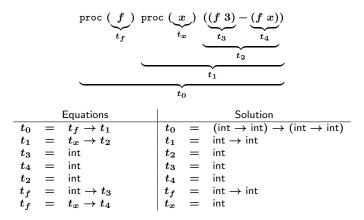
COSE212: Programming Languages Lecture 15 — Automatic Type Inference (3)

Hakjoo Oh 2019 Fall

Finding a Solution of Type Equations

Find the values of type variables that make all the equations true.



Static type systems find such a solution using unification algorithm.

The calculation is split into equations to be solved and substitution found so far. Initially, the substitution is empty:

Equations	Substitution
$t_0 ~=~ t_f ightarrow t_1$	
$t_1 \;=\; t_x o t_2$	
t_3 = int	
t_4 = int	
t_2 = int	
$t_f \;=\;$ int $ ightarrow t_3$	
$t_f = t_x o t_4$	

Consider each equation in turn. If the equation's left-hand side is a variable, move it to the substitution:

Equations	Substitution			
$t_1 = t_x ightarrow t_2$	$t_0 = t_f ightarrow t_1$			
t_3 = int				
t_4 = int				
t_2 = int				
$t_f \;=\; { m int} o t_3$				
$t_f ~=~ t_x ightarrow t_4$				

Move the next equation to the substitution and propagate the information to the existing substitution (i.e., substitute the right-hand side for each occurrence of t_1):

		Equations			Substitution
t_3	=		t_0	=	$t_f ightarrow (t_x ightarrow t_2)$
t_4	=	int	t_1	=	$t_x ightarrow t_2$
t_2	=	int			
t_{f}	=	$int \to t_3$			
t_{f}	=	$t_x ightarrow t_4$			

Same for the next three equations:

Equations	Substitution			
t_4 = int	$t_0 = t_f \rightarrow (t_x \rightarrow t_2)$			
t_2 = int	$t_1 = t_x ightarrow t_2$			
$t_f = ext{int} o t_3$	$t_3 = int$			
$t_f = t_x o t_4$				
Equations	Substitution			
$t_2 = int$	$t_0 = t_f ightarrow (t_x ightarrow t_2)$			
$t_f = ext{int} o t_3$	$t_1 = t_x ightarrow t_2$			
$t_f = t_x ightarrow t_4$	$t_3 = int$			
-	t_4 = int			
Equations	Substitution			
$t_f = \operatorname{int} ightarrow t_3$	$t_0 = t_f ightarrow (t_x ightarrow ext{int})$			
$t_f = t_x ightarrow t_4$	$t_1 = t_x ightarrow ext{int}$			
	$t_3 = int$			
	$egin{array}{rcl} t_4&=& ext{int}\ t_2&=& ext{int} \end{array}$			
	$t_2 = int$			

Consider the next equation $t_f = \text{int} \rightarrow t_3$. The equation contains t_3 , which is already bound to int in the substitution. Substitute int for t_3 in the equation. This is called *applying* the substitution to the equation.

Equations	Substitution		
$t_f = \text{int} ightarrow ext{int}$	$t_0 =$	$egin{array}{ll} t_f ightarrow (t_x ightarrow { m int}) \ t_x ightarrow { m int} \ { m int} \ { m int} \end{array}$	
$t_f \;\;=\;\; t_x o t_4$	$t_1 =$	$t_x ightarrow { m int}$	
	$t_3 =$	int	
	$t_4 =$	int	
	$t_2 =$	int	

Move the resulting equation to the substitution and update it.

Equations	Substitution			
$t_f = t_x ightarrow t_4$	t_0	=	$(int \to int) \to (t_x \to int)$	
	t_1	=	$t_x ightarrow$ int	
	t_3	=	int	
	t_4	=	int	
	t_2	=	int	
	$\mid t_{f}$	=	$(\operatorname{int} \to \operatorname{int}) \to (t_x \to \operatorname{int})$ $t_x \to \operatorname{int}$ int int $\operatorname{int} \to \operatorname{int}$	

Apply the substitution to the equation:

