

CS 2740 Knowledge Representation
Lecture 6

Propositional logic:
Horn clauses

Milos Hauskrecht
milos@cs.pitt.edu
5329 Sennott Square

Administration

- **Homework assignment 2**
 - Propositional logic exercises
 - Programming a resolution solver

Logical inference problem

Logical inference problem:

- **Given:**
 - a knowledge base KB (a set of sentences) and
 - a sentence α (called **a theorem**),
- **Does a KB semantically entail α ?** $KB \models \alpha$?

In other words: In all interpretations in which sentences in the KB are true, is also α true?

Solving logical inference problem

In the following:

How to design the procedure that answers:

$$KB \models \alpha ?$$

Three approaches:

- **Truth-table approach**
- **Inference rules**
- **Conversion to the inverse SAT problem**
 - **Resolution-refutation**

KB in restricted forms

If the sentences in the KB are restricted to some special forms
some of the sound inference rules may become complete

Example:

- **Horn form (Horn normal form)**

$$(A \vee \neg B) \wedge (\neg A \vee \neg C \vee D)$$

Can be written also as: $(B \Rightarrow A) \wedge ((A \wedge C) \Rightarrow D)$

- **Two inference rules that are sound and complete with respect to propositional symbols for KBs in the Horn normal form:**
 - **Resolution (positive unit resolution)**
 - **Modus ponens**

KB in Horn form

- **Horn form:** a clause with **at most one positive literal**

$$(A \vee \neg B) \wedge (\neg A \vee \neg C \vee D)$$

- **Not all sentences in propositional logic can be converted into the Horn form**

- **KB in Horn normal form:**

– Three types of propositional statements:

- **Rules** $(\neg B_1 \vee \neg B_2 \vee \dots \neg B_k \vee A)$

$$(\neg(B_1 \wedge B_2 \wedge \dots B_k) \vee A) \quad (B_1 \wedge B_2 \wedge \dots B_k \Rightarrow A)$$

- **Facts** B

- **Integrity constraints** $(\neg B_1 \vee \neg B_2 \vee \dots \neg B_k)$

$$(B_1 \wedge B_2 \wedge \dots B_k \Rightarrow \text{False})$$

KB in Horn form

- **Horn form:** a clause with **at most one positive literal**

$$(A \vee \neg B) \wedge (\neg A \vee \neg C \vee D)$$

- **Not all sentences in propositional logic can be converted into the Horn form**

- **KB in Horn normal form:**

– Three types of propositional statements:

- **Rules** $(\neg B_1 \vee \neg B_2 \vee \dots \neg B_k \vee A)$

$$(\neg(B_1 \wedge B_2 \wedge \dots B_k) \vee A) \quad (B_1 \wedge B_2 \wedge \dots B_k \Rightarrow A)$$

- **Facts** B

- **Integrity constraints** $(\neg B_1 \vee \neg B_2 \vee \dots \neg B_k)$

$$(B_1 \wedge B_2 \wedge \dots B_k \Rightarrow \text{False})$$

KB in Horn form

- **Modus ponens:**

$$\frac{B \Rightarrow A, \quad B}{A}$$

– More general version of the rule:

$$\frac{(B_1 \wedge B_2 \wedge \dots B_k \Rightarrow A), B_1, B_2, \dots B_k}{A}$$

– Modus ponens is **sound and complete with respect to propositional symbols** for the KBs in the Horn normal form

– We assume only logical inference problems for which the theorem α is a **propositional symbol**:

- **Note:** no negation of a propositional symbol is allowed

KB in Horn form

- **Inferences:**

- **Resolution rule:**

$$\frac{A \vee B, \quad \neg B \vee C}{A \vee C}$$

- Resolution is **sound and complete with respect to propositional symbols** for the KBs in the Horn normal form

Complexity of inferences for KBs in HNF

Question:

How efficient the inferences in the HNF wrt propositional symbols can be?

Answer:

Procedures linear in the size of the KB in the Horn form exist.

- Size of a clause: the number of literals it contains.
- Size of the KB in the HNF: the sum of the sizes of its elements.

