Small Proofs

Dan Licata
Wesleyan University

Thanks AFOSR (FA9550-15-1-0053,FA9550-16-1-0292), NSF (DMS 1321794)

```
   movl $0, %eax
   addl %eax, %ebx
   popl %eax
   looptop:
=> imul %edx
   andl $0xFF, %eax
   cmpl $100, %eax
   jb looptop
   leal 4(%esp), %ebp
   movl %esi, %edi
   subl $8, %edi
   shrl %cl, %ebx
   movw %bx, -2(%ebp)
```

x86

```
   movl $0, %eax
   addl %eax, %ebx
   popl %eax
   looptop:
=> imul %edx
   andl $0xFF, %eax
   cmpl $100, %eax
   jb looptop
   leal 4(%esp), %ebp
   movl %esi, %edi
   subl $8, %edi
   shrl %cl, %ebx
   movw %bx, -2(%ebp)
```

x86 

ARM 

```
   movl $0, %eax
   addl %eax, %ebx
   popl %eax
   looptop:
=> imul %edx
   andl $0xFF, %eax
   cmpl $100, %eax
   jb looptop
   leal 4(%esp), %ebp
   movl %esi, %edi
   subl $8, %edi
   shrl %cl, %ebx
   movw %bx, -2(%ebp)
```

x86 

ARM 

```
   movl $0, %eax
   addl %eax, %ebx
   popl %eax
   looptop:
=> imul %edx
   andl $0xFF, %eax
   cmpl $100, %eax
   jb looptop
   leal 4(%esp), %ebp
   movl %esi, %edi
   subl $8, %edi
   shrl %cl, %ebx
   movw %bx, -2(%ebp)
```

```
LDR r0,[p_a]
LDR r1,[p_b]
ADD r3,r0,r1
STR r3,[p_w]
LDR r2,[p_c]
ADD r0,r2,r3
STR r0,[p_x]
LDR r0,[p_d]
ADD r3,r2,r0
STR r3,[p_y]
```

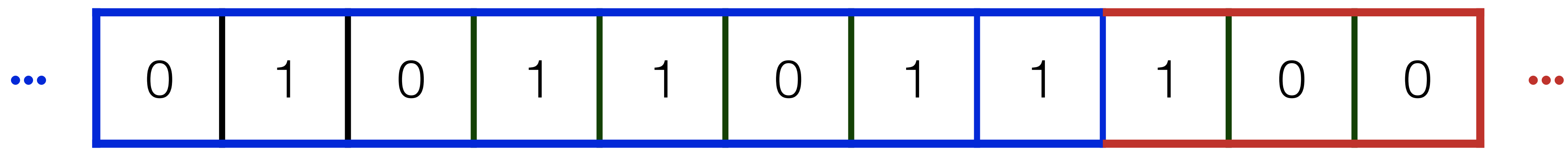
High-Level Language

C

```
for (int i=0; i<N; i++)
{
  A[i]++;
}
```

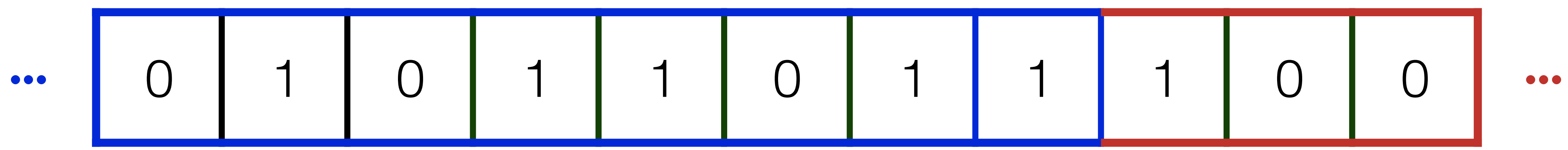
Portable Assembly Language

```
A : bit array[8]
x : int
```



Portable Assembly Language

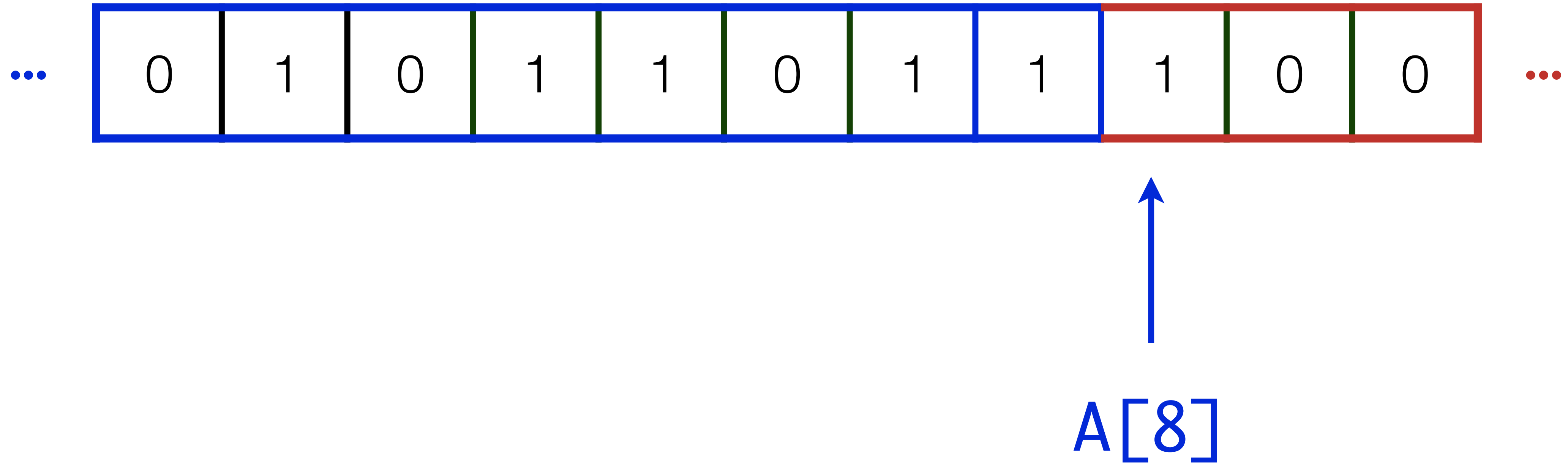
```
A : bit array[8]
x : int
```



A[8]

Portable Assembly Language

```
A : bit array[8]
x : int
```



High-Level Language

ML

High-Level Language

ML

```
map (λ x → x + 1) A
```

High-Level Language

ML

`map (λ x → x + 1) A`

enforces abstractions: what you ***can't*** *do matters*

CUDA

```
void matrixMultiplication(float *A, float *B, float *C, int N){

    // declare the number of blocks per grid and the number of threads per block
    // use 1 to 512 threads per block
    dim3 threadsPerBlock(N, N);
    dim3 blocksPerGrid(1, 1);
        if (N*N > 512){
            threadsPerBlock.x = 512;
            threadsPerBlock.y = 512;
            blocksPerGrid.x = ceil(double(N)/double(threadsPerBlock.x));
            blocksPerGrid.y = ceil(double(N)/double(threadsPerBlock.y));
        }

    matrixMultiplicationKernel<<<blocksPerGrid,threadsPerBlock>>>(A, B, C, N);
}
```

Domain-specific HLLs

C#

```
pfor (int i=0; i<N; i++)
{
  A[i]++;
}
```

NESL

```
pmap (λ x → x + 1) A
```

SQL

```
SELECT LAT_N, CITY, TEMP_F
FROM STATS, STATION
WHERE MONTH = 7
AND STATS.ID = STATION.ID
ORDER BY TEMP_F;
```

LAT_N	CITY	TEMP_F
47	Caribou	65.8
40	Denver	74.8
33	Phoenix	91.7

Target Architecture

ARM 

x86 

High-level Languages

C/ML/…

Domain-specific Languages

NESL 

SQL 

Target Architecture

ARM  ZFC/w.f. trees

x86 

High-level Languages

C/ML/…

Domain-specific Languages

NESL 

SQL 

Target Architecture

ARM  ZFC/w.f. trees

x86  type theory/meaning expl.

High-level Languages

C/ML/…

Domain-specific Languages

NESL 

SQL 

Target Architecture

ARM  ZFC/w.f. trees

x86  type theory/meaning expl.

High-level Languages

C/ML/… Agda/Gallina/Isar/…

Domain-specific Languages

NESL 

SQL 

Target Architecture

ARM  ZFC/w.f. trees

x86  type theory/meaning expl.

High-level Languages

C/ML/… Agda/Gallina/Isar/…

Domain-specific Languages

NESL  MLTT+UA+HITs

SQL 

Target Architecture

ARM 

x86 

ZFC/w.f. trees

type theory/meaning expl.

High-level Languages

C/ML/…

Agda/Gallina/Isar/…

Domain-specific Languages

NESL 

SQL 

MLTT+UA+HITs

Cubical type theory

Target Architecture

ARM  ZFC/w.f. trees

x86  type theory/meaning expl.

High-level Languages

C/ML/… Agda/Gallina/Isar/…

Domain-specific Languages

NESL 

MLTT+UA+HITs

Cubical type theory

SQL  Modal type theory the **big proof** is the compiler/model of the DSL the **big proof** is the compiler/model of the DSL

math *in* the DSL is **smaller**

internal languages:
short statements *inside* a particular categorical setting can unpack to long external statements

internal languages:
short statements *inside* a particular categorical setting can unpack to long external statements

synthetic X

Objections

- Requires cleverness; may never apply
- Requires specialists
- Computability (Gödel) / complexity theory (Blum) limits on any particular language
- Practical limits on DSLs

HoTT'13 (MLTT+UA+HITs)

HoTT'13 (MLTT+UA+HITs)

- DSL for homotopy types that should compile to set theory: simplicial sets model **and** other settings for homotopy theory (∞-toposes)

HoTT'13 (MLTT+UA+HITs)

- DSL for homotopy types that should compile to set theory: simplicial sets model **and** other settings for homotopy theory (∞-toposes)
- General purpose/foundational: includes all traditional (and classical) math

Blakers-Massey Theorem

$$\begin{tikzcd} Z \arrow[r, bend left, "g"] \arrow[d, bend right, "f"'] & Y \\ X & \end{tikzcd}$$

Blakers-Massey Theorem

$$\begin{tikzcd}
Z \arrow[drr, bend left, "g"] \arrow[dr, bend right, "f"'] & & \\
& & Y \arrow[d, "\nu_2"] \\
& X \arrow[r, "\nu_1"] & W := X +_Z Y
\end{tikzcd}$$

Blakers-Massey Theorem

$$
\begin{tikzcd}
Z \arrow[drr, bend left, "g"] \arrow[ddr, bend right, "f"'] & & \\
& X \times_W Y \arrow[r, "\pi_2"] \arrow[d, "\pi_1"] & Y \arrow[d, "\nu_2"] \\
& X \arrow[r, "\nu_1"] & W := X +_Z Y
\end{tikzcd}
$$

Blakers-Massey Theorem

$$
\begin{tikzcd}
Z \arrow[dr] \arrow[drr, bend left, "g"] \arrow[ddr, bend right, "f"'] & & \\
& X \times_W Y \arrow[r, "\pi_2"] \arrow[d, "\pi_1"] & Y \arrow[d, "\nu_2"] \\
& X \arrow[r, "\nu_1"] & W := X +_Z Y
\end{tikzcd}
$$

Blakers-Massey Theorem

$$
\begin{tikzcd}
Z \arrow[dr] \arrow[drr, bend left, "g"] \arrow[ddr, bend right, "f"'] & & \\
& X \times_W Y \arrow[r, "\pi_2"] \arrow[d, "\pi_1"] & Y \arrow[d, "\nu_2"] \\
& X \arrow[r, "\nu_1"] & W := X +_Z Y
\end{tikzcd}
$$

m-connected

Blakers-Massey Theorem

n-connected

$$\begin{tikzcd}
Z \arrow[dr] \arrow[drr, bend left, "g"] \arrow[ddr, bend right, "f"'] & & \\
& X \times_W Y \arrow[r, "\pi_2"] \arrow[d, "\pi_1"] & Y \arrow[d, "\nu_2"] \\
& X \arrow[r, "\nu_1"] & W := X +_Z Y
\end{tikzcd}$$

m-connected

Blakers-Massey Theorem

$(m+n)$-connected

n-connected

$$
\begin{tikzcd}
Z \arrow[dr] \arrow[drrr, bend left, "g"] \arrow[ddr, bend right, "f"'] & & & \\
& X \times_W Y \arrow[rr, "\pi_2"] \arrow[d, "\pi_1"] & & Y \arrow[d, "\nu_2"] \\
& X \arrow[rr, "\nu_1"] & & W := X +_Z Y
\end{tikzcd}
$$

m-connected

Homotopy groups of spheres

kth homotopy group

n-dimensional sphere

	π_1	π_2	π_3	π_4	π_5	π_6	π_7	π_8	π_9	π_{10}	π_{11}	π_{12}	π_{13}	π_{14}	π_{15}
S^0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
S^1	$\mathbf{Z}$	0	0	0	0	0	0	0	0	0	0	0	0	0	0
S^2	0	$\mathbf{Z}$	$\mathbf{Z}$	$\mathbf{Z}_2$	$\mathbf{Z}_2$	$\mathbf{Z}_{12}$	$\mathbf{Z}_2$	$\mathbf{Z}_2$	$\mathbf{Z}_3$	$\mathbf{Z}_{15}$	$\mathbf{Z}_2$	$\mathbf{Z}_2^2$	$\mathbf{Z}_{12}\times\mathbf{Z}_2$	$\mathbf{Z}_{84}\times\mathbf{Z}_2^2$	$\mathbf{Z}_2^2$
S^3	0	0	$\mathbf{Z}$	$\mathbf{Z}_2$	$\mathbf{Z}_2$	$\mathbf{Z}_{12}$	$\mathbf{Z}_2$	$\mathbf{Z}_2$	$\mathbf{Z}_3$	$\mathbf{Z}_{15}$	$\mathbf{Z}_2$	$\mathbf{Z}_2^2$	$\mathbf{Z}_{12}\times\mathbf{Z}_2$	$\mathbf{Z}_{84}\times\mathbf{Z}_2^2$	$\mathbf{Z}_2^2$
S^4	0	0	0	$\mathbf{Z}$	$\mathbf{Z}_2$	$\mathbf{Z}_2$	$\mathbf{Z}\times\mathbf{Z}_{12}$	$\mathbf{Z}_2^2$	$\mathbf{Z}_2^2$	$\mathbf{Z}_{24}\times\mathbf{Z}_3$	$\mathbf{Z}_{15}$	$\mathbf{Z}_2$	$\mathbf{Z}_2^3$	$\mathbf{Z}_{120}\times\mathbf{Z}_{12}\times\mathbf{Z}_2$	$\mathbf{Z}_{84}\times\mathbf{Z}_2^5$
S^5	0	0	0	0	$\mathbf{Z}$	$\mathbf{Z}_2$	$\mathbf{Z}_2$	$\mathbf{Z}_{24}$	$\mathbf{Z}_2$	$\mathbf{Z}_2$	$\mathbf{Z}_2$	$\mathbf{Z}_{30}$	$\mathbf{Z}_2$	$\mathbf{Z}_2^3$	$\mathbf{Z}_{72}\times\mathbf{Z}_2$
S^6	0	0	0	0	0	$\mathbf{Z}$	$\mathbf{Z}_2$	$\mathbf{Z}_2$	$\mathbf{Z}_{24}$	0	$\mathbf{Z}$	$\mathbf{Z}_2$	$\mathbf{Z}_{60}$	$\mathbf{Z}_{24}\times\mathbf{Z}_2$	$\mathbf{Z}_2^3$
S^7	0	0	0	0	0	0	$\mathbf{Z}$	$\mathbf{Z}_2$	$\mathbf{Z}_2$	$\mathbf{Z}_{24}$	0	0	$\mathbf{Z}_2$	$\mathbf{Z}_{120}$	$\mathbf{Z}_2^3$
S^8	0	0	0	0	0	0	0	$\mathbf{Z}$	$\mathbf{Z}_2$	$\mathbf{Z}_2$	$\mathbf{Z}_{24}$	0	0	$\mathbf{Z}_2$	$\mathbf{Z}\times\mathbf{Z}_{120}$

[image from wikipedia]

Pushouts

$$
\begin{tikzcd}
Z \arrow[rr, bend left, "g"] \arrow[dr, bend right, "f"'] & & Y \arrow[d, "\nu_2"] \\
& X \arrow[r, "\nu_1"] & W := X +_Z Y
\end{tikzcd}
$$

Pushouts

$$
\begin{tikzcd}
Z \arrow[drr, bend left, "g"] \arrow[ddr, bend right, "f"'] & & \\
& & Y \arrow[d, "\nu_2"] \\
& X \arrow[r, "\nu_1"] & W := X +_Z Y
\end{tikzcd}
$$

$\mathtt{inl} : X \to X +_Z Y$

Pushouts

$$\begin{tikzcd}
Z \arrow[r, bend left, "g"] \arrow[d, bend right, "f"'] & Y \arrow[d, "\nu_2"] \\
X \arrow[r, "\nu_1"] & W := X +_Z Y
\end{tikzcd}$$

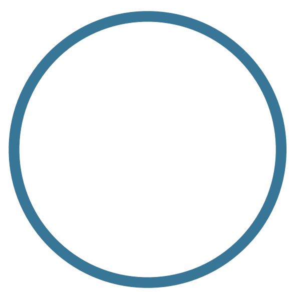
```
inl : X → X +Z Y
inr : Y → X +Z Y
```

Pushouts

$$
\begin{tikzcd}
Z \arrow[rrd, bend left, "g"] \arrow[rdd, bend right, "f"'] & & \\
& & Y \arrow[d, "\nu_2"] \\
& X \arrow[r, "\nu_1"] & W := X +_Z Y
\end{tikzcd}
$$

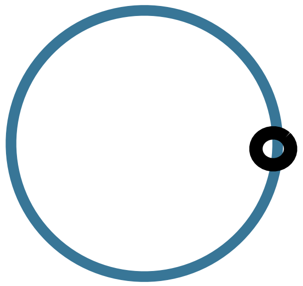
```
 inl  : X → X +Z Y
 inr  : Y → X +Z Y
glue  : Π(z : A) → inl(f z) = inr (g z)
```

Pushouts



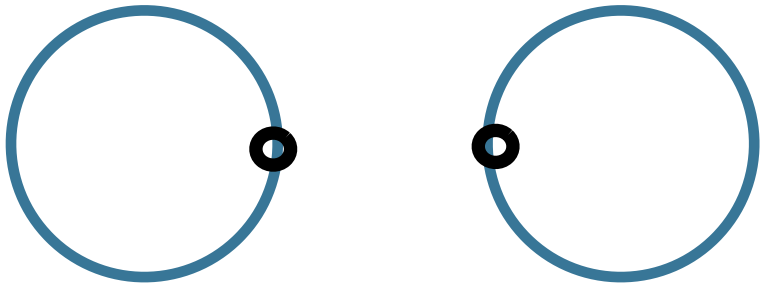
Pushouts

`base : 1 → Circle`

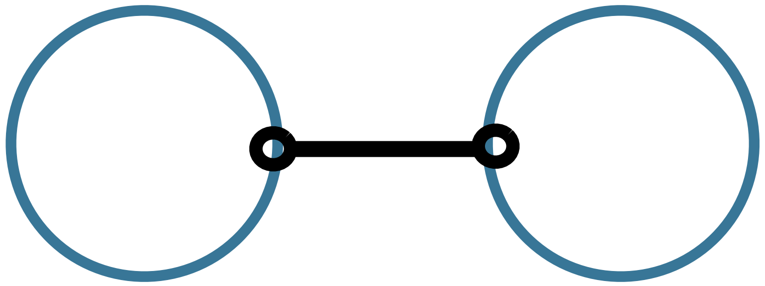


Pushouts

`base : 1 → Circle`



Pushouts

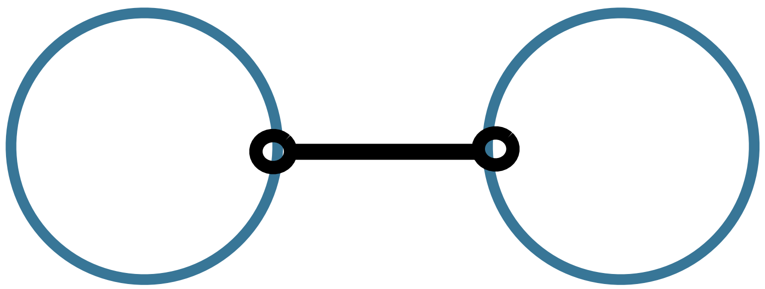


`base : 1 → Circle`

$\texttt{Circle} +_1 \texttt{Circle}$

Pushouts

`base : 1 → Circle`

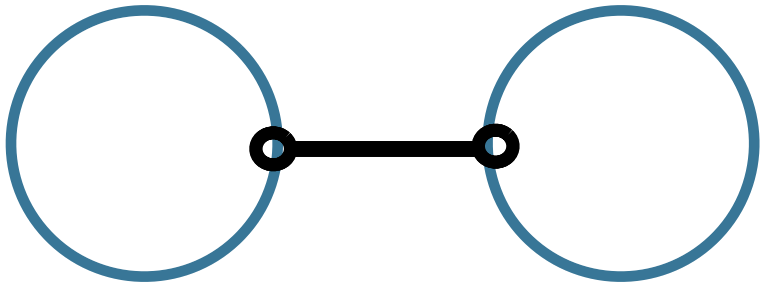


`Circle +₁ Circle`

`inl : Circle → Circle +₁ Cirlce`

Pushouts

base : 1 → Circle

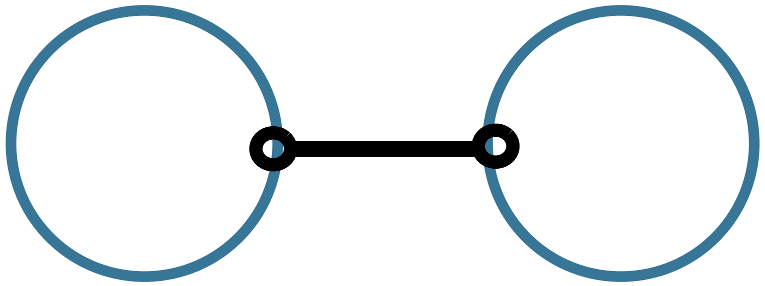


Circle $+_1$ Circle

inl : Circle → Circle $+_1$ Cirlce
inr : Circle → Circle $+_1$ Circle

Pushouts

base : 1 → Circle



Circle $+_1$ Circle

inl : Circle → Circle $+_1$ Cirlce

inr : Circle → Circle $+_1$ Circle

glue : inl(base<>) = inr (base<>)

