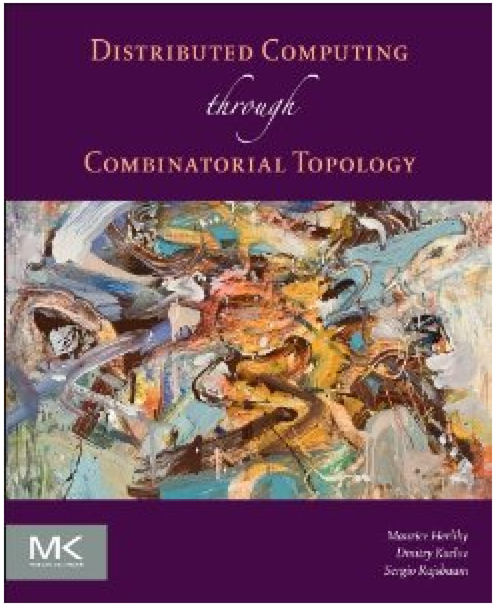
Colorless Wait-Free Computation

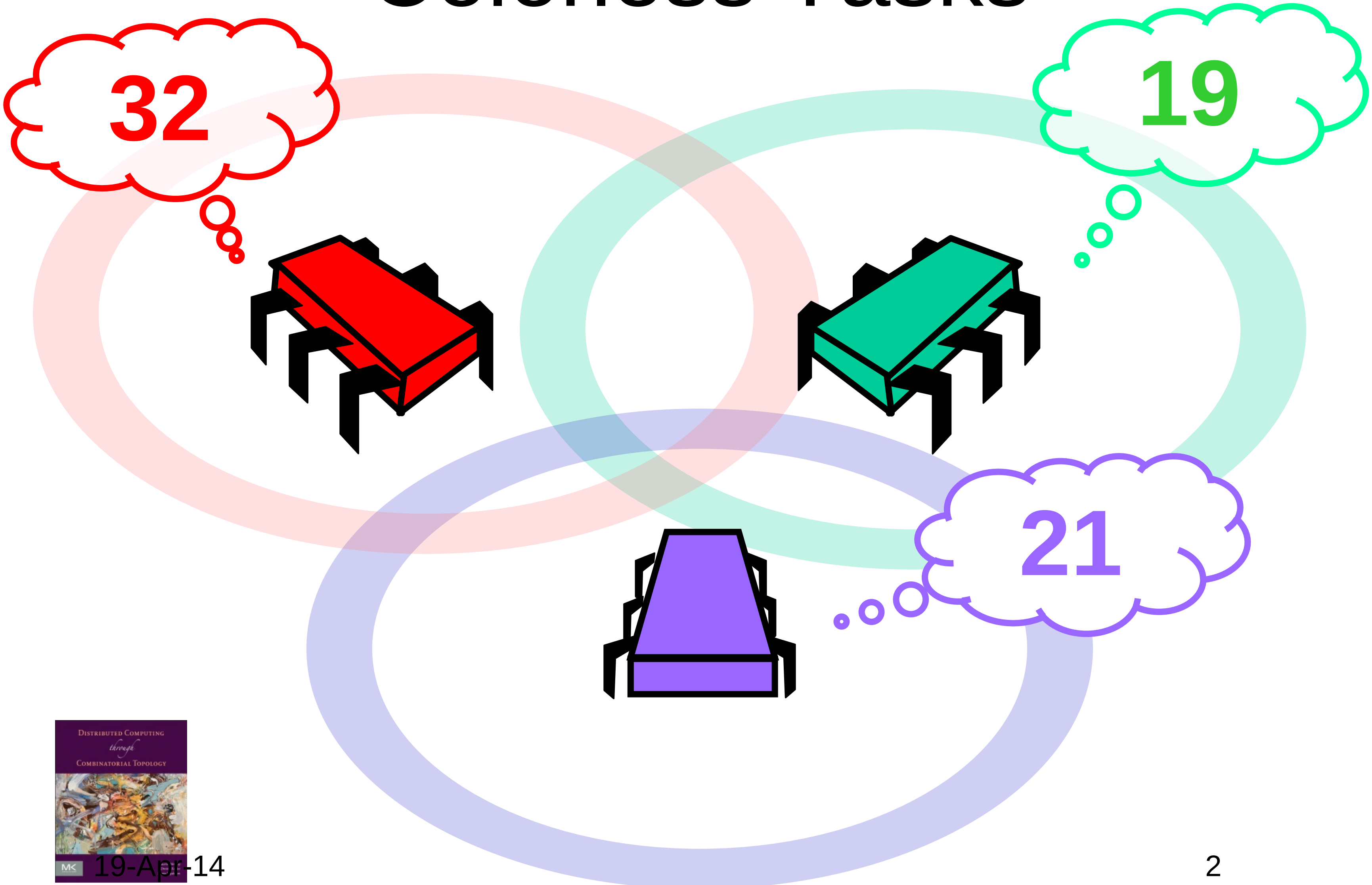


Companion slides for

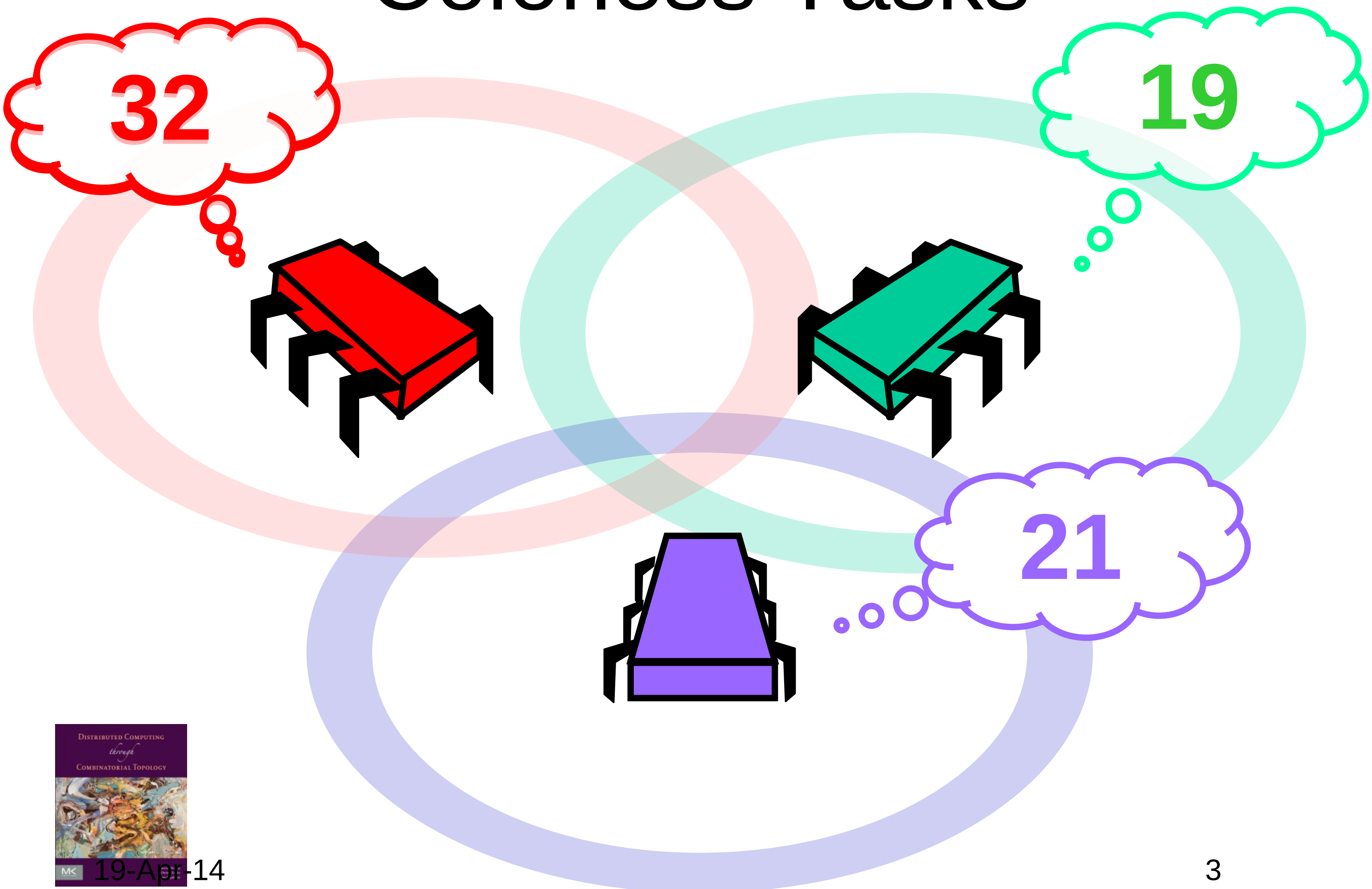
Distributed Computing Through Combinatorial Topology

Maurice Herlihy & Dmitry Kozlov & Sergio Rajsbaum

Colorless Tasks



Colorless Tasks



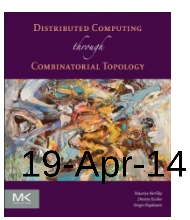
Colorless Tasks

The *set* of input values …

determines the *set* of output values.

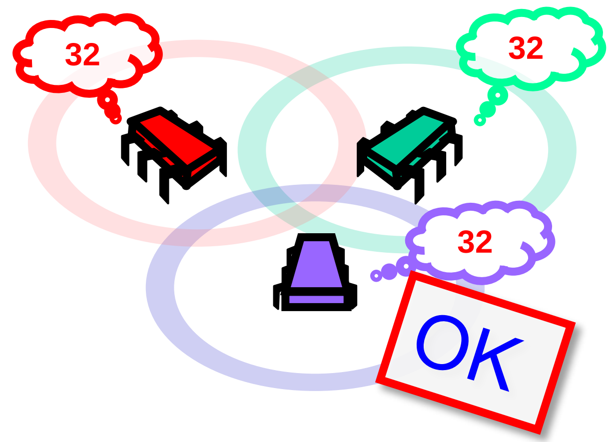
Number and identities irrelevant…

for both input and output values

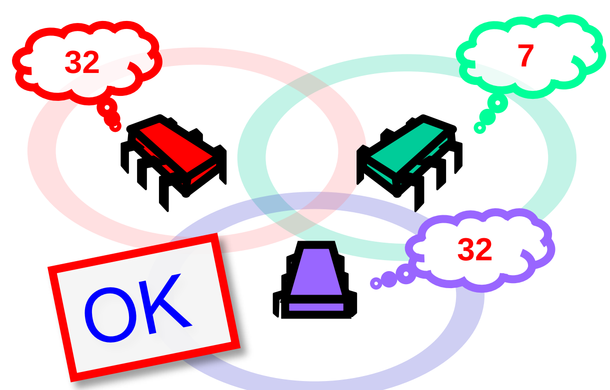


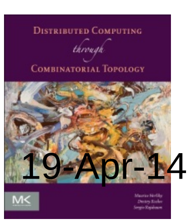
Examples

Consensus



k-set agreement



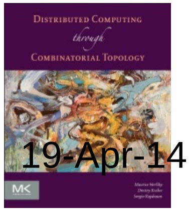


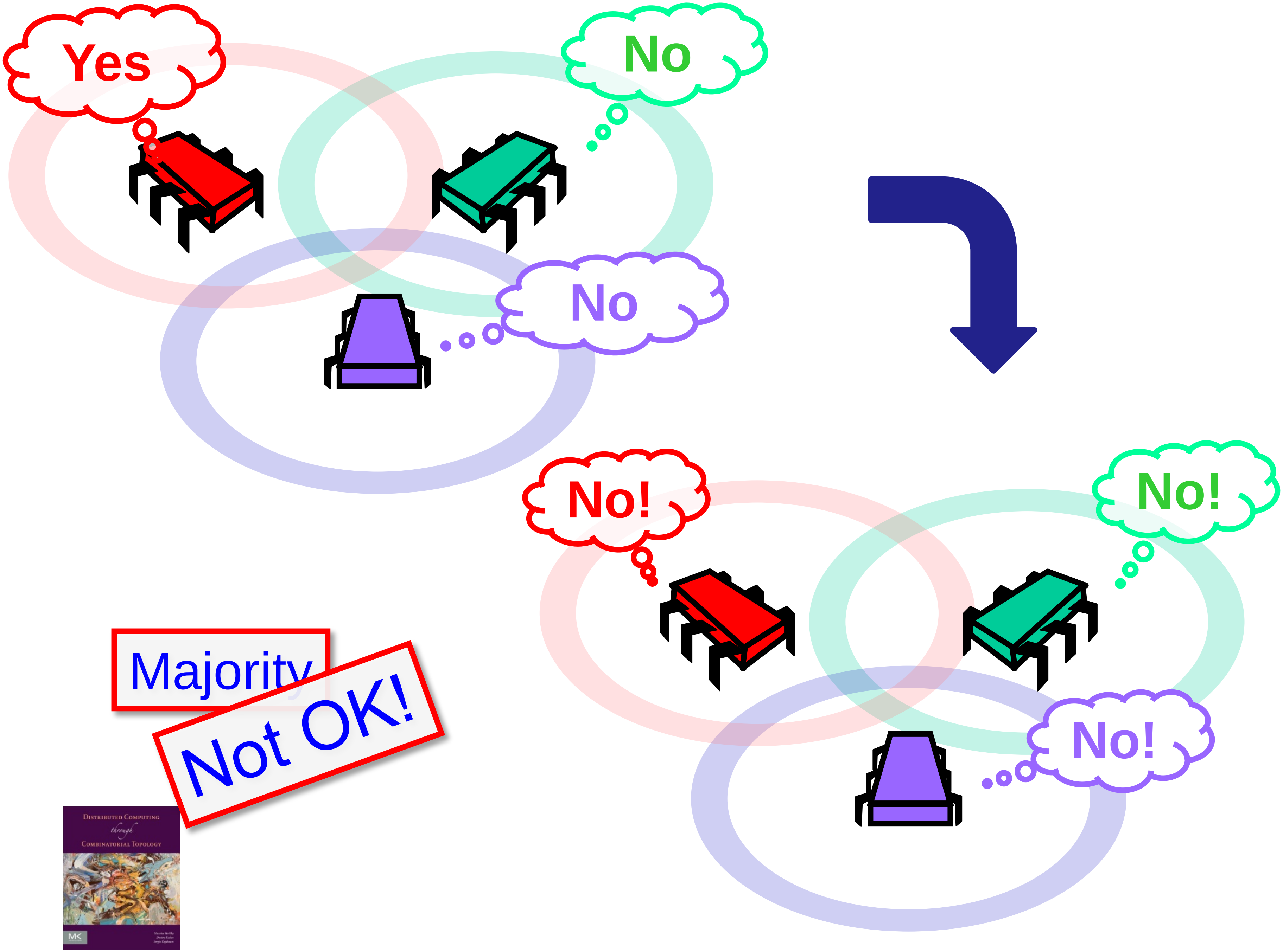
Non-Examples

Weak Symmetry-Breaking

When all participate …

At least one on group 0, group 1



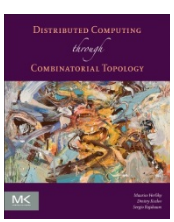

Yes
No
No
No!
No!
No!
Majority
Not OK!
Distributed Computing through Combinatorial Topology
MK

