

# Directed homotopy type theory

Paige Randall North

Utrecht University

14 December 2022

# Outline

Introduction

Directed homotopy theory

The hom-type former

Modal homotopy type theory

# Outline

Introduction

Directed homotopy theory

The hom-type former

Modal homotopy type theory

# Goal

## Goal

To develop a directed type theory.

# Goal

## Goal

To develop a directed type theory.

To formalize theorems about:

# Goal

## Goal

To develop a directed type theory.

To formalize theorems about:

- ▶ Higher category theory

# Goal

## Goal

To develop a directed type theory.

To formalize theorems about:

- ▶ Higher category theory
- ▶ Directed homotopy theory

# Goal

## Goal

To develop a directed type theory.

To formalize theorems about:

- ▶ Higher category theory
- ▶ Directed homotopy theory
  - ▶ Concurrent processes
  - ▶ Rewriting
  - ▶ Neural networks
  - ▶ ...



# Goal

## Goal

To develop a directed type theory.

To formalize theorems about:

- ▶ Higher category theory
- ▶ Directed homotopy theory
  - ▶ Concurrent processes
  - ▶ Rewriting
  - ▶ Neural networks
  - ▶ ...

# The Id-type

## Id-type

In MLTT, the *identity type* internalizes the notion of equality.

- ▶ There is a type  $\text{Id}_T(a, b)$  for any type  $T$  and  $a, b : T$
- ▶ Inductively generated by  $\text{refl}_a : \text{Id}_T(a, a)$

# The Id-type

## Id-type

In MLTT, the *identity type* internalizes the notion of equality.

- ▶ There is a type  $\text{Id}_T(a, b)$  for any type  $T$  and  $a, b : T$
- ▶ Inductively generated by  $\text{refl}_a : \text{Id}_T(a, a)$

## We can show:

- ▶ The relation  $\text{Id}_T(a, b)$  is (reflexive), transitive, and symmetric
- ▶ The identity type can be iterated:  
 $p, q : \text{Id}_T(a, b), \quad \alpha, \beta : \text{Id}_{\text{Id}_T(a, b)}(p, q), \quad \dots$
- ▶ It is possible for  $\text{Id}_{\text{Id}_T(a, b)}(p, q)$  to be empty.

Thus the Id-type constructor endows each type with the structure of an  $\infty$ -groupoid, or *space*.

# The Id-type

## Id-type

In MLTT, the *identity type* internalizes the notion of equality.

- ▶ There is a type  $\text{Id}_T(a, b)$  for any type  $T$  and  $a, b : T$
- ▶ Inductively generated by  $\text{refl}_a : \text{Id}_T(a, a)$

## We can show:

- ▶ The relation  $\text{Id}_T(a, b)$  is (reflexive), transitive, and symmetric
- ▶ The identity type can be iterated:  
 $p, q : \text{Id}_T(a, b), \quad \alpha, \beta : \text{Id}_{\text{Id}_T(a, b)}(p, q), \quad \dots$
- ▶ It is possible for  $\text{Id}_{\text{Id}_T(a, b)}(p, q)$  to be empty.

Thus the Id-type constructor endows each type with the structure of an  $\infty$ -groupoid, or *space*.

→ homotopy type theory

# What does directed mean? (Syntactically)

## Id-type is symmetric/undirected

For any type  $T$ , and terms  $a, b : T$ , there is a function

$$i : \text{Id}_T(a, b) \rightarrow \text{Id}_T(b, a)$$

so that any *path*  $p : \text{Id}_T(a, b)$  can be *inverted* to obtain a path  $ip : \text{Id}_T(b, a)$ .

# What does directed mean? (Syntactically)

## Id-type is symmetric/undirected

For any type  $T$ , and terms  $a, b : T$ , there is a function

$$i : \text{Id}_T(a, b) \rightarrow \text{Id}_T(b, a)$$

so that any *path*  $p : \text{Id}_T(a, b)$  can be *inverted* to obtain a path  $ip : \text{Id}_T(b, a)$ .

