Concurrency and directed algebraic topology

Martin Raussen

Department of Mathematical Sciences
Aalborg University
Denmark

Topology of Manifolds and Transformation Groups
Joint Meeting AMS & PTM
Warsaw, 2.8.2007

Outline

1. Motivations, mainly from Concurrency Theory
2. Directed topology: algebraic topology with a twist
3. A categorical framework (with examples)
4. “Compression” of ditopological categories: generalized congruences via homotopy flows

Main Collaborators:

- Lisbeth Fajstrup (Aalborg), Éric Goubault, Emmanuel Haucourt (CEA, France)

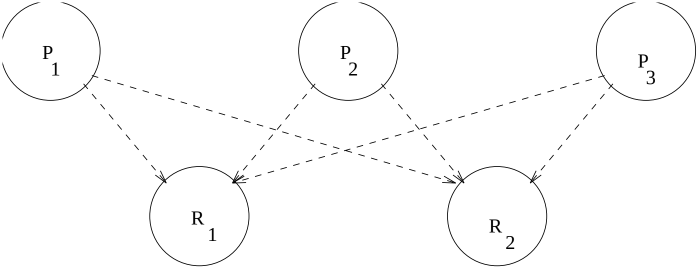
Outline

1. Motivations, mainly from Concurrency Theory
2. Directed topology: algebraic topology with a twist
3. A categorical framework (with examples)
4. “Compression” of ditopological categories: generalized congruences via homotopy flows

Main Collaborators:

- Lisbeth Fajstrup (Aalborg), Éric Goubault, Emmanuel Haucourt (CEA, France)

Mutual exclusion occurs, when n processes P_i compete for m resources R_j.



Only k processes can be served at any given time.

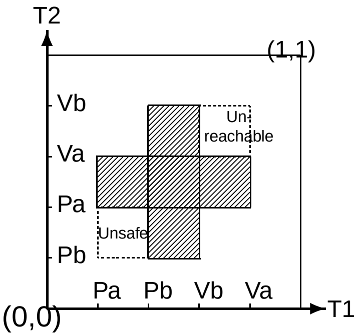
Semaphores!

Semantics: A processor has to lock a resource and relinquish the lock later on!

Description/abstraction $P_i : \dots PR_j \dots VR_j \dots$ (Dijkstra)

Schedules in "progress graphs"

The Swiss flag example



PV-diagram from

$P_1 : P_aP_bV_bV_a$

$P_2 : P_bP_aV_aV_b$

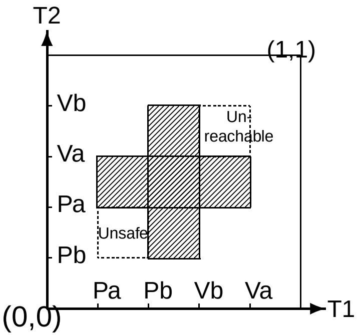
Executions are directed paths – since time flow is irreversible – avoiding a forbidden region (shaded).

Dipaths that are dihomotopic (through a 1-parameter deformation consisting of dipaths) correspond to equivalent executions.

Deadlocks, unsafe and unreachable regions may occur.

Schedules in "progress graphs"

The Swiss flag example

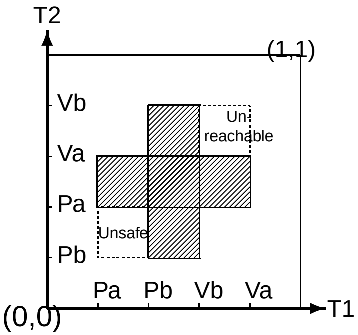


PV-diagram from
$P_1 : P_a P_b V_b V_a$
$P_2 : P_b P_a V_a V_b$

Executions are directed paths – since time flow is irreversible – avoiding a forbidden region (shaded).

Dipaths that are dihomotopic (through a 1-parameter deformation consisting of dipaths) correspond to equivalent executions.

Deadlocks, unsafe and unreachable regions may occur.



PV-diagram from

$P_1 : P_a P_b V_b V_a$

$P_2 : P_b P_a V_a V_b$

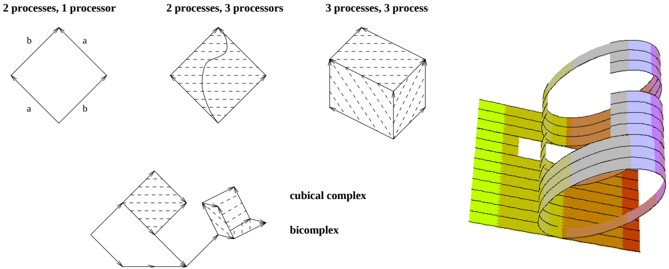
Executions are directed paths – since time flow is irreversible – avoiding a forbidden region (shaded).

