Two presentations of equality

Li Nuo

December 10, 2010

1 Background

We can define equality in two ways: either as an inductively defined relation or as a parameterized inductive predicate:

As a binary relation **data** Id (A : Set) : A → A → Set **where**
refl : (a : A) → Id A a a

This one was first proposed by Per Martin-Löf as intentional equality[?]. There is one instance for each element. We can treat equality relation as $(a, b) \bowtie$ Id A. We can describe it in another way: it is a partition of the set A x A.

Thorsten: As proposed Martin-Löf

As a predicate **data** Id' (A : Set) (a : A) : A → Set **where**
refl : Id' A a a

This one is just what used in the Agda standard library. It is only possible for us to define this one with dependent types because the type depends on a value. We can treat Id A a it as a predicate of whether certain element of A is the same as a. It also represents the singleton set with only one element refl. This one was proposed by Christine Paulin-Mohring. **Thorsten:** proposed by Christine Paulin-Mohring

For each of them, we have a corresponding elimination rule, defined as

As a binary relation J : (A : Set) (P : (a b : A) → Id A a b → Set)
→ (m : (a : A) → P a a (refl a))

$\rightarrow$ (a b : A) (p : Id A a b) $\rightarrow$ P a b p
J A P m .b b (refl .b) = m b

The P and m are both indexed by different a. P is actually a ternary relation. **Thorsten:** Use doublebar for all the inlined Agda code!

m can be seen as an introduction rule for P. For all a, (a, a, refl a) is inhabited in P. And the result is a more general property, For all a b, (a, b, x : Id A a b) is inhabited in P.

J actually maps

$$\forall \text{ (a : A)} \rightarrow \text{P a a (refl a)} \Rightarrow \forall \text{ (a b : A) (p : Id A a b)} \rightarrow \text{P a b p}$$

.

As a predicate J' : (A : Set) (a : A)
$\rightarrow$ (P : (b : A) $\rightarrow$ Id' A a b $\rightarrow$ Set)
$\rightarrow$ (m : P a refl)
$\rightarrow$ (b : A) (p : Id' A a b) $\rightarrow$ P b p
J' A .b P m b refl = m

The P and m are now restricted by the same a as the the identity predicate. P and m here are actually special cases of the P (P [a]) and m (m [a]) in J. 'a' can be regarded as a constant in the discourse.

J' actually maps

$$\text{P a refl} \Rightarrow \text{(b : A) (p : Id' A a b)} \rightarrow \text{P b p}$$

. m! can be seen as the only object in P and the result is used to unify elements equal to a (a constant) to get the unique object.

2 The Problem

Now the problem is: how to implement J using only J' (also we use the equality Id') and vice versa? We will still use corresponding equality for each elimination rule, otherwise it cannot eliminate the identity.

3 Solution

From J' to J is quite simple. **Thorsten:** Which is the first direction? When we eliminate the predicate identity, we only need to create the special cases of P and m for J'.

$$
\begin{aligned}
&\mathsf{JId'} : (\mathsf{A} : \mathsf{Set})\ (\mathsf{P} : (\mathsf{a}\ \mathsf{b} : \mathsf{A}) \to \mathsf{Id'}\ \mathsf{A}\ \mathsf{a}\ \mathsf{b} \to \mathsf{Set}) \\
&\quad \to ((\mathsf{a} : \mathsf{A}) \to \mathsf{P}\ \mathsf{a}\ \mathsf{a}\ \mathsf{refl}) \\
&\quad \to (\mathsf{a}\ \mathsf{b} : \mathsf{A})\ (\mathsf{p} : \mathsf{Id'}\ \mathsf{A}\ \mathsf{a}\ \mathsf{b}) \to \mathsf{P}\ \mathsf{a}\ \mathsf{b}\ \mathsf{p} \\
&\mathsf{JId'}\ \mathsf{A}\ \mathsf{P}\ \mathsf{m}\ \mathsf{a} = \mathsf{J'}\ \mathsf{A}\ \mathsf{a}\ (\mathsf{P}\ \mathsf{a})\ (\mathsf{m}\ \mathsf{a})
\end{aligned}
$$

Thorsten: Check that $\mathsf{JId'}\ \mathsf{A}\ \mathsf{P}\ \mathsf{m}\ .\mathsf{b}\ \mathsf{b}\ (\mathsf{refl}\ .\mathsf{b}) = \mathsf{m}\ \mathsf{b}$ holds definitionally.

The other direction is more tricky. We first define subst from J

$$
\begin{aligned}
&\mathsf{subst} : (\mathsf{A} : \mathsf{Set})\ (\mathsf{a}\ \mathsf{b} : \mathsf{A})\ (\mathsf{p} : \mathsf{Id}\ \mathsf{A}\ \mathsf{a}\ \mathsf{b}) \\
&\quad (\mathsf{B} : \mathsf{A} \to \mathsf{Set}) \to \mathsf{B}\ \mathsf{a} \to \mathsf{B}\ \mathsf{b} \\
&\mathsf{subst}\ \mathsf{A}\ \mathsf{a}\ \mathsf{b}\ \mathsf{p}\ \mathsf{B} = \mathsf{J}\ \mathsf{A}\ (\lambda\ \mathsf{a'}\ \mathsf{b'}\ \_ \to \mathsf{B}\ \mathsf{a'} \to \mathsf{B}\ \mathsf{b'})\ (\lambda\ \_ \to \mathsf{id})\ \mathsf{a}\ \mathsf{b}\ \mathsf{p}
\end{aligned}
$$