- Can think of identity terms as *undirected* paths

# What does directed mean? (Syntactically)

## Id-type is symmetric/undirected

For any type  $T$ , and terms  $a, b : T$ , there is a function

$$i : \text{Id}_T(a, b) \rightarrow \text{Id}_T(b, a)$$

so that any *path*  $p : \text{Id}_T(a, b)$  can be *inverted* to obtain a path  $ip : \text{Id}_T(b, a)$ .

- ▶ Can think of identity terms as *undirected* paths
- ▶ Can we design a type former of *directed* paths that resembles  $\text{Id}$  but without its inversion operation  $i$ ?

# What does directed mean? (Semantically)

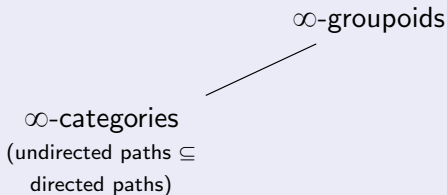
Categories vs. spaces

$\infty$ -groupoids



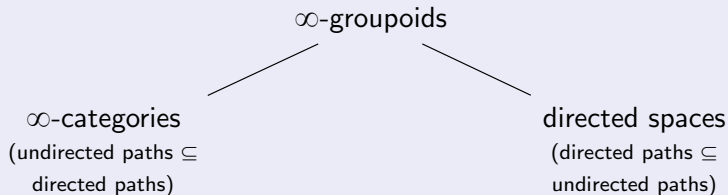
# What does directed mean? (Semantically)

## Categories vs. spaces



# What does directed mean? (Semantically)

## Categories vs. spaces



# Outline

Introduction

Directed homotopy theory

The hom-type former

Modal homotopy type theory

# Directed spaces

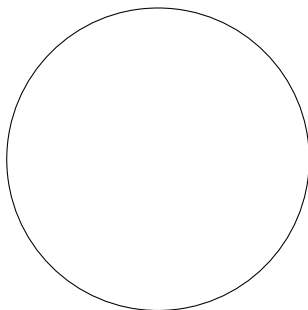
## Rough definition

A space together with a subset of its paths that are marked as 'directed'

# Directed spaces

## Rough definition

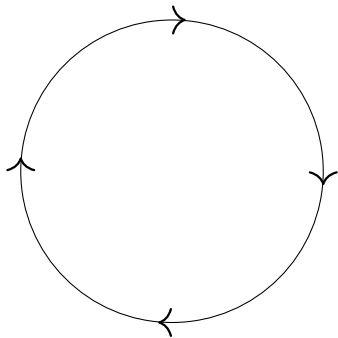
A space together with a subset of its paths that are marked as 'directed'



# Directed spaces

## Rough definition

A space together with a subset of its paths that are marked as 'directed'

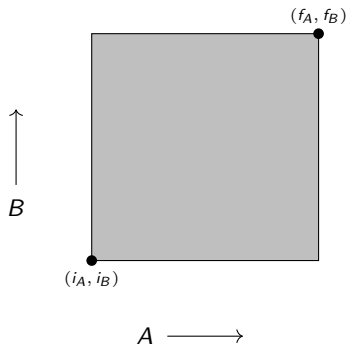


## Application: concurrency

Concurrent processes can be represented by directed spaces.

## Application: concurrency

Concurrent processes can be represented by directed spaces.

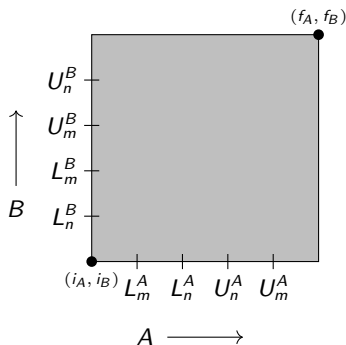


►  $A, B$  are two processes



## Application: concurrency

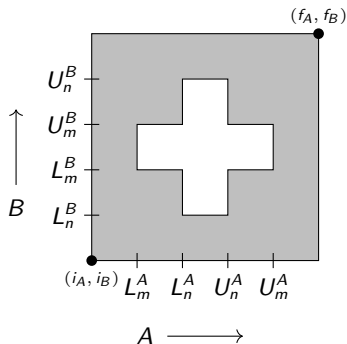
Concurrent processes can be represented by directed spaces.



