CS652 Smalltalk VM Operational Semantics

Terence Parr

April 3, 2015

$T \bowtie x$	Resolve x in scope T
$o \in \texttt{X}$	o instance of X
$o_{class} \in \texttt{STMetaClassObject}$	Metaclass (type) of object o
$o_{class_{class}} = o_{class}$	A metaclass object is its own type
$o_{superclass} \in \texttt{STMetaClassObject}$	Superclass (type) of object o
o_{field_i}	The ith field of object o
$f_{literal_i}$	The ith literal of method f
$f_s^{block_i} \in \texttt{BlockDescriptor}$	The ith block of method f associated with instance self=s
$f_s^{block_i}[\_,\_,\_] \in \texttt{BlockContext}$	The ith block of method f invoked with self=s
$f_s^{block_i}[\_,\_,\_]^d \in \texttt{BlockContext}$	The ith block of method f invoked with self=s and having depth d counting from zero at the method block; e.g., `f [|x| [|y|]]` has a method block at depth 0 with `x` and a nested block at depth 1 with `y`
$\gamma \in \texttt{MethodContext}^*$	Stack of method invocations growing to the right
$\delta \in \texttt{Object}^*$	Operand stack of objects growing to the right
$\mathbb{S}$	The state of the VM system dictionary
$(\mathbb{S}, \gamma)$	VM state is the system dictionary and a method invocation stack with zero or more elements
$(\mathbb{S}, \gamma) \Rightarrow (\mathbb{S}', \gamma')$	VM state transition
$(\mathbb{S}, \gamma) \Rightarrow^* (\mathbb{S}', \gamma')$	Zero-or-more state transitions
$f_s[ip, l_0, ..l_{n-1}, \delta]$	Method invocation context that derived from sending message f to receiver s (self); $f \in \texttt{MethodContext}$; l_i is local variable or argument, indexed from 0 and arguments first; δ is the operand stack; f can also represent a nested code block not just a method
$f[ip, l_0, ..l_{n-1}, \delta]$	Same as previous but the receiver is unknown or irrelevant
$f[ip, \_, \_]$	A method invitation context with "don't care" for locals and operand stack

