# Directed Topology and Concurrency Theory.

Lisbeth Fajstrup

Department of Mathematics
Aalborg University
Denmark

Sapporo 2017

# Directed Topology/Concurrency

**Take home message:**

Models of concurrency $\rightarrow$ geometry/topology questions $\rightsquigarrow$ New mathematics - usual topology does not suffice. $\rightsquigarrow$

- Results in concurrency.
- A new mathematical area.
- New questions in concurrency.

# Sequential programs

$x = 1;$
$y = x + 1;$
$x = y + 2;$

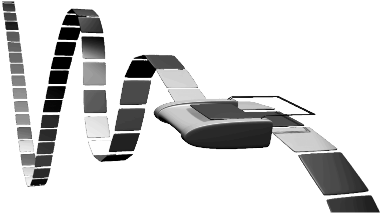
Another program:

- Pick up chopsticks
- Eat
- Put down chopsticks
- Think.

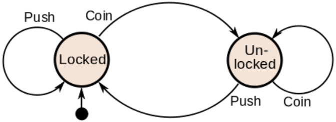
Instructions are executed in the order they are written.

# Models of sequential programs.

Turing machine

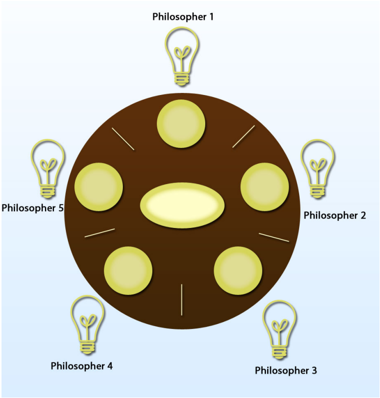


Graph (automaton).




Models needed to understand programs. To write code.

# Concurrent program. Dining philosophers.




Program for each philosopher

- Pick up right chopstick
- Pick up left chopstick
- Eat
- Put down chopsticks
- Think.

They interleave - no global prescribed order.

Philosopher 1 and 3 may proceed independently of each other.

Concurrent execution is faster than serial.

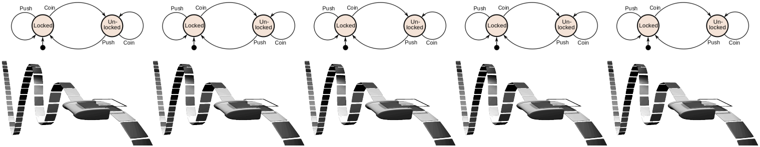
# Concurrency $\rightsquigarrow$ new problems

- Output depends on interleaving (and input)
- Deadlocks may appear
- Statespace explosion problem $\rightsquigarrow$ verification is often timeconsuming/impossible $\rightsquigarrow$ Limited use in critical software - airplanes, nuclear power plants, ...
- Resource starvation - one proces never gets access to resources needed

Need good models to understand and attack these problems.

# Concurrent programs - several models

Lots of graphs? Lots of Turing Machines? One for each thread



# Parallel programs $\neq$ Many copies of sequential programs.

Threads interact - share resources - communicate.
What model?

- Petri Nets
- Process calculus
- Event structures
- Higher Dimensional Automata

  $\vdots$
- Here: Locks.

Focus: Resources.

# The $PV$-model

E.W. Dijkstra, 1965, Mutual Exclusion. Resources cannot be used above their capacity.

$$Pa.Pb.Vb.Va|Pc.Pa.Vb.Va|Pb.Pc.Vb.Vc$$

$Pa$: request access to $a$. $Va$: release resource $a$.

Capacity of $a$: How many threads can use $a$ at the same time. I.e., be between $Pa$ and $Va$.

