CS344: Introduction to Artificial Intelligence (associated lab: CS386)

Pushpak Bhattacharyya
CSE Dept.,
IIT Bombay

Lecture–30, 31: Predicate Calculus; Interpretation

26th and 29th March, 2012

(Course seminars and discussion on g(n)=g(n) on 27th)*

Himalayan Club example

- Introduction through an example *(Zohar Manna, 1974):*
  - Problem: A, B and C belong to the Himalayan club. Every member in the club is either a mountain climber or a skier or both. A likes whatever B dislikes and dislikes whatever B likes. A likes rain and snow. No mountain climber likes rain. Every skier likes snow. *Is there a member who is a mountain climber and not a skier?*
- Given knowledge has:
  - Facts
  - Rules

Example contd.

- Let *mc* denote mountain climber and *sk* denotes skier. Knowledge representation in the given problem is as follows:
  1. *member(A)*
  2. *member(B)*
  3. *member(C)*
  4. $\forall x[member(x) \rightarrow (mc(x) \vee sk(x))]$
  5. $\forall x[mc(x) \rightarrow \sim like(x,rain)]$
  6. $\forall x[sk(x) \rightarrow like(x, snow)]$
  7. $\forall x[like(B, x) \rightarrow \sim like(A, x)]$
  8. $\forall x[\sim like(B, x) \rightarrow like(A, x)]$
  9. *like(A, rain)*
  10. *like(A, snow)*
  11. *Question:* $\exists x[member(x) \wedge mc(x) \wedge \sim sk(x)]$
- We have to infer the 11^{th} expression from the given 10.
- Done through Resolution Refutation.

Club example: Inferencing

1. $member(A)$
2. $member(B)$
3. $member(C)$
4. $\forall x[member(x) \rightarrow (mc(x) \vee sk(x))]$
   - Can be written as
   - $[member(x) \rightarrow (mc(x) \vee sk(x))]$ $\sim member(x) \vee mc(x) \vee sk(x)$
5. $\forall x[sk(x) \rightarrow lk(x, snow)]$
   - $\sim sk(x) \vee lk(x, snow)$
6. $\forall x[mc(x) \rightarrow \sim lk(x, rain)]$
   - $\sim mc(x) \vee \sim lk(x, rain)$
7. $\forall x[like(A, x) \rightarrow \sim lk(B, x)]$
   - $\sim like(A, x) \vee \sim lk(B, x)$

