Geometry and Distributed Systems

Eric Goubault

CEA/Saclay

Eric.Goubault@cea.fr

http://www.di.ens.fr/~goubault

Based on work by M. Herlihy, S. Rajsbaum, N. Shavit...

Aim of the talk

Can we implement some functions on some distributed architecture, even if there are some crashes?

> Example: consensus on an asynchronous system
>
> NO: FLP'85!

- There is a nice "geometrization" of the problem
- We will solve easy problems to make you understand
- But it has also solved some new problems!
- ... and this is an active research area!

Decision tasks

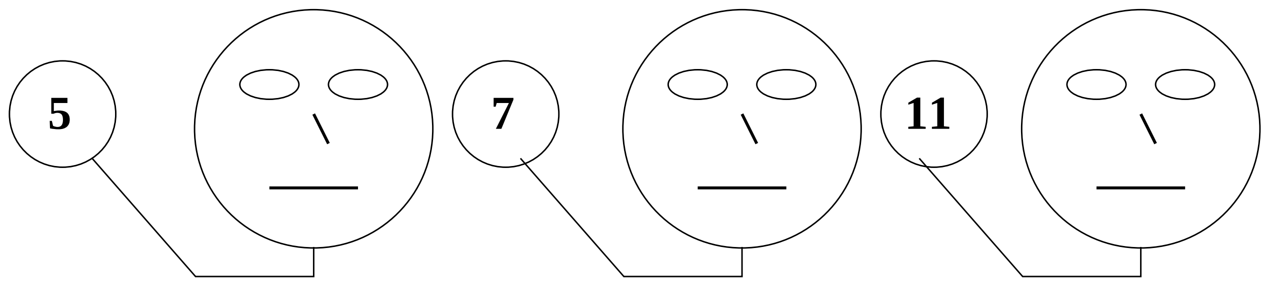
Can we implement a function...given an "architecture" (faults? shared memory / message passing, synchronous / semi-synchronous / asynchronous etc.)?

Each problem is given by:

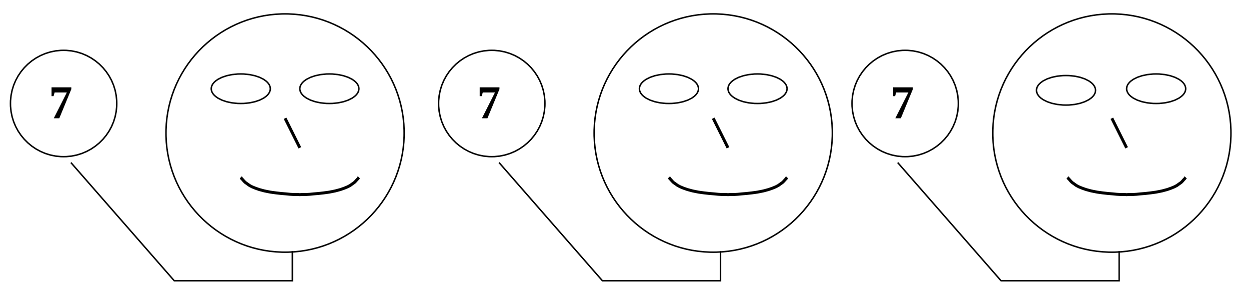
- For each processor $P_0, \ldots, P_{n-1}$ a set of possible initial values (in a domain $\mathcal{K} = \mathbb{N}$ or $\mathbf{R}$ etc.), i.e. a subset $\mathcal{I}$ of $\mathcal{K}^n$: "*input*"
- Similarly, we are given a set of possible final values $\mathcal{J}$ in $\mathcal{K}^n$: "*output*"
- Finally, we are given a map, the "*decision map*" $\delta : \mathcal{I} \to \wp(\mathcal{J})$ associating to each possible initial value, the set of authorized output values

Example: consensus

Before

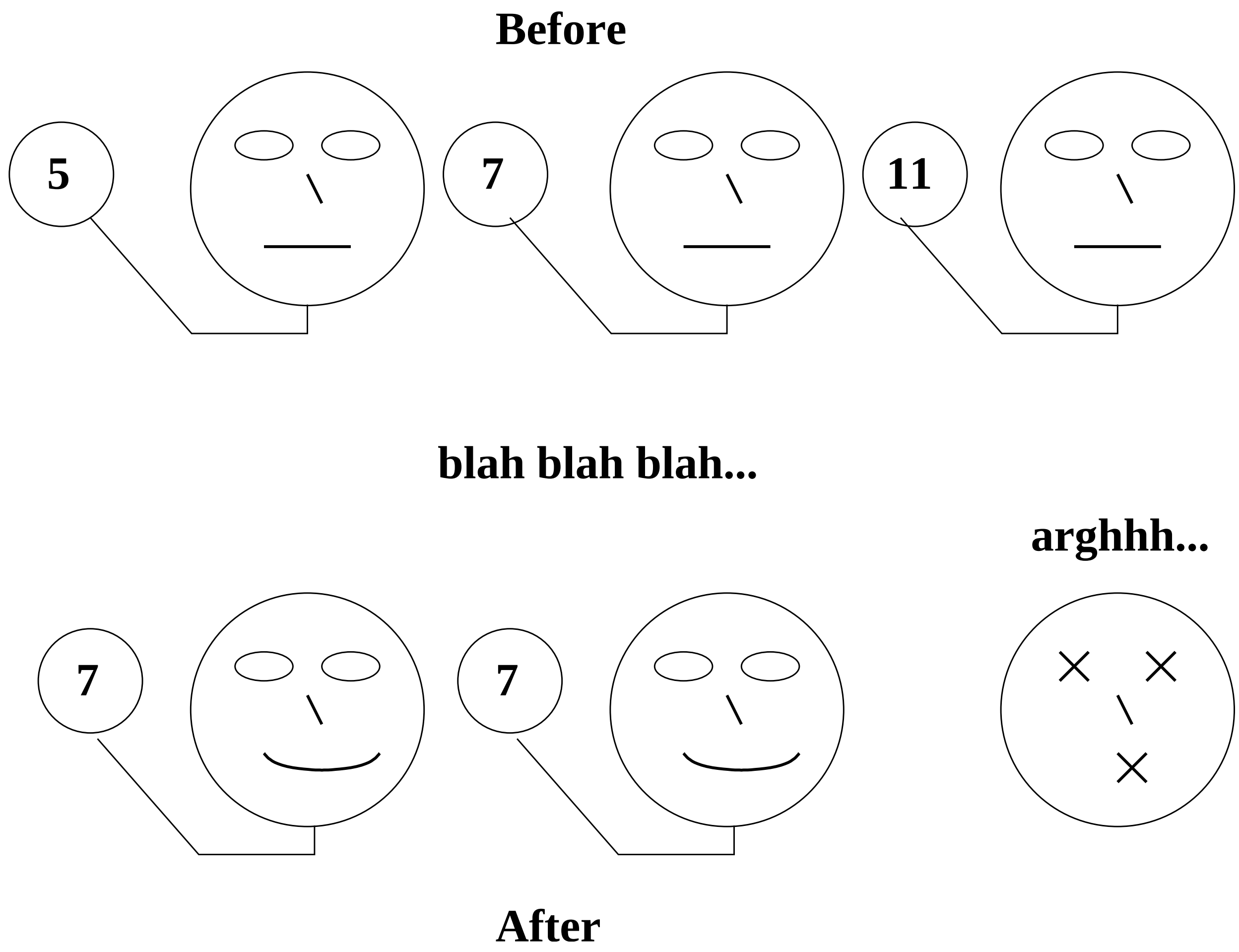


blah blah blah...



After

Even if...



Example

- $\mathcal{K} = \mathbb{N}, \mathcal{I} = \mathbb{N}^n$,
- $\mathcal{J} = \{(n, n, \ldots, n) \mid n \in \mathbb{N}\}$,
- $\delta(x_0, x_1, \ldots, x_{n-1}) = \left\{ \begin{array}{c} \{(x_0, x_0, \ldots, x_0), \\ (x_1, x_1, \ldots, x_1), \\ \ldots, \\ (x_{n-1}, x_{n-1}, \ldots, x_{n-1})\} \end{array} \right.$

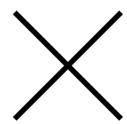
Main idea

- The *input* set and output sets have a geometrical structure (simplicial set)
- According to the architecture type, not all decision maps can be programmed
- There are geometrical constraints on the decision maps
- Very much like mainstream results in geometry, such as Brouwer's fixed point theorem...

Road map

- Input and output sets as simplicial sets (examples)
- Some basic algebraic topology
- The dynamics as sets of simplicial sets (protocol simplicial set, or complex)
- Some results and references

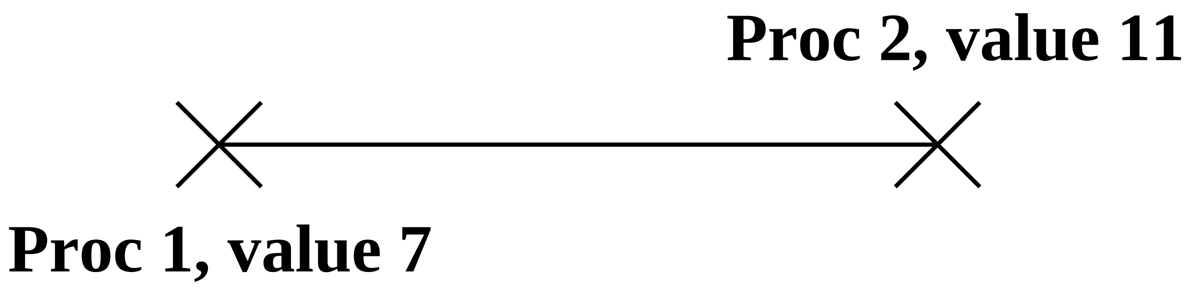
Simplicial model of states



Proc 1, value 7

(local state)

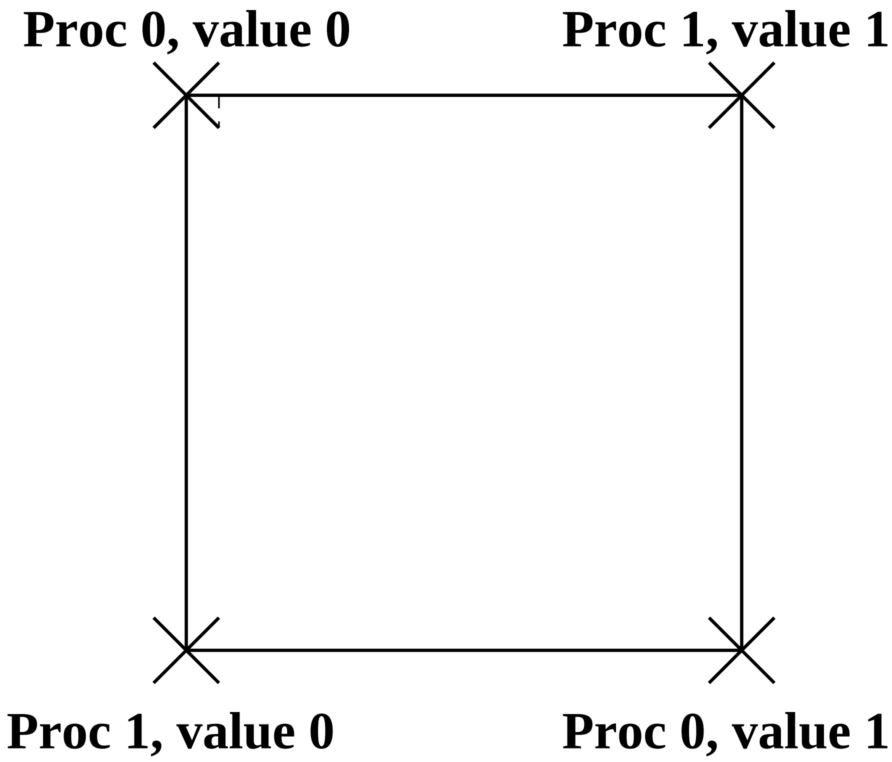
Simplicial model of states



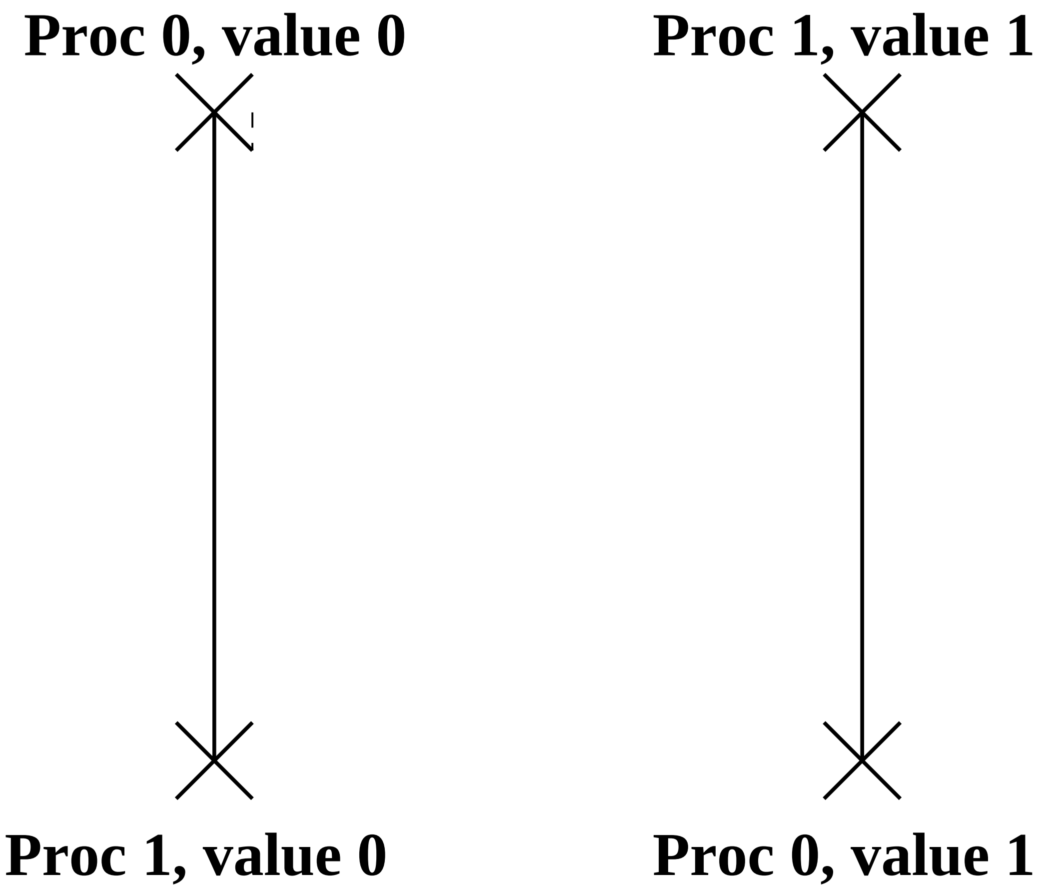
(compound state)

