

Ex2:

$$S1 = \{A/?x, B/?y, C/?w, D/?z\}$$

$$S2 = \{g(?x, ?y)/?z\}$$

$$S1S2 = \{A/?x, B/?y, C/?w, D/?z\}$$

Change the order of the substitutions in the previous example.

$$S2S1 = \{A/?x, B/?y, C/?w, g(A, B)/?z\}$$

$$S1S2 \neq S2S1$$



Verifying Consistency of Substitutions

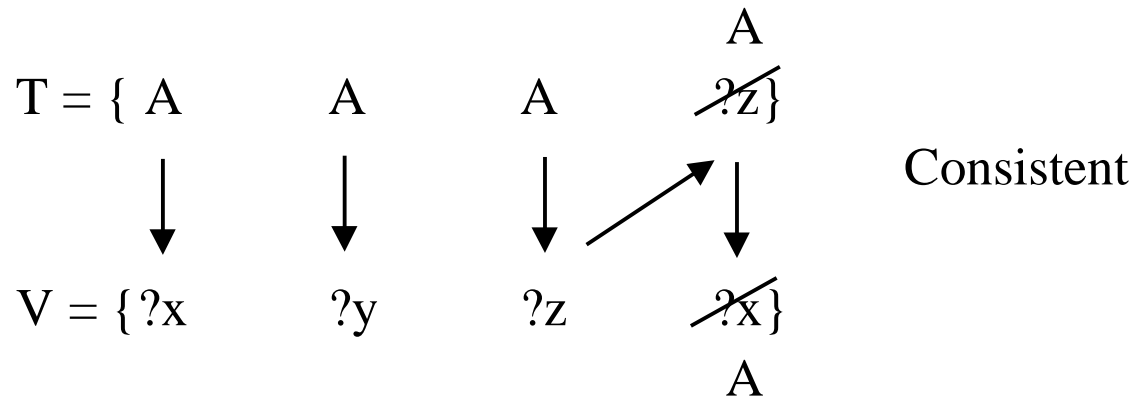
To determine consistency, create two sets, T , V . Put all terms for all involved substitutions in T , and all variables for all involved substitutions in V . Then propagate values of the variables. If any variable has to take more than one constant value, then substitutions are inconsistent.



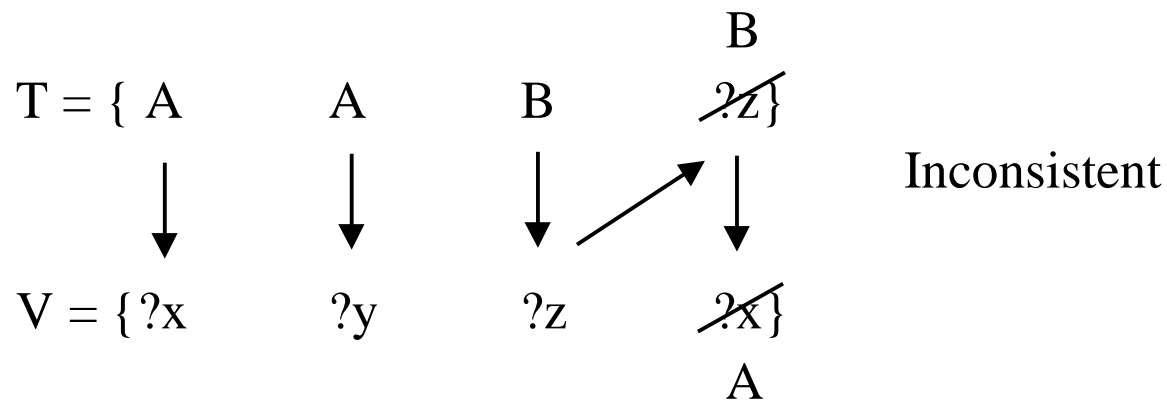
Ex: We have the following substitutions:

$$S1 = \{A/?x, A/?y\}$$

$$S2 = \{A/?z, ?z/?x\}$$



Using B/?z in S2:



How do we find substitutions?

$$\left. \begin{array}{l} W1(?x) \Rightarrow W2(?x) \\ W1(A) \end{array} \right\} \Rightarrow W2(A)$$

Use the recursive Unify procedure for unifying formulas (wffs) and finding substitutions.



Recursive Procedure **UNIFY**(E1, E2)

```
1  if either E1 or E2 is an atom (that is, a predicate symbol, a function
   symbol, a constant symbol, a negation symbol or a variable),
   interchange the arguments E1 and E2 (if necessary) so that E1 is an
   atom, and do:
2      begin
3          if E1 and E2 are identical, return NIL
4          if E1 is a variable, do:
5              begin
6                  if E1 occurs in E2, return FAIL
7                  return {E2/E1}
8              end
9          if E2 is a variable, return {E1/E2}
10         return FAIL
11     end
```

contd...



