Normal Paths as Function Sub-Types

by Sven Nilsen, 2017

Here I represent a new insight that unites logical equivalence of paths with sub-types of functions defined using paths. This makes function sub-types similar to variable sub-types. This new notation has strengths and weaknesses different from both paths and their equivalent equations.

In path semantics one can define sub-types of variables by using functions. Such sub-types makes sense only when the existential path returns `true`:

$$a : [g]\ b$$
$$b : [\exists g]\ \mathrm{true}$$

$$g : A \rightarrow B$$
$$\exists g : B \rightarrow \mathrm{bool}$$

The new idea is that sub-types of functions can be defined using paths:

$$f : [[g]]\ h$$

$$f : A \times A \rightarrow A$$
$$g : A \rightarrow B$$
$$h : B \times B \rightarrow B$$

This is the same as writing:

$$f[g] \Leftrightarrow h$$
$$g(f(a, b)) = h(g(a, b))$$

In asymmetric notation:

$$f : [[g_{i \rightarrow n}]]\ h$$

Just like for normal sub-types, there is an existential path that returns `true` if the path set is non-empty:

$$h : [\exists[g]]\ \mathrm{true}$$

$$\exists[g] \Leftrightarrow \exists[g \times g \rightarrow g]$$

The `∃[g]` notation is used because the brackets makes it easy to use path semantical notation for symmetric and asymmetric paths. `∃g` would lead to confusion since it is used for existential path of `g`.

There is still motivation to use the notation of logical equivalence of functions in proofs. One does not need to write double brackets `f[g] <=> h` and path sets `f[g] <=> {h_0, h_1}` are useful for describing partial functions in terms of function spaces.