- ▶  $A, B$  are two processes
- ▶  $m, n$  are two memory locations
- ▶ which can be locked ( $L$ ) or unlocked ( $U$ ) by each process

# Application: concurrency

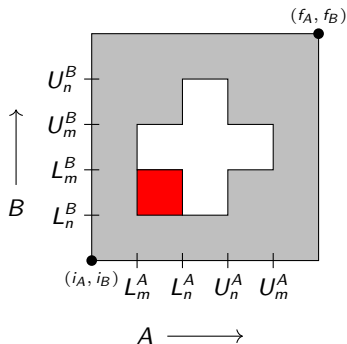
Concurrent processes can be represented by directed spaces.



- ▶  $A, B$  are two processes
- ▶  $m, n$  are two memory locations
- ▶ which can be locked ( $L$ ) or unlocked ( $U$ ) by each process

# Application: concurrency

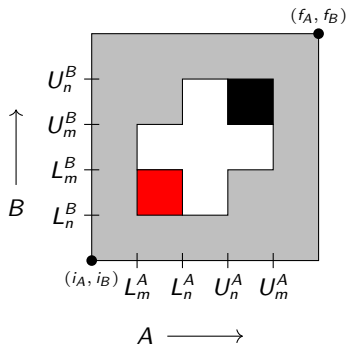
Concurrent processes can be represented by directed spaces.



- ▶  $A, B$  are two processes
- ▶  $m, n$  are two memory locations
- ▶ which can be locked ( $L$ ) or unlocked ( $U$ ) by each process

## Application: concurrency

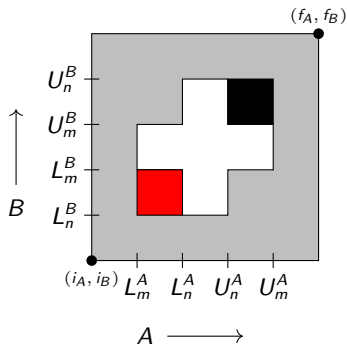
Concurrent processes can be represented by directed spaces.



- ▶  $A, B$  are two processes
- ▶  $m, n$  are two memory locations
- ▶ which can be locked ( $L$ ) or unlocked ( $U$ ) by each process

# Application: concurrency

Concurrent processes can be represented by directed spaces.



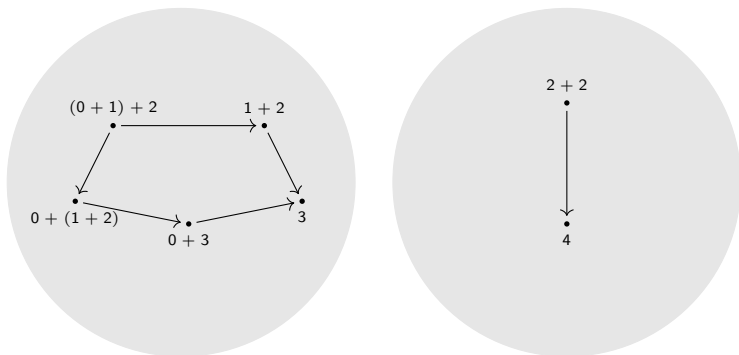
- ▶  $A, B$  are two processes
- ▶  $m, n$  are two memory locations
- ▶ which can be locked ( $L$ ) or unlocked ( $U$ ) by each process

## Fundamental questions:

- ▶ Which states are safe?
- ▶ Which states are reachable?

# Application: Term rewriting systems

Consider expressions in the monoid  $N = (\mathbb{N}, 0, +)$ .



