Logical Agent

CCAI 221: AI fundaments

Assembled from:

- Artificial intelligence: a modern approach, Russel and Norving (pearson.com)
-

Amended by Shahd Alahdal

Pearson

Outline:

- Knowledge-based agents
- Wumpus world
- Logic in general—models and entailment
- Propositional (Boolean) logic
- Inference rules and theorem proving
  - forward chaining
  - backward chaining (SELF LEARNING)
- Propositional logic examples
- First-order (Predicate) logic (SELF LEARNING)

Knowledge based agent

- In which we design agents that can **form** **representations** of a complex world, use a process of **inference** to derive **new representations** about the world, and use these new representations to **deduce** what to do.

Knowledge bases

Inference engine	⟵	**domain−independent algorithms**
Knowledge base	⟵	**domain−specific content**

- Knowledge base = set of sentences in a formal language called a knowledge representation language.
- Declarative approach to building an agent (or other system):
  - Tell it what it needs to know
- Then it can Ask itself what to do—answers should follow from the KB

A simple knowledge-based agent

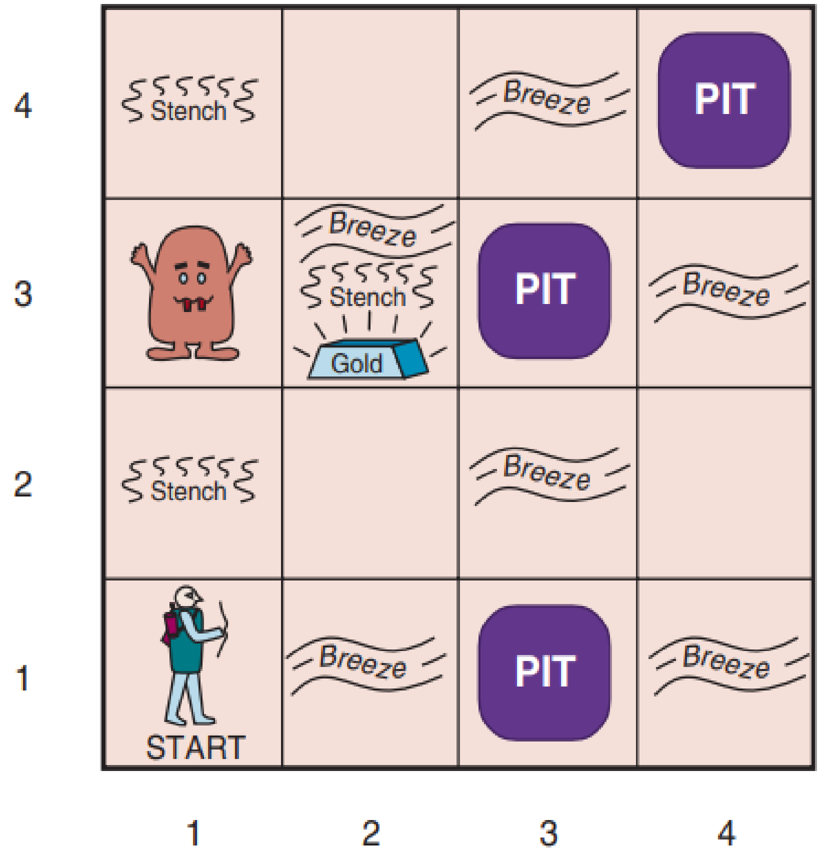
```
function KB-Agent( percept) returns an action
   static: KB, a knowledge base
           t, a counter, initially 0, indicating time
   Tell(KB, Make-Percept-Sentence( percept, t))
   action ← Ask(KB, Make-Action-Query(t))
   Tell(KB, Make-Action-Sentence(action, t))
   t ← t + 1
   return action
```

- The agent must be able to: Represent states, actions, etc.
  - Incorporate (enter) new percepts
  - Update internal representations of the world
  - Deduce hidden properties of the world
  - Deduce appropriate actions

Wumpus World

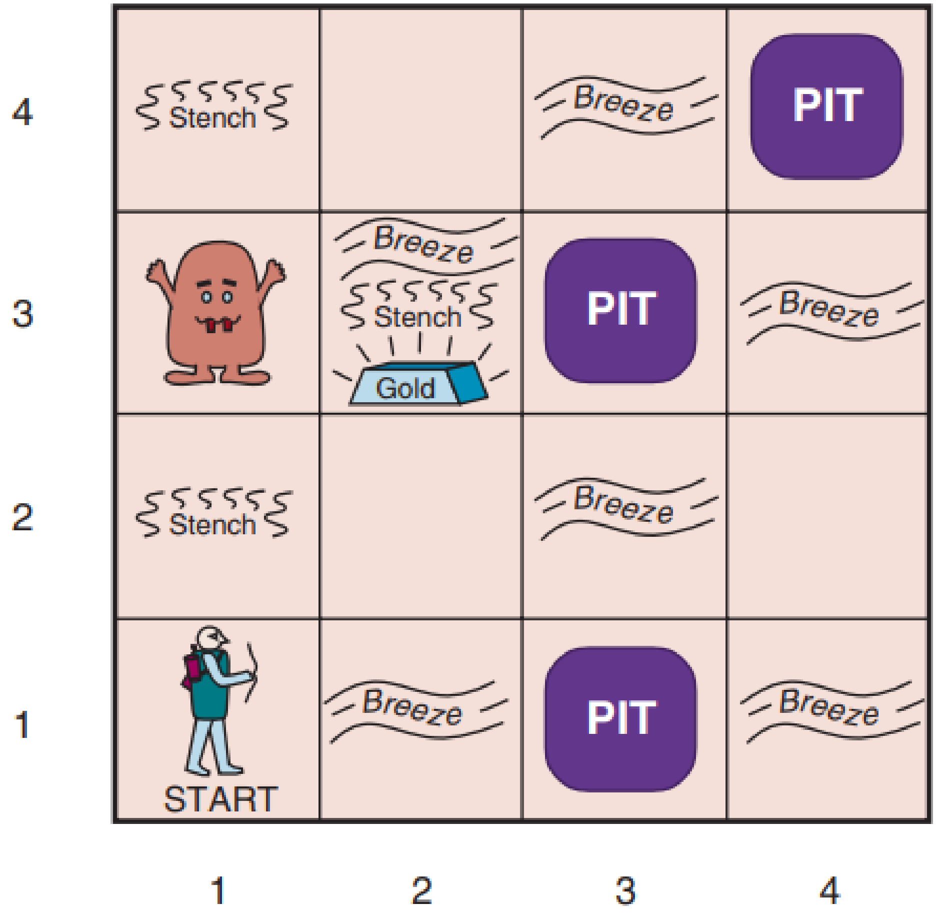
The Wumpus World

- The Wumpus World is 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 (holes) that will trap anyone who wanders into these rooms (except for the wumpus, which is too big to fall in).
- The only mitigating feature of this bleak environment is the possibility of finding a heap of gold.



