# Unification

Nao Hirokawa

JAIST



## Composition of Substitutions and Subsumption

**Definition**

$\sigma\tau = \{x \mapsto (x\sigma)\tau \mid x \in \mathcal{V}\}$

**Example**

$\sigma = \{x \mapsto \mathsf{s}(y), y \mapsto x + \mathsf{s}(0)\}$ $\quad\tau = \{x \mapsto \mathsf{s}(0), z \mapsto \mathsf{s}(\mathsf{s}(y))\}$

- $\sigma\tau = \{x \mapsto \mathsf{s}(y), y \mapsto \mathsf{s}(0) + \mathsf{s}(0), z \mapsto \mathsf{s}(\mathsf{s}(y))\}$
- $\tau\sigma = \{x \mapsto \mathsf{s}(0), y \mapsto x + \mathsf{s}(0), z \mapsto \mathsf{s}(\mathsf{s}(x + \mathsf{s}(0)))\}$

**Definition**

$\sigma \leq \tau \iff \exists\rho : \sigma\rho = \tau$



## Unification Problem

**Definition (Unification Problem)**

instance: terms $s$, $t$
question: $s\sigma = t\sigma$ for some substitution $\sigma$ ($\sigma$ is **unifier** of $s$ and $t$)

**Definition**

$\leq$-minimal unifier is called most general unifier (mgu)

**Example**

for $x + (0 + \mathsf{s}(y))$ and $\mathsf{s}(z) + (0 + x)$

- $\{x \mapsto \mathsf{s}(z)\}$ is not unifier
- $\{x \mapsto \mathsf{s}(z),\ y \mapsto z\}$ is mgu
- $\{x \mapsto \mathsf{s}(0),\ y \mapsto 0,\ z \mapsto 0\}$ is unifier but not mgu



## Unification Algorithm

let $s \approx t$ denote unordered pair of $s$ and $t$

**Definition (unification rules, $E \Rightarrow_\sigma E'$)**

- $\dfrac{\{f(s_1,\ldots,s_n) \approx f(t_1,\ldots,t_n)\} \uplus E}{\{s_1 \approx t_1,\ \ldots,\ s_n \approx t_n\} \cup E}$
- $\dfrac{\{x \approx x\} \uplus E}{E}$ if $x \in \mathcal{V}$
- $\dfrac{\{x \approx t\} \uplus E}{E\{x \mapsto t\}}\ \{x \mapsto t\}$ if $x \notin \mathcal{V}\mathrm{ar}(t)$ (occurs check)

**Theorem**

- *$s$ and $t$ are unifiable if and only if* $\{s \approx t\} \Rightarrow^* \varnothing$
- $\sigma_1\sigma_2\cdots\sigma_n$ *is mgu of $s$ and $t$ if* $\{s \approx t\} \Rightarrow_{\sigma_1} \cdots \Rightarrow_{\sigma_n} \varnothing$

$$\{x + (0 + \mathsf{s}(y)) \approx \mathsf{s}(z) + (0 + x)\}$$
$$\Downarrow$$
$$\{x \approx \mathsf{s}(z),\ 0 + \mathsf{s}(y) \approx 0 + x\}$$
$$\Downarrow \ \{x \mapsto \mathsf{s}(z)\}$$
$$\{0 + \mathsf{s}(y) \approx 0 + \mathsf{s}(z)\}$$
$$\Downarrow$$
$$\{0 \approx 0,\ \mathsf{s}(y) \approx \mathsf{s}(z)\}$$
$$\Downarrow$$
$$\{\mathsf{s}(y) \approx \mathsf{s}(z)\}$$
$$\Downarrow$$
$$\{y \approx z\}$$
$$\Downarrow \ \{y \mapsto z\}$$
$$\varnothing$$

mgu

$$\begin{aligned} & \{x \mapsto \mathsf{s}(z)\}\{y \mapsto z\} \\ = & \{x \mapsto \mathsf{s}(z), y \mapsto z\} \end{aligned}$$



## Exercise 1

Compute an mgu of $E$ if it exists.

