CSC 372, Spring 2025

SML Type Inference and Prolog Example Problems

Michelle Strout

THE UNIVERSITY OF ARIZONA

February 25, 2025

Plan

- **Announcements**
  - SA4 was due last night
  - Grades for ICA8 have been posted
  - Anki deck with prolog and type inference questions has been posted
  - LA2 will be posted tomorrow
- **Last time**
  - Some more Prolog
  - ICA8/Quiz8 about Prolog Introduction Reading assignment
  - Type inference for SML
- **Today**
  - Type inference for SML
  - Example Prolog problems

Outline for rest of today

- Type inference in SML
- Prolog example problems

SML Type Inference Rules

- **Constants**
  - `true, false`, type is `bool`
  - Integer literals (e.g. `42`), type is `int`
- **Variable use**
  - If x : tau is in the environment, then x has type tau
- **Lambda functions `(fn x => e)`**
  - If `x : tau1` and `e : tau2`, then `fn x => e : tau1 -> tau2`
- **Function application `(e1 e2)`**
  - If `e1 : tau1 -> tau2` and `e2 : tau1`, then `e1 e2 : tau2`
- **Addition, or multiplication `(e1 + e2)`**
  - Both `e1` and `e2` must be `int`, and the result is `int`
- **If expression `(if e1 then e2 else e3)`**
  - If `e1 : bool, e2 : tau, e3 : tau,` **result is** `tau`
- **Let expression `(let val x = e1 in e2 end)`**
  - Infer `e1 : tau1`, then infer `e2` with `x : tau1` in environment

Using an AST to help guide the process

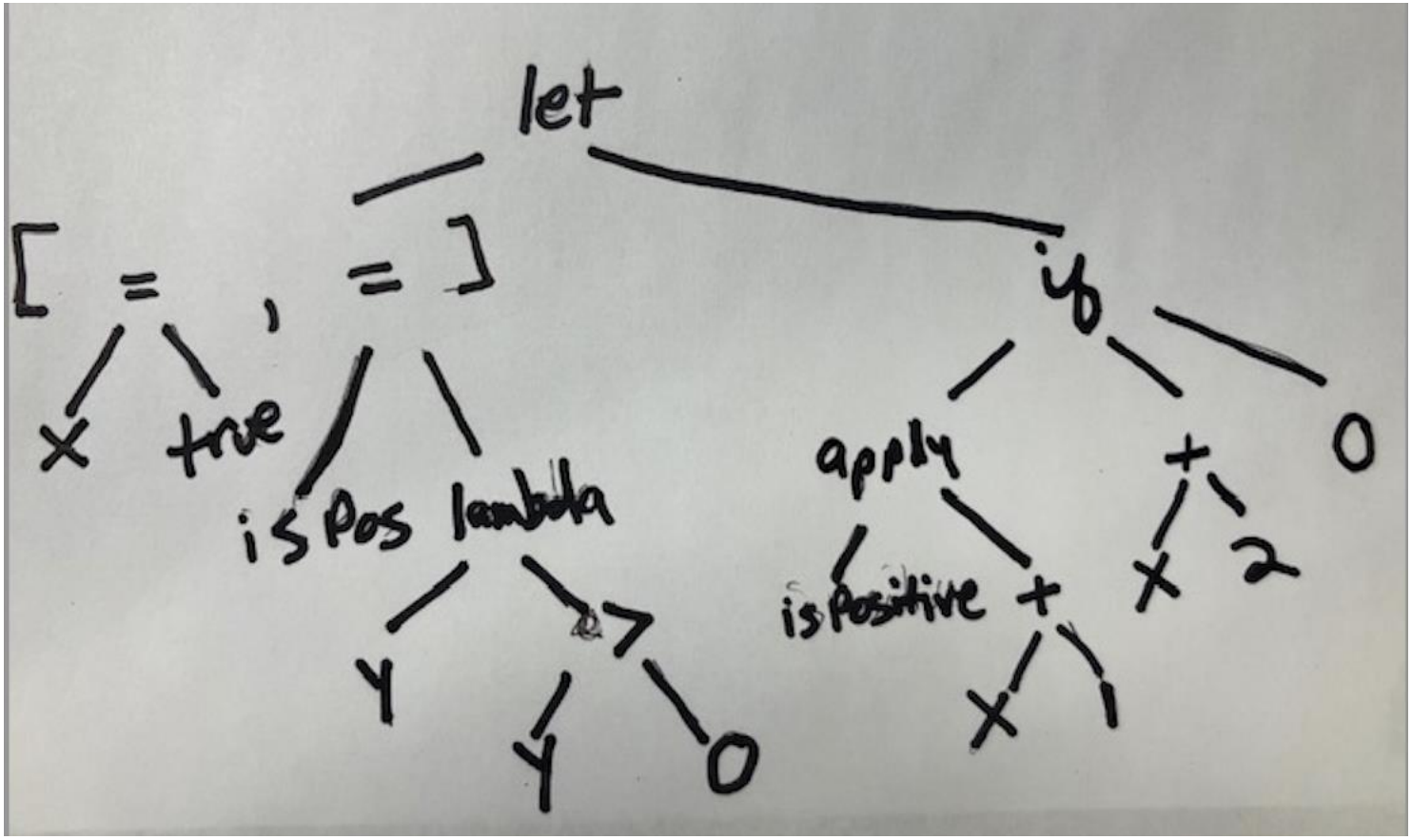
- Example

```
let
  val x = true
  val isPositive = fn y => y > 0
in
  if isPositive (x + 1) then x + 2 else 0
end
```

