Material Implications From Existential Paths

by Sven Nilsen, 2019

In this paper I show a method to derive material implications from existential paths.

The major result of this paper is the following law:

$$\exists f\{g\} \iff g \qquad \text{<~=>} \qquad g \Rightarrow h$$

$$f : h \to T$$
$$h : T \to bool$$
$$g : T \to bool$$

Notation:

`<=>`	Logical equivalence (one expression can be replaced by another)
`=>`	Material implication (extended with HOOO for boolean functions)
`<~=>`	Equivalence (everything provable in one system is provable in the other system)
`f{g}`	`f` constrained by `g` (the domain of `f` is constrained by `g`)
`∃f`	The existential path of `f` (codomain)
`∃f{g}`	The existential path of `f` constrained by `g` (`∃f{g} <=> ∃(f{g})`)
`→`	Function type (`A → B` means taking type `A` and returning type `B`)
`T`	A generic type
`bool`	Boolean (`true` or `false`)

For every output of a function, there exists an input:

$$\forall\ y\ \{\ (\exists f)(y) \iff \exists\ x\ \{\ f(x) == y\ \}\ \}$$

This is because the existential path means that there exist some input such that `f` returns the value:

$$\exists f \qquad \iff \qquad \backslash(y) = \exists\ x\ \{\ f(x) == y\ \}$$

Here, the `\(...) = ...` notation means a lambda expression.

So, imagine that if `∃f <=> g`, then there must exist some input for every member of `g`.
However, since `∃f{g} <=> g`, each input must also be a member of `g`.
The set of the input is the same as the set of the output.
Two sets that are the same have the same number of members.
When both the input and the output have the same number of members, the function can not map any two members to the same output. It must map each input to some unique member of the output.
This means that `f{g}` maps every member of `g` to some unique member of `g`.
Because of this, the function `f{g}` is invertible.

However, since `f` has the type:

$$f : h \to T$$

It means that the input of `f{g}` is also implicitly constrained by `h`:

$$f\{g\} \quad \Leftarrow\!\Rightarrow \quad f\{g \wedge h\}$$

Yet, each member of the input must map to some unique member of the output.
The set of `h` can not be smaller than the set of `g`, because then there would not exist some input of `f{g ∧ h}` for every member of `g`.

$$|h| >= |g|$$

The set of `h` covers the set of `g`.
This means that every input of `g` which returns `true` also returns `true` for `h`:

$$x : [g]\ \text{true} \qquad => \qquad x : [h]\ \text{true}$$

This can be written:

$$x : g \qquad => \qquad x : h$$

Or simply:

$$g => h$$

This gives the following law:

$$\exists f\{g\} <=> g \qquad <\sim=> \qquad g => h$$