UCS1524 – Logic Programming

Horn Clause Programs

Session Meta Data

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

Session Objectives

- Understanding horn clause programs using predicate logic
- Learn the horn clause program using program clauses and goal clause with SLD resolution.

Session Outcomes

- At the end of this session, participants will be able to
  - explain the horn clause program using SLD resolution.

Agenda

- Horn clause program
- Procedure clauses and calls
- Program clause, goal clause and halt clause
- SLD resolution
- Answer extraction

Horn Clause Programs

- Logic programming is restricted to horn clause programs.
- Reasons
  - Most of the mathematical theories seem to be axiomatizable in terms of Horn formulas
  - Allowing clauses that are not Horn leads to more complicated answer situations.
  - Efficient testing for satisfiability
  - Completeness of SLD-resolution is possible with Horn clauses

Procedure Clauses and calls

- Clauses that consist of a single positive literal are called *facts*.
- *Procedure clauses have the form* $\{P, \neg Q_1, \ldots, \neg Q_k\}$

  Where $P, Q_1, \ldots, Q_k$ are certain atomic formulas of predicate logic.
- The notation in PROLOG, namely

$$P :- Q_1, Q_2, \ldots, Q_k$$

- Here, P is called the *procedure head, and the sequence* $Q_1, \ldots, Q_k$ *is called the procedure body.*
- A single *Qi is considered as a procedure call.*

Program clause, goal clause and halting clause

- *A Horn clause program (simply logic program) consists* of a finite set of facts and procedure calls. An element of a logic program is also called *program clause or definite clause.*
- Finally, a logic program is *called or activated by a goal clause. A goal* clause (also called *query clause) is a Horn clause too, but one containing* negative literals only. Such a clause has the form $\{\neg Q_1, \neg Q_2, \ldots, \neg Q_k\}$ or in the PROLOG notation, $?- Q_1, Q_2, \ldots, Q_k$
- In this context the empty clause [] is called the *halting clause. It can* be considered to be the special case of a goal clause (with k = 0) where all procedure calls are successfully performed.

Example

- Let the clauses for recursive definition of the addition

$$(1) \quad \{A(x, 0, x)\}$$
$$(2) \quad \{A(x, s(y), s(z)), \neg A(x, y, z)\}$$

Base clause x+0=x

$$x + y' = (x + y)'$$

Y' Successor of y

A(x, y, z) mean x+y=z

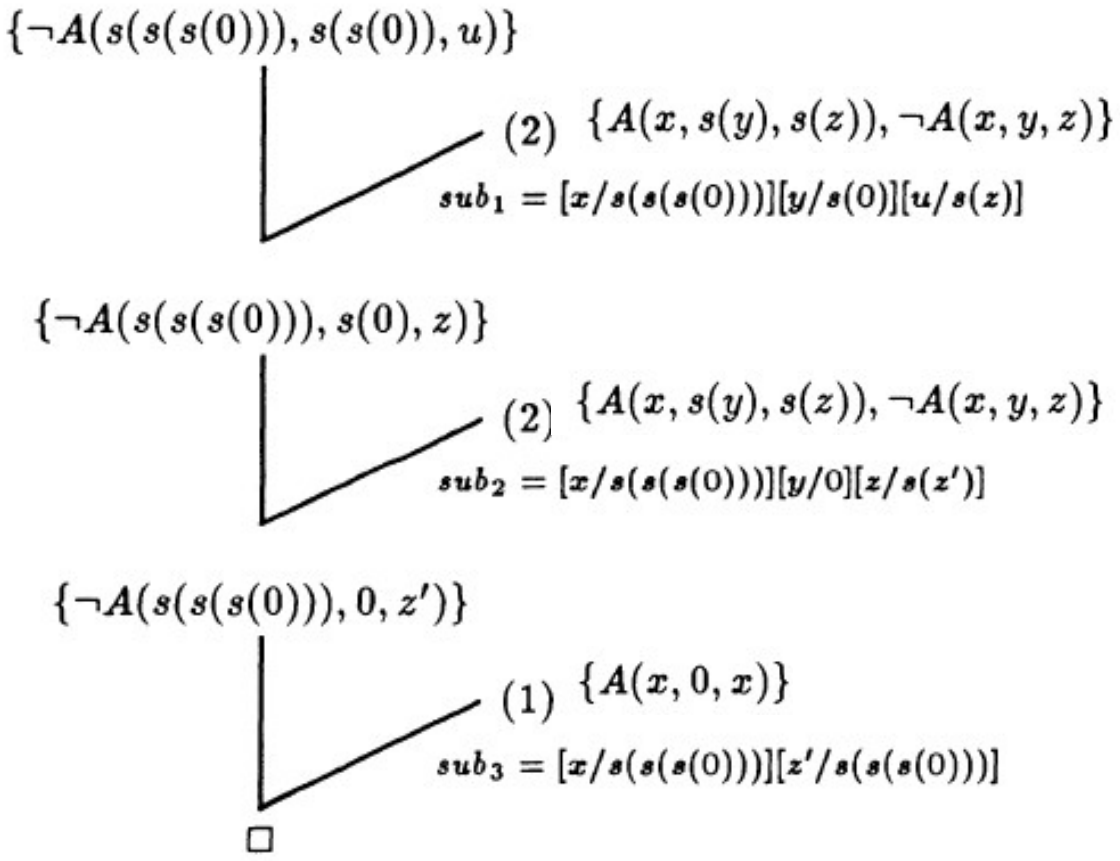
- To compute 3+2 (the result in u), the goal clause is

$$\{\neg A(s(s(s(0))), s(s(0)), u)\}$$

SLD resolution

- The SLD resolution is

In SLD. Start with base clause(-ve), intermediate clauses (-ve)



(1) $\{A(x, 0, x)\}$

(2) $\{A(x, s(y), s(z)), \neg A(x, y, z)\}$

Answer extraction

- An answer, a result of the computation, can be obtained by applying the computed most general unifiers *sub1, sub2, sub3 to the original goal clause.*
- We obtain

$$\{\neg A(s(s(s(0))), s(s(0)), u\} sub_1 sub_2 sub_3 =$$
$$\{\neg A(s(s(s(0))), s(s(0)), s(s(s(s(s(0))))))\}$$

- we can apply the substitution *sublsub2sub3 directly to the variable u occurring in the goal* clause.

$$\begin{aligned} u\, sub_1 sub_2 sub_3 &= s(z) sub_2 sub_3 \\ &= s(s(z')) sub_3 \\ &= s(s(s(s(0)))) \end{aligned}$$

- The result is 5

Summary

- Horn clause program
- Procedure clauses and calls
- Program clause, goal clause and halt clause
- SLD resolution
- Answer extraction

Check your understanding

- Consider the knowledge base
  - 1. ancestor(X,X).
  - 2. ancestor(X,Z) :- parent(X,Y), ancestor(Y,Z).
  - 3. parent(george,sam).
  - 4. parent(george,andy).
  - 5. parent(andy,mary).
  - 6. male(george).
  - 7. male(sam).
  - 8. male(andy).
  - 9. female(mary).
- Query: Find a female descendant of george
- Hint: The Skolemization of Query is "¬ ancestor(george,Q) V ¬ female(Q)"