Logical Inference

willie

Representation in FOL

Raj has only two brothers, Jose and German:

No region in South America borders any region in Europe

No two adjacent countries have the same map color

Representation in FOL

Raj has only two brothers, Jose and German:

$Brother(Jose, Raj) \land Brother(German, Raj) \land \neg(Jose = German) \land$

$\forall x\ Brother(x, Raj) \Rightarrow (x = Jose \lor x = German)$

No region in South America borders any region in Europe

$\forall c,d\ In(c, SouthAmerica) \land In(d, Europe) \Rightarrow \neg Border(c,d)$

No two adjacent countries have the same map color

$\forall x,y\ Country(x) \land Country(y) \land Border(x,y) \Rightarrow \neg(Color(x) = Color(y)) \land \neg(x=y)$

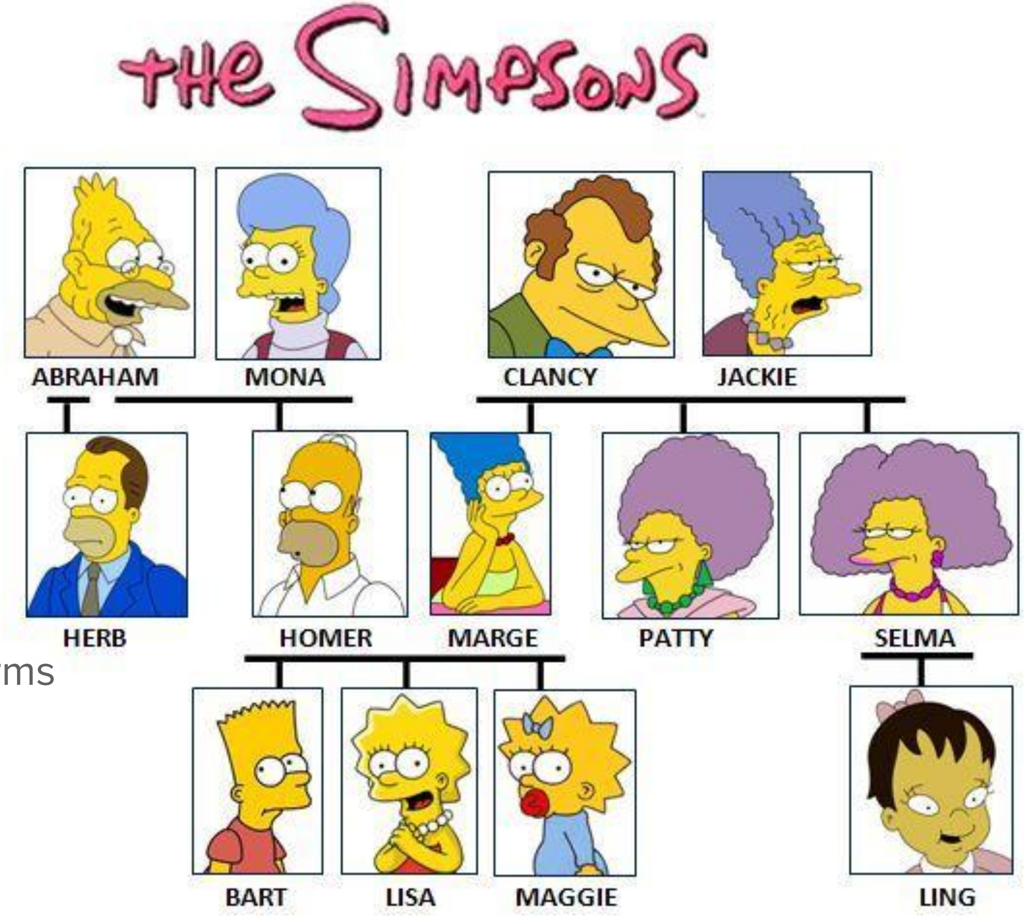
Kinship

Using properties, relations, and functions; express

- Homer likes donuts
- Lisa is smart
- Marge has 3 children
- Bart is male

Using quantifiers, define the following in terms of parent(x,y), male(x), and female(x)

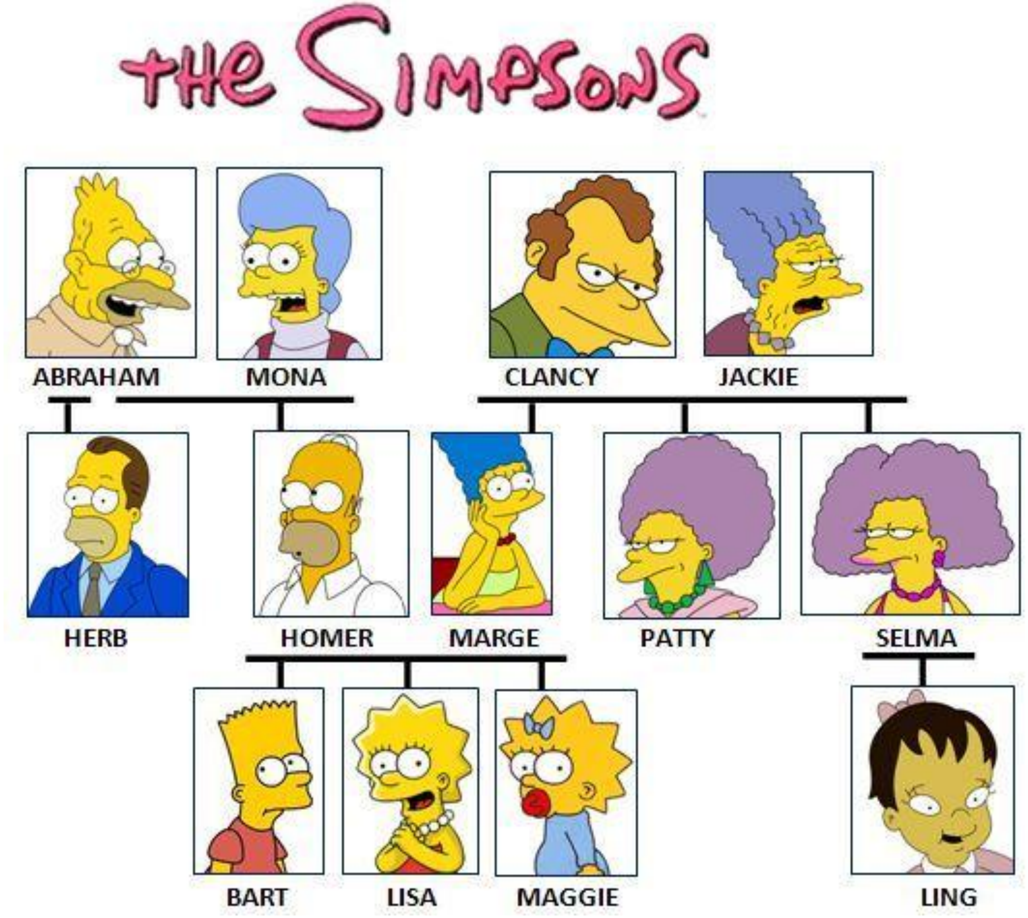
- grandparent(x,y)
- sibling(x,y)
- aunt(x,y)
- cousin(x,y)



Simpsons logic

Using properties, relations, and functions; express

- Homer likes donuts
  likes(Homer, Donuts)
- Lisa is smart
  smart(Lisa)
- Marge has 3 children
  numChildren(Marge) = 3
- Bart is male
  male(Bart)



Kinship

Using quantifiers, define the following in terms of parent(x,y), male(x), and female(x)

- grandparent(x,y)

  $\forall x,y\ \exists p\ \text{grandparent}(x,y) \Leftrightarrow \text{parent}(x,p) \land \text{parent}(p,y)$

- sibling(x,y)

  $\forall x,y\ \exists p\ \text{sibling}(x,y) \Leftrightarrow \text{parent}(p,x) \land \text{parent}(p,y) \land \neg(x = y)$

- aunt(x,y)

  $\forall x,y\ \exists p\ \text{aunt}(x,y) \Leftrightarrow \text{parent}(p,y) \land \text{sibling}(x,p) \land \text{female}(x)$

- cousin(x,y)

  $\forall x,y\ \exists p,q\ \text{cousin}(x,y) \Leftrightarrow \text{parent}(p,x) \land \text{parent}(q,y) \land \text{sibling}(p,q)$

Sherlock Robison & Dr. Wilson

The Sign of the #2

Inference in Propositional Logic

Resolution

Complete and sound inference algorithm

$$\frac{a \lor b \;,\quad \neg a \lor c}{b \lor c}$$

Only works on disjunction of literals

Convert to conjunctive normal form (CNF)

$(L_{11} \lor L_{12} \lor .. \lor L_{1i}) \land (L_{21} \lor L_{22} \lor .. \lor L_{2j}) \land .. \land (L_{n1} \lor L_{n2} \lor .. \lor L_{nk})$

Resolution

To show that KB $\models$ a

We show that {KB, ¬a} is unsatisfiable

Every pair that contains complementary literals is resolved

Continue until there are no new clauses KB $\not\models$ a

Or we derive a contradiction (from {a , ¬a}) KB $\models$ a

Example

$$\begin{tikzcd}
a \lor b \lor \neg c \arrow[dr] & & d \lor c \arrow[dl] & \boxed{KB} & & \\
& a \lor b \lor d \arrow[dr] & & \neg b \arrow[dl] & & \\
& & a \lor d \arrow[dr] & & \neg d \arrow[dl] & \\
& & & a \arrow[dr] & & \neg a \arrow[dl] \\
& & & & \bot &
\end{tikzcd}$$

$KB \vDash b$??

