Homotopy and Directed Type Theory: a Sample

Dan Doel

October 24, 2011

Type Theory Overview

- Judgments

$$\Gamma \vdash \mathrm{ctx}$$
$$\Gamma \vdash A \text{ type}$$
$$\Gamma \vdash M : A$$

- Families

$$\Gamma, x : A \vdash B(x) \text{ type}$$

- Inference

$$\frac{\Gamma_1 \vdash J_1}{\Gamma_2 \vdash J_2}$$

Type Theory Overview

Π and Σ

- Formation

$$\frac{\Gamma, x : A \vdash B(x) \text{ type}}{\Gamma \vdash \Pi_{x:A}.B(x) \text{ type}} \qquad \frac{\Gamma, x : A \vdash B(x) \text{ type}}{\Gamma \vdash \Sigma_{x:A}.B(x) \text{ type}}$$

- Introduction

$$\frac{\Gamma, x : A \vdash M : B(x)}{\Gamma \vdash (\lambda x : A.\, M) : \Pi_{x:A}.B(x)} \qquad \frac{\Gamma \vdash M : A \quad \Gamma \vdash N : B(M)}{\Gamma \vdash (M, N) : \Sigma_{x:A}.B(x)}$$

- Elimination

$$\frac{\Gamma \vdash F : \Pi_{x:A}.B(x) \quad \Gamma \vdash M : A}{\Gamma \vdash F\, M : B(M)}$$

$$\frac{\Gamma \vdash P : \Sigma_{x:A}.B(x)}{\Gamma \vdash \pi_1 P : A} \qquad \frac{\Gamma \vdash P : \Sigma_{x:A}.B(x)}{\Gamma \vdash \pi_2 P : B(\pi_1 P)}$$

Type Theory Overview

Inductive types

- Define a type as built out of constructors:

```
data ℕ : type where
  zero : ℕ
  suc  : ℕ -> ℕ
```

- Induction principle

```
induction :  (P : ℕ -> type)
          -> P(zero)
          -> ((n : ℕ) -> P(n) -> P(suc n))
          -> (n : ℕ) -> P(n)
```

Type Theory Overview

Identity types

- Definition

```
data Id A x : A -> type where
  refl : Id A x x
```

- Elimination

```
J :  (A : type) -> (x : A)
  -> (P : (z : A) -> Id A x z -> type)
  -> P x refl
  -> (y : A) -> (eq : Id A x y) -> P y eq
```

- Computation

```
J A M P PM M refl = PM
```

- Fancy notation: $M \simeq_A N$

Type Theory Overview

Identity types

- Simplified (but often useful) version of J:

```
subst :  (A : type) -> (P : A -> type)
         -> (x y : A) -> Id A x y
         -> P x -> P y
```

- Defining property of equality: respected by all predicates
- Very convenient: we needn't know anything about P to know that it respects equality
- Recurring theme: how far can we extend this respect?

Type Theory Overview

The universe U

- Formation

$$\frac{}{\Gamma \vdash \mathsf{U}\ \mathsf{type}} \qquad \frac{\Gamma \vdash S : \mathsf{U}}{\Gamma \vdash \mathsf{T}(S)\ \mathsf{type}}$$

- Introduction

$$\frac{}{\Gamma \vdash 0, 1, 2 : \mathsf{U}}$$

$$\frac{\Gamma \vdash S : \mathsf{U} \qquad \Gamma, x : \mathsf{T}(S) \vdash F : \mathsf{U}}{\Gamma \vdash \Pi\, S\, F : \mathsf{U}}$$

$$\dots$$

- No Elimination
- Keep in mind: $\simeq_{\mathsf{U}}$

Type Theory Overview

Set-theoretic model

- Types denote sets — including U
- Inductive types denote appropriate inductively defined sets
- The identity type denotes equality on said sets
  - We expect identities to be *propositions*.
  - This suggests a second eliminator for identities ...

