Type Systems
07 – Type Reconstruction

Jörn W. Janneck

Computer Science Dept.

Lund University

Slides begged, borrowed, and copied from the Type Systems course at EPFL.
Courtesy of Martin Odersky.
Some material also lifted from TAPL by Benjamin C. Pierce.

Type Reconstruction

Type Checking and Type Reconstruction

We now come to the question of type checking and type reconstruction.

Type checking: Given Γ, t and T, check whether $\Gamma \vdash t : T$

Type reconstruction: Given Γ and t, find a type T such that $\Gamma \vdash t : T$

Type checking and reconstruction seem difficult since parameters in lambda calculus do not carry their types with them.

Type reconstruction also suffers from the problem that a term can have many types.

Idea: : We construct all type derivations in parallel, reducing type reconstruction to a unification problem.

substitutions

Substitution: mapping σ from type variables to types

eg. {T : Nat, U : X → Nat, V : Bool → Bool}

extends naturally to mappings on

types:

$$
\begin{aligned}
\sigma(\mathsf{X}) &= \begin{cases} \mathsf{T} & \text{if } (\mathsf{X} \mapsto \mathsf{T}) \in \sigma \\ \mathsf{X} & \text{if X is not in the domain of } \sigma \end{cases} \\
\sigma(\mathsf{Nat}) &= \mathsf{Nat} \\
\sigma(\mathsf{Bool}) &= \mathsf{Bool} \\
\sigma(\mathsf{T}_1 \to \mathsf{T}_2) &= \sigma\mathsf{T}_1 \to \sigma\mathsf{T}_2
\end{aligned}
$$

contexts:

$$\sigma(x_1{:}\mathsf{T}_1, \ldots, x_n{:}\mathsf{T}_n) = (x_1{:}\sigma\mathsf{T}_1, \ldots, x_n{:}\sigma\mathsf{T}_n).$$

type equations:

$$\sigma(S == T) = \sigma(S) == \sigma(T)$$

22.1.2 THEOREM [PRESERVATION OF TYPING UNDER TYPE SUBSTITUTION]: If σ is any type substitution and $\Gamma \vdash \mathtt{t} : \mathsf{T}$, then $\sigma\Gamma \vdash \sigma\mathtt{t} : \sigma\mathsf{T}$. □

solutions

22.2.1 **Definition:** Let Γ be a context and $\mathtt{t}$ a term. A *solution* for $(\Gamma, \mathtt{t})$ is a pair (σ, T) such that $\sigma\Gamma \vdash \sigma\mathtt{t} : \mathsf{T}$. □

How do we find a solution (or all solutions) for some (Γ, t)?

constraint-based typing

Problem:

Given (Γ, t) find solution (σ, T).

Basic approach:

1. transform problem $\Gamma \vdash t : T$ into set of equations (aka constraints)

$$\Gamma \vdash \mathtt{t} : \mathsf{T} \;\; |_{\chi} \; C$$

2. compute the solutions for the equations

constraint-based typing: equations

$$\Gamma \vdash \mathtt{t} : \mathsf{T} \;|_{\mathcal{X}}\; C$$

$$\frac{\mathtt{x}{:}\mathsf{T} \in \Gamma}{\Gamma \vdash \mathtt{x} : \mathsf{T} \;|_{\varnothing}\; \{\}} \quad \text{(CT-VAR)}$$

$$\frac{\Gamma, \mathtt{x}{:}\mathsf{T}_1 \vdash \mathtt{t}_2 : \mathsf{T}_2 \;|_{\mathcal{X}}\; C}{\Gamma \vdash \lambda \mathtt{x}{:}\mathsf{T}_1 . \mathtt{t}_2 : \mathsf{T}_1 \to \mathsf{T}_2 \;|_{\mathcal{X}}\; C} \quad \text{(CT-ABS)}$$