Road Map

Operational Model

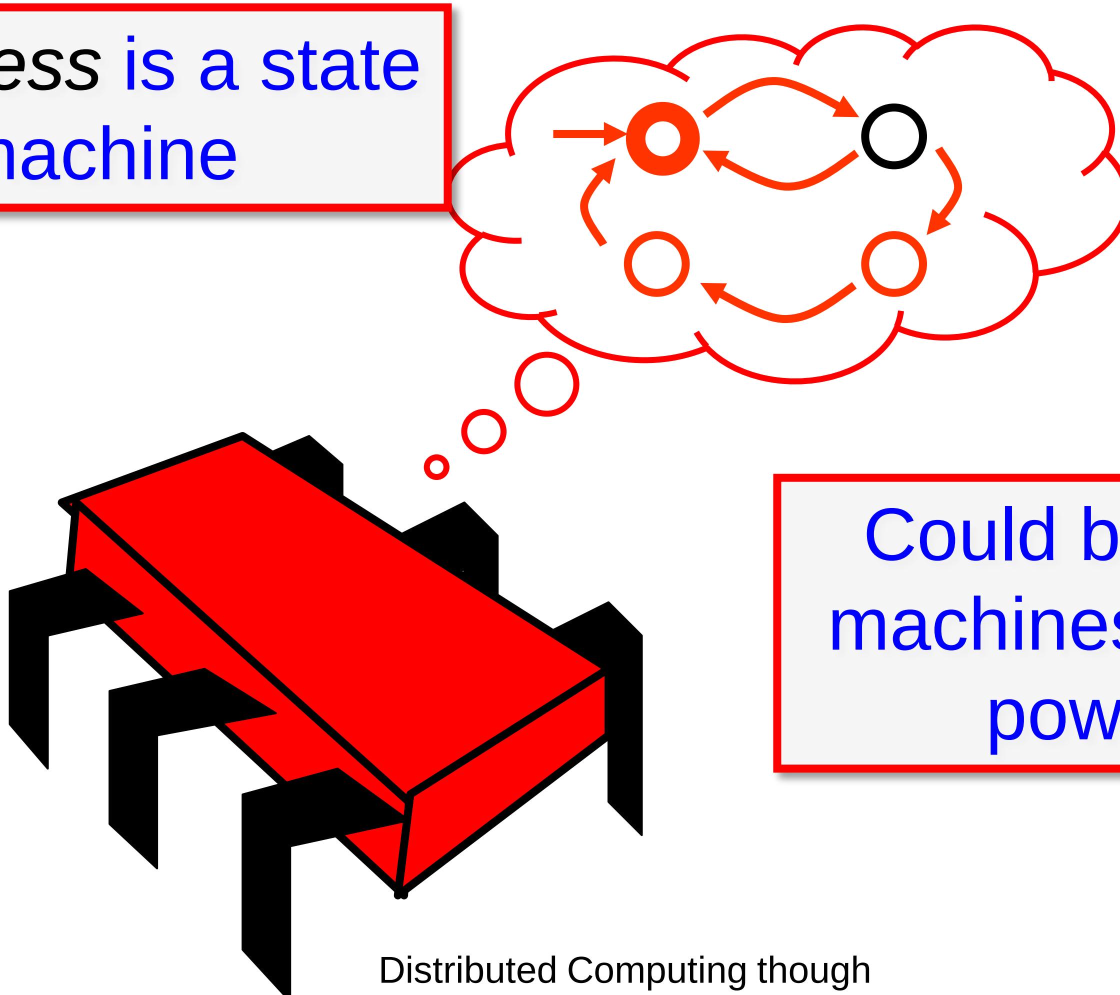
Combinatorial Model

Main Theorem

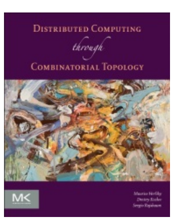


Processes

A *process* is a state machine

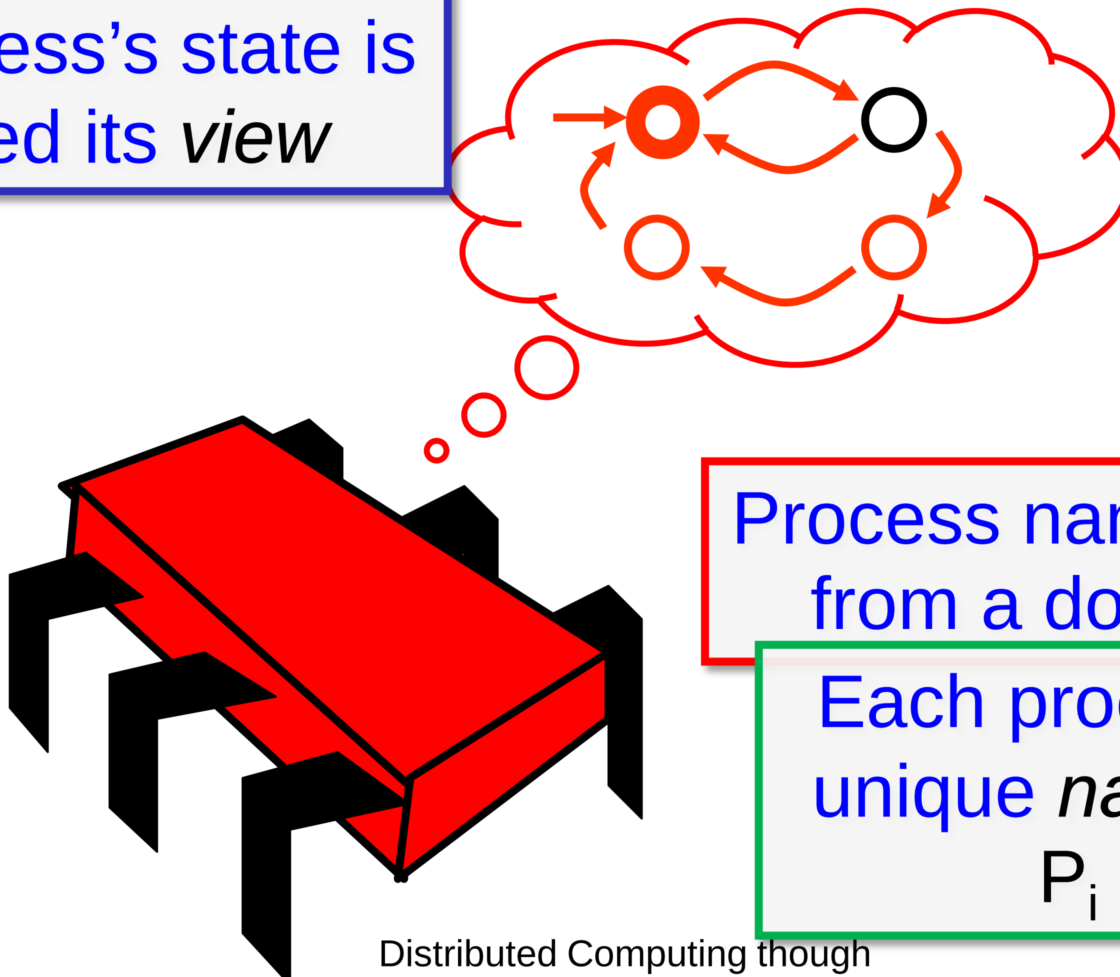


Could be Turing machines or more powerful



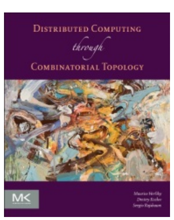
Processes

A process's state is called its *view*



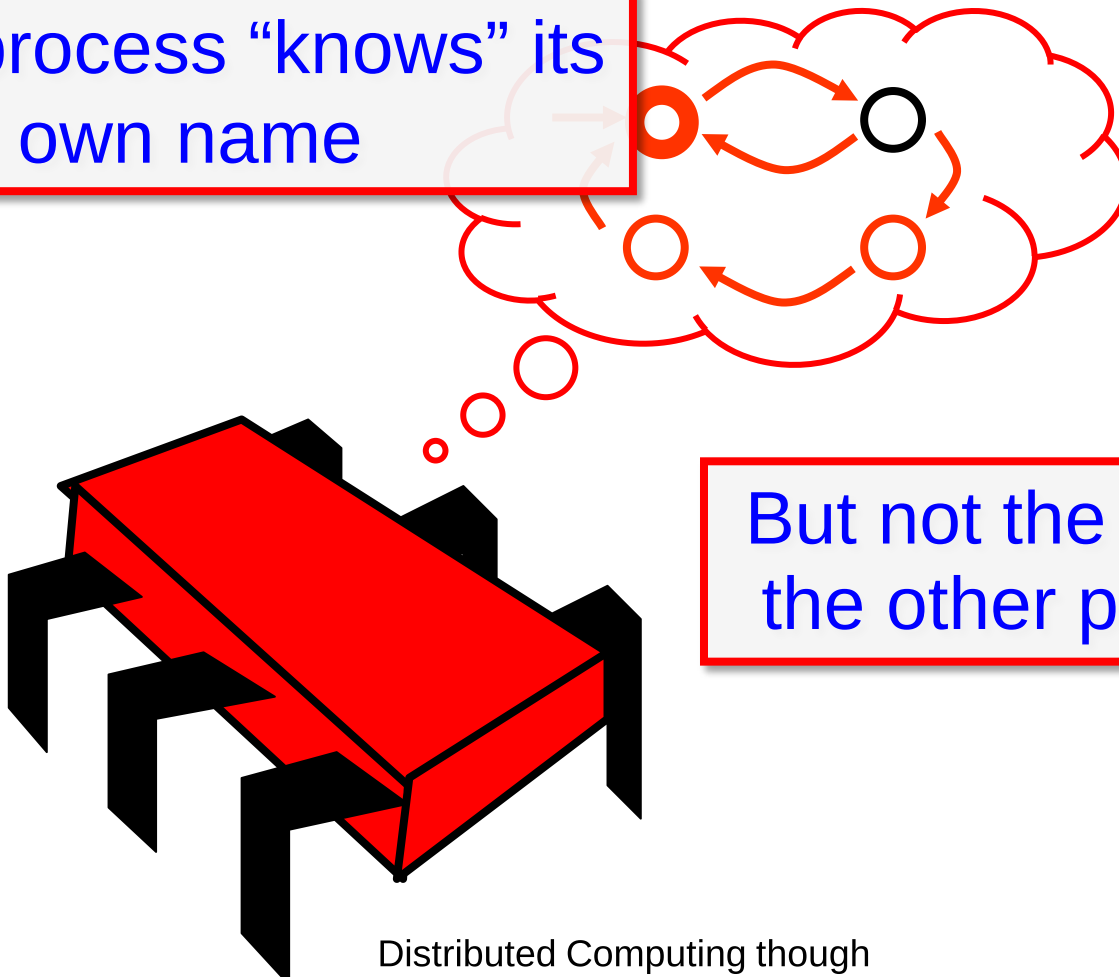
Process names taken from a domain Π

Each process has a unique *name* (color) $P_i \in \Pi$

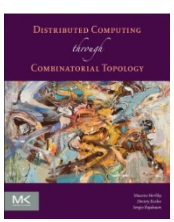


Processes

Each process “knows” its own name

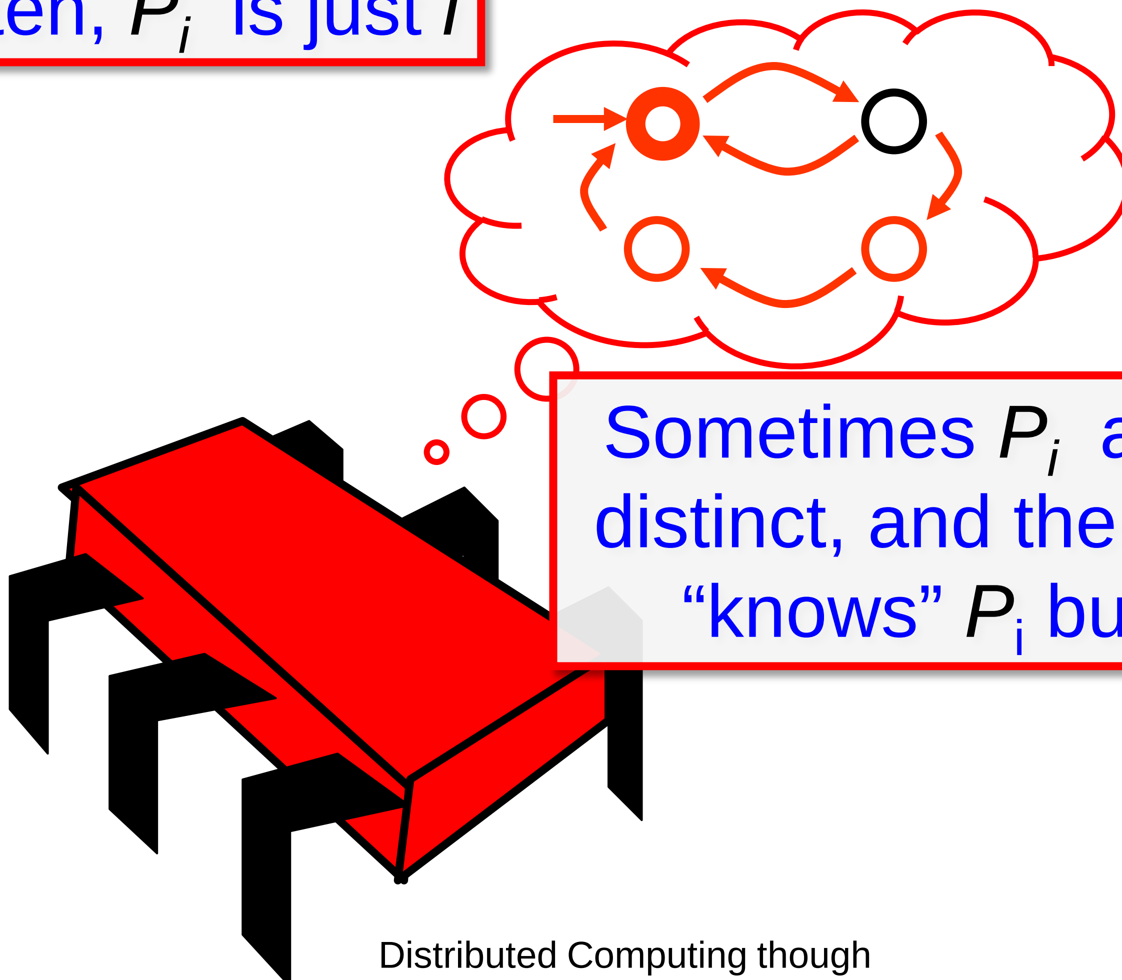


But not the names of the other processes

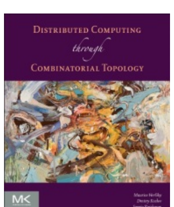


Processes

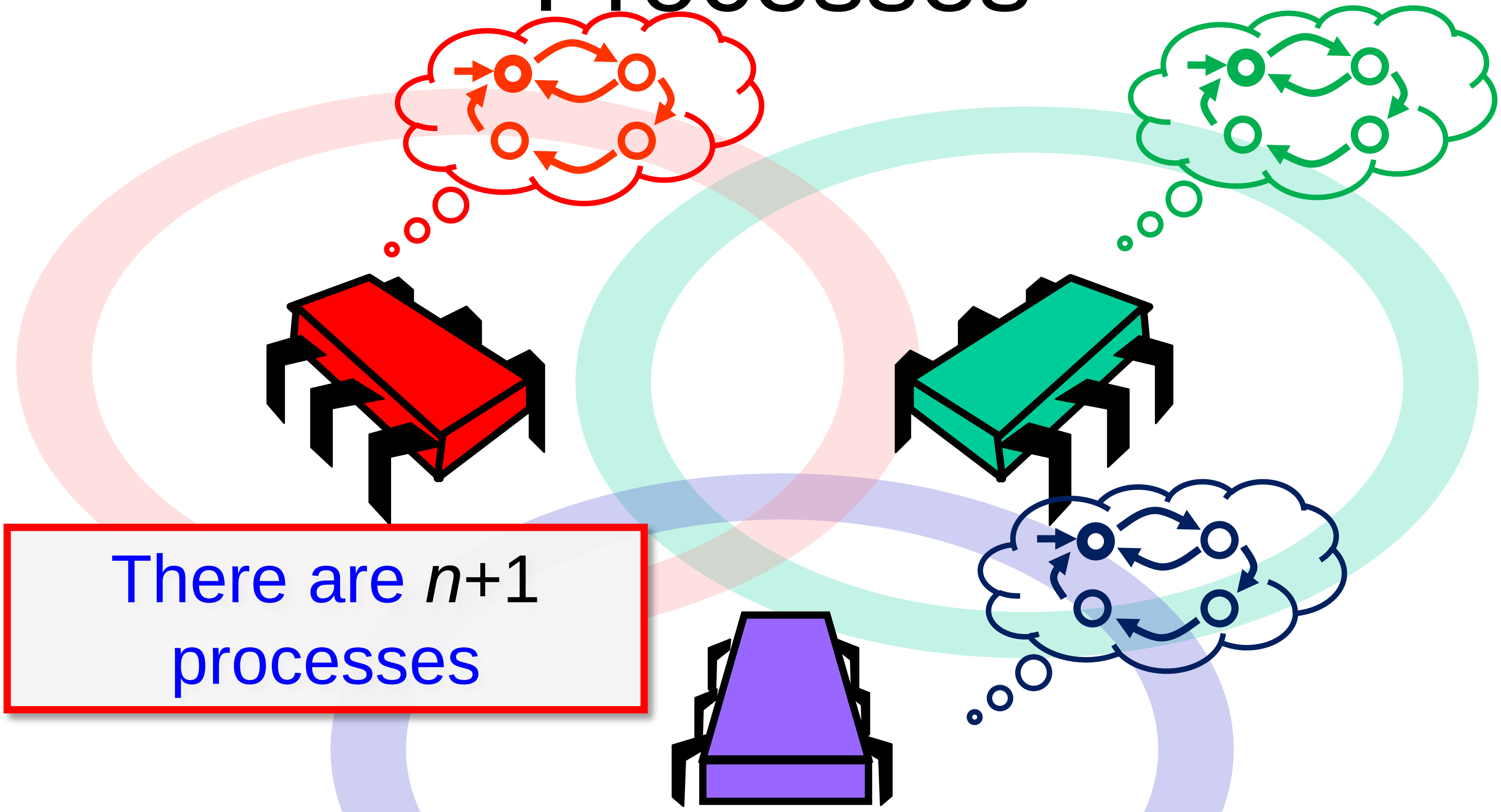
Often, P_i is just i



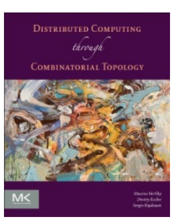
Sometimes P_i and i are distinct, and the process "knows" P_i but not i



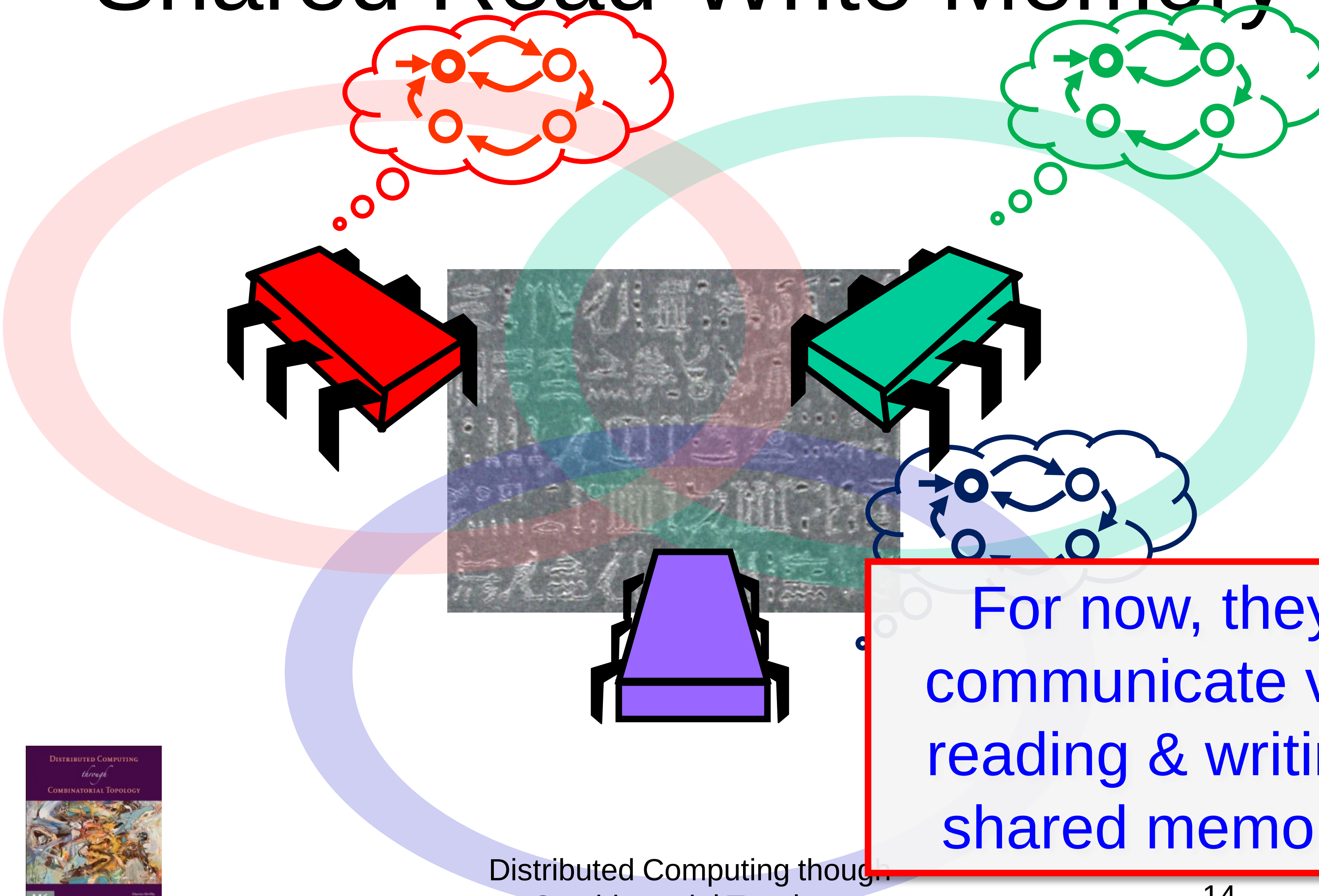
Processes



There are n+1 processes



Shared Read-Write Memory



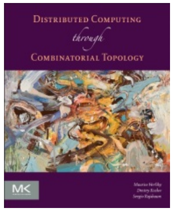
For now, they communicate via reading & writing shared memory

Immediate Snapshot

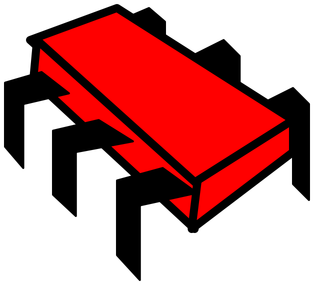
Individual reads & writes are too low-level ...

A *snapshot* = atomic read of all memory

We will use *immediate snapshot* ...



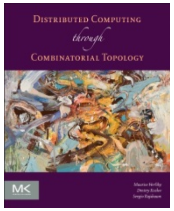
Immediate Snapshot



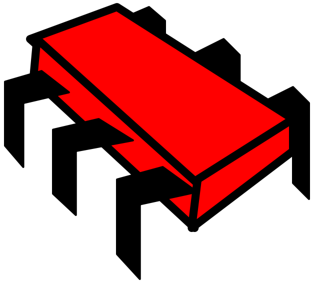
write view to memory

take snapshot

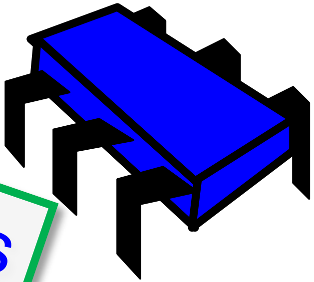
adjacent steps!



Immediate Snapshot



Concurrent steps

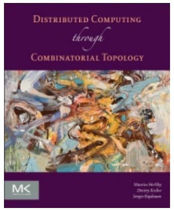


write view to memory

take snapshot

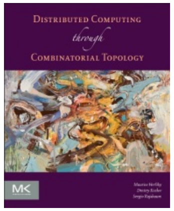
write view to memory

take snapshot



Immediate Snapshot

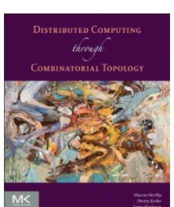
```
immediate
  mem[i] := view;
  snap := snapshot(mem[*])
```



P	Q	R
write		
snap		
	write	
	snap	
		write
		snap
{p}	{p,q}	{p,q,r}

P	Q	R
write		
snap		
	write	write
	snap	snap
{p}	{p,q,r}	{p,q,r}

P	Q	R
write	write	write
snap	snap	snap
{p,q,r}	{p,q,r}	{p,q,r}



Realistic?

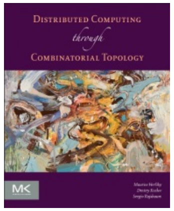
No!

My laptop reads only a few contiguous memory words at a time

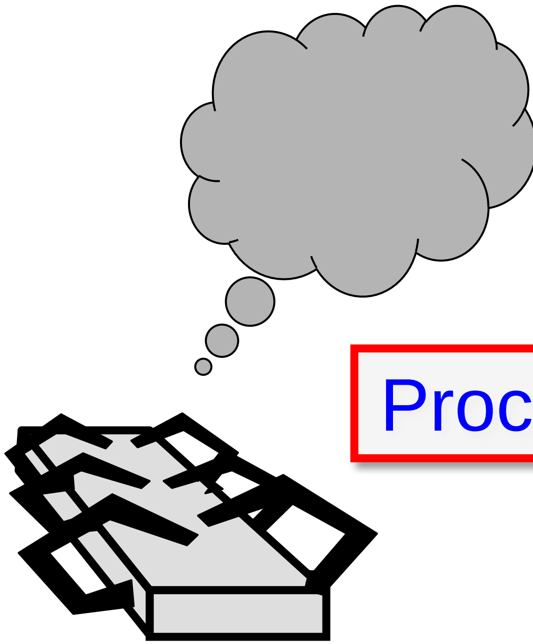
Yes!

Simpler lower bounds: if it's impossible with IS, it's impossible on your laptop.

Can implement IS from read-write

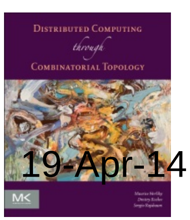


Crashes

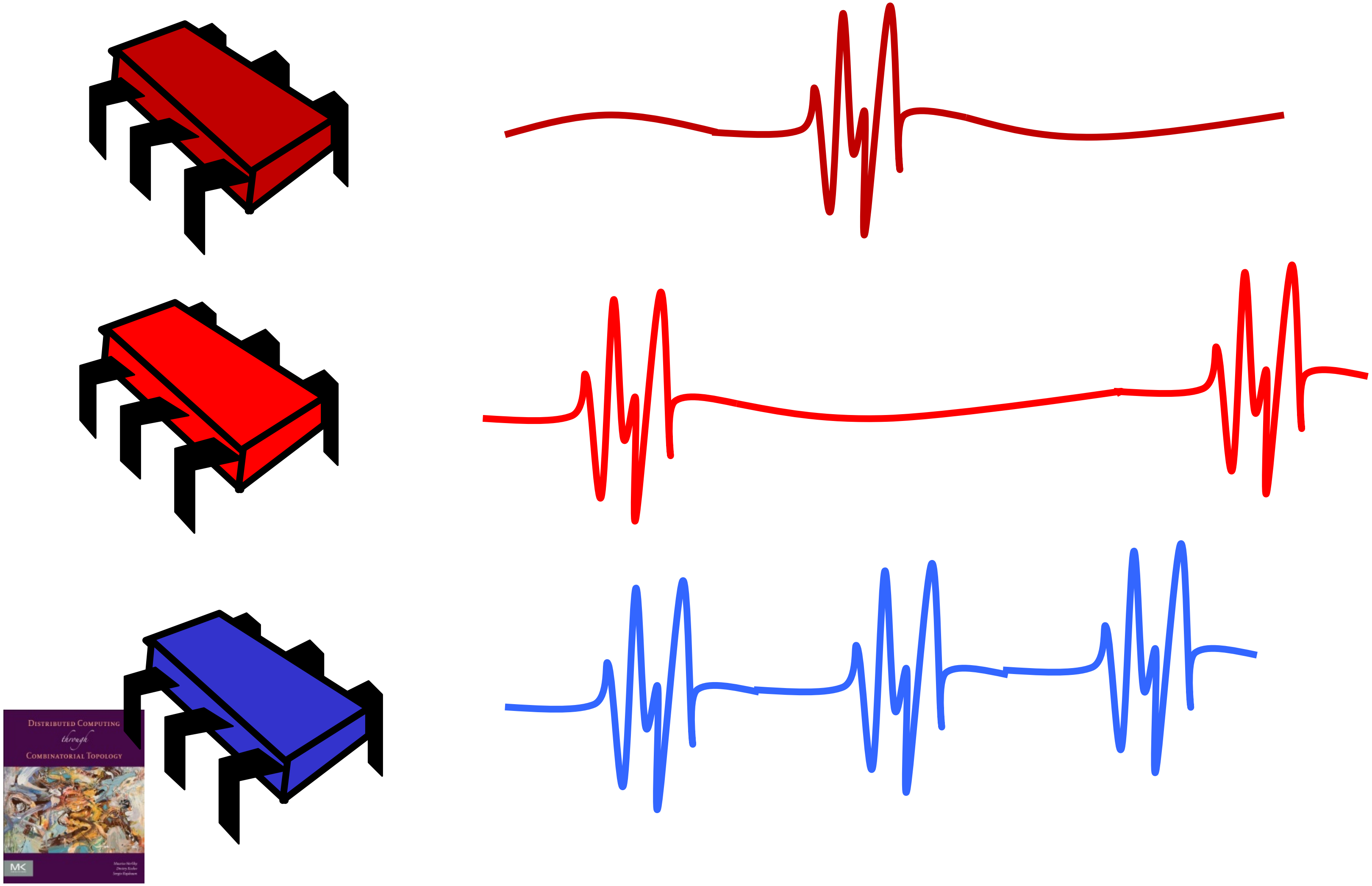


Processes may halt without warning

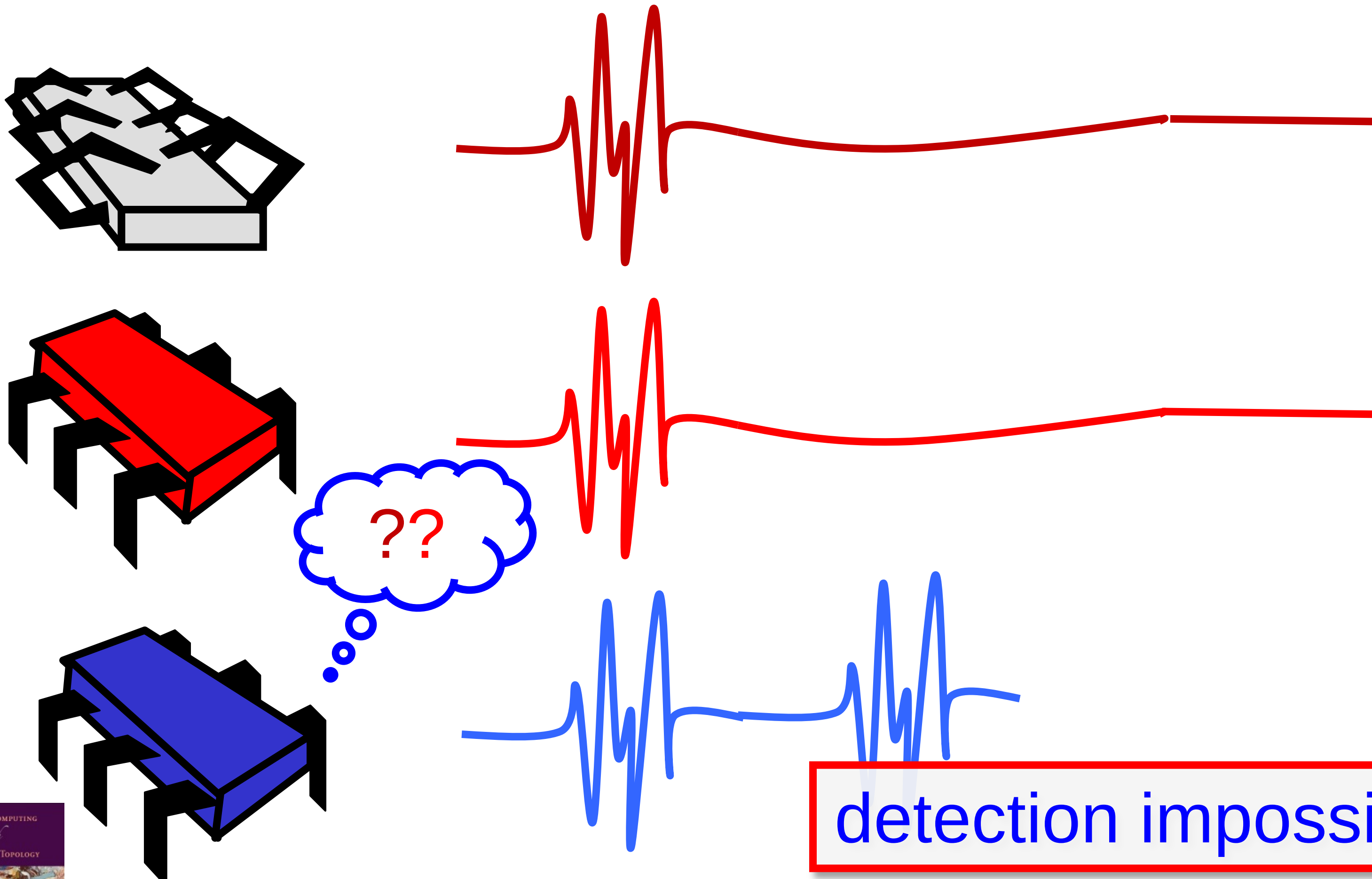
as many as n out of n+1



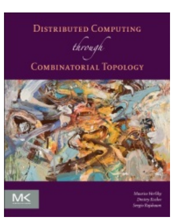
Asynchronous



Asynchronous Failures



detection impossible



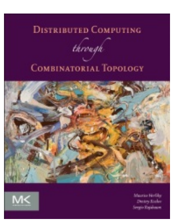
Configurations

$$\boxed{C = \{s_0, \ldots, s_n\}}$$

set of *simultaneous* process states

initial configurations

final configurations



Executions

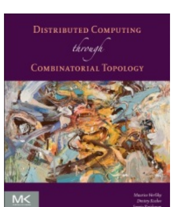
processes that communicate

$C_0, S_0, C_1, S_1, \ldots, S_r, C_{r+1}$

initial configuration

final configuration

next configuration



Executions

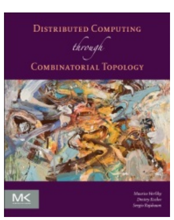
triple is a *concurrent step*

$C_0, S_0, C_1, S_1, \ldots, S_r, C_{r+1}$

Processes in S_0 said to *participate* in step

Only $P_i \in S_0$ can change between C_0 and C_1

state change result of communication



Executions

finite

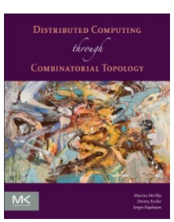
$C_0, S_0, C_1, S_1, \ldots, S_r, C_{r+1}$

infinite

$C_0, S_0, C_1, S_1, \ldots,$

partial

$C_0, S_0, C_1, S_1, \ldots, S_r, C_{r+1}$



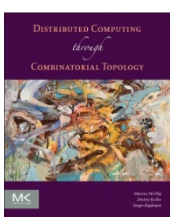
Crashes are Implicit

$$C_0,\ S_0,\ C_1,\ S_1,\ \ldots,\ S_r,\ C_{r+1}$$

If P_i's state not final in the final configuration

then P_i has *crashed*.

