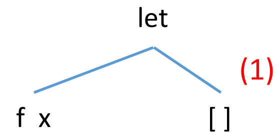
Hindley-Milner Type Inference

CSE340 Fall 2019

HW 5 Solution

Problem 1. `let f x = []`



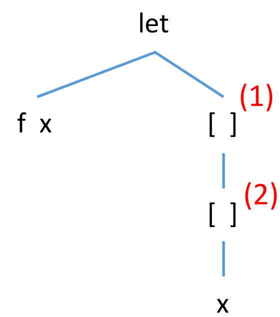
Visiting (1): $T_{(1)}$ = T List where T is not constrained.

$T_f = T_x \text{ -> } T_{(1)} = T_x \text{ -> } T\ list$

T_x is not constrained because there is no constraint that involves T_x

T is not constrained

Problem 2. `let  f x = [[x]]`



Visiting (1): $T_{(1)} = T_{(2)}$ list

Visiting (2): $T_{(2)} = T_x$ list

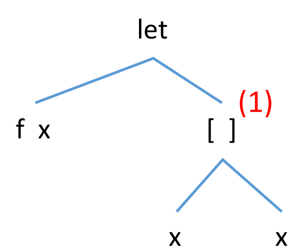
So, $T_{(1)} = (T_x$ list) list

There is no constraint involving T_x, so

$T_f = T_x \text{ -> } T_{(1)} = T_x \text{ -> } (T_x\ list)\ list$

T_x is not constrained

Problem 3. `let f x = [ x ; x ]`



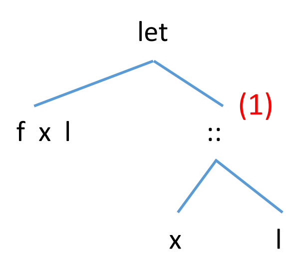
Visiting (1): $T_{(1)} = T_x$ list

$T_x = T_x$ ✔

$T_f = T_x \text{ -> } T_{(1)} = T_x \text{ -> } T_x\ list$

T_x is not constrained. The only constraint involving T_x is $T_x = T_x$ which is always satisfied for any type

Problem 4. `let f x l = x::l`



Visiting (1): $T_{(1)} = T_x$ list

$T_l = T_x$ list

T_x is not constrained

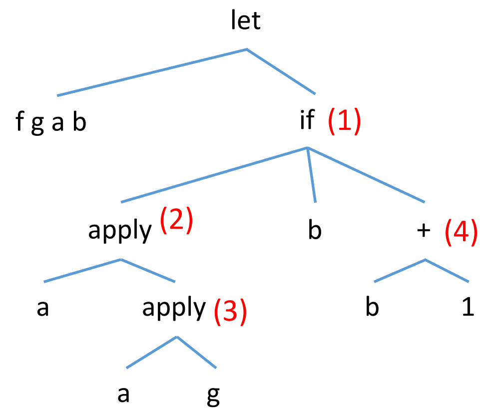
So,

$T_f = T_x \text{ -> } T_l \text{ -> } T_{(1)} = T_x \text{ -> } T_x\ list \text{ -> } T_x\ list$

T_x is not constrained

$T_l = T_x$ list

Problem 5. `let f g a b = if a (a g) then b else b + 1`



Visiting (4): $T_{(4)} = T_b$ = type of 1 = int

Visiting (1): $T_{(1)} = T_b = T_{(4)}$ = int

$T_{(2)}$ = bool

Visiting (2): $T_a = T_{(3)} \rightarrow T_{(2)} = T_{(3)} \rightarrow$ bool

Visiting (3): $T_a = T_g \rightarrow T_{(3)}$, but we know that

$T_a = T_{(3)} \rightarrow$ bool, so

$T_g = T_{(3)}$ and $T_{(3)}$= bool, it follows

that $T_g = T_{(3)}$ = bool, and T_a= bool -> bool

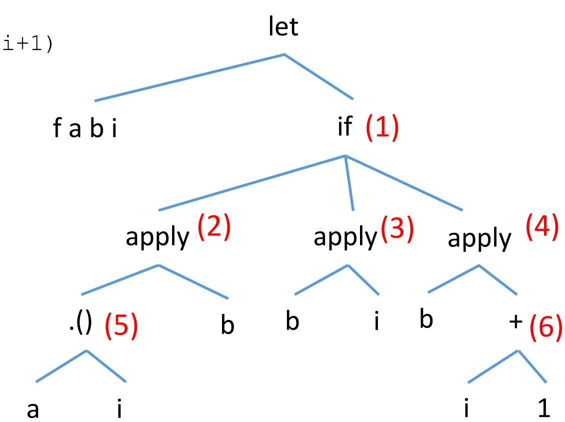
T_a= bool -> bool

T_g = bool

T_g = int

$T_f = T_g \rightarrow T_a \rightarrow T_b \rightarrow T_{(1)}$ = bool -> (bool -> bool) -> int -> int

