Mini Toolkit in Dyon for Boolean Path Semantics

by Sven Nilsen, 2018

Boolean Path Semantics is path semantics restricted to functions of type `bool$^n$ → bool`. In this paper I represent a mini toolkit in Dyon for doing Boolean Path Semantics. This toolkit fits on a single page.

```
// Functions of type `bool -> bool -> bool`.
and() = \(a) = \(b) = (grab a) && b
or() = \(a) = \(b) = (grab a) || b
eq() = \(a) = \(b) = (grab a) == b
xor() = \(a) = \(b) = (grab a)^b
imply() = \(a) = \(b) = (grab !a) || b
exc() = \(a) = \(b) = (grab a) && !b
fst() = \(a) = \(_) = grab a
snd() = \(_) = \(b) = clone(b)
true_2() = \(_) = \(_) = true
false_2() = \(_) = \(_) = false
nand() = \(a) = \(b) = !((grab a) && b)
nor() = \(a) = \(b) = !((grab a) || b)

// Functions of type `bool -> bool`.
false_1() = \(_) = false
not() = \(a) = !a
id() = \(a) = clone(a)
true_1() = \(_) = true

// Returns a random boolean.
fn bool_0() -> any {return random() < 0.5}

// Generates a random boolean function of type `bool^n -> bool`.
bool_n(n) =
    if n <= 0 {bool_0()}
    else {\(x: bool) = if x {grab bool_n(n-1)} else {grab bool_n(n-1)}}

// Existential path `∃f{g}`.
ex_bool_n(f, g) =
    if typeof(f) == "boolean" {
        \(x: bool) = (grab g) && (x == grab f)
    } else {
        \(x: bool) = {
            fa := grab ex_bool_n(\f(false), \g(false))
            tr := grab ex_bool_n(\f(true), \g(true))
            \fa(x) || \tr(x)
        }
    }

// Compose `f ∘ g`.
comp(f, g) = \(a) = \(b) = {
    g := grab '2 g
    f := grab '2 f
    g2 := \g(grab a)
    \f(\g2(b))
}

// Combine sub-types `(f, g) : bool -> bool -> bool`.
// Where `f, g : bool -> bool`
tup(f, g) = \(a) = \(b) = {
    f := grab '2 f
    g := grab '2 g
    \f(grab a) && \g(b)
}
```