$$\frac{\begin{array}{c} \Gamma \vdash \mathtt{t}_1 : \mathsf{T}_1 \;|_{\mathcal{X}_1}\; C_1 \qquad \Gamma \vdash \mathtt{t}_2 : \mathsf{T}_2 \;|_{\mathcal{X}_2}\; C_2 \\ \mathcal{X}_1 \cap \mathcal{X}_2 = \mathcal{X}_1 \cap FV(\mathsf{T}_2) = \mathcal{X}_2 \cap FV(\mathsf{T}_1) = \varnothing \\ \mathsf{X} \notin \mathcal{X}_1, \mathcal{X}_2, \mathsf{T}_1, \mathsf{T}_2, C_1, C_2, \Gamma, \mathtt{t}_1, \text{or } \mathtt{t}_2 \\ C' = C_1 \cup C_2 \cup \{\mathsf{T}_1 = \mathsf{T}_2 \to \mathsf{X}\} \end{array}}{\Gamma \vdash \mathtt{t}_1\ \mathtt{t}_2 : \mathsf{X} \;|_{\mathcal{X}_1 \cup \mathcal{X}_2 \cup \{\mathsf{X}\}}\; C'} \quad \text{(CT-APP)}$$

$$\Gamma \vdash \mathtt{0} : \mathsf{Nat} \;|_{\varnothing}\; \{\} \quad \text{(CT-ZERO)}$$

$$\frac{\begin{array}{c} \Gamma \vdash \mathtt{t}_1 : \mathsf{T} \;|_{\mathcal{X}}\; C \\ C' = C \cup \{\mathsf{T} = \mathsf{Nat}\} \end{array}}{\Gamma \vdash \mathtt{succ}\ \mathtt{t}_1 : \mathsf{Nat} \;|_{\mathcal{X}}\; C'} \quad \text{(CT-SUCC)}$$

$$\frac{\begin{array}{c} \Gamma \vdash \mathtt{t}_1 : \mathsf{T} \;|_{\mathcal{X}}\; C \\ C' = C \cup \{\mathsf{T} = \mathsf{Nat}\} \end{array}}{\Gamma \vdash \mathtt{pred}\ \mathtt{t}_1 : \mathsf{Nat} \;|_{\mathcal{X}}\; C'} \quad \text{(CT-PRED)}$$

$$\frac{\begin{array}{c} \Gamma \vdash \mathtt{t}_1 : \mathsf{T} \;|_{\mathcal{X}}\; C \\ C' = C \cup \{\mathsf{T} = \mathsf{Nat}\} \end{array}}{\Gamma \vdash \mathtt{iszero}\ \mathtt{t}_1 : \mathsf{Bool} \;|_{\mathcal{X}}\; C'} \quad \text{(CT-ISZERO)}$$

$$\Gamma \vdash \mathtt{true} : \mathsf{Bool} \;|_{\varnothing}\; \{\} \quad \text{(CT-TRUE)}$$

$$\Gamma \vdash \mathtt{false} : \mathsf{Bool} \;|_{\varnothing}\; \{\} \quad \text{(CT-FALSE)}$$

$$\frac{\begin{array}{c} \Gamma \vdash \mathtt{t}_1 : \mathsf{T}_1 \;|_{\mathcal{X}_1}\; C_1 \\ \Gamma \vdash \mathtt{t}_2 : \mathsf{T}_2 \;|_{\mathcal{X}_2}\; C_2 \qquad \Gamma \vdash \mathtt{t}_3 : \mathsf{T}_3 \;|_{\mathcal{X}_3}\; C_3 \\ \mathcal{X}_1, \mathcal{X}_2, \mathcal{X}_3 \text{ nonoverlapping} \\ C' = C_1 \cup C_2 \cup C_3 \cup \{\mathsf{T}_1 = \mathsf{Bool}, \mathsf{T}_2 = \mathsf{T}_3\} \end{array}}{\Gamma \vdash \mathtt{if}\ \mathtt{t}_1\ \mathtt{then}\ \mathtt{t}_2\ \mathtt{else}\ \mathtt{t}_3 : \mathsf{T}_2 \;|_{\mathcal{X}_1 \cup \mathcal{X}_2 \cup \mathcal{X}_3}\; C'} \quad \text{(CT-IF)}$$