Example:

$A, B, (A \wedge B \Rightarrow C), (C \Rightarrow D), (C \Rightarrow E), (E \wedge F \Rightarrow G)$

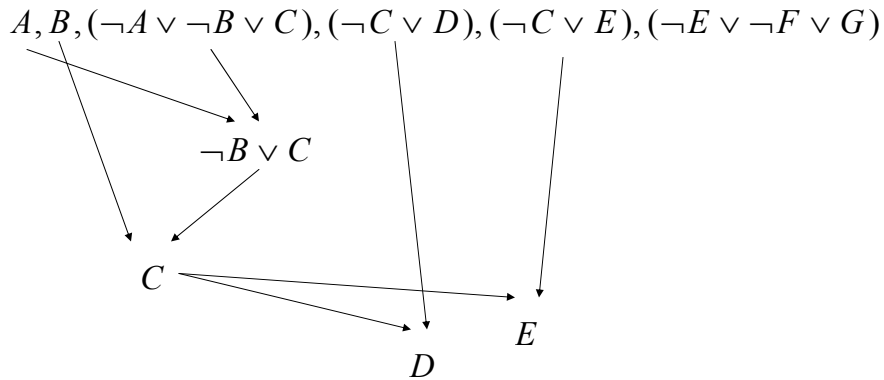
or

$A, B, (\neg A \vee \neg B \vee C), (\neg C \vee D), (\neg C \vee E), (\neg E \vee \neg F \vee G)$

The size is: 12

Complexity of inferences for KBs in HNF

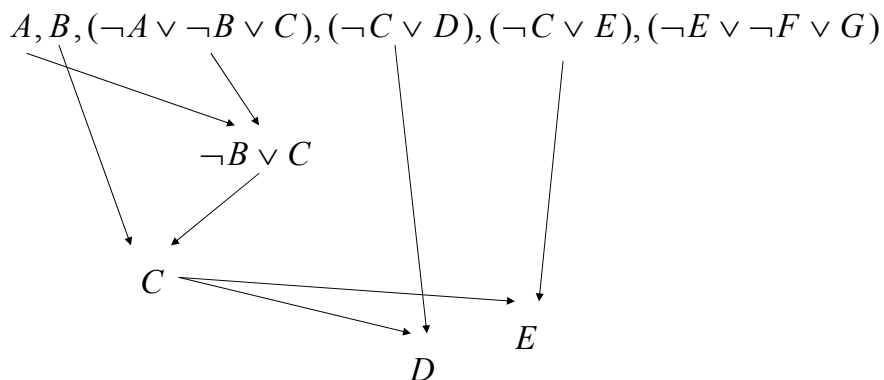
How to do the inference? If the HNF (is in clausal form) we can apply resolution.



Complexity of inferences for KBs in HNF

Features:

- Every resolution is a **positive unit resolution**; that is, a resolution in which **one clause is a positive unit clause** (i.e., a proposition symbol).

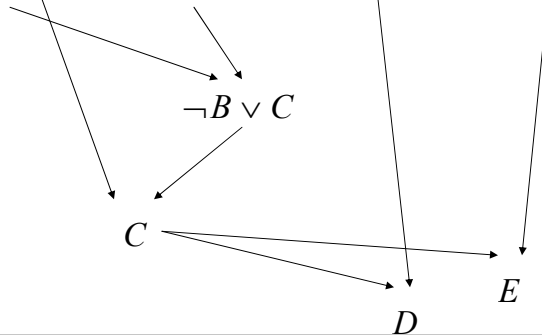


Complexity of inferences for KBs in HNF

Features:

- At each resolution, the input clause which is not a unit clause is a logical consequence of the result of the resolution. (Thus, the input clause may be deleted upon completion of the resolution operation.)

$A, B, (\neg A \vee \neg B \vee C), (\neg C \vee D), (\neg C \vee E), (\neg E \vee \neg F \vee G)$

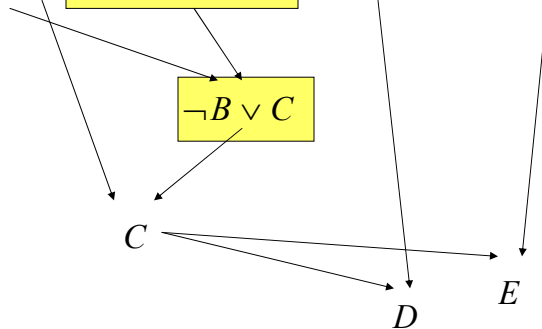


Complexity of inferences for KBs in HNF

Features:

- At each resolution, the input clause which is not a unit clause is a logical consequence of the result of the resolution. (Thus, the input clause may be deleted upon completion of the resolution operation.)

