

# Type Inference

CS 152 (Spring 2022)

Harvard University

Thursday, March 24, 2022

# Today, we will learn about

- ▶ Type inference
- ▶ Constraint-based typing
- ▶ Unification

# Type annotations

$$\text{T-ABS} \frac{\Gamma, x:\tau \vdash e:\tau'}{\Gamma \vdash \lambda x:\tau. e:\tau \rightarrow \tau'}$$

# Type inference

- ▶ Infer (or reconstruct) the types of a program
- ▶ Example:  $\lambda a. \lambda b. \lambda c. \text{if } a (b + 1) \text{ then } b \text{ else } c$

# Example

$\lambda a. \lambda b. \lambda c. \text{if } a (b + 1) \text{ then } b \text{ else } c$

# Example

$\lambda a. \lambda b. \lambda c. \text{if } a (b + 1) \text{ then } b \text{ else } c$

$\lambda a:\mathbf{int} \rightarrow \mathbf{bool}. \lambda b:\mathbf{int}. \lambda c:\mathbf{int}. \text{if } a (b + 1) \text{ then } b \text{ else } c$

# Example

$$\lambda x. \lambda y. x$$

# Example

$$\lambda x. \lambda y. x$$
$$\lambda x : X. \lambda y : Y. x$$



# Example

$$\lambda x. \lambda y. x$$
$$\lambda x : X. \lambda y : Y. x$$
$$\lambda x : \mathbf{int}. \lambda y : \mathbf{int}. x$$

# Example

$$\lambda x. \lambda y. x$$
$$\lambda x : X. \lambda y : Y. x$$
$$\lambda x : \mathbf{int}. \lambda y : \mathbf{int}. x$$
$$\lambda x : \mathbf{int}. \lambda y : \mathbf{int} \rightarrow \mathbf{int}. x$$

# Example

$$\lambda x. \lambda y. x$$
$$\lambda x : X. \lambda y : Y. x$$
$$\lambda x : \mathbf{int}. \lambda y : \mathbf{int}. x$$
$$\lambda x : \mathbf{int}. \lambda y : \mathbf{int} \rightarrow \mathbf{int}. x$$

...

# Example

$$\lambda x. \lambda f. f x$$

# Example

$$\lambda x. \lambda f. f x$$
$$\lambda x : X. \lambda f : X \rightarrow Y. f x$$

# Constraint-based Type Inference

- ▶ *Type variables*  $X, Y, Z, \dots$   
placeholders for types.
- ▶ Judgment  $\Gamma \vdash e : \tau \triangleright C$ 
  - ▶ Expression  $e$  has type  $\tau$  provided every constraint in set  $C$  is satisfied.
  - ▶ Constraints are of the form  $\tau_1 \equiv \tau_2$ .

# Language

$$e ::= x \mid \lambda x:\tau. e \mid e_1 e_2 \mid n \mid e_1 + e_2$$
$$\tau ::= \mathbf{int} \mid X \mid \tau_1 \rightarrow \tau_2$$

# Inference rules

$$\text{CT-VAR} \frac{}{\Gamma \vdash x:\tau \triangleright \emptyset} x:\tau \in \Gamma$$

$$\text{CT-INT} \frac{}{\Gamma \vdash n:\mathbf{int} \triangleright \emptyset}$$



## Inference rules (2)

$$\text{CT-ADD} \frac{\Gamma \vdash e_1 : \tau_1 \triangleright C_1 \quad \Gamma \vdash e_2 : \tau_2 \triangleright C_2}{\Gamma \vdash e_1 + e_2 : \mathbf{int} \triangleright C_1 \cup C_2 \cup \{\tau_1 \equiv \mathbf{int}, \tau_2 \equiv \mathbf{int}\}}$$

## Inference rules (3)

$$\text{CT-ABS} \frac{\Gamma, x:\tau_1 \vdash e:\tau_2 \triangleright C}{\Gamma \vdash \lambda x:\tau_1. e:\tau_1 \rightarrow \tau_2 \triangleright C}$$

$$\text{CT-APP} \frac{\begin{array}{c} \Gamma \vdash e_1:\tau_1 \triangleright C_1 \\ \Gamma \vdash e_2:\tau_2 \triangleright C_2 \\ C' = C_1 \cup C_2 \cup \{\tau_1 \equiv \tau_2 \rightarrow X\} \end{array}}{\Gamma \vdash e_1 e_2:X \triangleright C'} \quad X \text{ is fresh}$$

