Numbers, quotients, and definability of reals

Li Nuo

November 13, 2012

Since Agda is known as a proof assistant, the library of numbers is crucial. In such kind of proof assistants which are based on Martin-Löf type theory , we need to construct the type of numbers and the usual properties of them should be verifiable rather than axiomatic.

There are different ways of defining numbers, even though they are mathematically equivalent, they are technically different, which means the proving of theorems about the numbers varies. For example, integers can be defined by exploiting the isomorphism between $\mathbb{Z}$ and $\mathbb{N} + \mathbb{N}$:

```
data ℤ : Set where
  -[1+_] : (n : ℕ) → ℤ   - -[1+ n ] stands for - (1 + n).
  +_     : (n : ℕ) → ℤ   - + n stands for n.
```

And this is exactly the definition in Agda standard library version 0.6. This definition is easy to use since it has two cases and for each integer you have one unique representation. However intuitively we lose the "special position" held by 0. Of course we can define three cases definition with distinct 0 constructor but too many cases are not ideal for proving.

Alternatively we have another isomorphism between $\mathbb{Z}$ and $\mathbb{N} \times \mathbb{N}/\sim$, namely constructing the set of integers from quotienting the set of $\mathbb{N} \times \mathbb{N}$ by the following equivalence relation :

$$(n_1, n_2) \sim (n_3, n_4) \iff n_1 + n_4 = n_3 + n_2 \tag{1}$$

This implementation better exploits the relationship between the set of natural numbers and the set of integers, because any integer is a result of subtracting two natural numbers which means we uniformly represent all integers, and the laws for basic operations are simpler to lifted from the ones for natural numbers.

This kind of relationship between setoids and sets can be generalized as quotient. If we have a setoid, we can define a corresponding quotient type as $Q : Set$ and a function $[\cdot] : A \to Q$ such that we have

1. soundness properties

$$sound : \forall ab : A \to a \sim b \to [a] \equiv [b]$$

2. an eliminator of quotient types

$$\begin{gathered} qelim : B : Q \to Set \\ \to (f : (a : A) \to B[a]) \\ \to ((ab : A) \to (p : a \sim b) \\ \to substB(soundp)(fa) \equiv fb) \\ \to (q : Q) \to Bq \end{gathered}$$

3. the computational rule for the eliminator

$$qelim - B : \forall Bafq \to qelimBfq[a] \equiv fa$$