

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



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

ARM



```
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

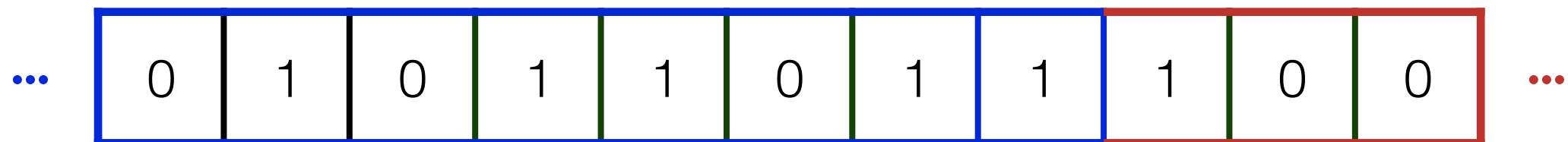
C

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

Portable Assembly Language

A : bit array[8]

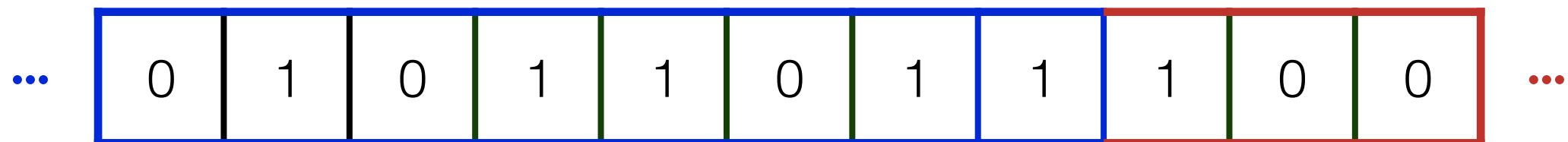
x : int



Portable Assembly Language

A : bit array[8]

x : int

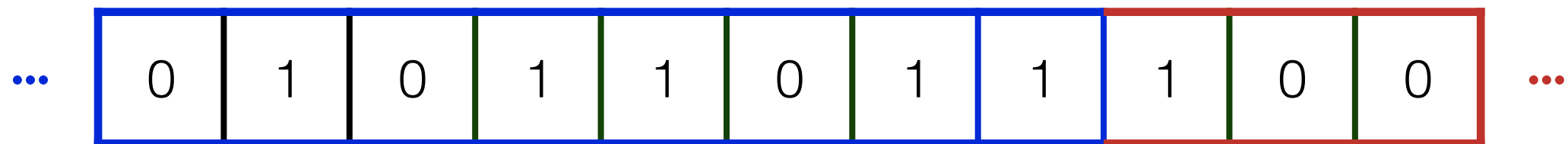


A[8]

Portable Assembly Language

A : bit array[8]

x : int



A[8]

High-Level Language

ML

High-Level Language

ML `map ( $\lambda x \rightarrow x + 1$ ) A`

High-Level Language

ML `map ( $\lambda x \rightarrow x + 1$ ) 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 ( $\lambda x \rightarrow x + 1$ ) 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



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



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)

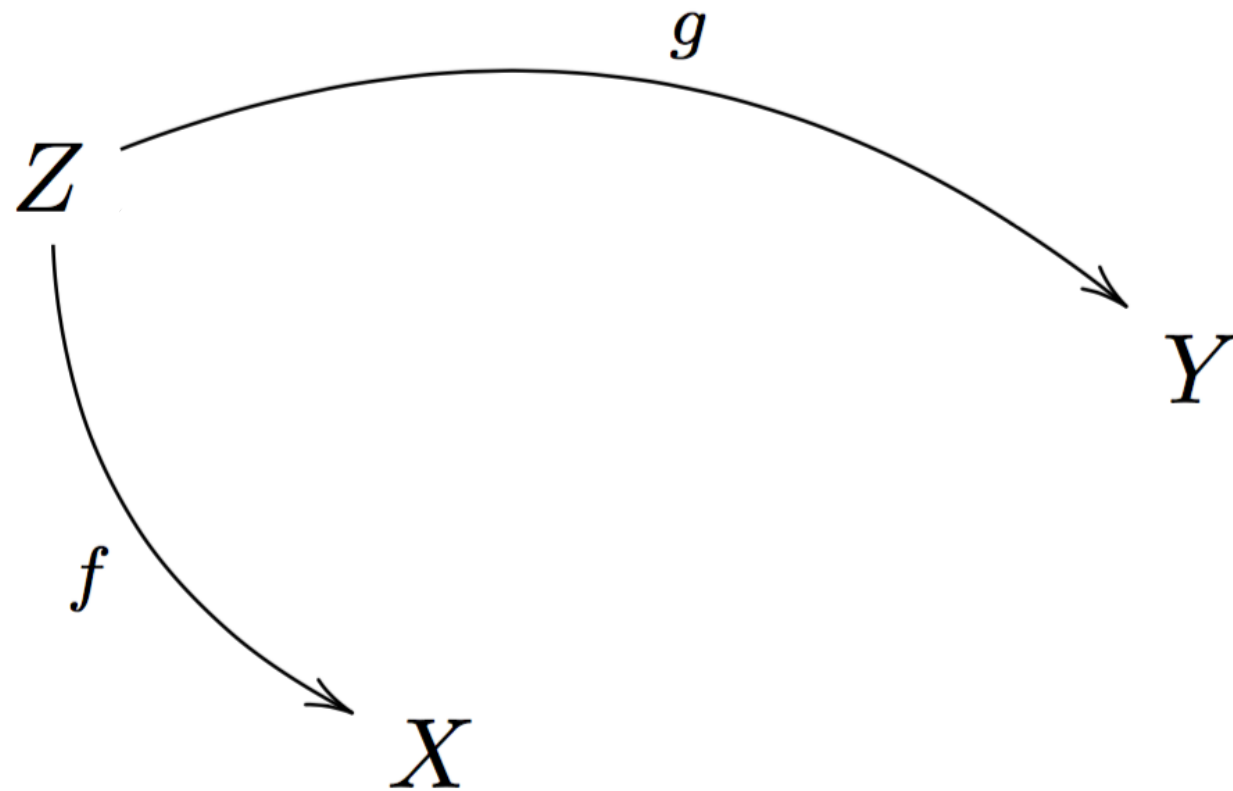
HoTT'13 (MLTT+UA+HITs)

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

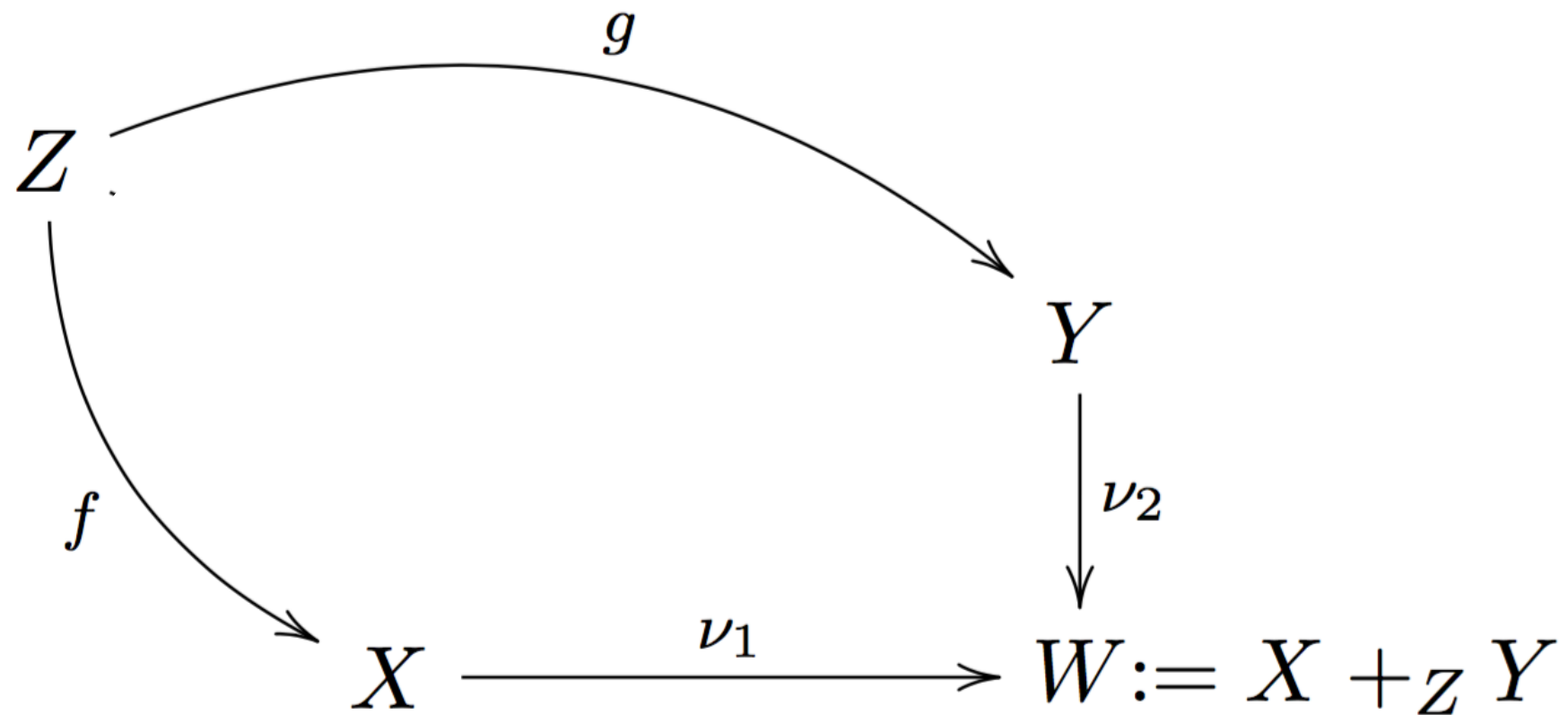
HoTT'13 (MLTT+UA+HITs)

- * 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

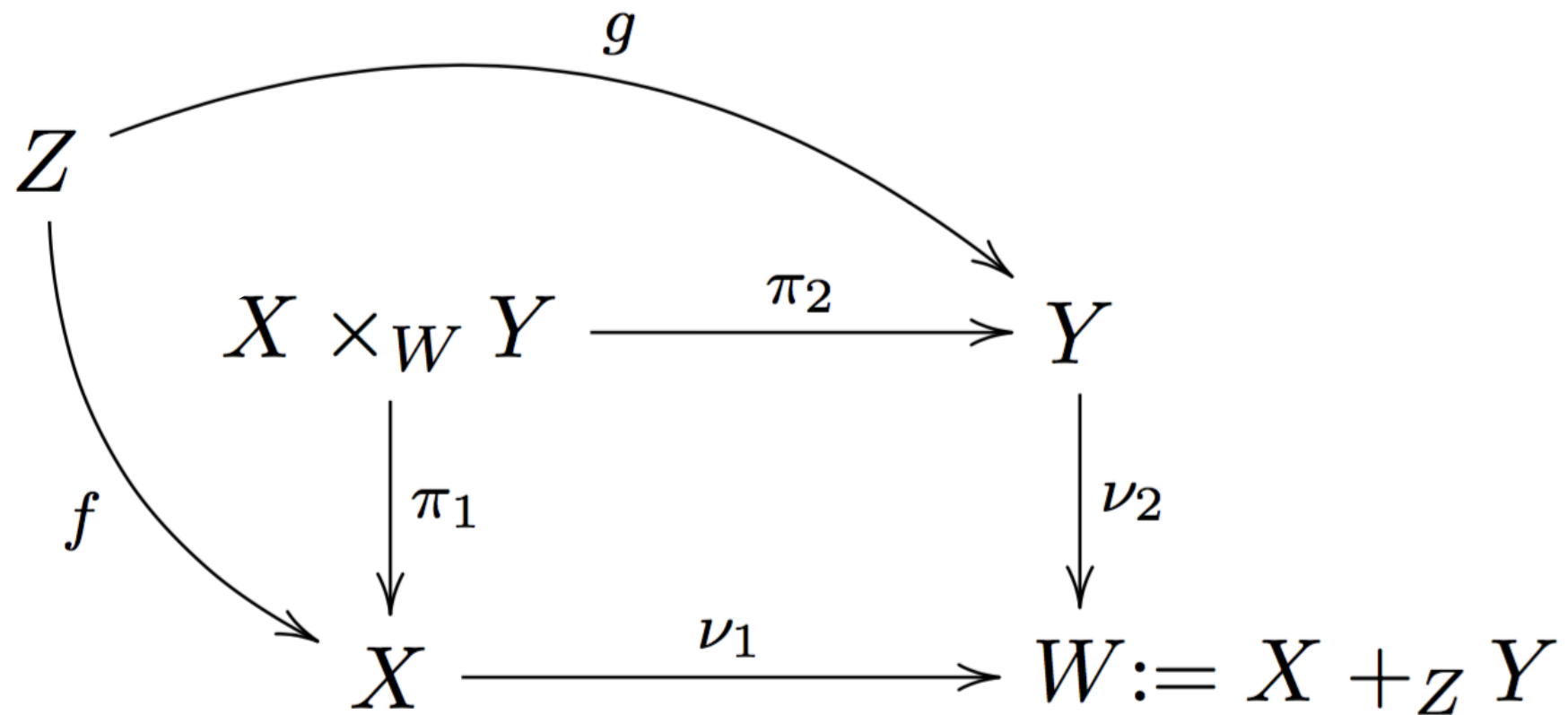
Blakers-Massey Theorem



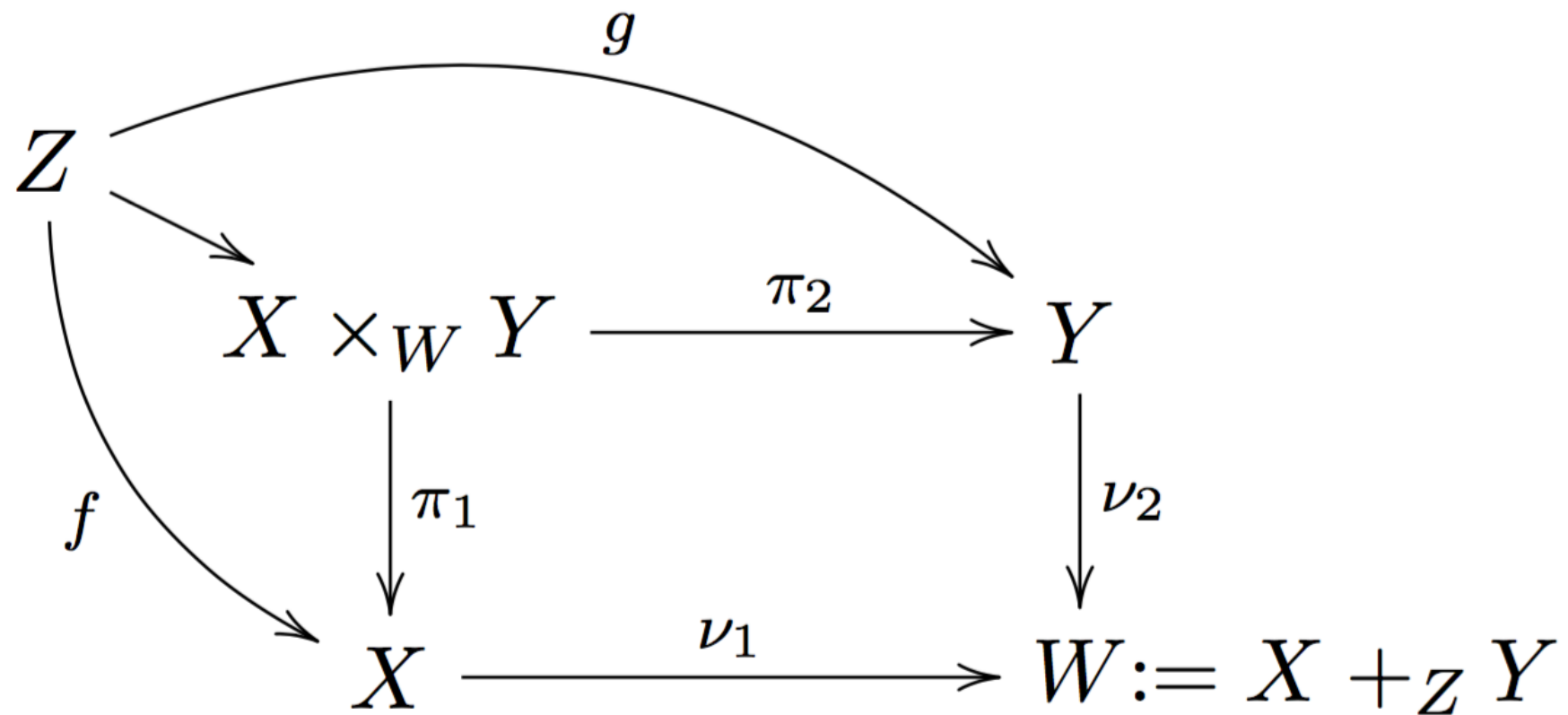
Blakers-Massey Theorem



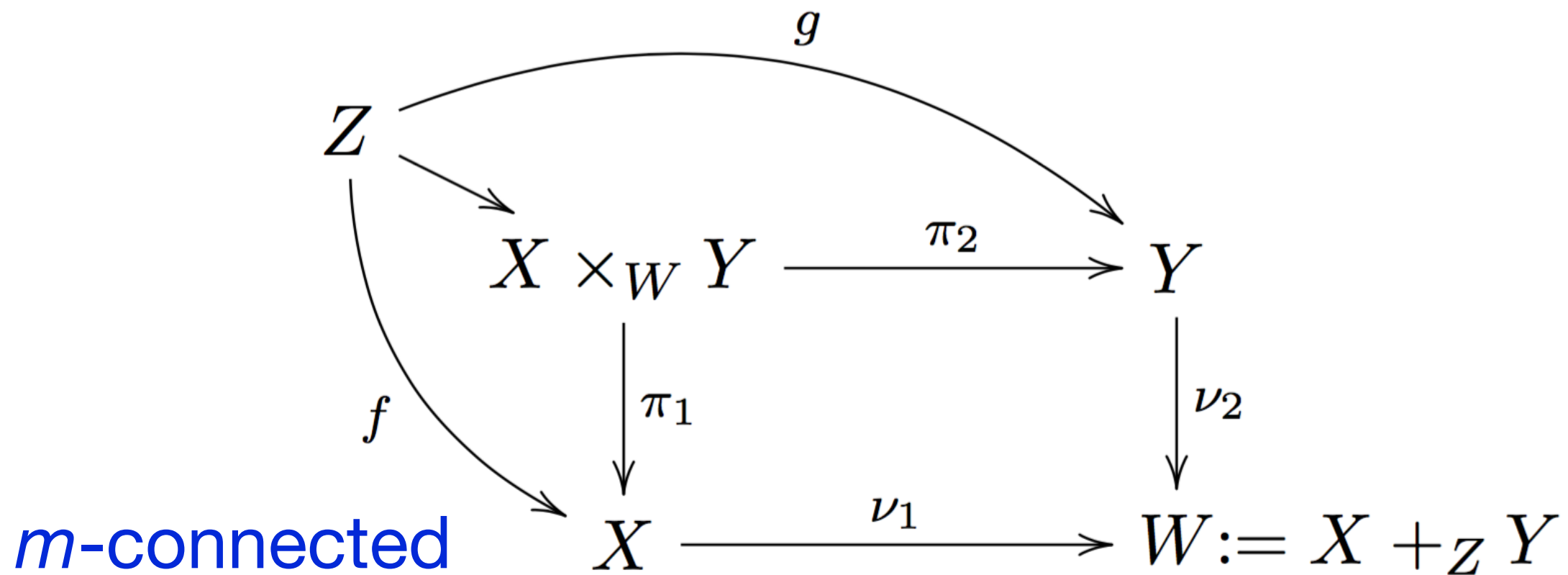
Blakers-Massey Theorem



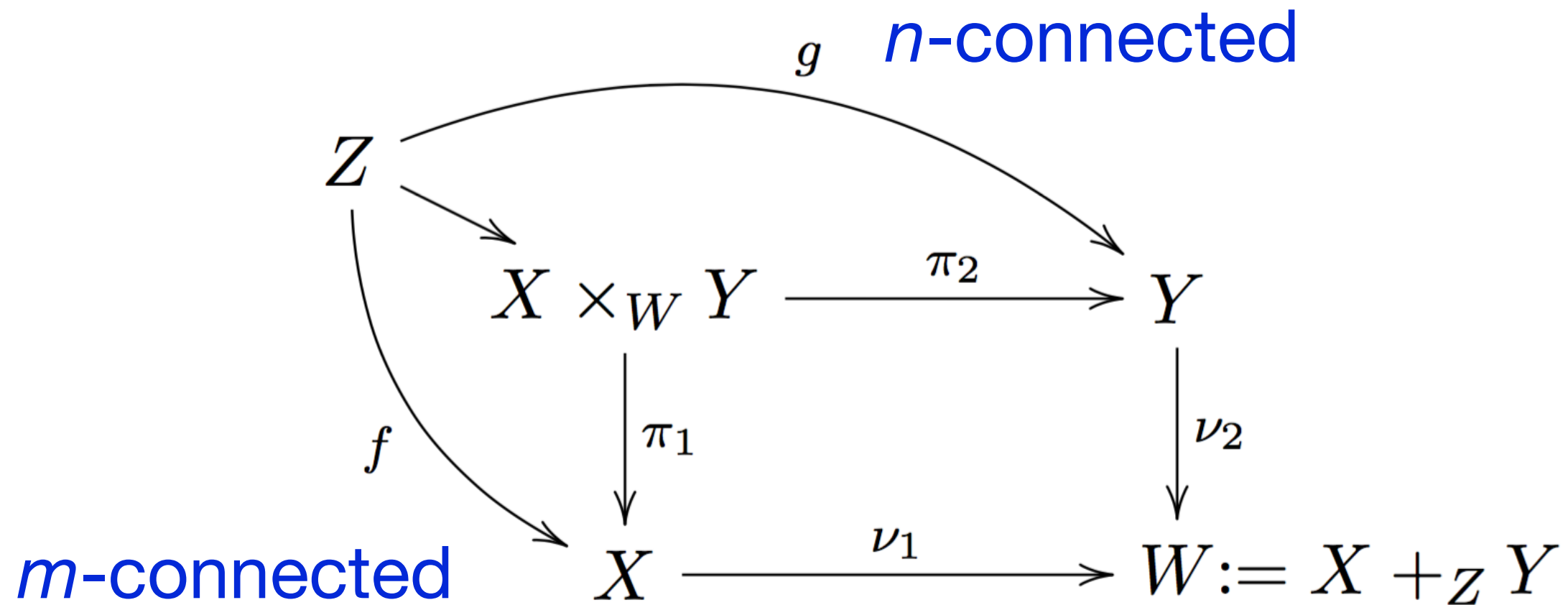
Blakers-Massey Theorem



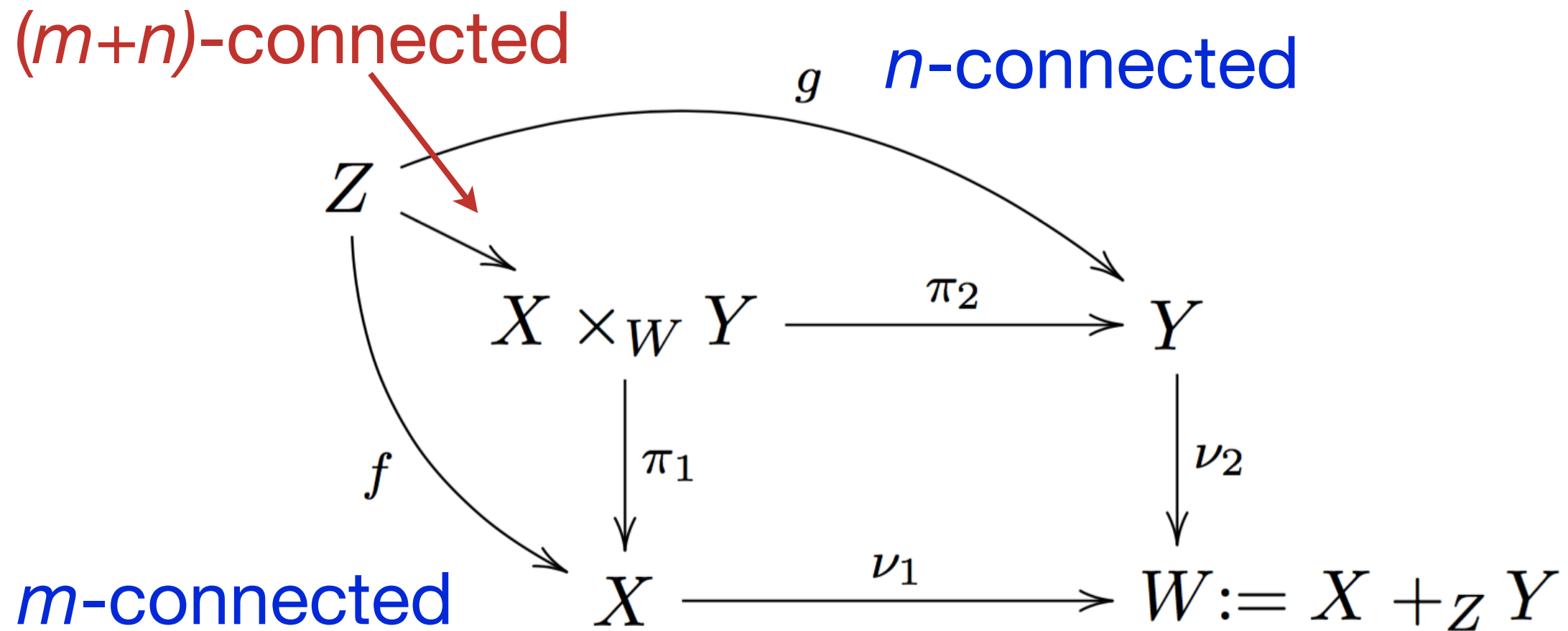
Blakers-Massey Theorem



Blakers-Massey Theorem



Blakers-Massey Theorem



Homotopy groups of spheres

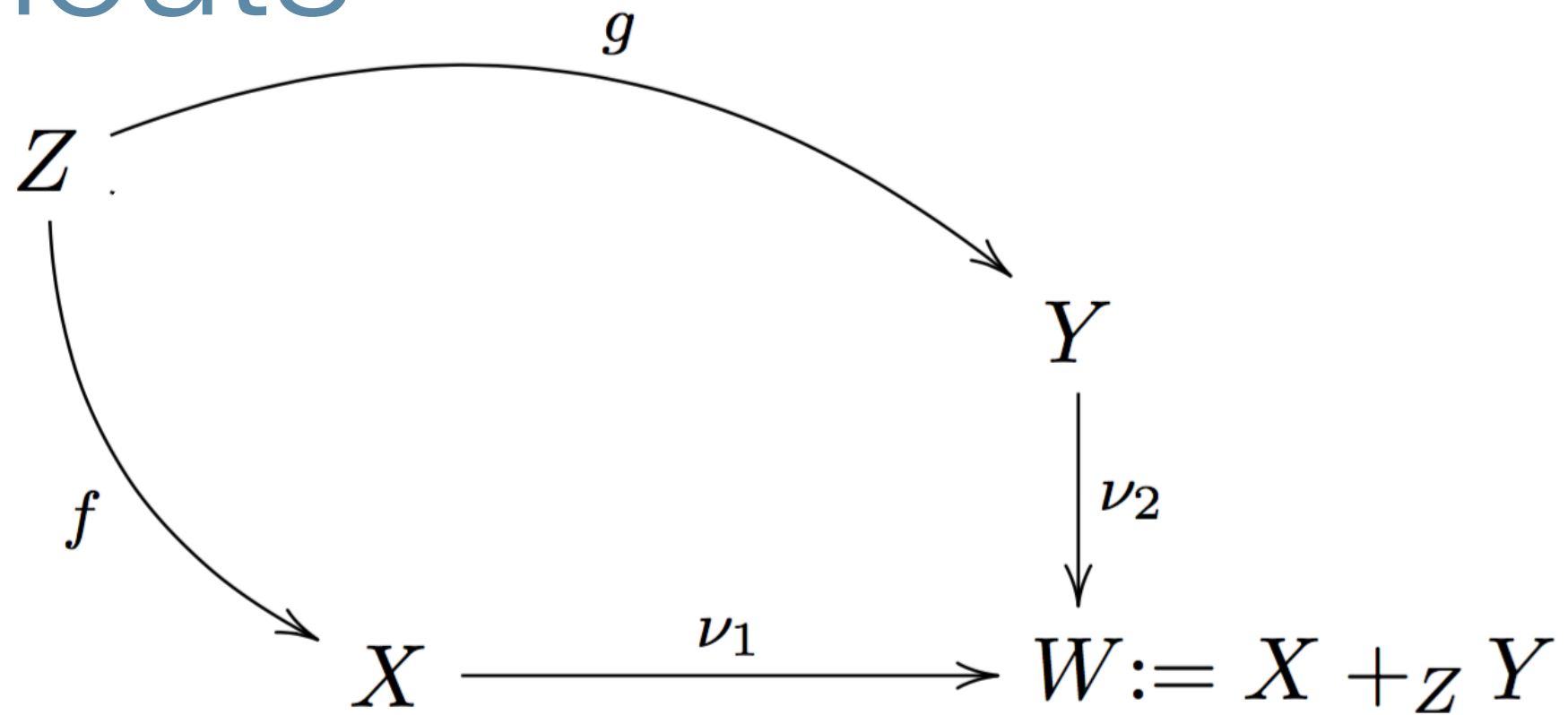
k^{th} homotopy group

n-dimensional sphere

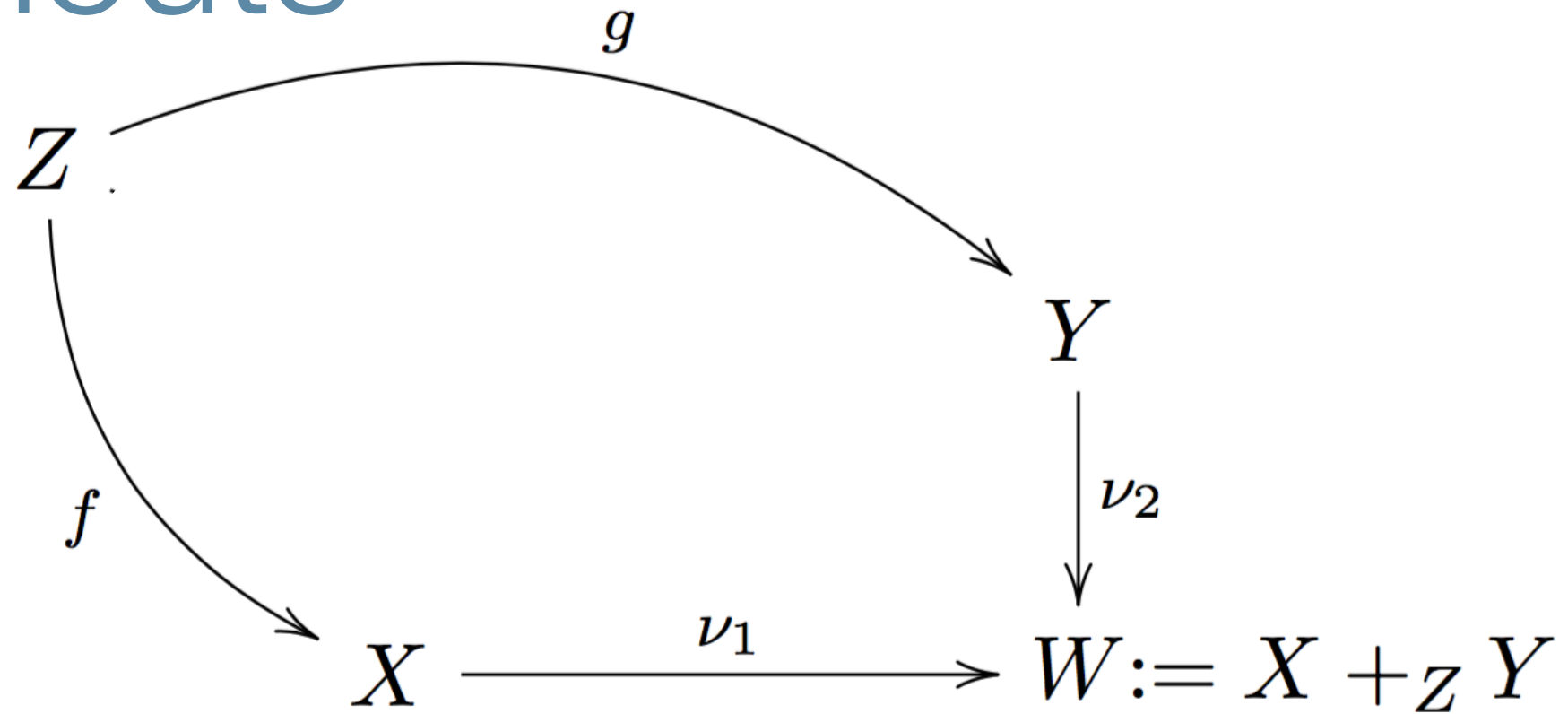
	π_1	π_2	π_3	π_4	π_5	π_6	π_7	π_8	π_9	π_{10}	π_{11}	π_{12}	π_{13}	π_{14}	π_{15}
S^0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
S^1	$\mathbb{Z}$	0	0	0	0	0	0	0	0	0	0	0	0	0	0
S^2	0	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}_2$	$\mathbb{Z}_2$	$\mathbb{Z}_{12}$	$\mathbb{Z}_2$	$\mathbb{Z}_2$	$\mathbb{Z}_3$	$\mathbb{Z}_{15}$	$\mathbb{Z}_2$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

[image from wikipedia]

Pushouts

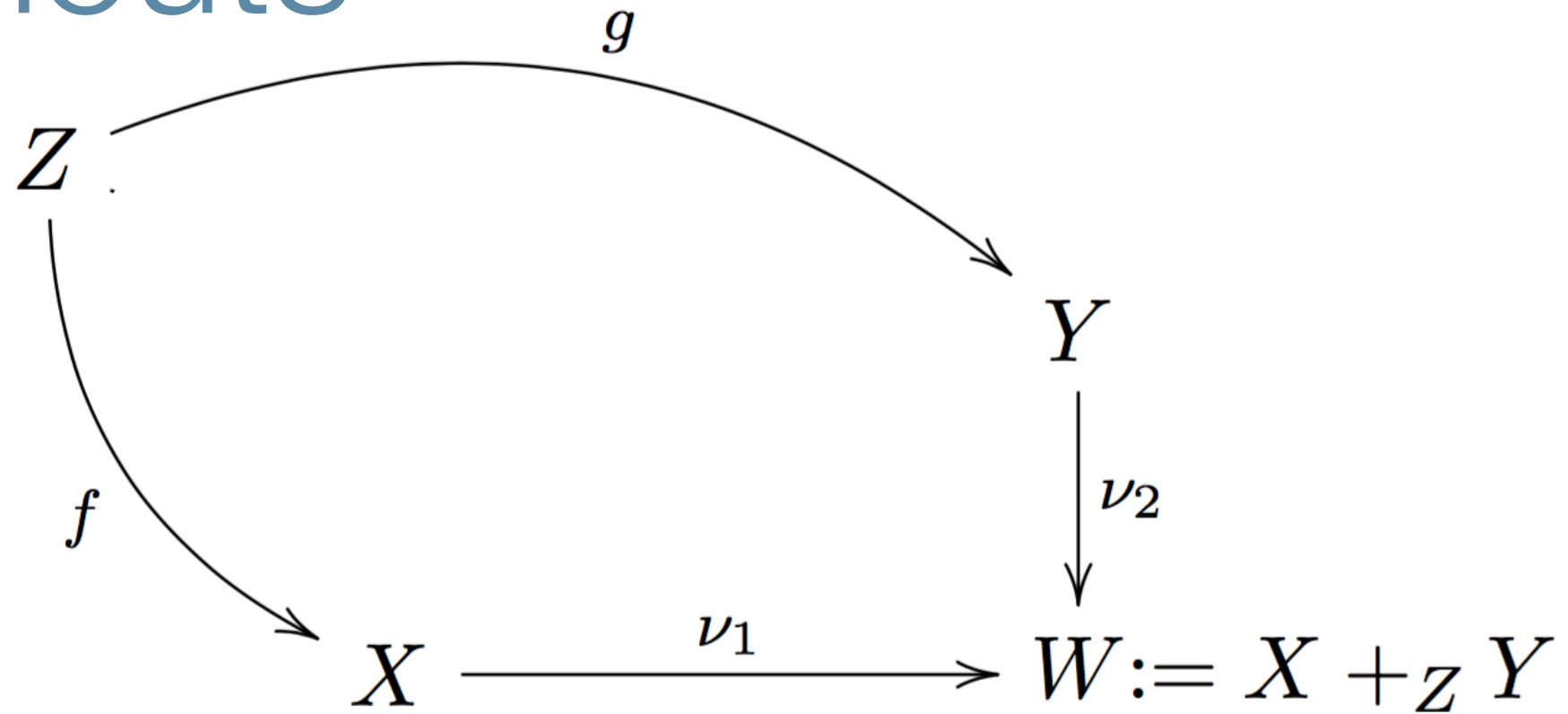


Pushouts



$\text{inl} : X \rightarrow X +_Z Y$

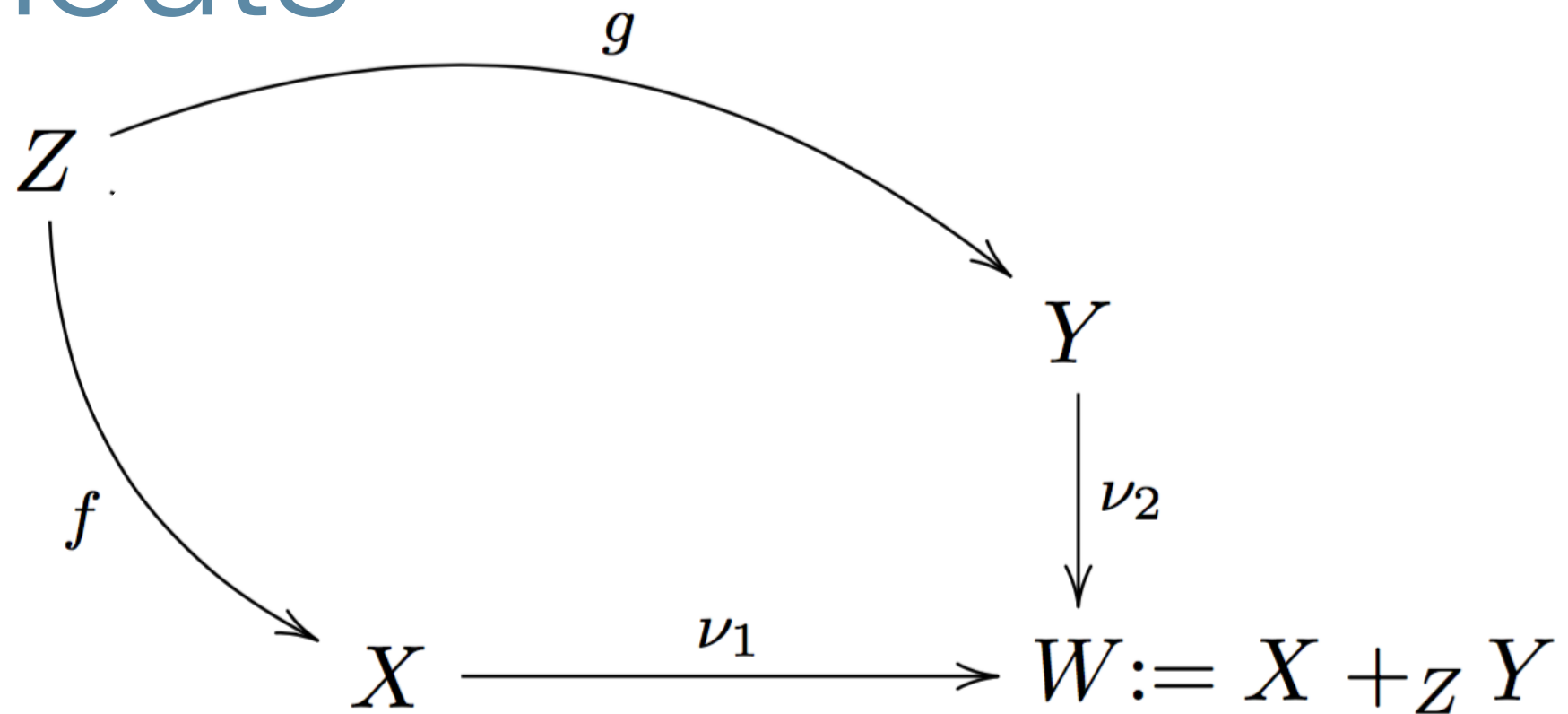
Pushouts



`inl : X → X +Z Y`

`inr : Y → X +Z Y`

Pushouts

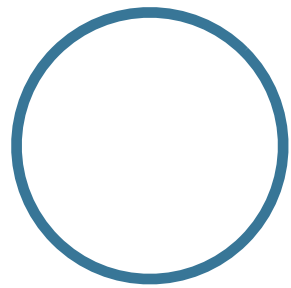


$\text{inl} : X \rightarrow X +_Z Y$

$\text{inr} : Y \rightarrow X +_Z Y$

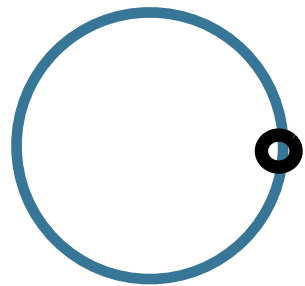
$\text{glue} : \prod (z : A) \rightarrow \text{inl}(f\ z) = \text{inr}\ (g\ z)$

Pushouts



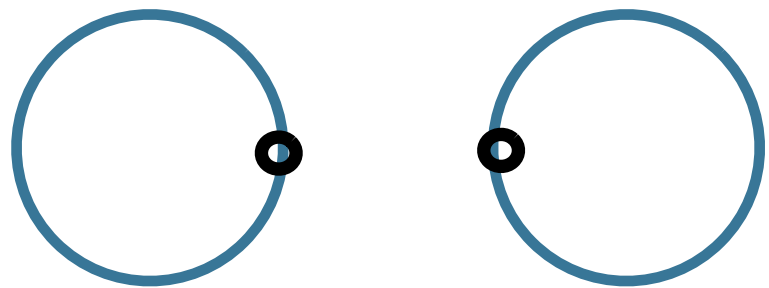
Pushouts

`base : 1 → Circle`



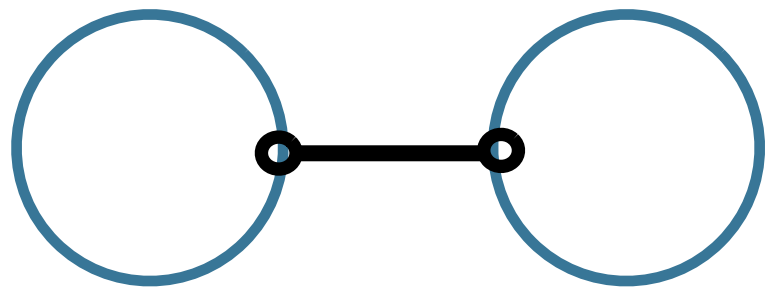
Pushouts

base : 1 $\rightarrow$ Circle



Pushouts

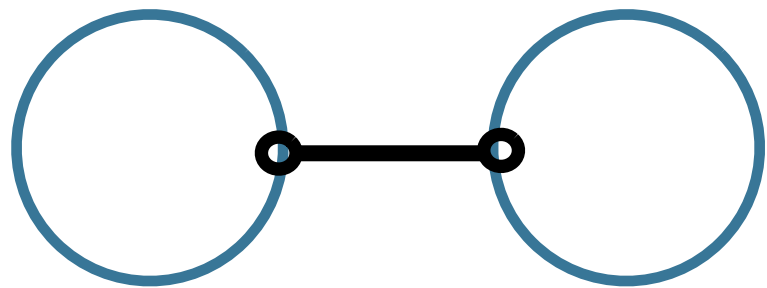
$\text{base} : 1 \rightarrow \text{Circle}$



$\text{Circle} +_1 \text{Circle}$

Pushouts

$\text{base} : 1 \rightarrow \text{Circle}$

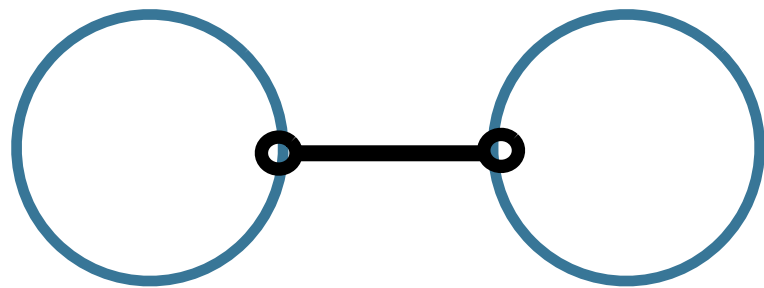


$\text{Circle} +_1 \text{Circle}$

$\text{inl} : \text{Circle} \rightarrow \text{Circle} +_1 \text{Circle}$

Pushouts

$\text{base} : 1 \rightarrow \text{Circle}$



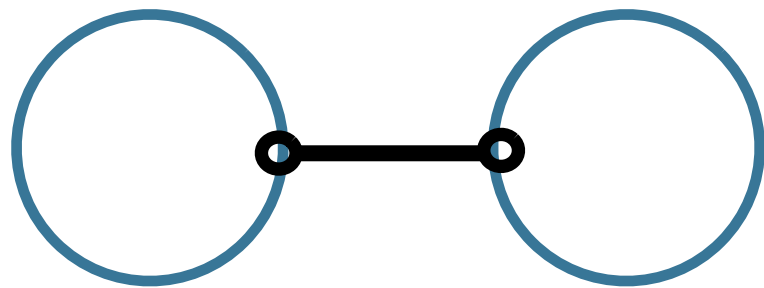
$\text{Circle} +_1 \text{Circle}$

$\text{inl} : \text{Circle} \rightarrow \text{Circle} +_1 \text{Circle}$

$\text{inr} : \text{Circle} \rightarrow \text{Circle} +_1 \text{Circle}$

Pushouts

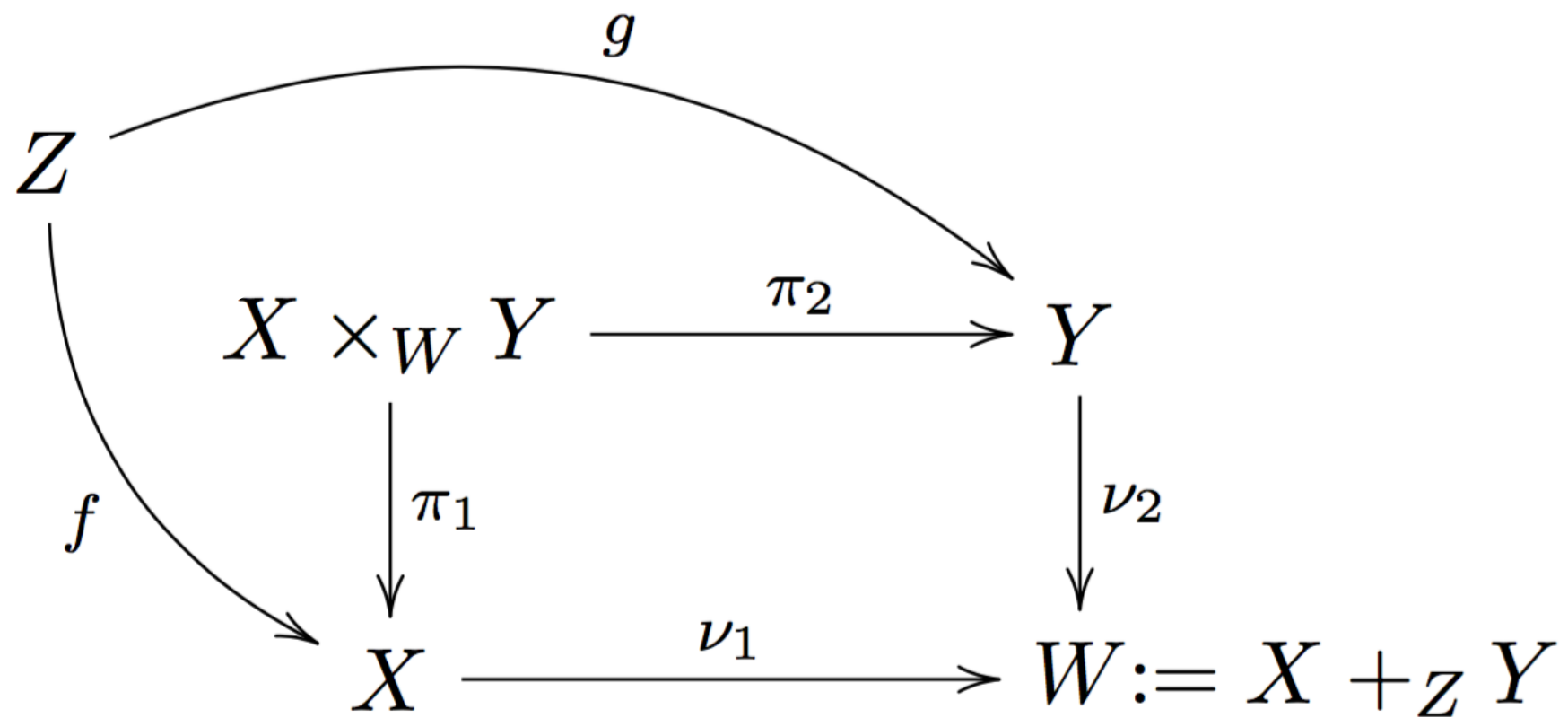
$\text{base} : 1 \rightarrow \text{Circle}$



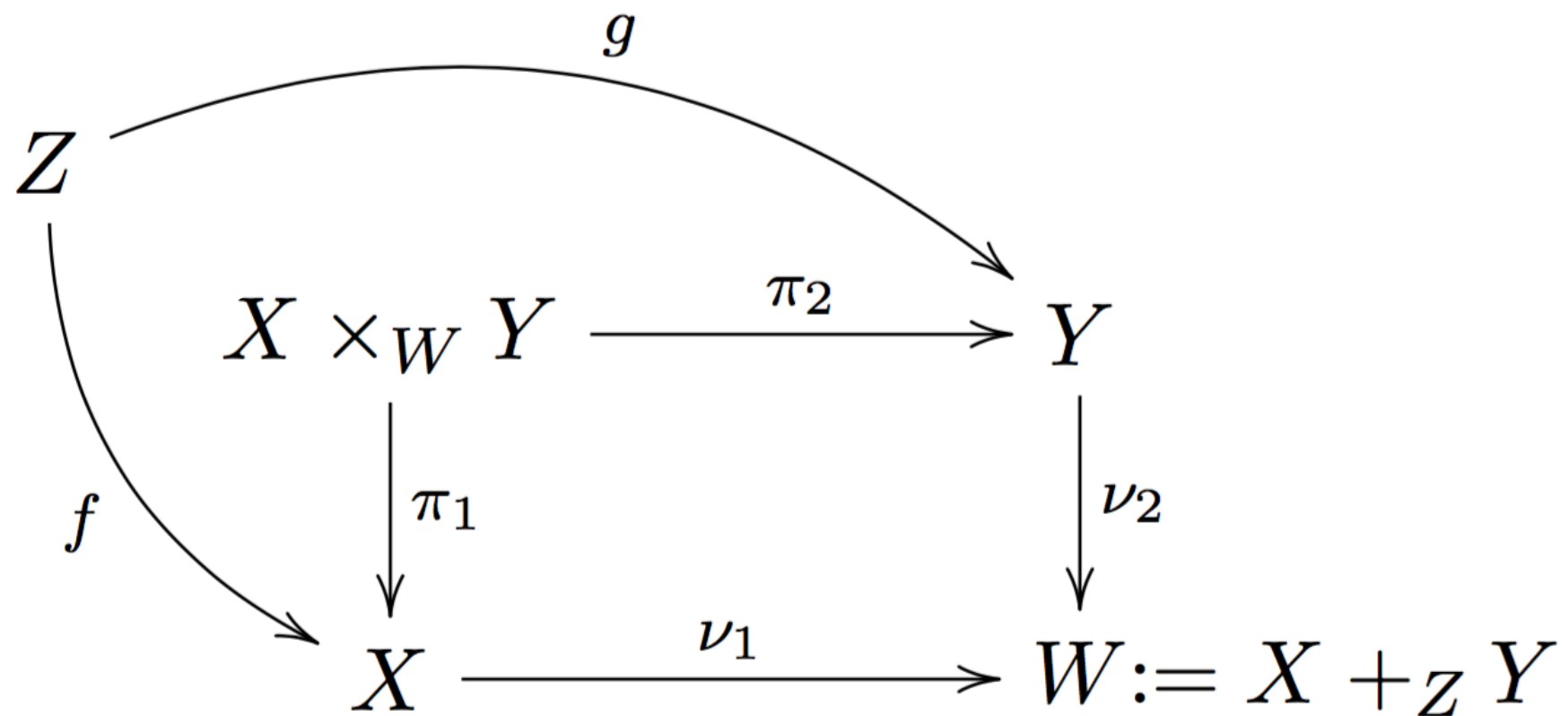
$\text{Circle} +_1 \text{Circle}$

$\text{inl} : \text{Circle} \rightarrow \text{Circle} +_1 \text{Circle}$
 $\text{inr} : \text{Circle} \rightarrow \text{Circle} +_1 \text{Circle}$
 $\text{glue} : \text{inl}(\text{base} \langle \rangle) = \text{inr}(\text{base} \langle \rangle)$

Pullbacks



Pullbacks



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

Homotopy Groups

$x : X$

$p : x =_x y$

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

$r : q_1 =_{p_1=p_2} q_2$

$\vdots$

Homotopy Groups

$$x : X$$

$$p : x =_x y$$

$$\pi_1(X)$$

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

$$r : q_1 =_{p_1=p_2} q_2$$

⋮

Homotopy Groups

$$x : X$$

$$p : x =_x y \quad \pi_1(X)$$

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

$$r : q_1 =_{p_1=p_2} q_2$$

⋮

Homotopy Groups

$$x : X$$

$$p : x =_x y \quad \pi_1(X)$$

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

$$r : q_1 =_{p_1=p_2} q_2 \quad \pi_3(X)$$

⋮

Connectedness

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

$$\pi_1(Z) \cong \pi_1(X)$$

$$\pi_2(Z) \cong \pi_2(X)$$

$$\pi_3(Z) \cong \pi_3(X)$$

$$\vdots$$
$$\vdots$$

$$\pi_n(Z) \cong \pi_n(X)$$

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

Connectedness

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

$$\pi_1(Z) \cong \pi_1(X)$$

$$\pi_2(Z) \cong \pi_2(X)$$

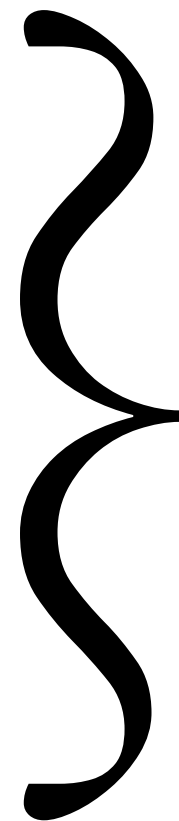
$$\pi_3(Z) \cong \pi_3(X)$$

$$\vdots$$

$$\vdots$$

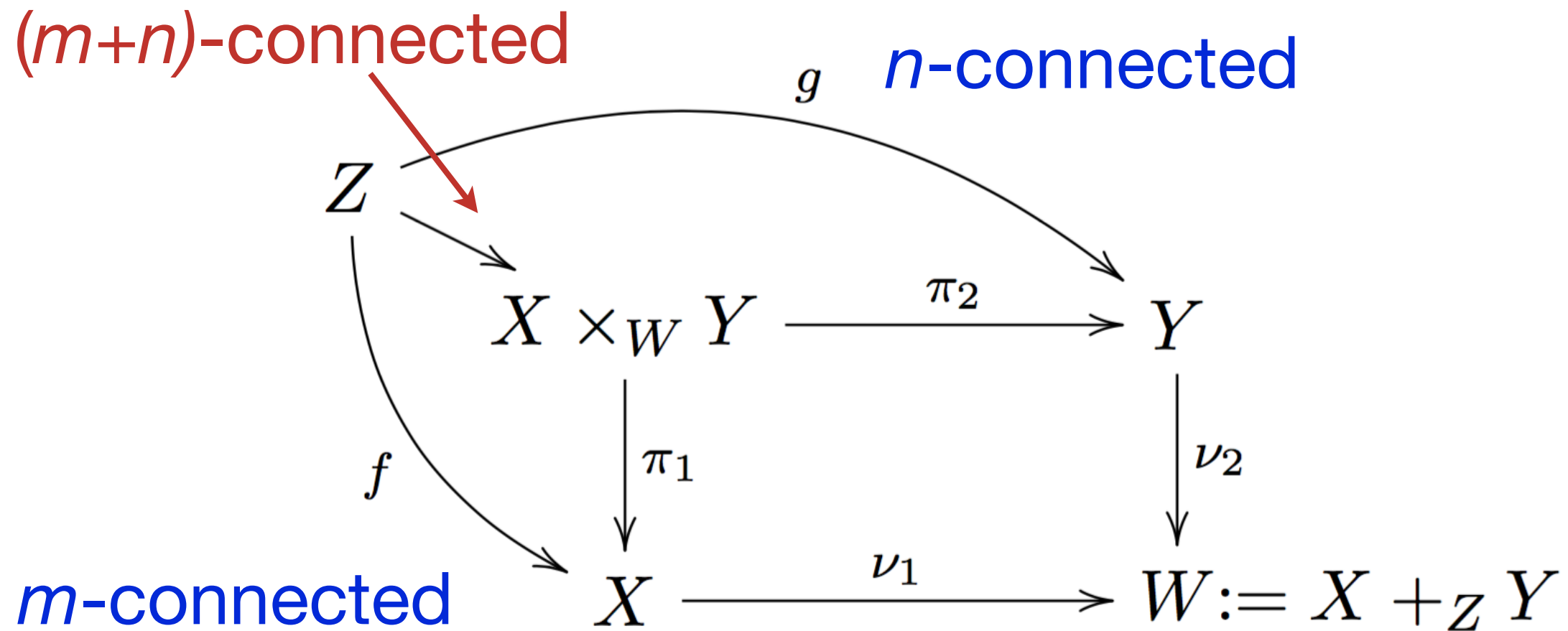
$$\pi_n(Z) \cong \pi_n(X)$$

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

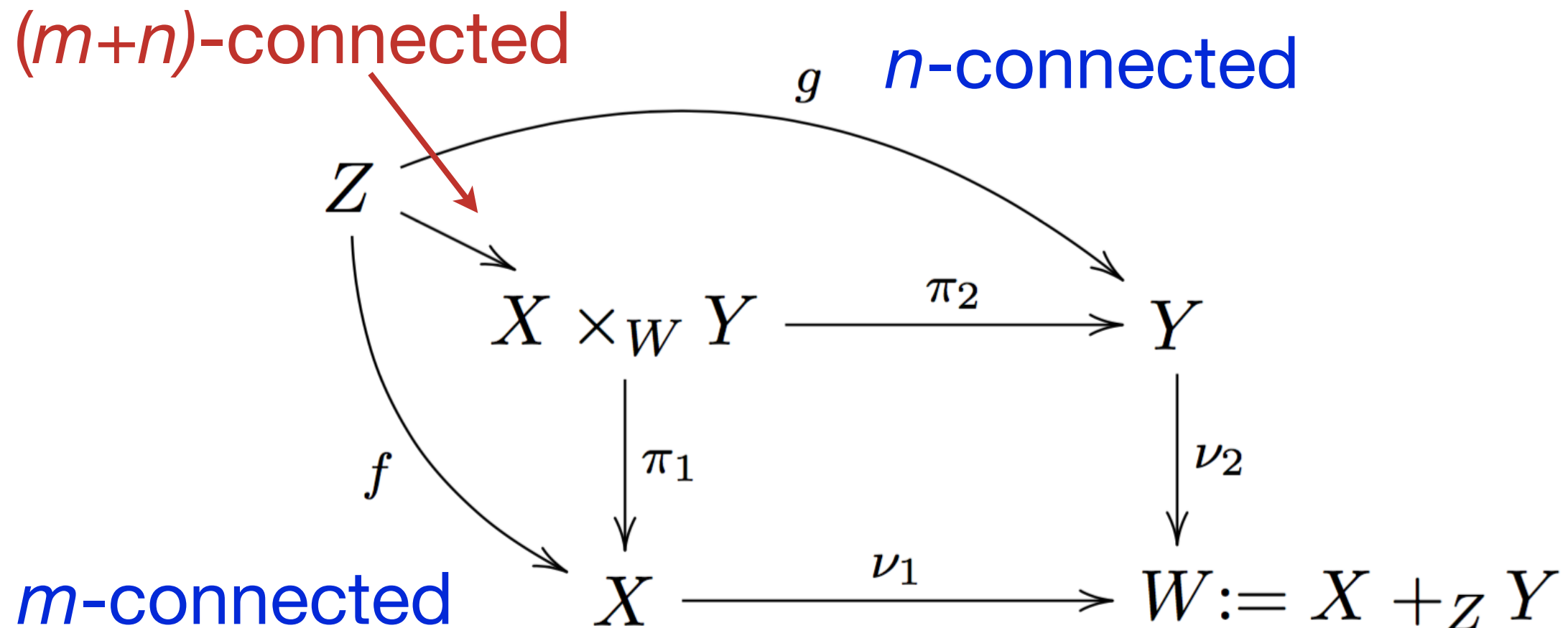


$$\|Z\|_n \cong \|X\|_n$$

Blakers-Massey Theorem



Blakers-Massey Theorem



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]

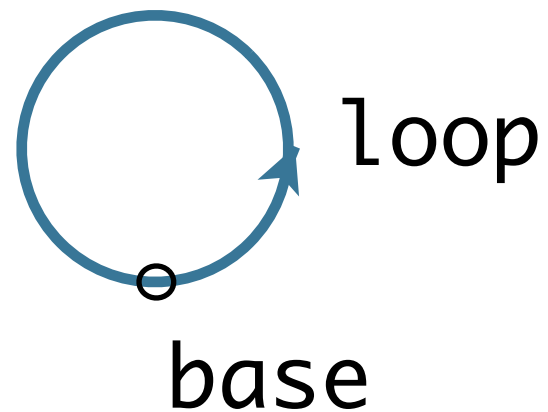
Blakers-Massey Theorem

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)))
```

