Precubical and Continuous Control Flow

Topology workshop
Paris 2014

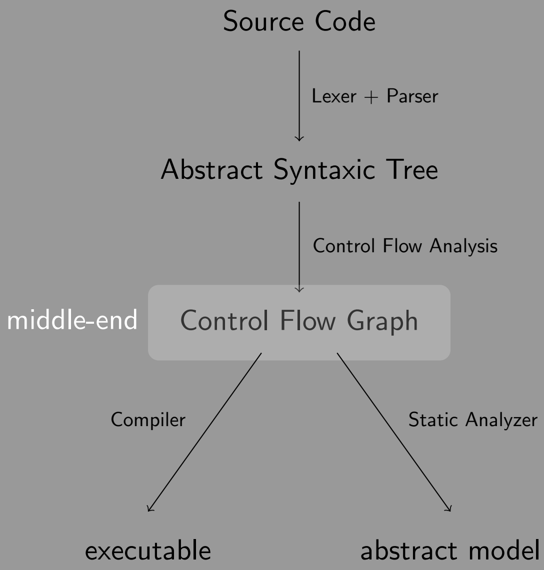
Emmanuel Haucourt

CEA-Tech, NanoInnov

The 15th of July

Control Flow Graphs of Sequential Processes

Control Flow Analysis, Frances E. Allen, SIGPLAN Notices 1970



The overall idea

of Static Analysis

The model of a program should be a finite representation of an overapproximation of the collection of all its execution traces.

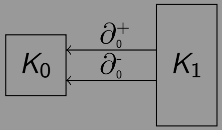
Precubical sets

as presheaves over $\square^+$

K_0

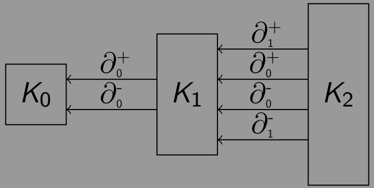
Precubical sets

as presheaves over $\square^+$



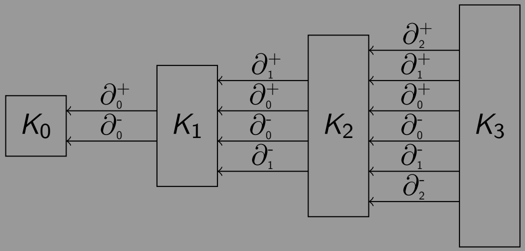
Precubical sets

as presheaves over $\square^+$



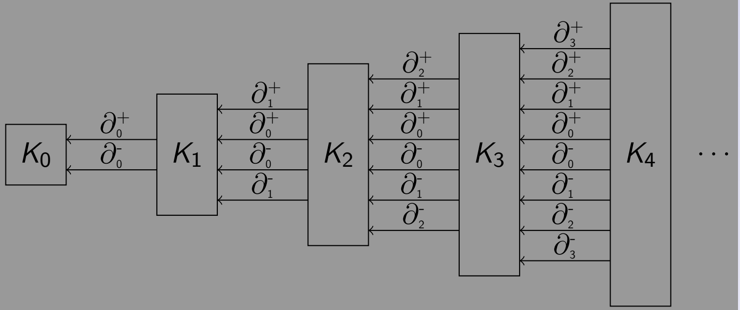
Precubical sets

as presheaves over $\square^+$



Precubical sets

as presheaves over $\square^+$



Tensor product
of precubical sets

Given precubical sets K and K' of dimension p and q, the set of n-cubes for $0 \leqslant n \leqslant p+q$

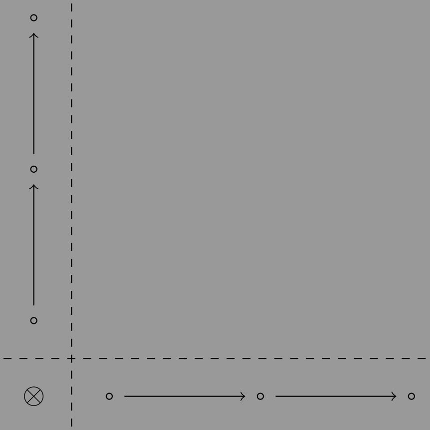
$$(K \otimes K')_n = \bigsqcup_{i+j=n} K_i \times K_j$$

For $x \otimes y \in K_i \times K'_j$ with $i + j = n$ the k^{th} face map, with $0 \leqslant k < n$, is given by

$$\partial_k^{\pm}(x \otimes y) = \begin{cases} \partial_k^{\pm}(x) \otimes y & \text{if } 0 \leqslant k < i \\ x \otimes \partial_{k-p}^{\pm} y) & \text{if } i \leqslant k < n \end{cases}$$