Horn Form

Horn Form (restricted)

KB = **conjunction** of **Horn clause**

- Horn clause =
  - proposition symbol; or
  - (conjunction of symbols) $\Rightarrow$ symbol
- Examples
  - C
  - $A \Rightarrow B$
  - $(C \land D) \Rightarrow B$
  - $(C \land D \land E) \Rightarrow B$

Forward and backward chaining

Modus Ponens (for Horn Form): complete for Horn KBs

$$\frac{\alpha \quad , \quad \alpha \Rightarrow \beta}{\beta}$$

$$\frac{\alpha_1, \ldots, \alpha_n, \qquad \alpha_1 \land \ldots \land \alpha_n \Rightarrow \beta}{\beta}$$

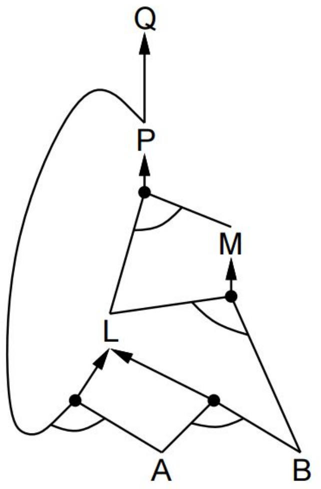
- Can be used with **forward chaining** or **backward chaining**
- These algorithms are very natural and run in linear time

Forward chaining

Idea: fire any rule whose premises are satisfied in the *KB*,

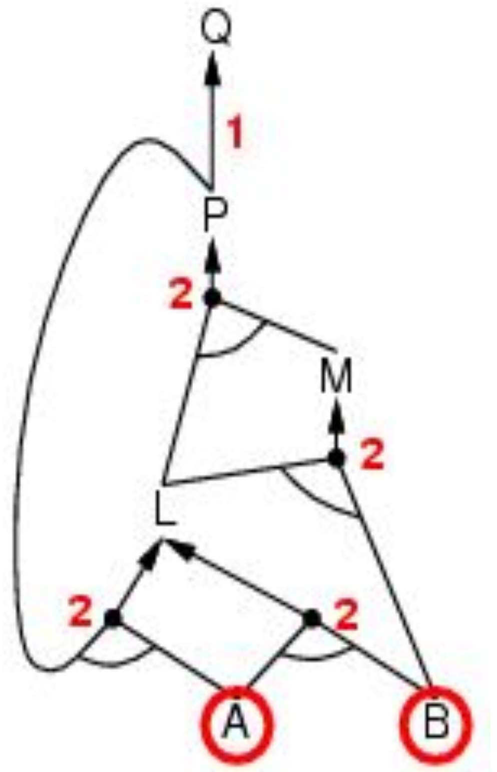
- add its conclusion to the *KB*, until query is found

$$P \Rightarrow Q$$
$$L \wedge M \Rightarrow P$$
$$B \wedge L \Rightarrow M$$
$$A \wedge P \Rightarrow L$$
$$A \wedge B \Rightarrow L$$
$$A$$
$$B$$



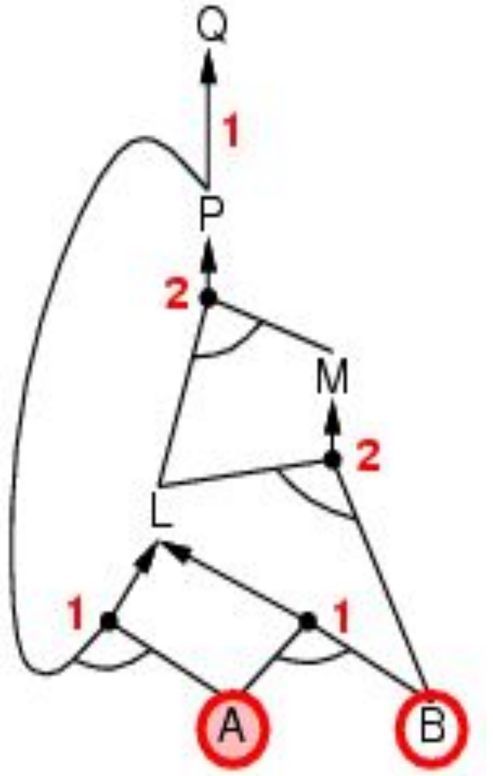
Forward chaining example

$$P \Rightarrow Q$$
$$L \wedge M \Rightarrow P$$
$$B \wedge L \Rightarrow M$$
$$A \wedge P \Rightarrow L$$
$$A \wedge B \Rightarrow L$$
$$A$$
$$B$$



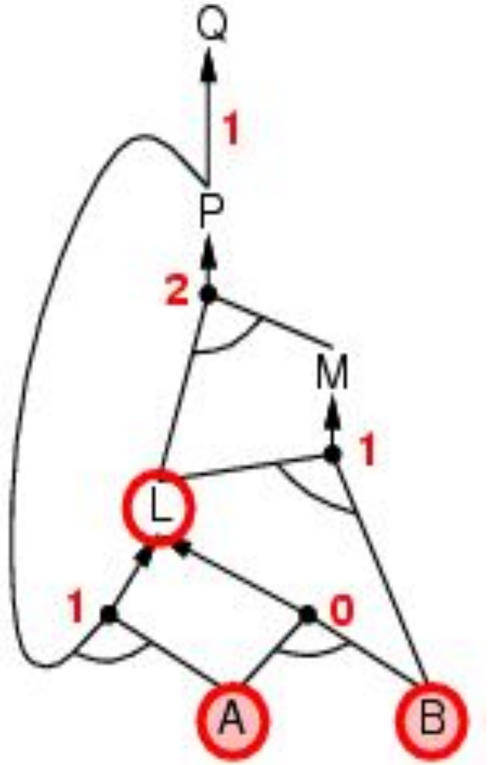
Forward chaining example

$$P \Rightarrow Q$$
$$L \wedge M \Rightarrow P$$
$$B \wedge L \Rightarrow M$$
$$A \wedge P \Rightarrow L$$
$$A \wedge B \Rightarrow L$$
$$A$$
$$B$$



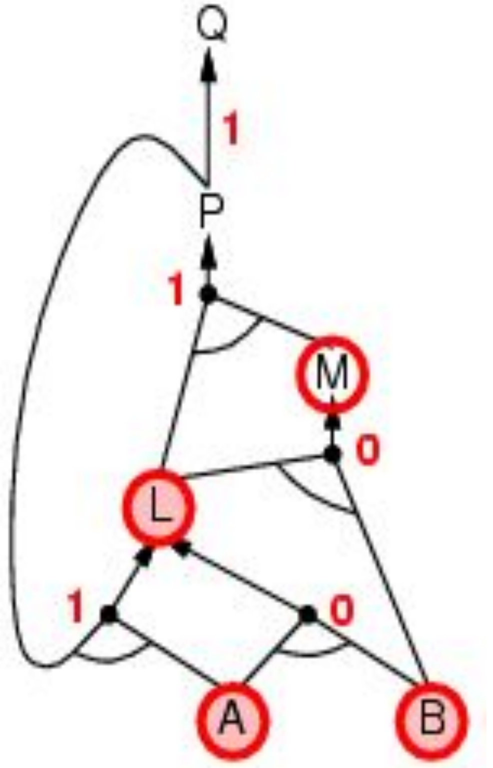
Forward chaining example

$$P \Rightarrow Q$$
$$L \wedge M \Rightarrow P$$
$$B \wedge L \Rightarrow M$$
$$A \wedge P \Rightarrow L$$
$$A \wedge B \Rightarrow L$$
$$A$$
$$B$$