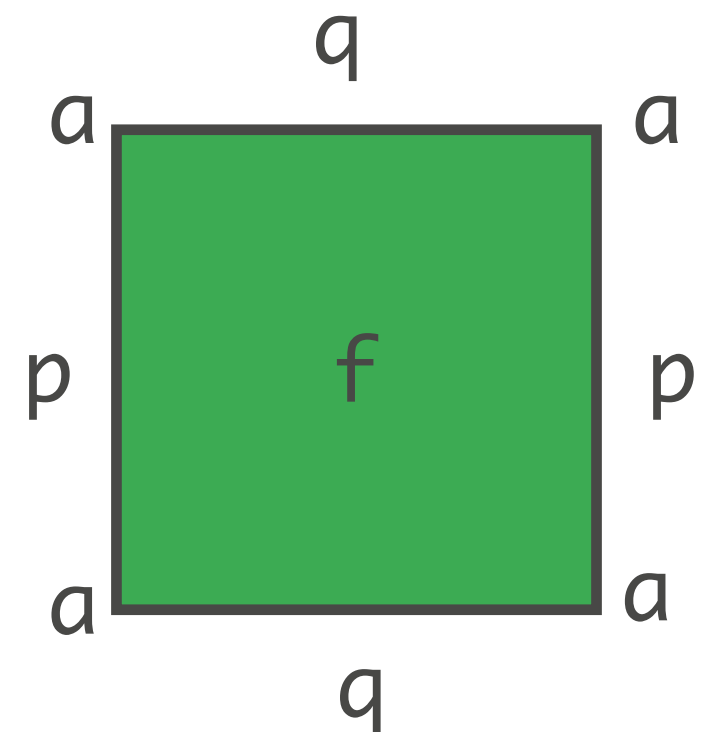
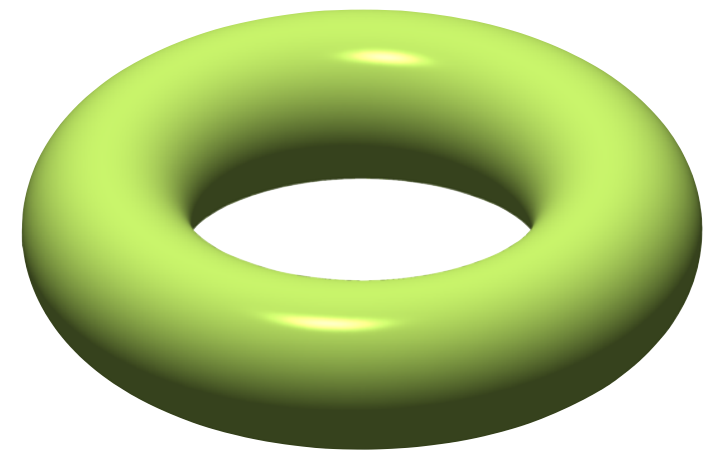
Cubical type theory

