Quotient Types by Normalization in Cedille

Andrew Marmaduke, Christopher Jenkins, Aaron Stump

6/12/2019

The University of Iowa

Mod 2

0, 1, 2, 3, 4, 5, 6, ...

0, 1, 2, 3, 4, 5, 6, ...

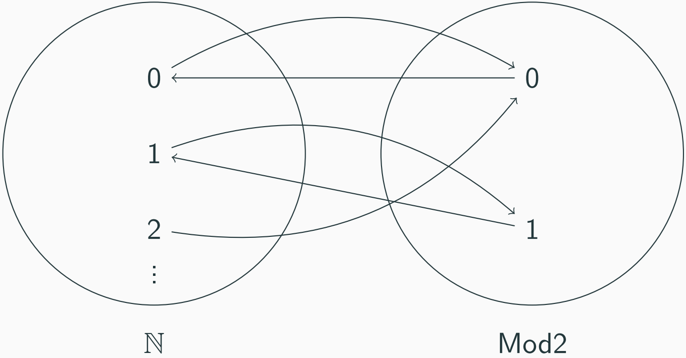
$$\forall n, m \in \mathbb{N}.\ n \sim m \iff n \bmod 2 = m \bmod 2$$

- We quotient $\mathbb{N}$ by an equivalence relation
- Not all equivalence relations have decidable choices for canonical elements

Definitions

A function f is a **normalization function** or **canonizer** if it is idempotent. It picks a canonical element of an equivalence class consistently.
A **canonical element** is the representative element in the quotient type of the equivalence class of elements in the carrier type.
A **quotient by normalization** is a type formed from the carrier type where the normalization function f is used to choose canonical elements.

Mod 2, Quotient by Normalization



$$\mathsf{Mod2} = \iota\ n : \mathbb{N}.\ \{\mathsf{mod}\ n\ 2 \simeq n\}$$

Background on Cedille

Introducing Cedille

CC

$\forall\ x : T.\ T'$	implicit products (Miquel)
$\iota\ x : T.\ T'$	dependent intersections (Kopylov)
$\{\ t \simeq t'\}$	untyped equality

- Small theory, formal syntax and semantics
- Core checker implemented in $<$ 1000loc Haskell
- Logically sound
- Turing complete(!)
- Supports inductive lambda-encodings

But if you are using intersections...

You must have an extrinsic (Curry-style) type theory.

- Unannotated terms of pure lambda calculus
- Assign multiple different types to same term (Intrinsic type theories usually have unique types.)
- Completely different from Coq, Agda
- Much less explored TT...

Equality type

Formation:

$$\frac{FV(t\ \ t') \subseteq dom(\Gamma)}{\Gamma \vdash \{t \simeq t'\} : \star}$$

Introduction and elimination:

$$\frac{FV(t') \subseteq dom(\Gamma)}{\Gamma \vdash t : \{t' \simeq t'\}} \quad \frac{\Gamma \vdash t' : \{t_1 \simeq t_2\} \qquad \Gamma \vdash t : [t_1/x]T}{\Gamma \vdash t : [t_2/x]T}$$

Direct computation rule:

$$\frac{\Gamma \vdash t : \{t_1 \simeq t_2\} \qquad \Gamma \vdash t_1 : T}{\Gamma \vdash t_2 : T}$$

Annotations:

$$\begin{aligned} |\beta\{t\}| &= |t| \\ |\rho\ t' \text{ - } t| &= |t| \end{aligned}$$

The Kleene trick

- Any term can be assigned a trivially true equality

$$\boxed{\mathrm{t}} : \{t' \simeq t'\}$$

- Restricted form of subset type, combined with *conversion*

$$\frac{\Gamma \vdash t : T' \qquad \Gamma \vdash T : \star \qquad T \cong T'}{\Gamma \vdash t : T}$$

Aside on rewriting

$$\boxed{\rho\ t\ \text{-}\ t'}$$

- Suppose $t : \{t_1 \simeq t_2\}$, and T is type for t'.
- Find each subterm of T convertible to t_1 and rewrite.
- Use ρ anywhere in a term

Dependent intersections

Formation:

$$\frac{\Gamma \vdash T : \star \qquad \Gamma, x : T \vdash T' : \star}{\Gamma \vdash \iota\, x : T.\ T'}$$

Introduction and elimination:

$$\frac{\Gamma \vdash t : T \qquad \Gamma \vdash t : [t/x]T'}{\Gamma \vdash t : \iota\, x : T.\ T'} \quad \frac{\Gamma \vdash t : \iota\, x : T.\ T'}{\Gamma \vdash t : T} \quad \frac{\Gamma \vdash t : \iota\, x : T.\ T'}{\Gamma \vdash t : [t/x]T'}$$

