

Chapter 7

Logical Agents

CS361 Artificial Intelligence

Dr. Khaled Wassif

Spring 2021

(This is the instructor's notes and student has to read the textbook for complete material.)

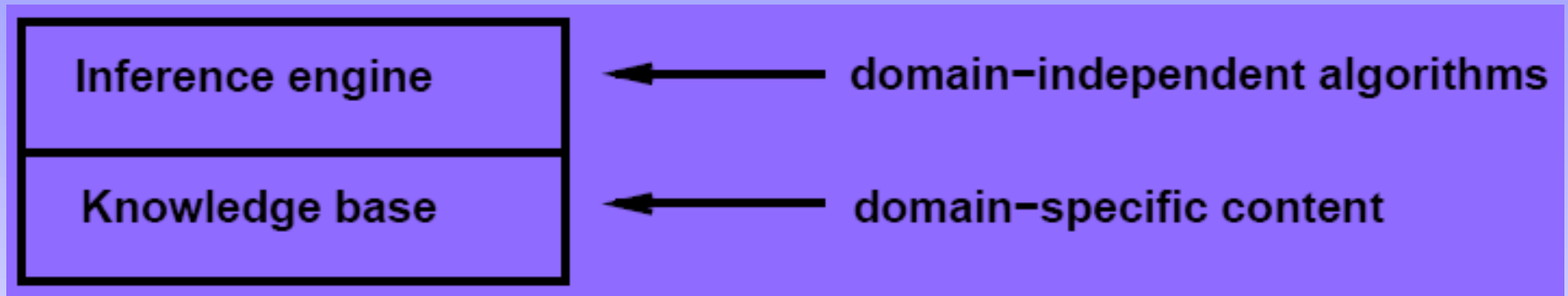
Chapter Outline

- Knowledge-based Agents
- Wumpus World
- Logic in General - Models and Entailment
- Propositional (Boolean) Logic
- Equivalence, Validity, Satisfiability
- Inference Rules and Theorem Proving
 - Resolution
 - Forward Chaining
 - Backward Chaining

Knowledge-based Agents

- Central component of a knowledge-based agent is its **knowledge base** (KB).
- A knowledge base is a set of sentences:
 - Expressed in a formal language (called **knowledge representation language**).
 - Represent some assertion about the world.
- There must be a way to add new sentences to the knowledge base, and a way to query what is known.
 - Standard names for these operations are TELL and ASK.
 - Main requirement when ASKs a question, the answer should follow from what has been TELLED to the KB previously.
- Determining what follows from what the KB has been TELLED is the job of the **inference mechanism**, the other main component of a knowledge-based agent.

Knowledge-based Agents



- **Declarative approach** to building a knowledge-based agent:
 - Tell it what it needs to know (called *background knowledge*)
 - Then it can Ask itself what to do
 - » Answers should follow from the KB.
 - » Extensive reasoning may be done.

Simple Knowledge-based Agent

```
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  $\leftarrow$  ASK(KB, MAKE-ACTION-QUERY(t))  
  TELL(KB, MAKE-ACTION-SENTENCE(action, t))  
  t  $\leftarrow$  t + 1  
  return action
```

Simple Knowledge-based Agent

- 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

Knowledge-based Agents

- A knowledge-based agent can be described at three levels:
 - Knowledge level
 - » Most abstract level that specify only what the agent knows and what its goals.
 - Logical level
 - » The level at which the knowledge is encoded into sentences using a knowledge representation language.
 - Implementation level
 - » The level that runs on the agent architecture.
 - » At which there are physical representations of the sentences of the logical level.

The WUMPUS World



The WUMPUS World (cont.)

- An early computer game:
 - Based on an agent who explores a cave consisting of rooms connected by passageways.
 - Lurking somewhere in the cave is the terrible wumpus, a beast that eats anyone who enters its room.
 - The wumpus can be shot by an agent, but the agent has only one arrow.
 - Some rooms contain bottomless pits that will trap anyone who wanders into these rooms (except the big wumpus).
 - The only explanatory feature of living in this environment is the possibility of finding a heap of gold.
- It makes an excellent test bed environment for intelligent agents.

Wumpus World PEAS Description

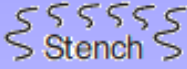
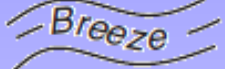



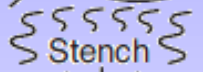


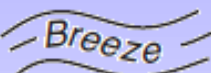
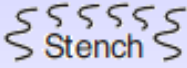
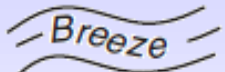

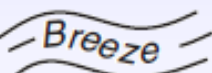

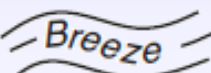
■ Performance measure:

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

■ Environment:

- A 4×4 grid of rooms and the agent always starts in [1,1]
- Squares adjacent to wumpus are smelly
- 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

Wumpus World PEAS Description

4				
3		  		
2				
1	 START			
	1	2	3	4

Wumpus World PEAS Description

■ Sensors:

- In the cell containing the wumpus, and in the directly (not diagonally) adjacent cells, the agent will perceive a Stench.
- In the cells adjacent to a pit, the agent will perceive a Breeze.
- In the cell containing the gold, the agent will perceive a Glitter.
- If the agent walks into a wall, it perceives a Bump.
- When the wumpus is killed, the agent perceives a Scream.
- The agent cannot perceive its location.

Thus, a percept is a quintuple

< Stench, Breeze, Glitter, Bump, Scream >

Wumpus World PEAS Description

■ **Actuators:**

- Move Forward.
- Turn Right.
- Turn Left.
- Grab.
- Shoot.
- Climb.

■ **Goals:**

- Get gold and return back to the start without entering a pit or wumpus square.

■ **Can search help the Wumpus world agent?**

Wumpus World Characterization

- Fully 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

Exploring Wumpus World

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

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

Exploring Wumpus World

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

Exploring Wumpus World

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 A OK	2,1 V 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

Exploring Wumpus World

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 V 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

Exploring Wumpus World

1,4	2,4	3,4	4,4
1,3 W	2,3 OK	3,3	4,3
1,2 V S OK	2,2 A OK	3,2 OK	4,2
1,1 V OK	2,1 V 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

Exploring Wumpus World

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 V S OK	2,2 V OK	3,2 OK	4,2
1,1 V OK	2,1 V 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

The WUMPUS World (cont.)

- Percepts provide deeper knowledge of the environment.
 - For example, sensing no stench while in [1,1], the agent should know that the wumpus is neither in [1,2] nor in [2,1].
- In order to update the state, given a new percept, the agent needs the following:
 1. Background knowledge linking percepts to possible contents of cells.
 2. Some way of representing this knowledge.
 3. Some way of mapping percepts to structures of the same representation.
 4. A method that manipulates these structures to update the agent's of the state of the environment.
- That is, the agent needs full-fledged **logical reasoning**.

What is Logic?

- There are different uses of the word “logic”.
 - But think of “logic” as a language for representing knowledge such that conclusions can be drawn.
 - » For example, knowledge of the Wumpus-world agent.
- Logic is a formal language and should have precise **syntax** and **semantics**.
 - Syntax defines sentences in the representation language.
 - Semantics define the "meaning" of sentences.
 - » Define the truth of each sentence with respect to each possible world.
 - E.g., arithmetic language
 - » “ $x + y = 4$ ” is a well-formed sentence; but “ $x4y+ =$ ” is not
 - » $x + 2 \geq y$ is true iff the number $x + 2$ is not less than the number y
 - In standard logics, every sentence must be either true or false in each possible world.

What is Logic?

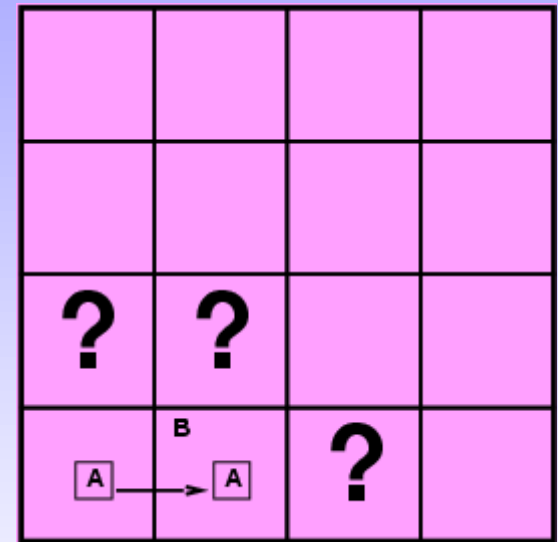
- **Model** is a word used instead of “possible world” for sake of precision.
 - If a sentence α is true in model m , we say that m **satisfies** α (or m **is a model of** α).
 - We use the notation $M(\alpha)$ to mean the set of all models of α .
- **Definition:**
 - Models are mathematical abstractions, each of which simply fixes the truth or falsehood of every relevant sentence.
 - Example:
 - » x number of men and y number of women sitting at a table playing bridge.
 - » $x + y = 4$ is a sentence which is true when the total number is four.
 - » Models: all possible assignments of real numbers to the variables x and y and each assignment fixes the truth of any arithmetic sentence whose variables are x and y .

Logical Entailment (Implication)

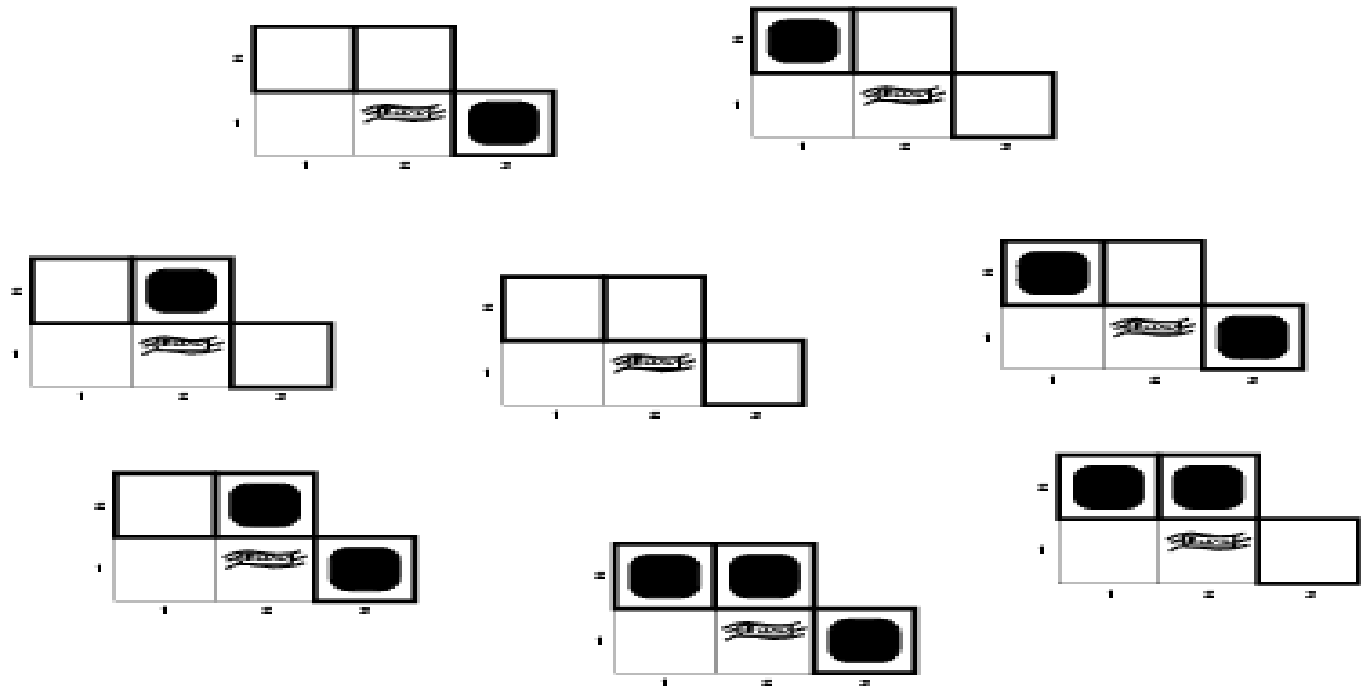
- A sentence α logically follows (**implies** or **entails**) a sentence β if and only if every model in which α is true, β is also true (written $\alpha \models \beta$).
 - Example
 - “All men are mortal and Socrates is a man”
logically implies “Socrates is mortal”
- Knowledge base KB entails sentence α if and only if α is true in all models where KB is true (written $KB \models \alpha$).
 - E.g., the KB containing “Both Ahmed and Gamal came” entails “Ahmed came”
 - E.g., $x = 0$ entails $x \cdot y = 0$
- Entailment is based only on semantics and does not depend at all on logical inference.