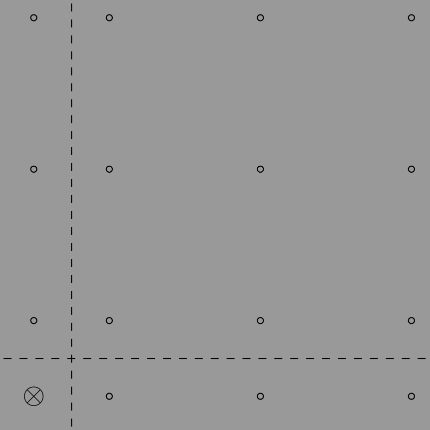
Example of tensor product

of precubical sets



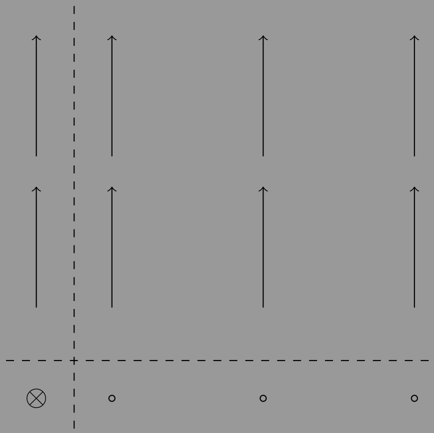
Example of tensor product

of precubical sets

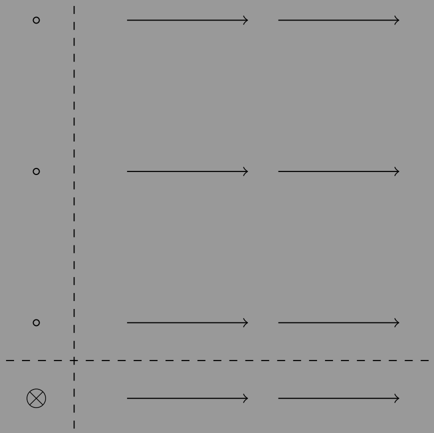


Example of tensor product

of precubical sets

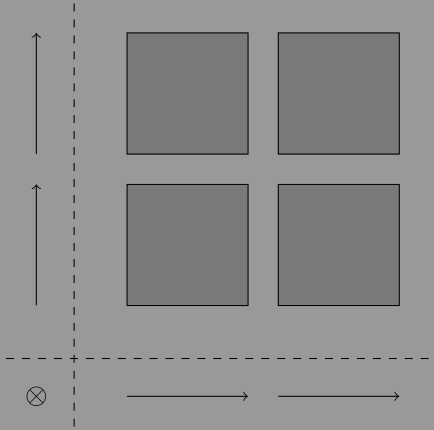


Example of tensor product
of precubical sets



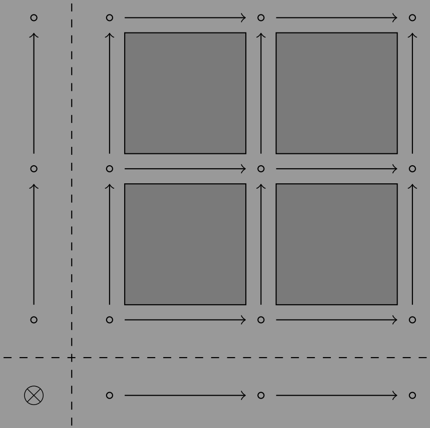
Example of tensor product

of precubical sets



Example of tensor product

of precubical sets



The PV language

Dijkstra 68 - Input language for ALCOOL in an extended form

- *Sem*: set of semaphores with arity in $\mathbb{N} \setminus \{0, 1\}$
- *Mtx*: set of mutex, an alias for a semaphore of arity 2
- A semaphore `x` of arity n is a resource offering $n - 1$ tokens, each process can hold one token or more
- A process acquire a token executing the instruction `P(x)` and release it executing the instruction `V(x)`
- A mutex can be held by only one process at the time
- Trying to perform `P(x)` though `x` is not available blocks the execution unless `x` is a mutex already held by the process
- The instruction `V(x)` is not blocking
- *Wait*: set of synchronization bareers with arity in $\mathbb{N} \setminus \{0, 1\}$
- Instruction `W(x)` blocks the execution of the process until n (arity of `x`) processes are blocked by `x` then all the execution are resumed at the same time

Extending the middle-end representation

Conservative process

A process is said to be conservative when for all paths γ, the amount of resources available at the arrival of γ only depends on the amounts of resources that were available at the origin of γ.

$$\partial^{-}\gamma = \partial^{-}\gamma' \text{ and } \partial^{+}\gamma = \partial^{+}\gamma' \;\Rightarrow\; [\![\gamma]\!] \cdot \delta(x) = [\![\gamma']\!] \cdot \delta(x)$$

Being conservative is decidable and induces a potential function.