Dipaths that are dihomotopic (through a 1-parameter deformation consisting of dipaths) correspond to equivalent executions.

Deadlocks, unsafe and unreachable regions may occur.

Higher dimensional automata
seen as (geometric realizations of) cubical sets

Vaughan Pratt, Rob van Glabbeek, Eric Goubault...



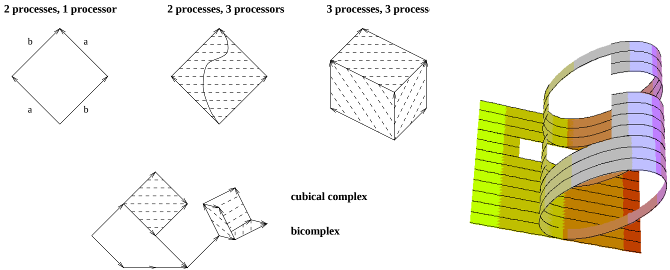
Squares/cubes/hypercubes are filled in iff actions on boundary are independent.

Higher dimensional automata are cubical sets:

- like simplicial sets, but modelled on (hyper)cubes instead of simplices; glueing by face maps (and degeneracies)
- additionally: preferred directions – not all paths allowable.

Higher dimensional automata
seen as (geometric realizations of) cubical sets

Vaughan Pratt, Rob van Glabbeek, Eric Goubault...



Squares/cubes/hypercubes are filled in iff actions on boundary are independent.

Higher dimensional automata are cubical sets:

- like simplicial sets, but modelled on (hyper)cubes instead of simplices; glueing by face maps (and degeneracies)
- additionally: preferred directions – not all paths allowable.

Discrete versus continuous models

How to handle the state-space explosion problem?

The state space explosion problem for discrete models for concurrency (transition graph models): The number of states (and the number of possible schedules) grows exponentially in the number of processors and/or the length of programs.

Need clever ways to find out which of the schedules yield equivalent results for general reasons – e.g., to check for correctness.

Alternative: Infinite continuous models allowing for well-known equivalence relations on paths (homotopy = 1-parameter deformations) – but with an important twist!

Analogy: Continuous physics as an approximation to (discrete) quantum physics.

The state space explosion problem for discrete models for concurrency (transition graph models): The number of states (and the number of possible schedules) grows exponentially in the number of processors and/or the length of programs.

Need clever ways to find out which of the schedules yield equivalent results for general reasons – e.g., to check for correctness.

Alternative: Infinite continuous models allowing for well-known equivalence relations on paths (homotopy = 1-parameter deformations) – but with an important twist!

Analogy: Continuous physics as an approximation to (discrete) quantum physics.

X a topological space. $\vec{P}(X) \subseteq X^I = \{p : I = [0,1] \to X \text{ cont.}\}$ a set of d-paths ("directed" paths $\leftrightarrow$ executions) satisfying

- { constant paths } $\subseteq \vec{P}(X)$
- $\varphi \in \vec{P}(X)(x,y), \psi \in \vec{P}(X)(y,z) \Rightarrow \varphi * \psi \in \vec{P}(X)(x,z)$
- $\varphi \in \vec{P}(X), \alpha \in I^I$ a nondecreasing reparametrization $\Rightarrow \varphi \circ \alpha \in \vec{P}(X)$

The pair $(X, \vec{P}(X))$ is called a d-space.

Observe: $\vec{P}(X)$ is in general not closed under reversal:

$$\alpha(t) = 1 - t,\ \varphi \in \vec{P}(X) \not\Rightarrow \varphi \circ \alpha \in \vec{P}(X)!$$

Examples:

- An HDA with directed execution paths.
- A space-time(relativity) with time-like or causal curves.

X a topological space. $\vec{P}(X) \subseteq X^I = \{p : I = [0,1] \to X \text{ cont.}\}$
a set of d-paths ("directed" paths $\leftrightarrow$ executions) satisfying

- $\{$ constant paths $\} \subseteq \vec{P}(X)$
- $\varphi \in \vec{P}(X)(x,y), \psi \in \vec{P}(X)(y,z) \Rightarrow \varphi * \psi \in \vec{P}(X)(x,z)$
- $\varphi \in \vec{P}(X), \alpha \in I^I$ a nondecreasing reparametrization
$\Rightarrow \varphi \circ \alpha \in \vec{P}(X)$

The pair $(X, \vec{P}(X))$ is called a d-space.
Observe: $\vec{P}(X)$ is in general not closed under reversal:

$$\alpha(t) = 1 - t,\ \varphi \in \vec{P}(X) \not\Rightarrow \varphi \circ \alpha \in \vec{P}(X)!$$

Examples:

- An HDA with directed execution paths.
- A space-time(relativity) with time-like or causal curves.