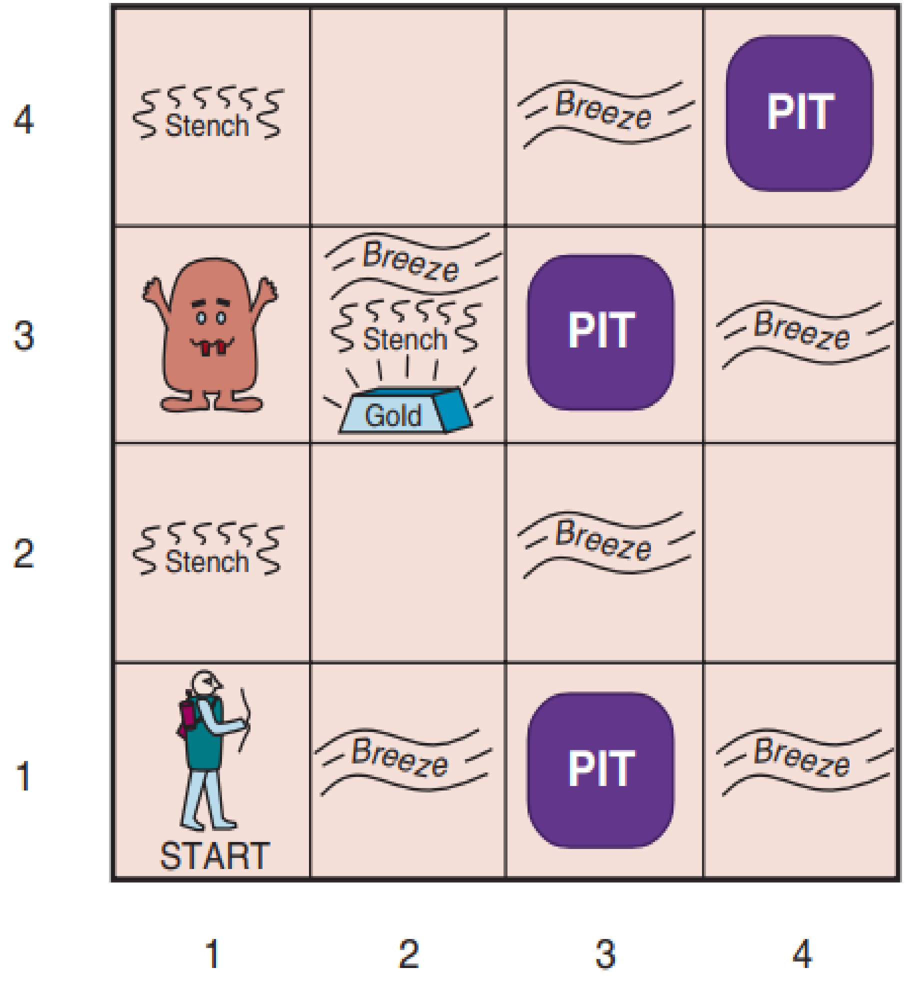
Wumpus World PEAS discretion

- Performance measure
  - gold +1000, death -1000
- -1 per step, -10 for using the arrow
- Environment
  - A 4 × 4 grid of rooms.
  - The agent always starts in the square labeled [1,1], facing to the right.
  - The locations of the gold and the wumpus are chosen randomly, from the squares other than the start square.



Wumpus World PEAS discretion

- **Sensors**
  - The agent has five sensors, each of which gives a single bit of information
    - Squares adjacent to wumpus are smelly (perceive a “stench”)
    - Squares adjacent to the pit are "breezy"
    - “Glitter" if gold is in the same square
    - “Bump” if it walks into the wall
    - “A scream” if the Wumpus is killed
- **Actuators** to help a gent to move: Forward, Left turn by 90◦, Right turn by 90◦, Grab, and Shoot
  - Shooting kills Wumpus if you are facing it. Shooting uses up the only arrow
  - Grabbing picks up gold if in the same square



Wumpus World characteristics

Observable??

Wumpus World characteristics

Observable?? No—only local perception

Deterministic??

Wumpus World characteristics

Observable?? No—only local perception

Deterministic?? Yes—outcomes exactly specified

Episodic??

Wumpus World characteristics

Observable?? No—only local perception

Deterministic?? Yes—outcomes exactly specified

Episodic?? No—sequential at the level of actions

Static??

Wumpus World characteristics

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??

Wumpus World characteristics

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??

Wumpus World characteristics

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

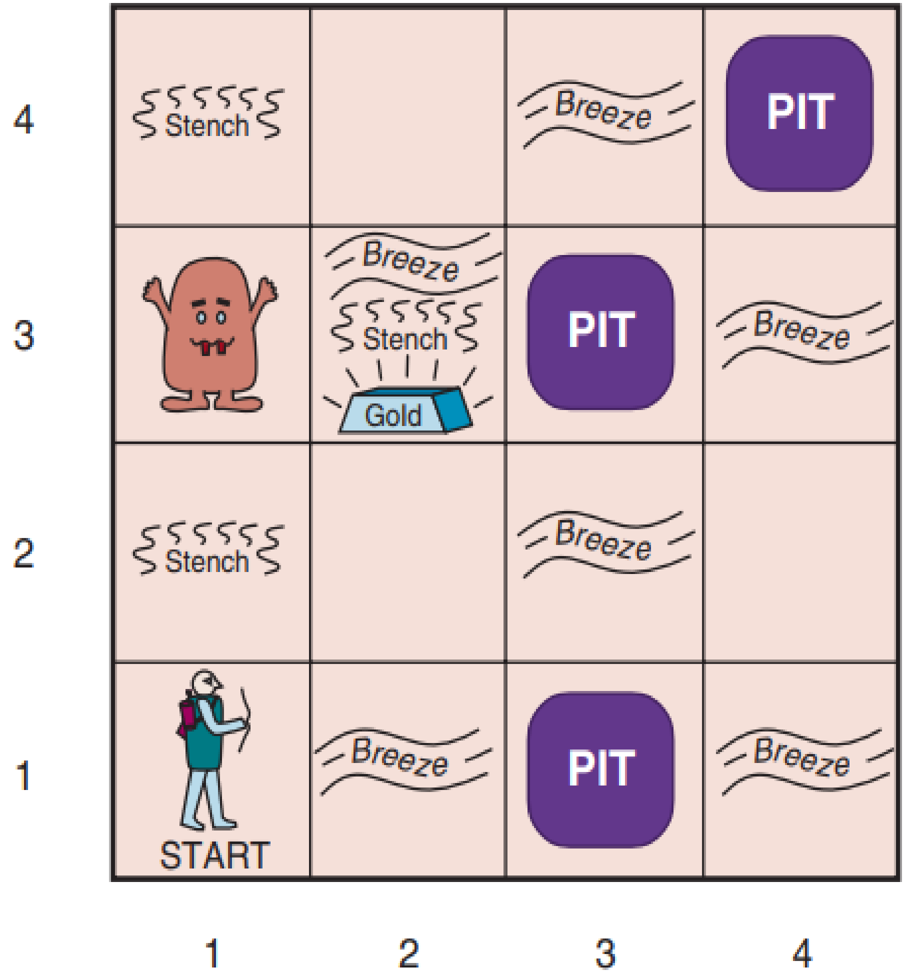
Discrete?? Yes

Single-agent?? Yes—Wumpus is essentially a natural feature

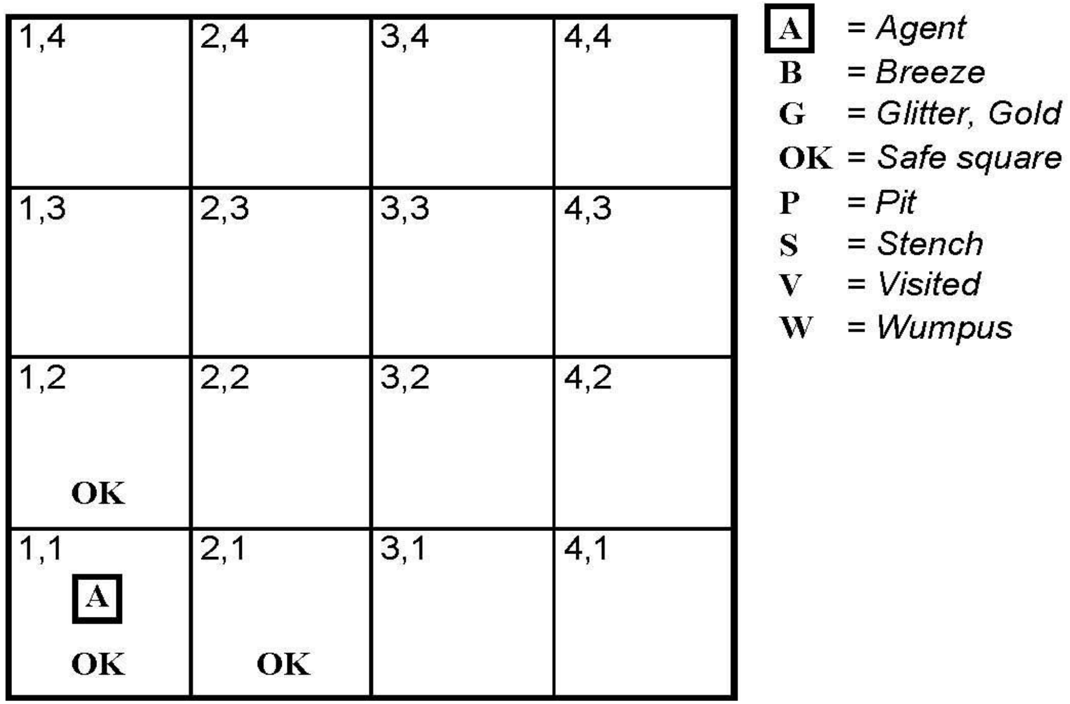
Wumpus World

- Percepts given to the agent
  1. Stench
  2. Breeze
  3. Glitter
  4. Bumb (ran into a wall)
  5. Scream (wumpus has been hit by an arrow)

- **Principle Difficulty**: the agent is initially ignorant of the configuration of the environment – going to have to reason to figure out where the gold is without getting killed!



