Geometry of Trace Spaces and (some) Applications

Eric Goubault

CEA LIST and Ecole Polytechnique, France

joint work with Samuel Mimram, Emmanuel Haucourt, Christine Tasson, Lisbeth Fajstrup, Martin Raussen

Workshop 15th years of LSV and CNRS Silver Medal to my elder brother ;-)
LSV, ENS Cachan

6th of February 2012

Geometry of Trace Spaces

Eric Goubault

CEA LIST and Ecole Polytechnique, France

joint work with Samuel Mimram, Emmanuel Haucourt, Christine Tasson, Lisbeth Fajstrup, Martin Raussen

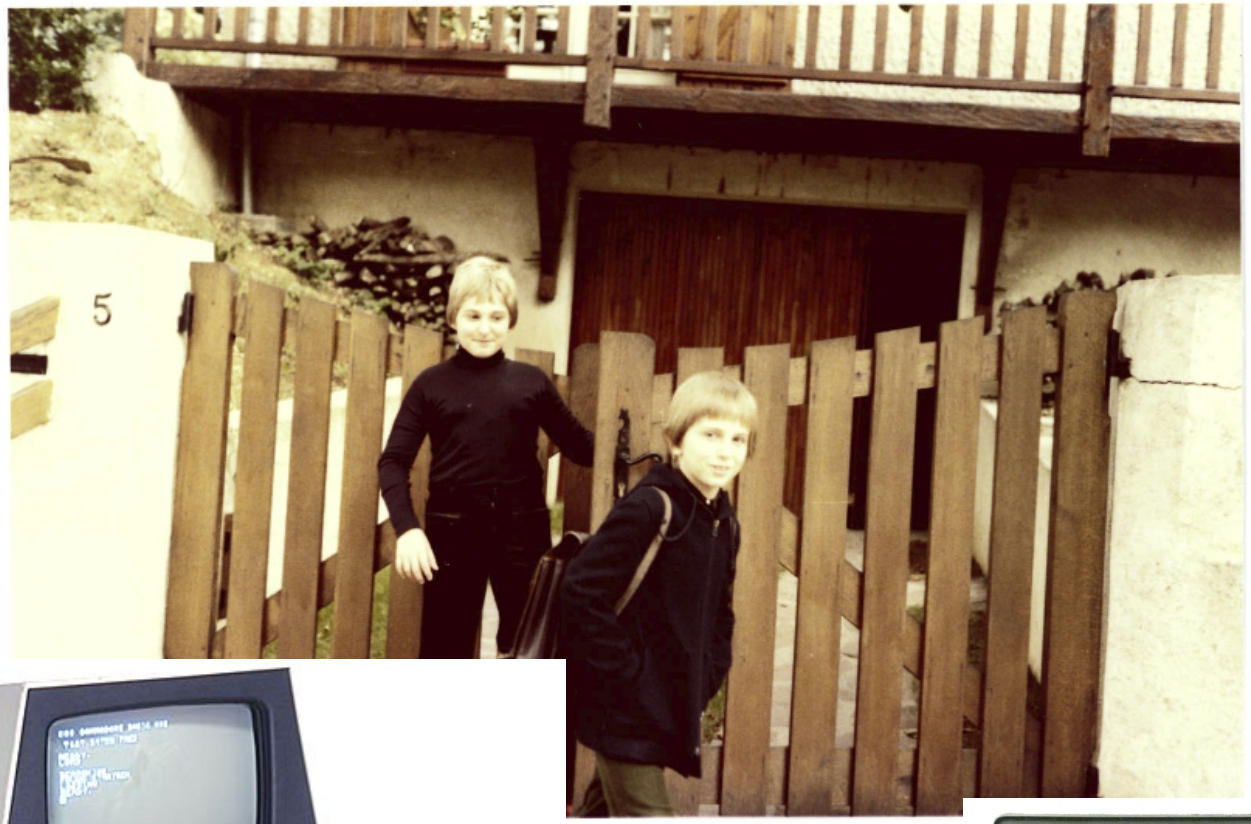
Workshop 15th years of LSV and CNRS Silver Medal to my elder brother ;-)
LSV, ENS Cachan

6th of February 2012

Jean

Jean

Jean




commodore
PET
2001 Series
professional computer


3, 1 4 1 5 9 2 6 5 4
OFF
ON
PRGM
RUN
FIX DEG SCI RAD ENG GRD
A
B
GSB
f
g
GTO
STO
RCL
h
MEM
PREFIX
PRGM
REG
ENTER
CHS
EEX
CLX

Jean


5


3,14159265 4
OFF
ON
PRGM
RUN
FIX DEG SCI RAD ENG GRD
A
B
GSB
f
g
R↓ I DSE (i) ISG
GTO
STO
RCL
h
MEM PREFIX PRGM REG Σ
ENTER
CHS
EEX
CLx
SIN
COS
TAN
7
8
9
4
5
6
LN
LOG

Jean

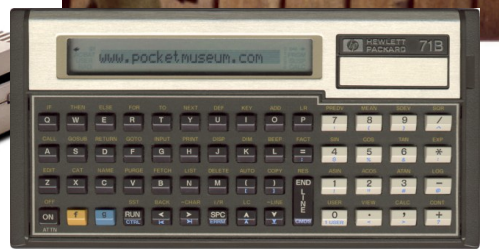
ÉCOLE POLYTECHNIQUE

Jean

Jean

Jean


5


www.pocketmuseum.com
HEWLETT PACKARD
71B


Object Type
Symbol
Example
HEWLETT PACKARD
ADVANCE SCIENTIFIC CALCULATOR
28S

And then...

- Programming (Forth, LISP, functional languages etc.)
- Computer algebra (Risch formal integration algorithm...)
- Proof theory / Semantics
- ...

And then...

Homology, Homotopy and Applications, vol.5(2), 2003, pp.137–209

ON THE GEOMETRY OF INTUITIONISTIC S4 PROOFS

JEAN GOUBAULT-LARRECQ and ÉRIC GOUBAULT

(*communicated by Gunnar Carlsson*)

Abstract

The Curry-Howard correspondence between formulas and types, proofs and programs, proof simplification and program execution, also holds for intuitionistic modal logic S4. It turns out that the S4 modalities translate as a monoidal comonad on the space of proofs, giving rise to a canonical augmented simplicial structure. We study the geometry of these augmented simplicial sets, showing that each type gives rise to an augmented simplicial set which is a disjoint sum of nerves of finite lattices of points, plus isolated (-1)-dimensional subcomplexes. As an application, we give semantics of modal proofs (a.k.a., programs) in categories of augmented simplicial sets and of topological spaces, and prove a completeness result in the style of Friedman: if any two proofs have the same denotations in each augmented simplicial model, then they are convertible. This result rests both on the fine geometric structure of the constructed spaces of proofs and on properties of subscone categories—the categorical generalization of the notion of logical relations used in lambda-calculus.

And then...

Homology, Homotopy and Applications, vol.5(2), 2003, pp.137–209

ON THE GEOMETRY OF INTUITIONISTIC S4 PROOFS

JEAN GOUBAULT-LARRECQ AND ÉRIC GOUBAULT

(*communicated by Gunnar Carlsson*)

Abstract

The Curry-Howard correspondence between formulas and types, proofs and programs, proof simplification and program execution, also holds for intuitionistic modal logic S4. It turns out that the S4 modalities translate as a monoidal comonad on the space of proofs, giving rise to a canonical augmented simplicial structure. We study the geometry of these augmented simplicial sets, showing that each type gives rise to an augmented simplicial set which is a disjoint sum of nerves of finite lattices of points, plus isolated (-1)-dimensional subcomplexes. As an application, we give semantics of modal proofs (a.k.a., programs) in categories of augmented simplicial sets and of topological spaces, and prove a completeness result in the style of Friedman: if any two proofs have the same denotations in each augmented simplicial model, then they are convertible. This result rests both on the fine geometric structure of the constructed spaces of proofs and on properties of subscone categories—the categorical generalization of the notion of logical relations used in lambda-calculus.

Computing
DOI 10.1007/s00607-011-0182-8

A generalization of p-boxes to affine arithmetic

Olivier Bouissou · Eric Goubault · Jean Goubault-Larrecq · Sylvie Putot

Received: 18 March 2011 / Accepted: 2 December 2011

Abstract We often need to deal with information that contains both interval and probabilistic uncertainties. P-boxes and Dempster–Shafer structures are models that unify both kind of information, but they suffer from the main defect of intervals, the wrapping effect. We present here a new arithmetic that mixes, in a guaranteed manner, interval uncertainty with probabilities, while using some information about variable dependencies, hence limiting the loss from not accounting for correlations. This increases the precision of the result and decreases the computation time compared to standard p-box arithmetic.

Keywords Affine arithmetic · P-boxes · Dempster–Shafer structures

Mathematics Subject Classification (2010) 60A86 · 65G30 · 65G50 · 65C50

And then...