Circle and Torus

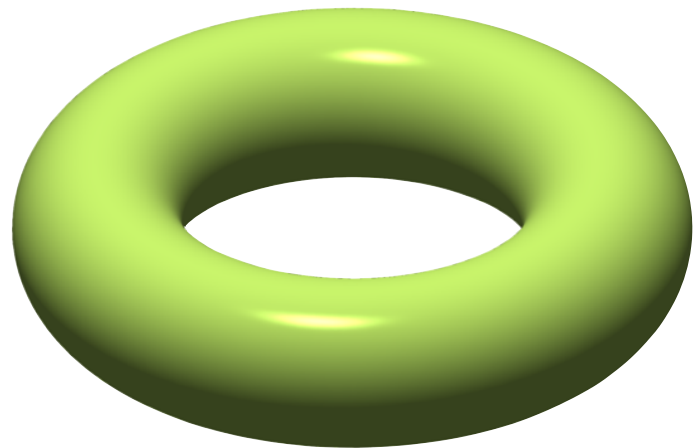
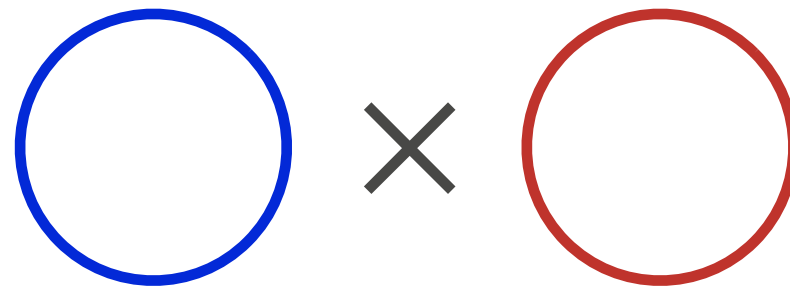


base : S^1

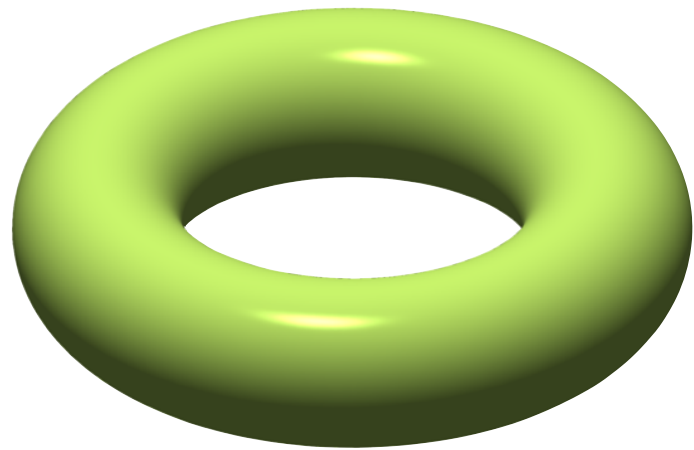
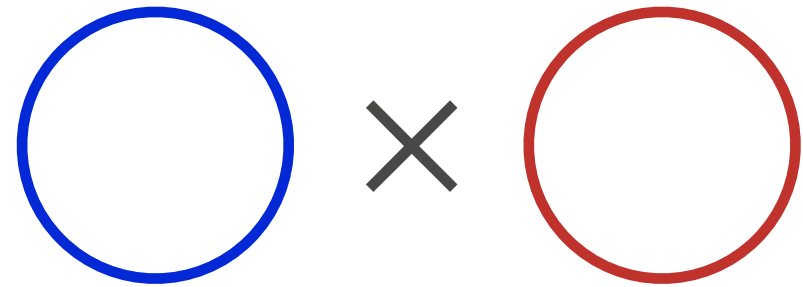
loop : base $\rightarrow_{S^1}$ base



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

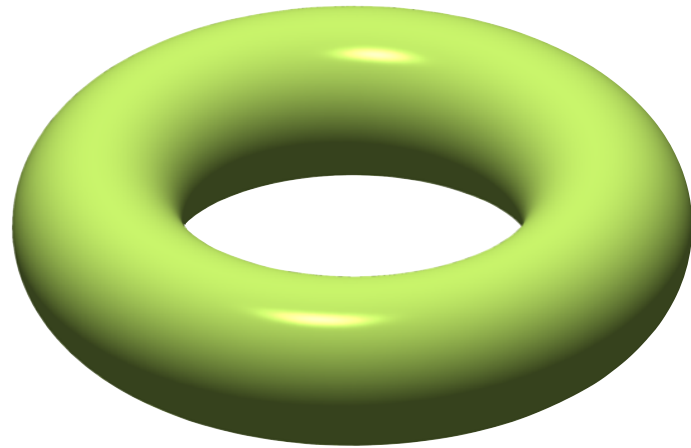
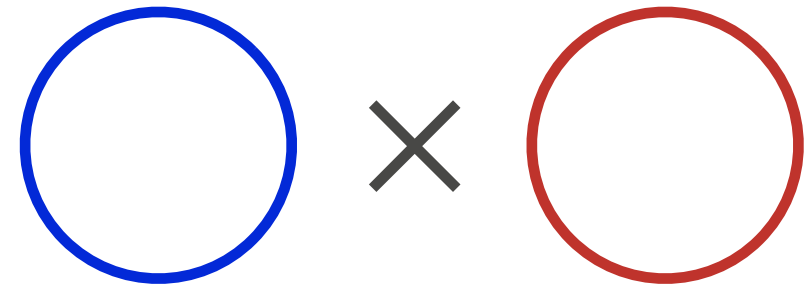
 $\simeq$ 

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

 $\simeq$ 

$$\text{t2c} : T \rightarrow S^1 \times S^1$$

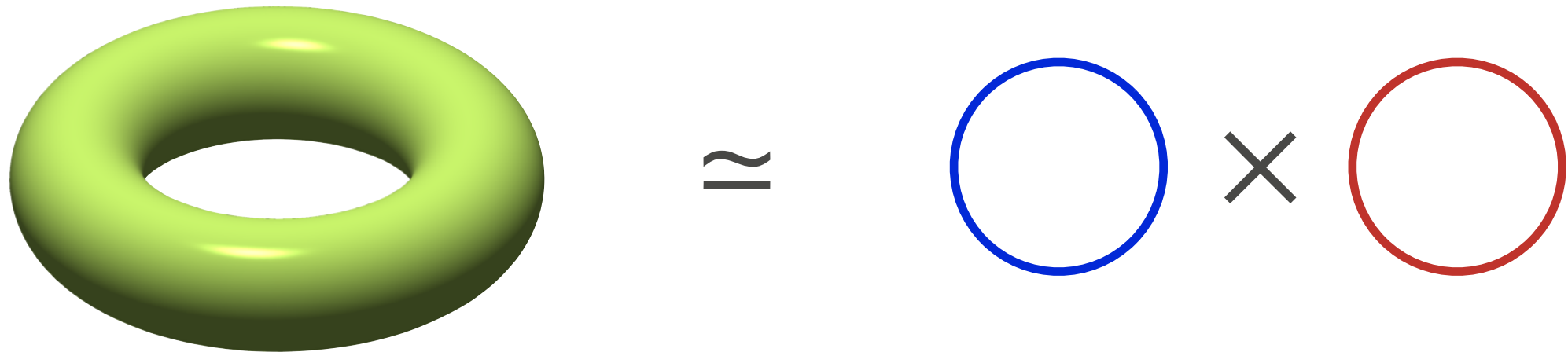
$$T \simeq S^1 \times S^1$$


 $\simeq$


$$t_{2c} : T \rightarrow S^1 \times S^1$$

$$c_{2t} : S^1 \times S^1 \rightarrow T$$

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

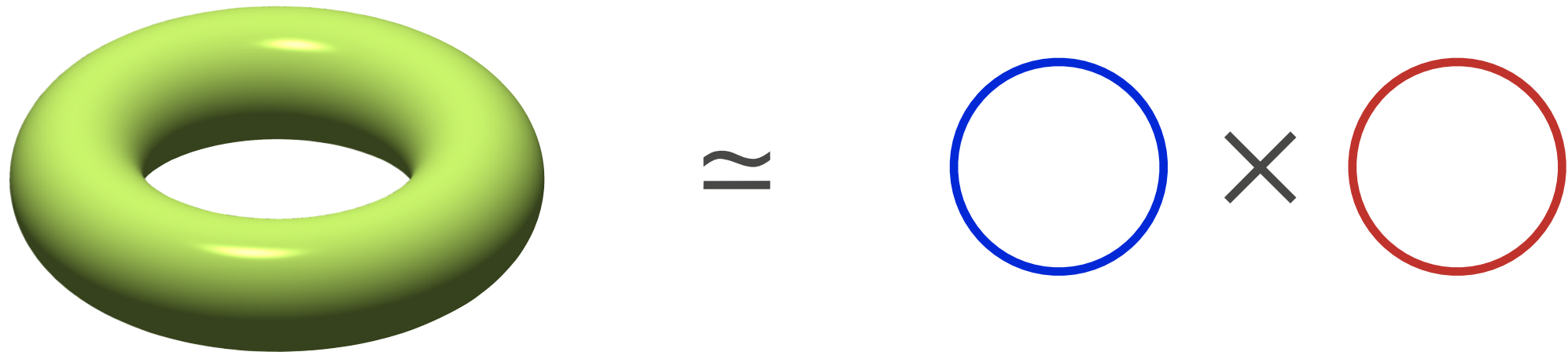


$$t_{2c} : T \rightarrow S^1 \times S^1$$

$$c_{2t} : S^1 \times S^1 \rightarrow T$$

$$c_{2t} t_{2c} : \Pi p : S^1 \times S^1. \quad t_{2c} (c_{2t} \ p) = p$$

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



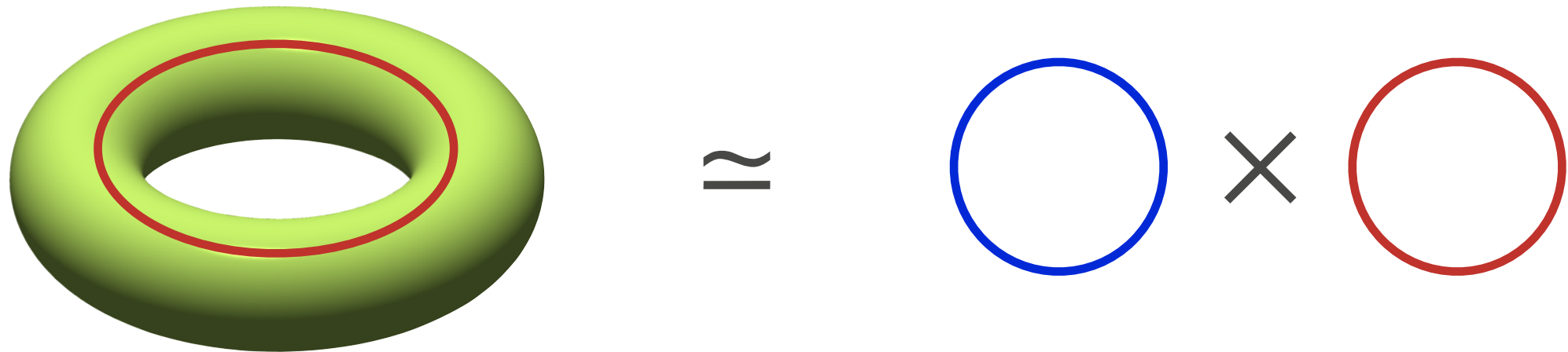
$$t_{2c} : T \rightarrow S^1 \times S^1$$

$$c_{2t} : S^1 \times S^1 \rightarrow T$$

$$c_{2t} t_{2c} : \prod p : S^1 \times S^1. \quad t_{2c} (c_{2t} p) = p$$

$$t_{2c} c_{2t} : \prod t : T. \quad c_{2t} (t_{2c} t) = t$$

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



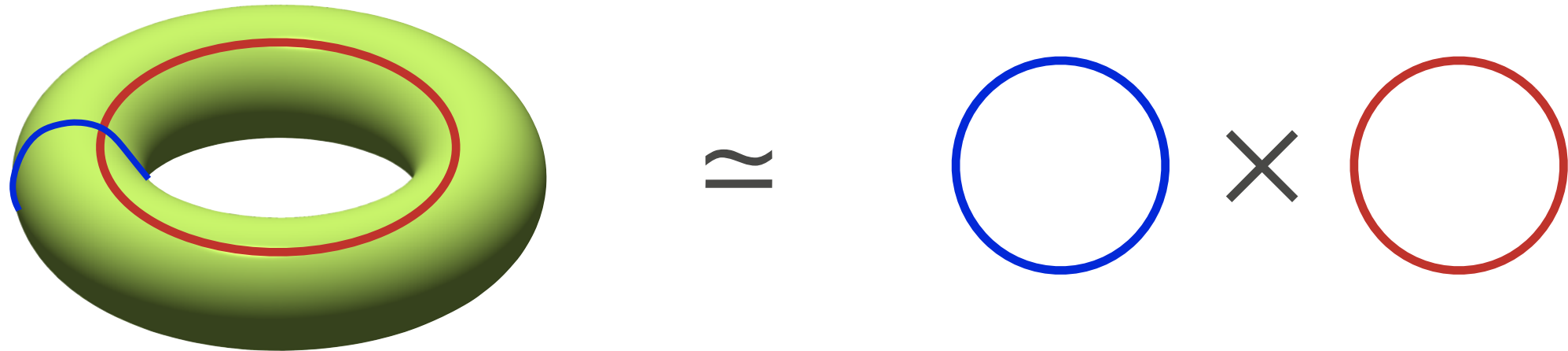
$$t_{2c} : T \rightarrow S^1 \times S^1$$

$$c_{2t} : S^1 \times S^1 \rightarrow T$$

$$c_{2t} t_{2c} : \Pi p : S^1 \times S^1. \quad t_{2c} (c_{2t} \ p) = p$$

$$t_{2c} c_{2t} : \Pi t : T. \quad c_{2t} (t_{2c} \ t) = t$$

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



$$t_{2c} : T \rightarrow S^1 \times S^1$$

$$c_{2t} : S^1 \times S^1 \rightarrow T$$

$$c_{2t} t_{2c} : \Pi p : S^1 \times S^1. \quad t_{2c} (c_{2t} \ p) = p$$

$$t_{2c} c_{2t} : \Pi t : T. \quad c_{2t} (t_{2c} \ t) = t$$

[Sojakova'13]

where for any family $E : T^2 \rightarrow \mathbf{type}$, paths $\alpha : x =_{T^2} y$, $\alpha' : y =_{T^2} z$ and point $u : E(x)$, the path

$$\mathcal{T}_f(E, \alpha, \alpha', u) : \text{trans}^E(\alpha \cdot \alpha', u) = \text{trans}^E(\alpha', \text{trans}^E(\alpha, u))$$

is obtained by path induction on α and α' . The inductor f then has the property that $f(b) \equiv b'$. Furthermore, there exist paths $\beta : \text{ap}_f(p) = p'$ and $\gamma : \text{ap}_f(q) = q'$ satisfying a higher coherence law, which we omit since we do not need it.

4 Logical equivalence between $\mathbf{S}^1 \times \mathbf{S}^1$ and T^2

Left-to-right We define a function $f : \mathbf{S}^1 \rightarrow T^2$ by circle recursion, mapping $\text{base} \mapsto b$ and $\text{loop} \mapsto p$. Thus, we have a definitional equality $f(\text{base}) \equiv b$ and a path $\beta_f : \text{ap}_f(\text{loop}) = p$. We define a function $F^{-1} : \mathbf{S}^1 \rightarrow \mathbf{S}^1 \rightarrow T^2$ again by circle recursion, mapping $\text{base} \mapsto f$ and $\text{loop} \mapsto \text{funext}(H)$, where $H : \Pi_{x:S^1} f(x) = f(x)$ is defined by circle induction as follows. We map base to q and loop to the path

$$\frac{\text{trans}^{b \mapsto f(x)} = f(x)(\text{loop}, q)}{\mathcal{T}_f(\text{loop}, q)} \text{ap}_f(\text{loop})^{-1} \cdot q \cdot \text{ap}_f(\text{loop})$$

Furthermore, we have a path $\beta_{f^{-1}} : \text{ap}_{f^{-1}}(\text{loop}) = \text{funext}(H)$. Since hap and funext form a quasi-equivalence, we have a path

$$\beta_{f^{-1}}^{-1} : \text{hap}(\text{ap}_{f^{-1}}(\text{loop})) = H$$

The function H is a homotopy between f and f such that $H(\text{base}) \equiv q$ and the following diagram commutes:

$$\begin{array}{ccc} \text{ap}_f(\text{loop}) \cdot q & \xrightarrow{\text{nat}_{H}(\text{loop})} & q \cdot \text{ap}_f(\text{loop}) \\ \text{via } \beta_f \downarrow & (1) & \downarrow \text{via } \beta_f \\ p \cdot q & \xrightarrow{\tau} & q \cdot p \end{array}$$

To show this, we note that for any $\alpha : x =_{S^1} y$, applying T^{-1} to the path

$$\frac{\text{ap}_f(\alpha)^{-1} \cdot H(x) \cdot \text{ap}_f(\alpha)}{\mathcal{T}_f(\alpha, H(x))^{-1}} \text{trans}^{b \mapsto f(x)} = f(x)(\alpha, H(x))$$

Right-to-left We define a function $G : T^2 \rightarrow \mathbf{S}^1 \times \mathbf{S}^1$ by torus recursion as follows. We map $b \mapsto (\text{base}, \text{base})$, $p \mapsto \text{pair}^{-1}(\text{base}, \text{loop})$, $q \mapsto \text{pair}^{-1}(\text{loop}, \text{base})$, and $\tau \mapsto \Phi_{\text{loop}, \text{loop}}$, where for any $\alpha : x =_{S^1} x'$, $\alpha' : y =_{S^1} y'$, the path

$$\Phi_{\alpha, \alpha'} : (\text{pair}^{-1}(\text{base}, \alpha') \cdot \text{pair}^{-1}(\alpha, \text{base})) = (\text{pair}^{-1}(\alpha, \text{base}) \cdot \text{pair}^{-1}(\text{base}, \alpha'))$$

is defined by induction on α and α' .

Then we have a definitional equality $G(b) \equiv (\text{base}, \text{base})$ and paths

$$\beta_G^p : \text{ap}_G(p) = \text{pair}^{-1}(\text{base}, \text{loop}) \\ \beta_G^q : \text{ap}_G(q) = \text{pair}^{-1}(\text{loop}, \text{base})$$

which make the following diagram commute:

$$\begin{array}{ccc} \text{ap}_G(p \cdot q) & \xrightarrow{\text{via } \tau} & \text{ap}_G(q \cdot p) \\ \text{ap}_G(p) \cdot \text{ap}_G(q) \downarrow & (2) & \downarrow \text{ap}_G(q) \cdot \text{ap}_G(p) \\ \text{via } \beta_G^p, \beta_G^q & & \downarrow \text{via } \beta_G^q, \beta_G^p \end{array}$$

$$\begin{array}{c} \text{ap}_G(\text{ap}_f(\text{loop})) \cdot 1_{(\text{base}, \text{base})} \\ \downarrow \text{via } \beta_f \\ \text{ap}_G(p) \\ \downarrow \beta_G^p \\ \text{pair}^{-1}(\text{base}, \text{loop}) \\ \downarrow 1_{(\text{base}, \text{base})} \cdot \text{pair}^{-1}(\text{base}, \text{loop}) \end{array}$$

This finishes the definition of ϵ . We now need to prove that

$$\text{trans}^{b \mapsto \epsilon}(y, S^1 \times G(F(x, y))) = (x, y)(\text{loop}, \epsilon) = \epsilon$$

By function extensionality, it suffices to show that for any $y : \mathbf{S}^1$ we have

$$\text{trans}^{b \mapsto \epsilon}(x, S^1 \times G(F(x, y))) = (x, y)(\text{loop}, \epsilon) \quad y = \epsilon(y)$$

Straightforward path induction shows that for any $\alpha : \text{base} =_{S^1} x$, we have

All that now remains to show is

$$\text{trans}^{b \mapsto \text{ap}_G(H(y)) \cdot \epsilon(y)} = \epsilon(y) \cdot \text{pair}^{-1}(\text{loop}, 1_b)(\text{loop}, \eta) = \eta$$

However, this follows at once from the fact that the circle $\mathbf{S}^1$, and hence the product $\mathbf{S}^1 \times \mathbf{S}^1$, is a 1-type (as shown e.g., by Licata and Shulman in [9]); this means that for any two points $x, y : \mathbf{S}^1 \times \mathbf{S}^1$, any two paths $\alpha, \alpha' : x = y$, and any two higher paths $\gamma, \gamma' : \alpha = \alpha'$, we necessarily have $\gamma = \gamma'$.

Right-to-left We need to show that for any $x : T^2$ we have $F(G(x)) = x$. We use torus induction with $b' := 1_b$. We let p' be the path

$$\frac{\text{trans}^{b \mapsto F(G(x))} = x(p, 1_b)}{\mathcal{T}_S(p, 1_b)} \text{ap}_F(\text{ap}_G(p))^{-1} \cdot 1_b \cdot p$$

To show this, we proceed by path induction on α_1 and α_2 . Hence we have to establish the claim for $\alpha_1 := 1_x$, $\alpha_2 := 1_x$, $u_x, u_y, u_z : F(G(x)) = x$, and $\eta_1 : 1_{F(G(x))} \cdot u_y = u_x \cdot 1_x$, $\eta_2 : 1_{F(G(x))} \cdot u_z = u_y \cdot 1_x$.

We note, however, that the types of η_1, η_2 are equivalent to $u_x = u_y$ and $u_y = u_z$ respectively. Hence it suffices to show that given $u_x, u_y, u_z : F(G(x)) = x$, $\eta'_1 : u_x = u_y$, $\eta'_2 : u_y = u_z$, we can establish the claim for the special case when η_1 and η_2 have been obtained from η'_1 and η'_2 , respectively, by using the aforementioned equivalences.

But we can now perform path induction on η'_1 and η'_2 , leaving us with $u_x : F(G(x)) = x$ and $\eta'_1 := 1_{u_x}$, $\eta'_2 := 1_{u_x}$. We finish the proof by generalizing the endpoints of u_x and performing a final path induction.

By what we have just shown, it suffices to prove that the following diagram commutes:

$$\begin{array}{ccc} \text{trans}^{b \mapsto F(G(x))} = w(p \cdot q, 1_b) & \xrightarrow{\text{via } \tau} & \text{trans}^{b \mapsto F(G(x))} = w(q \cdot p, 1_b) \\ \downarrow \mathcal{T}_S(p \cdot q, 1_b) & & \downarrow \mathcal{T}_S(q \cdot p, 1_b) \\ \text{ap}_F(\text{ap}_G(p \cdot q))^{-1} \cdot 1_b \cdot (p \cdot q) & & \text{ap}_F(\text{ap}_G(q \cdot p))^{-1} \cdot 1_b \cdot (q \cdot p) \\ \downarrow \mathcal{I}(\langle p, q, p, q, 1_b, 1_b, 1_b, \kappa_p, \kappa_q \rangle) & & \downarrow \mathcal{I}(\langle q, p, q, p, 1_b, 1_b, 1_b, \kappa_q, \kappa_p \rangle) \\ 1_b & & 1_b \end{array}$$

$$\begin{array}{ccc} \text{ap}_F(\text{ap}_G(\alpha)) \cdot u_y & \xrightarrow{\text{via } \theta} & \text{ap}_F(\text{ap}_G(\alpha')) \cdot u_y \\ \eta \downarrow & & \downarrow \eta' \\ u_x \cdot \alpha & \xrightarrow{\text{via } \theta} & u_x \cdot \alpha' \end{array}$$

To show this, we proceed by path induction on θ and a subsequent path induction on α . After simplifying it remains to prove that for $u_x, u_y : F(G(x)) = x$, $\eta, \eta' : 1_{F(G(x))} \cdot u_y = u_x \cdot 1_x$, we have $(\mathcal{I}(\eta) = \mathcal{I}(\eta')) \simeq (\eta = \eta')$. But this follows since $\mathcal{I}$ is an equivalence.

By what we have just shown, it suffices to prove that the following diagram commutes:

$$\begin{array}{ccc} \text{ap}_F(\text{ap}_G(p \cdot q)) \cdot 1_b & \xrightarrow{\text{via } \tau} & \text{ap}_F(\text{ap}_G(q \cdot p)) \cdot 1_b \\ \downarrow \zeta(p, q, p, q, 1_b, 1_b, 1_b, \kappa_p, \kappa_q) & & \downarrow \zeta(q, p, q, p, 1_b, 1_b, 1_b, \kappa_q, \kappa_p) \\ 1_b \cdot (p \cdot q) & \xrightarrow{\text{via } \tau} & 1_b \cdot (q \cdot p) \end{array}$$

Step 3 We observe the following: for $k \in \{1, 2\}$ and $x_1, x_2, x_3 : T^2$ let terms $\alpha_k^1 : x_k = x_{k+1}$; $\alpha_k^2 : G(x_k) = G(x_{k+1})$; $\alpha_k^3 : F(G(x_k)) = F(G(x_{k+1}))$; $i_k^1 : \text{ap}_F(\alpha_k^1) = \alpha_k^2$; $i_k^2 : \text{ap}_F(\alpha_k^2) = \alpha_k^3$; $i_k^3 : \alpha_k^2 = \alpha_k^3$ be given. Then the path

$$\begin{array}{c} \text{ap}_F(\text{ap}_G(\alpha_1^1 \cdot \alpha_2^1)) \cdot 1_{F(G(x_3))} \\ \downarrow \\ \text{ap}_F(\text{ap}_G(\alpha_1^1 \cdot \alpha_2^1)) \\ \downarrow \\ \text{ap}_F(\text{ap}_G(\alpha_1^1) \cdot \text{ap}_G(\alpha_2^1)) \\ \downarrow \text{via } i_1^1, i_2^1 \\ \text{ap}_F(\alpha_1^2 \cdot \alpha_2^2) \\ \downarrow \\ \text{ap}_F(\alpha_1^2) \cdot \text{ap}_F(\alpha_2^2) \\ \downarrow \text{via } i_1^2, i_2^2 \\ \alpha_1^3 \cdot \alpha_2^3 \\ \downarrow \text{via } i_1^3, i_2^3 \\ \alpha_1^4 \cdot \alpha_2^4 \\ \downarrow 1_{F(G(x_3))} \cdot (\alpha_1^4 \cdot \alpha_2^4) \end{array}$$

is equal to the path $\zeta(\alpha_1^1, \alpha_2^1, \alpha_1^2, \alpha_2^2, 1_{F(G(x_1))}, 1_{F(G(x_2))}, 1_{F(G(x_3))}, \eta_1, \eta_2)$ where η_k is the path

$$\begin{array}{c} \text{ap}_F(\text{ap}_G(\alpha_k^1)) \cdot 1_{F(G(x_{k+1}))} \\ \downarrow \\ \text{ap}_F(\text{ap}_G(\alpha_k^1)) \\ \downarrow \text{via } i_k^1 \\ \text{ap}_F(\alpha_k^2) \\ \downarrow i_k^2 \\ \alpha_k^3 \\ \downarrow i_k^3 \\ \alpha_k^4 \end{array}$$

To show this, we proceed by path induction (with one endpoint fixed) on i_k^1, i_k^2, i_k^3 and a subsequent path induction on α_k^1 .