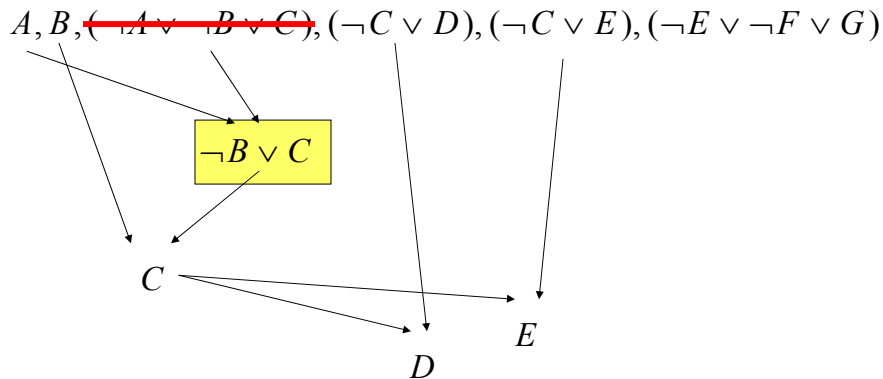
$A, B, (\neg A \vee \neg B \vee C), (\neg C \vee D), (\neg C \vee E), (\neg E \vee \neg F \vee G)$



Complexity of inferences for KBs in HNF

Features:

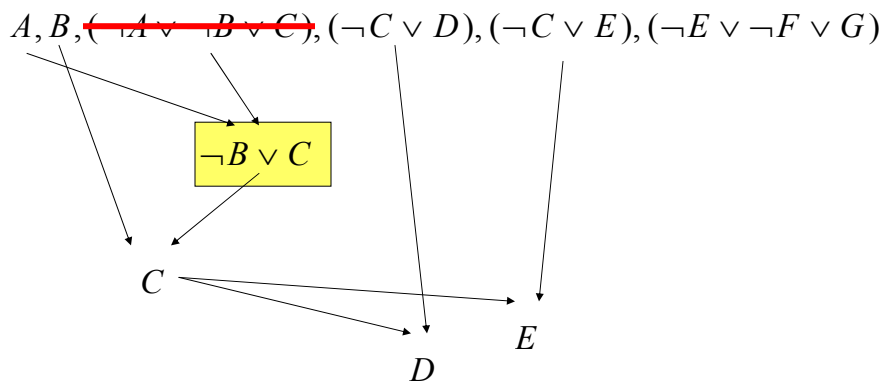
- Following this deletion, the size of the KB (the sum of the lengths of the remaining clauses) is one less than it was before the operation.)



Complexity of inferences for KBs in HNF

Features:

- If n is the size of the KB, then at most n positive unit resolutions may be performed on it.



Complexity of inferences for KBs in HNF

A linear time resolution algorithm:

- The number of positive unit resolutions is limited to the size of the formula (n)
- But to assure overall linear time we need to access each proposition in a constant time:
- Data structures indexed by proposition names may be accessed in constant time. (This is possible if the proposition names are number in a range (e.g., $1..n$), so that array lookup is the access operation.
- If propositions are accessed by name, then a symbol table is necessary, and the algorithm will run in time $O(n \cdot \log(n))$.

Forward and backward chaining

Two inference procedures based on **modus ponens** for **Horn KBs**:

- **Forward chaining**
Idea: Whenever the premises of a rule are satisfied, infer the conclusion. Continue with rules that became satisfied.
- **Backward chaining (goal reduction)**
Idea: To prove the fact that appears in the conclusion of a rule prove the premises of the rule. Continue recursively.

Both procedures are **complete for KBs in the Horn form !!!**

Forward chaining example

- **Forward chaining**

Idea: Whenever the premises of a rule are satisfied, infer the conclusion. Continue with rules that became satisfied.

