

Artificial Intelligence

Lecture 5: Knowledge and Reasoning

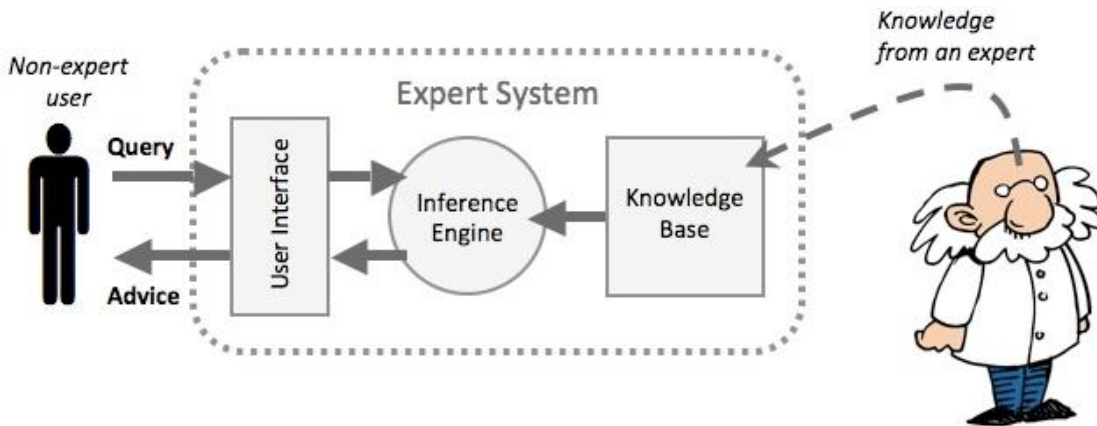
Xiaojin Gong

2021-03-29

Credits: AI Courses in Berkeley & JHU

Outline

- Knowledge-based Agent
- Propositional Logic
- First-Order Logic
- Knowledge Representation



Expert system



Knowledge graph

Knowledge-based Agents

- A knowledge-based agent is composed of a knowledge base and an inference mechanism.
- Knowledge base: a set of sentences in a knowledge representation language that is defined by its syntax and semantics.
 - Propositional logic
 - First-order logic
- Inference: derive new sentences from old.

Knowledge-based Agents

```
function KB-AGENT(percept) returns an action  
  persistent: KB, a knowledge base  
             t, a counter, initially 0, indicating time  
  
  TELL(KB, MAKE-PERCEPT-SENTENCE(percept, t))  
  action ← ASK(KB, MAKE-ACTION-QUERY(t))  
  TELL(KB, MAKE-ACTION-SENTENCE(action, t))  
  t ← t + 1  
  return action
```

- The agent must be able to
 - represent states, actions, etc.
 - incorporate new percepts
 - update internal representations of the world
 - deduce hidden properties of the world
 - deduce appropriate actions

Example: The Wumpus World

■ Performance measure

- gold +1000, death -1000
- -1 per step, -10 for using the arrow

■ Environment

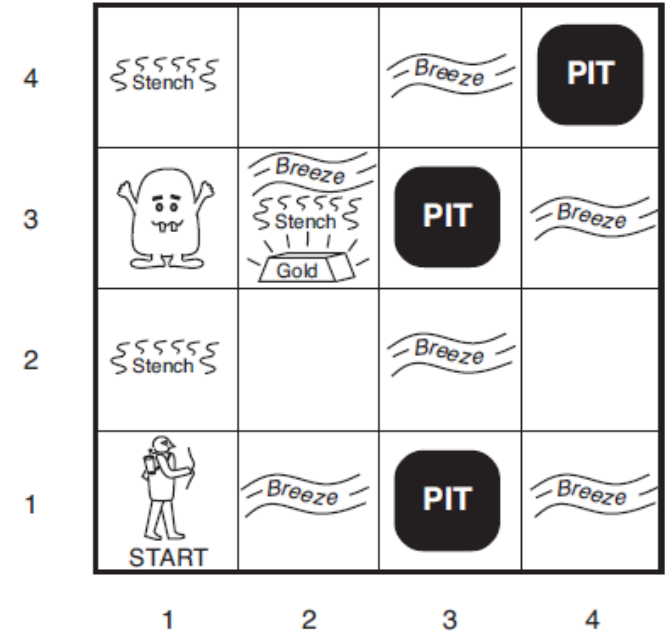
- squares adjacent to wumpus are stench
- squares adjacent to pit are breezy
- glitter iff gold is in the same square
- shooting kills wumpus if you are facing it
- shooting uses up the only arrow
- grabbing picks up gold if in same square
- releasing drops the gold in same square

■ Actuators

- Left turn, Right turn, Forward, Grab, Shoot, Climb

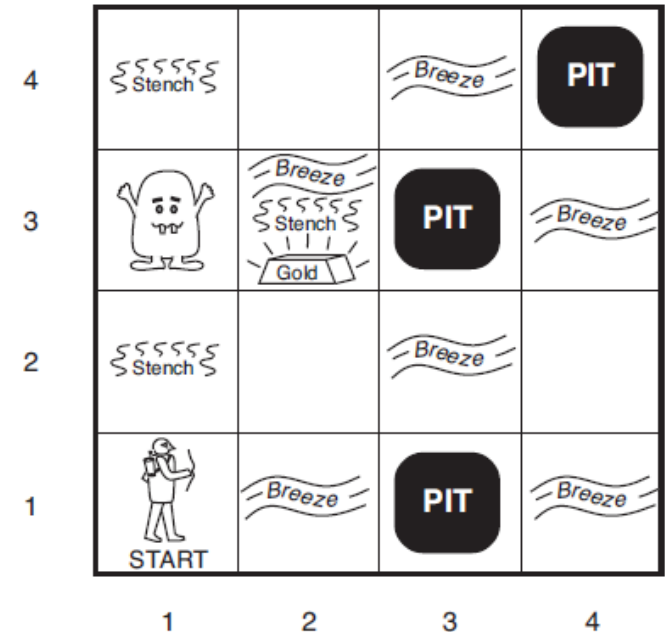
■ Sensors

- Stench, Breeze, Glitter, Bump, Scream



Example: The Wumpus World

- Observable?
 - No—only local perception
- Deterministic?
 - Yes—outcomes exactly specified
- Episodic?
 - No—sequential at the level of actions
- Static?
 - Yes—Wumpus and Pits do not move
- Discrete?
 - Yes
- Single-agent?
 - Yes—Wumpus is essentially a natural feature



Example: The Wumpus World

- A knowledge-based wumpus agent exploring the environment
 - Logical reasoning

1,4	2,4	3,4	4,4
1,3	2,3	3,3	4,3
1,2	2,2	3,2	4,2
1,1	2,1	3,1	4,1

OK

OK

OK

A = Agent
B = Breeze
G = Glitter, Gold
OK = Safe square
P = Pit
S = Stench
V = Visited
W = Wumpus

[None, None, None, None, None].

