Sample Document Using OOlong

November 28, 2017

1 Examples

This document exemplifies how to extract the LaTeX version of the OOlong semantics from Ott and our hand-written grammars. Figure 1 shows the syntax of OOlong. The corresponding LaTeX code is in `syntax.tex`. Figure 2 shows the syntax of OOlong. The corresponding LaTeX code is in `runtimeSyntax.tex`.

Figure 3 shows the well-formedness rules for classes and interfaces. The type rules are generated using the `\drules` command, defined by Ott. It is also possible to include single rules without the headers in the figure by using the `\drule` command:

WF-LET

$$\frac{\Gamma \vdash e_1 : t_1 \qquad \Gamma, x : t_1 \vdash e_2 : t}{\Gamma \vdash \textbf{let}\ x = e_1\ \textbf{in}\ e_2 : t}$$

The LaTeX sources for the type rules are generated from the file `oolong.ott` (by running the makefile in this directory). They look extra nice because we are using the `ottalt` package[1]! In order to use the `\drule` and `\drules` commands, the preamble must contain `\inputott{oolong.ott.tex}`.

Below follows the formal definitions of progress and preservation.

Progress A well-formed configuration is either done, has thrown an exception, has deadlocked, or can take one additional step:

$$\begin{array}{l}\forall \Gamma,\ H,\ V,\ T,\ t\ .\ \Gamma \vdash \langle H; V; T\rangle : t\ \Longrightarrow \\ \quad T = (\mathcal{L}, v) \vee T = \textbf{EXN} \vee \mathit{Blocked}(\langle H; V; T\rangle) \vee \\ \quad \exists \mathit{cfg}', \langle H; V; T\rangle \hookrightarrow \mathit{cfg}'\end{array}$$

Preservation If $\langle H; V; T\rangle$ types to t under some environment Γ, and $\langle H; V; T\rangle$ steps to some $\langle H'; V'; T'\rangle$, there exists an environment subsuming Γ which types $\langle H'; V'; T'\rangle$ to t.

$$\begin{array}{l}\forall \Gamma,\ H,\ H',\ V,\ V',\ T,\ T',\ t. \\ \quad \Gamma \vdash \langle H; V; T\rangle : t \wedge \langle H; V; T\rangle \hookrightarrow \langle H'; V'; T'\rangle\ \Longrightarrow \\ \quad\quad \exists \Gamma'.\Gamma' \vdash \langle H'; V'; T'\rangle : t \wedge \Gamma \subseteq \Gamma'\end{array}$$

P	::=	$Ids\ Cds\ e$	*(Programs)*
Id	::=	**interface** I `{`$Msigs$`}`	*(Interfaces)*
	|	**interface** I **extends** I_1, I_2	
Cd	::=	**class** C **implements** I `{`$Fds\ Mds$`}`	*(Classes)*
$Msig$	::=	$m(x : t_1) : t_2$	*(Signatures)*
Fd	::=	$f : t$	*(Fields)*
Md	::=	**def** $Msig$`{`e`}`	*(Methods)*
e	::=	$v \mid x \mid x.f \mid x.f = e \mid x.m(e)$	*(Expressions)*
	|	**let** x = e_1 **in** e_2 | **new** C | $(t)\ e$	
	|	**finish**`{`**async**`{`e_1`}` **async**`{`e_2`}}`; e_3	
	|	**lock**(x) **in** e | **locked**$_\iota${e}	
v	::=	**null** | ι	*(Values)*
t	::=	$C \mid I \mid$ **Unit**	*(Types)*
Γ	::=	$\epsilon \mid \Gamma, x : t \mid \Gamma, \iota : C$	*(Typing environment)*

