Semantics of Programming Languages

Dijkstra proposed the language of guarded commands with the following syntax. Commands take the form

\[ c := \text{skip} \mid \text{abort} \mid X := e \mid c; c \mid \text{if } gc \text{ fi} \mid \text{do } gc \text{ od} \]

where \( e \) is an arithmetic expression and \( gc \) stands for a guarded command of the form

\[ b_1 \rightarrow c_1 \parallel b_2 \rightarrow c_2 \]

for boolean expressions \( b_1 \) and \( b_2 \), called guards, and commands \( c_1 \) and \( c_2 \). Execution of the command \text{skip} does not result in a change of state. Following Dijkstra’s intentions, if no guard evaluates to true at a state, then the guarded command is said to fail, in which case, the guarded command does not yield a final state. Otherwise, the guarded command executes as one of the commands \( c_i \) whose associated guard \( b_i \) evaluates to true. The execution of the command \text{abort} does not yield a final state from any initial state. The command if \( gc \) does not fail, otherwise, it acts like \text{abort}. The command \text{do } gc \text{ od} executes repeatedly as the guarded command \( gc \), while \( gc \) continues not to fail, and terminates when \( gc \) fails.

(a) Assume that boolean and arithmetic expressions have no side effects and always terminate, and that the rules for their evaluation are given. Write down a collection of rules for an inductively defined evaluation relation of the form

\[ c, S \Rightarrow S' \]

whose sense is “starting from the initial state \( S \), the evaluation of the command \( c \) terminates at the final state \( S' \).”

(b) Give the commands in Dijkstra’s guarded language which simulate the standard imperative programming commands

\[ \text{if } b \text{ then } c_1 \text{ else } c_2 \quad \text{and} \quad \text{while } b \text{ do } c \]

respectively. You may assume that if \( b \) is a boolean expression, then so is \( \neg b \), the negation of \( b \).

(c) Give an appropriate definition of semantic equivalence of commands with respect to the evaluation relation defined in (a). Prove that for any boolean expression \( b \) and any command \( c \), the command

\[ \text{do } b \rightarrow c \parallel b \rightarrow c \text{ od} \]

is semantically equivalent to the command

\[ \text{if } b \rightarrow \left( c; \text{do } b \rightarrow c \parallel b \rightarrow c \text{ od} \right) \parallel \neg b \rightarrow \text{skip fi} \]