Assume the KB with the following rules and facts:

KB: R1: $A \wedge B \Rightarrow C$

R2: $C \wedge D \Rightarrow E$

R3: $C \wedge F \Rightarrow G$

F1: A

F2: B

F3: D

Theorem: E ?

Forward chaining example

Theorem: E

KB: R1: $A \wedge B \Rightarrow C$

R2: $C \wedge D \Rightarrow E$

R3: $C \wedge F \Rightarrow G$

F1: A

F2: B

F3: D

Forward chaining example

Theorem: E

KB: R1: $A \wedge B \Rightarrow C$

R2: $C \wedge D \Rightarrow E$

R3: $C \wedge F \Rightarrow G$

F1: A

F2: B

F3: D

Rule R1 is satisfied.

F4: C

Forward chaining example

Theorem: E

KB: R1: $A \wedge B \Rightarrow C$

R2: $C \wedge D \Rightarrow E$

R3: $C \wedge F \Rightarrow G$

F1: A

F2: B

F3: D

Rule R1 is satisfied.

F4: C

Rule R2 is satisfied.

F5: E



Forward chaining

- Efficient implementation: linear in the size of the KB
- **Example:**

```

$$P \Rightarrow Q$$

$$L \wedge M \Rightarrow P$$

$$B \wedge L \Rightarrow M$$

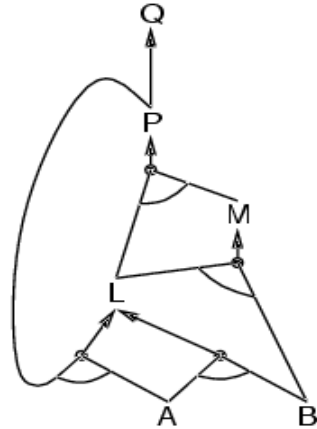
$$A \wedge P \Rightarrow L$$

$$A \wedge B \Rightarrow L$$

$$A$$

$$B$$

```



Forward chaining

- Runs in time linear in the number of literals in the Horn formulae

```
function PL-FC-ENTAILS?(KB, q) returns true or false
  local variables: count, a table, indexed by clause, initially the number of premises
                  inferred, a table, indexed by symbol, each entry initially false
                  agenda, a list of symbols, initially the symbols known to be true

  while agenda is not empty do
    p ← POP(agenda)
    unless inferred[p] do
      inferred[p] ← true
      for each Horn clause c in whose premise p appears do
        decrement count[c]
        if count[c] = 0 then do
          if HEAD[c] = q then return true
          PUSH(HEAD[c], agenda)
  return false
```

