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 value 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) $x * (12 - z)$

(c) $x - (z - 1)$

Solution

(a)

$$\begin{aligned}
[\![(x+y)-z]\!]_{\mathcal{A}}(\sigma) &= [\![x+y]\!]_{\mathcal{A}}(\sigma) - [\![z]\!]_{\mathcal{A}}(\sigma) \\
&= ([\![x]\!]_{\mathcal{A}}(\sigma) + [\![y]\!]_{\mathcal{A}}(\sigma)) - [\![z]\!]_{\mathcal{A}}(\sigma) \\
&= (\sigma(x) + \sigma(y)) - \sigma(z) \\
&= ((-1) + 11) - 10 \\
&= 0
\end{aligned}$$

(b)

$$\begin{aligned}
[\![x*(12-z)]\!]_{\mathcal{A}}(\sigma) &= [\![x]\!]_{\mathcal{A}}(\sigma) \cdot [\![12-z]\!]_{\mathcal{A}}(\sigma) \\
&= [\![x]\!]_{\mathcal{A}}(\sigma) \cdot ([\![12]\!]_{\mathcal{A}}(\sigma) - [\![z]\!]_{\mathcal{A}}(\sigma)) \\
&= \sigma(x) \cdot (12 - \sigma(z)) \\
&= (-1) \cdot (12 - 10) \\
&= -2
\end{aligned}$$

(c)

$$\begin{aligned}[\![x-(z-1)]\!]_{\mathcal{A}}(\sigma) &= [\![x]\!]_{\mathcal{A}}(\sigma) - [\![z-1]\!]_{\mathcal{A}}(\sigma)\\ &= [\![x]\!]_{\mathcal{A}}(\sigma) - ([\![z]\!]_{\mathcal{A}}(\sigma) - [\![1]\!]_{\mathcal{A}}(\sigma))\\ &= \sigma(x) - (\sigma(z) - 1)\\ &= (-1) - (10-1)\\ &= -10\end{aligned}$$

* 2. Compute the value 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) $x*(12-z)$

(c) $x-(z-1)$

Solution

This question depends on the convention that variables not explicitly mentioned by the state are assigned the value 0.

(a)

$$\begin{aligned}[\![(x+y)-z]\!]_{\mathcal{A}}(\sigma) &= [\![x+y]\!]_{\mathcal{A}}(\sigma) - [\![z]\!]_{\mathcal{A}}(\sigma)\\ &= [\![x]\!]_{\mathcal{A}}(\sigma) + [\![y]\!]_{\mathcal{A}}(\sigma) - [\![z]\!]_{\mathcal{A}}(\sigma)\\ &= \sigma(x) + \sigma(y) - \sigma(z)\\ &= (11+12) - 0\\ &= 23\end{aligned}$$

(b)

$$\begin{aligned}[\![x*(12-z)]\!]_{\mathcal{A}}(\sigma) &= [\![x]\!]_{\mathcal{A}}(\sigma) \cdot [\![12-z]\!]_{\mathcal{A}}(\sigma)\\ &= [\![x]\!]_{\mathcal{A}}(\sigma) \cdot ([\![12]\!]_{\mathcal{A}}(\sigma) - [\![z]\!]_{\mathcal{A}}(\sigma))\\ &= \sigma(x) \cdot (12 - \sigma(z))\\ &= 11 \cdot (12-0)\\ &= 132\end{aligned}$$

(c)

$$\begin{aligned}[\![x-(z-1)]\!]_{\mathcal{A}}(\sigma) &= [\![x]\!]_{\mathcal{A}}(\sigma) - [\![z-1]\!]_{\mathcal{A}}(\sigma)\\ &= [\![x]\!]_{\mathcal{A}}(\sigma) - ([\![z]\!]_{\mathcal{A}}(\sigma) - [\![1]\!]_{\mathcal{A}}(\sigma))\\ &= \sigma(x) - (\sigma(z) - 1)\\ &= 11 - (0-1)\\ &= 12\end{aligned}$$

* 3. Compute the value 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) $(x + y \le z)$ && true

