Why Extension-Based Proofs Fail

Faith Ellen, University of Toronto

Research done with

Dan Alistarh, James Aspnes, Rati Gelashvili, and Leqi Zhu

Consensus Problem

Each process p_i has a private input value $x_i \in \{0,1\}$.

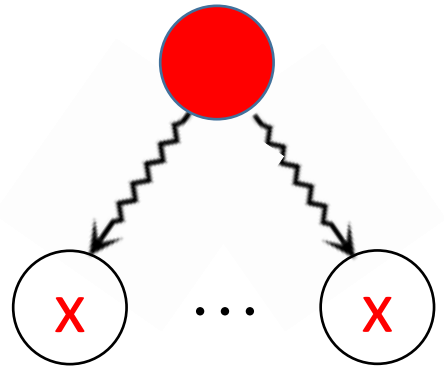
- Agreement: All output values are the same.
- Validity: The output value of each process is the input value of some process.
- Wait-Free Termination: Each non-faulty process outputs a value after taking a finite number of steps.

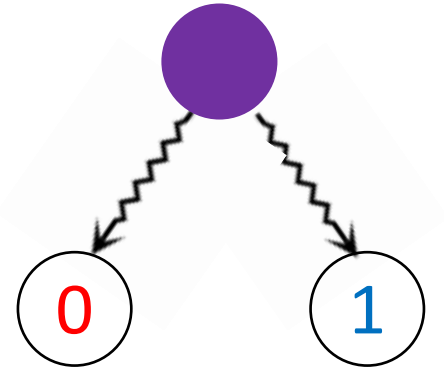
Impossibility of Consensus [FLP83]

A configuration C is:

x-univalent if all executions starting from C output x and

bivalent if there are two executions starting from C that output different values.

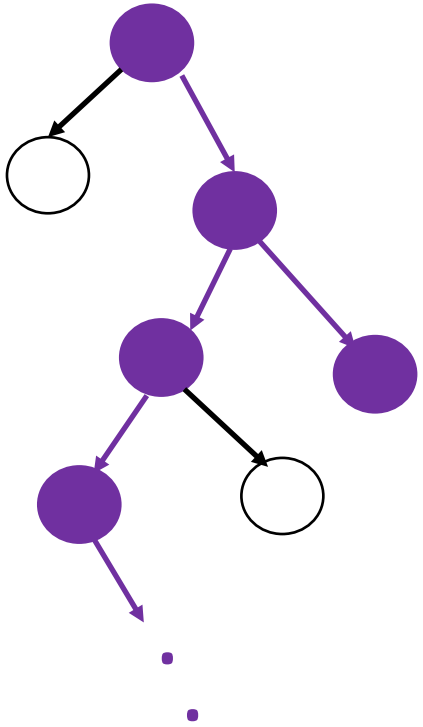




Impossibility of Consensus [FLP83]

LEMMA 1 Every consensus algorithm has a bivalent initial configuration.

LEMMA 2 From every bivalent configuration, there is a step that leads to a bivalent configuration.



This implies there is an infinite execution, consisting of only bivalent configurations, violating wait-free termination.

k-set Agreement Problem

Each process p_i has a private input value $x_i \in \{0,1,..,k\}$.

- Agreement: At most k different values are output.
- Validity: The output value of each process is the input value of some process.
- Wait-Free Termination: Each non-faulty process outputs a value after taking a finite number of steps.

Impossibility of k-set Agreement [BG,HS,SZ93]

The proofs of this result are MUCH more complicated.

Is there an extension based proof of this result?

Impossibility of k-set Agreement [BG,HS,SZ93]

The proofs of this result are MUCH more complicated.

Is there an extension based proof of this result? NO!

