Programming Languages and Compilers (CS 421)

Elsa L Gunter

2112 SC, UIUC

http://courses.engr.illinois.edu/cs421

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

Simple Example

- Let Γ = [x:int ; y:bool]
- Show Γ |- y || (x + 3 > 6) : bool
- Start building the proof tree from the bottom up

$$\frac{?}{\Gamma \vdash y \mid\mid (x + 3 > 6) : bool}$$

Simple Example

- Let Γ = [x:int ; y:bool]
- Show Γ |- y || (x + 3 > 6) : bool
- Which rule has this as a conclusion?

$$\frac{?}{\Gamma \mid\!- \; y \;||\; (x + 3 > 6) : bool}$$

Simple Example

- Let Γ = [x:int ; y:bool]
- Show Γ |- y || (x + 3 > 6) : bool
- Boolean Connectives: ||

$$\frac{\Gamma \text{ |- } y : \text{bool} \qquad\qquad \Gamma \text{ |- } x + 3 > 6 : \text{bool}}{\Gamma \text{ |- } y \text{ || } (x + 3 > 6) : \text{bool}}$$

Simple Example

- Let Γ = [x:int ; y:bool]
- Show Γ |- y || (x + 3 > 6) : bool
- Pick an assumption to prove

$$\dfrac{\dfrac{?}{\Gamma \text{ |- } y : \text{bool}} \qquad \Gamma \text{ |- } x + 3 > 6 : \text{bool}}{\Gamma \text{ |- } y \;||\; (x + 3 > 6) : \text{bool}}$$

Simple Example

- Let Γ = [x:int ; y:bool]
- Show Γ |- y || (x + 3 > 6) : bool
- Which rule has this as a conclusion?

$$\dfrac{\dfrac{?}{\Gamma \;|\!\!-\; y : \text{bool}} \qquad \Gamma \;|\!\!-\; x + 3 > 6 : \text{bool}}{\Gamma \;|\!\!-\; y \;||\; (x + 3 > 6) : \text{bool}}$$

Simple Example

- Let Γ = [x:int ; y:bool]
- Show Γ |- y || (x + 3 > 6) : bool
- Axiom for variables

$$\frac{\dfrac{}{\Gamma \ |\!\!-\ y : \text{bool}} \qquad \Gamma \ |\!\!-\ x + 3 > 6 : \text{bool}}{\Gamma \ |\!\!-\ y \ || \ (x + 3 > 6) : \text{bool}}$$

Simple Example

- Let Γ = [x:int ; y:bool]
- Show Γ |- y || (x + 3 > 6) : bool
- Pick an assumption to prove

$$\frac{\dfrac{}{\Gamma \text{ |- } y : \text{bool}} \qquad \dfrac{?}{\Gamma \text{ |- } x + 3 > 6 : \text{bool}}}{\Gamma \text{ |- } y \text{ || } (x + 3 > 6) : \text{bool}}$$

Simple Example

- Let Γ = [x:int ; y:bool]
- Show Γ |- y || (x + 3 > 6) : bool
- Which rule has this as a conclusion?

$$\dfrac{\dfrac{}{\Gamma \mid\!- \; y : \mathrm{bool}} \qquad \dfrac{?}{\Gamma \mid\!- \; x + 3 > 6 : \mathrm{bool}}}{\Gamma \mid\!- \; y \,||\, (x + 3 > 6) : \mathrm{bool}}$$

Simple Example

- Let Γ = [x:int ; y:bool]
- Show Γ |- y || (x + 3 > 6) : bool
- Arithmetic relations

$$\dfrac{\dfrac{}{\Gamma \mid\!- \; y : \text{bool}} \qquad \dfrac{\Gamma \mid\!- \; x + 3 : \text{int} \qquad \Gamma \mid\!- \; 6 : \text{int}}{\Gamma \mid\!- \; x + 3 > 6 : \text{bool}}}{\Gamma \mid\!- \; y \mid\mid (x + 3 > 6) : \text{bool}}$$

Simple Example

- Let Γ = [x:int ; y:bool]
- Show Γ |- y || (x + 3 > 6) : bool
- Pick an assumption to prove

$$\dfrac{\dfrac{}{\Gamma \mid\!- \; y : \text{bool}} \qquad \dfrac{\Gamma \mid\!- \; x + 3 : \text{int} \qquad \dfrac{?}{\Gamma \mid\!- \; 6 : \text{int}}}{\Gamma \mid\!- \; x + 3 > 6 : \text{bool}}}{\Gamma \mid\!- \; y \;||\; (x + 3 > 6) : \text{bool}}$$