Then to prove $\mathsf{J'}$ from J and Id,

$$
\begin{aligned}
&\mathsf{Q} : (\mathsf{A} : \mathsf{Set})\ (\mathsf{a} : \mathsf{A}) \to \mathsf{Set} \\
&\mathsf{Q}\ \mathsf{A}\ \mathsf{a} = \Sigma\ \mathsf{A}\ (\lambda\ \mathsf{b} \to \mathsf{Id}\ \mathsf{A}\ \mathsf{a}\ \mathsf{b}) \\
&\mathsf{J'Id} : (\mathsf{A} : \mathsf{Set})\ (\mathsf{a} : \mathsf{A}) \to (\mathsf{P} : (\mathsf{b} : \mathsf{A}) \to \mathsf{Id}\ \mathsf{A}\ \mathsf{a}\ \mathsf{b} \to \mathsf{Set}) \\
&\quad \to \mathsf{P}\ \mathsf{a}\ (\mathsf{refl}\ \mathsf{a}) \\
&\quad \to (\mathsf{b} : \mathsf{A})\ (\mathsf{p} : \mathsf{Id}\ \mathsf{A}\ \mathsf{a}\ \mathsf{b}) \to \mathsf{P}\ \mathsf{b}\ \mathsf{p} \\
&\mathsf{J'Id}\ \mathsf{A}\ \mathsf{a}\ \mathsf{P}\ \mathsf{m}\ \mathsf{b}\ \mathsf{p} = \mathsf{subst}\ (\mathsf{Q}\ \mathsf{A}\ \mathsf{a})\ (\mathsf{a}, \mathsf{refl}\ \mathsf{a})\ (\mathsf{b}, \mathsf{p}) \\
&\quad (\mathsf{J}\ \mathsf{A}\ (\lambda\ \mathsf{a'}\ \mathsf{b'}\ \mathsf{x} \to \mathsf{Id}\ (\mathsf{Q}\ \mathsf{A}\ \mathsf{a'})\ (\mathsf{a'}, \mathsf{refl}\ \mathsf{a'})\ (\mathsf{b'}, \mathsf{x})) \\
&\quad (\lambda\ \mathsf{a'} \to \mathsf{refl}\ (\mathsf{a'}, (\mathsf{refl}\ \mathsf{a'})))\ \mathsf{a}\ \mathsf{b}\ \mathsf{p})\ (\mathsf{uncurry}\ \mathsf{P})\ \mathsf{m}
\end{aligned}
$$

We can not just use J to eliminate the identity because J requires more general P and m. We need to formalise the result $\mathsf{P}\ \mathsf{b}\ \mathsf{p}$ from $\mathsf{P}\ \mathsf{a}\ (\mathsf{refl}\ \mathsf{a})$. We cannot substitute a or $\mathsf{refl}\ \mathsf{a}$ separately because the second argument is dependent on the first argument. So when we substitute we should reveal the dependent relation. **Thorsten:** Or : Instead we are going to substitute them simultanously using a dependent product.

We could use dependent productr to do this work. In this way, we can substitute them simultaneously. The problem now becomes substitute in

$$\mathsf{P}\ ((\lambda\ \mathsf{a} : \mathsf{A}\ \mathsf{p} : \mathsf{Id}\ \mathsf{A}\ \mathsf{a}\ \mathsf{b} \to (\mathsf{a}, \mathsf{p}))\ \mathsf{a}\ (\mathsf{refl}\ \mathsf{a}))$$

to

$$\mathsf{P}\ ((\lambda\ \mathsf{a} : \mathsf{A}\ \mathsf{p} : \mathsf{Id}\ \mathsf{A}\ \mathsf{a}\ \mathsf{b} \to (\mathsf{a}, \mathsf{p}))\ \mathsf{b}\ \mathsf{p})$$

From J, we have $\mathsf{Id}\ (\mathsf{Q}\ \mathsf{a})\ (\mathsf{a}, \mathsf{refl}\ \mathsf{a})\ (\mathsf{b}, \mathsf{x} : \mathsf{Id}\ \mathsf{a}\ \mathsf{b})$ so that we can prove $\mathsf{P'}\ (\mathsf{b}, \mathsf{p})$ from $\mathsf{P'}\ (\mathsf{a}, \mathsf{refl}\ \mathsf{a})$ using subst. Because $\mathsf{P'}\ (\mathsf{b}, \mathsf{p})$ is namely $\mathsf{P}\ \mathsf{b}\ \mathsf{p}$, we have proved.

Thorsten: Check that $\mathsf{J'Id}\ \mathsf{A}\ \mathsf{b}\ \mathsf{P}\ \mathsf{m}\ \mathsf{b}\ \mathsf{refl} = \mathsf{m}$ holds definitionally!

Thorsten: Add some references. For Id refer to the Nordstroem et al book, Thomas Streicher habil, Palmgren

Thorsten: Compare with the construction of the isomorphism.