Forward chaining

**

$$P \Rightarrow Q$$

$$L \wedge M \Rightarrow P$$

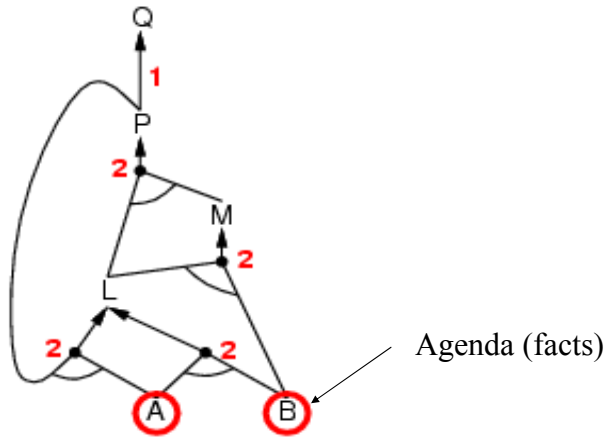
$$B \wedge L \Rightarrow M$$

$$A \wedge P \Rightarrow L$$

$$A \wedge B \Rightarrow L$$

A

B



Forward chaining

•

$$P \Rightarrow Q$$

$$L \wedge M \Rightarrow P$$

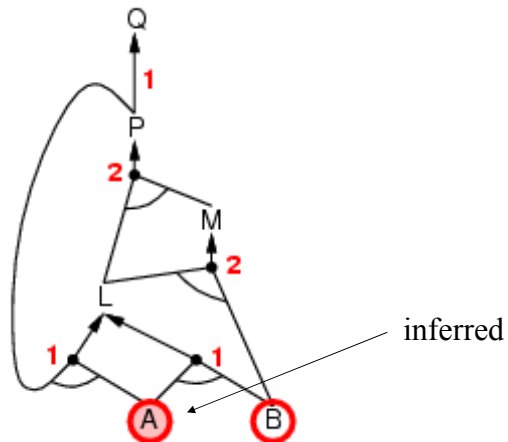
$$B \wedge L \Rightarrow M$$

$$A \wedge P \Rightarrow L$$

$$A \wedge B \Rightarrow L$$

A

B



Forward chaining

•

$$P \Rightarrow Q$$

$$L \wedge M \Rightarrow P$$

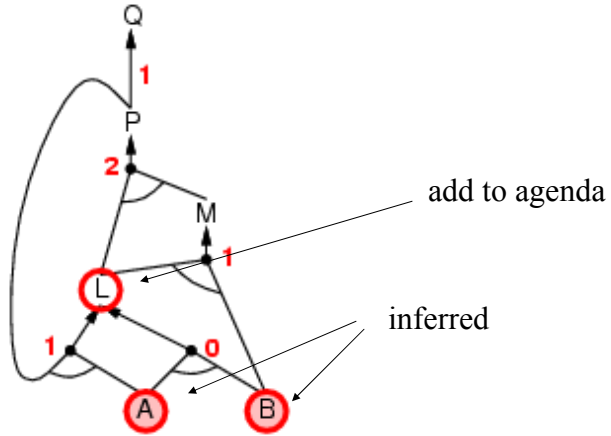
$$B \wedge L \Rightarrow M$$

$$A \wedge P \Rightarrow L$$

$$A \wedge B \Rightarrow L$$

A

B



Forward chaining

•

$$P \Rightarrow Q$$

$$L \wedge M \Rightarrow P$$

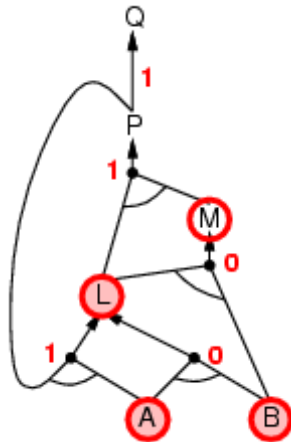
$$B \wedge L \Rightarrow M$$

$$A \wedge P \Rightarrow L$$

$$A \wedge B \Rightarrow L$$

A

B



Forward chaining

•

$$P \Rightarrow Q$$

$$L \wedge M \Rightarrow P$$

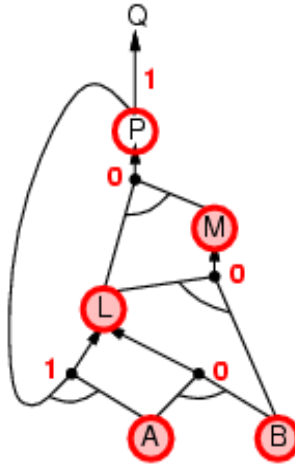
$$B \wedge L \Rightarrow M$$

$$A \wedge P \Rightarrow L$$

$$A \wedge B \Rightarrow L$$

A

B



Forward chaining