Outline

- Model of Computation
- Topological View of a Protocol in this model
- Definition of Extension-based proof
- Prove that there is no extension based proof of the impossibility of k-set agreement for more than k processes

Non-uniform Iterated Immediate Snapshot (NIIS) Model [HS99]

- There are n processes $p_0,\ldots,p_{n-1}$.
- Processes communicate using an infinite sequence $S_1,S_2,\ldots$ of snapshot objects, each with n components which are all initially -.
- Initially, p_i's state is its input.
- In its $2r-1$'st step, p_i updates component i of S_r with its state.
- In its $2r$'th step, p_i atomically scans S_r to get the values of all n components and updates its state to be this vector.

Non-uniform Iterated Immediate Snapshot (NIIS) Model [HS99]

- There is a decision function Δ that maps each pair (process, state) to either an output value or the special symbol $\perp$.
- If Δ maps (p_i, s) to $y \neq \perp$, then p_i outputs y in state s and terminates.
- If Δ maps (p_i, s) to $\perp$, this indicates that p_i hasn't yet decided in state s.

A protocol for a task in the NIIS model is completely specified by its decision function Δ.

Non-uniform Iterated Immediate Snapshot (NIIS) Model [HS99]

- An adversarial scheduler decides the order in which processes take steps.
- It repeatedly selects a set of processes that are all poised to perform updates on the same snapshot object, schedules all of these updates and then schedules all of their next scans.

$\{p_0,p_1,p_2\}\{p_3,p_6\}\{p_2\}\{p_4,p_5\}$

Non-uniform Iterated Immediate Snapshot (NIIS) Model [HS99]

- An adversarial scheduler decides the order in which processes take steps.
- It repeated selects a set of processes that are all poised to perform updates on the same snapshot object, schedules all of these updates and then schedules all of their next scans.
- Without loss of generality, the scheduler can schedule all operations on S_r before any operation on S_{r+1} for all $r \geq 1$.

$\{p_0,p_1,p_2\}\{p_3,p_7\}\{p_2\}\{p_4,p_5\}$ is indistinguishable from
$\{p_0,p_1,p_2\}\{p_3,p_7\}\{p_4,p_5\}\{p_2\}$

Non-uniform Iterated Immediate Snapshot Model [HS99]

- A configuration consists of the state of each process and the contents of each snapshot object at some point in a schedule.
- If a process has not terminated, but it is not subsequently scheduled, (i.e. the scheduler does not ever select it again), then the process is considered to have crashed.
- A process is active if it has not terminated or crashed.

Non-uniform Iterated Immediate Snapshot (NIIS) Model [HS06]

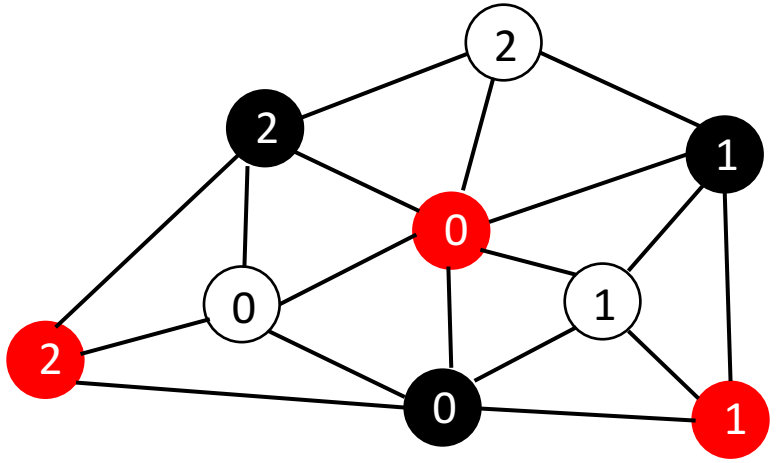
- This model is computationally equivalent to the standard asynchronous shared memory model, in which processes communicate by reading from and writing to shared registers.

- Any task that can be solved by a wait-free protocol in one of these models can be solved by a wait-free protocol in the other.

- It is much easier to use tools from combinatorial topology in the NIIS model.

Some Basic Combinatorial Topology

A simplicial complex σ is a collection of sets that is closed under subset, i.e. if $X \in \sigma$ and $Y \subseteq X$, then $Y \in \sigma$.