Entailment in Wumpus World

- A situation after detecting nothing in [1,1], moving right, and detecting breeze in [2,1]
- Consider possible models for *KB* assuming only pits
- 3 Boolean choices $\Rightarrow 2^3 = 8$ possible models

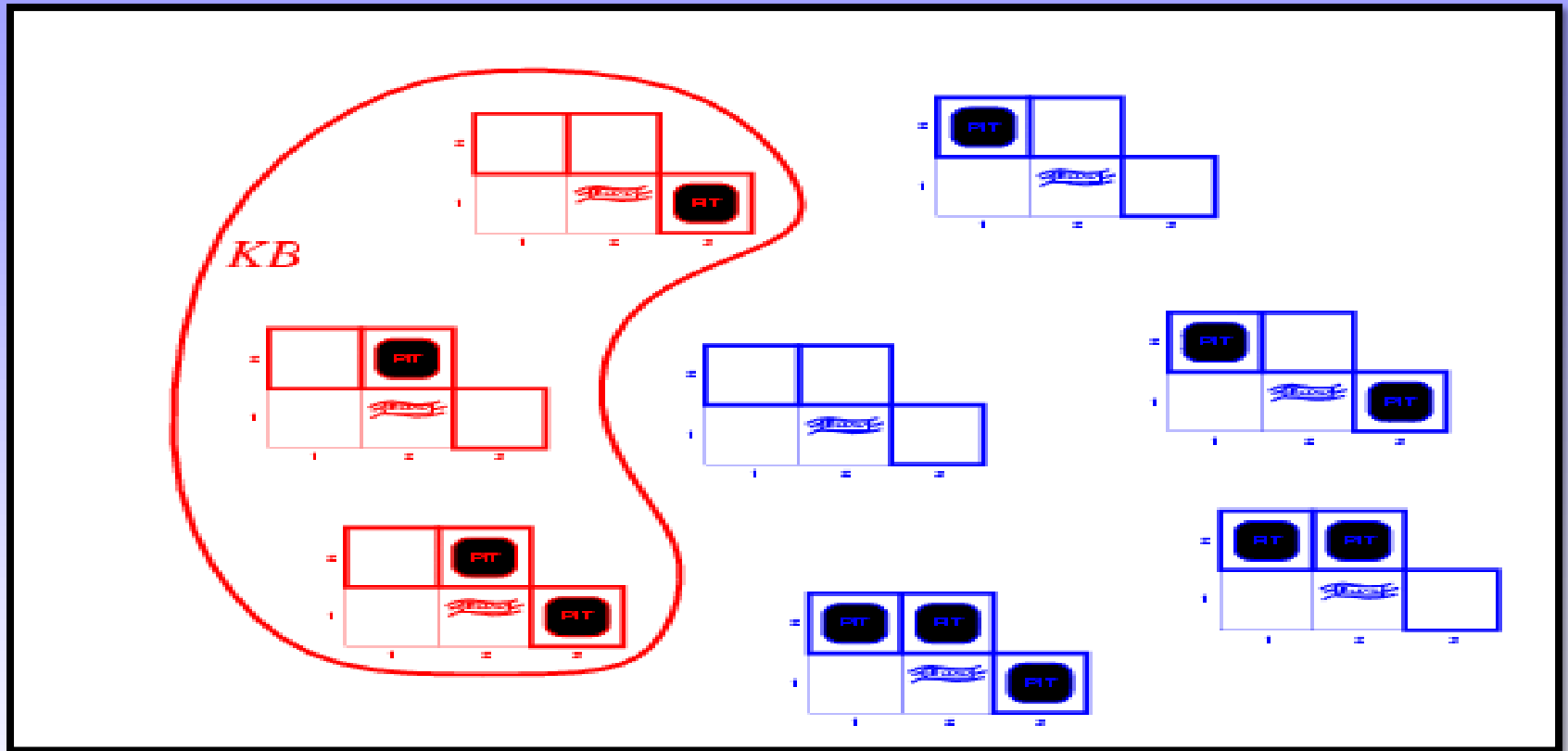


Wumpus Models



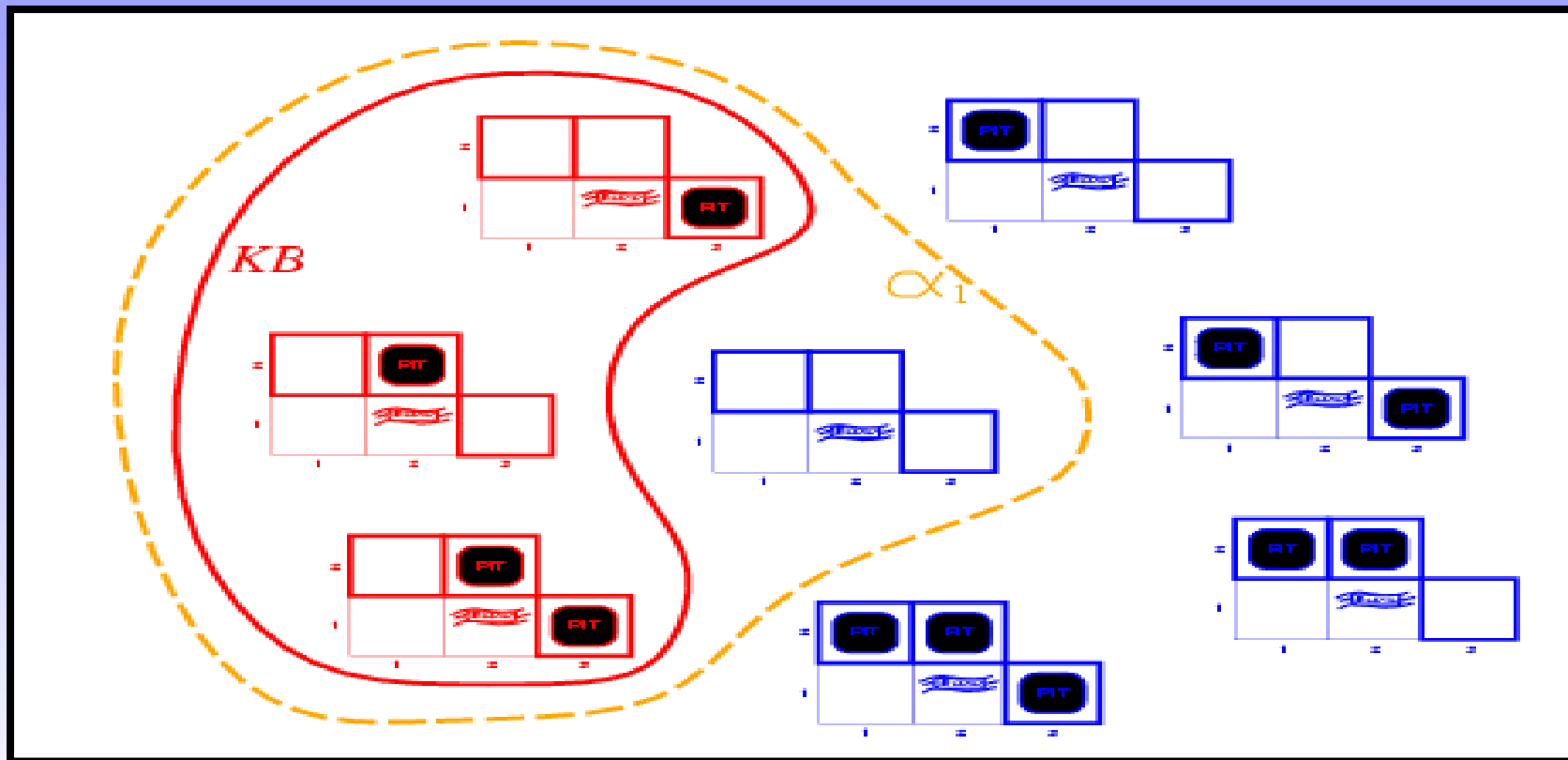
8 possible models

Wumpus Models



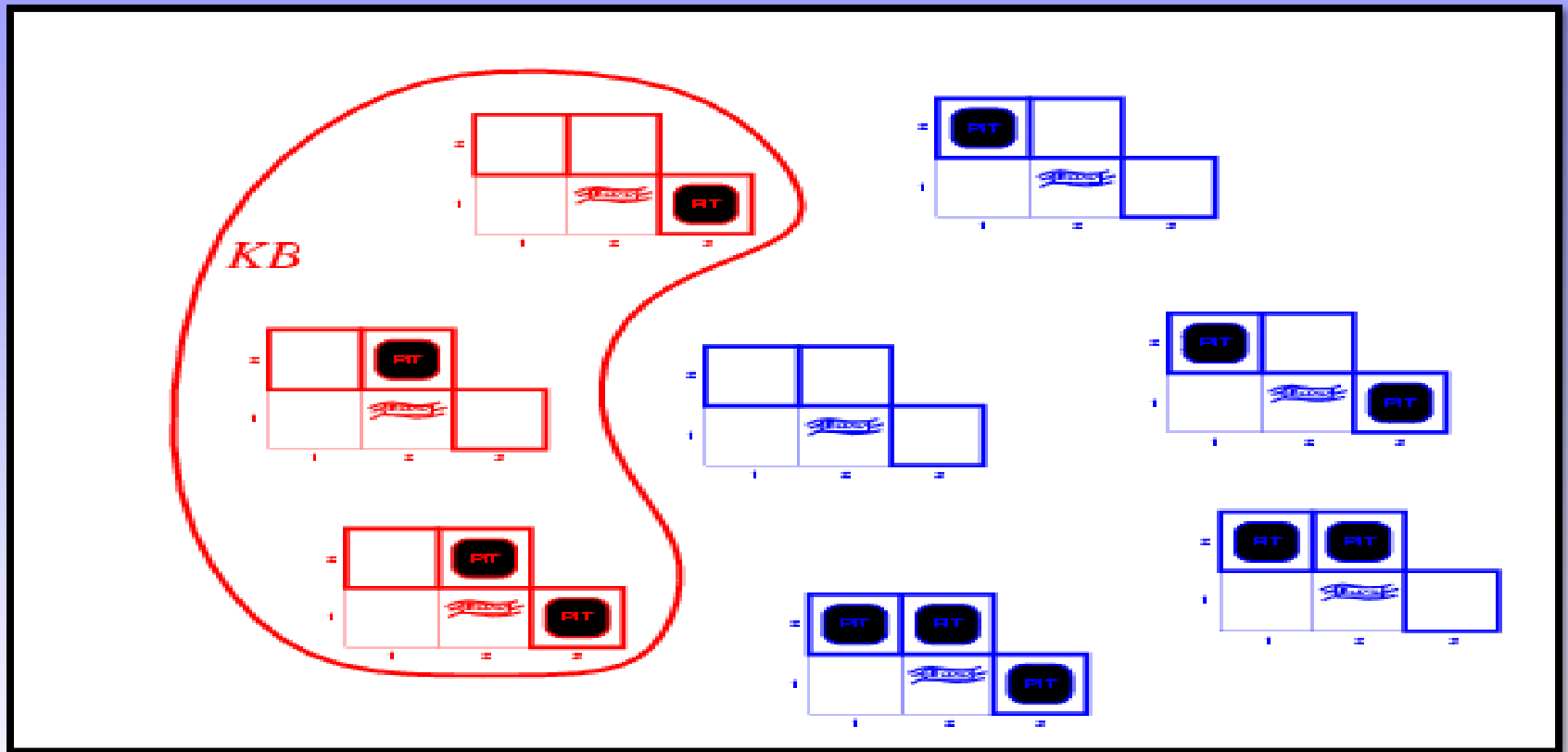
- $KB = \text{Wumpus-world rules} + \text{observations}$

