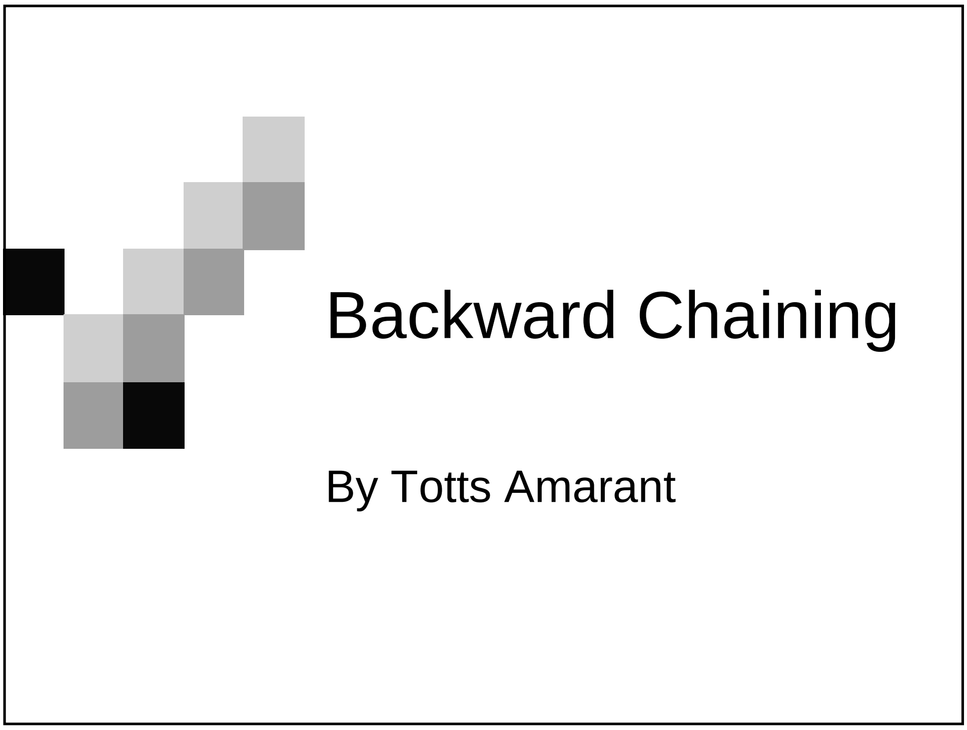
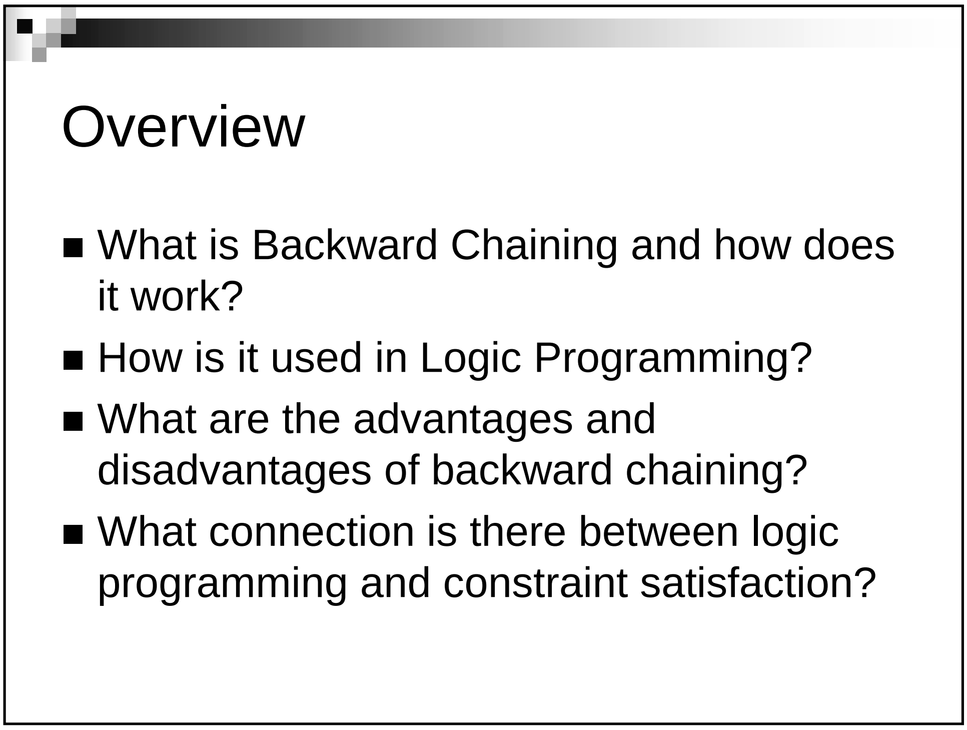


Backward Chaining

By Totts Amarant



Overview

- What is Backward Chaining and how does it work?
- How is it used in Logic Programming?
- What are the advantages and disadvantages of backward chaining?
- What connection is there between logic programming and constraint satisfaction?