Simple Example

- Let Γ = [x:int ; y:bool]
- Show Γ |- y || (x + 3 > 6) : bool
- Which rule has this as a conclusion?

$$\dfrac{\dfrac{}{\Gamma \mid\!\!- \; y : \mathrm{bool}} \qquad \dfrac{\Gamma \mid\!\!- \; x + 3 : \mathrm{int} \qquad \dfrac{?}{\Gamma \mid\!\!- \; 6 : \mathrm{int}}}{\Gamma \mid\!\!- \; x + 3 > 6 : \mathrm{bool}}}{\Gamma \mid\!\!- \; y \mid\mid (x + 3 > 6) : \mathrm{bool}}$$

Simple Example

- Let Γ = [x:int ; y:bool]
- Show Γ |- y || (x + 3 > 6) : bool
- Axiom for constants

$$\dfrac{\overline{\Gamma \;|\!\!-\; y : \text{bool}} \qquad \dfrac{\Gamma \;|\!\!-\; x + 3 : \text{int} \qquad \overline{\Gamma \;|\!\!-\; 6 : \text{int}}}{\Gamma \;|\!\!-\; x + 3 > 6 : \text{bool}}}{\Gamma \;|\!\!-\; y \;||\; (x + 3 > 6) : \text{bool}}$$

Simple Example

- Let Γ = [x:int ; y:bool]
- Show Γ |- y || (x + 3 > 6) : bool
- Pick an assumption to prove

$$\dfrac{\dfrac{}{\Gamma \mid\!\!- \; y : \text{bool}} \qquad \dfrac{\dfrac{?}{\Gamma \mid\!\!- \; x + 3 : \text{int}} \quad \dfrac{}{\Gamma \mid\!\!- \; 6 : \text{int}}}{\Gamma \mid\!\!- \; x + 3 > 6 : \text{bool}}}{\Gamma \mid\!\!- \; y \mid\mid (x + 3 > 6) : \text{bool}}$$

Simple Example

- Let Γ = [x:int ; y:bool]
- Show Γ |- y || (x + 3 > 6) : bool
- Which rule has this as a conclusion?

$$\dfrac{\dfrac{}{\Gamma\ \text{|-}\ y : \text{bool}} \qquad \dfrac{\dfrac{?}{\Gamma\ \text{|-}\ x + 3 : \text{int}} \quad \dfrac{}{\Gamma\ \text{|-}\ 6 : \text{int}}}{\Gamma\ \text{|-}\ x + 3 > 6 : \text{bool}}}{\Gamma\ \text{|-}\ y\ ||\ (x + 3 > 6) : \text{bool}}$$

Simple Example

- Let Γ = [x:int ; y:bool]
- Show Γ |- y || (x + 3 > 6) : bool
- Arithmetic operations

$$\dfrac{\dfrac{}{\Gamma \mathrel{|\!-} y : \mathrm{bool}} \qquad \dfrac{\dfrac{\Gamma \mathrel{|\!-} x : \mathrm{int} \quad \Gamma \mathrel{|\!-} 3 : \mathrm{int}}{\Gamma \mathrel{|\!-} x + 3 : \mathrm{int}} \quad \dfrac{}{\Gamma \mathrel{|\!-} 6 : \mathrm{int}}}{\Gamma \mathrel{|\!-} x + 3 > 6 : \mathrm{bool}}}{\Gamma \mathrel{|\!-} y \mathbin{||} (x + 3 > 6) : \mathrm{bool}}$$

Simple Example

- Let Γ = [x:int ; y:bool]
- Show Γ |- y || (x + 3 > 6) : bool
- Pick an assumption to prove

$$
\dfrac{
  \dfrac{}{\Gamma \mid\!- \; y : \text{bool}}
  \qquad
  \dfrac{
    \dfrac{
      \Gamma \mid\!- \; x : \text{int}
      \quad
      \dfrac{?}{\Gamma \mid\!- \; 3 : \text{int}}
    }{\Gamma \mid\!- \; x + 3 : \text{int}}
    \quad
    \dfrac{}{\Gamma \mid\!- \; 6 : \text{int}}
  }{\Gamma \mid\!- \; x + 3 > 6 : \text{bool}}
}{\Gamma \mid\!- \; y \mid\mid (x + 3 > 6) : \text{bool}}
$$

Simple Example

- Let Γ = [x:int ; y:bool]
- Show Γ |- y || (x + 3 > 6) : bool
- Which rule has this as a conclusion?

