Logic Programming using

PROLOG

Amir Tavasoli

What is PROLOG?

- Its a way of programming using logical statements. It's the most common logic programming language around.
- There are several implementations of PROLOG
  - **SWI PROLOG**
    - http://www.swi-prolog.org/
  - Turbo PROLOG
    - http://www.fraber.de/university/prolog/tprolog.html
  - Micro PROLOG
    - http://www.lpa.co.uk/dow_fre.htm
  - Visual PROLOG
    - http://www.visual-prolog.com/
- They have some differences in syntax.

PROLOG Syntax 1/3

- We have two kinds of statements in PROLOG.

1 – FACTS (or the Knowledge Bases)

- Like "Jim is a child."
  - In PROLOG we write "**child(jim).**"
- "Joe is father of Jim."
  - **father(joe, jim).**
- "Jill is mother of Jim"
  - **mother(jill, jim).**

PROLOG Syntax (cont.) 2/3

2 – RULES

- Like “A parent is either a father or mother.”
  - `parent(X,Y) :- father(X,Y); mother(X,Y).`
    - “;” is the logical disjunction OR
- “Two persons are siblings if they have the same parents.”
  - `siblings(X,Y) :- parent(Parent,X),parent(Parent,Y).`
    - “,” is the logical conjunction AND
- As you can notice variables are in Uppercase. They need to start with a Uppercase letter or “_”. Like “`Parent`” or “`_parent`” .

PROLOG Syntax (cont.) 3/3

- We call these clauses in logic Horn Clauses.
- Not all FOL statements can be said this way.
- Even thou, PROLOG has its limitations a lot of AI algorithms like **Expert Systems**, Heuristic Searches, Constraint Satisfaction Problems, and … can be easily implemented using PROLOG.

A test program 1/3

- Here are the FACTS of my program.
  - child(jim).
  - father(joe, jim).
  - mother(jill, jim).
  - child(jan).
  - father(joe, jan).
  - mother(jill, jan).

A test program (cont.) 2/3

- After loading the file in PROLOG. We can ask questions from PROLOG like:
  - Is joe is the father of jim?
    - ?father(joe, jim).
  - Who is the father of jim?
    - ?father(X,jim).
  - Whom mother is jill?
    - ?mother(jill, X).
- You can use . and ; to answer the questions.

A test program (cont.) 3/3

- We can add some rules to our program.
  - parent(X,Y) :- father(X,Y); mother(X,Y).
  - siblings(X,Y) :- parent(Parent,X),parent(Parent,Y).
- And we can ask questions from it:
  - "Who are the parents of jan?"
    - ?parent(X,jan).
  - "Who are siblings to each other?"
    - ?siblings(X,Y).

Recursive Rules

- We can have recursive rules in our program.
- For example in natural numbers we can say "successor of a number is also a number."
  - **numeral(succ(X)):-numeral(X).**
  - We should add the fact **numeral(0).** also to the program in order for it to start working. Because we simply need something to start from.
- (Jason Utt, 2005)

