Combinatorial Structures for Distributed Computing Models

Petr Kuznetsov
Telecom ParisTech

CIRM, 2019

This class is about distributed computing:

independent sequential processes that communicate

Communication models

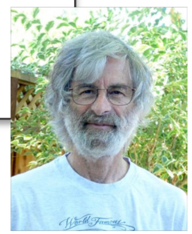
- Shared memory
  - ✓Processes apply operations on shared variables
  - ✓Failures and asynchrony

- Message passing
  - ✓Processes send and receive messages
  - ✓Communication graphs
  - ✓Message delays

Distributed ≠ Parallel

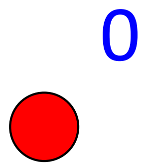
The main challenge is synchronization: resolving nondeterminism caused by the scheduler

> "you know you have a distributed system when the crash of a computer you've never heard of stops you from getting any work done" (Lamport)



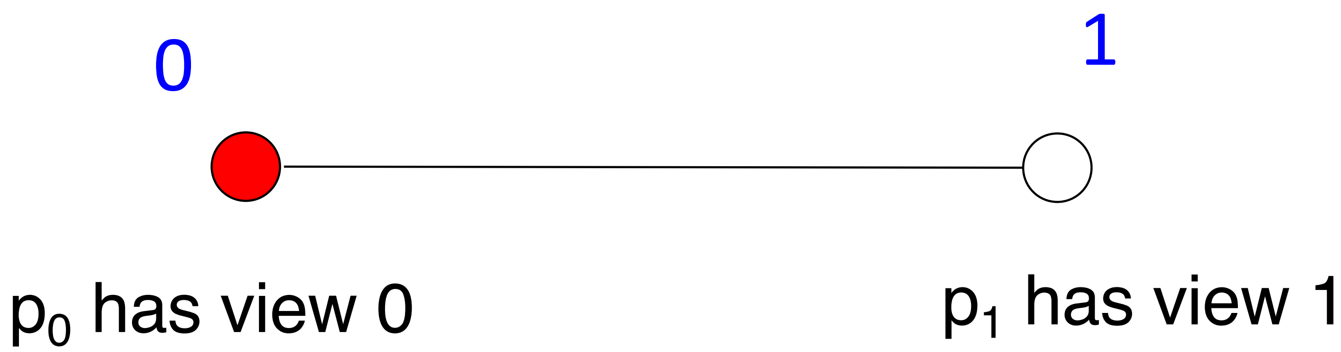
Indistinguishability: a local view can be compatible with multiple system states

Vertex: a local view



p_0 (red) has *view* 0

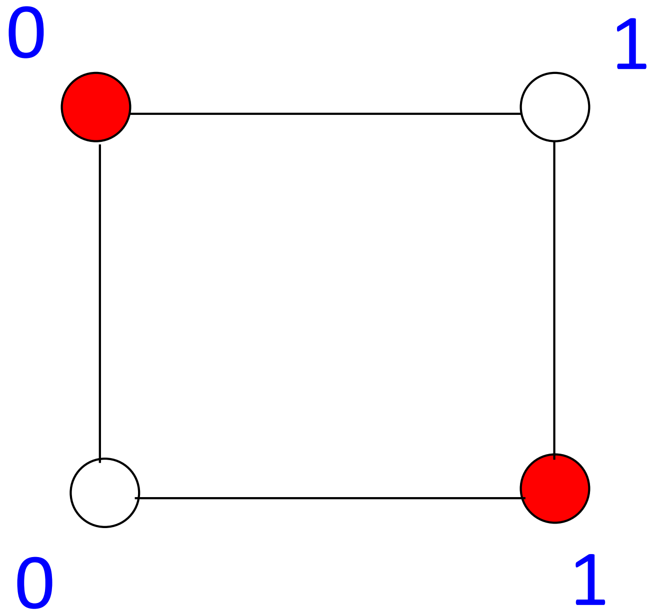
Simplex: a set of views that appear in the same state



There is a state in which
p_0 has view 0 and p_1 has view 1

1-dimensional simplex

Complex: a set of simplexes that represent possible states



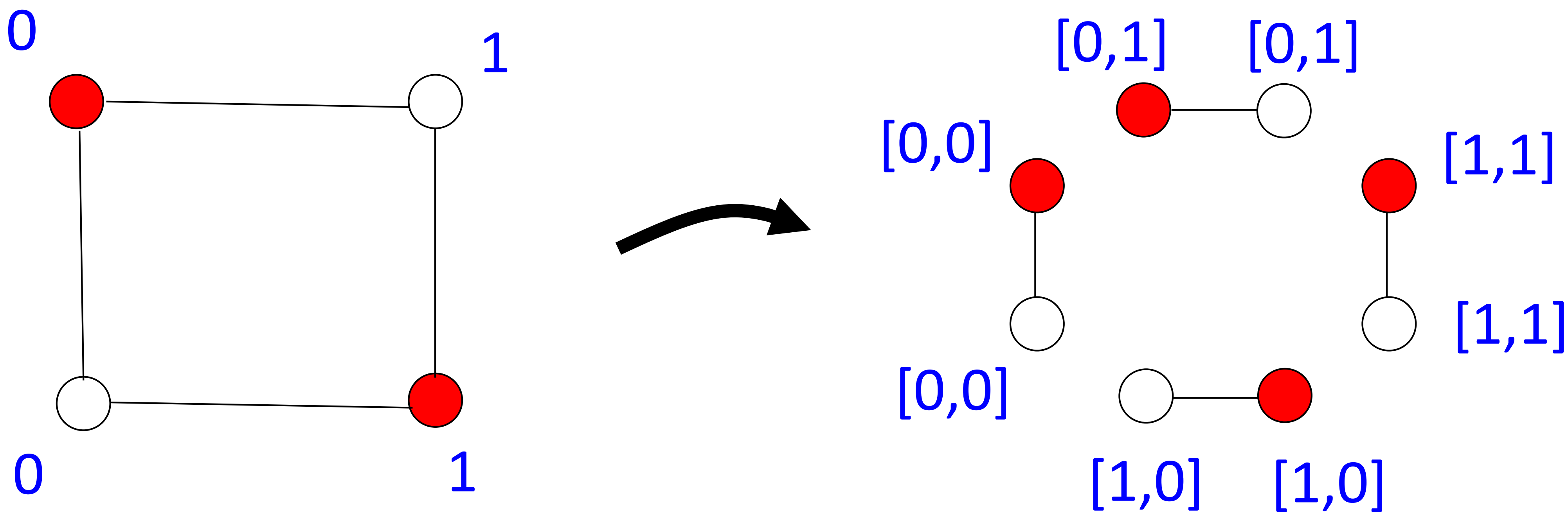
p_0 and p_1 pick up an input value in {0,1}

1-dimensional complex

Modeling computations

How the protocol complex looks like?

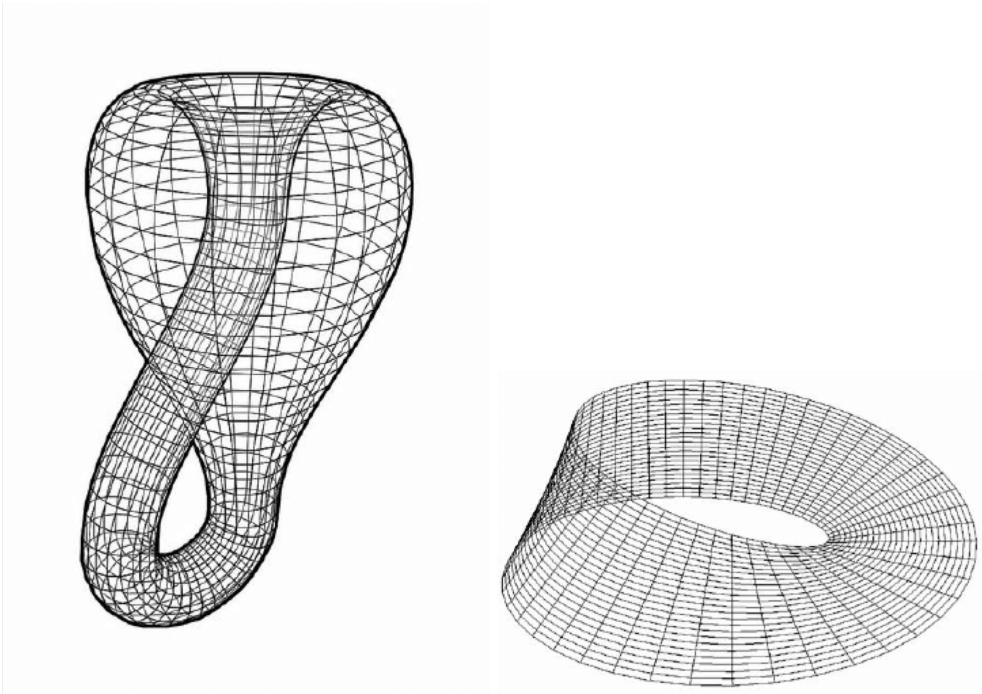
Suppose that p_0 and p_1 communicate via a reliable channel



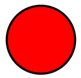
Roadmap

- Topology primer
- Shared memory models and set consensus
- Asynchronous Computability