(b) $!(x * y = z) \,||\, x = 11$

(c) $x \le y$ && $y \le x$

Solution

(a)

$$\begin{aligned}
[\![(x + y \le z) \text{ \&\& } \mathsf{true}]\!]_{\mathcal{B}}(\sigma) &= [\![x + y \le z]\!]_{\mathcal{B}}(\sigma) \wedge [\![\mathsf{true}]\!]_{\mathcal{B}}(\sigma) \\
&= ([\![x + y]\!]_{\mathcal{A}}(\sigma) \le [\![z]\!]_{\mathcal{A}}(\sigma)) \wedge \top \\
&= ([\![x]\!]_{\mathcal{A}}(\sigma) + [\![y]\!]_{\mathcal{A}}(\sigma) \le [\![z]\!]_{\mathcal{A}}(\sigma)) \wedge \top \\
&= (\sigma(x) + \sigma(y) \le \sigma(z)) \;\wedge \top \\
&= (11 + 12 \le 0) \wedge \top \\
&= \bot \wedge \top \\
&= \bot
\end{aligned}$$

(b)

$$\begin{aligned}
[\![!(x * y = z) \,||\, x = 11]\!]_{\mathcal{B}}(\sigma) &= [\![!(x * y = z)]\!]_{\mathcal{B}}(\sigma) \vee [\![x = 11]\!]_{\mathcal{B}}(\sigma) \\
&= \neg[\![x * y = z]\!]_{\mathcal{B}}(\sigma) \vee ([\![x]\!]_{\mathcal{A}}(\sigma) = [\![11]\!]_{\mathcal{A}}(\sigma)) \\
&= \neg([\![x * y]\!]_{\mathcal{A}}(\sigma) = [\![z]\!]_{\mathcal{A}}(\sigma)) \vee ([\![x]\!]_{\mathcal{A}}(\sigma) = [\![11]\!]_{\mathcal{A}}(\sigma)) \\
&= \neg([\![x]\!]_{\mathcal{A}}(\sigma) * [\![y]\!]_{\mathcal{A}}(\sigma) = [\![z]\!]_{\mathcal{A}}(\sigma)) \vee ([\![x]\!]_{\mathcal{A}}(\sigma) = [\![11]\!]_{\mathcal{A}}(\sigma)) \\
&= \neg(\sigma(x) * \sigma(y) = \sigma(z)) \vee (\sigma(x) = 11) \\
&= \neg(11 * 12 = 0) \vee (11 = 11) \\
&= \neg\bot \vee \top \\
&= \top \vee \top \\
&= \top
\end{aligned}$$

(c)

$$\begin{aligned}
[\![x \le y \text{ \&\& } y \le x]\!]_{\mathcal{B}}(\sigma) &= [\![x \le y]\!]_{\mathcal{B}}(\sigma) \wedge [\![y \le x]\!]_{\mathcal{A}}(\sigma) \\
&= [\![x]\!]_{\mathcal{A}}(\sigma) \le [\![y]\!]_{\mathcal{A}}(\sigma) \wedge ([\![y]\!]_{\mathcal{A}}(\sigma) \le [\![x]\!]_{\mathcal{A}}(\sigma)) \\
&= \sigma(x) \le \sigma(y) \wedge \sigma(y) \le \sigma(x) \\
&= 11 \le 12 \wedge 12 \le 11 \\
&= \top \wedge \bot \\
&= \bot
\end{aligned}$$

The next questions relate to when arithmetic and Boolean expressions are *syntactically equivalent* or *semantically equivalent*. Two arithmetic or Boolean expressions are syntactically equivalent if they have exactly 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? 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) $x + (y * 0)$

Solution

The only syntactically equivalent expressions are (d) and (e) (thanks to associativity). However, the expressions (a), (c), (d), and (e) are semantically equivalent and likewise so are (b) and (f).

* 5. Consider the Boolean expression $x \leq 2$ && $y \leq 3$. Find a semantically equivalent expression that does not use the && operator.

Solution

Lots of possible answers, e.g. $!(!(x \leq 2) \,||\, !(x \leq 3))$.