Backward Chaining

- Backward chaining is an algorithm that works backwards from the goal, chaining through rules to find known facts that support the proof.
- The FOL-BC-Ask is a backward chaining algorithm (figure 9.6). It is called with a list of goals containing an element, the original query, and returns the set of all substitutions satisfying the query.

A simple backward-chaining algorithm

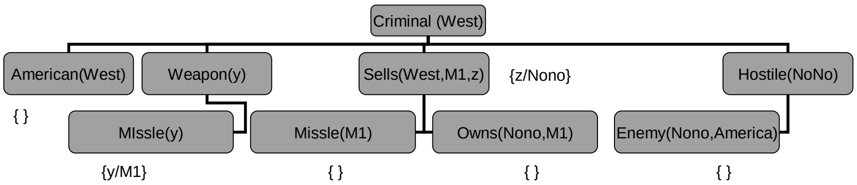
```
Function FOL-BC-Ask(KB,goals,@) returns a set of substitutions
  inputs:  KB, a knowledge base
           goals, a list of conjuncts forming a query (@ already applied)
           @, the current substitution, initially the empty subs. { }
  local variables: answers, a set of substitutions, initially empty { }

  if goals is empty then return {@}
  q' ← Subst(@,First(goals))
  for each sentence r in KB where STANDARDIZE-APART(r)=($p_1$ ^…^$p_n$
     →q) and @' ← Unify (q,q') succeeds
           new_goals ← [$p_1$, lll, $p_n$|Rest(goals)]
           answers ← FOL-BC-Ask(KB, new_goals, Compose(@',@)) U
     answers
return answers
```

Backward Chaining

- Backward chaining is a depth-first search algorithm.
- This means that it has problems with repeated states and incompleteness.
- There is a case of repeated states in the following example.

Example (Figure 9.7)

American(x)^Weapon(y)^Sells(x,y,z)^Hostile(z)→Criminal(x)
Owns(Nono,M1)
Missle(M1)
Missle(x)^Owns(Nono,x) → Sells(West,x,Nono)
Missle(x) → Weapon(x)
Enemy(x,America)→Hostile(x)
American(West)
Enemy(Nono,America)



Criminal (West)

American(West)

Weapon(y)

Sells(West,M1,z)

{z/Nono}

Hostile(NoNo)

{ }

MIssle(y)

Missle(M1)

Owns(Nono,M1)

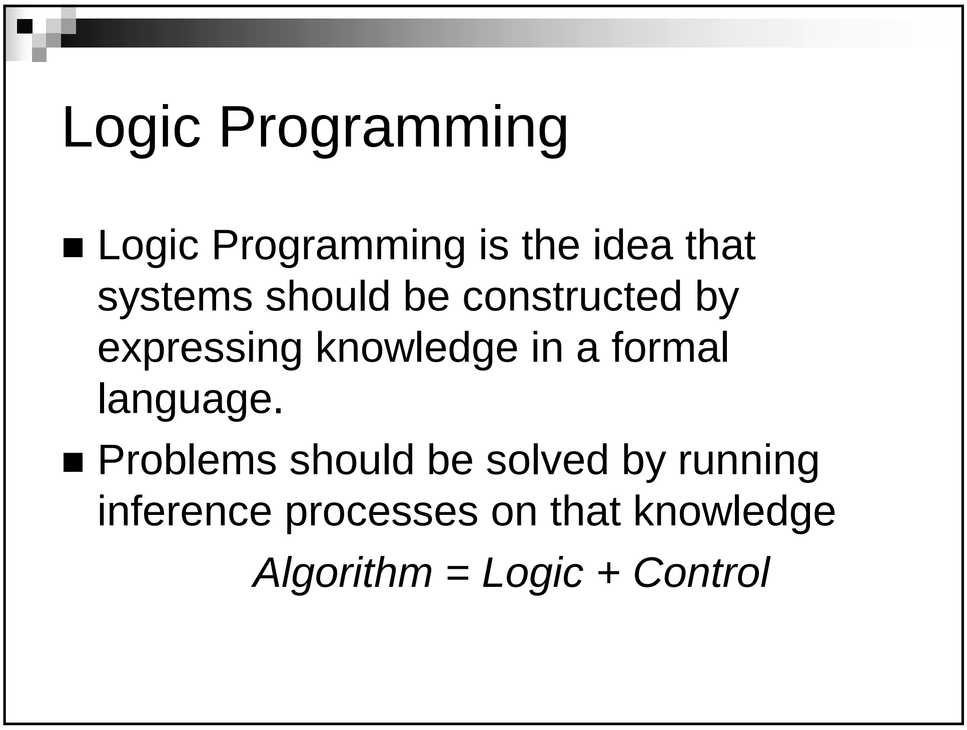
Enemy(Nono,America)