By what we have just shown, it suffices to prove that the outer rectangle in the following diagram commutes:

$$\begin{array}{ccc} \text{ap}_F(\text{ap}_G(p \cdot q)) \cdot 1 & \xrightarrow{\text{via } \tau} & \text{ap}_F(\text{ap}_G(q \cdot p)) \cdot 1 \\ \downarrow & A & \downarrow \\ \text{ap}_F(\text{ap}_G(p \cdot q)) & \xrightarrow{\text{via } \tau} & \text{ap}_F(\text{ap}_G(q \cdot p)) \\ \downarrow & B & \downarrow \\ \text{ap}_F(\text{ap}_G(p) \cdot \text{ap}_G(q)) & & \text{ap}_F(\text{ap}_G(q) \cdot \text{ap}_G(p)) \\ \downarrow \text{via } \beta_G^p, \beta_G^q & & \downarrow \text{via } \beta_G^q, \beta_G^p \\ \text{ap}_F(\text{pair}^{-1}(\text{base}, \text{loop}) \cdot \text{pair}^{-1}(\text{loop}, 1)) & \xrightarrow{\text{via } \Phi_{\text{loop}, \text{loop}}} & \text{ap}_F(\text{pair}^{-1}(\text{loop}, 1) \cdot \text{pair}^{-1}(\text{base}, \text{loop})) \\ \downarrow & C & \downarrow \\ \text{ap}_F(\text{pair}^{-1}(\text{base}, \text{loop}) \cdot \text{pair}^{-1}(\text{loop}, 1)) & & \text{ap}_F(\text{pair}^{-1}(\text{loop}, 1) \cdot \text{pair}^{-1}(\text{base}, \text{loop})) \\ \downarrow \text{via } \mu_{\text{base}}(\text{loop}), \mu_{\text{loop}}(\text{base}) & & \downarrow \text{via } \mu_{\text{loop}}(\text{base}), \mu_{\text{base}}(\text{loop}) \\ \text{ap}_F(\text{loop}) \cdot \text{hap}(\text{ap}_{f^{-1}}(\text{loop}), \text{base}) & \xrightarrow{\text{via } \text{nat}_{\text{hap}(\text{ap}_{f^{-1}}(\text{loop}), \text{base})}(\text{loop})} & \text{hap}(\text{ap}_{f^{-1}}(\text{loop}), \text{base}) \cdot \text{ap}_F(\text{loop}) \\ \downarrow \text{via } \beta_f, \text{hap}(\beta_{f^{-1}}, \text{base}) & D & \downarrow \text{via } \text{hap}(\beta_{f^{-1}}, \text{base}), \beta_f \\ p \cdot q & \xrightarrow{\tau} & q \cdot p \\ \downarrow \text{via } \tau & E & \downarrow \text{via } \tau \\ 1 \cdot (p \cdot q) & & 1 \cdot (q \cdot p) \end{array}$$

Step 4 It suffices to prove that each of the inner rectangles commutes. Rectangles A and E commute obviously. Rectangle B is just diagram (2) transported along ap_F , and hence commutes. Rectangle C commutes by the following generalization: for any $\alpha : x =_{S^1} y$, the diagram below commutes by path induction on α :

$$\begin{array}{ccc} \text{ap}_F(\text{pair}^{-1}(\text{base}, \alpha) \cdot \text{pair}^{-1}(\alpha, 1_b)) & \xrightarrow{\text{via } \Phi_{\alpha, \alpha}} & \text{ap}_F(\text{pair}^{-1}(\alpha, 1_b) \cdot \text{pair}^{-1}(\text{base}, \alpha)) \\ \downarrow \text{via } \mu_{\alpha}(\alpha), \mu_b(\alpha) & & \downarrow \text{via } \mu_b(\alpha), \mu_{\alpha}(\alpha) \\ \text{ap}_{f^{-1}(\alpha)}(\alpha) \cdot \text{hap}(\text{ap}_{f^{-1}}(\alpha), y) & \xrightarrow{\text{via } \text{nat}_{\text{hap}(\text{ap}_{f^{-1}}(\alpha), y)}(\alpha)} & \text{hap}(\text{ap}_{f^{-1}}(\alpha), x) \cdot \text{ap}_{f^{-1}(y)}(\alpha) \end{array}$$

It remains to show that rectangle D commutes. Consider the following diagram:

$$\begin{array}{ccc} \text{ap}_F(\text{loop}) \cdot \text{hap}(\text{ap}_{f^{-1}}(\text{loop}), \text{base}) & \xrightarrow{\text{via } \text{nat}_{\text{hap}(\text{ap}_{f^{-1}}(\text{loop}), \text{base})}(\text{loop})} & \text{hap}(\text{ap}_{f^{-1}}(\text{loop}), \text{base}) \cdot \text{ap}_F(\text{loop}) \\ \downarrow \text{via } \text{hap}(\beta_{f^{-1}}, \text{base}) & D_1 & \downarrow \text{via } \text{hap}(\beta_{f^{-1}}, \text{base}) \\ \text{ap}_F(\text{loop}) \cdot q & \xrightarrow{\text{via } \text{nat}_{H}(\text{loop})} & q \cdot \text{ap}_F(\text{loop}) \\ \downarrow \text{via } \beta_f & D_2 & \downarrow \text{via } \beta_f \\ p \cdot q & \xrightarrow{\tau} & q \cdot p \end{array}$$

Commutativity of the outer rectangle clearly implies the commutativity of D. It thus remains to show that D₁ and D₂ commute. The rectangle D₂ is precisely diagram (1), which commutes. Rectangle D₁ commutes by the following generalization: let $\gamma : h_1 =_{f^{-1}} h_2$ and $\alpha : x =_{S^1} y$ be given. Then the following diagram commutes by path induction on γ and α :

$$\begin{array}{ccc} \text{ap}_F(\alpha) \cdot h_1(y) & \xrightarrow{\text{via } \text{nat}_{h_1}(\alpha)} & h_1(x) \cdot \text{ap}_F(\alpha) \\ \downarrow \text{via } \text{hap}(\gamma, y) & & \downarrow \text{via } \text{hap}(\gamma, x) \\ \text{ap}_F(\alpha) \cdot h_2(y) & \xrightarrow{\text{via } \text{nat}_{h_2}(\alpha)} & h_2(x) \cdot \text{ap}_F(\alpha) \end{array}$$

This finishes the proof.

6 Conclusion

We have presented a homotopy-type theoretic proof that the torus T^2 is equivalent to the product of two circles $\mathbf{S}^1 \times \mathbf{S}^1$. To compare the proof described here to the one given by

[Sojakova'13]

Cubical Methods

Models

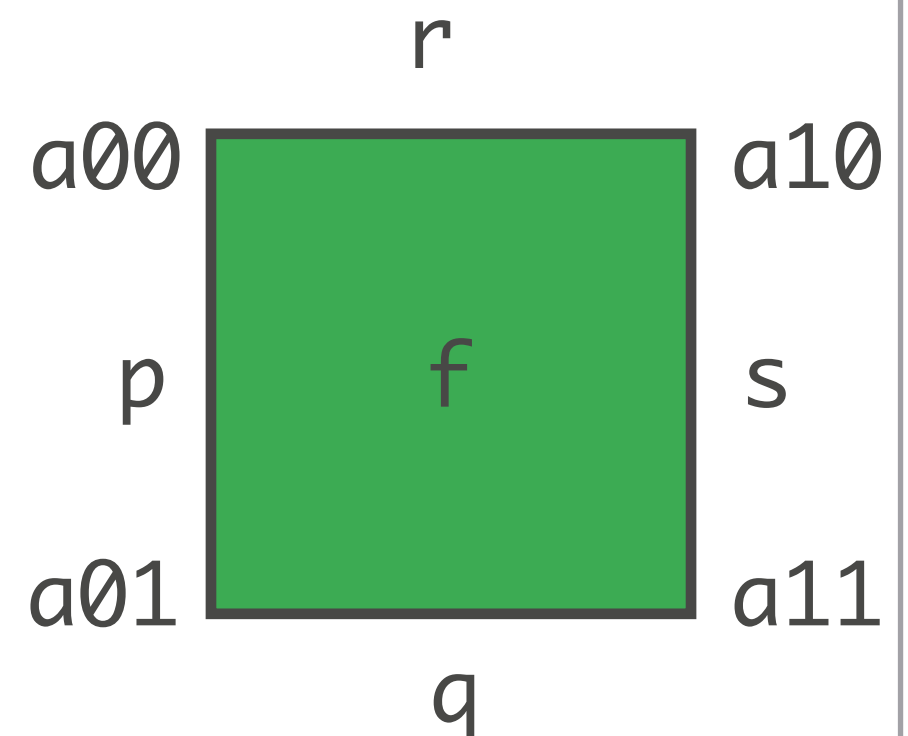
- * Bezem, Coquand, Huber '13
- * Cohen, Coquand, Huber, Mörtberg
- * Awodey, Frey, Hofstra
- * Gambino, Sattler
- * Orton, Pitts
- * Angiuli, Cavallo, Favonia, Harper, Sterling

Syntax

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

Cubical Types in MLTT

[L. and Brunerie'15]



Cubical Types in MLTT

data Square {A : Type} {a00 : A} : [L. and Brunerie'15]

{a01 a10 a11 : A}

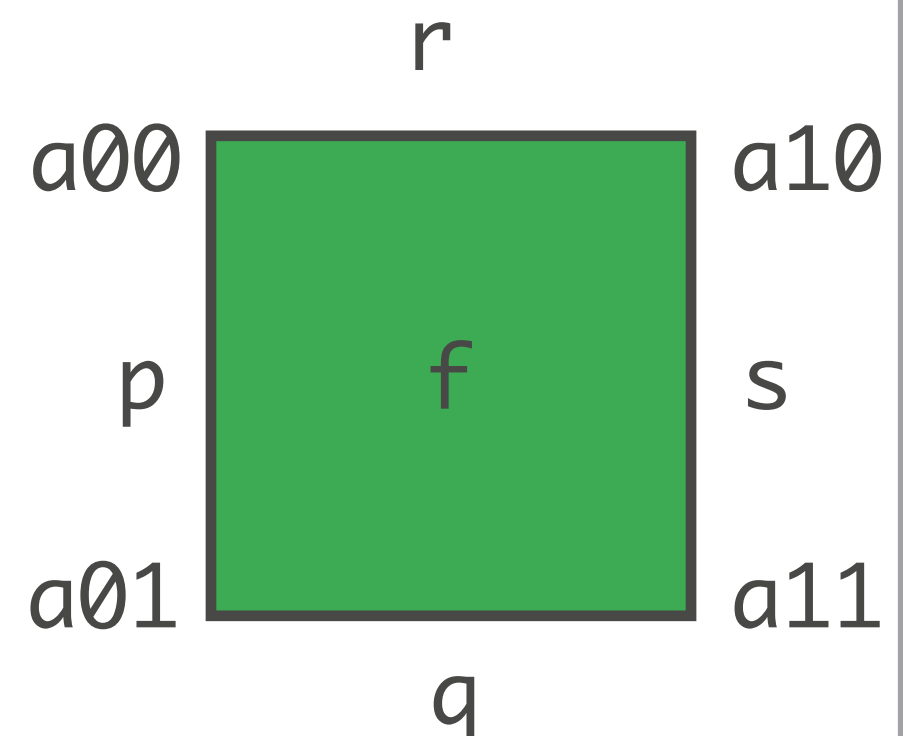
→ a00 == a01

→ a00 == a10

→ a01 == a11

→ a10 == a11 → Type where

id : Square id id id id



Cubical Types in MLTT

data Square {A : Type} {a00 : A} : [L. and Brunerie'15]

{a01 a10 a11 : A}

→ a00 == a01

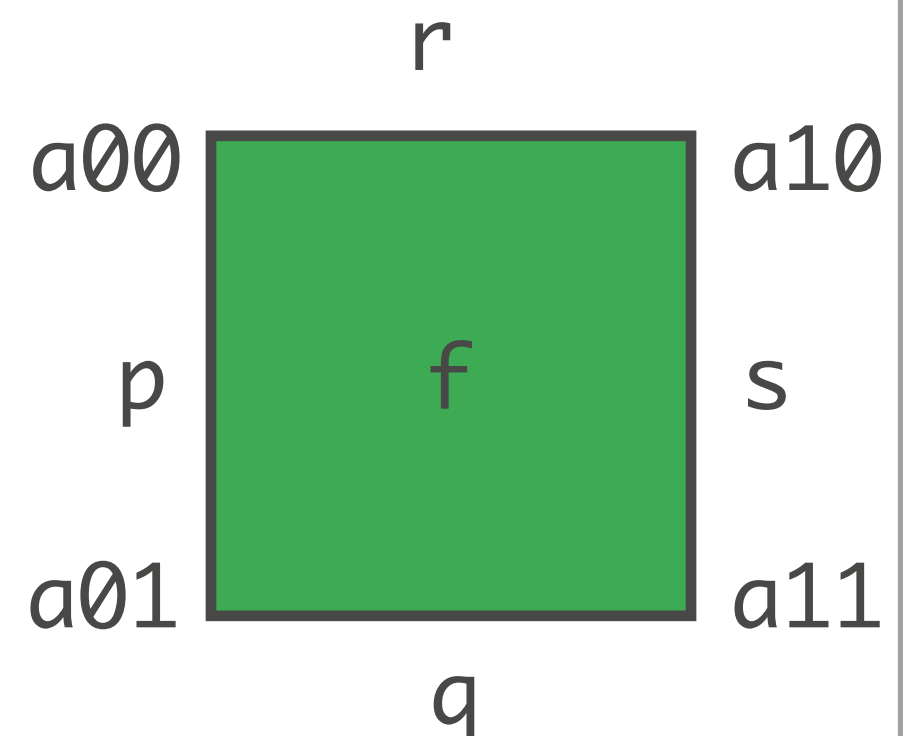
→ a00 == a10

→ 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 → S1 × S1
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 (S1-rec a q) loop) (ap (S1-rec a q) loop) p) →
  Cube s f hrefl-square βsquare βsquare hrefl-square
c2t-square-and-cube = (fill-cube-left _ _ _ _ _ _ _)
c2t-loop-homotopy = S1-elim _ p (in-PathOver = (fst c2t-square-and-cube))
c2t' : S1 → S1 → T
c2t' = S1-rec (S1-rec a q) (λ ≈ c2t-loop-homotopy)
c2t : S1 × S1 → 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-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))
    (S1.βloop/rec (S1-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 = (S1.βloop/elim _ 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 βcube --cube-h
      apdo-ap-cube-hv c2t' loop loop --cube-h
      snd (snd c2t'-β) --cube-h
      ap-square-id! f

c2t2c : (x y : S1) → Path (t2c (c2t' x y)) (x, y)
c2t2c = S1-elim _ (S1-elim _ id (in-PathOver = (square-symmetry loop2-case)))
  (coe (! PathOverII-NDdomain)
    (in-PathOver = od1 (S1-elim _ (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
      βcube --cube-h
      ap-square-id! _ --cube-h
      apdo-ap-cube-hv _,_ loop loop