•

$$P \Rightarrow Q$$

$$L \wedge M \Rightarrow P$$

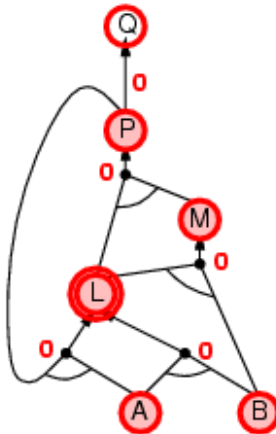
$$B \wedge L \Rightarrow M$$

$$A \wedge P \Rightarrow L$$

$$A \wedge B \Rightarrow L$$

A

B



Forward chaining

•

$$P \Rightarrow Q$$

$$L \wedge M \Rightarrow P$$

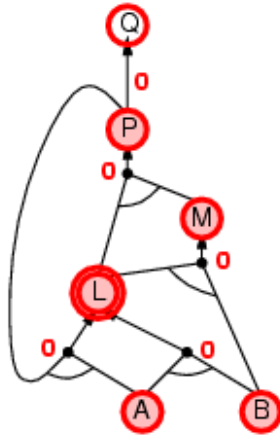
$$B \wedge L \Rightarrow M$$

$$A \wedge P \Rightarrow L$$

$$A \wedge B \Rightarrow L$$

A

B



Backward chaining example

Problem: KB: R1: $A \wedge B \Rightarrow C$

R2: $C \wedge D \Rightarrow E$

R3: $C \wedge F \Rightarrow G$

F1: A

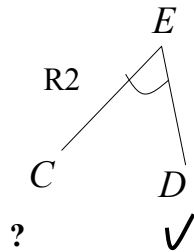
F2: B

F3: D

Theorem: E

- Goal: prove the theorem, try to be more theorem driven

Backward chaining example



KB: R1: $A \wedge B \Rightarrow C$

R2: $C \wedge D \Rightarrow E$

R3: $C \wedge F \Rightarrow G$

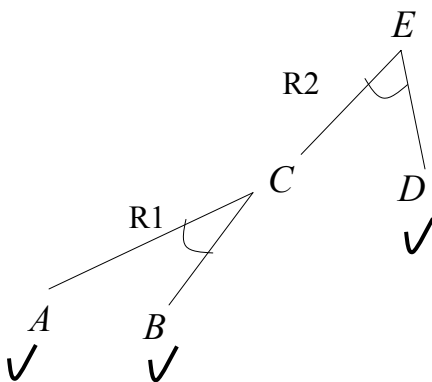
F1: A

F2: B

F3: D

- Backward chaining tries to prove a theorem
Procedure idea:
 - Checks if the theorem is true
 - If not, find the rule with the theorem in its conclusion and try to prove its premises
- Theorem: E