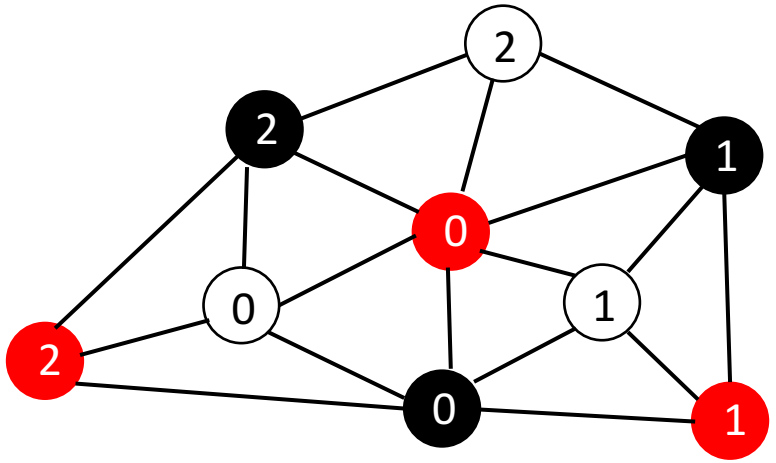


- A set in σ is called a simplex.
- A simplex of size 1 is called a vertex.
- A simplex of size 2 is called an edge.

Topological View of a Configuration

A simplicial complex σ is a collection of sets that is closed under subset, i.e. if $X \in \sigma$ and $Y \subseteq X$, then $Y \in \sigma$.

Each vertex represents a process (denoted by a colour) and a state of that process (denoted by a label).



Part of the input complex for 2-set agreement among 3 processes

A reachable configuration C is represented by the set of vertices corresponding to the processes that have not crashed in C and the states of those processes in C.

The input complex represents all initial configurations.

Topological View of a Protocol

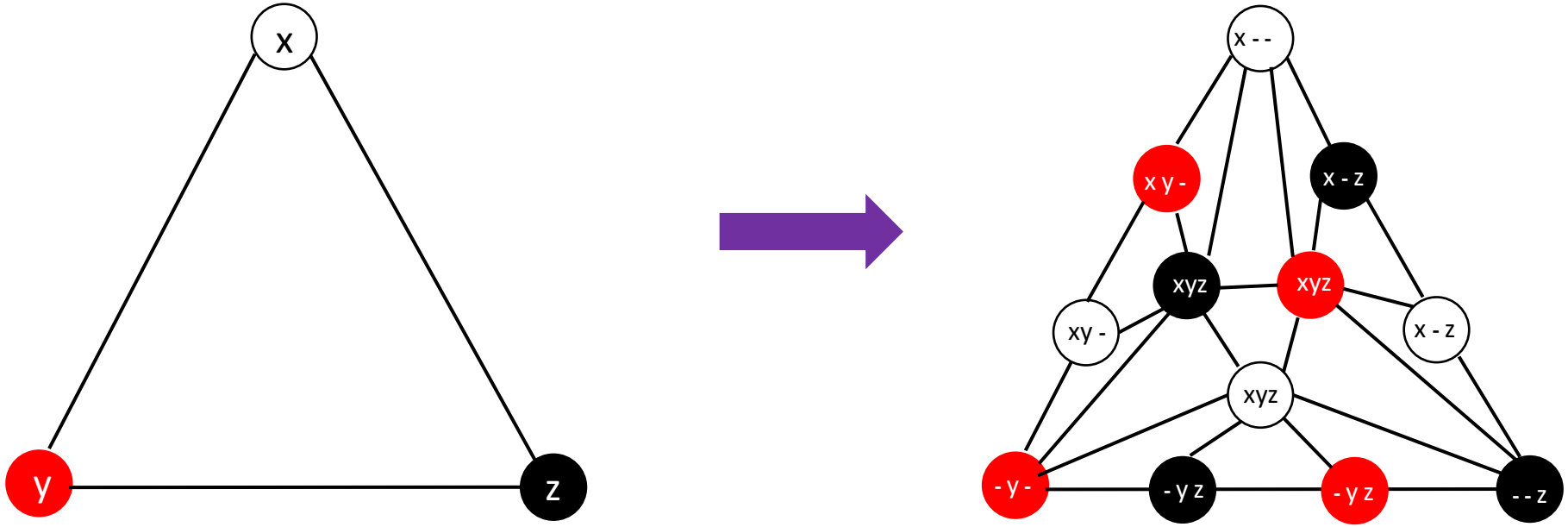
The simplicial complex of a protocol represents the set of all reachable configurations of the protocol in which no processes are active, i.e. every process has crashed or terminated.