Wumpus Models



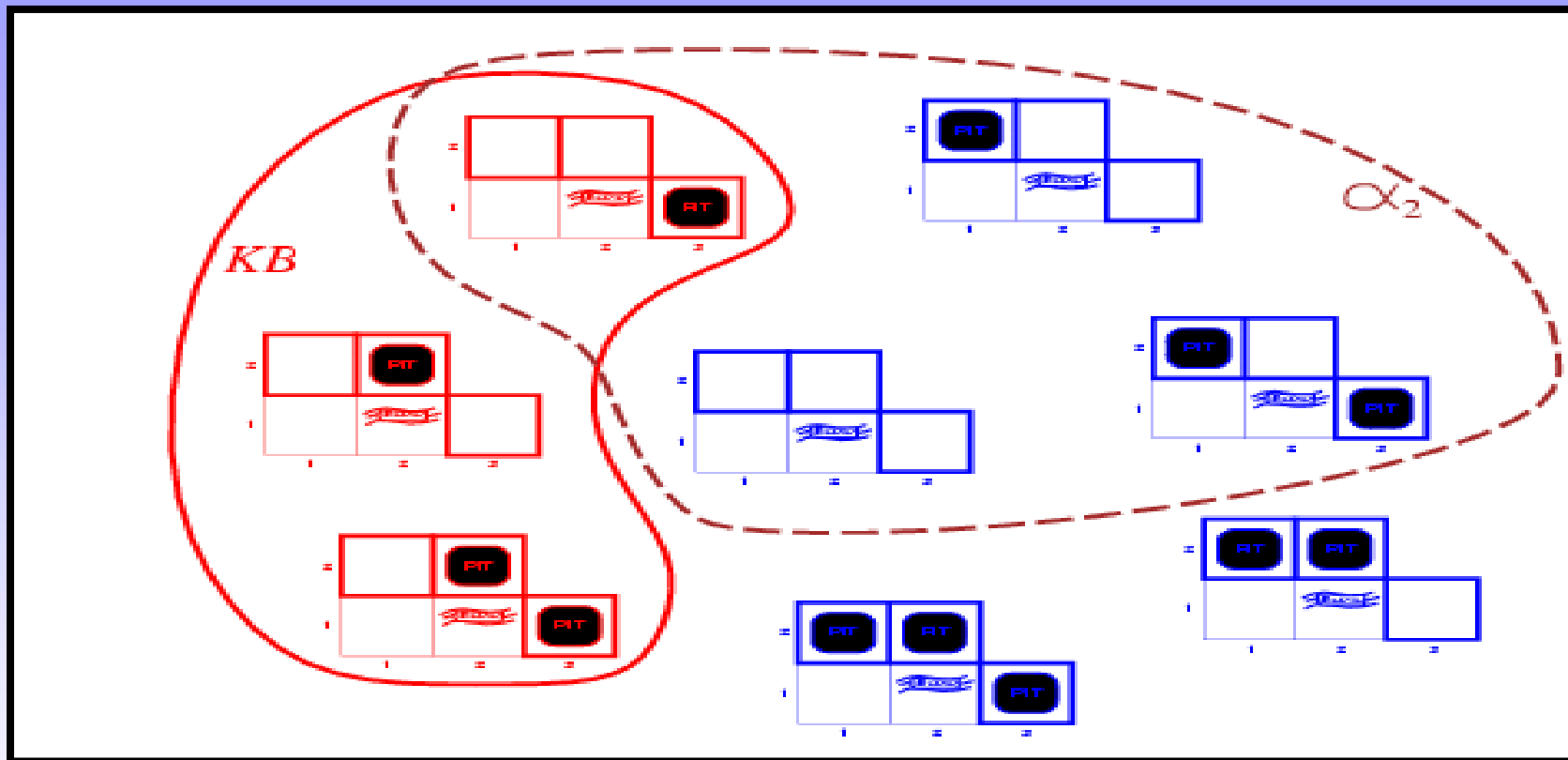
- KB = Wumpus-world rules + observations
- $\alpha_1 = "[1,2] \text{ is safe} "$, $KB \models \alpha_1$, proved by **model checking**

Wumpus Models



- $KB = \text{Wumpus-world rules} + \text{observations}$

Wumpus Models

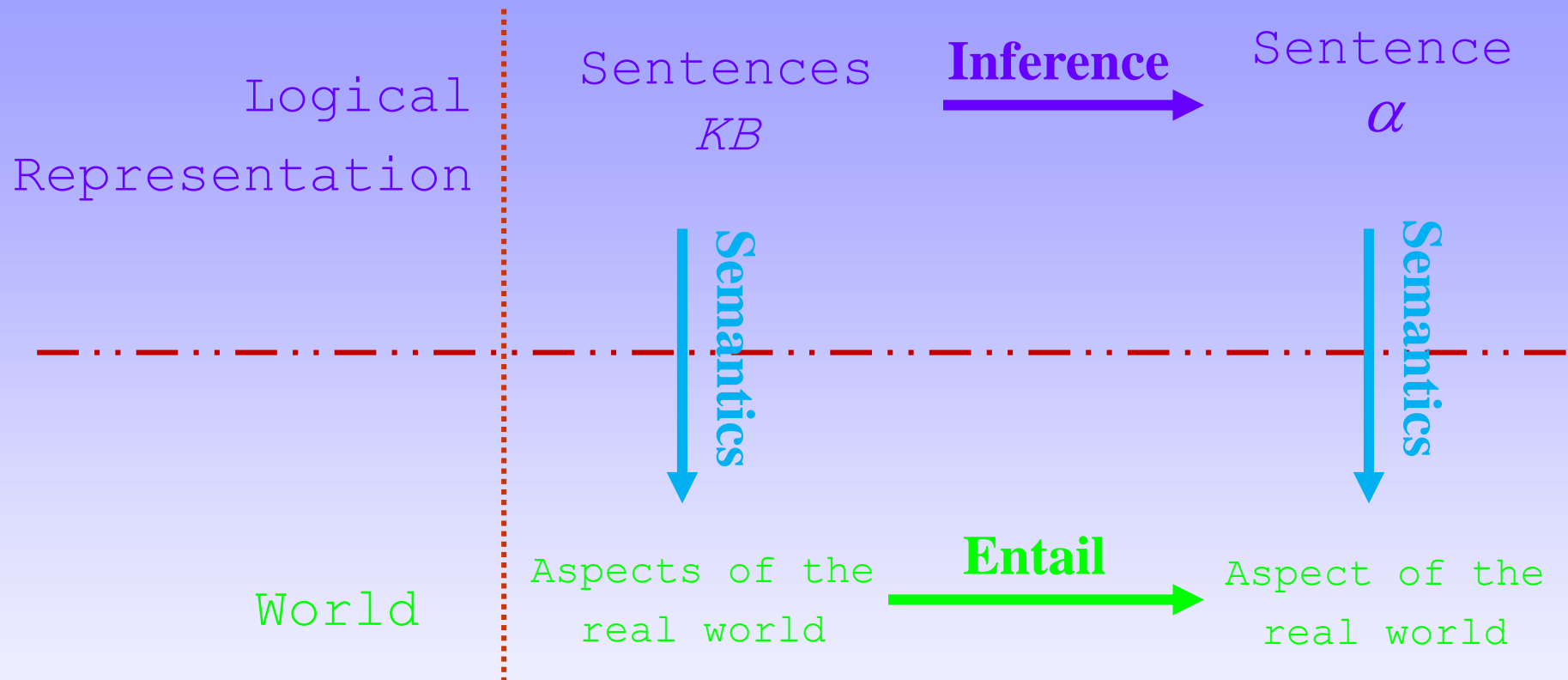


- KB = Wumpus-world rules + observations
- α_2 = "[2,2] is safe", $KB \not\models \alpha_2$

Logical Inference

- $KB \vdash_i \alpha$ = sentence α can be derived from KB by an inference procedure i .
- **Soundness**: i is sound if it derives *only* sentences that are entailed by KB .
$$KB \vdash_i \alpha \Rightarrow KB \models \alpha$$
 - It is easy to see that model checking is a sound procedure.
- **Completeness**: i is complete if it derives *all* sentences that are entailed by KB .
$$KB \models \alpha \Rightarrow KB \vdash_i \alpha$$
- There are sound and complete inference procedures for logics (first-order logic) that are sufficiently expressive to handle many knowledge bases.
- ***If KB is true in the real world, then any sentence α derived from KB by a sound inference procedure is also true in the real world.***

Logical Inference



Logical reasoning should ensure that the new configurations represent aspects of the world that actually follow from the aspects that the old configurations represent.

Propositional Logic

- Propositional logic is the simplest logic.
 - Syntax
 - Semantic
 - Entailment and Inference

Propositional Logic: Syntax

- Syntax defines the allowable sentences.
- Atomic sentence:
 - Single *proposition symbol*.
 - » Uppercase names for symbols must have some mnemonic value.
 - » Example: $W_{1,3}$ to say the Wumpus is in [1,3].
 - **True** and **False**: two proposition symbols with fixed meaning.
- Complex sentences:
 - Constructed from simpler sentences using *logical connectives*.

Propositional Logic: Syntax

■ Logical connectives:

1. \neg (not) **negation**.
2. \wedge (and) **conjunction**, its operands are conjuncts.
3. \vee (or) **disjunction**, its operands are disjuncts.
4. \Rightarrow (implies) **implication** or **conditional**.
 - » As $A \Rightarrow B$, A is the premise or antecedent and B is the conclusion or consequent.
 - » It is also known as rule or if-then statement.
5. \Leftrightarrow (if and only if) **equivalent** or **biconditional**.

Propositional Logic: Syntax

- Logical constants True and False are sentences.
- Proposition symbols P_1, P_2 etc. are sentences.
 - Symbols P_1 and negated symbols $\neg P_1$ are called **literals**.
- If S is a sentence, $\neg S$ is a sentence (**negation**).
- If S_1 and S_2 are sentences, $S_1 \wedge S_2$ is a sentence (**conjunction**).
- If S_1 and S_2 are sentences, $S_1 \vee S_2$ is a sentence (**disjunction**).
- If S_1 and S_2 are sentences, $S_1 \Rightarrow S_2$ is a sentence (**implication**).
- If S_1 and S_2 are sentences, $S_1 \Leftrightarrow S_2$ is a sentence (**biconditional**).

Propositional Logic: Syntax

■ Order of precedence

From highest to lowest:

- Parenthesis (Sentence) or [Sentence]
- Not \neg
- And \wedge
- Or \vee
- Implies \Rightarrow
- Equivalent \Leftrightarrow

Propositional Logic: Syntax

A BNF (Backus-Naur Form) grammar of sentences in propositional Logic is defined by the following rules:

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

Propositional Logic: Syntax

■ Example sentences:

- P means “It is hot.”
- Q means “It is humid.”
- R means “It is raining.”
- $(P \wedge Q) \Rightarrow R$
“If it is hot and humid, then it is raining”
- $Q \Rightarrow P$
“If it is humid, then it is hot”
- A better way:
Hot = “It is hot”
Humid = “It is humid”
Raining = “It is raining”

Propositional logic: Semantics

- Semantics define the rules for determining the truth of a sentence with respect to a particular model.
 - Each model specifies the truth value (*true* or *false*) for each proposition symbol.
 - E.g. $P_{1,2}$ $P_{2,2}$ $P_{3,1}$
false *false* *true*
 - With these three symbols, 8 possible models, can be enumerated automatically.