- Draw the AST for the example

- Apply type inference rules

AST from class



ASTs for parts of SML

- **Constants**
  - `bool(true)` **and** `bool(false)`, type is `bool`
  - `int(42), int(3), int(_)`, type is `int`
- **Variable use**
  - `var(x)`, If `(x,tau)` is in the environment, then `x` has type `tau`
- **Lambda functions `(fn x => e)`**
  - `lambda(x, E)`, E can be any other expression like `var(x)`
- **Function application `(e1 e2)`**
  - `apply(E1,E2)`
- **Addition, or multiplication `(e1 + e2)`**
  - `plus(E1,E2)` or `mult(E1,E2)`
- **If expression `(if e1 then e2 else e3)`**
  - `E=if(E1, E2, E3)`, **type of E1 must be bool, type of E2, E3, and E must all be the same (i.e., unify)**
- **Let expression `(let val x = e1 in e2 end)`**
  - `let(x, E1, E2)`, put `(x,tau)` into the environment

AST-based approach for earlier examples

- Earlier examples

```
fun square x = x * x;
fun baz f x = f (f x);

fun pairself x = (x,x)
```

Exercise.

Try the following queries and report the results. Don't forget to press ";" when there are multiple answers possible.

- `append([1,2,3],[4,5,6,7],X).`
- `append(X,[2,4],[0,2,4]).`
- `append([1,2],X,[1,2,3,4]).`
- `append(X,Y,[1,2,3,4]).`

What does `append(X,Y,Z)` mean?

Exercise.

Try the following queries and report the results. Don't forget to press ";" when there are multiple answers possible.

- `select(4, [1,2,4,3,5,4], X).`
- `select(1, X, [0,2,4]).`
- `select(X, [1,2,3,4], [1,3,4]).`
- `select(X, [1,2,3,4], Y).`

What does `select(X,Y,Z)` mean?

Exercise.

Try the following queries and report the results. Don't forget to press ";" when there are multiple answers possible.

- `nth0(0,[1,2,3,4],X).`
- `nth0(1,[1,2,3,4],X).`
- `nth0(5,[1,2,3,4],X).`
- `nth0(X,[1,2,3,1,2,3],2).`
- `nth0(4,X,4).`

What does `nth0(X,Y,Z)` mean?