Pullbacks

$$
\begin{tikzcd}
Z \arrow[drr, bend left, "g"] \arrow[ddr, bend right, "f"'] & & \\
& X \times_W Y \arrow[r, "\pi_2"] \arrow[d, "\pi_1"] & Y \arrow[d, "\nu_2"] \\
& X \arrow[r, "\nu_1"] & W := X +_Z Y
\end{tikzcd}
$$

Pullbacks

$$
\begin{tikzcd}
Z \arrow[drr, bend left, "g"] \arrow[ddr, bend right, "f"'] & & \\
& X \times_W Y \arrow[r, "\pi_2"] \arrow[d, "\pi_1"] & Y \arrow[d, "\nu_2"] \\
& X \arrow[r, "\nu_1"] & W := X +_Z Y
\end{tikzcd}
$$

$$X \times_W Y := \Sigma\ x{:}X,\ y{:}Y.\ \nu_1(x) =_W \nu_2(y)$$

Homotopy Groups

$x : X$

$p : x =_X y$

$q : p_1 =_{x=y} p_2$

$r : q_1 =_{p1=p2} q_2$

$\vdots$

Homotopy Groups

$x : X$

$p : x =_X y$ $\qquad\qquad \pi_1(X)$

$q : p_1 =_{x=y} p_2$

$r : q_1 =_{p1=p2} q_2$

$\vdots$

Homotopy Groups

$x : X$

$p : x =_X y$ $\qquad \pi_1(X)$

$q : p_1 =_{x=y} p_2$ $\qquad \pi_2(X)$

$r : q_1 =_{p1=p2} q_2$

$\vdots$

Homotopy Groups

$x : X$

$p : x =_X y$ $\qquad \pi_1(X)$

$q : p_1 =_{x=y} p_2$ $\qquad \pi_2(X)$

$r : q_1 =_{p1=p2} q_2$ $\qquad \pi_3(X)$

$\vdots$

Connectedness

$$Z \xrightarrow{f} X \quad \text{is n-connected}$$

$$\begin{array}{ccc}
\pi_1(Z) & \cong & \pi_1(X) \\
\pi_2(Z) & \cong & \pi_2(X) \\
\pi_3(Z) & \cong & \pi_3(X) \\
\vdots & & \vdots \\
\pi_n(Z) & \cong & \pi_n(X) \\
\pi_{n+1}(Z) & \rightarrowtail & \pi_{n+1}(X)
\end{array}$$

Connectedness

$Z \xrightarrow{f} X$ is n-connected

$$\left.\begin{array}{ccc} \pi_1(Z) & \cong & \pi_1(X) \\ \pi_2(Z) & \cong & \pi_2(X) \\ \pi_3(Z) & \cong & \pi_3(X) \\ \vdots & & \vdots \\ \pi_n(Z) & \cong & \pi_n(X) \end{array}\right\} \quad \| Z \|_n \cong \| X \|_n$$

$$\pi_{n+1}(Z) \rightarrowtail \pi_{n+1}(X)$$

Blakers-Massey Theorem

(*m*+*n*)-connected

n-connected

$$
\begin{tikzcd}
Z \arrow[dr] \arrow[drr, bend left, "g"] \arrow[ddr, bend right, "f"'] & & \\
& X \times_W Y \arrow[r, "\pi_2"] \arrow[d, "\pi_1"] & Y \arrow[d, "\nu_2"] \\
& X \arrow[r, "\nu_1"] & W := X +_Z Y
\end{tikzcd}
$$

m-connected

Blakers-Massey Theorem

(*m*+*n*)-connected

n-connected

$$
\begin{tikzcd}
Z \arrow[dr] \arrow[drr, bend left, "g"] \arrow[ddr, bend right, "f"'] & & \\
& X \times_W Y \arrow[r, "\pi_2"] \arrow[d, "\pi_1"] & Y \arrow[d, "\nu_2"] \\
& X \arrow[r, "\nu_1"] & W := X +_Z Y
\end{tikzcd}
$$

m-connected

Definitions make sense in an arbitrary ∞-topos…

Blakers-Massey Theorem

- Existing ∞-topos-theoretic proof worked by picking a particular site (model), but can't do that
- HoTT proof [Favonia, Finster, L., Lumsdaine,'16]

  stays *in* the language

  700 lines of code

  translates to ∞-topos language

  new applications [Anel,Biedermann,Finster,Joyal]

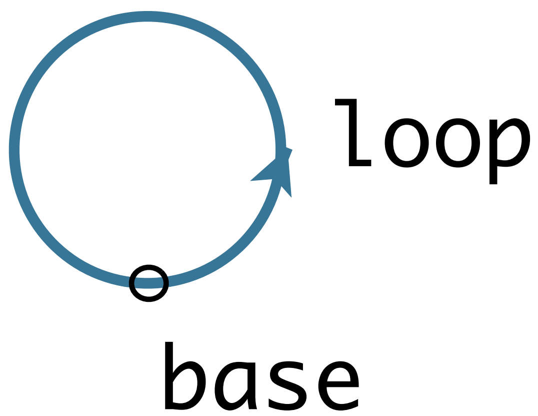
Blakers-Massey Theorem

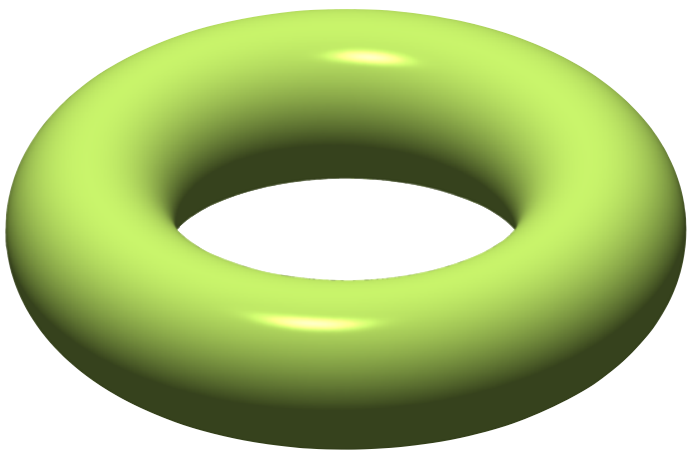
an “encode-decode” proof [L.,Shulman]

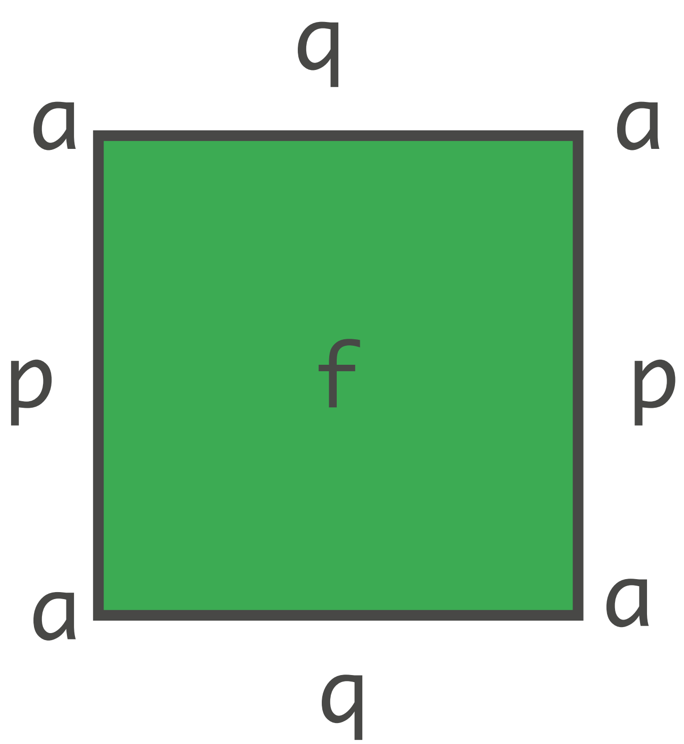
```
CodesFor : (w : W) (p : Path{W} (inm p0) w) → Type
CodesFor = Pushout-elim _
              (λ x α → ‖ Fiber gluel0 α ‖n+m)
              (λ x y p α → ‖ Fiber gluem (x,y,p,α) ‖n+m)
              (λ y α → ‖ Fiber gluer0 α ‖n+m)
              (λ x y pxy → (λ αm αl d → ua (glue-m-l pxy d)))
              (λ x y pxy → (λ αm αl d → ua (glue-m-r pxy d)))
```

Cubical type theory

Circle and Torus



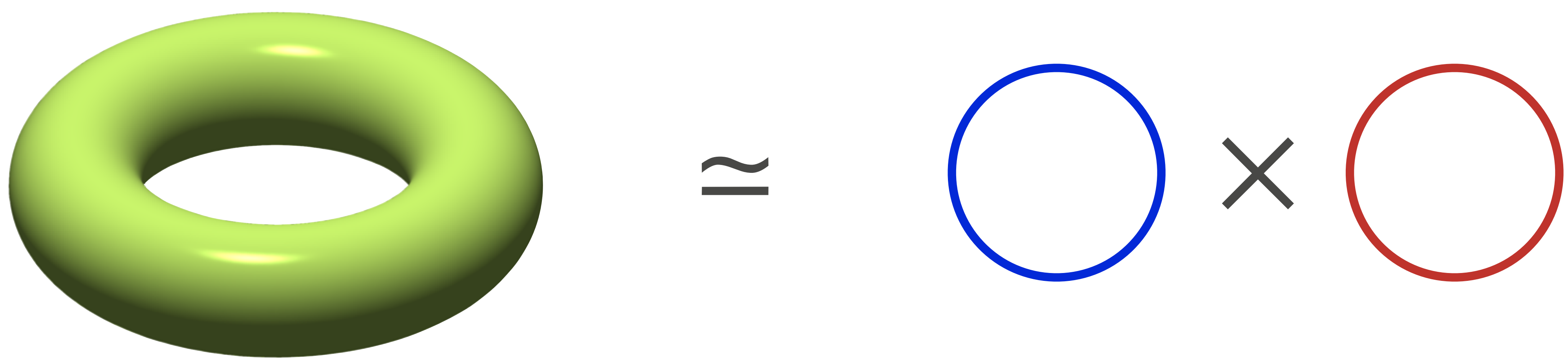




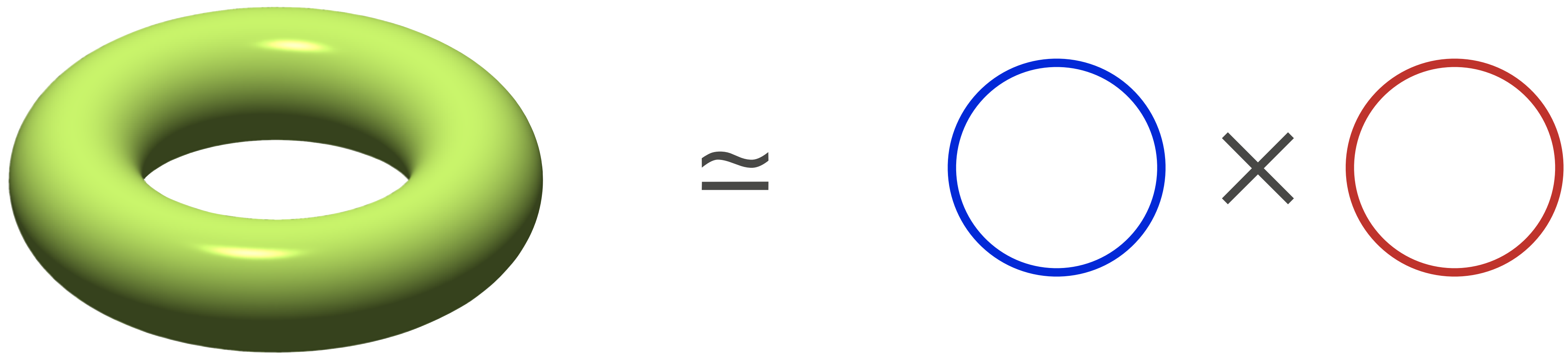
$$\mathtt{base} : S^1$$

$$\mathtt{loop} : \mathtt{base} =_{S^1} \mathtt{base}$$

$T \simeq S^1 \times S^1$

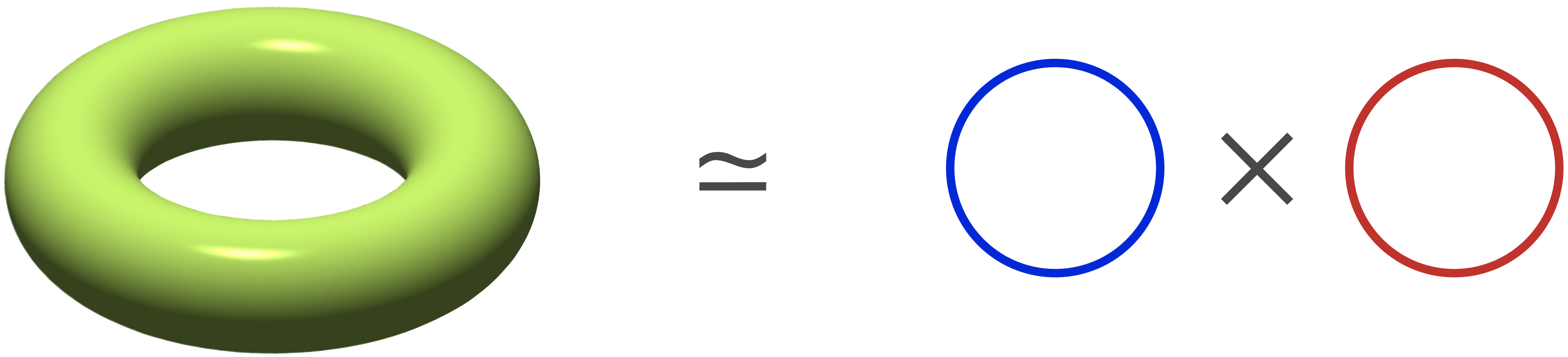


$T \simeq S^1 \times S^1$



`t2c : T → S¹ × S¹`

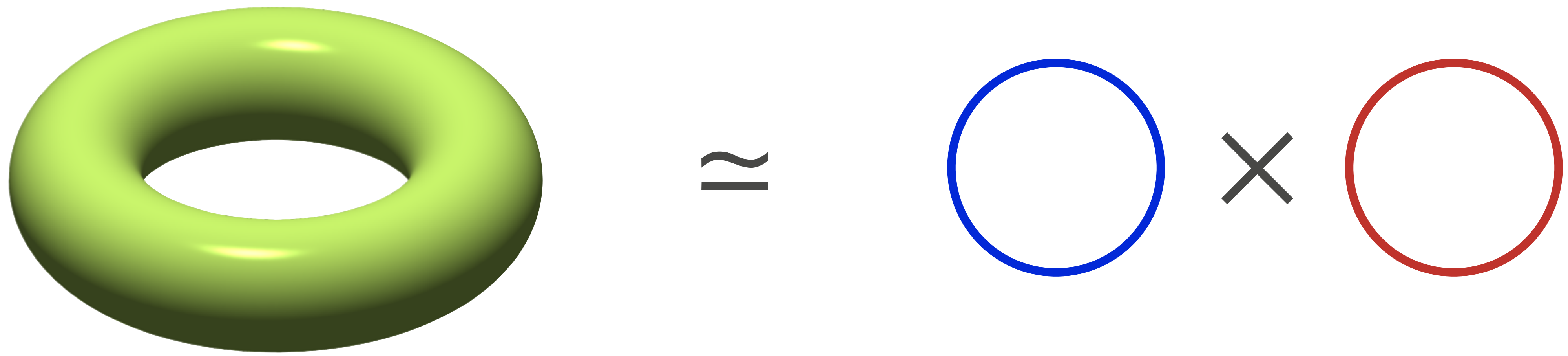
$T \simeq S^1 \times S^1$



$$t2c : T \to S^1 \times S^1$$

$$c2t : S^1 \times S^1 \to T$$

$T \simeq S^1 \times S^1$

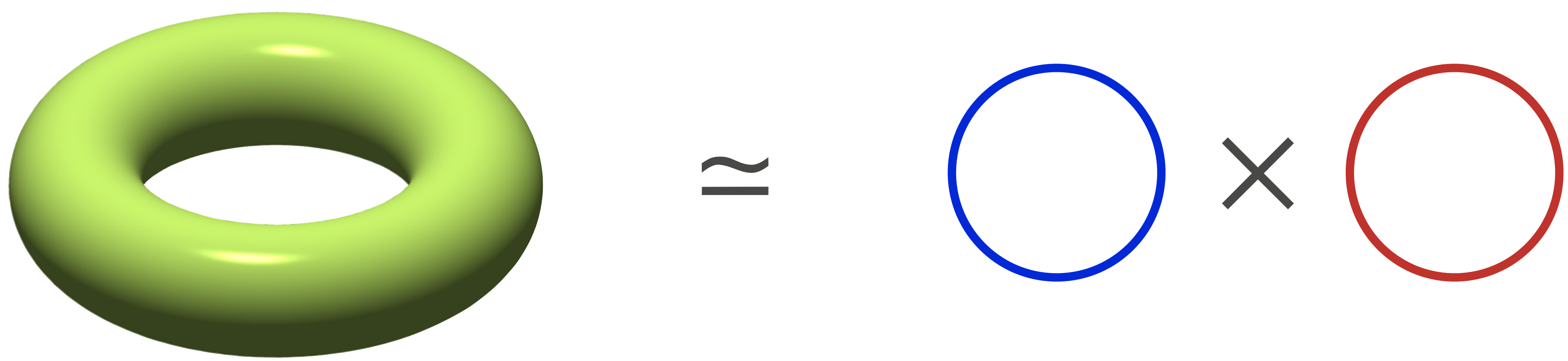


$$\mathrm{t2c} : T \to S^1 \times S^1$$

$$\mathrm{c2t} : S^1 \times S^1 \to T$$

$$\mathrm{c2t2c} : \Pi p{:}S^1{\times}S^1.\ \mathrm{t2c}\ (\mathrm{c2t}\ p) = p$$

$T \simeq S^1 \times S^1$

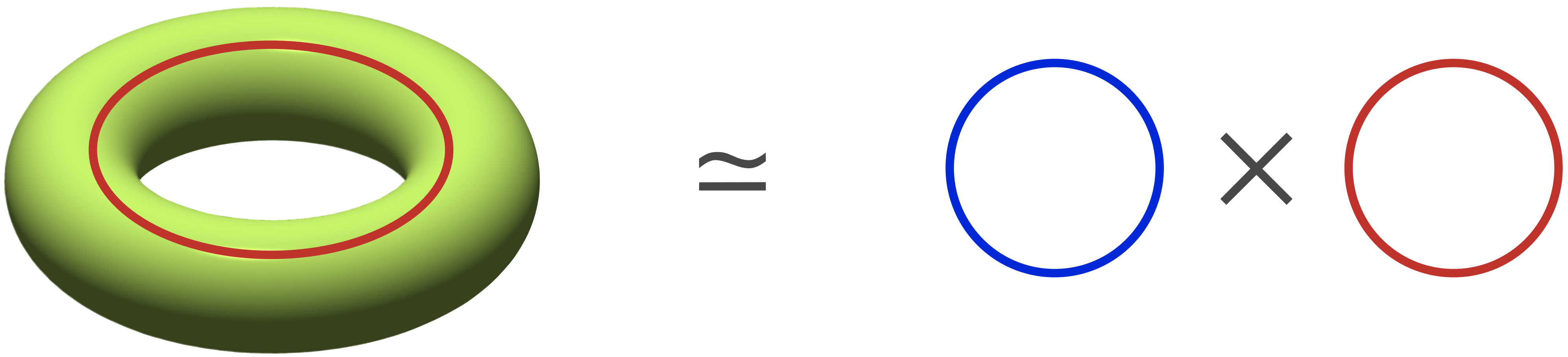


```
t2c : T → S¹ × S¹

c2t : S¹ × S¹ → T

c2t2c : Πp:S¹×S¹. t2c (c2t p) = p
t2c2t : Πt:T. c2t (t2c t) = t
```

$T \simeq S^1 \times S^1$

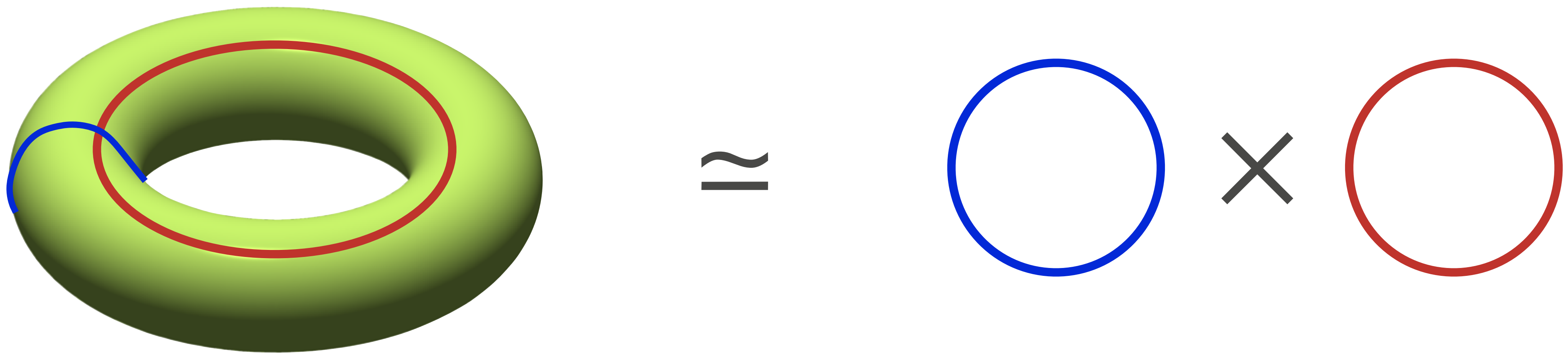


```
t2c : T → S¹ × S¹

c2t : S¹ × S¹ → T

c2t2c : Πp:S¹×S¹. t2c (c2t p) = p
t2c2t : Πt:T. c2t (t2c t) = t
```