- Stench, Breeze, Glitter, Bump, Scream

Example: The Wumpus World

- A knowledge-based wumpus agent exploring the environment
 - Logical reasoning

1,4	2,4	3,4	4,4
1,3	2,3	3,3	4,3
1,2 OK	2,2 P?	3,2	4,2
1,1 V OK	2,1 A B OK	3,1 P?	4,1

A = Agent
B = Breeze
G = Glitter, Gold
OK = Safe square
P = Pit
S = Stench
V = Visited
W = Wumpus

[None, Breeze, None, None, None]

Example: The Wumpus World

- A knowledge-based wumpus agent exploring the environment
 - Logical reasoning

1,4	2,4	3,4	4,4
1,3 W!	2,3	3,3	4,3
1,2 A S OK	2,2 OK	3,2	4,2
1,1 V OK	2,1 B V OK	3,1 P!	4,1

A = Agent
B = Breeze
G = Glitter, Gold
OK = Safe square
P = Pit
S = Stench
V = Visited
W = Wumpus

[Stench, None, None, None, None]

Example: The Wumpus World

- A knowledge-based wumpus agent exploring the environment
 - Logical reasoning

1,4	2,4 P?	3,4	4,4
1,3 W!	2,3 A S G B	3,3 P?	4,3
1,2 S V OK	2,2 V OK	3,2	4,2
1,1 V OK	2,1 B V OK	3,1 P!	4,1

A = Agent
 B = Breeze
 G = Glitter, Gold
 OK = Safe square
 P = Pit
 S = Stench
 V = Visited
 W = Wumpus

[Stench, Breeze, Glitter, None, None]

Logic in General

- Logics
 - Formal languages for representing information such that conclusions can be drawn
- Syntax
 - Defines the sentences in the language
- Semantics
 - Defines the truth of a sentence in a possible world (model)
- E.g., the language of arithmetic
 - $x + 2 \geq y$ is a sentence; $x^2 + y >$ is not a sentence
 - $x + 2 \geq y$ is true iff the number $x + 2$ is no less than the number y
 - $x + 2 \geq y$ is true in a world where $x = 7, y = 1$
 $x + 2 \geq y$ is false in a world where $x = 0, y = 6$

Models

- A possible world is represented by a **model**
- **Models** are mathematical abstractions, each of which simply fixes **the truth or falsehood** of every relevant sentences.
- m is a model of a sentence α if α is true in model m .
- $M(\alpha)$: the set of all models of α .

Entailment

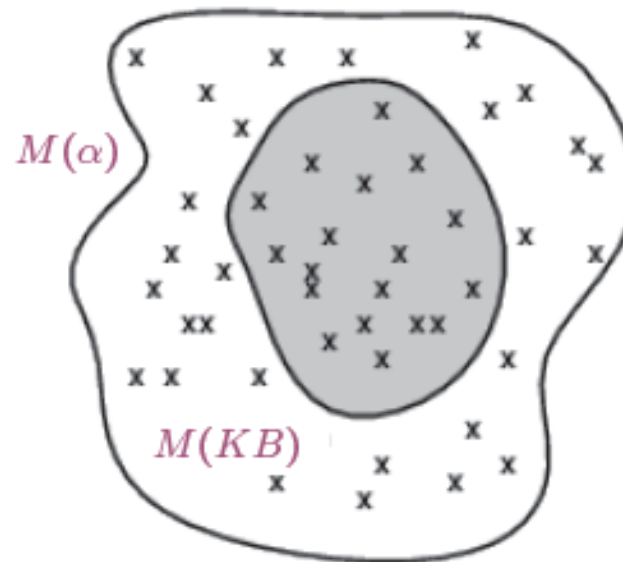
- **Entailment**: a sentence follows logically from another:

$$KB \models \alpha$$

- Knowledge base KB entails sentence α iff α is true in all models where KB is true

$$KB \models \alpha \text{ if and only if } M(KB) \subseteq M(\alpha)$$

- Eg. $x=0$ entails $xy=0$



Entailment in the Wumpus World

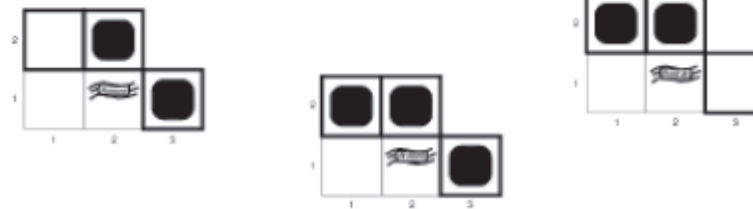
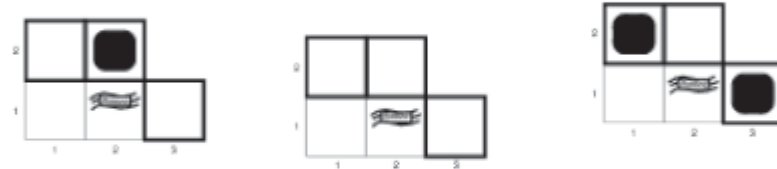
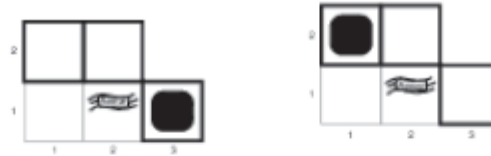
1,4	2,4	3,4	4,4
1,3 W!	2,3	3,3	4,3
1,2 ?	2,2 ?	3,2	4,2
1,1 A →	2,1 B	3,1 ?	4,1

- Situation after detecting nothing in [1,1], moving right, breeze in [2,1]
- Consider possible models for all ?, assuming only pits
- 3 Boolean choices \implies 8 possible models

Entailment in the Wumpus World

■ Possible Wumpus Models

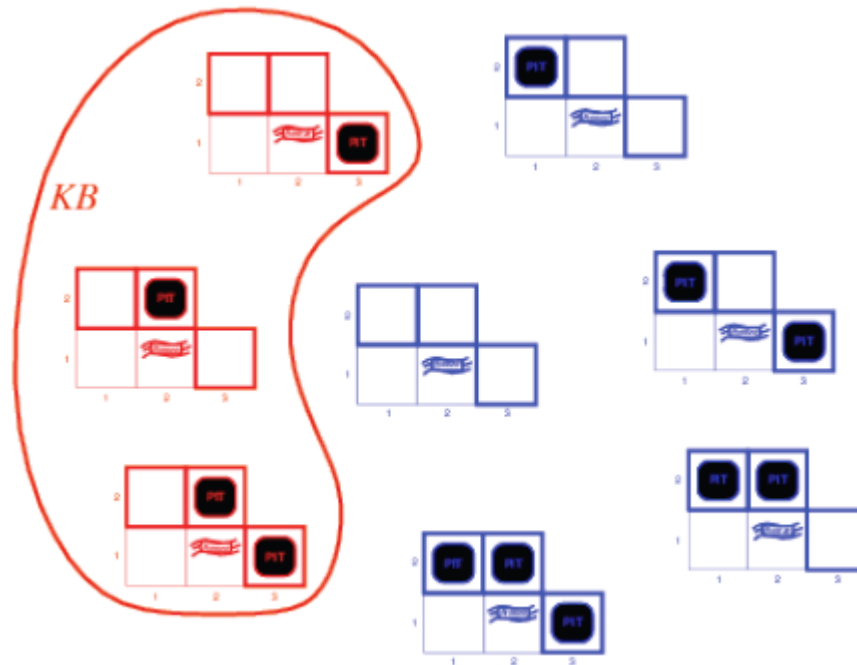
1,4	2,4	3,4	4,4
1,3 w!	2,3	3,3	4,3
1,2 ?	2,2 ?	3,2	4,2
1,1 A → A	2,1 B	3,1 ?	4,1