Problem 6. `let f a b i = if a.(i) b then b i else b (i+1)`



Visiting (6): $T_{(6)} = T_i$ = type of 1 = int

Visiting (4): $T_b = T_{(6)} \rightarrow T_{(4)}$ = int -> $T_{(4)}$

Visiting (3): $T_b = T_i \rightarrow T_{(3)}$ = int -> $T_{(3)}$

it follows that $T_{(3)} = T_{(4)}$

Visiting (1): $T_{(1)} = T_{(3)} = T_{(4)}$

$T_{(2)}$ = bool

$T_{(3)} = T_{(4)}$ is consistent with

constraint above (established when visiting (3))

Visiting (2): $T_{(5)} = T_b \rightarrow T_{(2)}$ = (int -> $T_{(4)}$) -> bool

Visiting (5): $T_a = T_{(5)}$ array and T_i = int

so T_a = ((int -> $T_{(4)}$) -> bool) array

T_a = (int -> $T_{(4)}$) -> bool array

T_b = int -> $T_{(4)}$

T_i = int

$T_f = T_a \rightarrow T_b \rightarrow T_i \rightarrow T_{(1)}$ = ((int -> $T_{(4)}$) -> bool) array -> (int -> $T_{(4)}$) -> int -> $T_{(4)}$

where $T_{(4)}$ is not constrained

Problem 7. `let f a b c i = if a c then (if b then i else i+1) else c`

Visiting (4): $T_{(4)}$ = T_i = type of 1 = int

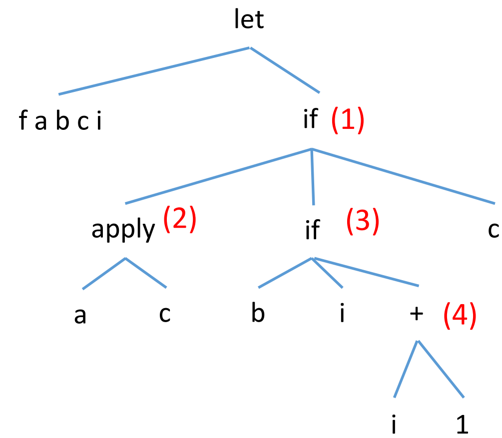
Visiting (3): $T_{(3)}$ = T_i= $T_{(4)}$ = int

T_b = bool

Visiting (1): $T_{(1)}$ = T_c= $T_{(3)}$ = int

$T_{(2)}$ = bool

Visiting (2): T_a = T_c -> $T_{(2)}$ = int -> bool



T_a = int -> bool

T_b = bool

T_c = int

T_i = int

T_f = T_a -> T_b -> T_c -> T_i -> $T_{(1)}$ = (int -> bool) -> bool -> int -> int -> int

Problem 8.
```
let rec max l = match l with
      []      -> None   (* max is not defined for empty list *)
    | h::l1   -> if h > max l1 then h else max l1
```