Figure 22-1: Constraint typing rules

constraint-based typing: equations

$\Gamma \vdash \mathtt{t} : \mathtt{T} \;|_{\mathcal{X}}\; C$

$$\Gamma \vdash \mathtt{true} : \mathtt{Bool} \;|_{\varnothing}\; \{\} \qquad \text{(CT-True)}$$

$$\Gamma \vdash \mathtt{false} : \mathtt{Bool} \;|_{\varnothing}\; \{\} \qquad \text{(CT-False)}$$

$$\Gamma \vdash \mathtt{0} : \mathtt{Nat} \;|_{\varnothing}\; \{\} \qquad \text{(CT-Zero)}$$

constraint-based typing: equations

$\Gamma \vdash \mathtt{t} : \mathsf{T} \ |_{\mathcal{X}} \ C$

$$\frac{\Gamma \vdash \mathtt{t}_1 : \mathsf{T} \ |_{\mathcal{X}} \ C \qquad C' = C \cup \{\mathsf{T} = \mathsf{Nat}\}}{\Gamma \vdash \mathtt{succ}\ \mathtt{t}_1 : \mathsf{Nat} \ |_{\mathcal{X}} \ C'} \quad \text{(CT-Succ)}$$

$$\frac{\Gamma \vdash \mathtt{t}_1 : \mathsf{T} \ |_{\mathcal{X}} \ C \qquad C' = C \cup \{\mathsf{T} = \mathsf{Nat}\}}{\Gamma \vdash \mathtt{pred}\ \mathtt{t}_1 : \mathsf{Nat} \ |_{\mathcal{X}} \ C'} \quad \text{(CT-Pred)}$$

$$\frac{\Gamma \vdash \mathtt{t}_1 : \mathsf{T} \ |_{\mathcal{X}} \ C \qquad C' = C \cup \{\mathsf{T} = \mathsf{Nat}\}}{\Gamma \vdash \mathtt{iszero}\ \mathtt{t}_1 : \mathsf{Bool} \ |_{\mathcal{X}} \ C'} \quad \text{(CT-IsZero)}$$

constraint-based typing: equations

$\Gamma \vdash \mathtt{t} : \mathtt{T} \;|_{\mathcal{X}}\; C$

$$\frac{\mathtt{x}{:}\mathtt{T} \in \Gamma}{\Gamma \vdash \mathtt{x} : \mathtt{T} \;|_{\varnothing}\; \{\}} \qquad \text{(CT-Var)}$$

$$\frac{\Gamma, \mathtt{x}{:}\mathtt{T}_1 \vdash \mathtt{t}_2 : \mathtt{T}_2 \;|_{\mathcal{X}}\; C}{\Gamma \vdash \lambda \mathtt{x}{:}\mathtt{T}_1.\mathtt{t}_2 : \mathtt{T}_1 \to \mathtt{T}_2 \;|_{\mathcal{X}}\; C} \qquad \text{(CT-Abs)}$$