8. $\forall x[\sim lk(A,x) \rightarrow lk(B,x)]$

– $lk(A,x) \vee lk(B,x)$

9. $lk(A,rain)$

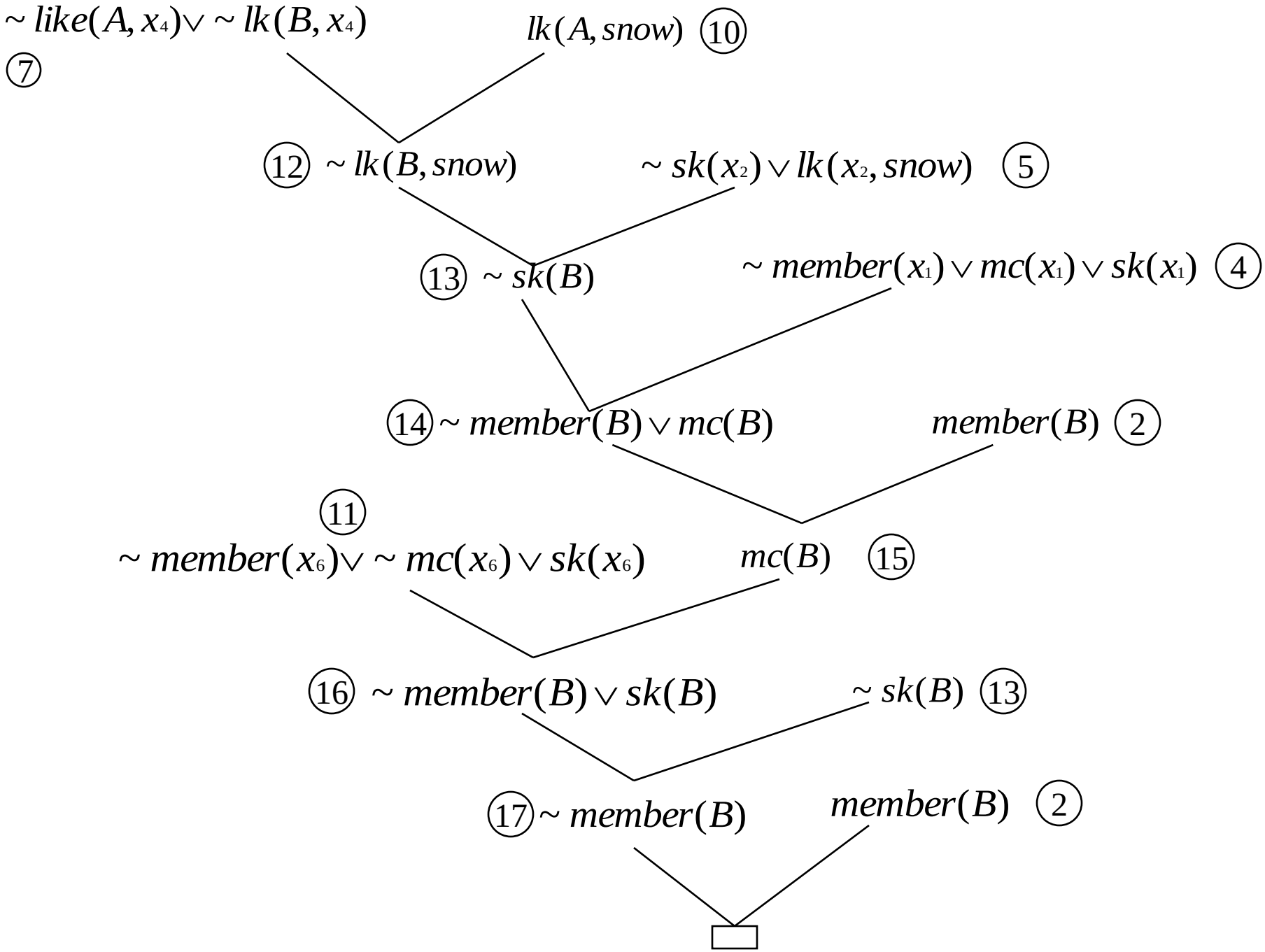
10. $lk(A,snow)$

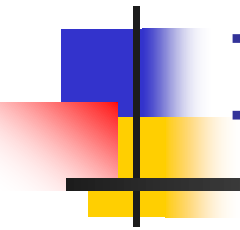
11. $\exists x[member(x) \wedge mc(x) \wedge \sim sk(x)]$

– Negate– $\forall x[\sim member(x) \vee \sim mc(x) \vee sk(x)]$

- Now standardize the variables apart which results in the following

1. $member(A)$
2. $member(B)$
3. $member(C)$
4. $\sim member(x_1) \vee mc(x_1) \vee sk(x_1)$
5. $\sim sk(x_2) \vee lk(x_2, snow)$
6. $\sim mc(x_3) \vee \sim lk(x_3, rain)$
7. $\sim like(A, x_4) \vee \sim lk(B, x_4)$
8. $lk(A, x_5) \vee lk(B, x_5)$
9. $lk(A, rain)$
10. $lk(A, snow)$
11. $\sim member(x_6) \vee \sim mc(x_6) \vee sk(x_6)$


~ like(A, x4) ∨ ~ lk(B, x4)
7
lk(A, snow)
10
12 ~ lk(B, snow)
~ sk(x2) ∨ lk(x2, snow)
5
13 ~ sk(B)
~ member(x1) ∨ mc(x1) ∨ sk(x1)
4
14 ~ member(B) ∨ mc(B)
member(B)
2
11
~ member(x6) ∨ ~ mc(x6) ∨ sk(x6)
mc(B)
15
16 ~ member(B) ∨ sk(B)
~ sk(B)
13
17 ~ member(B)
member(B)
2