$T \simeq S^1 \times S^1$



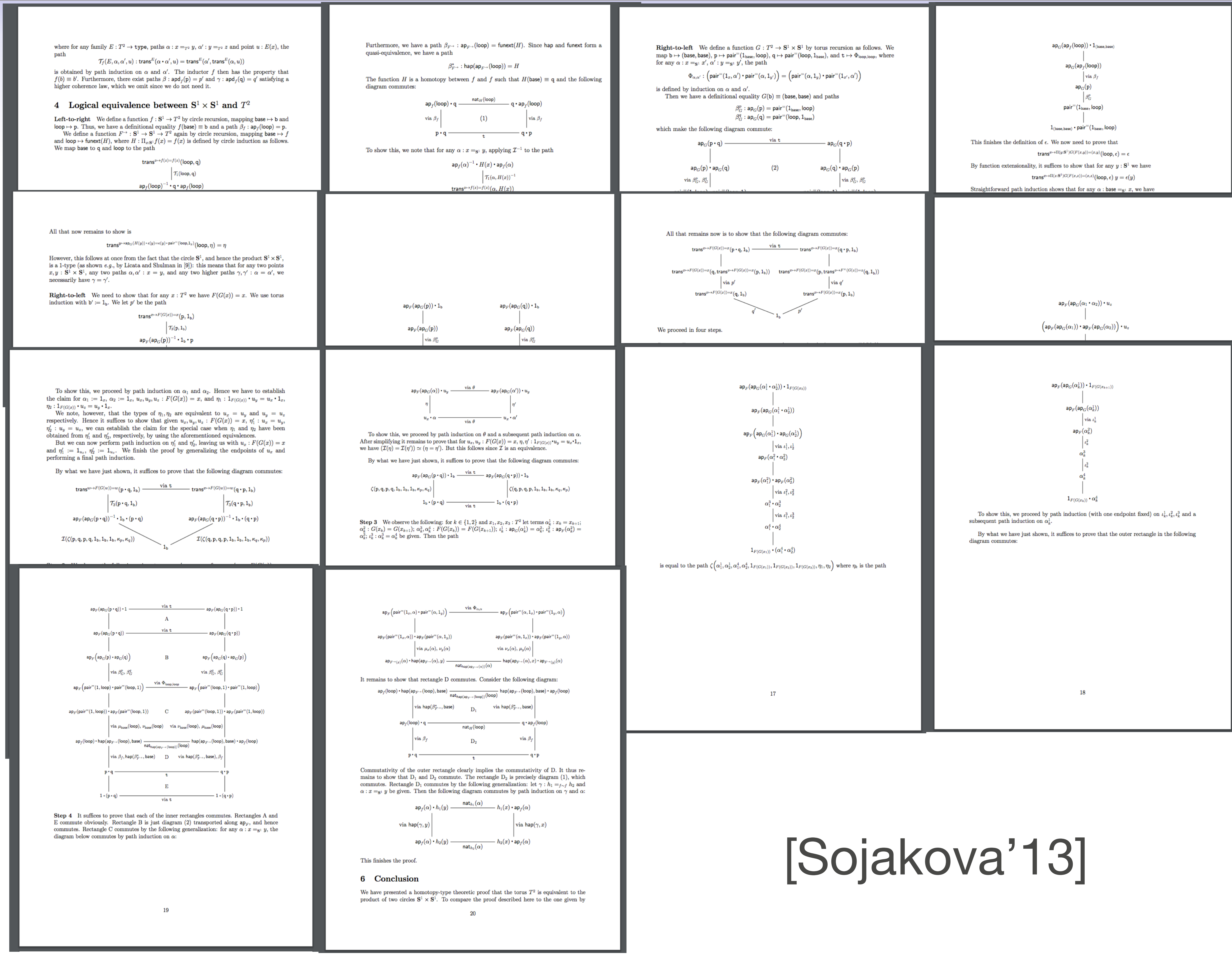
$$\texttt{t2c} : T \to S^1 \times S^1$$

$$\texttt{c2t} : S^1 \times S^1 \to T$$

$$\texttt{c2t2c} : \Pi p{:}S^1{\times}S^1.\ \texttt{t2c}\ (\texttt{c2t}\ p) = p$$

$$\texttt{t2c2t} : \Pi t{:}T.\ \texttt{c2t}\ (\texttt{t2c}\ t) = t$$

[Sojakova'13]



[Sojakova'13]

Cubical Methods

Models

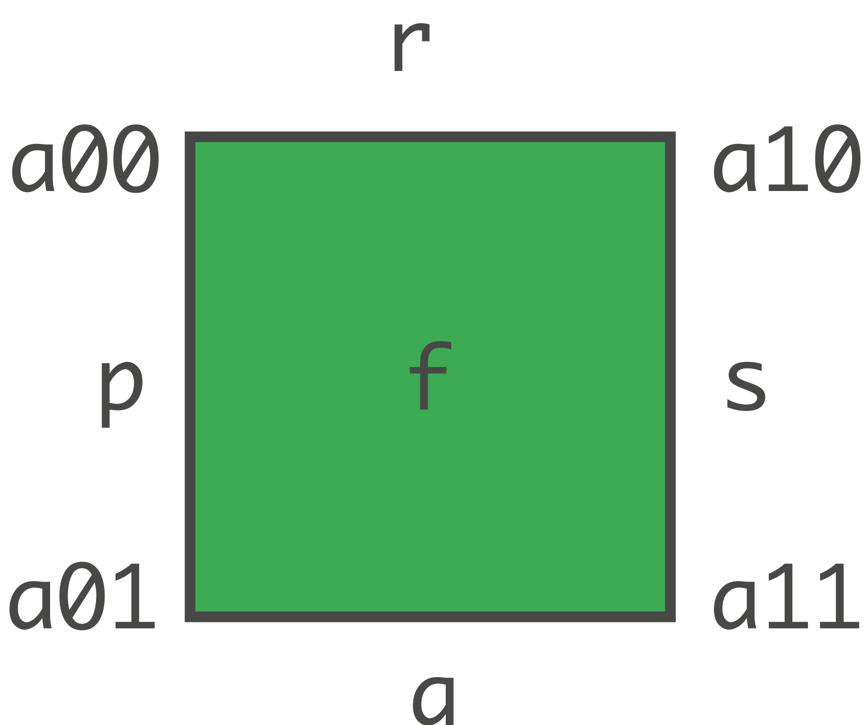
- Bezem, Coquand, Huber ‘13
- Cohen, Coquand, Huber, Mörtberg
- Awodey, Frey, Hofstra
- Gambino, Sattler
- Orton, Pitts
- Angiuli, Cavallo, Favonia, Harper, Sterling

Syntax

- Cohen, Coquand, Huber, Mörtberg
- Isaev
- Brunerie, Licata

Cubical Types in MLTT

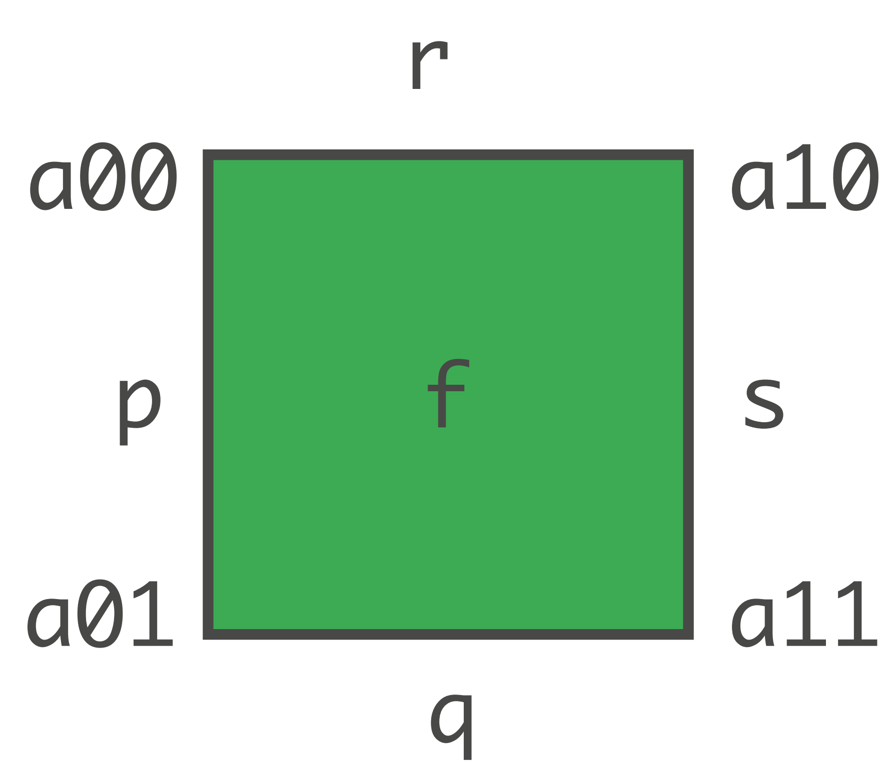
[L. and Brunerie'15]



Cubical Types in MLTT

[L. and Brunerie'15]

```
data Square {A : Type} {a00 : A} :
            {a01 a10 a11 : A}
            → a00 == a01
            → a00 == a10
            → a01 == a11
            → a10 == a11 → Type where
  id : Square id id id id
```

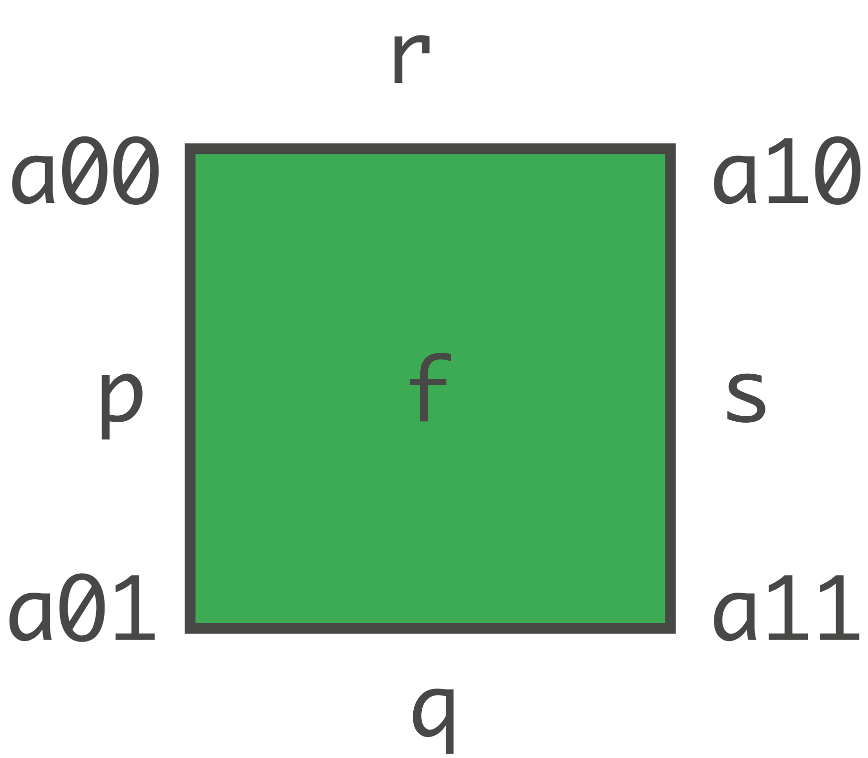


Cubical Types in MLTT

[L. and Brunerie'15]

```
data Square {A : Type} {a00 : A} :
            {a01 a10 a11 : A}
            → a00 == a01
            → a00 == a10
            → a01 == a11
            → a10 == a11 → Type where
  id : Square id id id id

Square' p q r s = (p;r = q;s)
```



Cubical Types in MLTT

[L. and Brunerie'15]

```
t2c : T → S^1 × S^1
t2c = T-rec (base, base) (pair-line loop id) (pair-line id loop) (pair-square hrefl-square vrefl-square)

c2t-square-and-cube : Σ λ (s : Square p (ap (S^1-rec a q) loop) (ap (S^1-rec a q) loop) p) →
                              Cube s f hrefl-square βsquare βsquare hrefl-square
c2t-square-and-cube = (fill-cube-left _ _ _ _ _)
c2t-loop-homotopy = S^1-elimo _ p (in-PathOver-= (fst c2t-square-and-cube))
c2t' : S^1 → S^1 → T
c2t' = S^1-rec (S^1-rec a q) (λ≃ c2t-loop-homotopy)
c2t : S^1 × S^1 → T
c2t (x,y) = c2t' x y

c2t'-β : Σ λ (c2t'-loop-2 : Square (ap (c2t' base) loop) id id q) →
         Σ λ (c2t'-loop-1 : Square (ap (λ x → c2t' x base) loop) id id p) →
             Cube (apdo-ap c2t' loop loop) f c2t'-loop-1 c2t'-loop-2 c2t'-loop-2 c2t'-loop-1
c2t'-β = _,_,
  out-SquareOver-= (apdo-by-equals _ _ _ (λ≃ (λ y → ap-o (λ f → f y) c2t' loop))) ·-cube-h
  out-SquareOver-= (apdo-by-equals _ _ _ (λ≃ (λ y → ap (ap (λ f → f y))
       (S^1.βloop/rec (S^1-rec a q) (λ≃ c2t-loop-homotopy))))) ·-cube-h
  out-SquareOver-= (apdo-by-equals _ _ _ (λ≃ (λ y → Π≃β c2t-loop-homotopy))) ·-cube-h
  degen-cube-h (ap out-PathOver-= (S^1.βloop/elimo _ p (in-PathOver-= (fst c2t-square-and-cube)))) ·-cube-h
  degen-cube-h (IsEquiv.β (snd out-PathOver-=-eqv) _) ·-cube-h
  (snd c2t-square-and-cube)

t2c2t : (x : T) → Path (c2t (t2c x)) x
t2c2t = T-elim _ id (in-PathOver-= (square-symmetry p-case)) (in-PathOver-= (square-symmetry q-case))
                (in-SquareOver-=
                   (whisker-cube (! (IsEquiv.β (snd out-PathOver-=-eqv) _)) (! (IsEquiv.β (snd out-PathOver-=-eqv) _))
                                 (! (IsEquiv.β (snd out-PathOver-=-eqv) _)) id id (! (IsEquiv.β (snd out-PathOver-=-eqv) _))
                                 (cube-symmetry-left-to-top f-case))) where
  p-case = _
  q-case = _
  f-case : Cube (ap-square (λ z → c2t (t2c z)) f) (ap-square (λ z → z) f) p-case q-case q-case p-case
  f-case = ap-square-o c2t t2c f ·-cube-h
           ap-cube c2t βfcube ·-cube-h
           apdo-ap-cube-hv c2t' loop loop ·-cube-h
           snd (snd c2t'-β) ·-cube-h
           ap-square-id! f

c2t2c : (x y : S^1) → Path (t2c (c2t' x y)) (x,y)
c2t2c = S^1-elimo _ (S^1-elimo _ id (in-PathOver-= (square-symmetry loop2-case)))
             (coe (! PathOverΠ-NDdomain)
               (in-PathOver-= od1 (S^1-elimo _ (square-symmetry loop1-case)
                                     (in-PathOver-Square _
                                       (whisker-cube id id red id id red
                                         (cube-symmetry-left-to-top looploop-case)))))) where
  loop1-case = _
  loop2-case = _
  looploop-case : Cube (apdo-ap (λ x y → t2c (c2t' x y)) loop loop) (apdo-ap _,_ loop loop)
                       loop1-case loop2-case loop2-case loop1-case
  looploop-case = apdo-ap-o t2c c2t' loop loop ·-cube-h
                  ap-cube t2c (snd (snd c2t'-β)) ·-cube-h
                  βfcube ·-cube-h
                  ap-square-id! _ ·-cube-h
                  apdo-ap-cube-hv _,_ loop loop

  red = ! (ap out-PathOver-= (S^1.βloop/elimo _ id (in-PathOver-= (square-symmetry loop2-case))) ·
          IsEquiv.β (snd out-PathOver-=-eqv) (square-symmetry loop2-case))
```

Cubical Syntax

Cubical Syntax

- Extends MLTT with abstract interval variables (endpoints 0 and 1) and new constructs using them (univalence and HITs)

Cubical Syntax

- Extends MLTT with abstract interval variables (endpoints 0 and 1) and new constructs using them (univalence and HITs)
- Models in cubical sets (Q: other ∞-toposes?)

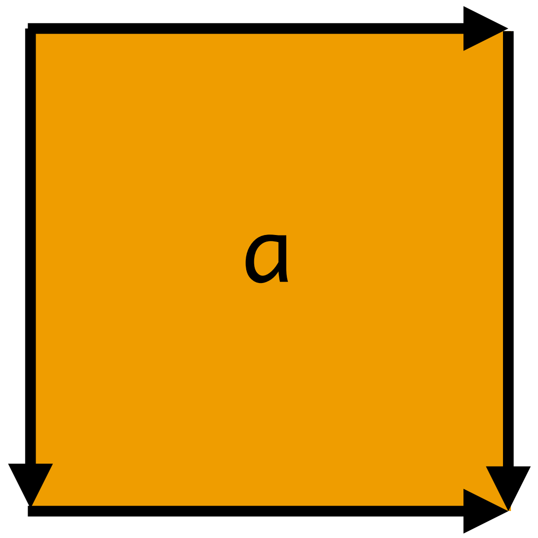
Faces

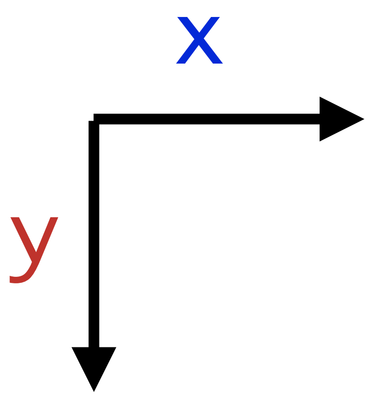
$$x{:}I \vdash a : A$$

$$\xrightarrow{x} \qquad a\langle 0/x\rangle \xrightarrow{a} a\langle 1/x\rangle$$