Exercise.

Try the following queries and report the results. Don't forget to press ";" when there are multiple answers possible.

- `nth1(0,[1,2,3,4],X).`
- `nth1(1,[1,2,3,4],X).`
- `nth1(5,[1,2,3,4],X).`
- `nth1(X,[1,2,3,1,2,3],2).`
- `nth1(4,X,4).`

What does `nth1(X,Y,Z)` mean?

Exercise.

Try the following queries and report the results. Don't forget to press ";" when there are multiple answers possible.

- `length([1,2,3,4],X).`
- `length([],X).`
- `length([_],X).`
- `length(X,5).`

What does `length(X,Y)` mean?

Exercise.

Try the following queries and report the results. Don't forget to press ";" when there are multiple answers possible.

- `reverse([1,2,3,4],X).`
- `reverse(X,[0,2,4,6,7]).`
- `reverse(X,Y).`
- `reverse([2,3|X], Y).`

What does `reverse(X,Y)` mean?

Exercise.

Try the following queries and report the results. Don't forget to press ";" when there are multiple answers possible.

- `sort([2,6,1,7,8,9,0],X).`
- `sort(X,[1,2,3,4,5]).`

(a) What does `sort(X,Y)` mean?

(b) Explain why the second query causes an error.

Exercise.

Try the following queries and report the results. Don't forget to press ";" when there are multiple answers possible.

- `member(3,[1,2,3,4,5]).`
- `member(X,[1,2,3,4,5]).`
- `member(3,X).`

What does `member(X,Y)` mean?

Exercise.

Given the definition below, what does `foo(X,Y)` mean?

```
foo(0,1).

foo(X,Y) :- X > 0, X1 is X-1, foo(X1,Y1), Y is
X*Y1.
```

Problem-solving in Prolog

Note: Answers to these will not be posted in these slides. Posting the answers in piazza is encouraged and will be rewarded with extra credit points.

Remember: Prolog is *declarative*.

Instead of thinking about *how* to find a solution, we need to think about *what* a solution looks like.

And then we let Prolog do the rest.

Other reminders

- Everything is unification (i.e. pattern matching).
- You do not have to specify when predicates should fail. You only need to specify what makes them true.

Man-Wolf-Goat-Cabbage

Problem Definition

A man has a wolf, a goat, and a cabbage. He also has a boat that can carry at most himself and one of his items at a time. He and all his items are on the west side of a river, and he needs to get them all over to the east side of the river. However, if at any point he leaves the wolf and the goat together unsupervised, the wolf will eat the goat. Similarly, if he ever leaves the goat and the cabbage together unsupervised, the goat will eat the cabbage. How can he get everyone safely across the river?

What does a solution to this problem look like?

The Knapsack Problem

Problem Definition

We start with a list of items, each with a *weight* and a *value*. We also have a knapsack that has a specific *weight limit*. The problem is to determine *the subset of items* that we can carry in the knapsack (without exceeding the weight limit) that will *maximize the total value*.

Example.

Here we see an example of the 0-1 knapsack problem, which means that each item can either be in (1) or out (0).

Let's say the knapsack can hold up to 5 kg, and we can choose from the following items:

- rock (w = 2 kg, v = $2)
- book (w = 1 kg, v = $4)
- diamond (w = 5 kg, v = $5)
- sandwich (w = 3 kg, v = $3)

The Decision Version

We start with a list of items, each with a *weight* and a *value*. We also have a knapsack that has a specific *weight limit*. The problem is to determine *if a subset of items exists* that fits within the weight limit and also has a total value that is greater than or equal to some threshold *V*.

What does a solution to this problem look like?

The 8-Queens Problem

Problem Definition

Given an 8X8 chessboard, find a way to place 8 Queens on the board so that none of them is holding any of the others in *check*. This means that no two pairs of Queens can be in the same row, in the same column, or in the same diagonal.

What does a solution to this problem look like?