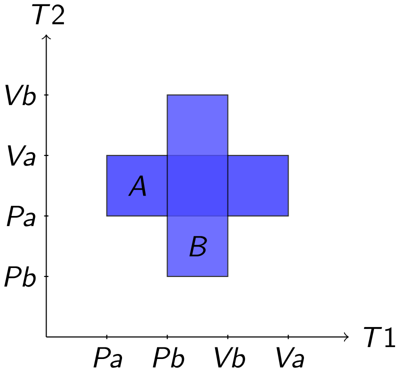
Vaughan Pratt "Modelling Concurrency with Geometry." 1992. Rob Van Glabbeek "Bisimulations for Higher Dimensional Automata," 1991

This model has geometry. (And needs topology.)

# The Swiss flag - two dining philosophers.

$$Pa.Pb.Vb.Va|Pb.Pa.Va.Vb$$

$T1 = PaPbVbVa$, $T2 = PbPaVaVb$



# $PV$-programs - controlling concurrency through locks

- A set of shared resources $\mathcal{R}$ - memory, printers,...
- A capacity function $\kappa : \mathcal{R} \to \mathbb{N}$. $\kappa(r)$ - maximal number of locks on $r$.

$PV$ programs

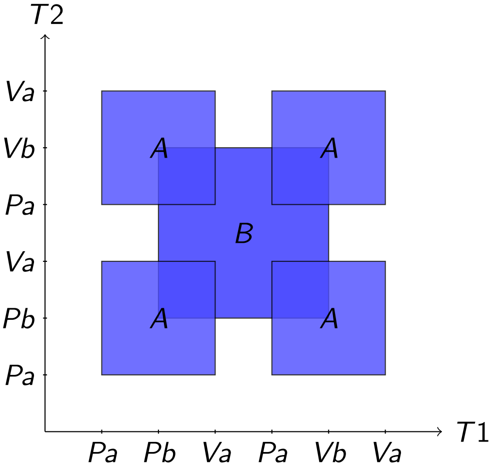
$$p ::= P_a \mid V_a \mid p.p \mid p|p \mid p + p \mid p^*$$

- $P_a$ - request to access $a$, if granted then lock $a$.
- $V_a$ -release $a$.
- Thread: No $p|p$. Sequential.
- At $\perp$ and $\top$: No locks.

# Geometrically

One thread - a graph.
$n$ threads in parallel - a product of $n$ graphs with “holes”.

$$T1 = T2 = Pa.Pb.Va.Pa.Vb.Va \qquad \kappa \equiv 1$$

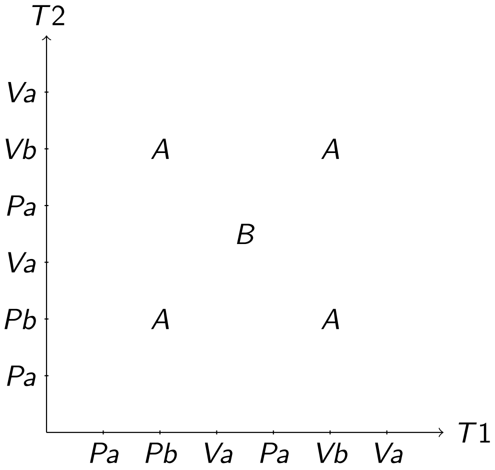


# Geometrically

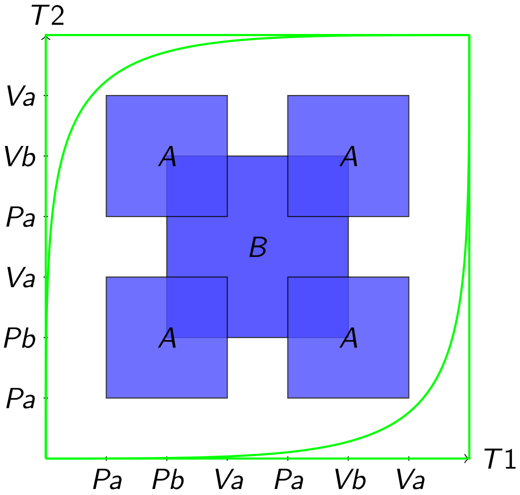
One thread - a graph.