Faces

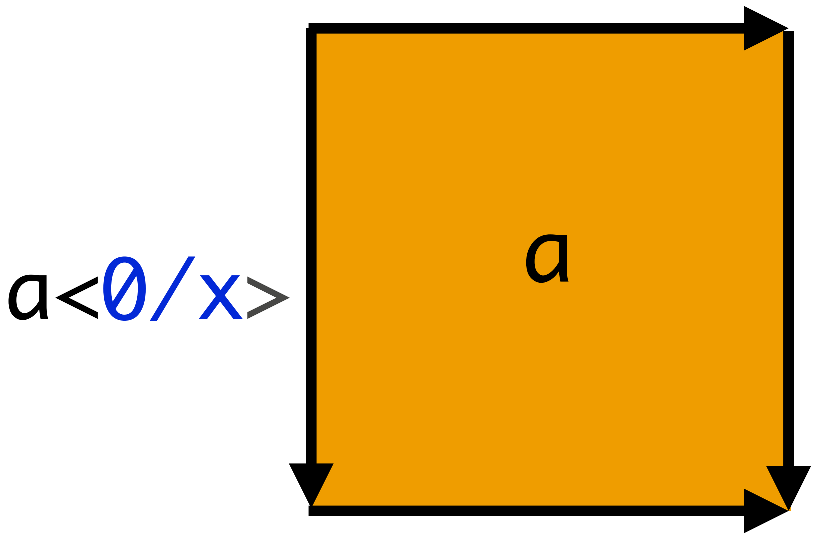
$$x{:}I,\ y{:}I \vdash a : A$$

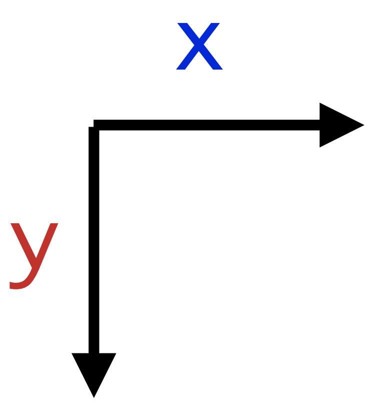




Faces

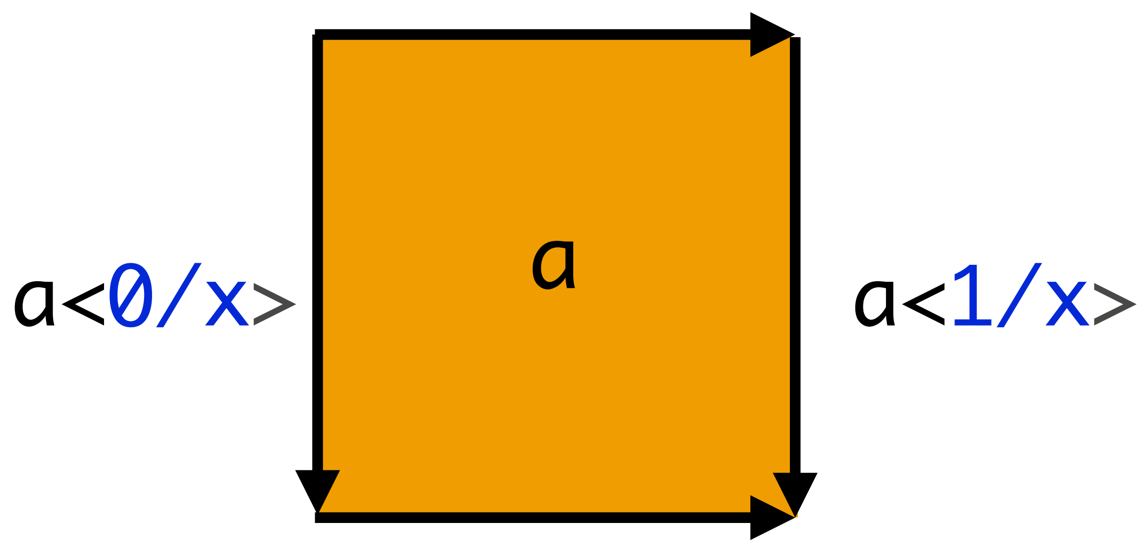
x:I, y:I ⊢ a : A

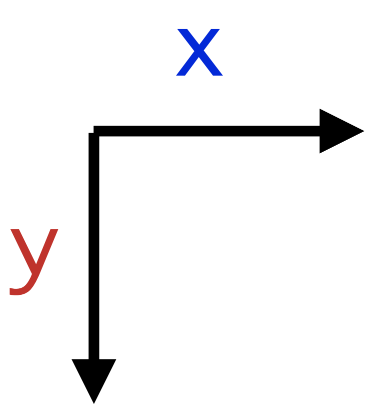




Faces

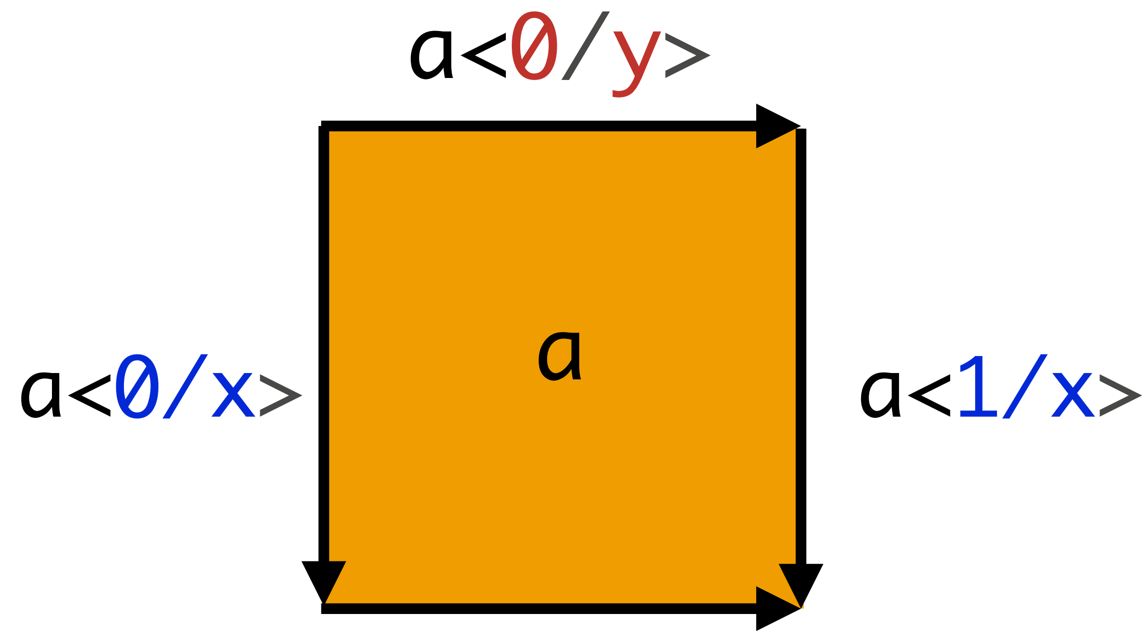
$x{:}I,\ y{:}I \vdash a : A$

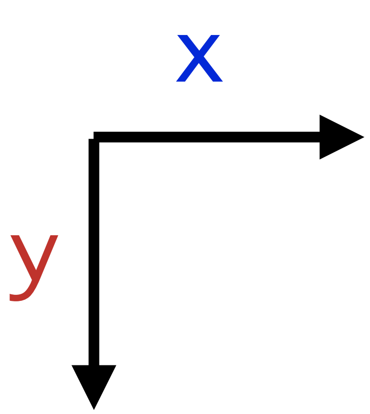




Faces

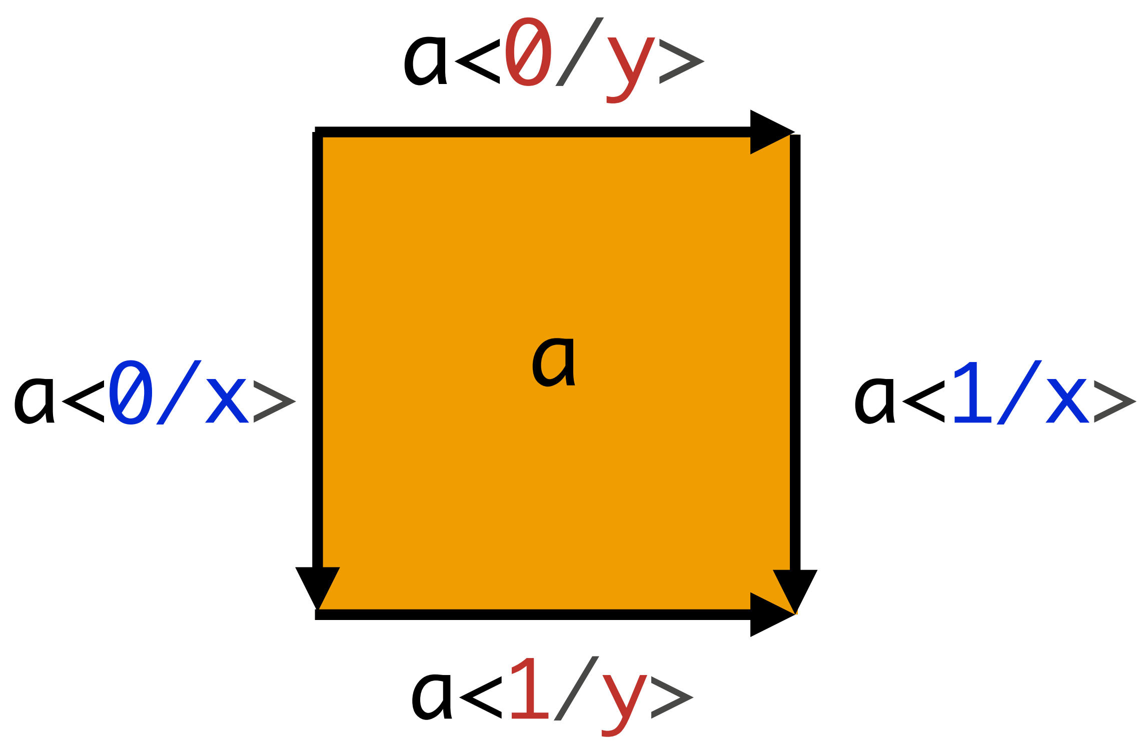
$$x{:}\mathrm{I},\ y{:}\mathrm{I} \vdash a : A$$

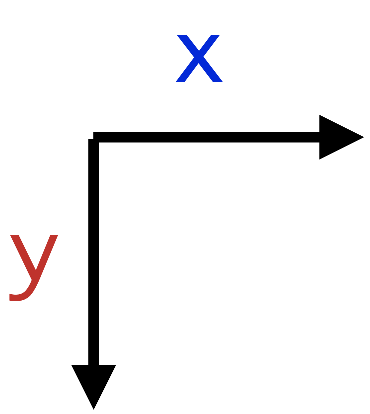




Faces

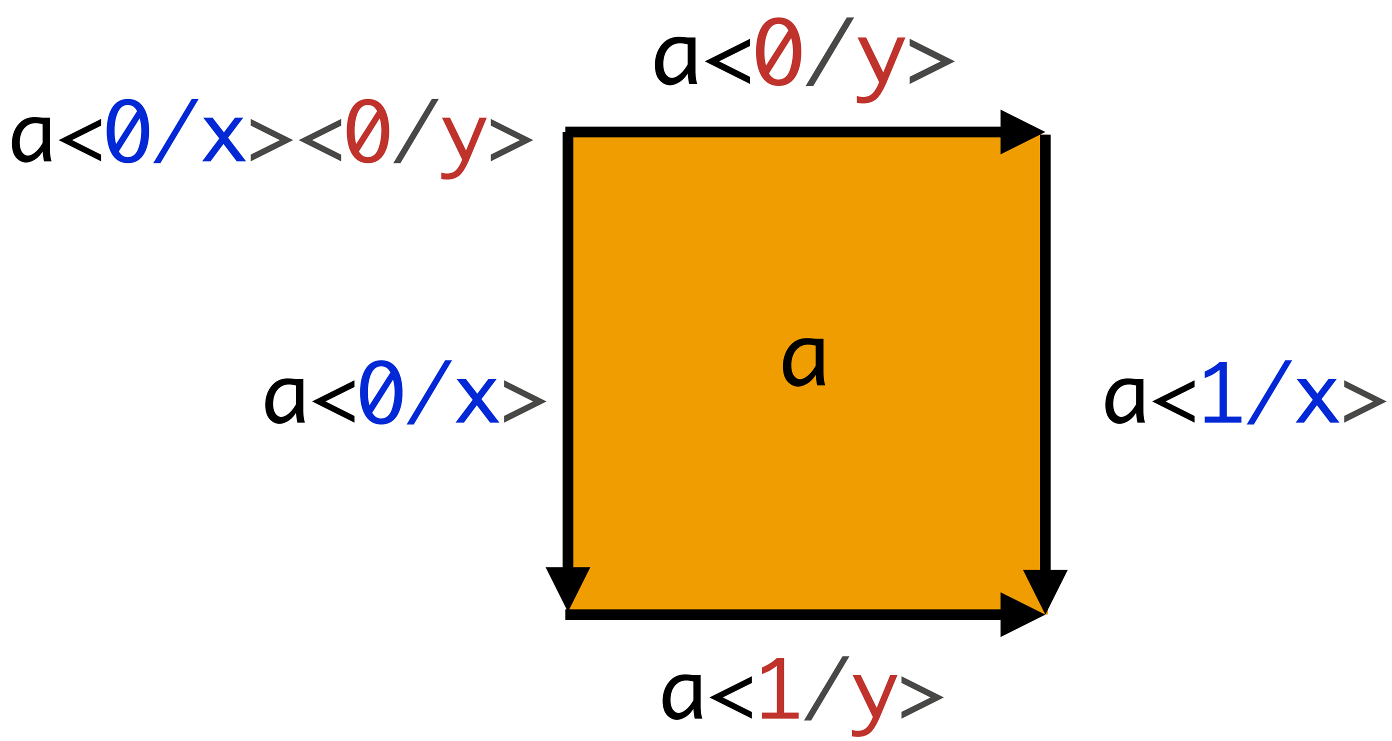
$$x{:}I,\ y{:}I \vdash a : A$$

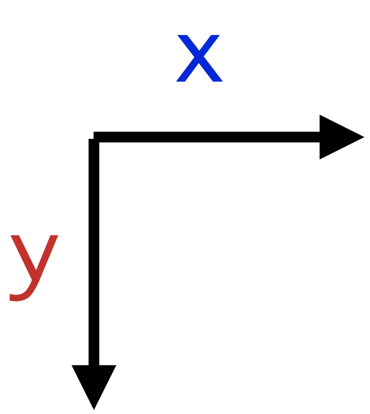




Faces

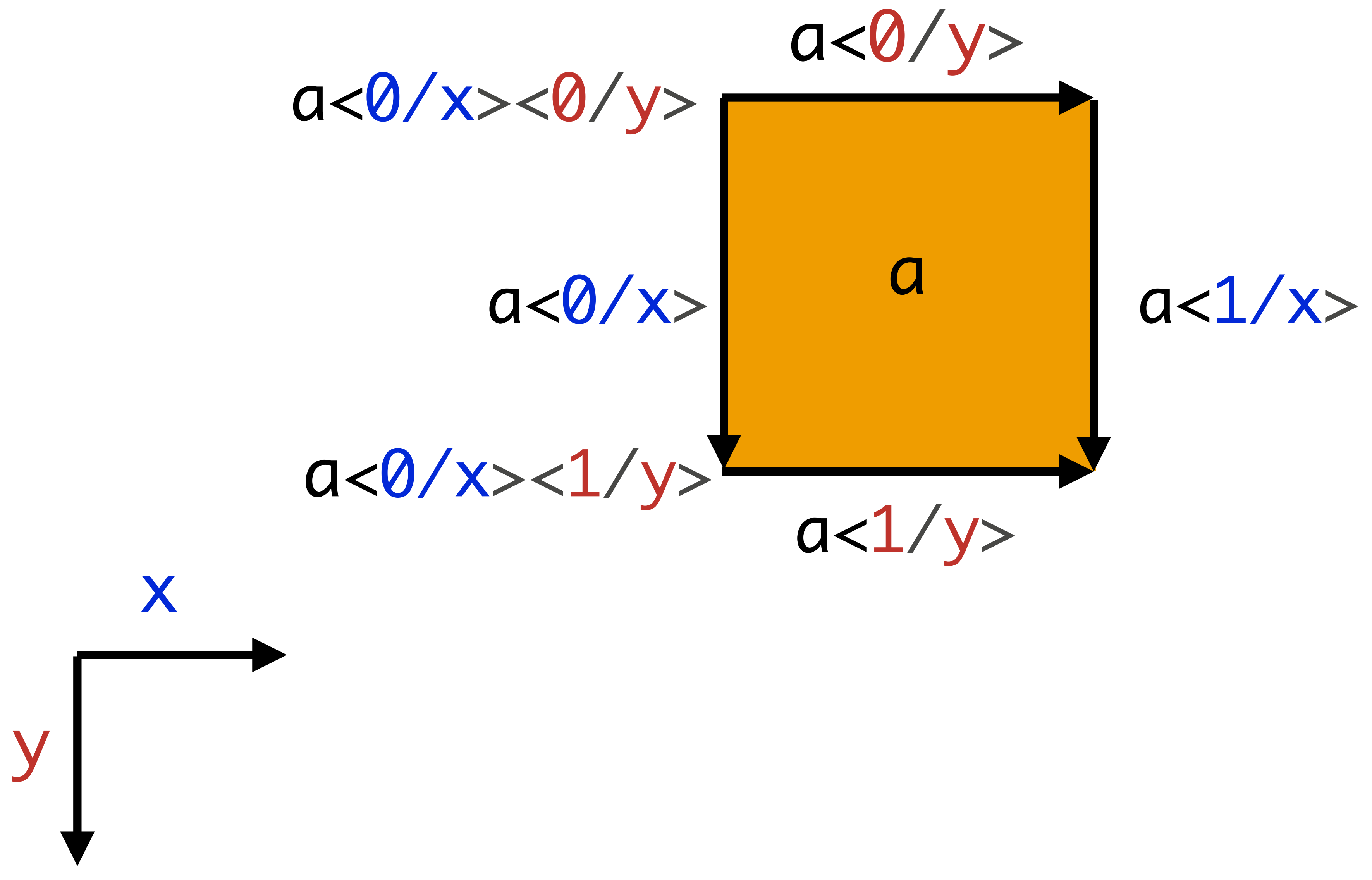
$$x{:}\mathrm{I},\ y{:}\mathrm{I} \vdash a : A$$