$$\dfrac{\dfrac{}{\Gamma\ |\!-\ y : \text{bool}} \qquad \dfrac{\dfrac{\Gamma\ |\!-\ x : \text{int} \quad \dfrac{?}{\Gamma\ |\!-\ 3 : \text{int}}}{\Gamma\ |\!-\ x + 3 : \text{int}} \quad \dfrac{}{\Gamma\ |\!-\ 6 : \text{int}}}{\Gamma\ |\!-\ x + 3 > 6 : \text{bool}}}{\Gamma\ |\!-\ y\ ||\ (x + 3 > 6) : \text{bool}}$$

Simple Example

- Let Γ = [x:int ; y:bool]
- Show Γ |- y || (x + 3 > 6) : bool
- Axiom for constants

$$\dfrac{\dfrac{}{\Gamma \;|\!\!- \; y : \text{bool}} \qquad \dfrac{\dfrac{\Gamma \;|\!\!- \; x : \text{int} \quad \dfrac{}{\Gamma \;|\!\!- \; 3 : \text{int}}}{\Gamma \;|\!\!- \; x + 3 : \text{int}} \quad \dfrac{}{\Gamma \;|\!\!- \; 6 : \text{int}}}{\Gamma \;|\!\!- \; x + 3 > 6 : \text{bool}}}{\Gamma \;|\!\!- \; y \;||\; (x + 3 > 6) : \text{bool}}$$

Simple Example

- Let Γ = [x:int ; y:bool]
- Show Γ |- y || (x + 3 > 6) : bool
- Pick an assumption to prove

$$\dfrac{\dfrac{}{\Gamma \vdash y : \text{bool}} \qquad \dfrac{\dfrac{\dfrac{?}{\Gamma \vdash x : \text{int}} \quad \dfrac{}{\Gamma \vdash 3 : \text{int}}}{\Gamma \vdash x + 3 : \text{int}} \quad \dfrac{}{\Gamma \vdash 6 : \text{int}}}{\Gamma \vdash x + 3 > 6 : \text{bool}}}{\Gamma \vdash y \mid\mid (x + 3 > 6) : \text{bool}}$$

Simple Example

- Let Γ = [x:int ; y:bool]
- Show Γ |- y || (x + 3 > 6) : bool
- Which rule has this as a conclusion?

$$\dfrac{\dfrac{}{\Gamma \mid\!- \; y : \mathrm{bool}} \qquad \dfrac{\dfrac{\dfrac{?}{\Gamma \mid\!- \; x : \mathrm{int}} \quad \dfrac{}{\Gamma \mid\!- \; 3 : \mathrm{int}}}{\Gamma \mid\!- \; x + 3 : \mathrm{int}} \quad \dfrac{}{\Gamma \mid\!- \; 6 : \mathrm{int}}}{\Gamma \mid\!- \; x + 3 > 6 : \mathrm{bool}}}{\Gamma \mid\!- \; y \mid\mid (x + 3 > 6) : \mathrm{bool}}$$

Simple Example

- Let Γ = [x:int ; y:bool]
- Show Γ |- y || (x + 3 > 6) : bool
- Axiom for variables

$$\dfrac{\dfrac{}{\Gamma \mathrel{|\!\!-} y : \text{bool}} \qquad \dfrac{\dfrac{\dfrac{}{\Gamma \mathrel{|\!\!-} x : \text{int}} \quad \dfrac{}{\Gamma \mathrel{|\!\!-} 3 : \text{int}}}{\Gamma \mathrel{|\!\!-} x + 3 : \text{int}} \qquad \dfrac{}{\Gamma \mathrel{|\!\!-} 6 : \text{int}}}{\Gamma \mathrel{|\!\!-} x + 3 > 6 : \text{bool}}}{\Gamma \mathrel{|\!\!-} y \mid\mid (x + 3 > 6) : \text{bool}}$$

Simple Example

- Let Γ = [x:int ; y:bool]
- Show Γ |- y || (x + 3 > 6) : bool
- No more assumptions! DONE!

$$\dfrac{\dfrac{}{\Gamma \mid\!\!-\ y : \text{bool}} \qquad \dfrac{\dfrac{\dfrac{}{\Gamma \mid\!\!-\ x : \text{int}} \quad \dfrac{}{\Gamma \mid\!\!-\ 3 : \text{int}}}{\Gamma \mid\!\!-\ x + 3 : \text{int}} \qquad \dfrac{}{\Gamma \mid\!\!-\ 6 : \text{int}}}{\Gamma \mid\!\!-\ x + 3 > 6 : \text{bool}}}{\Gamma \mid\!\!-\ y \mid\mid (x + 3 > 6) : \text{bool}}$$