$n$ threads in parallel - a product of $n$ graphs with "holes".

$$T1 = T2 = Pa.Pb.Va.Pa.Vb.Va \qquad \kappa > 1$$



# An execution is a *directed* path



# Equivalence of executions

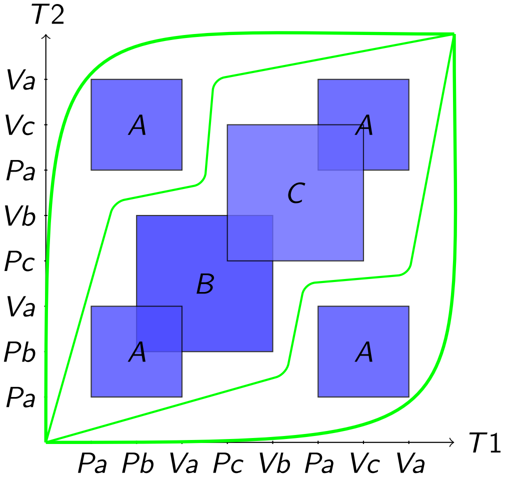
Executions are equivalent, if they have the same output given the same input.

Executions are equivalent, if the corresponding directed paths can be continuously deformed into eachother via directed paths!

**Why not piecewise linear. Why not only the 2-skeleton.**

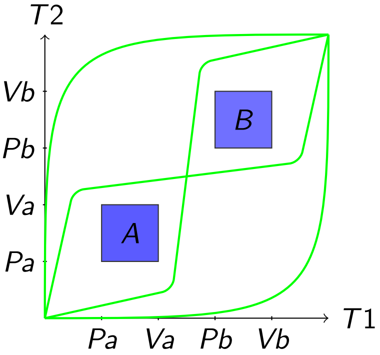
Want: Robust to subdivision. Also "true concurrency" - $k$ threads execute together $\sim$ a $k$-cube.

# Examples in the plane



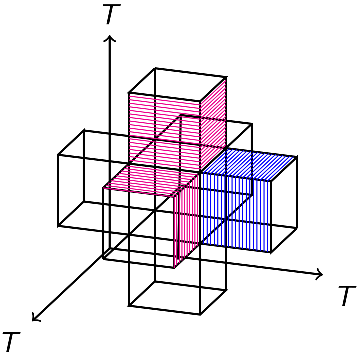

4 executions up to equivalence. Equivalent directed paths $\sim$ homotopic directed paths.

# Two holes



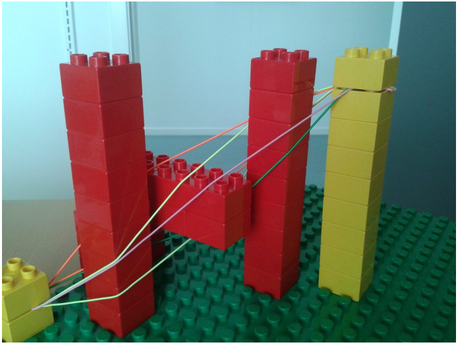

The number of directed paths up to equivalence depends on the order of the holes.

# $T = PaVa$, $\kappa = 1$, $T^3$




3! serial executions, pairwise inequivalent.

# With at least 3 threads, homotopic $\neq$ directed homotopic

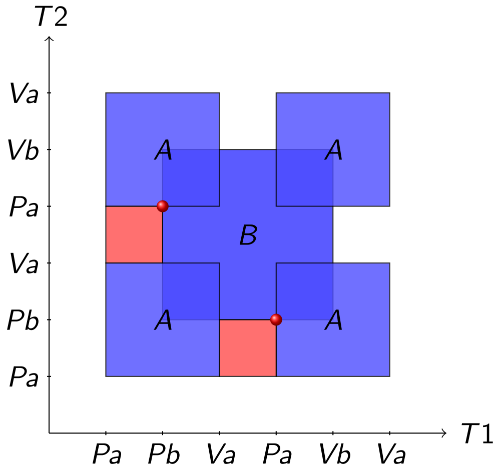


- 3 resources (red blocks)
- Capacity 2 and 3

Classification of executions $\rightsquigarrow$ verification for each equivalence class $\rightsquigarrow$ Too much identification is a serious PROBLEM. (Too little is just inefficient.)