Homology, Homotopy and Applications, vol.5(2), 2003, pp.137–209

ON THE GEOMETRY OF INTUITIONISTIC S4 PROOFS

JEAN GOUBAULT-LARRECQ AND ÉRIC GOUBAULT

(*communicated by Gunnar Carlsson*)

Abstract

The Curry-Howard correspondence between formulas and types, proofs and programs, proof simplification and program execution, also holds for intuitionistic modal logic S4. It turns out that the S4 modalities translate as a monoidal comonad on the space of proofs, giving rise to a canonical augmented simplicial structure. We study the geometry of these augmented simplicial sets, showing that each type gives rise to an augmented simplicial set which is a disjoint sum of nerves of finite lattices of points, plus isolated (-1)-dimensional subcomplexes. As an application, we give semantics of modal proofs (a.k.a., programs) in categories of augmented simplicial sets and of topological spaces, and prove a completeness result in the style of Friedman: if any two proofs have the same denotations in each augmented simplicial model, then they are convertible. This result rests both on the fine geometric structure of the constructed spaces of proofs and on properties of subscone categories—the categorical generalization of the notion of logical relations used in lambda-calculus.

Computing
DOI 10.1007/s00607-011-0182-8

A generalization of p-boxes to affine arithmetic

Olivier Bouissou · Eric Goubault · Jean Goubault-Larrecq · Sylvie Putot

Received: 18 March 2011 / Accepted: 2 December 2011

Abstract We often need to deal with information that contains both interval and probabilistic uncertainties. P-boxes and Dempster–Shafer structures are models that unify both kind of information, but they suffer from the main defect of intervals, the wrapping effect. We present here a new arithmetic that mixes, in a guaranteed manner, interval uncertainty with probabilities, while using some information about variable dependencies, hence limiting the loss from not accounting for correlations. This increases the precision of the result and decreases the computation time compared to standard p-box arithmetic.

Keywords Affine arithmetic · P-boxes · Dempster–Shafer structures

Mathematics Subject Classification (2010) 60A86 · 65G30 · 65G50 · 65C50

and I hope more!

Contents of the talk

- A rush tour of geometric semantics of concurrent programs
- Combinatorics/algorithmics of trace spaces
- Applications:
  - Static analysis...well not enough time!
  - Protocol complexes of fault-tolerant distributed systems...maybe but do not count on it too much!

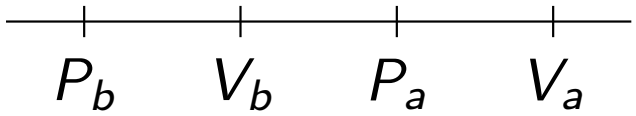
Geometric semantics

A program

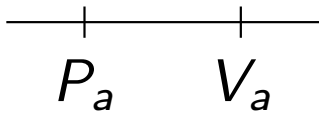
$$P_b\,;\texttt{x:=1}\,;V_b\,;P_a\,;\texttt{y:=2}\,;V_a \mid P_a\,;\texttt{y:=3}\,;V_a$$

will be interpreted as a **directed space**:

- $P_b.V_b.P_a.V_a$

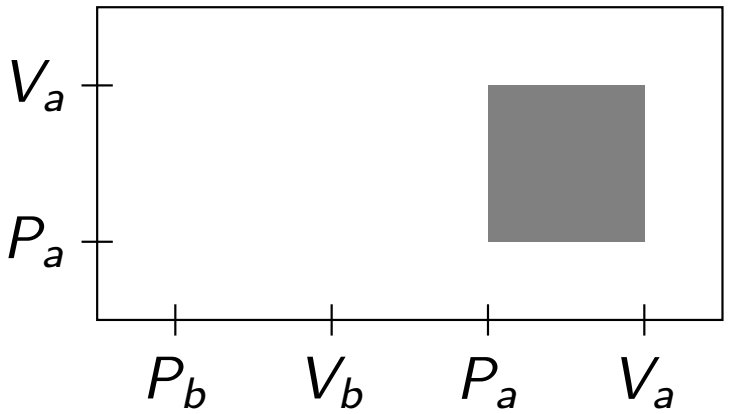


- $P_a.V_a$

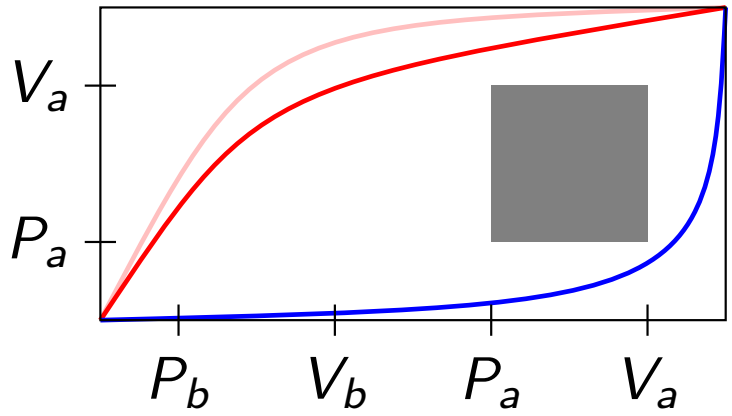


- $P_b.V_b.P_a.V_a \quad | \quad P_a.V_a$

Forbidden regions



A **scheduling** is the homotopy class of a path.



We want to compute *a path in every scheduling*.
P_b;x:=1;V_b;P_a;y:=2;V_a | P_a;y:=3;V_a can be scheduled in three different ways:

y:=3;x:=1;y:=2
$(x, y) = (1, 2)$

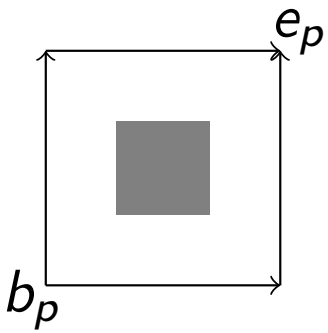
x:=1;y:=3;y:=2
$(x, y) = (1, 2)$

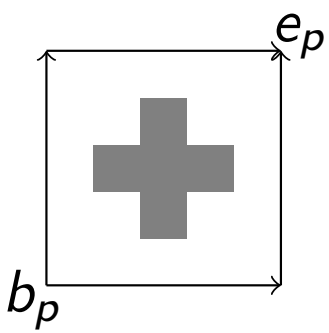
x:=1;y:=2;y:=3
$(x, y) = (1, 3)$

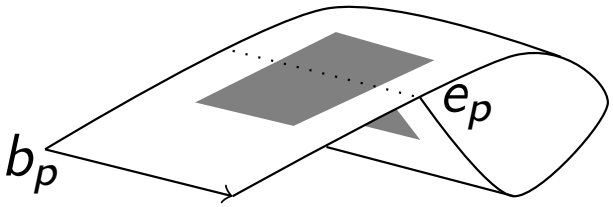
Examples of geometric semantics

To each program p we associate a d-space (H_p, b_p, e_p):

$P_a.V_a|P_a.V_a$ $\quad$ $P_a.P_b.V_b.V_a|P_b.P_a.V_a.V_b$ $\quad$ $P_a.(V_a.P_a)^*|P_a.V_a$

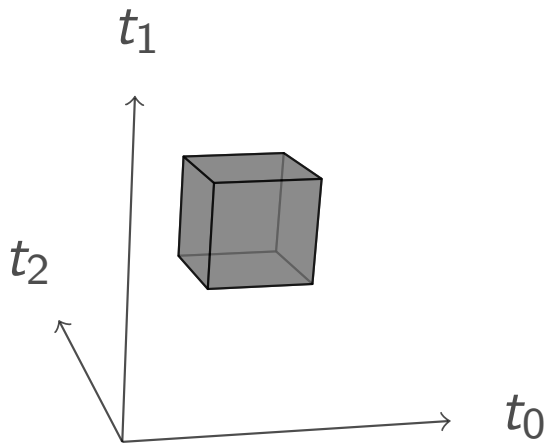




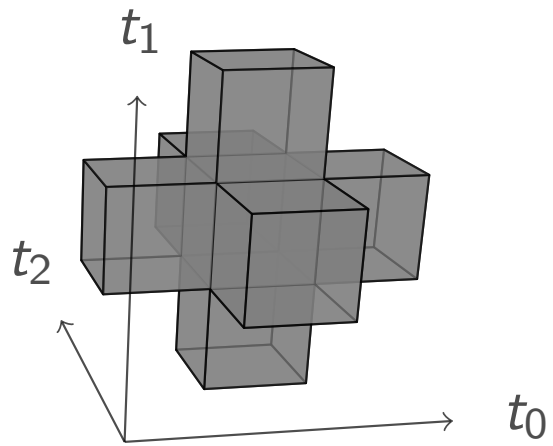


Examples of geometric semantics

$P_a.V_a|P_a.V_a|P_a.V_a$
$(\kappa_a = 2)$



$P_a.V_a|P_a.V_a|P_a.V_a$
$(\kappa_a = 1)$



Determining traces can be intricate!