The potential function

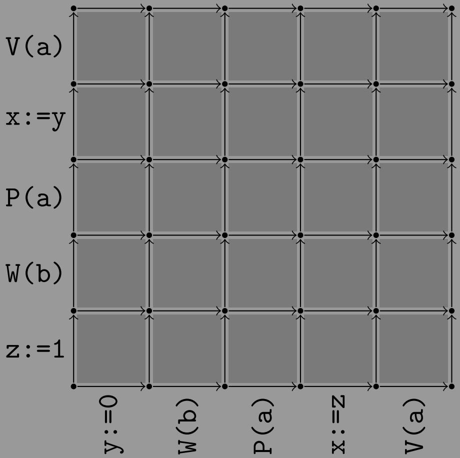
of a PV program $P_1|\cdots|P_d$

- assume each P_k is conservative
  and F_k the associated potential function
- let $K_0 = V_1 \times \cdots \times V_d$ the 0-dimensional cubes of the
  tensor product of the cfgs
- The potential function $F : K_0 \times \mathcal{R} \to \mathbb{N}$ is

$$F(v_1, \ldots, v_d, x) = \sum_{k=1}^{d} F_k(v_k, x)$$

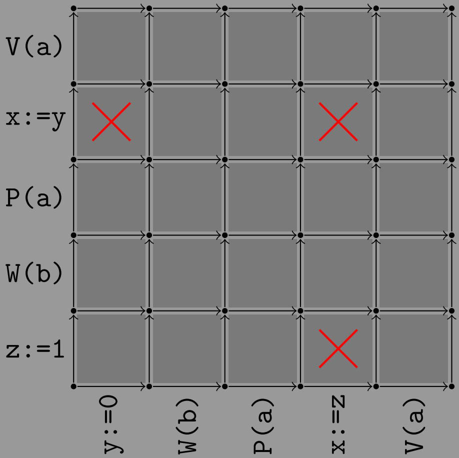
Control Flow Precubical Set: an example

`y:=0.W(b).P(a).x:=z.V(a)|z:=0.W(b).P(a).x:=y.V(a)`



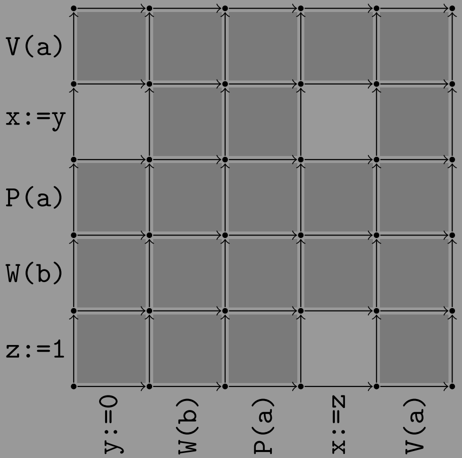
Control Flow Precubical Set: an example

`y:=0.W(b).P(a).x:=z.V(a)|z:=0.W(b).P(a).x:=y.V(a)`



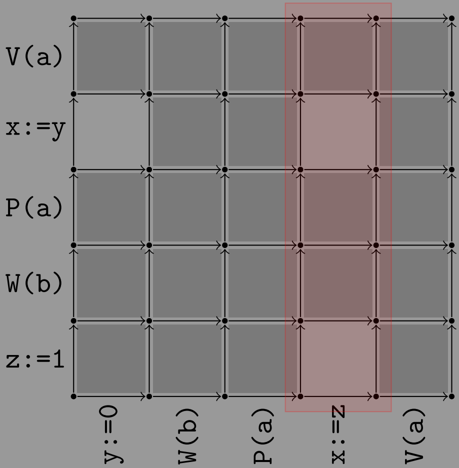
Control Flow Precubical Set: an example

y:=0.W(b).P(a).x:=z.V(a)|z:=0.W(b).P(a).x:=y.V(a)