Need: Topology with direction, directed topology. More later.... also categories....

# Deadlock - no time-directed paths leave the point

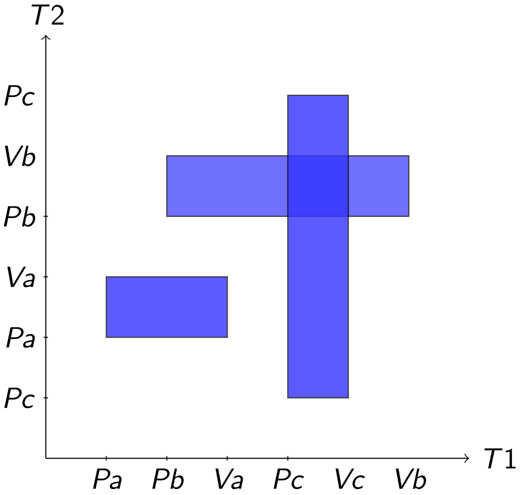



Red region: unsafe. No directed path leaves it. Red balls: deadlock points.

Algorithm finds deadlocks, unsafe region and unreachable region

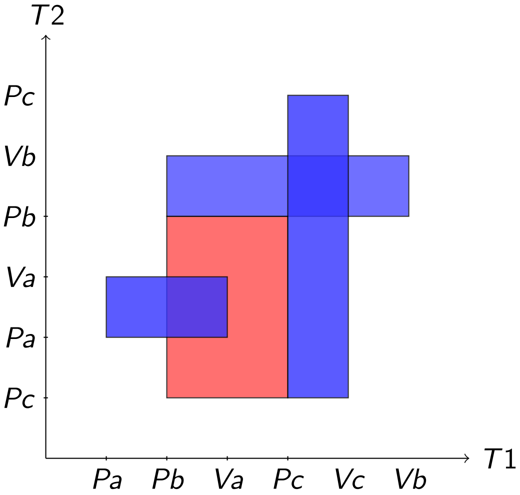
# Deadlocks

$T1 = PaPbVaPcVcVb, \; T2 = PcPaVaPbVbVc$



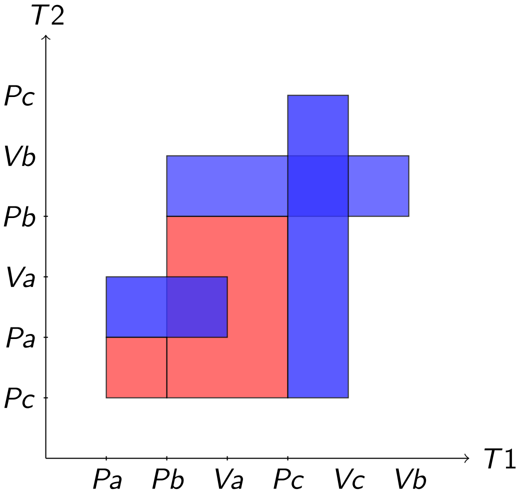
# Deadlocks

$T1 = PaPbVaPcVcVb,\ T2 = PcPaVaPbVbVc$



# Deadlocks

$T1 = PaPbVaPcVcVb,\ T2 = PcPaVaPbVbVc$



# Lipsky Papadimitriou

$$Px.Py.Pz.Vx.Pw.Vz.Vy.Vw|Pu.Pv.Px.Vu.Pz.Vv.Vx.Vz$$
$$|Py.Pw.Vy.Pu.Vw.Pv.Vu.Vv$$

Seems to have deadlocks in usual analysis (loops in request graph), but our methods show there is no deadlock.
Also building a physical model will work.

# A cut-off theorem for deadlocks

**Theorem 1, F.**

Let $T$ be a $PV$ thread accessing resources $\mathcal{R}$ with capacity $\kappa : \mathcal{R} \to \mathbb{N}$.
Let $T^n$ be $n$ copies of $T$ run in parallel.
$T^n$ is deadlock free for all $n$ if and only if $T^M$ is deadlock free, where $M = \Sigma_{r \in \mathcal{R}} \kappa(r)$.