crash cannot be detected in finite time



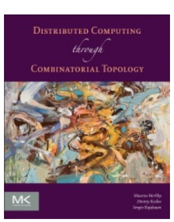
Colorless Tasks

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

(colorless) input assignment

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

(colorless) output assignment



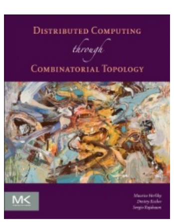
Input Assignments

$$\{(P_j, v_j), \mid P_j \in \Pi, v_j \in V^{in}\}$$

name, value pair

domain of *process names*

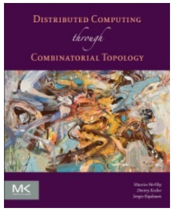
domain of *input values*



Colorless Input Assignments

$$\{(P_j, v_j), \mid P_j \in \Pi, v_j \in V^{in}\}$$

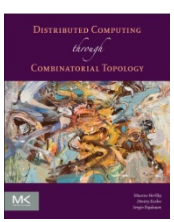
discard process names, keep values



(Colorless) Output Assignments

$$\{(P_j, v_j), \mid P_j \in \Pi, v_j \in V^{out}\}$$

$$\{(P_j, v_j), \mid P_j \in \Pi, v_j \in V^{out}\}$$



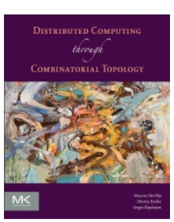
Example: Binary Consensus

$$\mathcal{I} = \{\{0\}, \{1\}, \{0,1\}\}$$

All start with 0

All start with 1

start with both



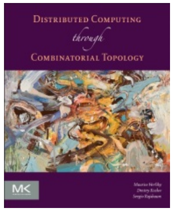
Example: Binary Consensus

$$\mathcal{I} = \{\{0\}, \{1\}, \{0,1\}\}$$

$$\mathcal{O} = \{\{0\}, \{1\}\}$$

All decide 0

All decide 1



Example: Binary Consensus

$$\Delta(\{0\}) = \{\{0\}\}$$

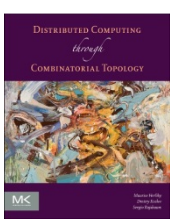
All start with 0,
all decide 0

Example: Binary Consensus

$\Delta(\{0\}) = \{\{0\}\}$

$\Delta(\{1\}) = \{\{1\}\}$

All start with 1,
all decide 1



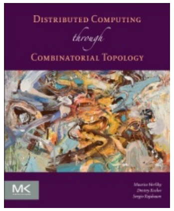
Example: Binary Consensus

$\Delta(\{0\}) = \{\{0\}\}$

$\Delta(\{1\}) = \{\{1\}\}$

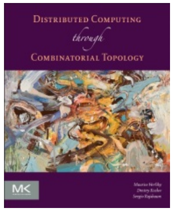
$\Delta(\{0,1\}) = \{\{0\},\{1\}\}$

with mixed inputs,
all decide 0,
or all decide 1



Colorless Layered Protocol

```
shared mem array 0..N-1,0..n of Value
view := input
for l := 0 to N-1 do
  immediate
    mem[l][i] := view;
    snap := snapshot(mem[l][*])
  view := set of values in snap
return δ(view)
```



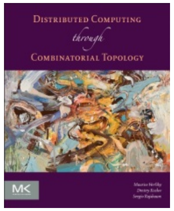
Colorless Layered Protocol

```
shared mem array 0..N-1,0..n of Value
view := input
for j := 0 to N-1 do
  immediate
    mem[j][i] := view;
    snap := snapshot(mem[j][*])
  view := set of values in snap
return δ(view)
```

2-dimensional memory array

row is clean *per-layer* memory

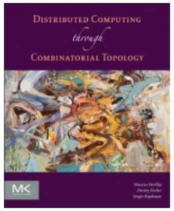
column is *per-process* word



Colorless Layered Protocol

```
shared mem array 0..N-1,0..n of Value
view := input
for j := 0 to N-1 do
  immediate
    mem[j][i] := view;
    snap := snapshot(mem[j][*])
  view := set of values in snap
return δ(view)
```

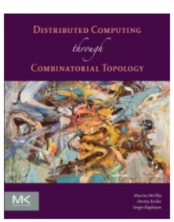
initial view is input value



Colorless Layered Protocol

```
shared mem array 0..N-1,0..n of Value
view := input
for l := 0 to N-1 do
  immediate
    mem[j][i] := view;
    snap := snapshot(mem[j][*])
  view := set of values in snap
return δ(view)
```

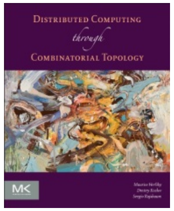
run for *N* layers



Colorless Layered Protocol

layer l : immediate write & snapshot of row l

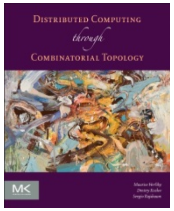
```
shared mem array 0..N-1,0..n of Value
view := input
for j := 0 to N-1 do
  immediate
    mem[l][i] := view;
    snap := snapshot(mem[l][*])
  view := set of values in snap
return δ(view)
```



Colorless Layered Protocol

```
shared mem array 0..N-1,0..n of Value
view := input
for j := 0 to N-1 do
  immediate
    mem[j][i] := view;
    snap := snapshot(mem[j][*])
  view := set of values in snap
return δ(view)
```

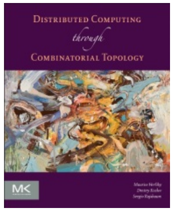
new view is set of values seen

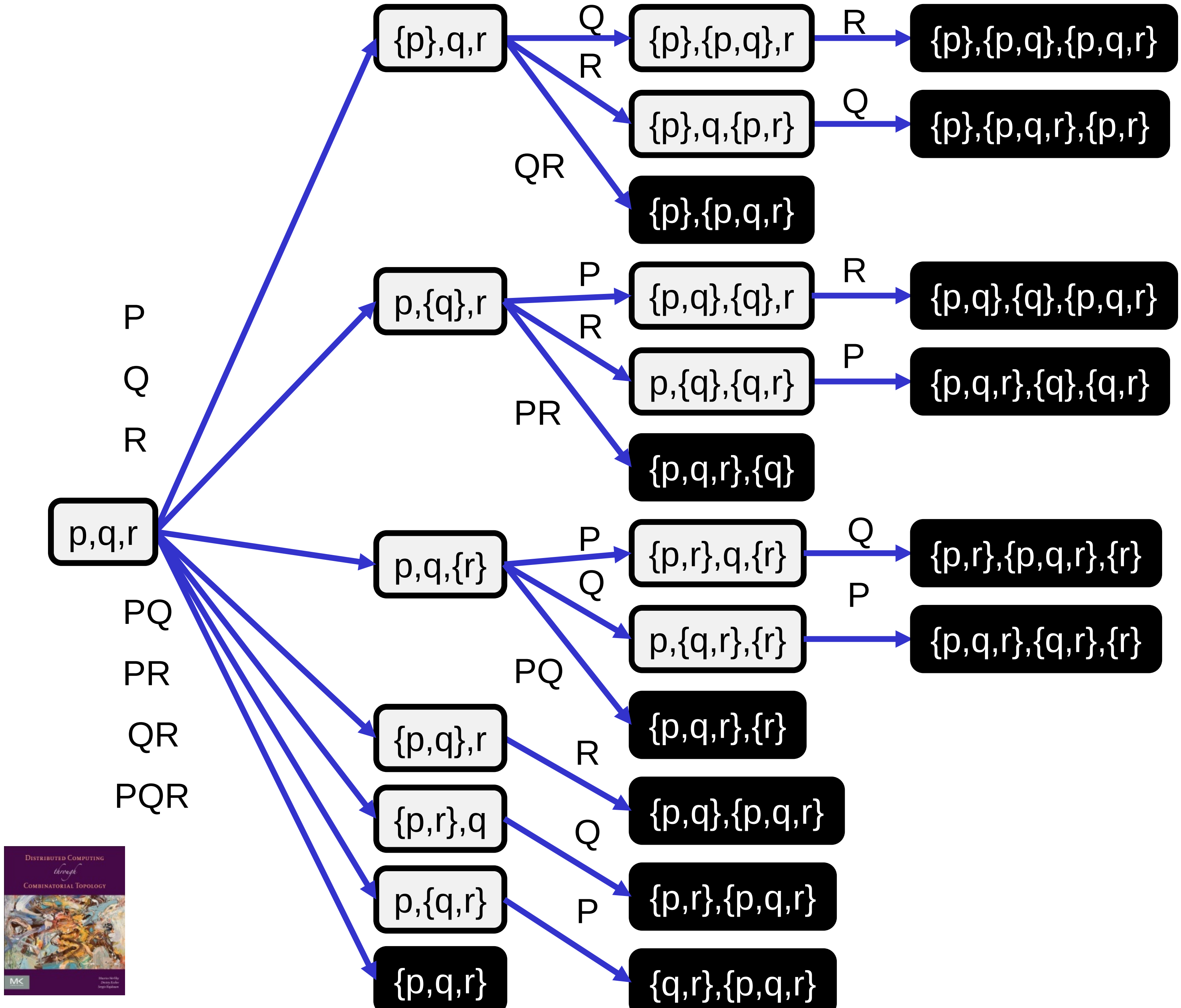


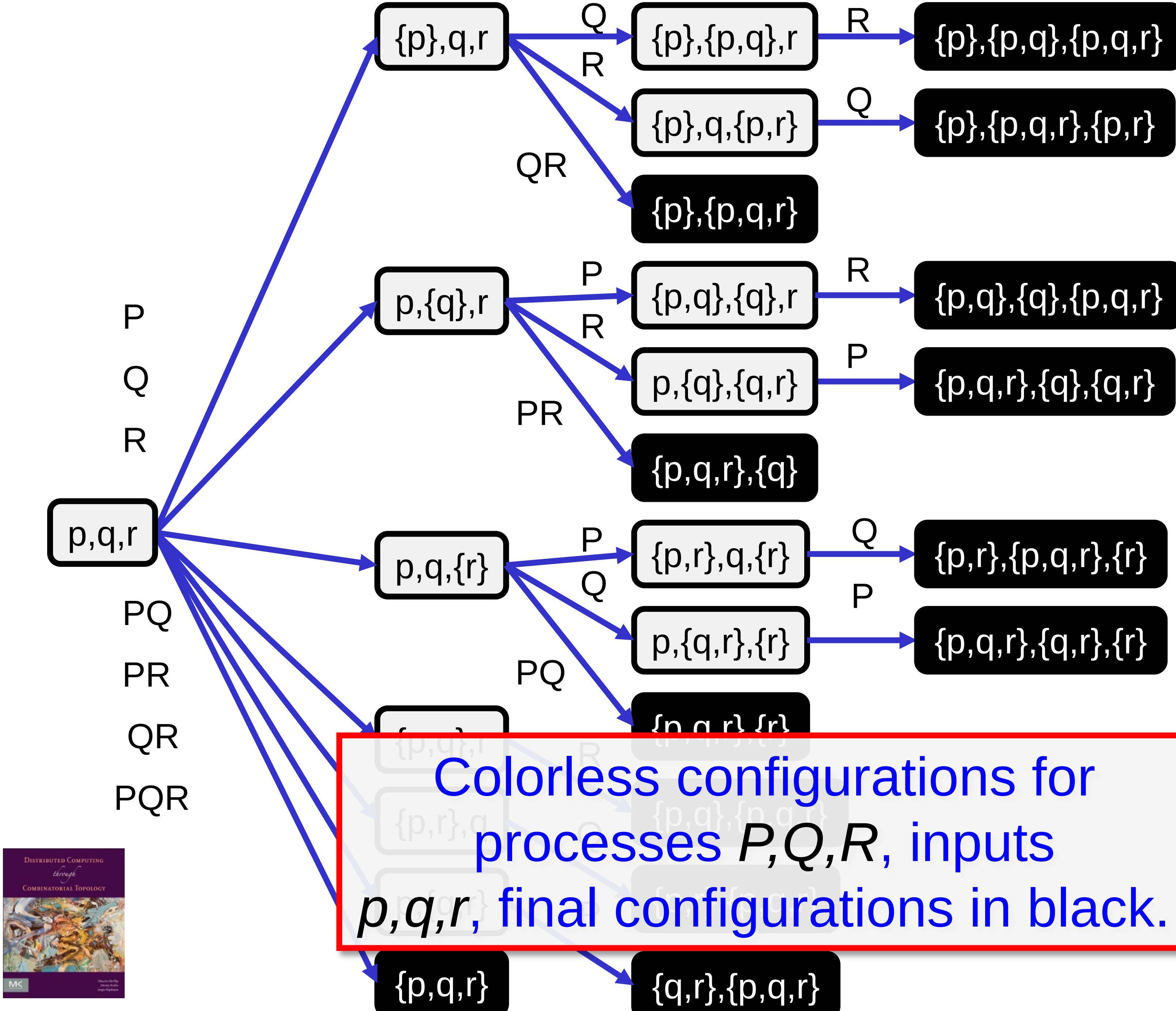
Colorless Layered Protocol

```
shared mem array 0..N-1,0..n of Value
view := input
for j := 0 to N-1 do
  immediate
    snap := snapshot(mem[j][*])
  view := set of values in snap
return δ(view)
```

finally apply decision value to final view




p,q,r
P
Q
R
PQ
PR
QR
PQR
{p},q,r
p,{q},r
p,q,{r}
{p,q},r
{p,r},q
p,{q,r}
{p,q,r}
Q
R
QR
{p},{p,q},r
{p},q,{p,r}
{p},{p,q,r}
R
Q
{p},{p,q},{p,q,r}
{p},{p,q,r},{p,r}
P
R
PR
{p,q},{q},r
p,{q},{q,r}
{p,q,r},{q}
R
P
{p,q},{q},{p,q,r}
{p,q,r},{q},{q,r}
P
Q
PQ
{p,r},q,{r}
p,{q,r},{r}
{p,q,r},{r}
Q
P
{p,r},{p,q,r},{r}
{p,q,r},{q,r},{r}
R
Q
P
{p,q},{p,q,r}
{p,r},{p,q,r}
{q,r},{p,q,r}
Distributed Computing
through
Combinatorial Topology
MK



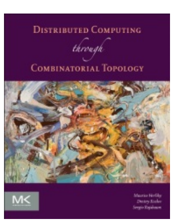
Colorless configurations for processes *P,Q,R*, inputs *p,q,r*, final configurations in black.

