

Programming Languages and Compilers (CS 421)

Elsa L Gunter

2112 SC, UIUC

[http://www.cs.uiuc.edu/class
/sp07/cs421/](http://www.cs.uiuc.edu/class/sp07/cs421/)

Based in part on slides by Mattox Beckman, as updated
by Vikram Adve and Gul Agha

Type Inference - The Problem

- Given an expression e , and a typing environment Γ , does there exist a type τ such that the judgment

$$\Gamma \vdash e : \tau$$

is valid - ie., follows from the typing rules?

Type Inference - Outline

- Begin by assigning a type variable as the type of the whole expression
- Decompose the expression into component expressions
- Use typing rules to generate constraints on components and whole
- Recursively gather additional constraints to guarantee a solution for components
- Solve system of constraints to generate a substitution
- Apply substitution to orig. type var. to get answer

Type Inference - Example

- What type can we give to
 $\text{fun } x \rightarrow \text{fun } f \rightarrow f \ x?$
- Start with a type variable and then look at the way the term is constructed

Type Inference - Example

- First approximate:

$$[] \vdash (\text{fun } x \rightarrow \text{fun } f \rightarrow f \ x) : \alpha$$

- Second approximate: use fun rule

$$\frac{[x : \beta] \vdash (\text{fun } f \rightarrow f \ x) : \gamma}{[] \vdash (\text{fun } x \rightarrow \text{fun } f \rightarrow f \ x) : \alpha}$$

- $\alpha \equiv (\beta \rightarrow \gamma)$

Type Inference - Example

- Third approximate: use fun rule

$$\frac{\frac{[f : \delta ; x : \beta] \vdash (f x) : \varepsilon}{[x : \beta] \vdash (\text{fun } f \rightarrow f x) : \gamma}}{[] \vdash (\text{fun } x \rightarrow \text{fun } f \rightarrow f x) : \alpha}$$

- $\alpha \equiv (\beta \rightarrow \gamma); \gamma \equiv (\delta \rightarrow \varepsilon)$

Type Inference - Example

- Fourth approximate: use app rule

$$[f : \delta; x : \beta] \vdash f : \varphi \rightarrow \varepsilon \quad [f : \delta; x : \beta] \vdash x : \varphi$$

$$[f : \delta ; x : \beta] \vdash (f x) : \varepsilon$$

$$[x : \beta] \vdash (\text{fun } f \text{ -> } f x) : \gamma$$

$$[] \vdash (\text{fun } x \text{ -> fun } f \text{ -> } f x) : \alpha$$

- $\alpha \equiv (\beta \rightarrow \gamma); \gamma \equiv (\delta \rightarrow \varepsilon)$

Type Inference - Example

- Fifth approximate: use var rule

$$\frac{}{[f : \delta ; x : \beta] \vdash f : \varphi \rightarrow \varepsilon} \quad [f : \delta ; x : \beta] \vdash x : \varphi$$

$$\frac{}{[f : \delta ; x : \beta] \vdash (f x) : \varepsilon}$$

$$\frac{}{[x : \beta] \vdash (\text{fun } f \text{ -> } f x) : \gamma}$$

$$\frac{}{[] \vdash (\text{fun } x \text{ -> fun } f \text{ -> } f x) : \alpha}$$

- $\alpha \equiv (\beta \rightarrow \gamma)$; $\gamma \equiv (\delta \rightarrow \varepsilon)$; $\delta \equiv (\varphi \rightarrow \varepsilon)$.

Type Inference - Example

- Sixth approximate: use var rule

$$\frac{\frac{\frac{[f : \delta ; x : \beta] \vdash f : \varphi \rightarrow \varepsilon}{[f : \delta ; x : \beta] \vdash x : \varphi}}{[f : \delta ; x : \beta] \vdash (f x) : \varepsilon}}{[x : \beta] \vdash (\text{fun } f \rightarrow f x) : \gamma}}{[] \vdash (\text{fun } x \rightarrow \text{fun } f \rightarrow f x) : \alpha}$$

- $\alpha \equiv (\beta \rightarrow \gamma)$; $\gamma \equiv (\delta \rightarrow \varepsilon)$; $\delta \equiv (\varphi \rightarrow \varepsilon)$; $\varphi \equiv \beta$

Type Inference - Example

- Done building proof tree; now solve!