- Interested in families  $D(n)$  indexed by  $n \in N$  for which rewrite rules  $n \rightarrow m$  induce rewrites  $D(n) \rightarrow D(m)$

# Outline

Introduction

Directed homotopy theory

The hom-type former

Modal homotopy type theory

## Rules for hom: core and op

$$\frac{T \text{ type}}{T^{\text{core}} \text{ type}}$$

$$\frac{T \text{ type}}{T^{\text{op}} \text{ type}}$$

$$\frac{T \text{ type} \quad t : T^{\text{core}}}{it : T}$$

$$\frac{T \text{ type} \quad t : T^{\text{core}}}{i^{\text{op}}t : T^{\text{op}}}$$



# Rules for hom: formation

## Id formation

$$\frac{T \text{ type} \quad s : T \quad t : T}{\text{Id}_T(s, t) \text{ type}}$$

## hom formation

$$\frac{T \text{ type} \quad s : T^{\text{op}} \quad t : T}{\text{hom}_T(s, t) \text{ type}}$$

## Rules for hom: introduction

### Id introduction

$$\frac{T \text{ type} \quad t : T}{r_t : \text{Id}_T(t, t) \text{ type}}$$

### hom introduction

$$\frac{T \text{ type} \quad t : T^{\text{core}}}{1_t : \text{hom}_T(i^{\text{op}}t, it) \text{ type}}$$

# Rules for hom: right elimination and computation

## Id elimination and computation

$$\frac{\begin{array}{c} T \text{ type} \\ s : T, t : T, f : \text{Id}_T(s, t) \vdash D(f) \text{ type} \quad s : T \vdash d(s) : D(r_s) \end{array}}{\begin{array}{c} s : T, t : T, f : \text{Id}_T(s, t) \vdash j(d, f) : D(f) \\ s : T \vdash j(d, r_s) \equiv d(s) : D(r_s) \end{array}}$$

## hom right elimination and computation

$$\frac{\begin{array}{c} T \text{ type} \quad s : T^{\text{core}}, t : T, f : \text{hom}_T(i^{\text{op}}s, t) \vdash D(f) \text{ type} \\ s : T^{\text{core}} \vdash d(s) : D(1_s) \end{array}}{\begin{array}{c} s : T^{\text{core}}, t : T, f : \text{hom}_T(i^{\text{op}}s, t) \vdash e_R(d, f) : D(f) \\ s : T^{\text{core}} \vdash e_R(d, 1_s) \equiv d(s) : D(1_s) \end{array}}$$

# Rules for hom: left elimination and computation

## Id elimination and computation

$$\frac{\begin{array}{c} T \text{ type} \\ s : T, t : T, f : \text{Id}_T(s, t) \vdash D(f) \text{ type} \quad s : T \vdash d(s) : D(r_s) \end{array}}{\begin{array}{c} s : T, t : T, f : \text{Id}_T(s, t) \vdash j(d, f) : D(f) \\ s : T \vdash j(d, r_s) \equiv d(s) : D(r_s) \end{array}}$$

## hom left elimination and computation

$$\frac{\begin{array}{c} T \text{ type} \quad s : T^{\text{op}}, t : T^{\text{core}}, f : \text{hom}_T(s, it) \vdash D(f) \text{ type} \\ s : T^{\text{core}} \vdash d(s) : D(1_s) \end{array}}{\begin{array}{c} s : T^{\text{op}}, t : T^{\text{core}}, f : \text{hom}_T(s, it) \vdash e_L(d, f) : D(f) \\ s : T^{\text{core}} \vdash e_L(d, 1_s) \equiv d(s) : D(1_s) \end{array}}$$

# Syntactic results

- ▶ Transport: for a dependent type  $t : T \vdash S(t)$ :

$$\begin{array}{l} t : T^{\text{core}}, t' : T, f : \text{hom}_T(i^{\text{op}}t, t'), s : S(it) \\ \vdash \text{transport}_R(s, f) : S(t') \end{array}$$

# Syntactic results

- ▶ Transport: for a dependent type  $t : T \vdash S(t)$ :

$$\begin{array}{l} t : T^{\text{core}}, t' : T, f : \text{hom}_T(i^{\text{op}}t, t'), s : S(it) \\ \vdash \text{transport}_R(s, f) : S(t') \end{array}$$

- ▶ Composition: for a type  $T$ :

$$\begin{array}{l} r : T^{\text{op}}, s : T^{\text{core}}, t : T, f : \text{hom}_T(r, is), g : \text{hom}_T(i^{\text{op}}s, t) \\ \vdash \text{comp}_R(f, g) : \text{hom}_T(r, t) \end{array}$$