Type Variables in Rules

- If_then_else rule:

$$\frac{\Gamma \mid\!- \ e_1 : \text{bool} \quad \Gamma \mid\!- \ e_2 : \tau \quad \Gamma \mid\!- \ e_3 : \tau}{\Gamma \mid\!- \ (\text{if } e_1 \text{ then } e_2 \text{ else } e_3) : \tau}$$

- τ is a type variable (meta-variable)
- Can take any type at all
- All instances in a rule application must get same type
- Then branch, else branch and if_then_else must all have same type

Function Application

- Application rule:

$$\frac{\Gamma \mid\!- \; e_1 : \tau_1 \rightarrow \tau_2 \qquad \Gamma \mid\!- \; e_2 : \tau_1}{\Gamma \mid\!- \; (e_1 \; e_2) : \tau_2}$$

- If you have a function expression e_1 of type $\tau_1 \rightarrow \tau_2$ applied to an argument of type τ_1, the resulting expression has type τ_2

Application Examples

$$\frac{\Gamma \text{ |- print\_int : int} \rightarrow \text{unit} \qquad \Gamma \text{ |- 5 : int}}{\Gamma \text{ |- (print\_int 5) : unit}}$$

- e_1 = print_int, e_2 = 5,
- τ_1 = int, τ_2 = unit

$$\frac{\Gamma \text{ |- map print\_int : int list} \rightarrow \text{unit list} \qquad \Gamma \text{ |- [3;7] : int list}}{\Gamma \text{ |- (map print\_int [3; 7]) : unit list}}$$

- e_1 = map print_int, e_2 = [3; 7],
- τ_1 = int list, τ_2 = unit list

Fun Rule

- Rules describe types, but also how the environment Γ may change
- Can only do what rule allows!
- fun rule:

$$\frac{[x : \tau_1] + \Gamma \mid\!\!- \; e : \tau_2}{\Gamma \mid\!\!- \text{fun } x \text{ -> } e : \tau_1 \rightarrow \tau_2}$$

Fun Examples

$$\frac{[y : int\ ] + \Gamma \ |\!-\ y + 3\ :\ int}{\Gamma \ |\!-\ \text{fun } y \text{ -> } y + 3\ :\ int \rightarrow int}$$

$$\frac{[f : int \rightarrow bool] + \Gamma \ |\!-\ f\ 2 :: [true]\ :\ bool\ list}{\Gamma \ |\!-\ (\text{fun } f \text{ -> } f\ 2 :: [true])\ :\ (int \rightarrow bool) \rightarrow bool\ list}$$

(Monomorphic) Let and Let Rec

- let rule:

$$\frac{\Gamma \mid\!- \; e_1 : \tau_1 \qquad [x : \tau_1] + \Gamma \mid\!- \; e_2 : \tau_2}{\Gamma \mid\!- \; (\text{let } x = e_1 \text{ in } e_2) : \tau_2}$$

- let rec rule:

$$\frac{[x: \tau_1] + \Gamma \mid\!- \; e_1 : \tau_1 \quad [x: \tau_1] + \Gamma \mid\!- \; e_2 : \tau_2}{\Gamma \mid\!- \; (\text{let rec } x = e_1 \text{ in } e_2) : \tau_2}$$

Example (Assume Prim Op rule for ::)

- Which rule do we apply?

$$\frac{?}{\begin{array}{l}\text{|- (let rec one = 1 :: one in} \\ \quad \text{let x = 2 in} \\ \qquad \text{fun y -> (x :: y :: one)) : int} \rightarrow \text{int list}\end{array}}$$

Example

- Let rec rule:

(2) [one : int list] |- (let x = 2 in fun y -> (x :: y :: one)) : int → int list

(1) [one : int list] |- (1 :: one) : int list

|- (let rec one = 1 :: one in
let x = 2 in
fun y -> (x :: y :: one)) : int → int list

Proof of 1

- Which rule?

