

ldf.1 Truth Values and Relations

lam:ldf:tvr:
sec We can encode truth values in the pure lambda calculus as follows:

$$\begin{aligned}\text{true} &\equiv \lambda x. \lambda y. x \\ \text{false} &\equiv \lambda x. \lambda y. y\end{aligned}$$

Truth values are represented as *selectors*, i.e., functions that accept two arguments and returning one of them. The truth value `true` selects its first argument, and `false` its second. For example, `true MN` always reduces to `M`, while `false MN` always reduces to `N`.

Definition ldf.1. We call a relation $R \subseteq \mathbb{N}^n$ **λ -definable** if there is a term R such that

$$R \bar{n}_1 \dots \bar{n}_k \xrightarrow{\beta} \text{true}$$

whenever $R(n_1, \dots, n_k)$ and

$$R \bar{n}_1 \dots \bar{n}_k \xrightarrow{\beta} \text{false}$$

otherwise.

For instance, the relation `IsZero = {0}` which holds of 0 and 0 only, is **λ -definable**.

$$\text{IsZero} \equiv \lambda n. n(\lambda x. \text{false}) \text{true}$$

How does it work? As Church numerals are defined as iterators (functions which apply their first argument n times to the second), we set the initial value to be `true`, and for every step of iteration, we return `false` whatever the result of last iteration is. This step will be applied to the initial value n times, and the result will be `true` if and only if the step is not applied at all, i.e., when $n = 0$.

On the basis of this representation of truth values, we can further define some truth functions. Here are two, the representations of negation and conjunction:

$$\begin{aligned}\text{Not} &\equiv \lambda x. x \text{false} \text{true} \\ \text{And} &\equiv \lambda x. \lambda y. xy \text{false}\end{aligned}$$

The function “Not” accepts one argument, and returns `true` if the argument is `false`, and `false` if the argument is `true`. The function “And” accepts two truth values as arguments, and should return `true` iff both arguments are `true`. Since we represent truth values as selectors (described above), and passes y and `false` to x , which will evaluate to y if x is `true`, or `false` if x is `false`, which is just what is desired.

Note that we assume here that only truth values are used as arguments to `And`. If it is passed other terms, the result (i.e., the normal form, if it exists) may well not be a truth value.

Problem 1df.1. Define the functions Or and Xor representing the truth functions of inclusive and exclusive disjunction using the encoding of truth values as λ -terms.

Photo Credits

Bibliography