Equations	Substitution
int $ ightarrow$ int $=$ t_x $ ightarrow$ int	$t_0 = (\operatorname{int} ightarrow \operatorname{int}) ightarrow (t_x ightarrow \operatorname{int})$
	$t_1 = t_x ightarrow ext{int}$
	$t_3 = \text{int}$
	t_4 = int
	t_2 = int
	$\begin{array}{rcl} t_0 &=& (\operatorname{int} \to \operatorname{int}) \to (t_x \to \operatorname{int}) \\ t_1 &=& t_x \to \operatorname{int} \\ t_3 &=& \operatorname{int} \\ t_4 &=& \operatorname{int} \\ t_2 &=& \operatorname{int} \\ t_f &=& \operatorname{int} \to \operatorname{int} \end{array}$

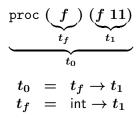
If neither side of the equation is a variable, simplify the equation by yielding two new equations:

Equations	Substitution			
int = t_x	t_0	=	$(\text{int} \rightarrow \text{int}) \rightarrow (t_x \rightarrow \text{int})$	
int = int	t_1	=	$t_x ightarrow$ int	
	t_3	=	int	
	t_4	=	int	
	t_2	=	int	
	t_{f}	=	$\begin{array}{l} (\operatorname{int} \to \operatorname{int}) \to (t_x \to \operatorname{int}) \\ t_x \to \operatorname{int} \\ \operatorname{int} \\ \operatorname{int} \\ \operatorname{int} \\ \operatorname{int} \to \operatorname{int} \end{array}$	

Switch the sides of the first equation and move it to the substitution:

Equations	Substitution			
int = int	t_0	=	$(int \rightarrow int) \rightarrow (int \rightarrow int)$ $int \rightarrow int$ int int int int $int \rightarrow int$ int int	
	t_1	=	$int \to int$	
	t_3	=	int	
	t_4	=	int	
	t_2	=	int	
	t_{f}	=	int \rightarrow int	
	t_{x}	=	int	

The final substitution is the solution of the original equations.



1 Substitution Equations $t_0 = t_f \rightarrow t_1$ $t_f = \text{int} \rightarrow t_1$ 2 Equations Substitution $t_f = \text{int} \rightarrow t_1$ $t_0 = t_f \rightarrow t_1$ 3 Equations Substitution $egin{array}{rcl} t_0 &=& (\operatorname{int}
ightarrow t_1)
ightarrow t_1 \ t_f &=& \operatorname{int}
ightarrow t_1 \end{array}$

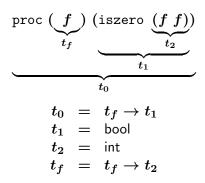
The type is *polymorphic* in t_1 .

$$\underbrace{ \begin{array}{c} \text{if } \underbrace{x}_{t_x} \text{ then } \underbrace{(x-1)}_{t_1} \text{ else } 0 \\ \underbrace{t_x}_{t_0} \\ t_x = \text{ bool} \\ t_1 = t_0 \\ \text{int } = t_0 \\ t_x = \text{ int} \\ t_1 = \text{ int} \\ \end{array} }$$

The equations have no solutions because, during the unification algorithm, we encounter the following contradictory state:

	Eqι	ations	Su	bstit	ution
bool	=	int	t_x		bool
t_1	=	int	t_1	=	int
			t_0	=	int

Because bool and int cannot be equal, there is no solution to the equations.



Solving as usual, we encounter a problem:

Equations	Substitution		
$t_f = t_f ightarrow ext{int}$			$t_f ightarrow$ bool
			bool
	t_2	=	int

- There is no type t_f that satisfies the equation, because the right-hand side of the equation is always larger than the left.
- If we ever deduce an equation of the form $t = \dots t \dots$ where the type variable t occurs in the right-hand side, we must conclude that there is no solution. This is called *occurrence check*.