Initial states for (binary) consensus

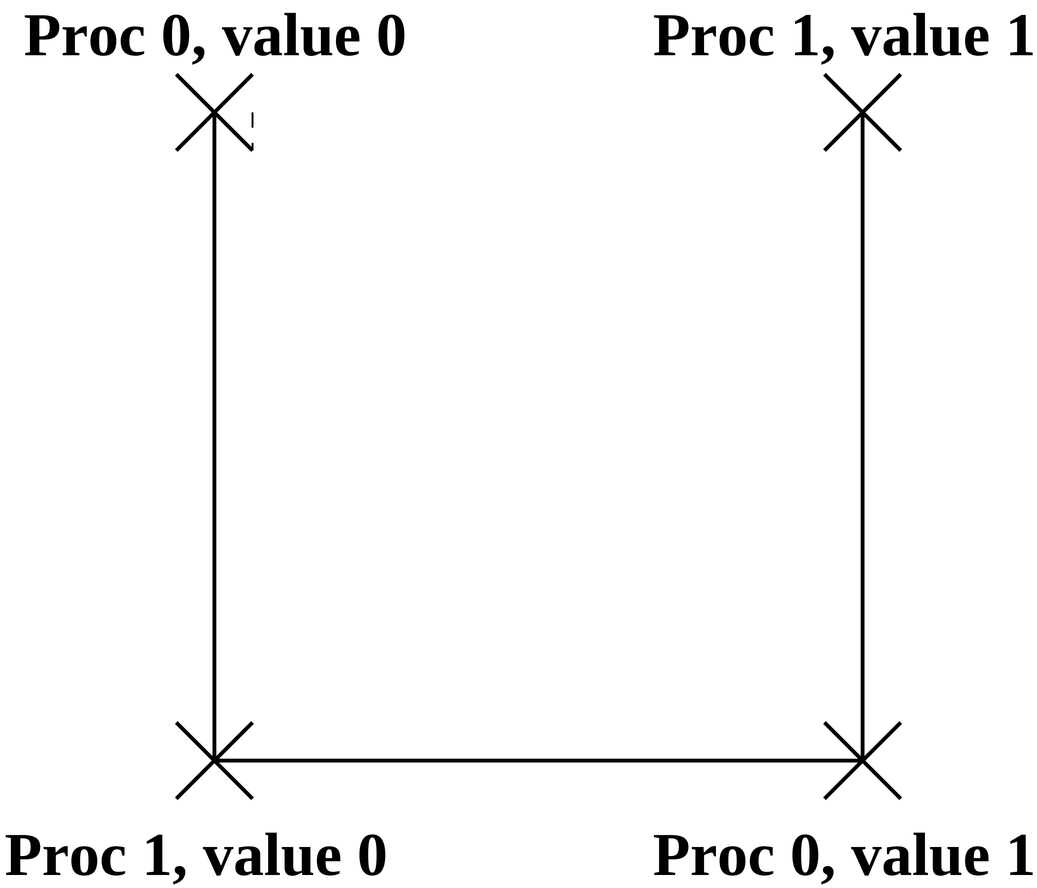
Here, 2 processors, i.e. dimension 2:



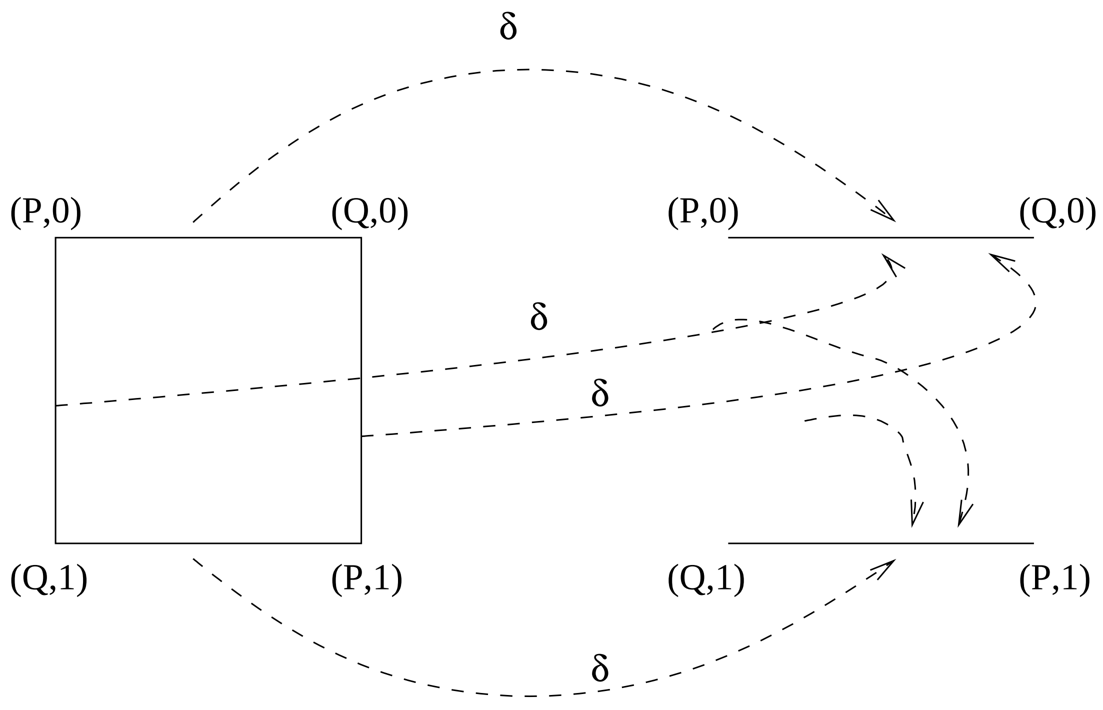
Final states for consensus



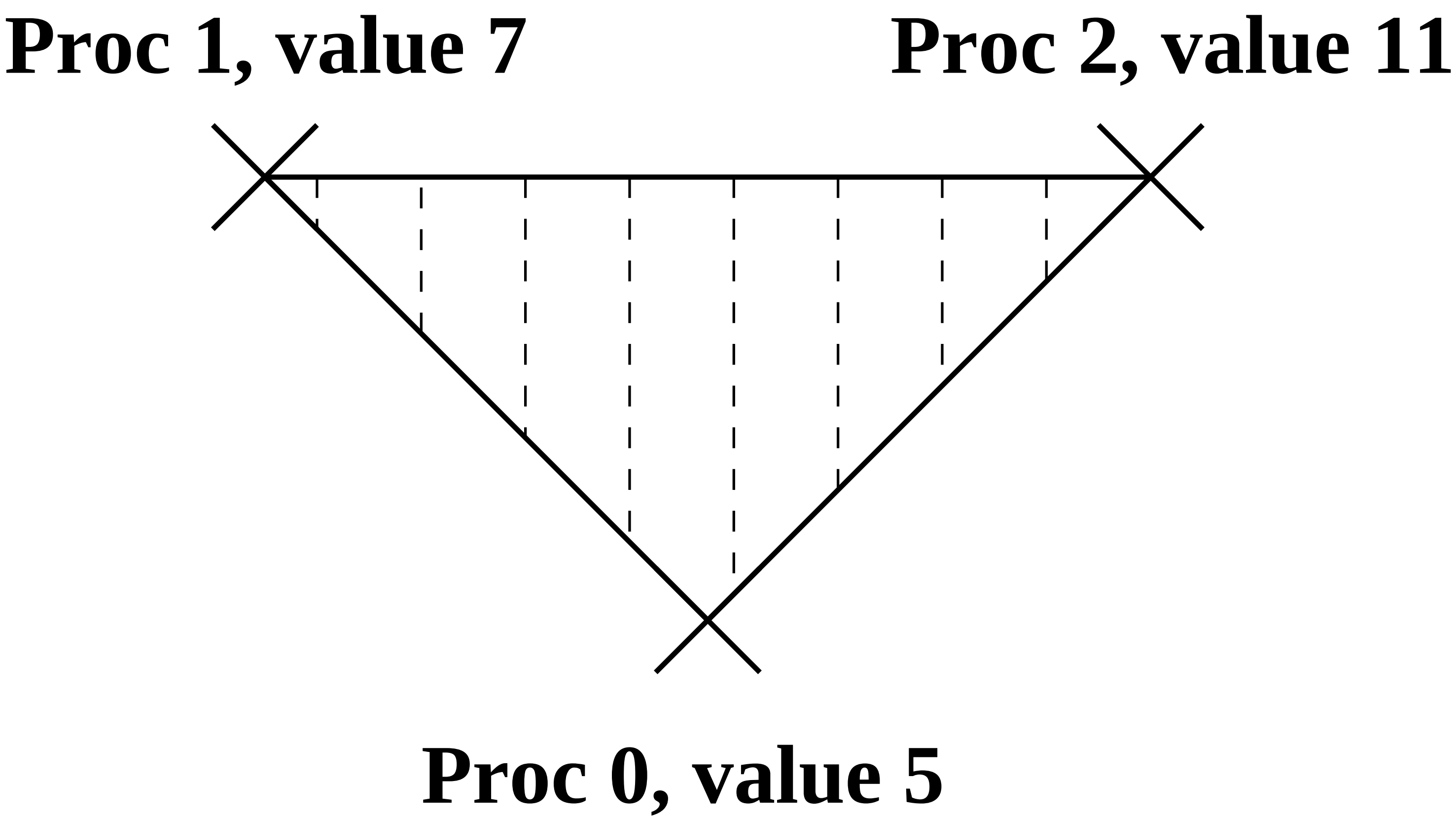
Final states for pseudo-consensus



Example: Consensus specification



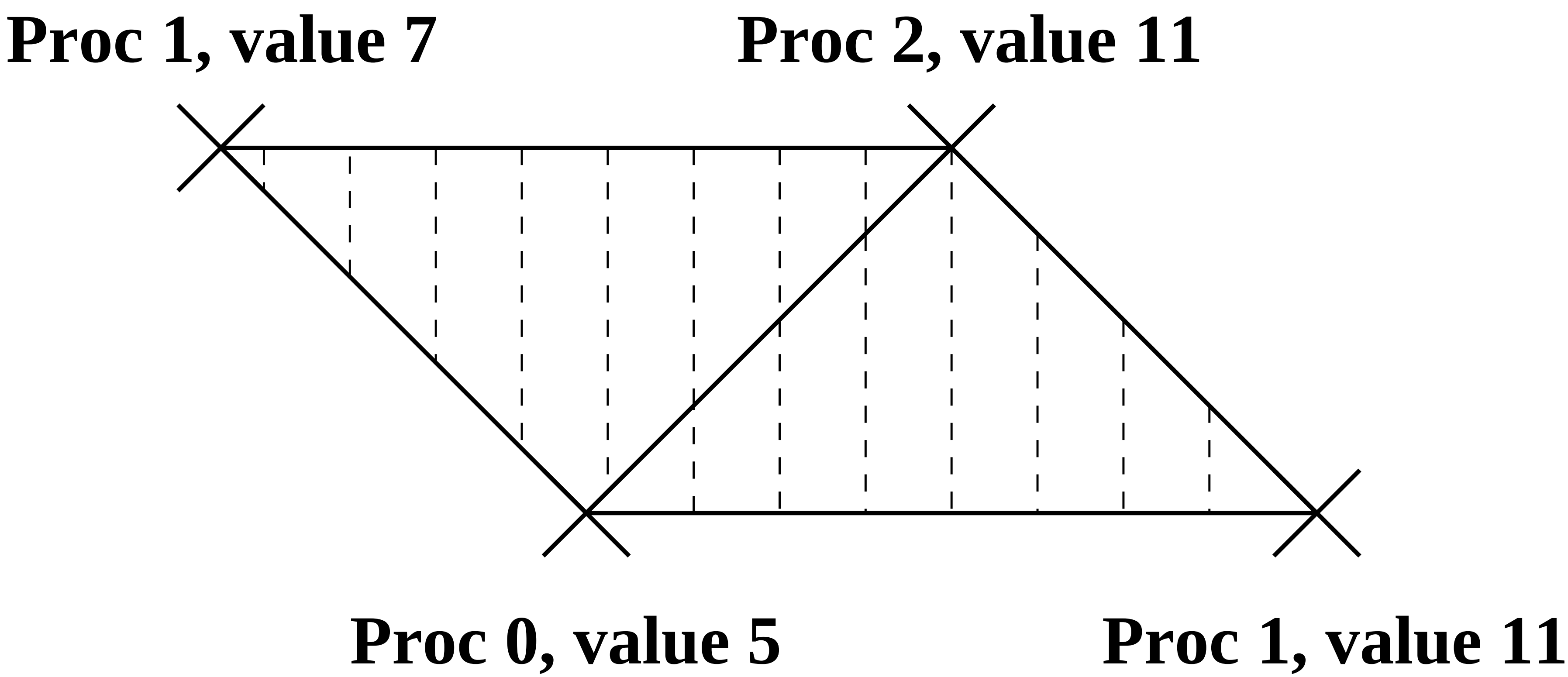
More generally: Simplicial model of states



(More generally [than a graph]: global state)

Example

Simplicial set=set of global states (with some common local states)



Simplicial complexes, formalization: n-simplex

- A n-simplex is just the convex hull of $n + 1$ points,
- For instance, a 0-simplex is a vertex, a 1-simplex is an edge, a 2-simplex is a triangle, a 3-simplex is a tetrahedron...
- A subsimplex (or face) of a n-simplex (defined by points $x_0, \ldots, x_n$) is any convex hull of a subset of points in $\{x_0, \ldots, x_n\}$.

Simplicial complex: A geometrical view

- Collection of *disjoint* simplices
- Glued along their faces (through “boundary relations”),
- Include, in dimension 1, all graphs.

Simplicial complex: A combinatorial view

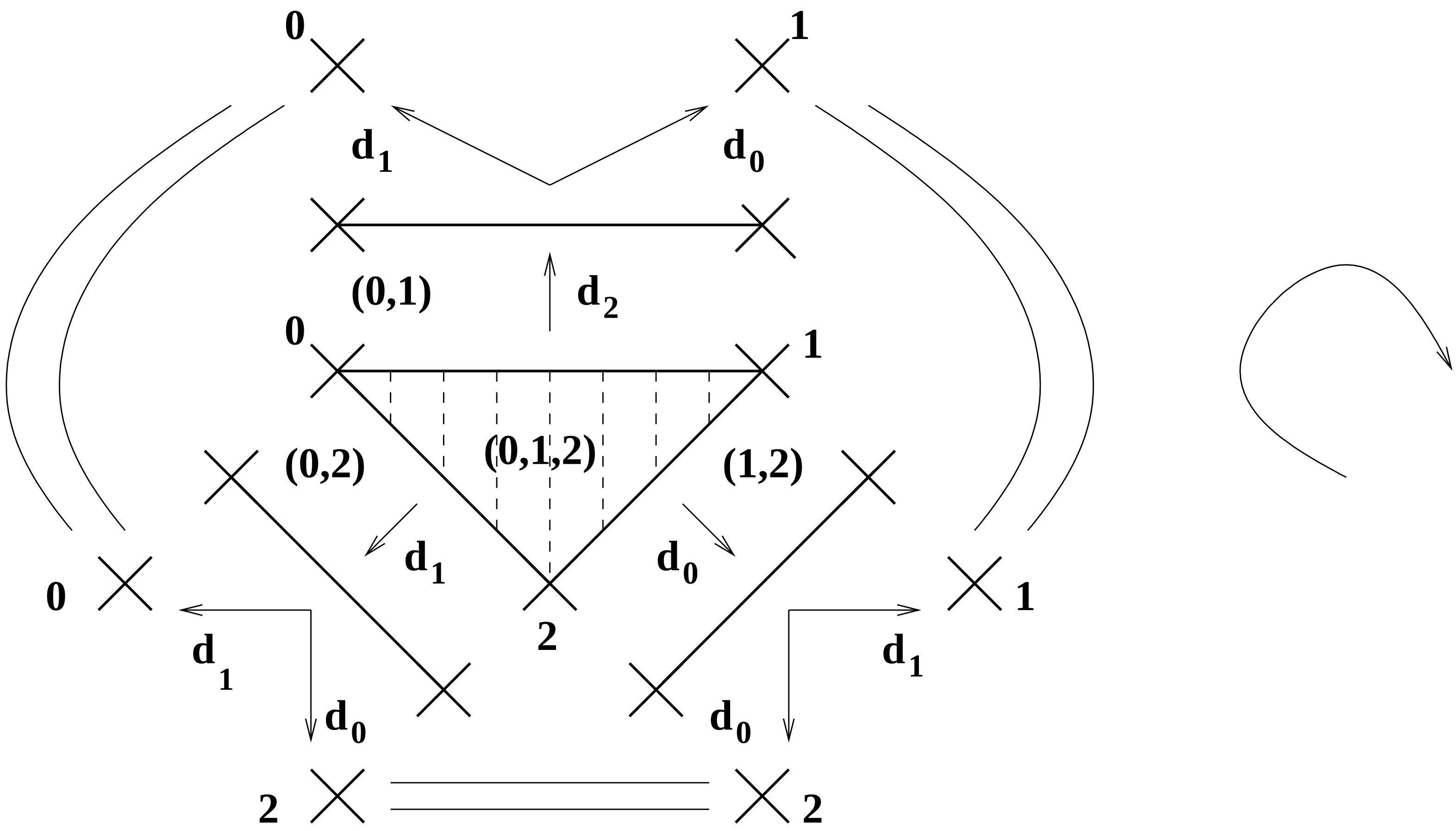
A simplicial set X_* is a family of sets X_n for $n \geq 0$ and maps called *faces* $d_i : X_n \to X_{n-1}$, $0 \leq i \leq n$, and *degeneracies* $s_j : X_n \to X_{n+1}$, $0 \leq j \leq n$, satisfying the following relations:

- $d_i d_j = d_{j-1} d_i$ for $i < j$;
- $s_i s_j = s_{j+1} s_i$ for $i \leq j$,
- $d_i s_j = \begin{cases} s_{j-1} d_i & \text{for } i < j, \\ \text{id} & \text{for } i = j,\ i = j+1, \\ s_j d_{i-1} & \text{for } i > j+1. \end{cases}$

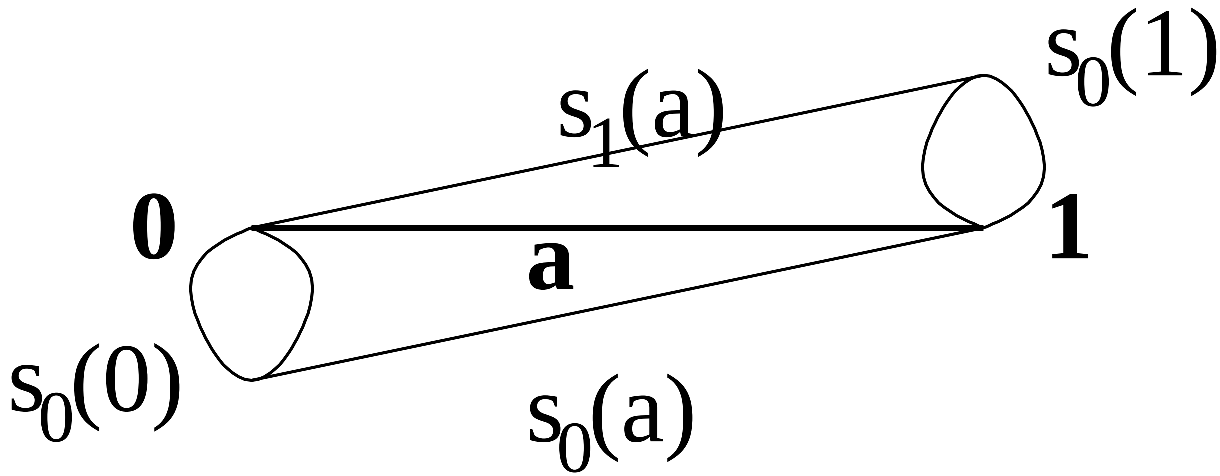
(if $u = (x_0, \ldots, x_n)$ is a n-simplex, set $d_i(u) = (x_0, \ldots, \hat{x}_i, \ldots, x_n)$ for its ith face - check the relations!)

Example

One can check the relations on the boundaries:



Example: degeneracies



Example

(of slide 16)

- $M_0 = \{P1V7, P0V5, P2V11, P1V11\}$,
- $M_1 = \{a, b, c, d, e\}$, $d_0(a) = P1V7$, $d_1(a) = P2V11$, $d_0(b) = P2V11$, $d_1(b) = P0V5$, $d_0(c) = P0V5$, $d_1(c) = P1V7$, $d_0(d) = P2V11$, $d_1(d) = P1V11$, $d_0(e) = P1V11$, $d_1(e) = P0V5$
- $M_2 = \{A, B\}$, $d_0(A) = a$, $d_1(A) = b$, $d_2(A) = c$, $d_0(B) = b$, $d_1(B) = d$, $d_2(B) = e$

The category of simplicial sets

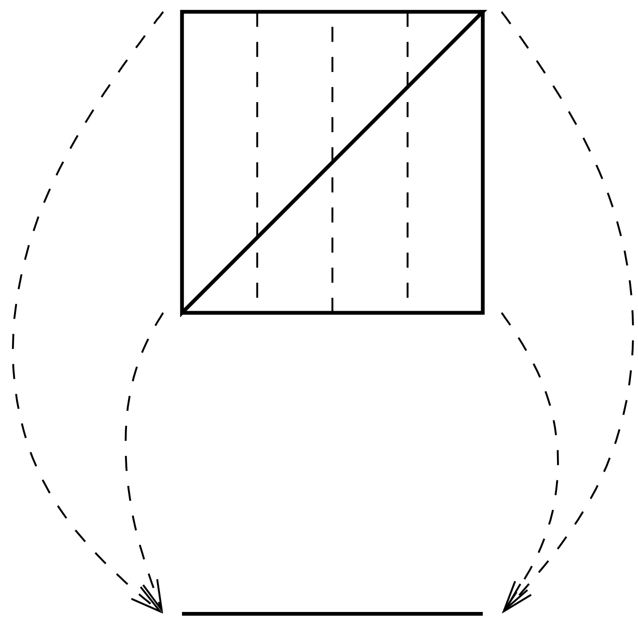
- A map $f : X_* \to Y_*$ (family of maps $f_n : X_n \to Y_n$) is called simplicial map if it commutes with faces and degeneracies:

$$\begin{array}{ccc} X_n & \xrightarrow{f_n} & Y_n \\ {\scriptstyle d_i}\downarrow & & \downarrow{\scriptstyle d_i} \\ X_{n-1} & \xrightarrow{f_{n-1}} & Y_{n-1} \end{array} \qquad \begin{array}{ccc} X_n & \xrightarrow{f_n} & Y_n \\ {\scriptstyle s_i}\downarrow & & \downarrow{\scriptstyle s_i} \\ X_{n+1} & \xrightarrow{f_{n+1}} & Y_{n+1} \end{array}$$

- A simplicial set can be viewed as a covariant functor from the category Δ of ordered finite sets and non-decreasing maps to the category of sets.

- Simplicial sets and simplicial maps form a category $\mathcal{S}$.

Example



Just a vertex to vertex map, extended to convex hulls (n-simplices)

Back to *protocols*

- Finite program
- Starts with input values
- Fixed number of rounds
- Halts with decision value

The full-information protocol is the one where the local value is the full history of communications

Generic protocol

```
s = empty;
for (i=0; i<r; i++) {
  broadcast messages;
  s = s + messages received;
}
return delta(s);
```

Example

Synchronous message passing; notion of round:

- at each round, every processor broadcasts its own value to the others
- in any order
- then every processor receives the broadcasted values and computes a new local value

Failure models

- crash (fail-stop),
- byzantine etc.

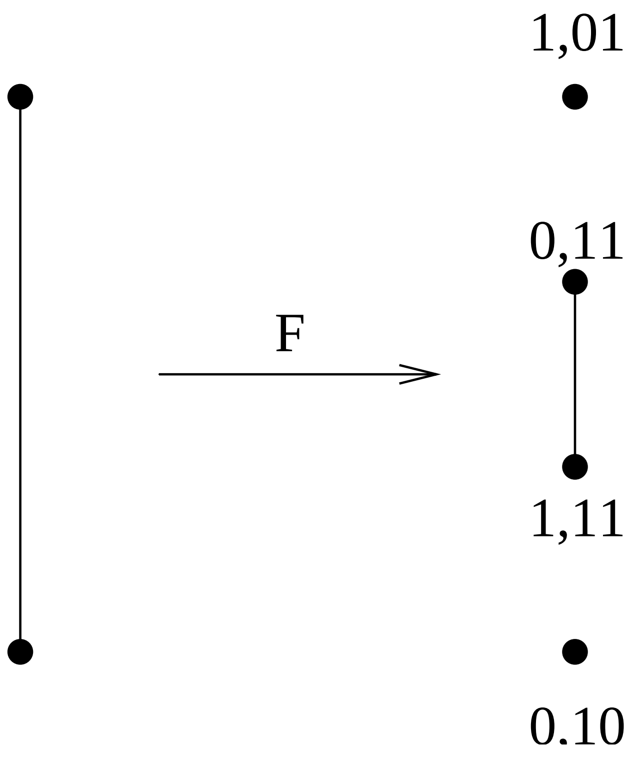
In what follows: crash failures only; can happen at any point of the broadcast, which can be done in any random order.

Protocol complex

Each protocol on some architecture defines:

- a simplicial set (for all rounds r):
  - vertices: sequence of messages received at a given round r
  - simplices: compound states at round r
- This is an operator on an input simplex
- A choice of model of computation entails some geometrical properties of the protocol complex

Synchronous protocol complex

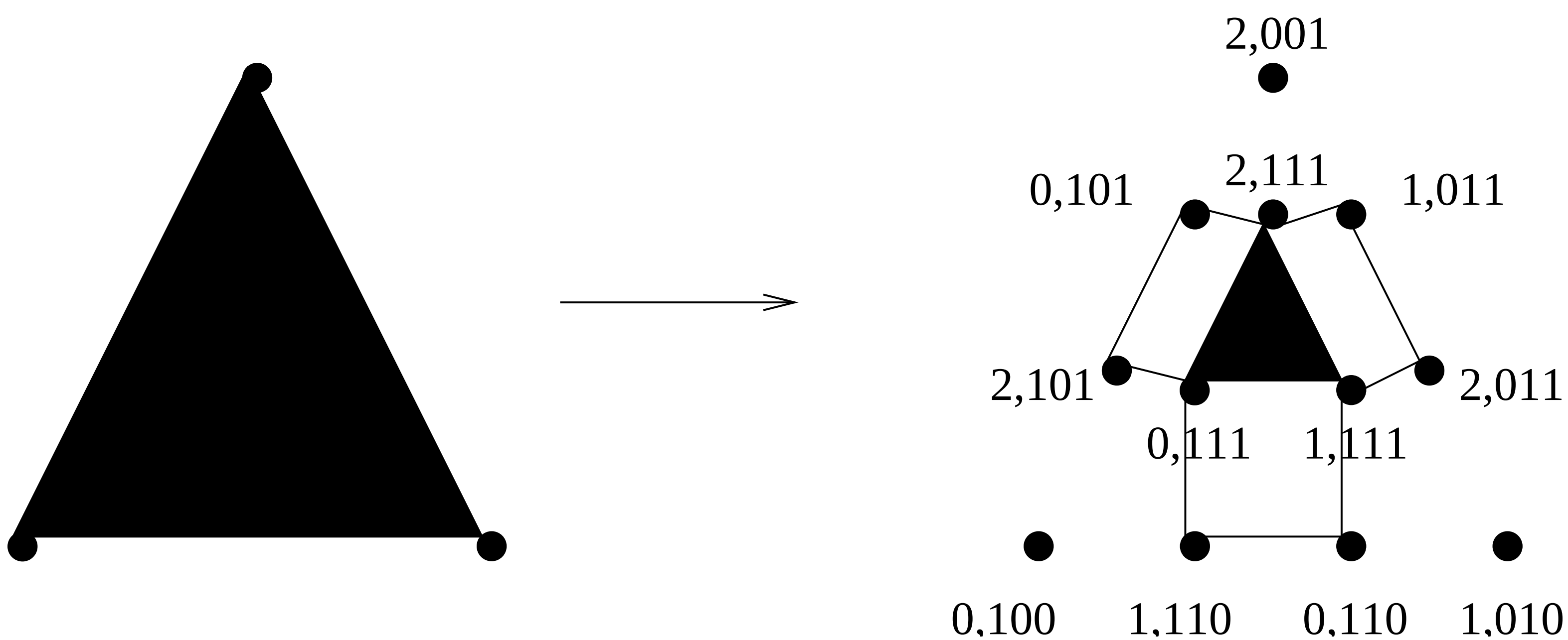


Explanation

In the synchronous model, at round 1:

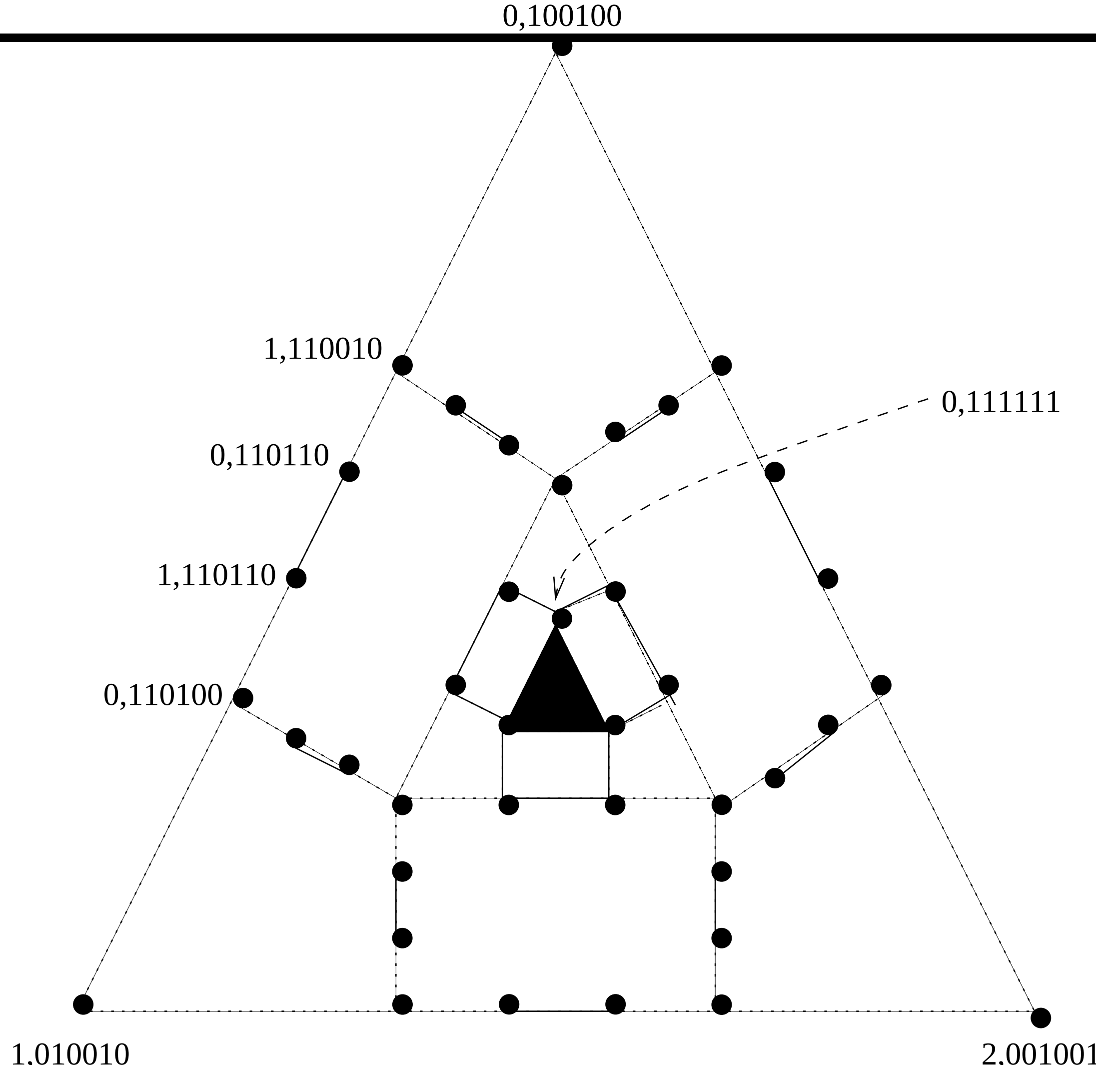
- no process has failed, hence everybody has received the message of the others (hence the central segment as global state)
- one process has failed, hence two points as possible states