A protocol is wait-free if every process terminates after being scheduled a finite number of times.

The simplicial complex of a wait-free protocol in the NIIS model has a special structure.

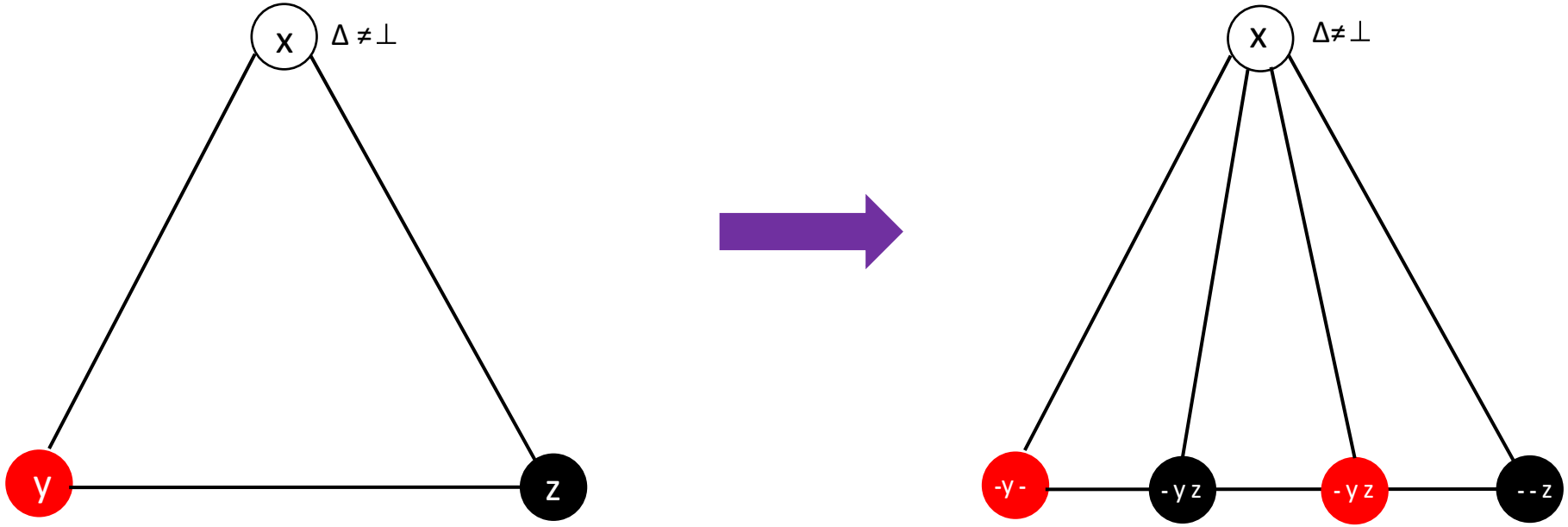
Chromatic Subdivision of a Simplex

Consider a simplex representing a configuration reached by a schedule in which every active process has been selected the same number of times.
Consider all schedules from this configuration in which every active process is selected at most once. The set of simplexes representing the resulting configurations is called the chromatic subdivision of the simplex.



Chromatic Subdivision of a Simplex

Consider a simplex representing a configuration reached by a schedule in which every active process has been selected the same number of times.
Consider all schedules from this configuration in which every active process is selected at most once. The set of simplexes representing the resulting configurations is called the chromatic subdivision of the simplex.