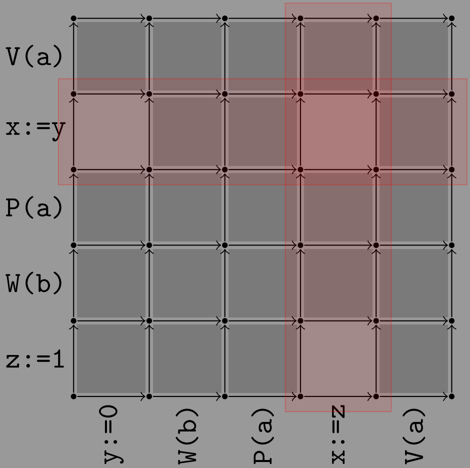
Control Flow Precubical Set: an example

y:=0.W(b).P(a).x:=z.V(a)|z:=0.W(b).P(a).x:=y.V(a)



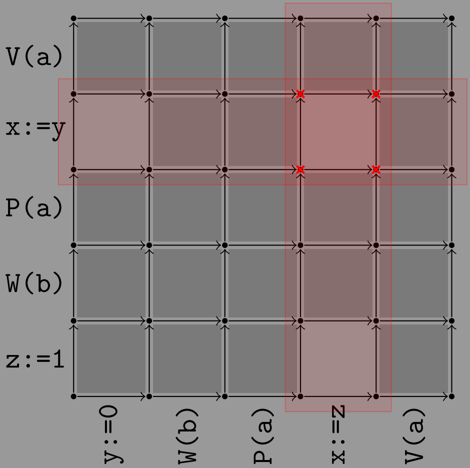
Control Flow Precubical Set: an example

y:=0.W(b).P(a).x:=z.V(a)|z:=0.W(b).P(a).x:=y.V(a)



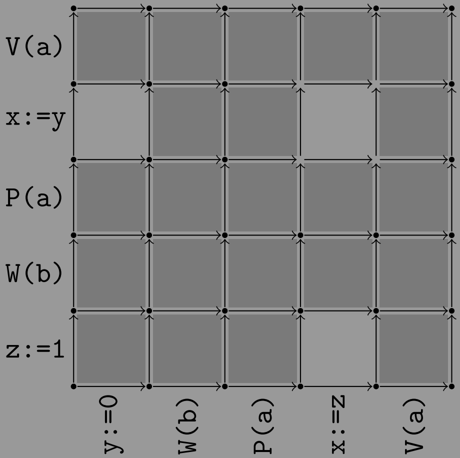
Control Flow Precubical Set: an example

y:=0.W(b).P(a).x:=z.V(a)|z:=0.W(b).P(a).x:=y.V(a)



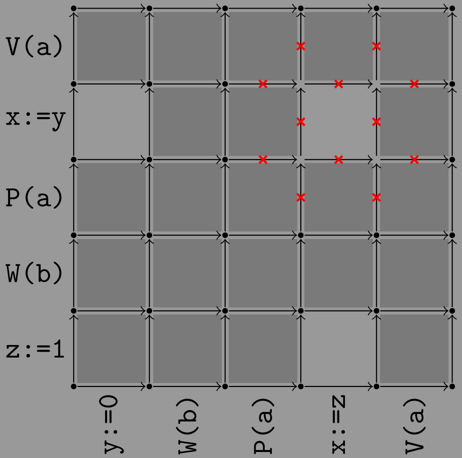
Control Flow Precubical Set: an example

y:=0.W(b).P(a).x:=z.V(a)|z:=0.W(b).P(a).x:=y.V(a)



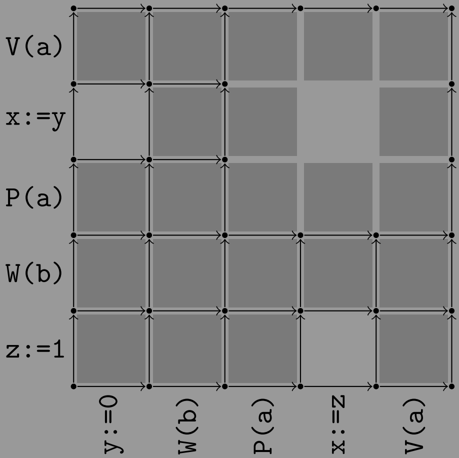
Control Flow Precubical Set: an example

`y:=0.W(b).P(a).x:=z.V(a)|z:=0.W(b).P(a).x:=y.V(a)`



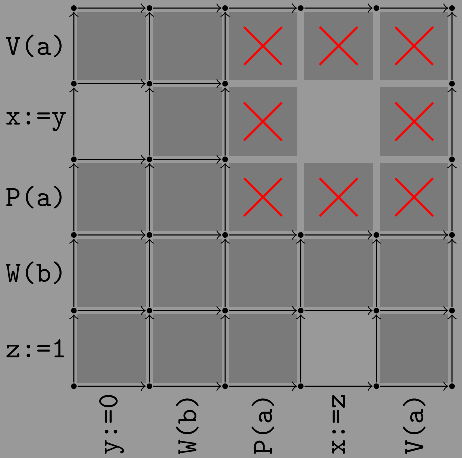
Control Flow Precubical Set: an example

y:=0.W(b).P(a).x:=z.V(a)|z:=0.W(b).P(a).x:=y.V(a)



