Type inferencing in ML

October 27, 2003

Outline

- What is type inferencing?

- How does ML infer types?

- Why infer types?

What is type inferencing?

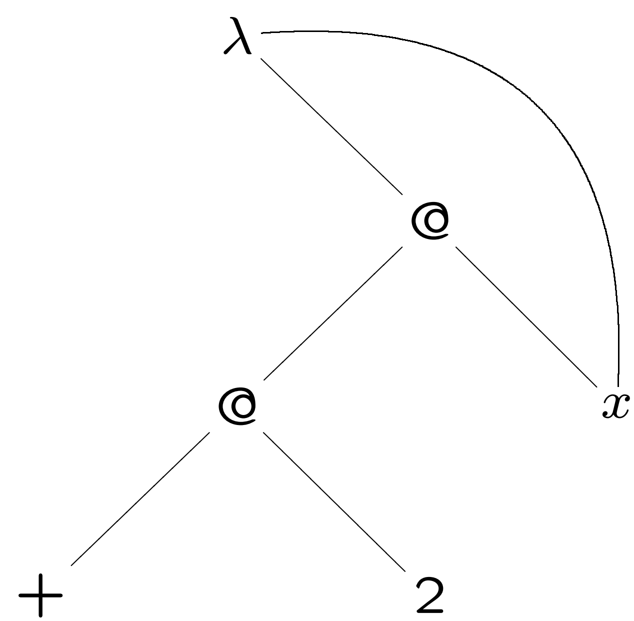
- When you determine the type of an object at compile/run-time without the user giving an explicit type to the object.

- Done using a semantically annotated parse-tree for the expression.

- Parse-tree is provided by compiler. Constructing the parse tree not important for us.

How does ML infer types?

We will use the following running example:



which is the annotated tree for $\lambda x.(2\ +\ x)$.