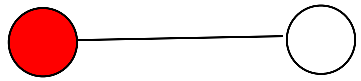
Topology primer



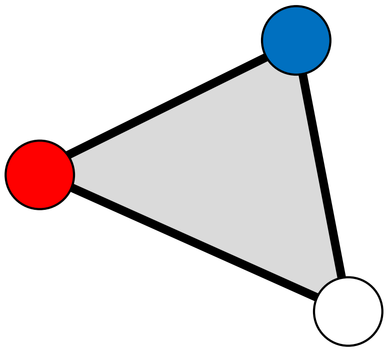
Simplexes



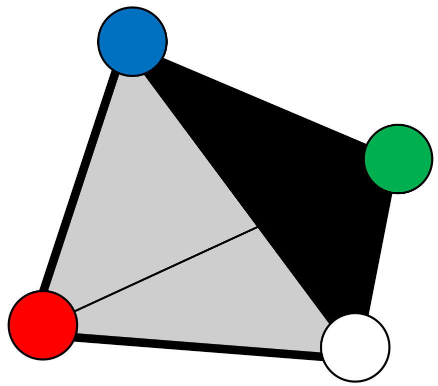
0-simplex



1-simplex

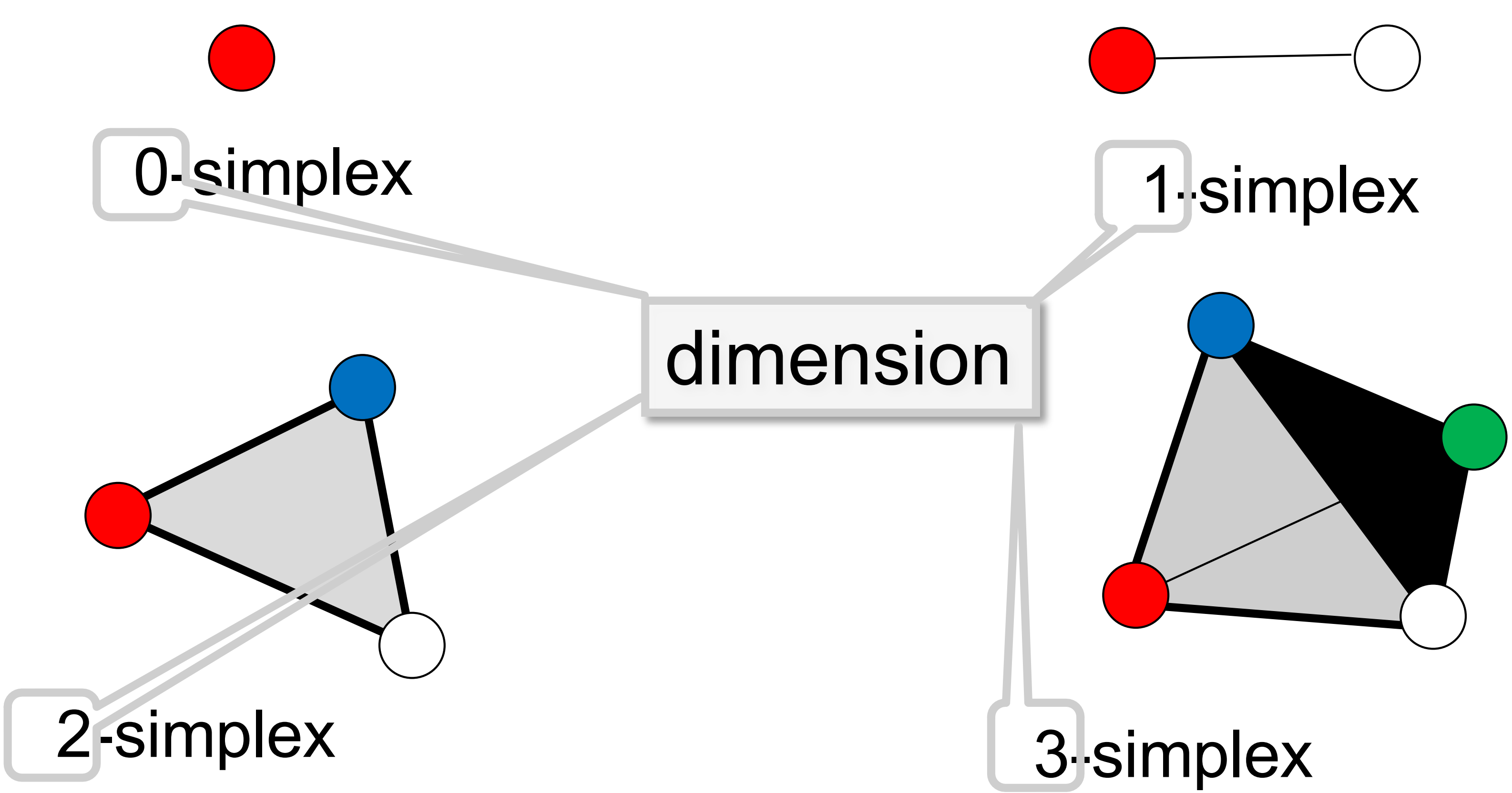


2-simplex



3-simplex

Simplexes



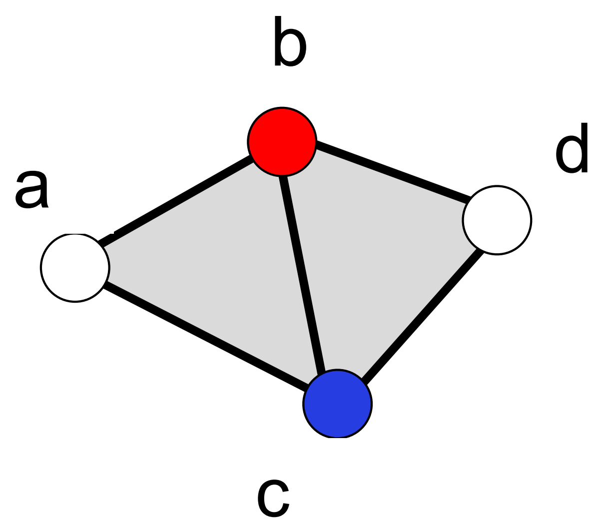
Combinatorial: a set of vertexes

Geometric: a convex hull on linearly independent points

Simplicial Complex

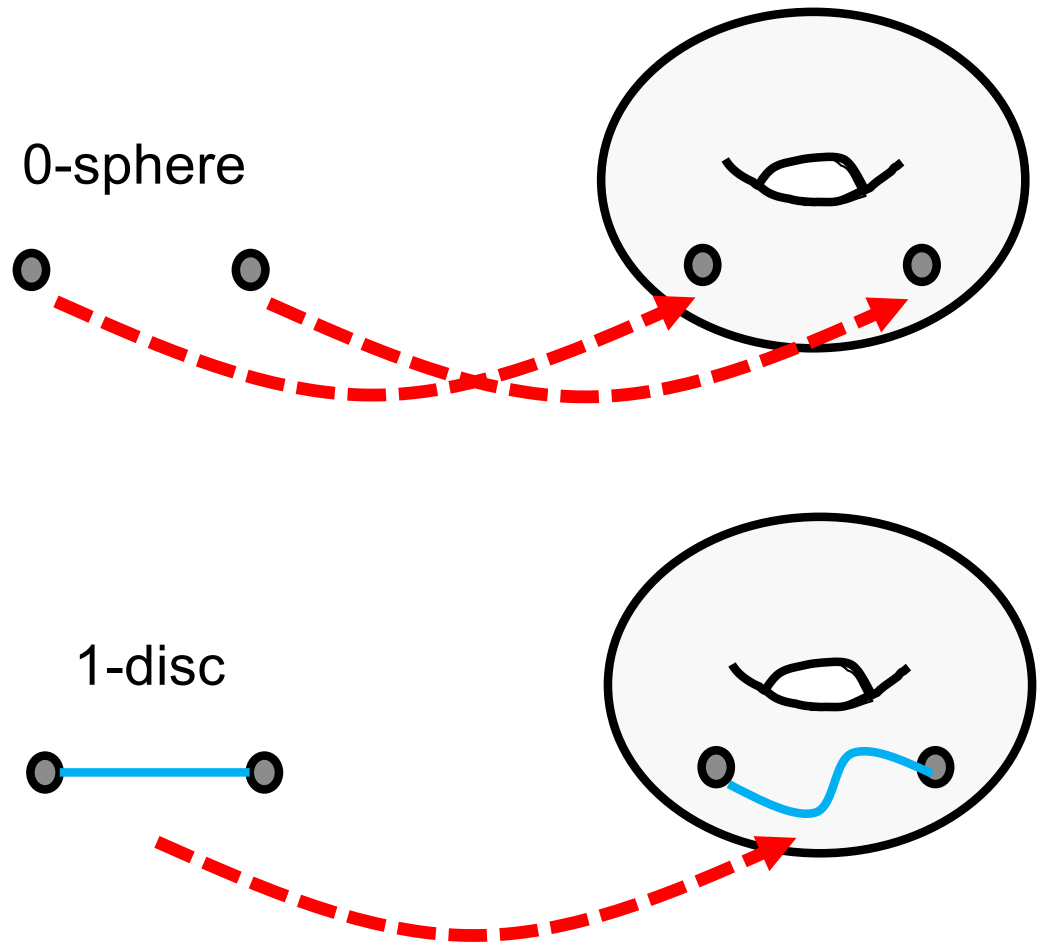
Combinatorial $\mathcal{A}$: set of simplices closed under inclusion

$\mathcal{A}$= {{a,b,c},{b,c,d}} + all subsets



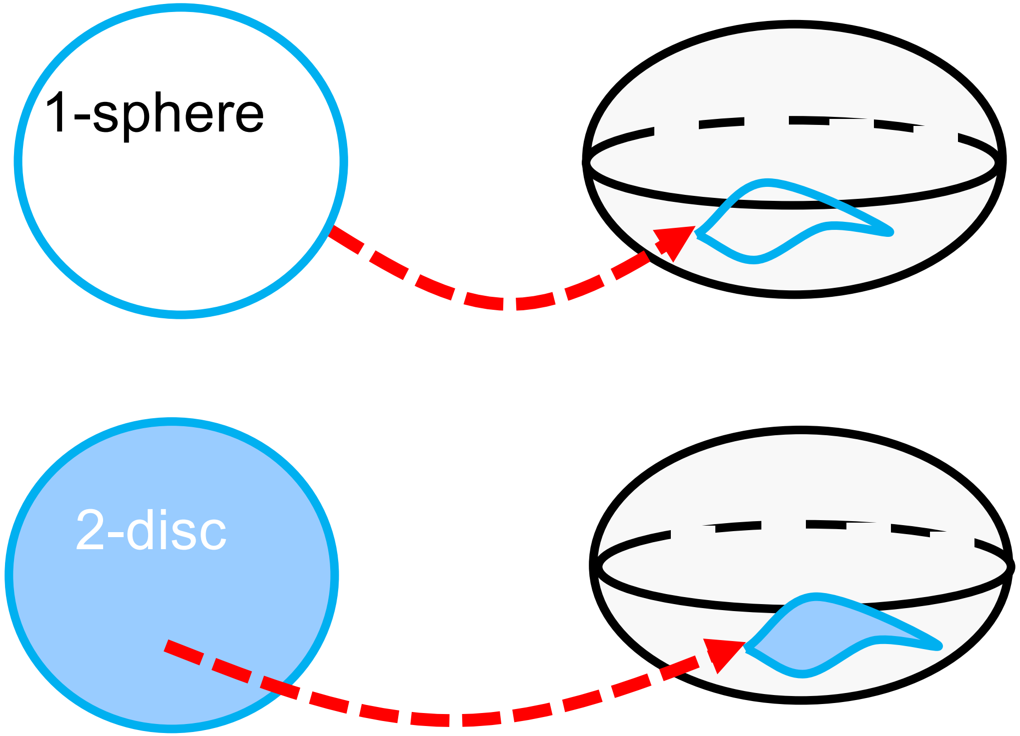
Geometric $|\mathcal{A}|$: set of geometric simplices, closed under containment

Connectivity



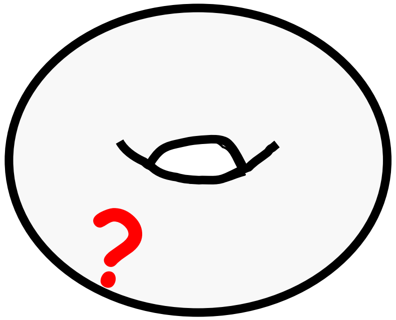
0-connected (path connected)

Connectivity



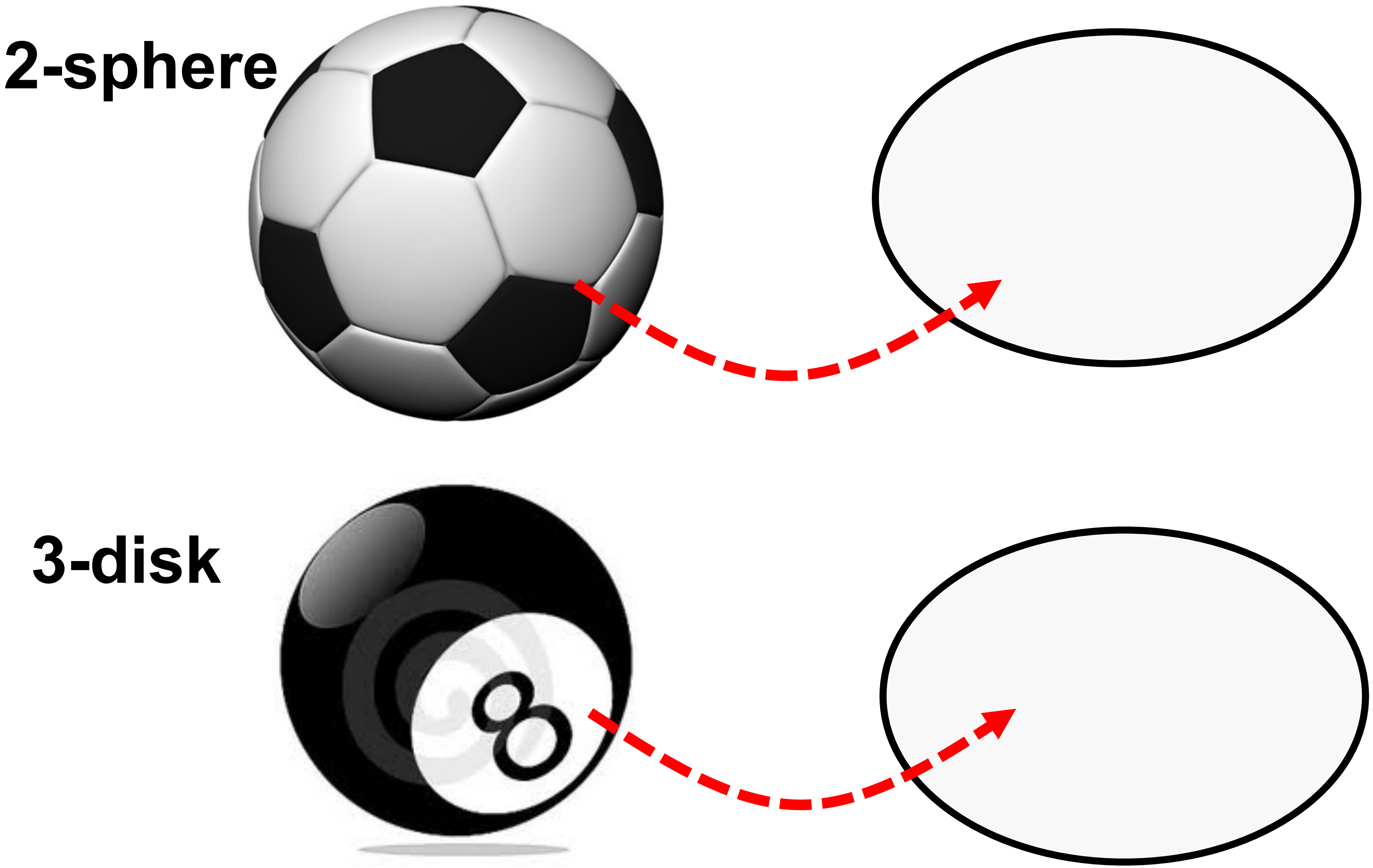
1-connected

(simply connected)