■ Rules:

$\neg S$ is *true* iff S is *false*

$S_1 \wedge S_2$ is *true* iff S_1 is *true* and S_2 is *true*

$S_1 \vee S_2$ is *true* iff S_1 is *true* or S_2 is *true*

$S_1 \Rightarrow S_2$ is *true* iff S_1 is *false* or S_2 is *true*

i.e., is *false* iff S_1 is *true* and S_2 is *false*

$S_1 \Leftrightarrow S_2$ is *true* iff S_1 and S_2 are both *true* or both *false*

Truth Tables for Connectives

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

- Most sentences are sometimes true.

$$P \wedge Q$$

- Some sentences are always true (valid).

$$\neg P \vee P$$

- Some sentences are never true (unsatisfiable).

$$\neg P \wedge P$$

Simple Inference Procedure

- **Model-checking** (or **Enumeration**) is an inference approach that implement directly the definition of entailment:
 - Enumerate all possible models by assigning *true* or *false* to every proposition symbol.
 - Check that α is *true* in every model in which KB is *true*.
- This procedure is **sound** and **complete**.
- Example:
 - Let $\alpha = A \vee B$ and $KB = (A \vee C) \wedge (B \vee \neg C)$
 - Is it the case that $KB \models \alpha$?

Simple Inference Procedure

<i>A</i>	<i>B</i>	<i>C</i>	<i>KB</i> <i>$(A \vee C) \wedge (B \vee \neg C)$</i>	<i>α</i> <i>$A \vee B$</i>
<i>False</i>	<i>False</i>	<i>False</i>	<i>False</i>	<i>False</i>
<i>False</i>	<i>False</i>	<i>True</i>	<i>False</i>	<i>False</i>
<i>False</i>	<i>True</i>	<i>False</i>	<i>False</i>	<i>True</i>
<i>False</i>	<i>True</i>	<i>True</i>	<i>True</i>	<i>True</i>
<i>True</i>	<i>False</i>	<i>False</i>	<i>True</i>	<i>True</i>
<i>True</i>	<i>False</i>	<i>True</i>	<i>False</i>	<i>True</i>
<i>True</i>	<i>True</i>	<i>False</i>	<i>True</i>	<i>True</i>
<i>True</i>	<i>True</i>	<i>True</i>	<i>True</i>	<i>True</i>

Simple Inference Procedure

<i>A</i>	<i>B</i>	<i>C</i>	<i>KB</i> <i>$(A \vee C) \wedge (B \vee \neg C)$</i>	<i>α</i> <i>$A \vee B$</i>
<i>False</i>	<i>False</i>	<i>False</i>	<i>False</i>	<i>False</i>
<i>False</i>	<i>False</i>	<i>True</i>	<i>False</i>	<i>False</i>
<i>False</i>	<i>True</i>	<i>False</i>	<i>False</i>	<i>True</i>
<i>False</i>	<i>True</i>	<i>True</i>	<i>True</i>	<i>True</i>
<i>True</i>	<i>False</i>	<i>False</i>	<i>True</i>	<i>True</i>
<i>True</i>	<i>False</i>	<i>True</i>	<i>False</i>	<i>True</i>
<i>True</i>	<i>True</i>	<i>False</i>	<i>True</i>	<i>True</i>
<i>True</i>	<i>True</i>	<i>True</i>	<i>True</i>	<i>True</i>

Simple Inference Procedure

<i>A</i>	<i>B</i>	<i>C</i>	<i>KB</i> $(A \vee C) \wedge (B \vee \neg C)$	α $A \vee B$
False	False	False	False	False
False	False	True	False	False
False	True	False	False	True
False	True	True	True	True
True	False	False	True	True
True	False	True	False	True
True	True	False	True	True
True	True	True	True	True

$KB \models \alpha$

Wumpus World Sentences

- Let $P_{i,j}$ be *true* if there is a pit in $[i,j]$ and $B_{i,j}$ be *true* if there is a breeze in $[i,j]$.

- There is no pit in $[1,1]$:

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

- “Pits cause breezes in adjacent squares”. But, we include just the relevant squares:

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

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

- Now we include the breeze percepts for the first two squares visited by the agent:

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

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

Inference by Enumeration

$B_{1,1}$	$B_{2,1}$	$P_{1,1}$	$P_{1,2}$	$P_{2,1}$	$P_{2,2}$	$P_{3,1}$	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
\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots
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
\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots
true	true	true	true	true	true	true	false	true	true	false	true	false

$$\mathbf{KB} = R_1 \wedge R_2 \wedge R_3$$

$$\alpha_1 = \neg P_{1,2}$$

$$KB \models \alpha_1$$

$$\alpha_2 = \neg P_{2,2}$$

$$KB \not\models \alpha_2$$

$$\alpha_3 = P_{3,1}$$

$$KB \not\models a_3$$

AI: A modern Approach © 2010 S. Russell and P. Norving
By Dr. Khaled Wassif

Propositional Theorem Proving

- Proof methods are divided (roughly) into two kinds:
 - **Model checking**
 - » Truth table enumeration (sound and complete for propositional logic).
 - » Suitable only in case of small numbers of propositional symbols.
 - » For n symbols, the time complexity is $O(2^n)$.
 - **Application of inference rules**
 - » More efficient than model checking.
 - » Proof without consulting models by applying set of inference rules.
 - » Legal (sound) generation of new sentences from current sentences.
 - » Can use inference rules as operators in a standard search algorithm.
 - » Typically require transformation of sentences into a **normal form**.

Additional Entailment Concepts

- Two sentences α and β are **logically equivalent** $\alpha \equiv \beta$ iff each of them entails the other: $\alpha \models \beta$ and $\beta \models \alpha$

$$\begin{array}{ll} (\alpha \wedge \beta) \equiv (\beta \wedge \alpha) & \text{commutativity of } \wedge \\ (\alpha \vee \beta) \equiv (\beta \vee \alpha) & \text{commutativity of } \vee \\ ((\alpha \wedge \beta) \wedge \gamma) \equiv (\alpha \wedge (\beta \wedge \gamma)) & \text{associativity of } \wedge \\ ((\alpha \vee \beta) \vee \gamma) \equiv (\alpha \vee (\beta \vee \gamma)) & \text{associativity of } \vee \\ \neg(\neg\alpha) \equiv \alpha & \text{double-negation elimination} \\ (\alpha \Rightarrow \beta) \equiv (\neg\beta \Rightarrow \neg\alpha) & \text{contraposition} \\ (\alpha \Rightarrow \beta) \equiv (\neg\alpha \vee \beta) & \text{implication elimination} \\ (\alpha \Leftrightarrow \beta) \equiv ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)) & \text{biconditional elimination} \\ \neg(\alpha \wedge \beta) \equiv (\neg\alpha \vee \neg\beta) & \text{de Morgan} \\ \neg(\alpha \vee \beta) \equiv (\neg\alpha \wedge \neg\beta) & \text{de Morgan} \\ (\alpha \wedge (\beta \vee \gamma)) \equiv ((\alpha \wedge \beta) \vee (\alpha \wedge \gamma)) & \text{distributivity of } \wedge \text{ over } \vee \\ (\alpha \vee (\beta \wedge \gamma)) \equiv ((\alpha \vee \beta) \wedge (\alpha \vee \gamma)) & \text{distributivity of } \vee \text{ over } \wedge \end{array}$$

Additional Entailment Concepts

- A sentence is **valid** (or **tautology**) if it is true in all models:

e.g., *True*, $A \vee \neg A$, $A \Rightarrow A$, $(A \wedge (A \Rightarrow B)) \Rightarrow B$

- Validity is connected to inference via the **Deduction Theorem**:

$KB \models \alpha$ if and only if $(KB \Rightarrow \alpha)$ is valid

- 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 **contradiction**:

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

Inference Rules

- An inference rule is sound if the conclusion is true in all cases where the premises are true:

$$\frac{\alpha}{\beta} \quad \begin{array}{l} \text{Premise} \\ \text{Conclusion} \end{array}$$

– Modus Ponens

- » From an implication and the premise of the implication, you can infer the conclusion:

$$\frac{\alpha \Rightarrow \beta, \alpha}{\beta} \quad \begin{array}{l} \text{Premise} \\ \text{Conclusion} \end{array}$$

– Modus Tollens

- » From an implication and the conclusion negation of the implication, you can infer the premise negation.

$$\frac{\alpha \Rightarrow \beta, \neg \beta}{\neg \alpha} \quad \begin{array}{l} \text{Premise} \\ \text{Conclusion} \end{array}$$

Inference Rules

– And-Elimination

» From a conjunction, you can infer any of the conjuncts.

$$\frac{\alpha_1 \wedge \alpha_2 \wedge \dots \wedge \alpha_n \quad \text{Premise}}{\alpha_i} \quad \text{Conclusion}$$

– And-Introduction

» From a list of sentences, you can infer their conjunction.

$$\frac{\alpha_1, \alpha_2, \dots, \alpha_n \quad \text{Premise}}{\alpha_1 \wedge \alpha_2 \wedge \dots \wedge \alpha_n} \quad \text{Conclusion}$$

– Or-Introduction

» From a sentence, you can infer its disjunction with anything else at all.

$$\frac{\alpha_i \quad \text{Premise}}{\alpha_1 \vee \alpha_2 \vee \dots \vee \alpha_n} \quad \text{Conclusion}$$

Inference Rules

– Unit Resolution

- » From a disjunction, if one of the disjuncts is false, then you can infer the other one is true.

$$\frac{\alpha \vee \beta, \neg\beta}{\alpha} \quad \begin{array}{l} \text{Premise} \\ \text{Conclusion} \end{array}$$

– Resolution

$$\frac{\alpha \vee \beta, \neg\beta \vee \gamma}{\alpha \vee \gamma} \quad \begin{array}{l} \text{Premise} \\ \text{Conclusion} \end{array}$$

or equivalently

$$\frac{\neg\alpha \Rightarrow \beta, \beta \Rightarrow \gamma}{\neg\alpha \Rightarrow \gamma} \quad \begin{array}{l} \text{Premise} \\ \text{Conclusion} \end{array}$$

Inference in Wumpus World

- Let $S_{i,j}$ be true if there is a stench in cell $[i,j]$.
- Let $B_{i,j}$ be true if there is a breeze in cell $[i,j]$.
- Let $W_{i,j}$ be true if there is a Wumpus in cell $[i,j]$.
- Let $P_{i,j}$ be true if there is a Pit in cell $[i,j]$.

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

Inference in Wumpus World

■ Given:

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

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

■ Let's make some inferences:

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

(By definition of the biconditional)

$$2. (P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1} \text{ (And-elimination)}$$

$$3. \neg B_{1,1} \Rightarrow \neg(P_{1,2} \vee P_{2,1}) \text{ (equivalence with contrapositive)}$$

$$4. \neg(P_{1,2} \vee P_{2,1}) \text{ (modus ponens)}$$

$$5. \neg P_{1,2} \wedge \neg P_{2,1} \text{ (DeMorgan's rule)}$$

$$6. \neg P_{1,2} \text{ (And Elimination)}$$

Inference in Wumpus World

Initial KB

Percept Sentences

$\neg S_{1,1}$	$\neg B_{1,1}$
$\neg S_{2,1}$	$B_{2,1}$
$S_{1,2}$	$\neg B_{1,2}$
...	

Environment Knowledge

$R_{11}: \neg S_{1,1} \Rightarrow \neg W_{1,1} \wedge \neg W_{2,1} \wedge \neg W_{1,2}$
 $R_{12}: S_{1,2} \Rightarrow W_{1,2} \vee W_{1,1} \vee W_{2,2} \vee W_{1,3}$
 $R_{13}: \neg B_{1,1} \Rightarrow \neg P_{1,1} \wedge \neg P_{2,1} \wedge \neg P_{1,2}$
 $R_{14}: B_{1,2} \Rightarrow P_{1,1} \vee P_{1,2} \vee P_{2,2} \vee P_{1,3}$
...