X a topological space. $\vec{P}(X) \subseteq X^I = \{p : I = [0,1] \to X \text{ cont.}\}$
a set of d-paths ("directed" paths $\leftrightarrow$ executions) satisfying

- $\{$ constant paths $\} \subseteq \vec{P}(X)$
- $\varphi \in \vec{P}(X)(x,y), \psi \in \vec{P}(X)(y,z) \Rightarrow \varphi * \psi \in \vec{P}(X)(x,z)$
- $\varphi \in \vec{P}(X), \alpha \in I^I$ a nondecreasing reparametrization $\Rightarrow \varphi \circ \alpha \in \vec{P}(X)$

The pair $(X, \vec{P}(X))$ is called a d-space.
Observe: $\vec{P}(X)$ is in general not closed under reversal:

$$\alpha(t) = 1 - t,\ \varphi \in \vec{P}(X) \not\Rightarrow \varphi \circ \alpha \in \vec{P}(X)!$$

Examples:

- An HDA with directed execution paths.
- A space-time(relativity) with time-like or causal curves.

D-maps, Dihomotopy, d-homotopy

A d-map $f : X \to Y$ is a continuous map satisfying

- $f(\vec{P}(X)) \subseteq \vec{P}(Y)$

special case: $\vec{P}(I) = \{\sigma \in I^I | \sigma$ nondecreasing reparametrization$\}$, $\vec{I} = (I, \vec{P}(I))$.
Then $\vec{P}(X) =$ set of d-maps from $\vec{I}$ to X.

- Dihomotopy $H : X \times I \to Y$, every H_t a d-map
- elementary d-homotopy = d-map $H : X \times \vec{I} \to Y$ – $H_0 = f \xrightarrow{H} g = H_1$
- d-homotopy: symmetric and transitive closure ("zig-zag")

L. Fajstrup, 05: In cubical models (for concurrency, e.g., HDAs), the two notions agree for d-paths ($X = \vec{I}$). In general, they do not.

D-maps, Dihomotopy, d-homotopy

A d-map $f : X \to Y$ is a continuous map satisfying

- $f(\vec{P}(X)) \subseteq \vec{P}(Y)$

special case: $\vec{P}(I) = \{\sigma \in I^I | \sigma$ nondecreasing reparametrization$\}$, $\vec{I} = (I, \vec{P}(I))$.
Then $\vec{P}(X) =$ set of d-maps from $\vec{I}$ to X.

- Dihomotopy $H : X \times I \to Y$, every H_t a d-map
- elementary d-homotopy = d-map $H : X \times \vec{I} \to Y$ – $H_0 = f \xrightarrow{H} g = H_1$
- d-homotopy: symmetric and transitive closure ("zig-zag")

L. Fajstrup, 05: In cubical models (for concurrency, e.g., HDAs), the two notions agree for d-paths ($X = \vec{I}$). In general, they do not.

D-maps, Dihomotopy, d-homotopy

A d-map $f : X \to Y$ is a continuous map satisfying

- $f(\vec{P}(X)) \subseteq \vec{P}(Y)$

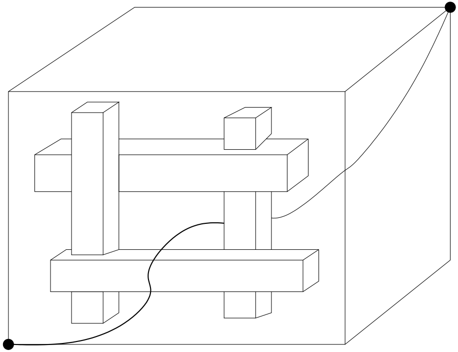
special case: $\vec{P}(I) = \{\sigma \in I^I | \sigma$ nondecreasing reparametrization$\}$, $\vec{I} = (I, \vec{P}(I))$.
Then $\vec{P}(X) =$ set of d-maps from $\vec{I}$ to X.

- Dihomotopy $H : X \times I \to Y$, every H_t a d-map
- elementary d-homotopy = d-map $H : X \times \vec{I} \to Y$ – $H_0 = f \xrightarrow{H} g = H_1$
- d-homotopy: symmetric and transitive closure (”zig-zag”)

L. Fajstrup, 05: In cubical models (for concurrency, e.g., HDAs), the two notions agree for d-paths ($X = \vec{I}$). In general, they do not.

Dihomotopy is finer than homotopy with fixed endpoints

Example: Two wedges in the forbidden region

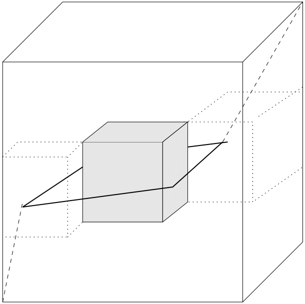


All dipaths from minimum to maximum are homotopic.
A dipath through the "hole" is not dihomotopic to a dipath on the boundary.