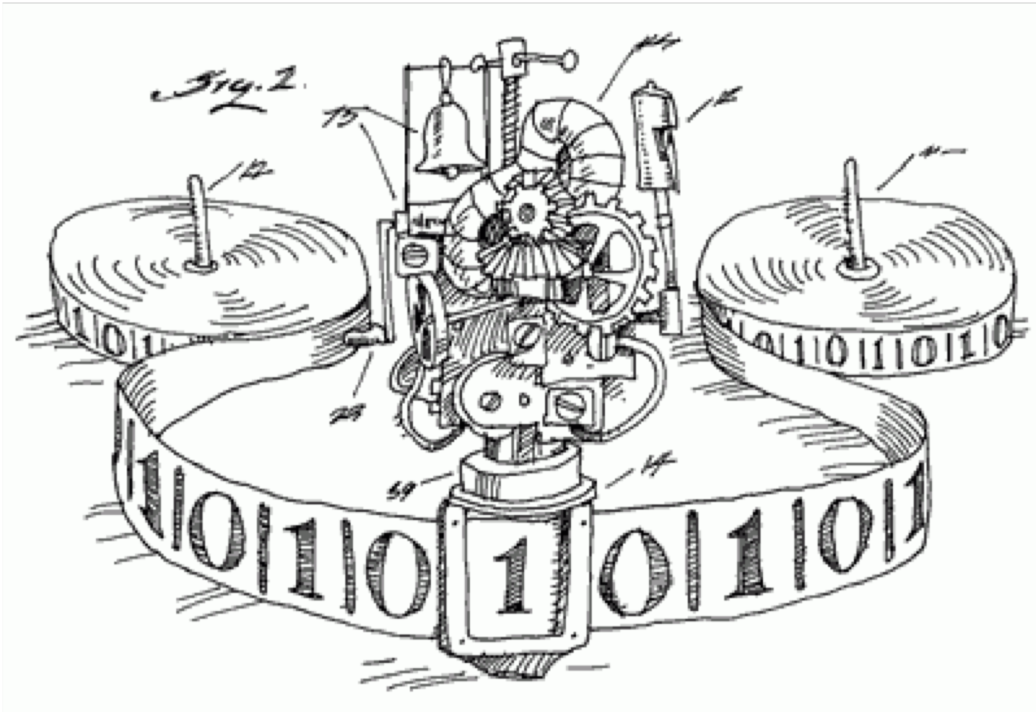


Not 1-connected

2-Connectivity



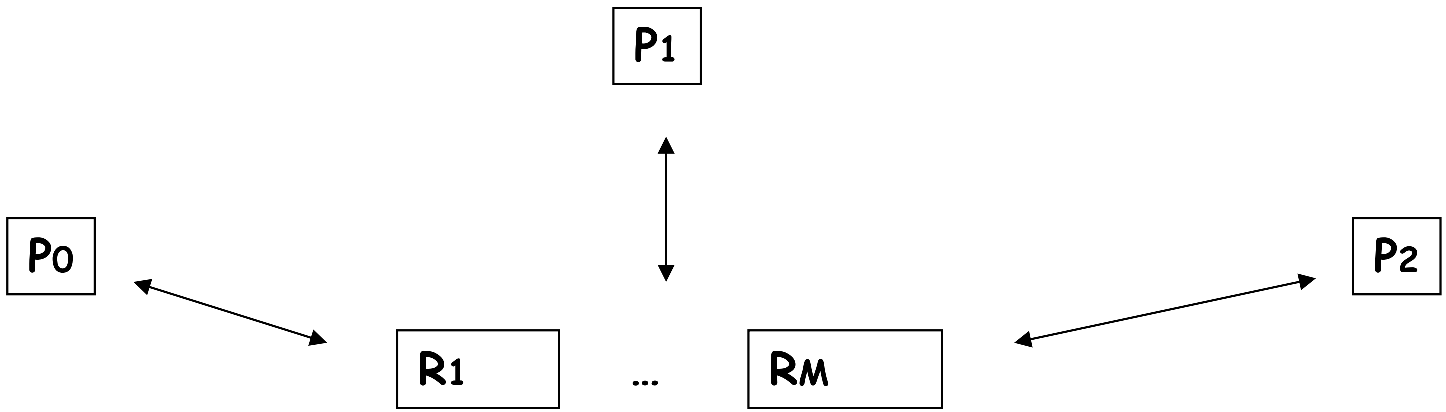
Back to computing



Read-write shared memory

- N+1 *asynchronous* (no bounds on relative speeds) processes $p_0,\ldots,p_N$ ($N \geq 1$) communicate via atomic read-write registers

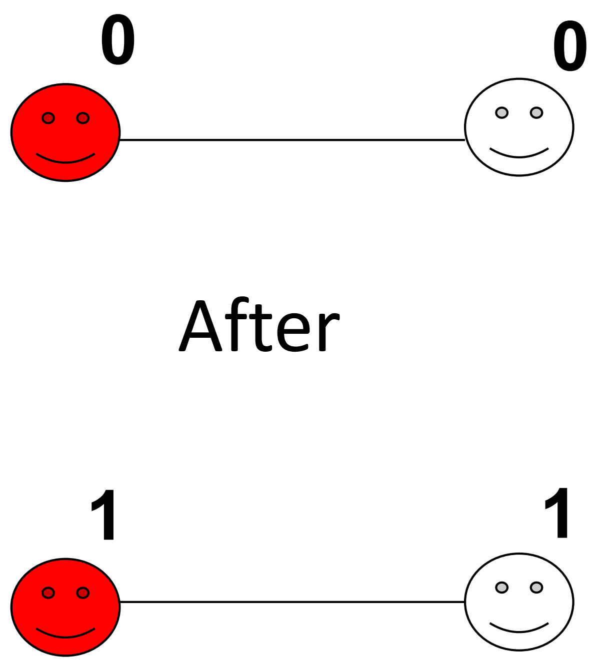
- Processes can fail by crashing
  - ✓A crashed process takes only finitely many steps (reads and writes)
  - ✓Up to t processes can crash: t-resilient system
  - ✓t=N: wait-free



Solving 2-process consensus?

Processes *propose* values and must *agree* on a common decision value so that the decided value is a proposed value of some process

Before



After

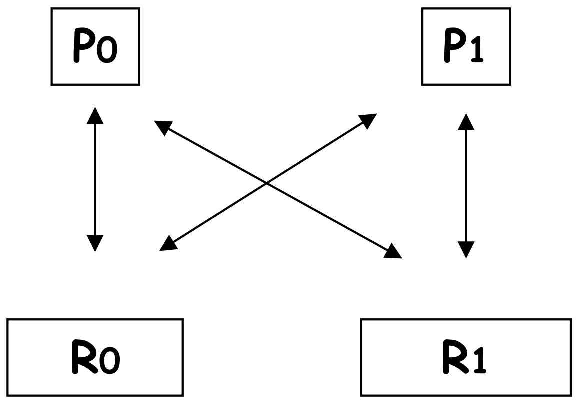
Key in state-machine replication [Paxos,BFT,…]

One-round interaction

Each process p_i (i=0,1):

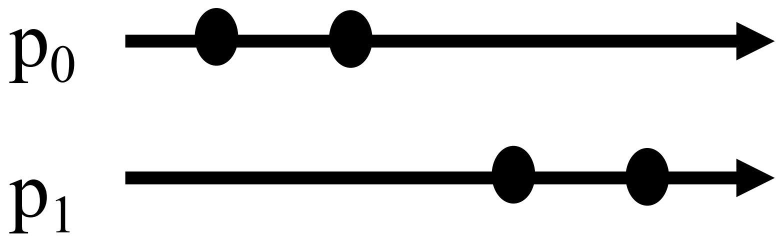
R_i:=v_i;   // write the input

S_i := R_{1-i} // read the input of p_{1-i}

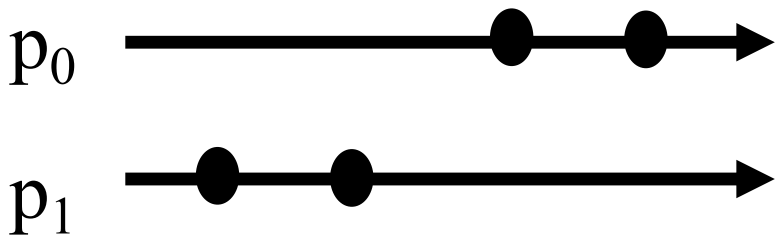


Three cases to consider

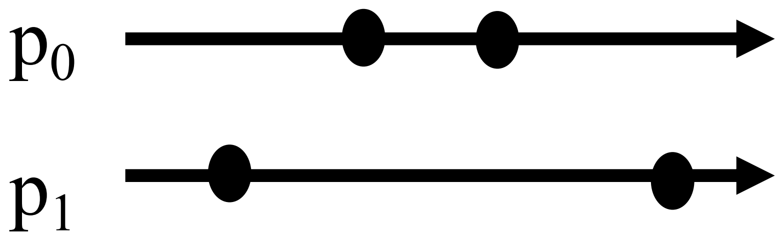
- p_0 reads before p_1 writes



- p_1 reads before p_0 writes



- p_0 and p_1 go “lock-step”