Backward chaining example



KB: R1: $A \wedge B \Rightarrow C$

R2: $C \wedge D \Rightarrow E$

R3: $C \wedge F \Rightarrow G$

F1: A

F2: B

F3: D

- Backward chaining is theorem driven

Backward chaining

•

$$P \Rightarrow Q \quad \leftarrow$$

$$L \wedge M \Rightarrow P$$

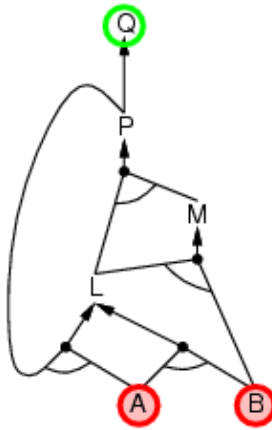
$$B \wedge L \Rightarrow M$$

$$A \wedge P \Rightarrow L$$

$$A \wedge B \Rightarrow L$$

$$A \quad \leftarrow$$

$$B \quad \leftarrow$$



Backward chaining

•

$$P \Rightarrow Q \quad \leftarrow$$

$$L \wedge M \Rightarrow P \quad \leftarrow$$

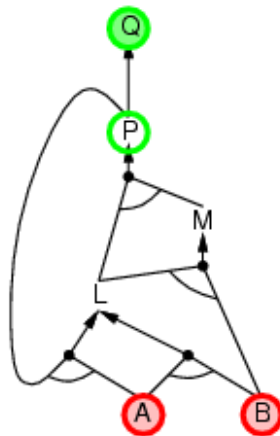
$$B \wedge L \Rightarrow M$$

$$A \wedge P \Rightarrow L$$

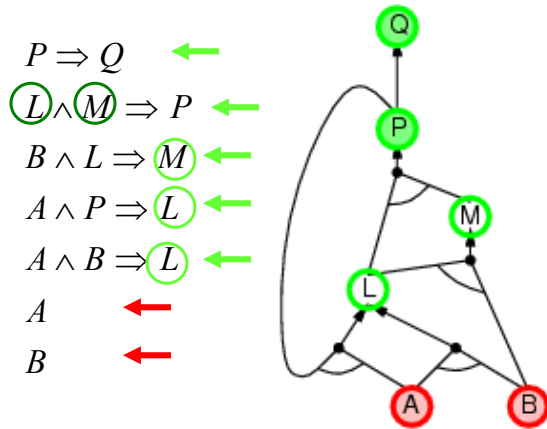
$$A \wedge B \Rightarrow L$$

$$A \quad \leftarrow$$

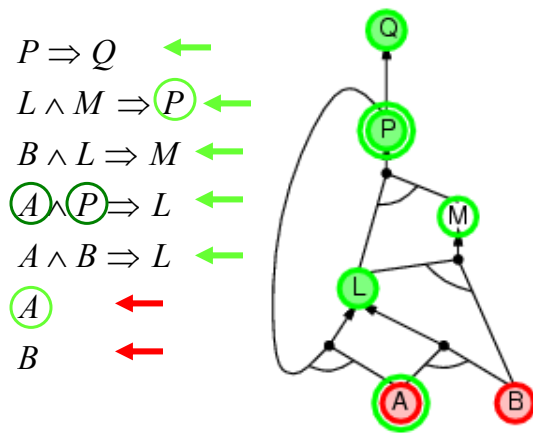
$$B \quad \leftarrow$$



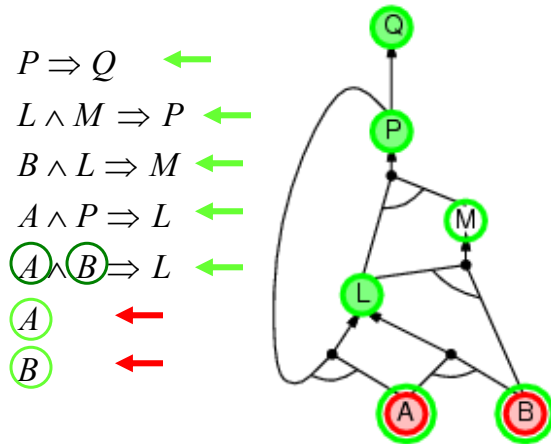
Backward chaining



Backward chaining



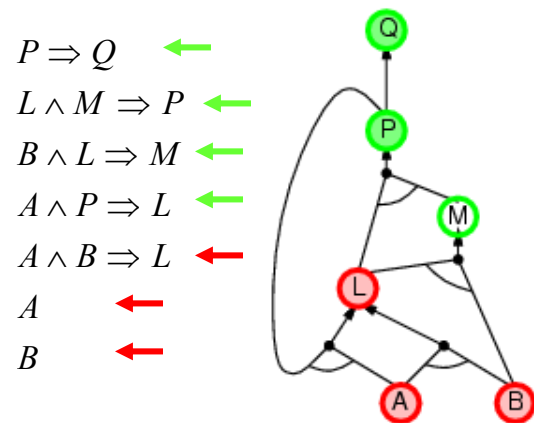
Backward chaining



CS 2740 Knowledge Representation

M. Hauskrecht

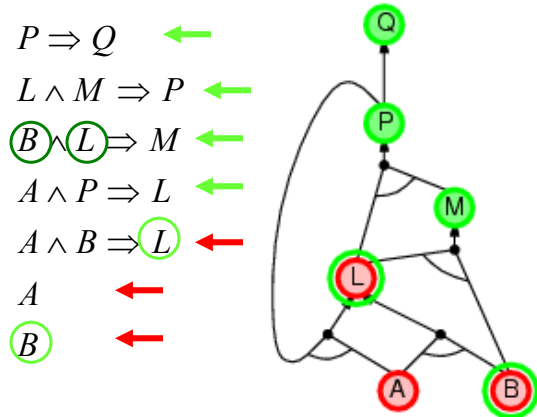
Backward chaining



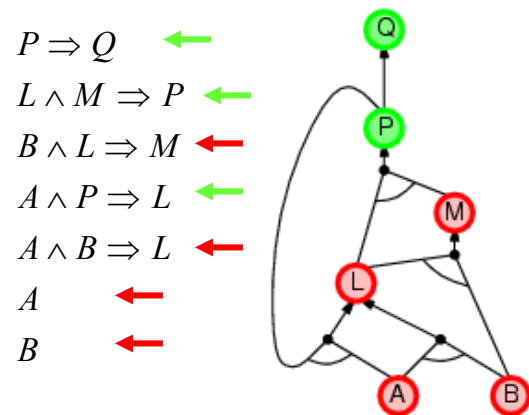
CS 2740 Knowledge Representation

M. Hauskrecht

Backward chaining



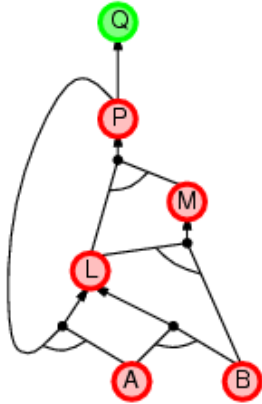
Backward chaining



Backward chaining

•

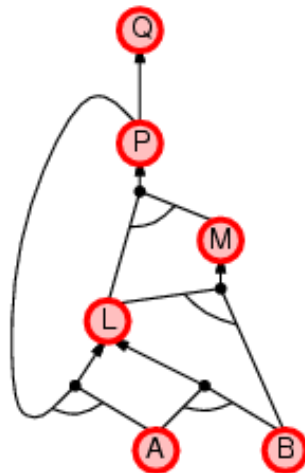
$P \Rightarrow Q$ ←
 $L \wedge M \Rightarrow P$ ←
 $B \wedge L \Rightarrow M$ ←
 $A \wedge P \Rightarrow L$ ←
 $A \wedge B \Rightarrow L$ ←
 A ←
 B ←



Backward chaining

•

$P \Rightarrow Q$ ←
 $L \wedge M \Rightarrow P$ ←
 $B \wedge L \Rightarrow M$ ←
 $A \wedge P \Rightarrow L$ ←
 $A \wedge B \Rightarrow L$ ←
 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?
- Complexity of BC can be **much less** than **linear in size of KB**

KB agents based on propositional logic

- Propositional logic allows us to build **knowledge-based agents** capable of answering queries about the world by inferring new facts from the known ones
- **Example:** an agent for diagnosis of a bacterial disease

Facts: The stain of the organism is gram-positive
The growth conformation of the organism is chains

Rules: **(If)** The stain of the organism is gram-positive \wedge
 The morphology of the organism is coccus \wedge
 The growth conformation of the organism is chains
(Then) \Rightarrow The identity of the organism is streptococcus