Some inferences:

Apply Modus Ponens to R_{11}

Add to KB

$$\neg W_{1,1} \wedge \neg W_{2,1} \wedge \neg W_{1,2}$$

Apply to this And-Elimination

Add to KB

$$\neg W_{1,1}$$

$$\neg W_{2,1}$$

$$\neg W_{1,2}$$

Inference in Wumpus World

- Recall that when we were at [2,1] we could not decide on a safe move, so we backtracked, and explored [1,2], which yielded $\neg B_{1,2}$.

$$\neg B_{1,2} \Leftrightarrow \neg P_{1,1} \wedge \neg P_{1,3} \wedge \neg P_{2,2}$$

this yields to $\neg P_{1,1} \wedge \neg P_{1,3} \wedge \neg P_{2,2}$

and consequently $\neg P_{1,1}$, $\neg P_{1,3}$, $\neg P_{2,2}$

Inference in Wumpus World

- Now we can consider the implications of $B_{2,1}$ and **prove by resolution**:

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

2. $B_{2,1} \Rightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$

(biconditional Elimination)

3. $P_{1,1} \vee P_{2,2} \vee P_{3,1}$ (modus ponens)

4. $P_{1,1} \vee P_{1,3}$ (resolution rule because no pit in [2,2])

5. $P_{3,1}$ (resolution rule because no pit in [1,1])

Conversion to CNF

- The resolution rule applies only to knowledge bases and queries consisting of **clauses** (disjunctions of literals).
- A sentence expressed as a conjunction of clauses is said to be in **Conjunctive Normal Form** (CNF).
 - E.g., $(A \vee \neg B) \wedge (B \vee \neg C \vee \neg D)$
- Every sentence of propositional logic is logically equivalent to a conjunction of clauses.
- Converting the propositional sentences into CNF allows using the resolution rule as part of a complete inference procedure for all of propositional logic.

Conversion to CNF: Example

■ Converting $B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$ into CNF:

1. Eliminate \Leftrightarrow by 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 and double-negation rules, replacing $\neg(\alpha \vee \beta)$ with $(\neg\alpha \wedge \neg\beta)$ and $\neg(\neg\alpha)$ with α :

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

4. Apply distributivity law (distributing \vee over \wedge) and flatten, replacing $(\alpha \vee (\beta \wedge \gamma))$ with $((\alpha \vee \beta) \wedge (\alpha \vee \gamma))$:

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

Proof by Contradiction

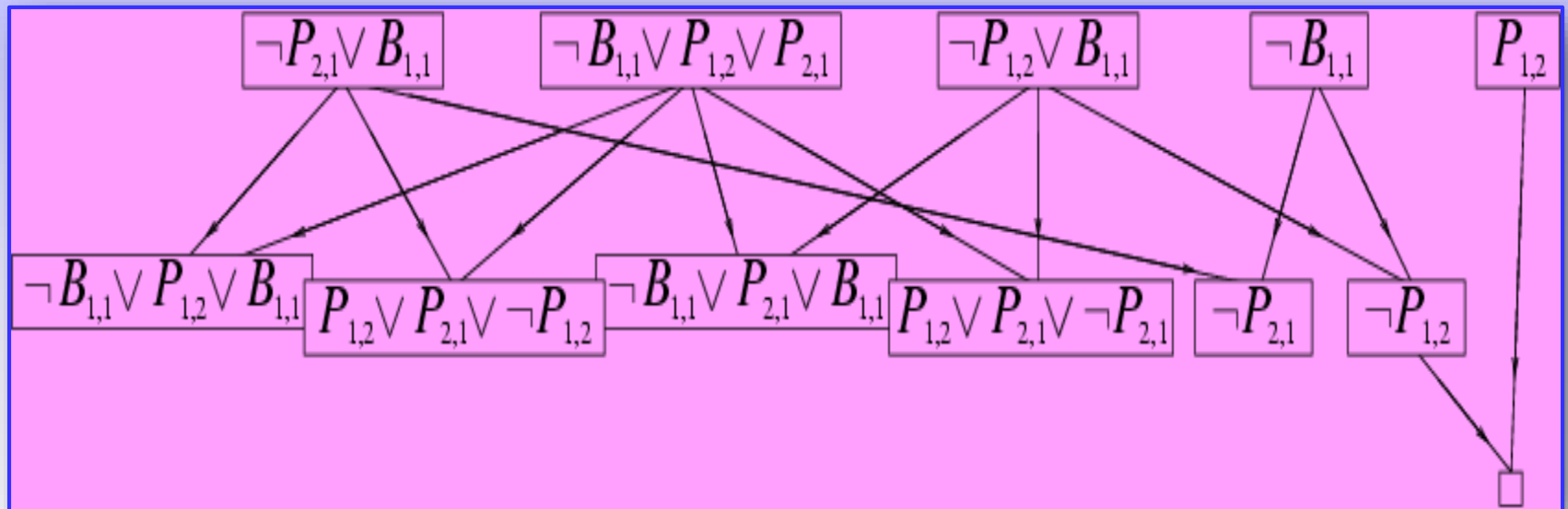
- Inference procedures based on resolution work by using the principle of proof by contradiction.
 - Show that $KB \models \alpha$ by showing that $(KB \wedge \neg\alpha)$ is unsatisfiable.
 1. First, convert $(KB \wedge \neg\alpha)$ into CNF.
 2. Then, apply the resolution rule to the resulting clauses.
 - » Each pair that contains complementary literals is resolved to produce a new clause, which is added to the set if it is not already present.
 3. The process continues until one of two things happens:
 - » There are no new clauses that can be added
 -  In which case KB does not entail α .
 - » Two clauses resolve to yield the empty clause
 -  In which case KB entails α .

Proof by Contradiction: Example

- $KB = R_2 \wedge R_4$

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

- $\alpha = \neg P_{1,2}$



- Then $KB \not\models \alpha$ (or $R_2 \wedge R_4 \Rightarrow \neg P_{1,2}$)

Definite Clauses and Horn Clauses

- Some real-world knowledge bases satisfy certain restrictions on the form of sentences they contain:
 - **Definite clauses** are clauses with exactly one positive literal.
 - » E.g., the clause $\neg\alpha \vee \neg\beta \vee \gamma$ is a definite clause.
 - **Horn clauses** are clauses with at most one positive literal.
- All definite clauses are Horn clauses.
 - Clauses with no positive literals are called **goal clauses**.
- Horn clauses are closed under resolution:
 - If two Horn clauses are resolved, we get back a Horn clause.

Definite Clauses and Horn Clauses

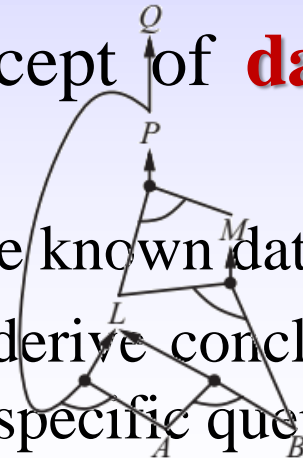
- Knowledge bases containing only definite clauses are interesting for three reasons:
 - Every definite clause can be written as an implication whose premise is a conjunction of positive literals and whose conclusion is a single positive literal.
 - » E.g., the clause $\neg\alpha \vee \neg\beta \vee \gamma$ can be written as $\alpha \wedge \beta \Rightarrow \gamma$.
 - » In Horn form, the premise is called the **body** and the conclusion is called the **head** – like Prolog.
 - » A sentence consisting of a single positive literal only is called a **fact**.
 - Inference with Horn clauses can be done through the **forward-chaining** and **backward chaining** algorithms.
 - Deciding entailment with Horn clauses can be done in time that is *linear* in the size of the knowledge base.

CNF & Definite and Horn Clauses Grammar

$$CNFSentence \rightarrow Clause_1 \wedge \dots \wedge Clause_n$$
$$Clause \rightarrow Literal_1 \vee \dots \vee Literal_m$$
$$Literal \rightarrow Symbol \mid \neg Symbol$$
$$Symbol \rightarrow P \mid Q \mid R \mid \dots$$
$$HornClauseForm \rightarrow DefiniteClauseForm \mid GoalClauseForm$$
$$DefiniteClauseForm \rightarrow (Symbol_1 \wedge \dots \wedge Symbol_l) \Rightarrow Symbol$$
$$GoalClauseForm \rightarrow (Symbol_1 \wedge \dots \wedge Symbol_l) \Rightarrow False$$

Forward Chaining

- To determine if a query is entailed by a knowledge base (KB) of definite clauses (implications):
 - Begin from known facts (positive literals) in KB .
 - If all premises of an implication (rule) are known, then its conclusion is added to the set of known facts.
 - This process continues until the query is added or no further inferences can be made.
- An example of the general concept of **data-driven** reasoning:
 - The focus of attention starts with the known data.
 - It can be used within an agent to derive conclusions from incoming percepts, often without a specific query in mind.



Forward Chaining Example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

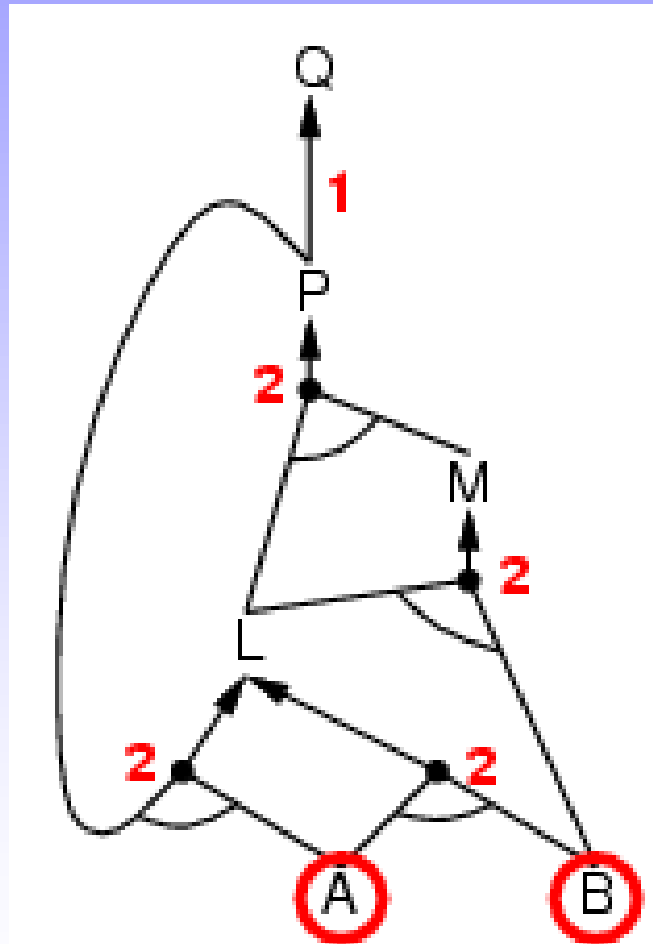
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



Forward Chaining Example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

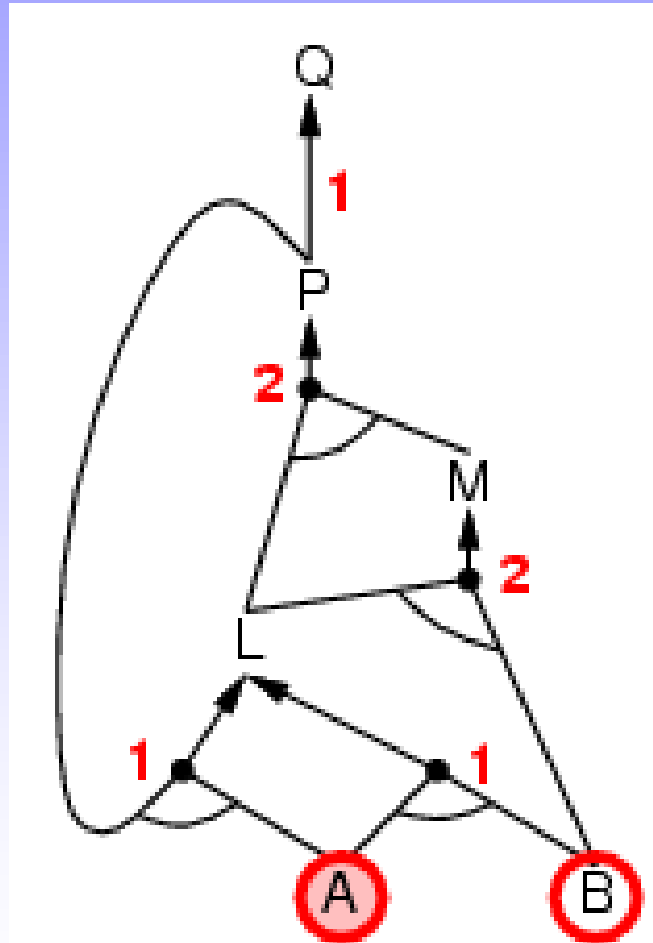
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