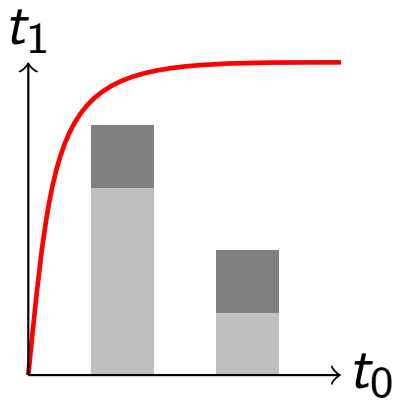
Px.Py.Pz.Vx.Pw.Vz.Vy.Vw | Pu.Pv.Px.Vu.Pz.Vv.Vx.Vz Py.Pw.Vy.Pu.Vw.Pv.Vu.Vv

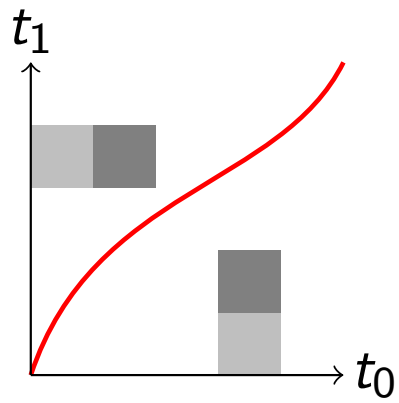
Lipsky

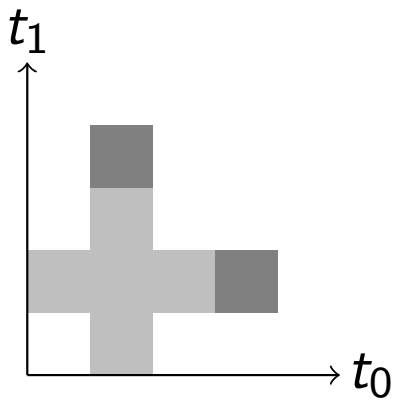
Determining trace spaces

The trace space between two points b and e is the space of directed paths modulo reparameterization, suitably topologized...

To determine it (its connected components?), the main (naive?) idea is to extend the forbidden cubes downwards in various directions and look whether there is a path from b to e.







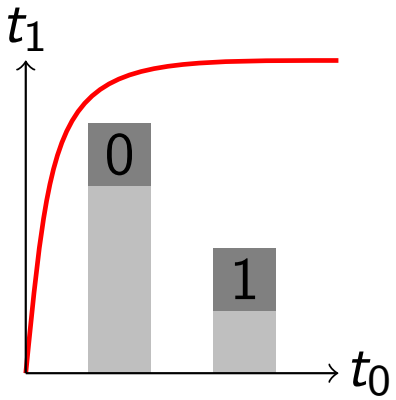
By combining those information, we will be able to compute traces modulo homotopy.

The directions in which to extend the holes will be coded by boolean matrices M.

The index poset

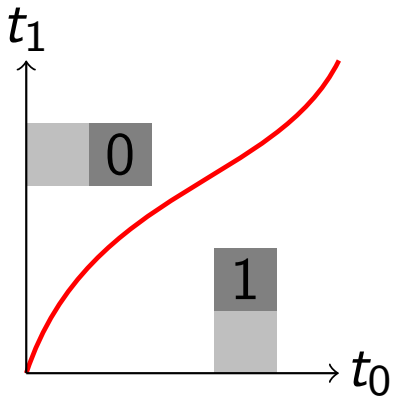
$\mathcal{M}_{l,n}$: boolean matrices with l rows and n columns.

X_M: space obtained by *extending*
for every (i,j) such that $M(i,j)=1$
the forbidden cube i downwards
in every direction other than j



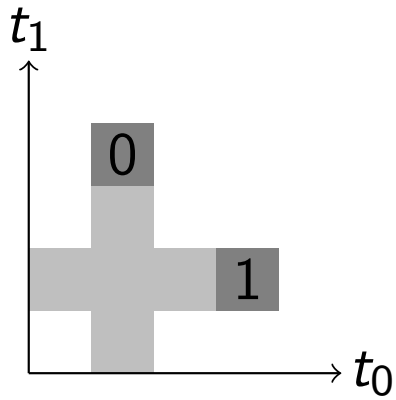
$$\begin{pmatrix} 1 & 0 \\ 1 & 0 \end{pmatrix}$$

alive



$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

alive

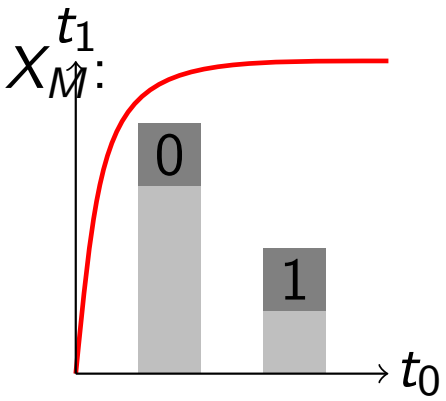


$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

dead

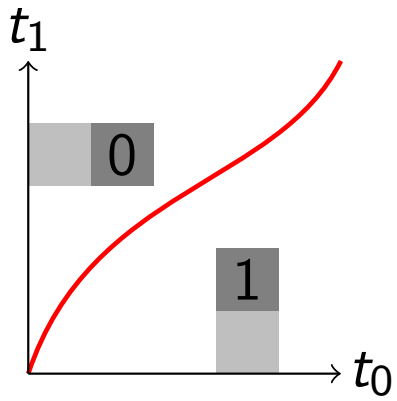
The index poset

$\mathcal{M}_{l,n}$: boolean matrices with l rows and n columns.



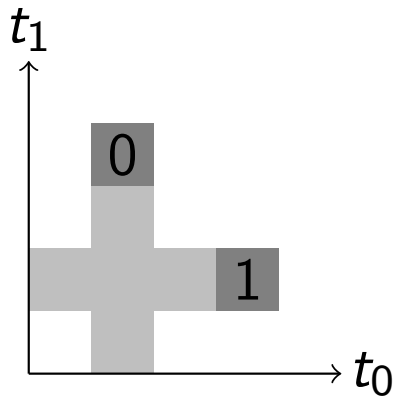
$$\begin{pmatrix} 1 & 0 \\ 1 & 0 \end{pmatrix}$$

alive



$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

alive



$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

dead

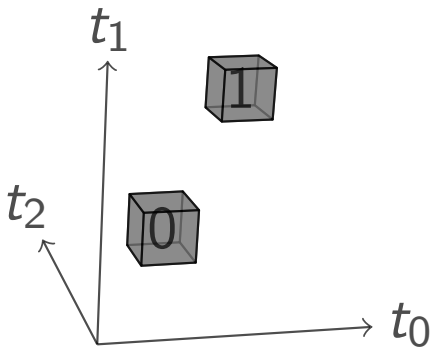
$\Psi : \mathcal{M}_{l,n} \to \{0, 1\}$:

- $\Psi(M) = 0$ if there is a path $b \to e$: M is alive
- $\Psi(M) = 1$ if there is no path $b \to e$: M is dead

The index poset

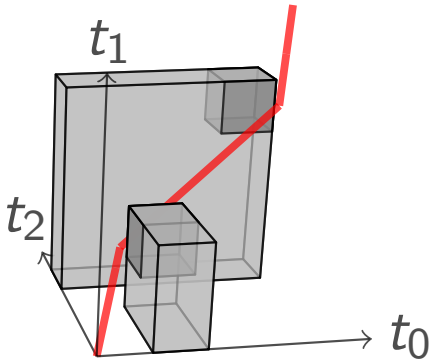
$$P_a.V_a.P_b.V_b \quad | \quad P_a.V_a.P_b.V_b \quad | \quad P_a.V_a.P_b.V_b$$

$(\kappa_a = 2,\ \kappa_b = 2)$



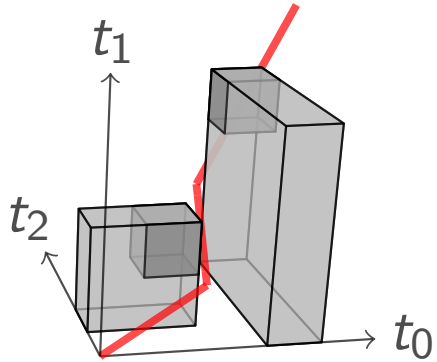
$$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

alive



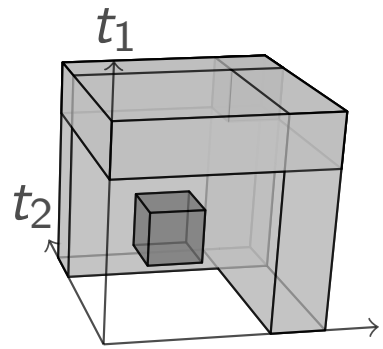
$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

alive



$$\begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$$

alive



$$\begin{pmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \end{pmatrix}$$

dead

Alive and dead?

Important matrices are

- the **dead poset** $D(X) = \{M \in \mathcal{M}_{I,n}^{\mathcal{C}} \ / \ \Psi(M) = 1\}$.
- the **index poset** $\mathcal{C}(X) = \{M \in \mathcal{M}_{I,n}^{R} \ / \ \Psi(M) = 0\}$ (the alive matrices).
- consider the entrywise ordering $(0 < 1)$ on matrices.

General results:

$$D(X) \rightsquigarrow \mathcal{C}(X) \rightsquigarrow \text{trace spaces, up to homotopy equivalence}$$

(hence at least homotopy classes of traces...)

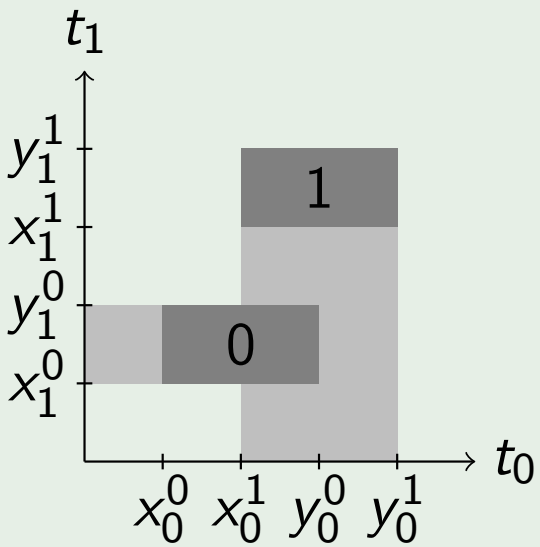
Proposition

A matrix $M \in \mathcal{M}^C_{l,n}$ is in $D(X)$ iff it satisfies

$$\forall (i,j) \in [0:l[\times[0:n[, \qquad M(i,j)=1 \quad \Rightarrow \quad x_j^i < \min_{i' \in R(M)} y_j^{i'}$$

where $R(M)$: indexes of non-null rows of M.