One-round protocol complex

p_0 reads before p_1 writes

p_1 reads after p_0 writes

p_0 reads after p_1 writes

p_1 reads before p_0 writes

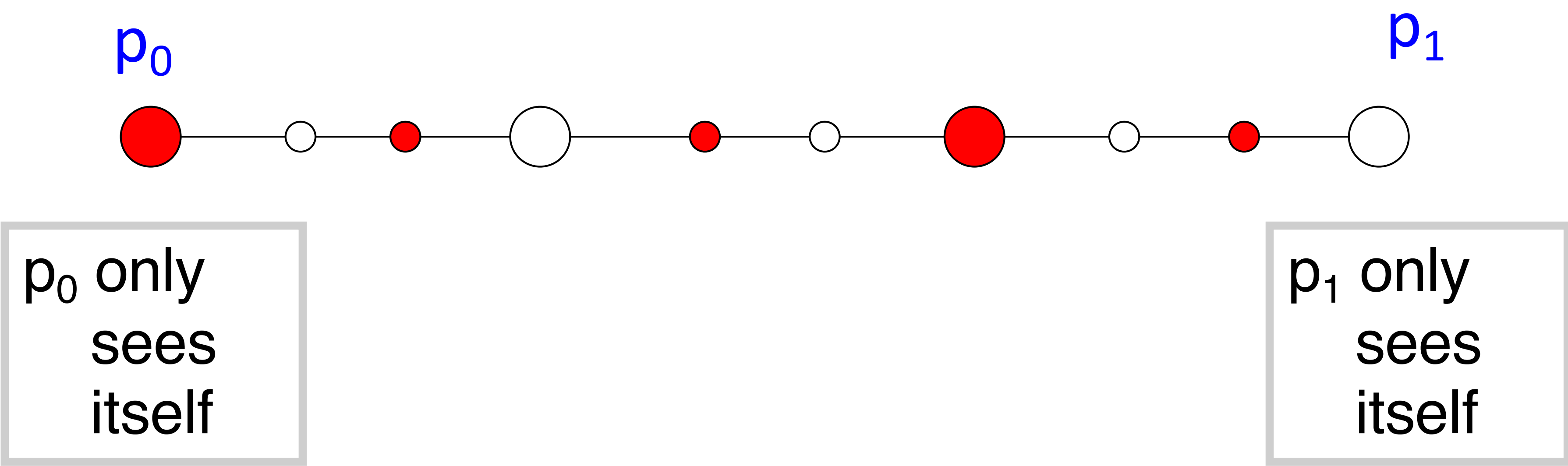
p_0 only sees itself

p_1 sees p_0

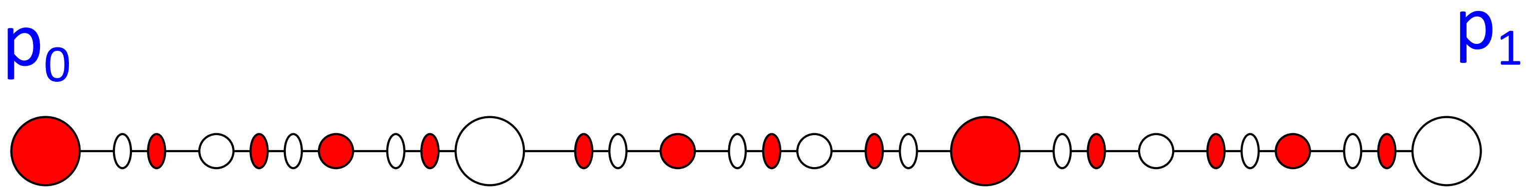
p_0 sees p_1

p_1 only sees itself

Two-round protocol complex



And so on…



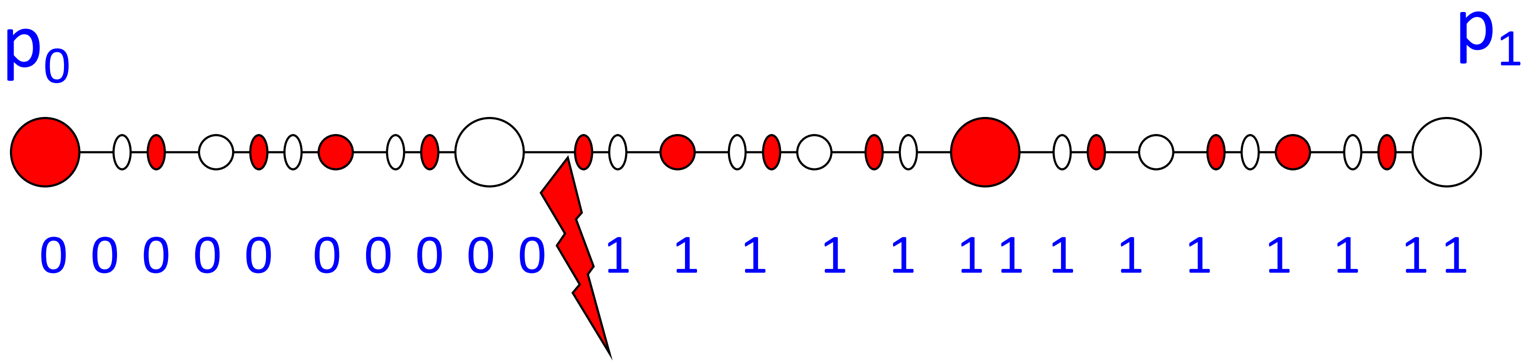
p_0 only sees itself

p_1 only sees itself

Solo runs remain connected - no way to decide!

Connectivity argument

- p_0 proposes 0, p_1 proposes 1
- p_i must decide i in a solo run!



There exists a run with conflicting decisions!

Impossibility of wait-free consensus
[FLP85,LA87]

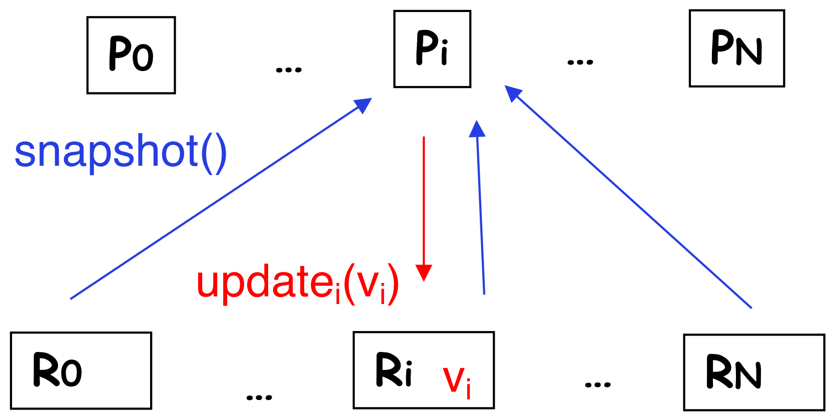
Theorem Consensus has no wait-free solution using reads and writes

(Can be strengthened to 1-resilient impossibility)

Immediate Snapshot model

Each process p_i $(i=0,..,N)$:

$\left.\begin{array}{l} \text{update}_i(v_i) \\ S_i := \text{snapshot}() \end{array}\right\}$ Atomically in batches

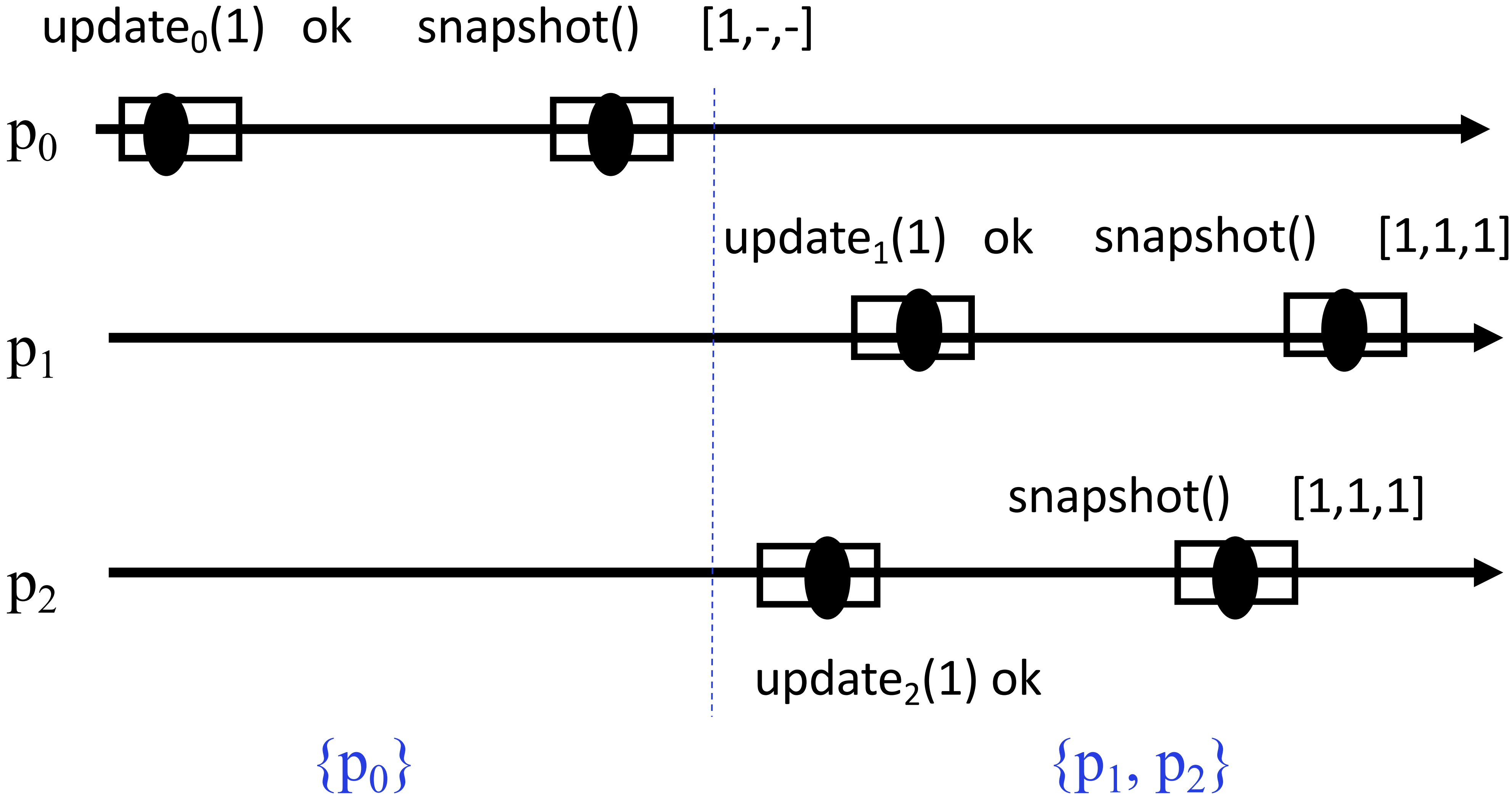


Vectors S_i satisfy:

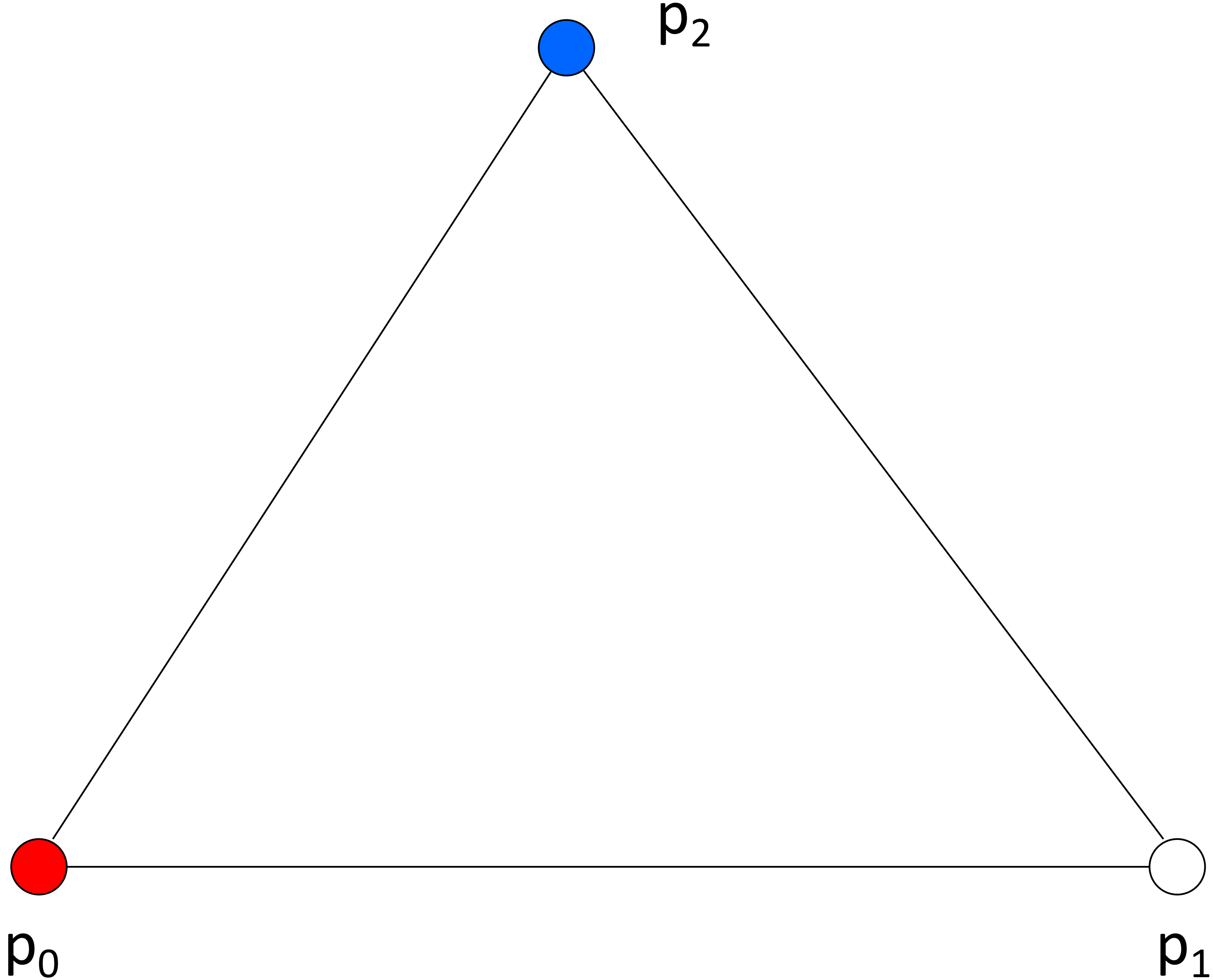
- Self-inclusion: for all i: $v_i \in S_i$
- Containment: for all i and j: $S_i \subseteq S_j$ or $S_j \subseteq S_i$
- Immediacy: for all i and j: $v_i \in S_j \Rightarrow S_i \subseteq S_j$

Can be implemented from atomic registers!