Forward Chaining Example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

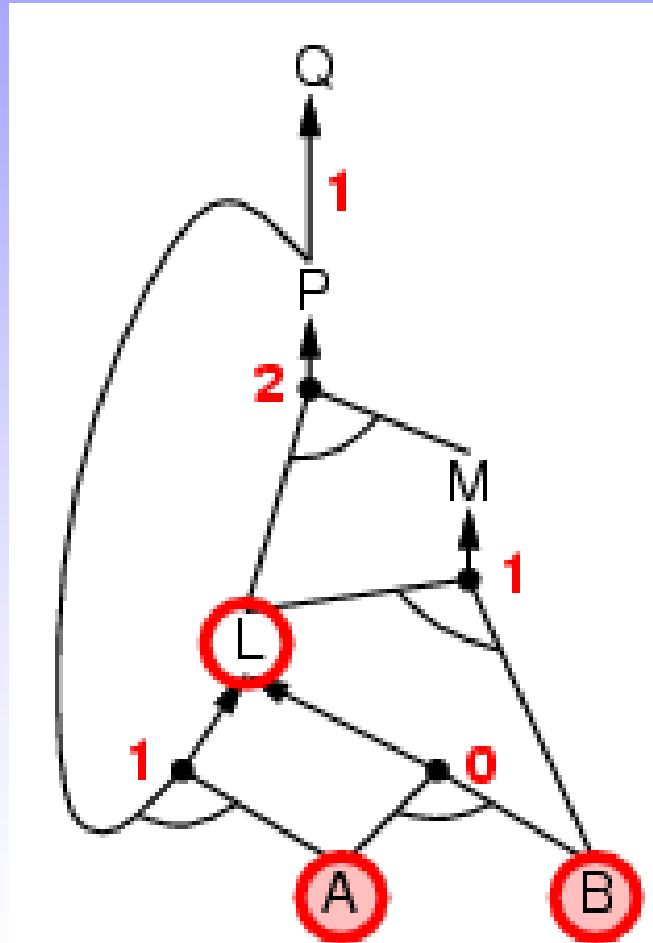
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

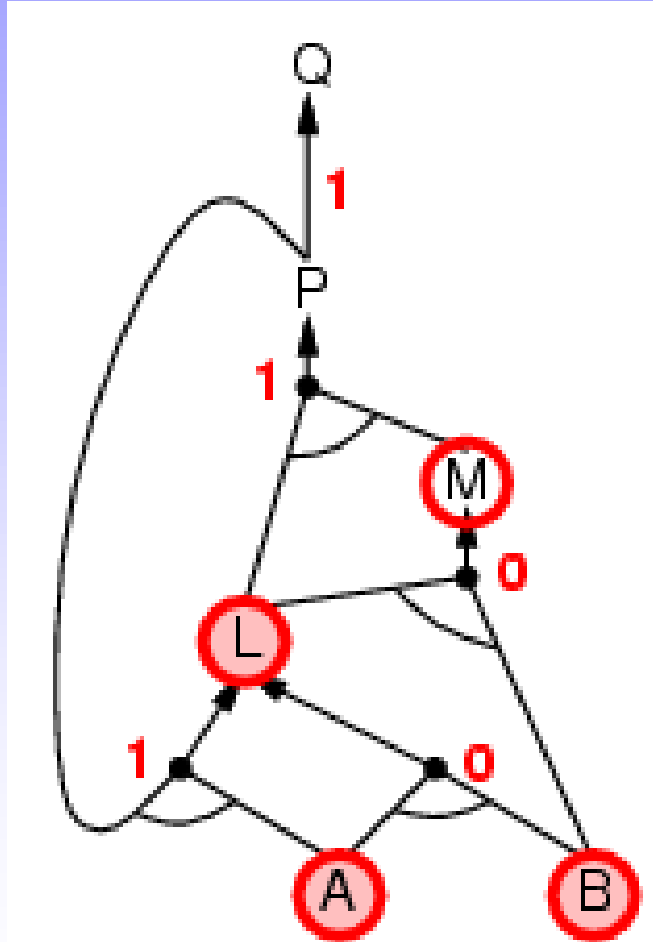
B



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

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

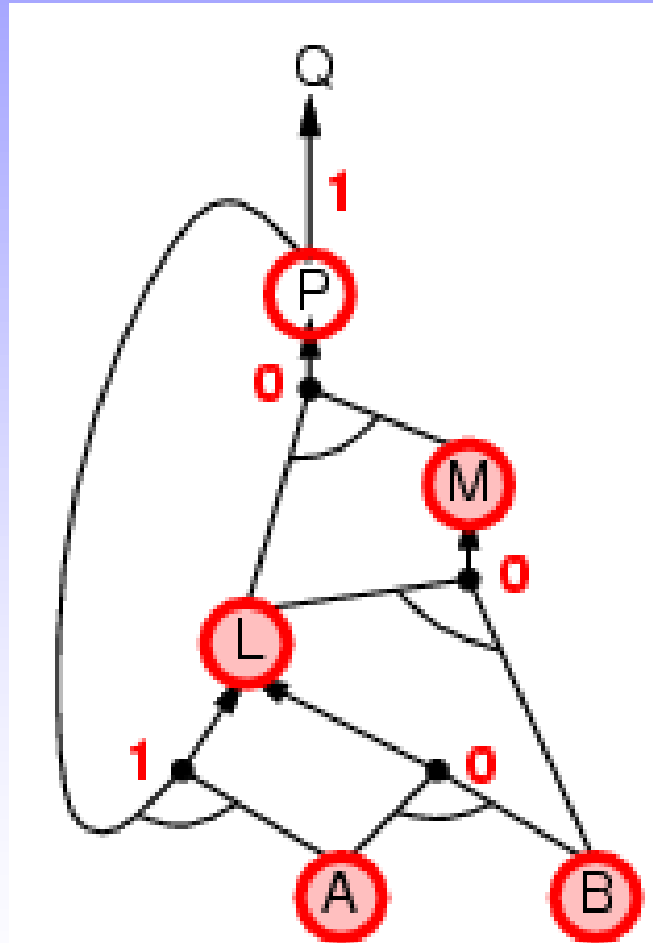
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



Forward Chaining Example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

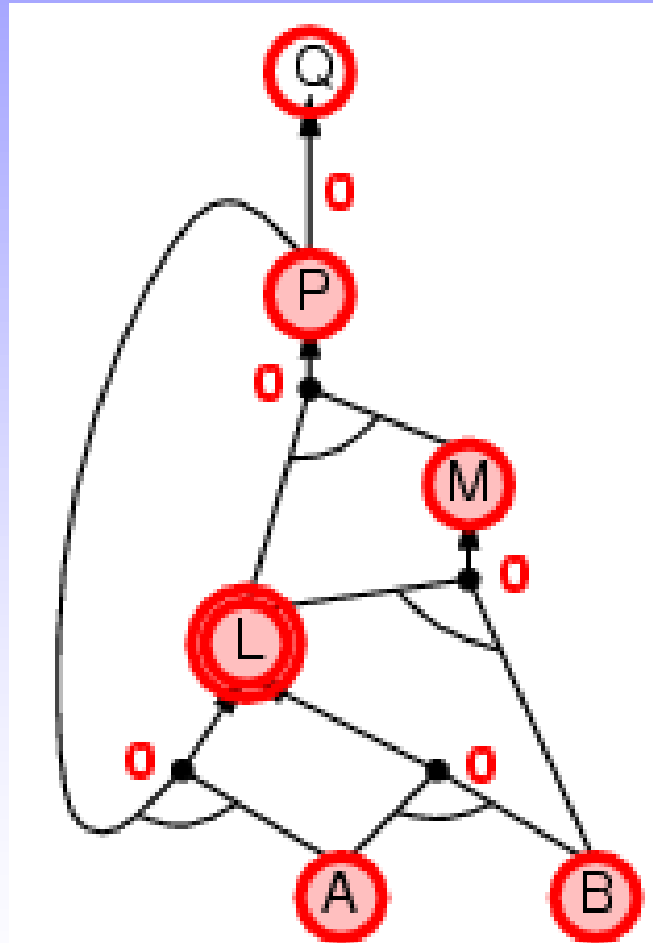
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



Forward Chaining Example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

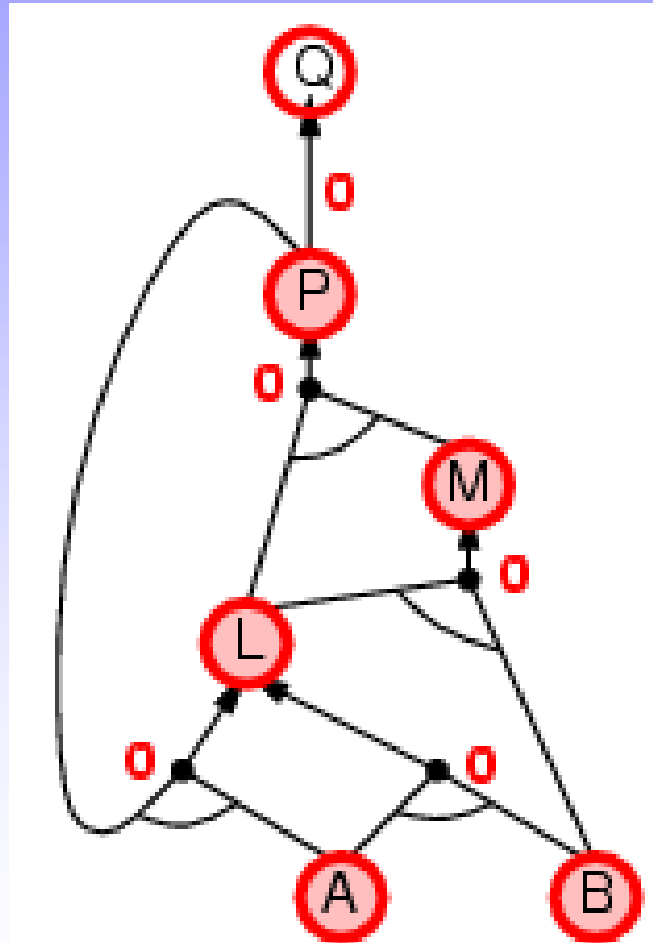
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

