Σ-types

Given $b:B \vdash E(b)$ type (in log $E\ (b:B) : \mathcal{U}$), want to form a type whose terms are dependent pairs $\langle b, e \rangle$ where $b:B$, $e:E(b)$.

Dependent pair types Σ

Σ-form:
$$\frac{\Gamma, x:P \vdash Q(x)}{\Gamma \vdash \sum_{x:P} Q(x)}$$

Σ-intro:
$$\frac{\Gamma \vdash p:P \qquad \Gamma \vdash q : Q(p)}{\Gamma \vdash \mathrm{pair}(p,q) : \sum_{x:P} Q(p)}$$

Σ-elim:
$$\frac{\begin{array}{c}\Gamma, z : \sum_{x:P} Q(p) \vdash D(z) \\ \Gamma, x:P, y:Q(x) \vdash a : D(\mathrm{pair}(x,y))\end{array}}{\Gamma, z : \sum_{x:P} Q(p) \vdash \mathrm{ind}_\Sigma(a,z) : D(z)}$$

Σ-comp:
$$\frac{\begin{array}{c}\Gamma, z : \sum_{x:P} Q(p) \vdash D(z) \\ \Gamma, x:P, y:Q(x) \vdash a : D(\mathrm{pair}(x,y))\end{array}}{\Gamma, x:P, y:Q(x) \vdash \mathrm{ind}_\Sigma(a, \mathrm{pair}(x,y)) \doteq a : D(\mathrm{pair}(x,y))}$$

Exercise.
- Construct a function $\pi_1 : \sum_{x:P} Q(x) \to P$.
- Construct a function $\pi_2 : \prod_{s : \sum_{x:P} Q(x)} Q(\pi_1 s)$.

Types as logic, sets, programs (Curry-Howard, Brouwer-Heyting-Kolmogorov)

	Logic	Sets	Program
Γ ctx	hypotheses	indexing set	names in scope
$\Gamma \vdash T$ type	predicate T on Γ	family T of sets on Γ $\begin{smallmatrix} T \\ \downarrow \\ \Gamma \end{smallmatrix}$	program specification using values from Γ
$\Gamma \vdash t : T$	proof of T	section, i.e., $T(\gamma)$ for all $\gamma \in \Gamma$	program
$\mathbb{N}$	—	$\mathbb{N}$	program w/ no input that outputs a nn
$S + T$ $\left(\sum_{i:bool} T_i\right)$	$\vee$	$\cup$	$\vee$
$S \times T$ $\left(\sum_{s:S} T\right)$	$\wedge$	$\times$	$\wedge$
$S \to T$ $\left(\prod_{s:S} T\right)$	$\Rightarrow$	$\to$	turn one kind of program into another
$\sum_{b:B} E(b)$	$\exists$	$\bigsqcup$	Σ
$\prod_{b:B} E(b)$	$\forall$	$\prod$ (set of sections)	Π

The strongest inductive type : Id

Why do we need the identity type?

(If we're not interested in homotopy.)

A1: There are many equalities that hold only propositionally.

Ex. $\mathrm{add}(x, 0) \doteq x$
$\mathrm{add}(x, sy) \doteq s\,\mathrm{add}(x,y)$
One cannot prove $\mathrm{add}(0,x) \doteq x$.

To prove this, we need to induct on n (i.e. use $\mathbb{N}$-elimination), but this only allows us to construct a term of a type.

We will be able to prove $\mathrm{add}(0,x) = x$.

A2: We already have a notion of equality:

judgmental equality $\doteq$

(The identity type is called propositional equality $=$.)

Logical interpretation: propositions are types / proofs are terms.

To prove an equality (and be consistent with the logical interpretation) we want to produce a term of a type of equalities.

Type constructors often internalize structure

- bool
- $\mathbb{N}$
- $\emptyset$
- $\mathbb{1}$

can also be seen as internalizing external versions.

- The universe type internalizes the judgment of the form

$$A \text{ type}$$

- We'll see how the identity type internalizes judgmental equality...

Identity type $=$

$=$-form

$$\frac{\Gamma \vdash A \text{ type} \quad \Gamma \vdash a : A \quad \Gamma \vdash b : A}{\Gamma \vdash a =_A b \text{ type}}$$

$=$-intro

$$\frac{\Gamma \vdash a : A}{\Gamma \vdash r_a : a =_A a}$$

$=$-elim

$$\frac{\Gamma, x:A, y:A, z : x =_A y \vdash D(x,y,z) \text{ type} \qquad \Gamma, x:A \vdash d : D(x,x,r_x)}{\Gamma, x:A, y:A, z : x =_A y \vdash \mathrm{ind}_=(d,x,y,z) : D(x,y,z) \text{ type}}$$