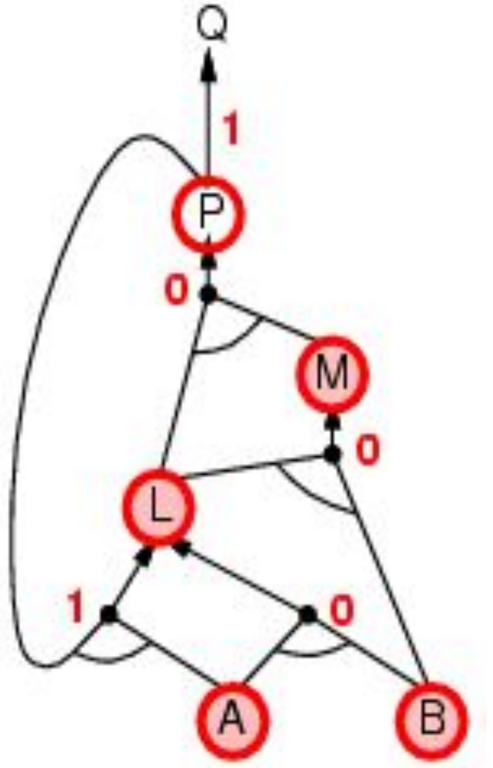
Forward chaining example

$$P \Rightarrow Q$$
$$L \wedge M \Rightarrow P$$
$$B \wedge L \Rightarrow M$$
$$A \wedge P \Rightarrow L$$
$$A \wedge B \Rightarrow L$$
$$A$$
$$B$$



Forward chaining example

$$
\begin{aligned}
P &\Rightarrow Q \\
L \wedge M &\Rightarrow P \\
B \wedge L &\Rightarrow M \\
A \wedge P &\Rightarrow L \\
A \wedge B &\Rightarrow L \\
A & \\
B &
\end{aligned}
$$



Forward chaining example

$P \Rightarrow Q$

$L \land M \Rightarrow P$

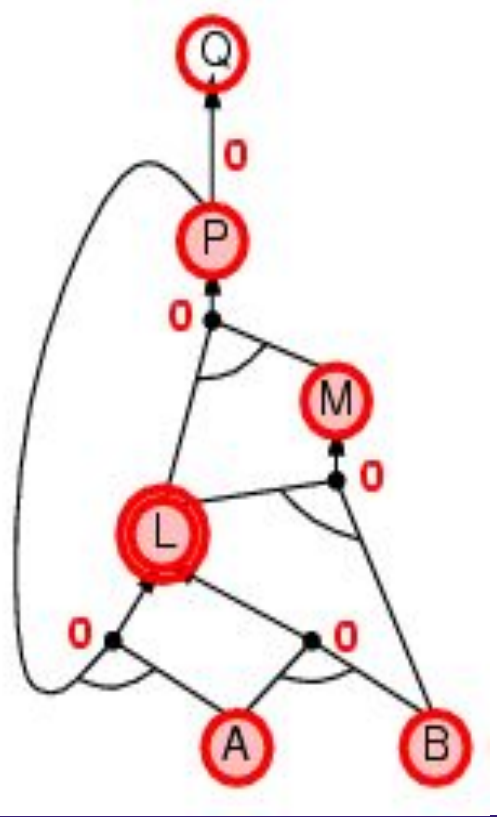
$B \land L \Rightarrow M$

$A \land P \Rightarrow L$

$A \land B \Rightarrow L$

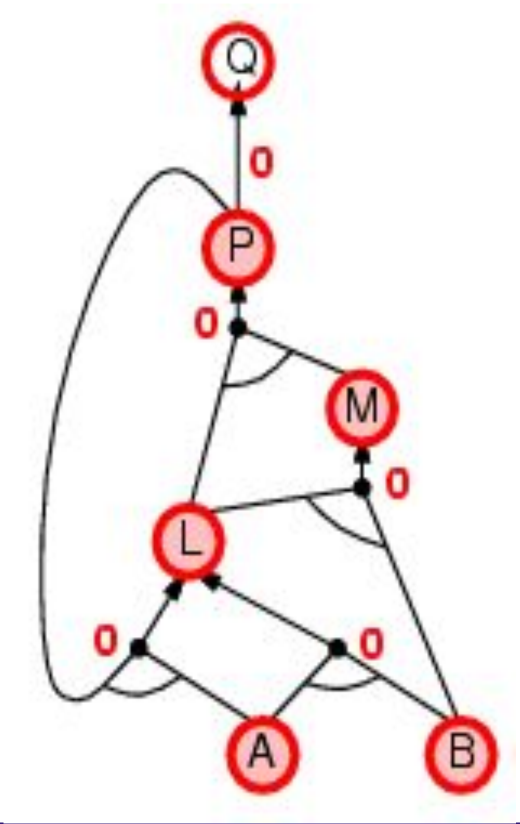
A

B



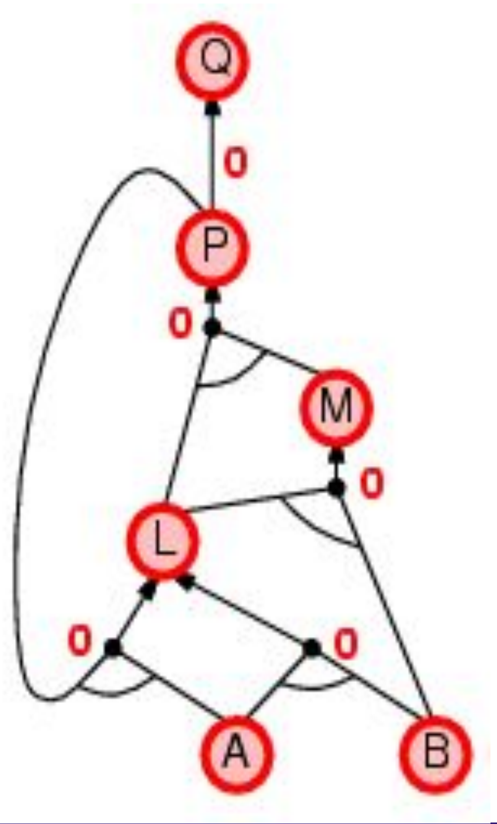
Forward chaining example

$$P \Rightarrow Q$$
$$L \wedge M \Rightarrow P$$
$$B \wedge L \Rightarrow M$$
$$A \wedge P \Rightarrow L$$
$$A \wedge B \Rightarrow L$$
$$A$$
$$B$$



Forward chaining example

$$P \Rightarrow Q$$
$$L \wedge M \Rightarrow P$$
$$B \wedge L \Rightarrow M$$
$$A \wedge P \Rightarrow L$$
$$A \wedge B \Rightarrow L$$
$$A$$
$$B$$



Backward chaining

Idea: work backwards from the query *q*:

    to prove *q* by BC,

        check if *q* is known already, or

        prove by BC all premises of some rule concluding *q*

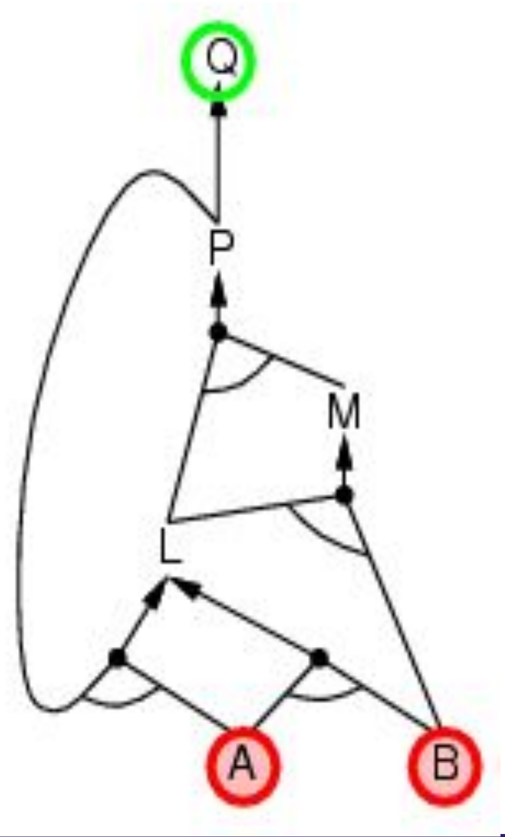
Avoid loops: check if new a subgoal is already on the goal stack

Avoid repeated work: check if new subgoal

1. has already been proven true, or
2. has already failed

Backward chaining example

$$P \Rightarrow Q$$
$$L \wedge M \Rightarrow P$$
$$B \wedge L \Rightarrow M$$
$$A \wedge P \Rightarrow L$$
$$A \wedge B \Rightarrow L$$
$$A$$
$$B$$