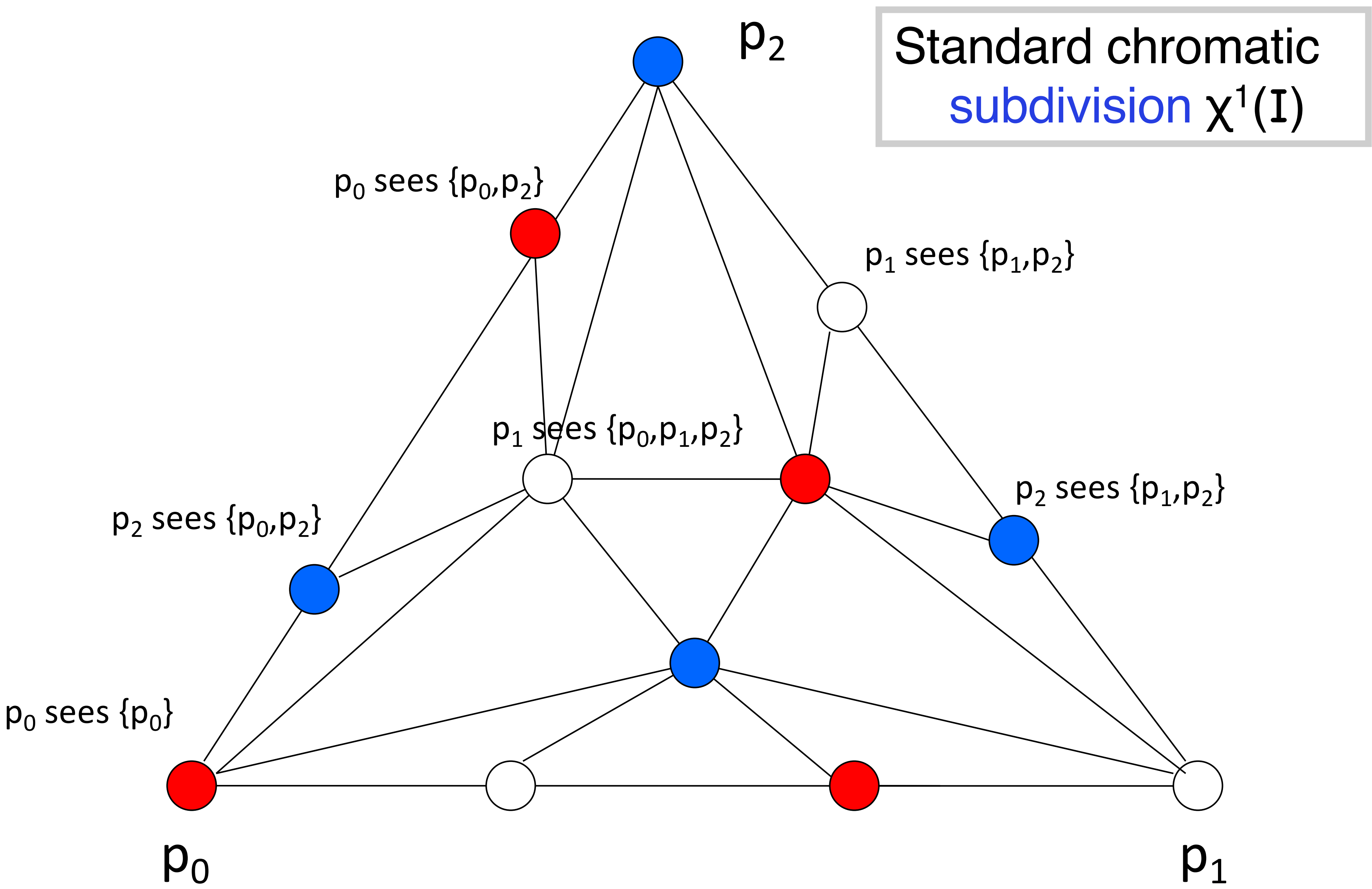
Immediate snapshot execution



Initial state I for three processes

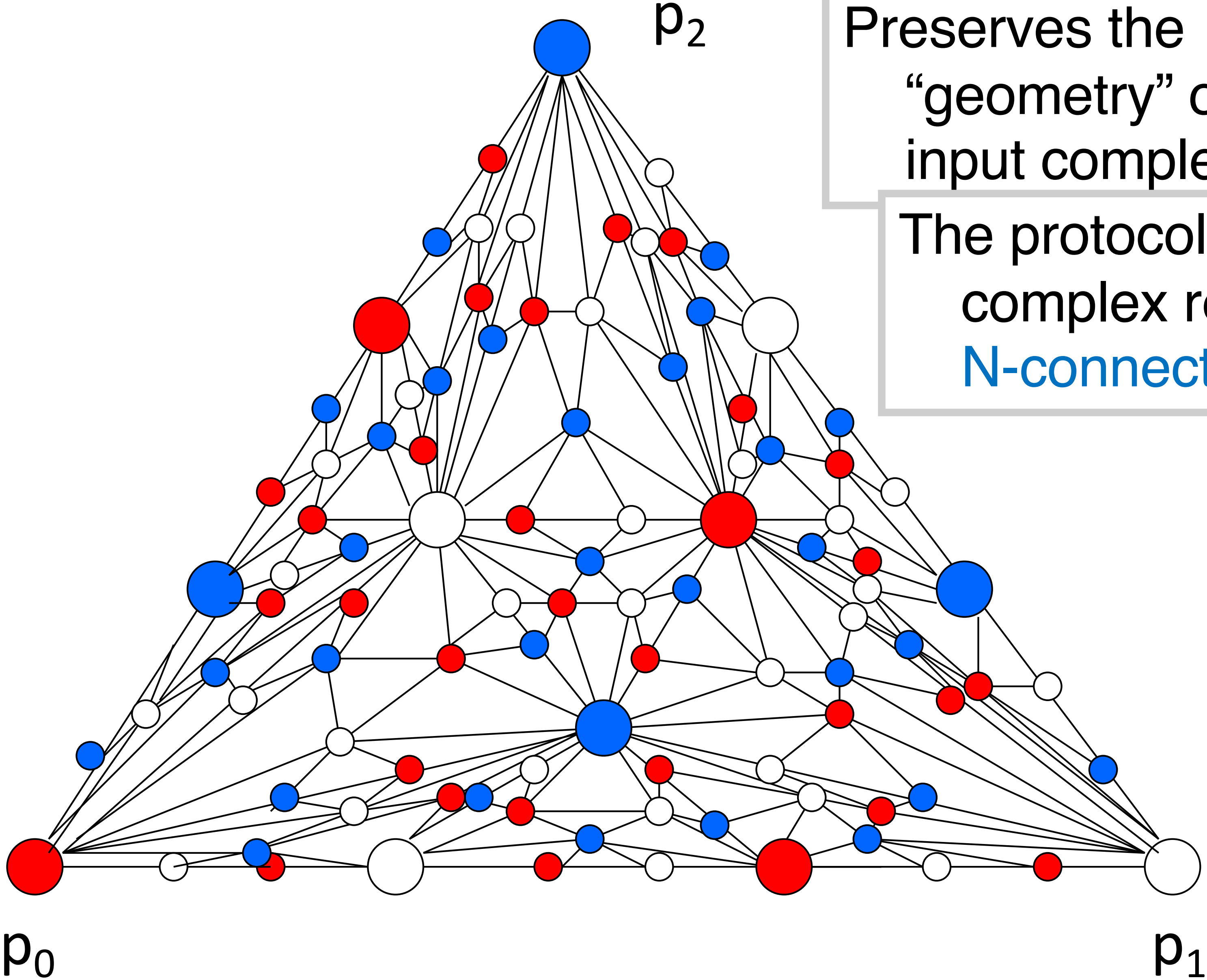


One round



p_2
Standard chromatic subdivision $\chi^1(\mathcal{I})$
p_0 sees $\{p_0,p_2\}$
p_1 sees $\{p_1,p_2\}$
p_1 sees $\{p_0,p_1,p_2\}$
p_2 sees $\{p_1,p_2\}$
p_2 sees $\{p_0,p_2\}$
p_0 sees $\{p_0\}$
p_0
p_1

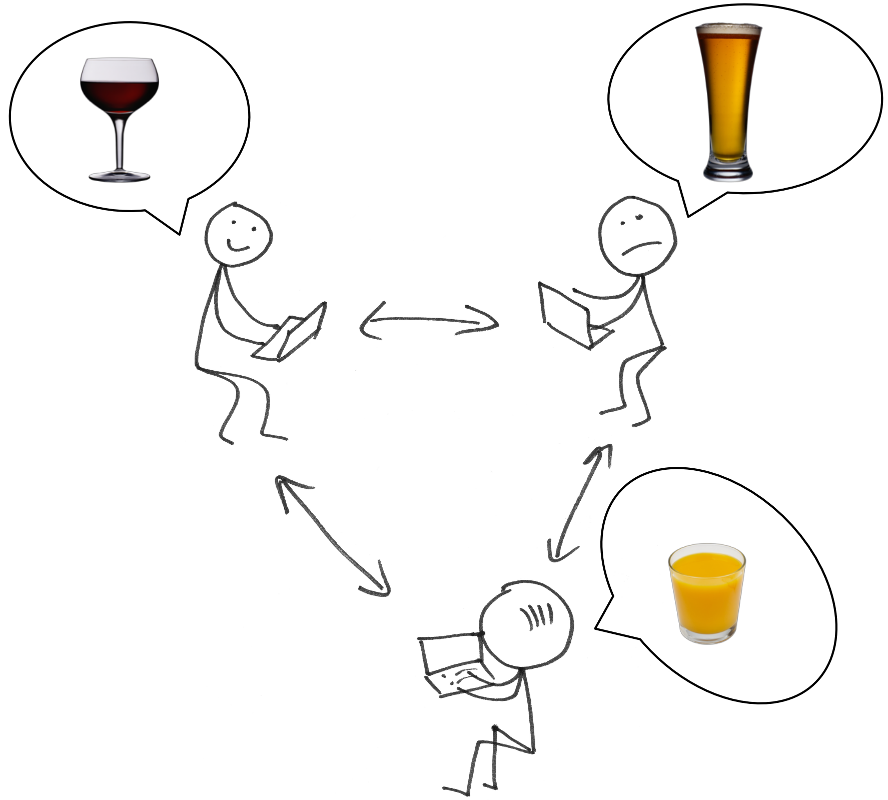
Two rounds: $\chi^2(\mathtt{I})$



Preserves the "geometry" of the input complex

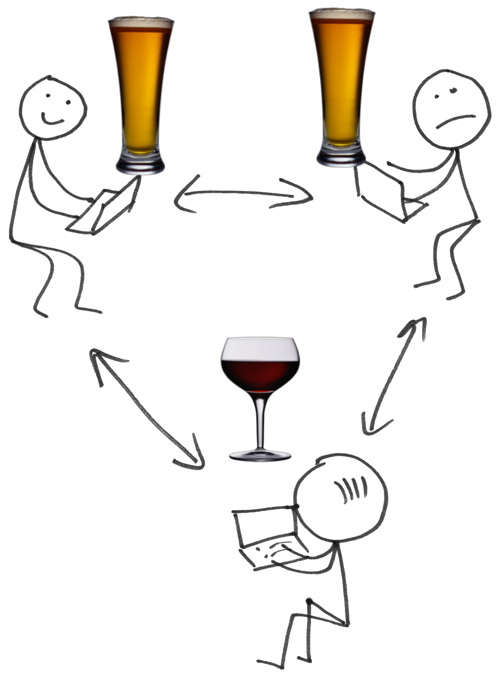
The protocol complex remains N-connected

k-set consensus



Processes start with private inputs

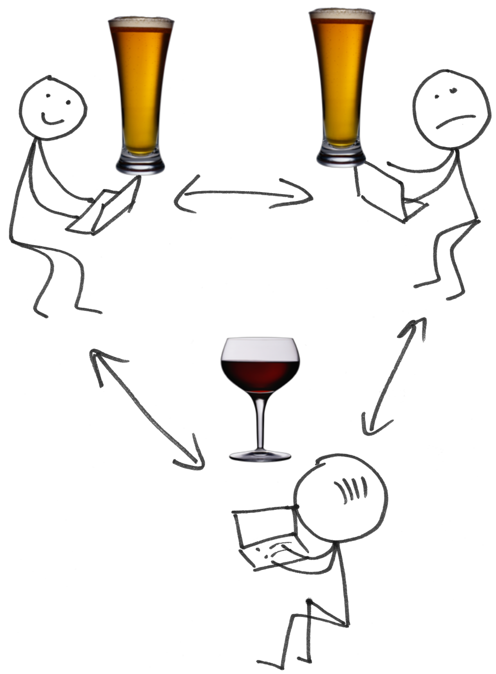
k-set consensus



Outputs should form a k-bounded subset of inputs

k-set consensus

2-set consensus



1-set consensus = consensus

N-set consensus = set consensus
(for N+1 processes)