Annotations:

$$\begin{array}{rcl} |[t, t']| & = & |t| \\ |t.1| & = & |t| \\ |t.2| & = & |t| \end{array}$$

Dependent Intersection as Set Comprehension

$$\text{Mod2} = \iota\ n : \mathbb{N}.\ \{\text{mod}\ n\ 2 \simeq n\}$$

$$\text{Mod2} = \{n \in \mathbb{N} \mid \text{mod}\ n\ 2 = n\}$$

Case Studies and Examples

Mod k

$$\text{Mod2} := \iota\ n : \mathbb{N}.\ \{\text{mod } n\ 2 \simeq n\}$$

Idempotency is needed only in the relevant argument:

$$\text{Modk} := \lambda\ k : \mathbb{N}.\ \iota\ n : \mathbb{N}.\ \{\text{mod } n\ k \simeq n\}$$

Even and Odd $\mathbb{N}$s

ecanon : $\mathbb{N} \to \mathbb{N}$

ecanon $0 = 0$

ecanon $1 = 0$

ecanon n $+ 2 =$ (ecanon n) $+ 2$

ocanon : $\mathbb{N} \to \mathbb{N}$

ocanon n $=$ (ecanon n) $+ 1$

$$\text{Even} := \iota\ n : \mathbb{N}.\ \{\text{ecanon } n \simeq n\}$$
$$\text{Odd} := \iota\ n : \mathbb{N}.\ \{\text{ocanon } n \simeq n\}$$

Aside: A difference from equivalence relations

- With Mod2 we pick 0 and 1 as canonical representatives
- We could also formulate an isomorphic type Mod2' with canonical representatives 2 and 3
- This is the same relationship between the types Even and Odd
- The equivalence relation for Even is the same as the one for Odd

0, 1, 2, 3, 4, 5, ...
0, 1, 2, 3, 4, 5, ...

Aside: Subtypes in Cedille

- Cedille can witness a subtyping relation between two types A and B
- If $f : A \to B$ exists such that $\forall a : A.\ \{f\ a \simeq (\lambda\ x.\ x)\ a\}$
- then we have a *cast* from A to B such that the type of any element in A can be changed to B
- In Cedille we write this type Cast A B

$$\text{Cast Mod2 } \mathbb{N}$$
$$\text{Cast Modk } \mathbb{N}$$
$$\text{Cast Even } \mathbb{N}$$
$$\text{Cast Odd } \mathbb{N}$$

Even Ordinals

data Ord =
| zero : Ord
| succ : Ord $\to$ Ord
| limit : ($\mathbb{N} \to$ Ord) $\to$ Ord

ecanon : Ord $\to$ Ord
ecanon 0 = 0
ecanon 1 = 0
ecanon n + 2 = (ecanon n) + 2
ecanon limit f = limit f

Binary Natural Numbers

data PreBNat =
| bzero : PreBNat
| b2 : PreBNat
| b2p : PreBNat

Define a constructor variant that preserves canonicity:
b2' : PreBNat $\rightarrow$ PreBNat

bnat : PreBNat $\rightarrow$ PreBNat
bnat bzero = bzero
bnat (b2 p) = b2' (bnat p)
bnat (b2p p) = b2p (bnat p)

Binary Natural Numbers

$\mathsf{BNat} = \iota\ b : \mathsf{PreBNat}.\ \{\ \mathsf{bnat}\ b \simeq b\ \}$

zero : BNat

twice : BNat $\to$ BNat

twice-plus-1 : BNat $\to$ BNat

Binary Natural Numbers

Need destructors to prove canonicity about subdata:

destruct-b2 :

$$\Pi\ p : \mathrm{PreBNat}.\ \{\ \mathrm{bnat}\ (\mathrm{b2}\ p) \simeq \mathrm{b2}\ p\ \} \Rightarrow \{\ \mathrm{bnat}\ p \simeq p\ \}$$

destruct-b2p :