$$\frac{\frac{\frac{[f : \delta ; x : \beta] \vdash f : \varphi \rightarrow \varepsilon}{[f : \delta ; x : \beta] \vdash x : \varphi}}{[f : \delta ; x : \beta] \vdash (f x) : \varepsilon}}{[x : \beta] \vdash (\text{fun } f \rightarrow f x) : \gamma}}{[] \vdash (\text{fun } x \rightarrow \text{fun } f \rightarrow f x) : \alpha}$$

- $\alpha \equiv (\beta \rightarrow \gamma)$; $\gamma \equiv (\delta \rightarrow \varepsilon)$; $\delta \equiv (\varphi \rightarrow \varepsilon)$; $\varphi \equiv \beta$

Type Inference - Example

- Type unification; solve like linear equations;

$$\frac{\frac{\frac{[f : \delta ; x : \beta] \vdash f : \varphi \rightarrow \varepsilon}{[f : \delta ; x : \beta] \vdash x : \varphi}}{[f : \delta ; x : \beta] \vdash (f x) : \varepsilon}}{[x : \beta] \vdash (\text{fun } f \text{ -> } f x) : \gamma}}{[] \vdash (\text{fun } x \text{ -> fun } f \text{ -> } f x) : \alpha}$$

- $\alpha \equiv (\beta \rightarrow \gamma)$; $\gamma \equiv (\delta \rightarrow \varepsilon)$; $\delta \equiv (\varphi \rightarrow \varepsilon)$; $\varphi \equiv \beta$

Type Inference - Example

- Eliminate φ :

$$\frac{\frac{\frac{[f : \delta ; x : \beta] \vdash f : \beta \rightarrow \varepsilon}{[f : \delta ; x : \beta] \vdash (f x) : \varepsilon}}{[x : \beta] \vdash (\text{fun } f \rightarrow f x) : \gamma}}{[] \vdash (\text{fun } x \rightarrow \text{fun } f \rightarrow f x) : \alpha}}$$

- $\alpha \equiv (\beta \rightarrow \gamma)$; $\gamma \equiv (\delta \rightarrow \varepsilon)$; $\delta \equiv (\beta \rightarrow \varepsilon)$; $\varphi \equiv \beta$

Type Inference - Example

- Next eliminate δ :

$$\frac{\frac{\frac{[f : \beta \rightarrow \varepsilon; x : \beta] \vdash f : \beta \rightarrow \varepsilon}{[f : \delta; x : \beta] \vdash x : \beta}}{[f : \beta \rightarrow \varepsilon; x : \beta] \vdash (f x) : \varepsilon}}{[x : \beta] \vdash (\text{fun } f \text{ -> } f x) : \gamma}}{[] \vdash (\text{fun } x \text{ -> fun } f \text{ -> } f x) : \alpha}$$

- $\alpha \equiv (\beta \rightarrow \gamma)$; $\gamma \equiv ((\beta \rightarrow \varepsilon) \rightarrow \varepsilon)$; $\delta \equiv (\beta \rightarrow \varepsilon)$;

Type Inference - Example

- Next eliminate γ :

$$\frac{\frac{\frac{[f : \beta \rightarrow \varepsilon; x : \beta] \vdash f : \beta \rightarrow \varepsilon}{[f : \beta \rightarrow \varepsilon; x : \beta] \vdash (f x) : \varepsilon}}{[x : \beta] \vdash (\text{fun } f \rightarrow f x) : ((\beta \rightarrow \varepsilon) \rightarrow \varepsilon)}}{[] \vdash (\text{fun } x \rightarrow \text{fun } f \rightarrow f x) : \alpha}}$$

- $\alpha \equiv (\beta \rightarrow ((\beta \rightarrow \varepsilon) \rightarrow \varepsilon)); \gamma \equiv ((\beta \rightarrow \varepsilon) \rightarrow \varepsilon);$

Type Inference - Example

- Next eliminate α :

$$\frac{}{[f : \beta \rightarrow \varepsilon; x : \beta] \vdash f : \beta \rightarrow \varepsilon} \quad \frac{}{[f : \delta; x : \beta] \vdash x : \beta}$$

$$\frac{}{[f : \beta \rightarrow \varepsilon; x : \beta] \vdash (f x) : \varepsilon}$$

$$\frac{}{[x : \beta] \vdash (\text{fun } f \text{ -> } f x) : ((\beta \rightarrow \varepsilon) \rightarrow \varepsilon)}$$

$$\frac{}{[] \vdash (\text{fun } x \text{ -> fun } f \text{ -> } f x) : (\beta \rightarrow ((\beta \rightarrow \varepsilon) \rightarrow \varepsilon))}$$