Synchronous protocol complex

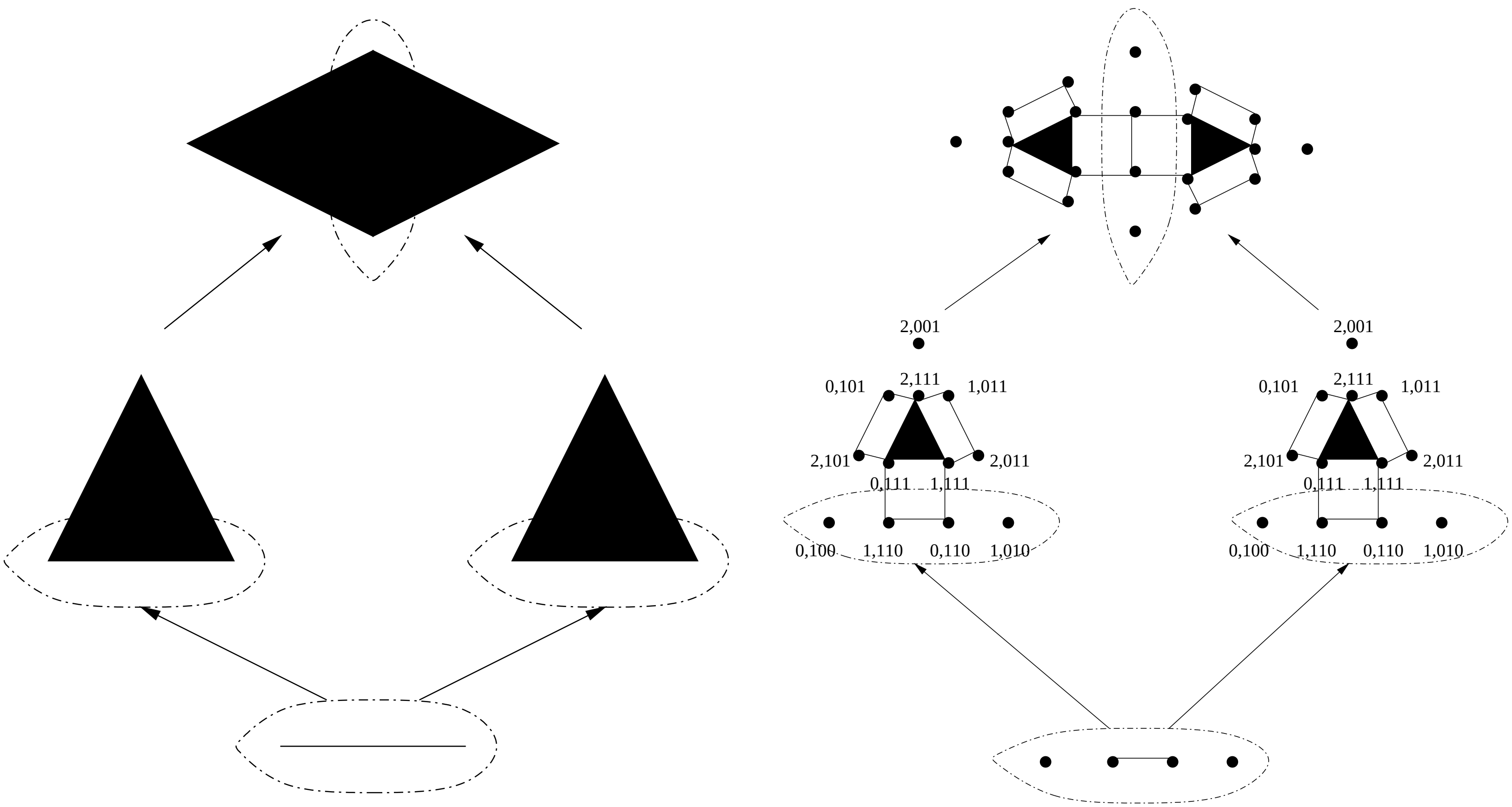


(wait-free - if up to 1 failure, forget the isolated points!)

Synchronous protocol complex - round 2



Synchronous protocol complex



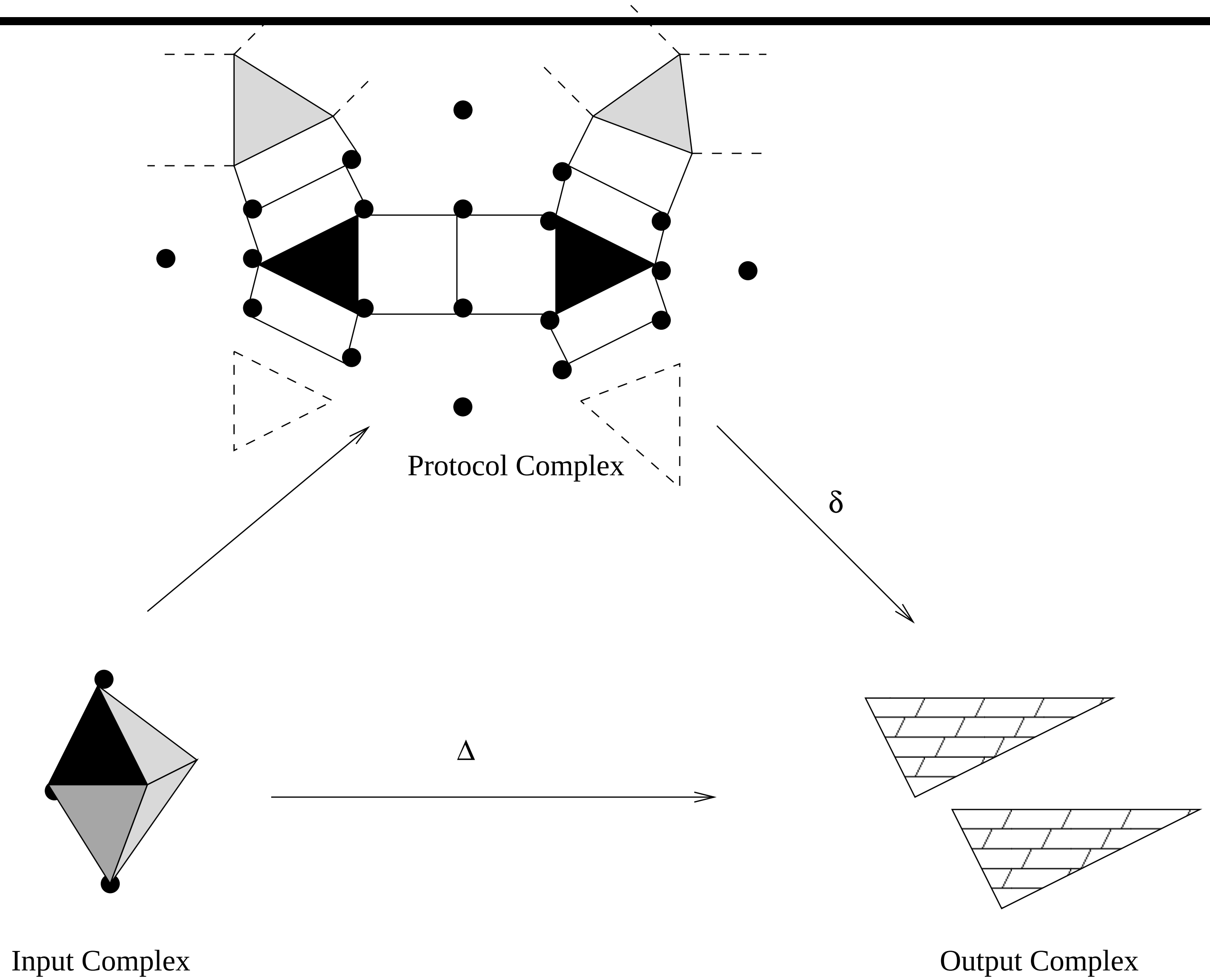
Decision map

The delta in the generic protocol is, mathematically speaking:

- is $\delta : P \to O$ (protocol to output complex)
- is a simplicial map (basically a function on vertices, extended on convex hulls)
- respects specification relation Δ, i.e. for all $x \in I$, for all $y \in P(I)$, $x\Delta(\delta(y))$

Proof strategy for impossibility/complexity results: find "topological obstruction" to the δ simplicial map (from protocol complex of any round/round up to k)

Main property



Continuous maps

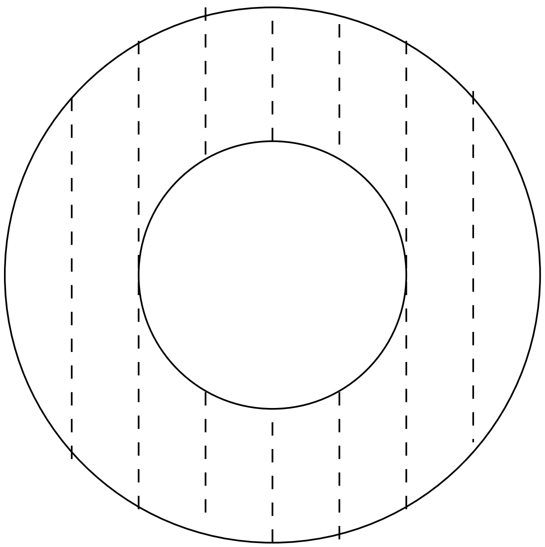
Let X and Y denote topological spaces.

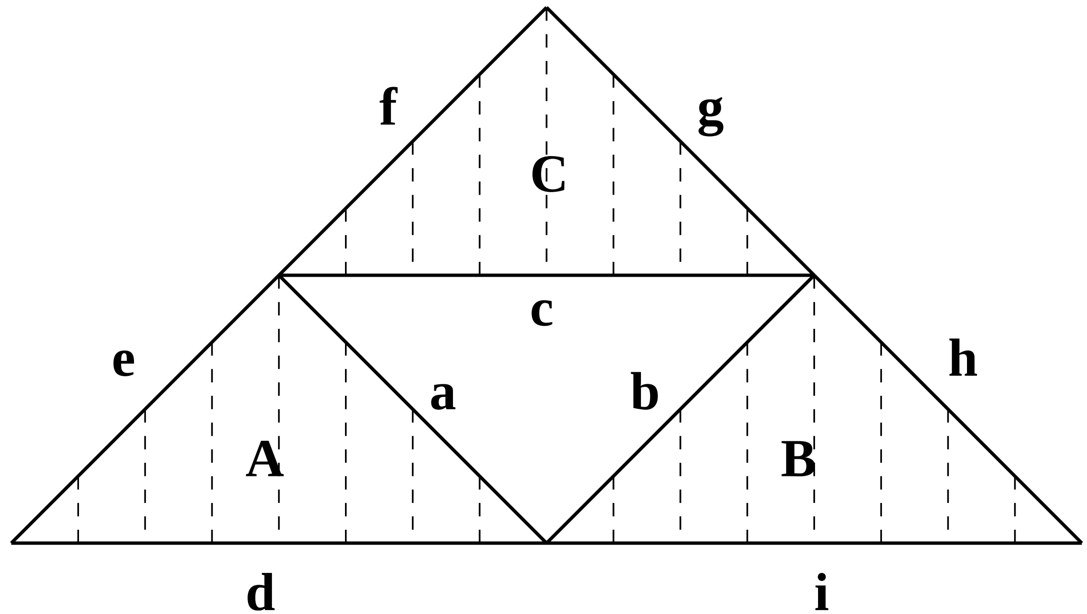
- A map $f : X \to Y$ is called continuous if and only if $f^{-1}(U) := \{x \in X | f(x) \in U\} \subset X$ is *open* for every open set $U \subseteq Y$.

- A continuous map $f : X \to Y$ is called a homeomorphism if
  - it is a bijection
  - its inverse $f^{-1} : Y \to X$ is continuous as well.

- Two topological spaces X, Y are called homeomorphic it and only if there exists a homeomorphism $f : X \to Y$.

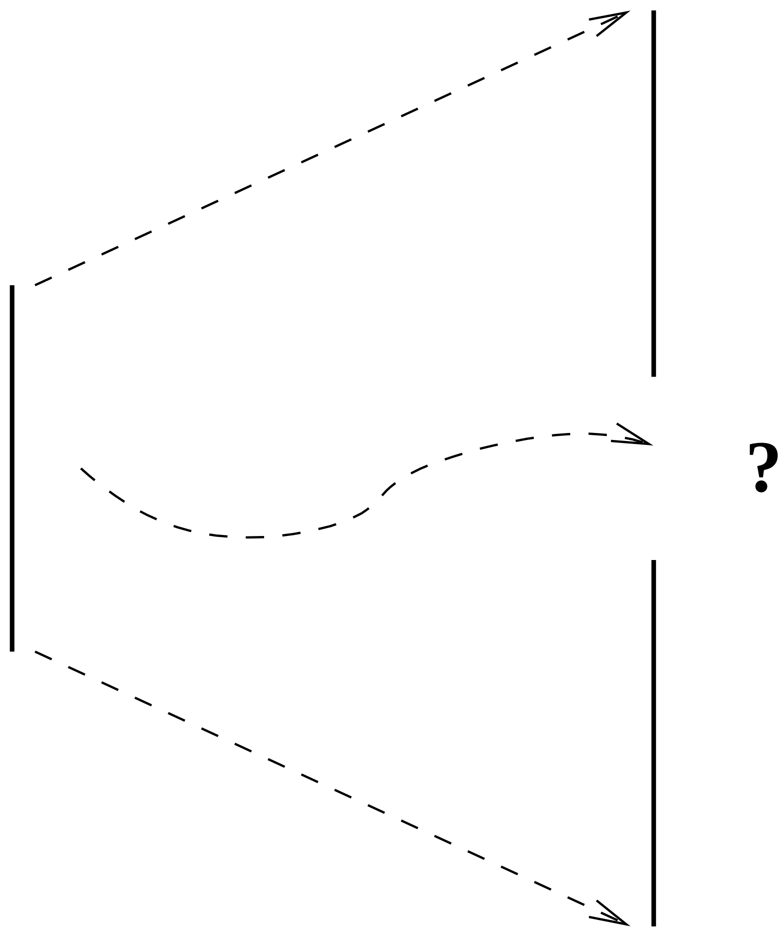
Relation with topological spaces





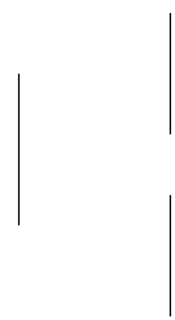
Simplicial/continuous maps

Commutation with boundaries is nothing than continuity:



Simple "invariants" of topological spaces

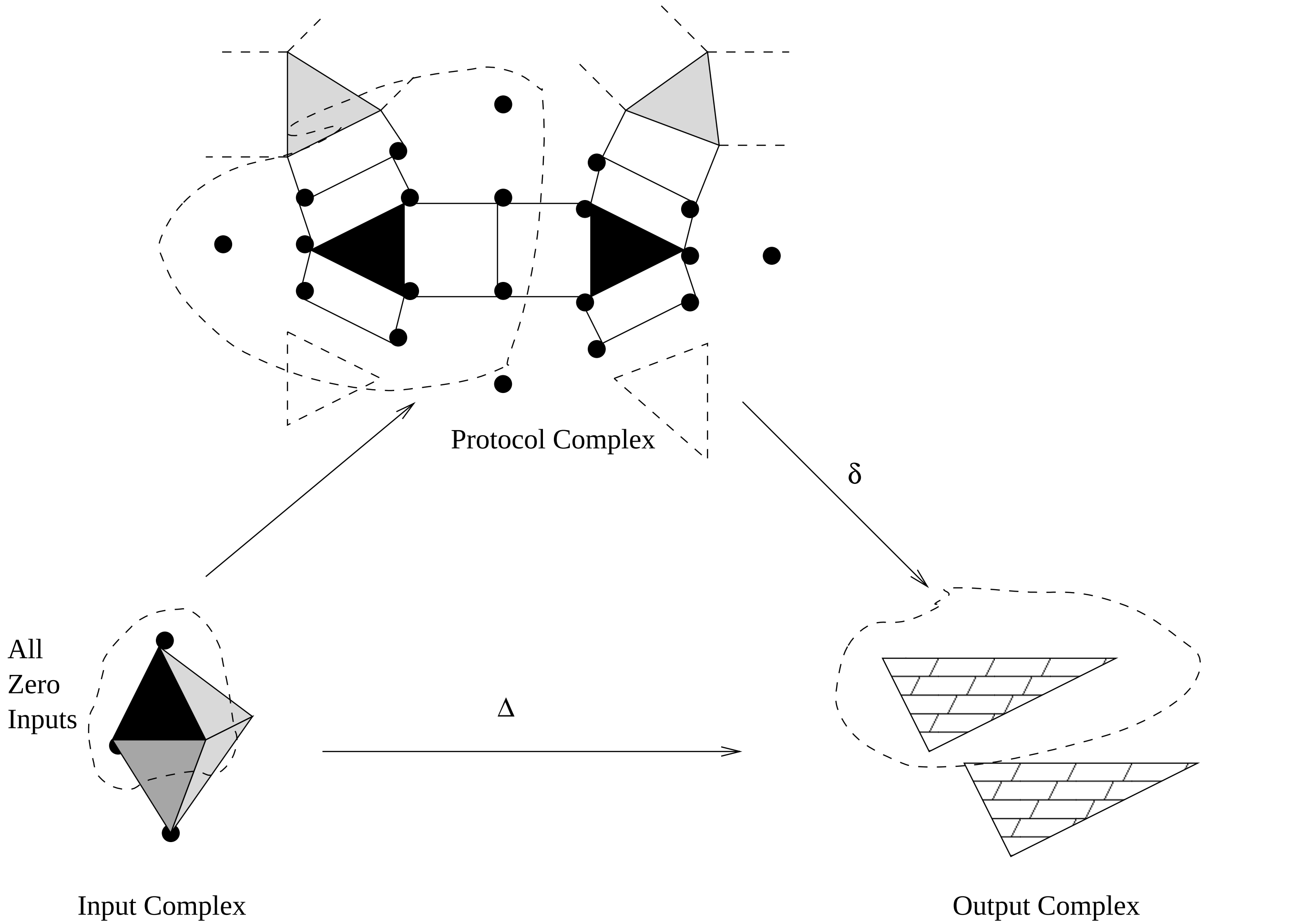
- If $f : X \rightarrow Y$ continuous (or simplicial...), then $Im(f)$ has a less or equal number of connected chunks
- Therefore: if X and Y are "equivalent" (i.e. homeomorphic), then they have the same number of "chunks" (numerical invariant)
- Example: these two spaces cannot be "equivalent" (i.e. homeomorphic):



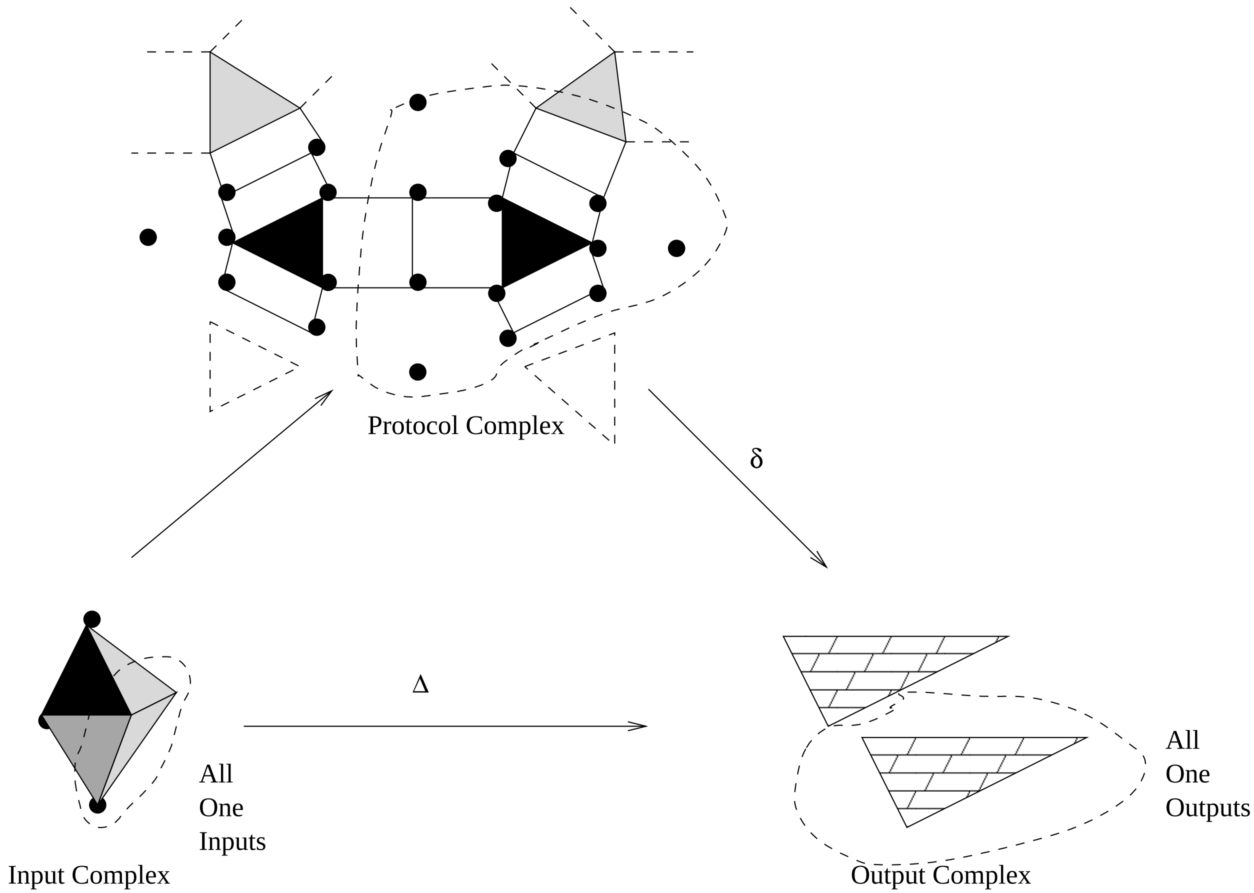
Easy application: consensus again...

- Binary consensus between 3 processes (synchronous message-passing model),
- Input complex is composed of 8 triangles: $(0,0,0)$, $(0,0,1)$, $(0,1,0)$, $(0,1,1)$, $(1,0,0)$, $(1,0,1)$, $(1,1,0)$ and $(1,1,1)$,
- Input complex is homeomorphic to a sphere (one connected component); the first four determine a “north” hemisphere, the last four create a “south” hemisphere
- Output complex is composed of 2 triangles: $(0,0,0)$ and $(1,1,1)$ (hence two connected components),
- Here: just one round.

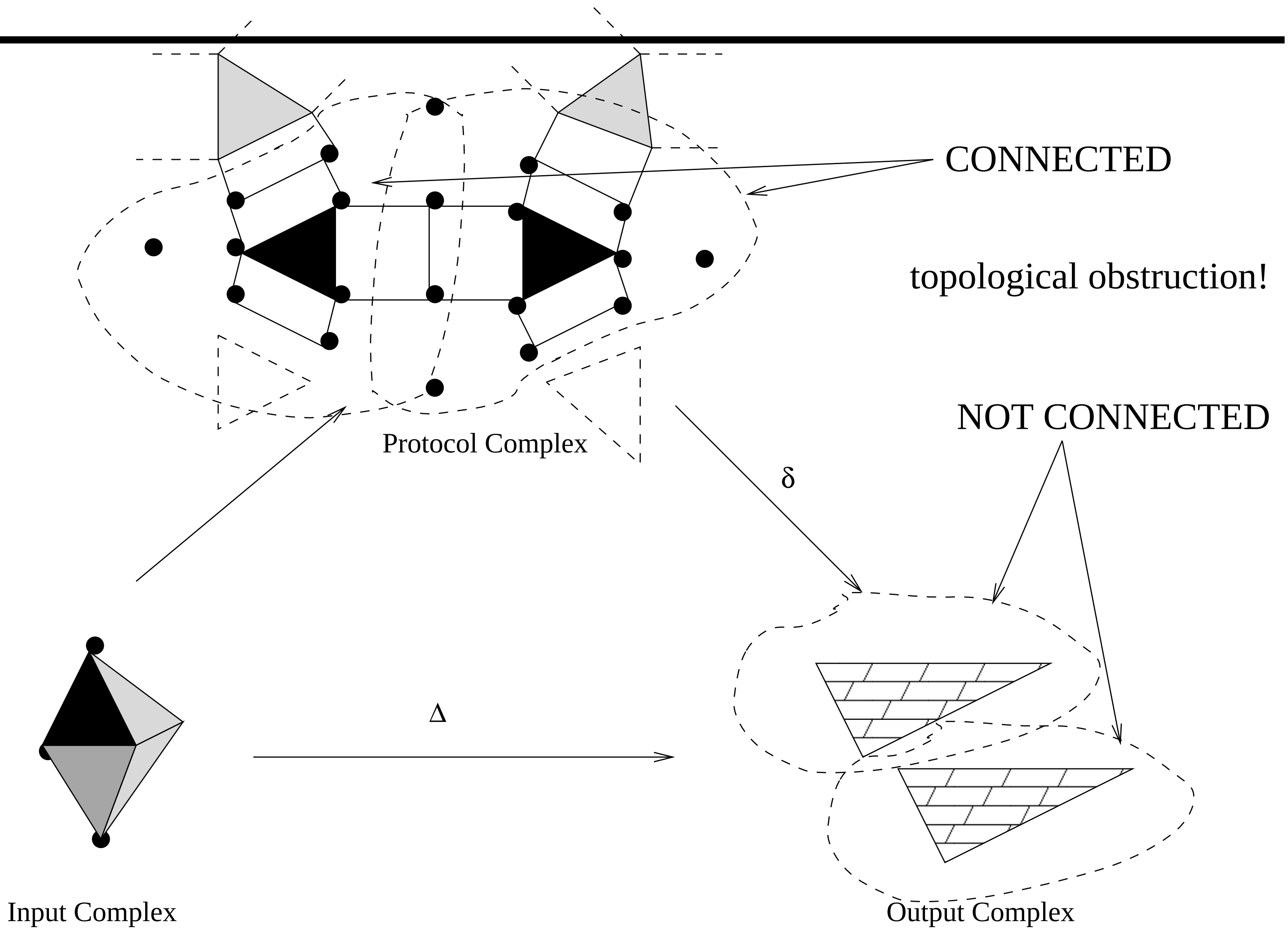
Easy application



Easy application



Easy application - for at most $n-2$ failures only!



More generally

- In any such $(n-2)$-round protocol complex, the all-zero subcomplex and the all-one subcomplex are connected
- Corollary: no $(n-2)$-round consensus protocol

Easy and not new... but gives the idea...

Even more generally...

- Synchronous message-passing model with r rounds, and at most k failures
- $P(S^{n-1})$ is $(n - rk - 2)$-connected: implies $(n-1)$-round consensus bound (for $k = 1$).

Paths in a topological space

- Let $I = [0, 1]$ denote the unit interval and let X denote a topological space. A path in X is a continuous map $\alpha : I \to X$. Its endpoints are $x_0 = \alpha(0)$ and $x_1 = \alpha(1)$.
- A path in X with $x_0 = \alpha(0) = \alpha(1) = x_1$ is called a loop in X.
- The concatenation (not associative in general) of α_1 and α_2 is undefined unless $\alpha_1(1) = \alpha_2(0)$. In that case,

$$(\alpha_1 * \alpha_2)(s) = \begin{cases} \alpha_1(2s), & t \leq \frac{1}{2} \\ \alpha_2(2s-1), & t \geq \frac{1}{2}. \end{cases}$$

Homotopy

- A homotopy is a "continuous deformation"
- A homotopy is a 1-parameter family $H_t : X \to Y, \; ; t \in I$ of maps, such that the associated map $H : X \times I \to Y$ is continuous.
- Two continuous maps $f, g : X \to Y$ are homotopic (symbol: $f \simeq g$) if and only if there is a homotopy $H : X \times I \to Y$ such that $H_0(x) = H(x, 0) = f(x)$ and $H_1(x) = H(x, 1) = g(x)$ for all $x \in X$.

Homotopy of loops

Let X denote a topological space, and let $x_0 \in X$ denote a *base point*.

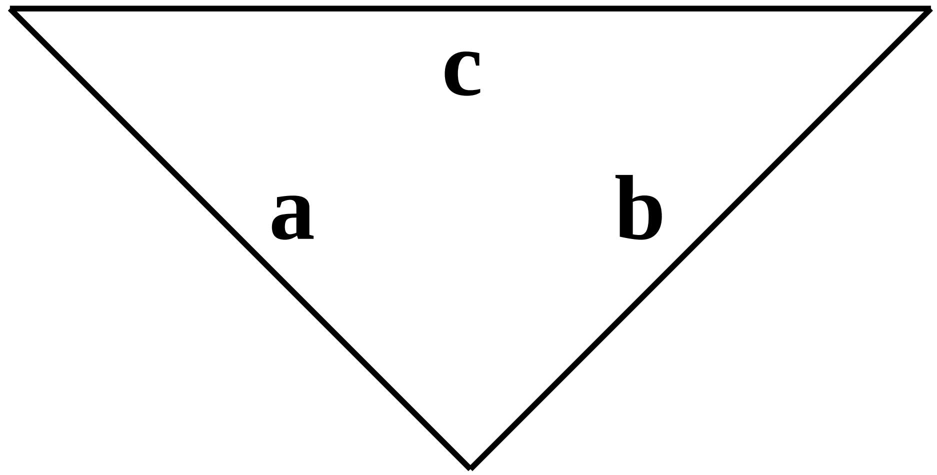
- A path $\alpha : I \to X$ is called a loop with basepoint x_0 if $\alpha(0) = \alpha(1) = x_0$. The set of loops with basepoint x_0 is denoted $\mathcal{P}_1(X; x_0)$.

- Concatenation defines a binary operation $C : \mathcal{P}_1(X; x_0) \times \mathcal{P}_1(X; x_0) \to \mathcal{P}_1(X; x_0)$.

- A *homotopy* of loops at x_0 is a family of loops $H_t : I \to X$ at x_0 such that the associated map $H : I \times I \to X, \; ; \; H(x,t) = H_t(x)$ is continuous.

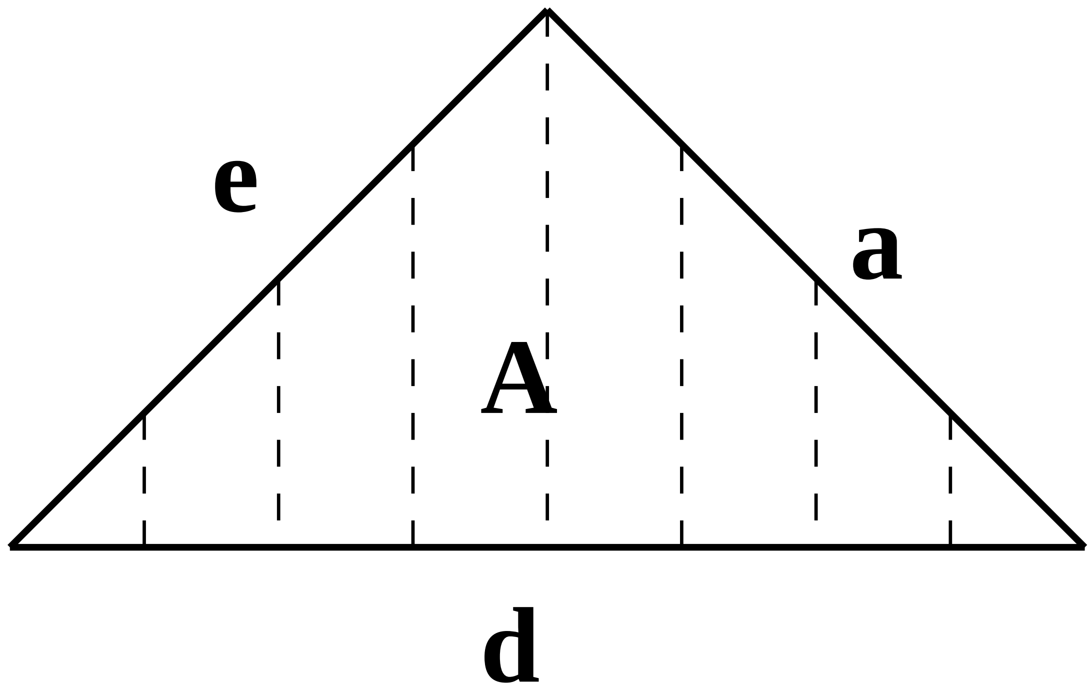
- Two loops α and β at x_0 are homotopic if there is a homotopy H_t of loops with $H_0 = \alpha$ and $H_1 = \beta$. We write: $\alpha \simeq \beta$.