Example



$$M = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \quad \begin{array}{l} x_1^0 = 1 < 2 = \min(y_1^0, y_1^1) \\ x_0^1 = 2 < 3 = \min(y_0^0, y_0^1) \end{array}$$

The index poset

Proposition

A matrix M is in $\mathcal{C}(X)$ iff for every $N \in D(X)$, $N \not\leqslant M$.

Remark

$N \not\leqslant M$: there exists (i,j) s.t. $N(i,j) = 1$ and $M(i,j) = 0$.

Remark

Since $\mathcal{C}(X)$ is downward closed it will be enough to compute the set $\mathcal{C}_{\max}(X)$ of maximal alive matrices.

Connected components

$M \wedge N$: pointwise min of M and N

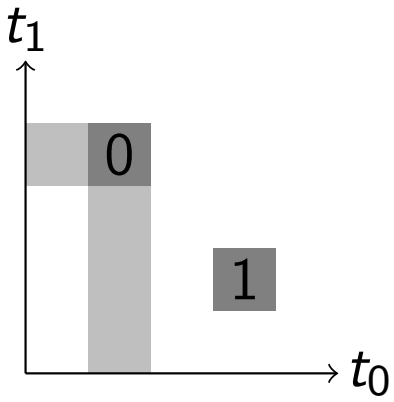
Definition

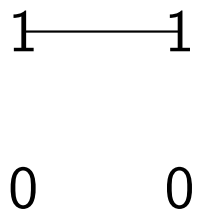
Two matrices M and N are **connected** when $M \wedge N$ does not contain any null row.

Proposition

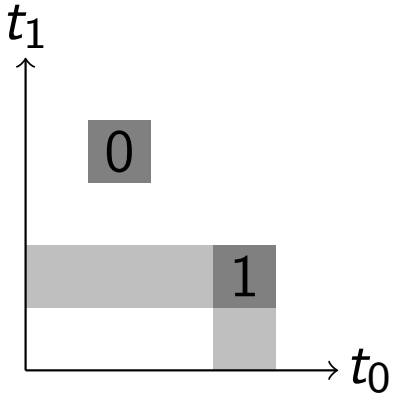
The connected components of $\mathcal{C}(X)$ are in bijection with homotopy classes of traces $b \to e$ in X.

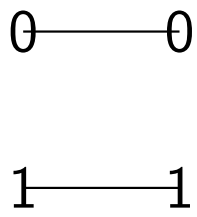
First dead matrix:



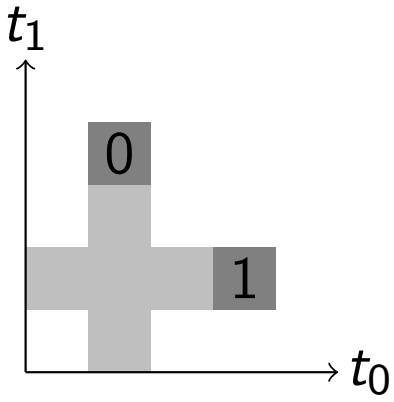


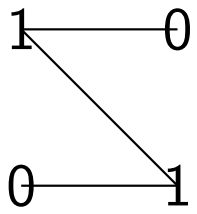
Second dead matrix:



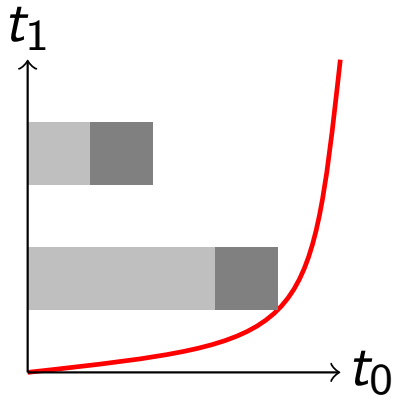


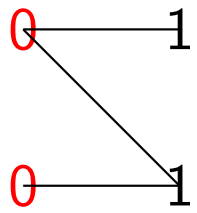
Third and last (minimal) dead matrix:



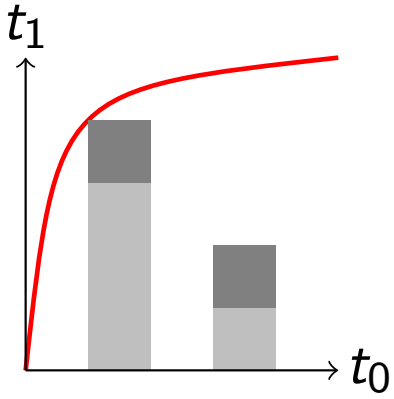


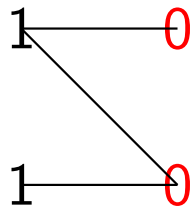
First (maximal) alive matrix:



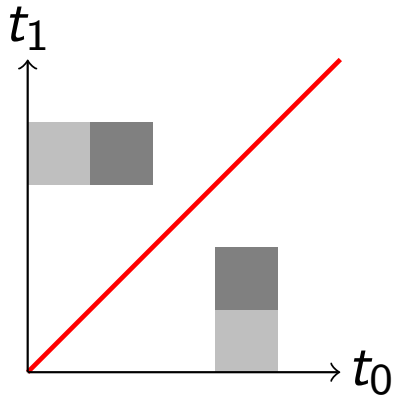


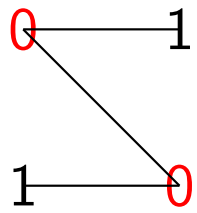
Second alive matrix:





Third (and last) maximal alive matrix:





Hypergraph transversal

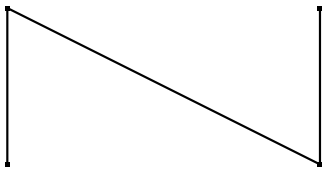
- An *hypergraph* $H = (V, E)$ consists of a set V of *vertices* and a set E of edges, where an *edge* is a subset of V
- A *transversal* T of H is a subset of V such that $T \cap e \neq \emptyset$ for every edge $e \in E$.

$D(X) \Rightarrow$ hypergraph H:

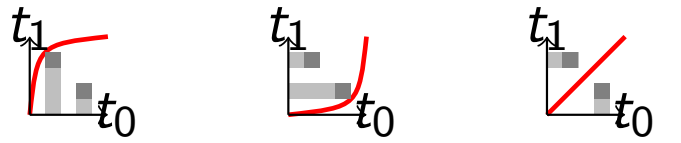
- vertices: $[0 : l[\times [0 : n[$
- hyperedges: $\{(i, j) \ / \ D(i, j) = 1\}$ (D is a matrix in $D(X)$)

The sets $\{(i, j) \ / \ M(i, j) = 0\}$, where M is a maximal matrix of $\mathcal{C}(X)$, correspond to *minimal transversals/hitting sets* (wrt inclusion order) of H.

...is itself an hypergraph (same vertices, but hitting sets as hyper-edges):



$M_2 \quad M_3 \quad M_1$



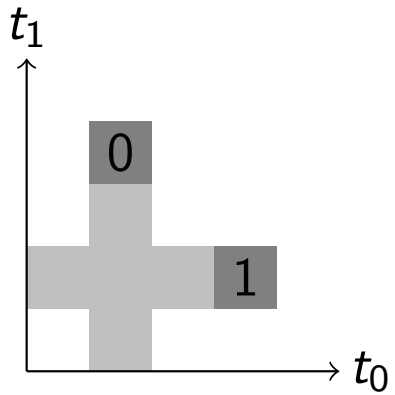
$$\begin{pmatrix} 1 & 0 \\ 1 & 0 \end{pmatrix} \quad \begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix} \quad \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

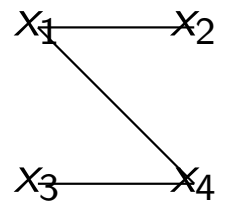
$M_1 \quad M_2 \quad M_3$

and they are all disconnected.
(linked with Herlihy/Rajsbaum protocol complex...)

Just the beginning...on the practical side

- Recent algorithmical advances in algorithmics of hypergraph transversals (although NP-complete...)
  - Link SAT/hypergraph transversals...



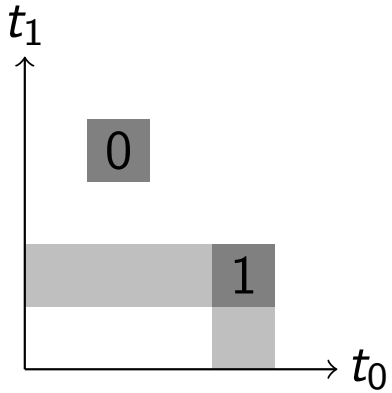


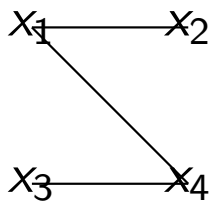
$\neg(x_1 \wedge x_4)$

  - Refined algorithmics
- lots of experimentations to do...