...contd

12 F1 <- the first element of E1, T1 <- the rest of E1

13 F2 <- the first element of E2, T2 <- the rest of E2

14 Z1 <- **UNIFY**(F1,F2)

15 **if** Z1 = FAIL, **return** FAIL

16 G1 <- result of applying Z1 to T1

17 G2 <- result of applying Z1 to T2

18 Z2 <- **UNIFY**(G1, G2)

19 **if** Z2 = FAIL, **return** FAIL

20 **return** the composition of Z1 and Z2



Ex: {P[A, ?x, f(?y)], P[?y, B, f(?z)]}

First convert everything into lists.

E1 = (P A ?x (f ?y))

E2 = (P ?y B (f ?z))

F1 = P T1 = (A ?x (f ?y))

F2 = P T2 = (?y B (f ?z))

Z1 ← UNIFY(F1, F2) → return NIL



$$G1 = T1 = (A \text{ ?x } (f \text{ ?y}))$$

$$G2 = T2 = (?y B (f ?z))$$

$$Z2 \leftarrow \text{UNIFY } ((A \text{ ?x } (f \text{ ?y})) \text{ (?y B (f ?z))))$$

E1 E2

$$F1 = A \qquad T1 = (?x (f ?y))$$

$$F2 = ?y \qquad T2 = (B (f ?z))$$

$$Z1 \leftarrow \text{UNIFY } (A, ?y) \rightarrow \text{return } (A/?y)$$

F1 F2

$$G1 = (?x (f A))$$

$$G2 = (B (f ?z))$$



$Z2 \leftarrow \text{UNIFY} \left(\underset{E1}{(?x (f A))} \underset{E2}{(B (f ?z))} \right)$

$F1 = ?x$ $T1 = ((f A))$
 $F2 = B$ $T2 = ((f ?z))$

$Z1 \leftarrow \text{UNIFY}(?x B) \rightarrow \text{return } (B/?x)$

$Z2 \leftarrow \text{UNIFY}(((f A)) ((f ?z)))$
:
 $Z2 \leftarrow \text{UNIFY}((A) (?z))$

$F1 = A$ $T1 = ()$
 $F2 = ?z$ $T2 = ()$

$Z1 \leftarrow \text{UNIFY} (A ?z) \rightarrow \text{return } (A/?z)$



Compose ((A/?z) (B/?x))

↓
((A/?z, B/?x))

Compose ((A/?y) (A/?z, B/?x))

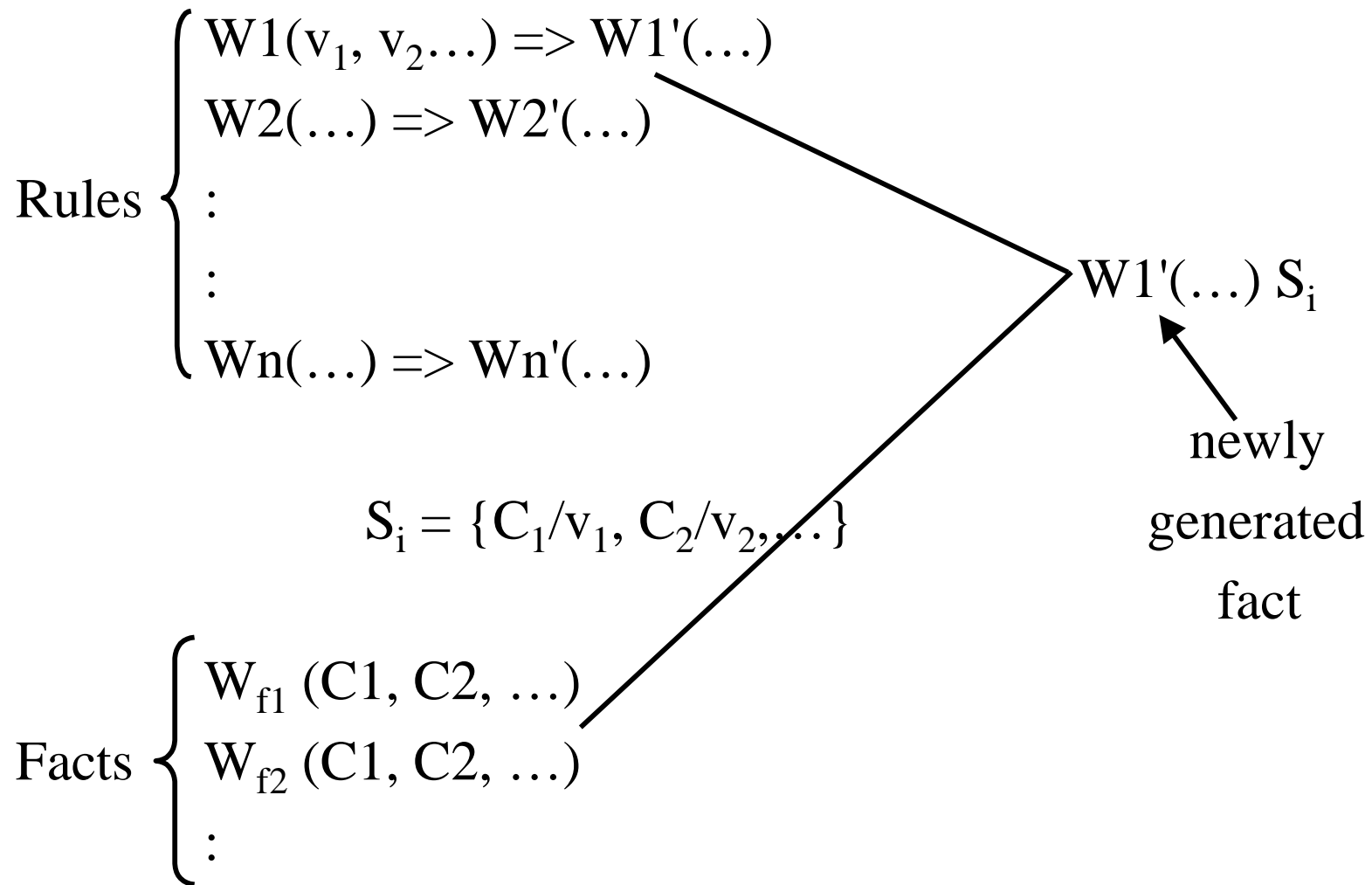
↓
((A/?y, A/?z, B/?x))