Backward chaining example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

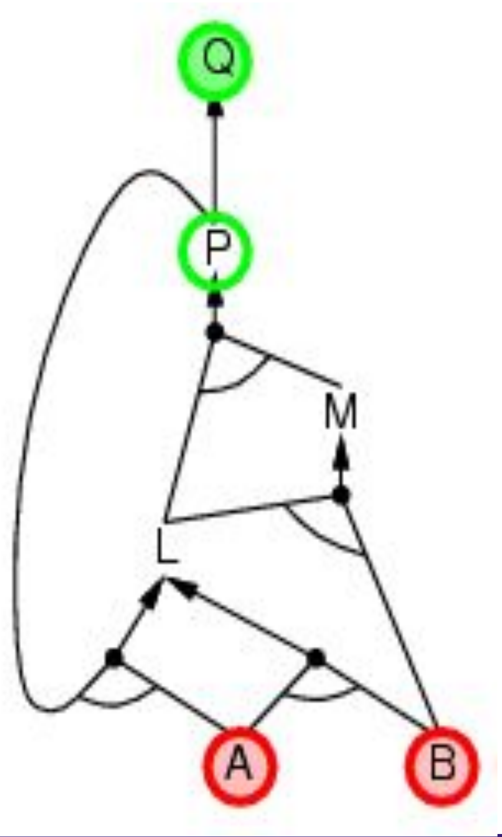
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

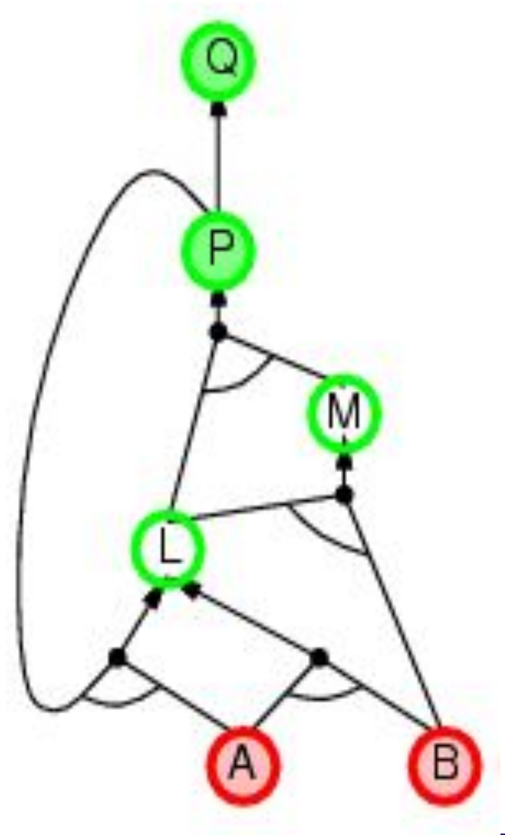
A

B



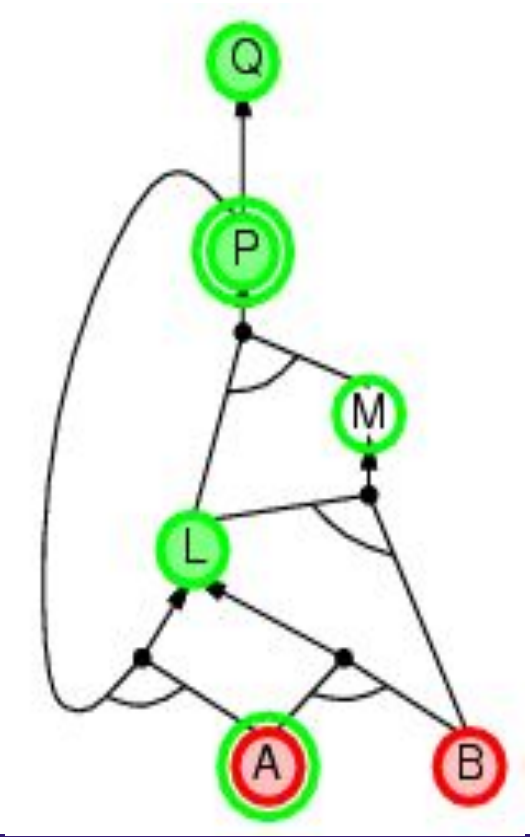
Backward chaining example

$$
\begin{aligned}
&P \Rightarrow Q \\
&L \land M \Rightarrow P \\
&B \land L \Rightarrow M \\
&A \land P \Rightarrow L \\
&A \land B \Rightarrow L \\
&A \\
&B
\end{aligned}
$$



Backward chaining example

$$P \Rightarrow Q$$
$$L \wedge M \Rightarrow P$$
$$B \wedge L \Rightarrow M$$
$$A \wedge P \Rightarrow L$$
$$A \wedge B \Rightarrow L$$
$$A$$
$$B$$



Backward chaining example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

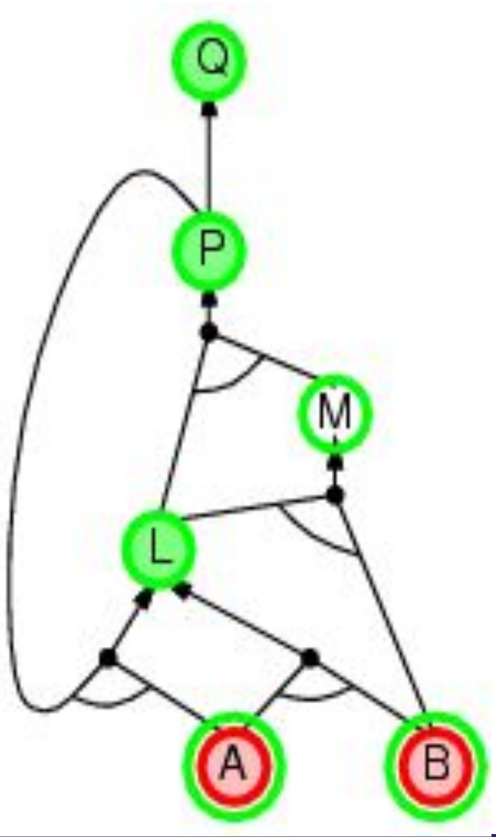
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

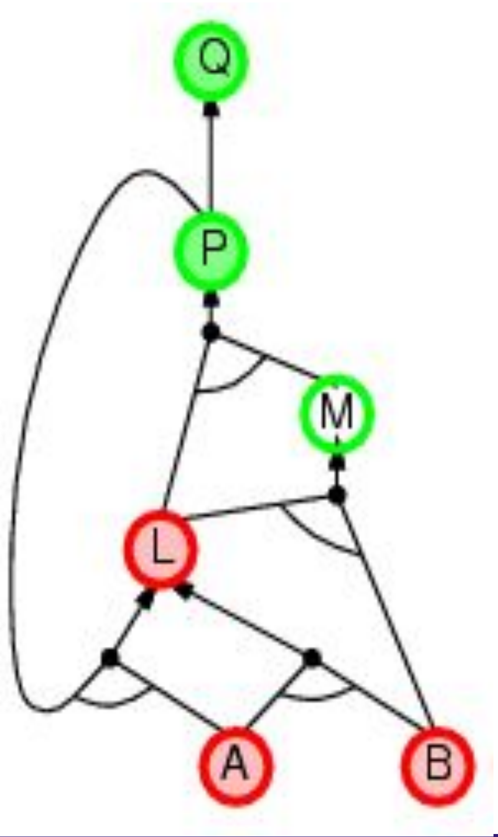
A

B



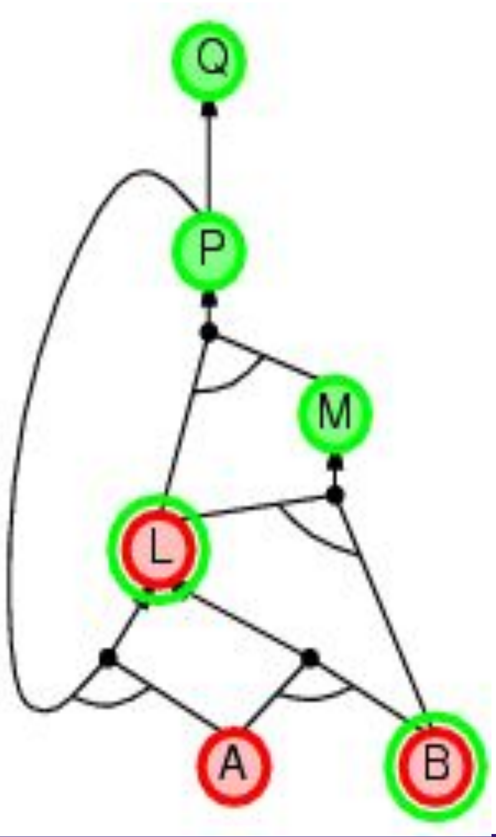
Backward chaining example

$$P \Rightarrow Q$$
$$L \wedge M \Rightarrow P$$
$$B \wedge L \Rightarrow M$$
$$A \wedge P \Rightarrow L$$
$$A \wedge B \Rightarrow L$$
$$A$$
$$B$$



