

Unification

1

Unification

Unification is the process of determining whether two expressions can be *unified*, i.e. made identical by appropriate substitutions for their variables.

2

Substitutions

A *substitution* is a finite set of pairs of variables and terms, called *replacements*.

$$\{x \leftarrow a, y \leftarrow f(b), v \leftarrow w\}$$

The result of applying a substitution σ to an expression φ is the expression $\varphi\sigma$ obtained from φ by replacing every occurrence of every variable in the substitution by its replacement.

$$p(x, x, y, z)\{x \leftarrow a, y \leftarrow f(b), v \leftarrow w\} = p(a, a, f(b), z)$$

Cascaded Substitutions

$$r\{x, y, z\}\{x \leftarrow a, y \leftarrow f(u), z \leftarrow v\} = r\{a, f(u), v\}$$

$$r\{a, f(u), v\}\{u \leftarrow d, v \leftarrow e, z \leftarrow g\} = r(a, f(d), e)$$

$$r\{x, y, z\}\{x \leftarrow a, y \leftarrow f(d), z \leftarrow e, u \leftarrow d, v \leftarrow e\} = r(a, f(d), e)$$

Composition of Substitutions

The *composition* of substitution σ and τ is the substitution (written $compose(\sigma, \tau)$ or, more simply, $\sigma\tau$) obtained by

- (1) applying τ to the replacements in σ
- (2) adding to σ pairs from τ with different variables
- (3) deleting any assignments of a variable to itself.

$$\begin{aligned} & \{x \leftarrow a, y \leftarrow f(u), z \leftarrow v\} \{u \leftarrow d, v \leftarrow e, z \leftarrow g\} \\ &= \{x \leftarrow a, y \leftarrow f(d), z \leftarrow e\} \{u \leftarrow d, v \leftarrow e, z \leftarrow g\} \\ &= \{x \leftarrow a, y \leftarrow f(d), z \leftarrow e, u \leftarrow d, v \leftarrow e\} \end{aligned}$$

5

Unification

A substitution σ is a *unifier* for an expression φ and an expression ψ if and only if $\varphi\sigma = \psi\sigma$.

$$\begin{aligned} p(x, y) \{x \leftarrow a, y \leftarrow b, v \leftarrow b\} &= p(a, b) \\ p(a, v) \{x \leftarrow a, y \leftarrow b, v \leftarrow b\} &= p(a, b) \end{aligned}$$

If two expressions have a unifier, they are said to be *unifiable*. Otherwise, they are *nonunifiable*.

$$\begin{aligned} & p(x, x) \\ & p(a, b) \end{aligned}$$

6

Non-Uniqueness of Unification

Unifier 1:

$$p(x,y)\{x \leftarrow a, y \leftarrow b, v \leftarrow b\} = p(a,b)$$

$$p(a,v)\{x \leftarrow a, y \leftarrow b, v \leftarrow b\} = p(a,b)$$

Unifier 2:

$$p(x,y)\{x \leftarrow a, y \leftarrow f(w), v \leftarrow f(w)\} = p(a,f(w))$$

$$p(a,v)\{x \leftarrow a, y \leftarrow f(w), v \leftarrow f(w)\} = p(a,f(w))$$

Unifier 3:

$$p(x,y)\{x \leftarrow a, y \leftarrow v\} = p(a,v)$$

$$p(a,v)\{x \leftarrow a, y \leftarrow v\} = p(a,v)$$

7

Most General Unifier

A substitution σ is a *most general unifier (mgu)* of two expressions if and only if it is as general as or more general than any other unifier.

Theorem: If two expressions are unifiable, then they have an mgu that is unique up to variable permutation.

$$p(x,y)\{x \leftarrow a, y \leftarrow v\} = p(a,v)$$

$$p(a,v)\{x \leftarrow a, y \leftarrow v\} = p(a,v)$$

$$p(x,y)\{x \leftarrow a, v \leftarrow y\} = p(a,y)$$

$$p(a,v)\{x \leftarrow a, v \leftarrow y\} = p(a,y)$$

8

Unification Procedure

One good thing about our language is that there is a simple and inexpensive procedure for computing a most general unifier of any two expressions if it exists.