Just the beginning...on the practical side

- Recent algorithmical advances in algorithmics of hypergraph transversals (although NP-complete...)
  - Link SAT/hypergraph transversals...



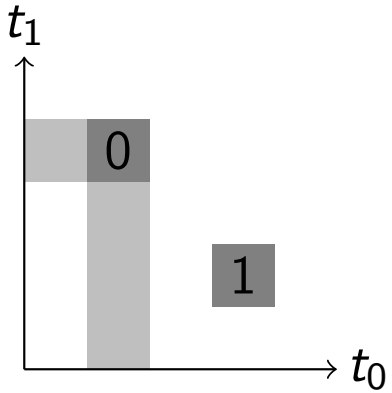


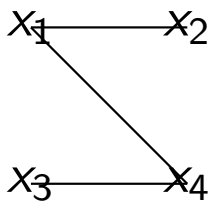
$$(\neg x_1 \vee \neg x_4) \wedge (\neg x_3 \vee \neg x_4)$$

  - Refined algorithmics
- lots of experimentations to do...

Just the beginning...on the practical side

- Recent algorithmical advances in algorithmics of hypergraph transversals (although NP-complete...)
  - Link SAT/hypergraph transversals...



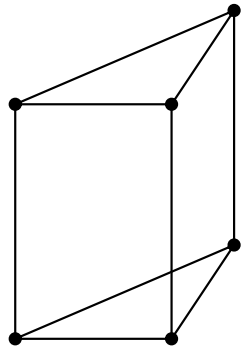


$$(\neg x_1 \vee \neg x_4) \wedge (\neg x_1 \vee \neg x_2) \wedge (\neg x_1 \vee \neg x_2)$$

  - Refined algorithmics
- lots of experimentations to do...

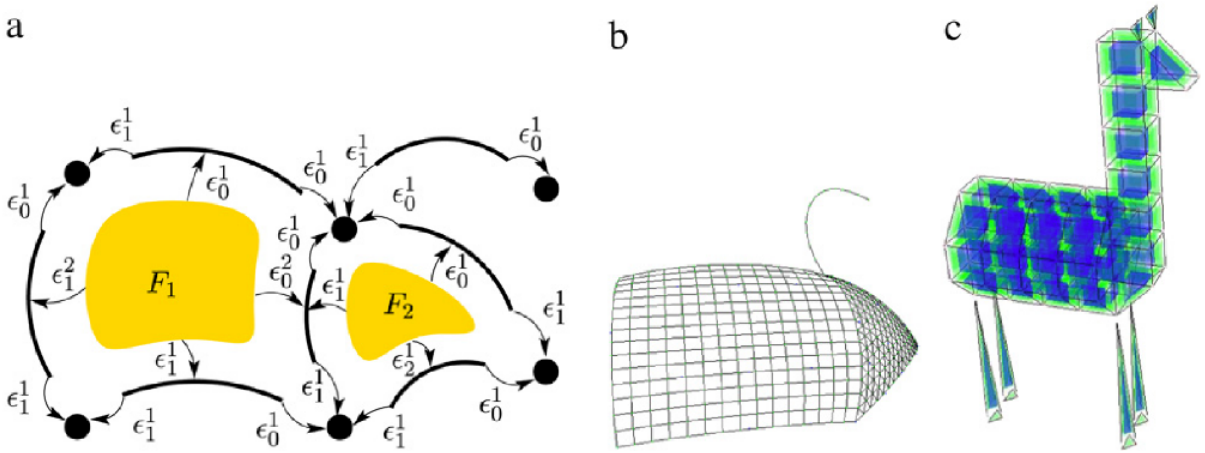
The prodsimplicial structure of trace spaces

- A prod-simplicial space is just a space made up of simplices, and products of simplices, glued together along their faces (natural generalization of cubical and simplicial sets)



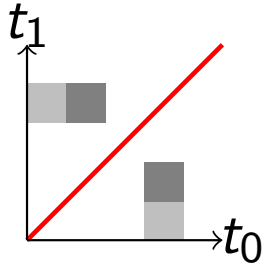
The prodsimplicial structure of trace spaces

- A prod-simplicial space is just a space made up of simplices, and products of simplices, glued together along their faces (natural generalization of cubical and simplicial sets)
- Example:



Each matrix of $\mathcal{C}$ represents a prod-simplex, product of one n-simplex per line, n=number of 1 per line minus 1...

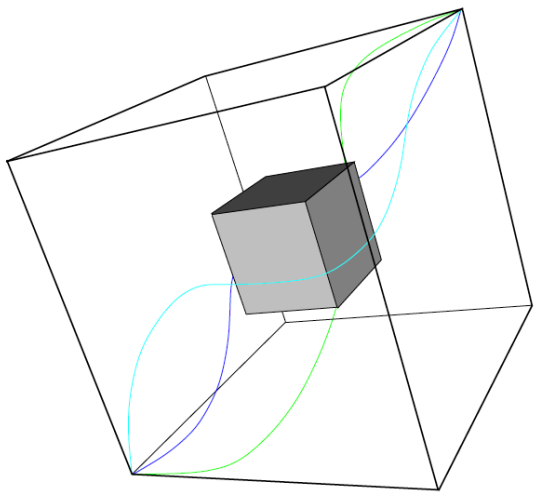
Recall:



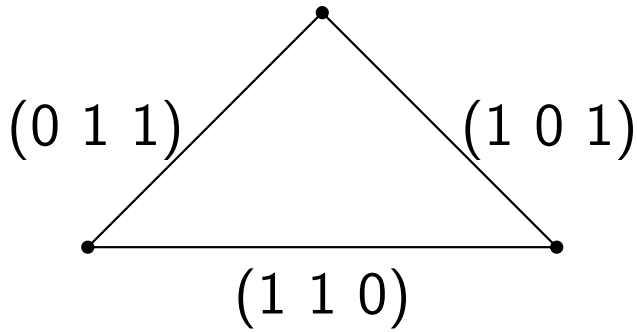
$$M_3 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

product of 2 0-simplices = point!

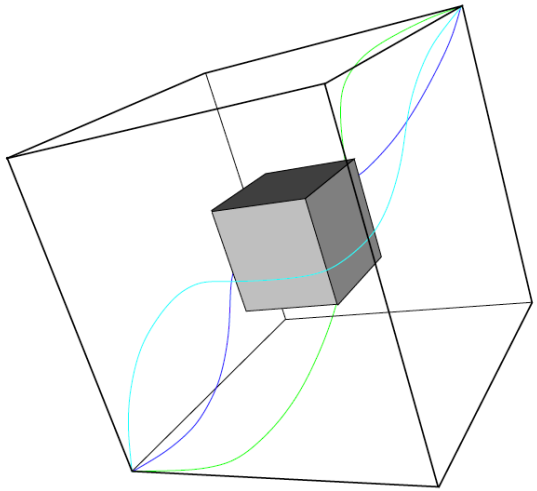
Each matrix of $\mathcal{C}$ represents a prod-simplex, product of one n-simplex per line, n=number of 1 per line minus 1...



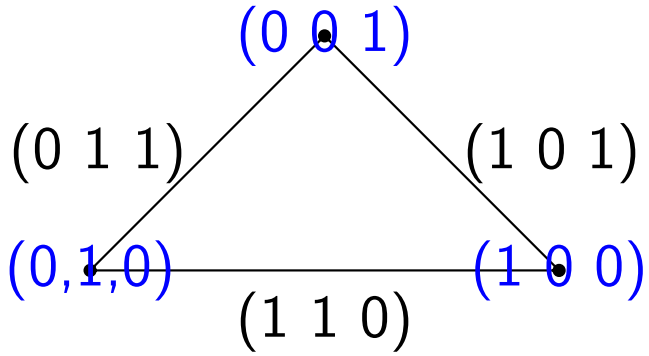
- $\mathcal{D}(X)(0,1) = \{(111)\}$
- $\mathcal{C}(X)(0,1) = \{(110), (101), (011)\}$
-



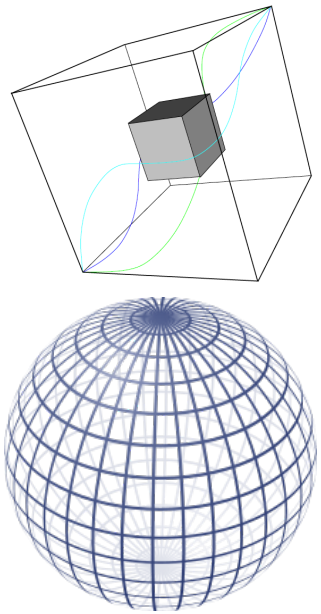
Each matrix of $\mathcal{C}$ represents a prod-simplex, product of one n-simplex per line, n=number of 1 per line minus 1...