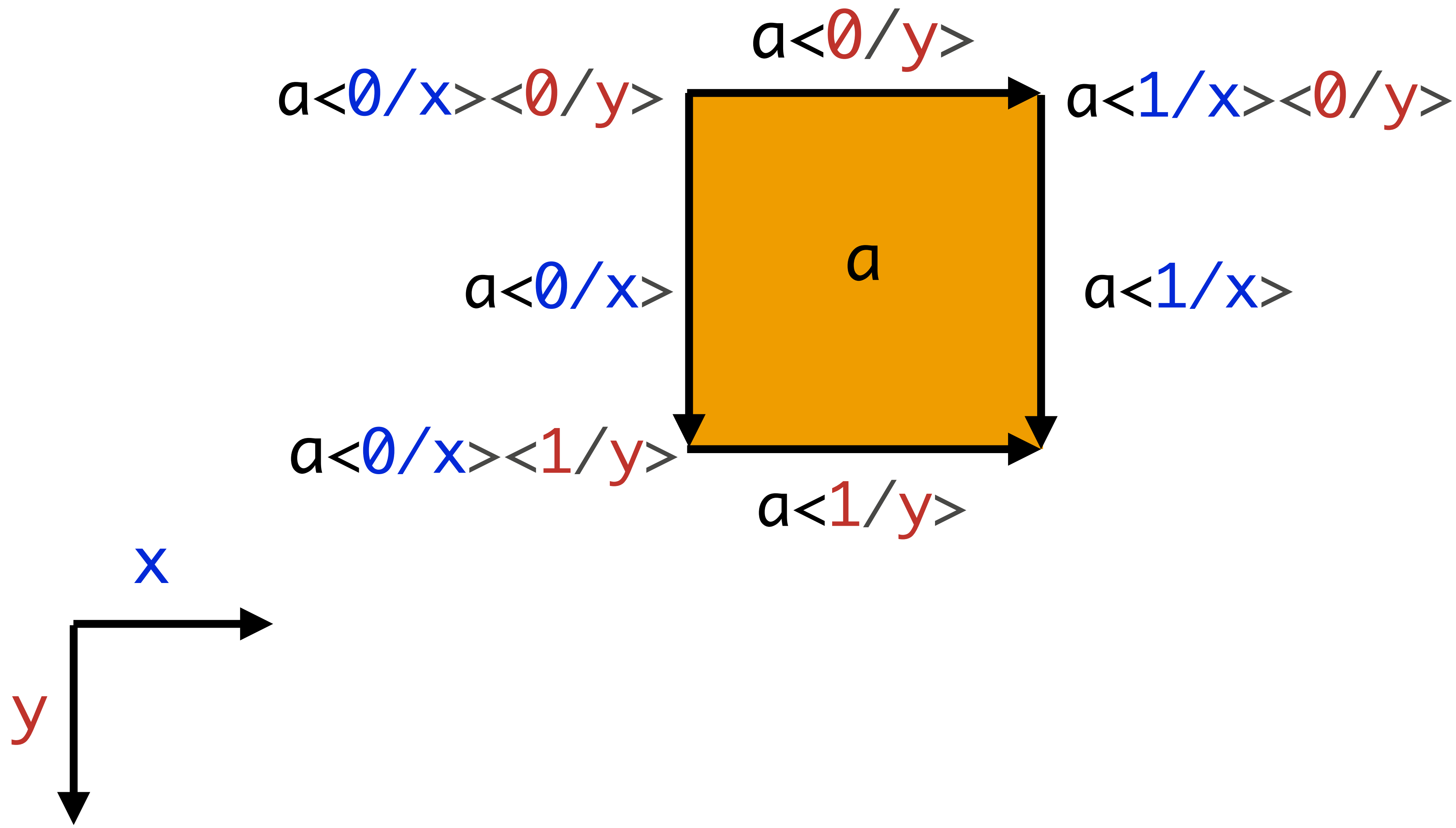
Faces

x:I, y:I ⊢ a : A



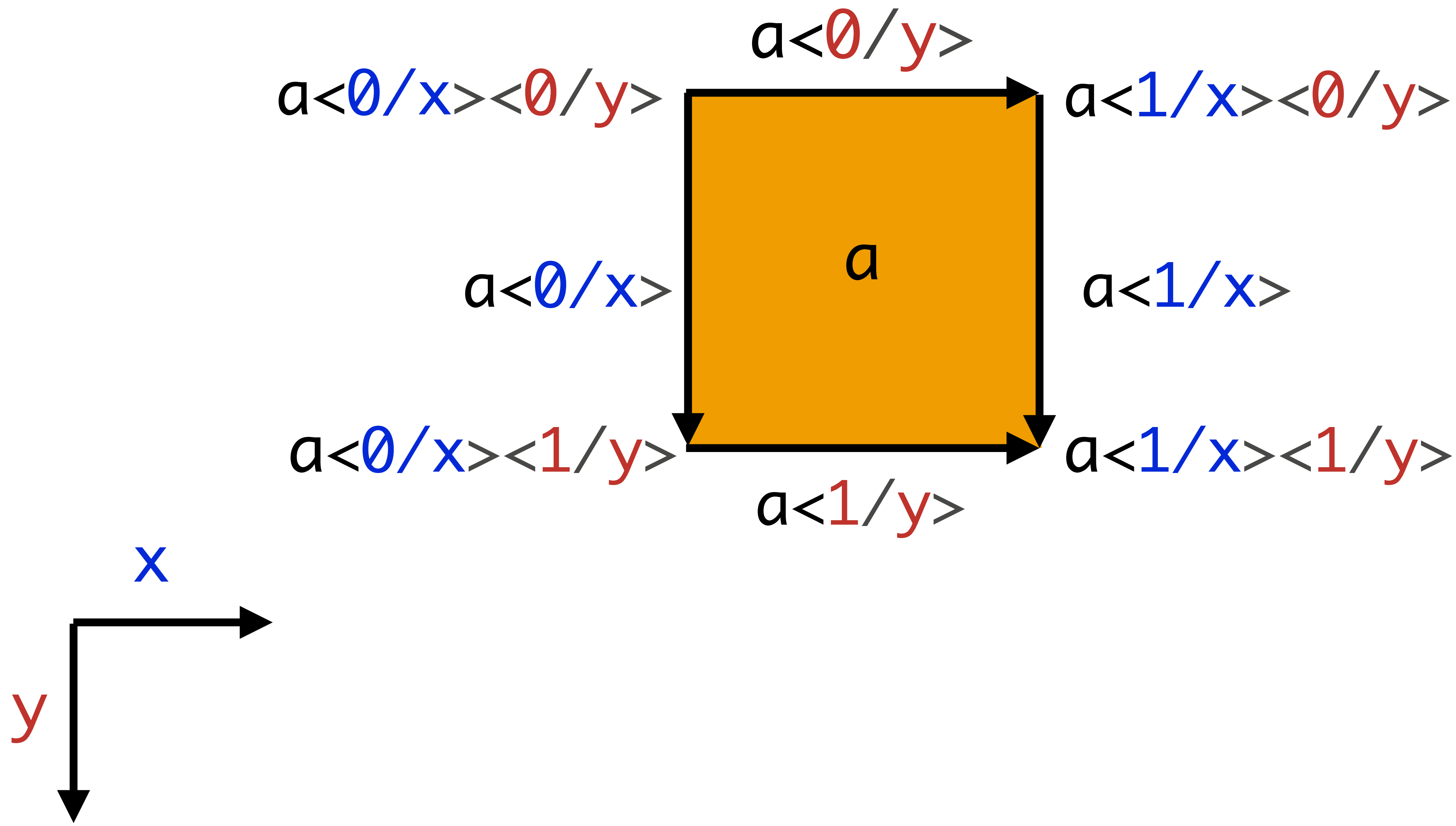
Faces

x:I, y:I ⊢ a : A



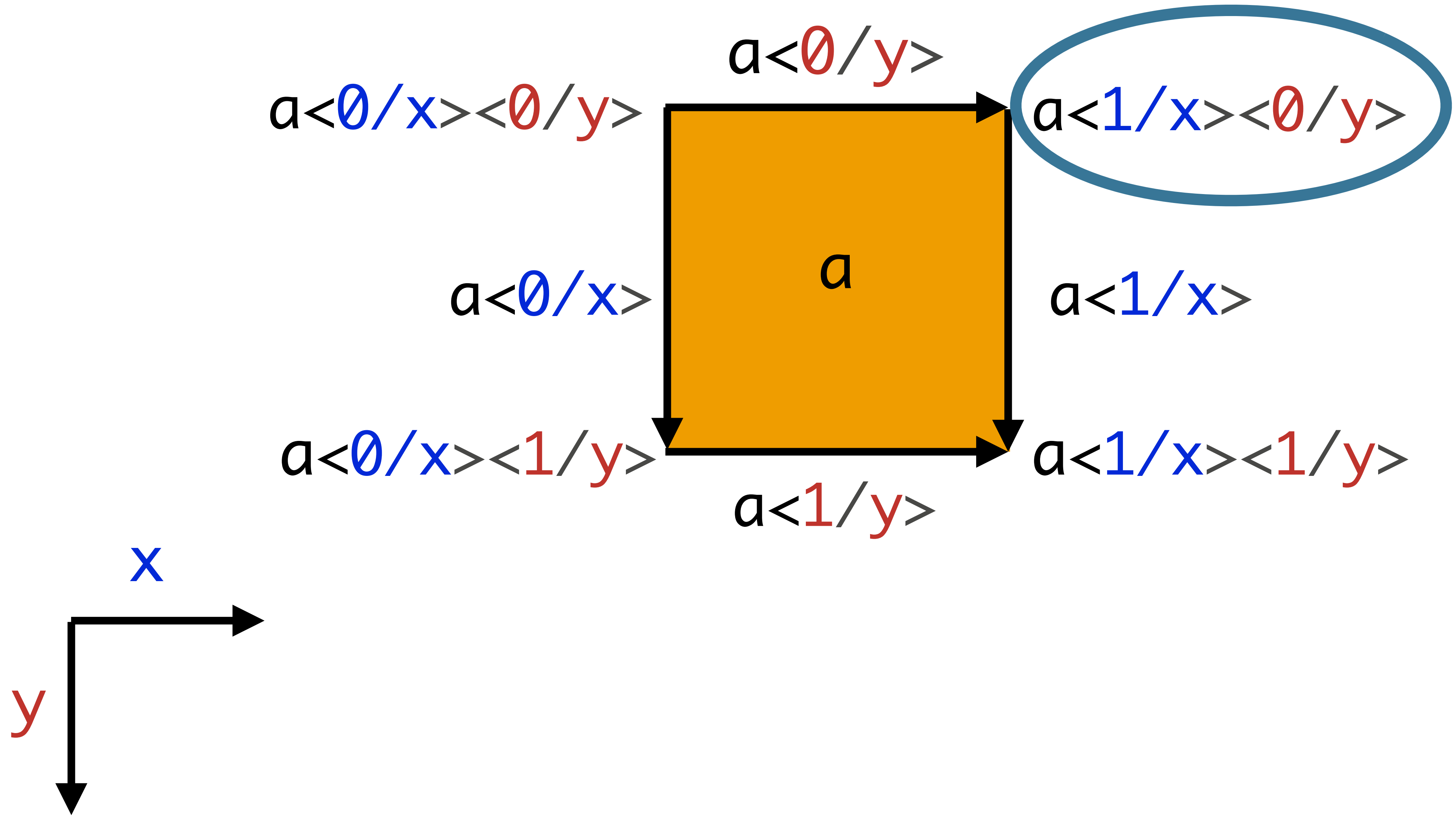
Faces

$$x{:}I,\ y{:}I \vdash a : A$$



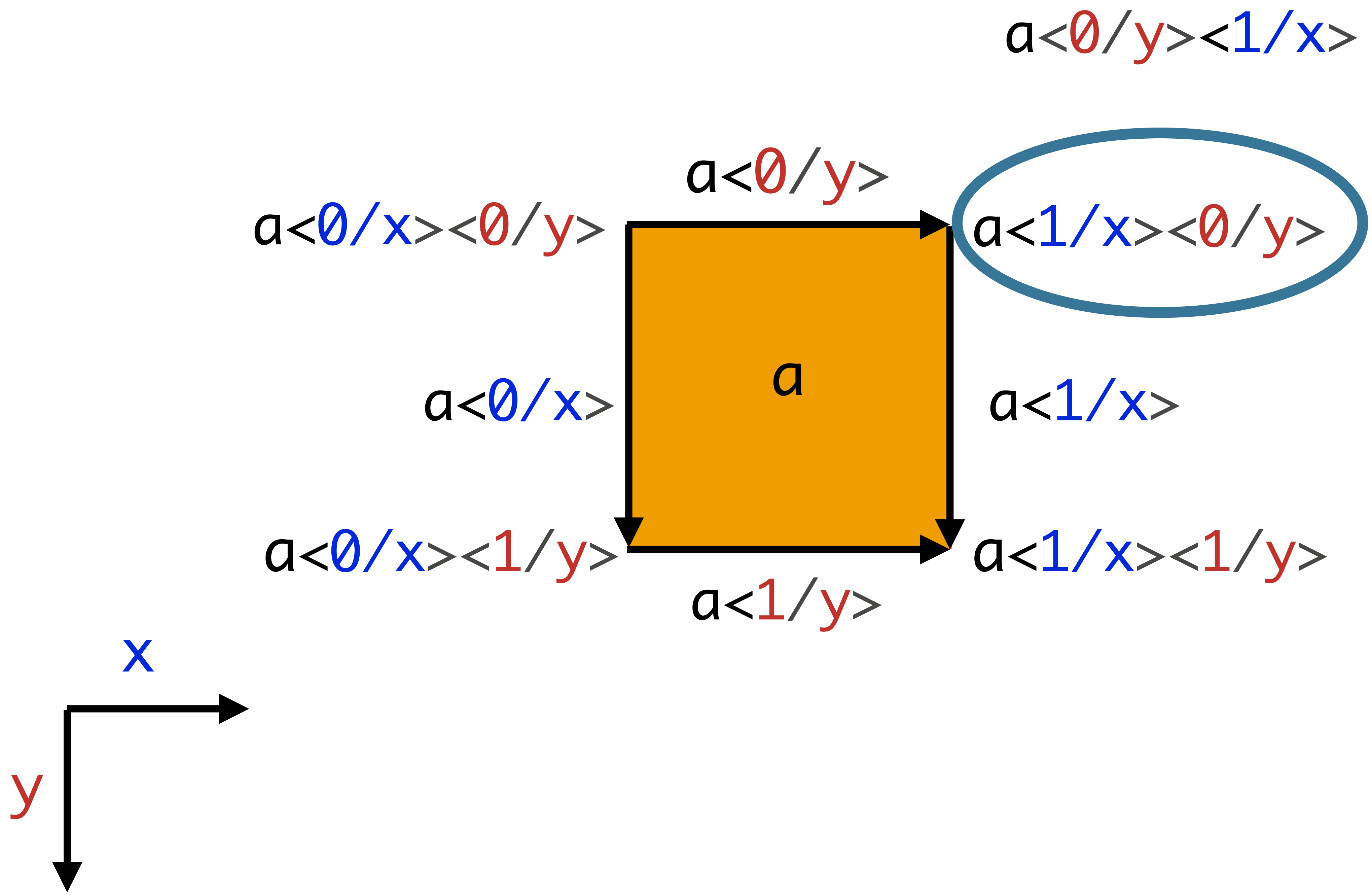
Faces

x:I, y:I ⊢ a : A



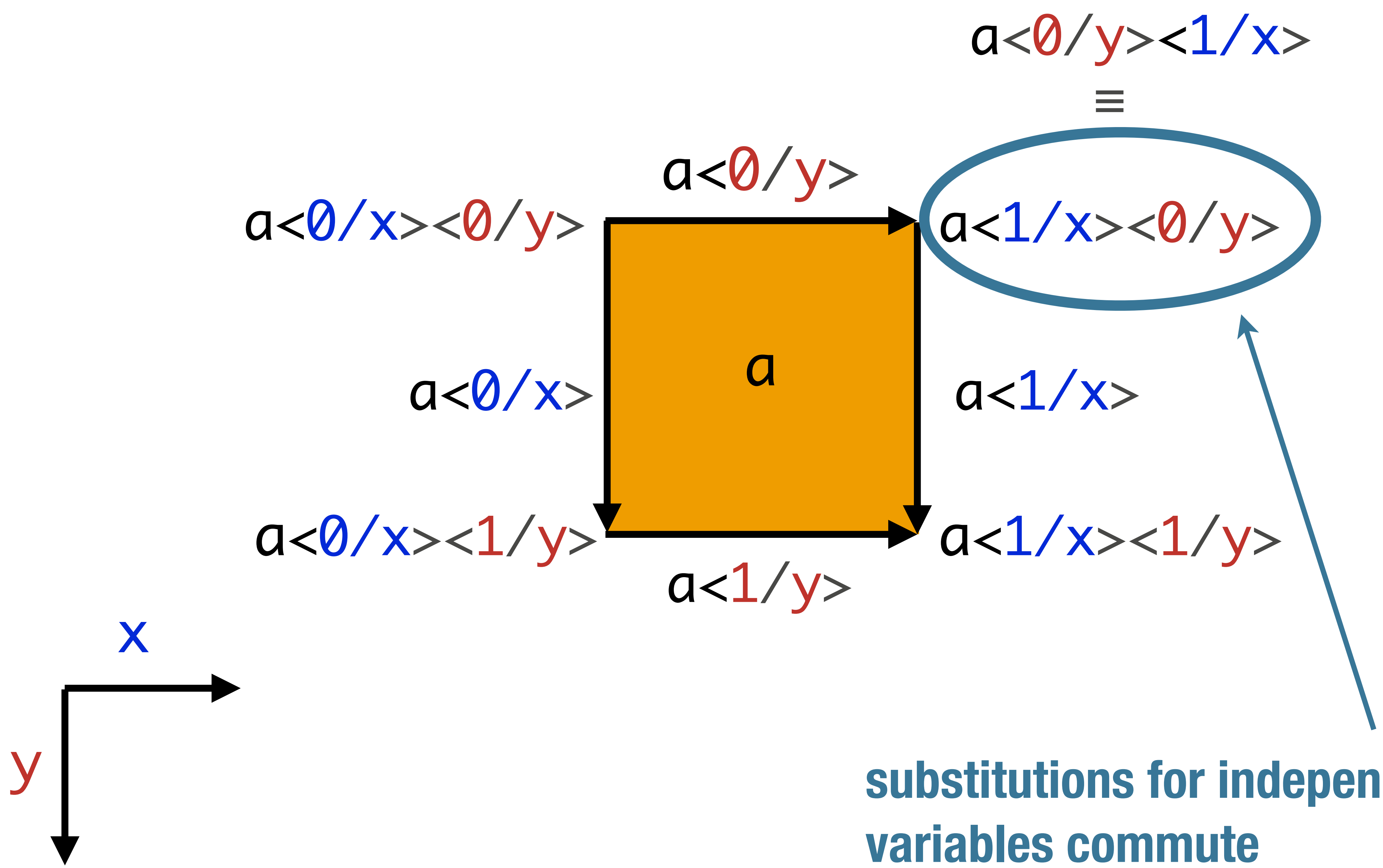
Faces

x:I, y:I ⊢ a : A



Faces

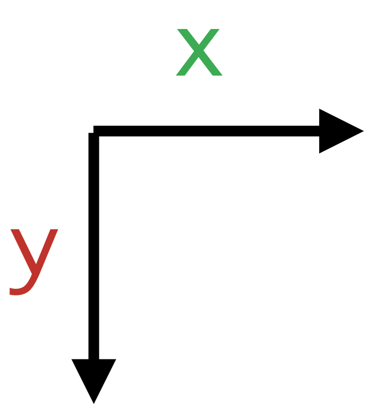
$$x{:}I,\ y{:}I \vdash a : A$$



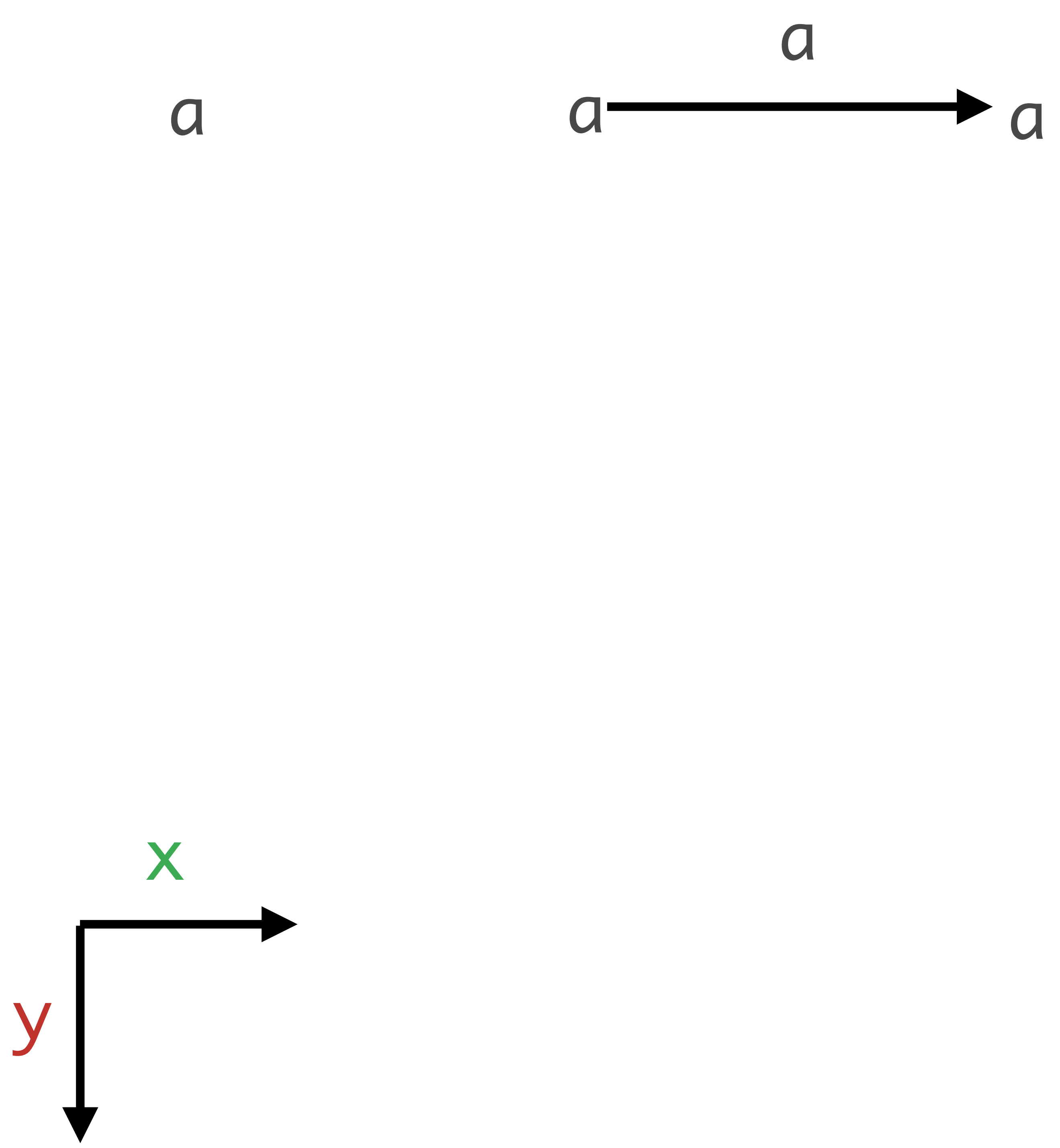
substitutions for independent variables commute

