CIS 500 Software Foundations

Homework Assignment 1

Induction; Operational Semantics

Due: Monday, September 20, 2004, by noon

Submission instructions:

This assignment should be done together with your *study group.* Only one assignment should be submitted per group, and it does not matter which group member submits the assignment. However, make sure that all group members' names are listed on the submission. These members will be in your group for the entire semester, unless there are special circumstances. If you receive help from someone outside of your study group, excluding the course staff, you must also acknowledge them on your assignment.

Solutions must be submitted electronically (in ascii, postscript, or PDF format). Follow the instructions at `http://www.seas.upenn.edu/~cis500/homework.html.`

1 Exercise Consider this simple grammar:

```
t ::= One
      Zero
      OneAnd t
      ZeroAnd t
```

Terms `t` can be thought of as representing binary integers abstractly. For example, the representation of 5 (i.e., `101` in binary) is `OneAnd (ZeroAnd One)`.

State the structural induction principle for terms `t`.

2 Exercise Using the big-step operational semantics for the Arith language (listed at the end of the assignment), we can show that

$$\texttt{if (if (iszero(succ 0)) then false else true) then succ(if true then 0 else (succ 0)) else 0} \Downarrow \texttt{succ 0}$$

Give a derivation witnessing this fact, noting the rule applied at each step. (Hint: you might want to start by drawing the *parse tree* for the term on the left-hand side first.)

3 Exercise 3.5.13 in TAPL.

4 Exercise 3.5.17 in TAPL.

5 Exercise The operational semantics of the Arith language (listed at the end of the assignment) is not total. In other words, the following property does not hold: *For all* `t`*, there exists a* $\texttt{t}'$ *such that* $\texttt{t} \Downarrow \texttt{t}'$.

In this problem, we will fix that by introducing a new expression to be the result of evaluating terms like `succ false`. To do so, we add `wrong` to the syntax of the terms of Arith and introduce two new syntactic categories.

```
t ::= ...
   wrong
badnat ::=              non-numeric normal forms
   bv                       boolean values
   wrong                    run-time error
badbool ::=             non-boolean normal forms
   nv                       numeric values
   wrong                    run-time error
```

We also augment the large-step evaluation relation with the following new rules:

$$\texttt{wrong} \Downarrow \texttt{wrong} \qquad \text{B-Wrong}$$

$$\frac{\texttt{t}_1 \Downarrow \texttt{badbool}}{\texttt{if } \texttt{t}_1 \texttt{ then } \texttt{t}_2 \texttt{ else } \texttt{t}_3 \Downarrow \texttt{wrong}} \qquad \text{B-IfWrong}$$

$$\frac{\texttt{t}_1 \Downarrow \texttt{badnat}}{\texttt{succ } \texttt{t}_1 \Downarrow \texttt{wrong}} \qquad \text{B-SuccWrong}$$

$$\frac{\texttt{t}_1 \Downarrow \texttt{badnat}}{\texttt{pred } \texttt{t}_1 \Downarrow \texttt{wrong}} \qquad \text{B-PredWrong}$$

$$\frac{\texttt{t}_1 \Downarrow \texttt{badnat}}{\texttt{iszero } \texttt{t}_1 \Downarrow \texttt{wrong}} \qquad \text{B-IsZeroWrong}$$

Show that the operational semantics of the language with these additions is total. Although many of the parts of this proof may be similar to each other, for this first assignment, write each case out in full.

6 Exercise Recall the abstract syntax of "binary" numbers, from the first exercise.

```
t ::= One
      Zero
      OneAnd t
      ZeroAnd t
```

Consider the following inductive definition of a function that returns the natural number that a term represents:

$$\begin{array}{lcl} \mathit{number}(\texttt{Zero}) & = & 0 \\ \mathit{number}(\texttt{One}) & = & 1 \\ \mathit{number}(\texttt{ZeroAnd t}) & = & 2 \cdot \mathit{number}(\texttt{t}) \\ \mathit{number}(\texttt{OneAnd t}) & = & 2 \cdot \mathit{number}(\texttt{t}) + 1 \end{array}$$

This function can be viewed as a relation $\mathit{number} \subseteq (\mathit{Terms}, \mathbb{N})$, between terms and natural numbers. Here's an equivalent definition of the function, using inference rules:

$$\frac{}{(\texttt{Zero}, 0) \in \mathit{number}}$$

$$\frac{}{(\texttt{One}, 1) \in \mathit{number}}$$

$$\frac{(\texttt{t}, i) \in \mathit{number}}{(\texttt{ZeroAnd t}, 2 \cdot i) \in \mathit{number}}$$

$$\frac{(\texttt{t}, i) \in \mathit{number}}{(\texttt{OneAnd t}, 2 \cdot i + 1) \in \mathit{number}}$$

