Introduction to Homotopy Type Theory

Nicolai Kraus

Midlands Graduate School 2021, 12–16 April

Ways to introduce (homotopy) type theory

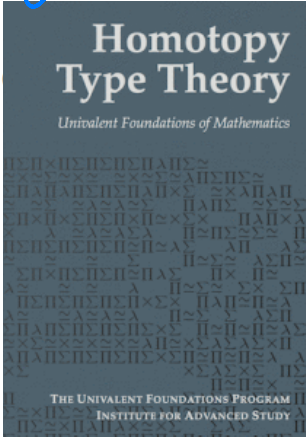
Agda

```
x·-mono : ∀ {x y z : Brw} → y ≤ z → x · y ≤ x · z
x·-mono ≤-zero = ≤-zero
x·-mono (≤-trans y≤w w≤z) = ≤-trans (x·-mono y≤w) (x·-mono w≤z)
x·-mono {x} (≤-succ-mono {y} {z} y≤z) = +x-mono x (x·-mono {x} y≤z)
x·-mono {x} {y} (≤-cocone f {k = k} y≤z) with decZero x
... | yes x≡zero = subst (λ z → z ≤ zero) (sym (zero·y≡zero y) • cong
... | no x≢zero = ≤-cocone (λ n → x · f n) {k = k} (x·-mono y≤z)
x·-mono {x} (≤-limiting f f≤z) with decZero x
... | yes x≡zero = ≤-zero
... | no x≢zero = ≤-limiting (λ n → x · f n) λ k → x·-mono (f≤z k)
x·-mono (≤-trunc p q i) = ≤-trunc (x·-mono p) (x·-mono q) i
```

inference rules

$$\frac{\Gamma, x{:}\mathbf{1} \vdash C : \mathcal{U}_i \qquad \Gamma \vdash c : C[\star/x] \qquad \Gamma \vdash a : \mathbf{1}}{\Gamma \vdash \mathsf{ind}_{\mathbf{1}}(x.C, c, a) : C[a/x]}\ \mathbf{1}\text{-}\mathrm{ELIM}$$

natural math language



Prerequisites

- *Basic examples of types* Nat Bool Unit Empty
- *Universe(s)* Type $A, B : \mathrm{Type}$
- *Function types* $A \to B$
- *Dependent function types* if $C : A \to \mathrm{Type}$, then $(a : A) \to C\,a$ [aka $\Pi(a:A).\,C\,a$]
- *Binary products (pairs) and coproducts (sums)* $A \times B$ $A \uplus B$
- *Dependent pairs* $\Sigma(a:A).\,C\,a$
- *Inductive types* see above

Program = Proof

Exercise: Formulate a type which says that there are infinitely many primes
(or simply: arbitrarily large primes)!

isPrime : $Nat \to Type$

isPrime(n) :≡ $\big((m\ k : N) \to (m \cdot k = n) \to (m = 1) \uplus (m = n)\big)$
$\times (n > 1)$

bigPrimes : $(n : Nat) \to \Sigma (p : Nat).\ isPrime(p)$
$\times\ p > n$

bigPrimes :≡ (exercise)

Caveat: judgments (meta-theoretic)

- A type "A is a valid type"
- $a : A$ "a is term of type A"
- $a \equiv b$ "a and b are the same" $(\lambda x. fx) y \equiv fy$
- $a :\equiv b$ same, by definition

We cannot ask or prove these statements *in* the language.

Exercise: Which of the following can we ask internally?

(1) Is 8 a natural number? no, b/c $8 : \mathrm{Nat}$ is judgment

(2) Is 8 a prime number? yes, b/c isPrime(8)

(3) Is "hello" a prime number? no, isPrime("hello") is not a type

(Identity *a.k.a.* equality *a.k.a.* identification *a.k.a.* path) types

- if $a, b : A$ then $a = b$ type "formation"
- given $a : A$ we have refl : $a = a$ "introduction"
- given $C : (a\, b : A) \to (a = b) \to$ Type "elimination"
  and $C\ a\ a$ refl (for all $a : A$) J aka path induction
  we get $C\ a\ b\ p$ (for all $\underbrace{a, b}_{A}$, $p : a = b$)
- applying this rule and asking for the refl case gives back the assumption "computation"