Backward chaining example

$$P \Rightarrow Q$$
$$L \wedge M \Rightarrow P$$
$$B \wedge L \Rightarrow M$$
$$A \wedge P \Rightarrow L$$
$$A \wedge B \Rightarrow L$$
$$A$$
$$B$$



Backward chaining example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

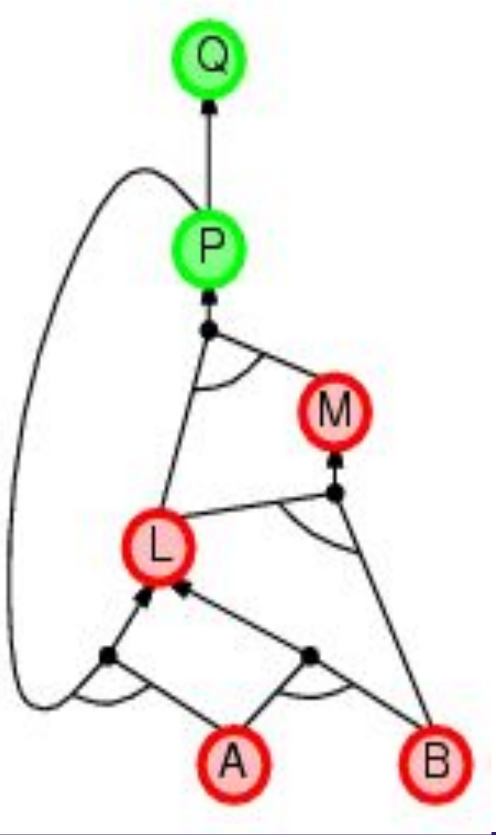
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

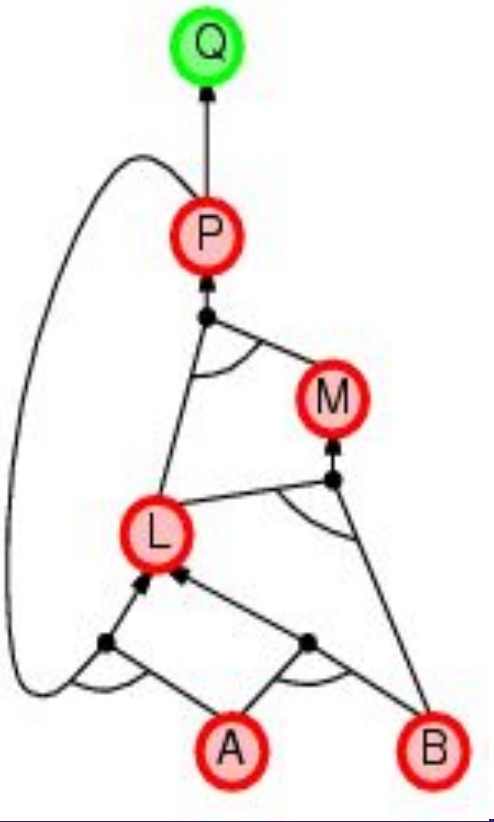
A

B



Backward chaining example

$$P \Rightarrow Q$$
$$L \wedge M \Rightarrow P$$
$$B \wedge L \Rightarrow M$$
$$A \wedge P \Rightarrow L$$
$$A \wedge B \Rightarrow L$$
$$A$$
$$B$$



Backward chaining example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

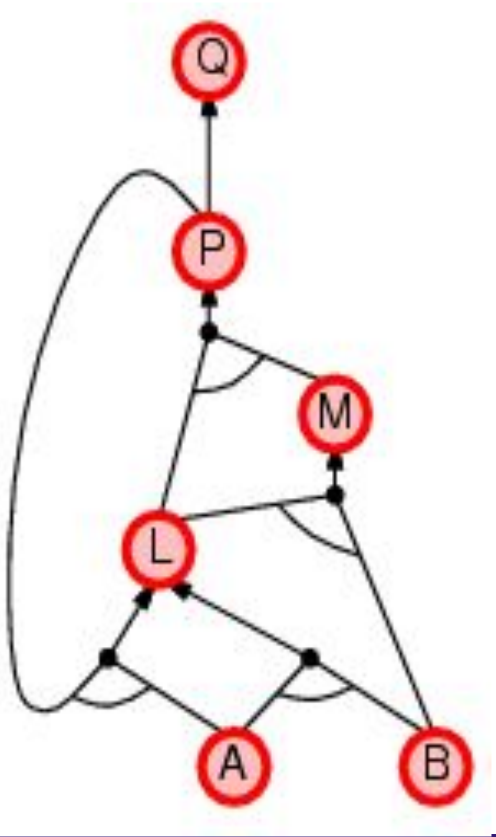
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



Forward vs. backward chaining

FC is **data-driven**, automatic, unconscious processing,

- e.g., object recognition, routine decisions
- May do lots of work that is irrelevant to the goal

BC is **goal-driven**, appropriate for problem-solving,

- e.g., Where are my keys? How do I get into a PhD program?

Inference in First-order Logic

Inference in FOL

Given

$\forall x\ King(x) \land Greedy(x) \Rightarrow Evil(x)$

$King(John)$

$Greedy(John)$

One can infer…

$King(John) \land Greedy(John) \Rightarrow Evil(John)$

$King(Richard) \land Greedy(Richard) \Rightarrow Evil(Richard)$

$Evil(John)$

$Evil(Richard)$

Inference in FOL

Existential Instantiation (in a $\exists$ rule, substitute one symbol, use the rule and discard)

Skolem Constant is a new variable that represents a new inference

Universal Instantiation (in a $\forall$ rule, substitute all symbols)

Existential Instantiation

Variable is replaced by a single new constant symbol

Suppose we have

$$\exists x \; Crown(x) \land OnHead(x, John)$$

Replacing the variable x with the constant C_1, we get

$$Crown(C_1) \land OnHead(C_1, John)$$

C_1 is a **Skolem constant**

Universal Instantiation

Suppose KB:

$\forall x\ King(x) \land Greedy(x) \Rightarrow Evil(x)$

King(John)

Greedy(John)

Brother(Richard, John)

Apply UI using {x/John} and {x/Richard} (substitute x for John or Richard)

$King(John) \land Greedy(John) \Rightarrow Evil(John)$

$King(Richard) \land Greedy(Richard) \Rightarrow Evil(Richard)$

And discard the Universally quantified sentence. We can get the KB to be propositions.

Universal Instantiation

Suppose KB:

$\forall x\ King(x) \land Greedy(x) \Rightarrow Evil(x)$

$King(John)$

$\forall y\ Greedy(y)$

Apply UI using {x/John} and {y/John}

Inference using Modus Ponens

KB is: King(John) , $\forall$ x King(x) $\Rightarrow$ Evil(x)

For atomic sentences p , p' and q, where there is a substitution θ such that SUBST(θ, p') = SUBST(θ, p)

$$\frac{p', \quad p \Rightarrow q}{\text{SUBST}(\theta, q)}$$

p' = King(John)

θ = {x/John}

SUBST(θ, q)

Evil(John)

p = King(x)

q = Evil(x)

Inference using Modus Ponens

For atomic sentences p_i , p_i’ and q, where there is a substitution θ such that

SUBST(θ, p_i’) = SUBST(θ, p_i), for all i

$$\frac{p_1', p_2', \ldots, p_n', \quad (p_1 \land p_2 \land \ldots \land p_n \Rightarrow q)}{\text{SUBST}(\theta, q)}$$

p_1’ = King(John)	p_1 = King(x)
p_2’ = Greedy(y)	p_2 = Greedy(x)
θ = {x/John, y/John}	q = Evil(x)
SUBST(θ, q)	

Unification

UNIFY(p, q) = θ Where SUBST(θ, p) = SUBST(θ, q)

For example:

UNIFY(Knows(John, x), Knows(John, Jane)) = {x/Jane}

UNIFY(Knows(John, x), Knows(y, Bill)) = {y/John, x/Bill}

UNIFY(Knows(John, x), Knows(y, Mother(y))) = {y/John, x/Mother(John)}

We ask ASK(Knows(John, x)) (Whom does John know?)

assignment 1:
when you go to do an ask, there might be a variable in what you're asking
you need to see what the variable can be, what it subsitutes for,
and then find something in the KB that matches what you asked.
matched algorithm is a simpler form of Unification

Inference example

“The Law says that it is a crime for an American to sell weapons to hostile nations. The country Nono, and enemy of America, has some missiles, and all of its missiles were sold to it by Colonel West, who is American”

Prove that Colonel West is a Criminal

Is Colonel West a criminal?