[one : int list] |- (1 :: one) : int list

Proof of 1

- Prim Op rule for :: (at int)

③ ④

$$\frac{[\text{one} : \text{int list}] \mid\!- \; 1 : \text{int} \qquad \begin{array}{c}[\text{one} : \text{int list}] \mid\!- \\ \text{one} : \text{int list}\end{array}}{[\text{one} : \text{int list}] \mid\!- \; (1 :: \text{one}) : \text{int list}}$$

Proof of 3

- Constant Rule

3 4

$$\dfrac{\dfrac{}{[\text{one} : \text{int list}] \;|\text{-}\; 1 : \text{int}} \qquad \begin{array}{c}[\text{one} : \text{int list}] \;|\text{-} \\ \text{one} : \text{int list}\end{array}}{[\text{one} : \text{int list}] \;|\text{-}\; (1 :: \text{one}) : \text{int list}}$$

Proof of 4

- Rule for variables

$$\frac{}{[\text{one} : \text{int list}] \mid\!- \text{one}:\text{int list}}$$

Proof of 2

5 [x:int; one : int list] |- fun y -> (x :: y :: one))

- Constant

$$\dfrac{\dfrac{}{\text{[one : int list] |- 2:int}} \qquad \text{[x:int; one : int list] |- fun y -> (x :: y :: one)) : int} \rightarrow \text{int list}}{\text{[one : int list] |- (let x = 2 in fun y -> (x :: y :: one)) : int} \rightarrow \text{int list}}$$

Proof of 5

Use function rule

$$\frac{?}{[x:int;\ one : int\ list]\ |\text{-}\ fun\ y \text{ -> } (x :: y :: one))\ : int \rightarrow int\ list}$$

Proof of 5

Use Prim Op rule for ::

$$\frac{\dfrac{?}{[y:int;\ x:int;\ one : int\ list] \mid\!- (x :: y :: one) : int\ list}}{[x:int;\ one : int\ list] \mid\!- fun\ y \text{ -> } (x :: y :: one)) : int \rightarrow int\ list}$$

Proof of 5

Use Prim Op rule for ::

6 7

[y:int; x:int; one:int list]|- x:int   [y:int; x:int; one:int list] |- (y :: one) : int list

---

[y:int; x:int; one : int list] |- (x :: y :: one) : int list

[x:int; one : int list] |- fun y -> (x :: y :: one)) : int → int list

Proof of 6

6 Variable Rule   7

$$\frac{\overline{[y{:}int;\ x{:}int;\ one{:}int\ list] \mid\!-\ x{:}int} \qquad \begin{array}{c}[y{:}int;\ x{:}int;\ one{:}int\ list] \mid\!- \\ (y :: one) : int\ list\end{array}}{\dfrac{[y{:}int;\ x{:}int;\ one : int\ list] \mid\!- (x :: y :: one) : int\ list}{[x{:}int;\ one : int\ list] \mid\!- fun\ y \to (x :: y :: one)) : int \to int\ list}}$$

Proof of 7

Prim Op Rule ::

8 9

[y:int; x:int; one:int list]|- y:int   [y:int; x:int; one:int list] |- one : int list

---

[y:int; x:int; one : int list] |- (y :: one)): int list

Proof of 8 and 9

Variable Rule

8

$$\frac{}{\text{[y:int; x:int; one:int list]|- y:int}}$$

9

$$\frac{}{\text{[y:int; x:int; one:int list] |- (y :: one) : int list}}$$

$$\frac{\text{[y:int; x:int; one:int list]|- y:int} \qquad \text{[y:int; x:int; one:int list] |- (y :: one) : int list}}{\text{[y:int; x:int; one : int list] |- (y :: one)): int list}}$$

Curry - Howard Isomorphism

- Type Systems are logics; logics are type systems
- Types are propositions; propositions are types
- Terms are proofs; proofs are terms

- Functions space arrow corresponds to implication; application corresponds to modus ponens

Curry - Howard Isomorphism

- Modus Ponens

$$\frac{A \Rightarrow B \quad A}{B}$$

- Application

$$\frac{\Gamma \mid\!- \; e_1 : \alpha \rightarrow \beta \quad \Gamma \mid\!- \; e_2 : \alpha}{\Gamma \mid\!- \; (e_1 \; e_2) : \beta}$$

Mia Copa

- The above system can't handle polymorphism as in OCAML
- No type variables in type language (only meta-variable in the logic)
- Would need:
  - Object level type variables and some kind of type quantification
  - **let** and **let rec** rules to introduce polymorphism
  - Explicit rule to eliminate (instantiate) polymorphism

Support for Polymorphic Types

- Monomorpic Types (τ):
  - Basic Types: int, bool, float, string, unit, …
  - Type Variables: α, β, γ δ ε
  - Compound Types: $\alpha \rightarrow \beta$, int * string, bool list, …
- Polymorphic Types:
  - Monomorphic types τ
  - Universally quantified monomorphic types
  - $\forall \alpha_1, \ldots, \alpha_n . \tau$
  - Can think of τ as same as $\forall . \tau$