Visiting (2): $T_{(2)}$ = T1 list

where T1 is unconstrained

Visiting (4): $T_{(4)}$ = T2 option

where T2 is unconstrained

Visiting (1): T_l= $T_{(1)}$ = $T_{(2)}$ = $T_{(3)}$ = T1 List

$T_{(4)}$ = $T_{(5)}$ = T2 option

Visiting (3): $T_{(3)}$ = T_{l1} = T_h list

so T_h = T1

Visiting (5): $T_{(5)}$ = T_h = $T_{(7)}$ = T2 option

$T_{(6)}$ = bool

so **T1 = T2 option**

T_l= = T1 List = T2 option list

Visiting (6): T_h = $T_{(8)}$

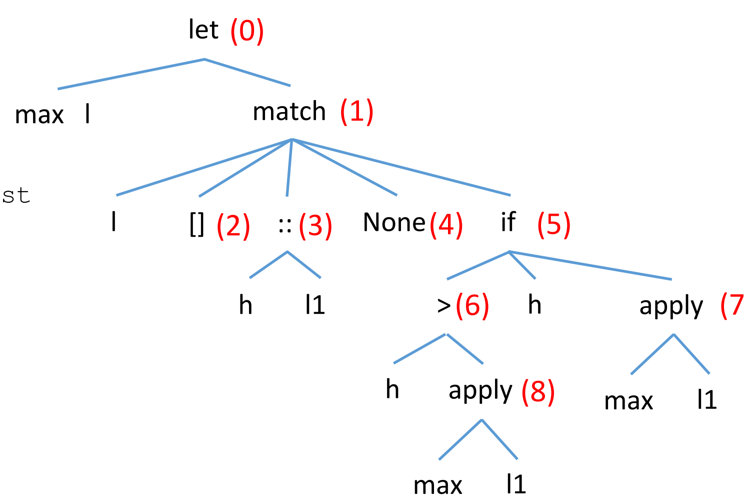
$T_{(6)}$ = bool ✔

so $T_{(8)}$ = T2 option

Visiting (8): T_{max} = T_{l1} -> $T_{(8)}$ = T2 option list -> T2 option

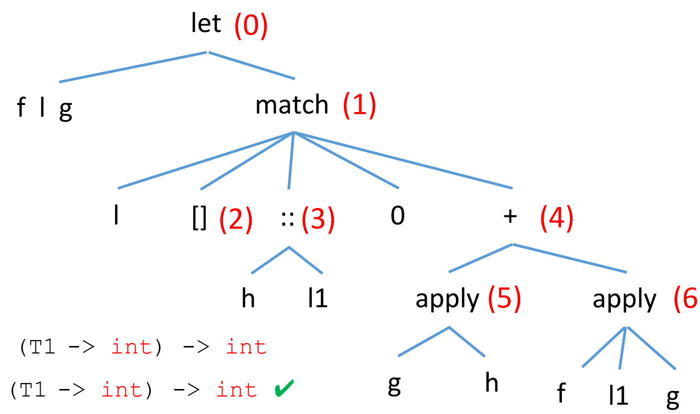
Visiting (8): T_{max} = T_{l1} -> $T_{(7)}$ = T2 option list -> T2 option ✔

Visiting (0): T_{max} = T_l -> $T_{(1)}$ = T2 option list -> T2 option ✔



T_l = T2 option list and T_{max} = T2 option list -> T2 option

Problem 9.
```
let rec f l g   = match l with (* this is a map-reduce function *)
           []       -> 0
         | h::l1    ->  (g h) + (f l1 g)     (* apply g to the head of l and f
                                              * to l1 (and g) and add the
                                              * results. The net effect is to
                                              * apply a to all the elements
                                              * and to sum the results up
                                              *)
```



Visiting (2): $T_{(2)}$ = T1 list

where T1 is unconstrained

Visiting (1): $T_l = T_{(2)} = T_{(3)}$ = T1 list

$T_{(4)}$ = type of 0 = int

Visiting (3): $T_{(3)} = T_{l1} = T_h$ list

so T_h = T1

Visiting (4): $T_{(5)} = T_{(6)} = T_{(4)}$ = int

Visiting (5): $T_g = T_h \rightarrow T_{(5)}$ = T1 -> int

Visiting (6): $T_f = T_{l1} \rightarrow T_g \rightarrow T_{(6)}$ = T1 List -> (T1 -> int) -> int

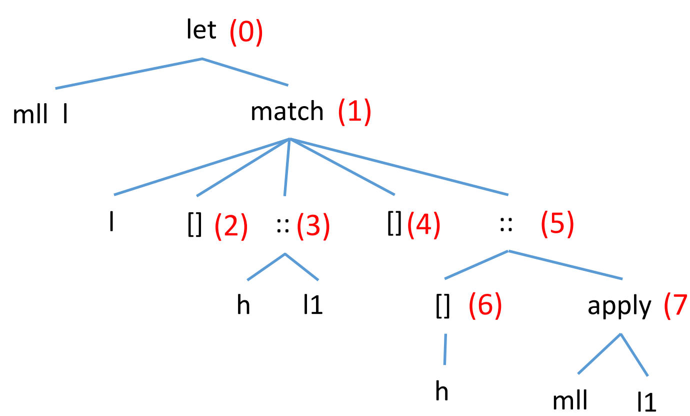
Visiting (0): $T_f = T_l \rightarrow T_g \rightarrow T_{(1)}$ = T1 list -> (T1 -> int) -> int ✔