Example: a loop which is not contractible



$$a - b + c$$

Example: a contractible loop



a−e+d

The fundamental group: Definition

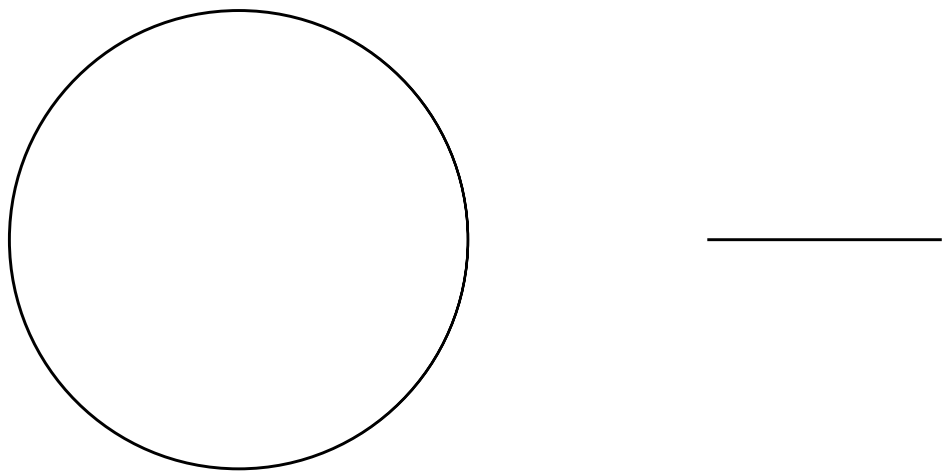
- Homotopy on loops with fixed basepoint defines an equivalence relation. The set of equivalence classes is denoted $\pi_1(X; x_0)$.
- Concatenation factors over the homotopy relation and thus defines a binary operation

$$C : \pi_1(X; x_0) \times \pi_1(X; x_0) \to \pi_1(X; x_0).$$

  We write $[\alpha] * [\beta]$ for $C([\alpha], [\beta])$.
- $\pi_1(X; x_0)$ with the operation $*$ is a group, the fundamental group of X with base point x_0.

Example

π_1 separates out (typical obstruction!):



(first case: $\mathbb{Z}$, second case: $*$)

The fundamental group: Properties and Examples

- If $x_0, x_1 \in X$ can be connected by a continuous path, the associated fundamental groups are isomorphic: $\pi_1(X; x_0) \cong \pi_1(X; x_1)$.
- $\pi_1(\mathbf{R}^n)$ is the (one-element) trivial group.
- The fundamental group of the circle S^1 is isomorphic to the integers: $\pi_1(S^1) \cong \mathbf{Z}$.
- The fundamental group of a higher-dimensional sphere $S^n, \; ; \; n > 1$ is trivial: $\pi_1(S^n) \cong 0, \; ; \; n > 1$.
- The fundamental group of the figure 8 (one-point identification of two circles) is the free group on two generators and thus *not* commutative.

Induced Homomorphisms: Definition and Properties

- A continuous map : $X \to Y$ induces a map $f_\# : \pi_1(X, x_0) \to \pi_1(Y, f(x_0))$, i.e., $f_\#[\alpha] = [f \circ \alpha]$.
- $f_\#$ is a group homomorphism.
- If $f_0, f_1 : X \to Y$ are homotopic via a *base-point preserving* homotopy $H : X \times I \to Y$, then the induced homomorphisms $f_{j\#} : \pi_1(X, x_0) \to \pi_1(Y, f(x_0))$ *agree*.

Higher homotopy groups: Definition

- Loops can be seen as continuous maps $\alpha : S^1 \to X$ defined on the circle.
- More generally, one considers continuous maps $\alpha : S^n \to X$ mapping base-point into base-point – and classifies them up to homotopy.
- The result is the n-th homotopy group $\pi_n(X)$ of X – an abelian group for $n > 1$.
- A continuous map $f : X \to Y$ induces a homomorphism $f_\# : \pi_n(X) \to \pi_n(Y)$.
- Homotopy invariance Homotopic maps induce the same homomorphism on the homotopy groups..

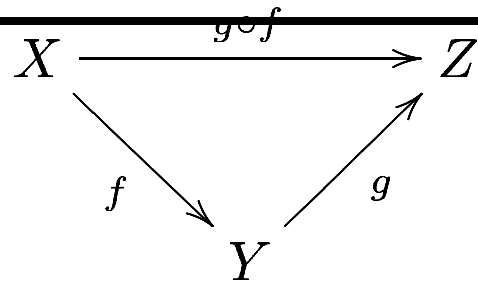
Higher homotopy groups: Properties and Examples

- Contractible spaces (like $\mathbf{R}^n$) have trivial fundamental groups.
- Homotopy groups of spheres:

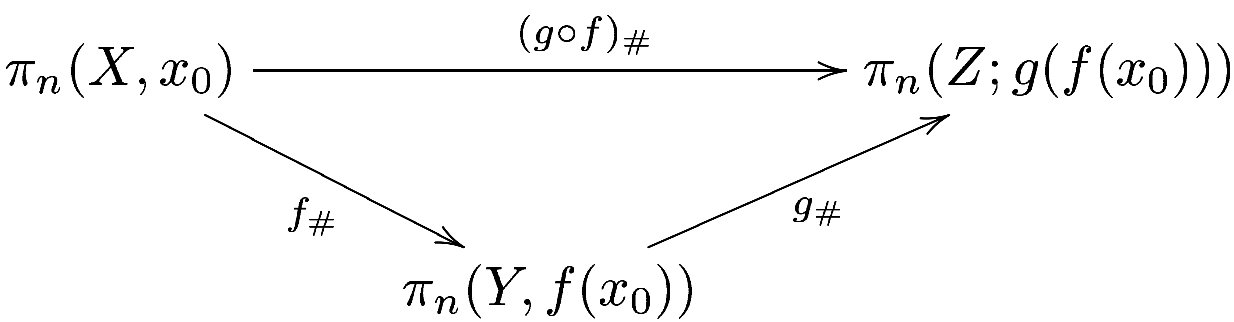
$$\pi_k(S^n) = \begin{cases} 0 & k < n \\ \mathbf{Z} & k = n \\ \text{unknown} & k > n \text{ in general} \end{cases}$$

- A space X is called k-connected if $\pi_i(X) = 0$ for $i \leq k$; example, S^n is $(n-1)$-connected.

Induced Homomorphisms: Naturality



induces

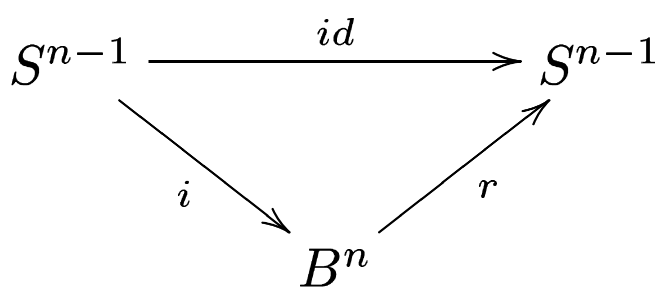


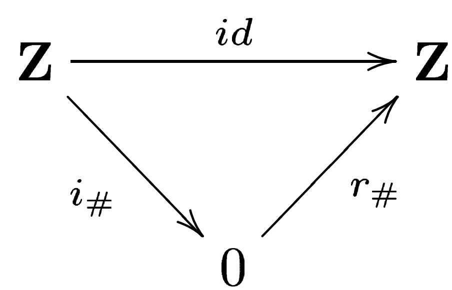
- $id : X \to X$ induces $id : \pi_n(X : x_0) \to \pi_n(X : x_0)$.
- Homotopy equivalent spaces have isomorphic homotopy groups.

Application: Brouwer's fixed point theorem

Let $B^n := \{x \in \mathbf{R}^n \; ; \mid ; \; \| \, x \, \| \leq 1\}$.

- (BFT) Every continuous self-map $f : B^n \to B^n$ has a fixed point $x_0 \in B^n$: $f(x_0) = x_0$.
- Lemma: There is no continuous map $r : B^n \to S^{n-1}$ extending the identity.





Application: Invariance of Dimension

Theorem $\mathbf{R}^n$ homeomorphic to $\mathbf{R}^m \Leftrightarrow m = n$.
(same for homotopy equivalent).
Proof:

- $\mathbf{R}^n \cong \mathbf{R}^m \Rightarrow S^{n-1} \simeq \mathbf{R}^n \setminus \{P\} \cong \mathbf{R}^m \setminus \{Q\} \simeq S^{m-1}$.
- Hence: S^{n-1} and S^{m-1} have isomorphic homotopy groups in all dimensions.
- Conclusion: $m = n$.

An application – and a warning

- Calculation of $\pi_1(S)$; S a surface (sphere, torus, Klein bottle etc.)
- Can be used to show that two given surfaces are *not* homotopy equivalent, and thus not homeomorphic.
- There is no immediate generalization to higher dimensions!
- In general, homotopy groups are easy to define – but difficult or impossible to calculate!

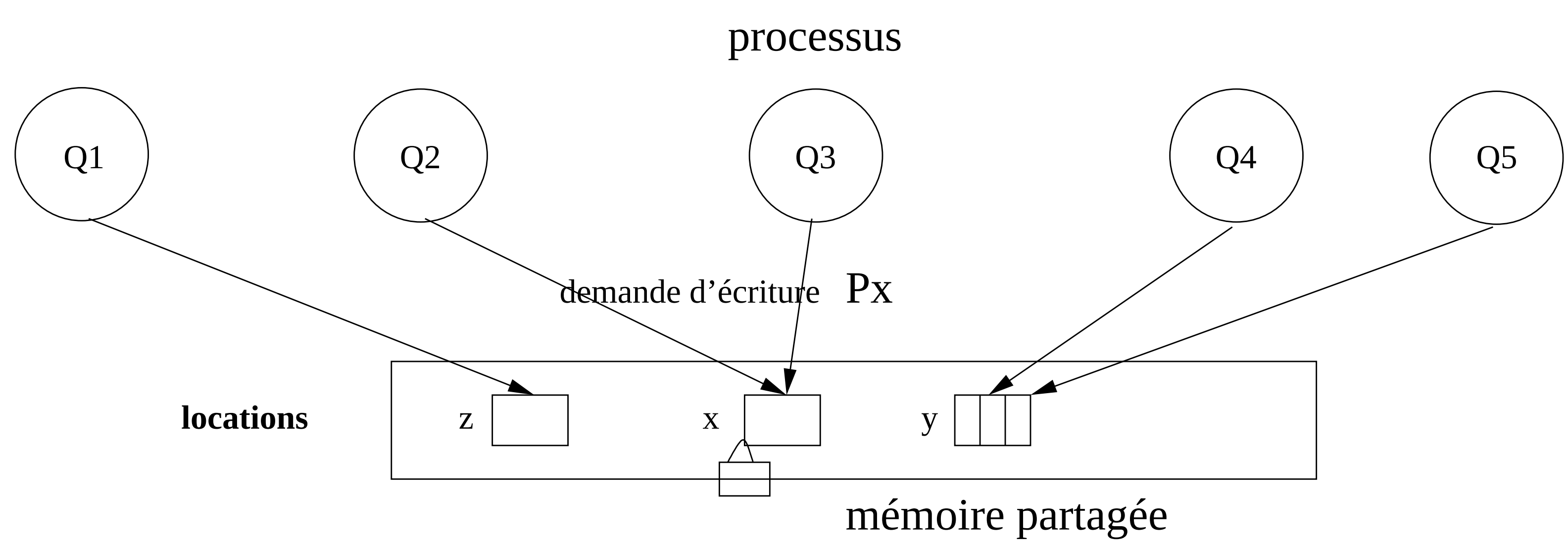
Homology

- Sequence of *abelian* groups
- Much simpler to compute or characterize than homotopy groups
- Invariants of homotopy, hence of homeomorphism
- Defined through a “simplicial” representative in general

Connections between homotopy and homology

- The homology groups of a (not too complicated) space can often be calculated inductively – much more accessible than homotopy groups.
- Knowledge of homology/cohomology can often help to deal with homotopy questions:
- **Hurewicz theorem** A simply connected topological space X $(\pi_1(X) = 0)$ is k-connected $(\pi_i(X) = 0, \; ; \; i \leq k)$ if and only if $H_i(X) = 0, \; ; \; i \leq k$.

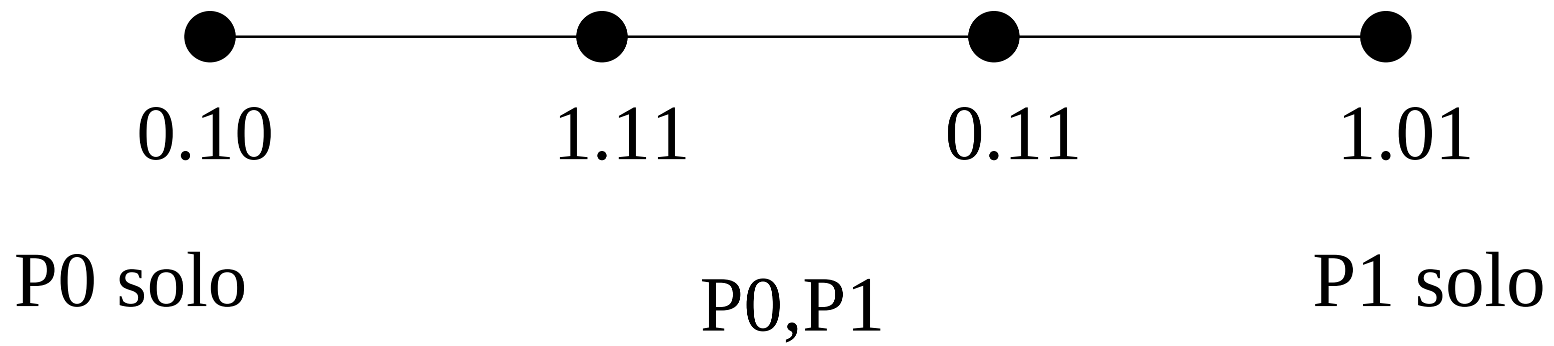
Shared-memory model



Asynchronous wait-free protocols

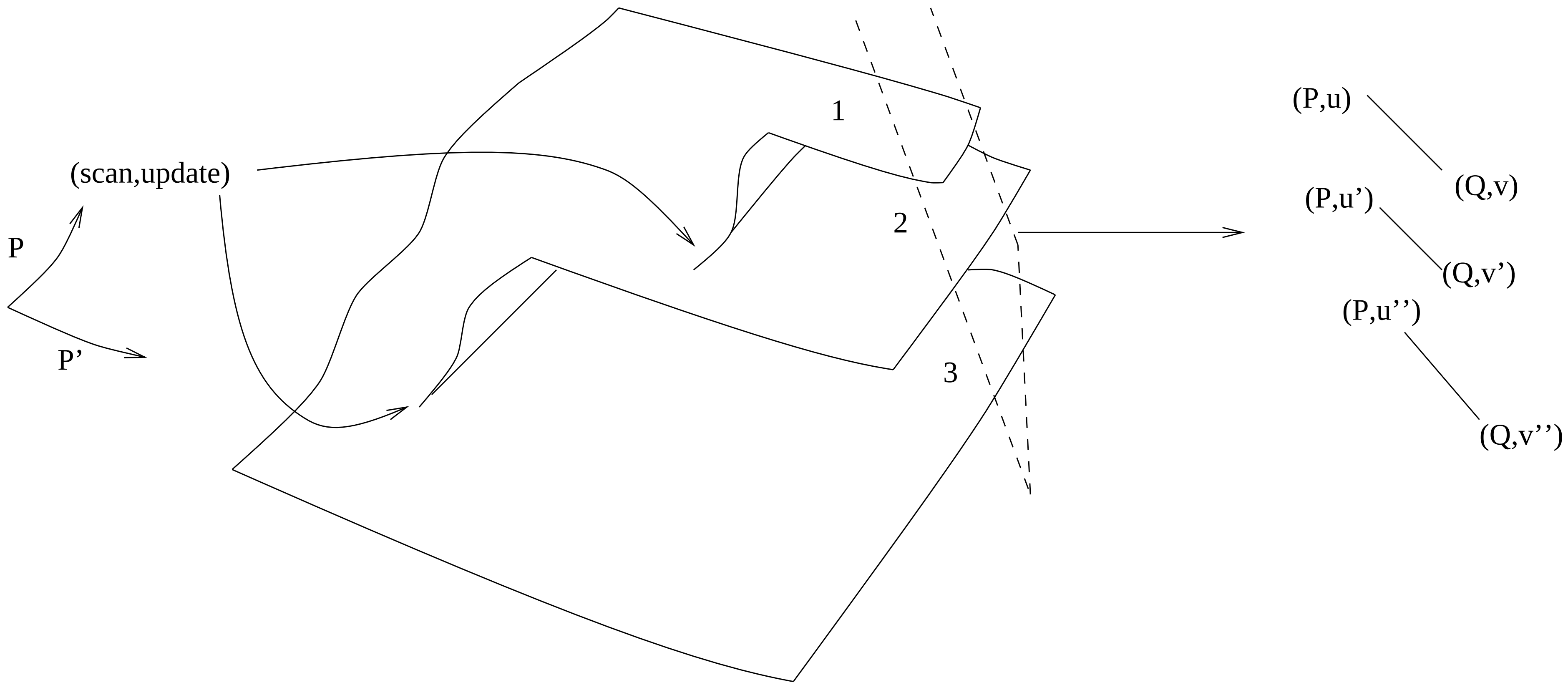
- n processes share memory (unbounded size), partitioned: one private chunk for each process
- Each process can:
  - atomically write to its location (update)
  - atomically scan (read) all of the memory into its local memory
- Equivalent to the usual read/write models
- We want *wait-free* protocols, i.e. robust to up to $n-1$ crash failures