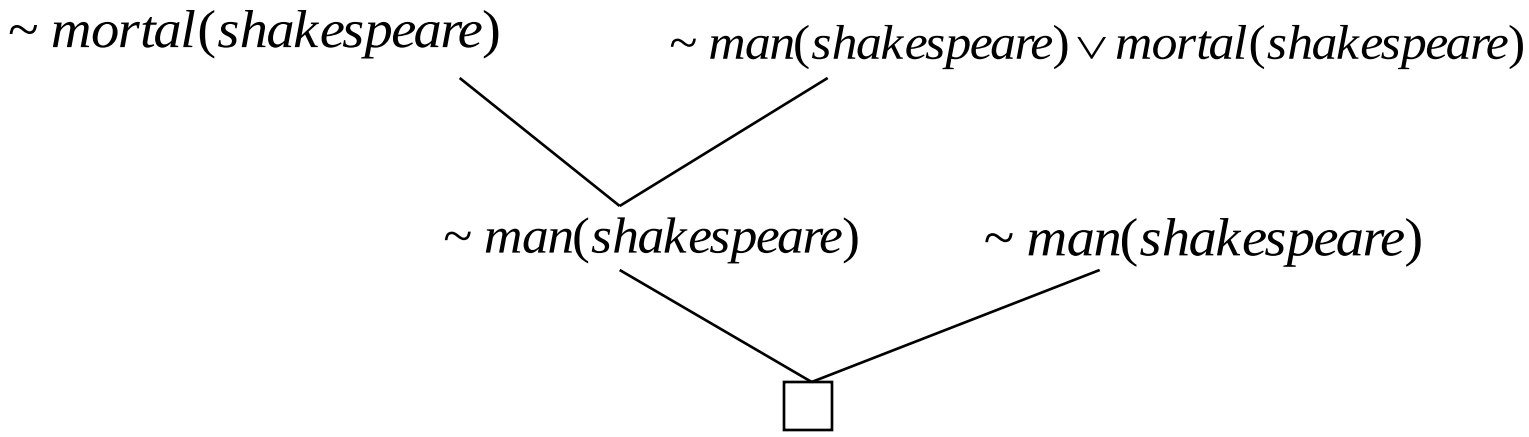
Insight into resolution

Resolution - Refutation

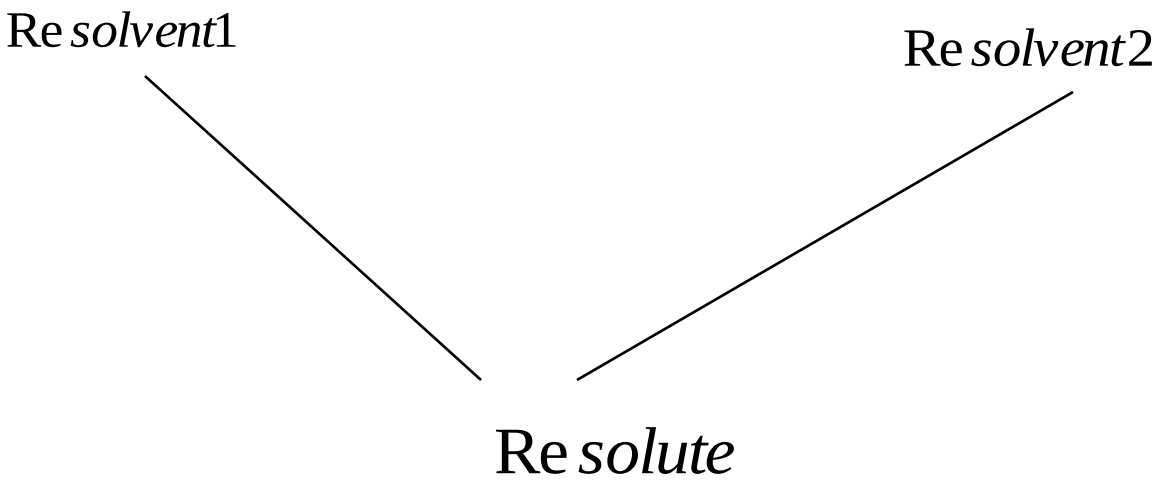
- $man(x) \rightarrow mortal(x)$
  - *Convert to clausal form*
  - *~man(shakespeare) \/ mortal(x)*
- Clauses in the knowledge base
  - *~man(shakespeare) \/ mortal(x)*
  - *man(shakespeare)*
  - *mortal(shakespeare)*

Resolution – Refutation contd

- *Negate the goal*
  - *~man(shakespeare)*
- Get a pair of resolvents



Resolution Tree



Search in resolution

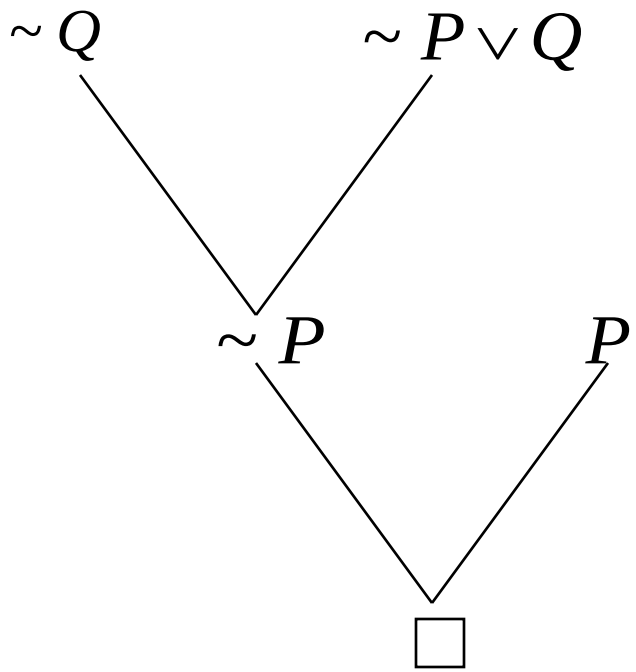
- Heuristics for Resolution Search
  - Goal Supported Strategy
    - Always start with the negated goal
  - Set of support strategy
    - Always one of the resolvents is the most recently produced resolute