Exploring the Wumpus World



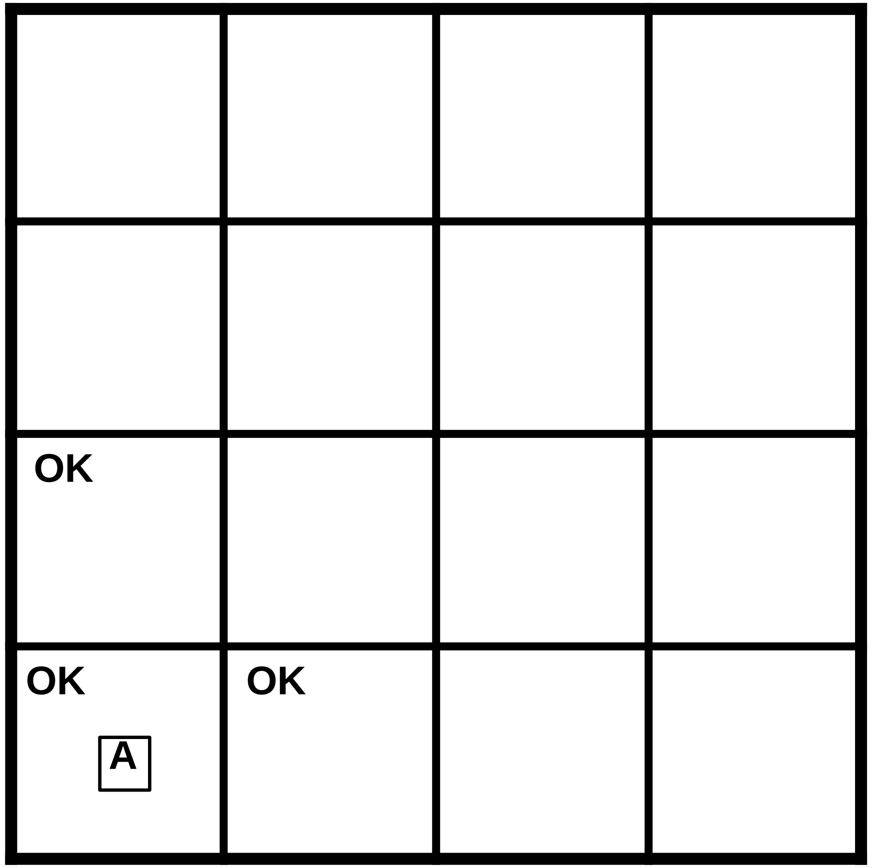
(a)

- **Initial situation:**
- **Agent in 1,1 and percept is [None, None, None, None, None]**
- **From this, the agent can infer the neighboring squares are safe (otherwise there would be a breeze or a stench)**

Agent always starts in square [1,1] facing right. Locations of the gold and the wumpus are chosen randomly

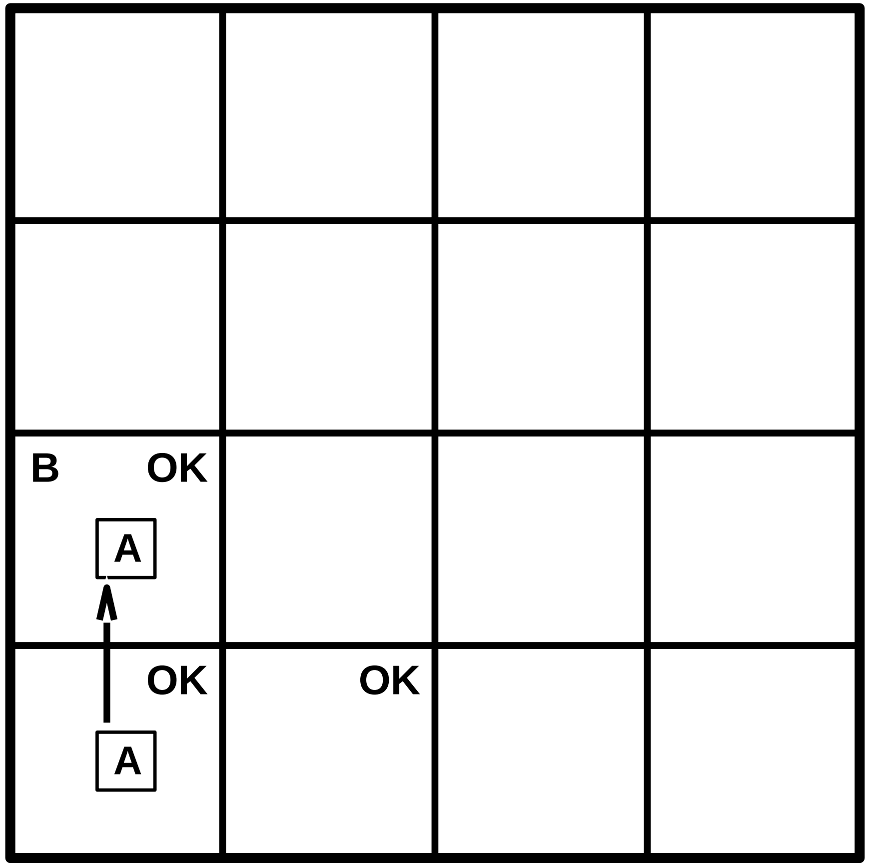
Note the agent perceives the environment - **percept** looks like **[stench, breeze, glitter, bump, scream]**