Non-Uniform Chromatic Subdivision of a Simplicial Complex

The non-uniform chromatic subdivision of a simplicial complex is obtained by replacing every set in the simplicial complex by its chromatic subdivision.

Let $\mathbb{Q}_0$ be the input simplicial complex for a protocol in the NIIS model.
Let $\mathbb{Q}_1$ be the non-uniform chromatic subdivision of $\mathbb{Q}_0$.

Then $\mathbb{Q}_1$ represents all configurations of the protocol reachable from an initial configuration by a schedule in which each active process is selected exactly once.

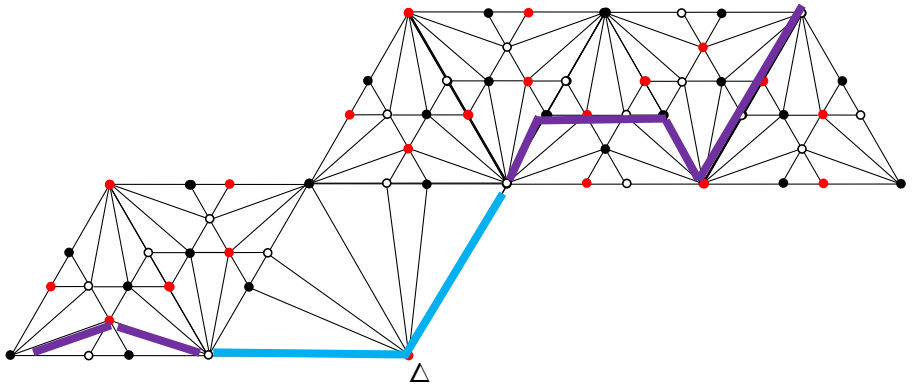
Topological View of a Protocol in the NIIS Model

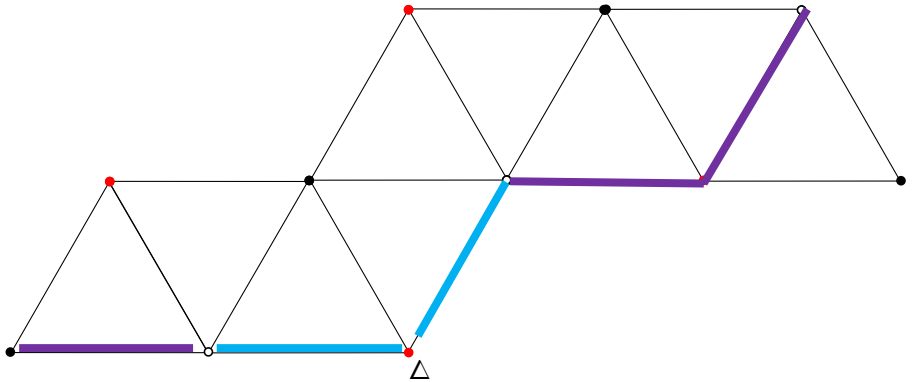
Let $\mathbb{Q}_0$ denote the input complex for a protocol in the NIIS model.
For $r \geq 0$, let $\mathbb{Q}_{r+1}$ denote the non-uniform chromatic subdivision of $\mathbb{Q}_r$.
Then $\mathbb{Q}_{r+1}$ represents all configurations of the protocol reachable from a configuration corresponding to a simplex in $\mathbb{Q}_r$ by a schedule in which each active process is selected exactly once.

Consider a protocol in the NIIS model in which every process terminates after it has been selected at most r times.
Then $\mathbb{Q}_r$ is the simplicial complex of the protocol specified by Δ.

LEMMA 1

Let $\mathbb{B}_r$ and $\mathbb{D}_r$ be disjoint subcomplexes of $\mathbb{Q}_r$.
Let $\mathbb{B}_{r+1}$ and $\mathbb{D}_{r+1}$ be the non-uniform chromatic subdivisions of $\mathbb{B}_r$ and $\mathbb{D}_r$ (which are subcomplexes of $\mathbb{Q}_{r+1}$).
If every path between a vertex of $\mathbb{B}_r$ and a vertex of $\mathbb{D}_r$ has length at least d and contains at least x edges between vertices that have not terminated, then every path between a vertex of $\mathbb{B}_{r+1}$ and a vertex of $\mathbb{D}_{r+1}$ has length at least d+x.