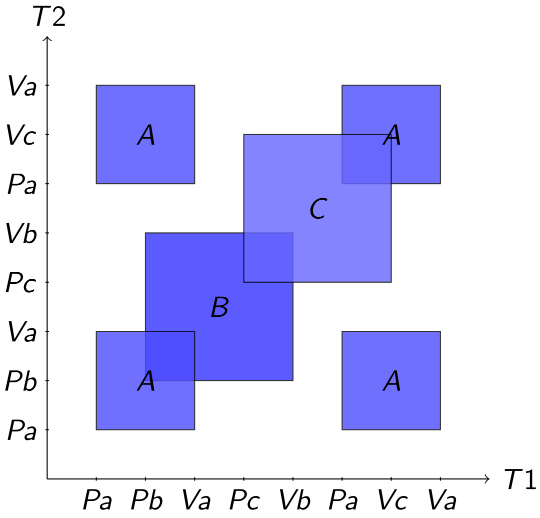
**Theorem 2, F.**

Given $\mathcal{R} = \{r_1, \ldots, r_k\}$ with capacity $\kappa : \mathcal{R} \to \mathbb{N}$, the thread
$T = Pr_1 Pr_2 Vr_1 Pr_3 Vr_2 \ldots Pr_k Vr_{k-1} Pr_1 Vr_k Vr_1$ satisfies:

- There is a deadlock in $T^M$ (and hence for all $n \geq M$)
- There are no deadlocks in $T^n$ for $n < M$

Deadlock at $\mathbf{x} = (\overbrace{x_1, \ldots, x_1}^{\kappa(r_k)}, \overbrace{x_2, \ldots, x_2}^{\kappa(r_1)}, \ldots, \overbrace{x_k \ldots x_k}^{\kappa(r_{k-1})})$ $x_i = Pr_i$, $x_1$ the *second* $Pr_1$

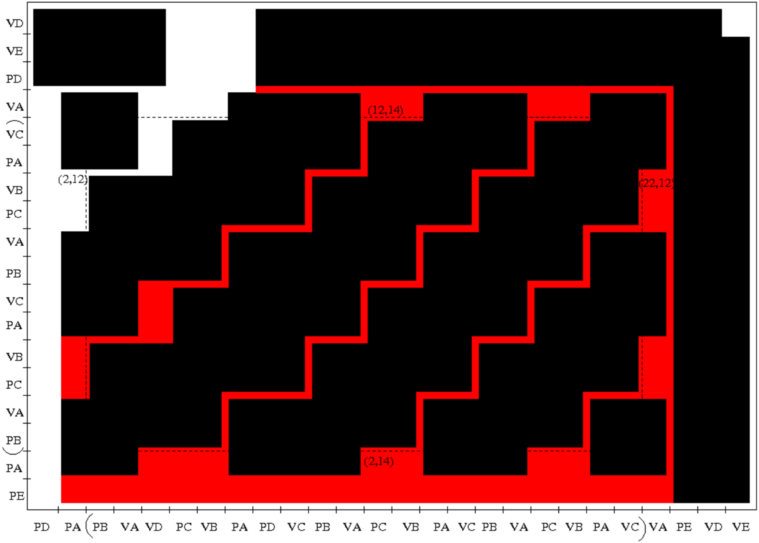
# Deadlock in $T^3$ Not in $T^2$




Deadlocks at $(6, 4, 2)$ $(4, 6, 2)$ $(2, 6, 4)$ $(6, 2, 4)$ $(4, 2, 6)$ $(2, 4, 6)$

# Programs with loops.

Deadlocks and unsafe area found in finitely many “deloopings”.



# Directed topology

Topological spaces with *direction*

**Definition**

Objects of **dTop** are d-Spaces: $(X, \vec{P}(X))$ where $X \in \mathbf{Top}$, $\vec{P}(X) \subseteq X^I$, the dipaths. $\vec{P}(X)$ is

- closed under concatenation,
- contains the constant paths
- closed under subpath, i.e., composition with $f : I \to I$ increasing but not necessarily surjective.