The twist has a price

Neither homogeneity nor cancellation nor group structure

Ordinary topology: Path space = loop space (within each path component).
A loop space is an H-space with concatenation, inversion, cancellation.



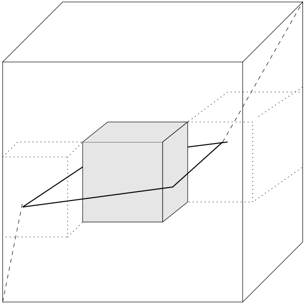
"Birth and death" of dihomotopy classes

Directed topology: Loops do not tell much; concatenation ok, cancellation not! Replace group structure by category structures!

The twist has a price

Neither homogeneity nor cancellation nor group structure

Ordinary topology: Path space = loop space (within each path component).
A loop space is an H-space with concatenation, inversion, cancellation.



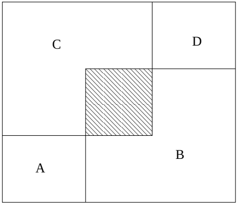
"Birth and death" of dihomotopy classes

Directed topology: Loops do not tell much; concatenation ok, cancellation not!
Replace group structure by category structures!

A first remedy: the fundamental category

$\vec{\pi}_1(X)$ of a d-space X [Grandis:03, FGHR:04]:

- Objects: points in X
- Morphisms: d- or dihomotopy classes of d-paths in X
- Composition: from concatenation of d-paths

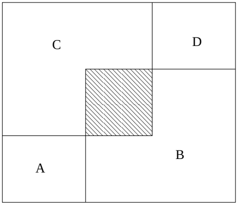


Property: van Kampen theorem (M. Grandis)
Drawbacks: Infinitely many objects. Calculations?
Question: How much does $\vec{\pi}_1(X)(x,y)$ depend on (x,y)?
Remedy: Localization, component category. [FGHR:04, GH:06]
Problem: This "compression" works only for loopfree categories

A first remedy: the fundamental category

$\vec{\pi}_1(X)$ of a d-space X [Grandis:03, FGHR:04]:

- Objects: points in X
- Morphisms: d- or dihomotopy classes of d-paths in X
- Composition: from concatenation of d-paths



Property: van Kampen theorem (M. Grandis)
Drawbacks: Infinitely many objects. Calculations?
Question: How much does $\vec{\pi}_1(X)(x,y)$ depend on (x,y)?
Remedy: Localization, component category. [FGHR:04, GH:06]
Problem: This "compression" works only for loopfree categories

X a (saturated) d-space.
$\varphi, \psi \in \vec{P}(X)(x,y)$ are called reparametrization equivalent if there are $\alpha, \beta \in \vec{P}(I)$ such that $\varphi \circ \alpha = \psi \circ \beta$ ("same oriented trace").
(Fahrenberg-R., 07): Reparametrization equivalence is an equivalence relation (transitivity).
$\vec{T}(X)(x,y) = \vec{P}(X)(x,y)/_{\simeq}$ makes $\vec{T}(X)$ into the (topologically enriched) trace category – composition associative.
A d-map $f : X \to Y$ induces a functor $\vec{T}(f) : \vec{T}(X) \to \vec{T}(Y)$.

X a (saturated) d-space.
$\varphi, \psi \in \vec{P}(X)(x,y)$ are called reparametrization equivalent if there are $\alpha, \beta \in \vec{P}(I)$ such that $\varphi \circ \alpha = \psi \circ \beta$ ("same oriented trace").
(Fahrenberg-R., 07): Reparametrization equivalence is an equivalence relation (transitivity).
$\vec{T}(X)(x,y) = \vec{P}(X)(x,y)/_{\simeq}$ makes $\vec{T}(X)$ into the (topologically enriched) trace category – composition associative.
A d-map $f : X \to Y$ induces a functor $\vec{T}(f) : \vec{T}(X) \to \vec{T}(Y)$.

Variant: $\vec{R}(X)(x,y)$ consists of regular d-paths (not constant on any non-trivial interval $J \subset I$). The contractible group $Homeo_+(I)$ of increasing homeomorphisms acts on these – freely if $x \neq y$.

Theorem (FR:07)

- *$\vec{R}(X)(x,y)/_{\simeq} \to \vec{P}(X)(x,y)/_{\simeq}$ is a homeomorphism.*
- *$\vec{R}(X)(x,y) \to \vec{R}(X)(x,y)/_{\simeq}$ is a (weak) homotopy equivalence.*

For X the geometric realisation of a cubical complex, all trace spaces $\vec{T}(X)(x,y)$ are locally contractible.