Road Map

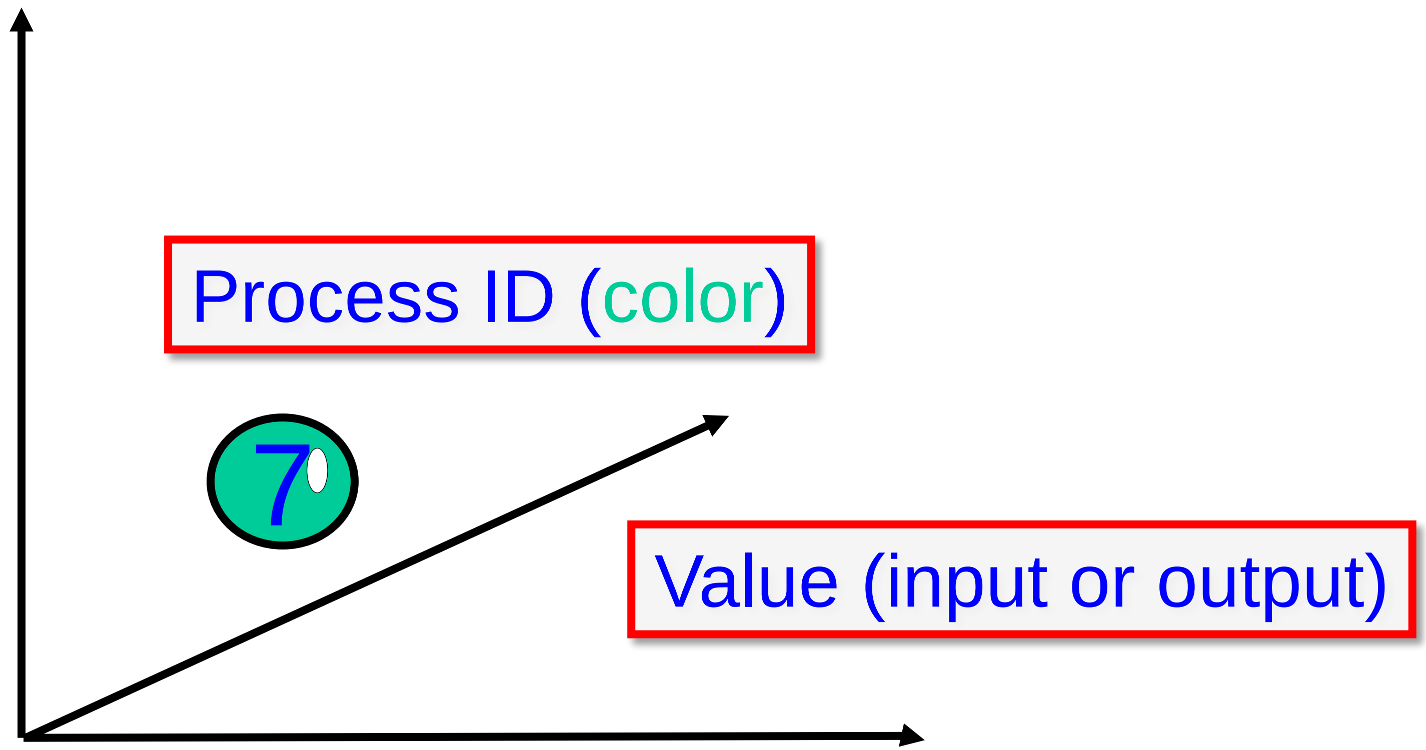
Operational Model

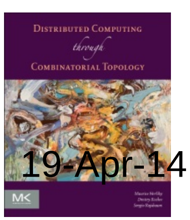
Combinatorial Model

Main Theorem

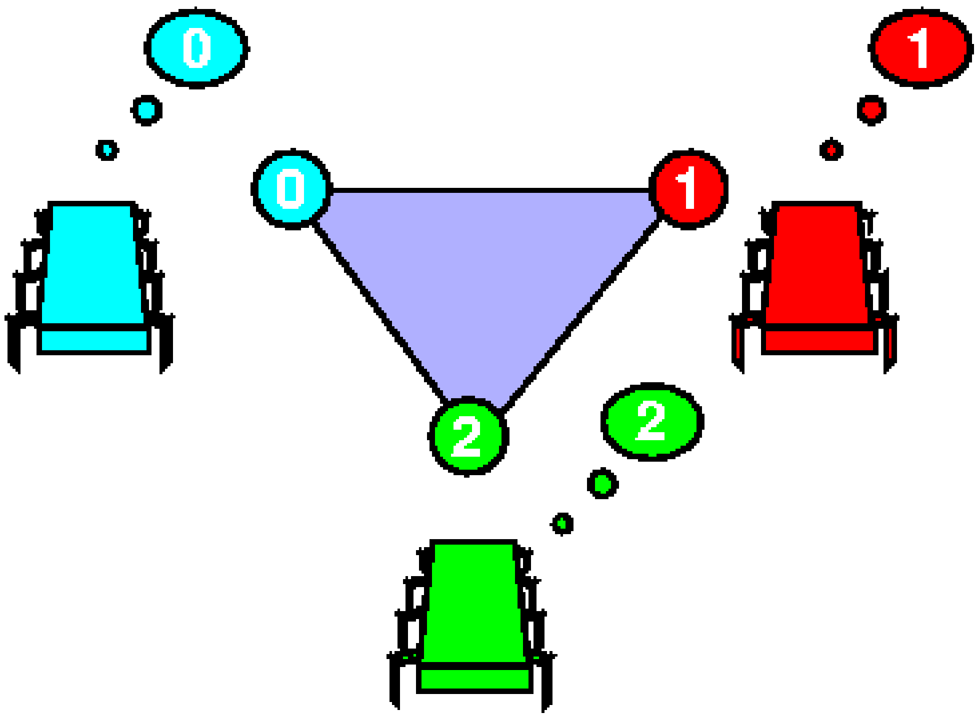


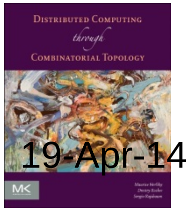
Vertex = Process State



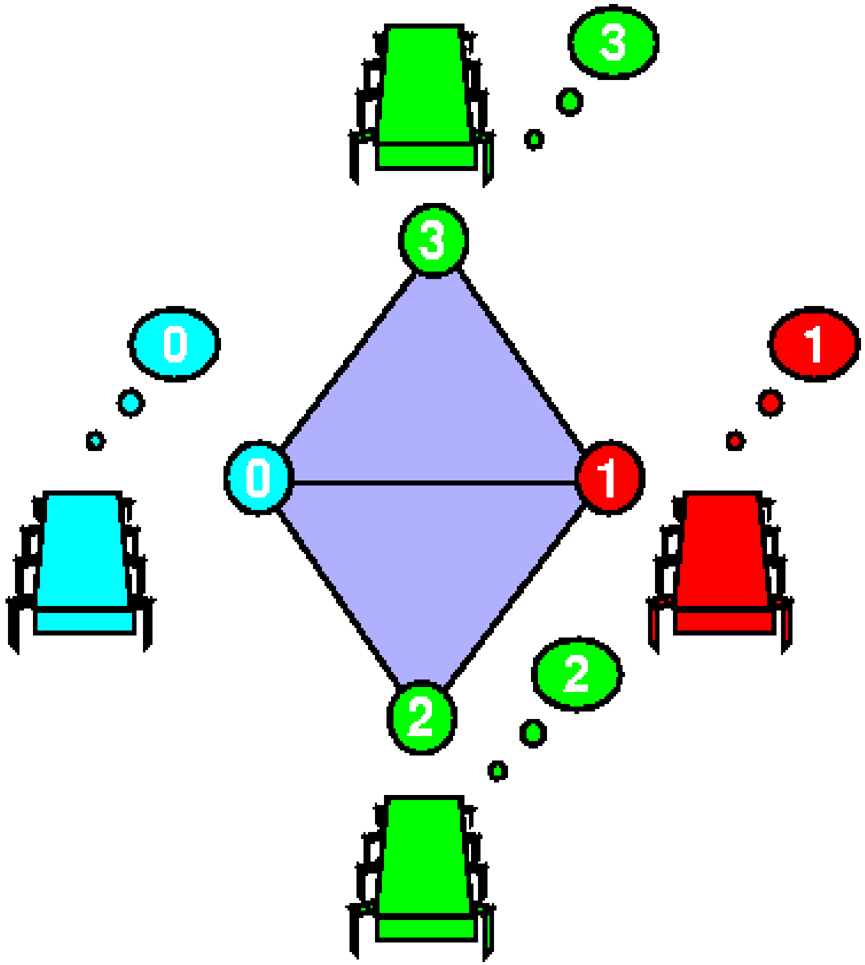


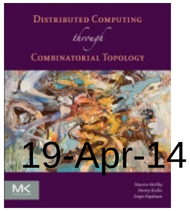
Simplex = Global State



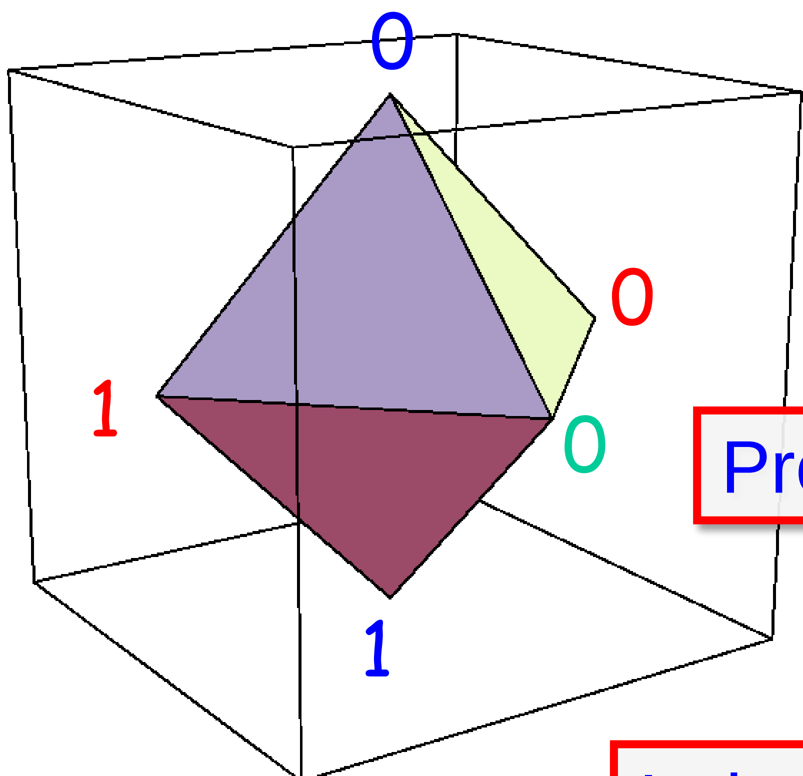


Complex = Global States





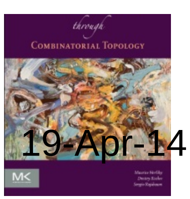
Input Complex for Binary Consensus



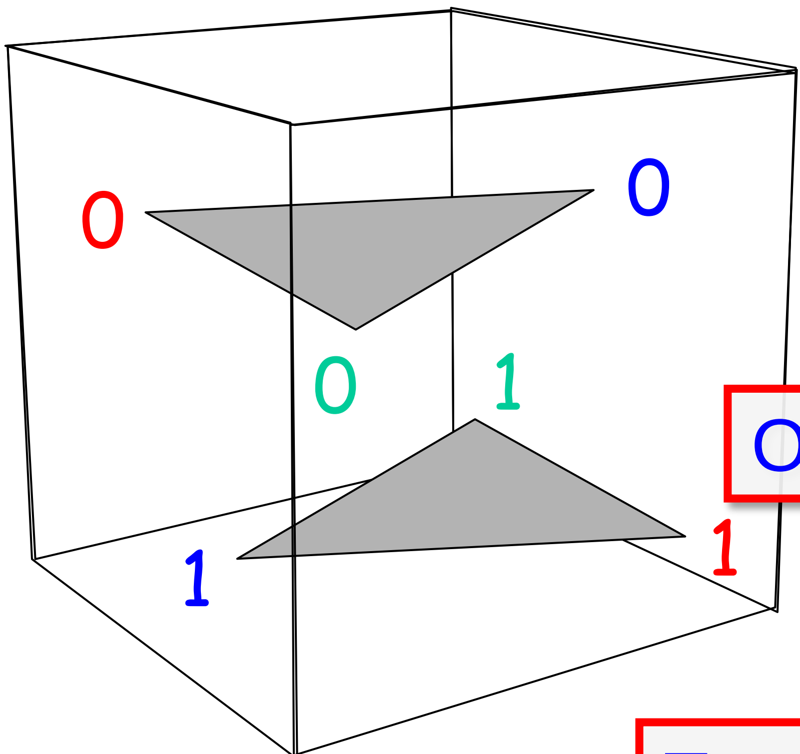
All possible initial states

Processes: red, green, blue

Independently assigned 0 or 1



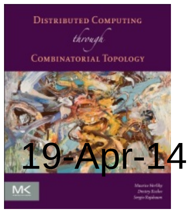
Output Complex for Binary Consensus



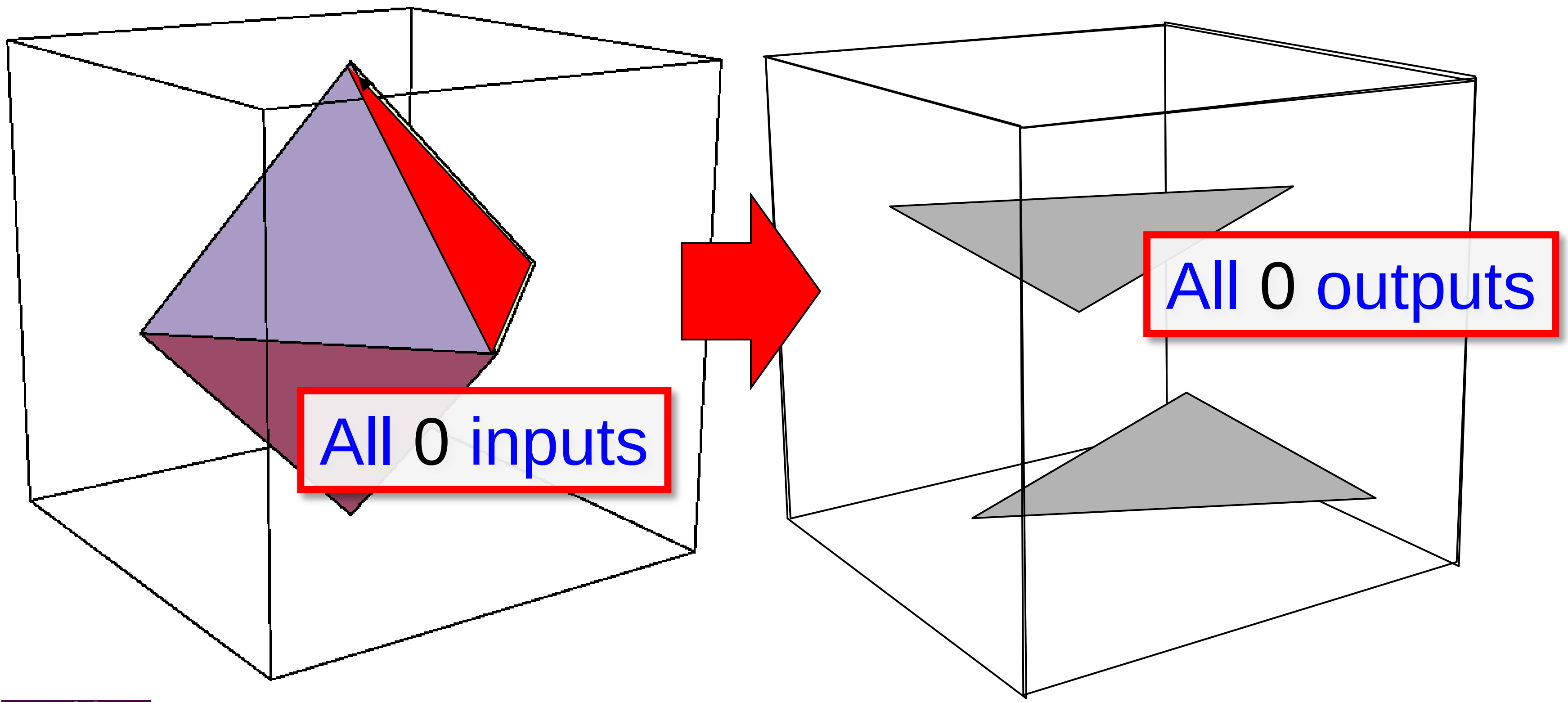
All possible final states

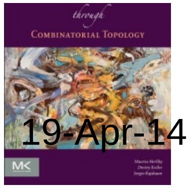
Output values all 0 or all 1

Two disconnected simplexes

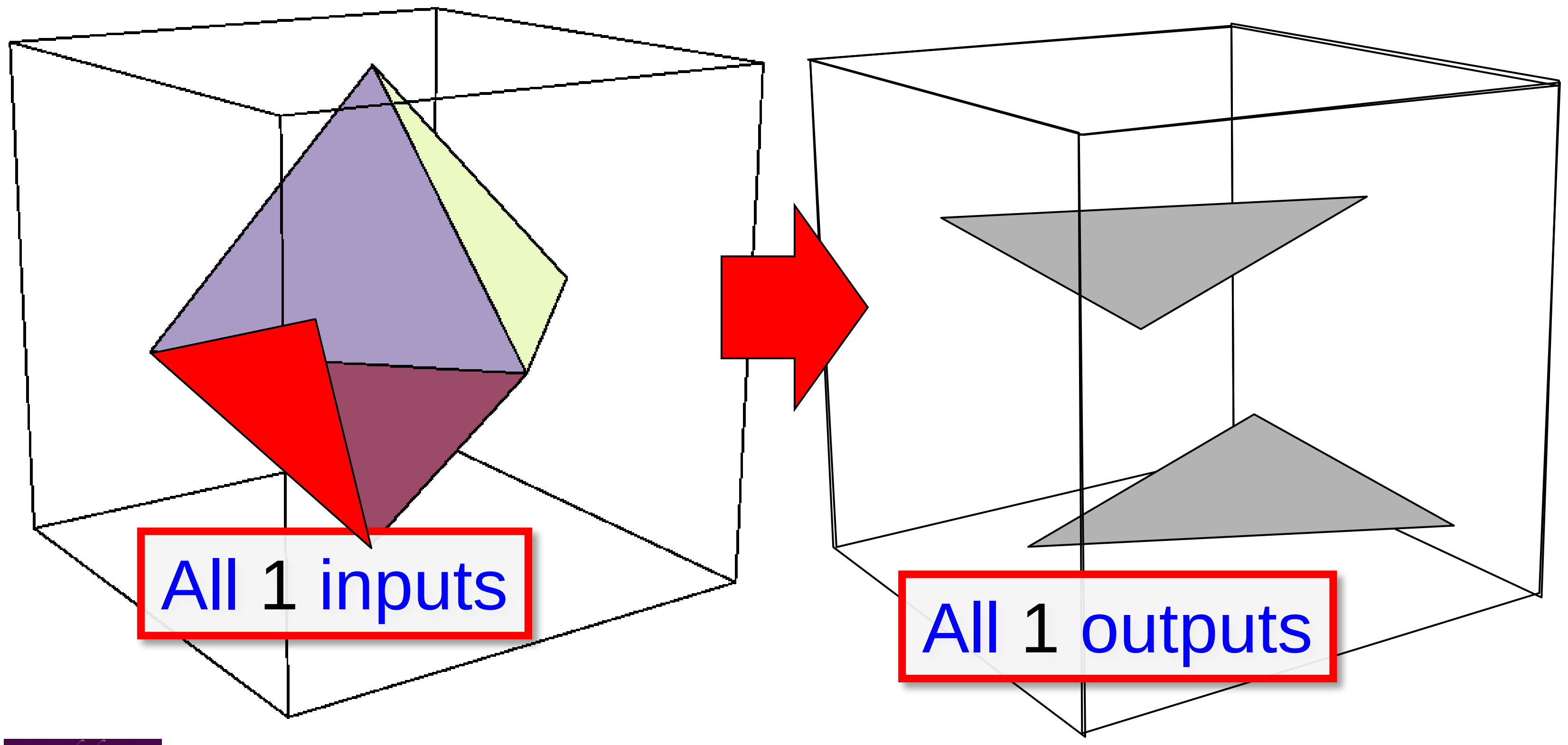


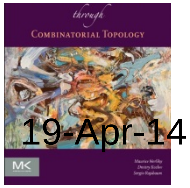
Carrier Map for Consensus



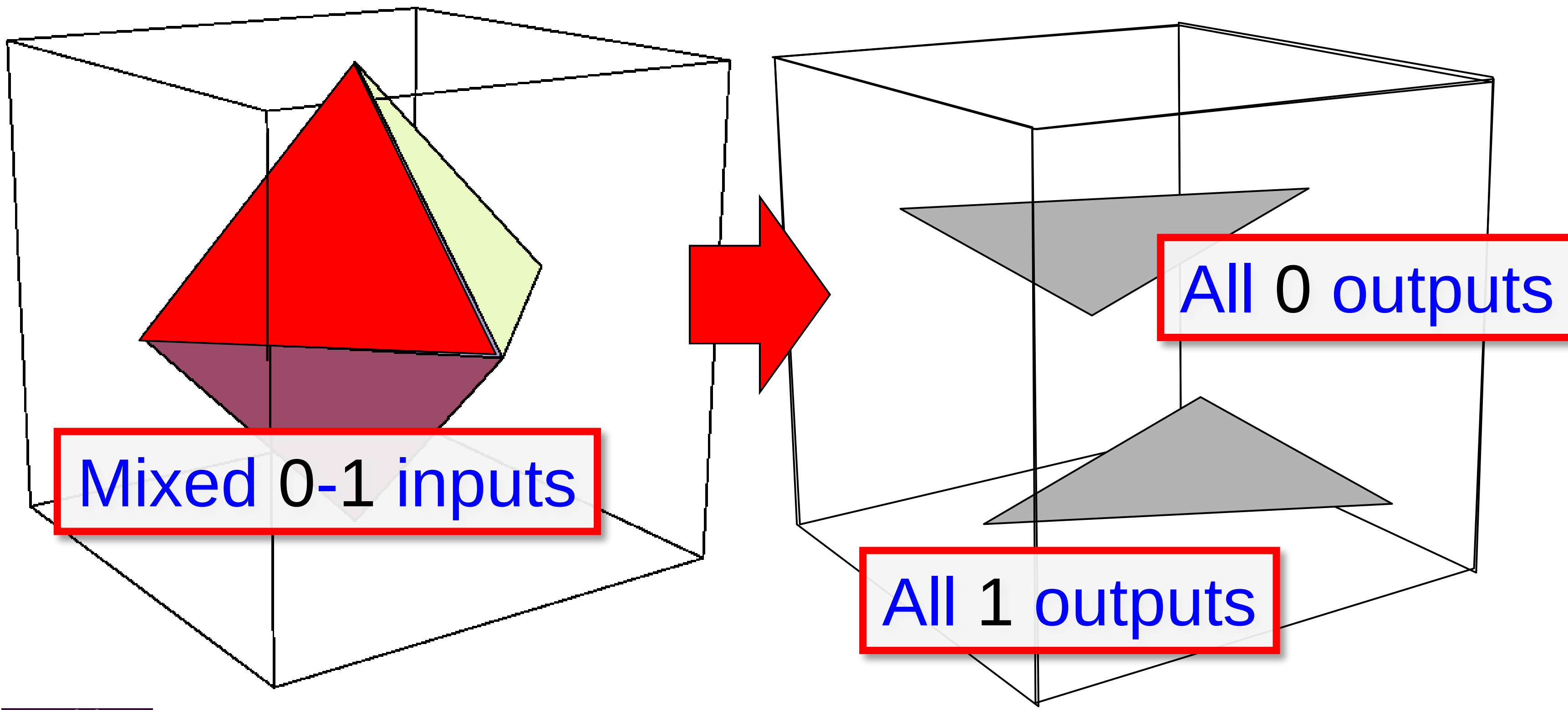


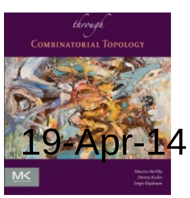
Carrier Map for Consensus





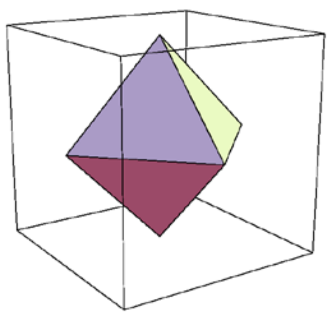
Carrier Map for Consensus



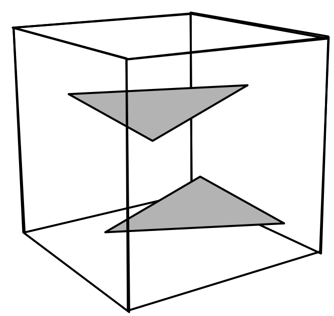


Task Specification

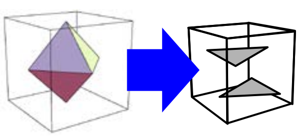
$$(\mathcal{I}, \mathcal{O}, \Delta)$$



Input complex

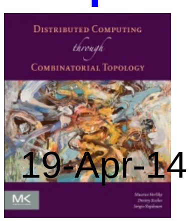


Output complex



Carrier map

Δ: $\mathcal{I} \rightarrow 2^{\mathcal{O}}$



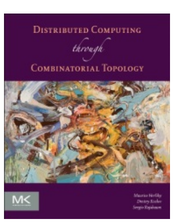
Colorless Tasks

$$(\mathcal{I}, \mathcal{P}, \Xi)$$

(colorless) input complex

strict carrier map
$\Xi: \mathcal{I} \rightarrow 2^{\mathcal{P}}$