Exploring Wumpus World



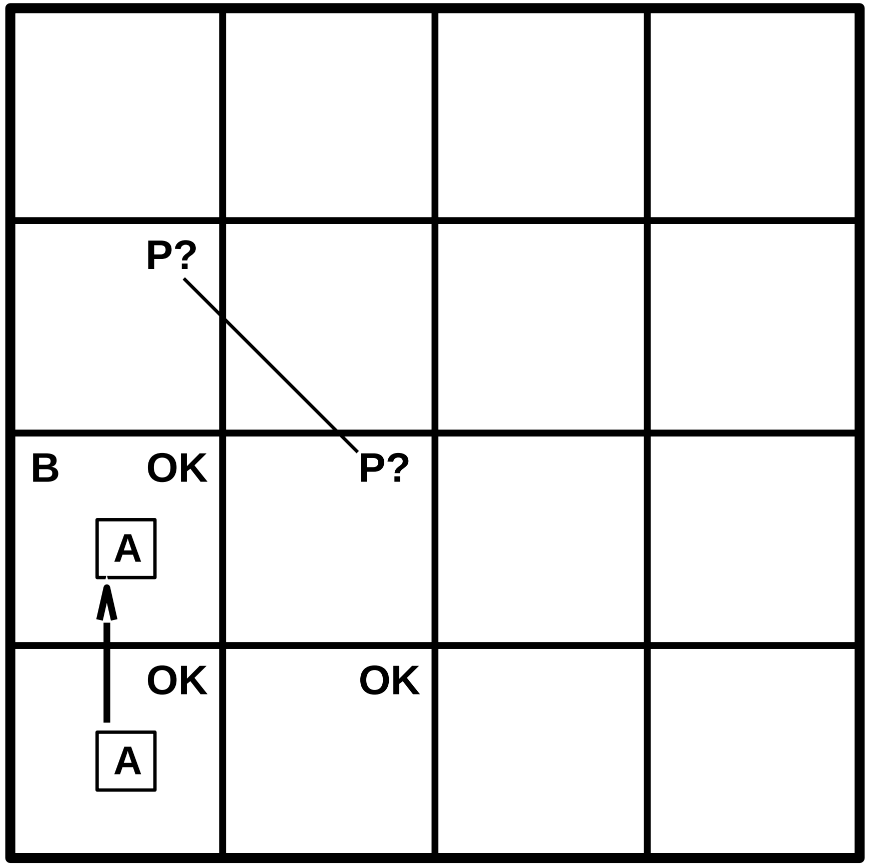
Exploring Wumpus World

- The agent decides to move up and feels a breeze – **what does this mean?**



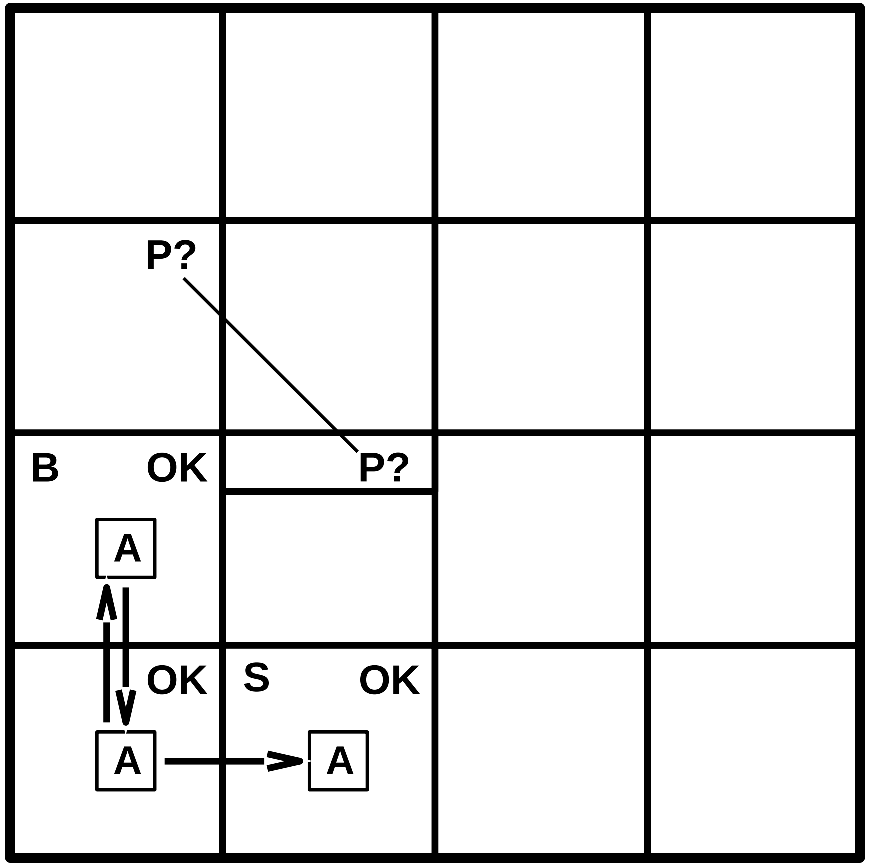
Exploring Wumpus World

- That there is a pit in either of the two neighboring squares.
- Better not move there since it knows there is a safer move if it backs up…



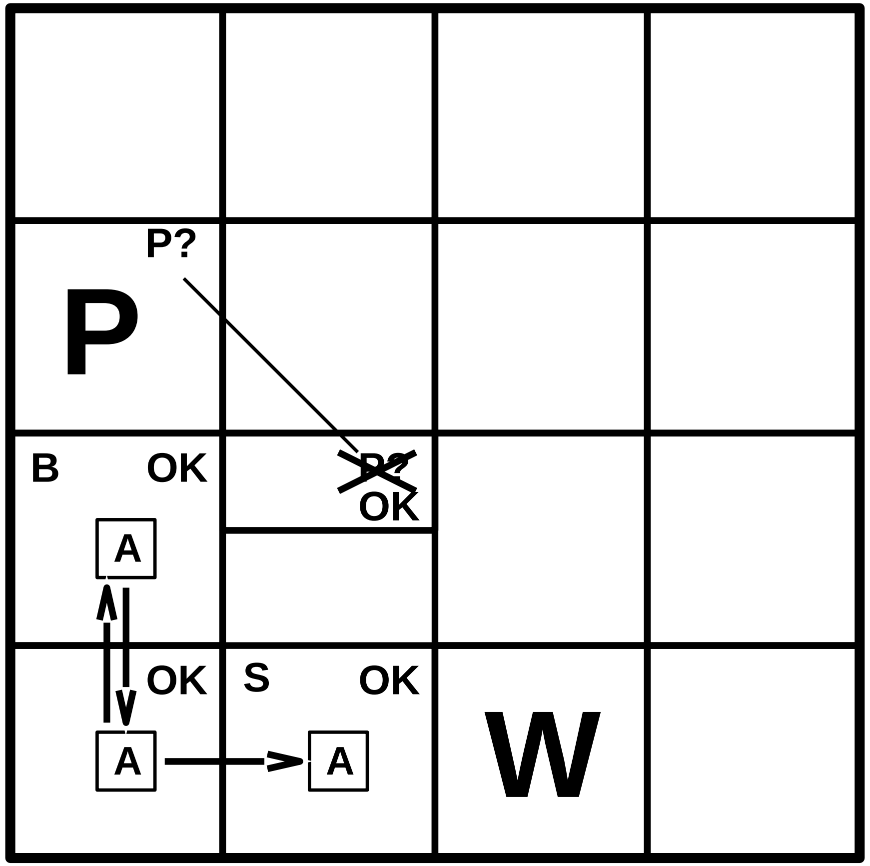
Exploring Wumpus World

- So, it does – and what can it infer from there?