Entailment in the Wumpus World

Valid Wumpus Models

1,4	2,4	3,4	4,4
1,3 w!	2,3	3,3	4,3
1,2 ?	2,2 ?	3,2	4,2
1,1 A → A	2,1 B	3,1 ?	4,1

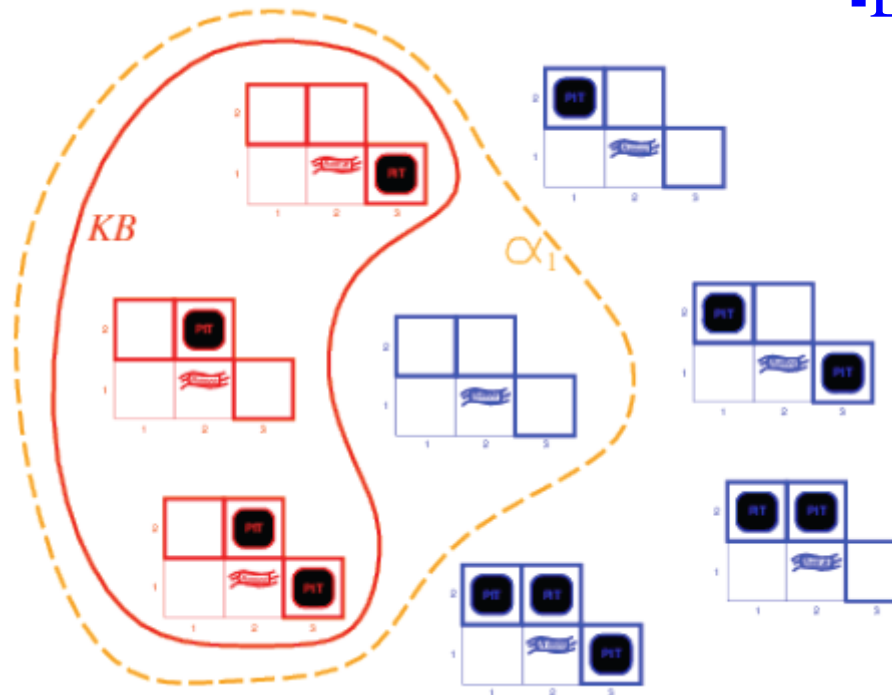


KB = wumpus-world rules + observations

Entailment in the Wumpus World

■ Entailment

1,4	2,4	3,4	4,4
1,3 W!	2,3	3,3	4,3
1,2 ? ?	2,2 ?	3,2	4,2
1,1 A → B ? 4,1	2,1 B	3,1 ?	



■ Logical inference

Model checking:

Enumerate the models, and check that α is true in every model in which KB is true

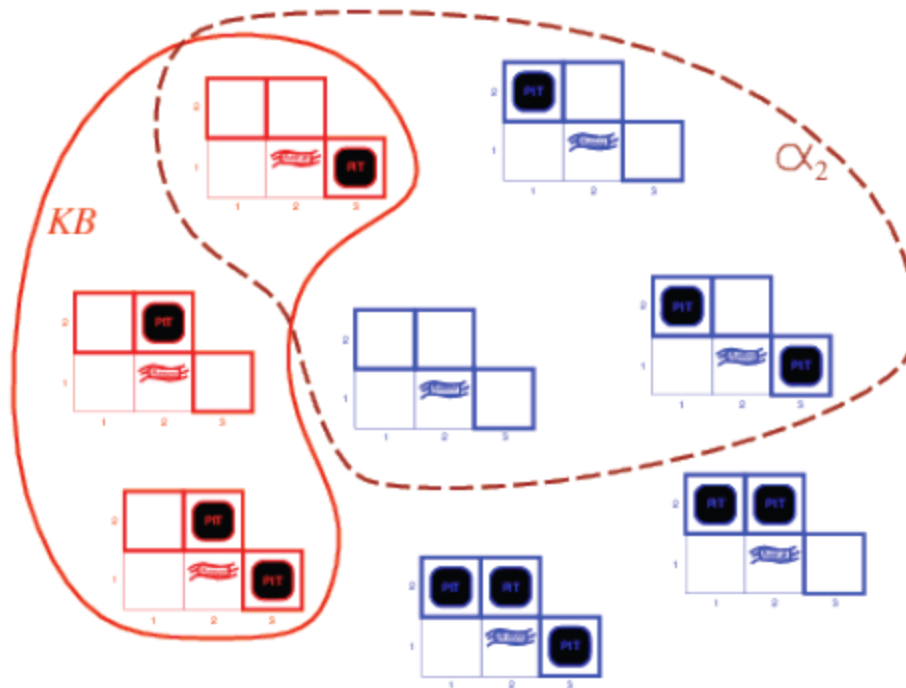
KB = wumpus-world rules + observations

α_1 = "[1,2] is safe", $KB \models \alpha_1$, proved by model checking

Entailment in the Wumpus World

■ Not Entailed

1,4	2,4	3,4	4,4
1,3 w!	2,3	3,3	4,3
1,2 ?	2,2 ?	3,2	4,2
1,1 A → A	2,1 B	3,1 ?	4,1



KB = wumpus-world rules + observations

α_2 = "[2,2] is safe", $KB \not\models \alpha_2$

Inference

- If an inference algorithm i can derive α from KB

$$KB \vdash_i \alpha$$

- Soundness

i is sound if

whenever $KB \vdash_i \alpha$, it is also true that $KB \models \alpha$

- Completeness

i is complete if

whenever $KB \models \alpha$, it is also true that $KB \vdash_i \alpha$

Propositional Logic: Syntax

- Propositional logic is the simplest logic—illustrates basic ideas
- The proposition symbols P_1, P_2 etc are sentences
- If P is a sentence, $\neg P$ is a sentence (negation) logical connectives
- If P_1 and P_2 are sentences, $P_1 \wedge P_2$ is a sentence (conjunction)
- If P_1 and P_2 are sentences, $P_1 \vee P_2$ is a sentence (disjunction)
- If P_1 and P_2 are sentences, $P_1 \implies P_2$ is a sentence (implication)
- If P_1 and P_2 are sentences, $P_1 \Leftrightarrow P_2$ is a sentence (biconditional)

Propositional Logic: Syntax

- BNF (Backus-Naur Form) grammar of sentences

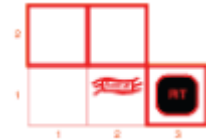
$$\begin{aligned} \textit{Sentence} &\rightarrow \textit{AtomicSentence} \mid \textit{ComplexSentence} \\ \textit{AtomicSentence} &\rightarrow \textit{True} \mid \textit{False} \mid P \mid Q \mid R \mid \dots \\ \textit{ComplexSentence} &\rightarrow (\textit{Sentence}) \mid [\textit{Sentence}] \\ &\mid \neg \textit{Sentence} \\ &\mid \textit{Sentence} \wedge \textit{Sentence} \\ &\mid \textit{Sentence} \vee \textit{Sentence} \\ &\mid \textit{Sentence} \Rightarrow \textit{Sentence} \\ &\mid \textit{Sentence} \Leftrightarrow \textit{Sentence} \end{aligned}$$