“The Law says that it is a crime for an American to sell weapons to hostile nations. The country Nono, and enemy of America, has some missiles, and all of its missiles were sold to it by Colonel West, who is American”

1: $\forall x,y,z\ American(x) \land Weapon(y) \land Sells(x, y, z) \land Hostile(z) \Rightarrow Criminal(x)$

2: $\exists x\ Owns(Nono, x)$

3: $\exists x\ Missile(x)$

4: $\forall x\ Missile(x) \Rightarrow Weapon(x)$

5: $\forall x\ Missile(x) \land Owns(Nono, x) \Rightarrow Sells(West, x, Nono)$

6: $\forall x\ Enemy(x, America) \Rightarrow Hostile(x)$

7: American(West)

8: Enemy(Nono, America)

“The Law says that it is a crime for an American to sell weapons to hostile nations. The country Nono, an enemy of America, has some missiles , and all of its missiles were sold to it by Colonel West, who is American”

R1: American(x) ∧ Weapon(y) ∧ Sells(x, y, z)

∧ Hostile(z) ⇒ Criminal(x)

R2: Owns(Nono, M1)

- Nono has some missiles

R3: Missile(M1)

R4: Missile(x) ⇒ Weapon(x)

- A missile is a weapon

R5: Missile(x) ∧ Owns(Nono, x) ⇒

Sells(West, x, Nono)

- All missiles sold by west

R6: Enemy(x, America) ⇒ Hostile(x)

- Enemies of America are hostile

R7: American(West)

- West is american

R8: Enemy(Nono, America)

Forward Chaining Iterations

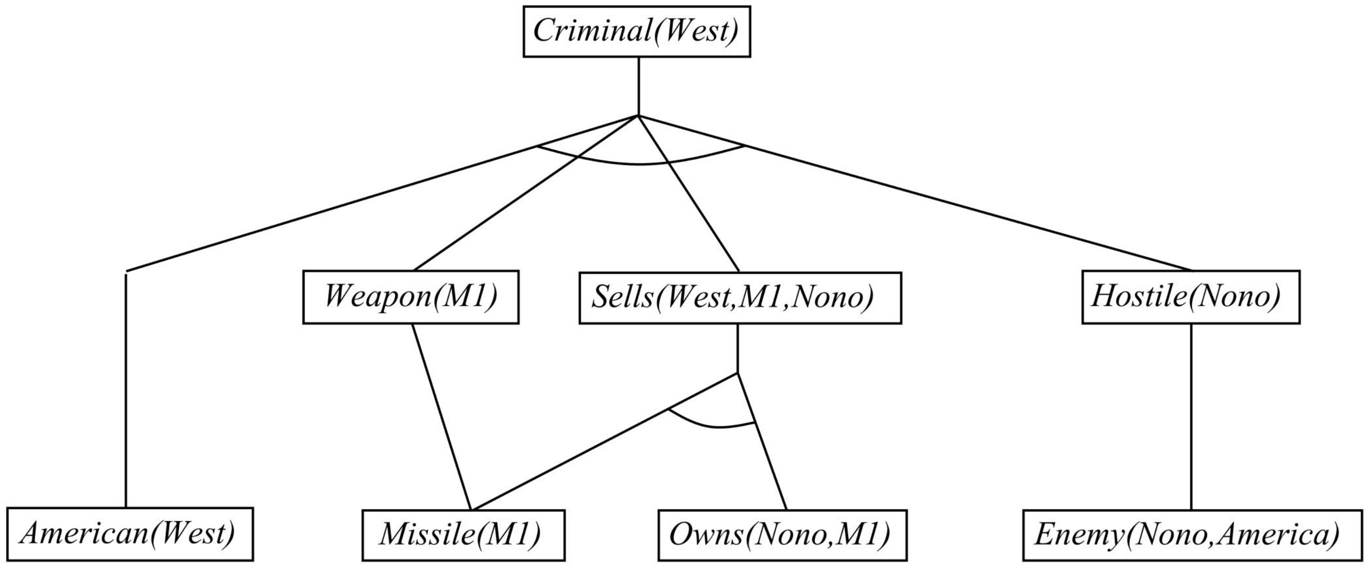
Iteration 1:

- R5 satisfied with {x/M1} so we add
  - R9: Sells(West, M1, Nono)
- R4 satisfied with {x/M1} so we add
  - R10: Weapon(M1)
- R6 satisfied with {x/Nono} so we add
  - R11: Hostile(Nono)

Iteration 2:

- R1 is satisfied with {x/West, y/M1, z/Nono}
  - Criminal(West) is added.