Figure 1: Smalltalk VM Bytecode Specification Notation

Bytecode Instruction	Transition
initial state	$state_0 = (\mathbb{S}[\texttt{nil}, \texttt{true}, \texttt{false}, \texttt{Transcript}], \texttt{main}_m[0, \epsilon, \epsilon])$ for $m \in \texttt{MainClass}$; program terminates if $\exists\, state_0 \Rightarrow^* (\mathbb{S}', \epsilon)$
$\texttt{nil}$	$(\mathbb{S}, \gamma f[ip, \_, \delta]) \Rightarrow (\mathbb{S}, \gamma f[ip + 1, \_, \delta\,\texttt{nil}])$
$\texttt{self}$	$(\mathbb{S}, \gamma f_s[ip, \_, \delta]) \Rightarrow (\mathbb{S}, \gamma f_s[ip + 1, \_, \delta\,s])$
$\texttt{true}$	$(\mathbb{S}, \gamma f[ip, \_, \delta]) \Rightarrow (\mathbb{S}, \gamma f[ip + 1, \_, \delta\,\texttt{true}])$
$\texttt{false}$	$(\mathbb{S}, \gamma f[ip, \_, \delta]) \Rightarrow (\mathbb{S}, \gamma f[ip + 1, \_, \delta\,\texttt{false}])$
$\texttt{push\_char } c$	$(\mathbb{S}, \gamma f[ip, \_, \delta]) \Rightarrow (\mathbb{S}, \gamma f[ip + 3, \_, \delta\,c])]$
$\texttt{push\_int } i$	$(\mathbb{S}, \gamma f[ip, \_, \delta]) \Rightarrow (\mathbb{S}, \gamma f[ip + 5, \_, \delta\,i])$
$\texttt{push\_float } i$	$(\mathbb{S}, \gamma f[ip, \_, \delta]) \Rightarrow (\mathbb{S}, \gamma f[ip + 5, \_, \delta\,intBitsToFloat(i)])$
$\texttt{push\_field } i$	$(\mathbb{S}, \gamma f_s[ip, \_, \delta]) \Rightarrow (\mathbb{S}, \gamma f_s[ip + 3, \_, \delta\,s_{field_i}])$
$\texttt{push\_local } 0, i$	$(\mathbb{S}, \gamma f[ip, ...l_i..., \delta]) \Rightarrow (\mathbb{S}, \gamma f[ip + 5, ...l_i..., \delta\,l_i])$
$\texttt{push\_local } n > 0, i$	$(\mathbb{S}, \gamma f[\_, \_, \_] ... g^{block}[\_, ...\boldsymbol{l_i}..., \_]^{d-n} ... g^{block'}[ip, \_, \_]^{d-1} ... g^{block''}[ip, \_, \delta]^d) \Rightarrow$ $(\mathbb{S}, \gamma\,...\,g^{block''}[ip + 5, \_, \delta\boldsymbol{l_i}]^d)$
$\texttt{push\_literal } i$	$(\mathbb{S}, \gamma f[ip, \_, \delta]) \Rightarrow (\mathbb{S}, \gamma f[ip + 3, \_, \delta\,f_{literal_i}])$
$\texttt{push\_global } i$	$(\mathbb{S}, \gamma f[ip, \_, \delta]) \Rightarrow (\mathbb{S}, \gamma f[ip + 3, \_, \delta\,\mathbb{S}[f_{literal_i}]])$
$\texttt{push\_array } n$	$(\mathbb{S}, \gamma f[ip, \_, \delta\,a_1..a_n]) \Rightarrow (\mathbb{S}, \gamma f[ip + 3, \_, \delta A])$ where $A = Array(a_1..a_n)$
$\texttt{store\_field } i$	$(\mathbb{S}, \gamma f_s[ip, \_, \delta\,\mathbf{v}]) \Rightarrow (\mathbb{S}[s_{field_i} = \mathbf{v}], \gamma f_s[ip + 3, \_, \delta])$
$\texttt{store\_local } n, i$	$(\mathbb{S}, \gamma f[ip, ...l_i..., \delta\,\mathbf{v}]) \Rightarrow (\mathbb{S}, \gamma f[ip + 5, ...l_{i-1}\mathbf{v}\,l_{i+1}..., \delta])$
$\texttt{pop}$	$(\mathbb{S}, \gamma f[ip, \_, \delta\,v]) \Rightarrow (\mathbb{S}, \gamma f[ip + 1, \_, \delta])$
$\texttt{send } n, i$	$(\mathbb{S}, \gamma f[ip, \_, \delta\,r\,p_1..p_n]) \Rightarrow (\mathbb{S}, \gamma f[ip + 5, \_, \delta]\,(r_{class} \bowtie f_{literal_i})_r[0, p_1..p_n, \epsilon])$
$\texttt{send\_super } n, i$	$(\mathbb{S}, \gamma f[ip, \_, \delta\,r\,p_1..p_n]) \Rightarrow (\mathbb{S}, \gamma f[ip + 5, \_, \delta]\,(r_{superclass} \bowtie f_{literal_i})_r[0, p_1..p_n, \epsilon])$
$\texttt{block } i$	$(\mathbb{S}, \gamma f[ip, \_, \delta]) \Rightarrow (\mathbb{S}, \gamma f[ip + 3, \_, \delta\,f_s^{block_i}])$
$\texttt{block\_return}$	$(\mathbb{S}, \gamma f[ip, \_, \delta]\,g^{block}[\_, \_, \delta'v]) \Rightarrow (\mathbb{S}, \gamma f[ip, \_, \delta\,v])$
(local method) $\texttt{return}$	$(\mathbb{S}, \gamma f[ip, \_, \delta]\,g[\_, \_, \delta'v]) \Rightarrow (\mathbb{S}, \gamma f[ip, \_, \delta\,v])$
(nonlocal method) $\texttt{return}$	$(\mathbb{S}, \gamma f[ip, \_, \delta]\,g_s[\_, \_, \_] ... h[\_, \_, \_]\,g_s^{block}[\_, \_, \delta'v]) \Rightarrow (\mathbb{S}, \gamma f[ip, \_, \delta\,v])$
$\texttt{dbg } i, loc$	$(\mathbb{S}, \gamma f[ip, \_, \_]) \Rightarrow (\mathbb{S}[file{=}f_{literal_i}, line{=}loc[31{:}8], col{=}loc[7{:}0]], \gamma f[ip + 7, \_, \_])$ Set VM current filename to $f_{literal_i}$ and split loc into char position (indexed from 0) from lower 8 bits and line number from the upper 24 bits.

Figure 2: Smalltalk VM State Transition Rules

Smalltalk fragment	Generated code
ϵ	ϵ
`class T : S [ ]`	ϵ

C
 $main$
	$main$
	 `self`
	 `return`

Figure 3: Smalltalk Compilation Rules