CMPSC 461: Programming Language Concepts, Fall 2024

Assignment 7 Practice Sheet (Types)

Last updated: **November 16, 2024**

Problem 1: Type Inference

1) Given the following lambda expressions, determine the most general type that can be inferred or mention why an expression is not typeable:
   a) $f_1 = \lambda x.\ \lambda y.\ (x + 10) == y$
   b) $f_2 = \lambda x.\ \lambda y.\ \lambda z.\ (x * z\ y > 0)\ \&\&\ y$
   c) $f_3 = \lambda w.\ \lambda x.\ \lambda y.\ \lambda z \text{if } x\ z \geq y \text{ then } z \text{ else } w$
   d) $f_4 = \lambda x.\ \lambda y.\ \lambda z.\ (x\ z\ \&\&\ y\ z) + 1$

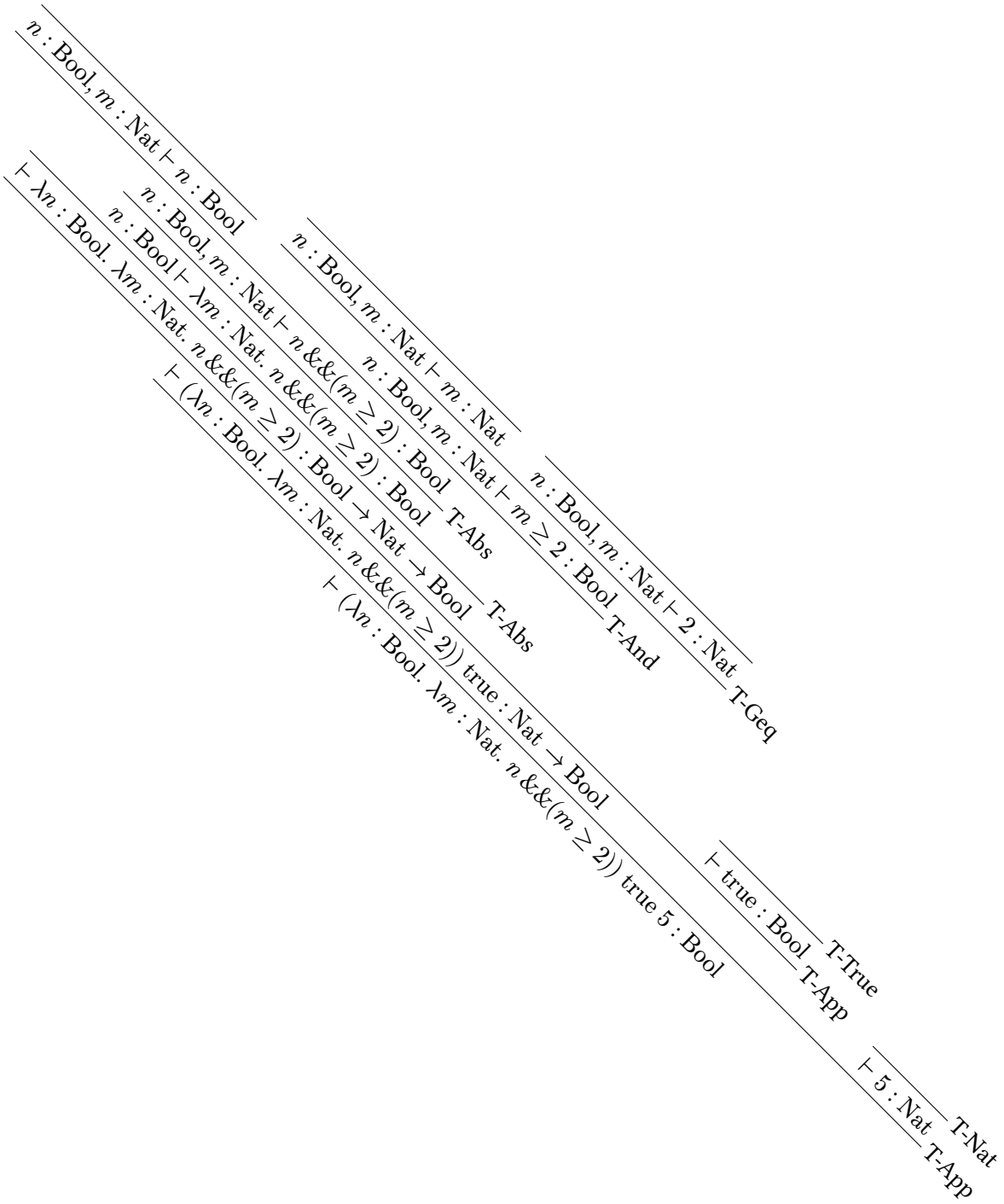
   Since we don't have polymorphic types, you may use capital letters to represent basic types.
2) What is the type of the following term?

$$(\lambda n : \text{Bool}.\ \lambda m : \text{Nat}.\ n\ \&\&(m \geq 2))\ \text{true}\ 5$$

   Justify your answer with the proof tree for the given term. You can refer to the typing rules in the Appendix. Make sure you mention which typing rules are you using at each step of the proof tree.