Compose (NIL (A/?y, A/?z, B/?x))

↓
((A/?y, A/?z, B/?x))



Rule Systems: Simplest Knowledge Systems



Forward Chaining: match fact expressions with antecedents, generate new facts.

Backward Chaining: match goal or subgoal expressions with consequents, generate new subgoals.



Figure - The forward-chaining inference algorithm. It adds to KB all the sentences that can be inferred from the sentence p . If p is already in KB , it does nothing. If p is new, consider each implication that has a premise that matches p . For each such implication, if all the remaining premises are in KB , then infer the conclusion. If the premises can be matched several ways, then infer each corresponding conclusion. The substitution of θ keeps track of the way things match.



Review and Exercises

- The following expressions are given:

Manager(PURCHASING-DEPT, JOHN-JONES)

Works-in(PURCHASING-DEPT, JOE-SMITH)

We also have the following production:

$$[\text{Works-in}(\text{?x}, \text{?y}) \wedge \text{Manager}(\text{?x}, \text{?z})]$$
$$\Rightarrow \text{Boss-of}(\text{?y}, \text{?z})$$

- a) Using these expressions, show the forward inference steps which show that JOHN-JONES is the boss of JOE-SMITH.
- b) Write a goal for finding the boss of JOE-SMITH. Using the above expressions, by setting up subgoals through backward inference steps and matching show JOHN-JONES is his boss.



1a)

W_{f1} Manager(PURCHASING-DEPT, JOHN-JONES) } F
 W_{f2} Works-in(PURCHASING-DEPT, JOE-SMITH) }

W_{R1} [Works - in(?x, ?y) \wedge Manager(?x, ?z)] } R
 \Rightarrow Boss - of(?y, ?z)
W'

S = {PURCHASING-DEPT/?x } Result
JOE-SMITH/?y, } from
JOHN-JONES /?z} Alg. Unify

Add to database W's

Boss-of (?y, ?z) S =

W_{f3} Boss-of(JOE-SMITH, JOHN-JONES)
goal



1b)

goal: Boss-of(JOE-SMITH, ?t)

S2 = {JOE-SMITH /?y, ?t /?z}

New subgoals:

$$\left\{ \begin{array}{l} \text{Works-in}(\text{?x}, \text{?y}) \text{ S2} \\ \text{Manager}(\text{?x}, \text{?z}) \text{ S2} \end{array} \right.$$

$$\left\{ \begin{array}{ll} \text{Works-in}(\text{?x}, \text{JOE-SMITH}) \text{ sg1} \\ \text{Manager}(\text{?x}, \text{?t}) \text{ sg2} \end{array} \right.$$


Match with facts in database:

$S3 = \{\text{PURCHASING-DEPT}/?x\}$

Works-in(PURCHASING-DEPT, JOE-SMITH)

sg1 $S3 = W_{f2}$

$S4 = \{\text{PURCHASING-DEPT}/?x, \text{JOHN-JONES } /?t\}$

sg2 $S4 = W_{f1}$



Compose S2, S3, S4 and check for consistency.

T = {JOE-SMITH ?t PURCHASING-DEPT ...*}

V = { ?y ?z ?x ...**}

*... PURCHASING -DEPT JOHN-JONES}

**... ?x ?t }



Compose S2, S3, S4 and check for consistency.

JOHN-JONES

$T = \{ \text{JOE-SMITH } \cancel{?t} \text{ PURCHASING-DEPT } \dots * \}$

$$\begin{array}{ccc}
 \downarrow & \downarrow & \downarrow \\
 V = \{ \cancel{?y} & \cancel{?z} & \cancel{?x} \dots ** \\
 \text{JOE-SMITH} & & \text{PURCHASING-DEPT}
 \end{array}$$

JOHN-JONES

$$\begin{array}{ccc}
 * \dots & \text{PURCHASING -DEPT} & \text{JOHN-JONES} \} \\
 & \downarrow & \updownarrow \\
 ** \dots & \cancel{?x} & ?t \} \\
 & \text{PURCHASING-DEPT} &
 \end{array}$$

Consistent substitutions

Answer: John-Jones/?t