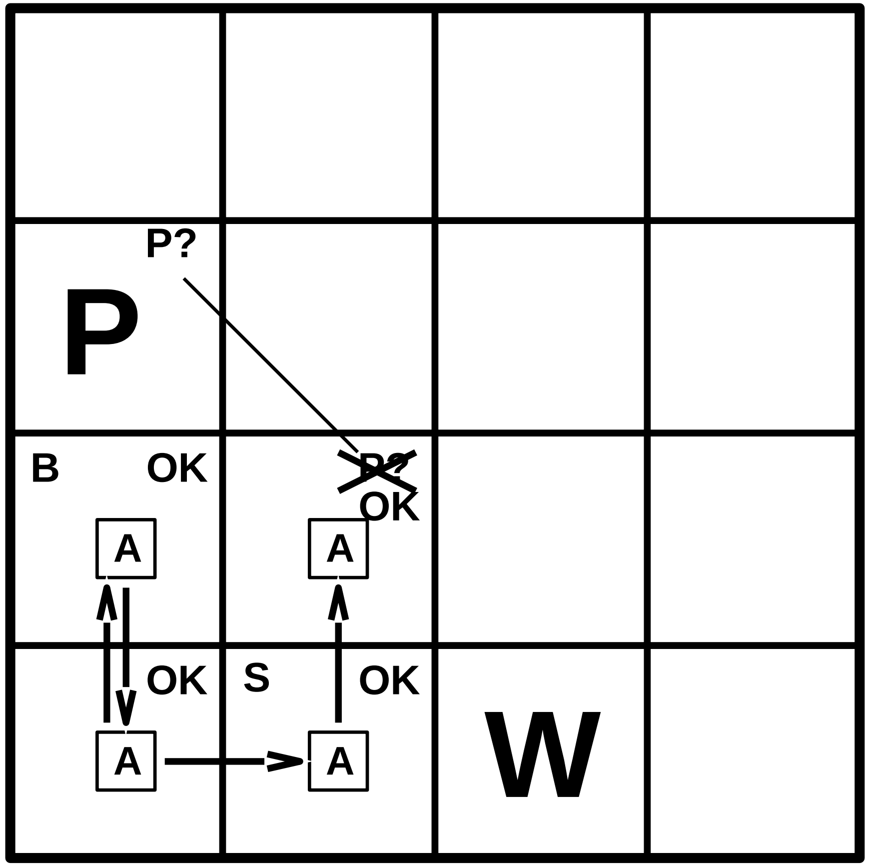


Exploring Wumpus World

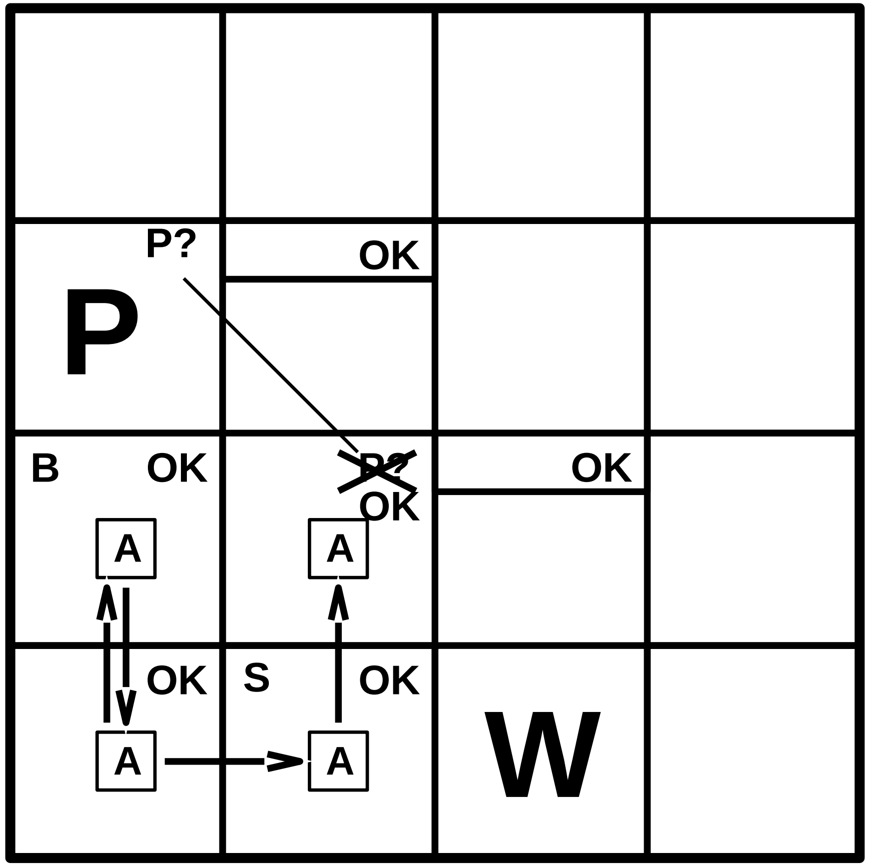
- A pretty difficult inference!; note how difficult because the inference of where the pit is depends on the lack of a percept (no B in 2,1) and percepts gathered over time.



Exploring Wumpus World



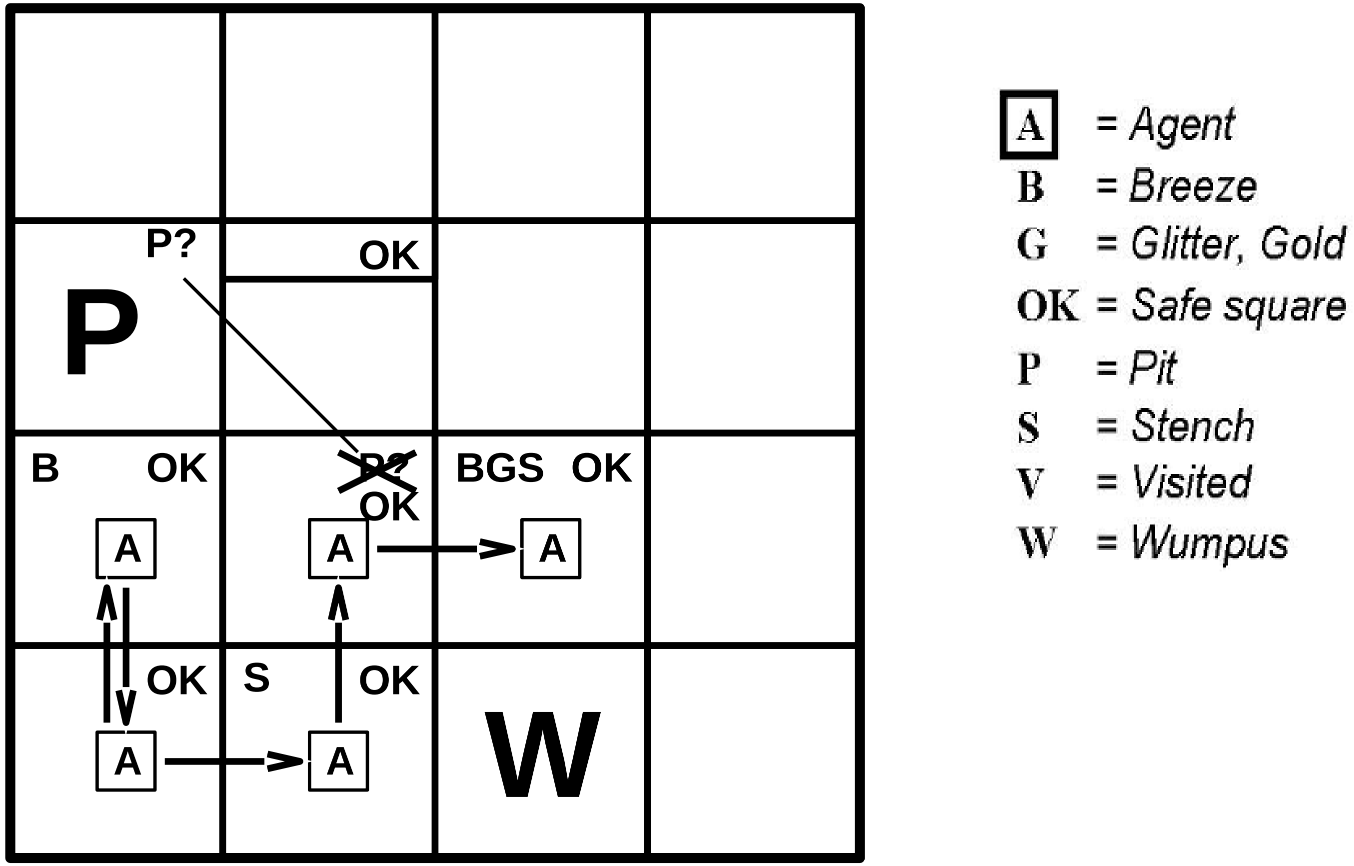
Exploring Wumpus World



Exploring Wumpus World

- In each case, the agent draws a conclusion from the available
- Information, that conclusion is guaranteed to be correct if the available
- The information is correct…

- This is a fundamental property of logical reasoning



Logic

Logic in General

- Logics are formal languages for representing information such that conclusions can be drawn
- Syntax defines the sentences in the language Semantics define the "meaning" of sentences;
  - i.e., define the truth of a sentence in a world
- E.g., the language of arithmetic
- $x + 2 \geq y$ is a sentence; $x2 + y >$ is not a sentence
- $x + 2 \geq y$ is true iff the number $x + 2$ is not 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 $x = 0$, $y = 6$

Pearson

Entailment

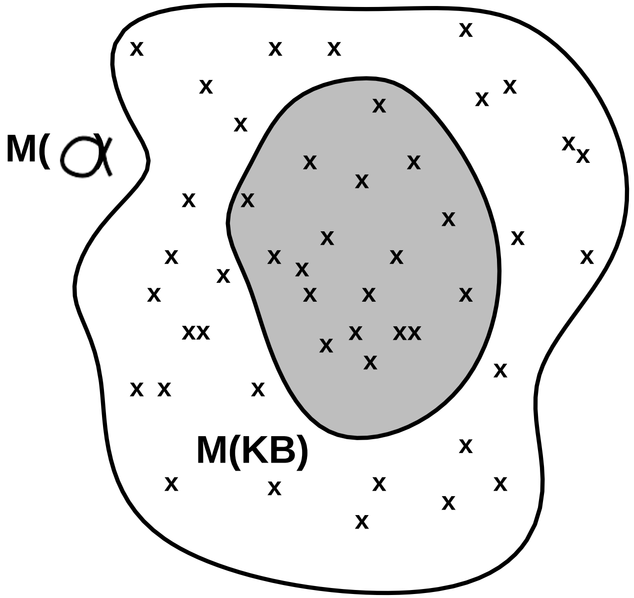
- Entailment means that one thing follows from another:
  - $KB \models a$
- Knowledge base KB entails sentence a
  - if and only if
- a is true in all worlds where KB is true
- E.g., $x + y = 4$ entails $4 = x + y$
- Entailment is a relationship between sentences (i.e., syntax) that is based on semantics

Models

- Logicians typically think in terms of models, which are mathematical abstractions, each of which simply fixes the truth or falsehood of every relevant sentence.