Support for Polymorphic Types

- Typing Environment Γ supplies polymorphic types (which will often just be monomorphic) for variables
- Free variables of monomorphic type just type variables that occur in it
  - Write FreeVars(τ)
- Free variables of polymorphic type removes variables that are universally quantified
  - $\mathrm{FreeVars}(\forall \alpha_1, \ldots, \alpha_n . \tau) = \mathrm{FreeVars}(\tau) - \{\alpha_1, \ldots, \alpha_n\}$
- FreeVars(Γ) = all FreeVars of types in range of Γ

Monomorphic to Polymorphic

- Given:
  - type environment Γ
  - monomorphic type τ
  - τ shares type variables with Γ
- Want most polymorphic type for τ that doesn't break sharing type variables with Γ
- $\mathrm{Gen}(\tau, \Gamma) = \forall \alpha_1, \ldots, \alpha_n . \tau$ where $\{\alpha_1, \ldots, \alpha_n\} = \mathrm{freeVars}(\tau) - \mathrm{freeVars}(\Gamma)$

Polymorphic Typing Rules

- A *type judgement* has the form

$$\Gamma \mid\!- \text{exp} : \tau$$

  - Γ uses polymorphic types
  - τ still monomorphic
- Most rules stay same (except use more general typing environments)
- Rules that change:
  - Variables
  - Let and Let Rec
  - Allow polymorphic constants
- Worth noting functions again

Polymorphic Let and Let Rec

- let rule:

$$\frac{\Gamma \mid\!- \; e_1 : \tau_1 \quad [x : \mathrm{Gen}(\tau_1,\Gamma)] + \Gamma \mid\!- \; e_2 : \tau_2}{\Gamma \mid\!- \; (\text{let } x = e_1 \text{ in } e_2) : \tau_2}$$

- let rec rule:

$$\frac{[x: \tau_1\,] + \Gamma \mid\!- \; e_1 : \tau_1 \quad [x : \mathrm{Gen}(\tau_1,\Gamma)] + \Gamma \mid\!- \; e_2 : \tau_2}{\Gamma \mid\!- \; (\text{let rec } x = e_1 \text{ in } e_2) : \tau_2}$$

Polymorphic Variables (Identifiers)

Variable axiom:

$$\frac{}{\Gamma \mid\!- \; x : \varphi(\tau)} \quad \text{if } \Gamma(x) = \forall \alpha_1, \ldots, \alpha_n \,.\, \tau$$

- Where φ replaces all occurrences of $\alpha_1, \ldots, \alpha_n$ by monotypes $\tau_1, \ldots, \tau_n$
- Note: Monomorphic rule special case:

$$\frac{}{\Gamma \mid\!- \; x : \tau} \quad \text{if } \Gamma(x) = \tau$$

- Constants treated same way

Fun Rule Stays the Same

- fun rule:

$$\frac{[x : \tau_1] + \Gamma \mid\!- \; e : \tau_2}{\Gamma \mid\!- \text{fun } x \text{ -> } e : \tau_1 \rightarrow \tau_2}$$

- Types τ_1, τ_2 monomorphic
- Function argument must always be used at same type in function body

Polymorphic Example

- Assume additional constants:
- hd : $\forall\alpha.\ \alpha$ list -> α
- tl: $\forall\alpha.\ \alpha$ list -> α list
- is_empty : $\forall\alpha.\ \alpha$ list -> bool
- :: : $\forall\alpha.\ \alpha$ -> α list -> α list
- [] : $\forall\alpha.\ \alpha$ list

Polymorphic Example

- Show:

$$\frac{?}{\text{\{\} |- let rec length = fun l -> if is\_empty l then 0 else 1 + length (tl l) in length ((::) 2 []) + length((::) true []) : int}}$$

```
{} |- let rec length =
        fun l -> if is_empty l then 0
                 else 1 + length (tl l)
  in length ((::) 2 []) + length((::) true []) : int
```

Polymorphic Example: Let Rec Rule

- Show: (1) (2)

(1)

{length:α list -> int}
|- fun l -> …
: α list -> int

(2)

{length:$\forall\alpha$. α list -> int}
|- length ((::) 2 []) +
length((::) true []) : int

{} |- let rec length =
fun l -> if is_empty l then 0
else 1 + length (tl l)
in length ((::) 2 []) + length((::) true []) : int

Polymorphic Example (1)