Type Theory Overview

Axiom K

- Axiom K

```
K :  (A : type) -> (x : A)
   -> (P : Id A x x -> type)
   -> P refl -> (eq : Id A x x) -> P eq
```

- Also called Uniqueness of Identity Proofs (UIP)
- Two motivations
  - Identities are propositions
  - Id is an (indexed) inductive type generated by refl
- But K is not definable from J

The Homotopy Model

Standard and non-standard models

- Peano Arithmetic formalizes the natural numbers
- Similar to our $\mathbb{N}$ type earlier
- Induction principle:
  $P(0) \to (\forall k. P(k) \to P(1 + k)) \to \forall n. P(n)$
- Motivation: every natural number is 0 or a successor thereof
  - But is this what it says?
- No, there are non-standard models

The Homotopy Model

J as an induction principle

- J, interpreted similarly, says, "every identity proof is refl."
  - K also says this, but evidently in a different way
- This is similarly a mistranslation of J
- Admits (∞-)groupoid/homotopy models

The Homotopy Model

$\infty-$groupoids

- A groupoid is a category ...

$$\frac{A : G}{id_A : A \rightarrow A} \quad \frac{f : A \rightarrow B \quad g : B \rightarrow C}{g \circ f : A \rightarrow C} \quad \ldots$$

- ... in which all elements are invertible

$$\frac{f : A \rightarrow B}{f^{-1} : B \rightarrow A} \quad \frac{f : A \rightarrow B}{f^{-1} \circ f = id_A} \quad \frac{f : A \rightarrow B}{f \circ f^{-1} = id_B}$$

- An ∞-groupoid has infinitely many levels of transformations, and equations are expected to hold only up to higher equivalences.

The Homotopy Model

Types as ∞-groupoids

- A suspicious coincidence ...

$$\frac{M : A}{\mathsf{refl} : M \simeq_A M} \qquad \frac{F : M \simeq_A N \quad G : N \simeq_A O}{\mathsf{trans}\, F\, G : M \simeq_A O} \qquad \cdots$$

$$\frac{F : M \simeq_A N}{\mathsf{sym}\, F : N \simeq_A M} \qquad \frac{F : M \simeq_A N}{\cdots : (\mathsf{trans}\, F\, (\mathsf{sym}\, F)) \simeq_{M \simeq_A M} \mathsf{refl}}$$

- The above can all be defined using J
- Types together with the identity type naturally form a groupoid
- Identity types $M \simeq_A N$ have their own identity types $F \simeq_{M \simeq_A N} G$ ...
- ... and equations in general hold only up to higher identity types
- So types are naturally ∞-groupoids

The Homotopy Model

Homotopy n-types

- Sometimes, an ∞-groupoid only has finitely many non-trivial levels
- Called an (∞, n)-groupoid, or homotopy n-type
- Groupoids can be seen as 1-types, sets as 0-types, propositions as -1-types
  - There are "contractible" types, -2-types at the low end
- The dimension of a type in this regard is definable in type theory

The Homotopy Model

Homotopy n-types

```
Contractible : type -> type
Contractible A = Σ(x : A). Π(y : A). (Id A x y)

Proposition : type -> type
Proposition A = Π(x y : A). Contractible (Id A x y)

Type : ℕ -> type -> type
Type zero    A = Π(x y : A). Proposition (Id A x y)
Type (suc n) A = Π(x y : A). Type n (Id A x y)
```

The Homotopy Model

Homotopy n-types

- Homotopy -2-types are trivial
  - There is an element such that all elements are equivalent to it
- A type is a homotopy $(n+1)$-type if its identity types are homotopy n-types
  - Elements (proofs) of a proposition are trivially equivalent to each other (proof irrelevance)
  - Equality of elements of sets is a proposition
  - Objects of a groupoid have sets of isomorphisms between them.
  - ...

The Homotopy Model

Conclusion

- Intuitionistic type theory already admits this higher-dimensional model
  - This model is incompatible with the K axiom, however
- Our earlier “standard” model treated U as a set. . . .
- However, without an inductive eliminator, there is nothing stopping U from being modeled as a higher dimensional type!
  - A groupoid of sets
- U is not provably higher dimensional in standard type theory, of course

The Univalence Axiom

- The traditional model of type theory led us to K
- What does the homotopy model suggest?
  - U should be higher dimensional, how can we get there?