Unification Algorithm

For each equation in turn,

- Apply the current substitution to the equation.
- If the equation is always true (e.g. int = int), discard it.
- If the left- and right-hand sides are contradictory (e.g. bool = int), the algorithm fails.
- If neither side is a variable (e.g. int $\rightarrow t_1 = t_2 \rightarrow$ bool), simplify the equation, which eventually generates an equation whose left- or right-hand side is a variable.
- If the left-hand side is not a variable, switch the sides.
- If the left-hand side variable occurs in the right-hand side, the algorithm fails.
- Otherwise, move it to the substitution and substitute the right-hand side for each occurrence of the variable in the substitution.

let
$$x = 4$$
 in $(x \ 3)$

let
$$f = \operatorname{proc}(z) \ z$$
 in $\operatorname{proc}(x) \ ((f \ x) - 1)$

let p = iszero 1 in if p then 88 else 99

let f = proc(x) x in if (f (iszero0)) then (f 11) else (f 22)

Substitution

Solutions of type equations are represented by substitution:

$$S \in Subst = TyVar
ightarrow T$$

Applying a substitution to a type:

$$egin{array}{rcl} S({
m int})&=&{
m int}\ S({
m bool})&=&{
m bool}\ S({
m a})&=&\left\{egin{array}{ll}t&{
m if}\ lpha\mapsto t\in S\ lpha&{
m otherwise}\ S(T_1 o T_2)&=&S(T_1) o S(T_2) \end{array}
ight.$$

Applying the substitution

$$S = \{t_1 \mapsto \mathsf{int}, t_2 \mapsto \mathsf{int} o \mathsf{int}\}$$

to to the type $(t_1 \rightarrow t_2) \rightarrow (t_3 \rightarrow \text{int})$:

$$\begin{split} S((t_1 \to t_2) \to (t_3 \to \text{int})) \\ &= S(t_1 \to t_2) \to S(t_3 \to \text{int}) \\ &= (S(t_1) \to S(t_2)) \to (S(t_3) \to S(\text{int})) \\ &= (\text{int} \to (\text{int} \to \text{int})) \to (t_3 \to \text{int}) \end{split}$$

Unification

Update the current substitution with equality $t_1 \doteq t_2$.

 $\mathsf{unify}: T \times T \times Subst \to Subst$

$$\begin{array}{rcl} & \mathsf{unify}(\mathsf{int},\mathsf{int},S) &=& S\\ & \mathsf{unify}(\mathsf{bool},\mathsf{bool},S) &=& S\\ & \mathsf{unify}(\alpha,\alpha,S) &=& S\\ & \mathsf{unify}(\alpha,\alpha,S) &=& S\\ & \mathsf{unify}(\alpha,t,S) &=& \left\{ \begin{array}{cc} \mathsf{fail} & \alpha \text{ occurs in } t\\ \mathsf{extend} \ S \text{ with } \alpha \doteq t & \mathsf{otherwise} \end{array} \right.\\ & \mathsf{unify}(t,\alpha,S) &=& \mathsf{unify}(\alpha,t,S)\\ \mathsf{unify}(t_1 \rightarrow t_2,t_1' \rightarrow t_2',S) &=& \mathsf{let} \ S' = \mathsf{unify}(t_1,t_1',S) \text{ in}\\ & \mathsf{let} \ S'' = \mathsf{unify}(S'(t_2),S'(t_2'),S') \text{ in}\\ & S''\\ & \mathsf{unify}(_,_,_) &=& \mathsf{fail} \end{array}$$

- $\operatorname{unify}(\alpha, \operatorname{int} \to \operatorname{int}, \emptyset) =$
- $\operatorname{unify}(\alpha, \operatorname{int} \rightarrow \alpha, \emptyset) =$
- $\operatorname{unify}(\alpha \to \beta, \operatorname{int} \to \operatorname{int}, \emptyset) =$
- $\operatorname{unify}(\alpha \to \beta, \operatorname{int} \to \alpha, \emptyset) =$

Solving Equations

Let \mathcal{U} be the final unification algorithm:

 $\mathcal{U}(u) = \mathsf{unifyall}(u, \emptyset)$

$\mathsf{typeof}: E \to T$

The final type inference algorithm that composes equation derivation (\mathcal{V}) and equation solving (\mathcal{U}) :

$$egin{aligned} \mathsf{typeof}(E) = \ \mathsf{let}\ S = \mathcal{U}(\mathcal{V}(\emptyset, E, lpha)) \quad (\mathsf{new}\ lpha) \ \mathsf{in}\ S(lpha) \end{aligned}$$

- typeof((proc (x) x) 1)
- typeof(let x = 1 in proc(y) (x + y))

Summary

Automatic type inference:

- derive type equations from the program text, and
- solve the equations by unification algorithm.