

# Programming Languages and Compilers (CS 421)

---

Elsa L Gunter

2112 SC, UIUC

[http://www.cs.uiuc.edu/class  
/fa06/cs421/](http://www.cs.uiuc.edu/class/fa06/cs421/)

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

# Background for Unification

---

- Terms made from constructors and variables (for the simple first order case)
- Constructors may be applied to arguments (other terms) to make new terms
- Variables and constructors with no arguments are base cases
- Constructors applied to different number of arguments (arity) considered different
- Substitution of terms for variables

# Simple Implementation Background

---

```
type term = Variable of string  
          | Const of (string * term list)
```

```
let rec subst vn residue term =  
  match term with Variable n ->  
    if vn = n then residue else term  
  | Const (c, tys) ->  
    Const (c, List.map (subst vn residue)  
             tys);;
```

# Unification Problem

---

Given a set of pairs of terms (“equations”)

$$\{(s_1, t_1), (s_2, t_2), \dots, (s_n, t_n)\}$$

(the *unification problem*) does there exist

a substitution  $\sigma$  (the *unification solution*)

of terms for variables such that

$$\sigma(s_i) = \sigma(t_i),$$

for all  $i = 1, \dots, n$ ?

# Uses for Unification

---

- Type Inference and type checking
- Pattern matching as in OCAML
  - Can used a simplified version of algorithm
- Logic Programming - Prolog
- Simple parsing

# Unification Algorithm

---

- Let  $S = \{(s_1, t_1), (s_2, t_2), \dots, (s_n, t_n)\}$  be a unification problem.
- Case  $S = \{ \}$ :  $\text{Unif}(S) = \text{Identity function}$  (ie no substitution)
- Case  $S = \{(s, t)\} \cup S'$ : Four main steps

# Unification Algorithm

---

- Delete: if  $s = t$  (they are the same term) then  $\text{Unif}(S) = \text{Unif}(S')$
- Decompose: if  $s = f(q_1, \dots, q_m)$  and  $t = f(r_1, \dots, r_m)$  (same  $f$ , same  $m!$ ), then  $\text{Unif}(S) = \text{Unif}(\{(q_1, r_1), \dots, (q_m, r_m)\} \cup S')$
- Orient: if  $t = x$  is a variable, and  $s$  is not a variable,  $\text{Unif}(S) = \text{Unif}(\{(x, s)\} \cup S')$

# Unification Algorithm

---

- Eliminate: if  $s = x$  is a variable, and  $x$  does not occur in  $t$  (the occurs check), then
  - Let  $\varphi = x \mapsto t$
  - Let  $\psi = \text{Unif}(\varphi(S'))$
  - $\text{Unif}(S) = \{x \mapsto \psi(t)\} \circ \psi$

# Tricks for Efficient Unification

---

- Don't return substitution, rather do it incrementally
- Make substitution be constant time
  - Requires implementation of terms to use mutable structures (or possibly lazy structures)
  - We haven't discussed these yet

# Example

---

- $x, y, z$  variables,  $f, g$  constructors
- $S = \{(f(x), f(g(y, z))), (g(y, f(y)), x)\}$

# Example

---

- $x, y, z$  variables,  $f, g$  constructors
- Pick a pair:  $(g(y, f(y)), x)$
- $S = \{(f(x), f(g(y, z))), (g(y, f(y)), x)\}$

# Example

---

- $x, y, z$  variables,  $f, g$  constructors
- Pick a pair:  $(g(y, f(y)), x)$
- Orient is first rule that applies
- $S = \{(f(x), f(g(y, z))), (g(y, f(y)), x)\}$

# Example

---

- $x, y, z$  variables,  $f, g$  constructors
- $S \rightarrow \{(f(x), f(g(y, z))), (x, g(y, f(y)))\}$

# Example

---

- $x, y, z$  variables,  $f, g$  constructors
- Pick a pair:  $(f(x), f(g(y, z)))$
- $S \rightarrow \{(f(x), f(g(y, z))), (x, g(y, f(y)))\}$

