Paths as Making Predictions

by Sven Nilsen, 2018

A path in path semantics (also called a “normal path”) is a function that predicts something about another function. Paths connect function spaces together into what is called “path semantical space”.

For example, when you add two even numbers, you get an even number. This is because:

```
add[even] <=> eq

add[even](true, true)
eq(true, true)
true
```

In equational form:

```
even(add(a, b)) = eq(even(a), even(b))
```

Paths are interesting because they are often cheaper to compute than the function they predict.

For example, when you concatenate two lists, the lengths are added to the length of the new list:

```
concat[len] <=> add
```

If you concatenate two lists of even lengths, you get a new list of even lengths, because:

```
concat[len][even]
add[even]
eq
```

The cost of concatenating lists is greater than the cost of adding two numbers. The cost of adding two numbers is greater than the cost of testing two boolean values for equality:

```
Cost(concat) > Cost(add) > Cost(eq)
```

This means that one can make predictions cheaper than performing the full computations, but only when the path exists and only within the types supported by the path. It means there are fundamental restrictions on which predictions can be made from available information.