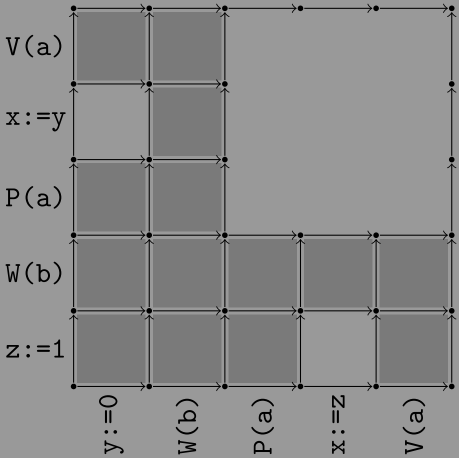
Control Flow Precubical Set: an example

y:=0.W(b).P(a).x:=z.V(a)|z:=0.W(b).P(a).x:=y.V(a)



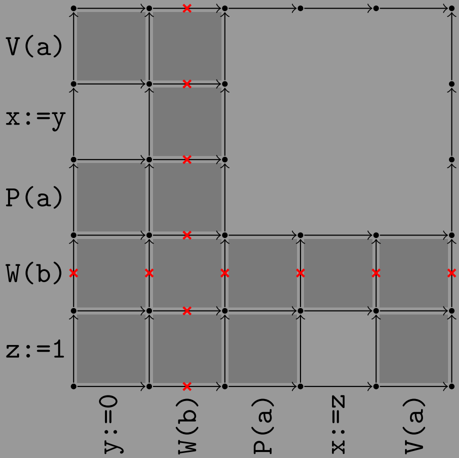
Control Flow Precubical Set: an example

`y:=0.W(b).P(a).x:=z.V(a)|z:=0.W(b).P(a).x:=y.V(a)`



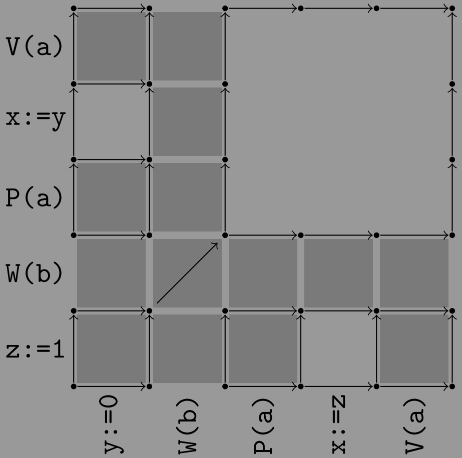
Control Flow Precubical Set: an example

y:=0.W(b).P(a).x:=z.V(a)|z:=0.W(b).P(a).x:=y.V(a)



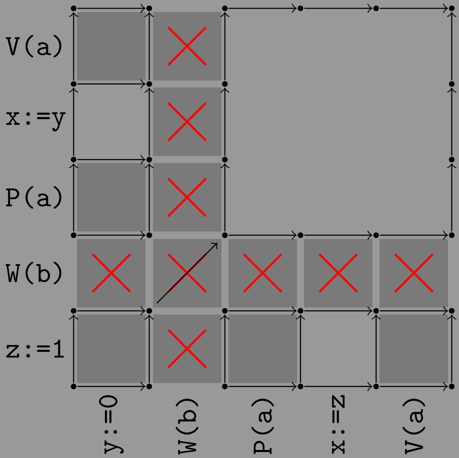
Control Flow Precubical Set: an example

y:=0.W(b).P(a).x:=z.V(a)|z:=0.W(b).P(a).x:=y.V(a)