$\mathrm{sym} : (a\, b : A) \to a = b \to b = a$

$\mathrm{sym}\ a\ \cancel{b}^{\,a}\ \cancel{p}^{\,\mathrm{refl}} :\equiv \mathrm{refl}$

here: $C\ a\ b\ p :\equiv (b = a)$

computation: $\mathrm{sym}\ a\ a\ \mathrm{refl} \equiv \mathrm{refl}$

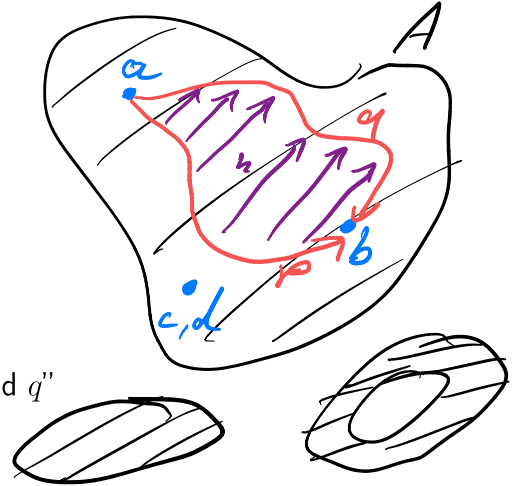
Model: types as spaces

The homotopical interpretation of dependent type theory works roughly like this:

judgment	interpretation
A type	"A is a space"
$a : A$	"a is a point in space A"
$c \equiv d$	"c and d are the same point"
$p : a = b$	"p is a path between a and b"
$h : p = q$	if p and q are equalities $a = b$: "h is a homotopy between p and q"

path in A

path in $a = b$

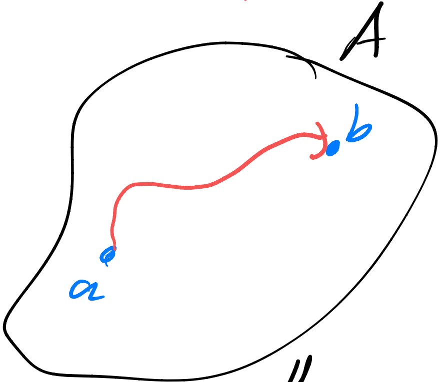


Types as spaces, examples: symmetry, transitivity

sym : $(a\, b : A) \;\to\; a = b \;\to\; b = a$

sym a $\cancel{b}$ $\cancel{p}$:≡ refl

 a $\;$ refl

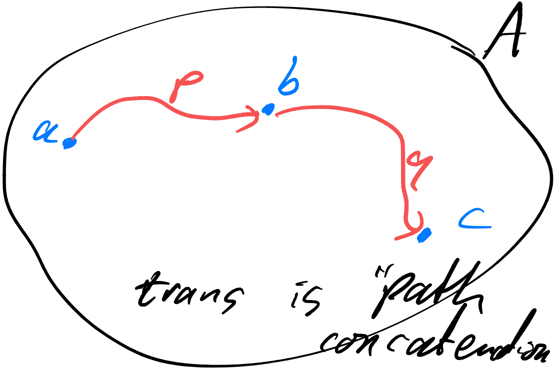


sym is "path reversal"

trans : $(a\, b\, c : A) \to$
$$a = b \;\to\; b = c \;\to\; a = c$$

trans a $\cancel{b}$ c $\cancel{p}$ q :≡ q

 a $\;$ refl



trans is "path concatenation"

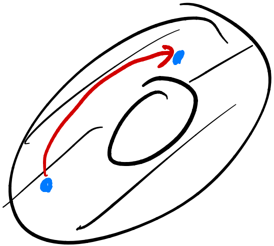
Types as spaces, example: $\underbrace{\text{uniqueness of identity proofs}}_{\text{UIP}}$

Can we prove this for all types A?

uip: $(a\,b : A) \to (p\,q : a = b) \to p = q$

uip a $\cancel{b}$ $\cancel{p}$ q $:\equiv$?

 a refl



A