

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
Department of Electrical Engineering and Computer Science

Problem Set 1A Solutions

A. Paper Problems - Due September 11

Problem 1 (PostFix + Pairs)

Do Exercise 3.37 on page 96 of the text.

Solution:

We need to change some domains before we leap ahead and write new transition rules. Command must be changed or our new commands won't be recognized, and Value must be changed to include pairs:

$$\begin{aligned}V &\in \text{Value} = \text{IntLit} + \text{CommandSeq} + \text{Pair} \\ \text{Pair} &= \text{Value} \times \text{Value}\end{aligned}$$

$$C ::= \text{pair} \mid \text{left} \mid \text{right} \mid \dots \text{PostFix commands} \dots$$

The transition relation \Rightarrow is extended with the following three axioms; the other axioms remain unchanged:

$$\langle \text{pair}.Q, V_1.V_2.S \rangle \Rightarrow \langle Q, \langle V_2, V_1 \rangle.S \rangle \quad [\text{pair}]$$

$$\langle \text{left}.Q, \langle V_1, V_2 \rangle.S \rangle \Rightarrow \langle Q, V_1.S \rangle \quad [\text{left}]$$

$$\langle \text{right}.Q, \langle V_1, V_2 \rangle.S \rangle \Rightarrow \langle Q, V_2.S \rangle \quad [\text{right}]$$

We handle errors just as the book does on page 50, by introducing stuck states that “are exactly those irreducible configurations that are non-final. . . The outcome of a program that reaches such a configuration will be stuck.”

Problem 2 (PostText)

Do Exercise 3.44 on page 99 of the text.

Solution:

- a. Assume a dictionary is a sequence of name-value pairs:

$$\text{Dictionary} = (\text{Identifier} \times \text{Value})^*$$

- i. The SOS for PostText looks very much like the SOS for PostFix with dictionaries added everywhere. We describe a PostText SOS which models all errors as stuck states as on page 50 of the text; you may have chosen to handle errors differently. We must change the Value domain to be $\text{IntLit} + \text{CommandSeq} + \text{Identifier}$, since an identifier can be left at the top of the stack.

$$\begin{aligned} \mathcal{C} &= \text{CommandSeq} \times \text{Stack} \times \text{Dictionary} \\ \mathcal{F}' &= \{\llbracket \text{Command} \rrbracket\} \times \text{FinalStack} \times \text{Dictionary} \\ \text{Stuck} &= \text{Irreducible} - \mathcal{F}' \\ \mathcal{F} &= \mathcal{F}' \cup \text{Stuck} \end{aligned}$$

$$\begin{aligned} \mathcal{I} &: \text{Program} \times \text{Inputs} \rightarrow \mathcal{C} \\ &= \lambda \langle (\text{posttext } N \ Q), [N_1, \dots, N_n] \rangle. \\ &\quad \mathbf{if } N = n \\ &\quad \mathbf{then } \langle Q, [(\text{IntLit} \mapsto \text{Value } N_1), \dots, (\text{IntLit} \mapsto \text{Value } N_n)], \llbracket \text{Identifier} \times \text{Value} \rrbracket \rangle \\ &\quad \mathbf{else } \langle \llbracket \text{Command} \rrbracket, \llbracket \text{Value} \rrbracket, \llbracket \text{Identifier} \times \text{Value} \rrbracket \rangle \\ \mathcal{O} &: \mathcal{F} \rightarrow \text{Answer} = \lambda c. \mathbf{if } c \in \text{Stuck} \\ &\quad \mathbf{then } (\text{Error} \mapsto \text{Answer } \text{error}) \\ &\quad \mathbf{else } (\mathcal{O}' \ c) \\ \mathcal{O}' &: \mathcal{F} \rightarrow \text{Answer} = \lambda \langle \llbracket \text{Command} \rrbracket, (\text{IntLit} \mapsto \text{Value } N) \cdot S', D \rangle. (\text{IntLit} \mapsto \text{Answer } N) \end{aligned}$$