- We say m is a model of a sentence a if a is true in m
  - $M(a)$ is the set of all models of a

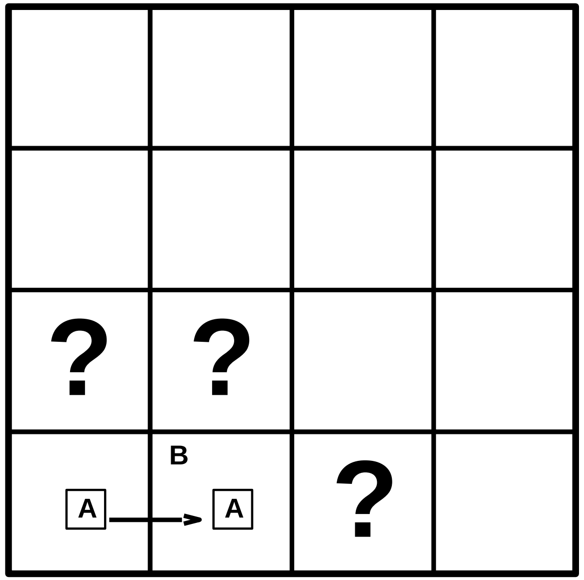
- Then $KB \models a$ if and only if $M(KB) \subseteq M(a)$



- For example, the sentence x = 0 entails the sentence xy = 0. Obviously, in any model
- Where x is zero, it is the case that xy is zero (regardless of the value of y).

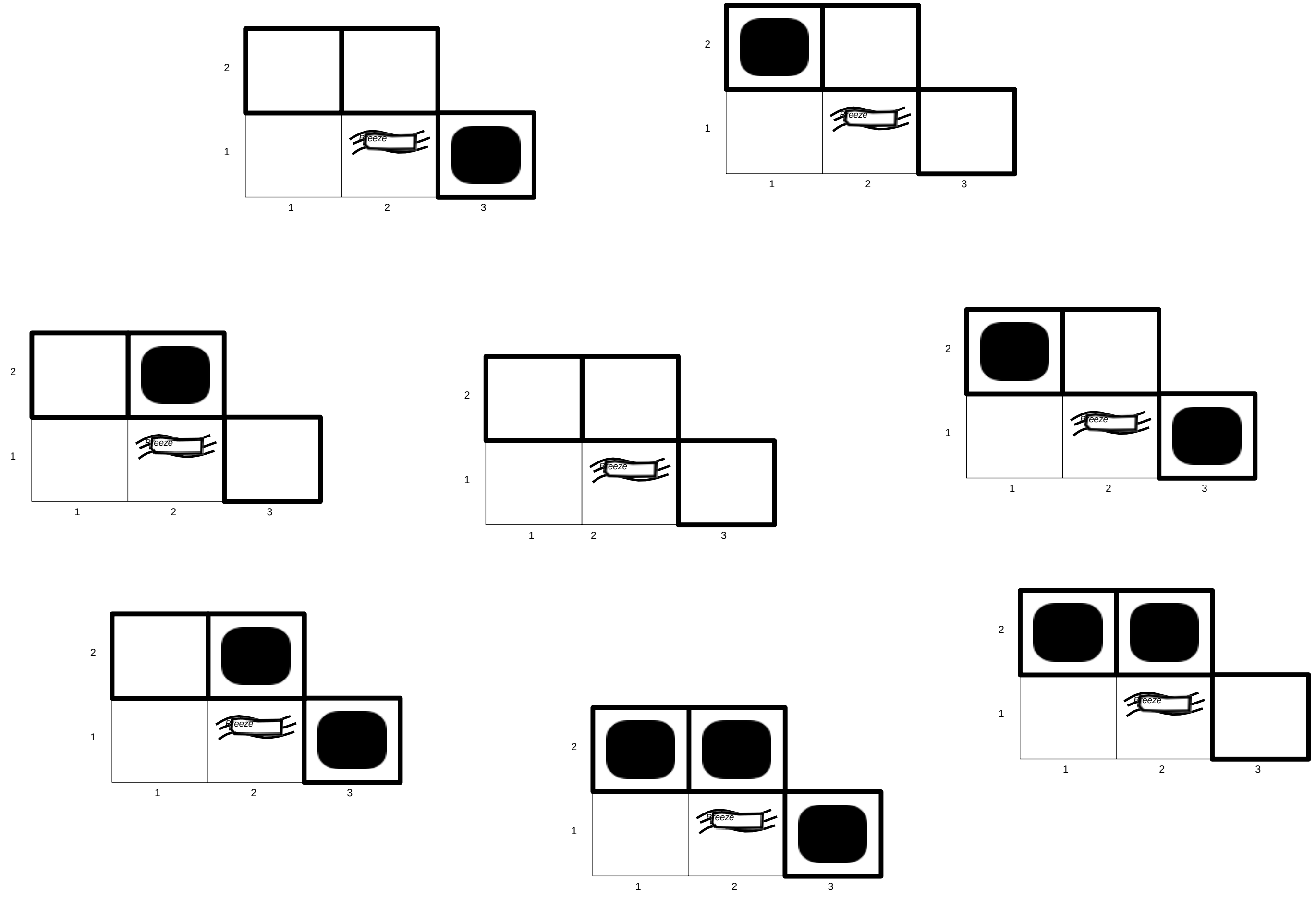
Entailment in the Wumpus world

- Situation after detecting nothing in [1,1], moving right, breeze in [2,1]

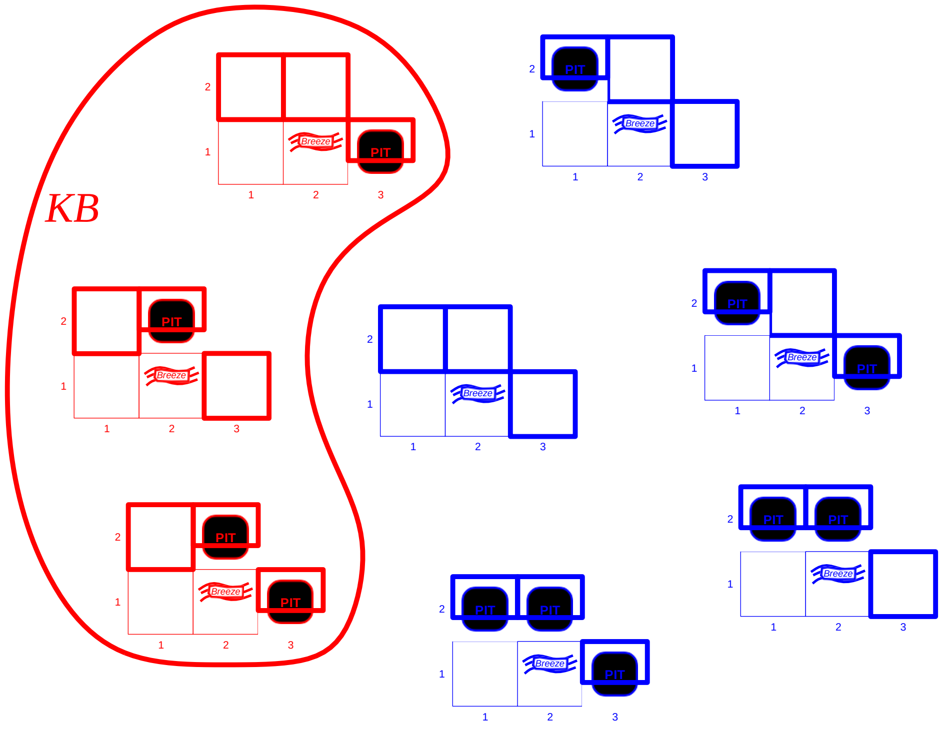


- Consider possible models for ?s assuming only pits
- 3 Boolean choices $\Rightarrow$ 8 possible models