constraint-based typing: equations

$\Gamma \vdash \mathtt{t} : \mathtt{T} \ |_{\mathcal{X}}\ C$

$$\frac{\begin{array}{c}\Gamma \vdash \mathtt{t}_1 : \mathtt{T}_1 \ |_{\mathcal{X}_1}\ C_1 \\ \Gamma \vdash \mathtt{t}_2 : \mathtt{T}_2 \ |_{\mathcal{X}_2}\ C_2 \qquad \Gamma \vdash \mathtt{t}_3 : \mathtt{T}_3 \ |_{\mathcal{X}_3}\ C_3 \\ \mathcal{X}_1, \mathcal{X}_2, \mathcal{X}_3 \text{ nonoverlapping} \\ C' = C_1 \cup C_2 \cup C_3 \cup \{\mathtt{T}_1 = \mathtt{Bool}, \mathtt{T}_2 = \mathtt{T}_3\}\end{array}}{\Gamma \vdash \mathtt{if}\ \mathtt{t}_1\ \mathtt{then}\ \mathtt{t}_2\ \mathtt{else}\ \mathtt{t}_3 : \mathtt{T}_2 \ |_{\mathcal{X}_1 \cup \mathcal{X}_2 \cup \mathcal{X}_3}\ C'}$$

(CT-IF)

constraint-based typing: equations

$\Gamma \vdash \mathtt{t} : \mathsf{T} \;|_{\mathcal{X}}\; C$

$$\frac{\begin{array}{c}\Gamma \vdash \mathtt{t}_1 : \mathsf{T}_1 \;|_{\mathcal{X}_1}\; C_1 \qquad \Gamma \vdash \mathtt{t}_2 : \mathsf{T}_2 \;|_{\mathcal{X}_2}\; C_2 \\ \mathcal{X}_1 \cap \mathcal{X}_2 = \mathcal{X}_1 \cap FV(\mathsf{T}_2) = \mathcal{X}_2 \cap FV(\mathsf{T}_1) = \emptyset \\ \mathsf{X} \notin \mathcal{X}_1, \mathcal{X}_2, \mathsf{T}_1, \mathsf{T}_2, C_1, C_2, \Gamma, \mathtt{t}_1, \text{or } \mathtt{t}_2 \\ C' = C_1 \cup C_2 \cup \{\mathsf{T}_1 = \mathsf{T}_2 \to \mathsf{X}\}\end{array}}{\Gamma \vdash \mathtt{t}_1\; \mathtt{t}_2 : \mathsf{X} \;|_{\mathcal{X}_1 \cup \mathcal{X}_2 \cup \{\mathsf{X}\}}\; C'} \quad \text{(CT-APP)}$$

constraint-based typing

Problem:

Given (Γ, t) find solution (σ, T).

Basic approach:

1. transform problem $\Gamma \vdash t : T$ into set of equations (aka constraints)

$$\Gamma \vdash \mathtt{t} : \mathsf{T} \ |_{\chi} \ C$$

2. compute the solutions for the equations

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

Constraint-based typing: unification

Recall: a substitution is a mapping from type variables to types.

Definition: A substitution σ *unifies* a set of equations “$S_i = T_i$” iff $\sigma S_i = \sigma T_i$ for all i.

Problem: Given a set of equations C, find σ that unifies C.

Subproblem: There might be more than one.

Constraint-based typing: unification

$$
\begin{array}{lll}
\textit{unify}(C) & = & \text{if } C = \emptyset, \text{ then } [\,] \\
& & \text{else let } \{\mathsf{S} = \mathsf{T}\} \cup C' = C \text{ in} \\
& & \quad \text{if } \mathsf{S} = \mathsf{T} \\
& & \qquad \text{then } \textit{unify}(C') \\
& & \quad \text{else if } \mathsf{S} = \mathsf{X} \text{ and } \mathsf{X} \notin \textit{FV}(\mathsf{T}) \\
& & \qquad \text{then } \textit{unify}([\mathsf{X} \mapsto \mathsf{T}]C') \circ [\mathsf{X} \mapsto \mathsf{T}] \\
& & \quad \text{else if } \mathsf{T} = \mathsf{X} \text{ and } \mathsf{X} \notin \textit{FV}(\mathsf{S}) \\
& & \qquad \text{then } \textit{unify}([\mathsf{X} \mapsto \mathsf{S}]C') \circ [\mathsf{X} \mapsto \mathsf{S}] \\
& & \quad \text{else if } \mathsf{S} = \mathsf{S}_1 \rightarrow \mathsf{S}_2 \text{ and } \mathsf{T} = \mathsf{T}_1 \rightarrow \mathsf{T}_2 \\
& & \qquad \text{then } \textit{unify}(C' \cup \{\mathsf{S}_1 = \mathsf{T}_1, \mathsf{S}_2 = \mathsf{T}_2\}) \\
& & \quad \text{else} \\
& & \qquad \textit{fail}
\end{array}
$$