Impossibility of wait-free set consensus
[BG93,HS93,SZ93]

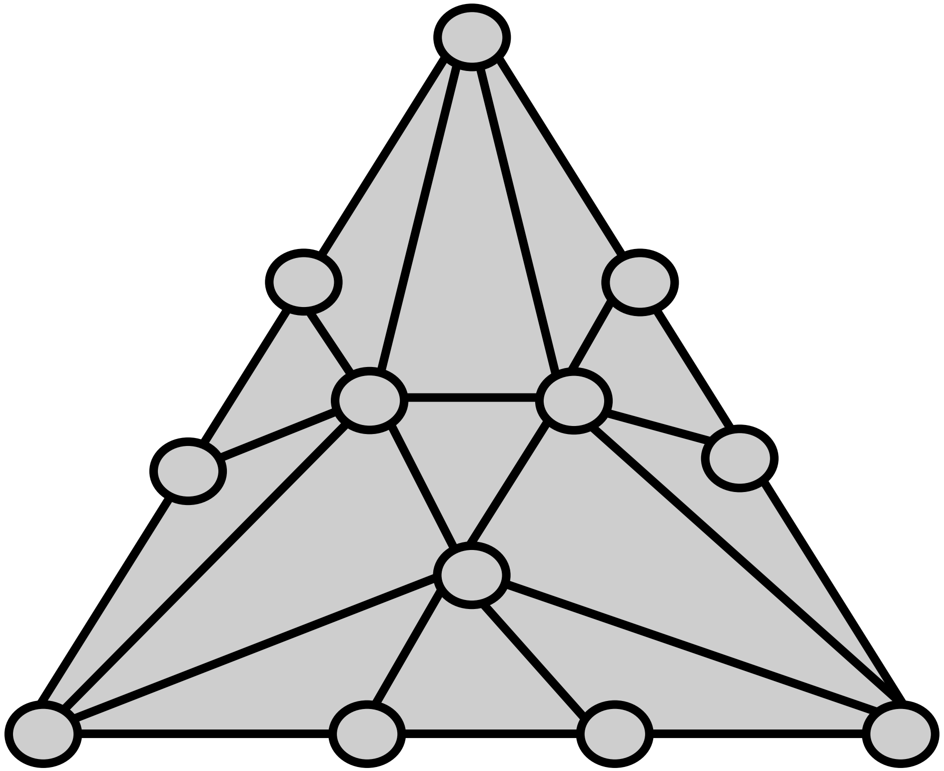
Theorem No (N+1)-process wait-free algorithm can solve N-set consensus in the iterated immediate snapshot model (IIS)

(and, thus, using read-write)

Gödel prize, 2004

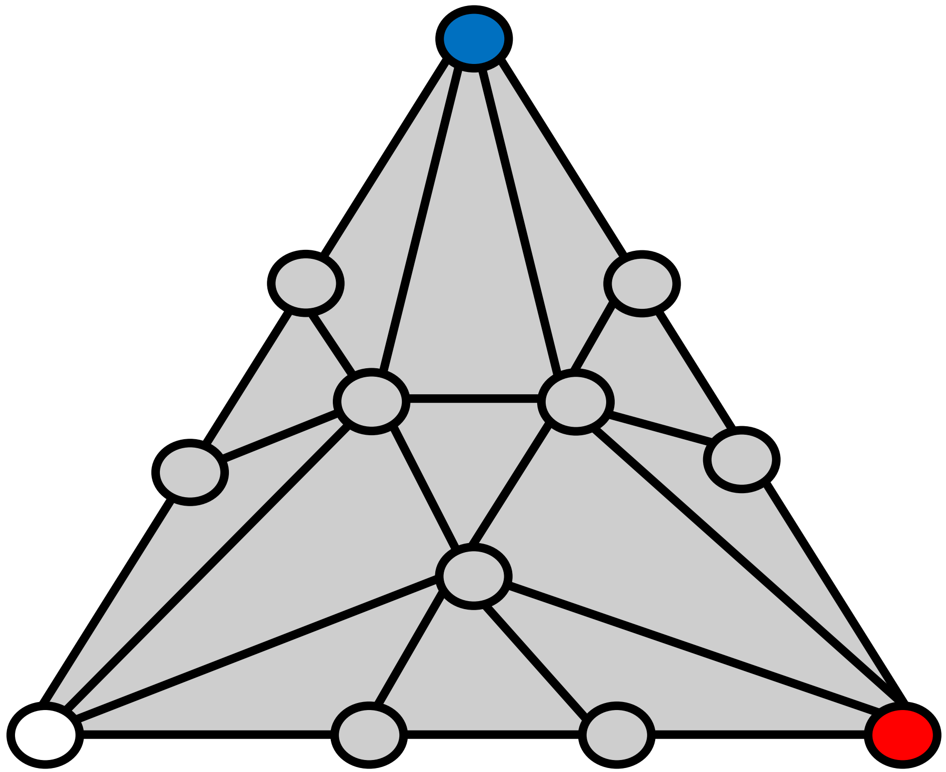
Reduces to Sperner's lemma: impossibility of Sperner coloring on a manifold

Sperner Coloring



Sperner Coloring

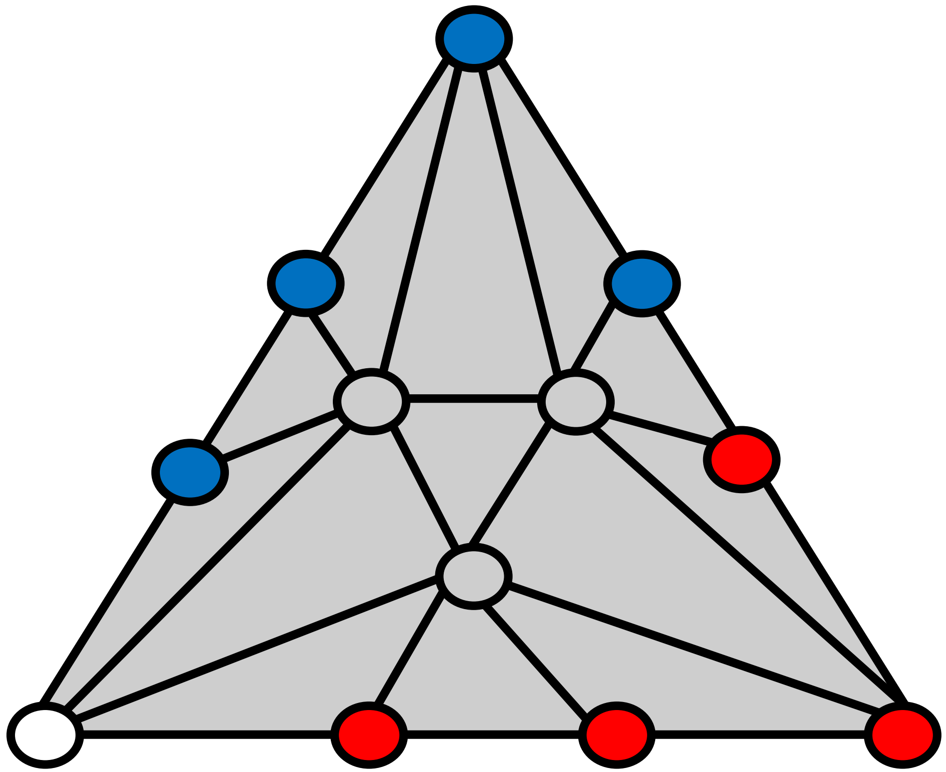
Corners get distinct colors



Sperner Coloring

Corners get distinct colors

Edges get corner colors

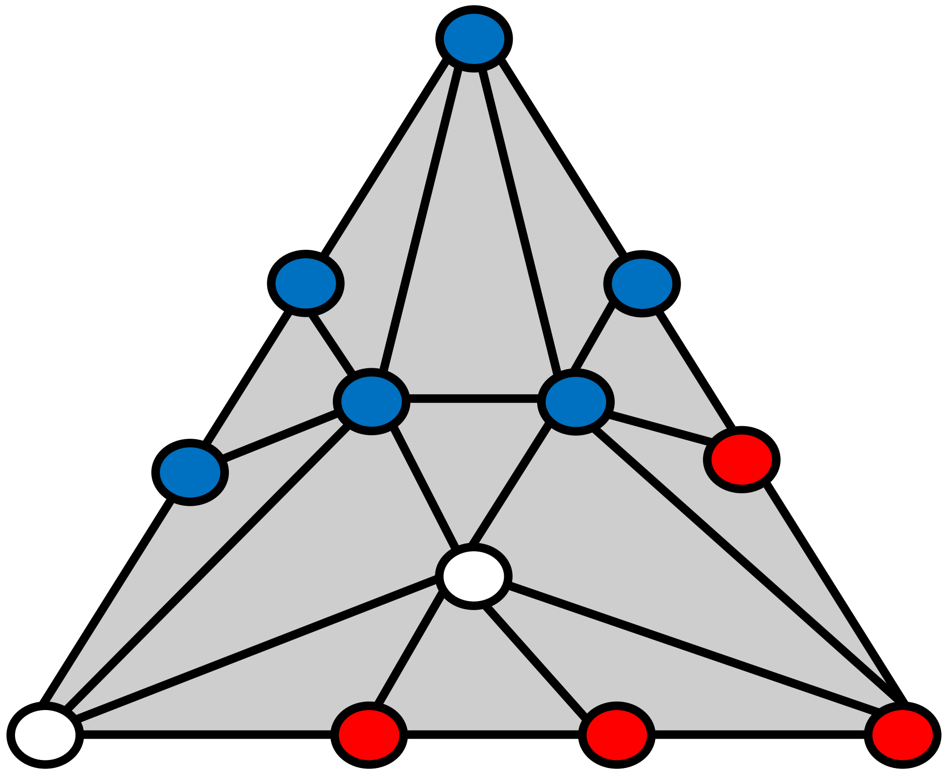


Sperner Coloring

Corners get distinct colors

Edges get corner colors

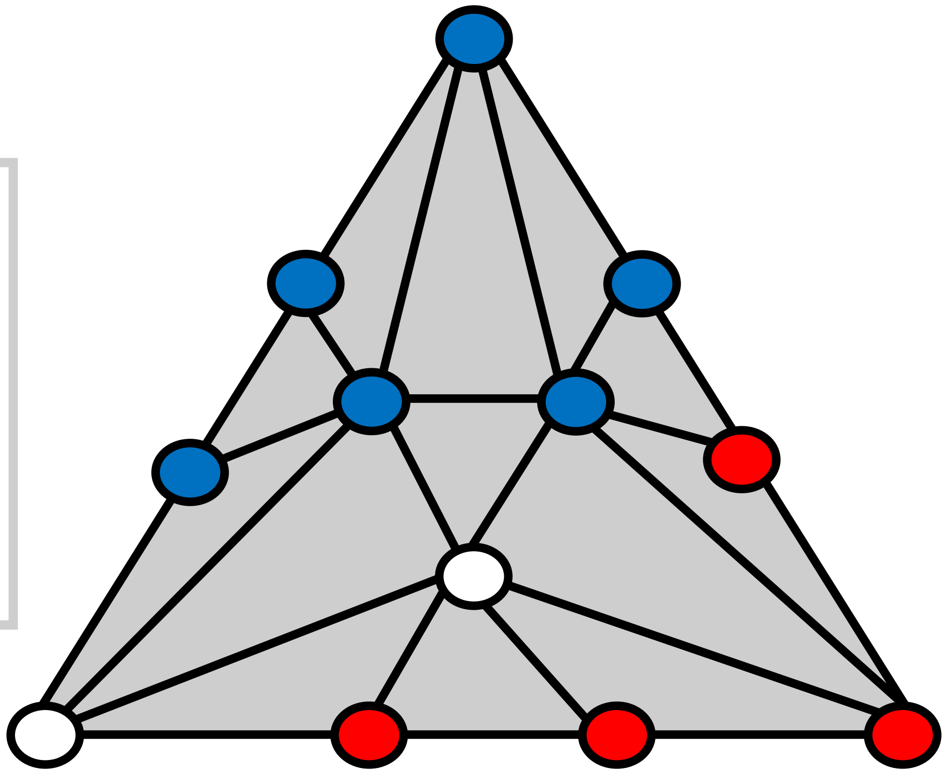
Every vertex gets colors of its **carrier face**