LEMMA 2

Let F be a simplex in $\mathbb{Q}_f$, where $f \geq 1$.
Let $\mathbb{F}$ be the largest subcomplex of $\mathbb{Q}_f$ which contains only vertices that are distance ≤ 1 from every vertex in F.
Then there exists an input value a such that every vertex in $\mathbb{F}$ contains a.

LEMMA 3

Let $\mathbb{F}$ be a subcomplex of $\mathbb{Q}_f$ and
let $\mathbb{F}'$ be the be the non-uniform chromatic subdivision of $\mathbb{F}$.
If every vertex in $\mathbb{F}$ contains a, then every vertex in $\mathbb{F}'$ contains a.

Extension-based proof

- Represented as an interaction between a prover and a protocol defined by a map Δ.
- Initially, the prover only knows the initial configurations of the protocol, where the state of each process is its input. $A(1)$ is this set of configurations. $A'(1)$ is initialized to ϕ.
- The prover may repeatedly query the protocol by choosing a configuration $C \in A(1) \cup A'(1)$ and selecting a set of processes P all poised to perform an update on the same snapshot object in C.
- For each $p_i \in P$, the protocol responds with $\Delta(p_i,s_i)$, where s_i is the state of p_i in the configuration C' resulting from scheduling P from C, and adds C' to $A'(1)$.

Extension-based proof

The prover wins (i.e. shows that the protocol is incorrect) if the protocol responds:

- inconsistently,
- with more than k different outputs in a configuration, or
- with an output a in a configuration that was obtained by a schedule from an initial configuration in which no process had a as its input.

Extension-based proof

- The prover may make a chain of queries $(C_0, P_0), (C_1, P_1), \ldots,$ such that, for each $i \geq 0$, C_{i+1} is the configuration resulting from scheduling P_i from C_i.
- If the protocol does not eventually terminate all of the processes in the chain, then the prover wins, since the protocol is not wait-free.
- After making finitely many chains of queries without winning, the prover must choose a configuration $C \in A'(1)$ and define $A(2)$ to be the set of all configurations C' such that, for all processes p_i, if p_i is in an initial state in C, then p_i is in an initial state in C' and if p_i is not in an initial state in C, then p_i is in the same state in C'.

Extension-based proof

The prover wins if the protocol responds:

- inconsistently,
- with more than k different outputs in a configuration, or
- with an output y in a configuration that was obtained by a schedule from an initial configuration in which no process had y as its input.

The prover wins if it asks an infinite chain of queries or there are an infinite number of phases.

The prover loses if $A'(r) = \phi$ at the end of some phase r.

There is no Extension-Based Proof for the Unsolvability of k-set agreement

To prove that k-set agreement is unsolvable using an extension-based proof, a prover must win against every protocol.

To show that there is no extension based proof, we give an adversarial protocol that is adaptively constructing the map Δ, and show that every prover loses against this protocol.

A Strategy for an Adversarial Protocol

Initialization:

- Construct the input complex $\mathbb{Q}_0$ for k-set agreement.
- For $i = 0,1,2,3$ define $\Delta(v) \leftarrow \perp$ for each $v \in \mathbb{Q}_i$ and construct the non-uniform chromatic subdivision $\mathbb{Q}_{i+1}$.
- For all $v \in \mathbb{Q}_4$ that contain only one input value a, define $\Delta(v) \leftarrow a$.
- Set $t \leftarrow 4$.

