UCS1524 – Logic Programming

Answer Generation in Logic Programming

Session Meta Data

Author	**Dr. D. Thenmozhi**
Reviewer	
Version Number	1.2
Release Date	27 July 2022

Session Objectives

- Understanding answer generation in logic programming
- Learn about the generation of answers from FOL formulas using resolution

Session Outcomes

- At the end of this session, participants will be able to
  - apply the answer generation in logic programming using resolution.

Agenda

- Answer Generation
- Generating single answer
- Generating multiple answer

Answer Generation

- A resolution proof as such shows only that the empty clause is derivable; an answer, in a sense, explains *how it* is obtained.
- Task : how to *generate an answer, a result of the computation,* from the resolution proof.
- Given : Satisfiable set of clauses F
- F consists of predicates, functions, and constants
- Example
  - "Eve likes apples"
  - "Eve likes wine"
  - "Adam likes everybody who likes wine“

Likes: predicate
Eve, apple, wine, Adam: constants

$$
\begin{aligned}
F \;=\; \{\, & \{likes(Eve, Apples)\}, \\
& \{likes(Eve, Wine)\}, \\
& \{likes(Adam, x), \neg likes(x, Wine)\}\,\}
\end{aligned}
$$

Answer Generation

- Call may be “Is there anybody whom Adam likes?”

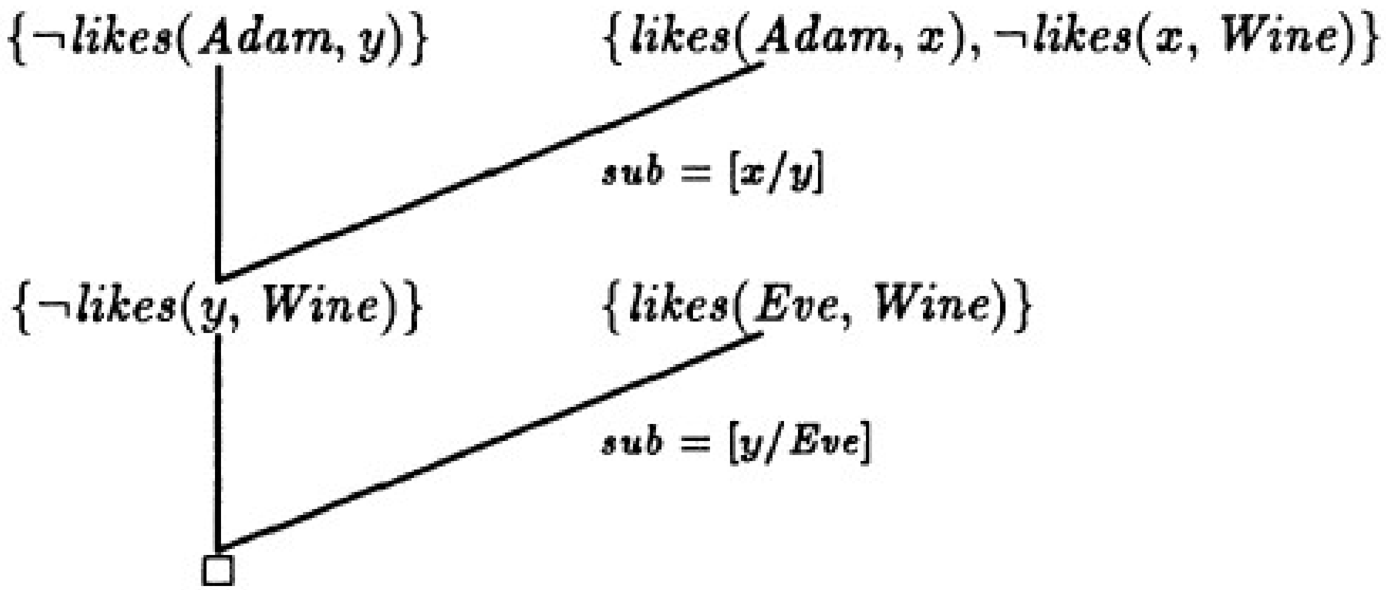
$$G = \exists y\ likes(Adam, y)$$

- We can verify this using resolution by checking whether there is a resolution refutation of $F \wedge \neg G$

$$\begin{aligned} F \wedge \neg G \quad \equiv \quad & \{\, \{likes(Eve, Apples)\}, \\ & \{likes(Eve, Wine)\}, \\ & \{likes(Adam, x), \neg likes(x, Wine)\}, \\ & \{\neg likes(Adam, y)\} \,\}. \end{aligned}$$

