## Typify types

## Oleg Grenrus

August 1, 2014

Parser and tools for type signatures of typify. Essentially the language is dependent type theory, with omitted lambda constructor.

## 1 Formal syntax definition

level	name		associativity	example
8	optional	?	postfix	a?
7	application	a b	left associative	$a b c \equiv (a b) c$
6	conjunction	$\wedge$	associative	$(a \wedge b) \wedge c \equiv a \wedge (b \wedge c)$
5	disjunction	$\vee$	associative	$(a \lor b) \lor c \equiv a \lor (b \lor c)$
4	product	×	associative	$(a \times b) \times c \equiv a \times (b \times c)$
3	ellipsis		postfix	$a\dots$
2	type	:	right associative	x:a
1	function	$\rightarrow$	right associative	$a \to b \to c \equiv a \to (b \to c)$
0	semicolon	;	associative	

Table 1: Operator precedence

name		code variants
any	Т	T *
nothing	$\perp$	⊥ _ _
optional	?	?
conjunction	$\wedge$	$\wedge$
disjunction	$\vee$	∨ &
ellipsis		(either three dots, or unicode ellipsis)
type	:	:
product	×	× ,
function	$\rightarrow$	$\rightarrow$ ->
semicolon	;	;

Table 2: Special character code representations

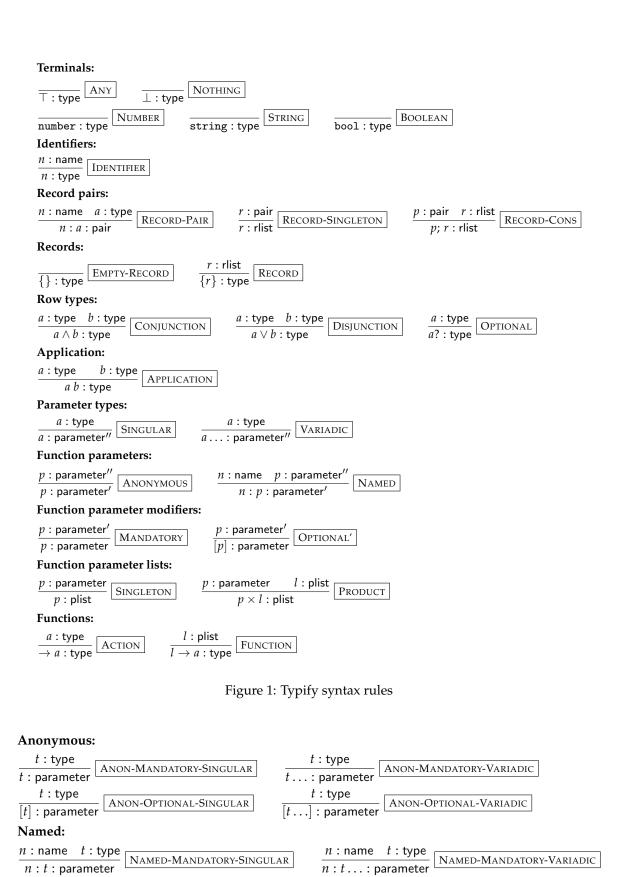


Figure 2: Alternative parameter rules

 $n: \mathsf{name} \quad t: \mathsf{type}$ 

 $n:t: \mathsf{Lype}$   $n:t: \mathsf{Lype}$  n:t:

 $\frac{n:\mathsf{name}\quad t:\mathsf{type}}{[n:t]:\mathsf{parameter}} \underbrace{ \begin{bmatrix} \mathsf{Named-Optional-Singular} \end{bmatrix} }$ 

```
a \times ys : b \dots \times c \to d \to e
\equiv x : a \times ys : (b \dots) \times c \to d \to e
\equiv (x : a) \times (ys : (b \dots)) \times c \to d \to e
\equiv ((x : a) \times (ys : (b \dots)) \times c) \to d \to e
\equiv ((x : a) \times (ys : (b \dots)) \times c) \to d \to e
\equiv ((x : a) \times (ys : (b \dots)) \times c) \to (d \to e)
a \times y : b \to c
\equiv a \times (y : b) \to c
\equiv (a \times (y : b)) \to c
a \times y : b \to c
\equiv (a \times (y : b)) \to c
```

Figure 3: Examples of operator precedence

```
Either a b \equiv \{type : "left"; value : a\} \lor \{type : "right"; value : b\}

flatMap : (@ : Observable \ A \times f : A \rightarrow Observable \ B \lor Event \ B \lor B) \rightarrow EventStream \ B
```

Figure 4: Examples of real world types