# Example

$$\vdash \lambda a: X. \lambda b: Y. 2 + (a (b + 3))$$

# Example

$$\vdash \lambda a: X. \lambda b: Y. 2 + (a (b + 3))$$

# Example

$$\vdash \lambda a:X. \lambda b:Y. 2 + (a (b + 3))$$
$$X \rightarrow Y \rightarrow \mathbf{int} \triangleright \{Z \equiv \mathbf{int}, X \equiv \mathbf{int} \rightarrow Z, Y \equiv \mathbf{int}, \mathbf{int} \equiv \mathbf{int}\}$$

# Example

$$\vdash \lambda a:X. \lambda b:Y. 2 + (a (b + 3))$$
$$X \rightarrow Y \rightarrow \mathbf{int} \triangleright \{Z \equiv \mathbf{int}, X \equiv \mathbf{int} \rightarrow Z, Y \equiv \mathbf{int}, \mathbf{int} \equiv \mathbf{int}\}$$
$$(\mathbf{int} \rightarrow \mathbf{int}) \rightarrow \mathbf{int} \rightarrow \mathbf{int}$$

# Unification

- ▶ What does it mean for a set of constraints to be satisfied?
- ▶ How do we find a solution to a set of constraints (i.e., infer the types)?
- ▶ To answer these questions: we define *type substitutions* and *unification*

# Type substitutions by example

- ▶ The substitution  $[X \mapsto \mathbf{int}, Y \mapsto \mathbf{int} \rightarrow \mathbf{int}]$  maps
  - ▶ type variable  $X$  to  $\mathbf{int}$ , and
  - ▶ type variable  $Y$  to  $\mathbf{int} \rightarrow \mathbf{int}$ .
- ▶ The same variable could occur in both the domain and range of a substitution.
- ▶ All substitutions are performed simultaneously.
- ▶ The substitution  $[X \mapsto \mathbf{int}, Y \mapsto \mathbf{int} \rightarrow X]$  maps
  - ▶  $Y$  to  $\mathbf{int} \rightarrow \mathbf{X}$
  - ▶ (not to  $\mathbf{int} \rightarrow \mathbf{int}$ ).



# Type substitutions (aka substitutions)

- ▶ Map from type variables to types
- ▶ Substitution of type variables, formally:

$$\sigma(X) = \begin{cases} \tau & \text{if } X \mapsto \tau \in \sigma \\ X & \text{if } X \text{ not in the domain of } \sigma \end{cases}$$

$$\sigma(\mathbf{int}) = \mathbf{int}$$

$$\sigma(\tau \rightarrow \tau') = \sigma(\tau) \rightarrow \sigma(\tau')$$

# Substitution in constraints

- ▶ Extended to substitution of constraints, and set of constraints:

$$\begin{aligned}\sigma(\tau_1 \equiv \tau_2) &= \sigma(\tau_1) \equiv \sigma(\tau_2) \\ \sigma(C) &= \{\sigma(c) \mid c \in C\}\end{aligned}$$

# Example

The substitution  $\sigma = [X \mapsto \mathbf{int}, Y \mapsto \mathbf{int} \rightarrow \mathbf{int}]$  unifies the constraint  $X \rightarrow (X \rightarrow \mathbf{int}) \equiv \mathbf{int} \rightarrow Y$ , since

$$\begin{aligned} & \sigma(X \rightarrow (X \rightarrow \mathbf{int})) \\ = & \mathbf{int} \rightarrow (\mathbf{int} \rightarrow \mathbf{int}) \\ = & \sigma(\mathbf{int} \rightarrow Y) \end{aligned}$$