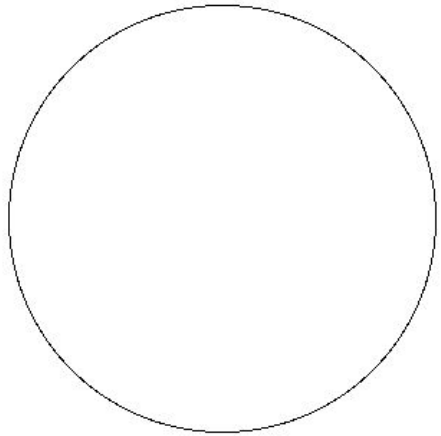
- $\mathcal{C}(X)(0,1) = \{(110), (101), (011)\}$
- and common faces are meet of matrices



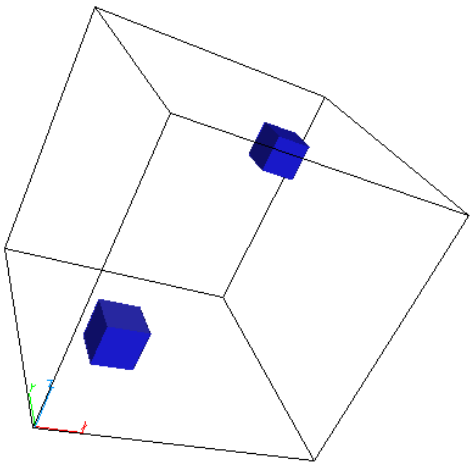
Each matrix of $\mathcal{C}$ represents a prod-simplex, product of one n-simplex per line, n=number of 1 per line minus 1...



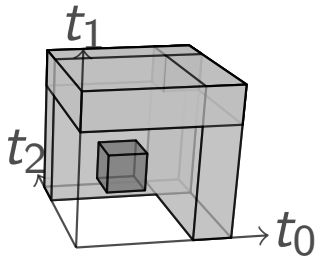
- $\mathcal{C}(X)(0,1) = \{(110),(101),(011)\}$
- connected, not simply-connected (reflecting the fact that $\pi_2(X) = \mathbb{Z}$)



A more intricate example



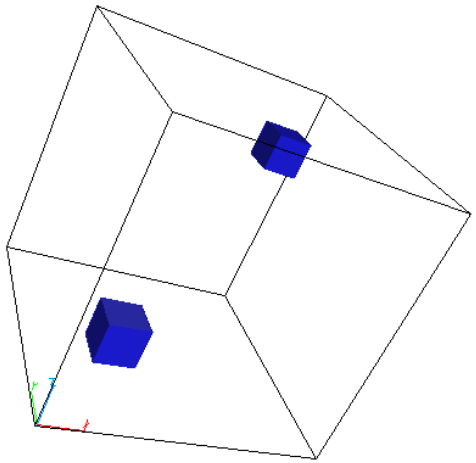
- $\mathcal{D}(X)(0,1) = \left\{ \begin{pmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \end{pmatrix}, \right.$



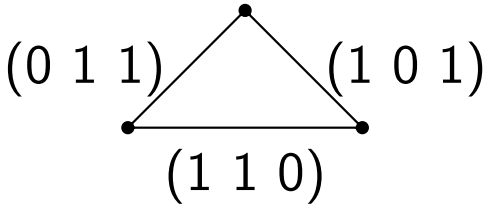
$\left. \begin{pmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \end{pmatrix} \right\}$

- $\mathcal{C}(X)(0,1) =$
$\left\{ \begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \end{pmatrix}, \begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}, \dots \right\}$

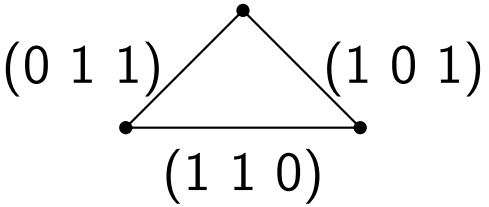
A more intricate example



- $\mathcal{C}(X)(0,1) =$
$$\left\{ \begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \end{pmatrix}, \begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}, \dots \right\}$$

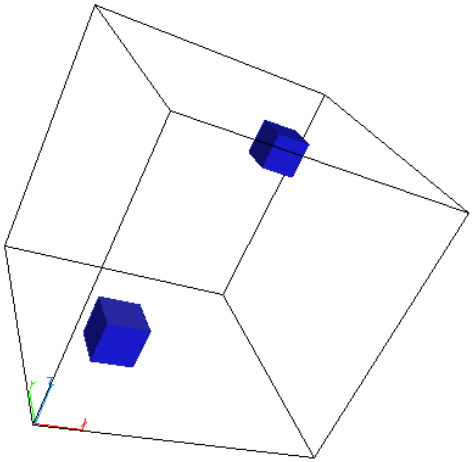


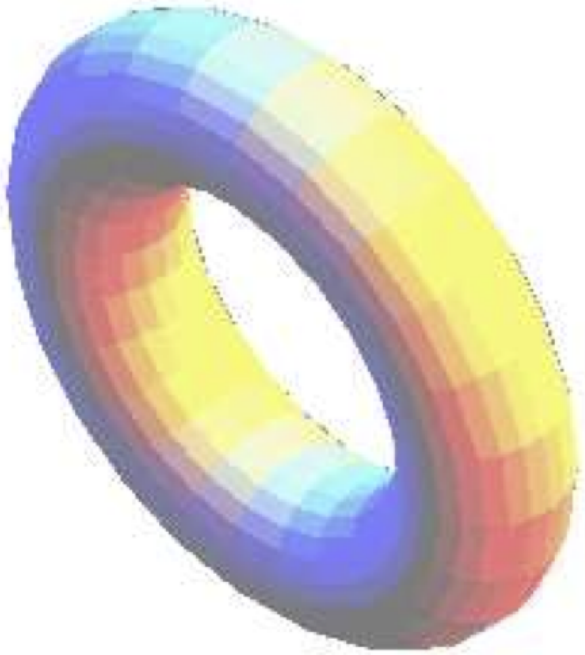
$\times$



A more intricate example

- $\mathcal{C}(X)(0,1) = \left\{ \begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \end{pmatrix}, \begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}, \dots \right\}$





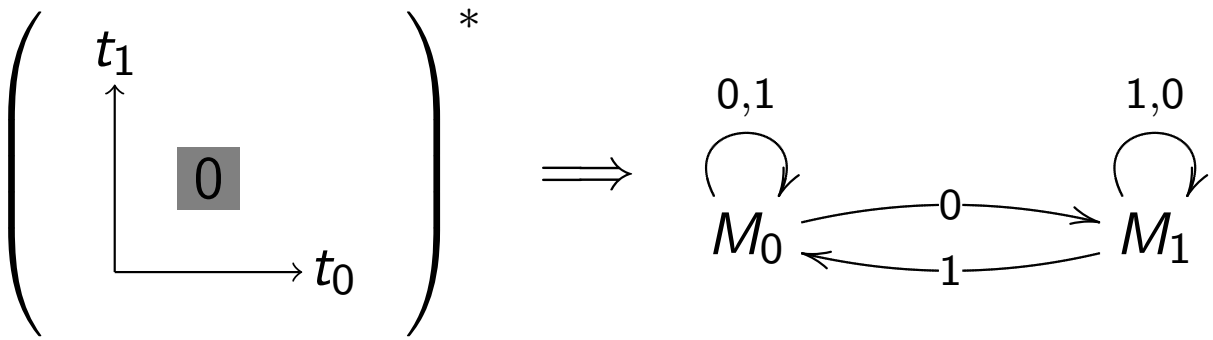
(π_1 is $\mathbb{Z} \times \mathbb{Z}$)

Just the beginning...on the theory side

- The transversal matroid:
  - general notion of dependence
  - $\rightarrow$combinatorics (Tutte polynomial)

Just the beginning...on the theory side

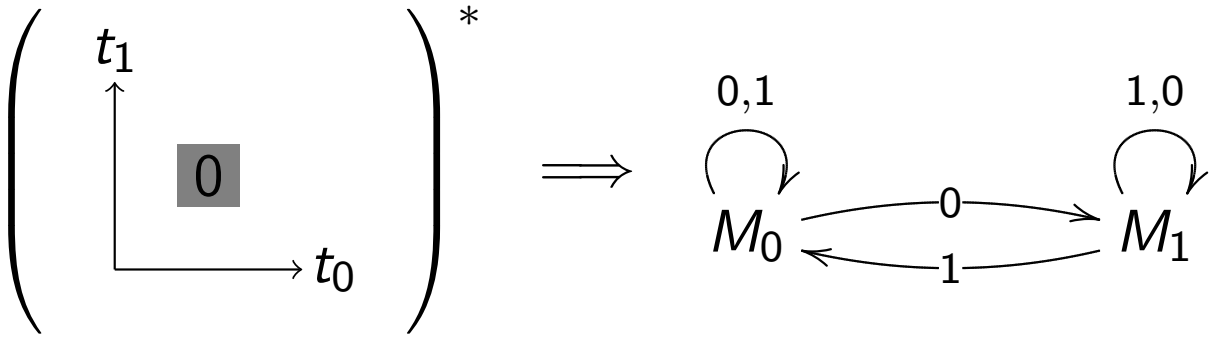
- The transversal matroid:
  - general notion of dependence
  - $\rightarrow$combinatorics (Tutte polynomial)
- Generalization to looping programs (see ESOP 2012)



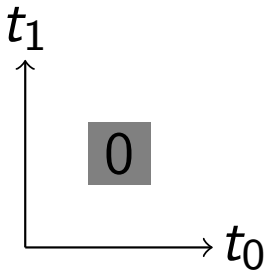
(linked to "generalized CFG" for static analysis)

Just the beginning...on the theory side

- Generalization to looping programs (see ESOP 2012)