Invariants maintained during phase 1:

- For each $0 \leq i < t$ and for each vertex $v \in \mathbb{Q}_i$, $\Delta(v)$ is defined.
- For each vertex in a configuration in $A(1) \cup A'(1)$, $\Delta(v)$ is defined.
- If $v, v' \in \mathbb{Q}_t$ are the states of processes that have terminated with different values, then the distance between v and v' is at least 6.
- If $v \in \mathbb{Q}_t$ is the state of a process that has terminated with value a, and $u \in \mathbb{Q}_t$ is a state that does not contain a, then the distance between v and u is at least 6.

A Strategy for the Adversarial Protocol

During phase 1, when the prover performs a query scheduling a set of processes P from (a previously explored) configuration C, resulting in configuration C':

- Let v and v' be the states of some process $p \in P$ in C and C'. If $v \in \mathbb{Q}_t$, but $v' \notin \mathbb{Q}_t$, then
  - for all vertices $u \in \mathbb{Q}_t$ where $\Delta(u)$ is undefined, define $\Delta(u) \leftarrow \perp$,
  - construct the non-uniform chromatic subdivision $\mathbb{Q}_{t+1}$, and
  - increment t.
- For all $v' \in C'$ such that $\Delta(v')$ is undefined:
  - If v' has a neighbour $u' \in \mathbb{Q}_t$ that has terminated, let $a = \Delta(u')$;
  - otherwise, let a = minimum value contained in state v'.
  - If there exists a vertex in $\mathbb{Q}_t$ at distance at most 5 from v' that does not contain a or is the state of a process that has decided a value other than a, define $\Delta(v') \leftarrow \perp$;
  - otherwise, define $\Delta(v') \leftarrow a$.

LEMMA There is no infinite chain of queries in phase 1.

A Strategy for the Adversarial Protocol

At the beginning of phase 2:

Let $f \geq 1$ denote the maximum number of times that some process has been scheduled in any configuration $C \in A(2)$.

- Define $\Delta(u) \leftarrow \perp$ for all vertices $u \in \mathbb{Q}_t$ where $\Delta(u)$ is undefined, construct the non-uniform chromatic subdivision $\mathbb{Q}_{t+1}$, and increment t.
- Let F be the vertices in $\mathbb{Q}_f$ that are states of the processes in C which have been scheduled f times.
- Let $\mathbb{F}$ be the largest sub-complex of $\mathbb{Q}_f$ that contains only vertices at distance ≤ 1 from every vertex in F. By Lemma 2, there is an input value a such that every vertex in $\mathbb{F}$ contains a.
- Let $\mathbb{F}'$ be the sub-complex of $\mathbb{Q}_t$ obtained from $\mathbb{F}$ by performing $t-f$ non-uniform chromatic subdivisions. By Lemma 3, every vertex in $\mathbb{F}$ contains a.

A Strategy for the Adversarial Protocol

Note that $\Delta(u) \neq \perp$ for every vertex in $\mathbb{Q}_t$.

By the Invariant, the distance is at least 6 between any $v, v' \in \mathbb{Q}_t$ that are the states of processes terminating with different values v .

Define $\Delta(u) \leftarrow a$ for all vertices $u \in \mathbb{F}'$ where $\Delta(u)$ is undefined.

Then, in every configuration corresponding to a simplex in $\mathbb{F}'$, all processes have terminated.

In every configuration corresponding to a simplex in $\mathbb{Q}_t$, at most 2 different values have been output.

In phases 2 and greater, the prover can only explore configurations reachable from A(2), so any sufficiently long schedule results in a configuration in $\mathbb{F}'$.

Thus, there are no infinite chains and the prover must eventually choose a configuration in $\mathbb{F}'$ at the end of some phase. At this point, the prover loses.

Extensions to extension-based proofs

Allow the prover to perform additional queries:

- Is there a schedule starting from a specific explored configuration in which some process answers value a?
- Provide a schedule starting from a specific explored configuration in which some process answers value a.
- What is the length of the longest schedule that can occur in the protocol? The prover can use the answer to such a query only when it has reached a configuration by a schedule that is longer than this, but not to decide which configurations to explore.

Covering arguments cannot be used to obtain space lower bounds for k-set agreement.