$=$-comp

$$\frac{\Gamma, x:A, y:A, z : x =_A y \vdash D(x,y,z) \text{ type} \qquad \Gamma, x:A \vdash d : D(x,x,r_x)}{\Gamma, x:A \vdash \mathrm{ind}_=(d,x,x,r_x) \doteq d : D(x,x,r_x)}$$

Type constructors often internalize structure

- At a 'meta' level, we can talk about judgmental equality:

  Ex. $a \doteq b : A$

  We can discuss this at the 'type-and-term' level by using identity types:

  Ex. $r_a : a =_A b$

Note that the rules governing equality say that

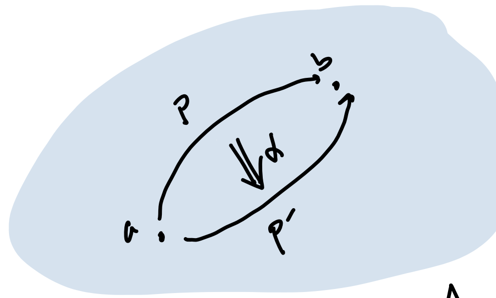
if $a \doteq b : A$, then $(a =_A a) \doteq (a =_A b)$, and
if $r_a : a =_A a$ and $(a =_A a) \doteq (a =_A b)$, then $r_a : a =_A b$.

→ Reflexivity (r_-) turns judgmental equalities into propositional equalities.

Exercise
- $add(0, n) = n$ for all $n : \mathbb{N}$.
  (Note: one of $add(0,n) \doteq n$ and $add(n,0) \doteq n$ will hold definitionally depending on how you defined add. Show the other one holds 'judgmentally'.)

The groupoidal behaviour of types

(The first homotopical phenomena)



A

We can now think of types as collections of points (terms) connected by homotopies/paths (equalities).

We can:

- have multiple equalities of the same type (ex: $p, p' : a =_A b$)
- (Ex.) take the inverse of an equality (if $q : b =_A c$, then $q^{-1} : c =_A b$)
- (Ex.) take composition of equalities (if $p : a =_A b$ and $q : b =_A c$, then $p \cdot q : a =_A c$)
- (Ex) have equalities of equalities ($\alpha : p =_{a =_A b} p'$)

Moreover: (Ex) functions $A \to B$ respect equality (i.e. map $a =_A a'$ to $fa =_B fa'$)
This is how <u>homotopies</u> in <u>spaces</u> behave.

The space interpretation

<u>Thm</u>. (Voevodsky) There is an interpretation of dependent type theory into Spaces (the category of Kan complexes) in which

types $\rightsquigarrow$ spaces
terms $\rightsquigarrow$ points
equalities $\rightsquigarrow$ paths

Transport

Prop. (Ex) For any dependent type $x:B \vdash E(x)$ type, any terms $b, b' : B$, and any equality $p : b =_B b'$, there is a function $tr_p : E(b) \to E(b')$.

- This ensures that everything respects propositional equality. If we think of E as a predicate on B, then if $E(b)$ is true and $b =_B b'$, so is $E(b')$.

- This is part of a more sophisticated relationship between type the and homotopy theory (Quillen model category theory). Transport sa that $\pi : \sum_{b:B} E(b) \longrightarrow B$ behaves like a fibration in a QMC.

Equivalence. For types S, T, there is a notion of equivalence

$$S \simeq T$$

Similar to

$$\sum_{f:S\to T} \sum_{g:T\to S} \left(\prod_{x:S} gfx = x\right) \times \left(\prod_{y:T} fgy = y\right).$$

(To be revisited later.)

Characterizing equality in standard types

bool: We can show false = false, true = true, false $\neq$ true.

$\mathbb{N}$: We have similar: $sn = sm \Rightarrow n = m$, $0 \neq sn$

Σ-types: For $s,t : \sum_{a:A} B(a)$, have $(s = t) \simeq \sum_{p : \pi_1 s = \pi_1 t} tr_p \, \pi_2 s = \pi_2 t$.

Π-types: For $f,g : \prod_{a:A} B(a)$, maybe want $(f = g) \simeq \prod_{x:A} fx = gx$.

Not provable. Called functional extensionality. (funext)

Validated by interpretations in logic, sets, spaces.

$=$-types: For $p,q : a = b$, maybe want

$$(p = q) \simeq \mathbb{1}.$$

Not provable. Called uniqueness of identity proofs. (UIP)

Validated by interpretations in logic, sets.

U-types: For $S,T : U$, maybe want

$$(S = T) \simeq (S \simeq T).$$

Not provable. Called univalence. (UA)

Validated by interpretation in spaces.

- UA $\Rightarrow$ funext.
- UIP $\wedge$ funext $\not\Rightarrow \perp$
- UA + UIP $\Rightarrow \perp$.

We choose UA.