Examples I^n the unit cube, ∂I^n its boundary.

- $\vec{T}(I^n;\mathbf{x},\mathbf{y})$ is contractible for all $x \preceq y \in I^n$;
- (Conjecture) $\vec{T}(\partial I^n;\mathbf{0},\mathbf{1})$ is (weakly) homotopy equivalent to S^{n-2}.

Variant: $\vec{R}(X)(x,y)$ consists of regular d-paths (not constant on any non-trivial interval $J \subset I$). The contractible group $Homeo_+(I)$ of increasing homeomorphisms acts on these – freely if $x \neq y$.

Theorem (FR:07)

- *$\vec{R}(X)(x,y)/_{\simeq} \to \vec{P}(X)(x,y)/_{\simeq}$ is a homeomorphism.*
- *$\vec{R}(X)(x,y) \to \vec{R}(X)(x,y)/_{\simeq}$ is a (weak) homotopy equivalence.*

For X the geometric realisation of a cubical complex, all trace spaces $\vec{T}(X)(x,y)$ are locally contractible.

Examples I^n the unit cube, ∂I^n its boundary.

- $\vec{T}(I^n; \mathbf{x}, \mathbf{y})$ is contractible for all $x \preceq y \in I^n$;
- (Conjecture) $\vec{T}(\partial I^n; \mathbf{0}, \mathbf{1})$ is (weakly) homotopy equivalent to S^{n-2}.

Variant: $\vec{R}(X)(x,y)$ consists of regular d-paths (not constant on any non-trivial interval $J \subset I$). The contractible group $Homeo_+(I)$ of increasing homeomorphisms acts on these – freely if $x \neq y$.

Theorem (FR:07)

- *$\vec{R}(X)(x,y)/_{\simeq} \to \vec{P}(X)(x,y)/_{\simeq}$ is a homeomorphism.*
- *$\vec{R}(X)(x,y) \to \vec{R}(X)(x,y)/_{\simeq}$ is a (weak) homotopy equivalence.*

For X the geometric realisation of a cubical complex, all trace spaces $\vec{T}(X)(x,y)$ are locally contractible.

Examples I^n the unit cube, ∂I^n its boundary.

- $\vec{T}(I^n;\mathbf{x},\mathbf{y})$ is contractible for all $x \preceq y \in I^n$;
- (Conjecture) $\vec{T}(\partial I^n;\mathbf{0},\mathbf{1})$ is (weakly) homotopy equivalent to S^{n-2}.

Preorder categories
Getting organised with indexing categories

A d-space structure on X induces the preorder $\preceq$:

$$x \preceq y \Leftrightarrow \vec{T}(X)(x,y) \neq \emptyset$$

and an indexing preorder category $\vec{D}(X)$ with

- Objects: (end point) pairs $(x,y), x \preceq y$
- Morphisms:
  $\vec{D}(X)((x,y),(x',y')) := \vec{T}(X)(x',x) \times \vec{T}(X)(y,y')$:

$$\begin{tikzcd} x' \arrow[r, shift left] \arrow[r, shift right] & x \arrow[r, "\preceq"] & y \arrow[r, shift left] \arrow[r, shift right] & y' \end{tikzcd}$$

- Composition: by pairwise contra-, resp. covariant concatenation.

A d-map $f : X \to Y$ induces a functor $\vec{D}(f) : \vec{D}(X) \to \vec{D}(Y)$.

A d-space structure on X induces the preorder $\preceq$:

$$x \preceq y \Leftrightarrow \vec{T}(X)(x,y) \neq \emptyset$$

and an indexing preorder category $\vec{D}(X)$ with

- Objects: (end point) pairs $(x,y), x \preceq y$
- Morphisms:
  $\vec{D}(X)((x,y),(x',y')) := \vec{T}(X)(x',x) \times \vec{T}(X)(y,y')$:

$$\begin{tikzcd} x' \arrow[r, bend left] \arrow[r, bend right] & x \arrow[r, "\preceq"] & y \arrow[r, bend left] \arrow[r, bend right] & y' \end{tikzcd}$$

- Composition: by pairwise contra-, resp. covariant concatenation.

A d-map $f : X \to Y$ induces a functor $\vec{D}(f) : \vec{D}(X) \to \vec{D}(Y)$.

