Programming Languages and Computation

Week 5: Denotational Semantics of Expressions

This problem sheet concerns the denotational semantics of the arithmetic and Boolean expressions.

* 1. Compute the denotation of the following arithmetic expressions in the state $\sigma = [x \mapsto -1, y \mapsto 11, z \mapsto 10]$. Your answer should be explicit about the steps you take and make reference to the definition of the denotation function.

   (a) $(x + y) - z$

   (b) $x * (12 - z)$

* 2. Compute the denotation of the following arithmetic expressions in the state $\sigma = [x \mapsto 11,\ y \mapsto 12]$. Your answer should be explicit about the steps you take and make reference to the definition of the denotation function.

   (a) $(x + y) - z$

   (b) $x * (12 - z)$

* 3. Compute the denotation of the following Boolean expressions in the state $\sigma = [x \mapsto 11,\ y \mapsto 12]$. Your answer should be explicit about the steps you take and make reference to the definition of the denotation function.

   (a) $(x + y \leq z)$ && true

   (b) $!(x * y = z) \mathbin{||} x = 11$

The next questions relate to when arithmetic expressions are *syntactically equivalent* or *semantically equivalent*. Two expressions are syntactically equivalent if they have the same abstract syntax tree and semantically equivalent is they denote the same function. Remember that two functions are equal if, and only if, they have the same value on every input.

* 4. Which of the following arithmetic expressions are syntactically equivalent and which are semantically equivalent (i.e. denote the same function)? Remember that the addition operator associates to the left.

(a) $x * 3$

(b) $x * 1$

(c) $x + (x + x)$

(d) $(x + x) + x$

(e) $x + x + x$

(f) $x + (y * 0)$

* 5. Suppose e_1 and e_2 are semantically equivalent arithmetic expressions. Prove that $e_1 + e_2$ is semantically equivalent to $e_1 * 2$. Your answer should make explicit reference to the denotation function.

** 6. Let us suppose we want to add a new construct to the language of arithmetic expressions:

$$A \to x \mid n \mid \cdots \mid \texttt{let}\ x = A\ \texttt{in}\ A$$

An expression $\texttt{let}\ x = e_1\ \texttt{in}\ e_2$ using this construct should evaluate the sub-expression e_2 in a state where the variable x is mapped to the value of e_1. For example, the expression $\texttt{let}\ x = 2\ \texttt{in}\ x + y$ when evaluated in the state $[x \mapsto 3,\ y \mapsto 2]$ should be 4.

Extend the definition of the denotation function $[\![\cdot]\!]_A$ with an equation for this construct. You do not need to change any other equations.

*** 7. The *free variables* of an expression is the set of variables that appear in that expression. Formally, we define a function $\mathsf{FV} : A \to \mathscr{P}(\mathsf{Var})$ from expressions to sets of variables by recursion over the structure of expressions as follows:

$$\begin{aligned}
\mathsf{FV}(x) &= \{x\} \\
\mathsf{FV}(n) &= \emptyset \\
\mathsf{FV}(a + b) &= \mathsf{FV}(a) \cup \mathsf{FV}(b) \\
\mathsf{FV}(a - b) &= \mathsf{FV}(a) \cup \mathsf{FV}(b) \\
\mathsf{FV}(a * b) &= \mathsf{FV}(a) \cup \mathsf{FV}(b)
\end{aligned}$$

(a) What are the free variables of the expression $x + 1$?

(b) Consider the denotation $[\![x + 1]\!]_A(\sigma)$ where σ is the state $[x \mapsto 2,\ y \mapsto 3]$. How does the denotation change if we instead consider the state $[x \mapsto 2,\ y \mapsto 4]$?

(c) Informally argue that, for *all* arithmetic expressions e and pair of states σ and σ' such that:

$$\forall x \in \mathsf{FV}(e).\, \sigma(x) = \sigma'(x)$$

that the denotations $[\![e]\!]_A(\sigma)$ and $[\![e]\!]_A(\sigma')$ are the same. How you could make this argument formal?