OPERATOR PRECEDENCE : $\neg, \wedge, \vee, \Rightarrow, \Leftrightarrow$

Propositional Logic: Semantics

- Each model specifies true/false for each proposition symbol

E.g. $m_1 = \{P_{1,2} = \text{false}, P_{2,2} = \text{false}, P_{3,1} = \text{true}\}$



(with these symbols, 8 possible models, can be enumerated automatically)■

- Rules for evaluating truth with respect to a model m :

$\neg P$	is true iff	P	is false		
$P_1 \wedge P_2$	is true iff	P_1	is true and	P_2	is true
$P_1 \vee P_2$	is true iff	P_1	is true or	P_2	is true
$P_1 \implies P_2$	is true iff	P_1	is false or	P_2	is true
i.e.,	is false iff	P_1	is true and	P_2	is false
$P_1 \Leftrightarrow P_2$	is true iff	$P_1 \implies P_2$	is true and	$P_2 \implies P_1$	is true■

- Simple recursive process evaluates an arbitrary sentence, e.g.,
 $\neg P_{1,2} \wedge (P_{2,2} \vee P_{3,1}) = \text{true} \wedge (\text{false} \vee \text{true}) = \text{true} \wedge \text{true} = \text{true}$

Truth Tables for Connectives

- Truth table

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
false	false	true	false	false	true	true
false	true	true	false	true	true	false
true	false	false	false	true	false	false
true	true	false	true	true	true	true

The Wumpus World

■ Knowledge base

- Let $P_{i,j}$ be true if there is a pit in $[i, j]$
 - observation $R_1 : \neg P_{1,1}$
- Let $B_{i,j}$ be true if there is a breeze in $[i, j]$.
- “Pits cause breezes in adjacent squares”
 - rule $R_2 : B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$
 - rule $R_3 : B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$
 - observation $R_4 : \neg B_{1,1}$
 - observation $R_5 : B_{2,1}$
- What can we infer about $P_{1,2}$, $P_{2,1}$, $P_{2,2}$, etc.?

1,4	2,4	3,4	4,4
1,3 w!	2,3	3,3	4,3
1,2 ?	2,2 ?	3,2	4,2
1,1 A →	2,1 B A	3,1 ?	4,1

The Wumpus World

- Model checking
 - Truth tables for inference

$$R_1 : \neg P_{1,1}$$

$$R_2 : B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$$

$$R_3 : B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$$

$$R_4 : \neg B_{1,1}$$

$$R_5 : B_{2,1}$$

$B_{1,1}$	$B_{2,1}$	$P_{1,1}$	$P_{1,2}$	$P_{2,1}$	$P_{2,2}$	$P_{3,1}$	R_1	R_2	R_3	R_4	R_5	KB
false	false	false	false	false	false	false	true	true	true	true	false	false
false	false	false	false	false	false	true	true	true	false	true	false	false
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
false	true	false	false	false	false	false	true	true	false	true	true	false
false	true	false	false	false	false	true	true	true	true	true	true	<u>true</u>
false	true	false	false	false	true	false	true	true	true	true	true	<u>true</u>
false	true	false	false	false	true	true	true	true	true	true	true	<u>true</u>
false	true	false	false	true	false	false	true	false	false	true	true	false
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
true	true	true	true	true	true	true	false	true	true	false	true	false

Inference by Model Checking

function TT-ENTAILS?(KB, α) **returns** *true* or *false*

inputs: KB , the knowledge base, a sentence in propositional logic
 α , the query, a sentence in propositional logic

$symbols \leftarrow$ a list of the proposition symbols in KB and α

return TT-CHECK-ALL($KB, \alpha, symbols, \{ \}$)

- Sound
- Complete
- $O(2^n)$

function TT-CHECK-ALL($KB, \alpha, symbols, model$) **returns** *true* or *false*

if EMPTY?($symbols$) **then**

if PL-TRUE?($KB, model$) **then return** PL-TRUE?($\alpha, model$)

else return *true* // when KB is *false*, always return *true*

else do

$P \leftarrow$ FIRST($symbols$)

$rest \leftarrow$ REST($symbols$)

return (TT-CHECK-ALL($KB, \alpha, rest, model \cup \{P = true\}$)

and

 TT-CHECK-ALL($KB, \alpha, rest, model \cup \{P = false\}$))

Figure 7.10 A truth-table enumeration algorithm for deciding propositional entailment. (TT stands for truth table.) PL-TRUE? returns *true* if a sentence holds within a model. The variable *model* represents a partial model—an assignment to some of the symbols. The keyword “and” is used here as a logical operation on its two arguments, returning *true* or *false*.

Inference

- Model checking:
 - Enumerating models and checking that α is true in every model in which KB is true.
- Theorem proving:
 - Applying rules of inference directly to the sentences in knowledge base to construct a proof of the desired sentence without consulting models.

Inference

▪ Logical equivalence

- Two sentences are logically equivalent iff true in same models:
 $\alpha \equiv \beta$ if and only if $\alpha \models \beta$ and $\beta \models \alpha$

$(\alpha \wedge \beta)$	\equiv	$(\beta \wedge \alpha)$	commutativity of \wedge
$(\alpha \vee \beta)$	\equiv	$(\beta \vee \alpha)$	commutativity of \vee
$((\alpha \wedge \beta) \wedge \gamma)$	\equiv	$(\alpha \wedge (\beta \wedge \gamma))$	associativity of \wedge
$((\alpha \vee \beta) \vee \gamma)$	\equiv	$(\alpha \vee (\beta \vee \gamma))$	associativity of \vee
$\neg(\neg\alpha)$	\equiv	α	double-negation elimination
$(\alpha \implies \beta)$	\equiv	$(\neg\beta \implies \neg\alpha)$	contraposition
$(\alpha \implies \beta)$	\equiv	$(\neg\alpha \vee \beta)$	implication elimination
$(\alpha \Leftrightarrow \beta)$	\equiv	$((\alpha \implies \beta) \wedge (\beta \implies \alpha))$	biconditional elimination
$\neg(\alpha \wedge \beta)$	\equiv	$(\neg\alpha \vee \neg\beta)$	De Morgan
$\neg(\alpha \vee \beta)$	\equiv	$(\neg\alpha \wedge \neg\beta)$	De Morgan
$(\alpha \wedge (\beta \vee \gamma))$	\equiv	$((\alpha \wedge \beta) \vee (\alpha \wedge \gamma))$	distributivity of \wedge over \vee
$(\alpha \vee (\beta \wedge \gamma))$	\equiv	$((\alpha \vee \beta) \wedge (\alpha \vee \gamma))$	distributivity of \vee over \wedge

Inference

■ Validity

- A sentence is **valid** if it is true in **all** models,
e.g., $True$, $A \vee \neg A$, $A \implies A$, $(A \wedge (A \implies B)) \implies B$
- Validity is connected to inference via the **Deduction Theorem**:
 $KB \models \alpha$ if and only if $(KB \implies \alpha)$ is valid■

■ Satisfiability

- A sentence is **satisfiable** if it is true in **some** model
e.g., $A \vee B$, C ■
- A sentence is **unsatisfiable** if it is true in **no** models
e.g., $A \wedge \neg A$ ■
- Satisfiability is connected to inference via the following:
 $KB \models \alpha$ if and only if $(KB \wedge \neg \alpha)$ is unsatisfiable