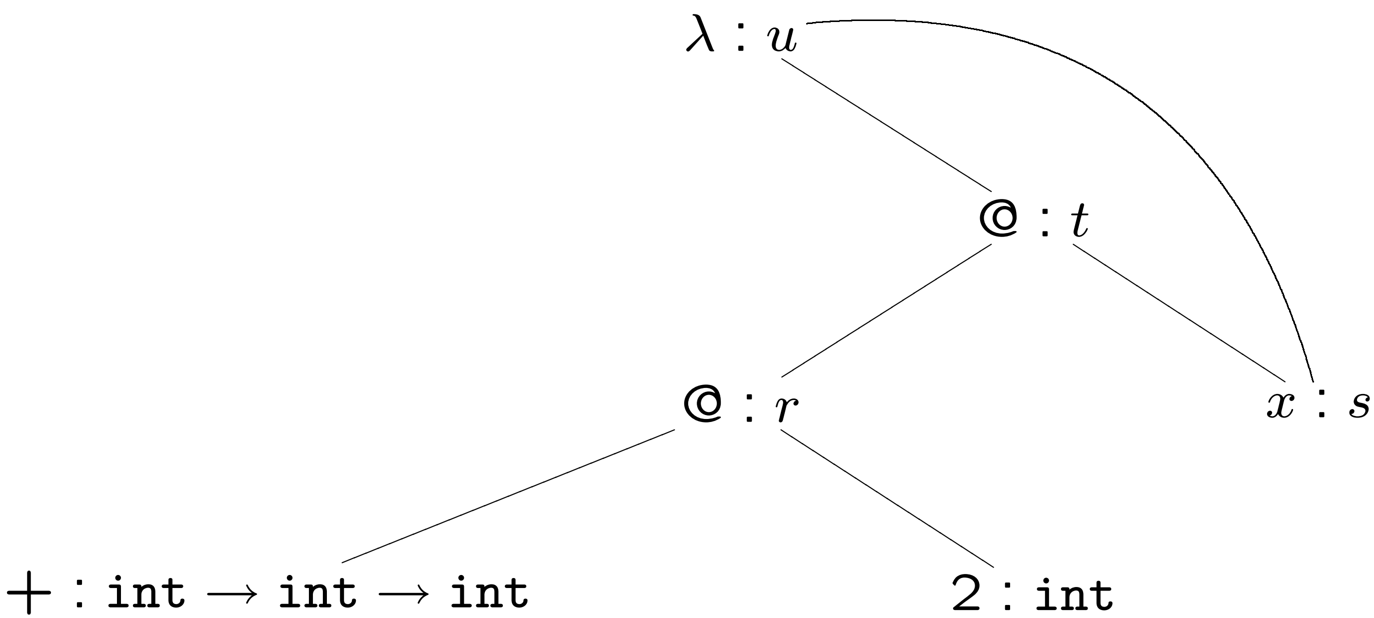
The type inferencing algorithm

1. The first step is to note all the information we know on the tree as well as assign type variables.

2. Next we build constraints on all the unknown types.

3. Finally we solve the constraints (through unification).

Assigning type variables

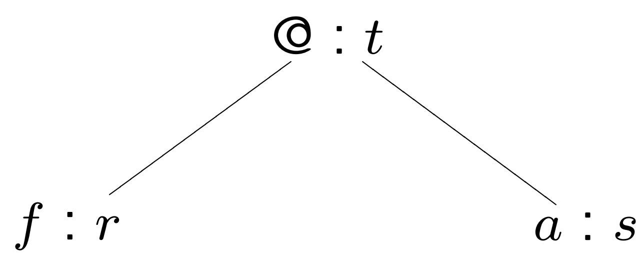


r, s, t, u are type variables.

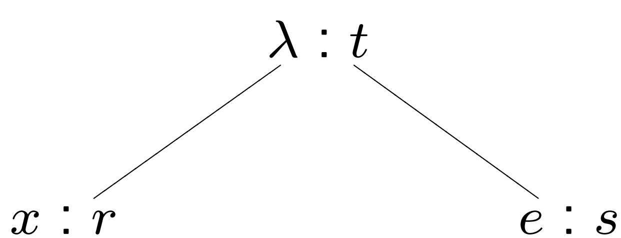
Building constraints

There are two kinds of interior nodes in an ML type-inferencing tree:

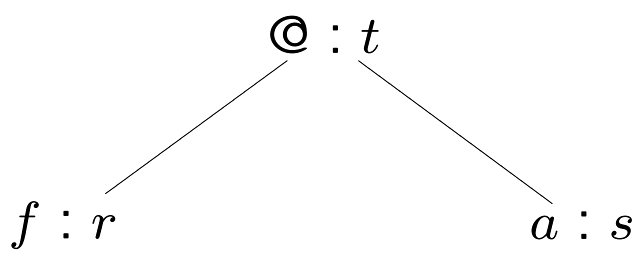
1. Application nodes:



2. λ nodes:

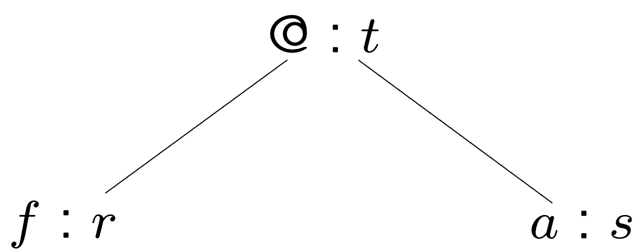


Application nodes



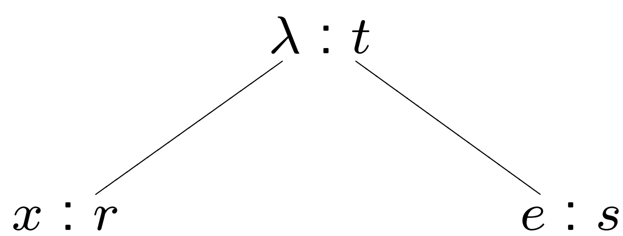
What is the relationship between r, s and t?

