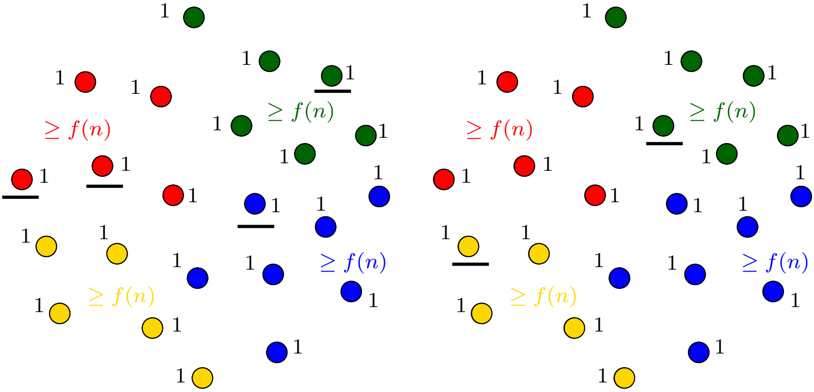


$2C_{stable}$: unstable

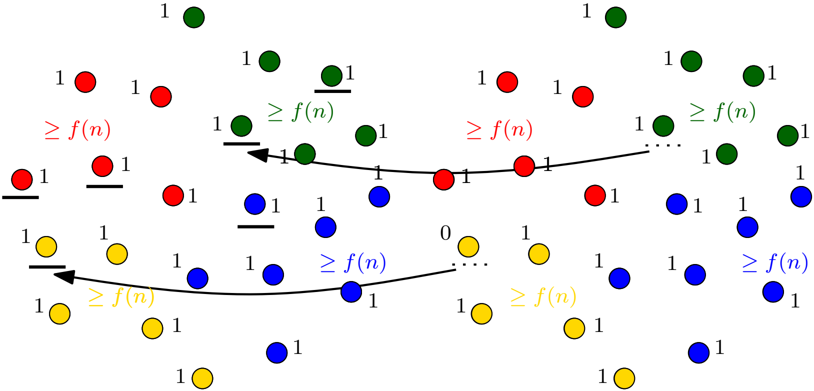
Parity Cannot be Computed with High Symmetry



after a while, an output 0 appears

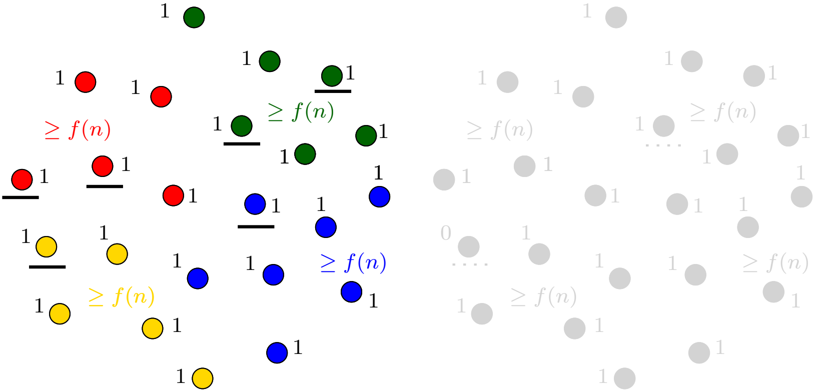


due to interactions inside a $C_{small} \subset 2C_{stable}$



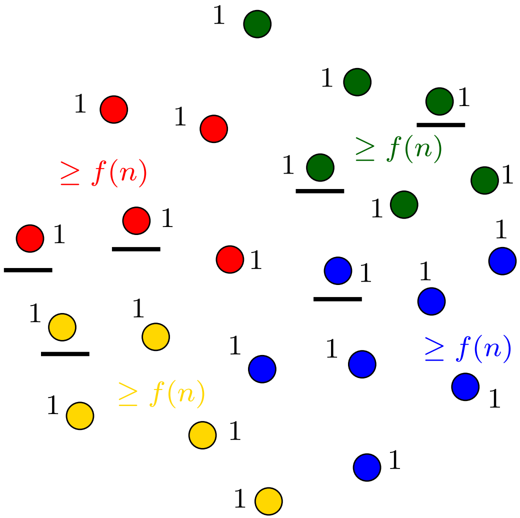
but also $C_{small} \subset C_{stable}$

Parity Cannot be Computed with High Symmetry

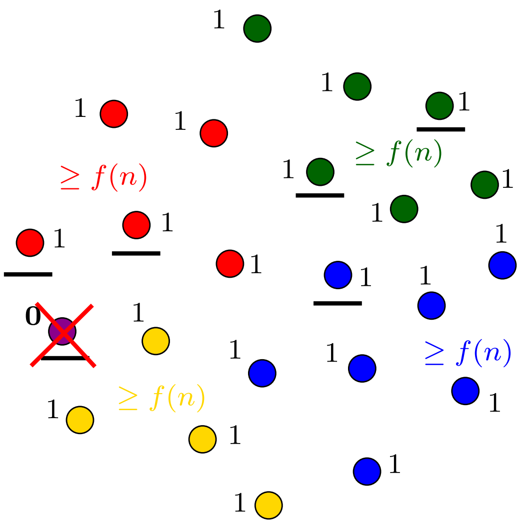


but also $C_{small} \subset C_{stable}$

Parity Cannot be Computed with High Symmetry



Output 0 can be produced
in $C_{small} \subset C_{stable}$



CONTRADICTION

Open Problems

- The impossibility excludes any protocol that would solve parity with symmetry depending on N_{min}
  - could be solvable with symmetry k, for any constant $k \geq 1$
- Exact characterization of the symmetry of all semilinear predicates
- Constant symmetry for parity can be achieved given auxiliary nodes
  - Can they be dropped? How is symmetry affected by auxiliary nodes?
- Networked systems (static or dynamic), much memory and/or UIDs
  - UIDs provide an *a priori* maximum symmetry breaking
  - Still, solving a task and avoiding an election may be highly non-trivial
  - How to define the "role" of a process here?
- More experimental and analytic work on the observed symmetry

Thank You!