Inference: Proof by Resolution

- Resolution Rule:

$$\frac{\ell_1 \vee \dots \vee \ell_k, \quad m_1 \vee \dots \vee m_n}{\ell_1 \vee \dots \vee \ell_{i-1} \vee \ell_{i+1} \vee \dots \vee \ell_k \vee m_1 \vee \dots \vee m_{j-1} \vee m_{j+1} \vee \dots \vee m_n}$$

where ℓ_i and m_j are complementary literals. E.g.,

$$\frac{P_{1,3} \vee P_{2,2}, \quad \neg P_{2,2}}{P_{1,3}}$$

Inference: Proof by Resolution

▪ Conjunctive Normal Form

$$CNFSentence \rightarrow Clause_1 \wedge \dots \wedge Clause_n$$

$$Clause \rightarrow Literal_1 \vee \dots \vee Literal_m$$


$$Literal \rightarrow Symbol \mid \neg Symbol$$

$$Symbol \rightarrow P \mid Q \mid R \mid \dots$$

$$HornClauseForm \rightarrow DefiniteClauseForm \mid GoalClauseForm$$

$$DefiniteClauseForm \rightarrow (Symbol_1 \wedge \dots \wedge Symbol_l) \Rightarrow Symbol$$

$$GoalClauseForm \rightarrow (Symbol_1 \wedge \dots \wedge Symbol_l) \Rightarrow False$$

conjunction of **disjunctions** of **literals**

clauses

E.g., $(A \vee \neg B) \wedge (B \vee \neg C \vee \neg D)$ ■

Inference: Proof by Resolution

- Example: The Wampus World

1,4	2,4	3,4	4,4
1,3	2,3	3,3	4,3
1,2 ?	2,2	3,2	4,2
1,1 A B OK	2,1 ?	3,1	4,1

- Rules such as: “If breeze, then a pit adjacent.”

$$B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$$

Inference: Proof by Resolution

▪ Example: The Wampus World

$$B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$$

1. Eliminate \Leftrightarrow , replacing $\alpha \Leftrightarrow \beta$ with $(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)$.

$$(B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$$

2. Eliminate \Rightarrow , replacing $\alpha \Rightarrow \beta$ with $\neg\alpha \vee \beta$.

$$(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg(P_{1,2} \vee P_{2,1}) \vee B_{1,1})$$

3. Move \neg inwards using de Morgan's rules and double-negation:

$$(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge ((\neg P_{1,2} \wedge \neg P_{2,1}) \vee B_{1,1})$$

4. Apply distributivity law (\vee over \wedge) and flatten:

$$(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg P_{1,2} \vee B_{1,1}) \wedge (\neg P_{2,1} \vee B_{1,1})$$

Inference: Proof by Resolution

■ Example: The Wampus World

- $KB = (B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1}))$

reformulated as:

$$(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg P_{1,2} \vee B_{1,1}) \wedge (\neg P_{2,1} \vee B_{1,1})$$

- Observation: $\neg B_{1,1}$
- Goal: disprove: $\alpha = \neg P_{1,2}$
- Resolution

$$\frac{\neg P_{1,2} \vee B_{1,1} \quad \neg B_{1,1}}{\neg P_{1,2}}$$

- Resolution

$$\frac{\neg P_{1,2} \quad P_{1,2}}{\text{false}}$$

1,4	2,4	3,4	4,4
1,3	2,3	3,3	4,3
1,2 ?	2,2	3,2	4,2
1,1 A OK	2,1 ?	3,1	4,1

$KB \models \alpha$ if and only if $(KB \wedge \neg \alpha)$ is unsatisfiable

Inference: Proof by Resolution

▪ Resolution Algorithm

```
function PL-RESOLUTION( $KB, \alpha$ ) returns true or false  
  inputs:  $KB$ , the knowledge base, a sentence in propositional logic  
            $\alpha$ , the query, a sentence in propositional logic  
  
   $clauses \leftarrow$  the set of clauses in the CNF representation of  $KB \wedge \neg\alpha$   
   $new \leftarrow \{ \}$   
  loop do  
    for each pair of clauses  $C_i, C_j$  in  $clauses$  do  
       $resolvents \leftarrow$  PL-RESOLVE( $C_i, C_j$ )  
      if  $resolvents$  contains the empty clause then return true  
       $new \leftarrow new \cup resolvents$   
  if  $new \subseteq clauses$  then return false  
   $clauses \leftarrow clauses \cup new$ 
```

- Resolution is **sound** and **complete** for propositional logic

Inference: Proof by Resolution

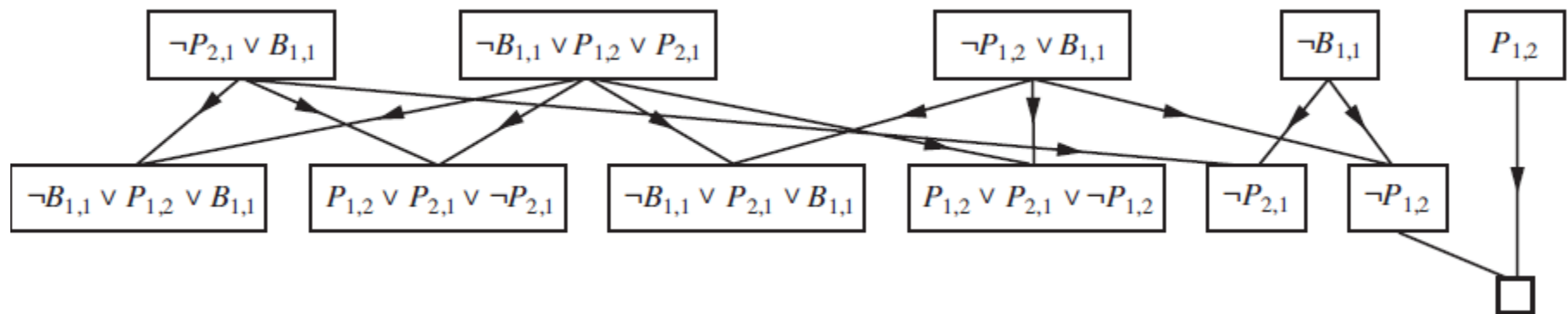
▪ Example: The Wampus World

- $KB = (B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1}))$

reformulated as:

$$(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg P_{1,2} \vee B_{1,1}) \wedge (\neg P_{2,1} \vee B_{1,1})$$

- Observation: $\neg B_{1,1}$
- Goal: disprove: $\alpha = \neg P_{1,2}$



Inference:

- **Definite clause:**
 - A disjunction of literals of which exactly one is positive.
 - Eg. $\neg L_{1,1} \vee \neg Breeze \vee B_{1,1}$
- **Horn clause:**
 - A disjunction of literals of which at most one is positive.
- **Goal clause:**
 - Clauses with no positive literals.
- **Horn clauses are closed under resolution.**
 - If you resolve two Horn clauses, you get back a Horn clause.