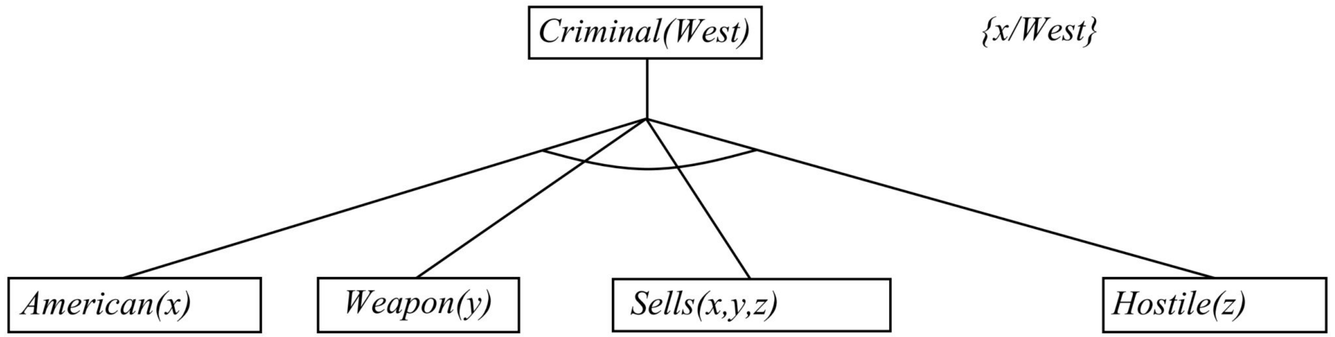
Forward chaining



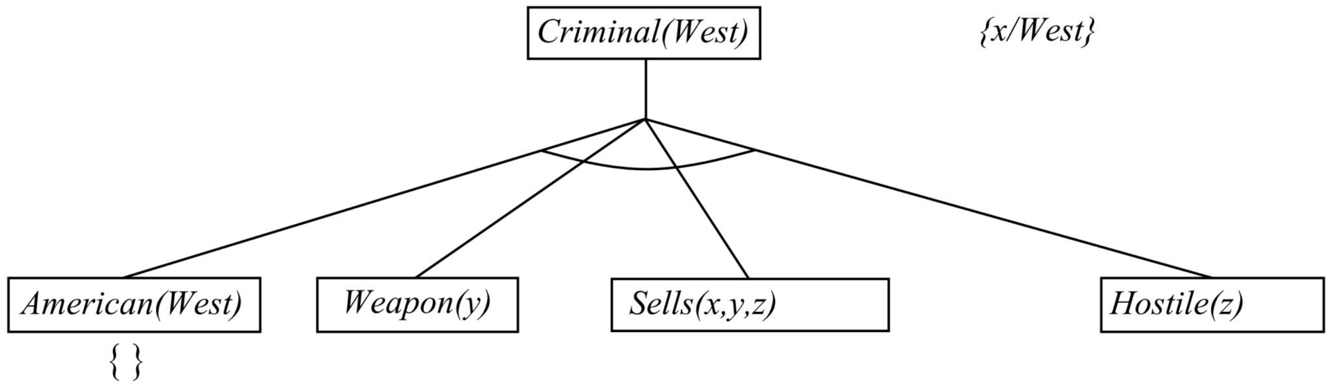
Backward Chaining

$$\boxed{Criminal(West)}$$

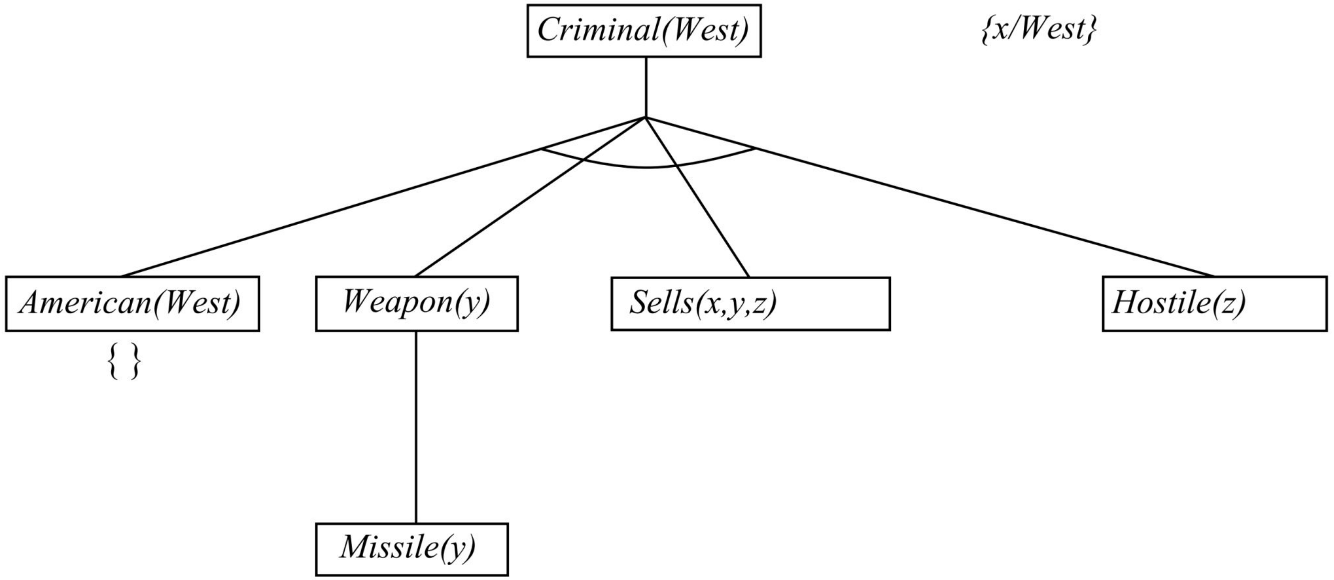
Backward Chaining



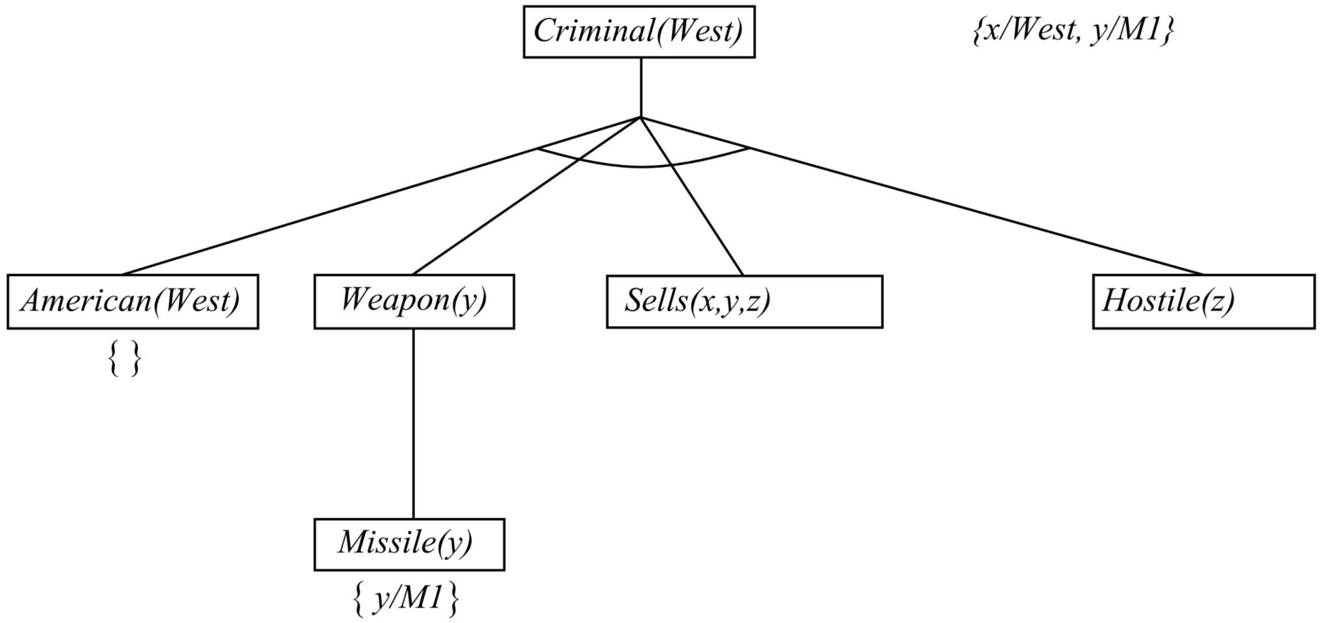
Backward Chaining



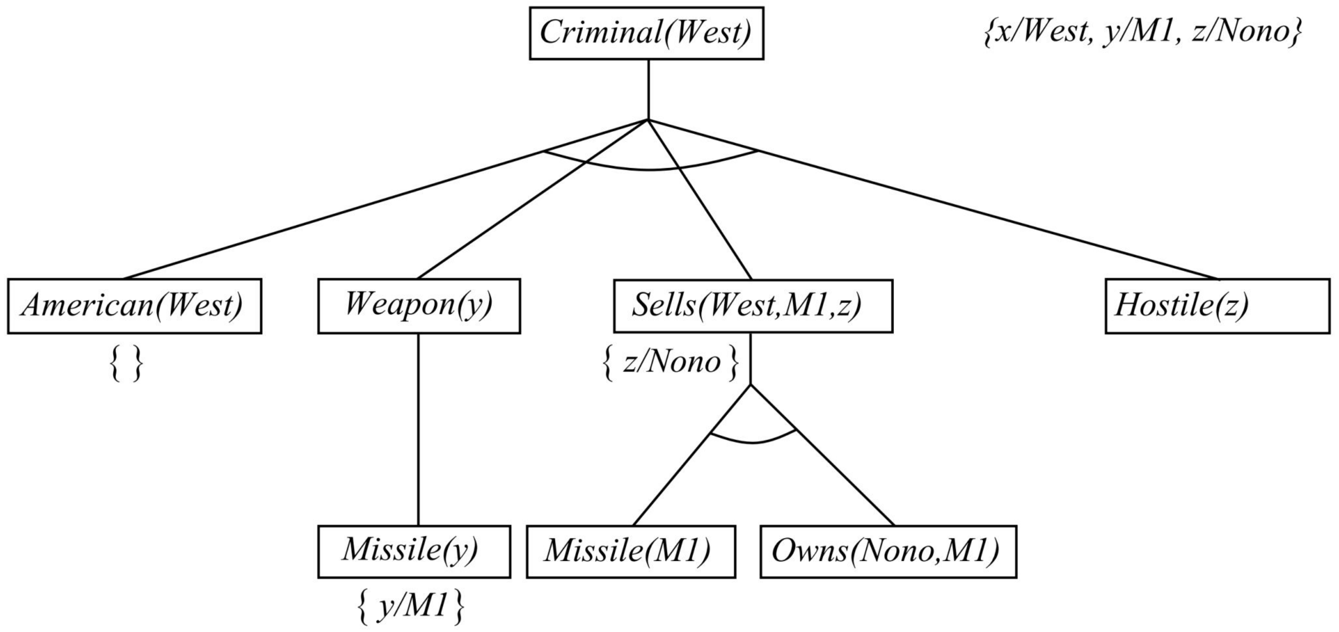
Backward Chaining



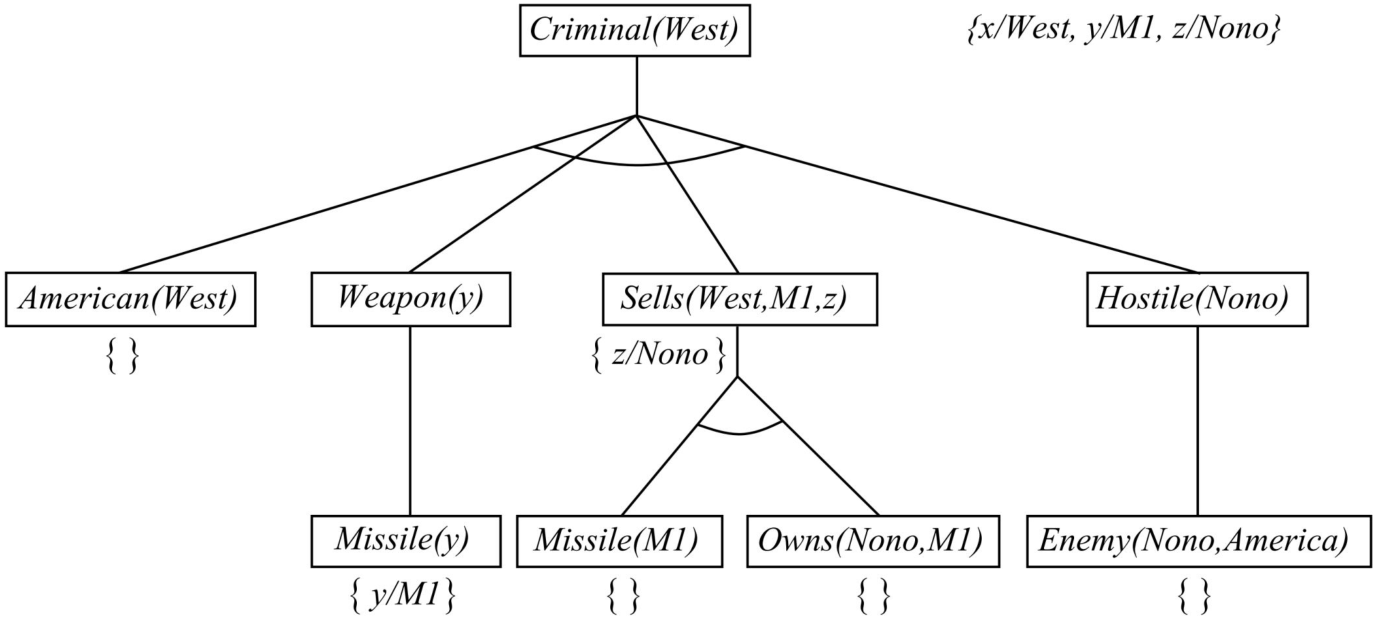
Backward Chaining



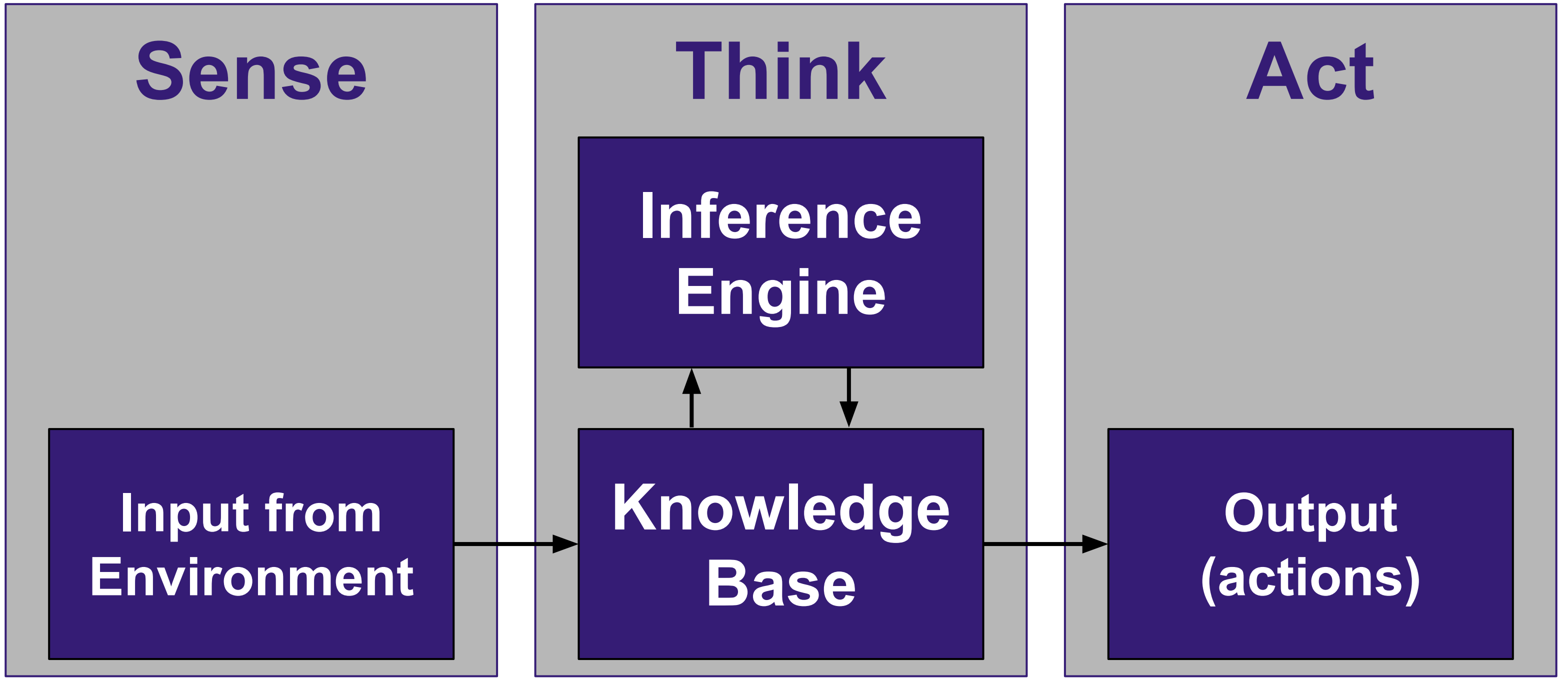
Backward Chaining



Backward Chaining



Knowledge-based agent



KB Interface

Assert

- Add a new fact or rule to the KB

Retract

- Remove an asserted fact from the KB

Ask

- Is a fact in the KB
- Ask with variables, get bindings

CLOSED-WORLD ASSUMPTION

- Things that are true, are known to be true
  - In the KB
- Anything not known to be true, assumed to be false
  - Not in the KB
- Only reason about truths
  - In the KB

Inference Engine

When a fact or rule is added to the KB

Use forward-chaining to infer new facts

Incremental approach allows us simplify how we match facts to rule

- Infer new rules

Infer New Facts

Start KB with a couple of facts

```
(barks Hershey)
(isa Dog Mammal)
(isa Mammal Animal)
```

Then add a rule

```
((barks ?d)) -> (inst ?d Dog)
```

Forward-chain to infer new facts

- Unify (barks ?d) with (barks Hershey)
  {?d/Hershey}
- Make substitutions
  ((barks Hershey)) -> (inst Hershey Dog)
- Add new fact to KB

More Complex Rules

If rules has multiple conditions, need to satisfy all before inferring new fact

Start KB with a couple of facts

```
(isa Dog Mammal)

(isa Mammal Animal)
```

Then add a rule

```
((isa?x ?y) (isa ?y ?z)) -> (isa ?x ?z)
```

Forward-chain to infer new rule

Infer New Rules

Start with first sentence in left-hand side (lhs) of rule