- Show:

$$\frac{?}{\{\text{length}:\alpha \text{ list -> int}\} \text{ |- fun l -> if is\_empty l then 0 else 1 + length (tl l)} : \alpha \text{ list -> int}}$$

Polymorphic Example (1): Fun Rule

- Show: (3)

{length: α list -> int, l: α list } |-
if is_empty l then 0
else length (hd l) + length (tl l) : int

{length: α list -> int} |-
fun l -> if is_empty l then 0
else 1 + length (tl l)
: α list -> int

Polymorphic Example (3)

- Let Γ ={length:α list -> int, l: α list }
- Show

$$\frac{?}{\Gamma \vdash \text{if is\_empty l then 0 else 1 + length (tl l) : int}}$$

Polymorphic Example (3):IfThenElse

- Let Γ ={length:α list -> int, l: α list }
- Show

$$\frac{\begin{array}{ccc} (4) & (5) & (6) \\ \Gamma \mid\!-\ \text{is\_empty l} & \Gamma \mid\!-\ 0{:}\text{int} & \Gamma \mid\!-\ 1\ + \\ :\ \text{bool} & & \text{length (tl l)}\ :\ \text{int} \end{array}}{\begin{array}{c} \Gamma \mid\!-\ \text{if is\_empty l then 0} \\ \text{else 1 + length (tl l)}\ :\ \text{int} \end{array}}$$

Polymorphic Example (4)

- Let Γ ={length:α list -> int, l: α list }
- Show

$$\frac{?}{\Gamma \mid\text{- is\_empty l : bool}}$$

Polymorphic Example (4):Application

- Let Γ ={length:α list -> int, l: α list }
- Show

$$\dfrac{\dfrac{?}{\Gamma |\!- \text{is\_empty} : \alpha \text{ list} \to \text{bool}} \qquad \dfrac{?}{\Gamma |\!- \text{l} : \alpha \text{ list}}}{\Gamma |\!- \text{is\_empty l} : \text{bool}}$$

Polymorphic Example (4)

- Let Γ ={length:α list -> int, l: α list }
- Show

By Const since α list -> bool is
instance of $\forall\alpha.\ \alpha$ list -> bool ?

$$\dfrac{\dfrac{}{\Gamma \mid\!- \text{is\_empty} : \alpha \text{ list} \to \text{bool}} \qquad \dfrac{}{\Gamma \mid\!- \text{l} : \alpha \text{ list}}}{\Gamma \mid\!- \text{is\_empty l} : \text{bool}}$$

Polymorphic Example (4)

- Let Γ ={length:α list -> int, l: α list }
- Show

$$\dfrac{\dfrac{\begin{array}{c}\text{By Const since } \alpha \text{ list -> bool is}\\ \text{instance of } \forall\alpha.\ \alpha \text{ list -> bool}\end{array}}{\Gamma|\text{- is\_empty : } \alpha \text{ list -> bool}} \qquad \dfrac{\begin{array}{c}\text{By Variable}\\ \Gamma(\text{l}) = \alpha \text{ list}\end{array}}{\Gamma|\text{- l : } \alpha \text{ list}}}{\Gamma|\text{- is\_empty l : bool}}$$

- This finishes (4)

Polymorphic Example (5):Const

- Let Γ ={length:α list -> int, l: α list }
- Show

By Const Rule

$$\frac{}{\Gamma \mid\!- \ 0\!:\!int}$$

Polymorphic Example (6):Arith Op

- Let Γ ={length:α list -> int, l: α list }
- Show

$$\dfrac{\dfrac{\text{By Const}}{\Gamma \mid\!- \ 1:\text{int}} \qquad \dfrac{\dfrac{\text{By Variable}}{\Gamma \mid\!- \ \text{length} : \alpha \ \text{list} \to \text{int}} \ (7) \qquad \Gamma \mid\!- \ (\text{tl l}) : \alpha \ \text{list}}{\Gamma \mid\!- \ \text{length (tl l)} : \text{int}}}{\Gamma \mid\!- \ 1 + \text{length (tl l)} : \text{int}}$$

Polymorphic Example (7):App Rule

- Let Γ ={length:α list -> int, l: α list }
- Show

$$\dfrac{\dfrac{\text{By Const}}{\Gamma \mid\!- \text{(tl l)} : \alpha \text{ list} \to \alpha \text{ list}} \qquad \dfrac{\text{By Variable}}{\Gamma \mid\!- \text{l} : \alpha \text{ list}}}{\Gamma \mid\!- \text{(tl l)} : \alpha \text{ list}}$$