(colorless) protocol complex



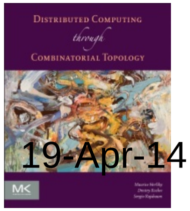
Protocol Complex

Vertex: process name, view

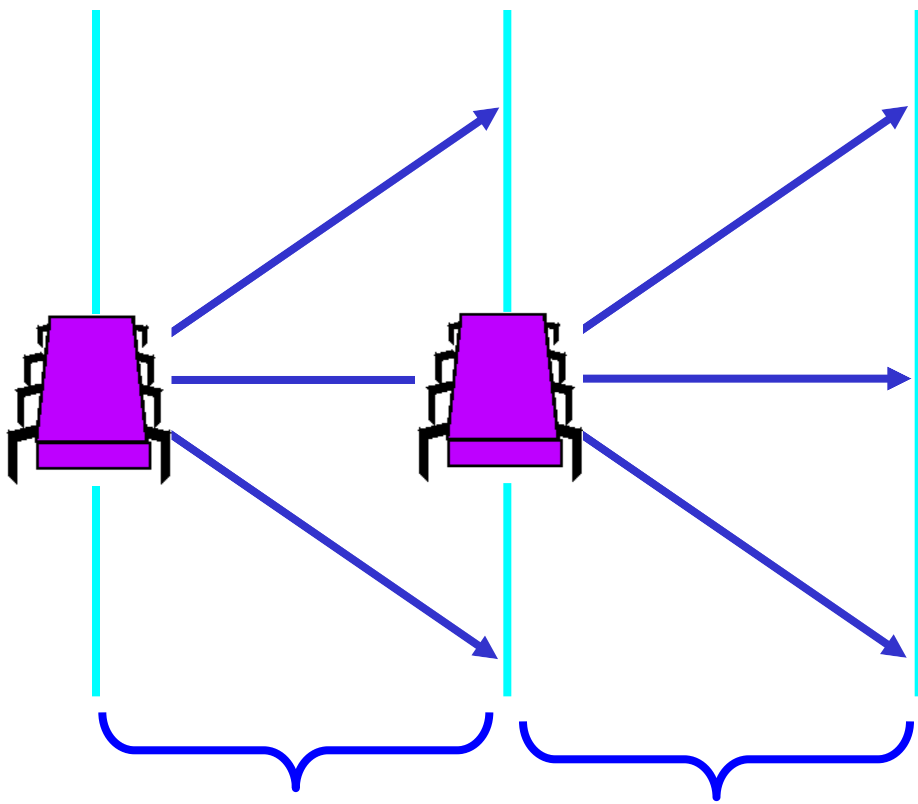
all values read and written

Simplex: compatible set of views

Each execution defines a simplex

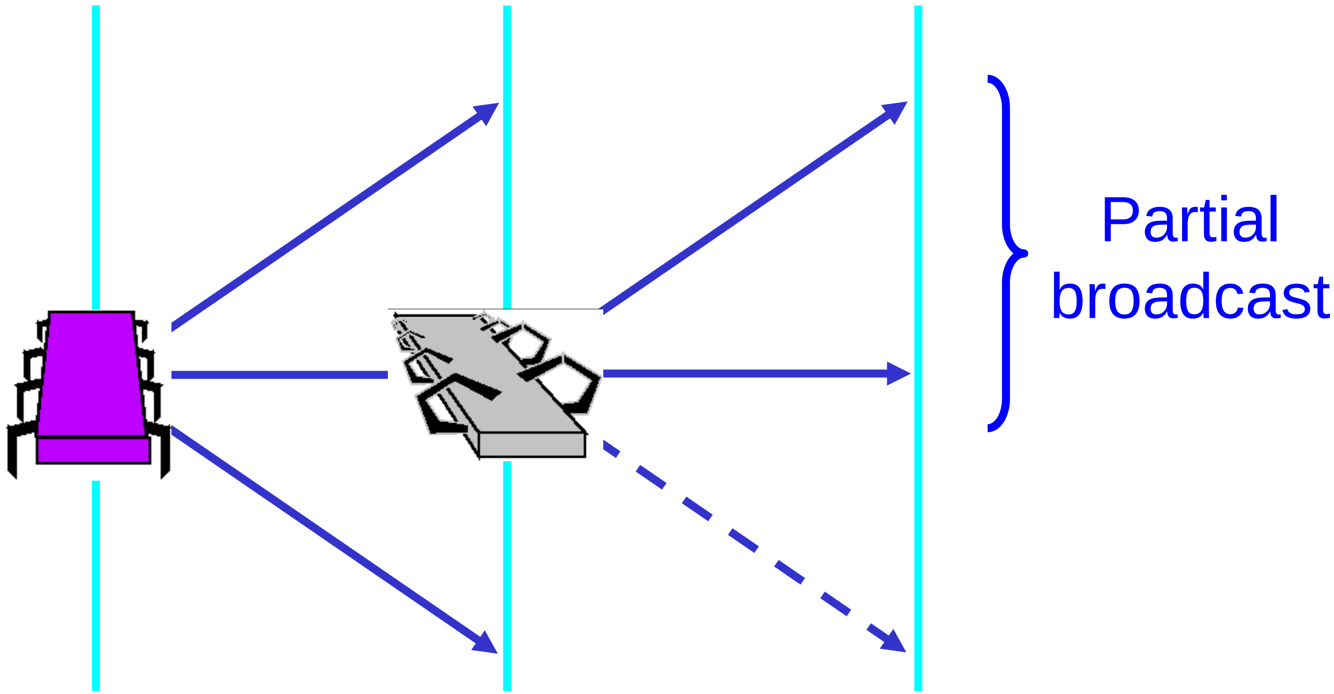


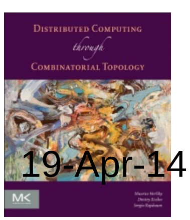
Example: Synchronous Message-Passing



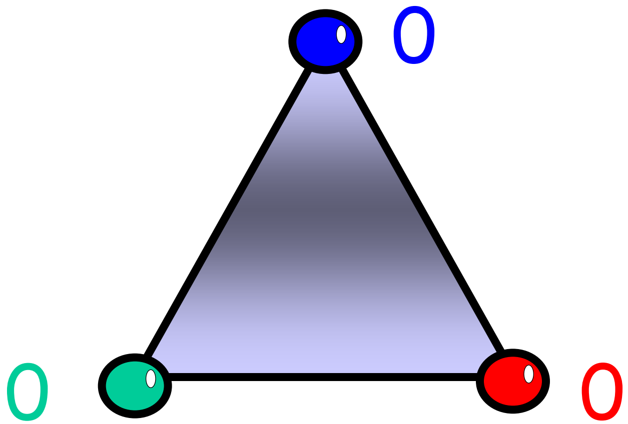
Round 0 Round 1

Failures: Fail-Stop





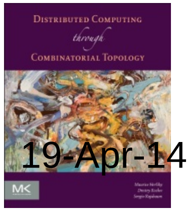
Single Input: Round Zero



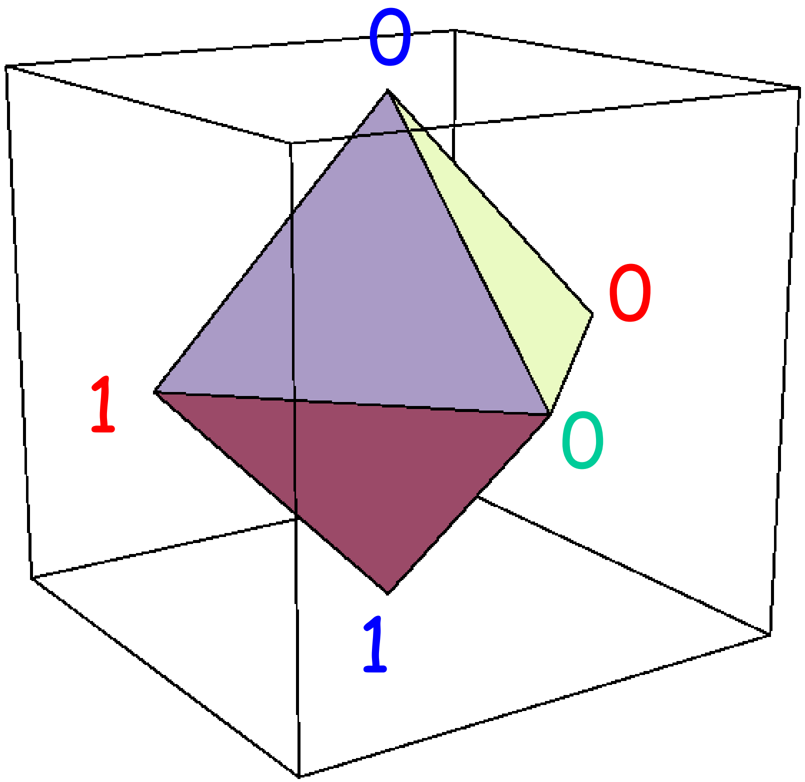
No messages sent

View is input value

Same as input simplex



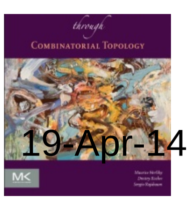
Round Zero Protocol Complex



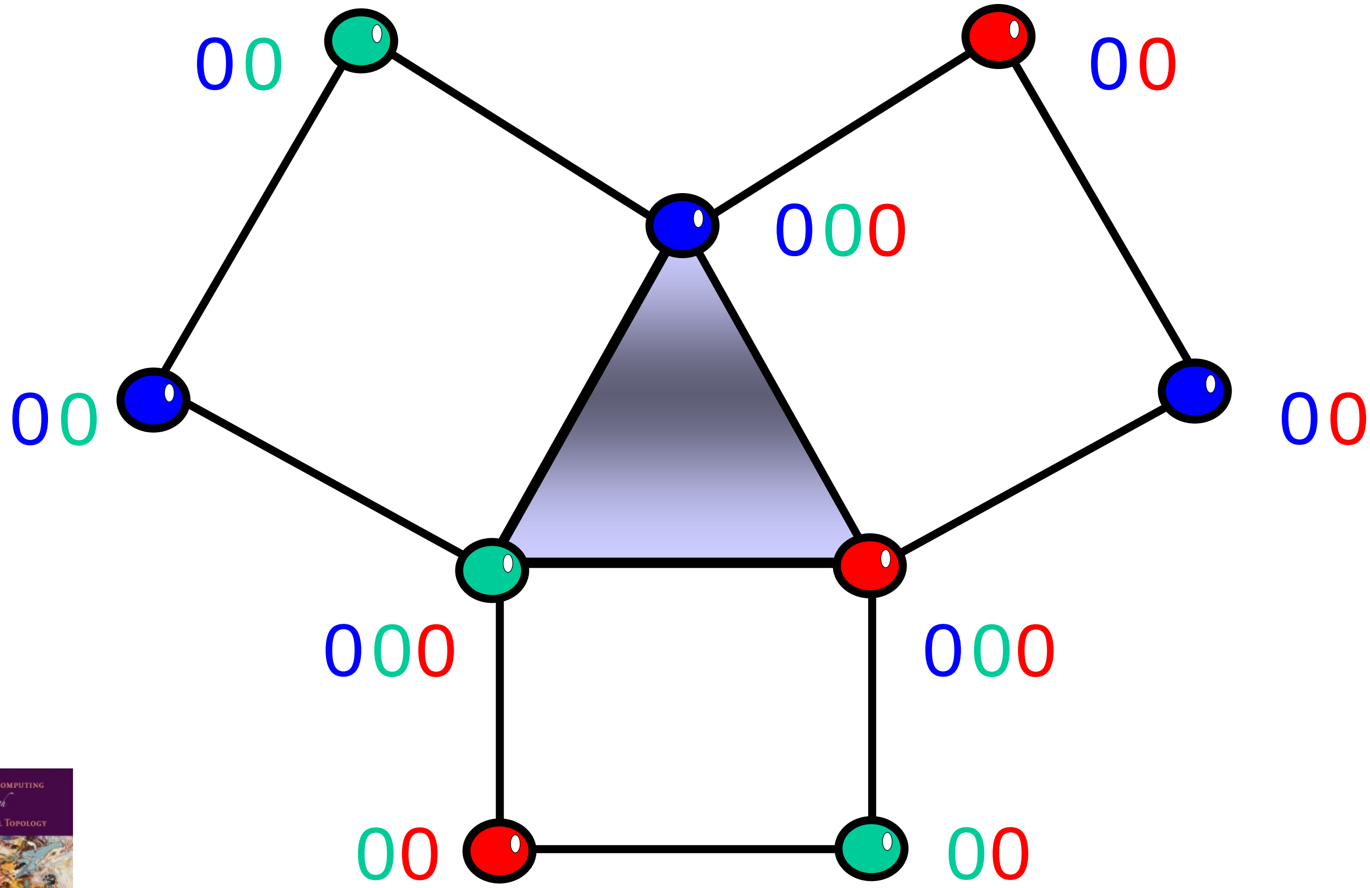
No messages sent

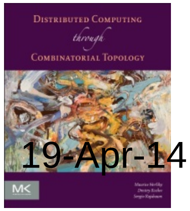
View is input value

Same as input complex

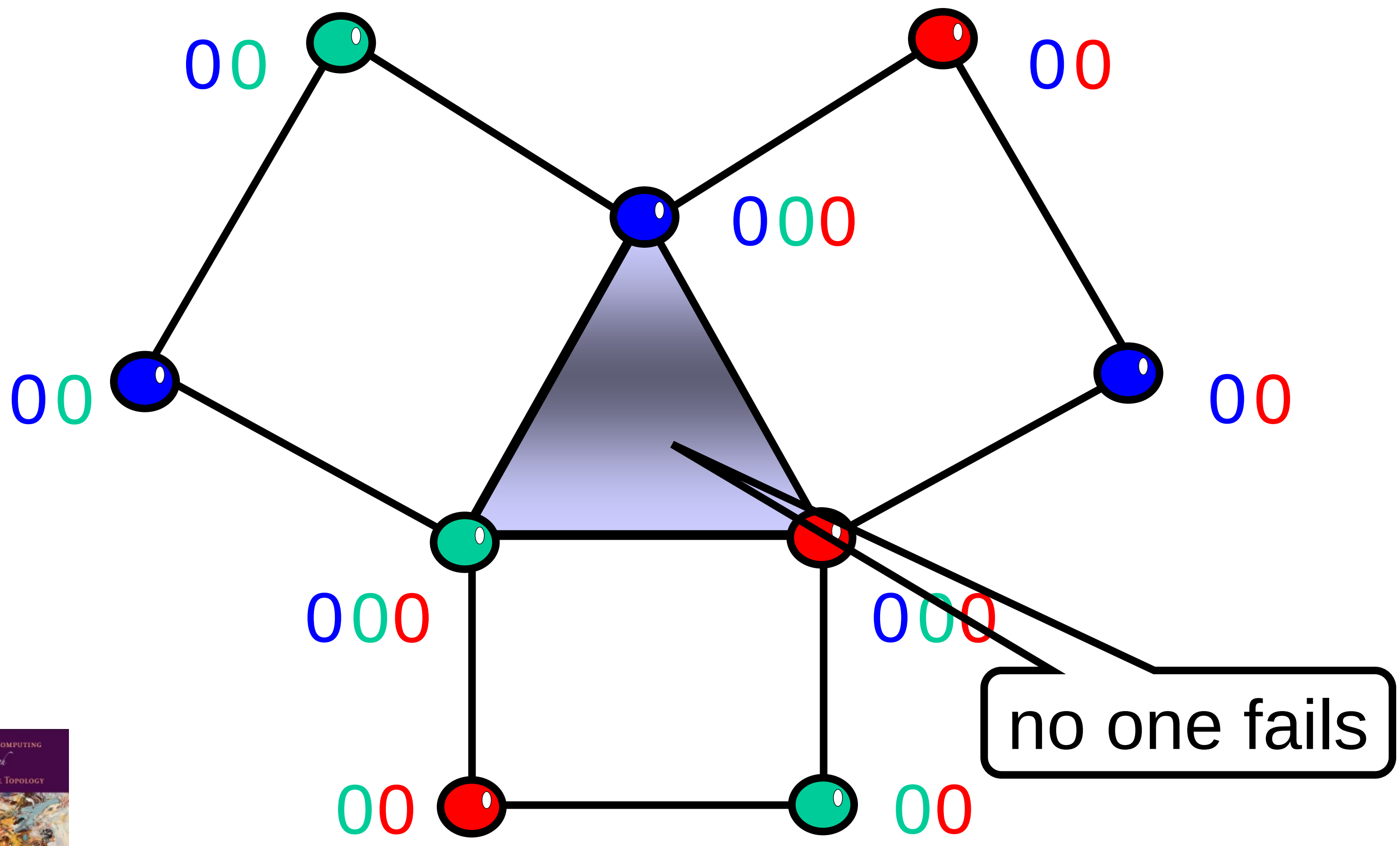


Single Input: Round One

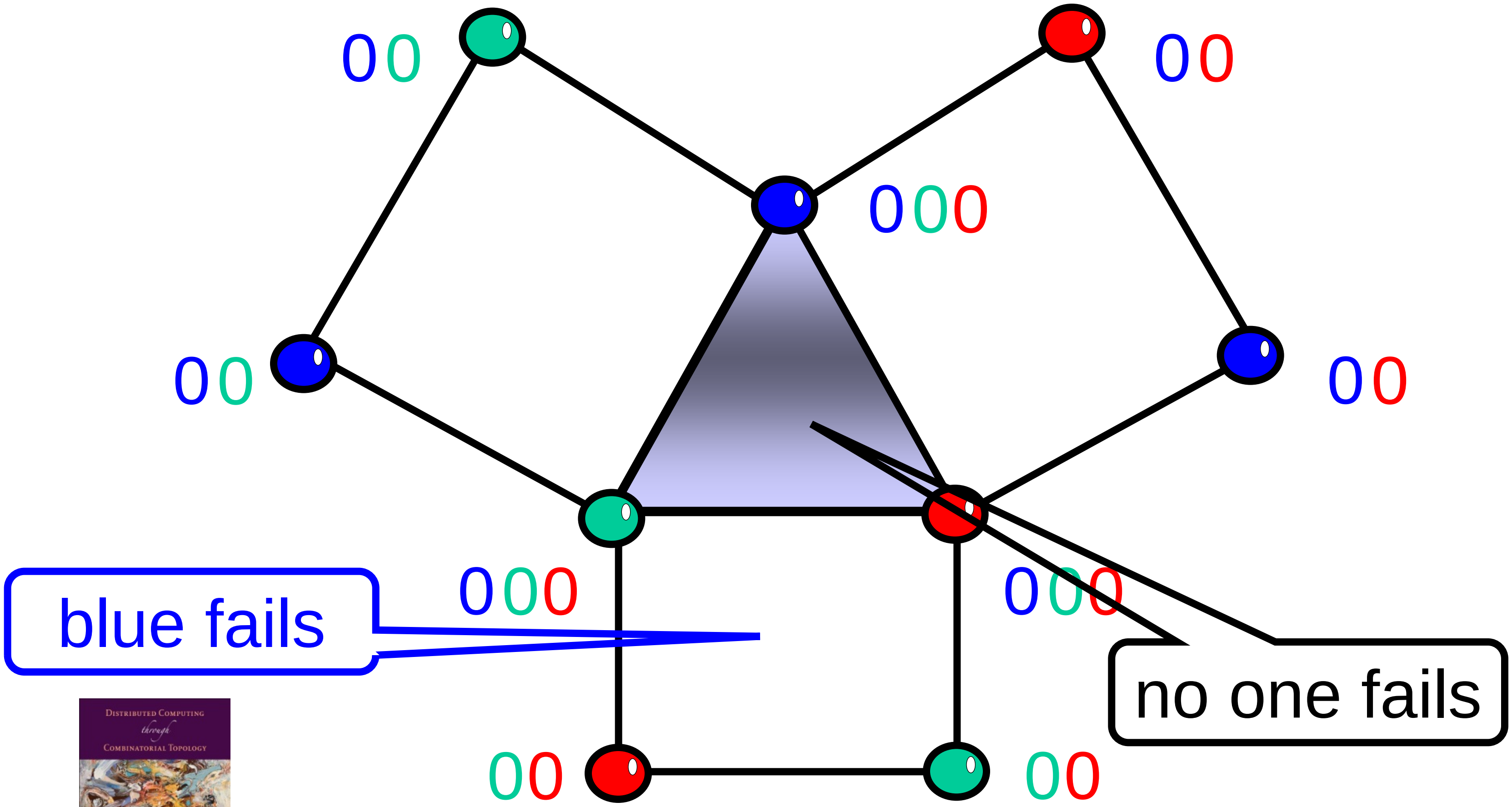




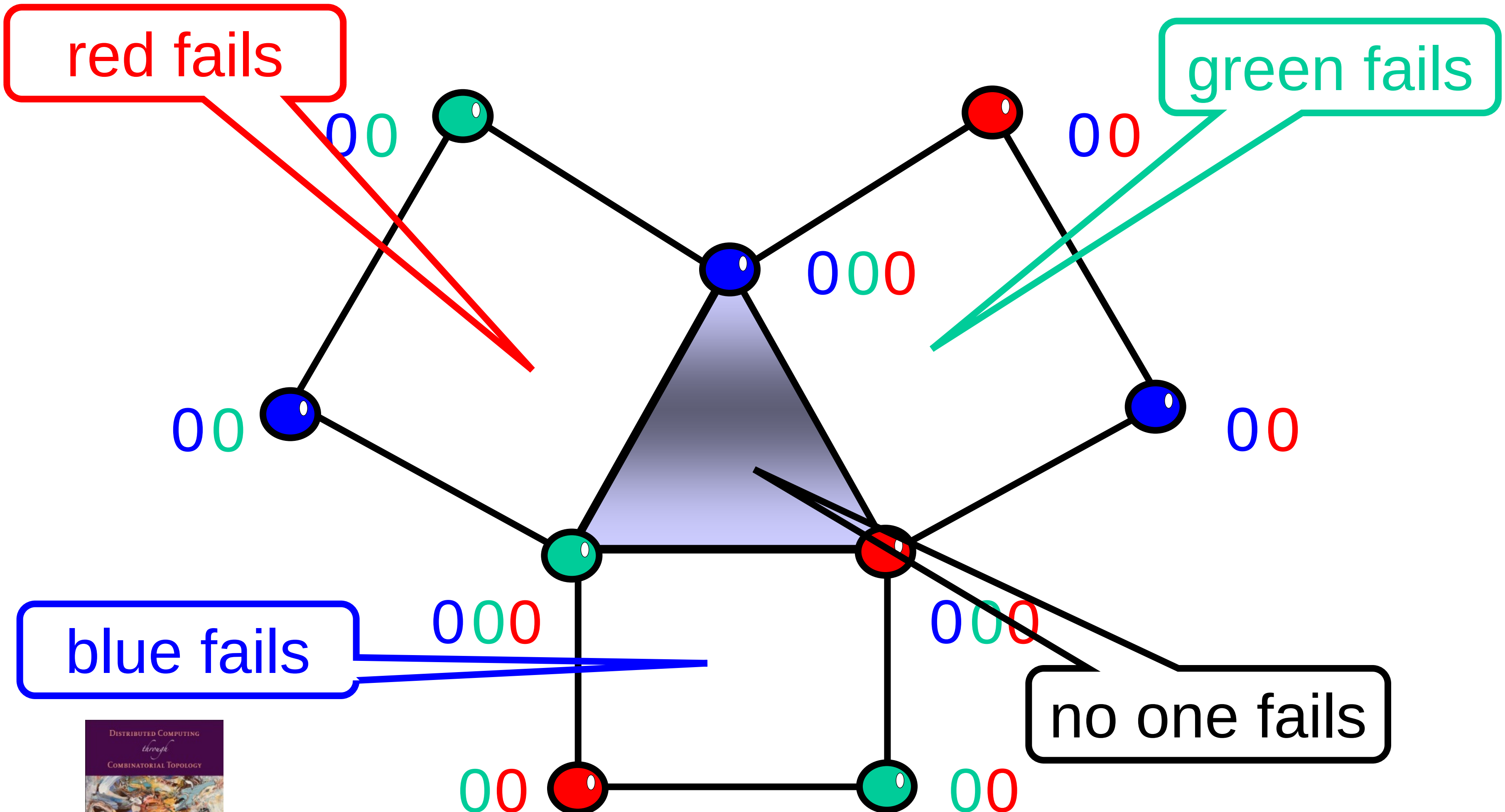
Single Input: Round One



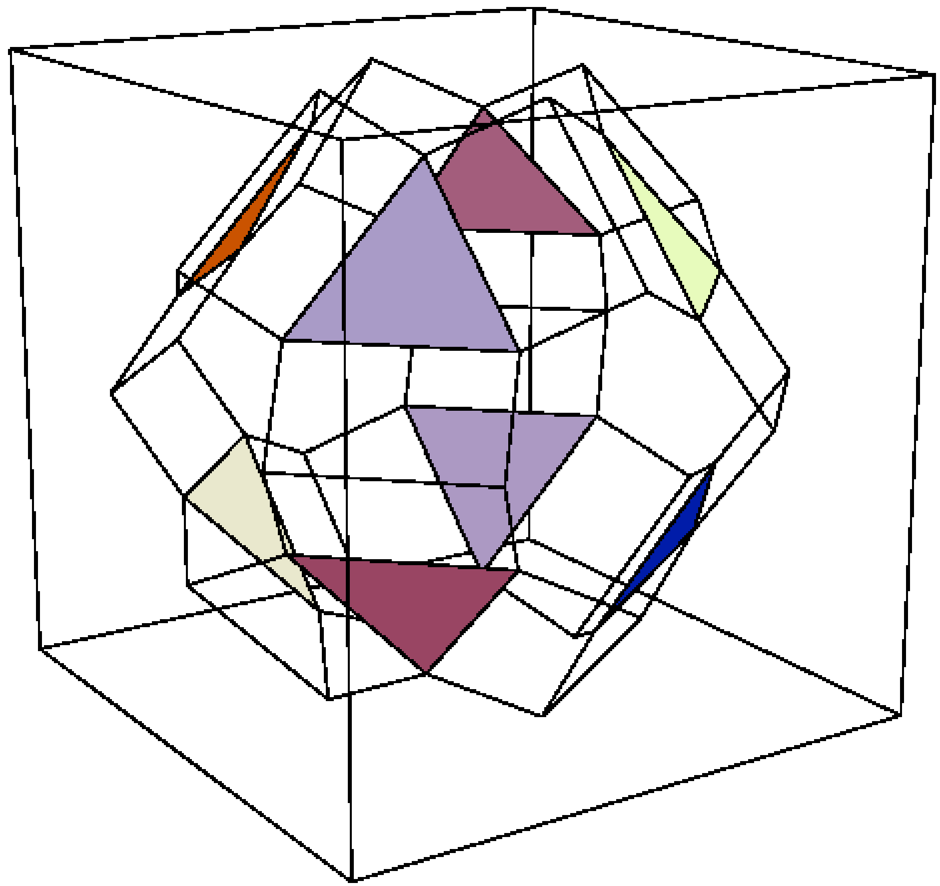
Single Input: Round One

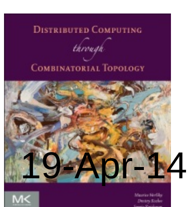


Single Input: Round One

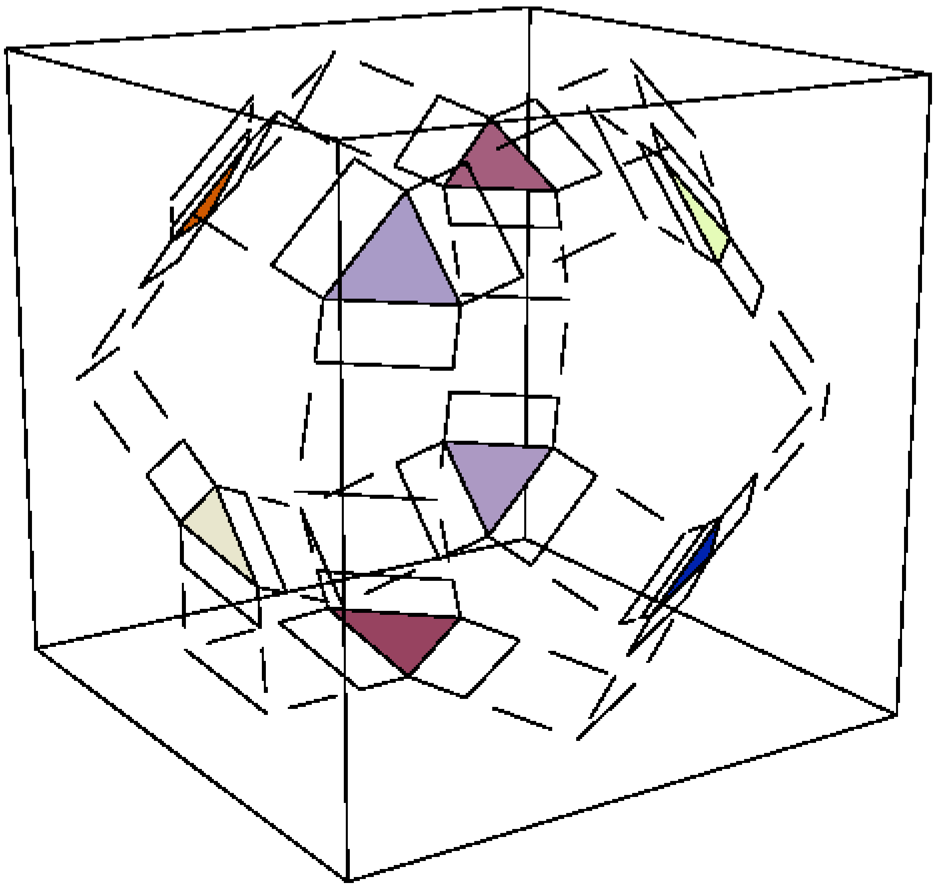


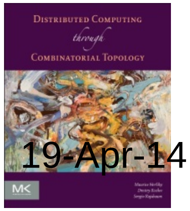
Protocol Complex: Round One



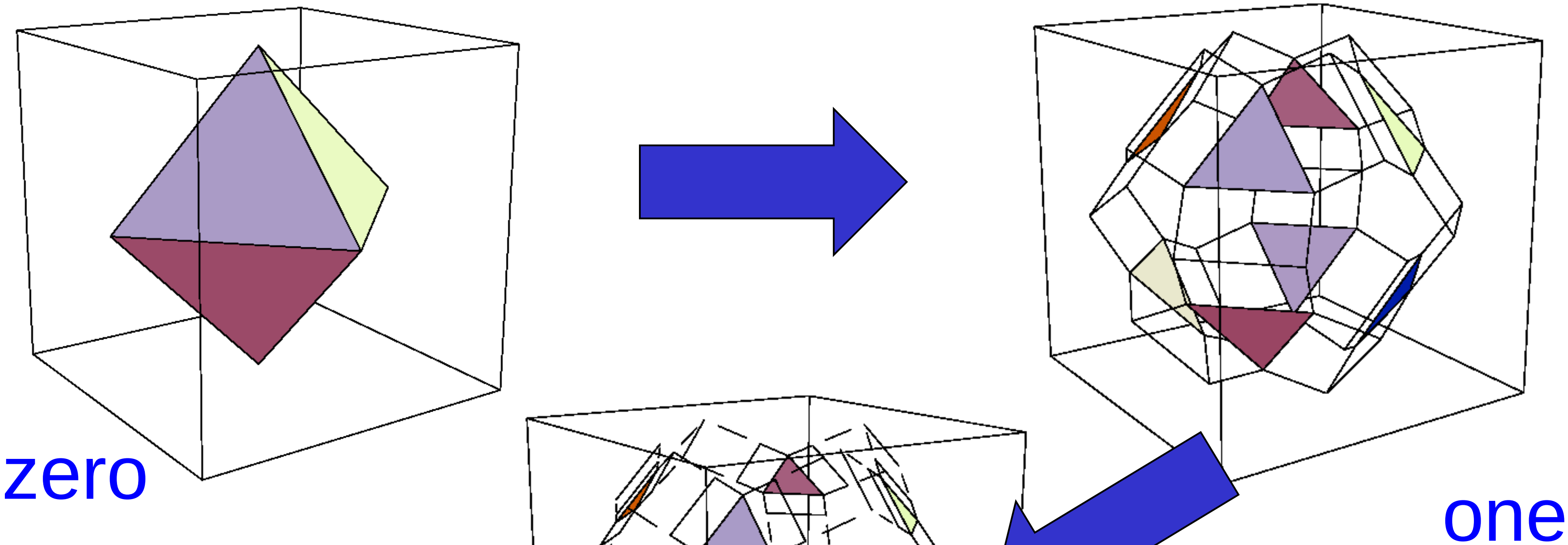


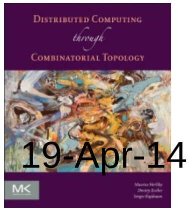
Protocol Complex: Round Two



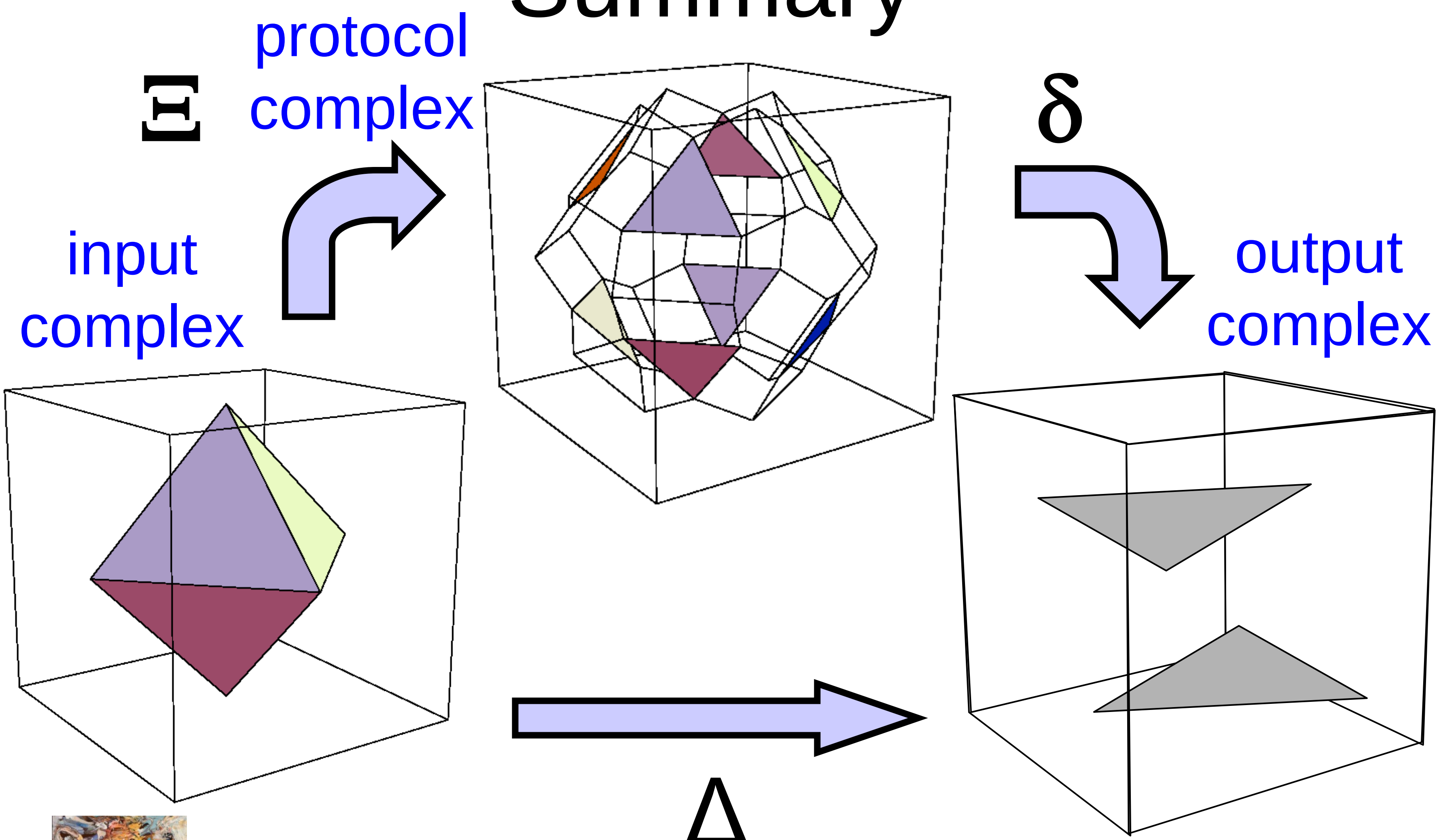


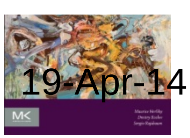
Protocol Complex Evolution



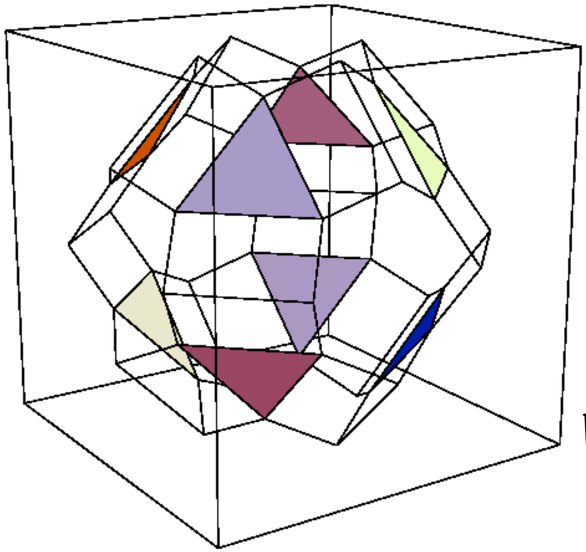


Summary

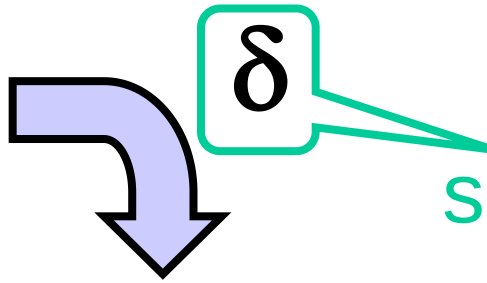




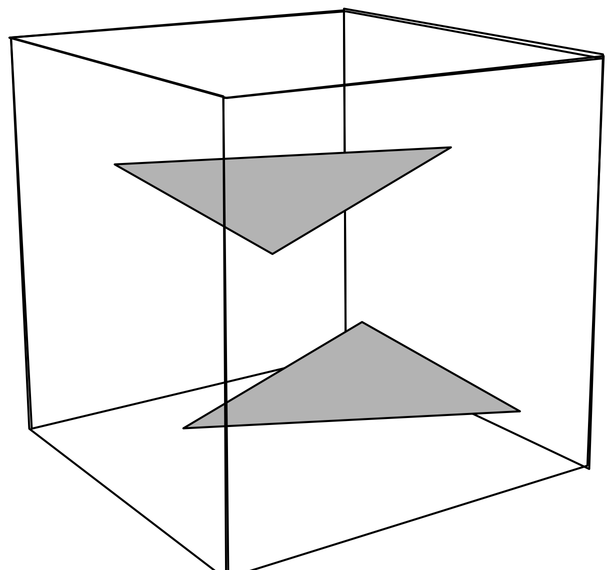
