Having read section , with the tools developed in this section a natural question to ask is: "can we generalize the notion of an n-type to sHoTT?" This question can be interpreted in many ways. There is a lot of freedom that is introduced, as we have free pick of which paths to change, and whether we want to change them to an isomorphism or morphism. For some types, all of these notions coincide. For example, the notion of set is one where parallel paths between points are equal. We can generalize this notion and instead ask for a type where parallel morphisms between points are equal. That is, we can ask for a type to "be a relation".

do a section about n-types

Definition 0.1. A type A is a relation if for all $x, y : A$ and $f, g : \hom_A(x, y)$, we have $f = g$.

As a first example of this, we can show that Unit is a relation.

Lemma 0.1. *Unit is a relation.*

Proof. It suffices to show $\hom_{\mathrm{Unit}}(\star, \star)$ is contractible. This is immediate as

$$\left\langle \Delta^1 \to \mathrm{Unit} \middle| {\partial\Delta^1 \atop [\star,\star]} \right\rangle \tag{1}$$

has contractible fibers, thus the entire extension type is contractible by relative function extensionality. □

Natural numbers are also a relation

Lemma 0.2. $\mathbb{N}$ *is a relation.*

Proof. Let $\mathrm{code} : \mathbb{N} \to \mathbb{N} \to \mathcal{U}$ be the type family defined recursively, with defining equations

$$\begin{aligned}
&\mathrm{code}\, 0\, 0 :\equiv \mathrm{Unit}\\
&\mathrm{code}\, 0\, S(n) :\equiv \mathrm{Void}\\
&\mathrm{code}\, S(n)\, 0 :\equiv \mathrm{Void}\\
&\mathrm{code}\, S(n)\, S(m) :\equiv \mathrm{code}\, n\, m
\end{aligned}$$

Before we define encode and decode, we define a map $\mathrm{pred} : \mathbb{N} \to \mathbb{N}$ as

$$\begin{aligned}
&\mathrm{pred}\, 0 :\equiv 0\\
&\mathrm{pred}\, S(n) :\equiv n
\end{aligned}$$

Moreover, there is a map $\phi : \prod_{(n:\mathbb{N})} \prod_{(m:\mathbb{N})} \hom_{\mathbb{N}}(S(n), S(m)) \to \hom_{\mathbb{N}}(n, m)$ defined as

$$\begin{aligned}
&\phi\, 0\, 0\, _ :\equiv \mathrm{idhom}_0\\
&\phi\, S(n)\, S(m)\, f :\equiv \lambda i.\, \mathrm{pred}(f(i))
\end{aligned}$$

Now, we define encode : $\prod_{(n:\mathbb{N})} \prod_{(m:\mathbb{N})} \hom_{\mathbb{N}}(n,m) \to \operatorname{code} n\, m$ with the defining equations

$$\begin{aligned}&\operatorname{encode} 0\, 0\, \_ :\equiv \star\\ &\operatorname{encode} S(n)\, S(m)\, f :\equiv \operatorname{encode} n\, m\, \phi(f)\end{aligned}$$

and we define decode : $\prod_{(n:\mathbb{N})} \prod_{(m:\mathbb{N})} \operatorname{code} n\, m \to \hom_{\mathbb{N}}(n,m)$ as

$$\begin{aligned}&\operatorname{decode} 0\, 0\, \_ :\equiv \operatorname{idhom}_0\\ &\operatorname{decode} S(n)\, S(m)\, u :\equiv \lambda i.\, S(\operatorname{decode} n\, m\, u)\end{aligned}$$

Before we can show that these maps form a pair in a bi invertible equivalence, we define a map $r : \prod_{(n:\mathbb{N})} \prod_{(f:\hom_{\mathbb{N}}(n,n))} f = \operatorname{idhom}_n$ Finally, we can show that these functions form a bi invertible equivalence by defining a map

$$\eta : \prod_{(n:\mathbb{N})} \prod_{(m:\mathbb{N})} \prod_{(f:\hom_{\mathbb{N}}(n,m))} \operatorname{decode} n\, m\, (\operatorname{encode} n\, m f) = f$$

as

$$\eta\, 0\, 0\, \_$$

□