# Unification

- ▶ Constraints are of form  $\tau_1 \equiv \tau_2$
- ▶ Substitution  $\sigma$  *unifies*  $\tau_1 \equiv \tau_2$  if  $\sigma(\tau_1)$  is the same as  $\sigma(\tau_2)$
- ▶ Substitution  $\sigma$  *unifies* (or *satisfies*) set of constraints  $C$  if it unifies every constraint in  $C$
- ▶ So given  $\vdash e:\tau \triangleright C$ , we want a substitution  $\sigma$  that unifies  $C$ . Moreover, type of  $e$  is  $\sigma(\tau)$

# Unification algorithm

$$\mathit{unify}(C) = \sigma$$

*unify*( $\emptyset$ )

[]

*unify*( $\{\tau \equiv \tau'\} \cup C$ )

if  $\tau = \tau'$  then

*unify*( $C$ )

else if  $\tau = X$  and  $X$  not a free variable of  $\tau'$  then

let  $\sigma = [X \mapsto \tau']$  in

*unify*( $\sigma(C)$ )  $\circ \sigma$

else if  $\tau' = X$  and  $X$  not a free variable of  $\tau$  then

let  $\sigma = [X \mapsto \tau]$  in

*unify*( $\sigma(C)$ )  $\circ \sigma$

else if  $\tau = \tau_0 \rightarrow \tau_1$  and  $\tau' = \tau'_0 \rightarrow \tau'_1$  then

*unify*( $C \cup \{\tau_0 \equiv \tau'_0, \tau_1 \equiv \tau'_1\}$ )

else *fail*

# Unification Algorithm (Things to Note)

- ▶ Choose a constraint from set  $C$ .
- ▶ Occurs check – free variable side conditions.



# Principal Types

- ▶ Recall: given  $\vdash e:\tau \triangleright C$ , we want a substitution  $\sigma$  that unifies  $C$ . Moreover, type of  $e$  is  $\sigma(\tau)$ .
- ▶ We want the most general solution.

# Principal unifier

- ▶ A substitution  $\sigma$  is *less specific* (or *more general*) than a substitution  $\sigma'$ , written  $\sigma \sqsubseteq \sigma'$  if  $\sigma' = \gamma \circ \sigma$  for some substitution  $\gamma$ .
- ▶ A *principal unifier* (or *most general unifier*) for a constraint set  $C$  is a substitution  $\sigma$  that satisfies  $C$  and such that  $\sigma \sqsubseteq \sigma'$  for every substitution  $\sigma'$  satisfying  $C$ .

# Examples

- ▶  $\{X \equiv \mathbf{int}, Y \equiv X \rightarrow X\}$
- ▶  $\{\mathbf{int} \rightarrow \mathbf{int} \equiv X \rightarrow Y\}$
- ▶  $\{X \rightarrow Y \equiv Y \rightarrow Z, Z \equiv U \rightarrow W\}$
- ▶  $\{\mathbf{int} \equiv \mathbf{int} \rightarrow Y\}$
- ▶  $\{Y \equiv \mathbf{int} \rightarrow Y\}$
- ▶  $\{\}$

# Examples

# Examples

- ▶  $\{X \equiv \mathbf{int}, Y \equiv X \rightarrow X\}$

# Examples

- ▶  $\{X \equiv \mathbf{int}, Y \equiv X \rightarrow X\}$   
 $[X \mapsto \mathbf{int}, Y \mapsto \mathbf{int} \rightarrow \mathbf{int}]$

# Examples

- ▶  $\{X \equiv \mathbf{int}, Y \equiv X \rightarrow X\}$   
 $[X \mapsto \mathbf{int}, Y \mapsto \mathbf{int} \rightarrow \mathbf{int}]$
- ▶  $\{\mathbf{int} \rightarrow \mathbf{int} \equiv X \rightarrow Y\}$

# Examples

- ▶  $\{X \equiv \mathbf{int}, Y \equiv X \rightarrow X\}$   
 $[X \mapsto \mathbf{int}, Y \mapsto \mathbf{int} \rightarrow \mathbf{int}]$
- ▶  $\{\mathbf{int} \rightarrow \mathbf{int} \equiv X \rightarrow Y\}$   
 $[X \mapsto \mathbf{int}, Y \mapsto \mathbf{int}]$