Preorder categories

Getting organised with indexing categories

A d-space structure on X induces the preorder $\preceq$:

$$x \preceq y \Leftrightarrow \vec{T}(X)(x,y) \neq \emptyset$$

and an indexing preorder category $\vec{D}(X)$ with

- Objects: (end point) pairs $(x,y), x \preceq y$
- Morphisms:
  $\vec{D}(X)((x,y),(x',y')) := \vec{T}(X)(x',x) \times \vec{T}(X)(y,y')$:

$$\begin{tikzcd} x' \arrow[r, bend left] \arrow[r, bend right] & x \arrow[r, "\preceq"] & y \arrow[r, bend left] \arrow[r, bend right] & y' \end{tikzcd}$$

- Composition: by pairwise contra-, resp. covariant concatenation.

A d-map $f : X \to Y$ induces a functor $\vec{D}(f) : \vec{D}(X) \to \vec{D}(Y)$.

The preorder category organises X via the
trace space functor $\vec{T}^X : \vec{D}(X) \to Top$

- $\vec{T}^X(x,y) := \vec{T}(X)(x,y)$
- $\vec{T}^X(\sigma_x,\sigma_y)$: $\vec{T}(X)(x,y) \longrightarrow \vec{T}(X)(x',y')$

$$[\sigma] \longmapsto [\sigma_x * \sigma * \sigma_y]$$

Homotopical variant $\vec{D}_\pi(X)$ with morphisms
$\vec{D}_\pi(X)((x,y),(x',y')) := \vec{\pi}_1(X)(x',x) \times \vec{\pi}_1(X)(y,y')$
and trace space functor $\vec{T}^X_\pi : \vec{D}_\pi(X) \to Ho-Top$ (with homotopy classes as morphisms).

The preorder category organises X via the trace space functor $\vec{T}^X : \vec{D}(X) \to Top$

- $\vec{T}^X(x,y) := \vec{T}(X)(x,y)$
- $\vec{T}^X(\sigma_x,\sigma_y)$:

$$\begin{array}{ccc} \vec{T}(X)(x,y) & \longrightarrow & \vec{T}(X)(x',y') \\ [\sigma] & \longmapsto & [\sigma_x * \sigma * \sigma_y] \end{array}$$

Homotopical variant $\vec{D}_\pi(X)$ with morphisms $\vec{D}_\pi(X)((x,y),(x',y')) := \vec{\pi}_1(X)(x',x) \times \vec{\pi}_1(X)(y,y')$ and trace space functor $\vec{T}^X_\pi : \vec{D}_\pi(X) \to Ho-Top$ (with homotopy classes as morphisms).

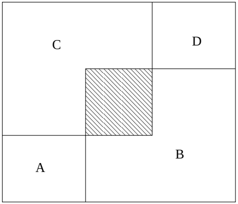
Sensitivity with respect to variations of end points
A persistence point of view

- How much does (the homotopy type of) $\vec{T}^X(x,y)$ depend on (small) changes of x,y?
- Which concatenation maps $\vec{T}^X(\sigma_x,\sigma_y):\vec{T}^X(x,y)\to\vec{T}^X(x',y'),\ [\sigma]\mapsto[\sigma_x*\sigma*\sigma_y]$ are homotopy equivalences, induce isos on homotopy, homology groups etc.?
- The persistence point of view: Homology classes etc. are born (at certain branchings/mergings) and may die (analogous to the framework of G. Carlsson etal.)
- Are there "components" with (homotopically/homologically) stable dipath spaces (between them)? Are there borders ("walls") at which changes occur?

- How much does (the homotopy type of) $\vec{T}^X(x,y)$ depend on (small) changes of x,y?
- Which concatenation maps $\vec{T}^X(\sigma_x,\sigma_y):\vec{T}^X(x,y)\to\vec{T}^X(x',y'),\ [\sigma]\mapsto[\sigma_x*\sigma*\sigma_y]$ are homotopy equivalences, induce isos on homotopy, homology groups etc.?
- The persistence point of view: Homology classes etc. are born (at certain branchings/mergings) and may die (analogous to the framework of G. Carlsson etal.)
- Are there “components” with (homotopically/homologically) stable dipath spaces (between them)? Are there borders (“walls”) at which changes occur?

- How much does (the homotopy type of) $\vec{T}^X(x,y)$ depend on (small) changes of x,y?
- Which concatenation maps $\vec{T}^X(\sigma_x,\sigma_y):\vec{T}^X(x,y)\to\vec{T}^X(x',y'),\ [\sigma]\mapsto[\sigma_x*\sigma*\sigma_y]$ are homotopy equivalences, induce isos on homotopy, homology groups etc.?
- The persistence point of view: Homology classes etc. are born (at certain branchings/mergings) and may die (analogous to the framework of G. Carlsson etal.)
- Are there "components" with (homotopically/homologically) stable dipath spaces (between them)? Are there borders ("walls") at which changes occur?