Application nodes



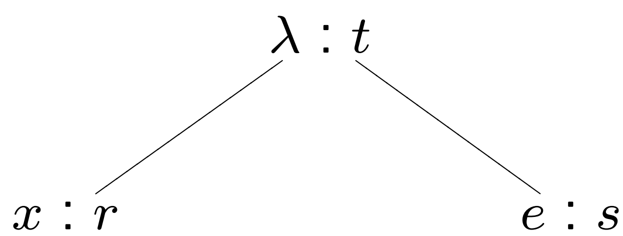
$$r = s \to t$$

Lambda nodes



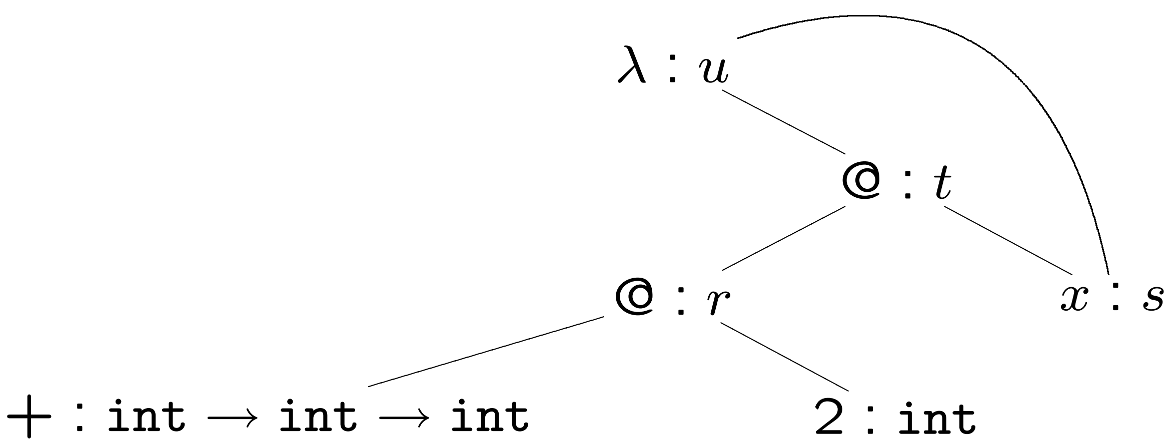
What is the relationship between r, s and t?

Lambda nodes



$$t = r \to s$$

Building the constraints



So 1. $\texttt{int} \to \texttt{int} \to \texttt{int} = \texttt{int} \to r$

2. $r = s \to t$

3. $u = s \to t$

Unifying

Our constraints are

1. $\texttt{int} \rightarrow \texttt{int} \rightarrow \texttt{int} = \texttt{int} \rightarrow r$
2. $r = s \rightarrow t$
3. $u = s \rightarrow t$

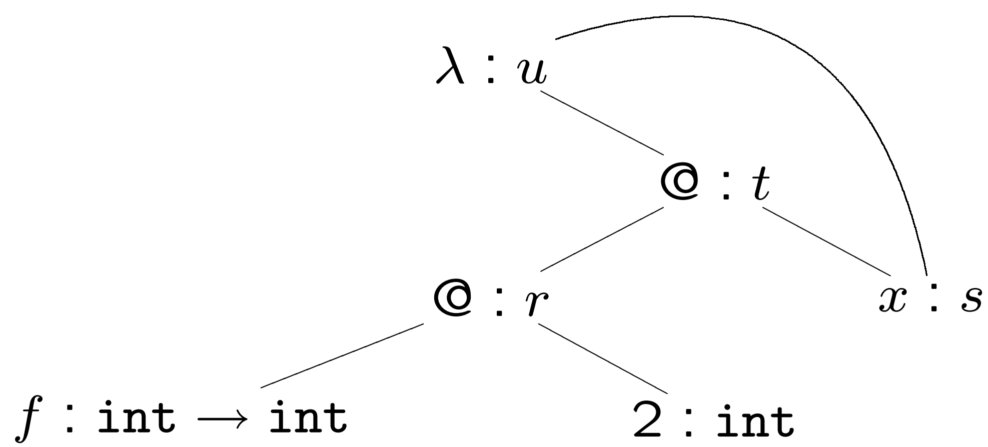
Therefore

4. $r = \texttt{int} \rightarrow \texttt{int}$ (from 1.)
5. $s = \texttt{int}$ (from 2. and 4.)
6. $t = \texttt{int}$ (from 2. and 4.)
7. $u = \texttt{int} \rightarrow \texttt{int}$ (from 3., 5. and 6.)

Thus, the type of the expression is $\texttt{int} \rightarrow \texttt{int}$ which is precisely the expected type of $\lambda x.(2\ +\ x)$.

Why do this?

Consider



Our constraints are

1. $\texttt{int} \rightarrow \texttt{int} = \texttt{int} \rightarrow r$
2. $r = s \rightarrow t$
3. $u = s \rightarrow t$

Cannot be unified since it implies that $\texttt{int} = s \rightarrow t$!