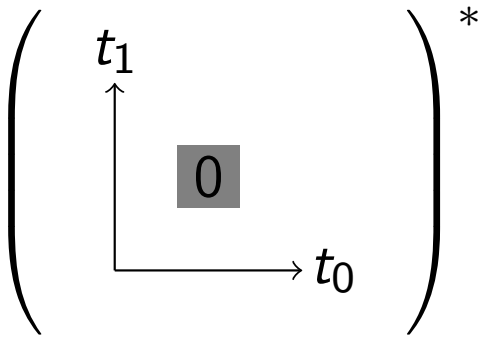


The (1,1)-delooping:

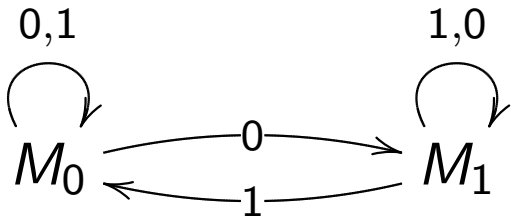


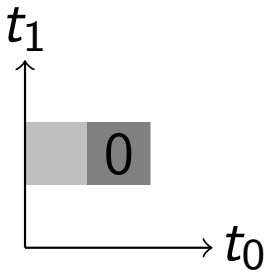
Just the beginning...on the theory side

- Generalization to looping programs (see ESOP 2012)

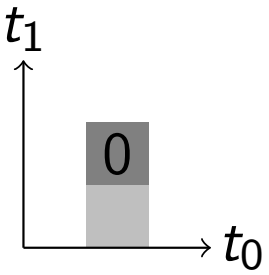


$\Longrightarrow$





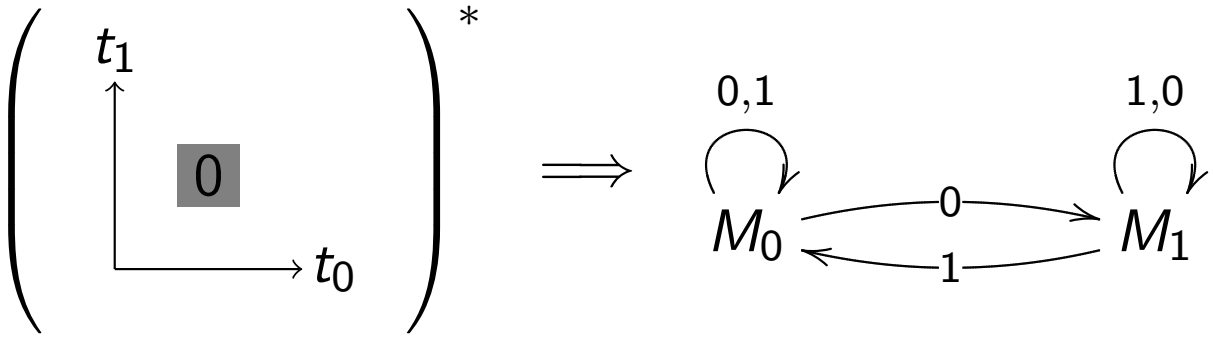
$M_0 = \begin{pmatrix} 0 & 1 \end{pmatrix}$



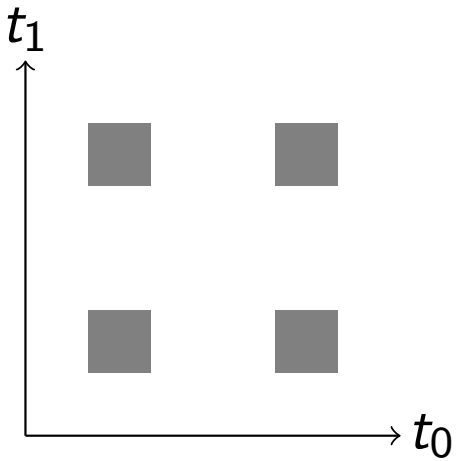
$M_1 = \begin{pmatrix} 1 & 0 \end{pmatrix}$

Just the beginning...on the theory side

- Generalization to looping programs (see ESOP 2012)



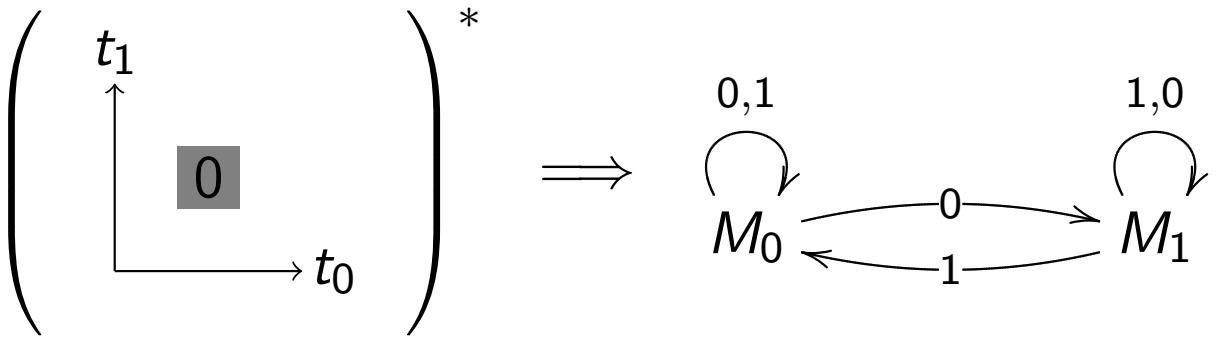
The (2,2)-delooping:



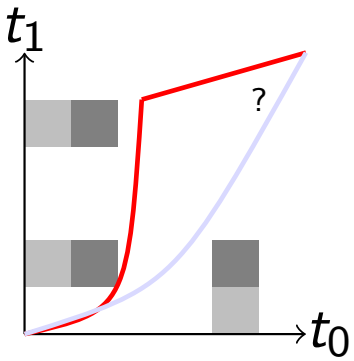
etc.

Just the beginning...on the theory side

- Generalization to looping programs (see ESOP 2012)



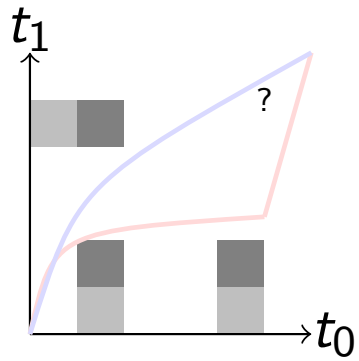
(plus some relations in fact...)



$$\begin{pmatrix} M_0 & ? \\ M_0 & M_1 \end{pmatrix}$$

$$= (M_0 *_1 M_0) *_0 (M_1 *_1 ?)$$

$$(M_0 *_0 M_1) *_1 (M_0 *_0 ?)$$



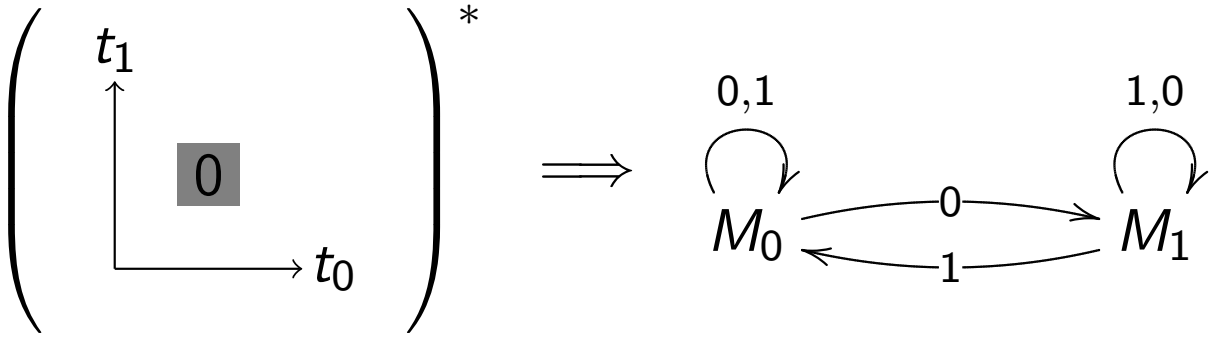
$$\begin{pmatrix} M_0 & ? \\ M_1 & M_1 \end{pmatrix}$$