- How much does (the homotopy type of) $\vec{T}^X(x,y)$ depend on (small) changes of x,y?
- Which concatenation maps $\vec{T}^X(\sigma_x,\sigma_y) : \vec{T}^X(x,y) \to \vec{T}^X(x',y'),\ [\sigma] \mapsto [\sigma_x * \sigma * \sigma_y]$ are homotopy equivalences, induce isos on homotopy, homology groups etc.?
- The persistence point of view: Homology classes etc. are born (at certain branchings/mergings) and may die (analogous to the framework of G. Carlsson etal.)
- Are there "components" with (homotopically/homologically) stable dipath spaces (between them)? Are there borders ("walls") at which changes occur?

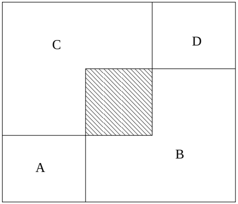


$$
\begin{tikzcd}
AA \arrow[rrd] \arrow[rdd] & & & & BB \arrow[lld] \arrow[ldd] \\
& & AB \arrow[d] & \# & \\
& AC \arrow[r] & AD & BD \arrow[l] & \\
& \# & CD \arrow[u] & & \\
CC \arrow[ruu] \arrow[rru] & & & & DD \arrow[llu] \arrow[luu]
\end{tikzcd}
$$

Figur: Standard example and component category

Components A, B, C, D – or rather $AA, AB, AC, AD, BB, BD, CC, CD, DD \subseteq X \times X$.

#: diagram commutes.



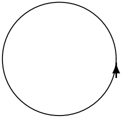
$$
\begin{tikzcd}
AA \arrow[drr] \arrow[ddr] & & & & BB \arrow[dll] \arrow[ddl] \\
& & AB \arrow[d] & \# & \\
& AC \arrow[r] & AD & BD \arrow[l] & \\
& \# & CD \arrow[u] & & \\
CC \arrow[uur] \arrow[urr] & & & & DD \arrow[ull] \arrow[uul]
\end{tikzcd}
$$

Figur: Standard example and component category

Components A, B, C, D – or rather $AA, AB, AC, AD, BB, BD, CC, CD, DD \subseteq X \times X$.

#: diagram commutes.

$X \quad = \quad \vec{S}^1$



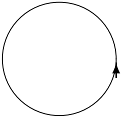
$$\mathcal{C} : \Delta \underset{b}{\overset{a}{\rightleftarrows}} \bar{\Delta}$$

Δ the diagonal, $\bar{\Delta}$ its complement. $\mathcal{C}$ is the free category generated by a, b.

oriented circle

- Remark that the components are no longer products!
- In order to get a discrete component category, it is essential to use an indexing category taking care of pairs (source, target).

$X \quad = \quad \vec{S}^1$



oriented circle

$$\mathcal{C}: \Delta \underset{b}{\overset{a}{\rightleftarrows}} \bar{\Delta}$$

Δ the diagonal, $\bar{\Delta}$ its complement. $\mathcal{C}$ is the free category generated by a, b.

- Remark that the components are no longer products!
- In order to get a discrete component category, it is essential to use an indexing category taking care of pairs (source, target).

Component categories
via generalized congruences and homotopy flows

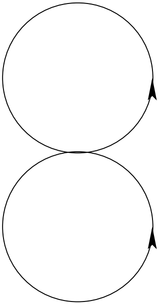
- How to identify morphisms in a category between different objects in an organised manner?
  Generalized congruence (Bednarczyk, Borzyszkowski, Pawlowski, TAC 1999) $\rightsquigarrow$ quotient category.
- Homotopy flows identify both elements and d-paths: Like flows in differential geometry. Instead of diffeotopies: Self-homotopies inducing homotopy equivalences on spaces of d-paths with given end points ("automorphic").
- Homotopy flows give rise to significant generalized congruences. Corresponding component category $\vec{D}_\pi(X)/\simeq$ identifies pairs of points on the same "homotopy flow line" and (chains of) morphisms.

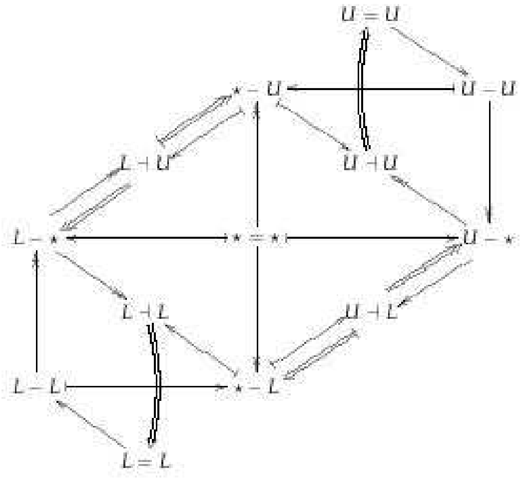
- How to identify morphisms in a category between different objects in an organised manner?
  Generalized congruence (Bednarczyk, Borzyszkowski, Pawlowski, TAC 1999) $\rightsquigarrow$ quotient category.
- Homotopy flows identify both elements and d-paths: Like flows in differential geometry. Instead of diffeotopies: Self-homotopies inducing homotopy equivalences on spaces of d-paths with given end points (“automorphic”).
- Homotopy flows give rise to significant generalized congruences. Corresponding component category $\vec{D}_\pi(X)/\simeq$ identifies pairs of points on the same “homotopy flow line” and (chains of) morphisms.