Limitations of the HNF

The HNF works with propositional symbols:

- Only non-negated propositional symbols may occur in the premise and the conclusion of the rule
- Only non-negated propositions can be used as facts

Dilemma:

- how to represent the negation so that we can express sentences like:
 - If it is not raining we will go swimming

Solution 1: make an explicit proposition for Not_Raining

Solution 2: negation as the failure

- The negation of the propositional symbol will become true if we fail to prove it is true

Negation as the failure

The negation of the propositional symbol will become true if we fail to prove it is true

Caveats:

not $Q \rightarrow P$

In terms of logic we have possibilities:

- $P = \text{True}$,
- or $Q = \text{True}$

Problem is if we use the failure to prove idea: Q is never on the right hand side of the rule so how we can prove it is not true

Solution: stable models

- Each atom (proposition) has a rule

Example. Animal identification system.

- I1. If the animal has hair then it is a mammal
- I2. If the animal gives milk then it is a mammal
- I3. If the animal has feathers then it is a bird
- I4. If the animal flies and it lays eggs then it is a bird
- I5. If the animal is a mammal and it eats meat then it is a carnivore
- I6. If the animal is a mammal and it has pointed teeth and it has claws and its eyes point forward then it is a carnivore
- I7. If the animal is a mammal and it has hoofs then it is an ungulate
- I8. If the animal is a mammal and it chews cud then it is an ungulate and it is even-toed
- I9. If the animal is a carnivore and it has a tawny color and it has dark spots then it is a cheetah
- I10. If the animal is a carnivore and it has a tawny color and it has black strips then it is a tiger
- I11. If the animal is an ungulate and it has long legs and it has a long neck and it has a tawny color and it has dark spots then it is a giraffe
- I12. If the animal is an ungulate and it has a white color and it has black stripes then it is a zebra
- I13. If the animal is a bird and it does not fly and it has long legs and it has a long neck and it is black and white then it is an ostrich,
- I14. If the animal is a bird and it does not fly and it swims and it is black and white then it is a penguin
- I15. If the animal is a bird and it is a good flyer then it is an albatross.