# Examples

- ▶  $\{X \equiv \mathbf{int}, Y \equiv X \rightarrow X\}$   
 $[X \mapsto \mathbf{int}, Y \mapsto \mathbf{int} \rightarrow \mathbf{int}]$
- ▶  $\{\mathbf{int} \rightarrow \mathbf{int} \equiv X \rightarrow Y\}$   
 $[X \mapsto \mathbf{int}, Y \mapsto \mathbf{int}]$
- ▶  $\{X \rightarrow Y \equiv Y \rightarrow Z, Z \equiv U \rightarrow W\}$

# Examples

- ▶  $\{X \equiv \mathbf{int}, Y \equiv X \rightarrow X\}$   
 $[X \mapsto \mathbf{int}, Y \mapsto \mathbf{int} \rightarrow \mathbf{int}]$
- ▶  $\{\mathbf{int} \rightarrow \mathbf{int} \equiv X \rightarrow Y\}$   
 $[X \mapsto \mathbf{int}, Y \mapsto \mathbf{int}]$
- ▶  $\{X \rightarrow Y \equiv Y \rightarrow Z, Z \equiv U \rightarrow W\}$   
 $[X \mapsto U \rightarrow W, Y \mapsto U \rightarrow W, X \mapsto U \rightarrow W]$

# Examples

- ▶  $\{X \equiv \mathbf{int}, Y \equiv X \rightarrow X\}$   
 $[X \mapsto \mathbf{int}, Y \mapsto \mathbf{int} \rightarrow \mathbf{int}]$
- ▶  $\{\mathbf{int} \rightarrow \mathbf{int} \equiv X \rightarrow Y\}$   
 $[X \mapsto \mathbf{int}, Y \mapsto \mathbf{int}]$
- ▶  $\{X \rightarrow Y \equiv Y \rightarrow Z, Z \equiv U \rightarrow W\}$   
 $[X \mapsto U \rightarrow W, Y \mapsto U \rightarrow W, X \mapsto U \rightarrow W]$
- ▶  $\{\mathbf{int} \equiv \mathbf{int} \rightarrow Y\}$

# Examples

- ▶  $\{X \equiv \mathbf{int}, Y \equiv X \rightarrow X\}$   
 $[X \mapsto \mathbf{int}, Y \mapsto \mathbf{int} \rightarrow \mathbf{int}]$
- ▶  $\{\mathbf{int} \rightarrow \mathbf{int} \equiv X \rightarrow Y\}$   
 $[X \mapsto \mathbf{int}, Y \mapsto \mathbf{int}]$
- ▶  $\{X \rightarrow Y \equiv Y \rightarrow Z, Z \equiv U \rightarrow W\}$   
 $[X \mapsto U \rightarrow W, Y \mapsto U \rightarrow W, X \mapsto U \rightarrow W]$
- ▶  $\{\mathbf{int} \equiv \mathbf{int} \rightarrow Y\}$   
Not unifiable

# Examples

- ▶  $\{X \equiv \mathbf{int}, Y \equiv X \rightarrow X\}$   
 $[X \mapsto \mathbf{int}, Y \mapsto \mathbf{int} \rightarrow \mathbf{int}]$
- ▶  $\{\mathbf{int} \rightarrow \mathbf{int} \equiv X \rightarrow Y\}$   
 $[X \mapsto \mathbf{int}, Y \mapsto \mathbf{int}]$
- ▶  $\{X \rightarrow Y \equiv Y \rightarrow Z, Z \equiv U \rightarrow W\}$   
 $[X \mapsto U \rightarrow W, Y \mapsto U \rightarrow W, X \mapsto U \rightarrow W]$
- ▶  $\{\mathbf{int} \equiv \mathbf{int} \rightarrow Y\}$   
Not unifiable
- ▶  $\{Y \equiv \mathbf{int} \rightarrow Y\}$