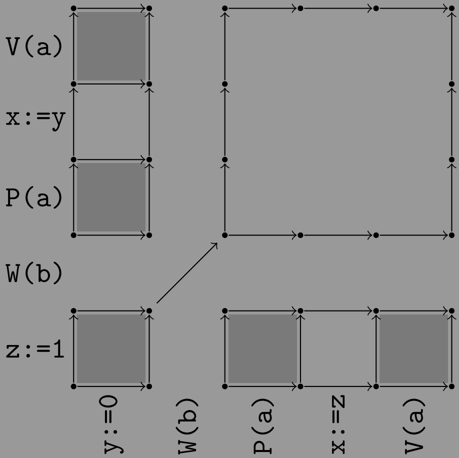
Control Flow Precubical Set: an example

`y:=0.W(b).P(a).x:=z.V(a)|z:=0.W(b).P(a).x:=y.V(a)`



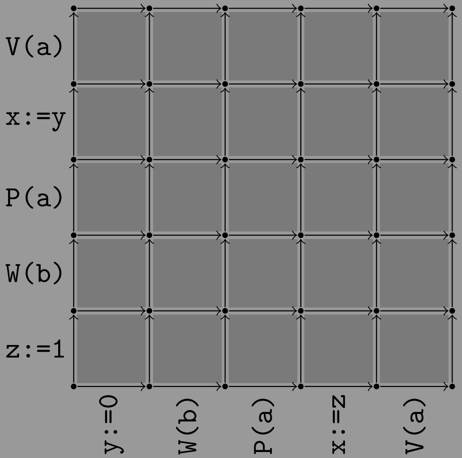
Control Flow Precubical Set: an example

y:=0.W(b).P(a).x:=z.V(a)|z:=0.W(b).P(a).x:=y.V(a)



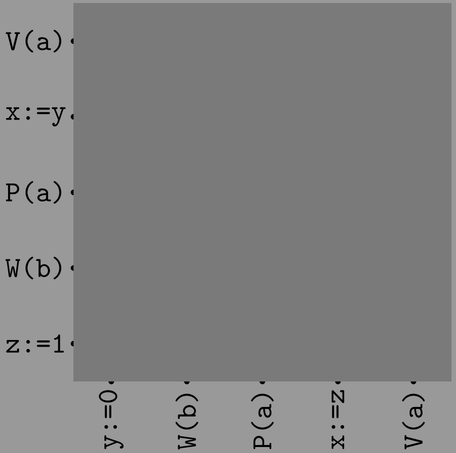
Geometric model: an example

y:=0.W(b).P(a).x:=z.V(a)|z:=0.W(b).P(a).x:=y.V(a)



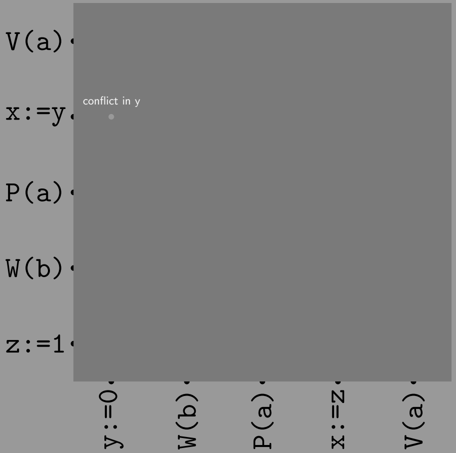
Geometric model: an example

y:=0.W(b).P(a).x:=z.V(a)|z:=0.W(b).P(a).x:=y.V(a)



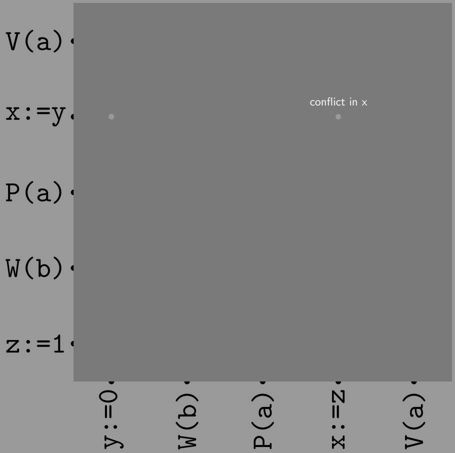
Geometric model: an example

y:=0.W(b).P(a).x:=z.V(a)|z:=0.W(b).P(a).x:=y.V(a)



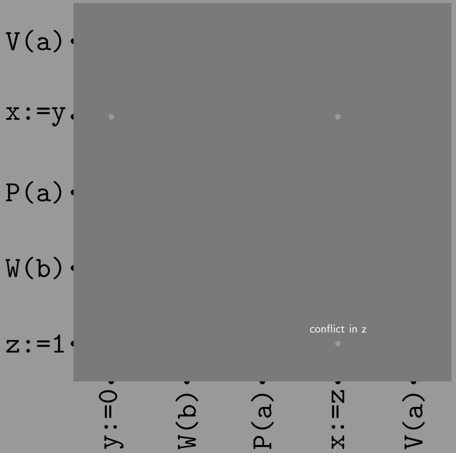
Geometric model: an example

y:=0.W(b).P(a).x:=z.V(a)|z:=0.W(b).P(a).x:=y.V(a)