A d-map $f : X \to Y$ is a continuous map satisfying
$\gamma \in \vec{P}(X) \Rightarrow f \circ \gamma \in \vec{P}(Y)$

# Examples of d-spaces

- $X = S^1$, $\vec{P}(X)$ the clock wise paths.
- $\vec{P}(X)$ the constant paths.
- $\vec{P}(X) = X^I$, all paths.
- $X = \mathbb{R}^n$ and $\gamma \in \vec{P}(X)$ if $s \leq t \Rightarrow \gamma_i(s) \leq \gamma_i(t)$, $i = 1, \ldots, n$
- $X = \Gamma_1 \times \ldots \times \Gamma_n$ a product of directed graphs. $\gamma \in \vec{P}(X)$ if $\gamma_i$ is a directed path in $\Gamma_i$, $i = 1, \ldots, n$
- Main examples: Directed paths are paths which are locally increasing wrt. a reasonable local order structure.

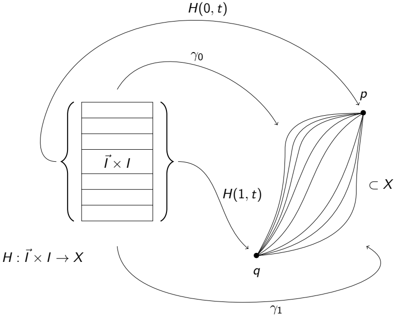
**In the model for $PV$-programs:**

$X$ is a state space. $\vec{P}(X)$ are (partial) executions.

# Equivalence of executions - dihomotopy

## Definition

Two dipaths $\gamma_0, \gamma_1 \in \vec{P}(X)(p, q)$ are dihomotopic, if there is a dimap $H : I \times \vec{I} \to X$ s.t. $H(t, 0) = \gamma_0(t)$, $H(t, 1) = \gamma_1(t)$, $H(0, I) = p$, $H(1, I) = q$. (Dipaths in $I$ are constant. Dipaths in $\vec{I}$ are non-decreasing)



# Equivalence of executions

$X$ is a state space - product of directed graphs, with holes. $\vec{P}(X) =$ locally non-decreasing paths. The following are equivalent.

- Execution paths $\gamma_1, \gamma_2 : \vec{I} \to X$ are dihomotopic
- They are in the same connected component of the path space $\vec{P}(X)(0,1)$ (Compact-Open topology)
- They are in the same connected component of the *trace space* $\vec{T}(X)(0,1)$ (dipaths modulo monotone reparametrization)

# Questions - concurrency

- Classify executions up to equivalence $\leftrightarrow$ Find components of $\vec{P}(X)(p,q)$ or $\vec{T}(X)(p,x)$.
- Algorithms - connections to configuration spaces. (Raussen, Ziemianski, Meshulam)
- Equivalence of programs - bisimulation $\leftrightarrow$ dicoverings. (F.)
- Verification: Will the program behave?
  - If there is a "bad" state, is it reachable? Is there a directed path to it from the initial state(s)
  - Will all possible executions give a "true" result? Study directed paths up to *equivalence.*
- State space explosion. Now we have infinitely many states - want dicomponents!

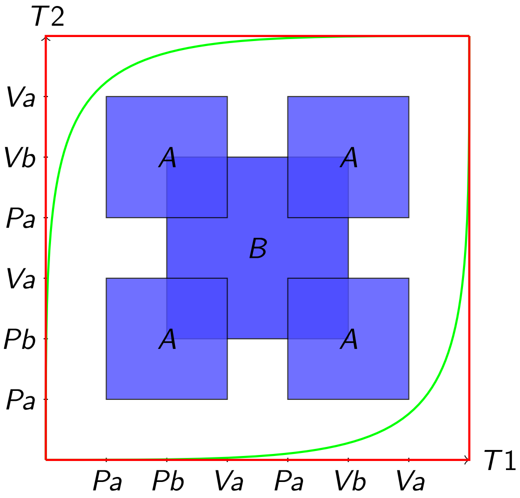
## It's complicated.