Decision Map



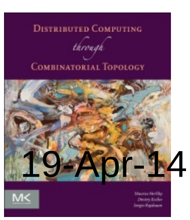
Protocol complex



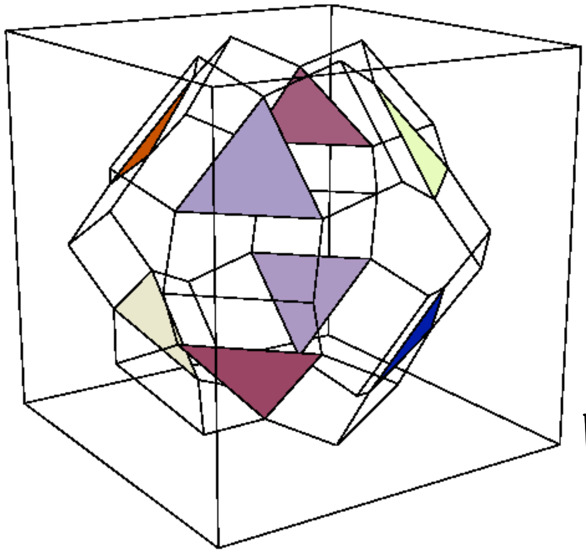
Simplicial map, sending simplexes to simplexes



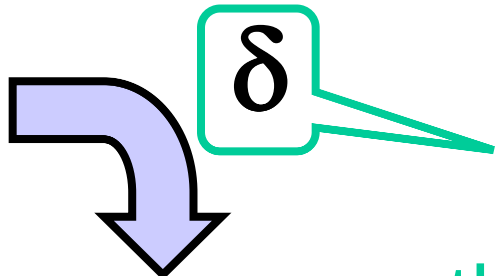
Output complex



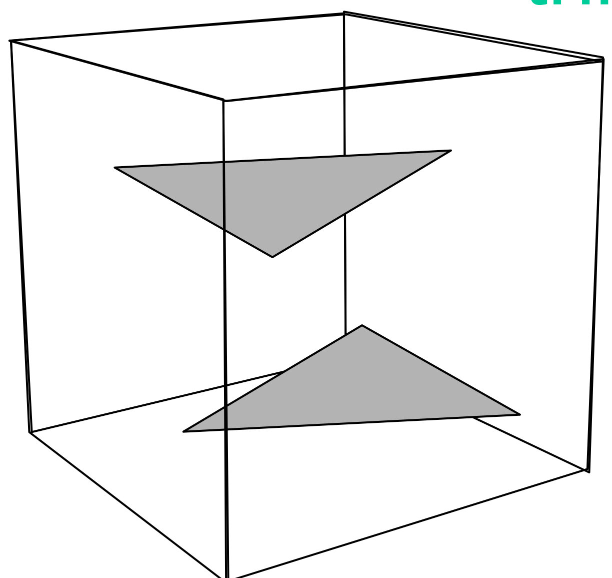
Lower Bound Strategy



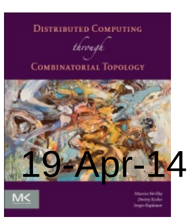
Protocol complex



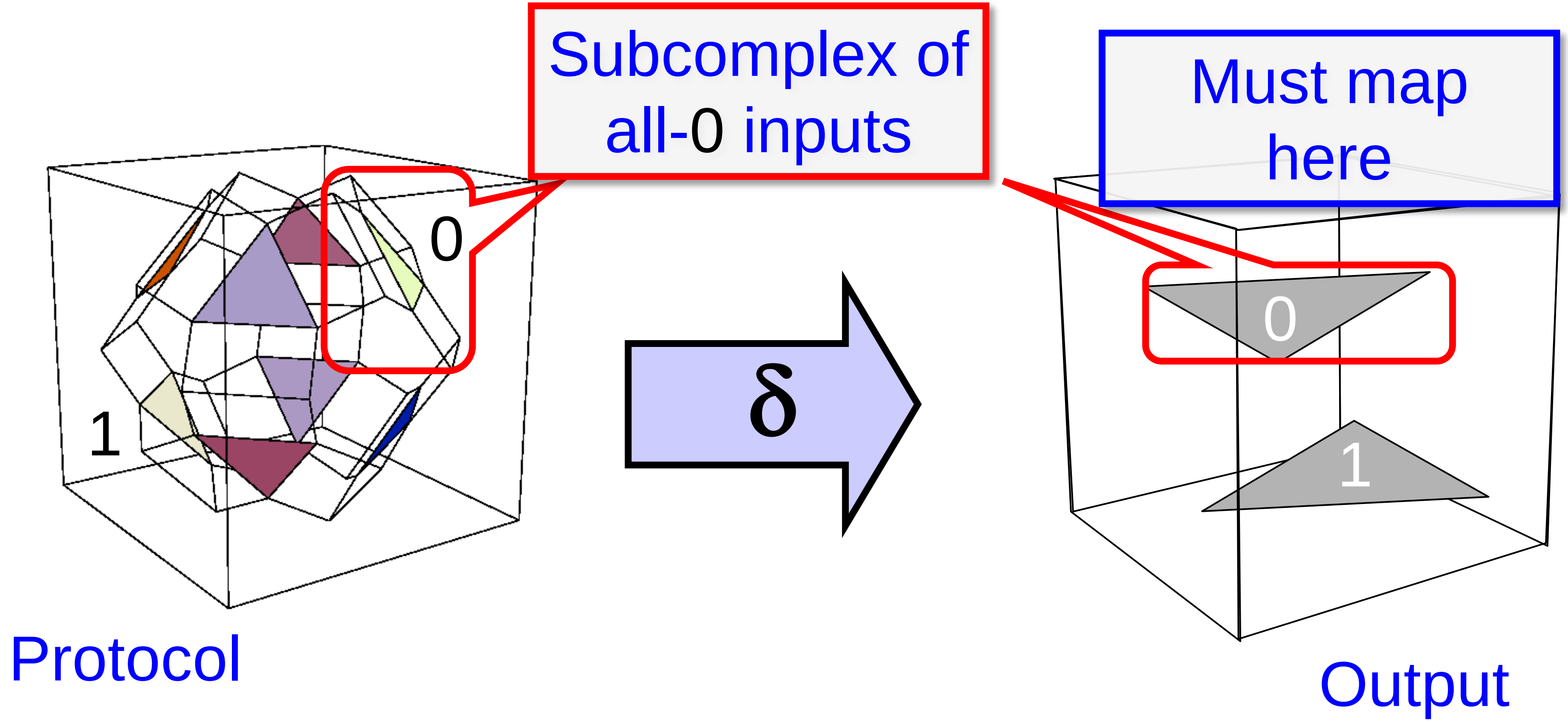
Find topological “obstruction” to this simplicial map

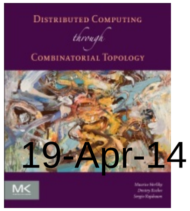


Output complex

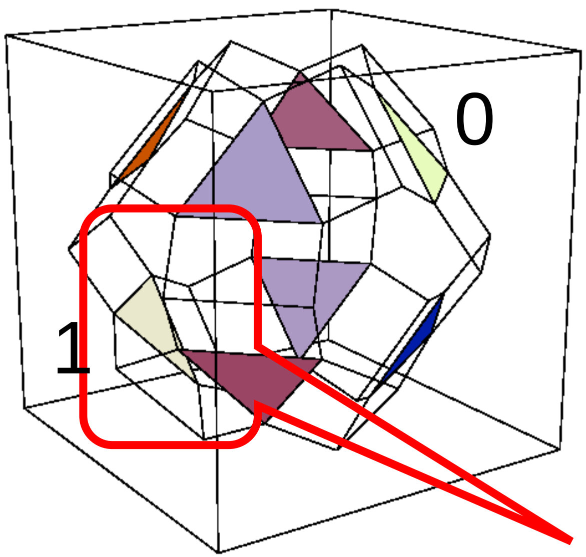


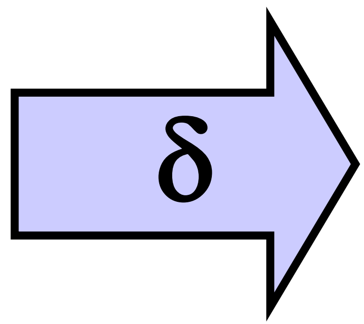
Consensus Example

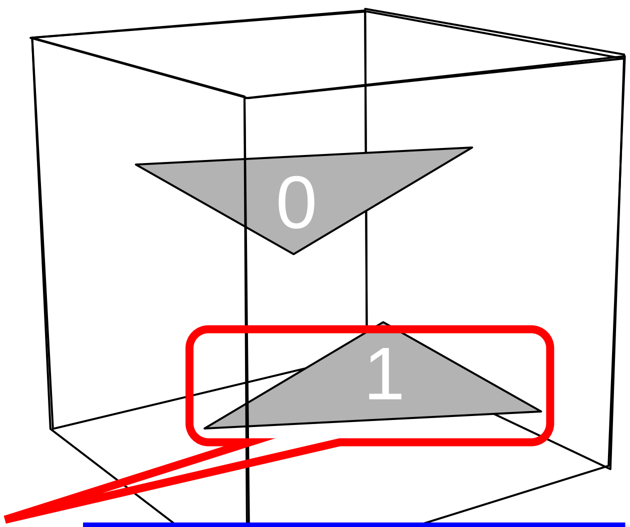




Consensus Example





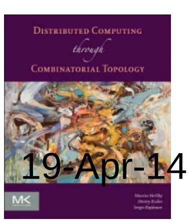


Protocol

Subcomplex of all-1 inputs

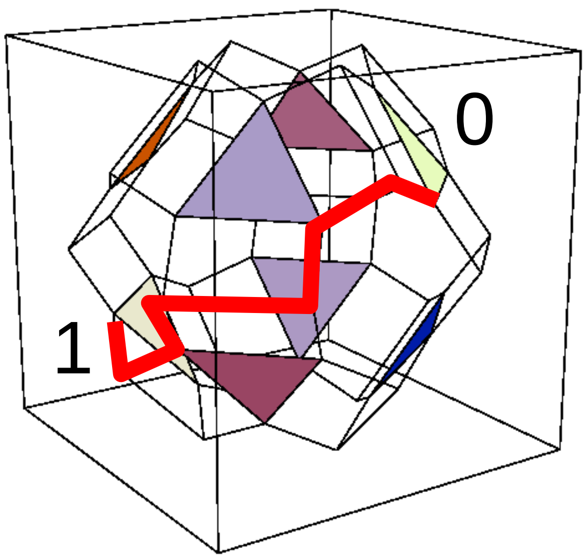
Must map here

Output



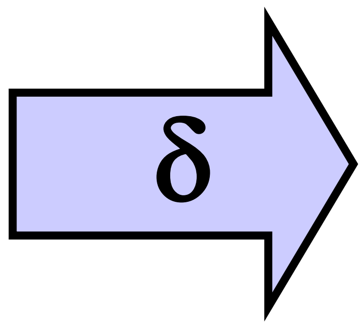
Consensus Example

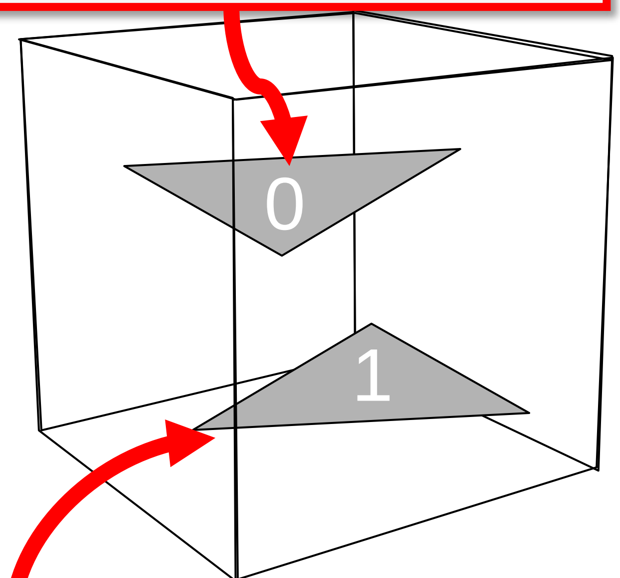
Image under δ must start here ..



Protocol

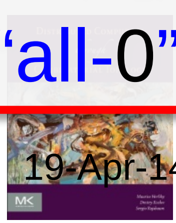
Path from "all-0" to "all-1"



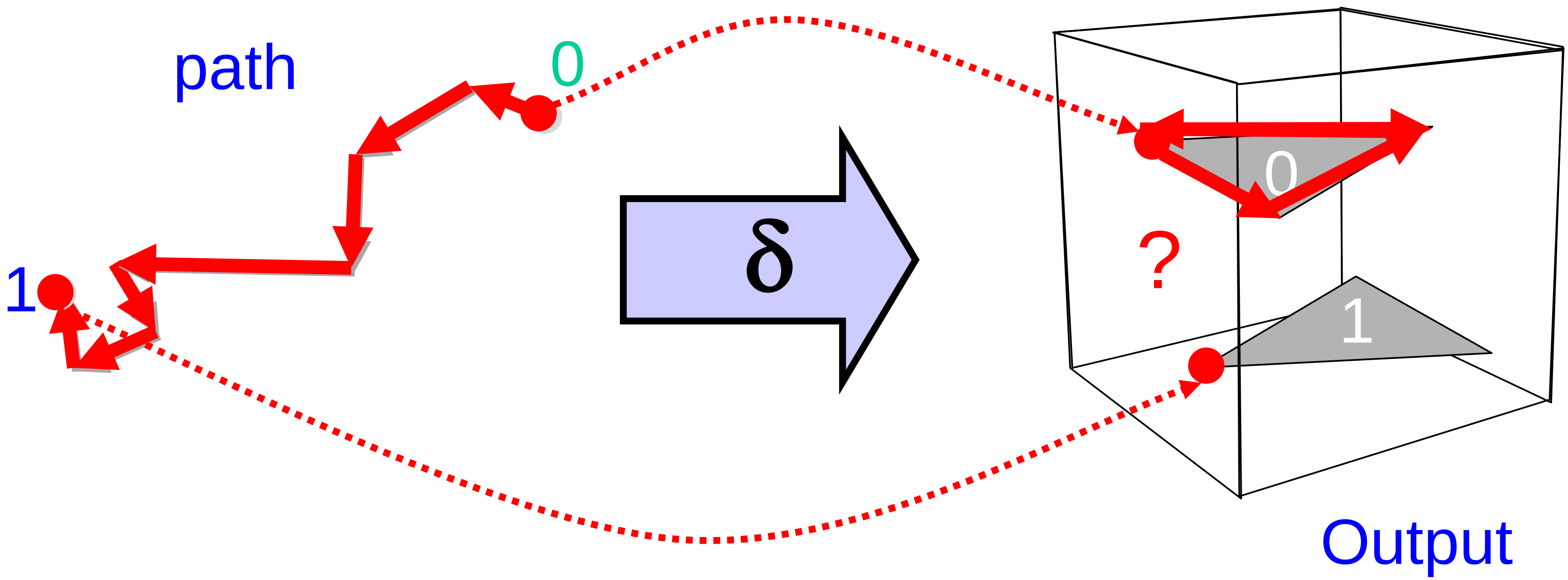


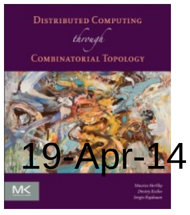
and end here

Output



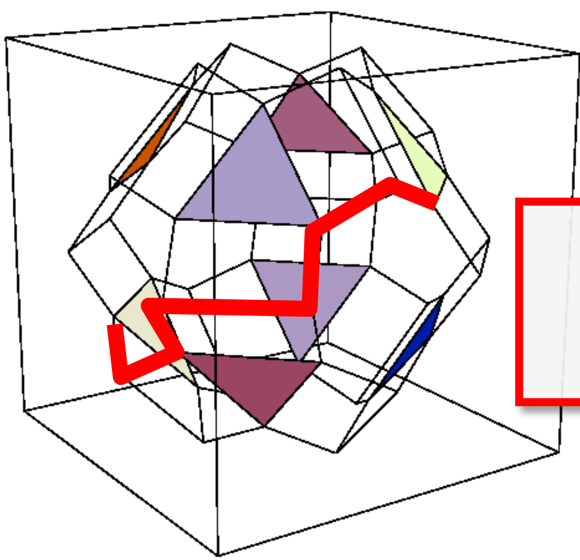
Consensus Example



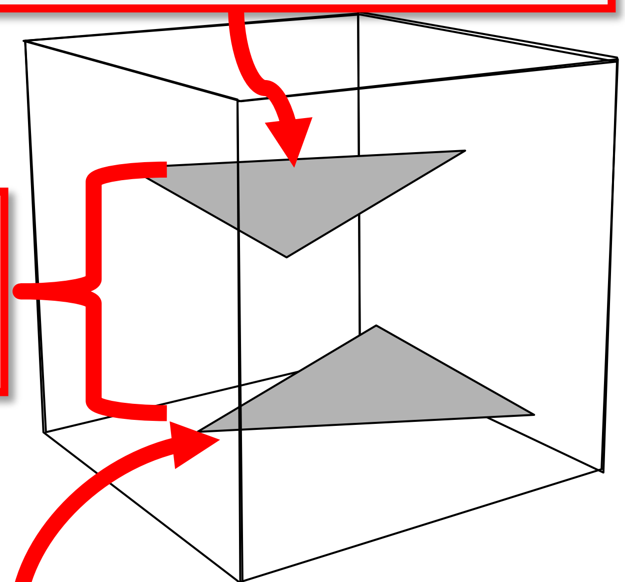


Consensus Example

Image under δ must start here ..

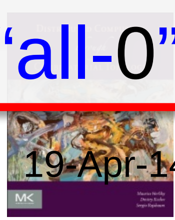


But this "hole" is an obstruction



Protocol

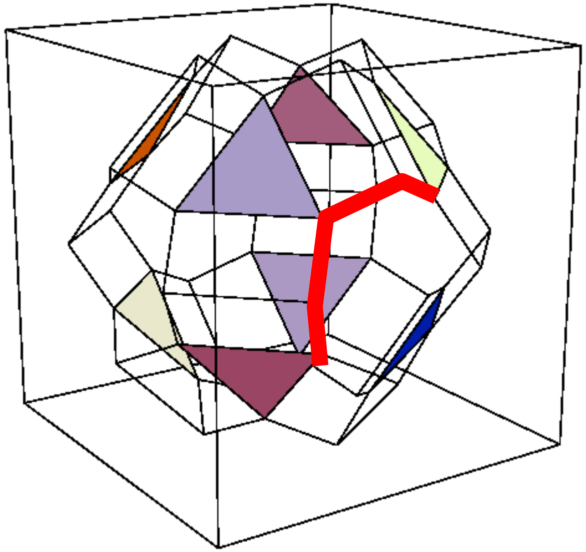
Path from "all-0" to "all-1"



and end here

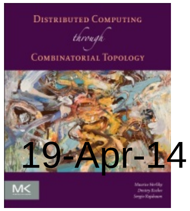
Output

Conjecture



A protocol cannot solve consensus if its complex is ***path-connected***

Model-independent!