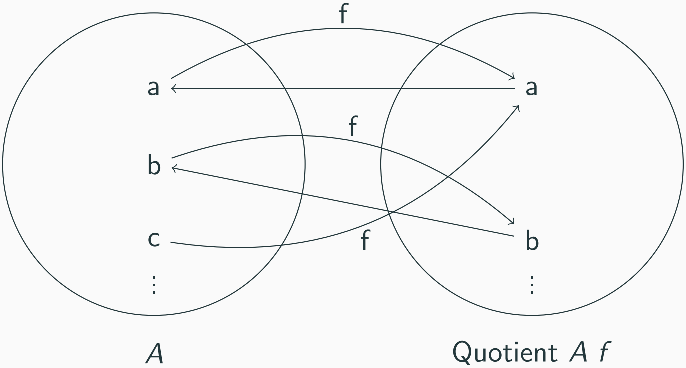
$$\Pi\ p : \mathrm{PreBNat}.\ \{\ \mathrm{bnat}\ (\mathrm{b2p}\ p) \simeq \mathrm{b2p}\ p\ \} \Rightarrow \{\ \mathrm{bnat}\ p \simeq\ \ p\ \}$$

Binary Natural Numbers

```
bnat-succ : BNat → BNat
bnat-succ bzero eq = twice-plus-1 zero
bnat-succ (b2 p) eq = twice-plus-1 [p, (destruct-b2 p eq)]
bnat-succ (b2p p) eq = twice (bnat-succ p (destruct-b2p p eq))
```

Generic Definition of Quotient Types and Lifting

Quotient by Normalization



$$\text{Quotient } A\ f := \iota\ a : A.\ \{f\ a \simeq a\}$$

Generic Definitions in Cedille

$$\begin{aligned} \mathsf{IdemFn} &= \lambda\ A.\ \iota\ f : A \to A.\ \Pi\ a : A.\ \{f\ (f\ a) \simeq f\ a\} \\ \mathsf{Quotient} &= \lambda\ A.\ \lambda\ f : \mathsf{IdemFn}\ A.\ \iota\ a : A.\ \{f\ a \simeq a\} \end{aligned}$$

canonize : A $\to$ Quotient A f

canonize $a = [f.1\ a,$ $\qquad f\ (f\ a) \simeq f\ a$

$\quad \rho\ (f.2\ a)$ $\qquad f\ a \simeq f\ a$

$\quad -\beta\{f.1\ a\}]$ $\qquad$ Kleene Trick

Cast for the Generic Definition

relax : Quotient A f $\to$ A
relax $q = q.1$

- We take the first projection of q to recover the element in A
- but, because of erasure $|q| = |q.1|$ thus $|\text{relax}| = \lambda\ x.\ x$
- Thus, for any quotient of A we have Cast (Quotient A f) A

Simple Lifting for Quotients

let Q = Quotient A f

lift_by_canonize : $(A \to A) \to$ Q $\to$ Q
lift_by_canonize f q = canonize (f (relax q))

Simple Lifting for Quotients

let Q = Quotient A f

Compatible $= \lambda\ op : A \to A.\ \forall\ a : A.\ \{f\ (op\ a) \simeq op\ (f\ a)\}$

lift : $\Pi\ op : A \to A.\ Compatible\ op \Rightarrow$ Q $\to$ Q

lift op -c q = [op q.1,	$f\ (op\ q) \simeq op\ q$
ρ (c -q.1)	$op\ (f\ q) \simeq op\ q$
$-\rho$ (q.2)	$op\ q \simeq op\ q$
$-\beta$\{op q\}]	Kleene Trick

Simple Lifting for Quotients

let Q = Quotient A f

Compatible $= \lambda\ op : A \to A.\ \forall\ a : A.\ \{f\ (op\ a) \simeq op\ (f\ \ a)\}$
lift : $\Pi\ op : A \to A.\ Compatible\ op \Rightarrow \mathrm{Q} \to \mathrm{Q}$

liftβ : $\Pi\ op : A \to A.\ \Pi\ a : A.\ \{\text{lift}\ op\ a \simeq op\ a\}$
liftβ op a $= \beta$

Lifting for Simple Types by Canonizer

simple types (over a type variable x) A ::= $x \mid T \mid A \to A'$

data IsSimple F =
| base : IsSimple (λ x. x)
| any : $\forall T$. IsSimple (λ x. T)
| arrow : $\forall A$ B. IsSimple $A \to$ IsSimple B
$\to$ IsSimple (λ x. A $x \to B$ x)

lift_simple : $\forall F$. IsSimple $F \to$ Pair (F $A \to F$ Q) (F $Q \to F$ A)

Conclusion

Related Work

- Quotients in NuPRL
- Definable Quotients of Li in Agda (which are equivalent to the formulation presented)
- Quotients of Cohen in Coq
- Quotient Theories in non-constructive theories like HOL Light
- Quotienting in Homotopy Type Theory

Future Work

- Expand compatibility lifting to simply typed functions
- Explore possibilities for undefinable quotients

Questions?