Figure 22-2: Unification algorithm

recap

type checking
Given Γ, t, and T, check whether

type reconstruction
Given Γ, and t, find T such that

$$\Gamma \vdash \mathtt{t} : \mathtt{T}$$

$\Gamma \vdash \mathtt{t} : \mathtt{T}$ ⇒ $\Gamma \vdash \mathtt{t} : \mathtt{T} \mid_{\chi} C$ ⇒ σ = unify(C)

(σ, σT)

two problems, two solutions

22.2.1 DEFINITION: Let Γ be a context and $\mathtt{t}$ a term. A *solution* for $(\Gamma, \mathtt{t})$ is a pair (σ, T) such that $\sigma\Gamma \vdash \sigma\mathtt{t} : \mathsf{T}$. □

$$\Gamma \vdash \mathtt{t} : \mathsf{T} \quad \Rightarrow \quad \Gamma \vdash \mathtt{t} : \mathsf{T} \;|_{\mathcal{X}}\; C \quad \Rightarrow \quad \sigma = \text{unify}(C)$$

22.3.4 DEFINITION: Suppose that $\Gamma \vdash \mathtt{t} : \mathsf{S} \mid C$. A *solution* for $(\Gamma, \mathtt{t}, \mathsf{S}, C)$ is a pair (σ, T) such that σ satisfies C and $\sigma\mathsf{S} = \mathsf{T}$. □

soundness and completeness

Soundness: If (σ, σT) is a solution for (Γ, t, T, C), then it is also a solution for (Γ, t).

(Theorem 22.3.5, p.323)

Completeness: If (σ, T) is a solution for (Γ, t), then there is a solution (σ', $\sigma' S$) for (Γ, t, S, C) such that $T = \sigma' S$ and $\sigma = \sigma'|_{dom(\sigma)}$.

(Theorem 22.3.7, p. 324)

principal unifier

preorder substitutions based on their 'specificity':

$\sigma \leq \sigma'$ iff exists γ such that $\gamma \circ \sigma = \sigma'$

principal unifier for C:
some σ such that
1. σ satisfies C and
2. for all σ' satisfying C, $\sigma \leq \sigma'$

unification theorem: (22.4.5, p.328)
1. unify(C) halts for all C
2. it only fails if there is no unifier for C
3. σ = unify(C) implies σ is a unifier for C
4. unify(C) is a principal unifier for C

principal types

principal solution, principal type:
A solution (σ, T) of (Γ, t, S, C) is a *principal solution* iff for all other solutions (σ', T) it is the case that $\sigma \leq \sigma'$. T is then the *principal type* of t under Γ.

(Def 22.5.1)

principal type theorem:
If (Γ, t, S, C) has a solution, it has a principal solution, and it's the one found by unify(C).

(Theorem 22.5.3)

implicit type annotations

$$\frac{\mathtt{X} \notin \mathcal{X} \qquad \Gamma, \mathtt{x}{:}\mathtt{X} \vdash \mathtt{t}_1 : \mathtt{T} \ |_{\mathcal{X}} \ C}{\Gamma \vdash \lambda\mathtt{x}.\mathtt{t}_1 : \mathtt{X}{\to}\mathtt{T} \ |_{\mathcal{X} \cup \{\mathtt{X}\}} \ C} \qquad \text{(CT-AbsInf)}$$

an odd case

$$(\lambda x{:}A.\ xx)(\lambda x{:}B.\ xx)$$