{y/M1}

{ }

{ }

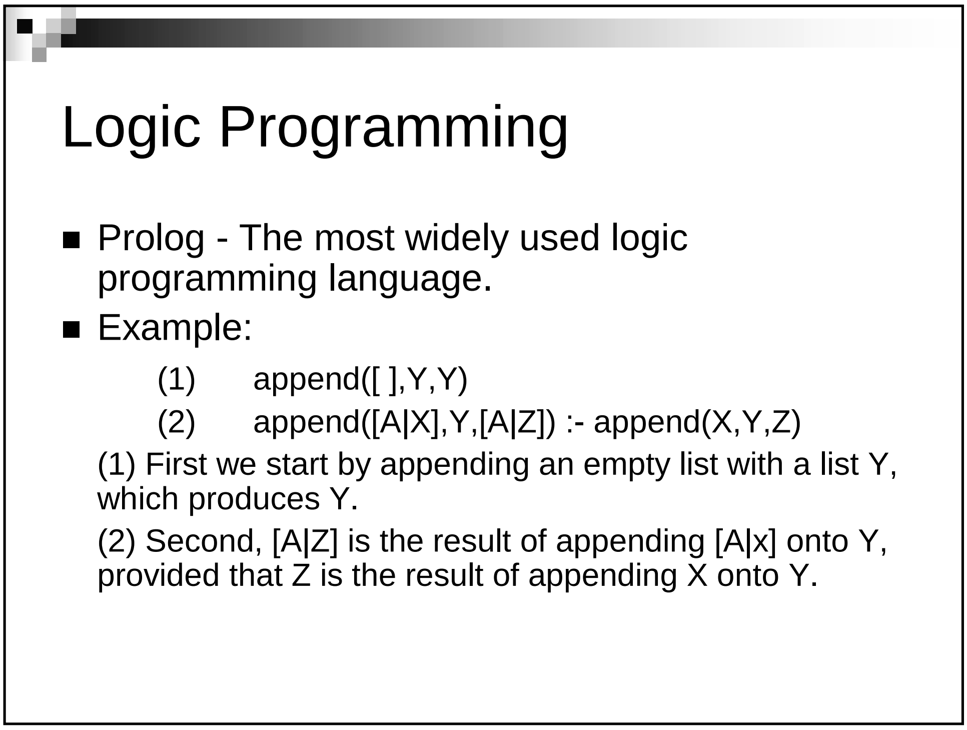
{ }



Logic Programming

- Logic Programming is the idea that systems should be constructed by expressing knowledge in a formal language.
- Problems should be solved by running inference processes on that knowledge

Algorithm = Logic + Control



Logic Programming

- Prolog - The most widely used logic programming language.
- Example:

```
(1)     append([ ],Y,Y)
(2)     append([A|X],Y,[A|Z]) :- append(X,Y,Z)
```

(1) First we start by appending an empty list with a list Y, which produces Y.

(2) Second, [A|Z] is the result of appending [A|x] onto Y, provided that Z is the result of appending X onto Y.

Logic Programming

- If we ask the query append(A,B,[1,2]): what two lists can be appended from A and B to give [1,2]? (Hint, 3 solutions)

Logic Programming

- If we ask the query append(A,B,[1,2]): what two lists can be appended from A and B to give [1,2]? (Hint, 3 solutions)

A = [ ]	B = [1,2]
A = [1]	B = [2]
A = [1,2]	B = [ ]

Logic Programming

- The execution of Prolog is done via depth-first backward chaining.
- A Prolog program can be executed in two modes: interpreted and compiled.
- Interpretation amounts to running the

  FOL-BC-Ask algorithm with the program as the knowledge base.
- Prolog interpreters can contain a variety of improvements to maximize speed and efficiency.