* 6. Find a state in which the arithmetic expressions $x * 2$ and $x + 3$ evaluate to the same value. Explain why they are *not* semantically equivalent.

Solution

A state in which they evaluate to the same value is $[x \mapsto 3]$. They are not semantically equivalent, however, as they evaluate to distinct values in the state $[x \mapsto 1]$.

** 7. Suppose that $e_1 \in \mathcal{A}$ and $e_2 \in \mathcal{A}$ 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.

Solution

To show that $e_1 + e_2$ is semantically equivalent to $e_1 * 2$, we must show that, for any state $\sigma \in \text{State}$, $[\![e_1 + e_2]\!]_{\mathcal{A}}(\sigma) = [\![e_1 * 2]\!]_{\mathcal{A}}(\sigma)$. Let $\sigma \in \text{State}$ be a state. By expanding the definitions, we can see that $[\![e_1 + e_2]\!]_{\mathcal{A}}(\sigma) = [\![e_1]\!]_{\mathcal{A}}(\sigma) + [\![e_2]\!]_{\mathcal{A}}(\sigma)$ and that $[\![e_1 * 2]\!]_{\mathcal{A}}(\sigma) = [\![e_1]\!]_{\mathcal{A}}(\sigma) \cdot 2$. As e_1 and e_2 are semantically equivalent, we know that $[\![e_1]\!]_{\mathcal{A}}(\sigma) = [\![e_2]\!]_{\mathcal{A}}(\sigma)$. Therefore, we have that:

$$\begin{aligned}
[\![e_1 + e_2]\!]_{\mathcal{A}}(\sigma) &= [\![e_1]\!]_{\mathcal{A}}(\sigma) + [\![e_2]\!]_{\mathcal{A}}(\sigma) \\
&= [\![e_1]\!]_{\mathcal{A}}(\sigma) + [\![e_1]\!]_{\mathcal{A}}(\sigma) \\
&= [\![e_1]\!]_{\mathcal{A}}(\sigma) * 2 \\
&= [\![e_1 * 2]\!]_{\mathcal{A}}(\sigma)
\end{aligned}$$

Therefore, we can conclude that the two expressions are semantically equivalent as required.

** 8. Suppose that $e_1 + 1$ and $e_2 + 1$ are two arithmetic expressions that are semantically equivalent for some $e_1, e_2 \in \mathcal{A}$. Prove that e_1 and e_2 are also semantically equivalent.

Solution

To show that e_1 and e_2 are semantically equivalent let $\sigma \in \text{State}$ be an arbitrary state. We have that $[\![e_1 + 1]\!]_{\mathcal{A}}(\sigma) = [\![e_2 + 1]\!]_{\mathcal{A}}(\sigma)$. It follows, by definition, that $[\![e_1]\!]_{\mathcal{A}}(\sigma) + 1 = [\![e_2]\!]_{\mathcal{A}}(\sigma) + 1$. Therefore, $[\![e_1]\!]_{\mathcal{A}}(\sigma) = [\![e_2]\!]_{\mathcal{A}}(\sigma)$ as required.

** 9. Suppose that $e_1 * e_2$ and $e_1 * 2$ are two arithmetic expressions that are *not* semantically equivalent for some arithmetic expressions $e_1, e_2 \in \mathcal{A}$.

(a) Prove that e_2 cannot be semantically equivalent to 2.

(b) Find concrete examples of expressions $e_1, e_2 \in \mathcal{A}$ such that $e_1 * e_2$ and $e_1 * 2$ are not semantically equivalent but where there exists a state $\sigma \in \text{State}$ such that $[\![e_2]\!]_{\mathcal{A}}(\sigma) = 2$.

Solution

(a) There exists a state $\sigma \in \text{State}$ such that $[\![e_1]\!]_{\mathcal{A}}(\sigma) \cdot [\![e_2]\!]_{\mathcal{A}}(\sigma) \neq [\![e_1]\!] \cdot 2$. Therefore, there exists a state in which $[\![e_2]\!]_{\mathcal{A}}(\sigma) \neq 2$. Thus, e_2 cannot be semantically equivalent to 2.