## (Almost) Every topological space is an execution space - same homology.

**Theorem (K. Ziemianski, 2013)**

Any finite simplicial complex $S$ on $n$ vertices is a component of a space of executions:
There is a subset $F \subset \vec{I}^n$ a union of $n$-rectangles such that $\vec{P}(\vec{I}^n \setminus F)(0,1)$ is homology equivalent to $S \sqcup S^{n-2}$

# Serializability - part of classifying executions




Two executions up to dihomotopy. Equivalent to the serial executions $T1.T2$ and $T2.T1$. Serial $\Rightarrow$ easier to verify.

# Serializability - a cut-off theorem

**Theorem (Fajstrup)**

Let $T$ be a $PV$-thread accessing resources $\mathcal{R}$ all of capacity 1.
Let $T^n$ be $n$ copies of $T$ run in parallel. Then $T^n$ is serializable if and only if $T^2$ is serializable.

Consequence: Easy to test for serializability of $T^n$ for all $n$.

# Serializability. General $\kappa$.

**Definition**

$\mathbf{x} = (x_1, \ldots, x_n) \in X$ is a local choice point if

1. For some $\tilde{r}$ there is $S = \{i_1, \ldots, i_m\} \subset [1:n]$, $m \geq 2$, such that $x_{i_j} = P\tilde{r}$
2. $\rho_{\tilde{r}}(x) = \kappa(\tilde{r}) - 1$,
3. for $i \notin S$ either $x_i = \top$ or $x_i = Pr$ and $\rho_r(\mathbf{x}) = \kappa(r)$

### Theorem (F., applying Raussen 2000)

No local choice points in $T^M$ where $M = 1 + \Sigma_{r\in\mathcal{R}}\kappa(r) \Rightarrow$ no local choice points in $T^n$ for any $n$.
If $\kappa(r) = 1$, then choice point in $T^n$ for all $n \geq 2$.
$T^n$ non-serializable for some $n \Rightarrow$ local choice points in $T^M$.

### Theorem F.

For $\kappa : \mathcal{R} \to \mathbb{N} \setminus \{1\}, T = Pr_1 Pr_2 Vr_1 Pr_3 Vr_2 \ldots Pr_k Vr_{k-1} Pr_1 Vr_k Vr_1$ satisfies:

- $T^M$ has a local choice point $M = \Sigma_{r\in\mathcal{R}}\kappa(r) + 1$
- $T^n$ has no choice point for $n \leq M - 2$.

Choice point in $T^{M-2}$ at $\mathbf{x} = (\overbrace{x_1, \ldots, x_1}^{\kappa(r_k)-1}, \overbrace{x_2, \ldots, x_2}^{\kappa(r_1)}, \ldots, \overbrace{x_k \ldots x_k}^{\kappa(r_{k-1})})$

# The directed path space in general

**Algorithm (M.Raussen)**

Calculates a prod-simplicial model of $\vec{P}(X)(0,1)$, when $X = I^n \setminus F$, $F$ is a union of $n$-rectangles. Method behind: Nerve Lemma.

Implemented in Alcool. Output: Homology of $\vec{P}(X)(0,1)$. A representative for each connected component. OBS: Verification only needed on those.

# Loops and the trace space algorithm.

- The trace space algorithm is combined with periodicity.
- An automaton which outputs the schedules, i.e., the info to build the trace space. (F. 2011, F., Goubault, Haucourt, Mimram, Raussen, 2012)

For programs $T = (PaVa)^*$ in parallel with itself, $T^n$, configurations spaces give good models. (Raussen, Zimianski, Meshulam)

# Maths questions - curiousity driven

Ditopology

- Dihomotopy
- Dihomology
- How many d-structures on a given space.
- Model structure?
- etc.