1. Define a function $\mathit{length}(.)$ that returns the length of a given term when viewed as a bitstring. For example it must be that:

$$\begin{array}{lcl} length(\texttt{Zero}) & = & 1 \\ length(\texttt{OneAnd (ZeroAnd (ZeroAnd One))}) & = & 4 \end{array}$$

As with $number(.)$, present $length(.)$ in two ways, with both the inductive definition and with inference rules.

2. Define a function $zeros(.)$ that returns the number of zeros that a given term contains when viewed as a bitstring. For example:

$$zeros(\texttt{ZeroAnd (OneAnd Zero)}) \quad = \quad 2$$

Similarly, provide both the inductive definition of the function as well as inference rules.

3. Using inference rules, define a relation $\texttt{t\#t}' \subseteq (Terms, Terms)$ such that:

$$\texttt{t\#t}' \Rightarrow zeros(\texttt{t}) = length(\texttt{t}')$$

There are many such relations – eg the empty relation is one of them! – but try to define the biggest relation you can. Prove that:

$$\texttt{t\#t}' \Rightarrow zeros(\texttt{t}) = length(\texttt{t}')$$

The following is **not** to be turned in. Depending on your definition of the relation above, you may or may not be able to prove that

$$zeros(\texttt{t}) = length(\texttt{t}') \Rightarrow \texttt{t\#t}'$$

You might want to try to prove this for the relation you came up with before, using induction on the pair of terms $(\texttt{t}, \texttt{t}')$. More details will be provided in the homework solutions.

7 Debriefing

1. How many hours did you spend on this assignment?
2. Would you rate it as easy, moderate, or difficult?
3. Did everyone in your study group participate?
4. How deeply do you feel you understand the material it covers (0%–100%)?

If you have any other comments, we would like to hear them; please send them `cis500@cis.upenn.edu`.

The Arith Language

Syntax

```
t ::=
   true
   false
   0
   succ t
   pred t
   iszero t
   if t then t else t
```

Big-step operational semantics

$$\mathtt{v} \Downarrow \mathtt{v} \qquad \text{B-Value}$$

$$\frac{\mathtt{t}_1 \Downarrow \mathtt{true} \qquad \mathtt{t}_2 \Downarrow \mathtt{v}}{\mathtt{if}\ \mathtt{t}_1\ \mathtt{then}\ \mathtt{t}_2\ \mathtt{else}\ \mathtt{t}_3 \Downarrow \mathtt{v}} \qquad \text{B-IfTrue}$$

$$\frac{\mathtt{t}_1 \Downarrow \mathtt{false} \qquad \mathtt{t}_3 \Downarrow \mathtt{v}}{\mathtt{if}\ \mathtt{t}_1\ \mathtt{then}\ \mathtt{t}_2\ \mathtt{else}\ \mathtt{t}_3 \Downarrow \mathtt{v}} \qquad \text{B-IfFalse}$$

$$\frac{\mathtt{t}_1 \Downarrow \mathtt{nv}}{\mathtt{succ}\ \mathtt{t}_1 \Downarrow \mathtt{succ}\ \mathtt{nv}} \qquad \text{B-Succ}$$

$$\frac{\mathtt{t}_1 \Downarrow \mathtt{succ}\ \mathtt{nv}}{\mathtt{pred}\ \mathtt{t}_1 \Downarrow \mathtt{nv}} \qquad \text{B-Pred}$$

$$\frac{\mathtt{t}_1 \Downarrow \mathtt{0}}{\mathtt{iszero}\ \mathtt{t}_1 \Downarrow \mathtt{true}} \qquad \text{B-IsZeroZero}$$

$$\frac{\mathtt{t}_1 \Downarrow \mathtt{succ}\ \mathtt{nv}}{\mathtt{iszero}\ \mathtt{t}_1 \Downarrow \mathtt{false}} \qquad \text{B-IsZeroSucc}$$

Solutions

Some solutions can be found in the back of the book. You should write out your answers to the assignment before looking. Note, however, that the solution to 3.5.13 is incorrect there – see
`http://www.cis.upenn.edu/~bcpierce/tapl/errata.txt`.