By Const since α list -> α list is instance of $\forall\alpha.\ \alpha$ list -> α list

Polymorphic Example: (2) by ArithOp

- Let Γ' = {length:$\forall\ \alpha.\ \alpha$ list -> int}
- Show:

(8) (9)

Γ' |- Γ' |-

length ((::) 2 []) :int length((::) true []) : int

{length: $\alpha.\ \alpha$ list -> int}

|- length ((::) 2 []) + length((::) true []) : int

Polymorphic Example: (8)AppRule

- Let Γ' = {length:$\forall\ \alpha.\ \alpha$ list -> int}
- Show:

$$\frac{\Gamma' \text{ |- length : int list ->int} \quad \Gamma' \text{ |- ((::)2 []):int list}}{\Gamma' \text{ |- length ((::) 2 []) :int}}$$

Polymorphic Example: (8)AppRule

- Let Γ' = {length:$\forall\, \alpha.\ \alpha$ list -> int}
- Show:

By Var since int list -> int is instance of $\forall\alpha.\ \alpha$ list -> int

$$\dfrac{\dfrac{}{\Gamma' \mid\!\!-\ \text{length : int list ->int list}} \qquad \overset{(10)}{\Gamma' \mid\!\!-\ ((::)2\ []):\text{int list}}}{\Gamma' \mid\!\!-\ \text{length } ((::)\ 2\ [])\ :\text{int}}$$

Polymorphic Example: (10)AppRule

- Let Γ' = {length:$\forall\, \alpha.\ \alpha$ list -> int}
- Show:
- By Const since α list is instance of $\forall\alpha.\ \alpha$ list

$$\dfrac{\Gamma' \text{ |-((::) 2) : int list -> int list} \qquad \dfrac{(11)}{\Gamma' \text{ |- [] : int list}}}{\Gamma' \text{ |- ((::) 2 []) :int list}}$$

Polymorphic Example: (11)AppRule

- Let Γ' = {length:$\forall\, \alpha.\ \alpha$ list -> int}
- Show:
- By Const since α list
  is instance of
  $\forall\alpha.\ \alpha$ list

$$\dfrac{\dfrac{\forall\alpha.\ \alpha \text{ list}}{\Gamma' \mid\!\!- (::) : \text{int} \to \text{int list} \to \text{int list}} \qquad \dfrac{\text{By Const}}{\Gamma' \mid\!\!- 2 : \text{int}}}{\Gamma' \mid\!\!- ((::)\ 2) : \text{int list} \to \text{int list}}$$

Polymorphic Example: (9)AppRule

- Let Γ' = {length:$\forall\, \alpha.\ \alpha$ list -> int}
- Show:

$$\frac{\Gamma' \text{ |- length:bool list ->int} \qquad \Gamma' \text{ |- ((::) true []):bool list}}{\Gamma' \text{ |- length ((::) true []) :int}}$$

Polymorphic Example: (9)AppRule

- Let Γ' = {length:$\forall\ \alpha.\ \alpha$ list -> int}
- Show:

By Var since bool list -> int is instance of
$\forall\alpha.\ \alpha$ list -> int

$$\dfrac{\dfrac{}{\Gamma'\ \vdash \text{length:bool list -> int}} \qquad \dfrac{(12)}{\Gamma'\ \vdash \text{((::) true []):bool list}}}{\Gamma'\ \vdash \text{length ((::) true []) :int}}$$

Polymorphic Example: (12)AppRule

- Let Γ' = {length:$\forall\ \alpha.\ \alpha$ list -> int}
- Show:
- By Const since α list is instance of $\forall\alpha.\ \alpha$ list

$$\dfrac{\dfrac{(13)}{\Gamma' \text{ |-((::)true):bool list ->bool list}} \quad \dfrac{}{\Gamma' \text{ |- []:bool list}}}{\Gamma' \text{ |- ((::) true []) :bool list}}$$

Polymorphic Example: (13)AppRule

- Let Γ' = {length:$\forall\,\alpha.\ \alpha$ list -> int}
- Show:

$$\dfrac{\dfrac{\text{By Const since bool list is instance of } \forall\alpha.\ \alpha \text{ list}}{\Gamma' \mid\!-\ (::):\text{bool} \to \text{bool list} \to \text{bool list}} \qquad \dfrac{\text{By Const}}{\Gamma' \mid\!-\ \text{true} : \text{bool}}}{\Gamma' \mid\!-\ ((::)\ \text{true}) : \text{bool list} \to \text{bool list}}$$