If Adversary keeps Protocol Complex path-connected …

Forever …

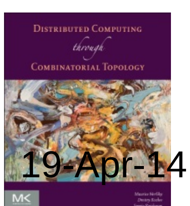
Consensus is ***impossible***

For *r* rounds …

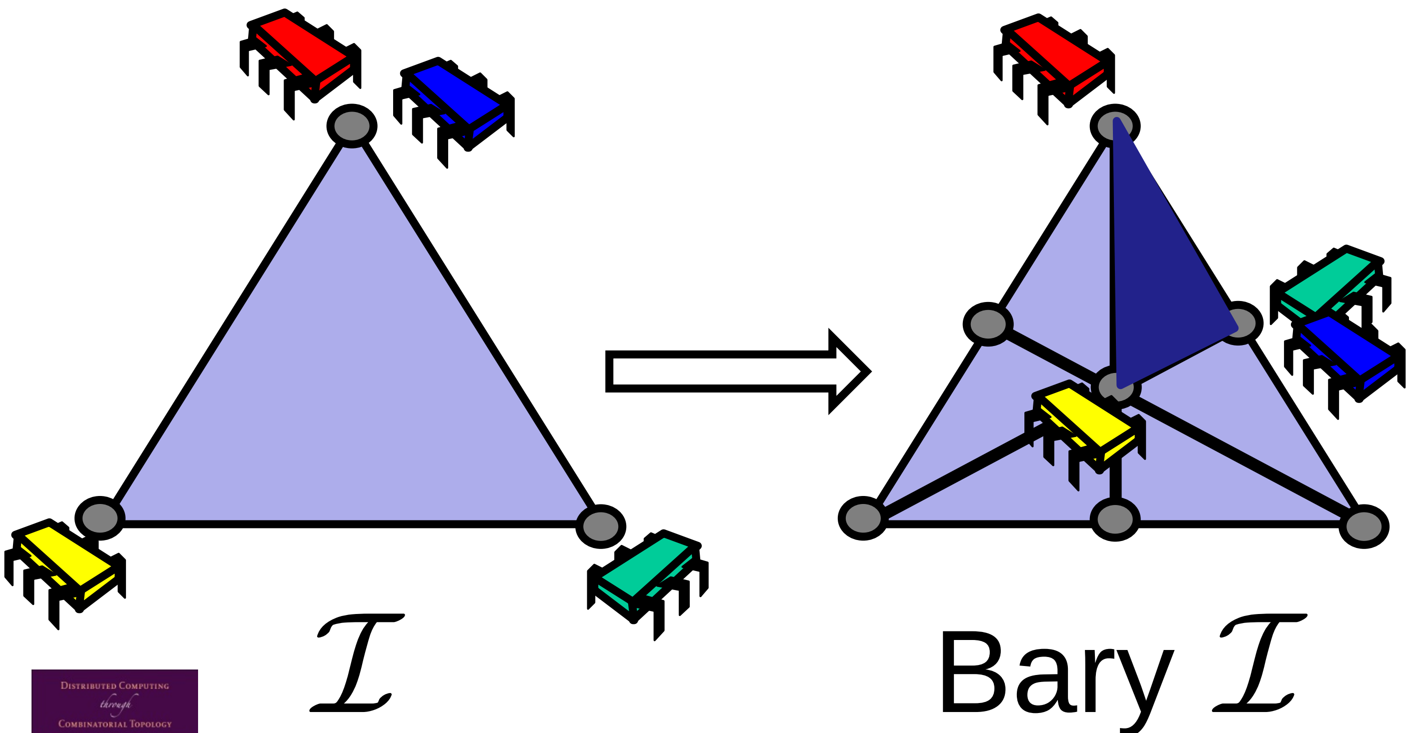
A ***round-complexity*** lower bound

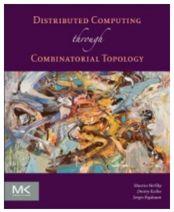
For time *t* …

A ***time-complexity*** lower bound



Barycentric Agreement





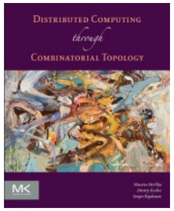
Barycentric Agreement

$$(\mathcal{I}, \text{bary } \mathcal{I}, \text{bary}(\cdot))$$

arbitrary input complex

subdivided output complex

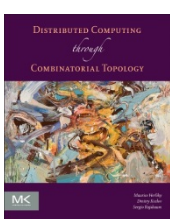
subdivision as carrier map



If There are n Processes

$$(\mathcal{I}, \text{bary}\ \text{skel}^n\ \mathcal{I}, \text{bary} \circ \text{skel}^n)$$

Inputs only from *n*-skeleton of input complex



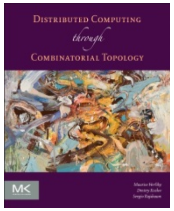
Theorem

A one-layer immediate snapshot protocol solves the *n*-process barycentric agreement task $(\mathcal{I}, \text{bary skel}^n\, \mathcal{I}, \text{bary}\circ\text{skel}^n)$

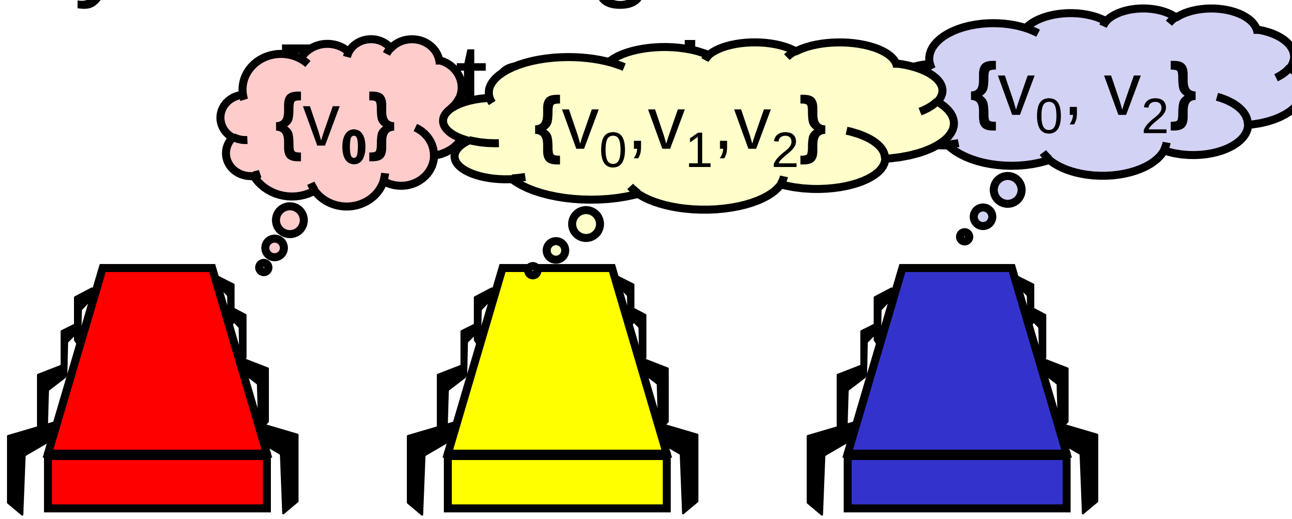
Proof

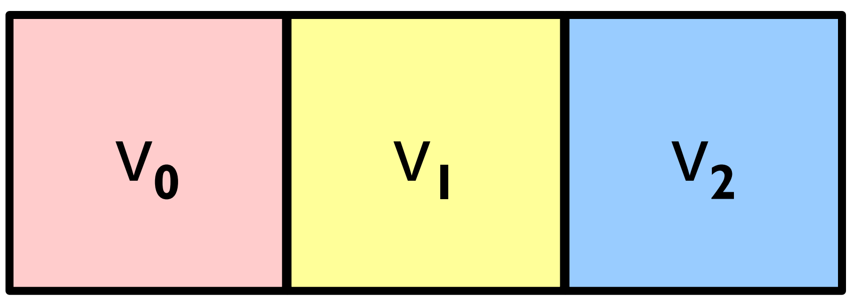
All input simplices belong to $\text{skel}^n\, \mathcal{I}$