Inference:

- Every **definite clause** can be written as
 - a conjunction of positive literals \Rightarrow a single positive literal

$$\neg L_{1,1} \vee \neg Breeze \vee B_{1,1} \quad (L_{1,1} \wedge Breeze) \Rightarrow B_{1,1}$$

- Inference with Horn clauses can be done through **forward chaining** or **backward chaining** algorithms.
- These algorithms are very natural and run in **linear** time

Inference: Forward Chaining

- Forward chaining algorithm

```
function PL-FC-ENTAILS?(KB, q) returns true or false
  inputs: KB, the knowledge base, a set of propositional definite clauses
         q, the query, a proposition symbol
  count  $\leftarrow$  a table, where count[c] is the number of symbols in c's premise
  inferred  $\leftarrow$  a table, where inferred[s] is initially false for all symbols
  agenda  $\leftarrow$  a queue of symbols, initially symbols known to be true in KB

  while agenda is not empty do
    p  $\leftarrow$  POP(agenda)
    if p = q then return true
    if inferred[p] = false then
      inferred[p]  $\leftarrow$  true
      for each clause c in KB where p is in c.PREMISE do
        decrement count[c]
        if count[c] = 0 then add c.CONCLUSION to agenda
  return false
```

- Forward chaining is sound and complete.

Inference: Forward Chaining

■ Example

- Given

$$P \implies Q$$

$$L \wedge M \implies P$$

$$B \wedge L \implies M$$

$$A \wedge P \implies L$$

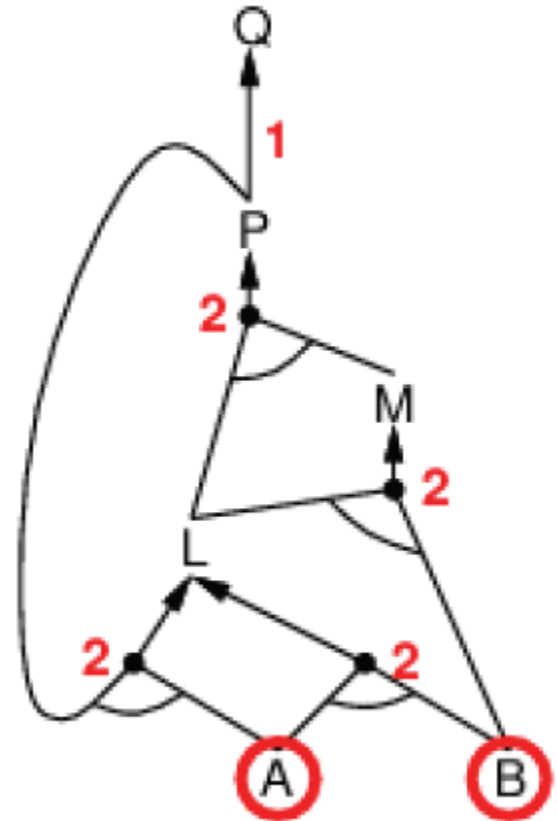
$$A \wedge B \implies L$$

A

B

- Agenda: A, B

- Annotate horn clauses with number of premises

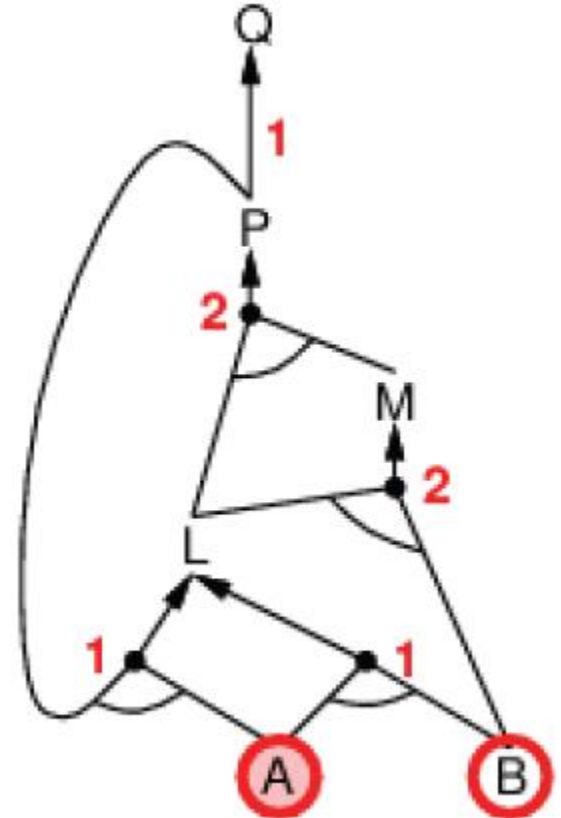


AND-OR Graph

Inference: Forward Chaining

■ Example

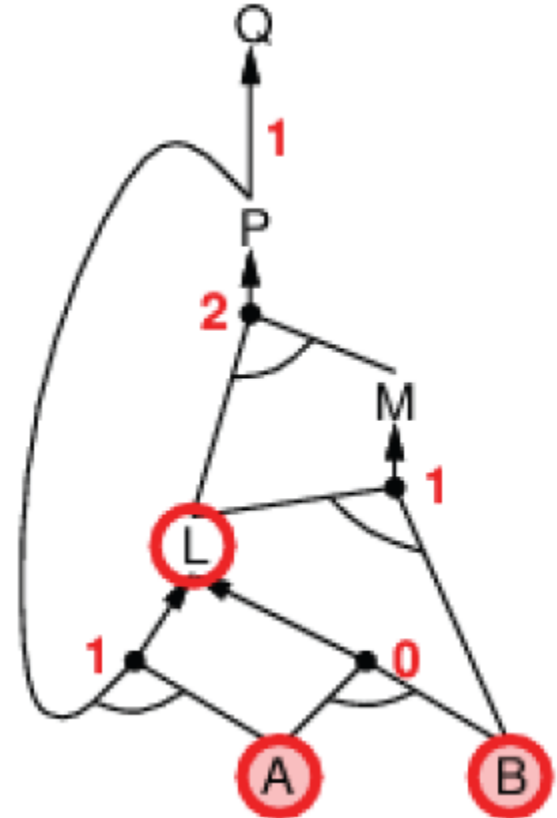
- Process agenda item A
- Decrease count for horn clauses in which A is premise



Inference: Forward Chaining

■ Example

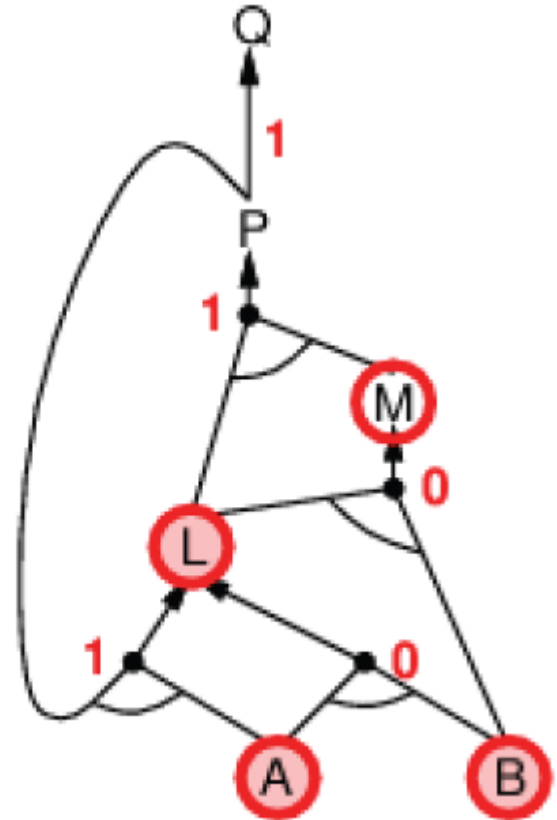
- Process agenda item B
- Decrease count for horn clauses in which B is premise
- $A \wedge B \implies L$ has now fulfilled premise
- Add L to agenda