Degeneracies

a



Degeneracies

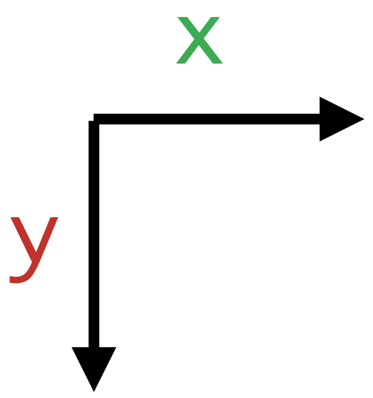


Degeneracies

a

$a \xrightarrow{a} a$

$a <0/x> \equiv a$



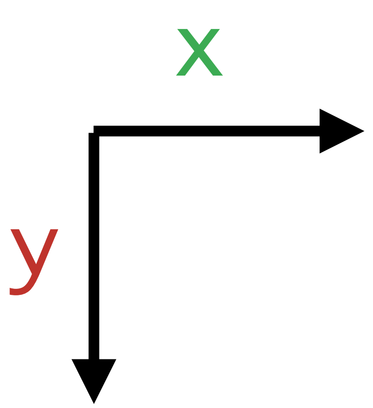
Degeneracies

a

$a \xrightarrow{a} a$

`a <0/x> ≡ a`

`a <1/x> ≡ a`



Degeneracies

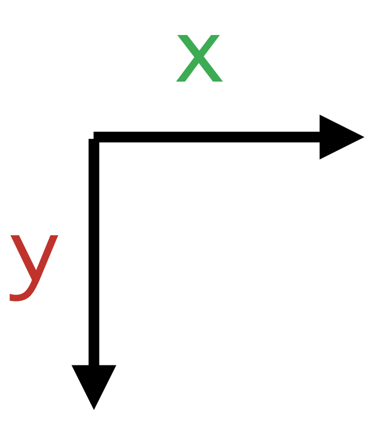
a

$a \xrightarrow{a} a$

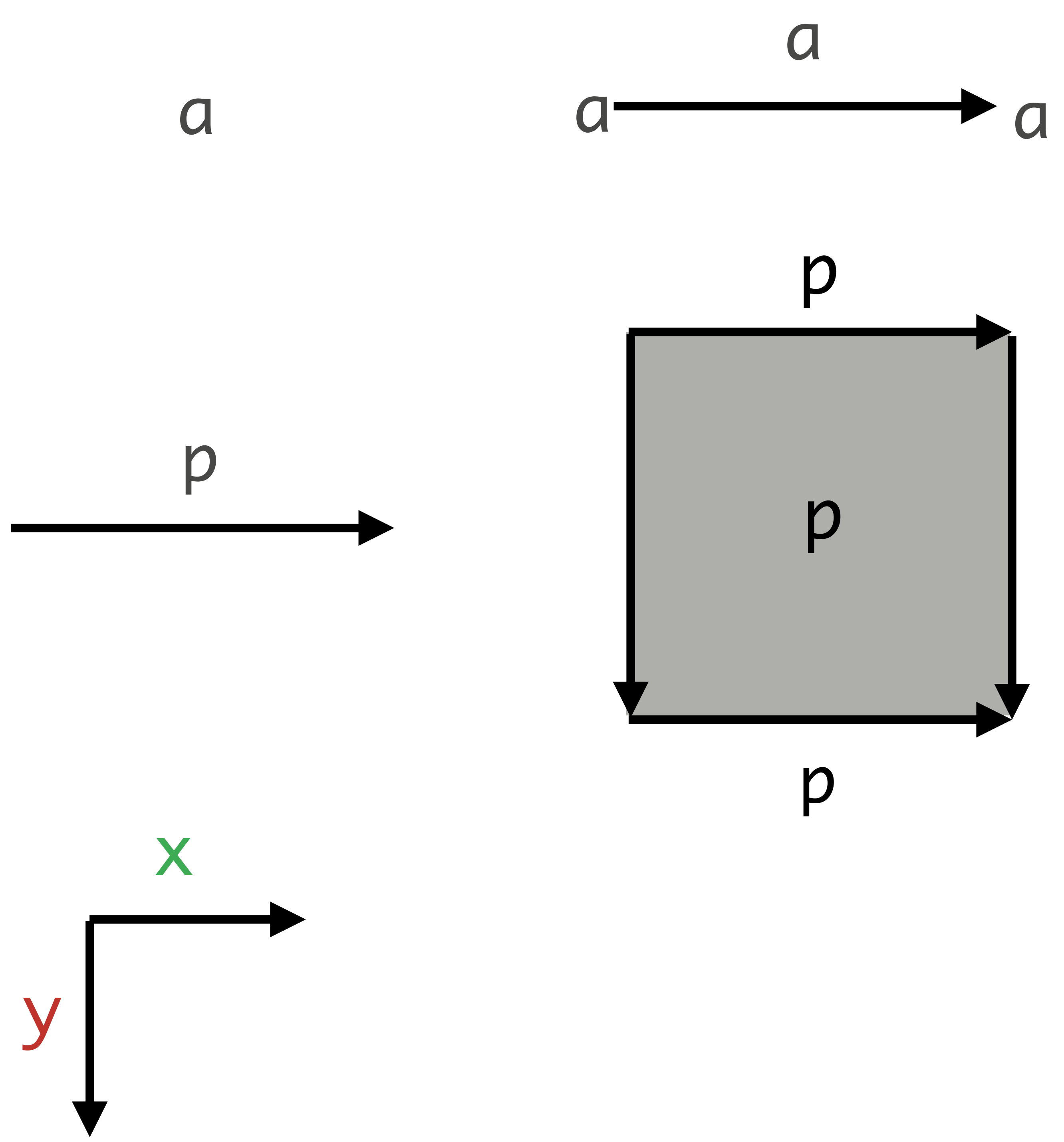
a <0/x> ≡ a

a <1/x> ≡ a

$\xrightarrow{p}$



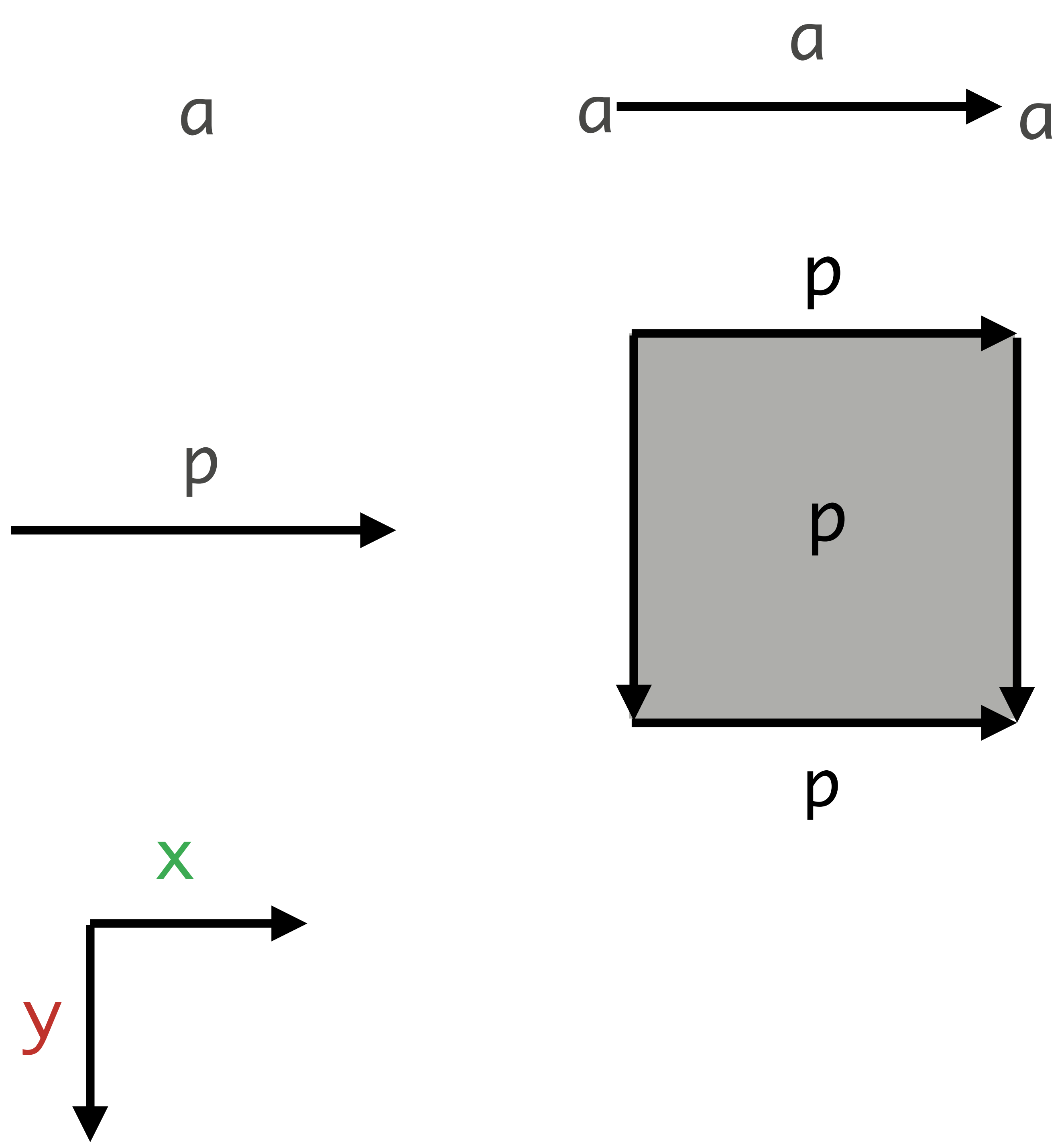
Degeneracies



$$a \langle 0/x \rangle \equiv a$$

$$a \langle 1/x \rangle \equiv a$$

Degeneracies



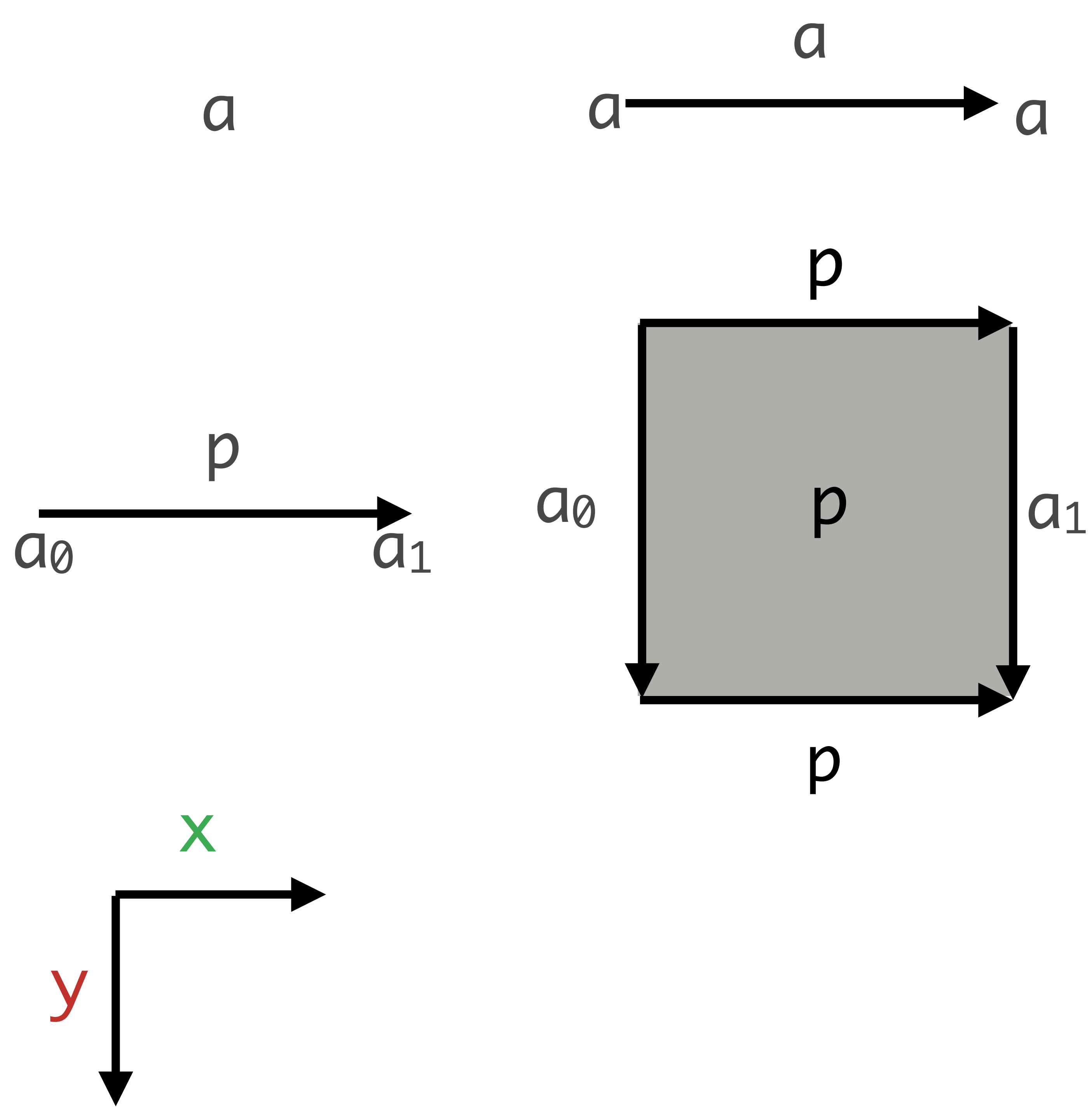
$a\langle 0/x\rangle \equiv a$

$a\langle 1/x\rangle \equiv a$

$p\langle 0/y\rangle \equiv p$

$p\langle 1/y\rangle \equiv p$

Degeneracies



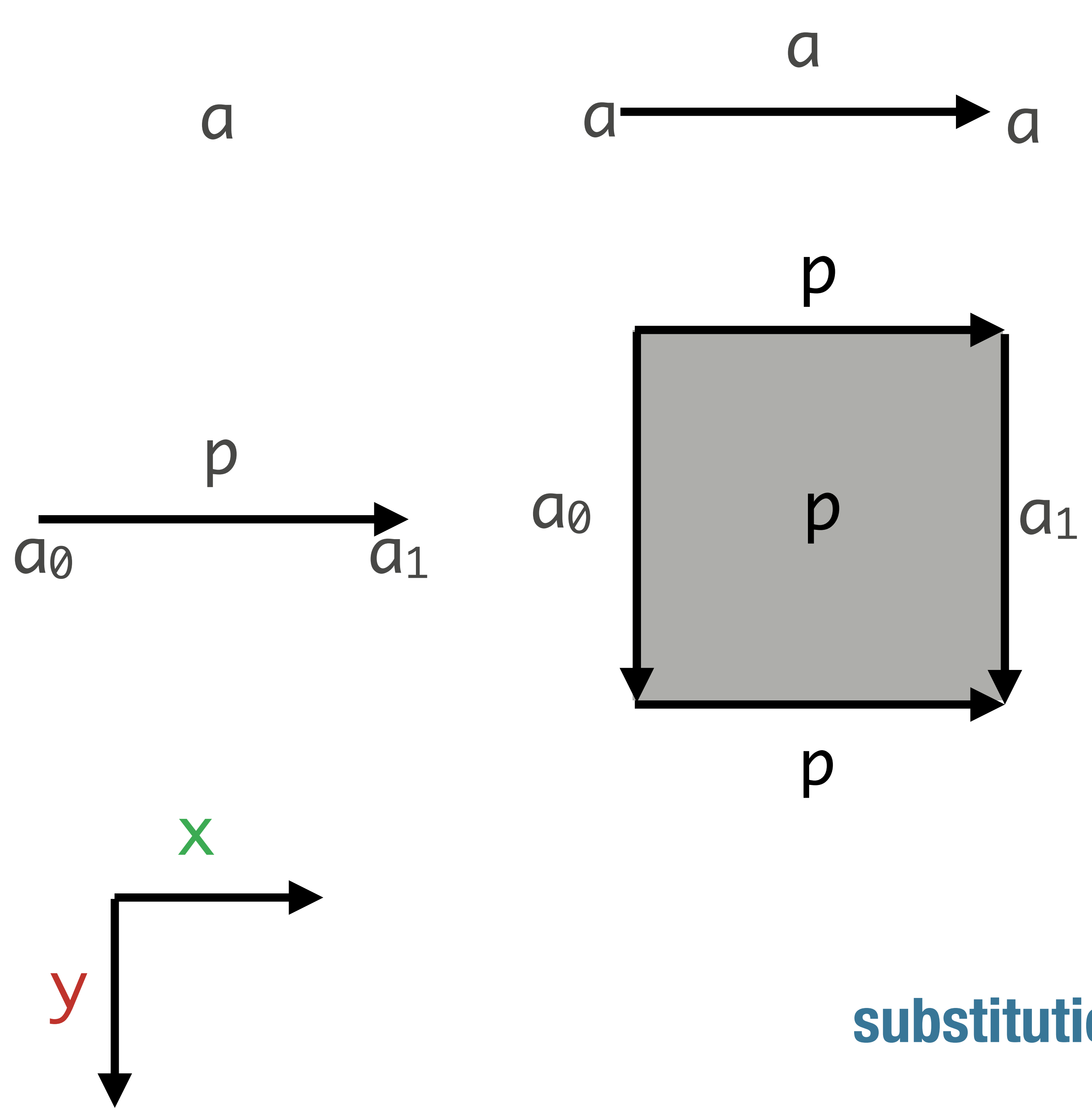
$$a\langle 0/x\rangle \equiv a$$

$$a\langle 1/x\rangle \equiv a$$

$$p\langle 0/y\rangle \equiv p$$

$$p\langle 1/y\rangle \equiv p$$

Degeneracies



$a\langle 0/x\rangle \equiv a$

$a\langle 1/x\rangle \equiv a$

$p\langle 0/y\rangle \equiv p$

$p\langle 1/y\rangle \equiv p$

substitution after weakening is identity

Diagonals

$$
\begin{tikzcd}
a\langle 0/x\rangle\langle 0/y\rangle \arrow[r, "a\langle 0/y\rangle"] \arrow[d, "a\langle 0/x\rangle"'] & a\langle 1/x\rangle\langle 0/y\rangle \arrow[d, "a\langle 1/x\rangle"] \\
a\langle 0/x\rangle\langle 1/y\rangle \arrow[r, "a\langle 1/y\rangle"'] & a\langle 1/x\rangle\langle 1/y\rangle
\end{tikzcd}
$$

Diagonals

$$
\begin{tikzcd}
a\langle 0/x\rangle\langle 0/y\rangle \arrow[r, "a\langle 0/y\rangle"] \arrow[d, "a\langle 0/x\rangle"'] & a\langle 1/x\rangle\langle 0/y\rangle \arrow[d, "a\langle 1/x\rangle"] \\
a\langle 0/x\rangle\langle 1/y\rangle \arrow[r, "a\langle 1/y\rangle"'] & a\langle 1/x\rangle\langle 1/y\rangle
\end{tikzcd}
$$

$a\langle x/y\rangle$ is a line ({x}-cube)

Diagonals

$$
\begin{tikzcd}
a\langle 0/x\rangle\langle 0/y\rangle \arrow[r, "a\langle 0/y\rangle"] \arrow[d, "a\langle 0/x\rangle"'] & a\langle 1/x\rangle\langle 0/y\rangle \arrow[d, "a\langle 1/x\rangle"] \\
a\langle 0/x\rangle\langle 1/y\rangle \arrow[r, "a\langle 1/y\rangle"'] & a\langle 1/x\rangle\langle 1/y\rangle
\end{tikzcd}
$$

$a\langle x/y\rangle$ is a line ({x}-cube)

$$a\langle x/y\rangle\langle 0/x\rangle \equiv a\langle 0/x\rangle\langle 0/y\rangle$$

Diagonals

$$\begin{tikzcd}[column sep=huge, row sep=huge]
a<0/x><0/y> \arrow[r, "a<0/y>"] \arrow[d, "a<0/x>"'] & a<1/x><0/y> \arrow[d, "a<1/x>"] \\
a<0/x><1/y> \arrow[r, "a<1/y>"'] & a<1/x><1/y>
\end{tikzcd}$$

`a<x/y>` is a line ({x}-cube)

`a<x/y><0/x> ≡ a<0/x><0/y>`

Diagonals

$$\begin{tikzcd}[column sep=huge, row sep=huge]
a<0/x><0/y> \arrow[r, "a<0/y>"] \arrow[d, "a<0/x>"'] & a<1/x><0/y> \arrow[d, "a<1/x>"] \\
a<0/x><1/y> \arrow[r, "a<1/y>"'] & a<1/x><1/y>
\end{tikzcd}$$

$a<x/y>$ is a line ({x}-cube)

$$a<x/y><0/x> \equiv a<0/x><0/y>$$

$$a<x/y><1/x> \equiv a<1/x><1/y>$$

Diagonals

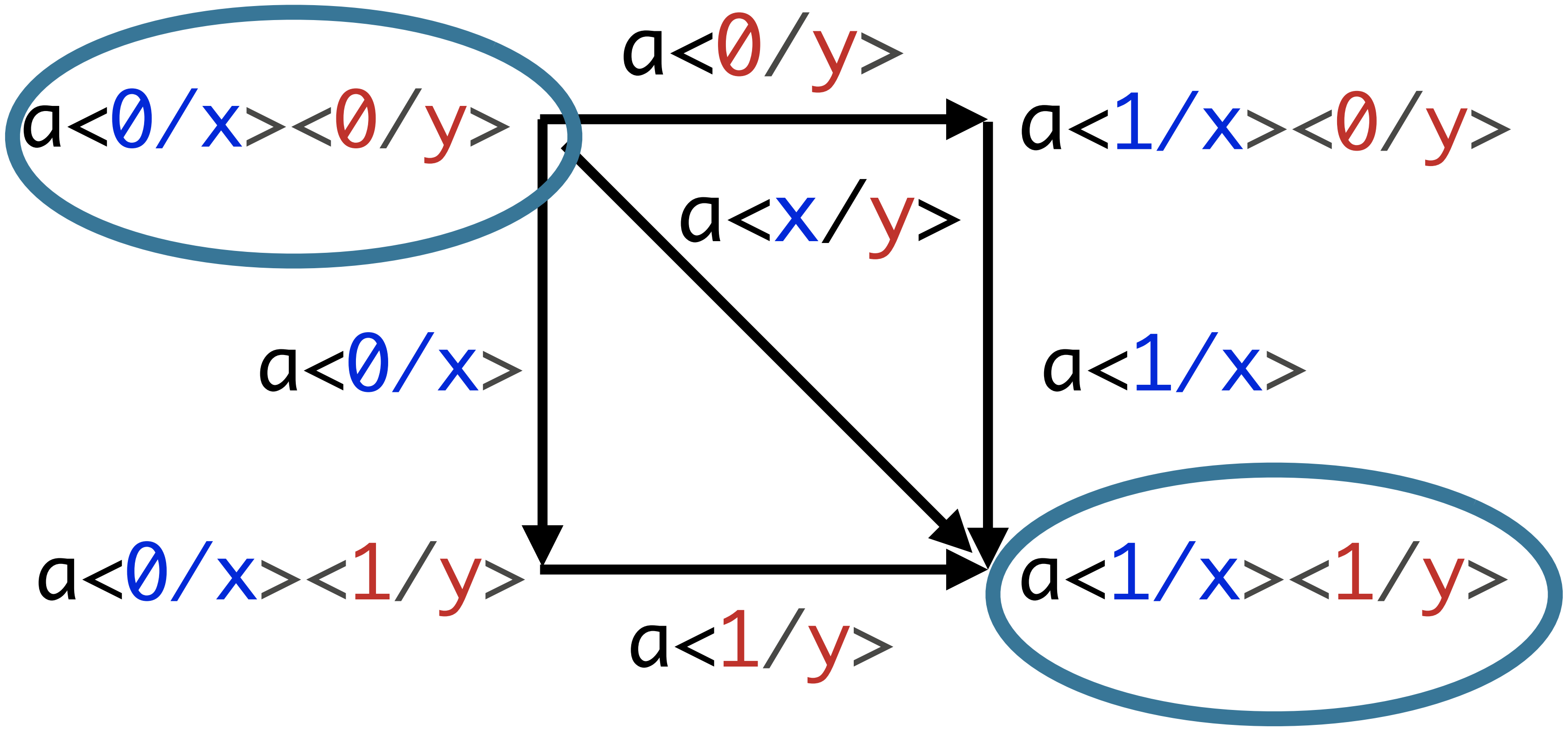
$$
\begin{tikzcd}
a<0/x><0/y> \arrow[r, "a<0/y>"] \arrow[d, "a<0/x>"'] & a<1/x><0/y> \arrow[d, "a<1/x>"] \\
a<0/x><1/y> \arrow[r, "a<1/y>"'] & a<1/x><1/y>
\end{tikzcd}
$$

$a<x/y>$ is a line ({x}-cube)

$$a<x/y><0/x> \equiv a<0/x><0/y>$$

$$a<x/y><1/x> \equiv a<1/x><1/y>$$

Diagonals

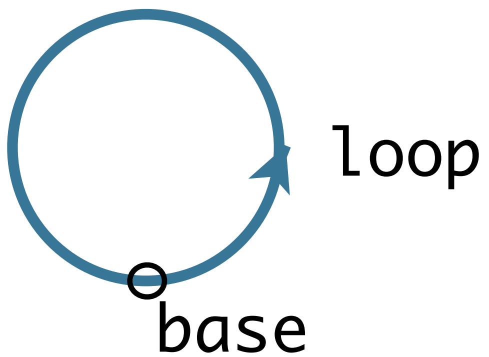


$a<x/y>$ is a line ({x}-cube)

$$a<x/y><0/x> \equiv a<0/x><0/y>$$

$$a<x/y><1/x> \equiv a<1/x><1/y>$$

Circle



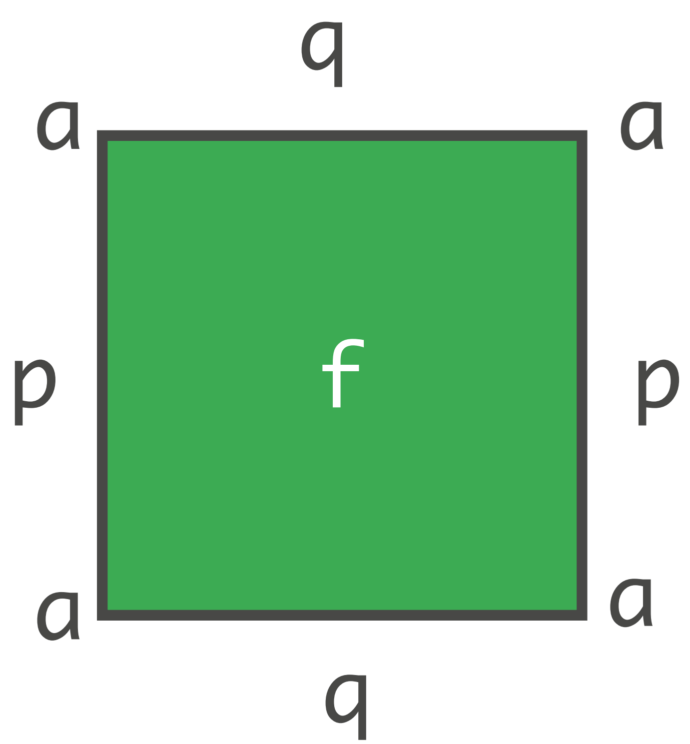
$$\mathtt{base} : S^1$$

$$x{:}I \vdash \mathtt{loop}_x : S^1$$

$$\mathtt{loop}_0 \equiv \mathtt{base}$$

$$\mathtt{loop}_1 \equiv \mathtt{base}$$

Torus



$$a : T$$

$$y{:}I \vdash p_y : T$$

$$x{:}I \vdash q_x : T$$

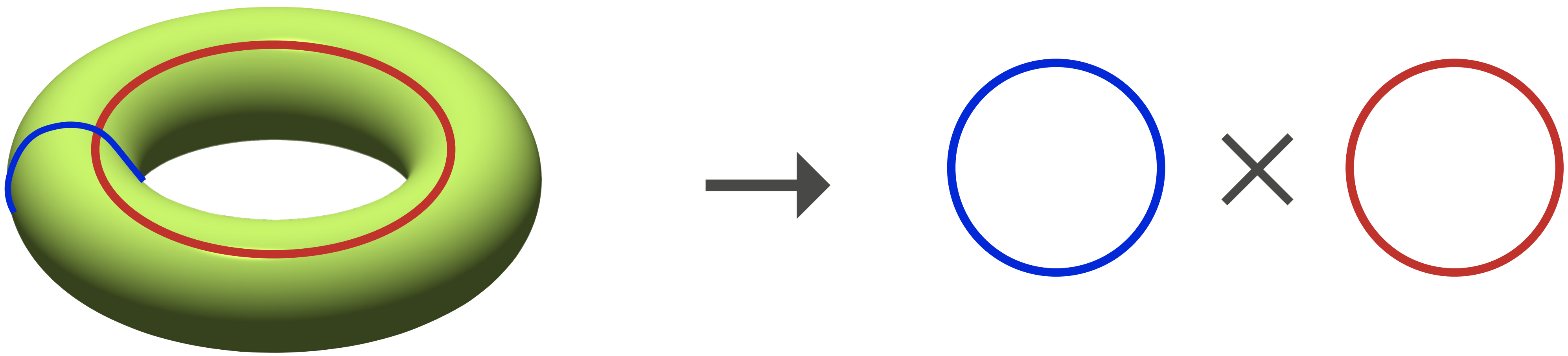
$$x{:}I, y{:}I \vdash f_{x,y} : T$$

$$p_0 \equiv p_1 \equiv q_0 \equiv q_1 \equiv a$$

$$f_{0,y} \equiv f_{1,y} \equiv p_y$$

$$f_{x,0} \equiv f_{x,1} \equiv q_x$$

```
t2c : T → S¹ × S¹
t2c a = (base,base)
t2c p_y = (loop_y,base)
t2c q_x = (base,loop_x)
t2c f_x,y = (loop_y,loop_x)
```