Relationships between **Top** and **dTop**:

- Forgetful functor $U : \mathbf{dTop} \to \mathbf{Top}$.
- Right adjoint: $\vec{P}(X) = X^I$. All continuous maps to $(X, \vec{P}(X))$ are dimaps.
- Left adjoint: $\vec{P}(X) =$ the constant paths. All continuous maps from $(X, \vec{P}(X))$ are dimaps.

**Theorem: (F. 2011, F.,J. Pita Costa 2016)**

The set of d-structures on $X \in \mathbf{Top}$ is a Heyting algebra under inclusion. $(\vec{P} \wedge \vec{Q} = \vec{P} \cap \vec{Q}$, $\vec{P} \vee \vec{Q} = \overline{\vec{P} \cup \vec{Q}}$, closure under concatenation and subpath.)

# Directed topology - invariants?

Remember $\vec{P}(X)(p,q)$ the space of dipaths, compact-open topology.

## The fundamental category

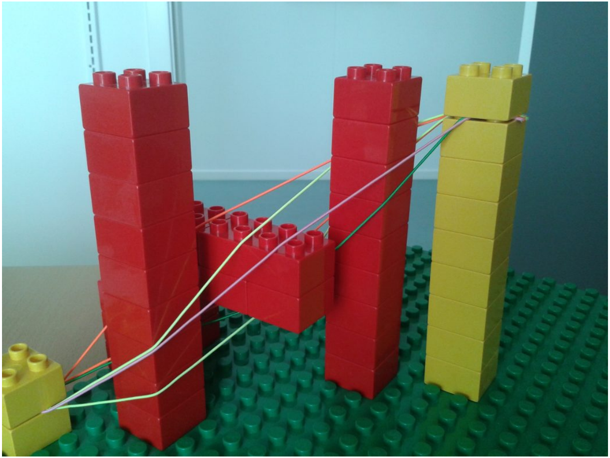
Components of $\vec{P}(X)(p,q)$ organised in The fundamental category

- Objects: All points in $X$
- Morphisms: $\vec{\pi}_1(X)(p,q)$
  - The connected components of $\vec{P}(X)(p,q)$
  - Equivalently: $\vec{\pi}_1(X)(p,q) = \vec{P}(X)(p,q)/\sim$ where $\sim$ is dihomotopy.

  $\vec{\pi}_1(X)(p,q)$ is the number of inequivalent (partial) executions.

OBS: It is *not* a group.

# No cancellation in $\vec{\pi}_1$



# Other invariants

(co)Homology, homotopy groups,... of $\vec{P}(X)(p,q)$.
*Variation* of basepoints $\rightarrow$ (homo)morphisms.
$\sigma \in \vec{P}(X)(p',p)$ induces

$$\sigma^* : \vec{P}(X)(p,q) \rightarrow \vec{P}(X)(p',q)$$

.

$$\sigma_* : \vec{P}(X)(r,p') \rightarrow \vec{P}(X)(r,p)$$

Dihomology - (co)Homology M.Grandis, 2009, S. Krishnan, 2013, J.Dubut, E.Goubault, J. Goubault Larrecq

# Directed coverings - equivalence of programs

Computer Science p.o.v. - geometric version:

Two programs with geometric model $X$ and $Y$ are equivalent (bisimilar), if there is a program $Z$ and maps

$$f : Z \to X, g : Z \to Y$$

such that every dipath in $X$ *lifts uniquely* to $Z$ along $f$: For $\gamma : \vec{I} \to X$ and $z_0 \in f^{-1}(\gamma(0))$ there is unique $\tilde{\gamma} : I \to Z$ s.t. $f \circ \tilde{\gamma} = \gamma$. I.e., $f$ and $g$ are *dicoverings* and similarly for $g$


- F., Dicovering spaces. Homology, Homotopy Appl., Vol. 5, Nr. 2, 2003
- F., J. Rosický, A convenient category for directed homotopy. I: Theory Appl. Categ., Vol. 21, Nr. 1, 2008.
- F. Classification of dicoverings. I: Topol. Appl., Vol. 157, Nr. 15, 2010. 2015.

# Commercial and main collaborators