You only decide on a value you heard of

Sperner's Lemma

Every Sperner coloring has a simplex with all N+1 colors



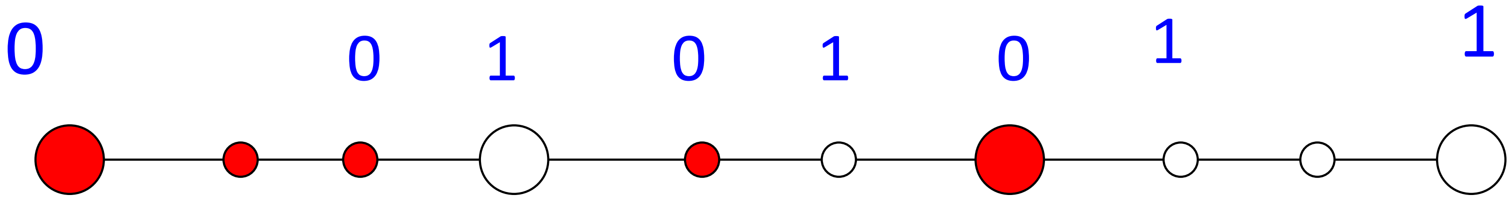
In at least one run, all N+1 values are decided => N-set consensus is impossible

Sperner's lemma: inductive step

Claim: for each k=0,…,N, face {0,…,k} contains an odd number of k-dimensional simplexes colored 0,…,k

By induction: k=0 - trivial (exactly one)

k=1, simple counting



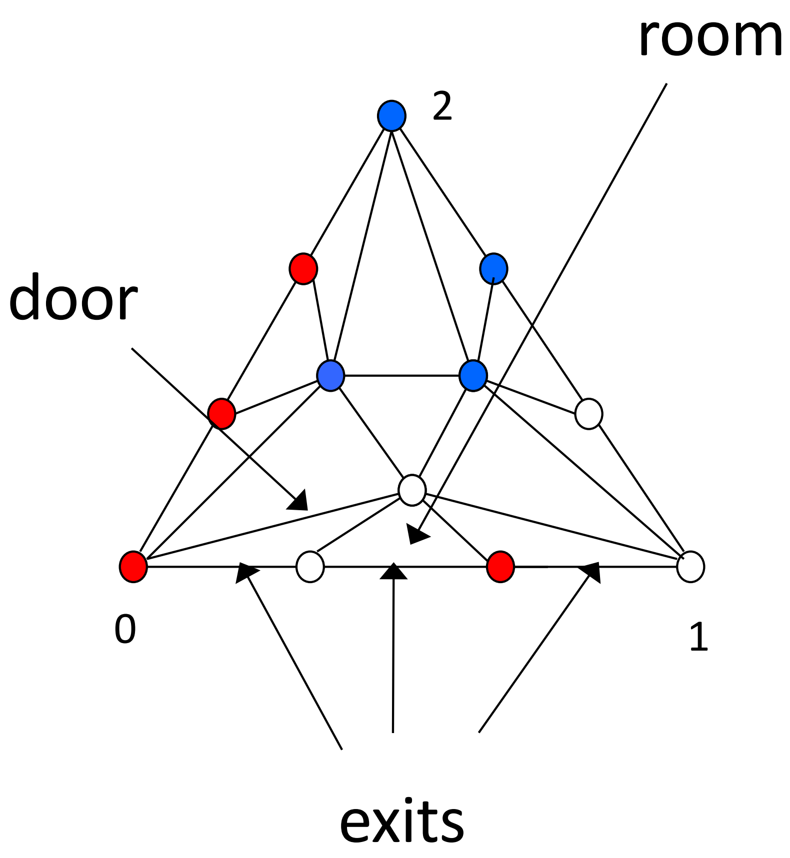
Suppose the claim holds for k=N-1 and consider the face 0,…,N

Sperner: rooms and doors

Each N-simplex is a room

An (N-1)-dimensional face
(a subset of N-1 vertices)
of a room colored in
0,…,N-1 is a door

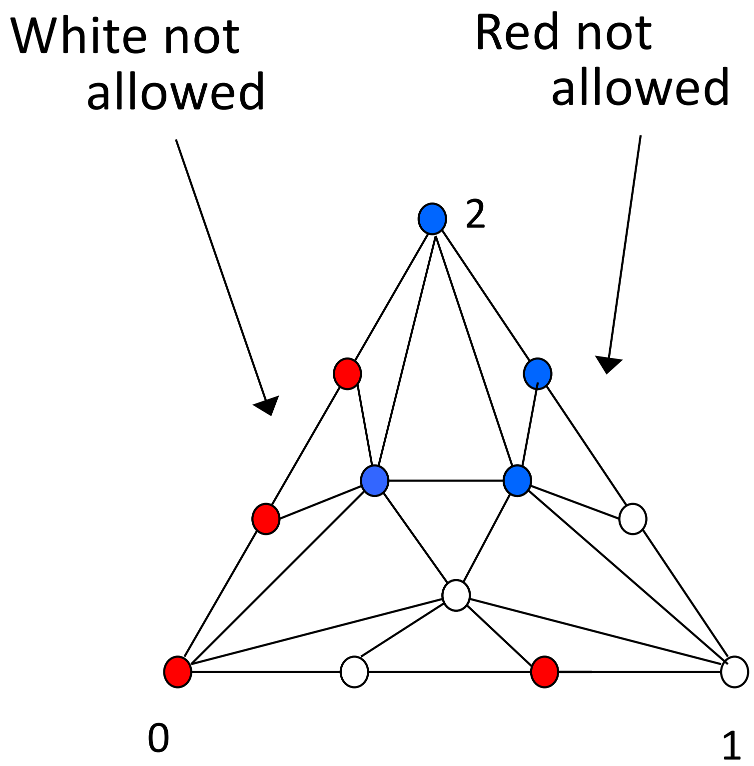
A door is an exit if it
belongs to the boundary



Sperner: exits

There is an odd number of exits!

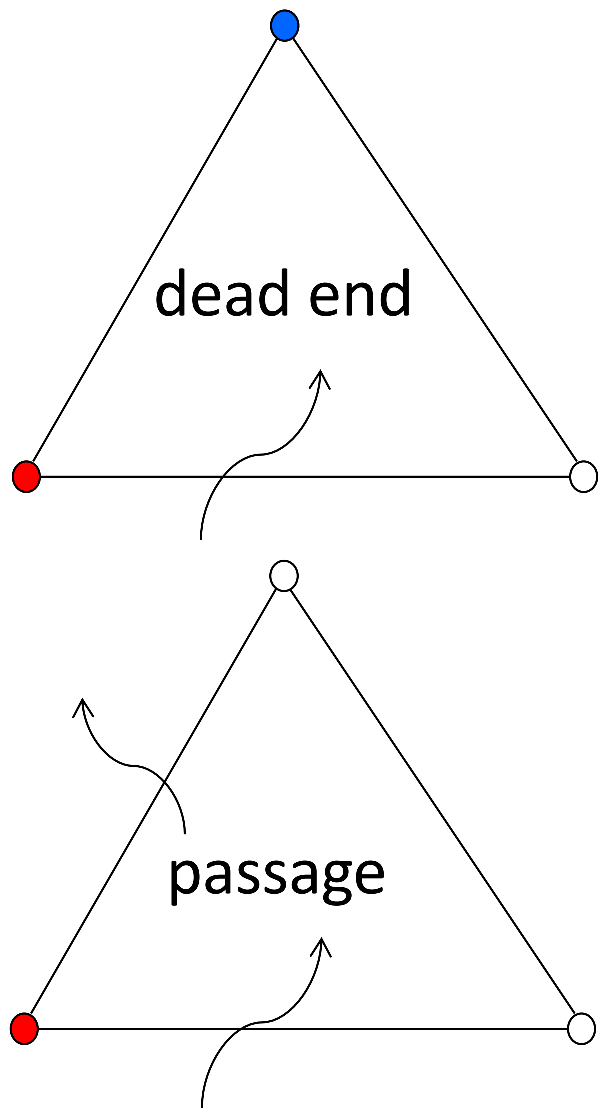
- No face other than 0,…,N-1 can contain simplexes colored 0,…,N-1
- Exits may only be contained in 0,…,N-1



Sperner: passages and dead ends

A room with a door is either:

- A passage (has two doors), or
- A dead end (has no doors)



We must show that there is an odd number of dead ends (fully colored simplexes)

Sperner: counting fully colored rooms

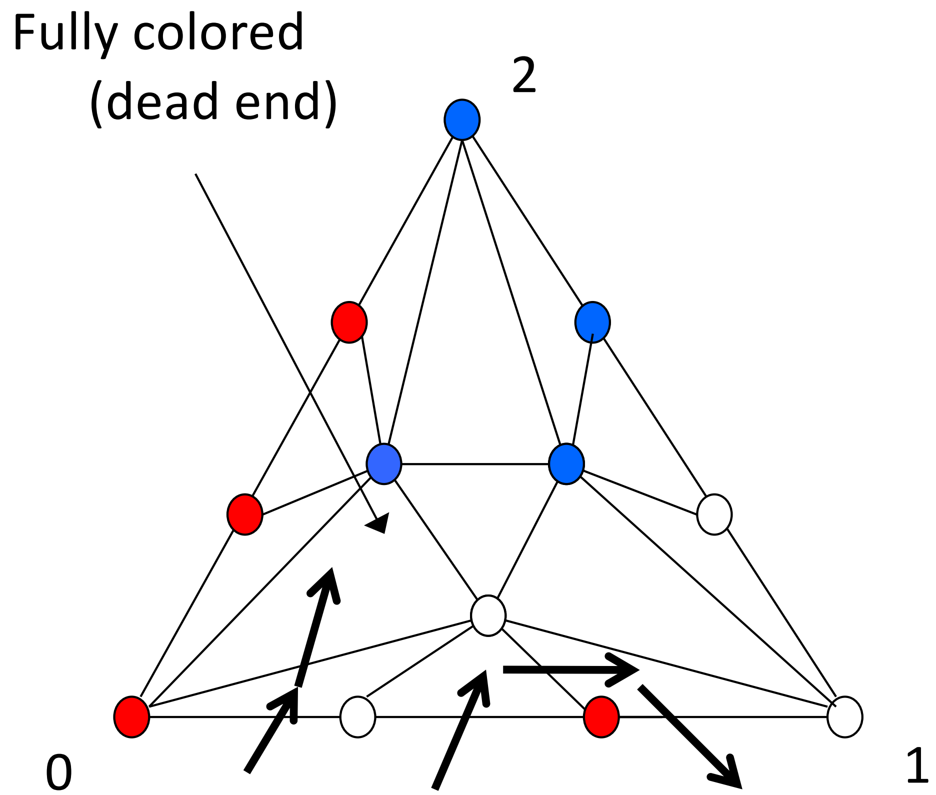
Start with an exit and walk through the doors