1. $E = \{x + \mathsf{s}(y) \approx \mathsf{s}(y) + \mathsf{s}(z)\}$
2. $E = \{x + \mathsf{s}(y) \approx \mathsf{s}(y) + \mathsf{s}(x)\}$
3. $E = \left\{ \begin{array}{lcl} a & \approx & \mathsf{TArr}(\mathsf{TApp}(\mathsf{List}, b), c) \\ a & \approx & \mathsf{TArr}(c, c) \end{array} \right\}$

Here $x$, $y$, $z$, $a$, $b$, and $c$ are variables.



# Type System



## Type Inference (Type Reconstruction Problem)

**Question**

what is type of f in next Haskell program?

```
data List a = Nil | Cons a (List a)
f (Cons x y) z = Cons x (f y z)
f Nil        z = z
```

**Note**

corresponding system is applicative TRS $\mathcal{R}$ over type environment $\Gamma$

$$\mathcal{R} = \left\{ \begin{array}{c} \mathsf{f}\ (\mathsf{c}\ x\ y)\ z \to \mathsf{c}\ x\ (\mathsf{f}\ y\ z) \\ \mathsf{f}\ \mathsf{nil}\ z \to z \end{array} \right\} \qquad \Gamma = \left\{ \begin{array}{l} \mathsf{c} : a \to \mathsf{List}\ a \to \mathsf{List}\ a \\ \mathsf{nil} : \mathsf{List}\ a\} \end{array} \right\}$$

# Typing

**Definition**

- **polymorphic type** $\tau$ is term of form:

$$\tau ::= \underbrace{a}_{\text{type variable}} \mid \underbrace{c}_{\text{type constructor}} \mid \underbrace{\tau\ \tau}_{\text{type application}} \mid \underbrace{\tau \to \tau}_{\text{function type}}$$

- **type environment** is partial function from symbols to types

**Example**

$$\left\{\begin{array}{l} x : \mathsf{Nat} \\ 0 : \mathsf{Nat} \\ \mathsf{f} : a \to (a \to \mathsf{Bool}) \end{array}\right\} \text{ is type environment}$$



**Definition (type judgement)**

given type environment $\Gamma$

$$\frac{\Gamma(x) = \tau}{x : \tau\sigma} \qquad \frac{t : \tau_1 \to \tau_2 \quad u : \tau_1}{t\ u : \tau_2}$$

where $\sigma$ is type version of substitution

**Example**

$$\Gamma = \left\{\begin{array}{l} x : \mathsf{Nat} \\ 0 : \mathsf{Nat} \\ \mathsf{f} : a \to a \to \mathsf{Bool} \end{array}\right\} \quad \dfrac{\dfrac{\dfrac{\Gamma(\mathsf{f}) = a \to a \to \mathsf{Bool}}{\Gamma \vdash \mathsf{f} : \mathsf{Nat} \to \mathsf{Nat} \to \mathsf{Bool}} \quad \dfrac{\Gamma(x) = \mathsf{Nat}}{\Gamma \vdash x : \mathsf{Nat}}}{\Gamma \vdash \mathsf{f}\ x : \mathsf{Nat} \to \mathsf{Bool}} \quad \dfrac{\Gamma(0) = \mathsf{Nat}}{\Gamma \vdash 0 : \mathsf{Nat}}}{\Gamma \vdash \mathsf{f}\ x\ 0 : \mathsf{Bool}}$$



**Definition**

- term $t$ is typable under $\Gamma$ if $\Gamma \vdash t : \tau$ for some $\tau$
- rule $\ell \to r$ is typable under $\Gamma$ if $\Gamma \vdash \ell : \tau$ and $\Gamma \vdash r : \tau$ for some type $\tau$
- applicative TRS is typable under $\Gamma$ if all rules are typable under $\Gamma$