Answer Generation

$$F \wedge \neg G \equiv \{\{likes(Eve, Apples)\}, \{likes(Eve, Wine)\}, \{likes(Adam, x), \neg likes(x, Wine)\}, \{\neg likes(Adam, y)\}\}.$$



$\{\neg likes(Adam, y)\}$ $\{likes(Adam, x), \neg likes(x, Wine)\}$

$sub = [x/y]$

$\{\neg likes(y, Wine)\}$ $\{likes(Eve, Wine)\}$

$sub = [y/Eve]$

$\square$

Empty clause is derivable thus unsatisfiable

If variable y is substituted by Eve, we will get the answer Eve

Answer Generation

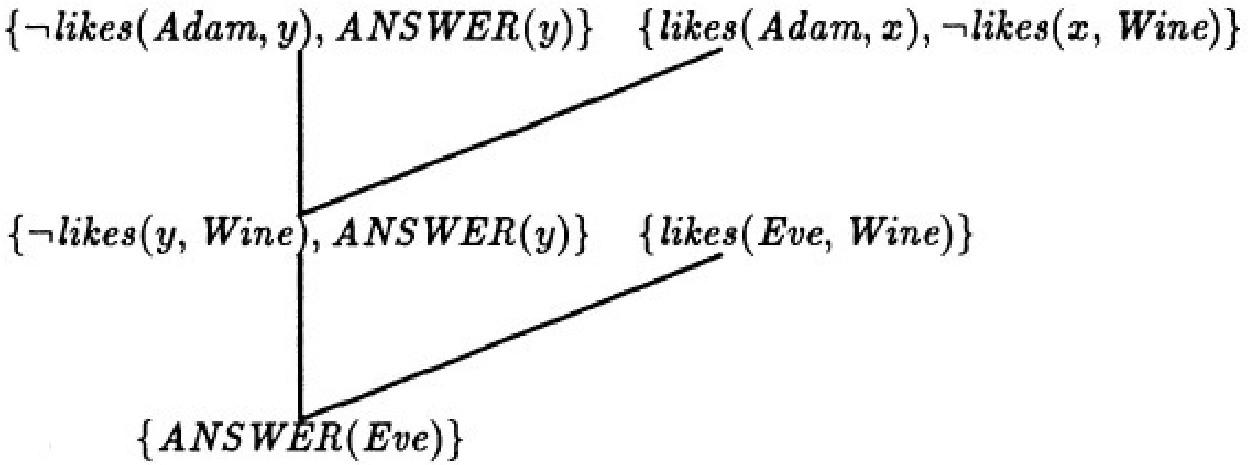
- Instead of

$$\{\neg likes(Adam, y)\}$$

- if we use

$$\{\neg likes(Adam, y), ANSWER(y)\}$$

- We get



$\{\neg likes(Adam, y), ANSWER(y)\}$

$\{likes(Adam, x), \neg likes(x, Wine)\}$

$\{\neg likes(y, Wine), ANSWER(y)\}$

$\{likes(Eve, Wine)\}$

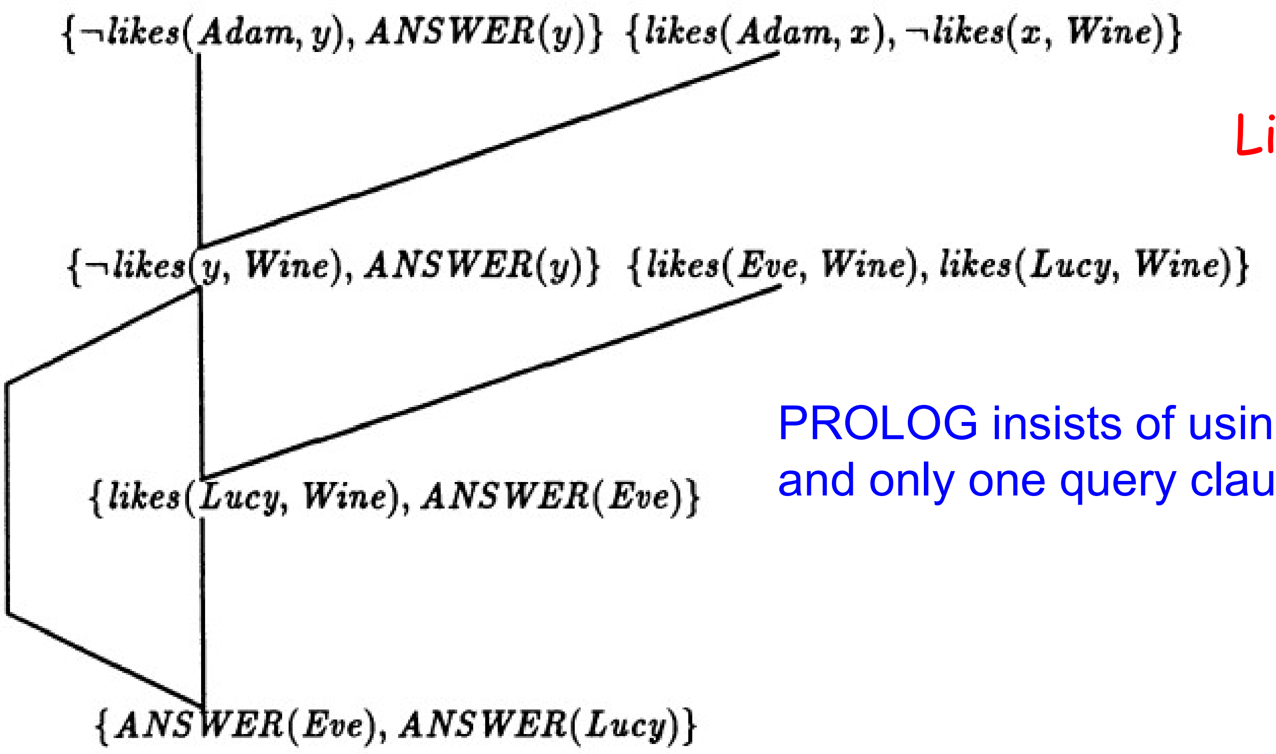
$\{ANSWER(Eve)\}$

Answer Generation

- If the clause set is
  - "Eve likes apples"
  - "Eve or Lucy (or both) like wine"
  - "Adam likes everyone who likes wine

$$\{\ \{likes(Eve, Apples)\},\ \{likes(Eve, Wine), likes(Lucy, Wine)\},\ \{likes(Adam, x), \neg likes(x, Wine)\},\ \{\neg likes(Adam, y), ANSWER(y)\}\ \}$$

- Query is "Who does Adam like?”



$\{\neg likes(Adam, y), ANSWER(y)\}$ $\{likes(Adam, x), \neg likes(x, Wine)\}$

$\{\neg likes(y, Wine), ANSWER(y)\}$ $\{likes(Eve, Wine), likes(Lucy, Wine)\}$

$\{likes(Lucy, Wine), ANSWER(Eve)\}$

$\{ANSWER(Eve), ANSWER(Lucy)\}$

Linear Resolution

PROLOG insists of using only Horn clauses, and only one query clause

Summary

- Answer Generation
- Generating single answer
- Generating multiple answer

Check your understanding

- Given clauses
  - Laxman is wherever Ram is. Ram is at Ayodhya. Where is Laxman?
- Generate answer