# Examples

- ▶  $\{X \equiv \mathbf{int}, Y \equiv X \rightarrow X\}$   
 $[X \mapsto \mathbf{int}, Y \mapsto \mathbf{int} \rightarrow \mathbf{int}]$
- ▶  $\{\mathbf{int} \rightarrow \mathbf{int} \equiv X \rightarrow Y\}$   
 $[X \mapsto \mathbf{int}, Y \mapsto \mathbf{int}]$
- ▶  $\{X \rightarrow Y \equiv Y \rightarrow Z, Z \equiv U \rightarrow W\}$   
 $[X \mapsto U \rightarrow W, Y \mapsto U \rightarrow W, X \mapsto U \rightarrow W]$
- ▶  $\{\mathbf{int} \equiv \mathbf{int} \rightarrow Y\}$   
Not unifiable
- ▶  $\{Y \equiv \mathbf{int} \rightarrow Y\}$   
Not unifiable

# Examples

- ▶  $\{X \equiv \mathbf{int}, Y \equiv X \rightarrow X\}$   
 $[X \mapsto \mathbf{int}, Y \mapsto \mathbf{int} \rightarrow \mathbf{int}]$
- ▶  $\{\mathbf{int} \rightarrow \mathbf{int} \equiv X \rightarrow Y\}$   
 $[X \mapsto \mathbf{int}, Y \mapsto \mathbf{int}]$
- ▶  $\{X \rightarrow Y \equiv Y \rightarrow Z, Z \equiv U \rightarrow W\}$   
 $[X \mapsto U \rightarrow W, Y \mapsto U \rightarrow W, X \mapsto U \rightarrow W]$
- ▶  $\{\mathbf{int} \equiv \mathbf{int} \rightarrow Y\}$   
Not unifiable
- ▶  $\{Y \equiv \mathbf{int} \rightarrow Y\}$   
Not unifiable
- ▶  $\{\}$

# Examples

- ▶  $\{X \equiv \mathbf{int}, Y \equiv X \rightarrow X\}$   
 $[X \mapsto \mathbf{int}, Y \mapsto \mathbf{int} \rightarrow \mathbf{int}]$
- ▶  $\{\mathbf{int} \rightarrow \mathbf{int} \equiv X \rightarrow Y\}$   
 $[X \mapsto \mathbf{int}, Y \mapsto \mathbf{int}]$
- ▶  $\{X \rightarrow Y \equiv Y \rightarrow Z, Z \equiv U \rightarrow W\}$   
 $[X \mapsto U \rightarrow W, Y \mapsto U \rightarrow W, X \mapsto U \rightarrow W]$
- ▶  $\{\mathbf{int} \equiv \mathbf{int} \rightarrow Y\}$   
Not unifiable
- ▶  $\{Y \equiv \mathbf{int} \rightarrow Y\}$   
Not unifiable
- ▶  $\{\}$   
 $\square$



# Properties of the algorithm *unify*

1.  $unify(C)$  halts, either by failing or by returning a substitution for all  $C$ ;
2. If  $unify(C) = \sigma$ , then  $\sigma$  is a unifier for  $C$ ;
3. if  $\delta$  is a unifier for  $C$ , then  $unify(C) = \sigma$  with  $\sigma \sqsubseteq \delta$ .

# Proving 1. termination

Define a lexicographic measure on a constraint set  $C$ , called *degree*:  
pair  $(m, n)$  where  
 $m$  is the number of distinct type variables in  $C$  and  
 $n$  is the total size of the types in  $C$ .

See that each recursive call has a smaller degree.

## Proving 2. unifier

Induction on the number of recursive calls in the computation of  $unify(C)$ .

Variable cases depend on observation:

If  $\sigma$  unifies  $[X \mapsto \tau]D$ , then  $\sigma \circ [X \mapsto \tau]$  unifies  $\{X \equiv \tau\} \cup D$  for any constraint set  $D$ .

## Proving 3. principal

Again, induction on the number of recursive calls in the computation of  $unify(C)$ .

- ▶ If  $C$  is empty, then  $unify(C)$  immediately returns the trivial substitution  $[]$ ; since  $\delta = \delta \circ []$ , we have  $[] \sqsubseteq \delta$  as required.
- ▶ If  $C$  is non-empty, then  $unify(C)$  chooses some constraint  $\tau \equiv \tau'$  and continues on the shape. (See TAPL.)

# Let-Polymorphism