Solution

1) a) $\text{Nat} \to \text{Nat} \to \text{Bool}$
   b) $\text{Nat} \to \text{Bool} \to (\text{Bool} \to \text{Nat}) \to \text{Bool}$
   c) $A \to (A \to \text{Nat}) \to \text{Nat} \to A \to A$
   d) This is not typeable because we cannot add a boolean and a natural number.
2) T-Var is omitted for brevity.



Problem 2: Type system

1) Write down a typing rule (call it T-EQ) for the term $t_1 == t_2$, such that the equality test takes in two natrual numbers and returns a Boolean value.
2) What is the type of the following term?

$$(\lambda x : \text{Bool}.\ \lambda y : \text{Nat}.\ (y == (8 + 3))\ x)\ \text{true}$$

Justify your answer with the proof tree for the given term. You can refer to the typing rules in the Appendix. Make sure you mention which typing rules are you using at each step of the proof tree.

Solution

1) See T-Eq in appendix.
2) T-Var is omitted for brevity.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{x : \text{Bool}, y : \text{Nat} \vdash y : \text{Nat}} \quad \dfrac{\dfrac{}{\vdash 8 : \text{Nat}}\text{T-Nat} \quad \dfrac{}{\vdash 3 : \text{Nat}}\text{T-Nat}}{\vdash 8 + 3 : \text{Nat}}\text{T-Add}}{x : \text{Bool}, y : \text{Nat} \vdash y == (8 + 3) : \text{Bool}}\text{T-Eq}}{x : \text{Bool} \vdash \lambda y : \text{Nat}.\ (y == (8 + 3)) : \text{Nat} \to \text{Bool}}\text{T-Abs}}{\lambda x : \text{Bool}.\ \lambda y : \text{Nat}.\ (y == (8 + 3)) : \text{Bool} \to \text{Nat} \to \text{Bool}}\text{T-Abs} \quad \dfrac{}{\vdash \text{true} : \text{Bool}}\text{T-True}}{\vdash (\lambda x : \text{Bool}.\ \lambda y : \text{Nat}.\ (y == (8 + 3)))\ x\ \text{true} : \text{Nat} \to \text{Bool}}\text{T-App}$$

Problem 3: Type Inference

Given the following lambda expressions, determine the most general type that can be inferred or mention why an expression is not typeable:

1) $f_1 = \lambda x.\ x \geq 10$
2) $f_2 = \lambda x.\ \lambda y.\ \lambda z.\ (x\ z \geq 10) + \text{false}$
3) $f_3 = \lambda x.\ \lambda y.\ \lambda z.\ \text{if}\ (x\ z \geq y)\ \text{then true else false}$
4) $f_4 = \lambda x.\ \lambda y.\ \lambda z.\ (x\ z == 3)\ \&\&(y\ z == 0)$

Since we don't have polymorphic types, you may use capital letters to represent basic types.

Solution

1) $\text{Nat} \to \text{Bool}$
2) This is not typeable because we cannot add a boolean and a natural number.
3) $(A \to \text{Nat}) \to \text{Nat} \to A \to \text{Bool}$
4) $(A \to \text{Nat}) \to (A \to \text{Nat}) \to A \to \text{Bool}$

Problem 4: Types, Type Systems and Type Inference

1) Mention two differences between Strongly typed and Weakly typed programming languages.
2) Consider a simply typed lambda calculus and its typing rules. You can refer to the typing rules in the Appendix. Write down a typing rule (T-Star) for the new unary operator $\star$, so that operation $\star\ n$ returning false if n is less than 3, and true otherwise.
3) Using your solution for part 2, infer what is the type of the term

$$(\lambda s : \text{Nat}.\ \lambda z : \text{Bool}.\ (10 \geq n)\ \&\&\ \star\ s)\ 8$$

Justify your answer with the proof tree for the given term. Make sure you mention which typing rules are you using at each step of the proof tree.

Solution

1) In strongly typed languages, strict type enforcement is present, meaning variables are bound to specific data types and conversions between types typically require explicit casting. This helps catch type-related errors at compile-time, enhancing code reliability. In contrast, weakly typed languages allow more flexibility, often implicitly converting between types, which can sometimes lead to unexpected behavior or errors that are harder to detect during development.