red = ! (ap out-PathOver = (S1.βloop/elim _ id (in-PathOver = (square-symmetry loop2-case)))) .
  IsEquiv.β (snd out-PathOver = eqv) (square-symmetry loop2-case))
  
```

Cubical Syntax

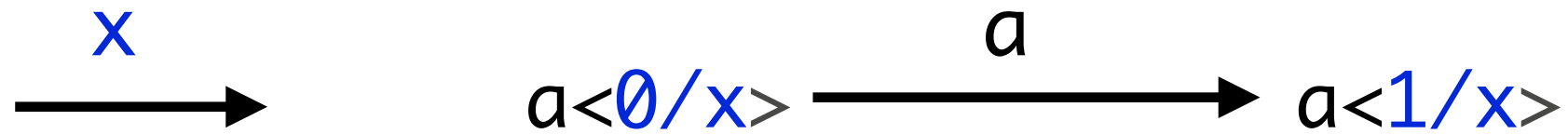
Cubical Syntax

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

Cubical Syntax

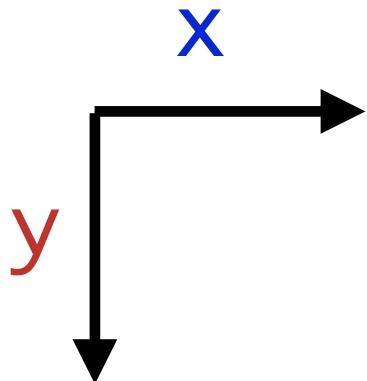
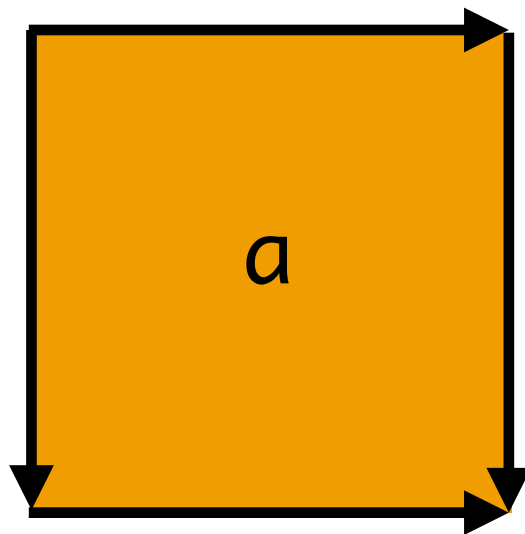
- * Extends MLTT with abstract interval variables (endpoints 0 and 1) and new constructs using them (univalence and HITs)
- * Models in cubical sets (Q: other ∞ -toposes?)

Faces

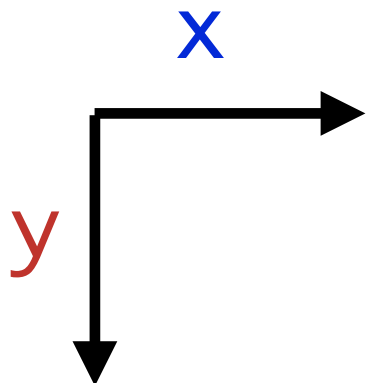
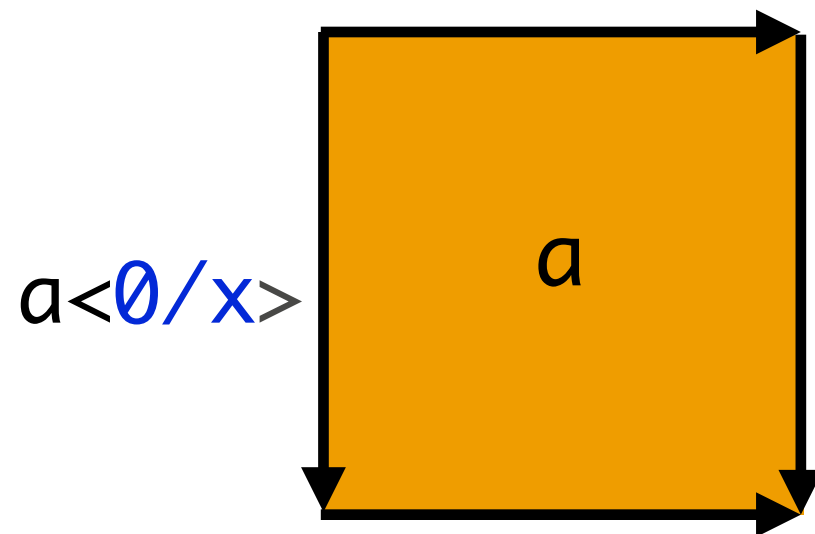
$$x:I \vdash a : A$$


Faces

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

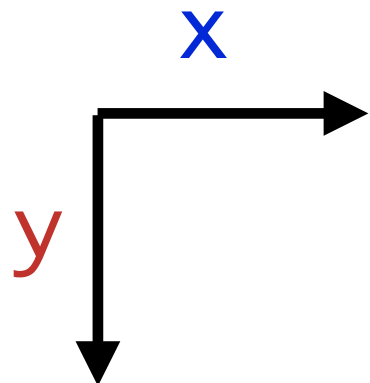
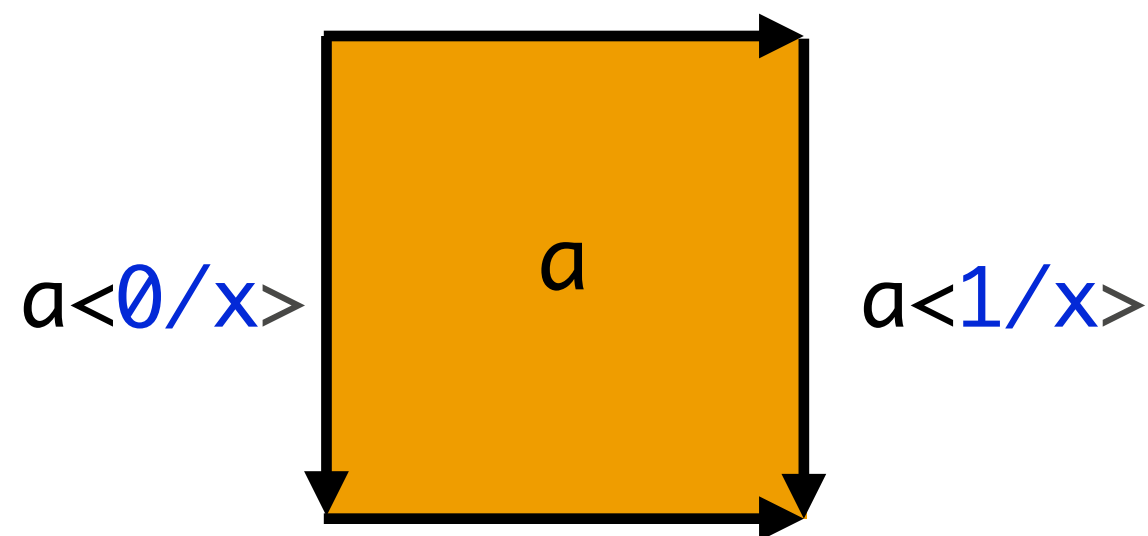


Faces

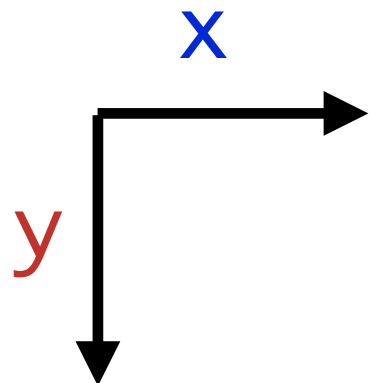
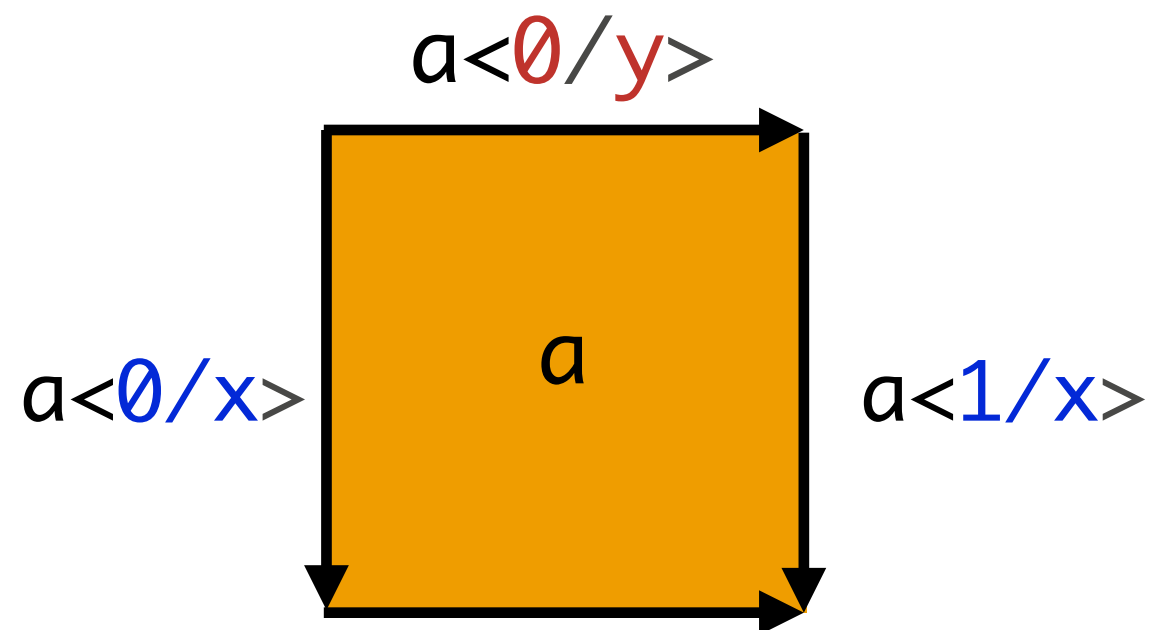
$$x:I, y:I \vdash a : A$$


Faces

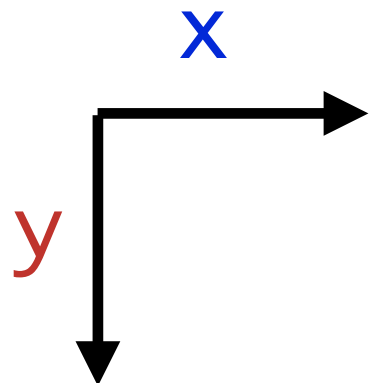
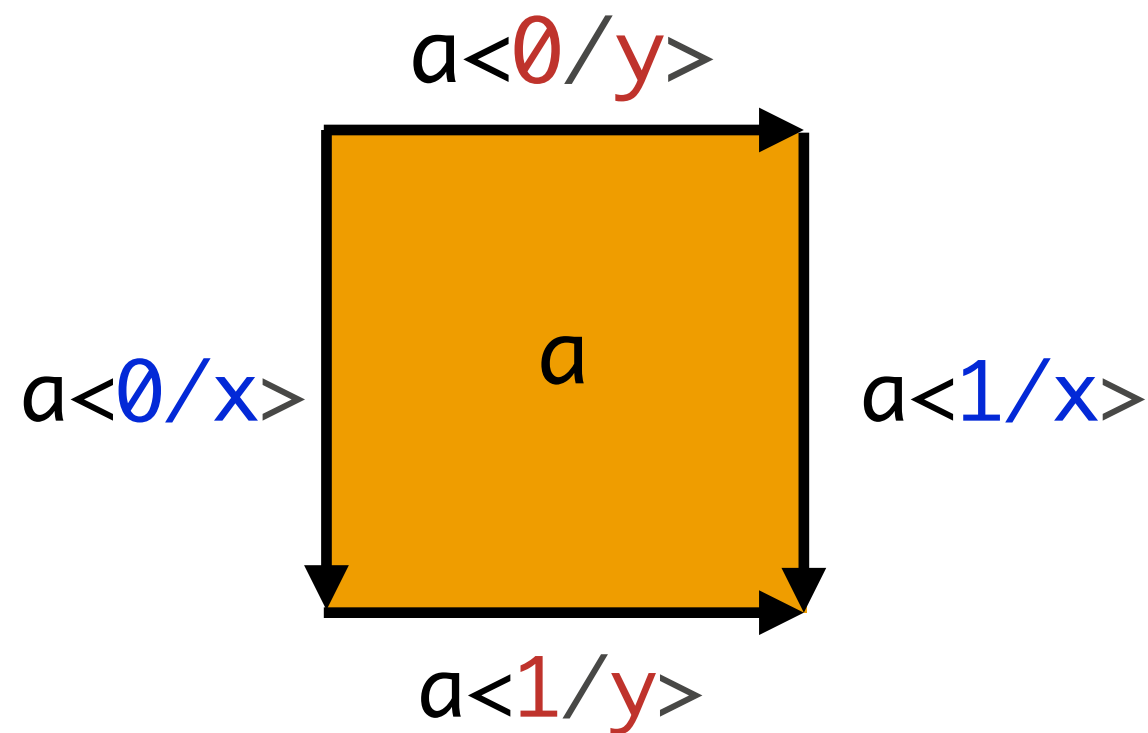
$x:I, y:I \vdash a : A$



Faces

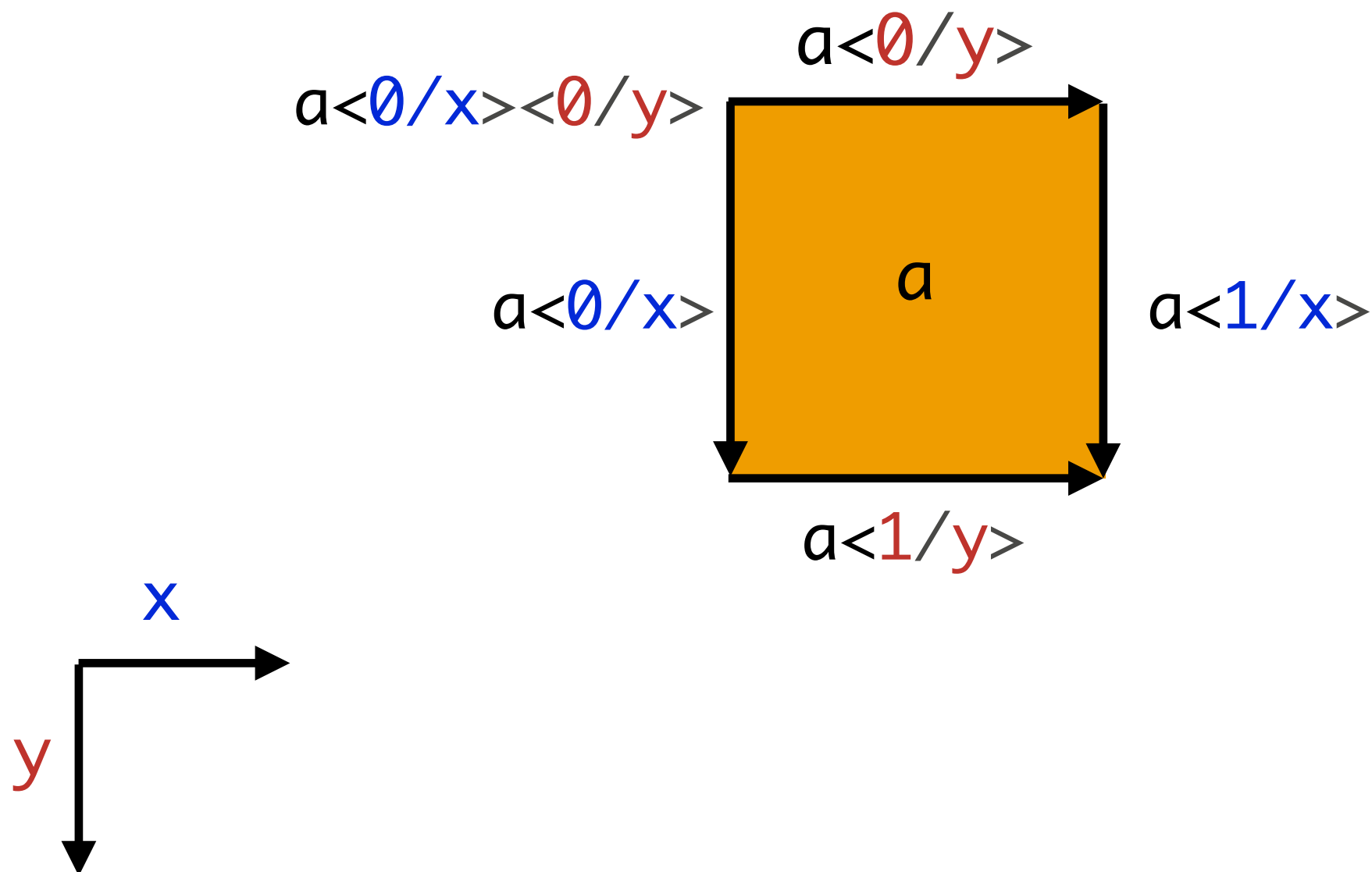
$$x:I, y:I \vdash a : A$$


Faces

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


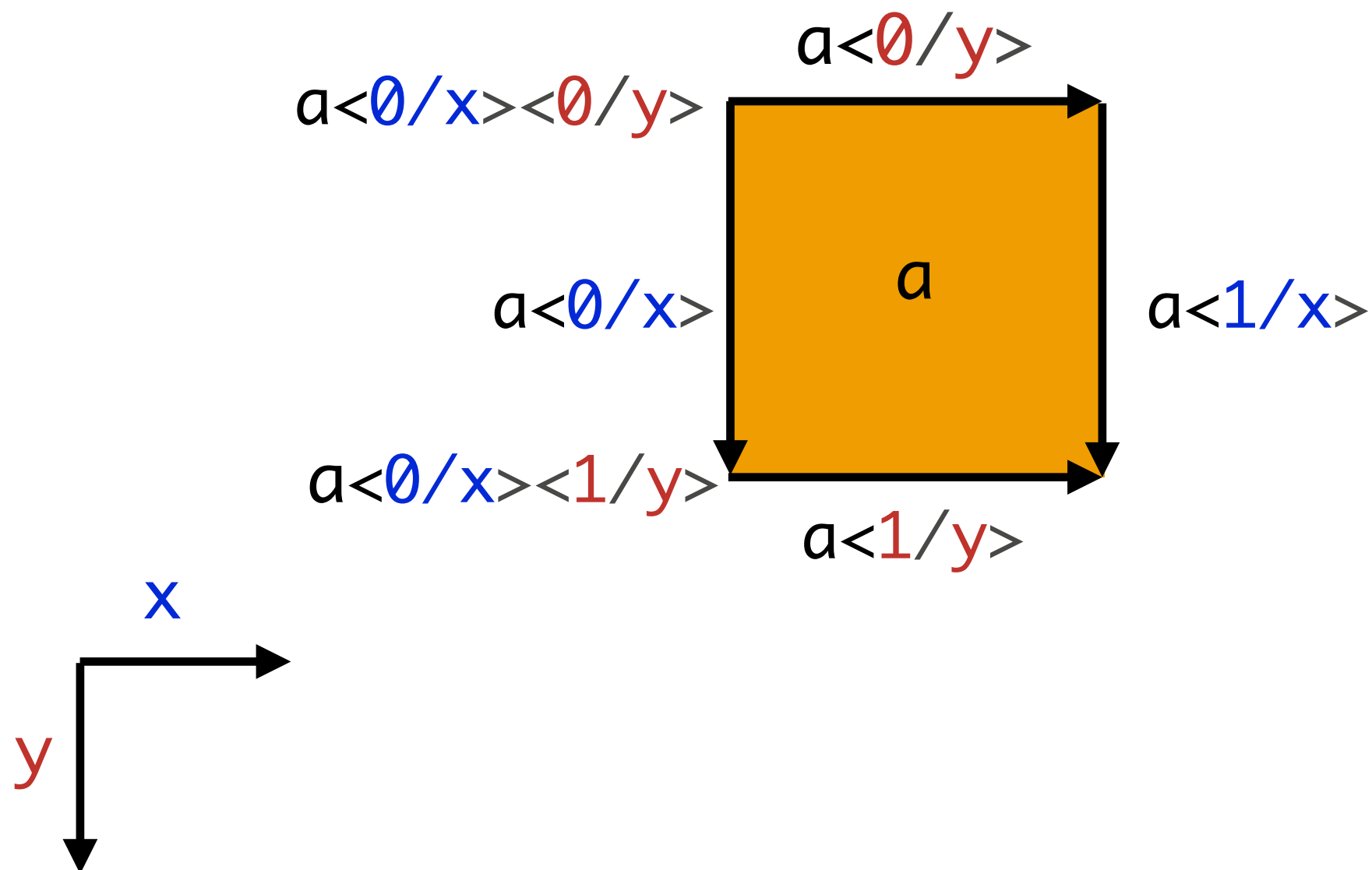
Faces

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

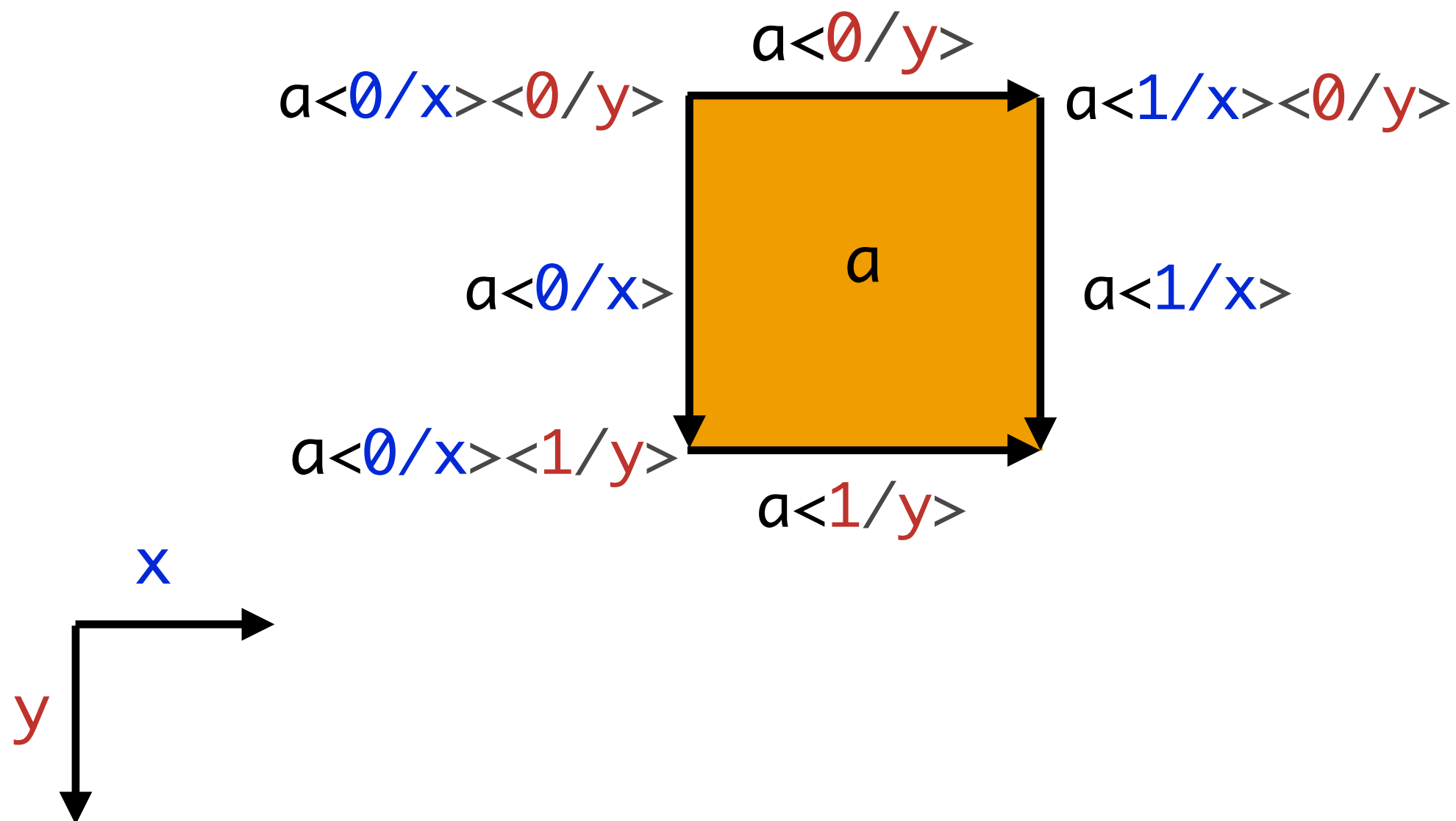


Faces

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

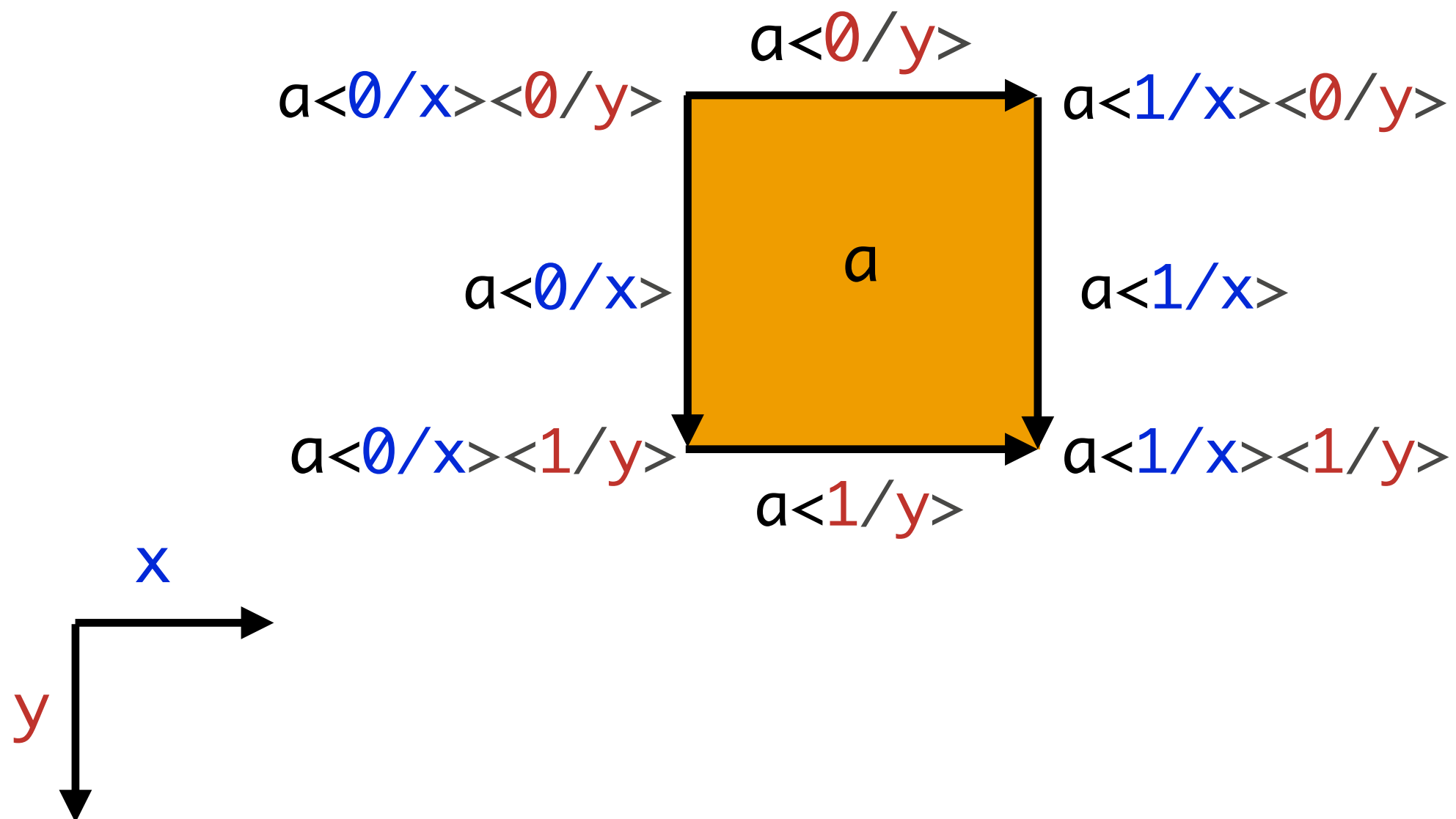


Faces

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


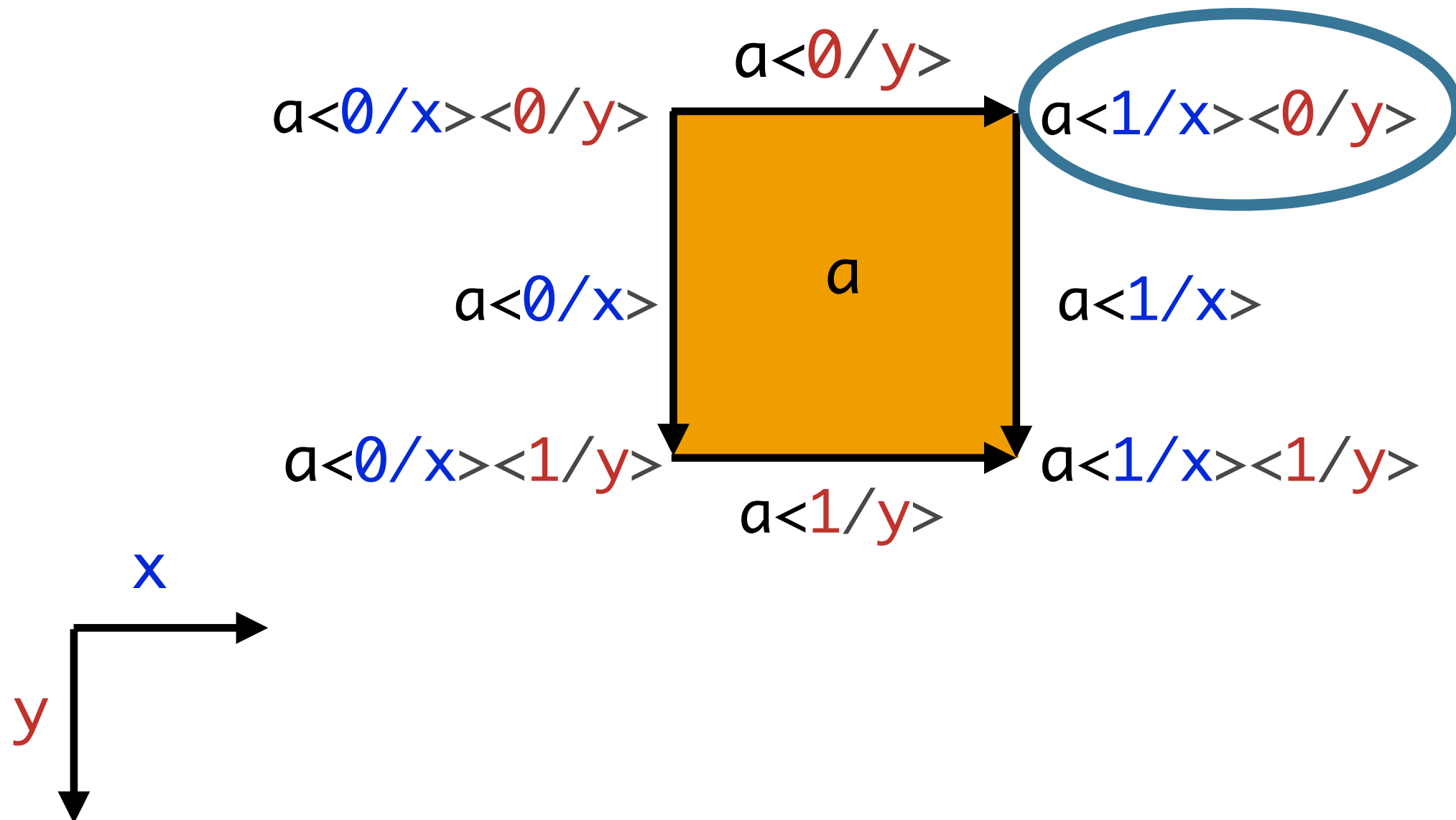
Faces

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



Faces

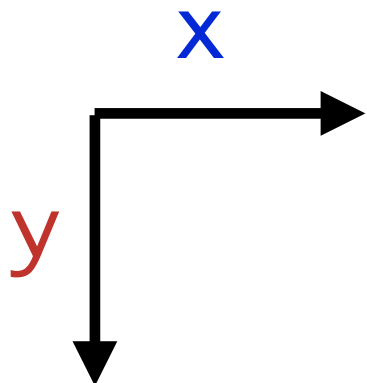
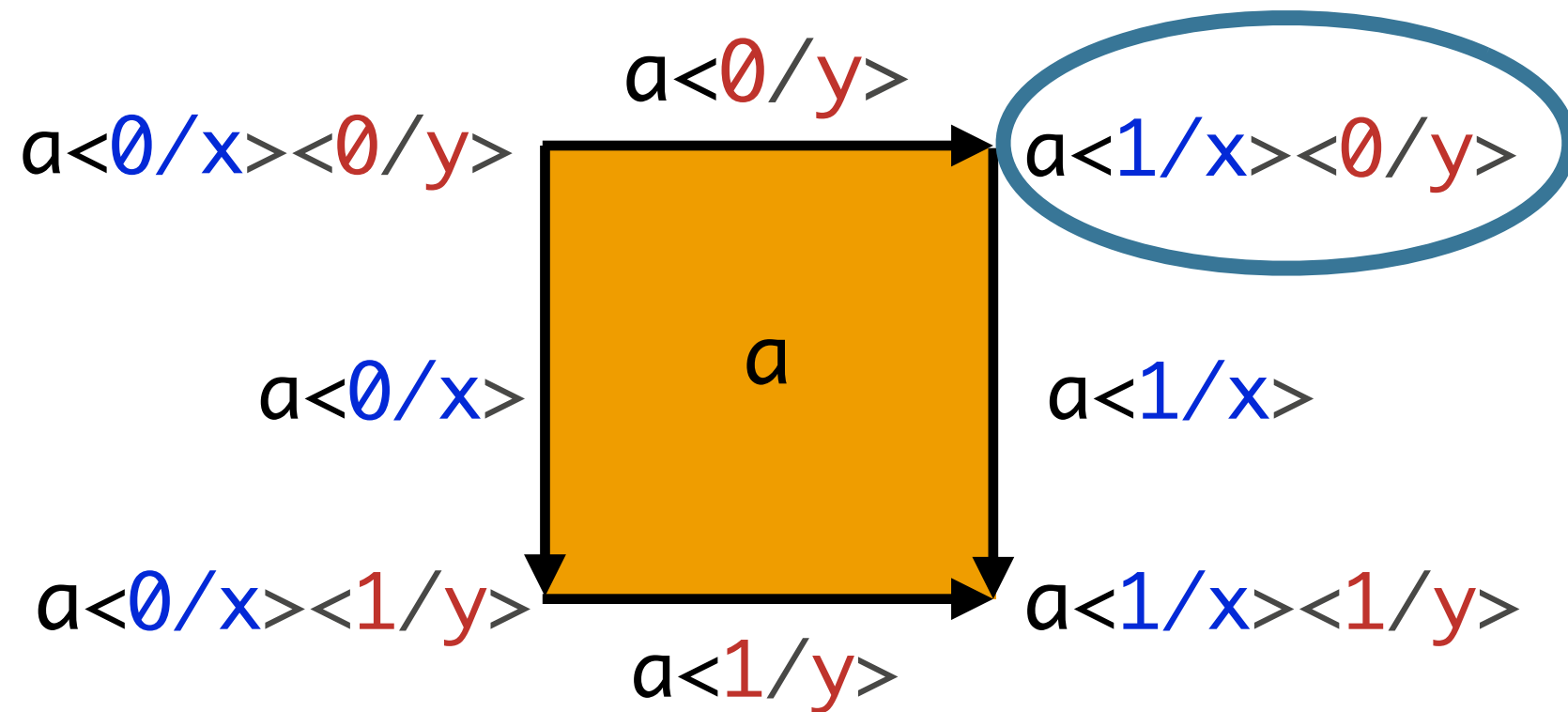
$x:I, y:I \vdash a : A$



Faces

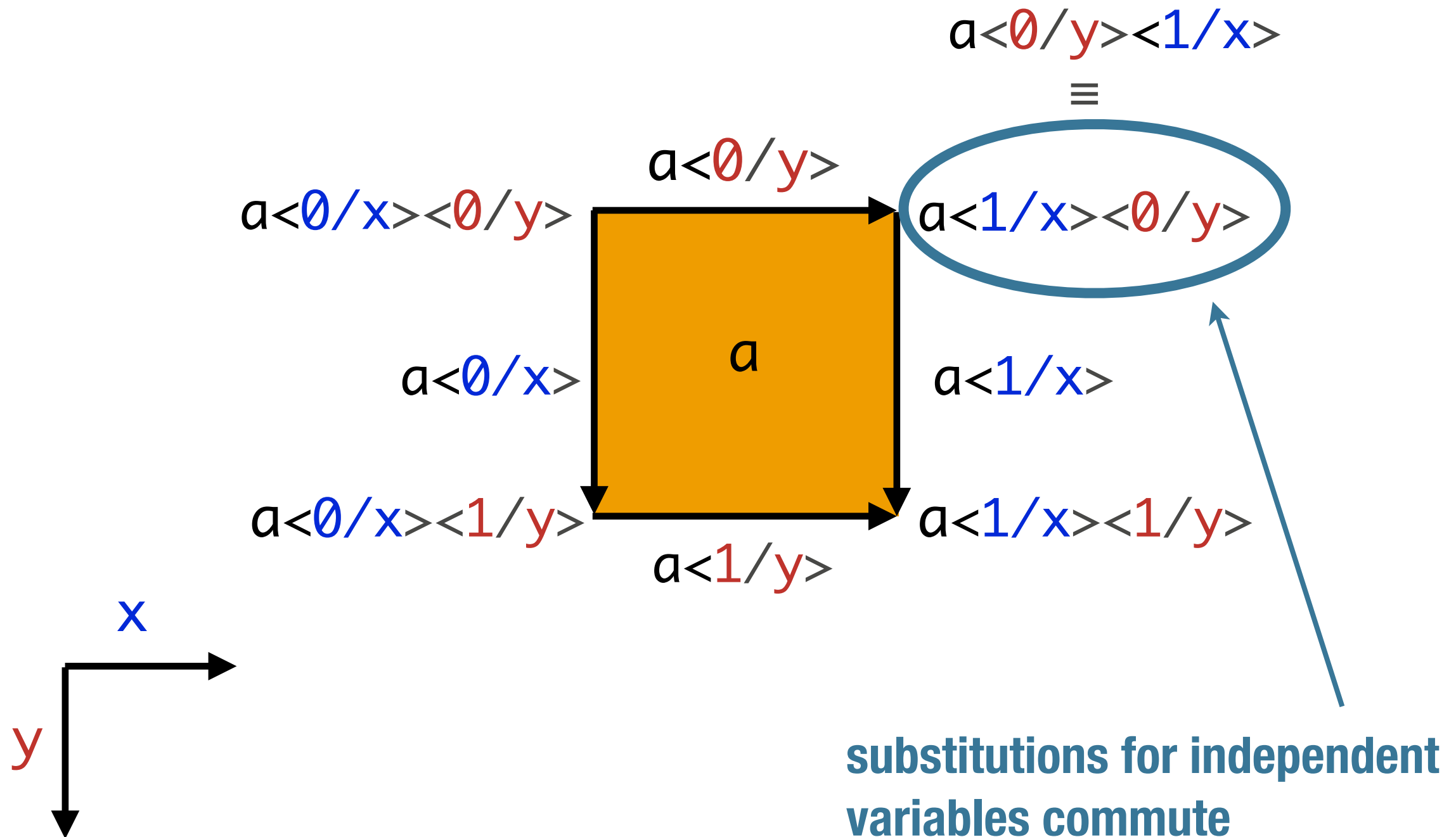
$x:I, y:I \vdash a : A$

$a\langle 0/y \rangle \langle 1/x \rangle$



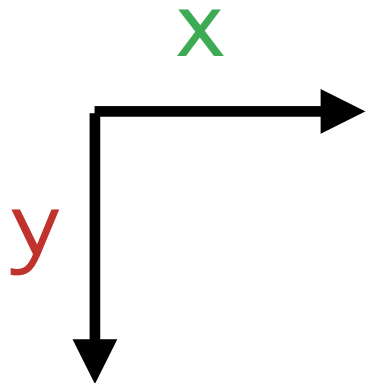
Faces

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

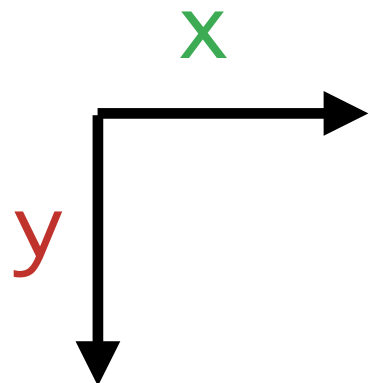
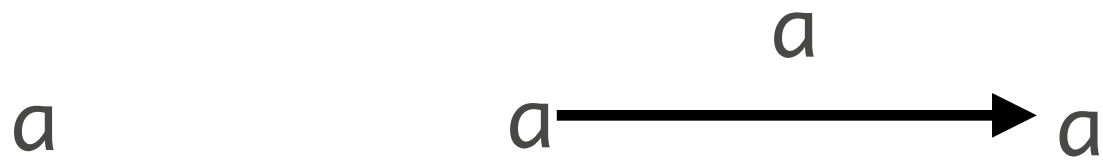


Degeneracies

a

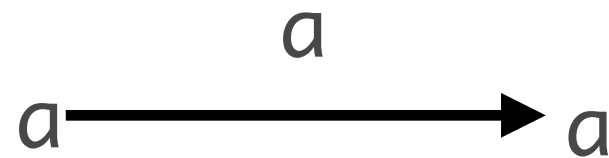


Degeneracies

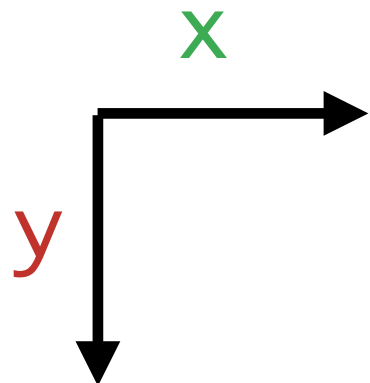


Degeneracies

a

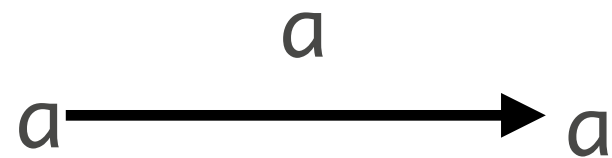


$$a \langle 0/x \rangle \equiv a$$



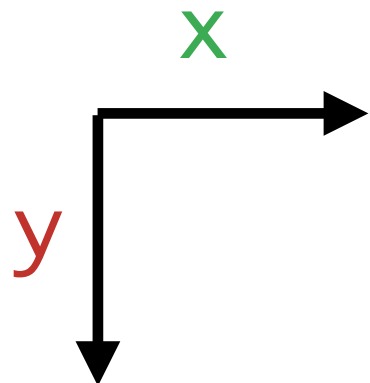
Degeneracies

a



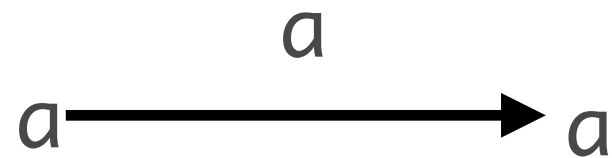
$$a \langle 0/x \rangle \equiv a$$

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



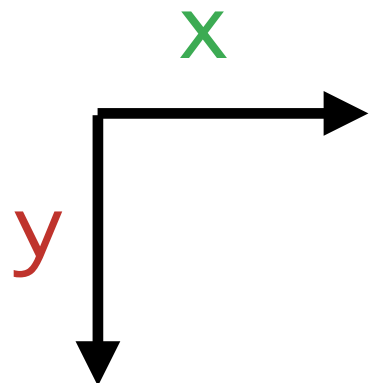
Degeneracies

a



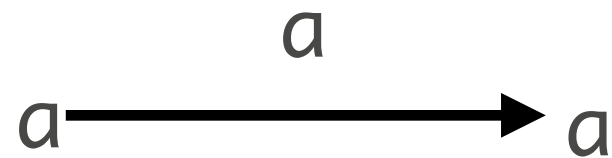
$$a \langle 0/x \rangle \equiv a$$

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



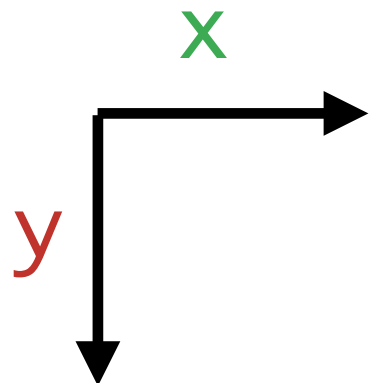
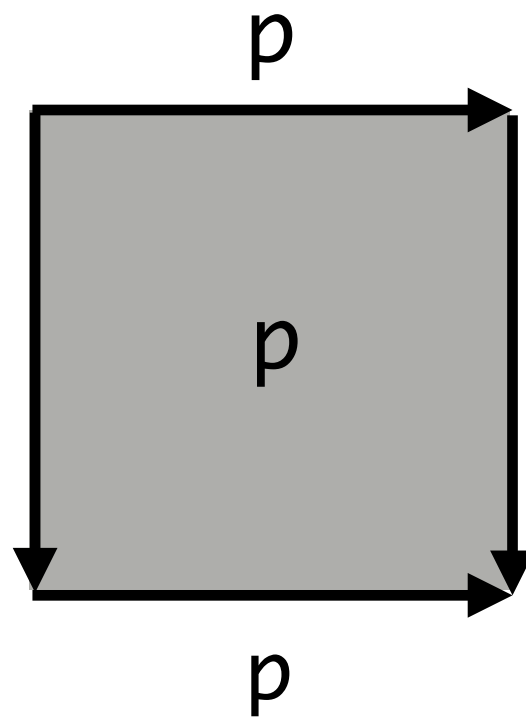
Degeneracies

a



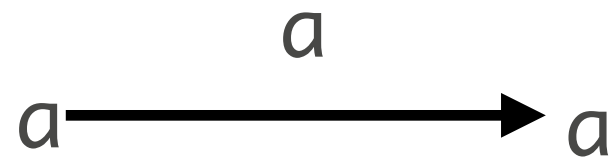
$$a \langle 0/x \rangle \equiv a$$

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



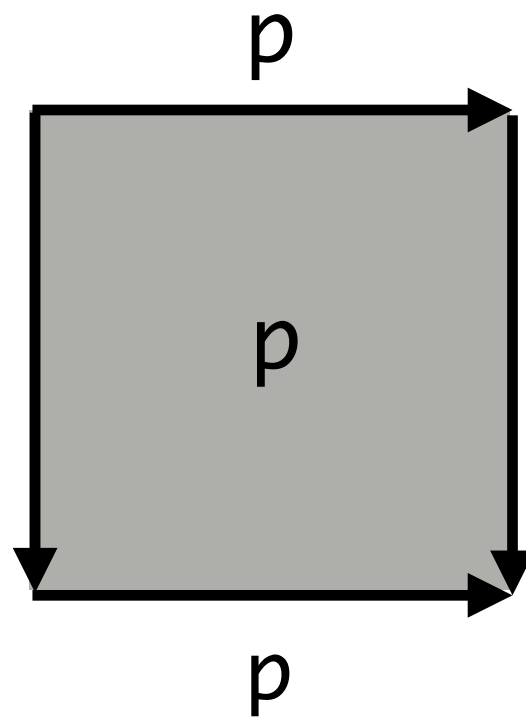
Degeneracies

a



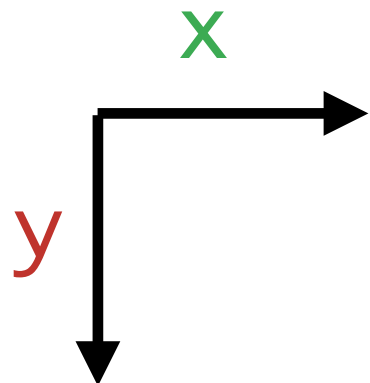
$$a \langle 0/x \rangle \equiv a$$

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



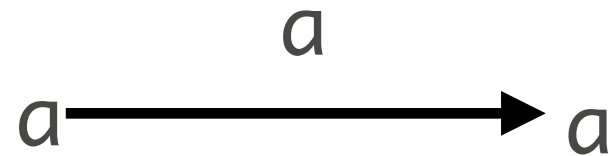
$$p \langle 0/y \rangle \equiv p$$

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



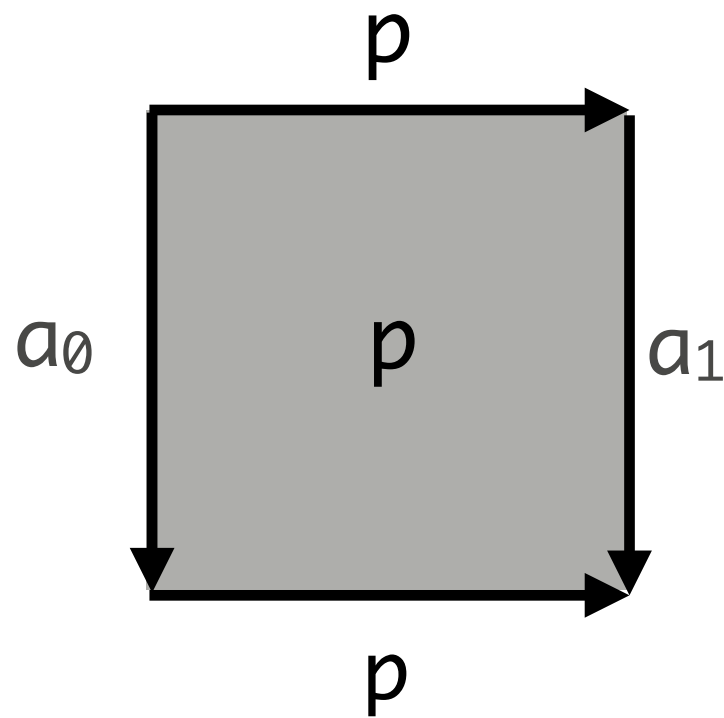
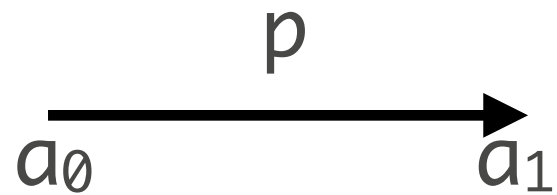
Degeneracies

a



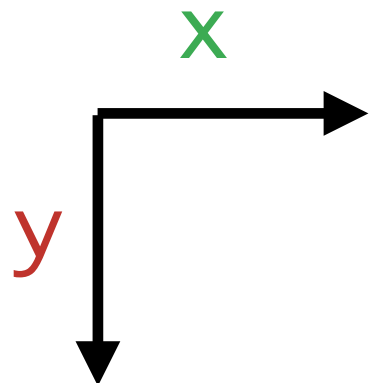
$$a \langle 0/x \rangle \equiv a$$

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



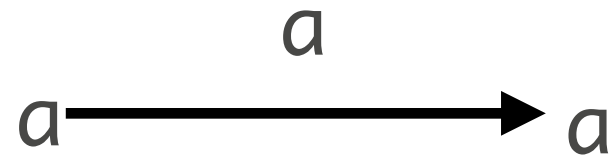
$$p \langle 0/y \rangle \equiv p$$

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



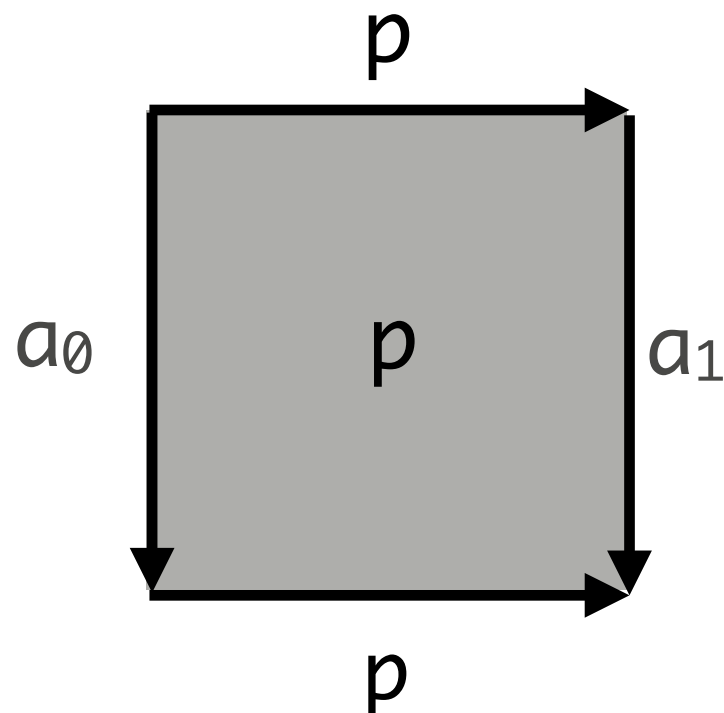
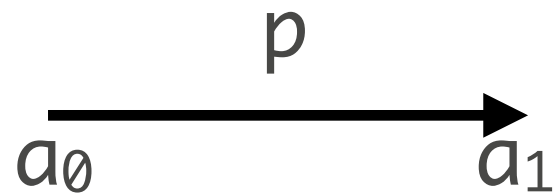
Degeneracies

a



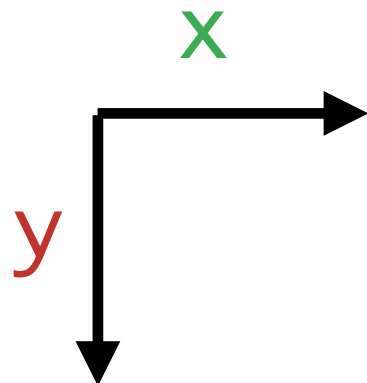
$$a \langle 0/x \rangle \equiv a$$

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



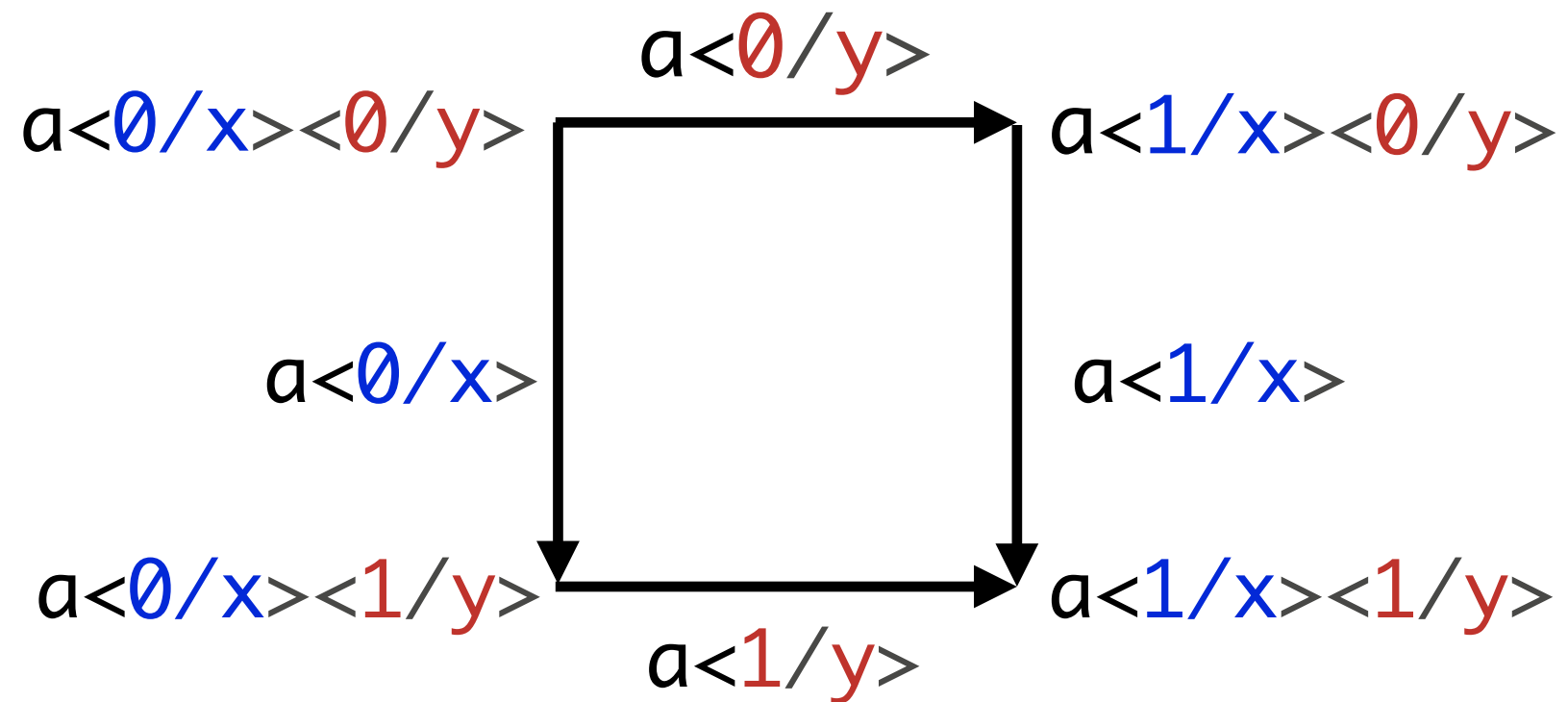
$$p \langle 0/y \rangle \equiv p$$

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

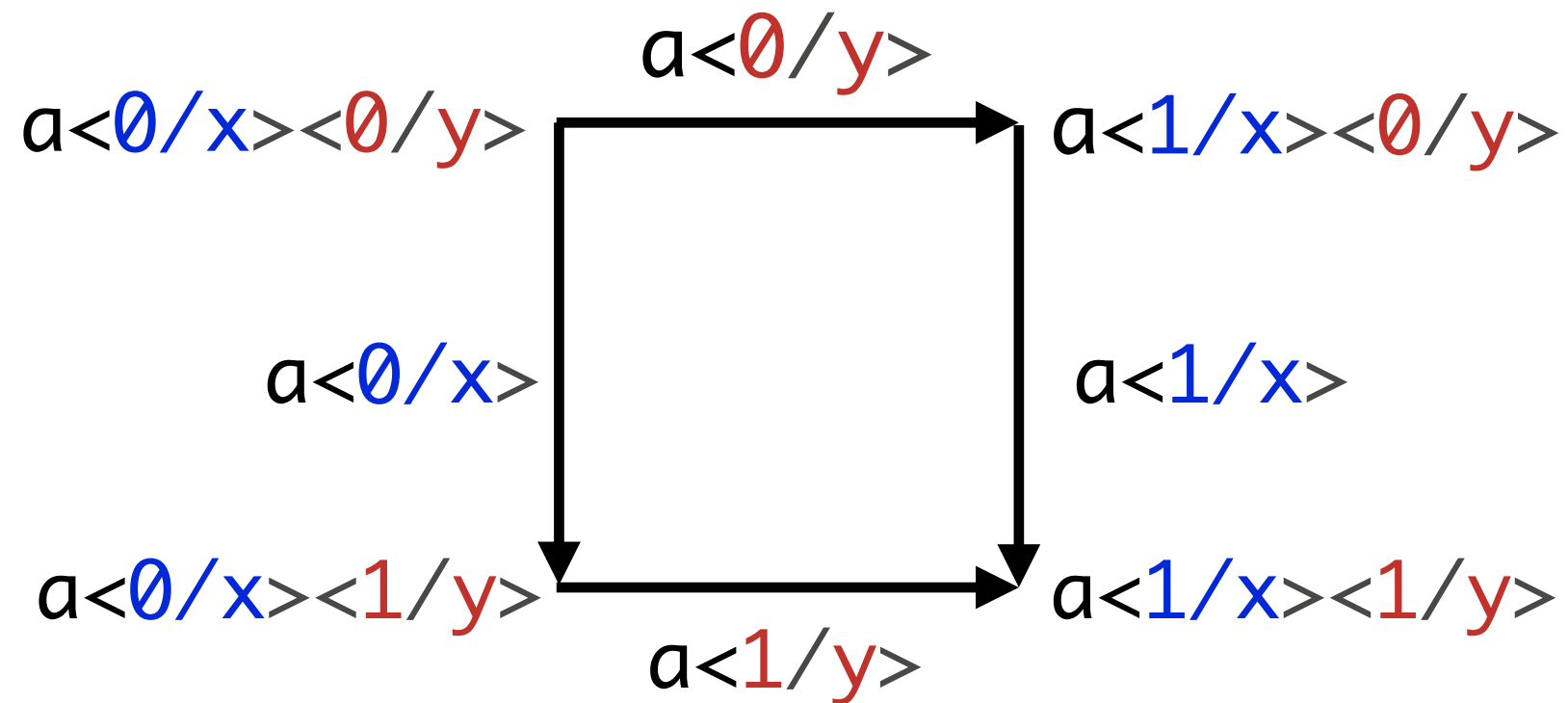


substitution after weakening is identity

Diagonals

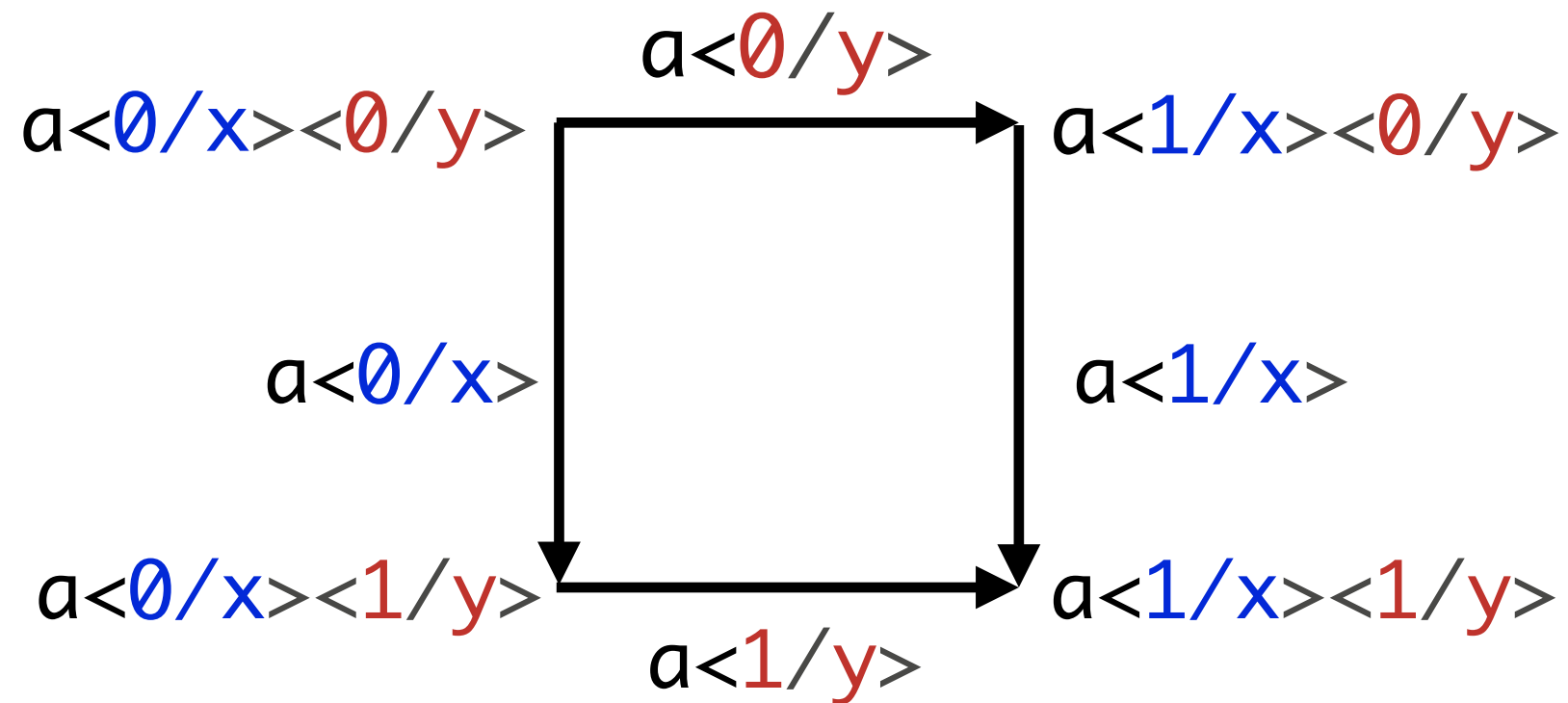


Diagonals



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

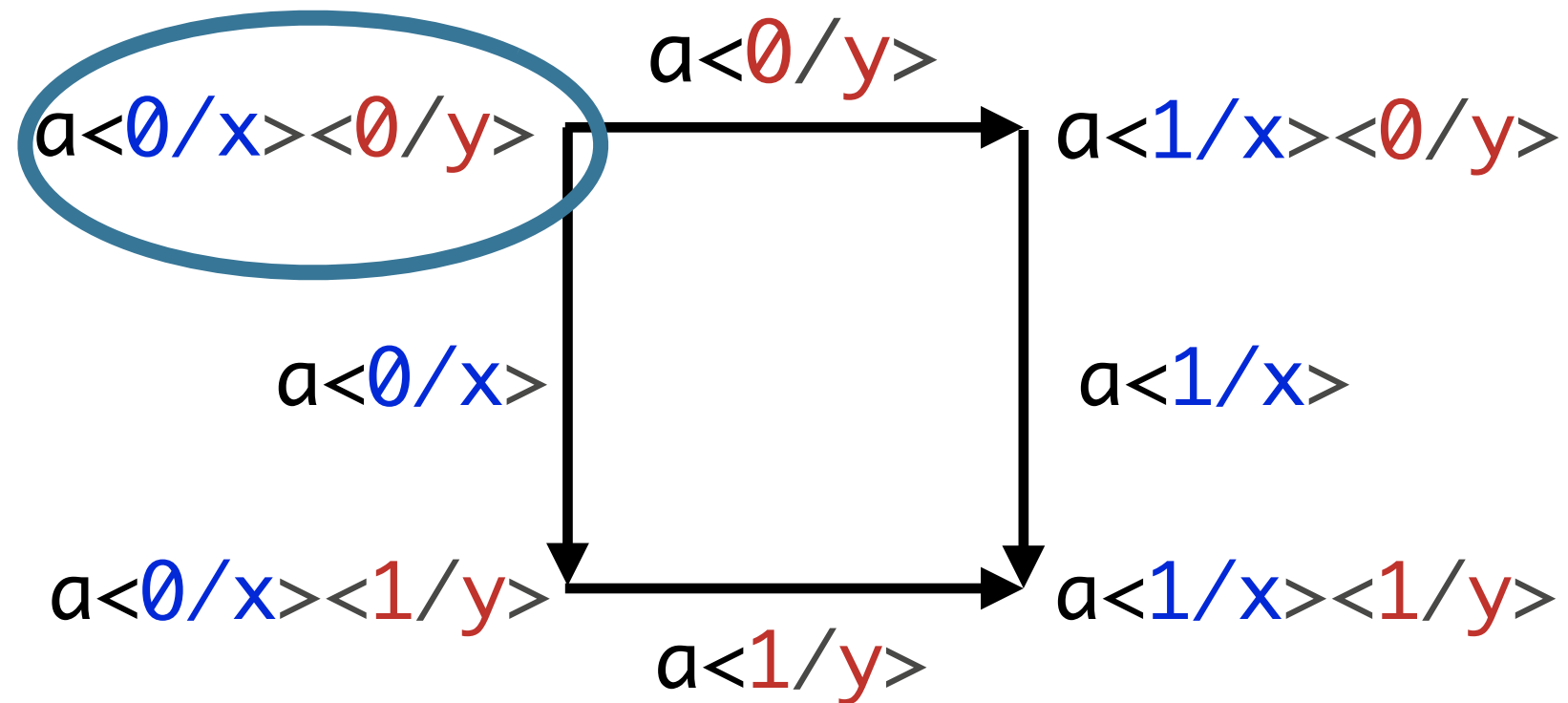