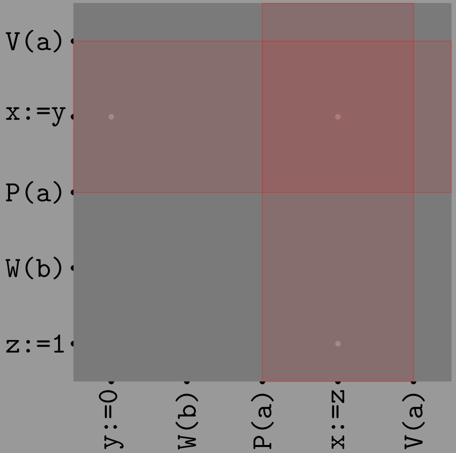
Geometric model: an example

y:=0.W(b).P(a).x:=z.V(a)|z:=0.W(b).P(a).x:=y.V(a)



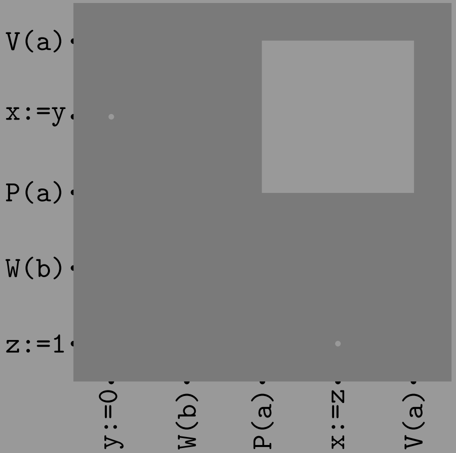
Geometric model: an example

`y:=0.W(b).P(a).x:=z.V(a)|z:=0.W(b).P(a).x:=y.V(a)`



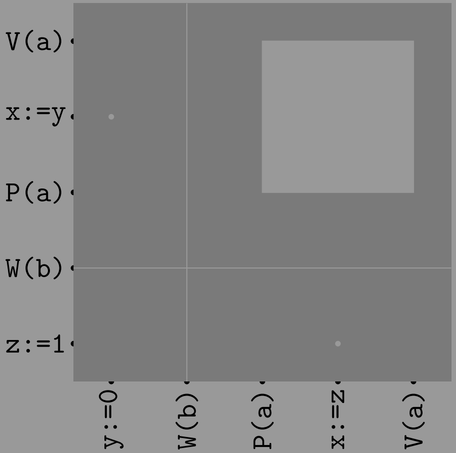
Geometric model: an example

y:=0.W(b).P(a).x:=z.V(a)|z:=0.W(b).P(a).x:=y.V(a)



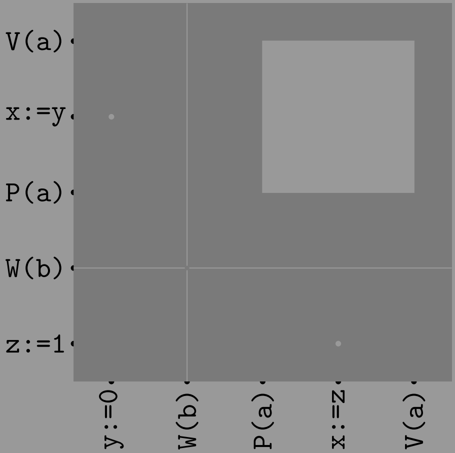
Geometric model: an example

y:=0.W(b).P(a).x:=z.V(a)|z:=0.W(b).P(a).x:=y.V(a)



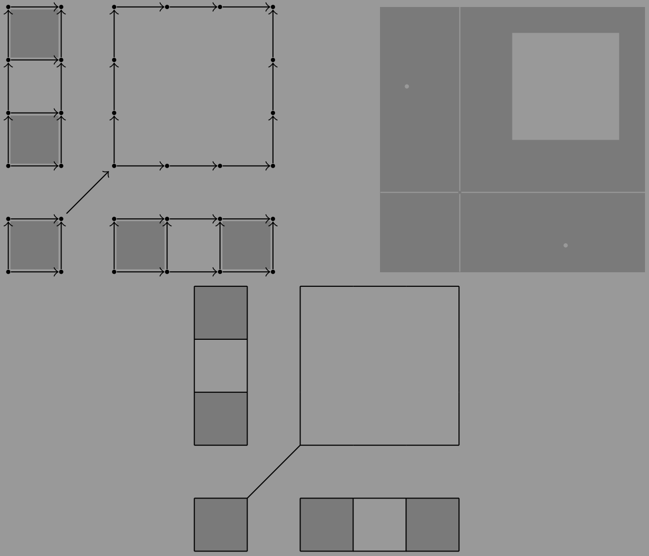
Geometric model: an example

y:=0.W(b).P(a).x:=z.V(a)|z:=0.W(b).P(a).x:=y.V(a)