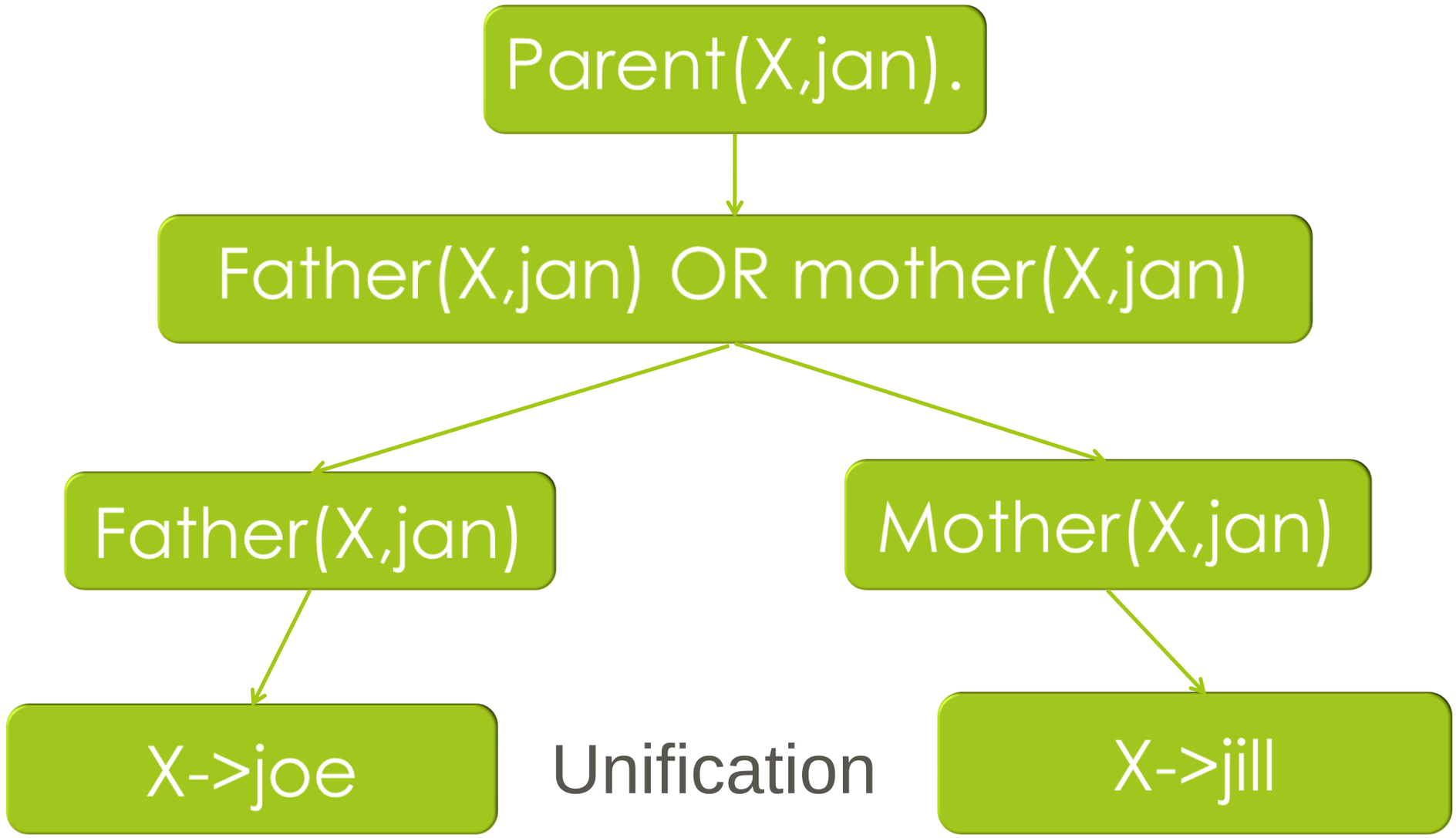
Unification

- The equality symbol "=" in PROLOG is not like the equality symbol in logic.
- The equality symbol in PROLOG is used to unify two terms.
- parent(X,tom)=parent(jim,Y).

How PROLOG works

- PROLOG uses the backward chaining process.
- It starts from the entered statement and tries to go through branches of possible answers that is made by rules using Depth First Search Algorithm.
- PROLOG uses a lot of techniques to make the process more and more efficient.
- You can find more about this in Chapter 9 of “Artificial Intelligence: A Modern Approach” by Russell and Norving 2003

Parent Tree


Parent(X,jan).
Father(X,jan) OR mother(X,jan)
Father(X,jan)
Mother(X,jan)
X->joe
Unification
X->jill

Extending PROLOG

- PROLOG Technology Theorem Prover, or PTTP (Stickel, 1988) is a sound and complete theorem prover. They took the PROLOG compiler and extend it such that, it can be used as a sound and complete reasoner for full First Order Logic. (Russell, 2003)
- In PTTP you can use all of the FOL statements but it has its own deficiencies.

References

- Artificial Intelligence: A Modern Approach, Second Edition, Stuart J. Russell and Peter Norving, 2003

- PROLOG versus You, An Introduction to Logic Programming, A.-L. Johansson, A. Eriksson-Granskog, A. Edman, 1989

- http://www.ims.uni-stuttgart.de/~uttjn/prolog/lpn/3.1.3%20Example%203_%20Successor.pdf (Jason Utt, 2005)