Inference: Forward Chaining

■ Example

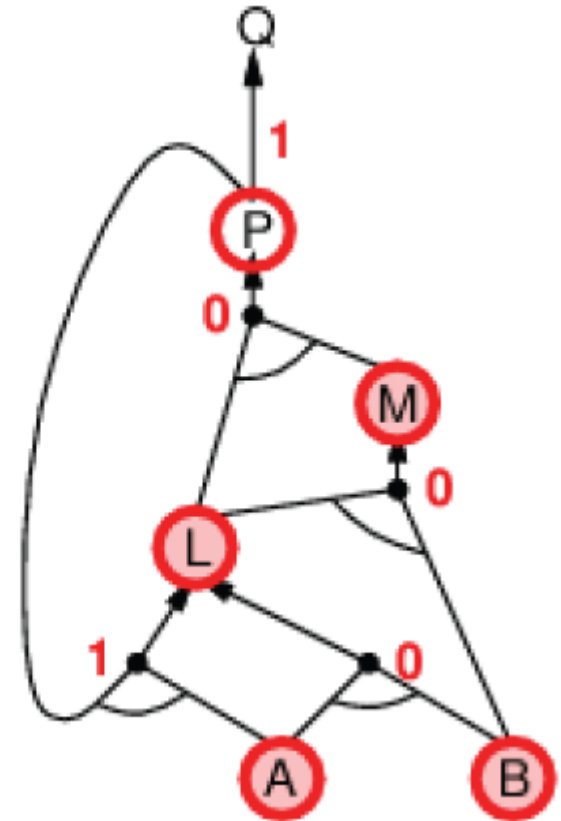
- Process agenda item L
- Decrease count for horn clauses in which L is premise
- $B \wedge L \implies M$ has now fulfilled premise
- Add M to agenda



Inference: Forward Chaining

■ Example

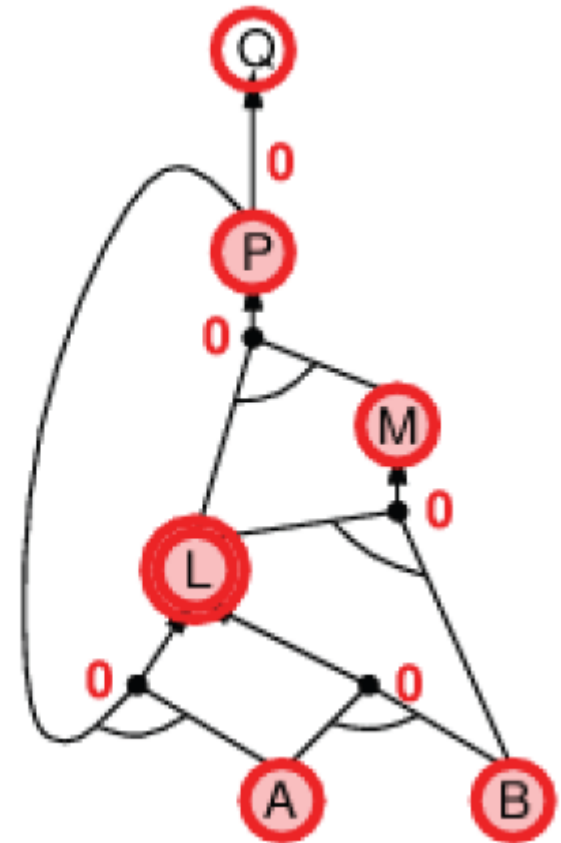
- Process agenda item M
- Decrease count for horn clauses in which M is premise
- $L \wedge M \implies P$ has now fulfilled premise
- Add P to agenda



Inference: Forward Chaining

■ Example

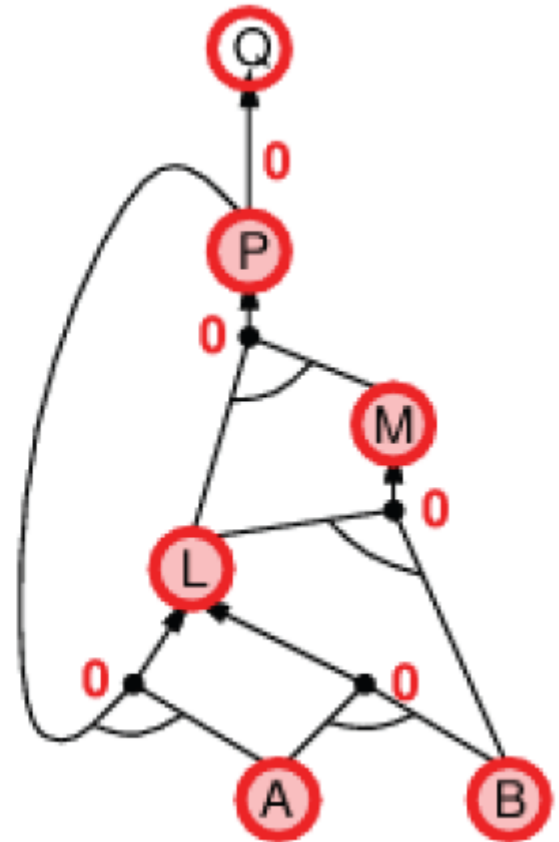
- Process agenda item P
- Decrease count for horn clauses in which P is premise
- $P \implies Q$ has now fulfilled premise
- Add Q to agenda
- $A \wedge P \implies L$ has now fulfilled premise



Inference: Forward Chaining

■ Example

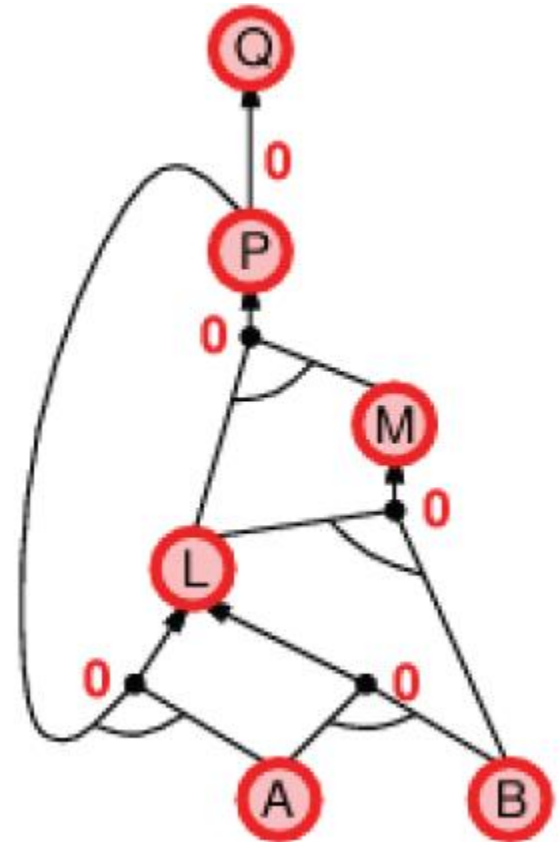
- Process agenda item P
- Decrease count for horn clauses in which P is premise
- $P \implies Q$ has now fulfilled premise
- Add Q to agenda
- $A \wedge P \implies L$ has now fulfilled premise
- But L is already inferred



