Notes from Semantics and verification of programs

Jacek Olczyk

October 2018

# Part I
# Notes from tutorials by Lorenzo Clemente

## 1 Small step semantics - continuation

### 1.1 Recap

- Global environments $\rho \vdash e \to e'$
- $\frac{\rho[x \to n] \vdash e \to e'}{\rho \vdash \text{ let } x = \underline{n} \text{ in } e \to \text{ let } x = \underline{n} \text{ in } e'}$

### 1.2 Local environments

- How do we define the semantics for 'let $x = e$ in $f$' expressions using local environments? More precisely, we need $e$ to have its own environment, so that its evaluation doesn't affect the environment of $f$, as is the case with global environments.
- We are given the following 2 rules:
- $\frac{}{(\rho, x) \to (\rho, \rho(x))}$
- $\frac{}{(\rho, \text{ let } x = \underline{n} \text{ in } e) \to (\rho[x \to n], e)}$
- Now we need to give a rule for evaluating let expressions where a non-numeric expression is assigned to $x$.
- $\frac{(\rho, e) \to (\rho', e')}{(\rho, \text{ let } x = e \text{ in } f) \to ((\rho\text{? or maybe } \rho'\text{?}), \text{ let } x = e' \text{ in } f)}$
- $\rho$ doesn't work, because then a nested let in expression can't change the value of their variables.

- Neither does $\rho'$, because then we don't get our original environment back at the end.
- Solution: new construct
- $e$ then $x = n$
- Now we have:
- $\frac{(\rho,e)\to(\rho',e')}{(\rho,e \text{ then } x=\underline{n})\to(\rho',e' \text{ then } x=\underline{n})}$
- $\frac{}{(\rho,\underline{m} \text{ then } x=\underline{n})\to(\rho[x\to\underline{n}],\underline{m})}$
- $\frac{}{(\rho, \text{ let } x=\underline{n} \text{ in } e)\to(\rho[x\to\underline{n}],e \text{ then } x=\rho(x))}$

# 2 Imperative language

- Syntax
- $C ::=$ Skip $|X := e|C;C|$if $b$ then $c$ else $c|$while $b$ do $c$
- $e ::= n|x|e + e$
- $b :: true|false|e \leq e|\neg b|b \wedge b$
- $E[[e]]_s \in \mathbb{Q}$, $B[[b]]_s \in \{true, false\}$
- $s \in State = Var \to \mathbb{Q}$
- Configurations
- $(c, s) \in C$
- $s \in C$ (final)
- Small step rules for C - expressions
- $\frac{}{(Skip,s)\to s}$
- $\frac{}{(x:=e,s)\to s[x\to E[[e]]_s}$
- $\frac{(c,s)\to s'}{(c;d,s)\to(d,s')}$
- $\frac{(c,s)\to(c',s')}{(c;d,s)\to(c';d,s')}$
- $\frac{B[[b]]_s=true}{(\text{if } b \text{ then } c \text{ else } d,s)\to(c,s)}$
- $\frac{B[[b]]_s=false}{(\text{if } b \text{ then } c \text{ else } d,s)\to(d,s)}$
- $\frac{B[[b]]_s=true}{(\text{while } b \text{ do } c,s)\to(c;\text{while } b \text{ do } c,s)}$

- $\frac{B[[b]]_s = false}{(\text{while } b \text{ do } c,s) \to s}$
- Adding "Repeat c until b"
- $\frac{}{(\text{Repeat } c \text{ until } b,s) \to (c; \text{if } b \text{ then } Skip \text{ else Repeat } c \text{ until } b,s)}$

# 3 Numbers as strings of bits

- Evaluate:
- $n ::= \$0|\$1|n0|n1|n+n$
- final configurations: numbers without "+", e.g. \$100101
- $n \to n'$
- $\frac{n \to n'}{n0 \to n'0}$
- $\frac{n \to n'}{n1 \to n'1}$
- $\frac{m \to m'}{m+n \to m'+n}$
- $\frac{n \to n'}{m+n \to m+n'}$
- $\frac{}{m0+n0 \to (m+n)0}$
- $\frac{}{m0+n1 \to (m+n)1}$
- $\frac{}{m1+n0 \to (m+n)1}$
- $\frac{}{m1+n1 \to (m+n+\$1)0}$
- Fill in the last 4
- I think we should add a rule to merge two doll

# 4 Next time

Add to the syntax:

- for x:=e to e do c
- do e times c
- do c while e