Chaining

63

Logical Inference

- Given a set of axioms (i.e., a set of sentences assumed to be true).
- Generate a set of theorems (i.e., a set of sentences inferred to be true from the axioms.
- Two approaches:
  - Forward Chaining
  - Backward Chaining

64

Forward Chaining

- From known facts, infer new facts by matching facts to l.h.s. of rules and inferring r.h.s.
- This approach makes use of Modus Ponens.
- Inference process continues by "chaining" through rules until desired conclusions are reached.

65

Forward Chaining

function FOL-FC-ASK(KB,α) **returns** a substitution or *false*
  inputs: KB, the knowledge base
      α, the query
  local variables: *new*, new inferred sentences

  repeat until *new* is empty
    new ← {}
    for each sentence r **in** KB **do**
      $(p_i \wedge \ldots \wedge p_n \Rightarrow q) \leftarrow$ STANDARDIZE-APART(r)
      for each θ such that SUBST(θ, $p_1' \wedge \ldots \wedge p_n'$) for some $p_1', \ldots, p_n'$ in KB
        $q' \leftarrow$ SUBST(θ,q)
        if q' is not a renaming of some sentence in KB or *new* **then do**
          add q' to *new*
          $\phi \leftarrow$ UNIFY(q',α)
          if ϕ is not *fail* **then return** ϕ
    add *new* to KB
  return *false*

66

Example

- Add facts in turn, firing rules as appropriate.
  1. $Buffalo(x) \wedge Pig(y) \Rightarrow Faster(x,y)$
  2. $Pig(y) \wedge Slug(z) \Rightarrow Faster(y,z)$
  3. $Faster(x,y) \wedge Faster(y,z) \Rightarrow Faster(x,z)$
  4. $Buffalo(Bob)$
  5. $Pig(Pat)$
     - $\rightarrow$ 6. $Faster(Bob,Pat)$ [1 $\{x/Bob,y/Pat\}$,4,5]
  7. $Slug(Steve)$
     - $\rightarrow$ 8. $Faster(Pat,Steve)$ [2 $\{y/Pat,z/Steve\}$,5,7]
       - $\rightarrow$ 9. $Faster(Bob,Steve)$ [3,6,8]

MONTANA STATE UNIVERSITY | NORM ASBJORNSON College of ENGINEERING

67

Backward Chaining

- Start with goal conclusion and state as hypothesis (i.e., something assumed to be true).
- Match goal to r.h.s of rules and take l.h.s. as new sub-goal.
- Chain back through rules until known facts are found.

MONTANA STATE UNIVERSITY | NORM ASBJORNSON College of ENGINEERING

68

Backward Chaining

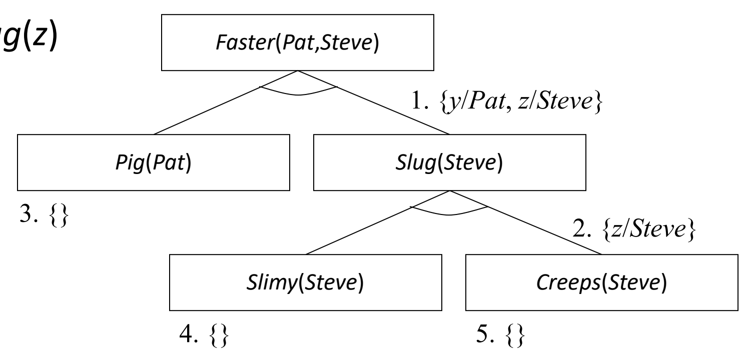
function FOL-BC-ASK(*KB*, *goals*, θ) **returns** a set of substitutions
  inputs: *KB*, the knowledge base
    goals, a list of conjuncts forming a query (θ already applied)
    θ, the current substitution, initial empty
  local variables: *answers*, a set of substitutions, initially empty

  if *goals* is empty **then return** $\{\theta\}$
  $q' \leftarrow$ SUBST(θ,FIRST(*goals*))
  for each sentence r **in** *KB* where
    STANDARDIZE-APART(r) = $(p_i \wedge \ldots \wedge p_n \Rightarrow q)$ and $\theta' \leftarrow$ UNIFY(q, q') succeeds
   new_goals $\leftarrow [p_1,\ldots,p_n|$REST(*goals*)]
   answers $\leftarrow$ FOL-BC-ASK(*KB*, *new_goals*, COMPOSE(θ', θ)) $\cup$ *answers*
  return *answers*

69

Example

1. $Pig(x) \wedge Slug(y) \Rightarrow Faster(x,y)$
2. $Slimy(z) \wedge Creeps(z) \Rightarrow Slug(z)$
3. $Pig(Pat)$
4. $Slimy(Steve)$
5. $Creeps(Steve)$



70