Diagonals



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

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

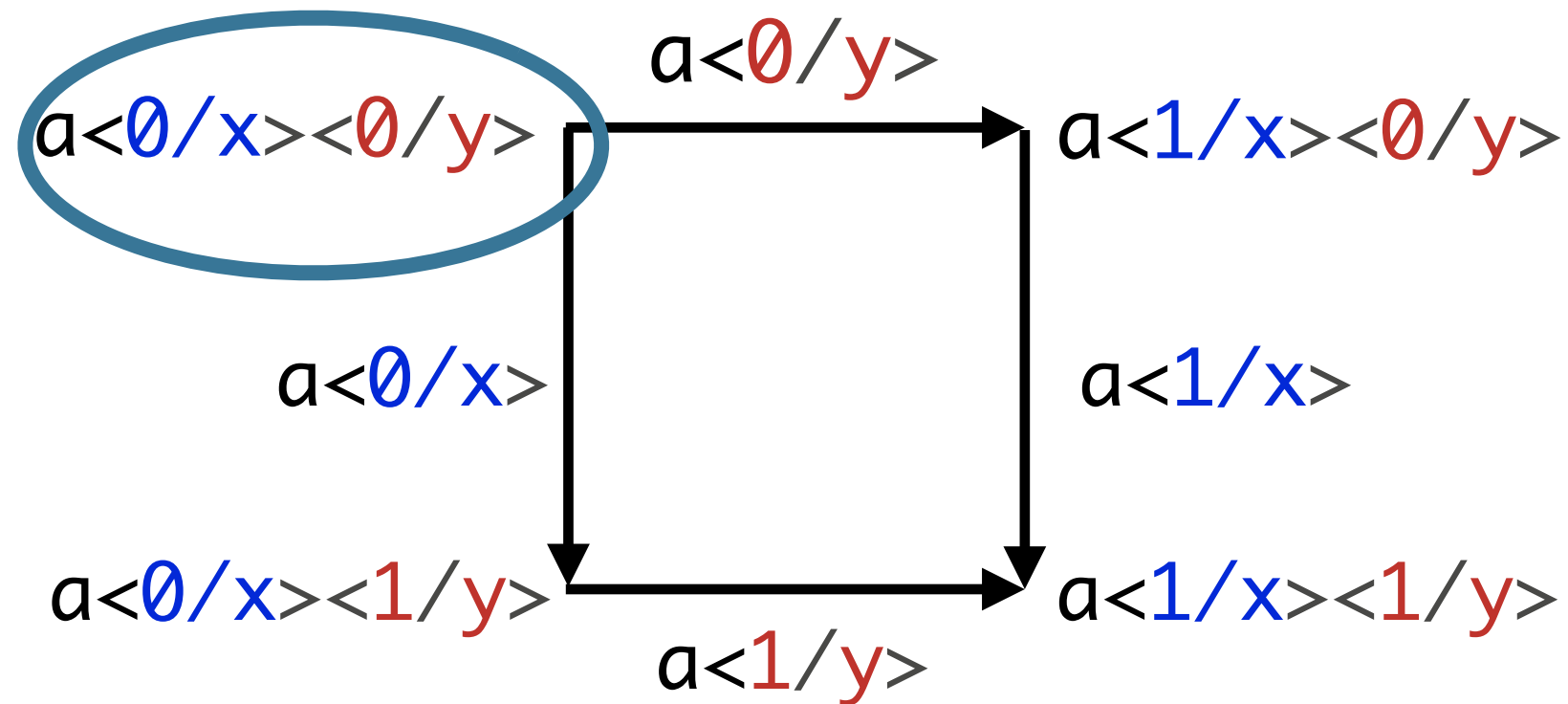
Diagonals



$a\langle x/y \rangle$ 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

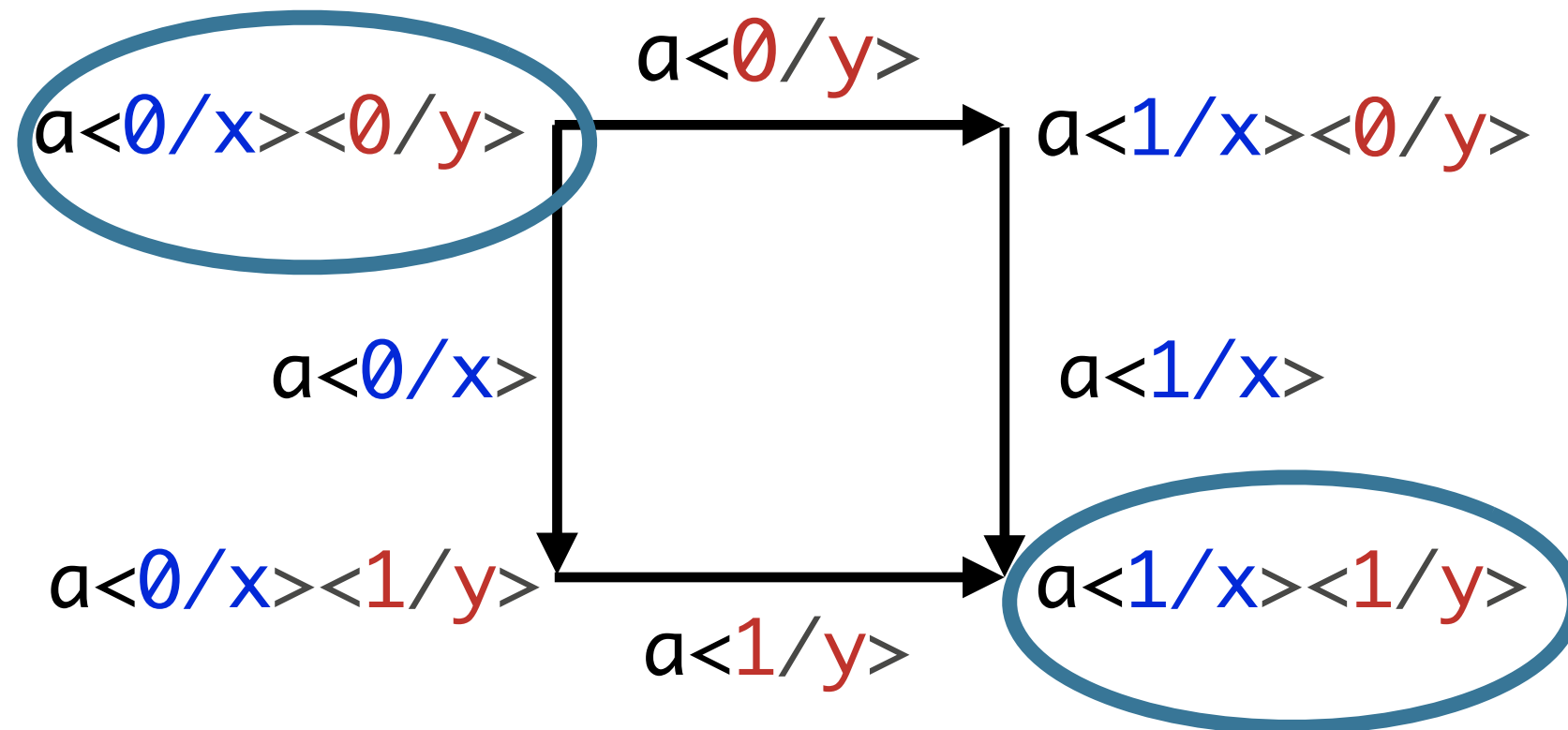


$a\langle x/y \rangle$ is a line ($\{x\}$ -cube)

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

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

Diagonals

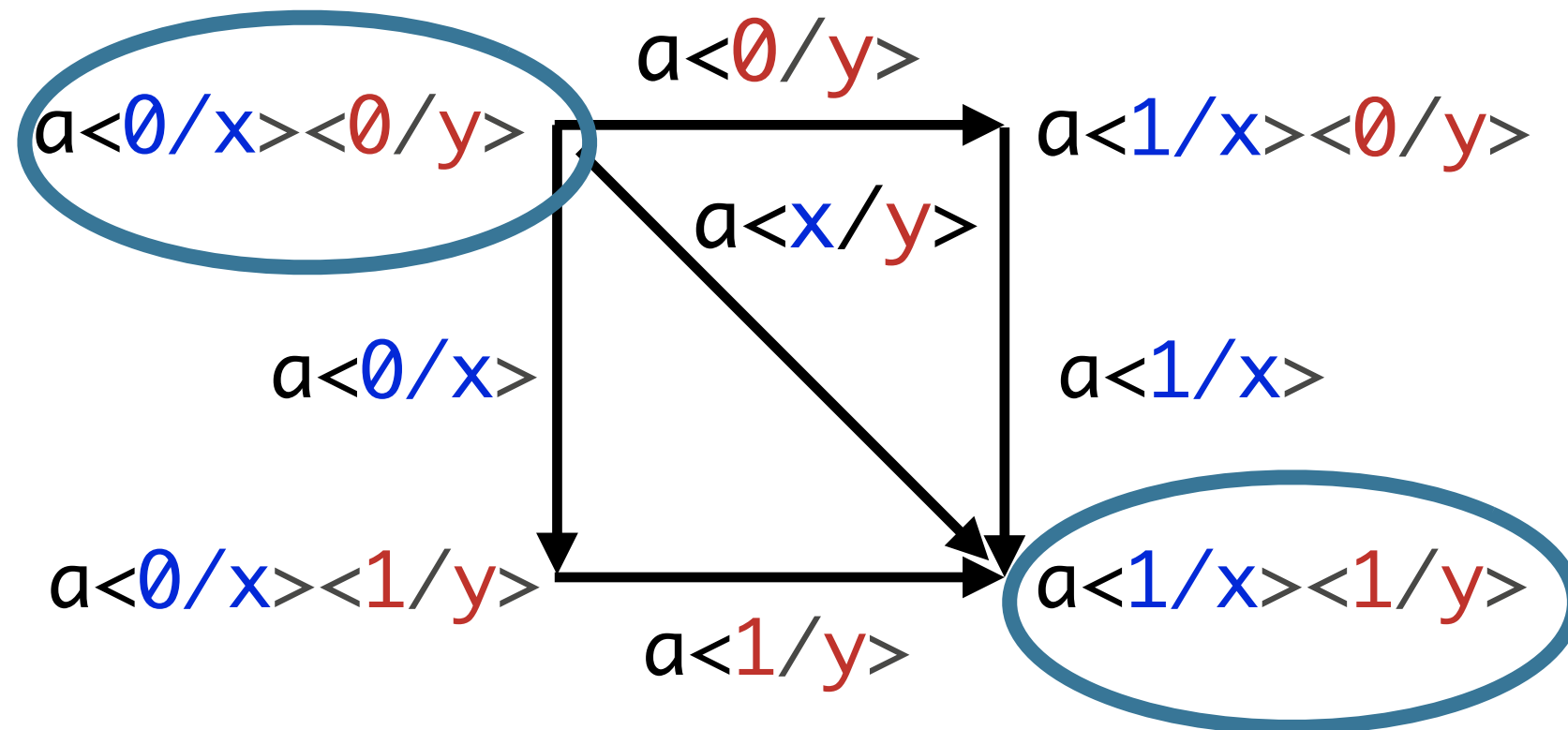


$a\langle x/y \rangle$ is a line ($\{x\}$ -cube)

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

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

Diagonals

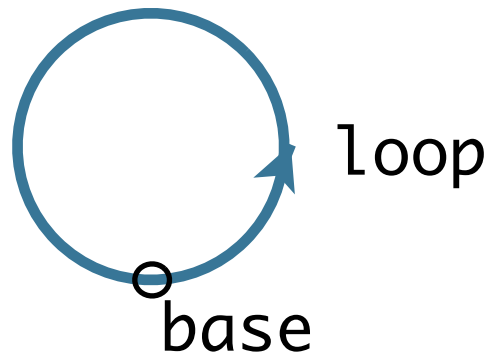


$a\langle x/y \rangle$ is a line ($\{x\}$ -cube)

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

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

Circle



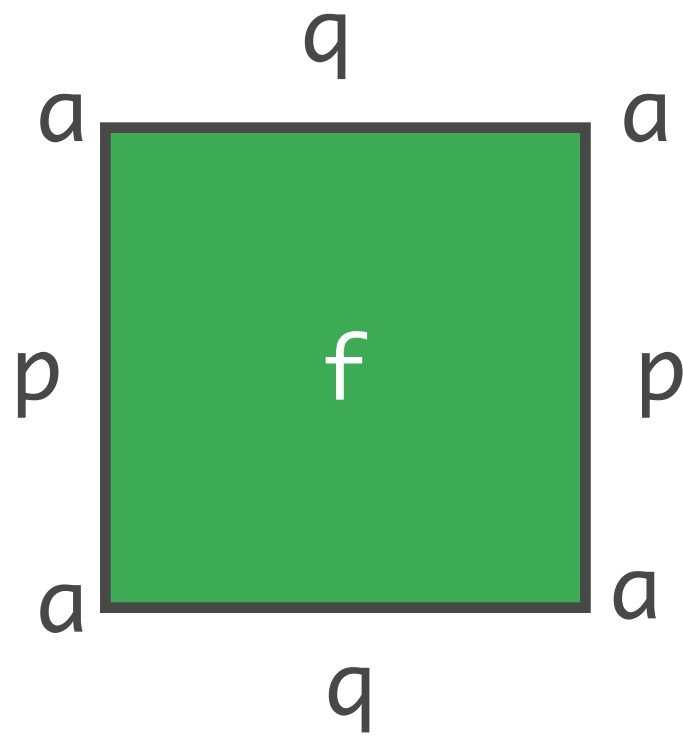
$\text{base} : S^1$

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

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

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

Torus



$$a : T$$

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

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

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

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

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

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

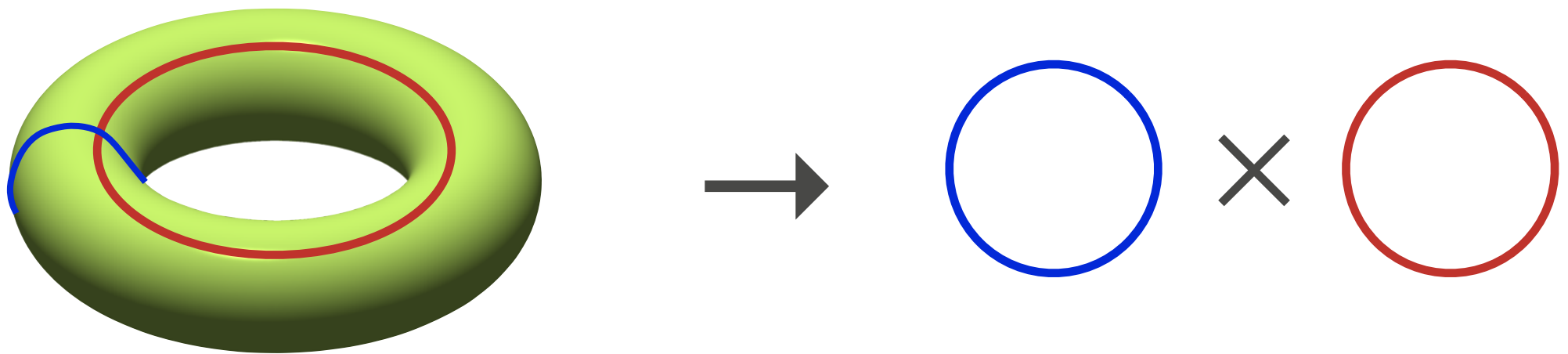
$$t2c : T \rightarrow S^1 \times S^1$$

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

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

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

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



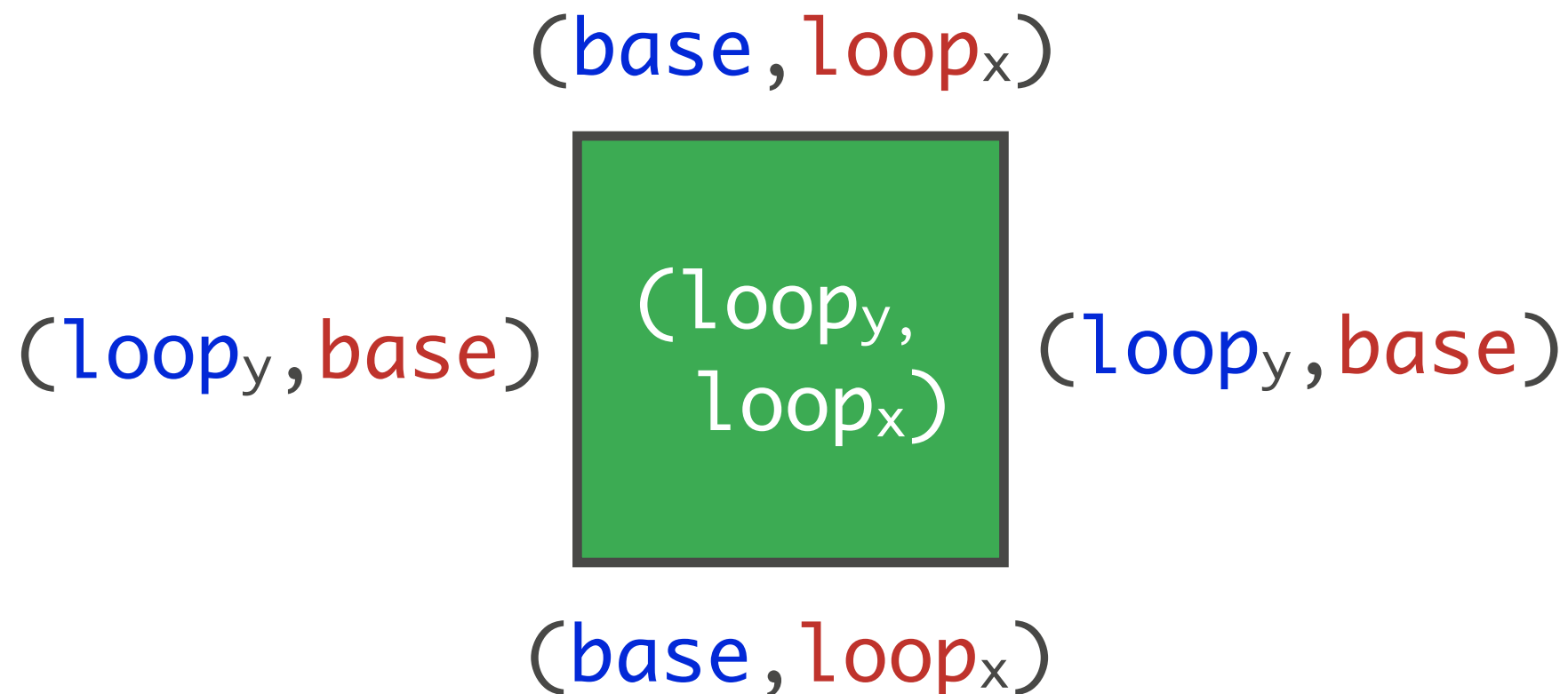
$$t2c : T \rightarrow S^1 \times S^1$$

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

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

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

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



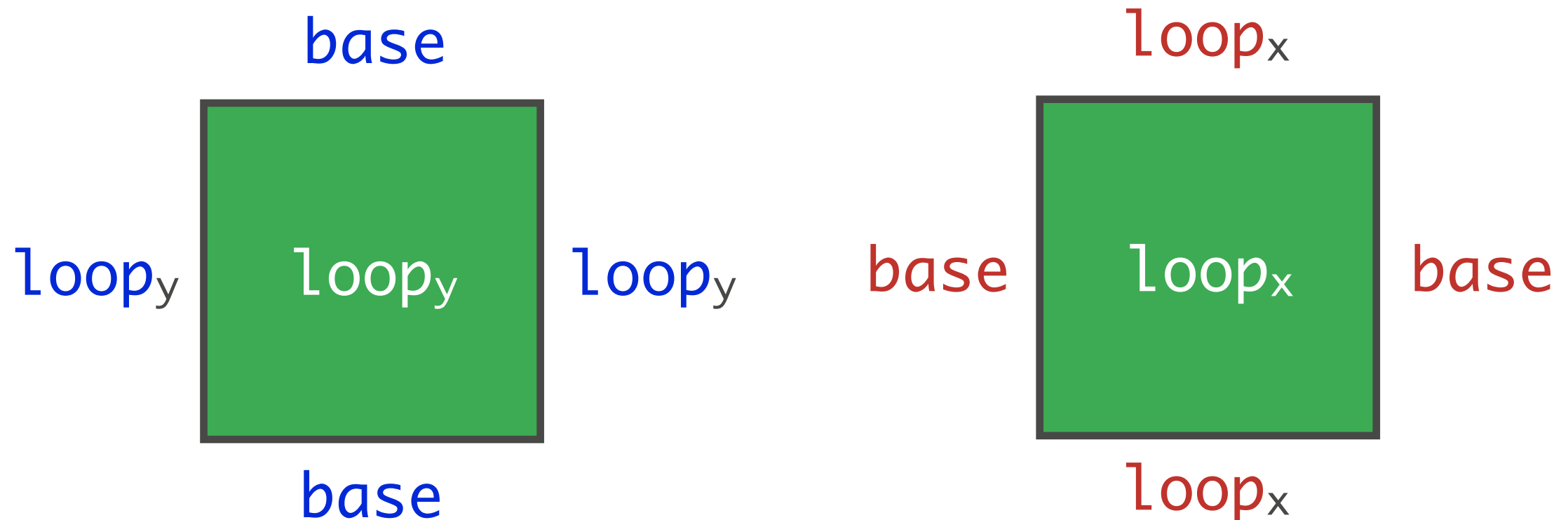
$$t2c : T \rightarrow S^1 \times S^1$$

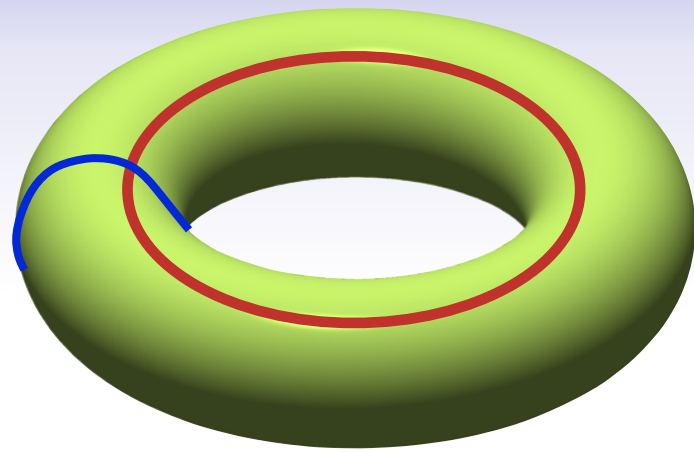
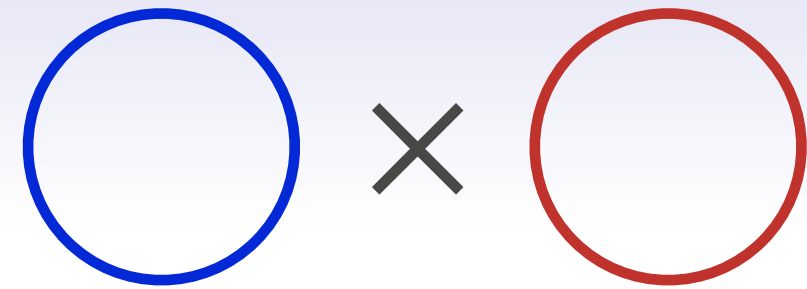
$$t2c \ a = (\text{base}, \text{base})$$

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

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

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




 $\cong$


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

$$t2c \ a = (base, base)$$

$$t2c \ p_y = (loop_y, base)$$

$$t2c \ q_x = (base, loop_x)$$

$$t2c \ f_{x,y} = (loop_y, loop_x)$$

$$c2t : S^1 \rightarrow (S^1 \rightarrow T)$$

$$c2t \ \text{blue } base = \text{red } base \mapsto a$$

$$\text{red } loop_x \mapsto q_x$$

$$c2t \ \text{blue } loop_y = \text{red } base \mapsto p_y$$

$$\text{red } loop_x \mapsto f_{x,y}$$

Composites: induct; reflexivity in each case

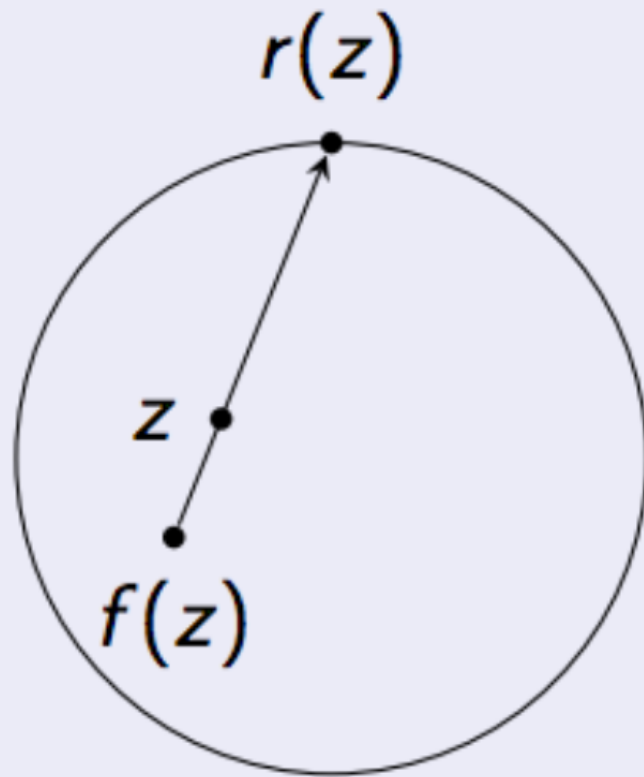
Modal homotopy type theories

Brouwer fixed point theorem

Theorem

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

Proof.

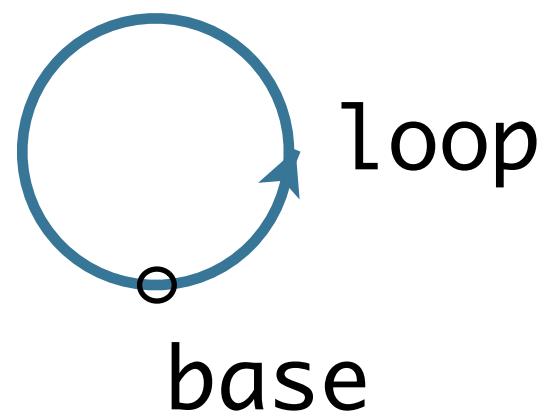


Suppose $f : \mathbb{D}^2 \rightarrow \mathbb{D}^2$ is continuous with no fixed point. For any $z \in \mathbb{D}^2$, draw the ray from $f(z)$ through z to hit $\partial\mathbb{D}^2 = \mathbb{S}^1$ at $r(z)$. Then r is continuous, and retracts $\mathbb{D}^2$ onto $\mathbb{S}^1$. Hence $\pi_1(\mathbb{S}^1)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



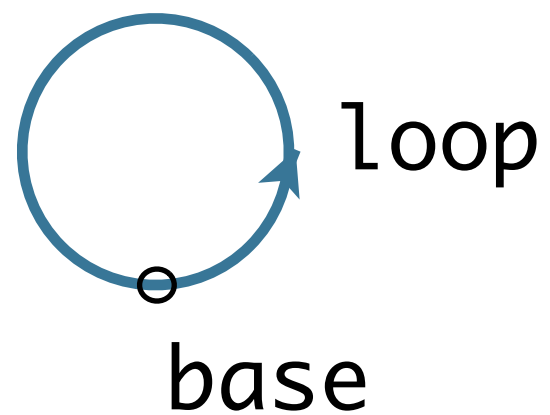
Brouwer fixed point theorem

homotopical circle



Brouwer fixed point theorem

homotopical circle

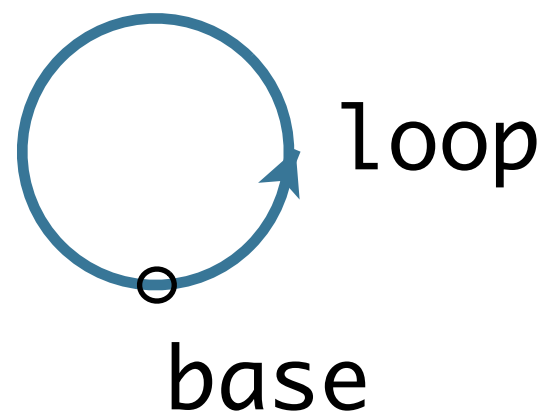


homotopical disc



Brouwer fixed point theorem

homotopical circle



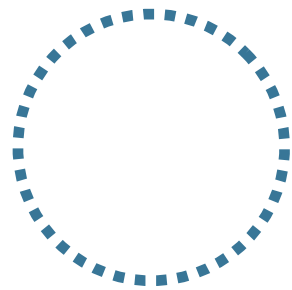
homotopical disc



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

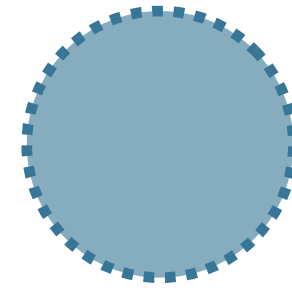
Brouwer fixed point theorem

topological circle



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

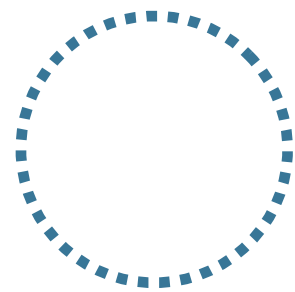
topological disc



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