- ii. Here is one way to write axioms extending the transition relation for the new commands. We're careful in the ref rule to return the latest value defined for an identifier I . (Note that all the operations inherited from PostFix don't affect the dictionary; in PostText, axioms corresponding to those operations are modified to pass the dictionary along unchanged.)

$$\begin{aligned} \langle I.Q, S, D \rangle &\Rightarrow \langle Q, I.S, D \rangle && \text{[identifier]} \\ \langle \text{def}.Q, V.I.S, D \rangle &\Rightarrow \langle Q, S, \langle I, V \rangle.D \rangle && \text{[def]} \\ \langle \text{ref}.Q, I.S, D_1 @ \llbracket \langle I, V \rangle \rrbracket @ D_2 \rangle &\Rightarrow \langle Q, V.S, D_1 @ \llbracket \langle I, V \rangle \rrbracket @ D_2 \rangle && \text{[ref]} \\ &\text{where } \langle I, V' \rangle \notin D_1 \text{ for any Value } V' \end{aligned}$$

- b. Assume dictionaries are functions from identifiers to values:

$$\begin{aligned} D \in \text{Dictionary} &= \text{Identifier} \rightarrow \text{Binding} \\ \text{Binding} &= \text{Value} + \text{Unbound} \\ \text{Unbound} &= \{\text{unbound}\} \end{aligned}$$

- i. This SOS is not very different from part (a) except for the input function, which must handle the different representation of a dictionary. Note that we handle errors in the

same way as part (a).

$$EmptyDict = \lambda I. (Unbound \mapsto Binding \text{ unbound})$$

$$\begin{aligned} \mathcal{C} &= CommandSeq \times Stack \times Dictionary \\ \mathcal{F}' &= \{\llbracket Command \rrbracket\} \times FinalStack \times Dictionary \\ Stuck &= Irreducible - \mathcal{F}' \\ \mathcal{F} &= \mathcal{F}' \cup Stuck \end{aligned}$$

$$\begin{aligned} \mathcal{I} &: Program \times Inputs \rightarrow \mathcal{C} \\ &= \lambda \langle (\text{posttext } N \ Q), [N_1, \dots, N_n] \rangle. \\ &\quad \mathbf{if} \ N = n \\ &\quad \mathbf{then} \ \langle Q, [(IntLit \mapsto Value \ N_1), \dots, (IntLit \mapsto Value \ N_n)], EmptyDict \rangle \\ &\quad \mathbf{else} \ \langle \llbracket Command \rrbracket, \llbracket Value \rrbracket, EmptyDict \rangle \\ \mathcal{O} : \mathcal{F} \rightarrow Answer &= \lambda c. \ \mathbf{if} \ c \in Stuck \\ &\quad \mathbf{then} \ (Error \mapsto Answer \ error) \\ &\quad \mathbf{else} \ (\mathcal{O}' \ c) \\ \mathcal{O}' : \mathcal{F} \rightarrow Answer &= \lambda \langle \llbracket Command \rrbracket, (IntLit \mapsto Value \ N).S', D \rangle. (IntLit \mapsto Answer \ N) \end{aligned}$$

- ii. Here, we extend the transition relation using the *bind* function provided. We're careful in the ref axiom not to put unbound on the stack. (In fact, unbound is not allowed as a stack element; the stack holds Values, and Unbound is not a Value.)

$$\begin{aligned} \langle I.Q, S, D \rangle &\Rightarrow \langle Q, I.S, D \rangle && \text{[identifier]} \\ \langle \text{def}.Q, V.I.S, D \rangle &\Rightarrow \langle Q, S, (\text{bind } I \ V \ D) \rangle && \text{[def]} \\ \langle \text{ref}.Q, I.S, D \rangle &\Rightarrow \langle Q, V.S, D \rangle && \text{[ref]} \\ &\text{where } V = (D \ I) \neq \text{unbound} \end{aligned}$$