

Homework 5

CSC 7101, Spring 2007

Due: 3 May 2007

This homework is just for practice. You don't need to hand it in.

1. Consider Dijkstra's guarded conditional:

$$\langle c \rangle ::= \text{if } \langle be \rangle_1 \rightarrow \langle c \rangle_1 \square \langle be \rangle_2 \rightarrow \langle c \rangle_2 \square \langle be \rangle_3 \rightarrow \langle c \rangle_3 \text{ fi}$$

where **if** and **fi** are keywords, $\langle be \rangle_i$ are boolean expressions and $\langle c \rangle_i$ are commands. (Dijkstra's guarded conditional has arbitrarily many clauses; we'll restrict ourselves to exactly three for simplicity.)

The guarded conditional is a generalization of the if-statement with multiple conditions ($\langle be \rangle_1 \dots \langle be \rangle_3$) and multiple branches ($\langle c \rangle_1 \dots \langle c \rangle_3$). The boolean expressions are *guards* for the commands that follow them. A command $\langle c \rangle_i$ can only be executed if its guard $\langle be \rangle_i$ is true. If more than one guard is true, **one** of the commands whose guard is true is chosen **nondeterministically** (at random) to be executed.

Define the denotational semantics of this command. Since we are dealing with commands, you need to define a semantic function with a state argument. In the semantic domains in denotational semantics, there is no notion of nondeterminism. You, therefore, have to make the command deterministic. This is basically an exercise in translating your operational semantics definition for this construct into denotation semantics notation, except that you have to make the command deterministic. Explain your semantic definition.

2. Suppose we change the intuitive operational model of the language whose semantics we have been defining as follows: Whenever the value of a variable is 'used' its value reduces by 1. In other words, expression evaluation has side effects. Thus executing an assignment command like ' $x := y + z$ ' not only assigns a value to x , it also reduces the values of y and z by 1 each.

What changes would you have to make to the denotational semantics of the language? Specify all the required changes. If you need to make any assumptions, state these assumptions clearly.

3. Define the semantic function for a denotational semantics with state for the following (simplified) loop construct adopted from Ada:

loop S_1 ; **exit when** B ; S_2 **end**

In the above statement, S_1 and S_2 are statements and B is a boolean expression. The semantics of '**exit when** B ' is the same as '**if** (B) **break**;' in C.

Do not rely on recursion, but define an appropriate functional τ . The semantics of the construct is then $fix(\tau)$, where fix is the least fixed point operator.

4. (10 pts)

Given the following functional:

$$\tau[f](x) = \begin{cases} \perp & \text{if } x = \perp \vee x < 0 \\ x & \text{if } x = 0 \vee x = 1 \\ f(x-1) + f(x-2) & \text{if } x > 1 \end{cases}$$

Compute $f^1 = \tau[\Omega]$, $f^2 = \tau[f^1]$, $f^3 = \tau[f^2]$, and $f^4 = \tau[f^3]$, where $f^0 = \Omega$ is the function that doesn't terminate for any argument.