- $\alpha \equiv (\beta \rightarrow ((\beta \rightarrow \varepsilon) \rightarrow \varepsilon));$

Type Inference - Example

- No more equations to solve; we are done

$$\frac{}{[f : \beta \rightarrow \varepsilon; x : \beta] \vdash f : \beta \rightarrow \varepsilon} \quad \frac{}{[f : \delta; x : \beta] \vdash x : \beta}$$

$$\frac{}{[f : \beta \rightarrow \varepsilon; x : \beta] \vdash (f x) : \varepsilon}$$

$$\frac{}{[x : \beta] \vdash (\text{fun } f \text{ -> } f x) : ((\beta \rightarrow \varepsilon) \rightarrow \varepsilon)}$$

$$\frac{}{[] \vdash (\text{fun } x \text{ -> } \text{fun } f \text{ -> } f x) : (\beta \rightarrow ((\beta \rightarrow \varepsilon) \rightarrow \varepsilon))}$$

- Any instance of $(\beta \rightarrow ((\beta \rightarrow \varepsilon) \rightarrow \varepsilon))$ is a valid type

Type Inference - The Problem

- Given an expression e , and a typing environment Γ , does there exist a type τ such that the judgment

$$\Gamma \vdash e : \tau$$

is valid - ie., follows from the typing rules?

Type Inference Algorithm

Let $\text{has_type}(\Gamma, e, \tau) = S$

- Γ is a typing environment
- e is an expression
- τ is a (generalized) type,
- S is a set of equations between generalized types

Type Inference Algorithm

- Let $\text{has_type}(\Gamma, e, \tau) = S$
 - Γ is a typing environment,
 - e is an expression,
 - τ is a (generalized) type,
 - S is a set of equations between generalized types.
Idea: S is the constraints on type variables necessary for $\Gamma \vdash e : \tau$
- Let $\text{Unif}(S)$ be a substitution of generalized types for type variables solving S
- Solution: $\text{Unif}(S)(\Gamma) \vdash e : \text{Unif}(S)(\tau)$

Type Inference Algorithm

has_type (Γ, exp, τ) =

- Case *exp* of
 - Var v --> return $\{\tau \equiv \Gamma(v)\}$
 - Const c --> return $\{\tau \equiv \sigma\}$ where $\Gamma \vdash c : \sigma$ by the constant rules
 - fun $x \rightarrow e$ -->
 - Let α, β be fresh variables
 - Let $S = \text{has_type} ([x: \alpha] \cup \Gamma, e, \beta)$
 - Return $\{\tau \equiv \alpha \rightarrow \beta\} \cup S$

Type Inference Algorithm (cont)

- Case *exp* of
 - App ($e_1 e_2$) \rightarrow
 - Let α be a fresh variable
 - Let $S_1 = \text{has_type}(\Gamma, e_1, \alpha \rightarrow \tau)$
 - Let $S_2 = \text{has_type}(\Gamma, e_2, \alpha)$
 - Return $S_1 \cup S_2$

Type Inference Algorithm (cont)

- Case *exp* of
 - let $x = e_1$ in e_2 -->
 - Let α be a fresh variable
 - Let $S_1 = \text{has_type}(\Gamma, e_1, \alpha)$
 - Let $S_2 =$
 $\text{has_type}([x: \alpha] \cup \Gamma, e_2, \tau)$
 - Return $S_1 \cup S_2$

Type Inference Algorithm (cont)

- Case *exp* of
 - let rec $x = e_1$ in e_2 -->
 - Let α be a fresh variable
 - Let $S_1 = \text{has_type}([x: \alpha] \cup \Gamma, e_1, \alpha)$
 - Let $S_2 = \text{has_type}([x: \alpha] \cup \Gamma, e_2, \tau)$
 - Return $S_1 \cup S_2$

Type Inference Algorithm (cont)

- To infer a type, introduce `type_of`
- Let α be a fresh variable
- `type_of` (Γ, e) =
 - Let $S = \text{has_type}(\Gamma, e, \alpha)$
 - Return $\text{Unif}(S)(\alpha)$
- Need an algorithm for `Unif`