Wumpus Model

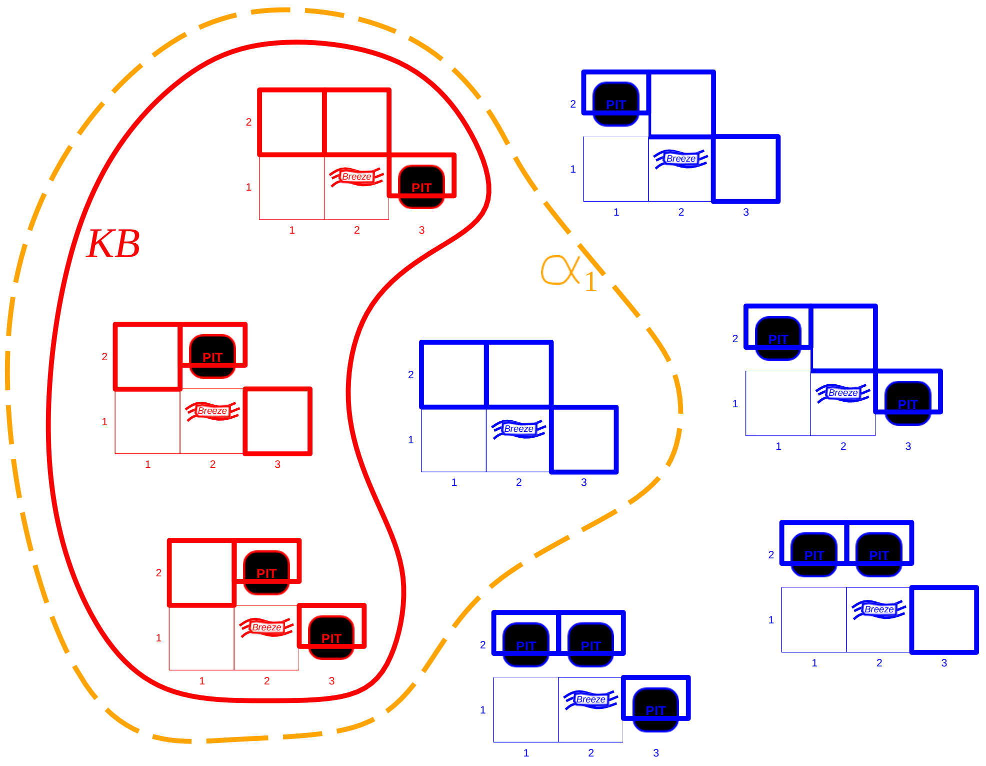


Wumpus Model



KB = wumpus-world rules + observations

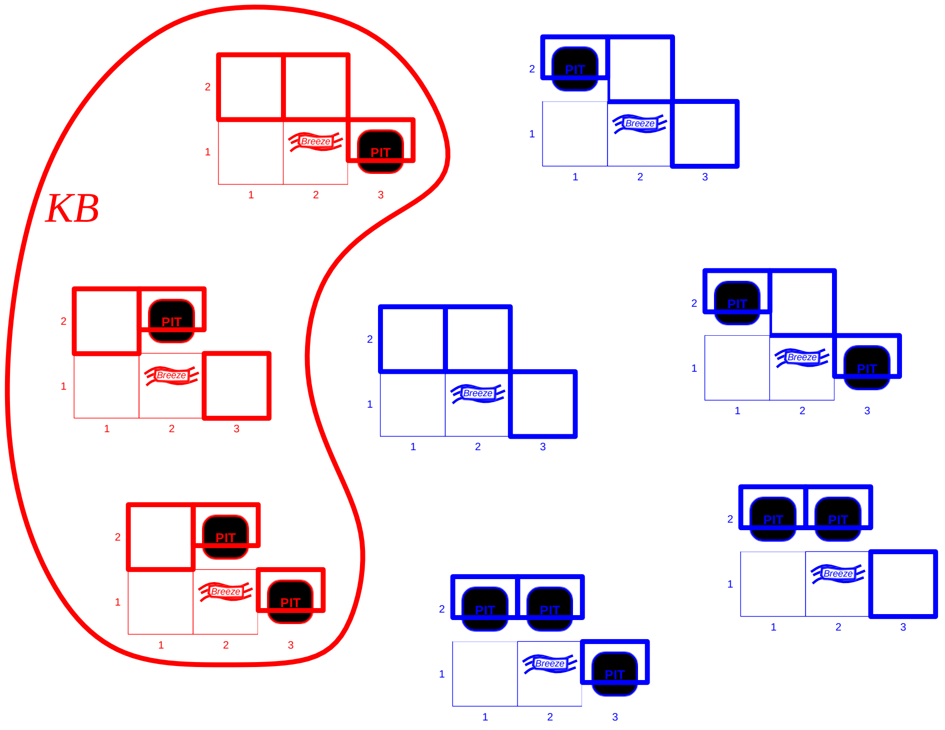
Wumpus Model



KB = wumpus-world rules + observations

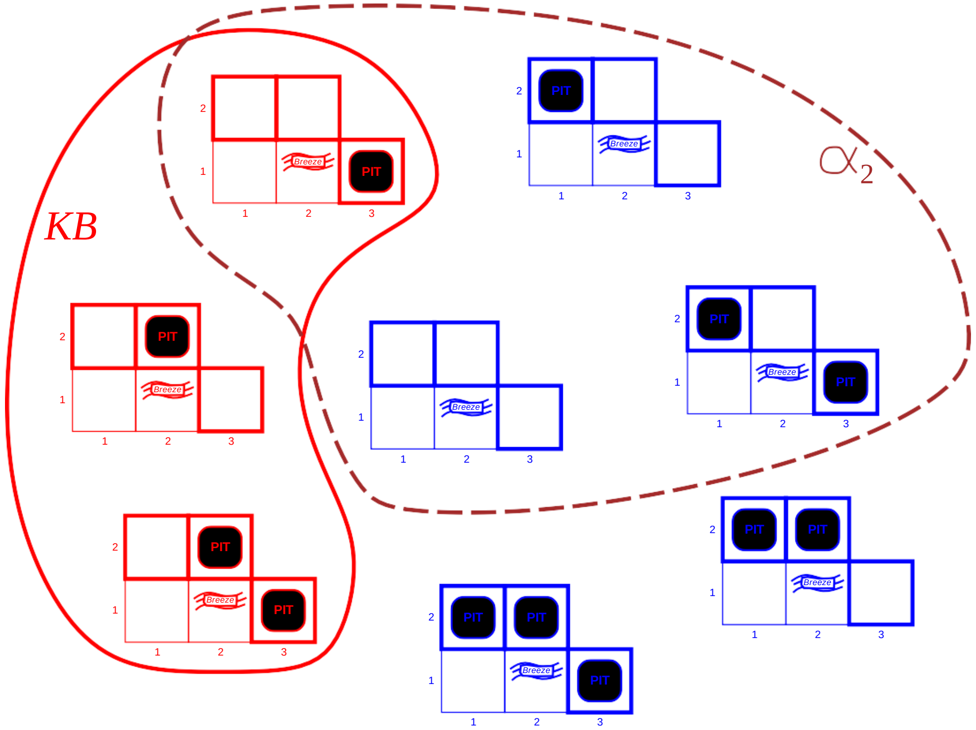
a_1 = "[1,2] is safe", $KB \models a_1$, proved by model checking

Wumpus Model



KB = wumpus-world rules + observations

Wumpus Model



KB = wumpus-world rules + observations

α_2 = "[2,2] is safe", $KB \models \alpha_2$

Propositional (Boolean) Logic

Propositional logic: Semantics

Each model specifies true/false for each proposition symbol

E.g.	$P_{1,2}$	$P_{2,2}$	$P_{3,1}$
	true	*true*	*false*

(With these symbols, 8 possible models, can be enumerated automatically.)