9

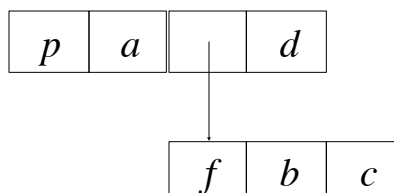
Expression Structure

Each expression is treated as a sequence of its immediate subexpressions.

Linear Version:

$p(a, f(b, c), d)$

Structured Version:



10

Unification Procedure

- (1) If two expressions being compared are identical, succeed.
- (2) If neither is a variable and at least one is a constant, fail.
- (3) If one of the expressions is a variable, proceed as described shortly.
- (4) If both expressions are sequences, iterate across the expressions, comparing as described above.

11

Dealing with Variables

If one of the expressions is a variable, check whether the variable has a binding in the current substitution.

- (a) If so, try to unify the binding with the other expression.
- (b) If no binding, check whether the other expression contains the variable. If the variable occurs within the expression, fail; otherwise, set the substitution to the composition of the old substitution and a new substitution in which variable is bound to the other expression.

12

Example

Call: $p(x,b), p(a,y), \{\}$

Call: $p, p, \{\}$

Exit: $\{\}$

Call: $x, a, \{\}$

Exit: $\{\}\{x \leftarrow a\} = \{x \leftarrow a\}$

Call: $b, y, \{x \leftarrow a\}$

Exit: $\{x \leftarrow a\}\{y \leftarrow b\} = \{x \leftarrow a, y \leftarrow b\}$

Exit: $\{x \leftarrow a, y \leftarrow b\}$

13

Example

Call: $p(x,x), p(a,y), \{\}$

Call: $p, p, \{\}$

Exit: $\{\}$

Call: $x, a, \{\}$

Exit: $\{\}\{x \leftarrow a\} = \{x \leftarrow a\}$

Call: $x, y, \{x \leftarrow a\}$

Call: $a, y, \{x \leftarrow a\}$

Exit: $\{x \leftarrow a\}\{y \leftarrow a\} = \{x \leftarrow a, y \leftarrow a\}$

Exit: $\{x \leftarrow a, y \leftarrow a\}$

Exit: $\{x \leftarrow a, y \leftarrow a\}$

14

Example

Call: $p(x, x), p(a, b), \{\}$

Call: $p, p, \{\}$

Exit: $\{\}$

Call: $x, a, \{\}$

Exit: $\{\}\{x \leftarrow a\} = \{x \leftarrow a\}$

Call: $x, b, \{x \leftarrow a\}$

Call: $a, b, \{x \leftarrow a\}$

Exit: *false*

Exit: *false*

Exit: *false*

15

Example

Call: $p(x, x), p(y, f(y)), \{\}$

Call: $p, p, \{\}$

Exit: $\{\}$

Call: $x, y, \{\}$

Exit: $\{\}\{x \leftarrow y\} = \{x \leftarrow y\}$

Call: $x, f(y), \{x \leftarrow y\}$

Call: $y, f(y), \{x \leftarrow y\}$

Exit: *false*

Exit: *false*

Exit: *false*

16

Reason

Circularity Problem:

$$\{x \leftarrow f(y), y \leftarrow f(y)\}$$

Unification Problem:

$$p(x, x) \{x \leftarrow f(y), y \leftarrow f(y)\} = p(f(y), f(y))$$

$$p(y, f(y)) \{x \leftarrow f(y), y \leftarrow f(y)\} = p(f(y), f(f(y)))$$

17

Solution

Before assigning a variable to an expression, first check that the variable does not occur within that expression.

This is called, oddly enough, the *occur check* test.

Prolog does not do the occur check (and is proud of it).

18