The Univalence Axiom

Equivalence

- We want inhabitants of U to be equivalent if there is an isomorphism between them.
- This is typically defined by the following progression:

$$\mathrm{IsEquiv} : (f : S \to T) \to \mathrm{type}$$

$$S \cong T = \Sigma_{f:S\to T}.\, \mathrm{IsEquiv}(f)$$

$$\mathrm{substEqv} : S \simeq_{\mathrm{U}} T \to S \cong T$$

$$\mathrm{univalence} : \mathrm{IsEquiv}(\mathrm{substEqv})$$

- substEqv being an equivalence implies that there is an inverse from $S \cong T$ to $S \simeq_{\mathrm{U}} T$

The Univalence Axiom

Consequences

- Isomorphism of sets implies identity
  - $\mathrm{Vec}\,A\,n \simeq \mathrm{Vec}\,A\,m \to n \simeq_{\mathbb{N}} m$?
- Univalence has been shown to imply extensionality of functions

$$(\Pi_{x:A}.f\ x \simeq_B g\ x) \to f \simeq_{A \to B} g$$

Higher Inductive Types

- We can define new sets via generators using inductive types
- Why not define new n-types?

```
data Circle : type where
  base : Circle
  loop : Id Circle base base

ind-Circle :
     (P  : Circle -> type)
  -> (p  : P base)
  -> (eq : Id (P base) (subst loop p) p)
  -> (c : Circle) -> P c
```

Benefits

- Mathematical
  - Working up to equivalence is common mathematical practice, handled automatically by homotopy type theory
  - Intuitionistic type theory is probably the best direct formulation of ∞-groupoids known
- "Practical"
  - Functional extensionality is a useful proof principle for reasoning about programs
  - Equivalence-implies-identity aids in code reuse and abstraction
    - List A and $\Sigma_{n:\mathbb{N}}.\,\mathsf{Vec}\,A\,n$ are isomorphic implementations of lists, so any construction on one automatically functions for the other
    - Abstract types and views can be related by equivalence, allowing one to program and prove via the view, while a more efficient abstract type is used internally

Directed Type Theory

- Homotopy type theory generalizes from sets to ∞-groupoids
- We can also generalize from groupoids (symmetric) to categories (directed)
  - Instead of $\simeq_A$ with refl, trans, subst, **sym** ...
  - ... we have $\Longrightarrow_A$ with id, $\circ$, map

Directed Type Theory

Some Details

- Contexts must now track variances:

$$\frac{\Gamma \vdash \mathsf{ctx}}{\Gamma^{op} \vdash \mathsf{ctx}} \qquad \frac{\Gamma \vdash A \ \mathsf{type}}{\Gamma,\ x : A^{+} \vdash \mathsf{ctx}} \qquad \frac{\Gamma^{op} \vdash A \ \mathsf{type}}{\Gamma,\ x : A^{-} \vdash \mathsf{ctx}}$$

$$\frac{\Gamma,\ x : A^{-} \vdash B(x) \ \mathsf{type}}{\Gamma \vdash \Pi_{x:A}.B(x) \ \mathsf{type}} \qquad \frac{\Gamma, x : A^{-} \vdash M : B(x)}{\Gamma \vdash (\lambda x : A.\, M) : \Pi_{x:A}.B(x)}$$

- map acts in response to variance

$$\frac{\Gamma,\ x : A^{+} \vdash B(x) \ \mathsf{type} \qquad \Gamma \vdash \alpha : M \Longrightarrow_{A} N}{\Gamma \vdash \mathsf{map}_{x:A^{+}.B(x)}\, \alpha : B(M) \to B(N)}$$

Directed Type Theory

Benefits

- Directed types allow an even larger class of transformations to be automatically respected by a large number of constructions
  - Sets and functions
  - Contexts and variable renaming
  - Lambda terms and reduction
- Programming/proving with views and abstract types now needn't require an equivalence between view and implementation
- Higher dimensional directed type theory has the tools for talking about naturality within the language, and may be able to internally support 'free' theorems

Caveats

- There is still work to be done in these areas
  - The univalence axiom has only been postulated thus far; its computational behavior is an open question
    - Licata and Harper have shown canonicity for a 2-dimensional directed theory, but the approach is different
  - Proper hom types have yet to be worked out
    - Instead of Id A x y, Hom A x y
    - Composition of Hom A x y with Hom A y z has y in both covariant and contravariant positions
    - Directed type theory works around the issue for now

Further Reading

- The Homotopy Type Theory website

  homotopytypetheory.org

- Univalent Foundations (Voevdosky)

  math.ias.edu/~vladimir/Site3/Univalent_Foundations.html

- Directed Type Theory (Licata and Harper)

  www.cs.cmu.edu/~drl/pubs.html
  www.cs.cmu.edu/~rwh/papers.htm