Immediate snapshot results are ordered

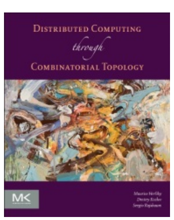


Barycentric Agreement

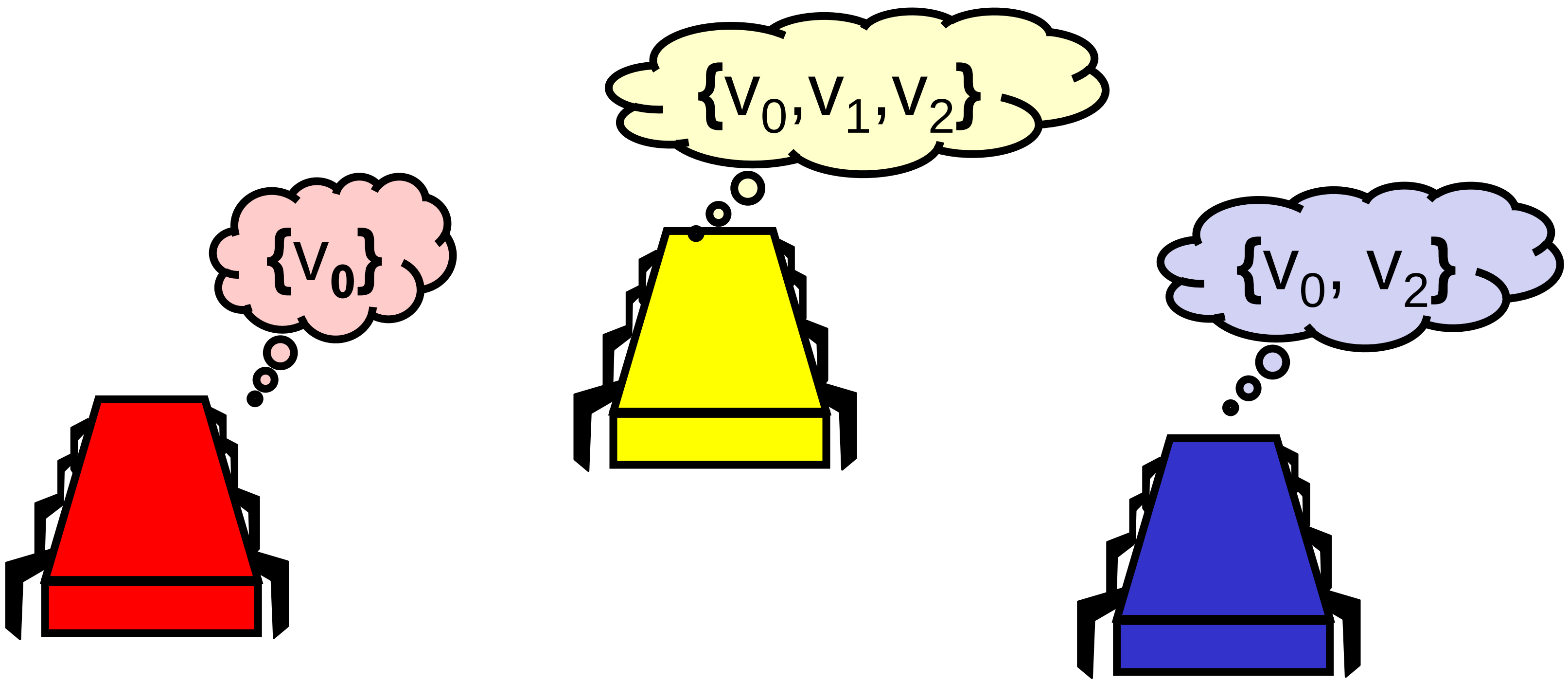


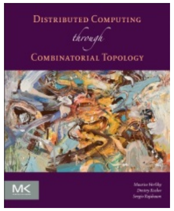


Snapshots are ordered



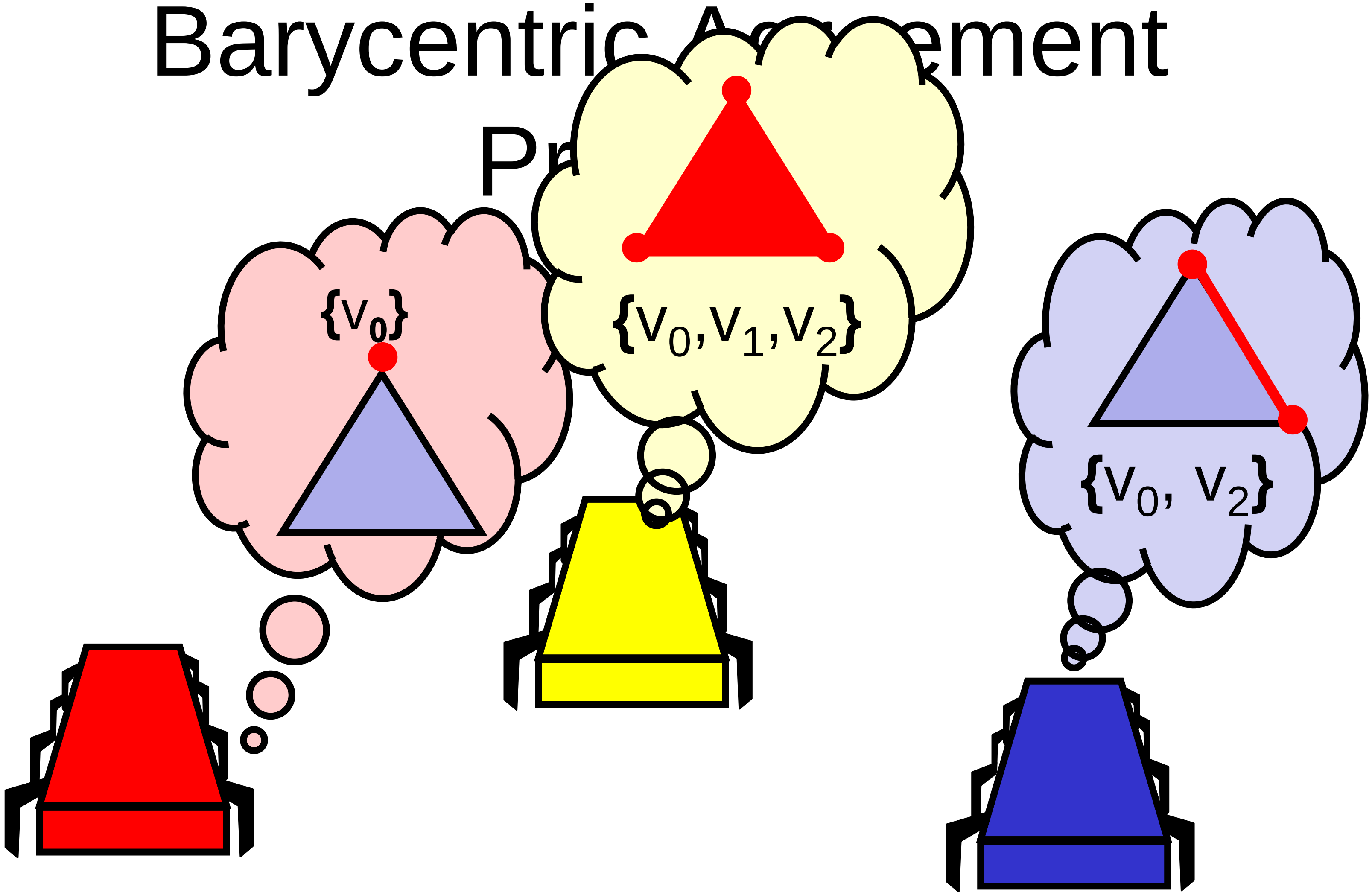
Barycentric Agreement Protocol

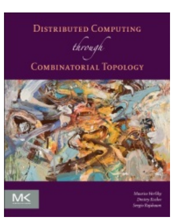




Each view is a face of σ

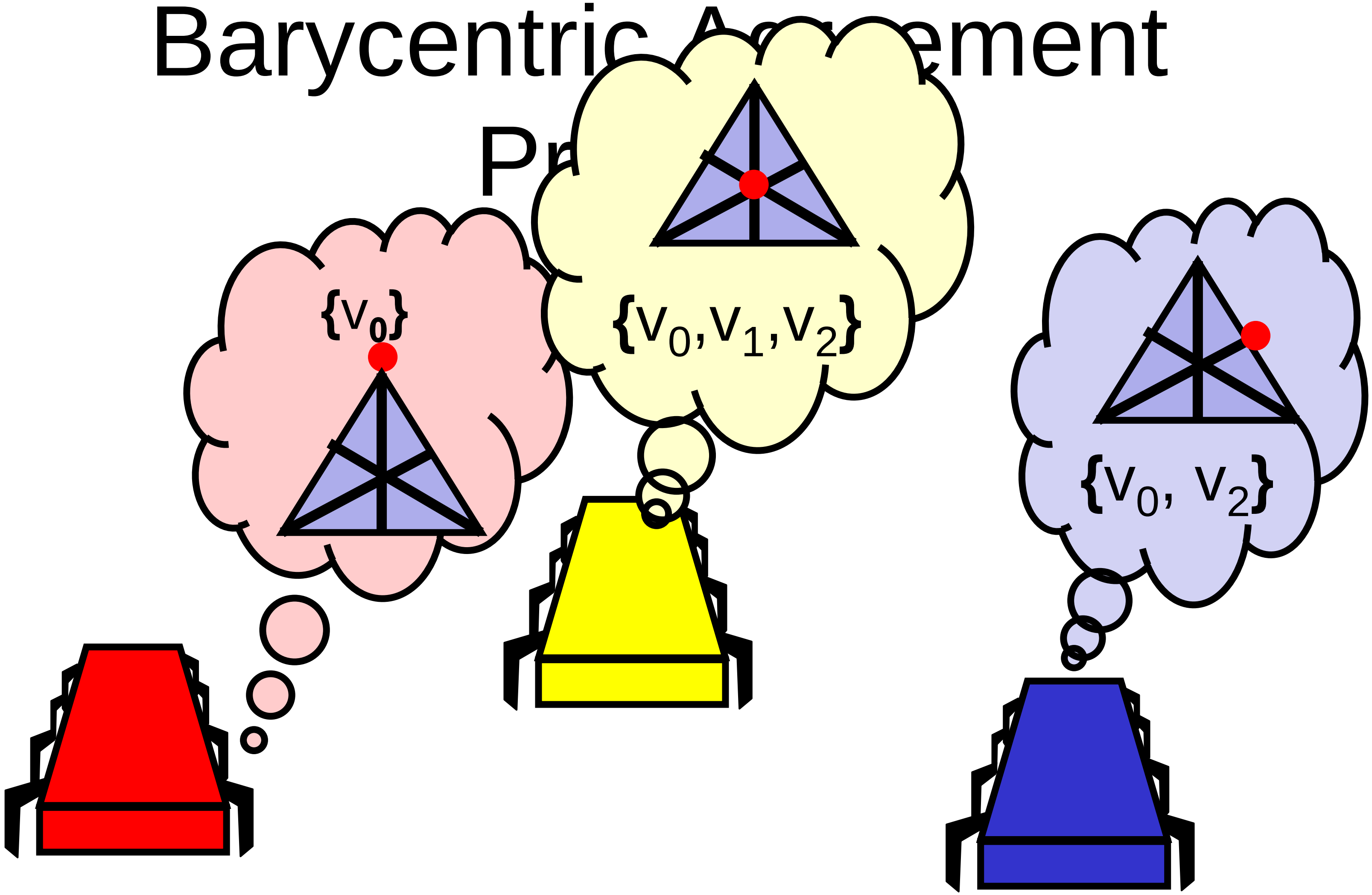
Barycentric Agreement Protocol

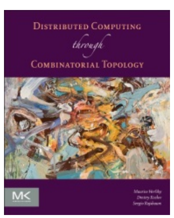




Each view is a face of σ

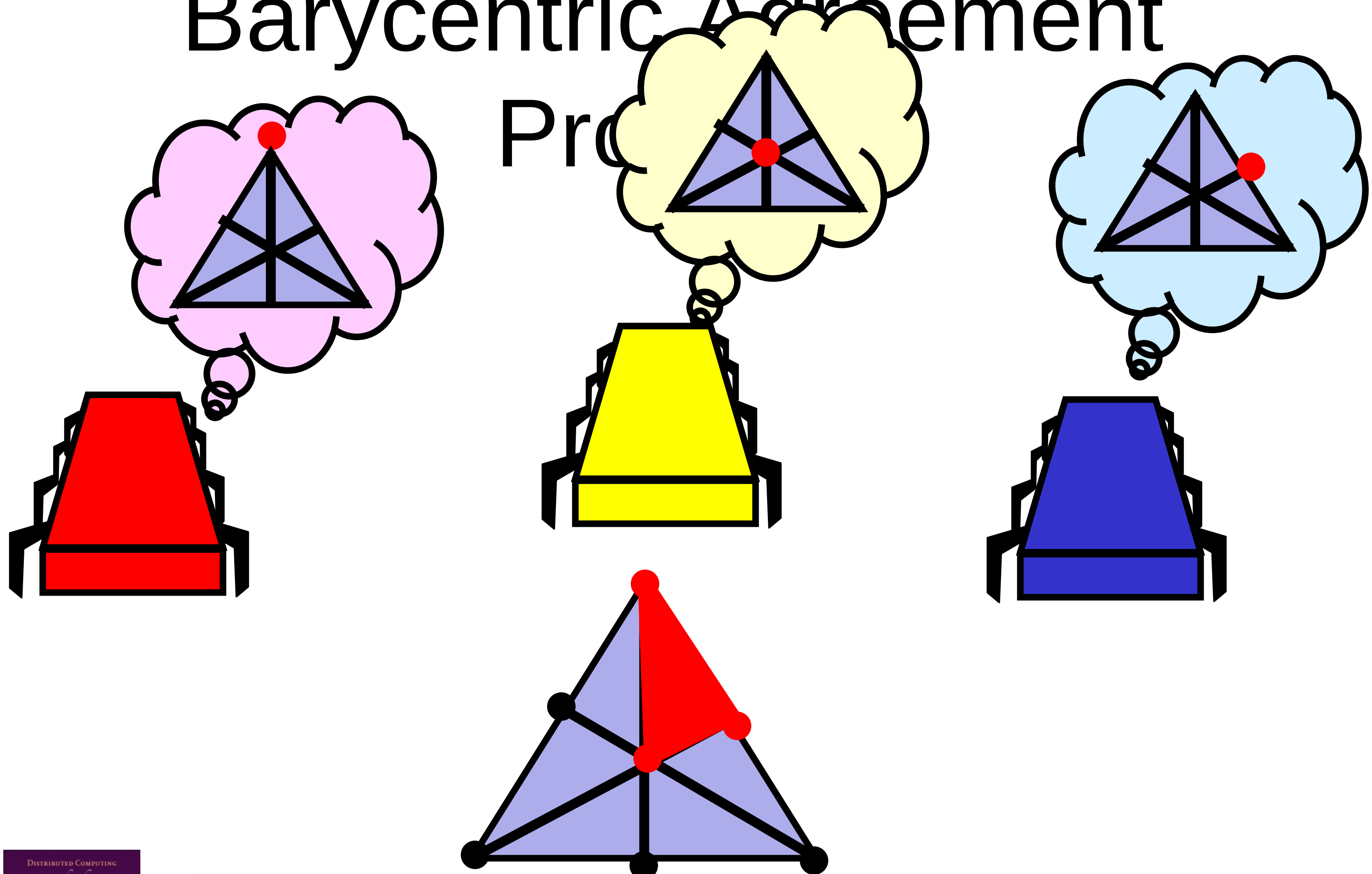
Barycentric Agreement Protocol

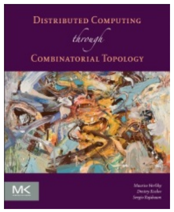




Each face of σ is a vertex of Bary σ

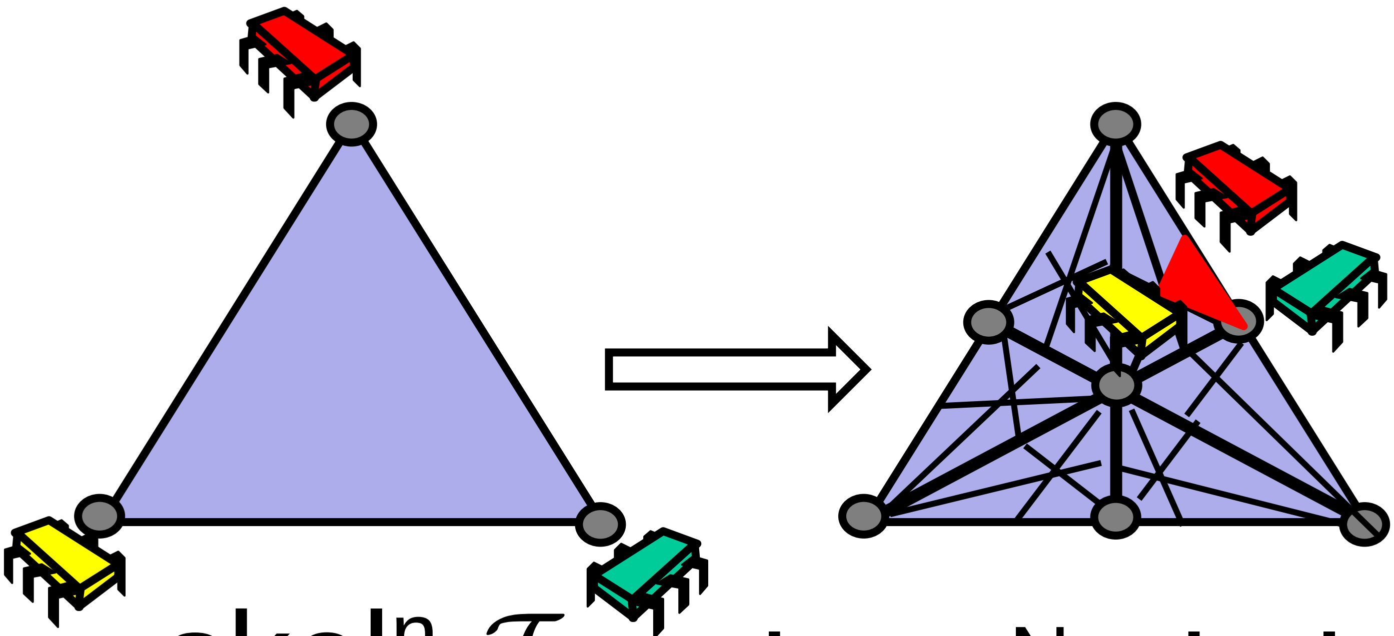
Barycentric Agreement
Pr



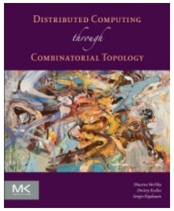


Ordered faces ⇒ simplex of Bary σ

Iterated Barycentric Agreement

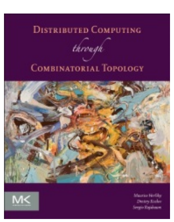


$\text{skel}^n\ \mathcal{I}$ $\quad$ $\text{bary}^N\ \text{skel}^n\ \mathcal{I}$

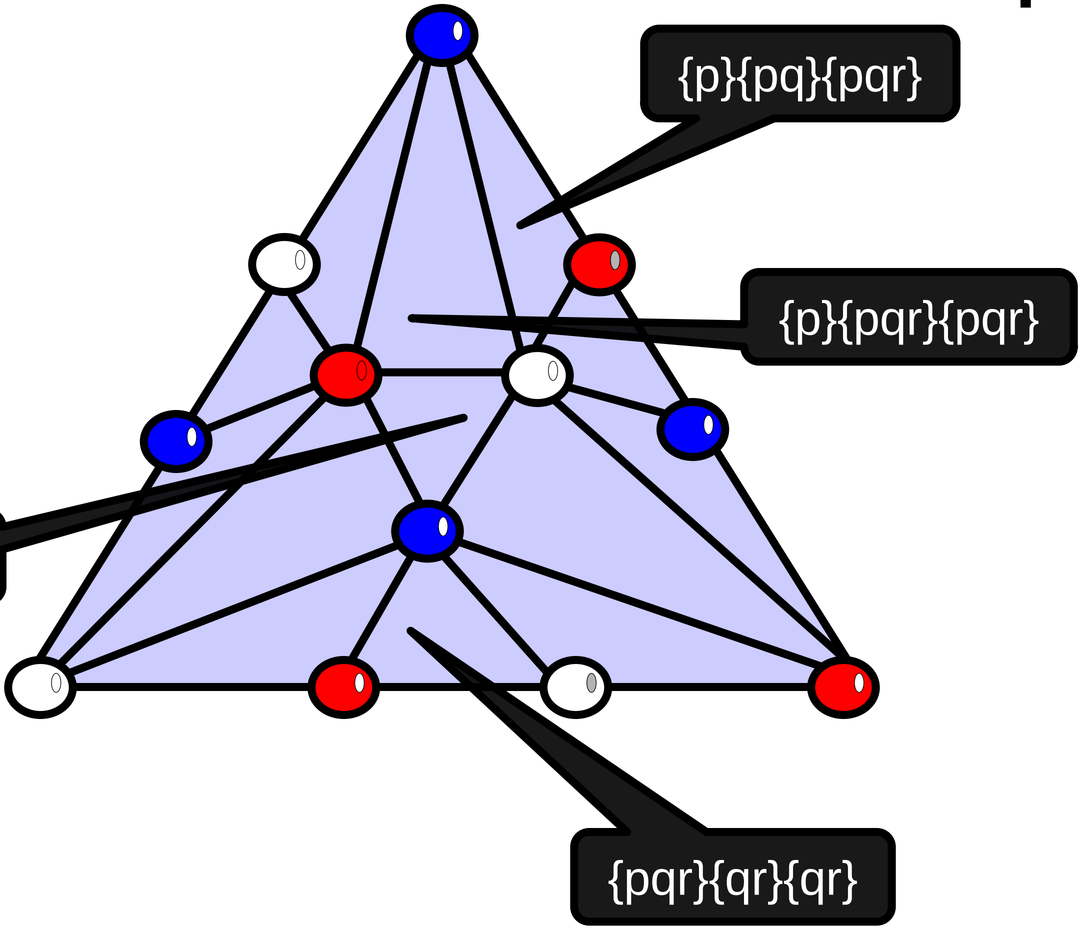


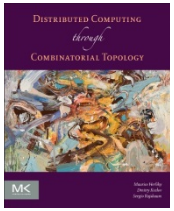
Iterated Barycentric Agreement

$$\boxed{(\mathcal{I}, \mathrm{bary}^N\, \mathrm{skel}^n\, \mathcal{I}, \mathrm{bary}^N \circ \mathrm{skel}^n)}$$

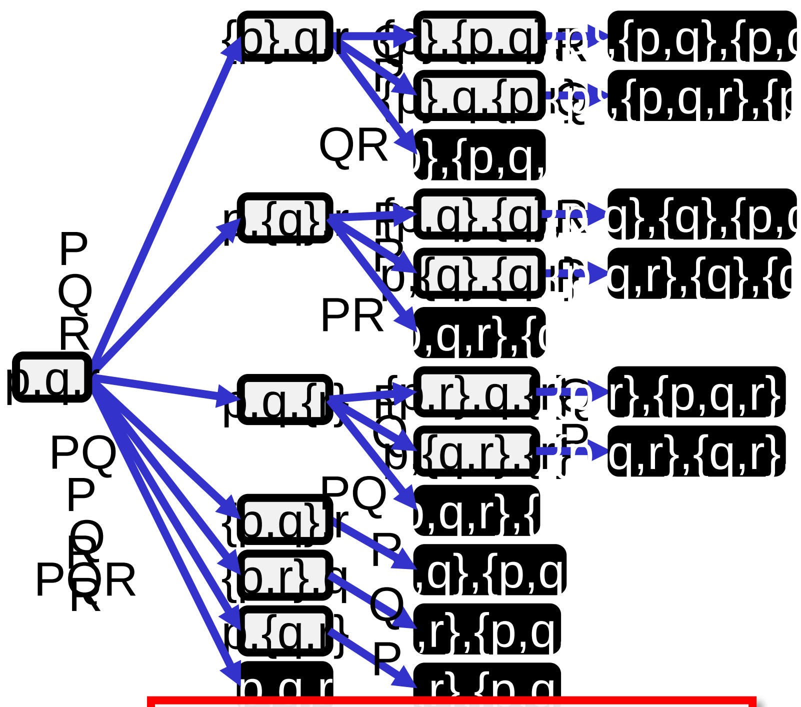


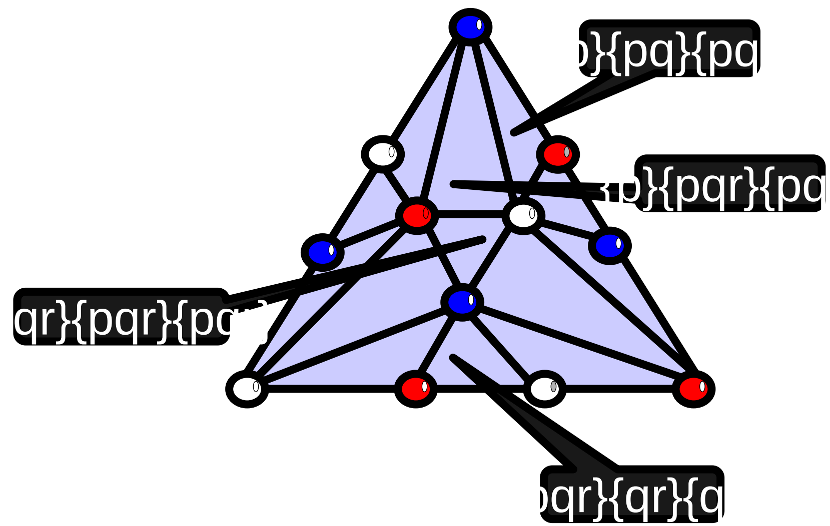
One-Layer Immediate Snapshot Protocol Complex





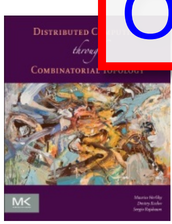
Compare Views





Operational view

Combinatorial view



Compositions

Given protocols

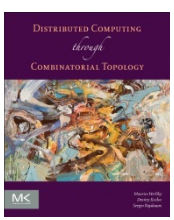
$(\mathcal{I},\mathcal{P},\Xi)$ $(\mathcal{I}',\mathcal{P}',\Xi')$

where $\mathcal{P} \subseteq \mathcal{I}'$

their *composition* is

$(\mathcal{I},\mathcal{P}'',\Xi'')$

where $\Xi'' = \Xi' \circ \Xi$ and $\mathcal{P}'' = \Xi''(\mathcal{I})$

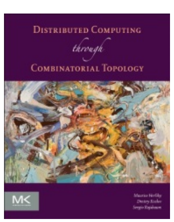


Road Map

Operational Model

Combinatorial Model

Main Theorem



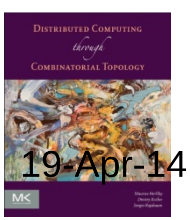
Fundamental Theorem

Theorem

$(\mathcal{I},\mathcal{O},\Delta)$ has a wait-free $(n+1)$-process layered protocol iff there is a continuous map

$f\colon |\mathrm{skel}^n\, \mathcal{I}| \to |\mathcal{O}|$...

carried by Δ



Proof Outline

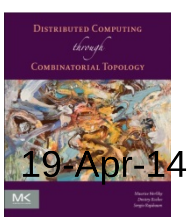
Lemma

If …

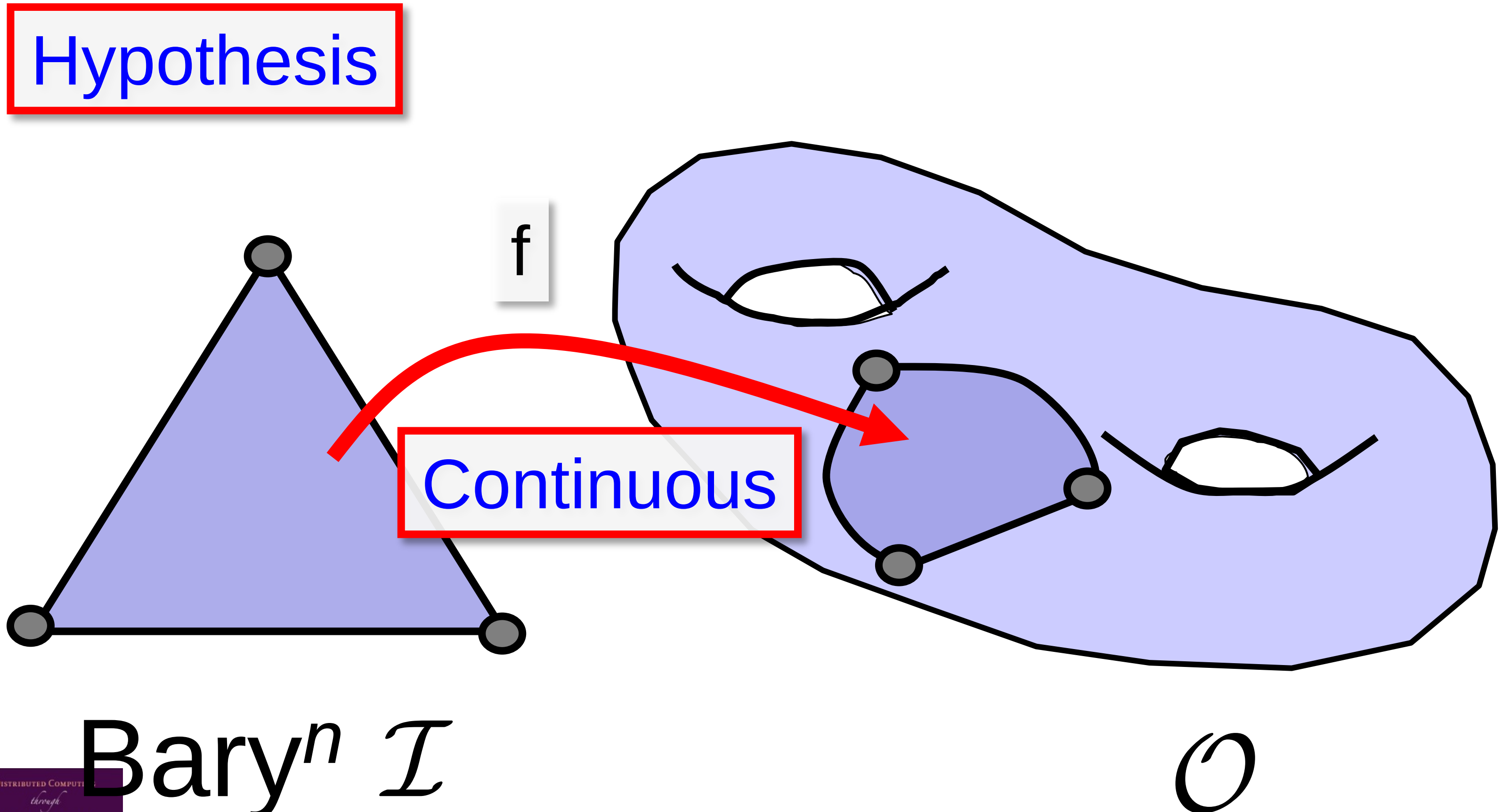
there is a WF layered protocol for $(\mathcal{I},\mathcal{O},\Delta)$ …

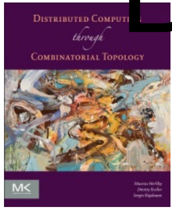
then …

there is a continuous $f: |\mathrm{skel}^n\, \mathcal{I}| \to |\mathcal{O}|$ carried by Δ.

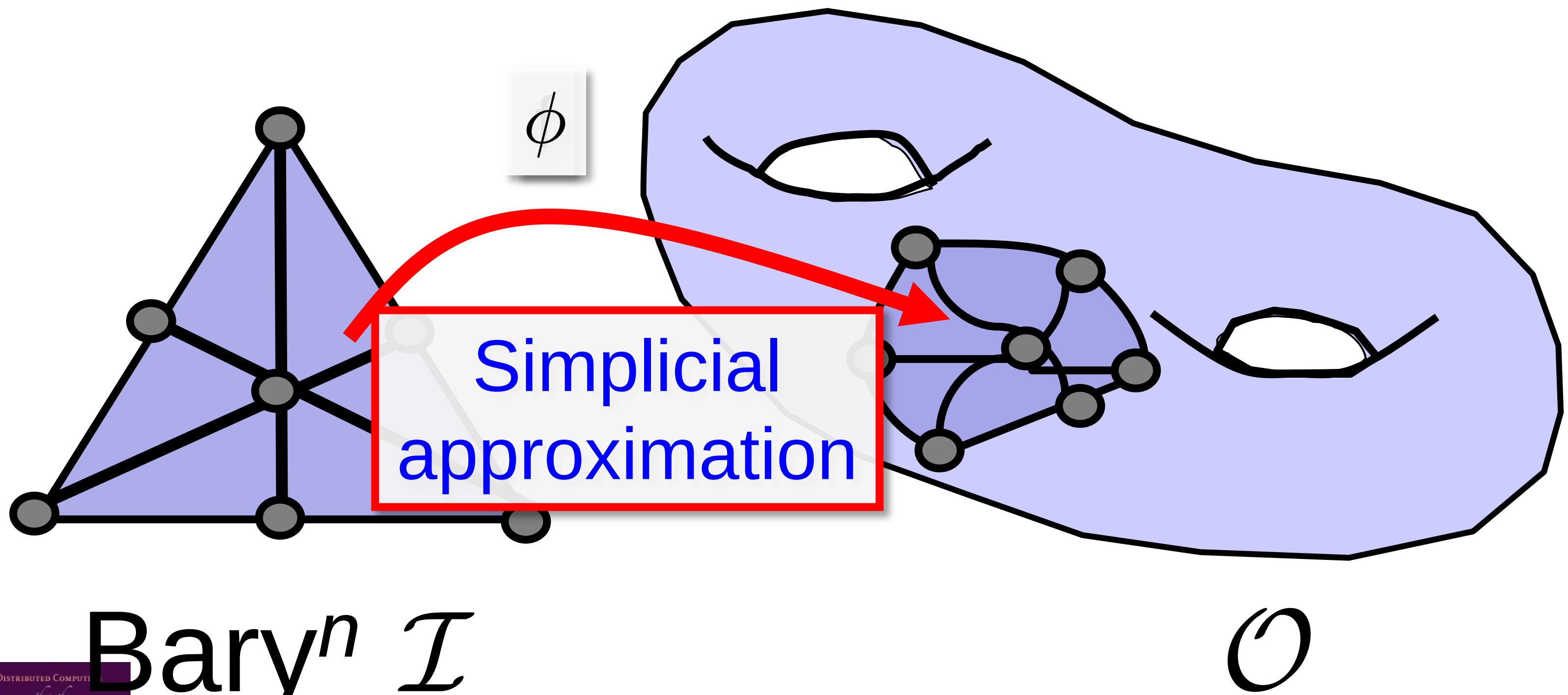


Map $\Rightarrow$ Protocol

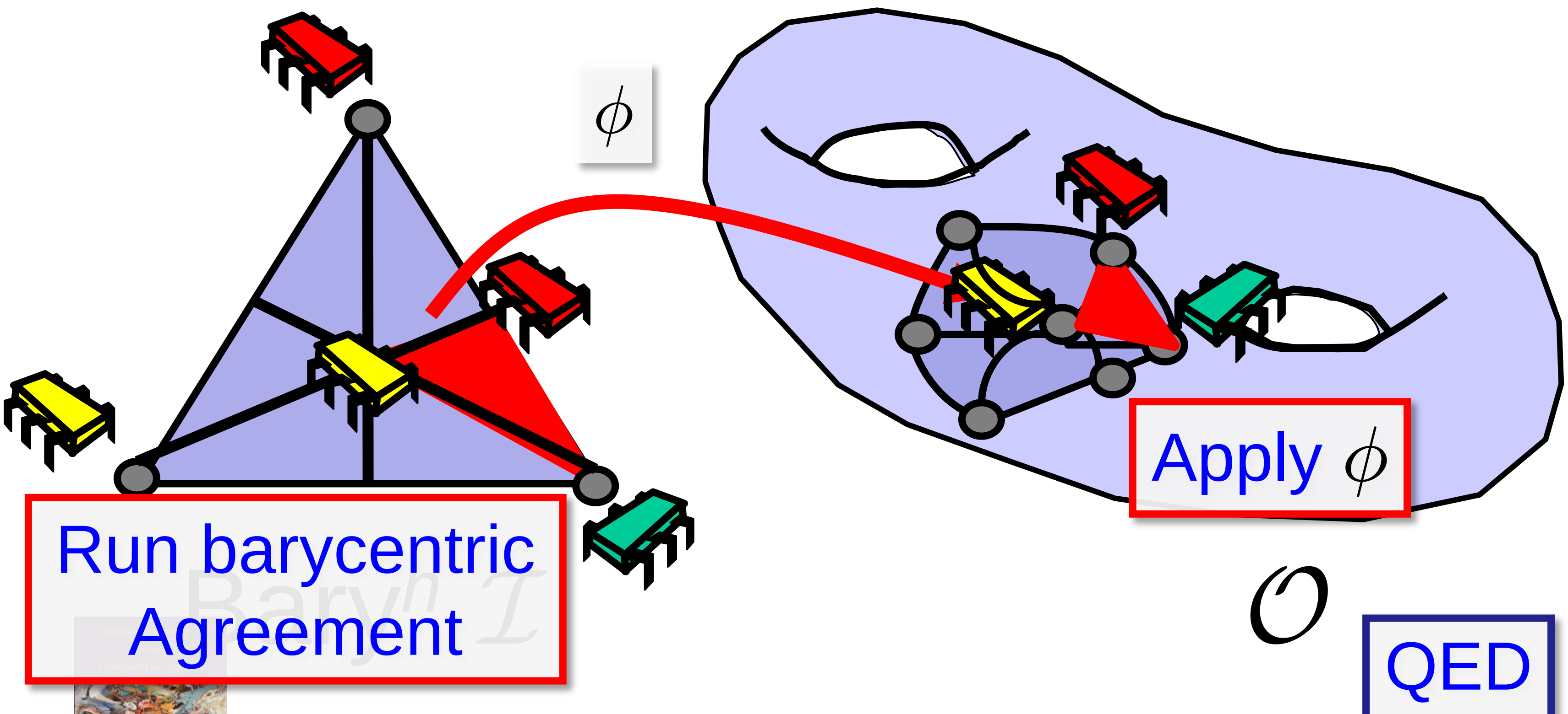




Map $\Rightarrow$ Protocol



Map $\Rightarrow$ Protocol



Protocol $\Rightarrow$ Map

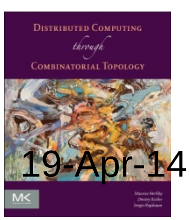
Lemma

If …

there is a WF-RW protocol for $(\mathcal{I},\mathcal{O},\Delta)$ …

then …

there is a continuous $f\colon |\mathrm{skel}^n\,\mathcal{I}| \to |\mathcal{O}|$ carried by Δ.



Protocol $\Rightarrow$ Map

Proof strategy

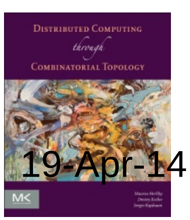
Inductive construction

g_d: $|\mathrm{skel}^d\, \mathcal{I}| \to |\Xi(\mathcal{I})|$.

Base $d = 0$

Define g_0: $|\mathrm{skel}^0\, \mathcal{I}| \to |\Xi(\mathcal{I})|$ …

Let $g_0(v)$ be any vertex in $\Xi(\{v\})$



Protocol $\Rightarrow$ Map

Induction Hypothesis

$g_{d\text{-}1}$: $|\text{skel}^{d\text{-}1}\,\mathcal{I}| \to |\Xi(\mathcal{I})|$

For all σ in $\text{skel}^{d\text{-}1}\,\mathcal{I}$

$g_{d\text{-}1}$ sends $|\text{skel}^{d\text{-}1}\,\sigma| \mapsto |\Xi(\sigma)|$

But $\Xi(\sigma)$ is $(d\text{-}1)$-connected (earlier theorem)

Can extend to g_d: $|\sigma| \mapsto |\Xi(\sigma)|$

Yielding g_d: $|\text{skel}^{d}\,\mathcal{I}| \to |\Xi(\mathcal{I})|$

Protocol $\Rightarrow$ Map

Constructed

g: $|\text{skel}^n \mathcal{I}| \to |\Xi(\mathcal{I})|$

Simplicial decision map

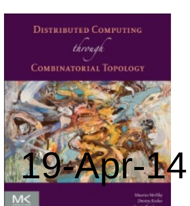
δ: $\Xi(\text{skel}^n \mathcal{I}) \to \mathcal{O}$

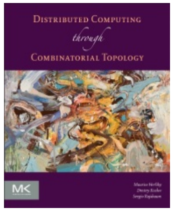
$|\delta|$: $|\Xi(\text{skel}^n \mathcal{I})| \to |\mathcal{O}|$

Composition $f = |\,\delta\,| \cdot g$ yields

f: $|\,\text{skel}^n \mathcal{I}| \to |\mathcal{O}|$ carried by Δ.

QED




Distributed Computing
through
Combinatorial Topology
MK