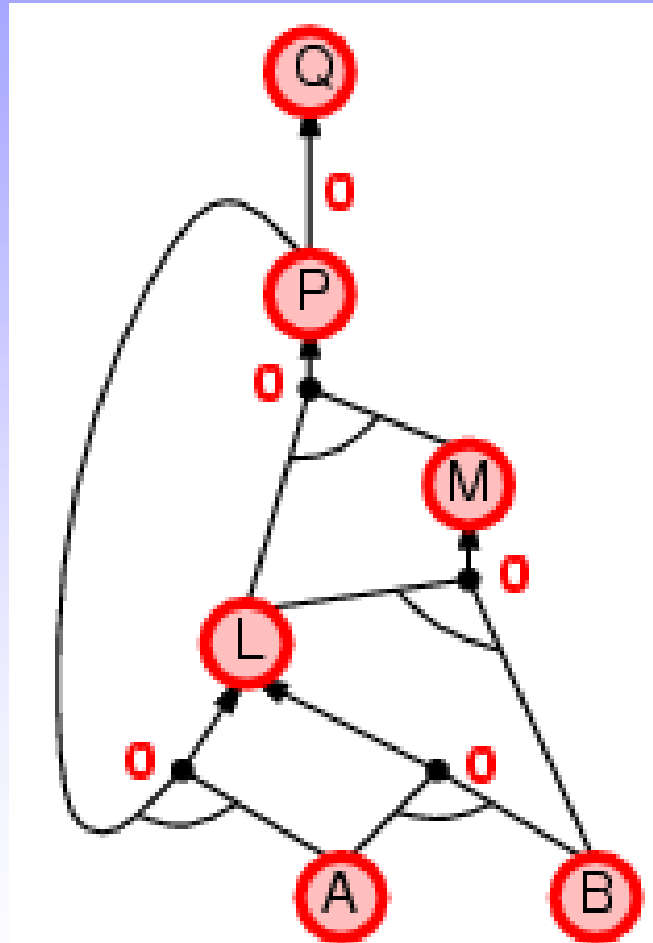
$A \wedge B \Rightarrow L$

A

B



Forward Chaining Example

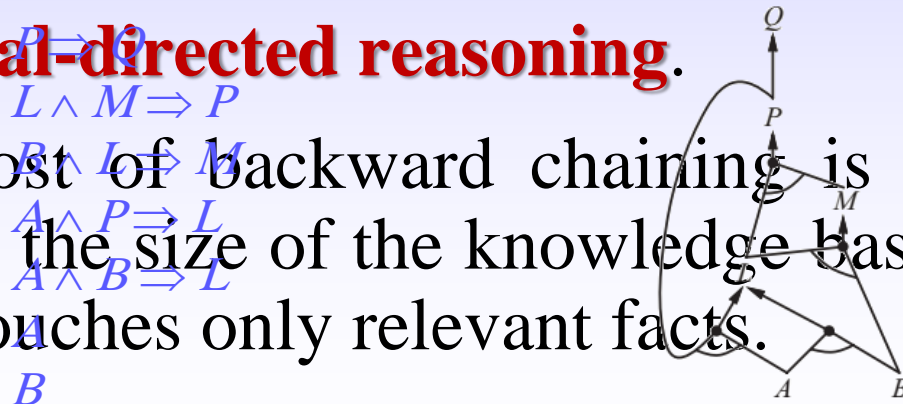


Backward chaining

- As its name suggests, work backward from the query:
 - If the query is known to be true, then no work is needed.
 - Otherwise, find those implications in the knowledge base whose conclusion is the query.
 - If all the premises of one of those implications can be proved true (by backward chaining), then the query is true.
 - It works backward until it reaches a set of known facts.

- A form of **goal-directed reasoning**.

- Often, the cost of backward chaining is *much less* than linear in the size of the knowledge base, because the process touches only relevant facts.



Backward Chaining Example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

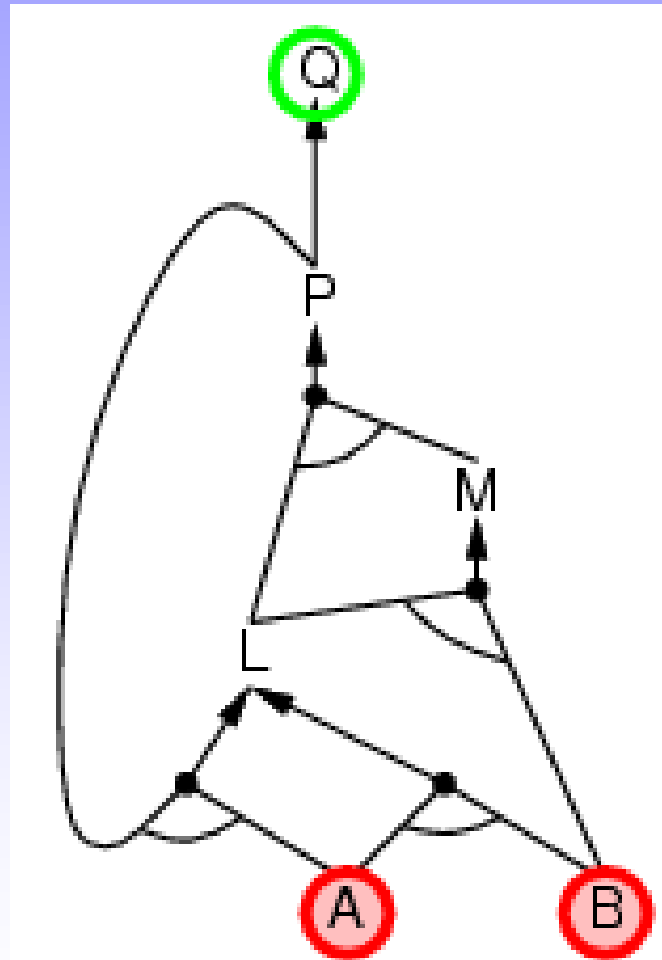
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



Backward Chaining Example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

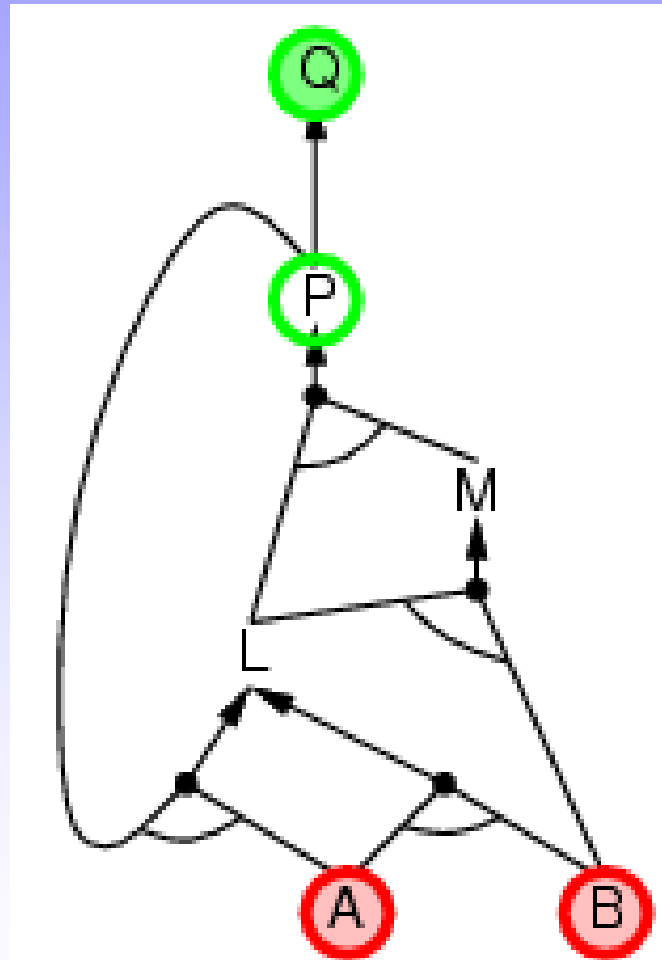
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



Backward Chaining Example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

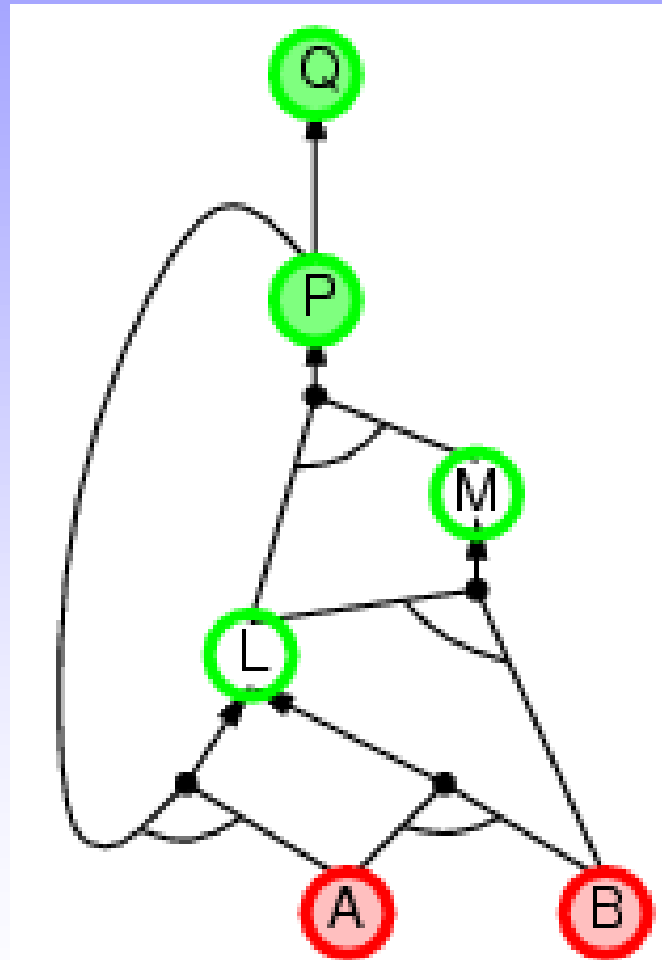
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



Backward Chaining Example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

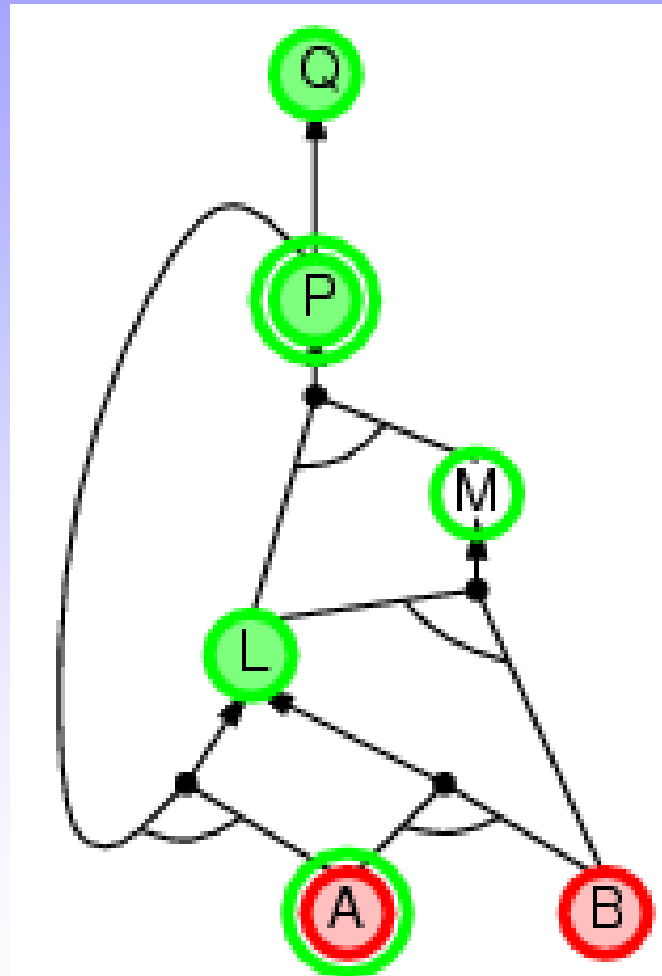
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



Backward Chaining Example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

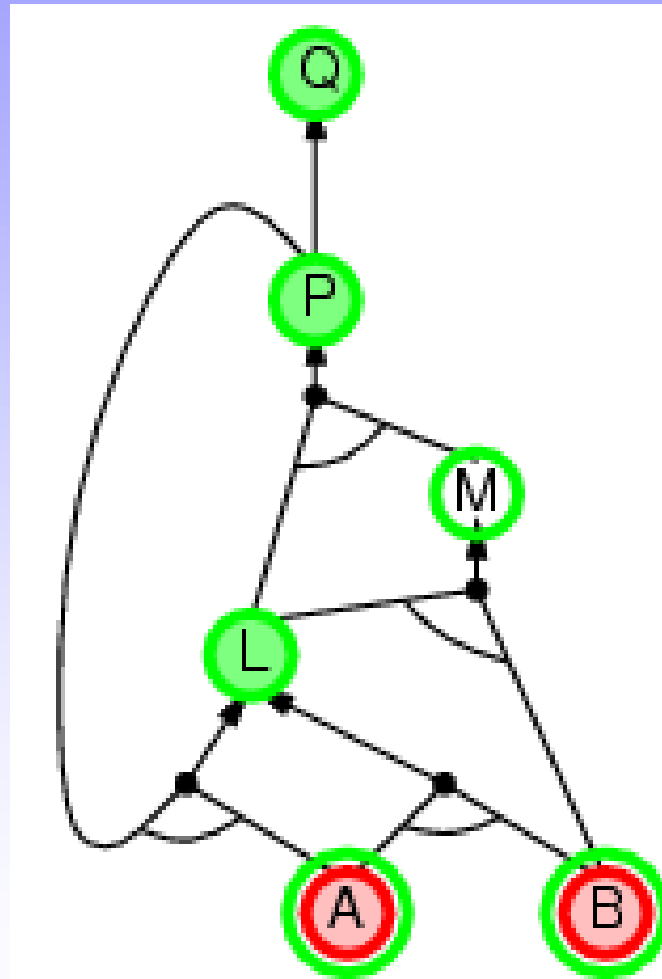
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