# The interpretation

- ▶ Dependent types are represented by functors  $T : \Gamma \rightarrow \mathbf{Cat}$ .
- ▶  $T^{\text{core}}$  is represented by the objects of  $T$
- ▶  $T^{\text{op}}$  is represented by the opposite of  $T$
- ▶  $\text{hom}$  is represented by  $\text{hom} : T^{\text{op}} \rightarrow T \rightarrow \mathbf{Set}$
- ▶  $1_\bullet$  is represented by the identity morphisms
- ▶ There are two computation rules since in  $\Sigma_{x,y} \text{hom}(x,y)$ , given a  $f : \text{hom}(x,y)$ , there are two arrows from an identity to  $f$ :  
postcomposing  $1_x$  with  $f$  and precomposing  $1_y$  with  $f$

# The interpretation

- ▶ Dependent types are represented by functors  $T : \Gamma \rightarrow \mathbf{Cat}$ .
- ▶  $T^{\text{core}}$  is represented by the objects of  $T$
- ▶  $T^{\text{op}}$  is represented by the opposite of  $T$
- ▶  $\text{hom}$  is represented by  $\text{hom} : T^{\text{op}} \rightarrow T \rightarrow \mathbf{Set}$
- ▶  $1_\bullet$  is represented by the identity morphisms
- ▶ There are two computation rules since in  $\Sigma_{x,y} \text{hom}(x,y)$ , given a  $f : \text{hom}(x,y)$ , there are two arrows from an identity to  $f$ :  
postcomposing  $1_x$  with  $f$  and precomposing  $1_y$  with  $f$

## Point

In this model, terms of  $\text{hom}$ -types are not always invertible, so they are not always invertible in the type theory.



# Outline

Introduction

Directed homotopy theory

The hom-type former

Modal homotopy type theory

# Problems with the first attempt

The functions  $\text{op}$ ,  $\text{core}$  are problematic.

- ▶ There are no introduction rules for  $T^{\text{core}}$  or  $T^{\text{op}}$
- ▶ Including the identity type causes the  $\text{hom}$  type to collapse to the identity type on elements of  $T^{\text{core}}$ .
- ▶ We need a  $\text{op}$  function on the universe; e.g. the **1-functor**  $\text{op} : \text{Cat} \rightarrow \text{Cat}$ . This does not exist for 2-categories and up.

# Modal directed homotopy type theory

## Solution

The solution is to properly account for core, op, etc.

- ▶ **syntactically**: modal type theory
- ▶ **semantically**: multisided weak factorization systems (jww van den Berg, McCloskey)

(The theory of multisided weak factorization systems accounts for multiple fibrations – e.g. Grothendieck fibrations, opfibrations, isofibrations – in one category and how they interact, inspired by the two-sided fibrations of Street.)

# Modal directed type theory

The idea:

- ▶ Forget about having a type constructors  $T \mapsto T^{\text{op}}, T^{\text{core}}$

$$x : R^{\text{op}}, y : S^{\text{core}} \vdash T$$

- ▶ Instead op and core should be descriptions of how variables can be used.

$$x \vdash^{\bar{\cdot}} R, y \vdash^{\circ} S \vdash T$$

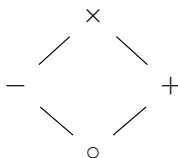
- ▶ Compare with linear / n-use type theory (Reed, McBride, Licata-Shulman-Riley, Abel...)

$$x \vdash^0 R, y \vdash^3 S \vdash T$$

## A modal approach

Following Licata-Shulman-Riley's (2017) modal framework for the sequent calculus.

- ▶ four modes that form a lattice:



- ▶ with an multiplication
  - ▶  $+$  is the identity
  - ▶  $o$  is almost an absorbing element:  $o a = a o = o$  except  $o x = x$
  - ▶  $x$  is almost an absorbing element:  $x a = a x = x$  except  $x o = o$
  - ▶  $-$  is a root of unity:  $-- = +$ .
- ▶ This is a sub-monoidal category of  $[\text{Cat}, \text{Cat}]$

