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	$(a \vee b) \vee c \equiv a \vee (b \vee c)$
4	product	$\times$	associative	$(a \times b) \times c \equiv a \times (b \times c)$
3	ellipsis	$\ldots$	postfix	$a \ldots$
2	type	:	right associative	$x : a$
1	function	$\rightarrow$	right associative	$a \rightarrow b \rightarrow c \equiv a \rightarrow (b \rightarrow c)$
0	semicolon	;	associative	

Table 1: Operator precedence

name		code variants
any	$\top$	$\top$ *
nothing	$\bot$	$\bot$ _|_
optional	?	?
conjunction	$\wedge$	$\wedge$ |
disjunction	$\vee$	$\vee$ &
ellipsis	$\ldots$	$\ldots$ (either three dots, or unicode ellipsis)
type	:	:
product	$\times$	$\times$,
function	$\rightarrow$	$\rightarrow$ ->
semicolon	;	;

Table 2: Special character code representations

Terminals:

$$\frac{}{\top : \text{type}}\ \boxed{\textsc{Any}} \qquad \frac{}{\bot : \text{type}}\ \boxed{\textsc{Nothing}}$$

$$\frac{}{\texttt{number} : \text{type}}\ \boxed{\textsc{Number}} \qquad \frac{}{\texttt{string} : \text{type}}\ \boxed{\textsc{String}} \qquad \frac{}{\texttt{bool} : \text{type}}\ \boxed{\textsc{Boolean}}$$

Identifiers:

$$\frac{n : \text{name}}{n : \text{type}}\ \boxed{\textsc{Identifier}}$$

Record pairs:

$$\frac{n : \text{name} \quad a : \text{type}}{n : a : \text{pair}}\ \boxed{\textsc{Record-Pair}} \qquad \frac{r : \text{pair}}{r : \text{rlist}}\ \boxed{\textsc{Record-Singleton}} \qquad \frac{p : \text{pair} \quad r : \text{rlist}}{p; r : \text{rlist}}\ \boxed{\textsc{Record-Cons}}$$

Records:

$$\frac{}{\{\} : \text{type}}\ \boxed{\textsc{Empty-Record}} \qquad \frac{r : \text{rlist}}{\{r\} : \text{type}}\ \boxed{\textsc{Record}}$$

Row types:

$$\frac{a : \text{type} \quad b : \text{type}}{a \wedge b : \text{type}}\ \boxed{\textsc{Conjunction}} \qquad \frac{a : \text{type} \quad b : \text{type}}{a \vee b : \text{type}}\ \boxed{\textsc{Disjunction}} \qquad \frac{a : \text{type}}{a? : \text{type}}\ \boxed{\textsc{Optional}}$$

Application:

$$\frac{a : \text{type} \qquad b : \text{type}}{a\ b : \text{type}}\ \boxed{\textsc{Application}}$$

Parameter types:

$$\frac{a : \text{type}}{a : \text{parameter}''}\ \boxed{\textsc{Singular}} \qquad \frac{a : \text{type}}{a \ldots : \text{parameter}''}\ \boxed{\textsc{Variadic}}$$

Function parameters:

$$\frac{p : \text{parameter}''}{p : \text{parameter}'}\ \boxed{\textsc{Anonymous}} \qquad \frac{n : \text{name} \quad p : \text{parameter}''}{n : p : \text{parameter}'}\ \boxed{\textsc{Named}}$$

Function parameter modifiers:

$$\frac{p : \text{parameter}'}{p : \text{parameter}}\ \boxed{\textsc{Mandatory}} \qquad \frac{p : \text{parameter}'}{[p] : \text{parameter}}\ \boxed{\textsc{Optional}'}$$

Function parameter lists:

$$\frac{p : \text{parameter}}{p : \text{plist}}\ \boxed{\textsc{Singleton}} \qquad \frac{p : \text{parameter} \qquad l : \text{plist}}{p \times l : \text{plist}}\ \boxed{\textsc{Product}}$$

Functions:

$$\frac{a : \text{type}}{\rightarrow a : \text{type}}\ \boxed{\textsc{Action}} \qquad \frac{l : \text{plist}}{l \rightarrow a : \text{type}}\ \boxed{\textsc{Function}}$$

Figure 1: Typify syntax rules

Anonymous:

$$\frac{t : \text{type}}{t : \text{parameter}}\ \boxed{\textsc{Anon-Mandatory-Singular}} \qquad \frac{t : \text{type}}{t \ldots : \text{parameter}}\ \boxed{\textsc{Anon-Mandatory-Variadic}}$$

$$\frac{t : \text{type}}{[t] : \text{parameter}}\ \boxed{\textsc{Anon-Optional-Singular}} \qquad \frac{t : \text{type}}{[t \ldots] : \text{parameter}}\ \boxed{\textsc{Anon-Optional-Variadic}}$$

Named:

$$\frac{n : \text{name} \quad t : \text{type}}{n : t : \text{parameter}}\ \boxed{\textsc{Named-Mandatory-Singular}} \qquad \frac{n : \text{name} \quad t : \text{type}}{n : t \ldots : \text{parameter}}\ \boxed{\textsc{Named-Mandatory-Variadic}}$$

$$\frac{n : \text{name} \quad t : \text{type}}{[n : t] : \text{parameter}}\ \boxed{\textsc{Named-Optional-Singular}} \qquad \frac{n : \text{name} \quad t : \text{type}}{[n : t \ldots] : \text{parameter}}\ \boxed{\textsc{Named-Optional-Variadic}}$$

Figure 2: Alternative parameter rules

$$
\begin{aligned}
& a \times ys : b \ldots \times c \to d \to e \\
& \quad \equiv x : a \times ys : (b \ldots) \times c \to d \to e \\
& \quad \equiv (x : a) \times (ys : (b \ldots)) \times c \to d \to e \\
& \quad \equiv ((x : a) \times \big(ys : (b \ldots)) \times c) \to d \to e \\
& \quad \equiv ((x : a) \times \big(ys : (b \ldots)) \times c) \to (d \to e) \\
& a \times y : b \to c \\
& \quad \equiv a \times (y : b) \to c \\
& \quad \equiv (a \times (y : b)) \to c
\end{aligned}
$$

$$
\begin{aligned}
& a \vee b? \wedge c\ d \ldots \to e \\
& \quad \equiv a \vee (b?) \wedge (c\ d) \ldots \to e \\
& \quad \equiv a \vee ((b?) \wedge (c\ d)) \ldots \to e \\
& \quad \equiv (a \vee ((b?) \wedge (c\ d))) \ldots \to e \\
& \quad \equiv ((a \vee ((b?) \wedge (c\ d))) \ldots) \to e
\end{aligned}
$$

Figure 3: Examples of operator precedence

$$
\begin{aligned}
& Either\ a\ b \equiv \{type : \text{"}\texttt{left}\text{"};\ value : a\} \vee \{type : \text{"}\texttt{right}\text{"};\ value : b\} \\
& flatMap : (@ : Observable\ A \times f : A \to Observable\ B \vee Event\ B \vee B) \to EventStream\ B
\end{aligned}
$$

Figure 4: Examples of real world types