Backward Chaining Example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

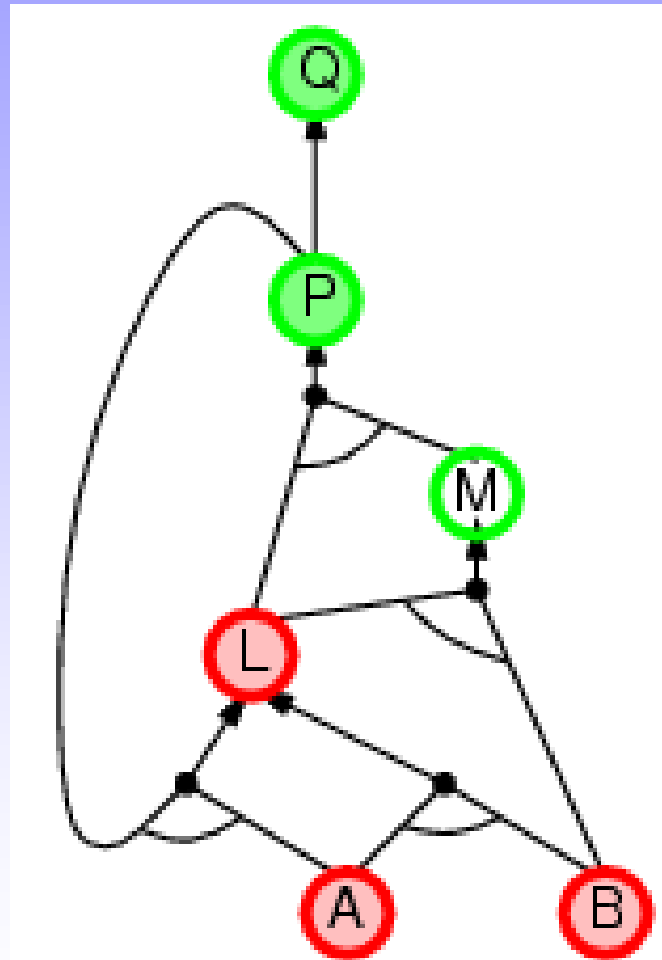
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



Backward Chaining Example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

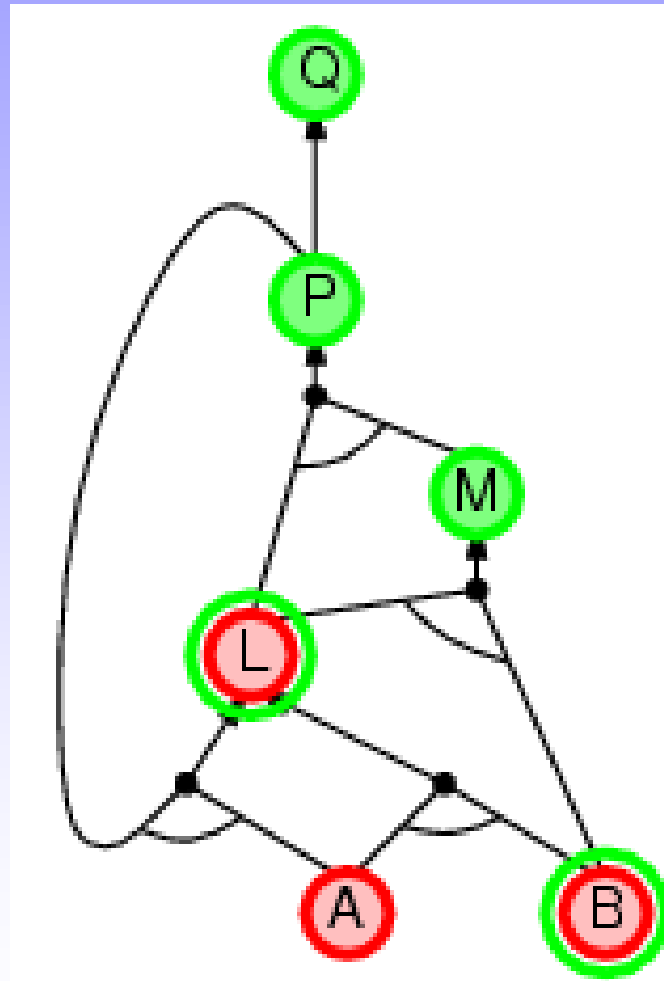
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



Backward Chaining Example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

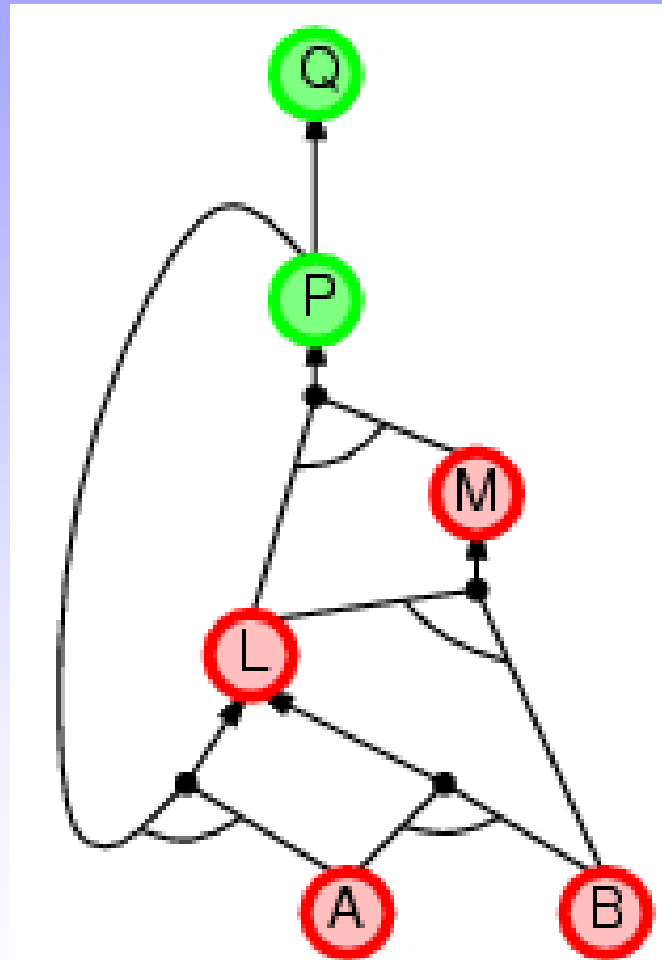
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



Backward Chaining Example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

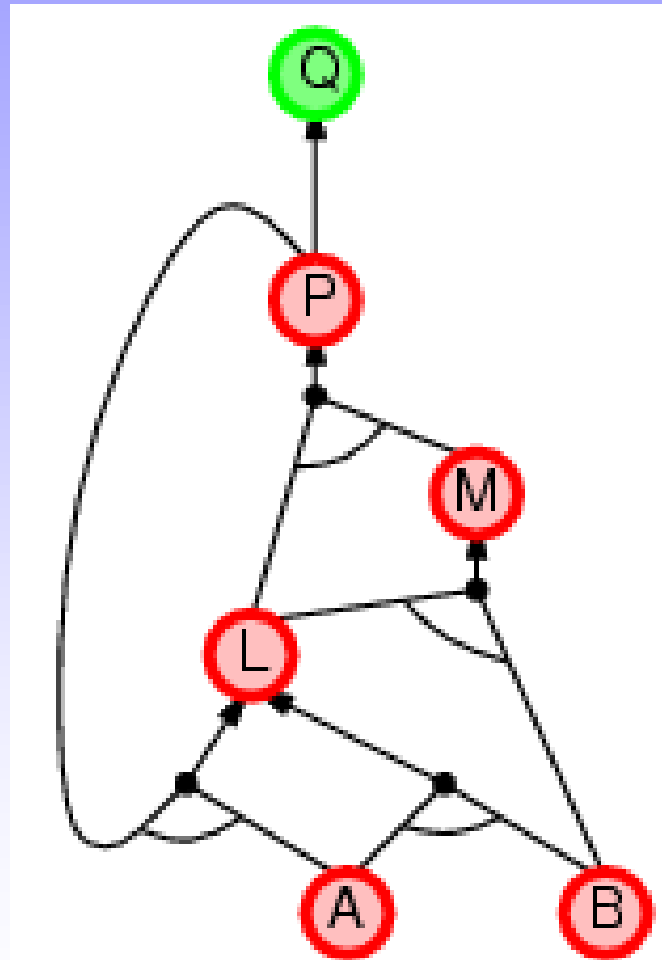
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

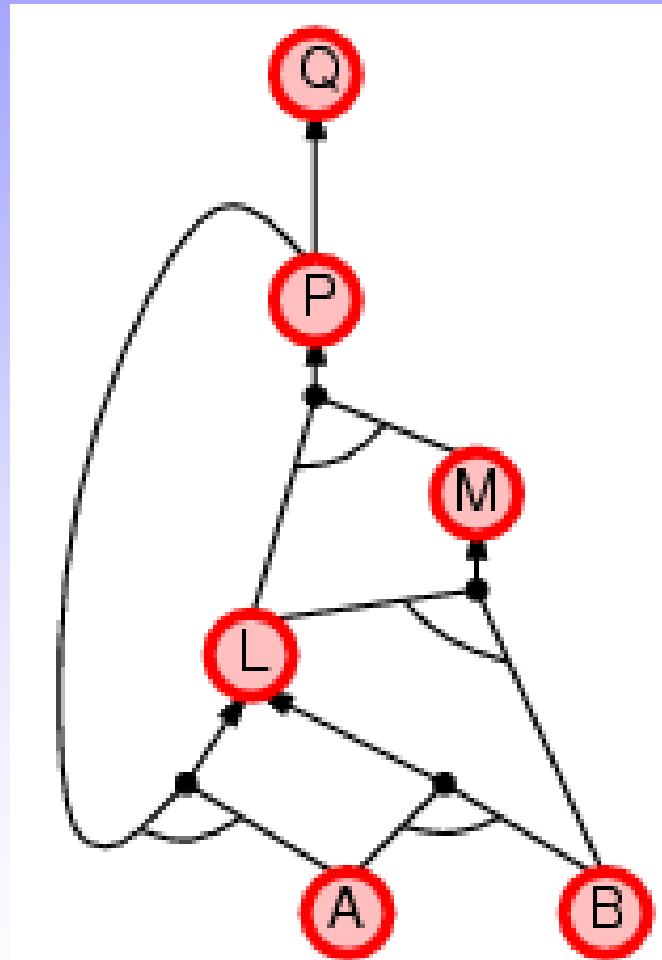
$A \wedge B \Rightarrow L$

A

B



Backward Chaining Example



SUMMARY

- Logical agents apply inference to a knowledge base to derive new information and make decisions.
- Basic concepts of logic:
 - **syntax**: formal structure of sentence.
 - **semantics**: truth of sentences models.
 - **entailment**: necessary truth of one sentence given another.
 - **inference**: deriving sentences from other sentences.
 - **soundness**: derivations produce only entailed sentences.
 - **completeness**: derivations can produce all entailed sentences.
- Wumpus world requires the ability to represent partial and negated information, reason by cases, etc.
- Resolution is complete inference for propositional logic.
- Inference can be done through forward or backward chaining algorithms.