One-round protocol simplicial set (2D)

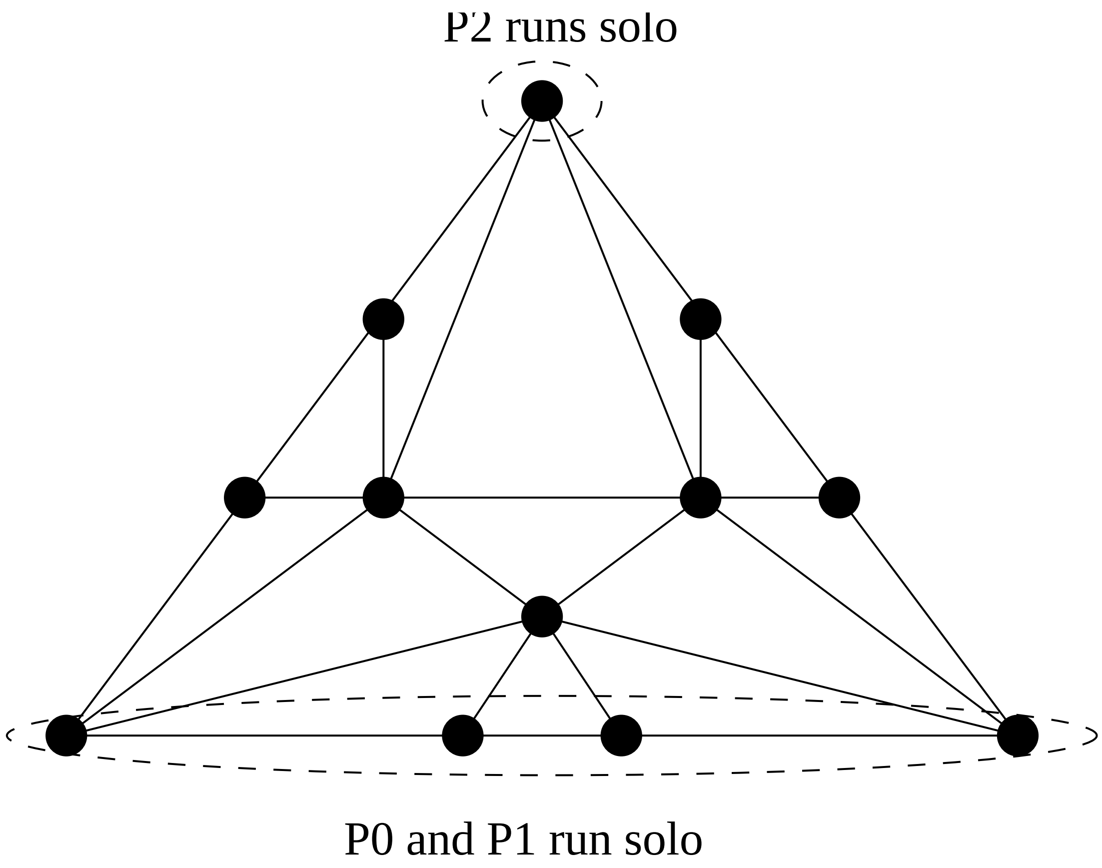


Semantics

Dynamics (and its cut up to time r=protocol complex):



One-round protocol simplicial set (3D)

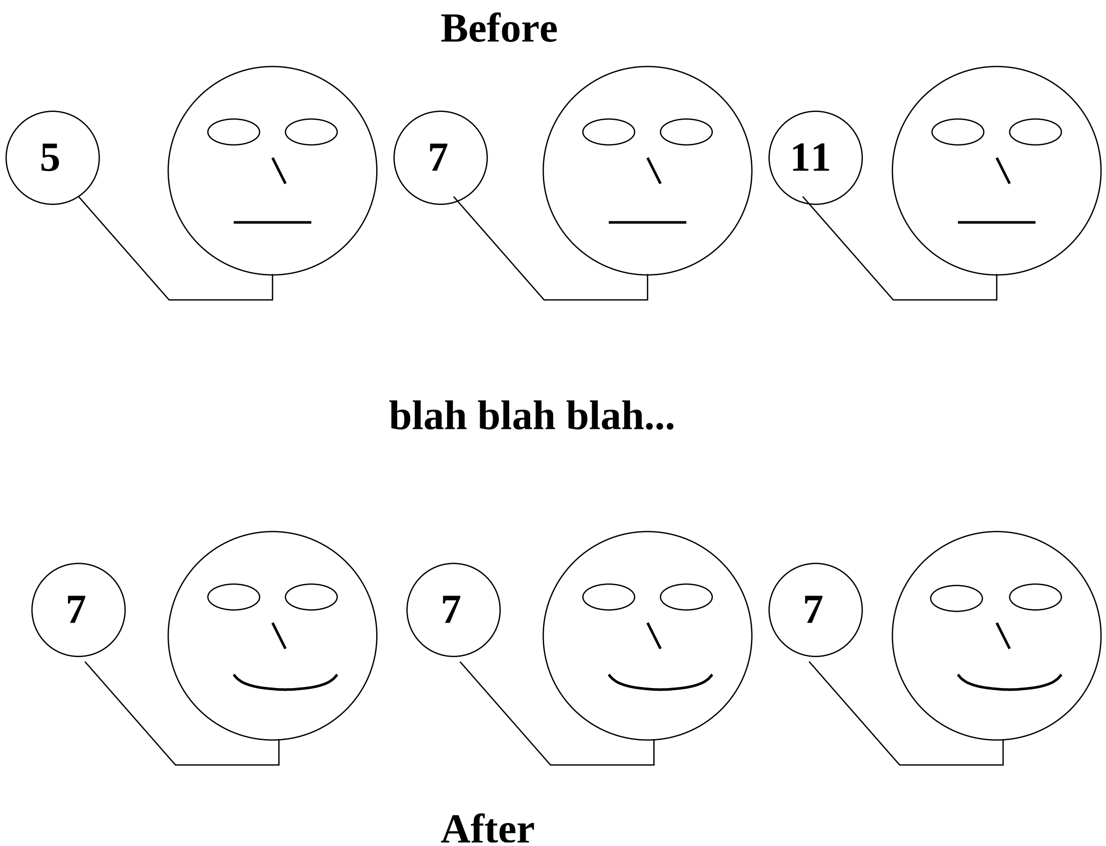


Theorem

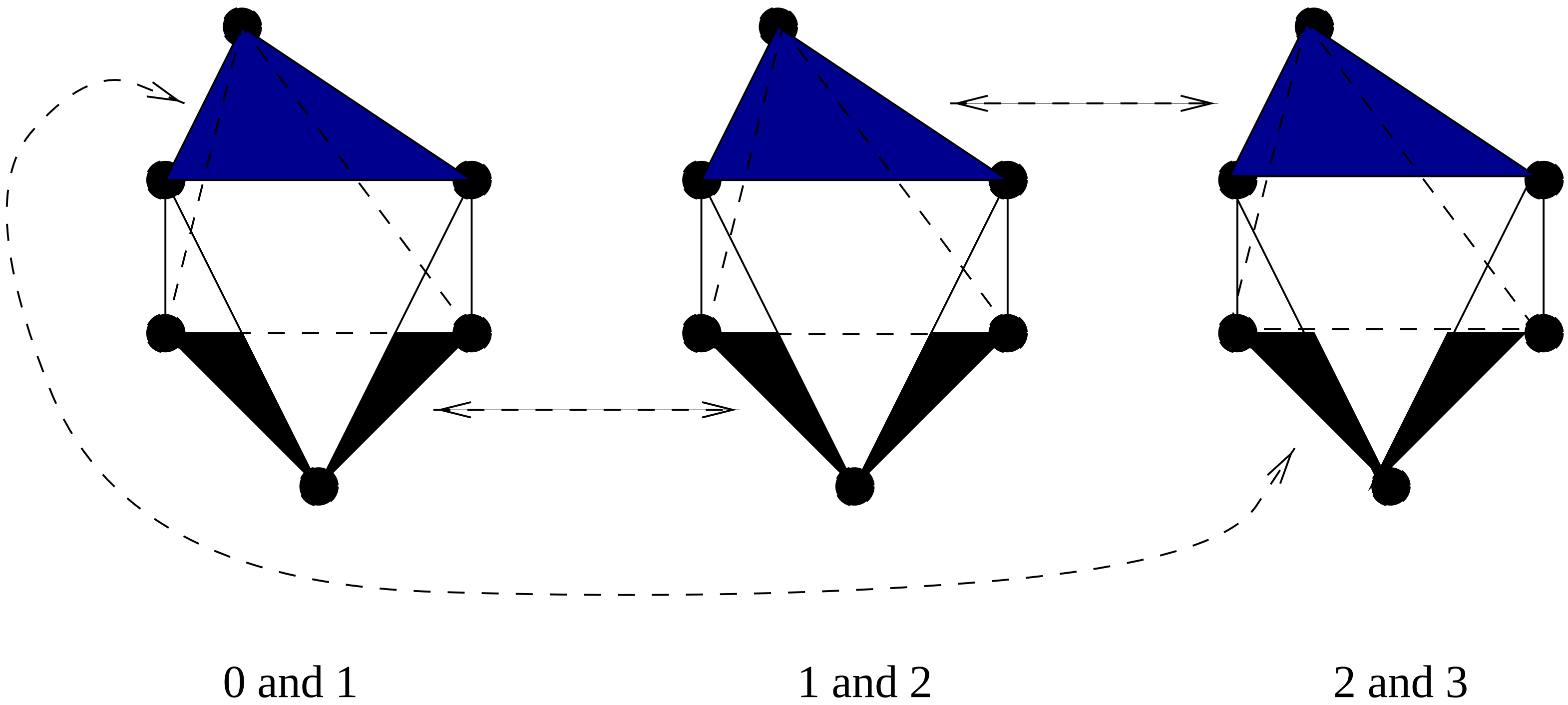
- Wait-free read/write protocol complexes are:
  - $(n-1)$-connected (no holes in any dimension)
  - no matter how long the protocol runs
- Application: k-set agreement

k-set agreement task

Generalization of consensus; processes must end up with at most k different values (taken from the initial values):



Output simplicial set ($n = 3$, $k = 2$)



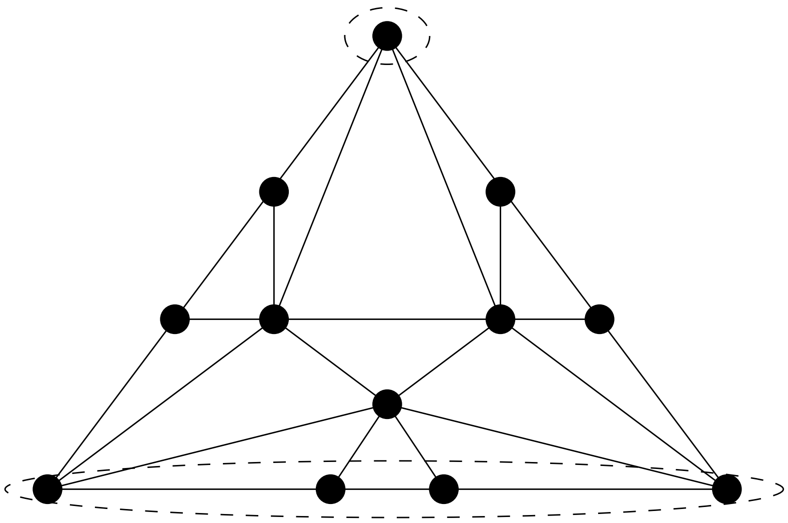
3 spheres glued together minus the simplex formed of all 3 values:
not 1-connected

Sketch of a proof

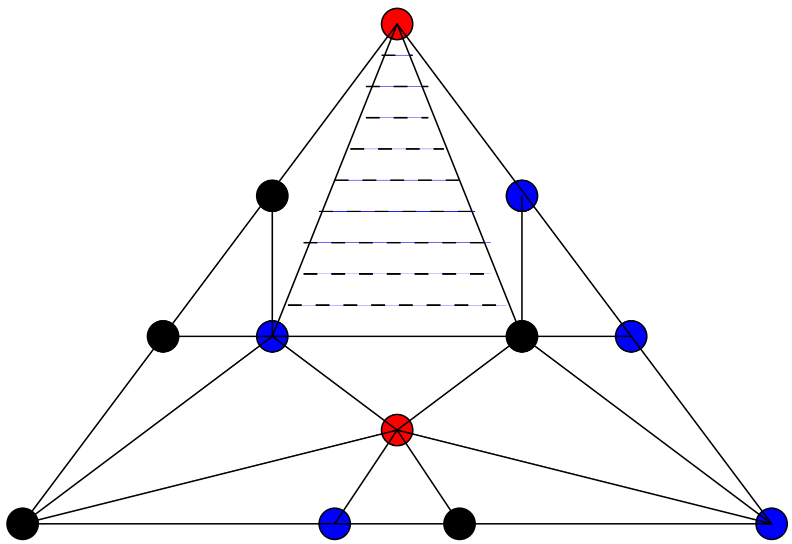
A tool from algebraic topology (Sperner's lemma):

- Subdivide a simplex
- Give each "corner" a distinct "color"
- Give each vertex a corner color
- Giver interior vertices any corner color