**Example**

- g g is typeable under $\{\mathsf{g} : a \to a\}$ but not under $\{\mathsf{g} : \mathsf{List}\ a \to \mathsf{List}\ a\}$
- $\{\mathsf{map}\ f\ (x : xs) \to f\ x : \mathsf{map}\ f\ xs\}$ is typable under

$$\{\mathsf{map} : (a \to b) \to \mathsf{List}\ a \to \mathsf{List}\ b, \ldots\}$$

**Fact (type preservation)**

for every applicative TRS $\mathcal{R}$ typable under $\Gamma$

$$\Gamma \vdash s : \tau \ \ \& \ \ s \to_{\mathcal{R}} t \implies \Gamma \vdash t : \tau$$



# Exercise 2

1. Identify the type environment $\Gamma$ and the applicative TRS $\mathcal{R}$ for the program:

$$\begin{array}{l} \mathbf{data}\ \mathsf{Nat} = \mathsf{Zero} \mid \mathsf{Succ}\ \mathsf{Nat} \\ \mathbf{data}\ \mathsf{List}\ a = \mathsf{Nil} \mid \mathsf{Cons}\ a\ (\mathsf{List}\ a) \\ \\ \mathsf{Len} :: \mathsf{List}\ a \to \mathsf{Nat} \\ \mathsf{Len}\ (\mathsf{Cons}\ x\ xs) = \mathsf{Succ}\ (\mathsf{Len}\ xs) \end{array}$$

2. Prove $\Gamma \vdash \mathsf{Cons}\ \mathsf{Zero}\ \mathsf{Nil} : \mathsf{List}\ \mathsf{Nat}$.
3. Prove $\Gamma \vdash \mathsf{Len}\ (\mathsf{Cons}\ x\ xs) : \mathsf{Nat}$.
4. Prove $\Gamma \vdash \mathsf{Succ}\ (\mathsf{Len}\ xs) : \mathsf{Nat}$.

### Definition (constraint typing)

let $\Gamma$ be type environment and $a, d_x, d_f$ type variables

$$\frac{}{\Gamma \vdash_{\mathcal{C}} x : a}[a \approx d_x] \qquad \frac{\Gamma(f) = \tau}{\Gamma \vdash_{\mathcal{C}} f : a}[a \approx \tau'] \qquad \frac{\Gamma(f) = \bot}{\Gamma \vdash_{\mathcal{C}} f : a}[a \approx d_f]$$

$$\frac{\Gamma \vdash_{\mathcal{C}} t : b \qquad \Gamma \vdash_{\mathcal{C}} u : c}{\Gamma \vdash_{\mathcal{C}} t\ u : a}[b \approx c \to a]$$

where $b, c$ are fresh variables, and $\tau'$ is renamed version of $\tau$ with fresh variables

### Notation

- $\mathcal{C}_\Gamma(t, a)$ is set of constraints in derivation of $\Gamma \vdash_{\mathcal{C}} t : a$
- $\mathcal{C}_\Gamma(\mathcal{R}) = \{e \mid e \in \mathcal{C}_\Gamma(\ell \to r) \text{ for some } \ell \to r \in \mathcal{R}\}$



## Example of (Monomorphic) Type Inference

let $\Gamma = \{\mathsf{c} : a \to \mathsf{L}\ a \to \mathsf{L}\ a,\ \mathsf{nil} : \mathsf{L}\ a\}$ (below, $\Gamma \vdash_{\mathcal{C}}$ is omitted)

$$\dfrac{\dfrac{\dfrac{\Gamma(\mathsf{f}) = \bot}{\mathsf{f} : a_2}[3] \quad \dfrac{\dfrac{\dfrac{\dfrac{\Gamma(\mathsf{c}) = a_6 \to \mathsf{L}\ a_6 \to \mathsf{L}\ a_6}{\mathsf{c} : a_5}[6] \quad \dfrac{\Gamma(x) = \bot}{x : a_7}[7]}{\mathsf{c}\ x : a_4}[5] \quad \dfrac{\Gamma(y) = \bot}{y : a_8}[8]}{\mathsf{c}\ x\ y : a_3}[4]}{}}{\mathsf{f}\ (\mathsf{c}\ x\ y) : a_1}[2] \quad \dfrac{\Gamma(z) = \bot}{z : a_9}[9]}{\mathsf{f}\ (\mathsf{c}\ x\ y)\ z : a_0}[1]$$