    (isa ?x ?y)

Try to unify it with each fact in KB

    (isa Dog Mammal)

When a substitution is found

- Make substitutions

  ((isa Dog Mammal) (isa Mammal ?z)) -> (isa Dog ?z)

- Create new rule with rest of rule

  ((isa Mammal ?z)) -> (isa Dog ?z)

- Add new rule to KB

Repeat

Added new rule

    ((isa Mammal ?z)) -> (isa Dog ?z)

Unify first sentence of LHS with facts in the KB

    (isa Mammal Animal)

    {?z/Animal}

Make substitutions

    ((isa Mammal Animal)) -> (isa Dog Animal)

Add new fact to KB

    (isa Dog Animal)

FC Algorithm For New Fact

```
Forward_chain(fact):
  For each rule in KB
    Get first statement in lhs of rule
    Unify fact with first statement
    If substitution found
      If lhs is only one statement
        Create new fact using the substitution on rhs
        Add new fact to kb
      Else
        Create new rule with rest of lhs and rhs
        Add new rule to kb
```

FC Algorithm For New Rule

```
Forward_chain(rule):
  For each fact in KB
    Get first statement in lhs of rule
    Unify fact with first statement
    If substitution found
      If lhs is only one statement
        Create new fact using the substitution on rhs
        Add new fact to kb
      Else
        Create new rule with rest of lhs and rhs
        Add new rule to kb
```

Truth Maintenance

Bookkeeping to track how facts came to be believed

Track which facts and rules support other facts and rules

Asserted facts and rules are assumed to be true

- No internal support for them

Everything else must have support

If some fact is inferred, then it is supported by the rule and fact used to infer the fact

Similarly, if some rule is inferred, then it is supported by the rule and fact used to infer the rule