Comparing

Discrete vs Continuous



Cubical areas

- A pospace is a topological space with a closed partial order
- The Morphisms of pospace are the continuous increasing maps
- A *n*-cube is the product of a *n*-uple of intervals of $\mathbb{R}$
- A *n*-cubical area is a finite union of *n*-cubes
- A *n*-cubical area inherits a pospace structure from $\mathbb{R}^n$

Prime decomposition theorem for areas

- areas form a commutative monoid with cartesian product
- this commutative monoid is free
- prime decomposition of $[\![P]\!]$ provide information about parallel decomposition of P.

Dipath

on a cubical area X

- Dipath are continuous increasing maps $\gamma : [0, r] \to X$ with $r \geqslant 0$, $\partial^- \gamma = \gamma(0)$ and $\partial^+ \gamma = \gamma(r)$
- Concatenation $\gamma \cdot \delta : [0, r + r'] \to X$ when $\partial^- \gamma = \partial^+ \delta$;

$$\gamma \cdot \delta(t) = \begin{cases} \delta(t) & \text{if } t \leqslant r \\ \gamma(t) & \text{if } r \leqslant t \end{cases}$$

- If X is the model of a program then the dipaths on X is an overapproximation of the execution traces
- Infinitely many paths between two points

Elementary homotopy

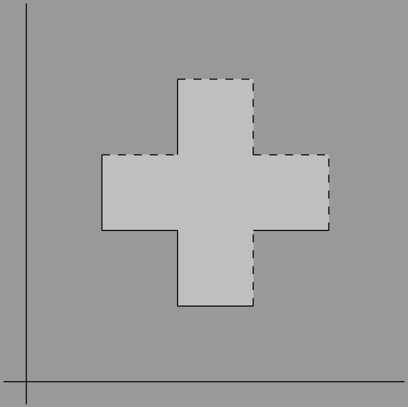
- dihomotopy $h:[0,r]\times[0,\rho]\to X$ a morphism s.t.
  $h(0,\_)$ and $h(r,\_)$ are both constant
- anti-dihomotopy $h:[0,r]\times[0,\rho]\to X$ such that
$(t,x)\mapsto h(t,-x)$ is a dihomotopy
- elementary homotopy $h_n*\cdots*h_1$ where each h_k
  is either a dihomotopy or an antidihomotopy
- $\gamma\sim\delta$ when there exist an elementary homotopy between
$\gamma\theta$ and $\delta\psi$ for some θ and ψ both increasing and surjective,
and sharing their domain of definition.

Characterizing dihomotopy classes through areas

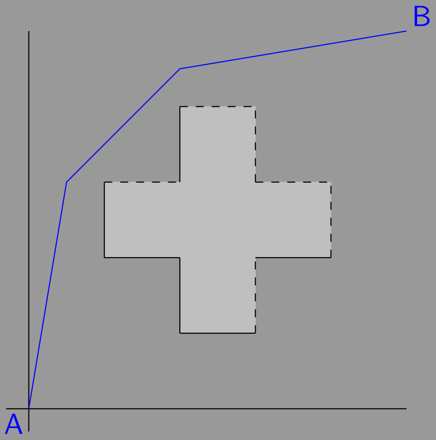
- X cubical area, for all dipath γ there exist a cubical area s.t. $\delta \sim \gamma$ iff img(δ)$\subseteq A_\gamma$
- in fact $\gamma \sim \delta$ iff $A_\gamma = A_\delta$
- further there is a finite collection $\mathcal{K}$ of subareas of X such that for all γ and δ sharing their extremities,

$\gamma \sim \delta$ iff for all $K \in \mathcal{K}$, img(γ)$\subseteq K \Leftrightarrow$ img(δ)$\subseteq K$

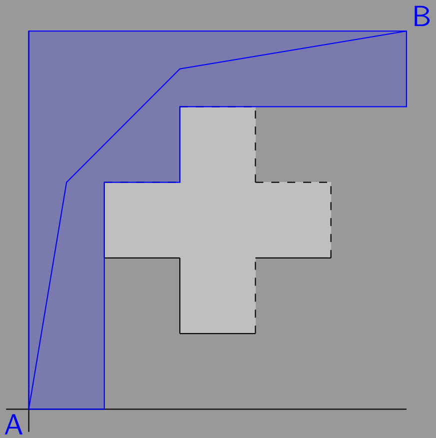
Dihomotopy classes as cubical areas



Dihomotopy classes as cubical areas



Dihomotopy classes as cubical areas



Thank you