- How to identify morphisms in a category between different objects in an organised manner?
  Generalized congruence (Bednarczyk, Borzyszkowski, Pawlowski, TAC 1999) $\rightsquigarrow$ quotient category.
- Homotopy flows identify both elements and d-paths: Like flows in differential geometry. Instead of diffeotopies: Self-homotopies inducing homotopy equivalences on spaces of d-paths with given end points (“automorphic”).
- Homotopy flows give rise to significant generalized congruences. Corresponding component category $\vec{D}_\pi(X)/\simeq$ identifies pairs of points on the same “homotopy flow line” and (chains of) morphisms.

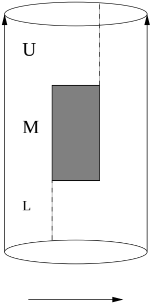
The component category of a wedge of two oriented circles

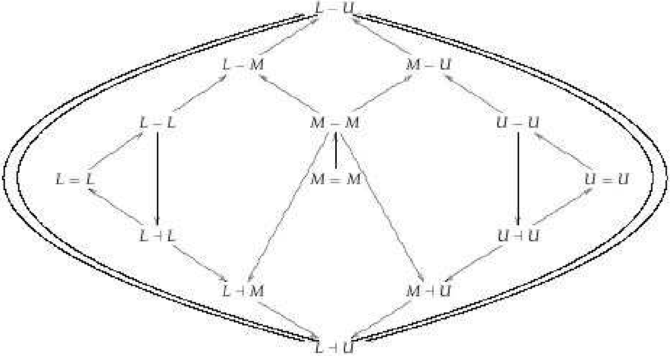
$X = \vec{S}^1 \vee \vec{S}^1$





The component category of an oriented cylinder with a deleted rectangle





Concluding remarks

- Component categories contain the essential information given by (algebraic topological invariants of) path spaces
- Compression via component categories is an antidote to the state space explosion problem
- Some of the ideas (for the fundamental category) are implemented and have been tested for huge industrial software from EDF (Éric Goubault & Co., CEA)
- Dihomotopy equivalence: Definition uses automorphic homotopy flows to ensure homotopy equivalences

$$\vec{T}(f)(x,y) : \vec{T}(X)(x,y) \to \vec{T}(Y)(fx,fy) \text{ for all } x \preceq y.$$

- Much more theoretical and practical work remains to be done!

Concluding remarks

- Component categories contain the essential information given by (algebraic topological invariants of) path spaces
- Compression via component categories is an antidote to the state space explosion problem
- Some of the ideas (for the fundamental category) are implemented and have been tested for huge industrial software from EDF (Éric Goubault & Co., CEA)
- Dihomotopy equivalence: Definition uses automorphic homotopy flows to ensure homotopy equivalences

$$\vec{T}(f)(x,y) : \vec{T}(X)(x,y) \to \vec{T}(Y)(fx,fy) \text{ for all } x \preceq y.$$

- Much more theoretical and practical work remains to be done!

Concluding remarks

- Component categories contain the essential information given by (algebraic topological invariants of) path spaces
- Compression via component categories is an antidote to the state space explosion problem
- Some of the ideas (for the fundamental category) are implemented and have been tested for huge industrial software from EDF (Éric Goubault & Co., CEA)
- Dihomotopy equivalence: Definition uses automorphic homotopy flows to ensure homotopy equivalences

$$\vec{T}(f)(x,y) : \vec{T}(X)(x,y) \to \vec{T}(Y)(fx,fy) \text{ for all } x \preceq y.$$

- Much more theoretical and practical work remains to be done!

Concluding remarks

- Component categories contain the essential information given by (algebraic topological invariants of) path spaces
- Compression via component categories is an antidote to the state space explosion problem
- Some of the ideas (for the fundamental category) are implemented and have been tested for huge industrial software from EDF (Éric Goubault & Co., CEA)
- Dihomotopy equivalence: Definition uses automorphic homotopy flows to ensure homotopy equivalences

$$\vec{T}(f)(x,y) : \vec{T}(X)(x,y) \to \vec{T}(Y)(fx,fy) \text{ for all } x \preceq y.$$

- Much more theoretical and practical work remains to be done!