2)
$$\frac{\Gamma \vdash t : \text{Nat}}{\Gamma \star t : \text{Nat} \to \text{Bool}}\ \text{T-Star}$$

3) T-Var is omitted for brevity. To avoid page overflow, let $\Delta = s : \text{Nat}, z : \text{Bool}$.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Delta \vdash 10 : \text{Nat}}\text{T-Nat} \quad \Delta \vdash s : \text{Nat}}{\Delta \vdash 10 \geq n : \text{Bool}}\text{T-Geq} \quad \dfrac{\Delta \vdash s : \text{Nat}}{\Delta \vdash \star\, s : \text{Bool}}\text{T-Star}}{\Delta \vdash (10 \geq s)\ \&\&\ \star\, s : \text{Bool}}\text{T-And}}{s : \text{Nat} \vdash \lambda z : \text{Bool}.\ (10 \geq s)\ \&\&\ \star\, s : \text{Bool} \to \text{Bool}}\text{T-Abs}}{\vdash \lambda s : \text{Nat}.\ \lambda z : \text{Bool}.\ (10 \geq s)\ \&\&\ \star\, s : \text{Nat} \to \text{Bool} \to \text{Bool}}\text{T-Abs} \quad \dfrac{}{\vdash 8 : \text{Nat}}\text{T-Nat}}{\vdash (\lambda s : \text{Nat}.\ \lambda z : \text{Bool}.\ (10 \geq s)\ \&\&\ \star\, s)\ 8 : \text{Nat} \to \text{Bool} \to \text{Bool}}\text{T-App}$$

Appendix

Simply Typed Lambda Calculus Extended with Booleans and Natural Numbers

Syntax

Terms

$t ::= x$	(variable)
$\mid \lambda x : \tau.\ t$	(abstraction)
$\mid t\ t$	(application)
$\mid$ true	(constant true)
$\mid$ false	(constant false)
$\mid$ if t then t else t	(conditional)
$\mid n$	(constant natrual number)
$\mid t + t$	(nat addition)
$\mid t\ \&\&\ t$	(bool and)
$\mid t \geq t$	(nat greater than or equal to)
$\mid t == t$	(nat equality)

Types

$\tau ::= \tau \to \tau$	(function type)
$\mid$ Bool	(boolean type)
$\mid$ Nat	(natural number type)

Typing

$$\frac{}{\Gamma \vdash n : \text{Nat}}\ \text{T-Nat}$$

$$\frac{}{\Gamma \vdash \text{true} : \text{Bool}}\ \text{T-True}$$

$$\frac{}{\Gamma \vdash \text{false} : \text{Bool}}\ \text{T-False}$$

$$\frac{x : \tau \in \Gamma}{\Gamma \vdash x : \tau}\ \text{T-Var}$$

$$\frac{\Gamma, x : \tau_1 \vdash t : \tau_2}{\Gamma \vdash \lambda x : \tau.\ t : \tau_1 \to \tau_2}\ \text{T-Abs}$$

$$\frac{\Gamma \vdash t_1 : \tau_1 \to \tau_2 \quad \Gamma \vdash t_2 : \tau_1}{\Gamma \vdash t_1\ t_2 : \tau_2}\ \text{T-App}$$

$$\frac{\Gamma \vdash t_1 : \text{Bool} \quad \Gamma \vdash t_2 : \tau \quad \Gamma \vdash t_3 : \tau}{\Gamma \vdash \text{if } t_1 \text{ then } t_2 \text{ else } t_3 : \tau}\ \text{T-If}$$

$$\frac{\Gamma \vdash t_1 : \text{Nat} \quad \Gamma \vdash t_2 : \text{Nat}}{\Gamma \vdash t_1 + t_2 : \text{Nat}}\ \text{T-Add}$$

$$\frac{\Gamma \vdash t_1 : \text{Bool} \quad \Gamma \vdash t_2 : \text{Bool}}{\Gamma \vdash t_1 \,\&\&\, t_2 : \text{Bool}}\ \text{T-And}$$

$$\frac{\Gamma \vdash t_1 : \text{Nat} \quad \Gamma \vdash t_2 : \text{Nat}}{\Gamma \vdash t_1 \geq t_2 : \text{Bool}}\ \text{T-Geq}$$

$$\frac{\Gamma \vdash t_1 : \text{Nat} \quad \Gamma \vdash t_2 : \text{Nat}}{\Gamma \vdash t_1 == t_2 : \text{Bool}}\ \text{T-Eq}$$