Sperner's lemma



P0 and P1 run solo



⇒ At least one simplex has all colors

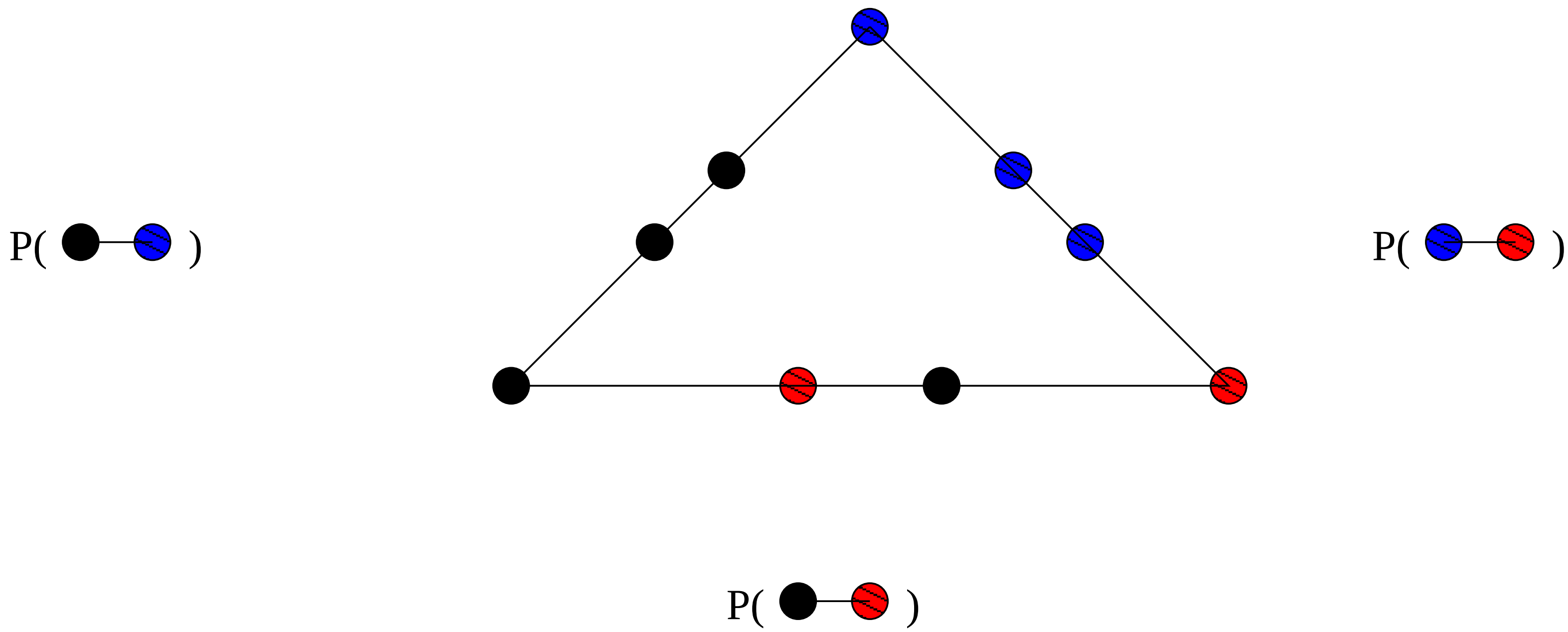
Input and protocol simplicial set

- Each process colored with distinct input
- Each vertex colored with decision

Protocol complex

- For a one-process execution: same vertex and same color (cannot decide anything else)
- For a two-process execution:
  - the protocol complex is connected
  - all vertices are of one of the two colors

Protocol complex - for all 2 process executions



Full protocol complex

- Because complex is simply-connected
- We can "fill-in" edge-paths
- Vertices colored with input colors

End of proof

Apply Sperner's Lemma:

- Some simplex has all three colors
- That simplex is a protocol execution that decides three values!

Converse

- In fact, even more:
- A task has a wait-free read/write protocol if and only if there exists a simplicial map μ:
  - from subdivided input complex
  - to output complex
  - that respects Δ

Principle of the proof

$\Rightarrow$

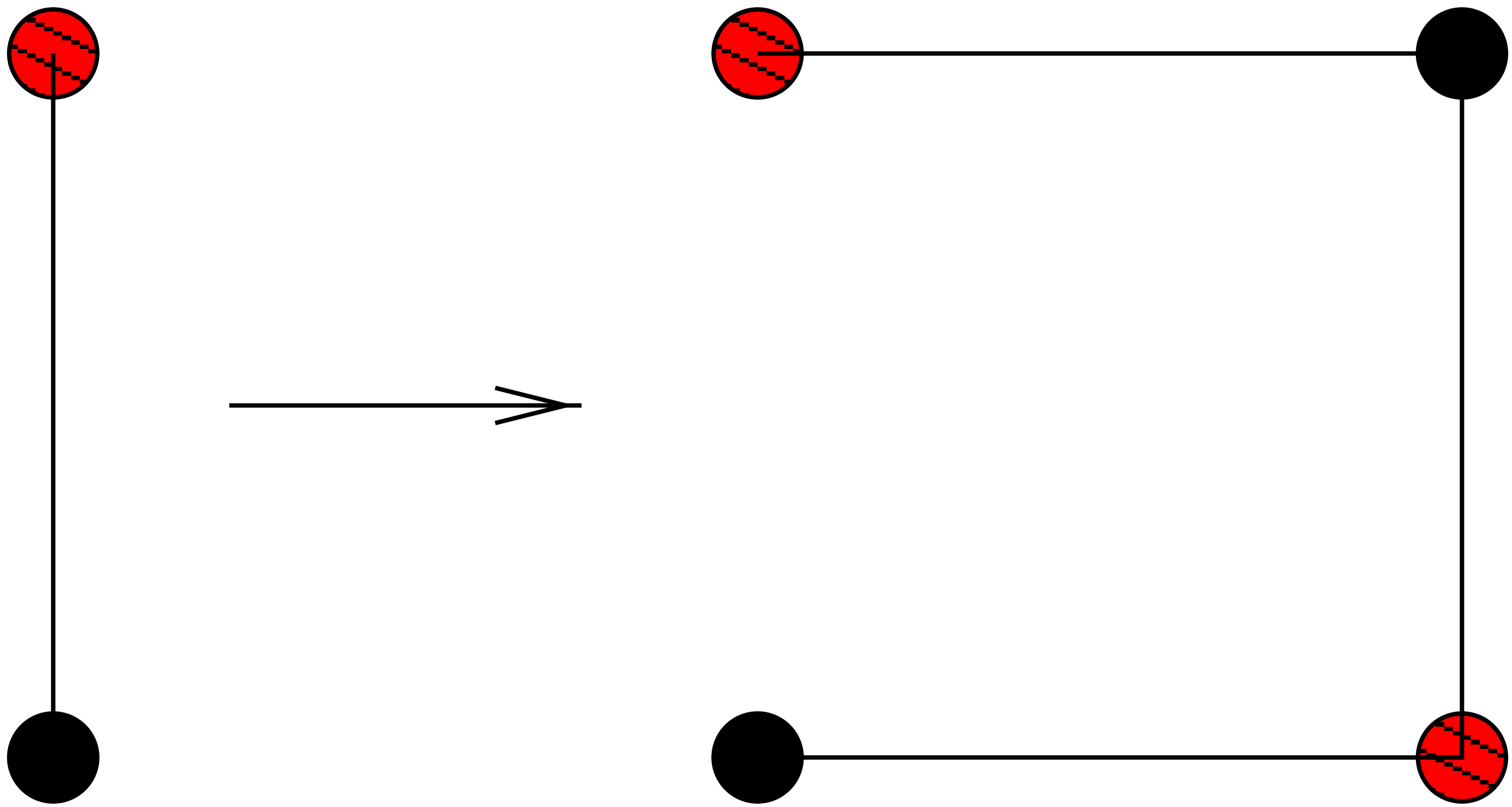
- Protocol complex is $(n-1)$-connected (using Mayer-Vietoris)
- Exploit connectivity to
  - embed subdivided input complex into protocol complex
  - map protocol complex to output complex
  - just like k-set agreement proof

Principle of the proof

$\Leftarrow$

- We can reduce any task to "simplex agreement" [using the participating set algorithm of Borowsky and Gafni 1993]
- Start out at corners of subdivided simplex
- Must rendez-vous on vertices of single simplex in subdivision

Example



Subdivision of a segment into three segments

Protocol

$$
\begin{aligned}
P \quad = \quad & update; \\
& scan; \\
& case\ (u, v)\ of \\
& \ (x, y') : \ u = x'; update; [] \\
& \ default : \ update
\end{aligned}
\qquad\qquad
\begin{aligned}
P' \quad = \quad & update; \\
& scan; \\
& case\ (u, v)\ of \\
& \ (x, y') : \ v = y; update; [] \\
& \ default : \ update
\end{aligned}
$$

Proof

Using the semantics, we have the following three possible 1-schedules (up to homotopy), since the only possible interactions are between the $scan$ and $update$ statements,

(i) Suppose the $scan$ operation of P is completed before the $update$ operation of P' is started: P does not know y so it chooses to write x. $Prog$ ends up with $((P,x),(P',y))$.

(ii) Symmetric case: $Prog$ ends up with $((P,x'),(P',y'))$.

(iii) The $scan$ operation of P is after the $update$ of P' and the $scan$ of P' is after the $update$ of P. $Prog$ ends up with $((P,x'),(P',y))$.

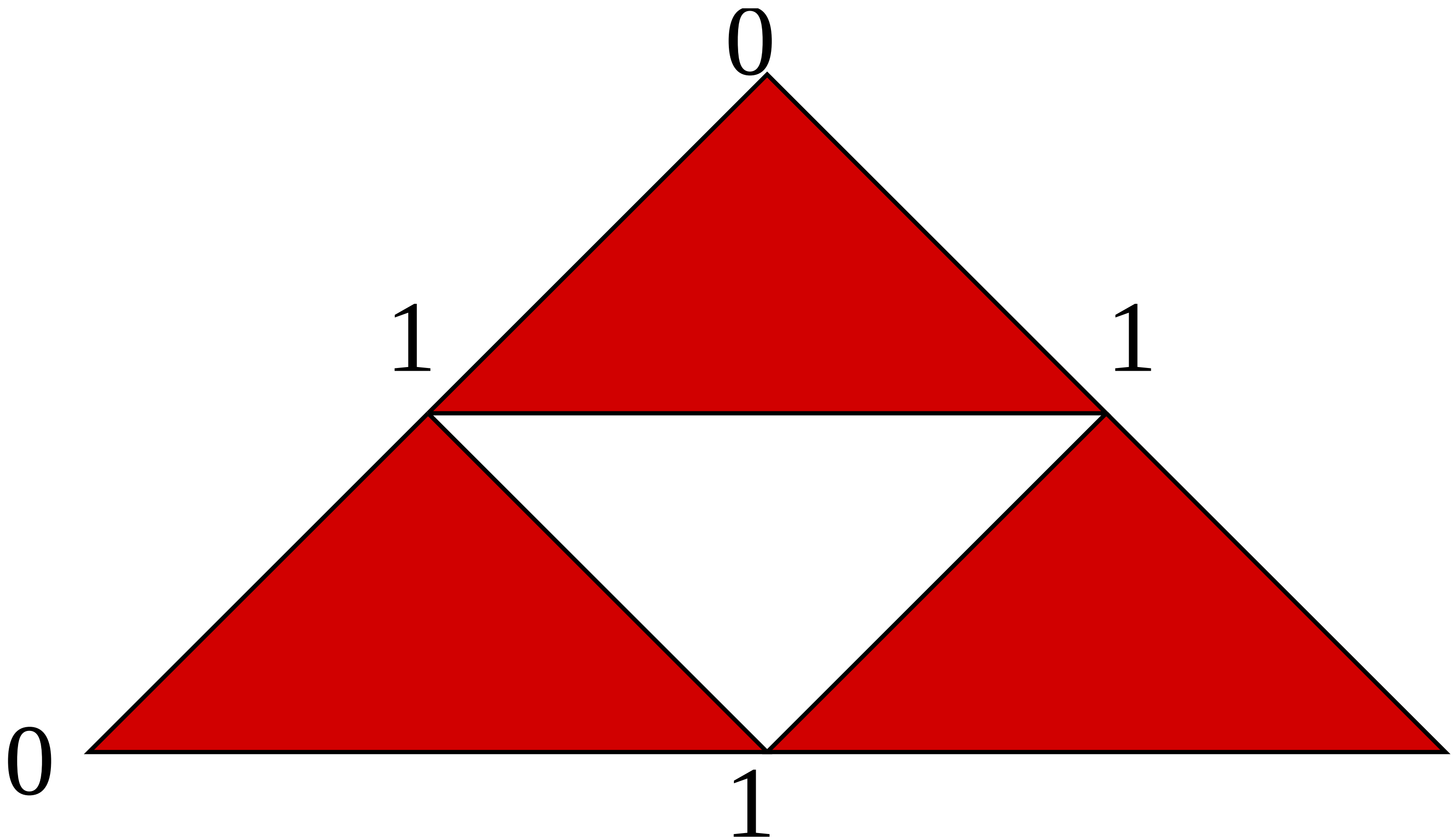
Other communication primitives

Real multiprocessors provide additional atomic synchronization:

- test&set
- fetch&add
- compare&swap
- queues...

Other protocol complexes...other results

Example: test&set protocol complex



Test&Set

- Wait-free Test&Set protocol complexes
  - are all $(n-3)$-connected
  - more powerful than read/write (2-process consensus)
  - but still no 3-process consensus
- Similar results hold for other synchronization operations

References and main results

- Begins with Fisher-Lynch-Patterson ("FLP") in 1985: there exists a simple task that cannot be solved in a (simple) message-passing system with at most one potential crash
- Created a very active research area, see for instance Nancy Lynch's book "Distributed Algorithms" (1996)

References and main results

- Later developed by Biran-Moran-Zaks in PoDC'88: characterization of the tasks that can be solved by a (simple) message-passing system in the presence of one failure
- The argument uses a "similarity chain", which could be seen as a 1-dimensional version of what we just developed
- Revealed to be difficult to extend to models with more failures

References and main results

Then, in PoDC'1993, independently,

- Borowsky-Gafni, Saks-Zaharoglou and Herlihy-Shavit derived lower bounds for the k-set agreement problem of Chaudhuri (proposed in 1990)
  [at least $\lfloor \frac{f}{k} \rfloor + 1$ steps in synchronous model]
- Saks-Zaharoglou and Herlihy-Shavit exploited topological properties to derive this lower bound

References and main results

- Renaming: Attiya-BarNoy-Dolev-Peleg JACM 1990,
- The $(n+1, K)$-renaming task starts with $n+1$ processes being given a unique input name in $0, \ldots, N$ and are required to choose unique output name in $0, \ldots, K$ with $n \leq K < N$ (independently of a "process id" - i.e. "anonymous renaming" in fact).
- Showed that (message-passing model) there is a wait-free solution for $K \geq 2n+1$, none when $K \leq n+2$
- Using these geometrical techniques: it has been shown that there is no renaming when $K \leq 2n$
- Herlihy and Shavit STOC'93: same result holds for the wait-free asynchronous model (using homology explicitely).

References and main results

Later results, on the same line, include:

- Full characterization of wait-free asynchronous tasks with atomic read/writes on registers, see “The topological structure of asynchronous computability”, M. Herlihy and N. Shavit, J. of the ACM, jan. 2000
- Use of algebraic spans in “Algebraic Spans”, M. Herlihy and S. Rajsbaum as a unified methods for renaming, k-set agreement problems etc.
- Use of pseudo-spheres...

References and research directions

- Consensus numbers (see M. Herlihy and then E. Ruppert SIAM J. Comput. vol 30, No 4, 2000 for instance). Importance based on the remark (M. Herlihy): an object which solves the consensus problem for n processes can simulate in a wait-free manner (together with read/write registers) any object for n or fewer processes.
- Example: R/W registers have consensus number 1, test&set, queues, stacks, fetch and add have consensus number 2 etc.
- Example: There is no wait-free $(n+1, 2j)$-renaming protocol if processes share a read/write memory and $(n+1, j)$-consensus objects.

References and research directions

- Afek et Strup: characterization of the effect of the register size in the power of synchronization primitives
- Characterization of complexity and not only computability, see for instance “Towards a Topological Characterization of Asynchronous Complexity”, G. Hoest and N. Shavit
- Links with (geometric) semantics [potential for more realistic models of distributed systems?], for instance my paper in CAAP’97 “Optimal Implementation of Wait-Free Binary Relations” ?
- Extension of this model for randomized algorithms etc.?