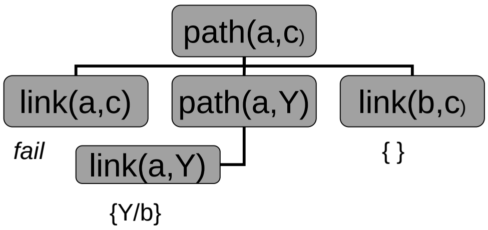
Disadvantages of Prolog

- The Achilles heel of Prolog: The mismatch between depth-first search and search trees that include repeated states and infinite paths.
- Example:

```
path(X,Z) :- link(X,Z).
path(X,Z) :- path(X,Y), link(Y,Z).
```

Disadvantages of Prolog

This generates the tree…



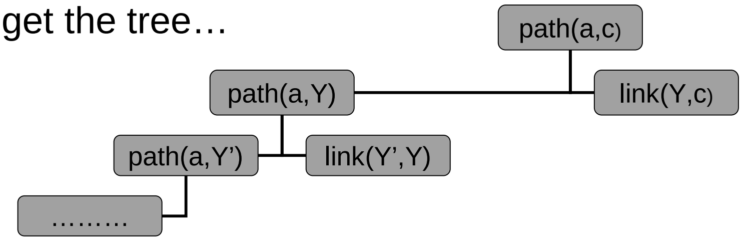
Disadvantages of Prolog

- But, if we have

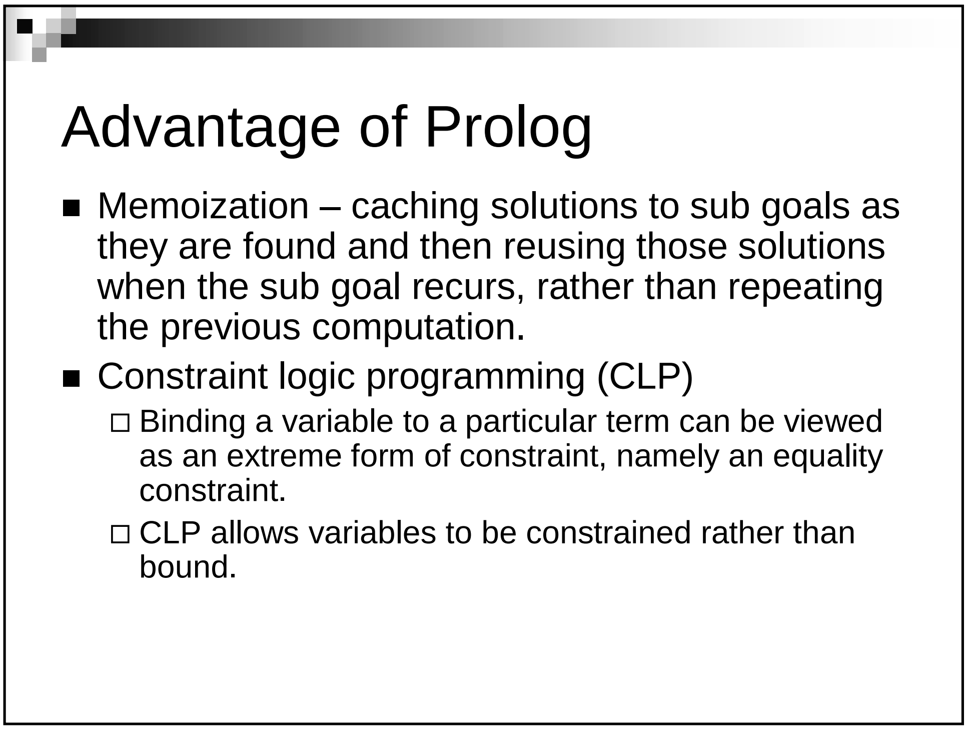
  path(X,Z) :- path(X,Y), link(Y,Z)

  path(X,Z) :- link(X,Z).

  we get the tree…



This shows the infinite path problem along the left side of the tree.



Advantage of Prolog

- Memoization – caching solutions to sub goals as they are found and then reusing those solutions when the sub goal recurs, rather than repeating the previous computation.
- Constraint logic programming (CLP)
  - □ Binding a variable to a particular term can be viewed as an extreme form of constraint, namely an equality constraint.
  - □ CLP allows variables to be constrained rather than bound.