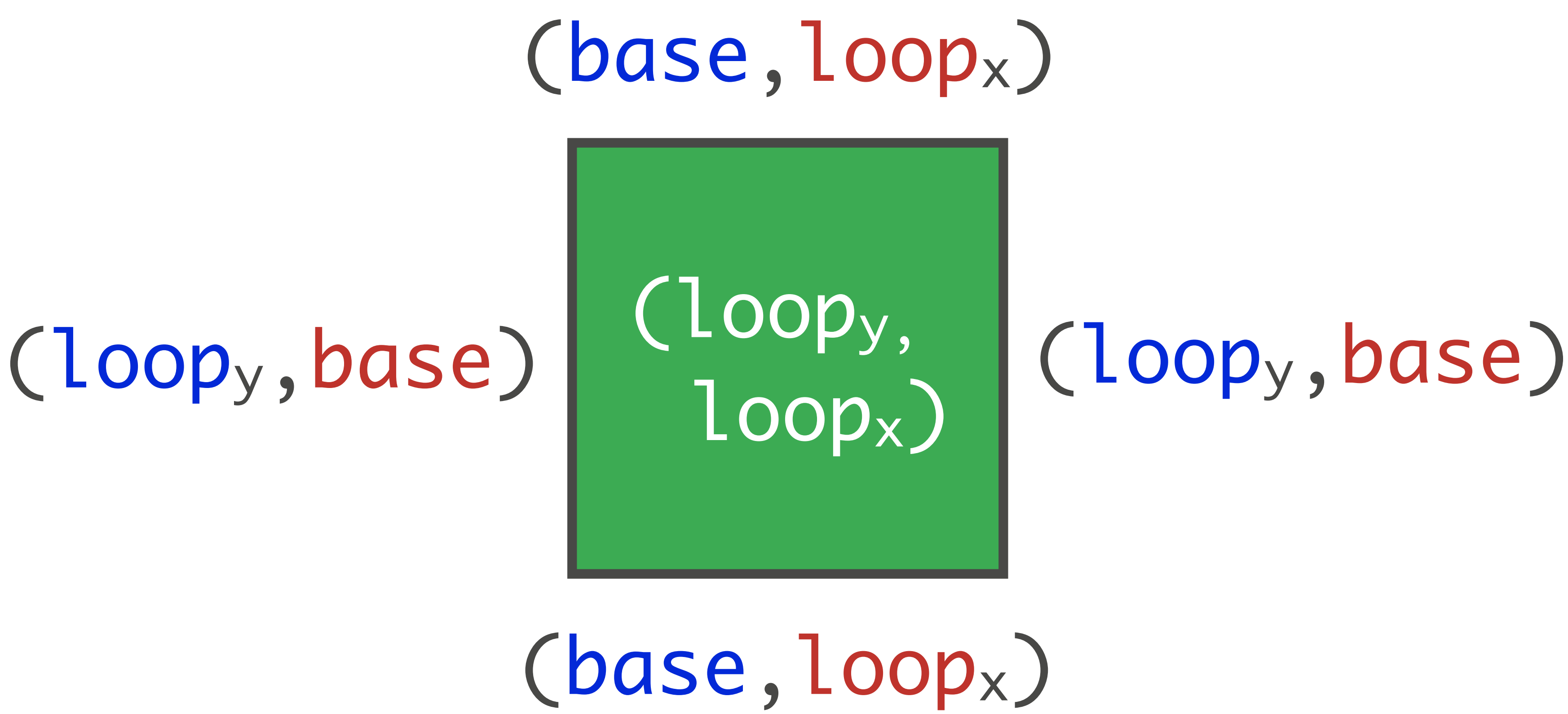
$\mathtt{t2c} : T \to S^1 \times S^1$

$\mathtt{t2c}\ a = (\mathtt{base}, \mathtt{base})$

$\mathtt{t2c}\ p_y = (\mathtt{loop}_y, \mathtt{base})$

$\mathtt{t2c}\ q_x = (\mathtt{base}, \mathtt{loop}_x)$

$\mathtt{t2c}\ f_{x,y} = (\mathtt{loop}_y, \mathtt{loop}_x)$



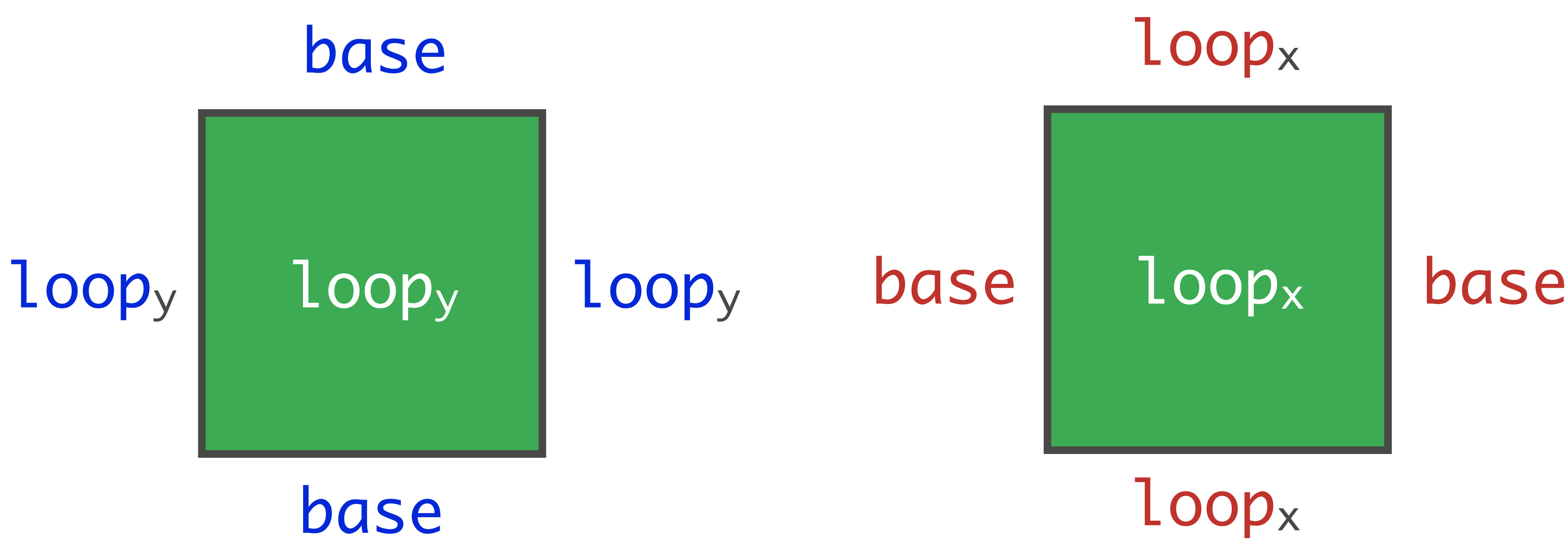
$$\mathrm{t2c} : T \to S^1 \times S^1$$

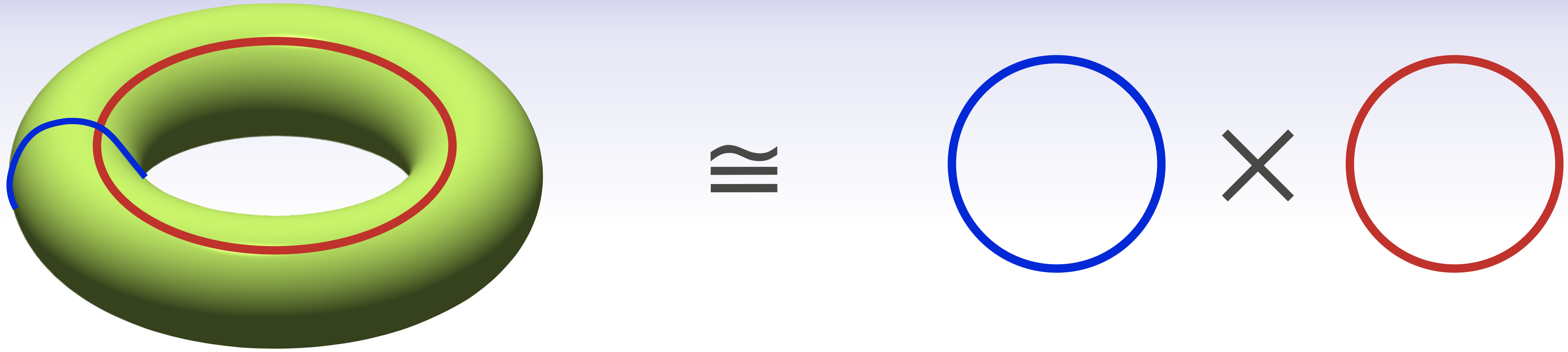
$$\mathrm{t2c}\ a = (\mathrm{base}, \mathrm{base})$$

$$\mathrm{t2c}\ p_y = (\mathrm{loop}_y, \mathrm{base})$$

$$\mathrm{t2c}\ q_x = (\mathrm{base}, \mathrm{loop}_x)$$

$$\mathrm{t2c}\ f_{x,y} = (\mathrm{loop}_y, \mathrm{loop}_x)$$





$\mathtt{t2c} : T \to S^1 \times S^1$

$\mathtt{t2c}\ a = (\mathtt{base}, \mathtt{base})$

$\mathtt{t2c}\ p_y = (\mathtt{loop}_y, \mathtt{base})$

$\mathtt{t2c}\ q_x = (\mathtt{base}, \mathtt{loop}_x)$

$\mathtt{t2c}\ f_{x,y} = (\mathtt{loop}_y, \mathtt{loop}_x)$

$\mathtt{c2t} : S^1 \to (S^1 \to T)$

$\mathtt{c2t}\ \mathtt{base} = \mathtt{base} \mapsto a$

$\qquad\qquad\quad \mathtt{loop}_x \mapsto q_x$

$\mathtt{c2t}\ \mathtt{loop}_y = \mathtt{base} \mapsto p_y$

$\qquad\qquad\quad \mathtt{loop}_x \mapsto f_{x,y}$

Composites: induct; reflexivity in each case

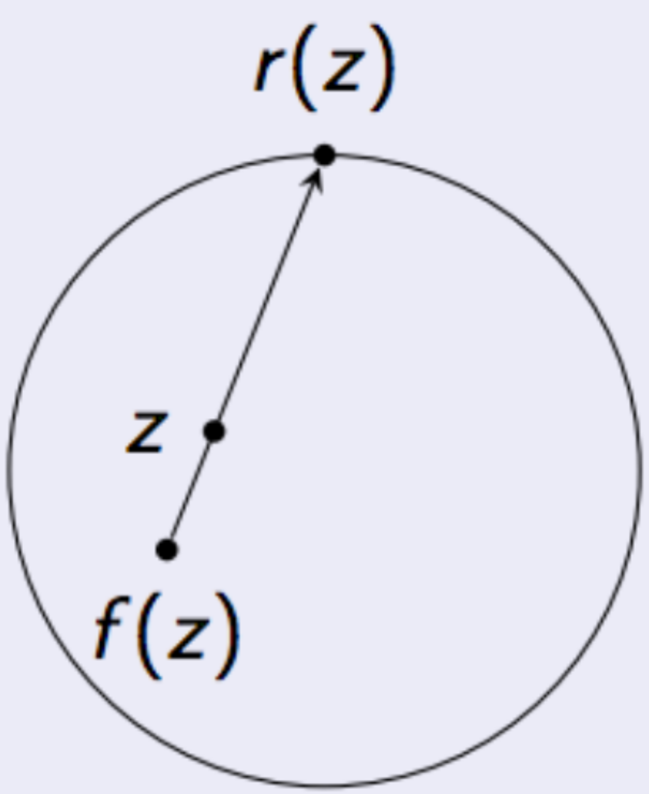
Modal homotopy type theories

Brouwer fixed point theorem

Theorem

Any continuous map $f : \mathbb{D}^2 \to \mathbb{D}^2$ has a fixed point.

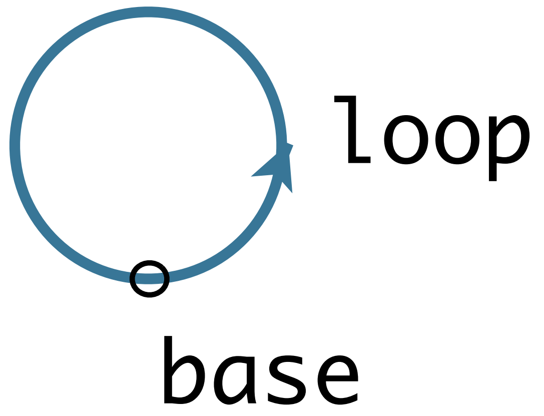
Proof.



Suppose $f : \mathbb{D}^2 \to \mathbb{D}^2$ is continuous with no fixed point. For any $z \in \mathbb{D}^2$, draw the ray from $f(z)$ through z to hit $\partial\mathbb{D}^2 = \mathbb{S}^1$ at $r(z)$. Then r is continuous, and retracts $\mathbb{D}^2$ onto $\mathbb{S}^1$. Hence $\pi_1(\mathbb{S}^1) = \mathbb{Z}$ is a retract of $\pi_1(\mathbb{D}^2) = 0$, a contradiction. $\square$

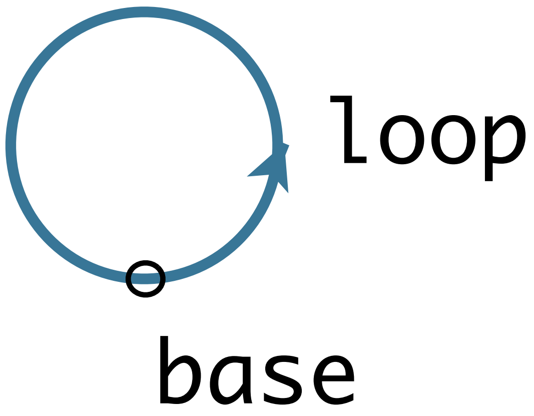
Brouwer fixed point theorem

homotopical circle



Brouwer fixed point theorem

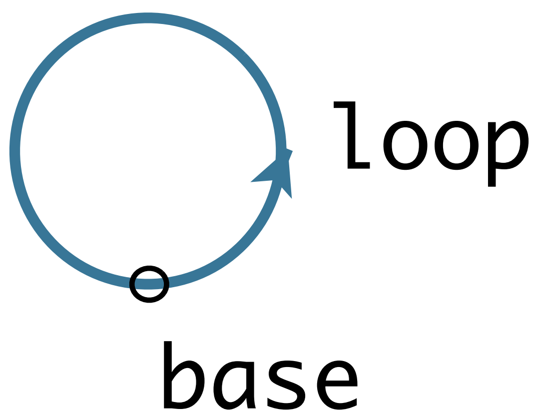
homotopical circle



homotopical disc

Brouwer fixed point theorem

homotopical circle

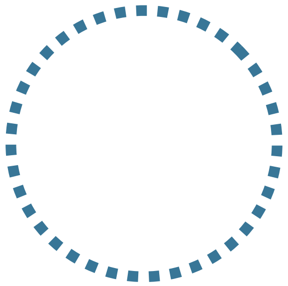


homotopical disc

theorem is about topological spaces,
continuous functions on them,
not homotopy types

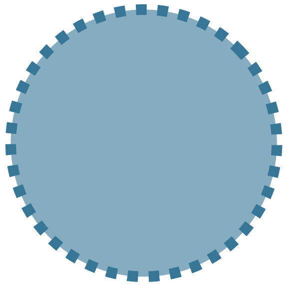
Brouwer fixed point theorem

topological circle



$\{(x,y):\mathbb{R}^2 \mid x^2 + y^2 = 1\}$

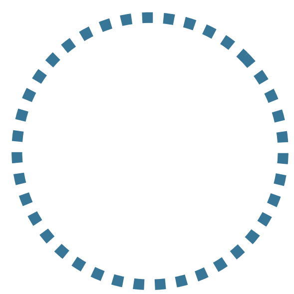
topological disc



$\{(x,y):\mathbb{R}^2 \mid x^2 + y^2 \leq 1\}$

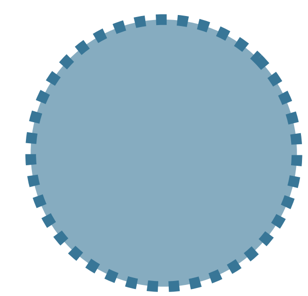
Brouwer fixed point theorem

topological circle



$\{(x,y):\mathbb{R}^2 \mid x^2 + y^2 = 1\}$

topological disc



$\{(x,y):\mathbb{R}^2 \mid x^2 + y^2 \leq 1\}$

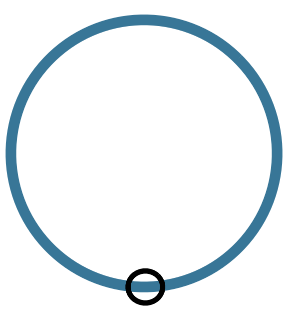
but want to use synthetic homotopy theory as a lemma:
need to connect topological circle and HIT circle
(without calculating homotopy groups of topological circle)

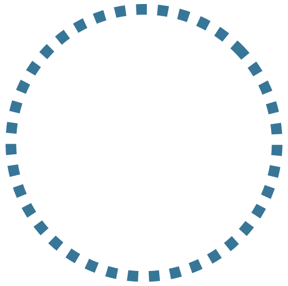
External View

type theory

↓

homotopy types
(e.g. simplicial sets) → topological spaces



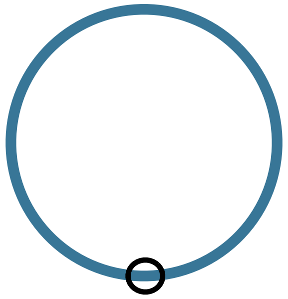


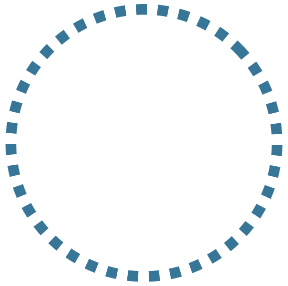
External View

type theory

↓

homotopy types
(e.g. simplicial sets) → topological spaces





but can’t see this *inside* the DSL

Types as spaces × 2

type theory

Types as spaces $\times$ 2

type theory

homotopy types

Types as spaces × 2

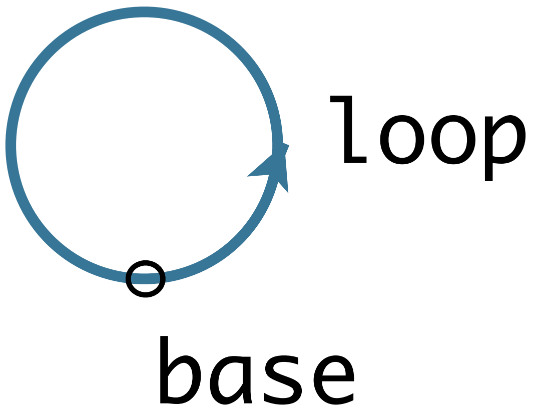
$$\begin{tikzcd}[column sep=small] & \text{type theory} \arrow[dl] \arrow[dr] & \\ \text{homotopy types} & & \text{topological spaces} \end{tikzcd}$$

Types as spaces × 2

$$\begin{tikzcd}
 & \text{type theory} \arrow[dl] \arrow[dr] & \\
\text{homotopy types} \arrow[dr] & & \text{topological spaces} \arrow[dl] \\
 & \text{real-cohesive HoTT [Shulman]} &
\end{tikzcd}$$

Circles

homotopical circle

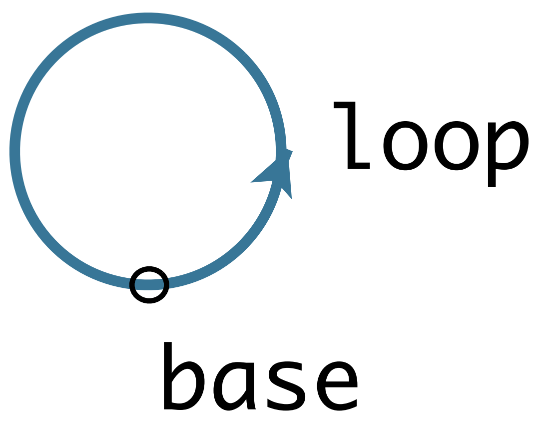


(base=base) ≃ $\mathbb{Z}$

topologically discrete

Circles

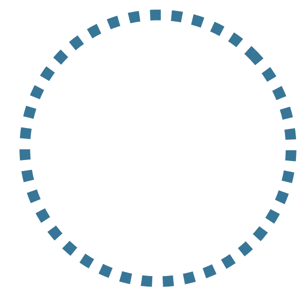
homotopical circle



(base=base) $\simeq \mathbb{Z}$

topologically discrete

topological circle

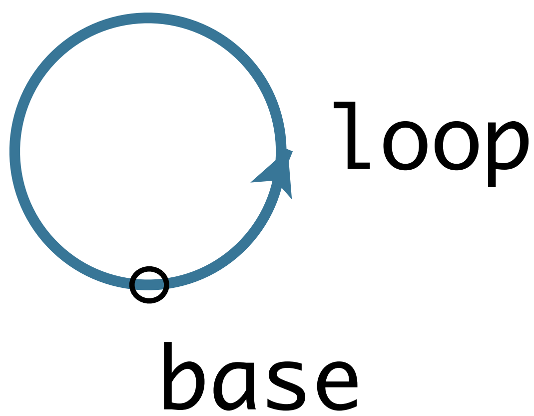


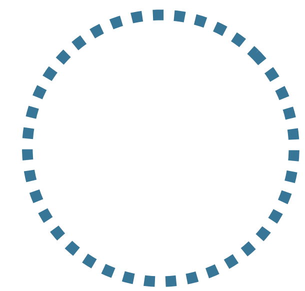
$\{(x,y):\mathbb{R}^2 \mid x^2 + y^2 = 1\}$