# Orientations

Contexts are annotated by *orientations*. We write:

$$w : A, x : B, y : C, z : D \vdash T$$

or

$$w : A, x : B, y : C, z : D \vdash_{x,y^-,z^0} T$$

# Orientations

Contexts are annotated by *orientations*. We write:

$$w : A, x : B, y : C, z : D \vdash T$$

or

$$w : A, x : B, y : C, z : D \vdash_{x,y^-,z^0} T$$

Orientations on contexts inherit an order from the lattice, so we use the following rule.

$$\frac{\Gamma \vdash_{\ell} \mathcal{J} \quad m \leq \ell \quad \Gamma\text{-ort}}{\Gamma \vdash_m \mathcal{J}} \text{ORT-SUBST}$$

# Structural rules

$$\frac{\Gamma, x : \sigma, \Delta \text{ ctx}}{\Gamma, x : \sigma, \Delta \vdash_x x : \sigma} \text{VAR}$$

$$\frac{\Gamma, \Delta \vdash_{\ell, m} \mathcal{J} \quad \Gamma \vdash_{\ell} \rho \text{ type}}{\Gamma, x : \rho, \Delta \vdash_{\ell, m} \mathcal{J}} \text{WEAK}$$

$$\frac{\begin{array}{cc} \Gamma, x : \rho, \Delta \vdash_{\ell, x^{\omega}, m} T & \Gamma \vdash_{\ell} U : \rho \\ n \leq \ell \text{ } \Gamma\text{-ort} & n \leq \omega \cdot \ell \text{ } \Gamma\text{-ort} \end{array}}{\Gamma, \Delta[U/x] \vdash_{n, m} T[U/x]} \text{SUBST}$$



# The new hom-type

## hom formation

$$\frac{\Gamma \vdash_{\ell} A \text{ type}}{\Gamma, x : A, y : A \vdash_{\ell, x^-, y} \text{hom}_A(x, y) \text{ type}} \text{ hom-FORM}$$

## hom introduction

$$\frac{\Gamma \vdash_{\ell} A \text{ type}}{\Gamma, x : A \vdash_{\ell, x^{\circ}} 1_x : \text{hom}_A(x, x) \text{ type}} \text{ hom-INTRO}$$

# The new Id-type

## Id<sup>◦</sup> formation

$$\frac{\Gamma \vdash_{\ell} A \text{ type}}{\Gamma, x : A, y : A \vdash_{\ell, x^{\circ}, y^{\circ}} \text{Id}_A^{\circ}(x, y) \text{ type}} \text{Id}^{\circ}\text{-FORM}$$

## Id<sup>◦</sup> introduction

$$\frac{\Gamma \vdash_{\ell} A \text{ type}}{\Gamma, x : A \vdash_{\ell, x^{\circ}} \text{refl}_x : \text{Id}_A^{\circ}(x, x) \text{ type}} \text{Id}^{\circ}\text{-INTRO}$$

# The new Id-type

## Id<sup>×</sup> formation

$$\frac{\Gamma \vdash_{\ell} A \text{ type}}{\Gamma, x : A, y : A \vdash_{\ell, x^{\times}, y^{\times}} \text{Id}^{\times}(x, y) \text{ type}} \text{Id}^{\times}\text{-FORM}$$

## Id<sup>×</sup> introduction

$$\frac{\Gamma \vdash_{\ell} A \text{ type}}{\Gamma, x : A \vdash_{\ell, x^{\times}} \text{refl}_x : \text{Id}^{\times}(x, x) \text{ type}} \text{Id}^{\times}\text{-INTRO}$$

# Inside the type theory

What can we do?

- ▶ Find inclusions  $\text{Id}^\circ(a, b) \rightarrow \text{hom}(a, b) \rightarrow \text{Id}^\times(a, b)$ , but not  $\text{hom}(a, b) \rightarrow \text{Id}^\circ(a, b)$ .
- ▶ Transport and compose.

What can't we do?

- ▶ Form all  $\Sigma$  types (F types in LSR). For example, the one you should get from  $a : A \vdash_{a-} 1$  is  $A^{\text{op}}$ .

Future work

- ▶ Connect this formally with the intended semantics (jww van den Berg-McCloskey and Ahrens-van der Weide)
- ▶ Understand which  $\Sigma$  types exist.
- ▶  $\Pi$ -types, directed univalence, higher inductive types, etc...

Thank you!