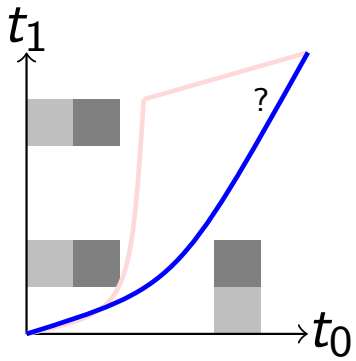
$$(M_1 *_0 M_1) *_1 (M_0 *_0 ?)$$

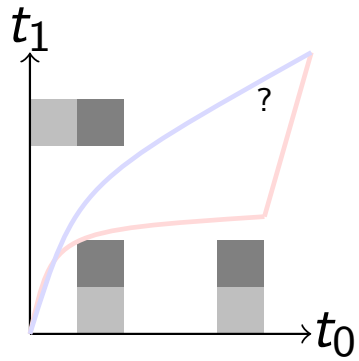
$$= (M_1 *_1 M_0) *_0 (M_1 *_1 ?)$$

Just the beginning...on the theory side

- Generalization to looping programs (see ESOP 2012)







$$\begin{pmatrix} M_0 & ? \\ M_0 & M_1 \end{pmatrix}$$

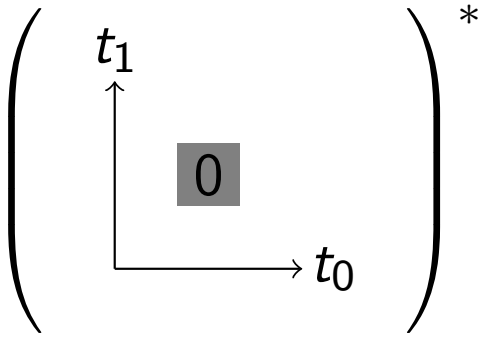
$$= (M_0 *_1 M_0) *_0 (M_1 *_1 ?)$$
$$(M_0 *_0 M_1) *_1 (M_0 *_0 ?)$$

$$\begin{pmatrix} M_0 & ? \\ M_1 & M_1 \end{pmatrix}$$

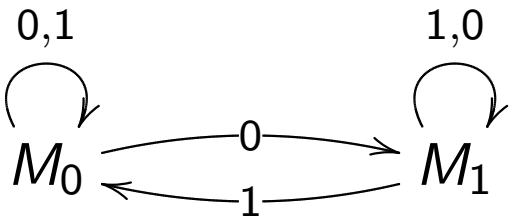
$$(M_1 *_0 M_1) *_1 (M_0 *_0 ?)$$
$$= (M_1 *_1 M_0) *_0 (M_1 *_1 ?)$$

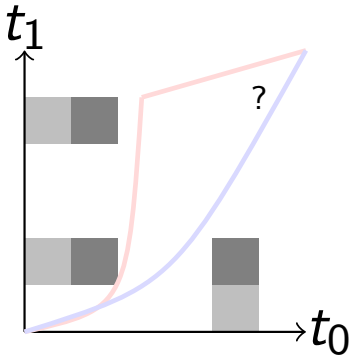
Just the beginning...on the theory side

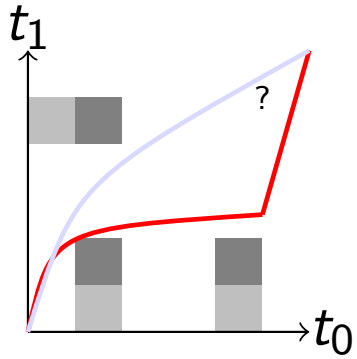
- Generalization to looping programs (see ESOP 2012)



$\Longrightarrow$







$$\begin{pmatrix} M_0 & ? \\ M_0 & M_1 \end{pmatrix}$$

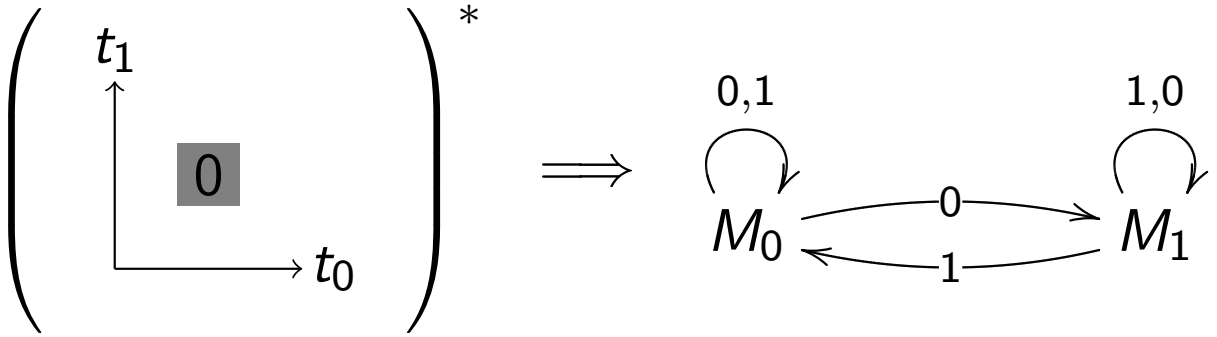
$$= (M_0 *_1 M_0) *_0 (M_1 *_1 ?)$$
$$(M_0 *_0 M_1) *_1 (M_0 *_0 ?)$$

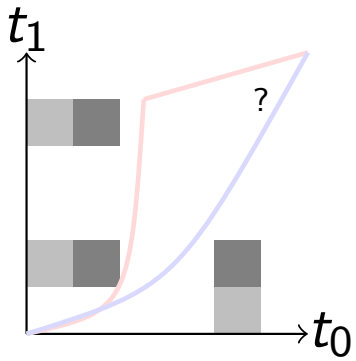
$$\begin{pmatrix} M_0 & ? \\ M_1 & M_1 \end{pmatrix}$$

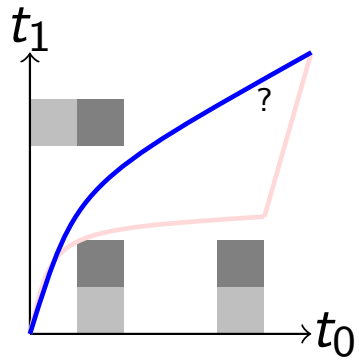
$$(M_1 *_0 M_1) *_1 (M_0 *_0 ?)$$
$$= (M_1 *_1 M_0) *_0 (M_1 *_1 ?)$$

Just the beginning...on the theory side

- Generalization to looping programs (see ESOP 2012)







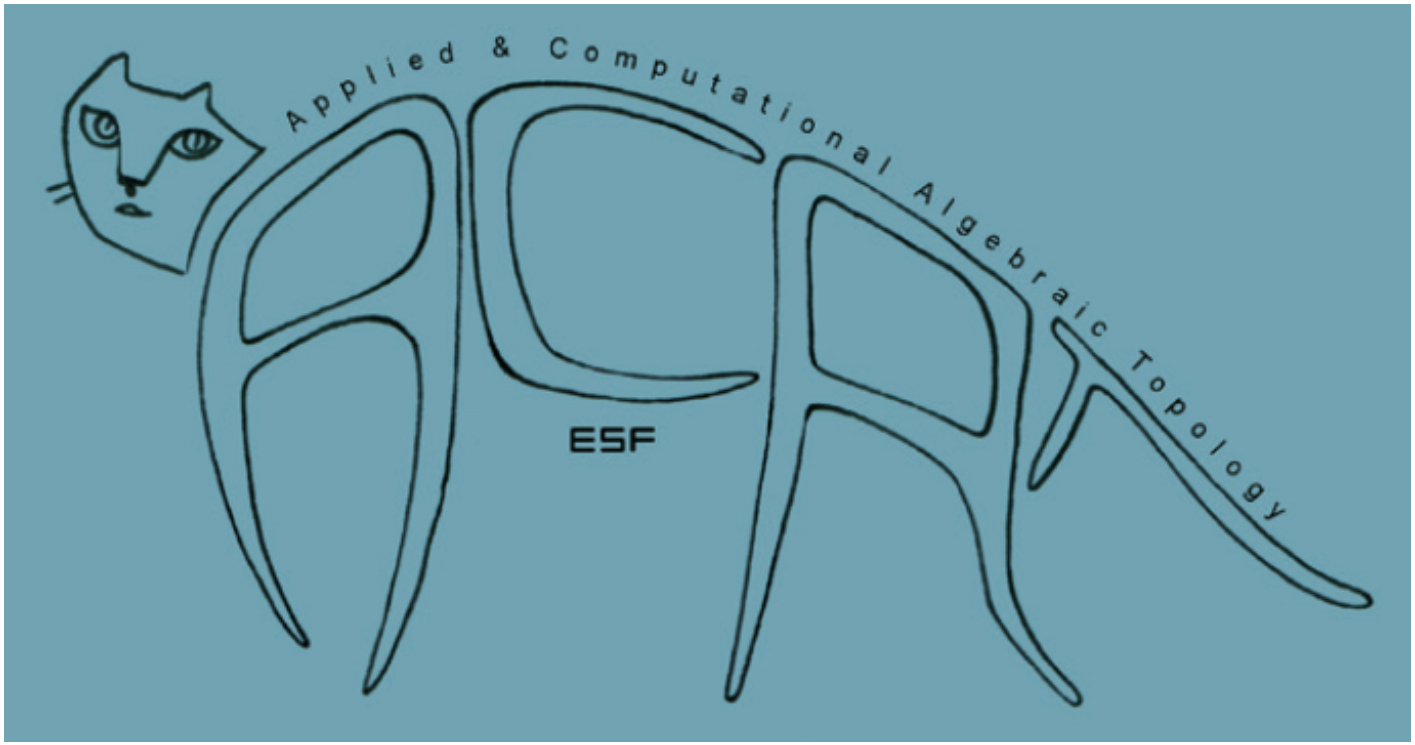
$$\begin{pmatrix} M_0 & ? \\ M_0 & M_1 \end{pmatrix}$$

$$= (M_0 *_1 M_0) *_0 (M_1 *_1 ?)$$
$$(M_0 *_0 M_1) *_1 (M_0 *_0 ?)$$

$$\begin{pmatrix} M_0 & ? \\ M_1 & M_1 \end{pmatrix}$$

$$(M_1 *_0 M_1) *_1 (M_0 *_0 ?)$$
$$= (M_1 *_1 M_0) *_0 (M_1 *_1 ?)$$

Thanks for your attention!



http://acat.lix.polytechnique.fr