316 10-22-09

Semantics of expressions given by $Eval(E, \alpha)$

Semantics of expressions given by $M(S, \alpha)$

$\boxed{\text{assignment } x_i = E}$

$M(x_i = E, \{\langle x_1, v_1\rangle, \ldots, \langle x_n, v_n\rangle\}) =$

if $(v_E = \perp_V)$ then $\perp_S$

else $\{\langle x_1, v_1\rangle, \ldots, \langle x_i, v_E\rangle, \ldots \langle x_n, V_n\rangle\}$

where $v_E = Eval(E, \{\langle x_1, v_1\rangle, \ldots \langle x_n, v_n\rangle\})$

if (B) $S_1$ else $S_2$

$M(\text{if}(B)\ S_1 \text{ else } S_2, \alpha) =$

if $(Eval(B, \alpha) = \perp_V)$ then $\perp_S$

else if $(Eval(B, \alpha) = \text{true})$ then $M(S_1, \alpha)$

else $M(S_2, \alpha)$

if (B) S

$M(\text{if}(B) S, \alpha) =$

if $(Eval(B, \alpha) = \perp_V)$ then $\perp_S$

else $Eval(B, \alpha) = \text{true}$ then $M(S, \alpha)$

else $\alpha$

while (B) do S

$M(\text{while}(B) \text{ do } S, \alpha) =$

if $(Eval(B, \alpha) = \perp_V)$ then $\perp_S$

else if $Eval(B, \alpha) = \text{false}$ then $\alpha$

else if $(M(S, \alpha) = \perp_S)$ then $\perp_S$

else $M(\text{while}(B) \text{ do } S, M(S, \alpha))$ //iterate the loop from the new state $M(S, \alpha)$

do S while (B)

$M(\text{do } S \text{ while}(B), \alpha) =$

if $(M(S, \alpha) = \perp_S)$ then $\perp_S$

else if $(Eval(B, M(S, \alpha) = \perp_V)$ then $\perp_S$

else if $(Eval(B, M(S, \alpha) == \text{false})$ then $M(S, \alpha)$

else $M(\text{do } S \text{ while}(B), M(S, \alpha))$

$\alpha \rightarrow$ $S_1; S_2; \dots ; S_n \rightarrow ?$

$M(S_n, \dots M(S_2, M(S_1, \alpha)) \dots)$

$\alpha_0$

$S_1$: $x = 3;$

$\alpha_1$

$S_2$: $a = 2;$

$\alpha_2$

$S_3$: if $(x < a)$ $\overset{S_4}{\boxed{x = x + a}}$; else $\overset{S_5}{\boxed{a = x + a}}$

$\alpha_3$

$\alpha_0$ = initial program state $= \{<x, \perp>, <a, \perp>\}$

$\alpha_1 = M(S_1, \alpha_0) = \{<x, 3>, <a, \perp>\}$

$\alpha_2 = M(S_2, \alpha_1) = \{<x, 3>, <a, 2>\}$

$\alpha_3 = M(S_3, \alpha_2)$ ?

Eval $(x < a, \alpha_2) = 3 < 2 =$ false

$\alpha_3 = M(S_3, \alpha_2) = M(S_5, \alpha_2) = \{<x, 3>, <a, 5>\}$

---

$\alpha_0$

$S_1$: $x = 3;$

$\alpha_1$

$S_2$: $a = 2;$

$\alpha_2$

$S_3$: while $(x < 5)$ do $\overset{S_4}{\boxed{x = x + a}}$

$\alpha_3$

$\alpha_0, \alpha_1, \alpha_2$ are the same as before

$\alpha_3 = M(S_3, \alpha_2)$ ?

Eval $(x < 5, \alpha_2) = 3 < 5 =$ true

$\alpha_4 = M(S_4, \alpha_2) = \{<x, 5>, <a, 2>\}$

$\alpha_3 = M(S_3, \alpha_2) = M(S_3, \alpha_4)$

Eval $(x < 5, \alpha_4) = 5 < 5 =$ false

$M(S_3, \alpha_4) = \alpha_4$

$\alpha_3 = M(S_3, \alpha_2) = M(S_3, \alpha_4) = \alpha_4$

- Eval $(E, \alpha)$ and $M(S, \alpha)$ can be implemented by a program (ie: by recursive functions)
- $E$ and $S$ are implemented by explicit parse trees.
- Data structures for program state $\alpha$
  - List or vector of $<x_i, v_i>$ pairs
  - More efficient table structures like Hash Tables.

- Resulting program is pure interpreter

<u>Semantics By Equivalence</u> = Applicable to any semantic description method (operational, denotational, axiomatic, etc.)

do S while(B) is equivalent to: S; while (B) do S

for(init_statement; B ; inc_statement) S is equivalent to:

```
inc_statement;
while (B) do
{
  S;
  inc_statement;
}
```

<u>Controlling Function Calls by Runtime Stack</u>