Argument Relevant Functions

by Sven Nilsen, 2019

In this paper I formalize argument relevant functions in path semantics.

An argument relevant function is defined as following using notation for path semantics[1]:

$$\text{argument\_relevant}_T(f : \backslash T) = \neg \exists \, i \, \{ \, \exists? f[\text{id\_unit}_i \rightarrow \text{id}] \, \}$$

The type `\T` means that `f` is a function of zero or more arguments returning type `T`[2].
The function `id_unit_i` constructs a function tuple erasing a single argument `i`[3].
The `∃?` operator means the existence of a normal path[4].

The output of an argument relevant function depends on all arguments,
such that no single argument can be erased while producing the same output.

Most standard functions used in programming are argument relevant.
For example, `and`, `or`, `add` etc.

References:

[1]   "Path Semantics"
Sven Nilsen, 2016-2019
https://github.com/advancedresearch/path_semantics/blob/master/papers-wip/path-semantics.pdf

[2]   "Higher Order Operator Overloading and Notation for Parameters"
Sven Nilsen, 2019
https://github.com/advancedresearch/path_semantics/blob/master/papers-wip/higher-order-operator-overloading-and-notation-for-parameters.pdf

[3]   "The Id Unit Function"
Sven Nilsen, 2019
https://github.com/advancedresearch/path_semantics/blob/master/papers-wip/the-id-unit-function.pdf

[4]   "Existence of Normal Paths"
Sven Nilsen, 2019
https://github.com/advancedresearch/path_semantics/blob/master/papers-wip/existence-of-normal-paths.pdf