Two cases are possible:

- Stop in a dead end
- Reach another exit

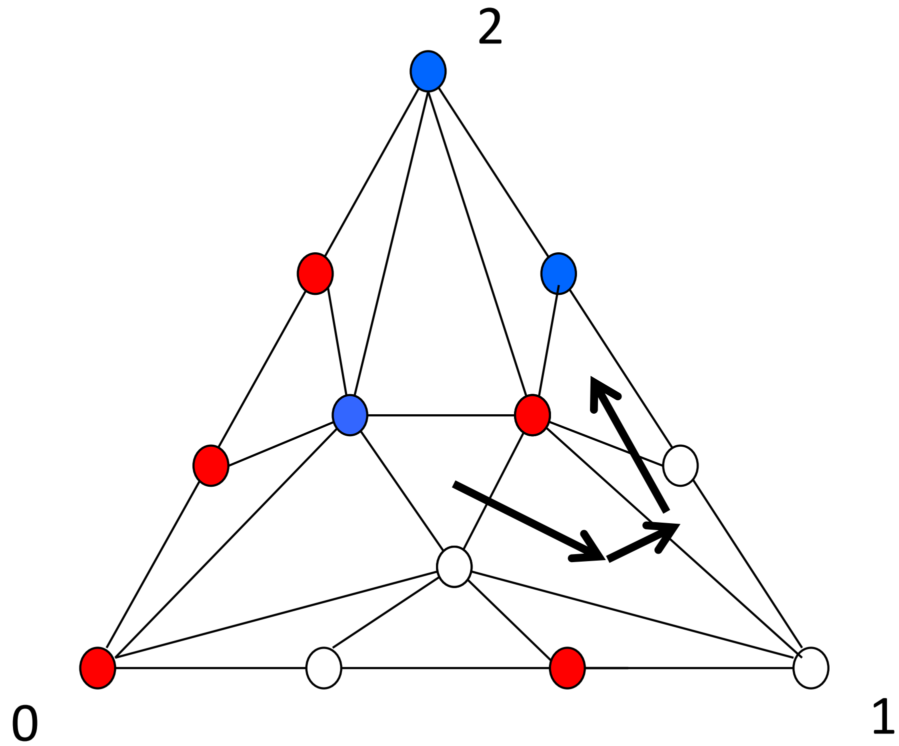
The number of exit doors is odd =>

The total number of fully colored rooms is odd



Sperner: internal dead ends

The number of inaccessible dead ends must be even



Impossibility of wait-free set consensus
[BG93,HS93,SZ93]

Theorem No (N+1)-process wait-free algorithm can solve N-set consensus in the iterated immediate snapshot model (IIS)

(and, thus, using read-write)

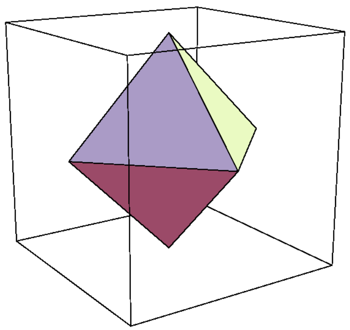
Generalization [BG93]: there is no k-resilient algorithm for k-set consensus

(BG agreement simulation technique)

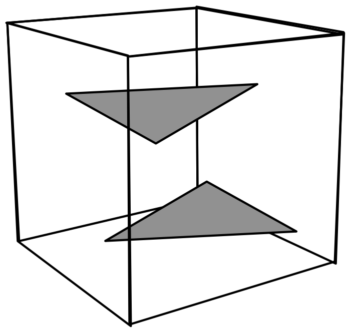
Asynchronous computability

Task specification

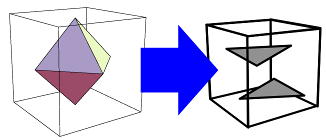
$$(\mathcal{I}, \mathcal{O}, \Delta)$$



Input complex



Output complex



Carrier map

$\Delta: \mathcal{I} \to 2^{\mathcal{O}}$

Asynchronous Computability Theorem [HS99]

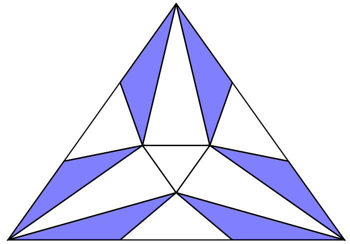
A task $(\mathcal{I},\mathcal{O},\Delta)$ is wait-free read-write solvable if and only if there is a chromatic simplicial map from a subdivision $\chi^r(\mathcal{I})$ to $\mathcal{O}$ carried by Δ

For colorless tasks (e.g., k-set consensus):

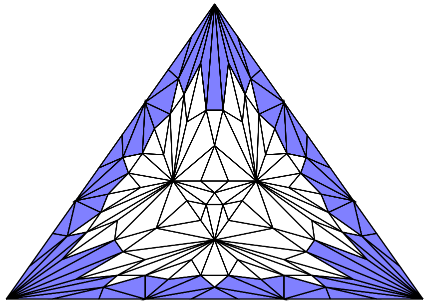
… there exists a continuous map from $|\mathcal{I}|$ to $|\mathcal{O}|$ carried by Δ

Generalizing ACT [KRH18]

- Models with stronger object (e.g., RW+TAS)
- Adversarial models specifying the possible correct sets (non-uniform/correlated faults)



Model $\mathcal{A}$ corresponds to an affine tasks $R_{\mathcal{A}}$ (a subset of $\chi^2(\mathtt{I})$)



A task $(\mathtt{I},O,\Delta)$ is solvable in model $\mathcal{A}$ if and only if there is a chromatic simplicial map from a subdivision $R_{\mathcal{A}}{}^r(\mathtt{I})$ to O carried by Δ

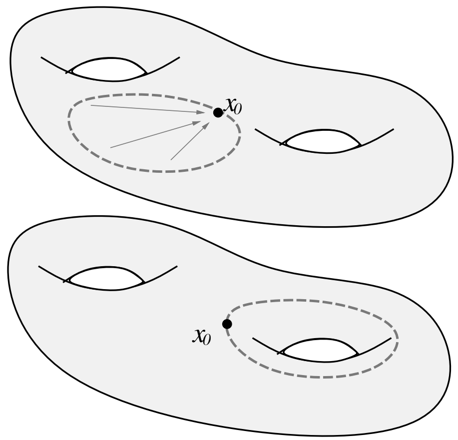
Automatic proofs: decidability?

Can we devise an algorithm to tell whether a task is wait-free solvable?

No

3-process wait-free task solvability is undecidable [GK95,HR97]

Loop agreement task is equivalent to loop contractibility (undecidable)



Under the rug…

- Is $\chi^r(\mathtt{I})$ a subdivision?
  - Yes! [Lin11,Koz15]
  - RW is a subdivision in general [AG09]
- Isn't the Iterated IS model weaker than read-write?
  - Not for task solvability: [BG93,BG97,GR10]
- Proof of ACT?
  - König's lemma

Takeaways/open questions

- Geometrical structure captures the computational power of a model
  - ✓Combinatorial vs. Operational
- Other problems/models?
  - ✓Long-lived abstractions (queues, hash tables, TMs…)
  - ✓Byzantine adversary: a faulty process deviates arbitrarily
  - ✓Partial synchrony

- Complexity bounds?
- Mathematics induced by DC?

Distributed jungle

Byzantine

Blockchain!

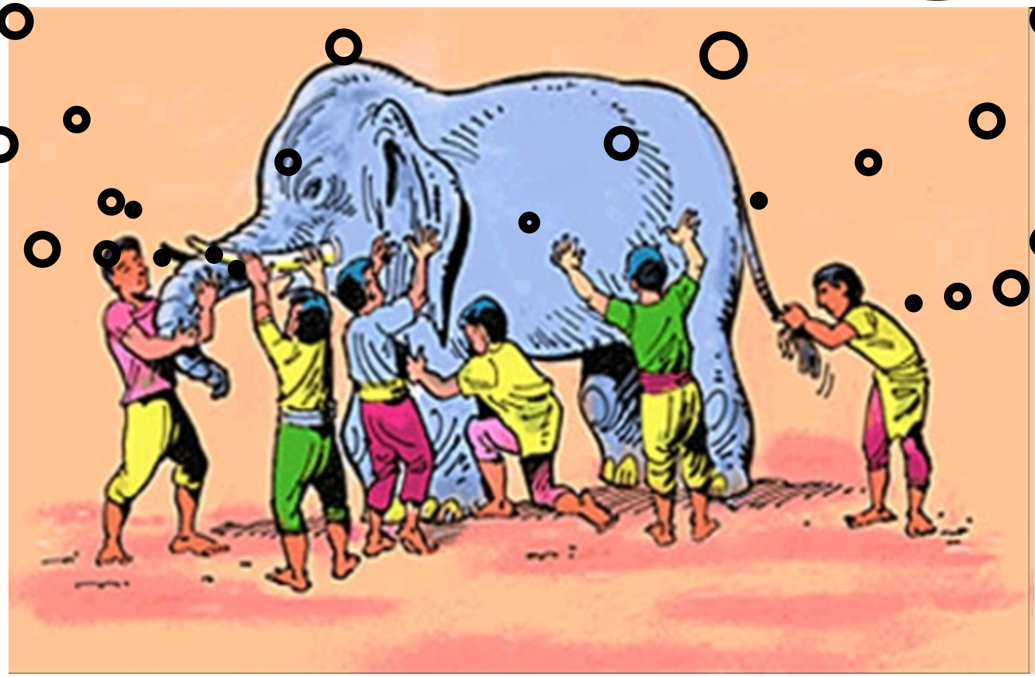
Message-passing

CAS

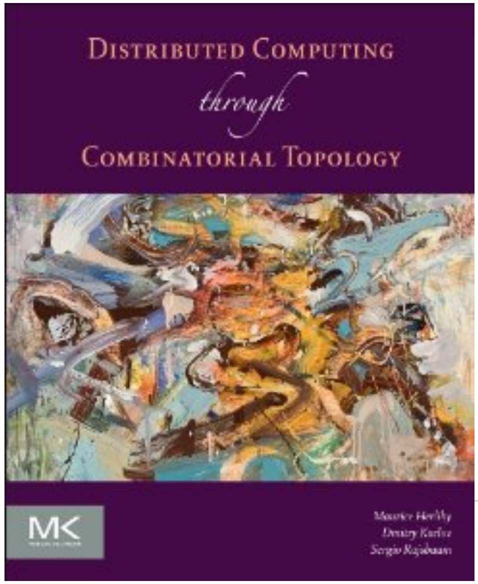
Shared-memory

Locks

Adversaries



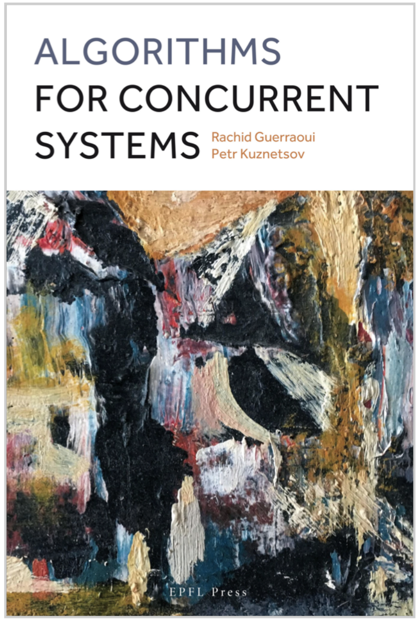
Distributed computability theory?



Distributed Computing through Combinatorial Topology

Maurice Herlihy, Dmitry Kozlov, Sergio Rajsbaum

Morgan Kaufman, 2013



Algorithms for Concurrent Systems

Rachid Guerraoui, Petr Kuznetsov

EPFL Press, 2019

Slides and exercises: https://perso.telecom-paristech.fr/kuznetso/CIRM2019


distributed
computing
combinatorial
models
Distributed
representative
properties
executions
diversity
structures
simplicial
corresponding
complexity
Combinatorial
systems
sets
basics
topology
Computing
analysis
model
capturing
power
complexes
discuss
via
mathematical
characterization
Structures
tools
Models
computability

Questions?