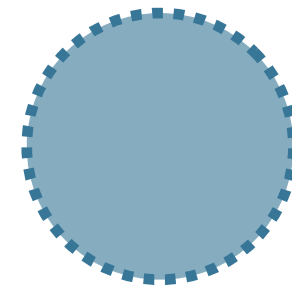
Brouwer fixed point theorem

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

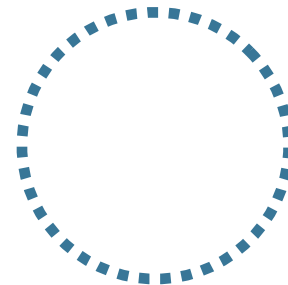
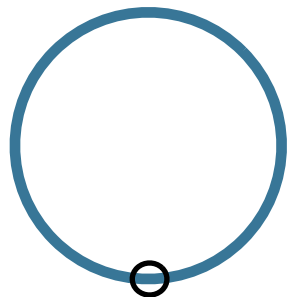
type theory



homotopy types
(e.g. simplicial sets)



topological spaces



External View

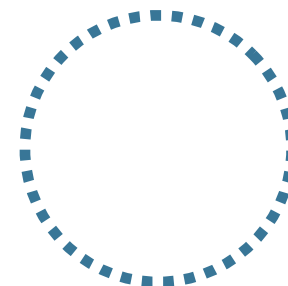
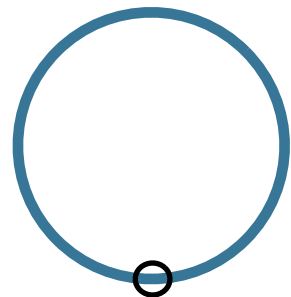
type theory



homotopy types
(e.g. simplicial sets)



topological spaces



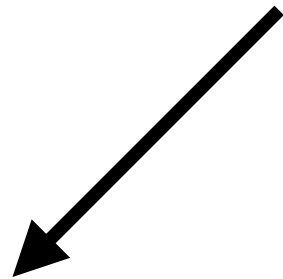
but can't see this *inside* the DSL

Types as spaces $\times 2$

type theory

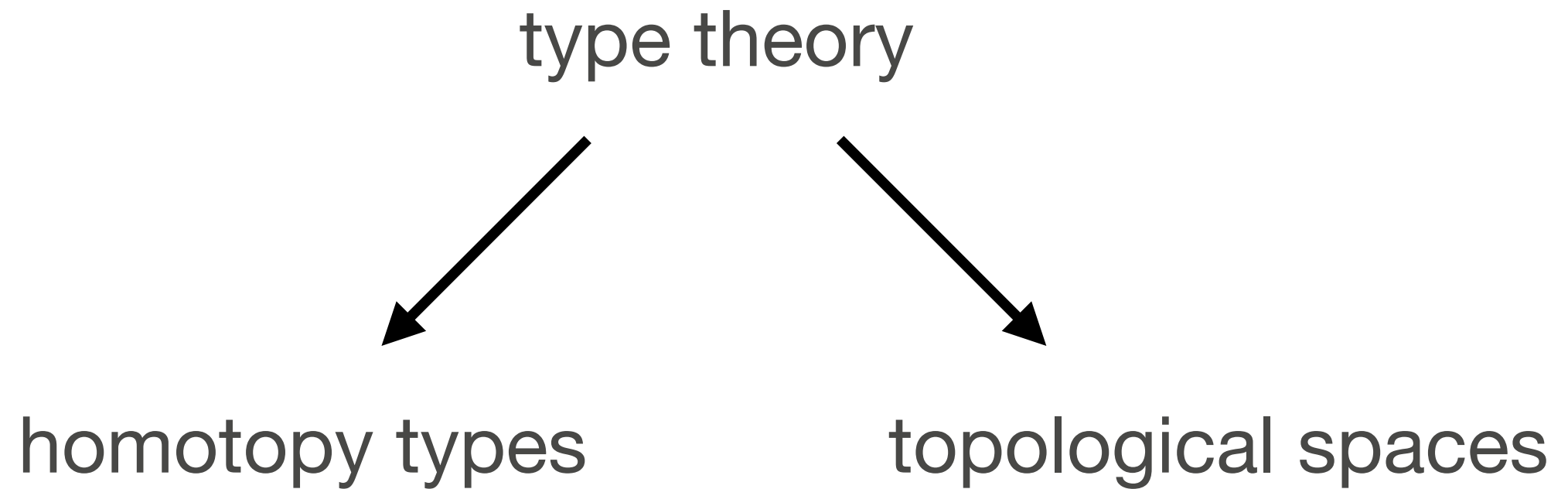
Types as spaces $\times 2$

type theory

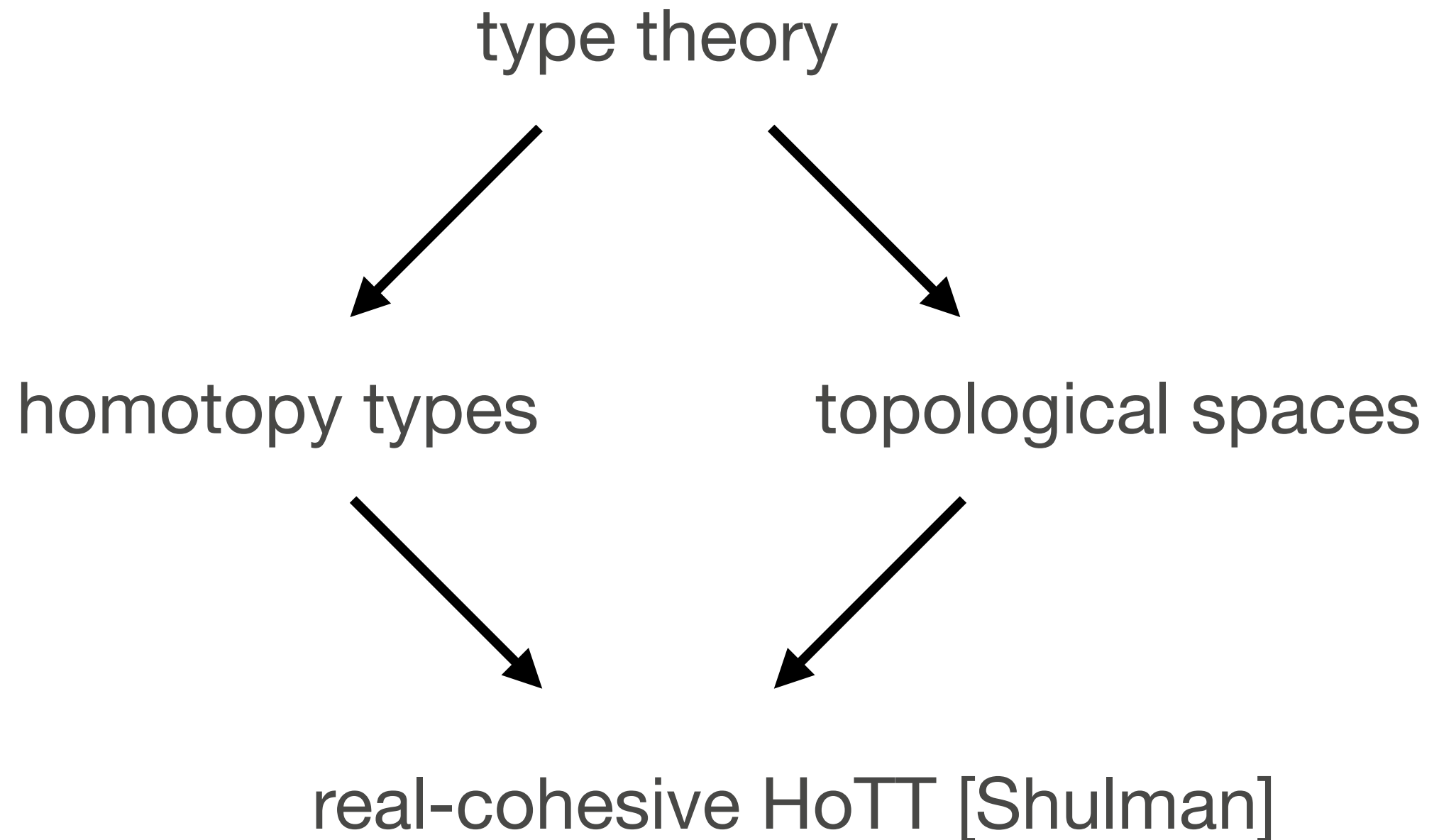


homotopy types

Types as spaces $\times 2$

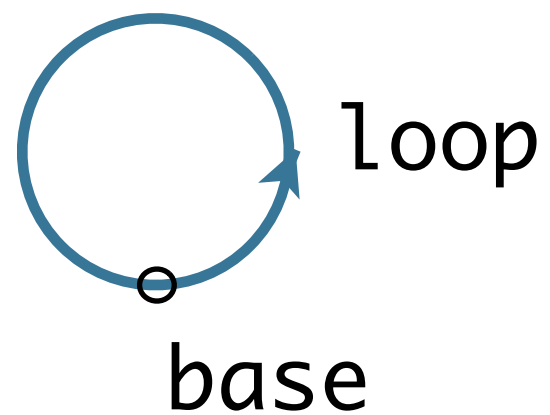


Types as spaces $\times 2$



Circles

homotopical circle

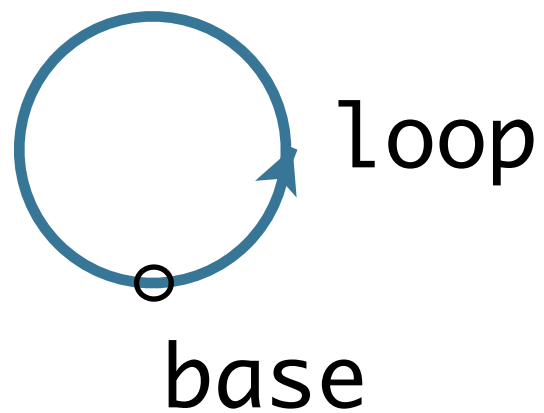


$$(\text{base}=\text{base}) \approx \mathbb{Z}$$

topologically discrete

Circles

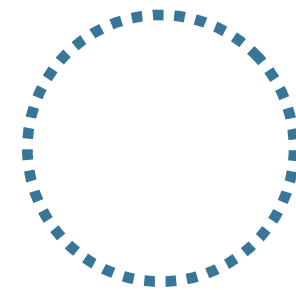
homotopical circle



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

topologically discrete

topological circle

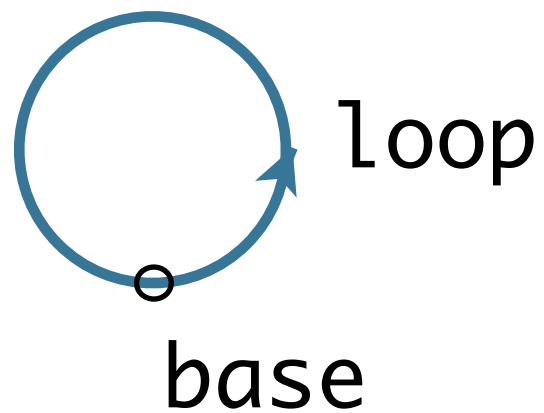


$$\{(x, y) : \mathbb{R}^2 \mid x^2 + y^2 = 1\}$$

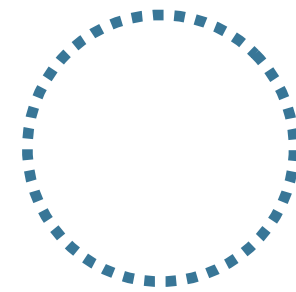
$\emptyset$ -type

usual topology

Modalities



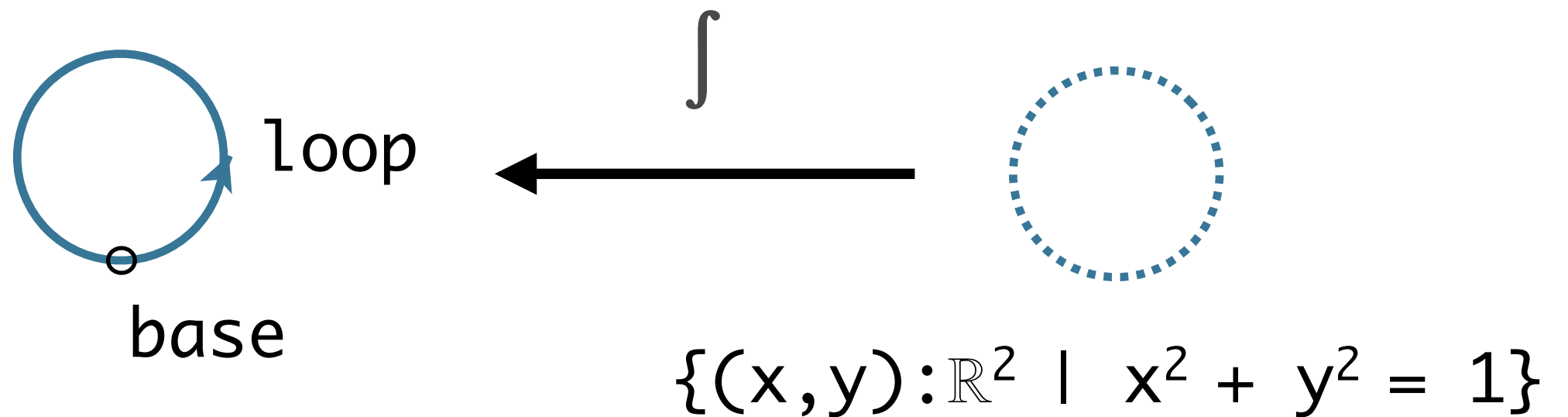
$(\text{base}=\text{base}) \simeq \mathbb{Z}$
topologically discrete



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

$\emptyset$ -type
usual topology

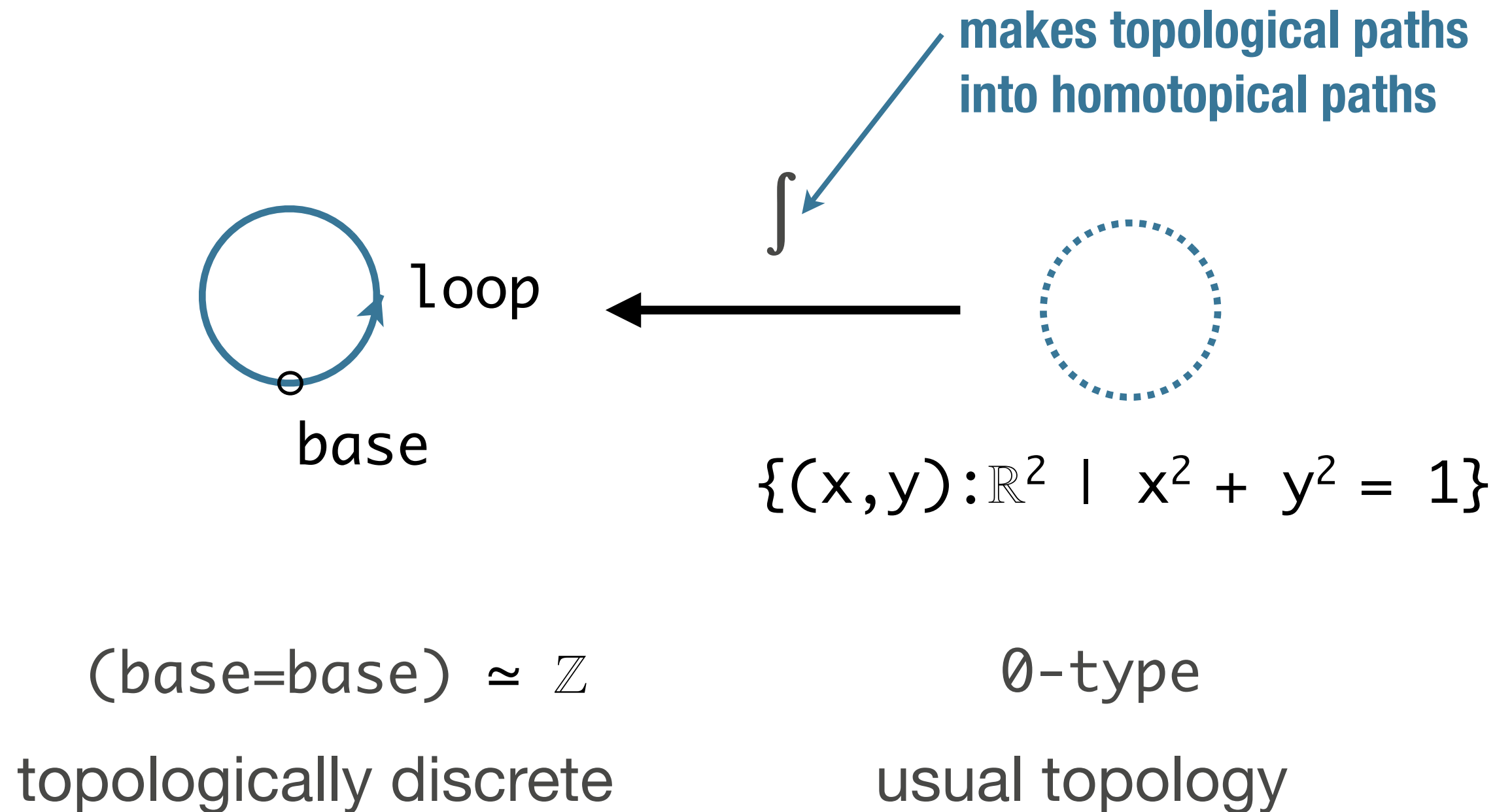
Modalities



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

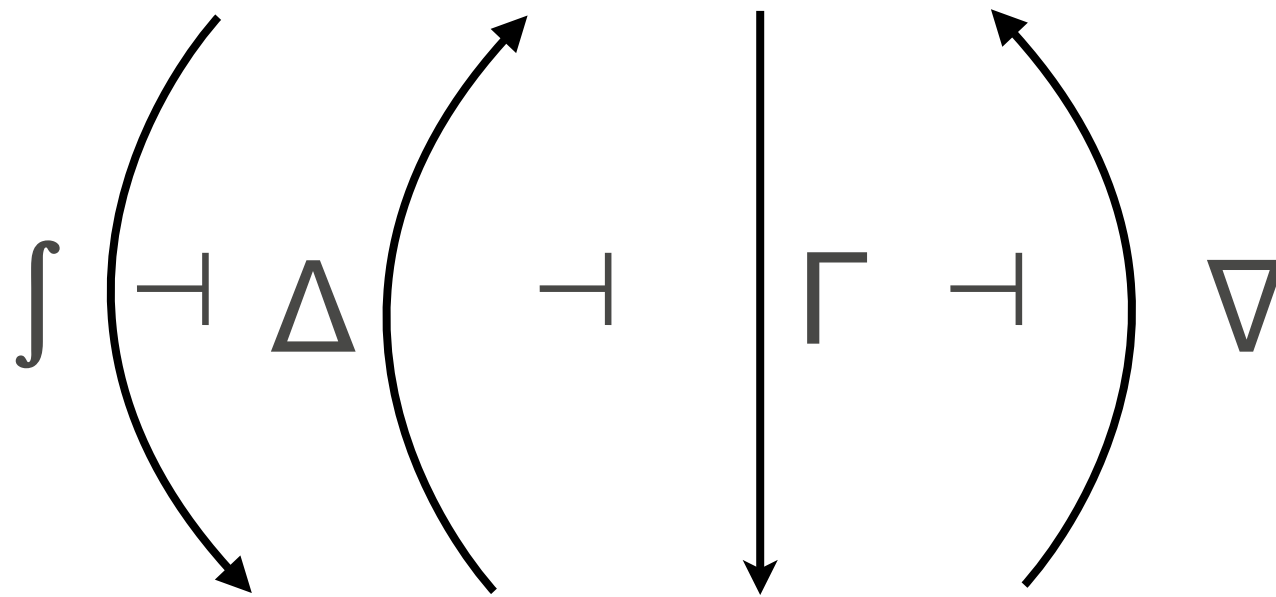
$\emptyset$ -type
usual topology

Modalities



Cohesion [Lawvere; Schrieber, Shulman]

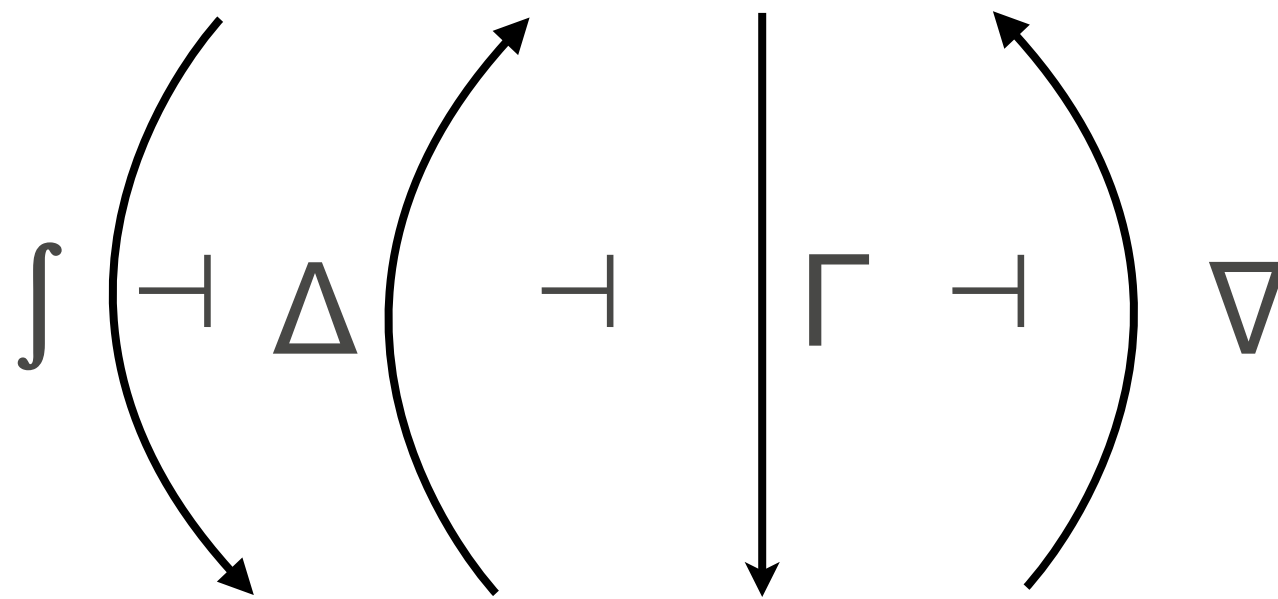
∞ -groupoids with topology on each level



∞ -groupoids

Cohesion [Lawvere; Schrieber, Shulman]

∞ -groupoids with topology on each level

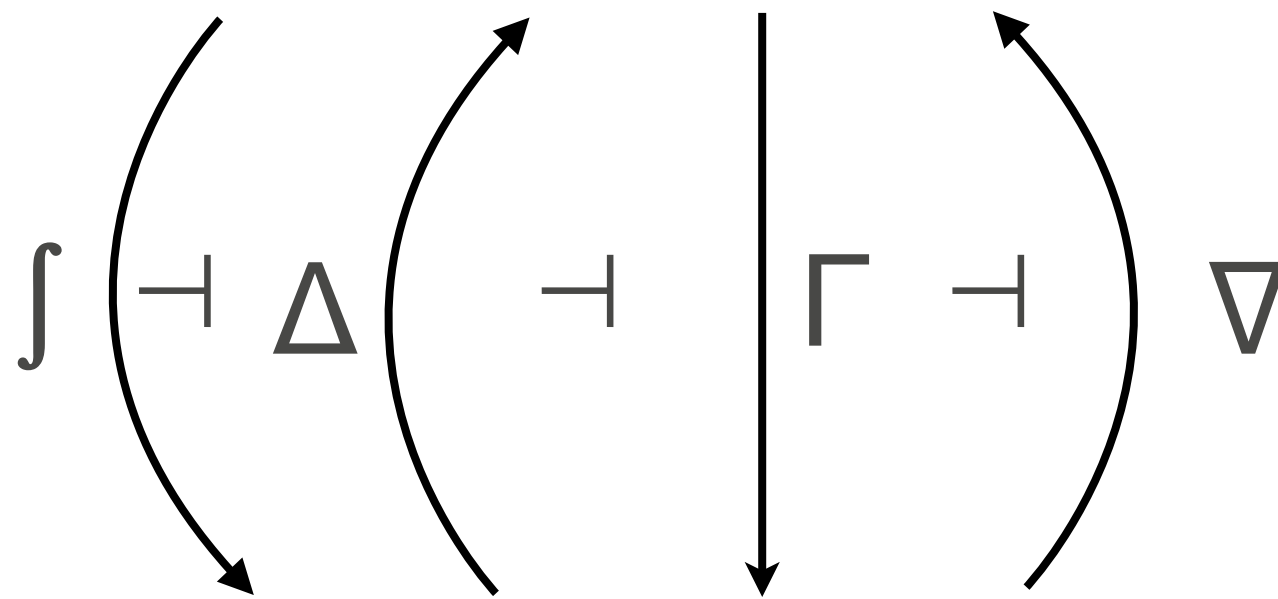


∞ -groupoids

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

Cohesion [Lawvere; Schrieber, Shulman]

∞ -groupoids with topology on each level

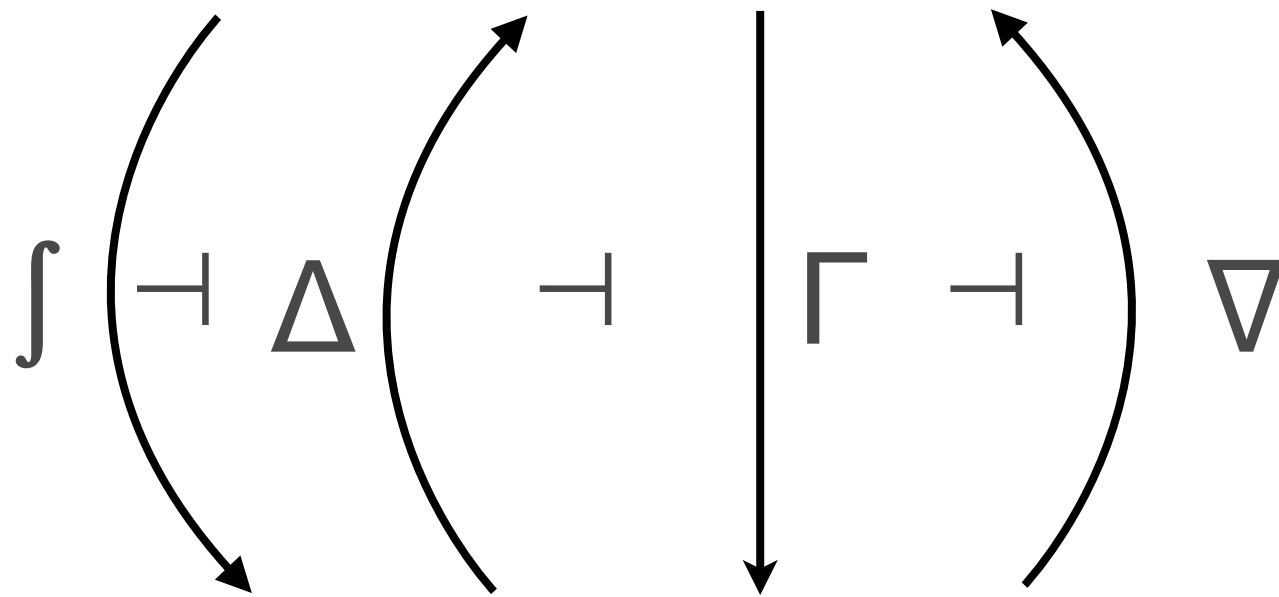


∞ -groupoids

$$\flat A := \Delta \Gamma A \quad \# A := \nabla \Gamma A$$

Cohesion [Lawvere; Schrieber, Shulman]

∞ -groupoids with topology on each level



∞ -groupoids

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

Cohesion [Lawvere; Schrieber, Shulman]

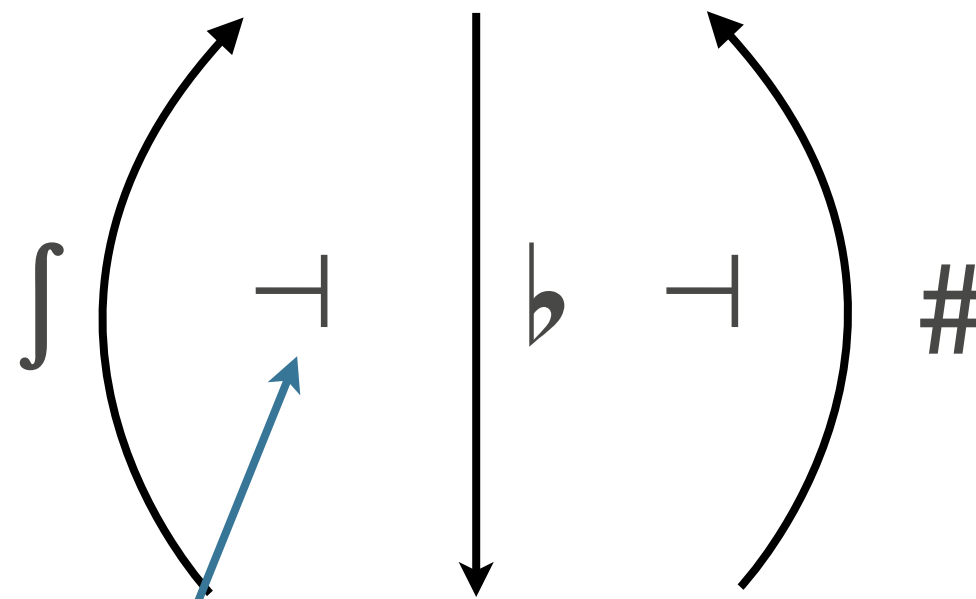
∞ -groupoids with topology on each level

$$\int \left(\dashv \quad \downarrow \quad \flat \quad \dashv \right) \#$$

∞ -groupoids with topology on each level

Cohesion [Lawvere; Schrieber, Shulman]

∞ -groupoids with topology on each level

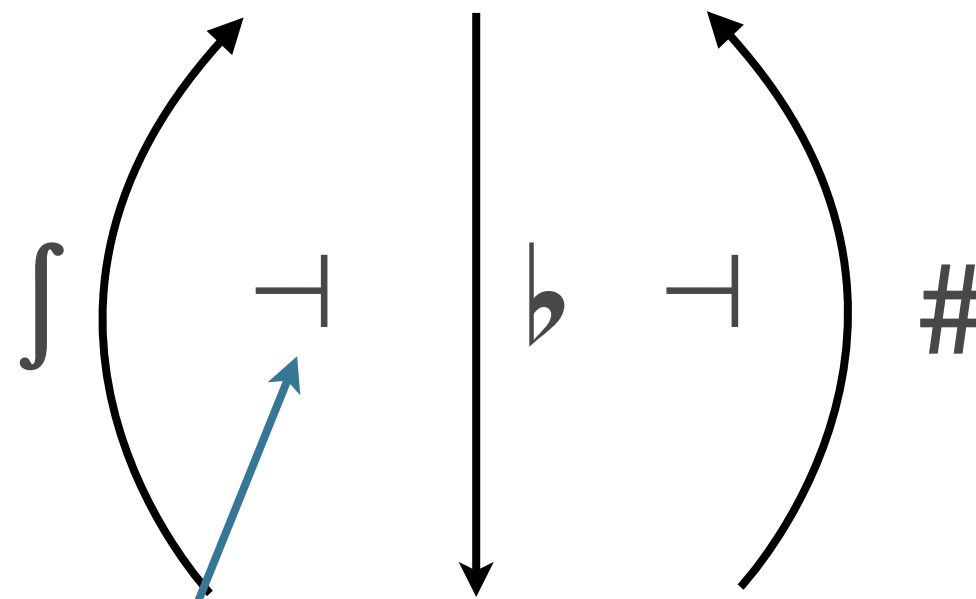


∞ -groupoids with topology on each level

$$\int \bigcirc \cong \bigcirc$$

Cohesion [Lawvere; Schrieber, Shulman]

∞ -groupoids with topology on each level

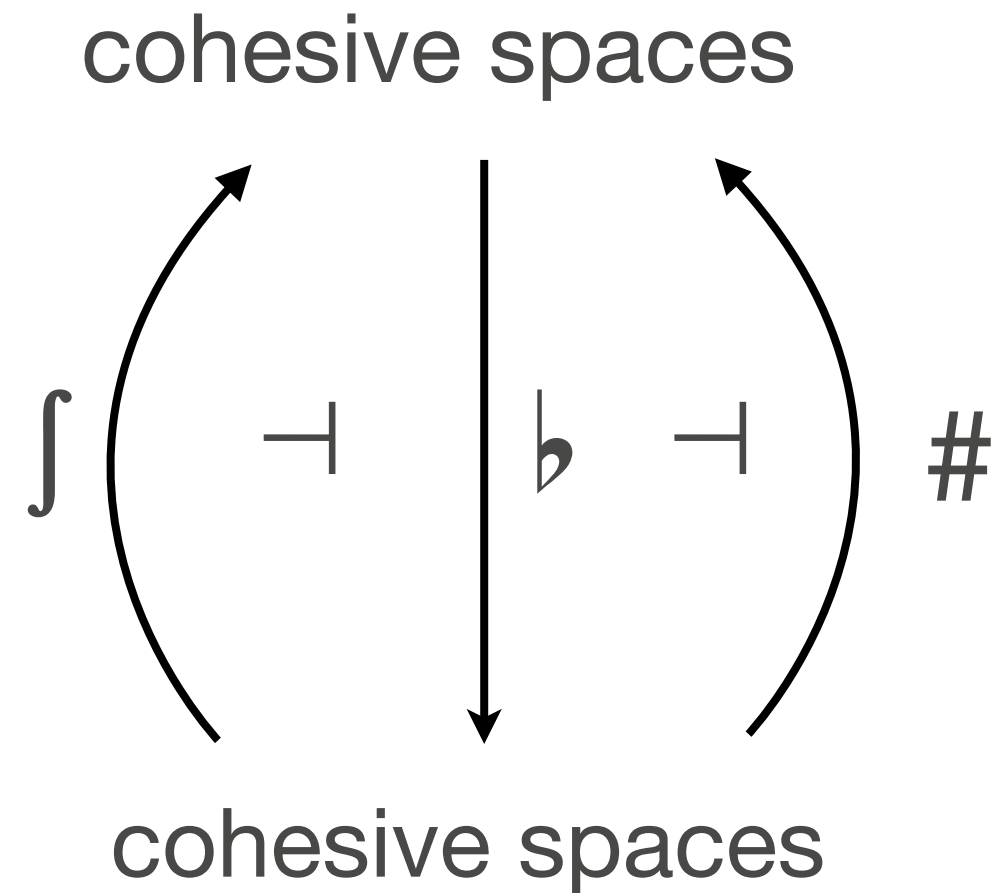


∞ -groupoids with topology on each level

$$\int \bigcirc \simeq \bigcirc$$

$$\int \bigcirc \simeq 1$$

Cohesion [Lawvere; Schrieber, Shulman]



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 $\#$, retraction $\flat A \rightarrow A \rightarrow \#A$)
[Finster, L., Morehouse, Riley]
- * Directed type theory [Riehl, Shulman]:
Involutive cohesion (opposites) [L., Riehl, Shulman]

Adjoint Logic

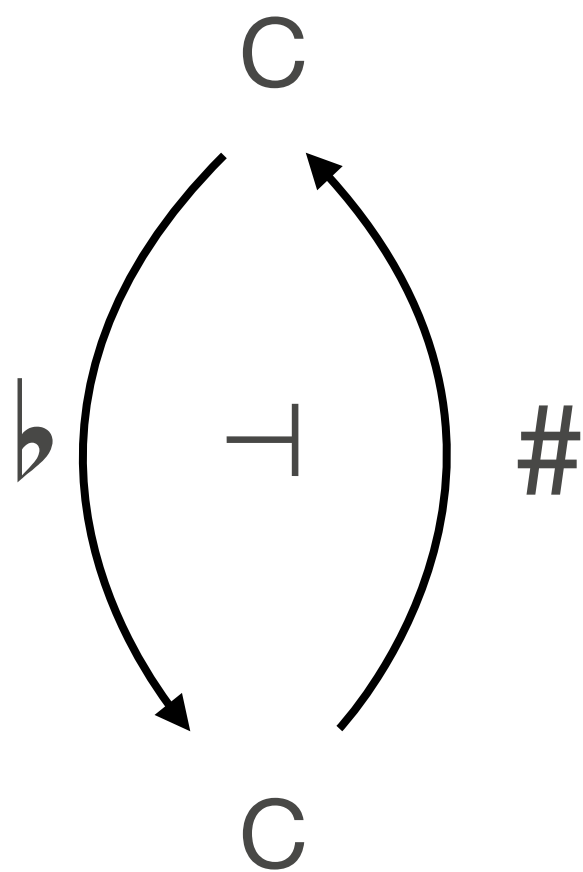
[Benton, Wadler'95; Atkey'04; Reed'09; L., Shulman'16]

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, $\square$ -strong)

Mode theories

generators and relations
in a cartesian 2-multicategory



$$b : C \rightarrow C$$

$$b \circ b = b$$

$$b \Rightarrow 1$$

$$b(x \times y) = bx \times by$$

F types

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

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

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

U types

$$\frac{\phi, c:q \vdash a : p \quad \Delta \text{ ctx}_\phi \quad A \text{ type}_p}{U_a(\Delta|A) \text{ type}_q}$$

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

$$\frac{}{\Delta, c:U_{c.a}(\Delta|A) \vdash_a A}$$

Adjoints

$$\flat \dashv \sharp$$