T_g = T1 -> int

T_l = T1 List

T_f = T1 List -> (T1 -> int) -> int

Problem 10.
```
let rec mll l = match l with
                    []       -> []
                  | h::l1    -> [h]::(mll l1)
```



Visiting (2): $T_{(2)}$ = T1 list

where T1 is unconstrained

Visiting (3): $T_{(3)} = T_{l1} = T_h$ list

where T_h is not constrained

Visiting (4): $T_{(4)}$ = T2 list

where T2 is unconstrained

Visiting (1): $T_l = T_{(2)} = T_{(3)} = T_h$ list

$T_{(5)} = T_{(4)}$ = T2 list

Visiting (6): $T_{(6)} = T_h$ list

Visiting (5): $T_{(5)} = T_{(7)} = T_{(6)}$ list = T_h list list

Visiting (7): $T_{mll} = T_{l1} = T_{(7)} = T_h$ list -> T_h list list

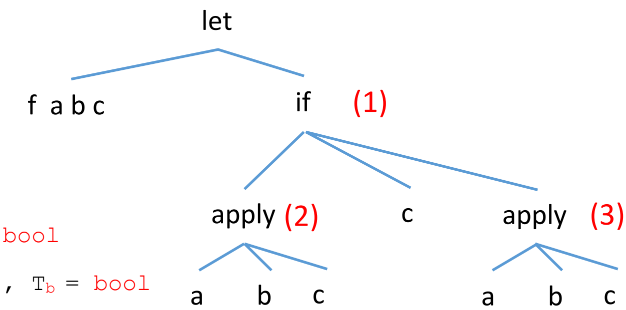
Visiting (0): $T_{mll} = T_l \rightarrow T_{(1)} = T_h$ list -> T_h list list ✔

$T_l = T_h$ list

$T_{mll} = T_h$ list -> T_h list list

where T_h is not constrained

Problem 11. let f a b c = if a b c then c else a c b



Visiting (1): $T_{(1)} = T_c = T_{(3)}$

$T_{(2)} = bool$

Visiting (2): $T_a = T_b \rightarrow T_c \rightarrow bool$

Visiting (3): $T_a = T_c \rightarrow T_b \rightarrow T_{(3)}$

It follows that $T_b = T_c$ and $T_c = T_b$ and $T_{(3)} = bool$

but $T_c = T_{(3)}$, so $T_c = bool$ and since $T_c = T_b$, $T_b = bool$

also, $T_{(1)} = T_c$, $T_{(1)} = bool$

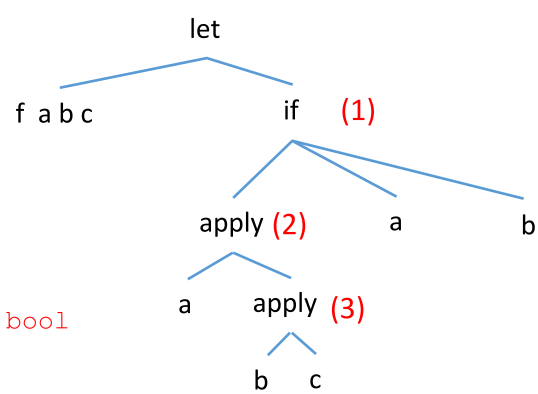
$T_a = bool \rightarrow bool \rightarrow bool$

$T_b = bool$

$T_c = bool$

$T_f = T_a \rightarrow T_b \rightarrow T_c \rightarrow T_{(1)} = (bool \rightarrow bool \rightarrow bool) \rightarrow bool \rightarrow bool \rightarrow bool$