(b) There are many possible answers, but the point is that even though e_2 cannot be equivalent to 2 there may still be states in which it evaluates to 2. For example, if $e_1 = 1$ and $e_2 = x$, then we have that both $e_1 * e_2$ and $e_1 * 2$ evaluate to 2 in the state $\sigma = [x \mapsto 2]$.

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

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

An expression $\mathsf{let}\ x = e_1\ \mathsf{in}\ e_2$ 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 $\mathsf{let}\ x = 2\ \mathsf{in}\ x + y$ when evaluated in the state $[x \mapsto 3, y \mapsto 2]$ should be 4.

Extend the definition of the denotation function $[\![\cdot]\!]_{\mathcal{A}}$ with an equation for this construct. You may find it useful to use the notation $\sigma[x \mapsto n]$ to represent the state σ updated such that $(\sigma[x \mapsto n])(x) = n$ and $(\sigma[x \mapsto n])(y) = \sigma(y)$ for all $y \neq x$. You do not need to change any other equations.

Solution

The required equation is:

$$[\![\mathsf{let}\ x = e_1\ \mathsf{in}\ e_2]\!]_{\mathcal{A}}(\sigma) = [\![e_2]\!]_{\mathcal{A}}(\sigma[x \mapsto [\![e_1]\!]_{\mathcal{A}}(\sigma)])$$

The key things to note are that the state is updated according to the value of e_1 as specified and that the equation is recursive in that the denotation for new construct is defined in terms of the denotation of its sub-expressions.

** 11. Now let us suppose we extend the language of arithmetic expressions with a different operator:

$$A \to x \mid n \mid \cdots \mid B\,?\,A : A$$

An instance of this *ternary operator* $e_1 ? e_2 : e_3$ for some Boolean expression $e_1 \in \mathcal{B}$ and arithmetic expressions $e_2, e_3 \in \mathcal{A}$ behaves as e_2 in states where e_1 is true and behaves as e_3 otherwise.

Extend the definition of the denotation function $[\![\cdot]\!]_{\mathcal{A}}$ with an equation for this construct. Your answer may make reference to the denotation function for Boolean expressions.

Solution

The required equation is:

$$[\![e_1 \mathbin{?} e_2 : e_3]\!]_{\mathcal{A}}(\sigma) = \begin{cases} [\![e_2]\!]_{\mathcal{A}}(\sigma) & \text{if } [\![e_1]\!]_{\mathcal{B}}(\sigma) \\ [\![e_3]\!]_{\mathcal{A}}(\sigma) & \text{otherwise} \end{cases}$$

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

$$\begin{aligned} \mathsf{FV}(x) &= \{x\} \\ \mathsf{FV}(n) &= \emptyset \\ \mathsf{FV}(e_1 + e_2) &= \mathsf{FV}(a) \cup \mathsf{FV}(b) \\ \mathsf{FV}(e_1 - e_2) &= \mathsf{FV}(a) \cup \mathsf{FV}(b) \\ \mathsf{FV}(e_1 * e_2) &= \mathsf{FV}(a) \cup \mathsf{FV}(b) \end{aligned}$$

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

(b) Consider the denotation $[\![x+1]\!]_{\mathcal{A}}(\sigma)$ 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 $[\![e]\!]_{\mathcal{A}}(\sigma)$ and $[\![e]\!]_{\mathcal{A}}(\sigma')$ are the same. How you could make this argument formal?

Solution

(a) The free variables of the expression $x + 1$ are just $\{x\}$.

(b) The denotation of an expression doesn't change when variables that do not appear in the expression are changed. The denotation $[\![x+1]\!]_{\mathcal{A}}(\sigma)$ equates to $\sigma(x) + 1$ and thus does not depend on $\sigma(y)$.

(c) To prove this formally one must use proof by induction, that we will cover in the second lecture of week 5. Informally, however, one can inspect each of the defining equations of the denotation function and see that they only depend on the free variables or sub-expressions which themselves will only use a subset of the free variables.