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

$$X \rightarrow \sharp Y$$

Adjoints

$$b \dashv \#$$

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

$$x:X \vdash_x U_b(Y)$$

$$X \rightarrow \#Y$$

Adjoints

$$\flat \dashv \sharp$$

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

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

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

$$X \rightarrow \sharp Y$$

Adjoints

$$b \dashv \#$$

$$bX \rightarrow Y$$

$$z:F_b(X) \vdash_z Y$$

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

$$x:X \vdash_x U_b(Y)$$

$$X \rightarrow \#Y$$

Adjoints

$$b \dashv \#$$

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

$$bX \rightarrow Y$$

$$z:F_b(X) \vdash_z Y$$

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

$$x:X \vdash_x U_b(Y)$$

$$X \rightarrow \#Y$$

Adjoints

$$b \dashv \#$$

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

$$bX \rightarrow Y$$

$$z:F_b(X) \vdash_z Y$$

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

$$x:X \vdash_x U_b(Y)$$

$$X \rightarrow \#Y$$

$$A \multimap Y$$

Adjoints

$$b \dashv \#$$

$$bX \rightarrow Y$$

$$z:F_b(X) \vdash_z Y$$

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

$$x:X \vdash_x U_b(Y)$$

$$X \rightarrow \#Y$$

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

$$x:X \vdash_x U_c \otimes a(a:A|Y)$$

$$A \multimap Y$$

Adjoints

$$\flat \dashv \sharp$$

$$\flat X \rightarrow Y$$

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

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

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

$$X \rightarrow \sharp Y$$

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

$$x:X, a:A \vdash_{x \otimes a} Y$$

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

$$A \multimap Y$$

Adjoints

$$\flat \dashv \sharp$$

$$\flat X \rightarrow Y$$

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

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

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

$$X \rightarrow \sharp Y$$

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

$$z:F_{x \otimes a}(x:X, a:A) \vdash_z Y$$

$$x:X, a:A \vdash_{x \otimes a} Y$$

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

$$A \multimap Y$$

Adjoints

$$\flat \dashv \sharp$$

$$\flat X \rightarrow Y$$

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

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

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

$$X \rightarrow \sharp Y$$

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

$$X \otimes A$$

$$z:F_{x \otimes a}(x:X, a:A) \vdash_z Y$$

$$x:X, a:A \vdash_{x \otimes a} Y$$

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

$$A \multimap Y$$

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

$$\begin{array}{c}
\frac{y:A \vdash_y A \quad z:B \vdash_z B}{y:A, z:B \vdash_{y \times z} A \times B} \\
\frac{f(y) \otimes f(z) \Rightarrow f(y \times z) \quad y:A, z:B \vdash_{y \times z} A \times B}{y:A, z:B \vdash_{f(y) \otimes f(z)} F_f (A \times B)} \\
\frac{y:A, z:F_f B \vdash_{f(y) \otimes z} F_f (A \times B)}{y:F_f A, z:F_f B \vdash_{y \otimes z} F_f (A \times B)} \\
\frac{y:F_f A, z:F_f B \vdash_{y \otimes z} F_f (A \times B)}{x: F_f A \otimes F_f B \vdash_x F_f (A \times B)}
\end{array}$$

Semantics

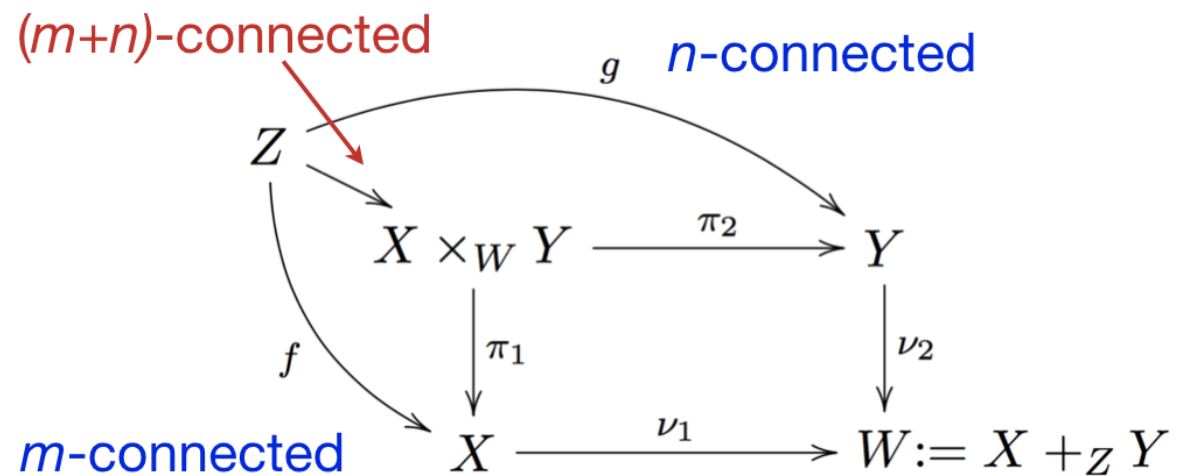
- * a functor from a cartesian multicategory (derivations) to another cartesian 2-multicategory (mode theory)
- * which is a locally discrete fibration (action of structural rules)
- * $F_\alpha \Delta$ makes this into an opfibration
- * $U_\alpha(\Delta|A)$ makes this into a fibration

internal languages:

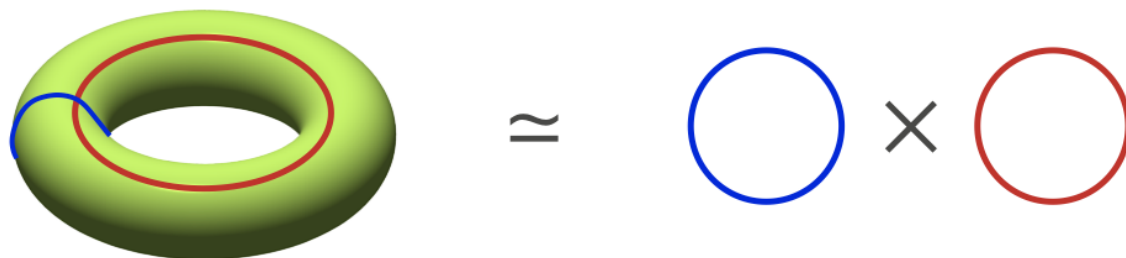
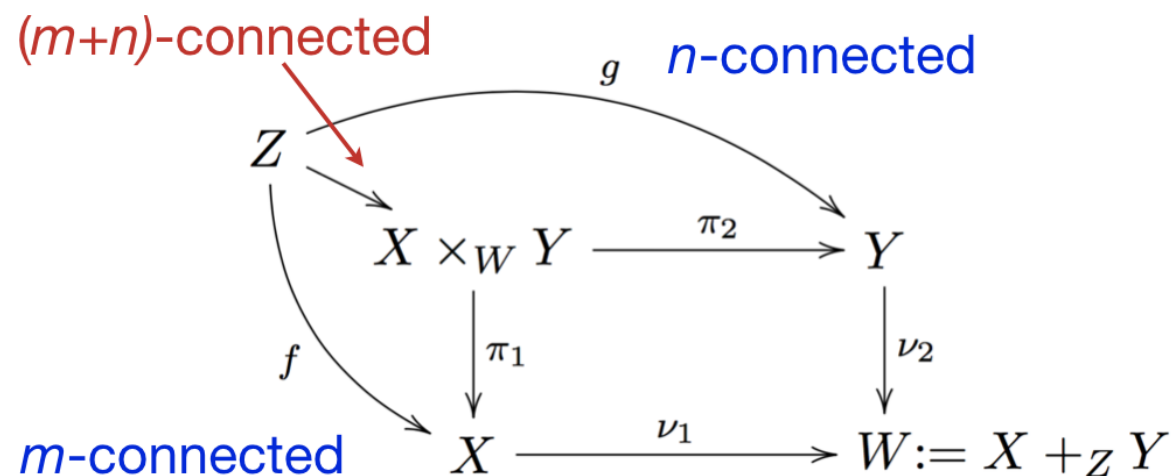
short statements *inside* a particular
categorical setting can unpack to
long external statements

internal languages:

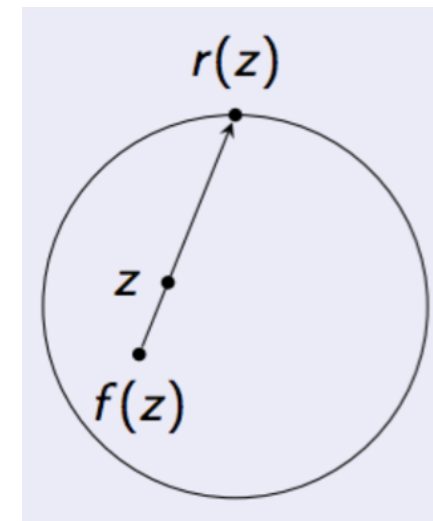
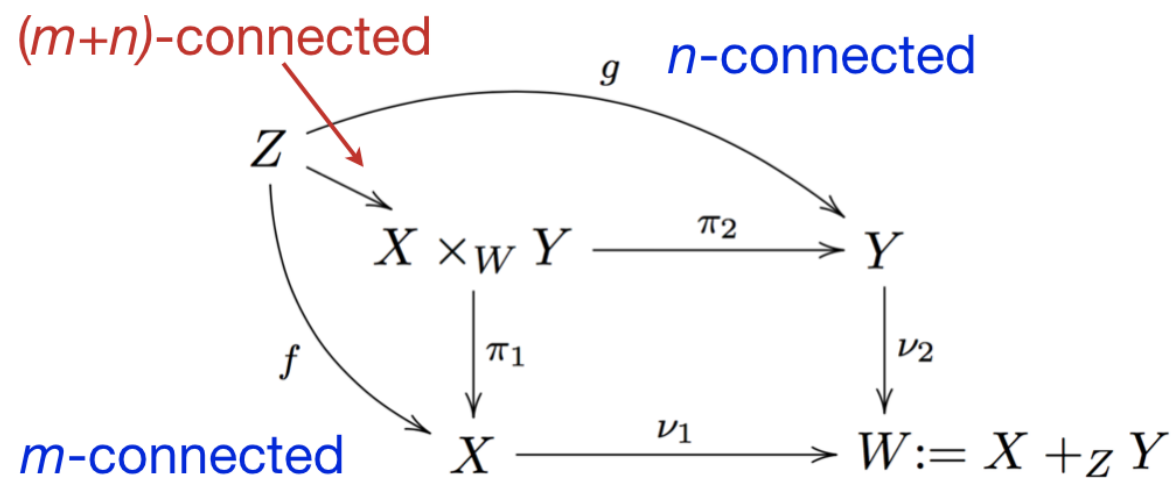
short statements *inside* a particular
categorical setting can unpack to
long external statements



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



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