Figure 1: Syntax of OOlong

cfg	::=	$\langle H; V; T \rangle$	*(Configuration)*
H	::=	$\epsilon \mid H, \iota \mapsto obj$	*(Heap)*
V	::=	$\epsilon \mid V, x \mapsto v$	*(Variable map)*
T	::=	$(\mathcal{L}, e) \mid T_1 \,\|\|\, T_2 \rhd e \mid$ **EXN**	*(Threads)*
obj	::=	(C, F, L)	*(Objects)*
F	::=	$\epsilon \mid F, f \mapsto v$	*(Field map)*
L	::=	**locked** | **unlocked**	*(Lock status)*
EXN	::=	**NullPointerException**	*(Exceptions)*

Figure 2: Syntax of runtime constructs of OOlong

$\boxed{\vdash P : t \quad \vdash Id \quad \vdash Cd \quad \vdash Fd \quad \vdash Md}$ *(Well-formed program)*

$$\textsc{wf-program}\ \frac{\forall\, Id \in Ids.\ \vdash Id \qquad \forall\, Cd \in Cds.\ \vdash Cd \qquad \epsilon \vdash e : t}{\vdash Ids\ Cds\ e : t}$$

$$\textsc{wf-interface}\ \frac{\forall\, m(x : t) : t' \in Msigs.\ \vdash t \wedge \vdash t'}{\vdash \textbf{interface}\ I\ \{\ Msigs\ \}} \qquad \textsc{wf-interface-extends}\ \frac{\vdash I_1 \qquad \vdash I_2}{\vdash \textbf{interface}\ I\ \textbf{extends}\ I_1, I_2}$$

$$\textsc{wf-class}\ \frac{\begin{array}{c}\forall\, m(x : t) : t' \in \textbf{msigs}\,(I).\textbf{def}\, m(x : t) : t'\ \{\ e\ \} \in Mds \\ \forall\, Fd \in Fds.\ \vdash Fd \qquad \forall\, Md \in Mds.\textbf{this} : C \vdash Md\end{array}}{\vdash \textbf{class}\ C\ \textbf{implements}\ I\ \{\ Fds\ Mds\ \}} \qquad \textsc{wf-field}\ \frac{\vdash t}{\vdash f : t}$$

$$\textsc{wf-method}\ \frac{\textbf{this} : C, x : t \vdash e : t'}{\textbf{this} : C \vdash \textbf{def}\, m(x : t) : t'\ \{\ e\ \}}$$

Figure 3: Well-formedness of classes and interfaces.

$$\textbf{vardom}(\Gamma) = \{x \mid x \in \textbf{dom}(\Gamma)\}$$

$$\textbf{msigs}(I) = \begin{cases} Msigs & \text{if } \textbf{interface}\ I\{Msigs\} \in P \\ \textbf{msigs}(I_1) \cup \textbf{msigs}(I_2) & \text{if } \textbf{interface}\ I\ \textbf{extends}\ I_1, I_2 \in P \end{cases}$$

$$\textbf{msigs}(C) = \{Msig \mid \textbf{def}\ Msig\{e\} \in Mds\} \text{ if } \textbf{class}\ C \ldots \{\_\ Mds\} \in P$$

$$\textbf{msigs}(t)(m) = x : t_1 \to t_2 \ \text{ if } \ m(x : t_1) : t_2 \in \textbf{msigs(t)}$$

$$\textbf{heldLocks}(T) = \begin{cases} \mathcal{L} & \text{if } T = (\mathcal{L}, e) \\ \textbf{heldLocks}(T_1) \cup \textbf{heldLocks}(T_2) & \text{if } T = T_1 \,||\, T_2 \rhd e \end{cases}$$

$$\textbf{locks}(e) = \{\iota \mid \textbf{locked}_\iota\{e'\} \in e\}$$

$$\begin{array}{l} \mathit{distinctLocks}(e) \equiv |\textbf{locks}(e)| = |\textbf{lockList}(e)| \\ \quad \text{where } \textbf{lockList}(e) = [\iota \mid \textbf{locked}_\iota\{e'\} \in e] \end{array}$$

Figure 4: Helper functions

References

[1] Jesse Tov, The `ottalt` LaTeX package:
`http://users.eecs.northwestern.edu/~jesse/code/latex/ottalt/ottalt.pdf`.