0-type

usual topology

Modalities





$\{(x,y):\mathbb{R}^2 \mid x^2 + y^2 = 1\}$

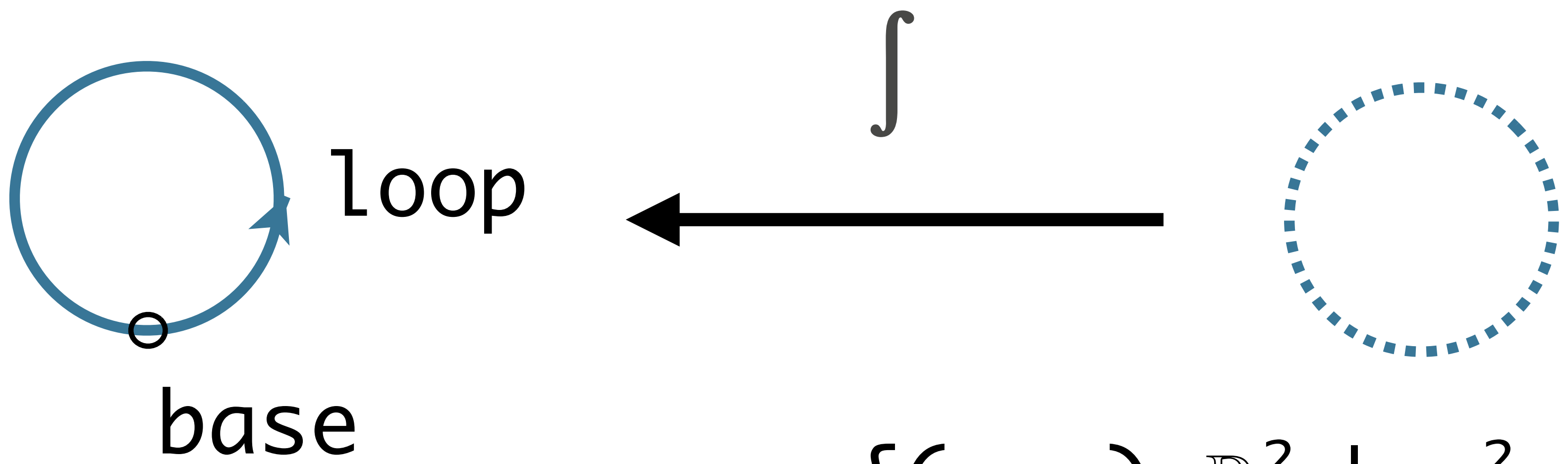
$(\text{base}=\text{base}) \simeq \mathbb{Z}$

topologically discrete

0-type

usual topology

Modalities



$\{(x,y):\mathbb{R}^2 \mid x^2 + y^2 = 1\}$

(base=base) $\simeq \mathbb{Z}$

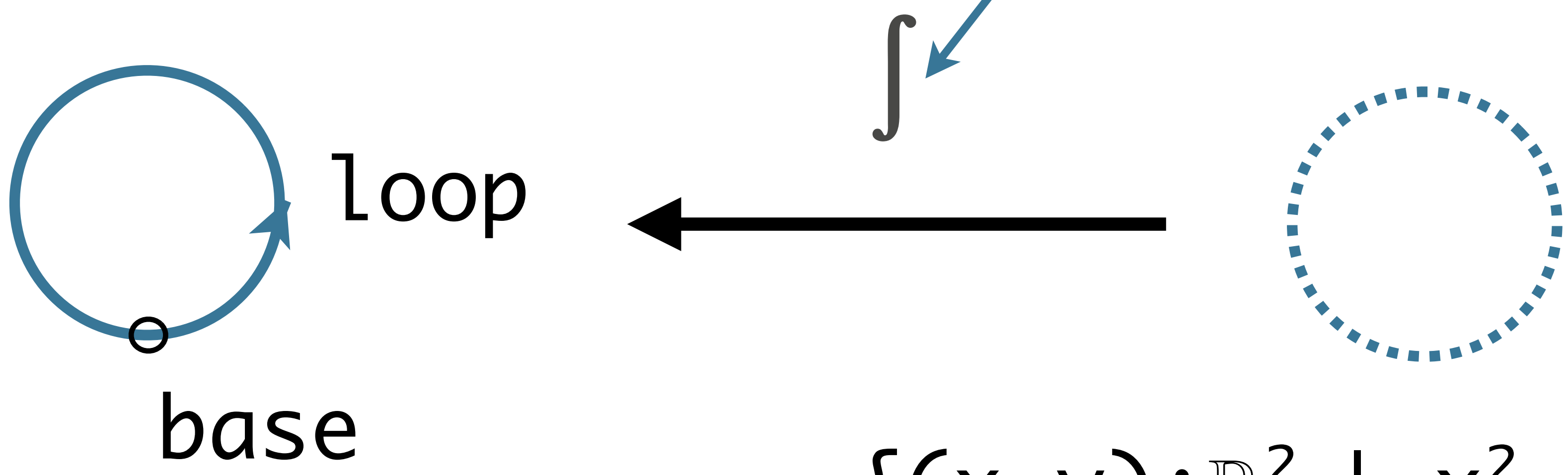
topologically discrete

0-type

usual topology

Modalities

makes topological paths into homotopical paths



$\{(x,y):\mathbb{R}^2 \mid x^2 + y^2 = 1\}$

$(base=base) \simeq \mathbb{Z}$

topologically discrete

0-type

usual topology

Cohesion [Lawvere; Schrieber,Shulman]

$$\begin{tikzcd}[row sep=huge, column sep=large]
\infty\text{-groupoids with topology on each level} \arrow[d, bend right=60, "\int"'] \arrow[d, phantom, shift right=6ex, "\dashv\ \Delta"] \arrow[d, phantom, shift left=3ex, "\dashv"] \arrow[d, "\Gamma"] \arrow[d, phantom, shift left=9ex, "\dashv"] \\
\infty\text{-groupoids} \arrow[u, bend right=30] \arrow[u, bend right=60, "\nabla"']
\end{tikzcd}$$

Cohesion [Lawvere; Schrieber,Shulman]

$$
\begin{tikzcd}[column sep=large]
\infty\text{-groupoids with topology on each level} \arrow[d, bend right=60, "\int"'] \arrow[d, "\Gamma"] \\
\infty\text{-groupoids} \arrow[u, bend left=30, "\Delta"] \arrow[u, bend right=60, "\nabla"']
\end{tikzcd}
$$

$$\int \dashv \Delta \dashv \Gamma \dashv \nabla$$

$$\flat\, A := \Delta\Gamma\, A$$

Cohesion [Lawvere; Schrieber,Shulman]

$$\begin{tikzcd}[row sep=huge, column sep=huge]
\infty\text{-groupoids with topology on each level}
\arrow[d, bend right=60, "\int"']
\arrow[d, bend right=60, phantom, xshift=1.2em, "\dashv"]
\arrow[d, "\Gamma"]
\arrow[d, bend left=60, phantom, xshift=-1.2em, "\dashv"]
\\
\infty\text{-groupoids}
\arrow[u, bend left=30, "\Delta"]
\arrow[u, phantom, xshift=-1.5em, "\dashv"]
\arrow[u, bend right=30, "\nabla"']
\end{tikzcd}$$

$$\flat\, A := \Delta\Gamma\, A \qquad \#\, A := \nabla\Gamma\, A$$

Cohesion [Lawvere; Schrieber,Shulman]

$$\begin{tikzcd}[column sep=large]
\infty\text{-groupoids with topology on each level} \arrow[d, bend right=60, "\int"'] \arrow[d, bend right=60, phantom, "\dashv", shift left=4ex] \arrow[d, "\Gamma"] \arrow[d, phantom, "\dashv", shift left=6ex] \\
\infty\text{-groupoids} \arrow[u, bend left=40, "\Delta"] \arrow[u, bend right=40, "\nabla"']
\end{tikzcd}$$

$$\int A := \Delta \int A \qquad \flat A := \Delta\Gamma A \qquad \# A := \nabla\Gamma A$$

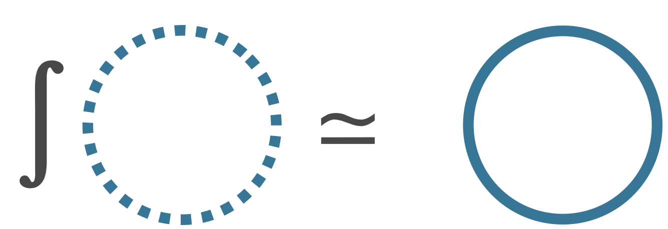
Cohesion [Lawvere; Schrieber,Shulman]

$$\begin{tikzcd}[column sep=large]
\infty\text{-groupoids with topology on each level} \arrow[d, "\flat"] \\
\infty\text{-groupoids with topology on each level} \arrow[u, bend left=60, "\int"] \arrow[u, bend right=60, "\sharp"']
\end{tikzcd}$$

$$\int \dashv \flat \dashv \sharp$$

Cohesion [Lawvere; Schrieber,Shulman]

$$\begin{tikzcd}[column sep=large]
\text{∞-groupoids with topology on each level} \arrow[d, "\flat" description] \arrow[d, phantom, shift right=6ex, "\dashv"] \arrow[d, phantom, shift left=6ex, "\dashv"] \\
\text{∞-groupoids with topology on each level} \arrow[u, bend left=70, "\int"] \arrow[u, bend right=70, "\#"']
\end{tikzcd}$$



Cohesion [Lawvere; Schrieber,Shulman]

$$\begin{tikzcd}[column sep=huge]
\infty\text{-groupoids with topology on each level} \arrow[d, "\flat"] \\
\infty\text{-groupoids with topology on each level} \arrow[u, bend left=60, "\int"] \arrow[u, bend right=60, "\#"']
\end{tikzcd}$$

$$\int \dashv \flat \dashv \#$$

Cohesion [Lawvere; Schrieber,Shulman]

$$\begin{tikzcd}
\text{cohesive spaces} \arrow[d, "\flat" description] \\
\text{cohesive spaces} \arrow[u, bend left=60, "\int"] \arrow[u, bend right=60, "\#"']
\end{tikzcd}$$

$$\int \dashv \flat \dashv \#$$

WIP: variations on cohesion

- Types as smooth manifolds [Schreiber,Wellen]: Differential cohesion (three additional modalities) [Gross,L.,New,Paykin,Riley,Shulman,Wellen]
- Types as parametrized spectra [Finster,Joyal]: Bireflective cohesion ($\flat$ is $\sharp$, retraction $\flat A \to A \to \sharp A$) [Finster,L.,Morehouse,Riley]
- Directed type theory [Riehl,Shulman]: Involutive cohesion (opposities) [L.,Riehl,Shulman]

Adjoint Logic

[Benton,Wadler'95;Atkey'04;Reed'09;L.,Shulman'16]

Framework [L.,Shulman,Riley,'17] for type theories with:

- Products and closed (non-assoc, monoidal, sym monoidal, semicartesian, relevant, cartesian, bunched)
- Adjoint functors (monoidal, lax, non- left adjoints)
- Comonads (monoidal, lax, non-monoidal)
- Monads (non-strong, strong, $\Box$-strong)

Mode theories

$$
\begin{tikzcd}
c \arrow[d, bend right=50, "\flat"'] \arrow[d, phantom, "\dashv"] \\
c \arrow[u, bend right=50, "\#"']
\end{tikzcd}
$$

generators and relations
in a cartesian 2-multicategory

c

$\flat : c \to c$

$\flat \circ \flat = \flat$

$\flat \Rightarrow 1$

$\flat(x \times y) = \flat x \times \flat y$

F types

$$\frac{\beta \Rightarrow \alpha[\gamma] \qquad \Gamma \vdash_{\gamma} \Delta}{\Gamma \vdash_{\beta} F_{\alpha}\, \Delta}$$

$$\frac{\Gamma, \Delta \vdash_{\beta[\alpha/x]} B}{\Gamma,\ x{:}F_{\alpha}(\Delta) \vdash_{\beta} B}$$

F/U-types for above mode theory explain spatial type theory [Shulman'15]

U types

$$\frac{\phi, c{:}q \vdash \alpha : p \qquad \Delta\ \mathrm{ctx}_{\phi} \qquad A\ \mathrm{type}_p}{U_{\alpha}(\Delta|A)\ \mathrm{type}_q}$$

$$\frac{\Gamma, \Delta \vdash_{\alpha[\beta/c]} A}{\Gamma \vdash_{\beta} U_{c.\alpha}(\Delta|A)}$$

$$\frac{}{\Delta,\ c{:}U_{c.\alpha}(\Delta|A) \vdash_{\alpha} A}$$

Adjoints

$$\flat \dashv \sharp$$

$$\frac{x{:}X \vdash_x U_\flat(Y)}{X \to \sharp Y}$$

Adjoints

$$\flat \dashv \sharp$$

$$\frac{x{:}X \vdash_{\flat(x)} Y}{x{:}X \vdash_x U_\flat(Y)}$$

$$X \to \sharp Y$$

Adjoints

$$\flat \dashv \sharp$$

$$\frac{\dfrac{z{:}F_\flat(X) \vdash_z Y}{x{:}X \vdash_{\flat(x)} Y}}{x{:}X \vdash_x U_\flat(Y)}$$

$$X \to \sharp Y$$

Adjoints

$$\flat \dashv \sharp$$

$$\flat X \to Y$$

$$z{:}F_\flat(X) \vdash_z Y$$

$$=\!=\!=\!=\!=\!=\!=$$

$$x{:}X \vdash_{\flat(x)} Y$$

$$=\!=\!=\!=\!=\!=\!=$$

$$x{:}X \vdash_x U_\flat(Y)$$

$$X \to \sharp Y$$

Adjoints

$\flat \dashv \sharp$

$- \otimes A \dashv A \multimap -$

$$\flat X \to Y$$

$$\dfrac{\dfrac{z{:}F_\flat(X) \vdash_z Y}{x{:}X \vdash_{\flat(x)} Y}}{x{:}X \vdash_x U_\flat(Y)}$$

$$X \to \sharp Y$$

Adjoints

$\flat \dashv \sharp$

$- \otimes A \dashv A \multimap -$

$$\flat X \to Y$$

$$\cfrac{\cfrac{z{:}F_\flat(X) \vdash_z Y}{x{:}X \vdash_{\flat(x)} Y}}{x{:}X \vdash_x U_\flat(Y)}$$

$$X \to \sharp Y$$

$$A \multimap Y$$

Adjoints

$$\flat \dashv \#$$

$$- \otimes A \dashv A \multimap -$$

$$\flat X \to Y$$

$$z{:}F_\flat(X) \vdash_z Y$$

$$x{:}X \vdash_{\flat(x)} Y$$

$$x{:}X \vdash_x U_\flat(Y)$$

$$X \to \# Y$$

$$x{:}X \vdash_x U_{c \otimes a}(a{:}A \mid Y)$$

$$A \multimap Y$$

Adjoints

$$\flat \dashv \#$$

$$
\begin{array}{c}
\flat X \to Y \\
z{:}F_\flat(X) \vdash_z Y \\
\hline\hline
x{:}X \vdash_{\flat(x)} Y \\
\hline\hline
x{:}X \vdash_x U_\flat(Y) \\
X \to \#Y
\end{array}
$$

$$- \otimes A \dashv A \multimap -$$

$$
\begin{array}{c}
x{:}X, a{:}A \vdash_{x \otimes a} Y \\
\hline\hline
x{:}X \vdash_x U_{c \otimes a}(a{:}A | Y) \\
A \multimap Y
\end{array}
$$

Adjoints

$\flat \dashv \#$

$- \otimes A \dashv A \multimap -$

$\flat X \to Y$

$$\frac{\frac{z{:}F_\flat(X) \vdash_z Y}{x{:}X \vdash_{\flat(x)} Y}}{x{:}X \vdash_x U_\flat(Y)}$$

$$\frac{\frac{z{:}F_{x\otimes a}(x{:}X, a{:}A) \vdash_z Y}{x{:}X, a{:}A \vdash_{x\otimes a} Y}}{x{:}X \vdash_x U_{c\,\otimes\,a}(a{:}A|Y)}$$

$X \to \#Y$

$A \multimap Y$

Adjoints

$$\flat \dashv \#$$

$$\flat X \to Y$$

$$\frac{\dfrac{z{:}F_{\flat}(X) \vdash_z Y}{x{:}X \vdash_{\flat(x)} Y}}{x{:}X \vdash_x U_{\flat}(Y)}$$

$$X \to \# Y$$

$$- \otimes A \dashv A \multimap -$$

$$X \otimes A$$

$$\frac{\dfrac{z{:}F_{x \otimes a}(x{:}X, a{:}A) \vdash_z Y}{x{:}X, a{:}A \vdash_{x \otimes a} Y}}{x{:}X \vdash_x U_{c \otimes a}(a{:}A | Y)}$$

$$A \multimap Y$$

$$x{:}\ F_f\, A \otimes F_f\, B \vdash_x F_f\, (A \times B)$$

$$\frac{y{:}F_f\, A,\ z{:}F_f\, B \vdash_{y \otimes z} F_f\, (A \times B)}{x{:}\ F_f\, A \otimes F_f\, B \vdash_{x} F_f\, (A \times B)}$$

$$\dfrac{\dfrac{y{:}A,\ z{:}F_f\,B \vdash_{f(y)\otimes z} F_f\,(A \times B)}{y{:}F_f\,A,\ z{:}F_f\,B \vdash_{y\otimes z} F_f\,(A \times B)}}{x{:}\ F_f\,A \otimes F_f\,B \vdash_{x} F_f\,(A \times B)}$$

$$\dfrac{\dfrac{\dfrac{y{:}A,\ z{:}B \vdash_{f(y)\otimes f(z)} F_f\,(A \times B)}{y{:}A,\ z{:}F_f\,B \vdash_{f(y)\otimes z} F_f\,(A \times B)}}{y{:}F_f\,A,\ z{:}F_f\,B \vdash_{y\otimes z} F_f\,(A \times B)}}{x{:}\ F_f\,A \otimes F_f\,B \vdash_{x} F_f\,(A \times B)}$$

$f(y) \otimes f(z) \Rightarrow f(?)$

$$\dfrac{\dfrac{\dfrac{\dfrac{}{y{:}A,\ z{:}B \vdash_{f(y) \otimes f(z)} F_f\,(A \times B)}}{y{:}A,\ z{:}F_f\,B \vdash_{f(y) \otimes z} F_f\,(A \times B)}}{y{:}F_f\,A,\ z{:}F_f\,B \vdash_{y \otimes z} F_f\,(A \times B)}}{x{:}\ F_f\,A \otimes F_f\,B \vdash_{x} F_f\,(A \times B)}$$

$$
\begin{array}{c}
\hline
y{:}A,\ z{:}B \vdash_{f(y)\otimes f(z)} F_f\,(A \times B) \\
\hline
y{:}A,\ z{:}F_f\,B \vdash_{f(y)\otimes z} F_f\,(A \times B) \\
\hline
y{:}F_f\,A,\ z{:}F_f\,B \vdash_{y\otimes z} F_f\,(A \times B) \\
\hline
x{:}\ F_f\,A \otimes F_f\,B \vdash_{x} F_f\,(A \times B)
\end{array}
$$

$f(y) \otimes f(z) \Rightarrow f(y \times z)$

$$\dfrac{\dfrac{\dfrac{y{:}A,\ z{:}B \vdash_{f(y)\otimes f(z)} F_f\,(A \times B)}{y{:}A,\ z{:}F_f\,B \vdash_{f(y)\otimes z} F_f\,(A \times B)}}{y{:}F_f\,A,\ z{:}F_f\,B \vdash_{y\otimes z} F_f\,(A \times B)}}{x{:}\ F_f\,A \otimes F_f\,B \vdash_{x} F_f\,(A \times B)}$$

$$\cfrac{\cfrac{\cfrac{\cfrac{f(y)\otimes f(z)\Rightarrow f(y\times z) \qquad \cfrac{y{:}A\vdash_y A \qquad z{:}B\vdash_z B}{y{:}A,z{:}B\vdash_{y\times z} A\times B}}{y{:}A,\ z{:}B \vdash_{f(y)\otimes f(z)} F_f\,(A\times B)}}{y{:}A,\ z{:}F_f\,B\vdash_{f(y)\otimes z} F_f\,(A\times B)}}{y{:}F_f\,A,\ z{:}F_f\,B\vdash_{y\otimes z} F_f\,(A\times B)}}{x{:}\ F_f\,A\otimes F_f\,B\vdash_x F_f\,(A\times B)}$$

Semantics

- a functor from a cartesian multicategory (derivations) to another cartesian 2-multicategory (mode theory)
- which is a locally discrete fibration (action of structural rules)
- $F_\alpha \Delta$ makes this into an opfibration
- $U_\alpha(\Delta|A)$ makes this into a fibration

internal languages:
short statements *inside* a particular categorical setting can unpack to long external statements

internal languages:

short statements *inside* a particular categorical setting can unpack to long external statements

(*m*+*n*)-connected

n-connected

$$
\begin{tikzcd}
Z \arrow[dr] \arrow[drr, bend left, "g"] \arrow[ddr, bend right, "f"'] & & \\
& X \times_W Y \arrow[r, "\pi_2"] \arrow[d, "\pi_1"] & Y \arrow[d, "\nu_2"] \\
& X \arrow[r, "\nu_1"] & W := X +_Z Y
\end{tikzcd}
$$

m-connected

internal languages:

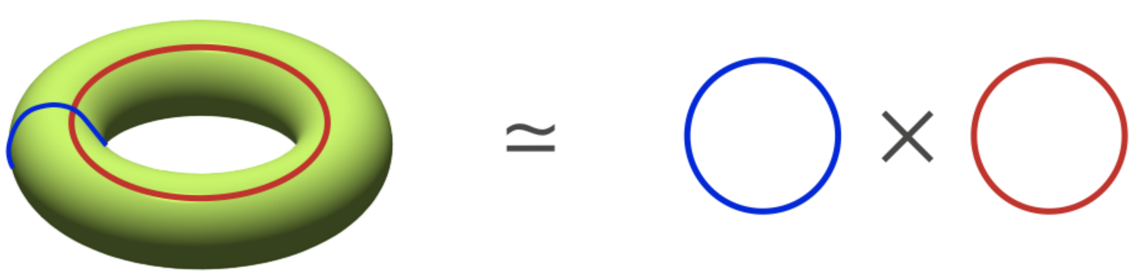
short statements *inside* a particular categorical setting can unpack to long external statements

(*m*+*n*)-connected

n-connected

m-connected

$$
\begin{tikzcd}
Z \arrow[dr] \arrow[drr, bend left, "g"] \arrow[ddr, bend right, "f"'] & & \\
& X \times_W Y \arrow[r, "\pi_2"] \arrow[d, "\pi_1"] & Y \arrow[d, "\nu_2"] \\
& X \arrow[r, "\nu_1"] & W := X +_Z Y
\end{tikzcd}
$$



internal languages:

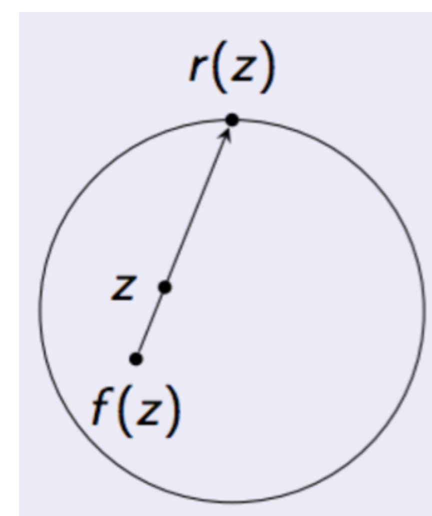
short statements *inside* a particular categorical setting can unpack to long external statements

(*m*+*n*)-connected

n-connected

$$\begin{tikzcd}
Z \arrow[dr] \arrow[drr, bend left, "g"] \arrow[ddr, bend right, "f"'] & & \\
& X \times_W Y \arrow[r, "\pi_2"] \arrow[d, "\pi_1"] & Y \arrow[d, "\nu_2"] \\
& X \arrow[r, "\nu_1"] & W := X +_Z Y
\end{tikzcd}$$

m-connected



Target Architecture

ARM  ZFC/w.f. trees

x86  type theory/meaning expl.

High-level Languages

C/ML/… Agda/Gallina/Isar/…

Domain-specific Languages

NESL  MLTT+UA+HITs

Cubical type theory

SQL  Modal type theory