$\mathcal{C}_\Gamma(\mathsf{f}\ (\mathsf{c}\ x\ y)\ z, a_0)$ consists of following equations:

$$\begin{array}{lll}
1:\ a_1 \approx a_9 \to a_0 & 4:\ a_4 \approx a_8 \to a_3 & 7:\ a_7 \approx d_x \\
2:\ a_2 \approx a_3 \to a_1 & 5:\ a_5 \approx a_7 \to a_4 & 8:\ a_8 \approx d_y \\
3:\ a_2 \approx d_{\mathsf{f}} & 6:\ a_5 \approx a_6 \to \mathsf{L}\ a_6 \to \mathsf{L}\ a_6 & 9:\ a_9 \approx d_z
\end{array}$$



variable-renamed applicative TRS $\mathcal{R}$ over $\Gamma = \{\mathsf{c} : a \to \mathsf{L}\ a \to \mathsf{L}\ a,\ \mathsf{nil} : \mathsf{L}\ a\}$:

$$\begin{array}{rcl}
\ell_1 = & \mathsf{f}\ (\mathsf{c}\ x\ y)\ z \to \mathsf{c}\ x\ (\mathsf{f}\ y\ z) & = r_1 \\
\ell_2 = & \mathsf{f}\ \mathsf{nil}\ w \to w & = r_2
\end{array}$$

$$\mathcal{C}_\Gamma(\mathcal{R}) = \mathcal{C}_\Gamma(\ell_1, a_0) \cup \mathcal{C}_\Gamma(r_1, a_{10}) \cup \{a_0 \approx a_{10}\} \ \cup\ \mathcal{C}_\Gamma(\ell_2, a_{20}) \cup \mathcal{C}_\Gamma(r_2, a_{25}) \cup \{a_{20} \approx a_{25}\}$$

$$\left\{\begin{array}{lll}
1:\ a_1 \approx a_9 \to a_0 & 4:\ a_4 \approx a_8 \to a_3 & 7:\ a_7 \approx d_x \\
2:\ a_2 \approx a_3 \to a_1 & 5:\ a_5 \approx a_7 \to a_4 & 8:\ a_8 \approx d_y \\
3:\ a_2 \approx d_{\mathsf{f}} & 6:\ a_5 \approx a_6 \to \mathsf{L}\ a_6 \to \mathsf{L}\ a_6 & 9:\ a_9 \approx d_z \\
10:\ a_{11} \approx a_{15} \to a_{10} & 13:\ a_{14} \approx d_x & 16:\ a_{17} \approx d_{\mathsf{f}} \\
11:\ a_{12} \approx a_{14} \to a_{11} & 14:\ a_{16} \approx a_{19} \to a_{15} & 17:\ a_{18} \approx d_y \\
15:\ a_{12} \approx a_{13} \to \mathsf{L}\ a_{13} \to \mathsf{L}\ a_{13} & 12:\ a_{17} \approx a_{18} \to a_{16} & 18:\ a_{19} \approx d_z \\
19:\ a_0 \approx a_{10} & & \\
20:\ a_{21} \approx a_{24} \to a_{20} & 22:\ a_{22} \approx d_{\mathsf{f}} & 24:\ a_{24} \approx d_w \\
21:\ a_{22} \approx a_{23} \to a_{21} & 23:\ a_{23} \approx d_{\mathsf{nil}} & 25:\ a_{25} \approx d_w \\
26:\ a_{20} \approx a_{25} & &
\end{array}\right\}$$

since $\mu(d_{\mathsf{f}}) = \mathsf{L}\ d_x \to \mathsf{L}\ d_x \to \mathsf{L}\ d_x$ for mgu $\mu$ of $\mathcal{C}_\Gamma(\mathcal{R})$, type of $\mathsf{f}$ is $\mathsf{L}\ a \to \mathsf{L}\ a \to \mathsf{L}\ a$