Problem 12. let f a b c = if a (b c) then a else b



Visiting (1): $T_{(1)} = T_a = T_b$

$T_{(2)} = bool$

Visiting (2): $T_a = T_{(3)} \rightarrow T_{(2)} = T_{(3)} \rightarrow bool$

Visiting (3): $T_b = T_c \rightarrow T_{(3)}$

but $T_a = T_b$, so $T_c \rightarrow T_{(3)} = T_{(3)} \rightarrow bool$

It follows that $T_{(3)} = bool$ and $T_c = T_{(3)} = bool$

and $T_a = T_b = bool \rightarrow bool$

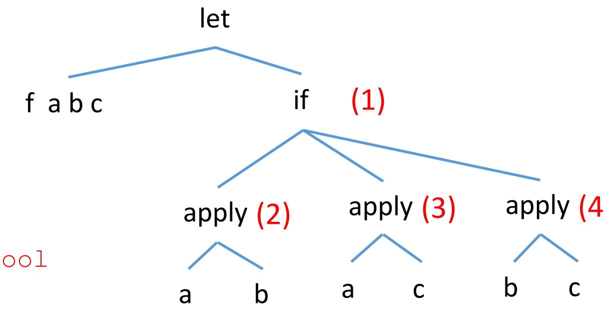
$T_a = bool \rightarrow bool$

$T_b = bool \rightarrow bool$

$T_c = bool$

$T_f = T_a \rightarrow T_b \rightarrow T_c \rightarrow T_{(1)} = (bool \rightarrow bool) \rightarrow (bool \rightarrow bool) \rightarrow bool \rightarrow bool \rightarrow bool$

Problem 13. `let f a b c = if a b then a c else b c`



Visiting (1): $T_{(1)} = T_{(3)} = T_{(4)}$

$T_{(2)} = bool$

Visiting (2): $T_a = T_b \rightarrow T_{(2)} = T_b \rightarrow bool$

Visiting (3): $T_a = T_c \rightarrow T_{(3)}$

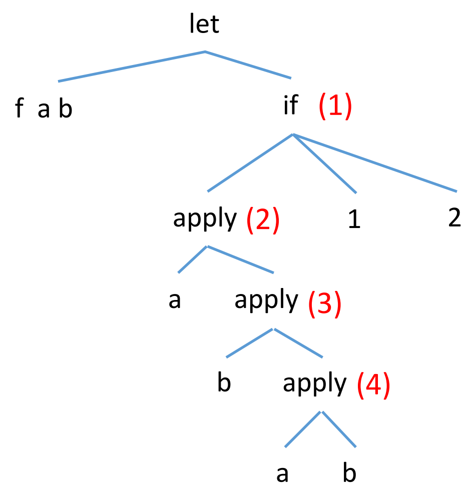
it follows that $T_c = T_b$ and $T_{(3)} = T_{(2)} = bool$

so $T_{(1)} = T_{(3)} = T_{(4)} = bool$

Visiting (4): $T_b = T_c \rightarrow T_{(4)} = T_c \rightarrow bool$

but $T_c = T_b$ so it follows that $T_b = T_b \rightarrow bool$ **TYPE MISMATCH**

Problem 14. `let f a b  = if  a (b (a b)) then 1  else 2`



Visiting (1): $T_{(1)}$ = type of 1 = type of 2 = int

$T_{(2)} = bool$

Visiting (2): $T_a = T_{(3)} \rightarrow T_{(2)} = T_{(3)} \rightarrow bool$

Visiting (4): $T_a = T_b \rightarrow T_{(4)}$

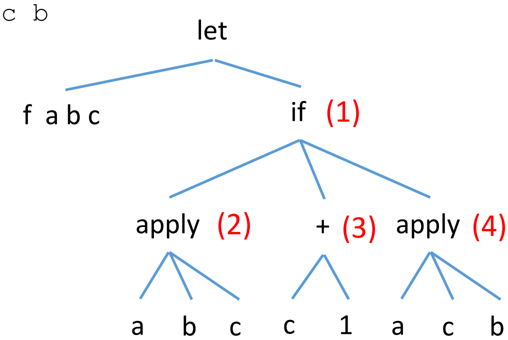
but $T_a = T_{(3)} \rightarrow bool$ and it follows that $\mathbf{T_b = T_{(3)}}$

and $T_{(4)} = bool$

Visiting (3): $T_b = T_{(4)} \rightarrow T_{(3)}$

so $T_b = bool \rightarrow T_b$ **TYPE MISMATCH**

Problem 15. `let f a b c = if  a b c then c + 1 else a c b`



Visiting (1): $T_{(1)} = T_{(3)} = T_{(4)}$

$T_{(2)} = bool$

Visiting (3): $T_{(3)} = T_c$ = Type of 1 = int

it follows that $T_{(4)} = int$

Visiting (2): $T_a = T_b \rightarrow T_c \rightarrow T_{(2)} = T_b \rightarrow int \rightarrow bool$

Visiting (4): $T_a = T_c \rightarrow T_b \rightarrow T_{(4)} = int \rightarrow T_b \rightarrow int$

It follows that $T_b = T_c = int$

and bool = int **TYPE MISMATCH**