Inference: Forward Chaining

■ Example

- Process agenda item Q
- Q is inferred
- Done



More Logics

■

Language	Ontological Commitment	Epistemological Commitment
Propositional logic	facts	true / false / unknown
First-order logic	facts, objects, relations	true / false / unknown
Temporal logic	facts, objects, relations, times	true / false / unknown
Probability theory	facts	degree of belief
Fuzzy logic	facts + degree of truth	known interval value

Higher-order logic:

relations and functions operate not only on objects,
but also on relations and functions

First-Order Logic

-
- Propositional logic: world contains **facts**
- First-order logic: the world contains **objects**, **relations**, and **functions**■
- **Objects**: people, houses, numbers, theories, Ronald McDonald, colors, baseball games, wars, centuries ... ■
- **Relations**: red, round, bogus, prime, multistoried ..., brother of, bigger than, inside, part of, has color, occurred after, owns, comes between, ... ■
- **Functions**: father of, best friend, third inning of, one more than, end of ...

First-Order Logic

■ Syntax

- Constants: *KingJohn, 2, UCB, ...*
- Predicates: *Brother, >, ...*
- Functions: *Sqrt, LeftLegOf, ...*
- Variables: *x, y, a, b, ...*
- Connectives: $\wedge \vee \neg \implies \iff$
- Equality: $=$
- Quantifiers: $\forall \exists$

First-Order Logic

■ Syntax

- Atomic sentence = *predicate(term₁, ..., term_n)*
or *term₁ = term₂*
- Term = *function(term₁, ..., term_n)*
or *constant* or *variable*
- E.g., *Brother(KingJohn, RichardTheLionheart)*
> (Length(LeftLegOf(Richard)), Length(LeftLegOf(KingJohn)))

First-Order Logic

■ Syntax

- Complex sentences are made from atomic sentences using connectives

$$\neg S, \quad S_1 \wedge S_2, \quad S_1 \vee S_2, \quad S_1 \implies S_2, \quad S_1 \leftrightarrow S_2$$

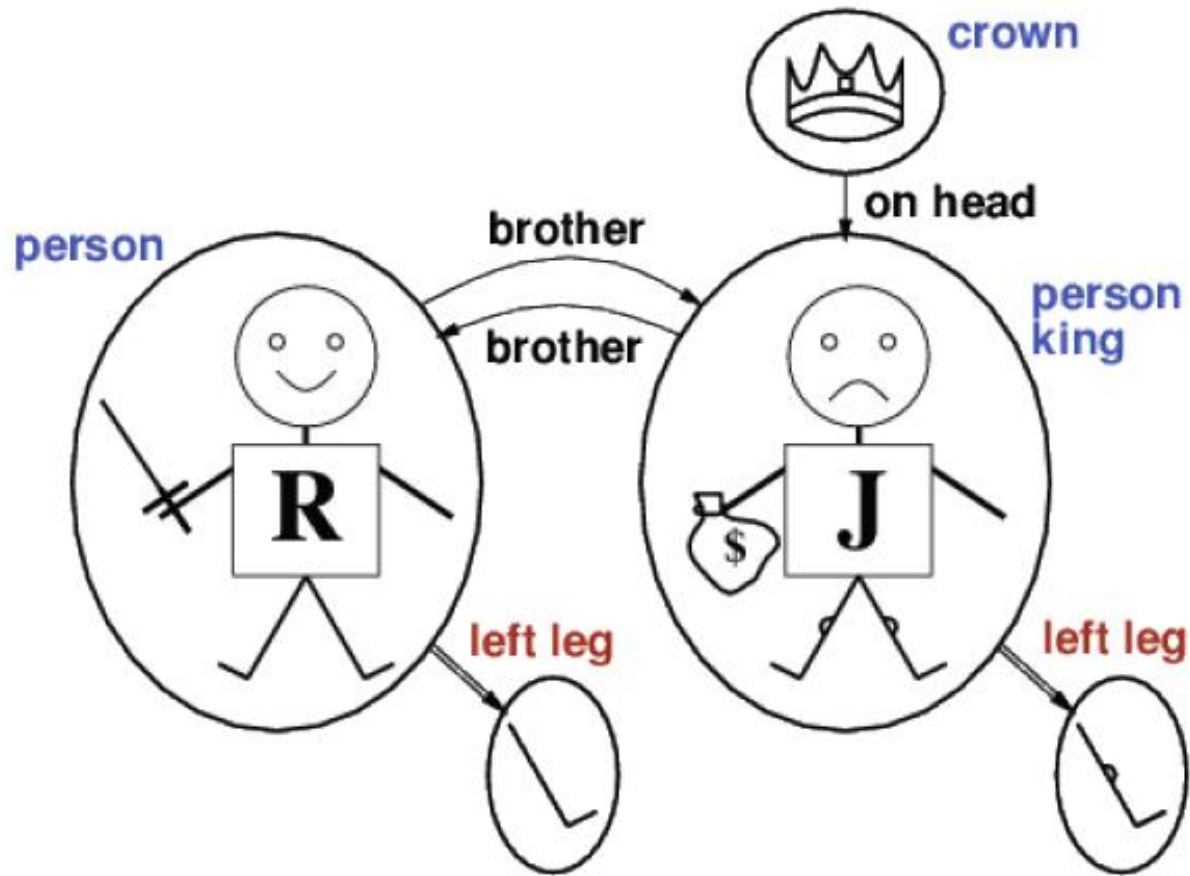
- For instance
 - $Sibling(KingJohn, Richard) \implies Sibling(Richard, KingJohn)$
 - $>(1, 2) \vee \leq(1, 2)$
 - $>(1, 2) \wedge \neg >(1, 2)$

First-Order Logic

- Semantics
 - Sentences are true with respect to a model and an interpretation
 - Model contains ≥ 1 objects (domain elements) and relations among them
 - Interpretation specifies referents for
 - constant symbols \rightarrow objects
 - predicate symbols \rightarrow relations
 - function symbols \rightarrow functional relations
 - An atomic sentence $\textit{predicate}(\textit{term}_1, \dots, \textit{term}_n)$ is true iff the objects referred to by $\textit{term}_1, \dots, \textit{term}_n$ are in the relation referred to by $\textit{predicate}$

First-Order Logic

- An example



Knowledge Representation

- Type of knowledge
 - Objects
 - Events
 - Procedures
 - Relations
 - Mental states
 - Meta knowledge

Knowledge Representation

- Properties of Representation Systems
 - Representational adequacy
 - ability to represent the required knowledge
 - Inferential adequacy
 - ability to manipulate knowledge
 - produce new knowledge
 - ability to direct inference methods into productive directions
 - ability to respond with limited resources (time, storage)
 - Acquisitional efficiency
 - ability to acquire new knowledge
 - ideally, automatically

Knowledge Representation

- Knowledge Graph



<https://googleblog.blogspot.com/2012/05/introducing-knowledge-graph-things-not.html>

Assignments

- Reading assignment:
 - Ch. 7.1-7.5, Ch. 8.1-8.2
- Homework 3:
 - Due by Apr. 12, 2021.