

PhD Qualifying exam on Programming Languages

Adriana Compagnoni

September 2004

This is an open-books and open-notes exam.

The language EXP is defined by the following grammar:

$$\begin{aligned}
 E & ::= true|false|N \\
 & \quad | E + E | E - E | E \wedge E | E \vee E | not E \\
 & \quad | if E then E else E | (E)
 \end{aligned}$$

where N is a countable set of numeric constants, and we will say $n \in N$ to express the fact that n is an element of the set N .

The typed language T-EXP consists of adding types Bool and Num to EXP expressions with the following axioms and rules:

$$\begin{array}{ll}
 true : Bool & (<T-BOOL>) \\
 false : Bool & (<F-BOOL>) \\
 \frac{n \in N}{n : Num} & (<N-NUM>) \\
 \frac{e : Num \quad f : Num}{e + f : Num} & (<+-NUM>) \\
 \frac{e : Num \quad f : Num}{e - f : Num} & (<--NUM>) \\
 \frac{e : Bool \quad f : Bool}{e \wedge f : Bool} & (<\wedge-BOOL>) \\
 \frac{e : Bool \quad f : Bool}{e \vee f : Bool} & (<\vee-BOOL>) \\
 \frac{e : Bool}{not e : Bool} & (<NOT-BOOL>) \\
 \frac{e : Bool \quad f : T \quad g : T}{if e then f else g : T} & (<IF-T>) \\
 \frac{e : T}{(e) : T} & (<()-T>)
 \end{array}$$

Observation: How to read axioms and rules

Consider the following examples.

The axiom $\langle T\text{-BOOL} \rangle$ means that the constant *true* has type *Bool*.

The rule $\langle +\text{-NUM} \rangle$ means that if expression *e* has type *Num* and expression *f* has type *Num*, then expression $e + f$ has type *Num* as well.

This notation allows us to justify a statement of the form

e has type T

constructing the derivation tree of a statement of the form

$e : T$

in the language T-EXP.

For example, to show that

$(3 + 4) - 1$ has type *Num*

we construct a derivation tree ending with conclusion:

$(3 + 4) - 1 : \text{Num}$

as follows:

$$\begin{array}{c}
 \begin{array}{cc}
 3 \text{ in } N & 4 \text{ in } N \\
 \langle n\text{-Num} \rangle \text{ -----} & \langle n\text{-Num} \rangle \text{ -----} \\
 3 : N & 4 : N \\
 \langle +\text{-Num} \rangle \text{ -----} & \\
 3+4 : \text{Num} & \\
 \langle ()\text{-T} \rangle \text{ -----} & \langle n\text{-Num} \rangle \text{ -----} \\
 (3+4) : \text{Num} & 1 \text{ in } N \\
 \langle --\text{Num} \rangle \text{ -----} & \\
 & 1 : N \\
 & (3+4)-1 : \text{Num}
 \end{array}
 \end{array}$$

On the other hand,

$6 \wedge \text{true} : \text{Num}$

is not derivable in T-EXP, because the rule $\langle \wedge\text{-BOOL} \rangle$, the only rule to derive a statement of the form $e \wedge f : T$, requires T to be *Bool*.

Exercises

1. Show that the expression

if (7 + true) then (8 - false) else false

is in the language EXP. In other words, write a derivation of:

$E \rightarrow^* \textit{if (7 + true) then (8 - false) else false}$

2. Show that

if (7 + true) then (8 - false) else false

cannot be derived using the given typing rules of T-EXP. In other words, show that

if (7 + true) then (8 - false) else false

is an expression that cannot be given a type in the language T-EXP.

3. Write a derivation in T-EXP of

if (not false) then 4 else (9 + 1) : Num

4. Write a parser for EXP.
5. Implement a typechecker for T-EXP. You may use ML or SCHEME.

Good luck!