# Example

---

- $x, y, z$  variables,  $f, g$  constructors
- Pick a pair:  $(f(x), f(g(y, z)))$
- Decompose it  $(x, g(y, z))$
- $S \rightarrow \{(x, g(y, z)), (x, g(y, f(y)))\}$

# Example

---

- $x, y, z$  variables,  $f, g$  constructors
- Pick a pair:  $(x, g(y, f(y)))$
- $S \rightarrow \{(x, g(y, z)), (x, g(y, f(y)))\}$

# Example

---

- $x, y, z$  variables,  $f, g$  constructors
- Pick a pair:  $(x, g(y, f(y)))$
- Substitute:
- $S \rightarrow \{(g(y, f(y)), g(y, z))\}$

With  $\{x \mapsto g(y, f(y))\}$

# Example

---

- $x, y, z$  variables,  $f, g$  constructors
- Pick a pair:  $(g(y, f(y)), g(y, z))$
- $S \rightarrow \{(g(y, f(y)), g(y, z))\}$   
With  $\{x \mapsto g(y, f(y))\}$

# Example

---

- $x, y, z$  variables,  $f, g$  constructors
- Pick a pair:  $(g(y, f(y)), g(y, z))$
- Decompose:  $(y, y)$  and  $(f(y), z)$
- $S \rightarrow \{(y, y), (f(y), z)\}$

With  $\{x \mapsto g(y, f(y))\}$

# Example

---

- $x, y, z$  variables,  $f, g$  constructors
- Pick a pair:  $(y, y)$
- $S \rightarrow \{(y, y), (f(y), z)\}$   
With  $\{x \mid \rightarrow g(y, f(y))\}$

# Example

---

- $x, y, z$  variables,  $f, g$  constructors
- Pick a pair:  $(y, y)$
- Eliminate
- $S \rightarrow \{(f(y), z)\}$

With  $\{x \mapsto g(y, f(y))\}$

# Example

---

- $x, y, z$  variables,  $f, g$  constructors
- Pick a pair:  $(f(y), z)$

- $S \rightarrow \{(f(y), z)\}$

With  $\{x \mapsto g(y, f(y))\}$

# Example

---

- $x, y, z$  variables,  $f, g$  constructors
- Pick a pair:  $(f(y), z)$
- Orient
- $S \rightarrow \{(z, f(y))\}$

With  $\{x \mid \rightarrow g(y, f(y))\}$

# Example

---

- $x, y, z$  variables,  $f, g$  constructors
- Pick a pair:  $(z, f(y))$

- $S \rightarrow \{(z, f(y))\}$

With  $\{x \mapsto g(y, f(y))\}$

# Example

---

- $x, y, z$  variables,  $f, g$  constructors
- Pick a pair:  $(z, f(y))$
- Substitute
- $S \rightarrow \{ \}$

With  $\{x \mapsto \{z \mapsto f(y)\} (g(y, f(y))) \} \circ \{z \mapsto f(y)\}$

# Example

---

- $x, y, z$  variables,  $f, g$  constructors
- Pick a pair:  $(z, f(y))$
- Substitute
- $S \rightarrow \{ \}$

With  $\{x \mapsto g(y, f(y))\} \circ \{(z \mapsto f(y))\}$

# Example

---

$$S = \{(f(x), f(g(y,z))), (g(y,f(y)),x)\}$$

Solved by  $\{x \mapsto g(y,f(y))\} \circ \{(z \mapsto f(y))\}$

$$\underbrace{f(g(y,f(y)))}_x = f(g(y,\underbrace{f(y)}_z))$$

and

$$g(y,f(y)) = \underbrace{g(y,f(y))}_x$$

# Example of Failure

---

- $S = \{(f(x,g(y)), f(h(y),x))\}$
- Decompose
- $S \rightarrow \{(x,h(y)), (g(y),x)\}$
- Orient
- $S \rightarrow \{(x,h(y)), (x,g(y))\}$
- Substitute
- $S \rightarrow \{(h(y), g(y))\}$  with  $\{x \mapsto h(y)\}$
- No rule to apply! Decompose fails!