Inferencing in Predicate Calculus

- Forward chaining
  - Given P, $P \rightarrow Q$, to infer Q
  - P, match *L.H.S* of
  - Assert Q from *R.H.S*
- Backward chaining
  - Q, Match *R.H.S* of $P \rightarrow Q$
  - assert P
  - Check if P exists
- Resolution – Refutation
  - Negate goal
  - Convert all pieces of knowledge into clausal form (disjunction of literals)
  - See if contradiction indicated by null clause $\square$ can be derived

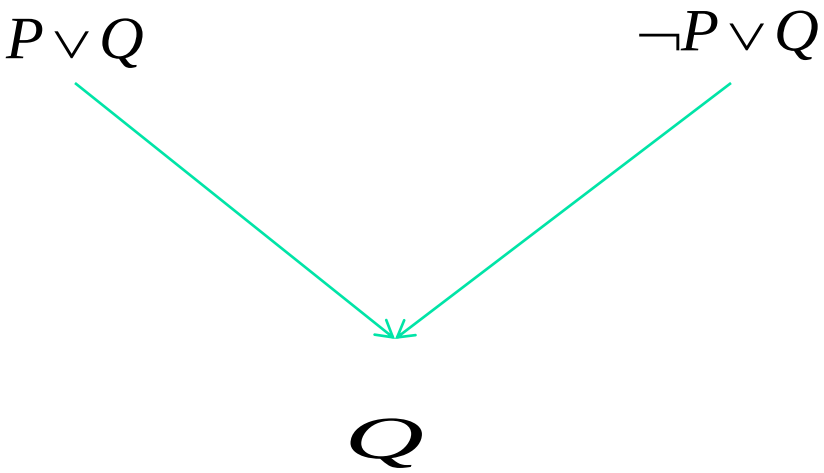
1. P
2. $P \rightarrow Q$ converted to $\sim P \vee Q$
3. $\sim Q$

Draw the resolution tree (actually an inverted tree). Every node is a clausal form and branches are intermediate inference steps.



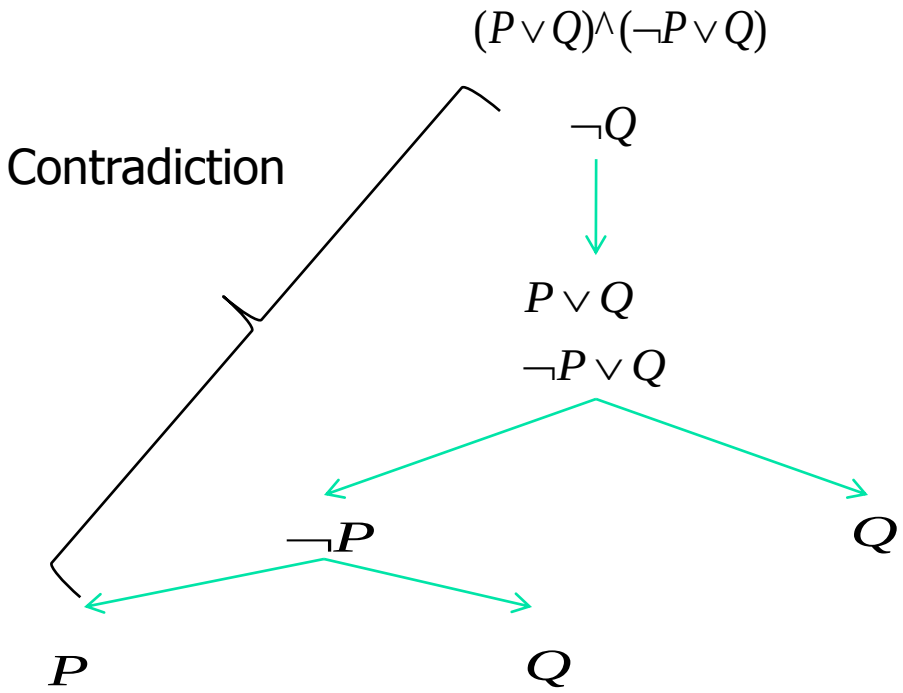
Theoretical basis of Resolution

- Resolution is proof by contradiction
- ***resolvent1 .AND. resolvent2 => resolute*** is a tautology



Tautologiness of Resolution

- Using Semantic Tree



Theoretical basis of Resolution (cont …)

- Monotone Inference
  - Size of Knowledge Base goes on increasing as we proceed with resolution process since intermediate resolvents added to the knowledge base
- Non-monotone Inference
  - Size of Knowledge Base does not increase
  - Human beings use non-monotone inference

Terminology

- Pair of clauses being resolved is called the Resolvents. The resulting clause is called the Resolute.
- Choosing the correct pair of resolvents is a matter of search.