Rules for evaluating truth with respect to a model m:

$\neg S$	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	$S_1 \Rightarrow S_2$	is true and	$S_2 \Rightarrow S_1$	is true

Simple recursive process evaluates an arbitrary sentence, e.g.,

$\neg P_{1,2} \wedge (P_{2,2} \vee P_{3,1}) = true \wedge (false \vee true) = true \wedge true = true$

Propositional logic: Semantics

- Propositional logic is the simplest logic—illustrates basic ideas
- The proposition symbols P_1, P_2 etc are sentences If S is a sentence, $\neg S$ is a sentence (negation)
- and/but
  If S_1 and S_2 are sentences, $S_1 \wedge S_2$ is a sentence (conjunction) If S_1 and S_2 are
  either or
  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)

Truth table for connectives

$\neg Q$

T
F
T
F

$(\neg P \vee Q) \wedge (\neg Q \vee P)$

$\neg P \vee Q$

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

T
F
F
T

$P \Rightarrow Q = \neg P \vee Q$

$P \Leftrightarrow Q = (p \to q) \wedge (q \to p)$

$= (\neg P \vee Q) \wedge (\neg Q \vee P)$

$\neg Q \vee P$

T
F
T
T

Pearson

Wumpus world sentences

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

$$\neg P_{1,1}$$
$$\neg B_{1,1}$$
$$B_{2,1}$$

"Pits cause breezes in adjacent squares"

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

"A square is breezy if and only if there is an adjacent pit"

Truth tables for inference

$B_{1,1}$	$B_{2,1}$	$P_{1,1}$	$P_{1,2}$	$P_{2,1}$	$P_{2,2}$	$P_{3,1}$	......	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*	*true*
false	*true*	*false*	*false*	*false*	*true*	*false*	...	*true*	*true*	*true*	*true*	*true*	*true*
false	*true*	*false*	*false*	*false*	*true*	*true*	...	*true*	*true*	*true*	*true*	*true*	*true*
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*

Enumerate rows (different assignments to symbols),
if KB is true in row, check that α is too

Logical equivalence c

Two sentences are logically equivalent iff true in same models:

$\alpha \equiv \beta$ if and only if $\alpha \models \beta$ and $\beta \models \alpha$

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

Proof methods

- **Proof methods are divided into (roughly) two kinds:**

- **Application of inference rules**
  - Legitimate (sound) generation of new sentences from old
  - Proof = a sequence of inference rule applications
    - Can use inference rules as operators in a standard search algorithm.
  - Typically require translation of sentences into a normal form

- **Model checking**
  - truth table enumeration (always exponential in n)
  - improved backtracking, e.g., Davis–Putnam–Logemann–Loveland heuristic search in model space (sound but incomplete)
    - e.g., min-conflicts-like hill-climbing algorithms

Forward Chaining

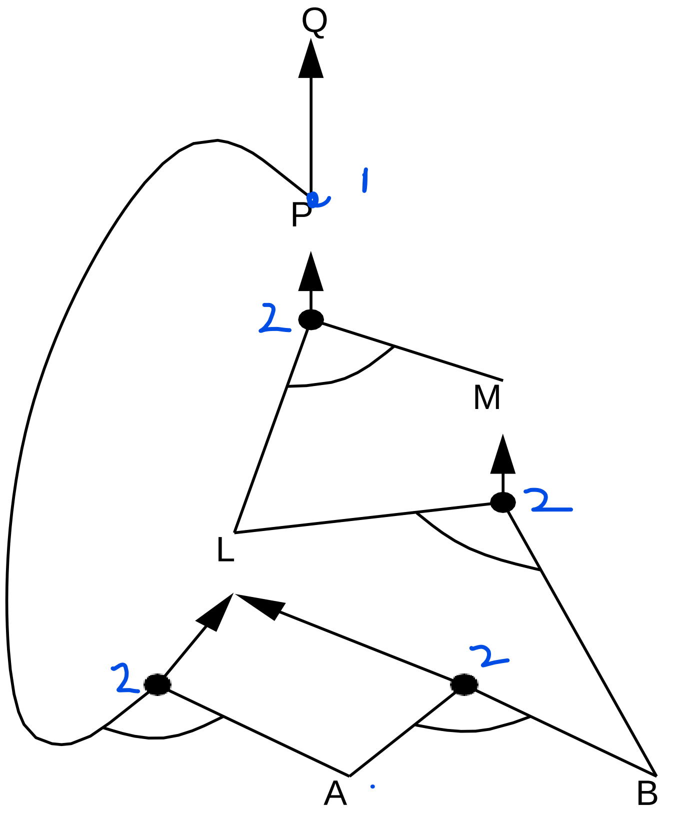
https://www.youtube.com/watch?v=EZJs6w2YFRM

From min 5:40

Forward chaining

- Idea: fire any rule whose premises are satisfied in the KB, and add its conclusion to the KB, until the query is found

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



Forward chaining algorithm

```
function PL-FC-Entails?(KB, q) returns true or false
  inputs: KB, the knowledge base, a set of propositional Horn clauses
          q, the query, a proposition symbol
  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 in KB

  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 example

We want to prove if the relation

$P \Rightarrow Q$

is true or not.

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

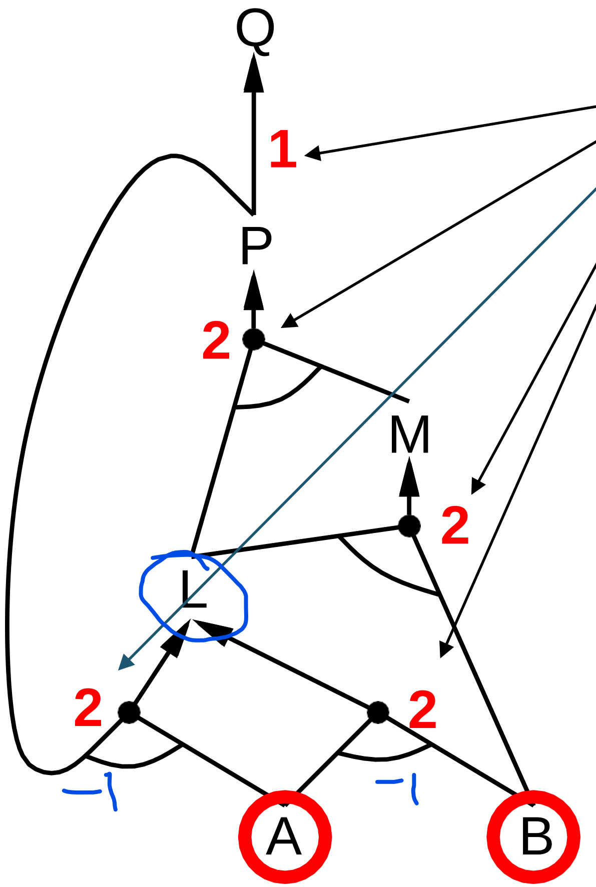
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



Here we added the number of parameters in each rule

$L \wedge M \Rightarrow P$

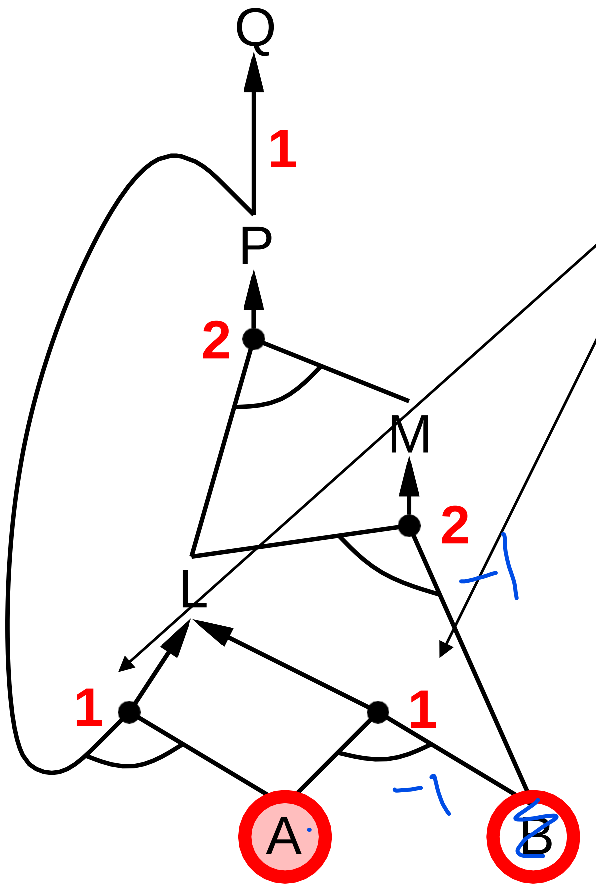
To get (p) we need two parameters (L and M)

And so on...

We have two parameters A and B they are TRUE

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



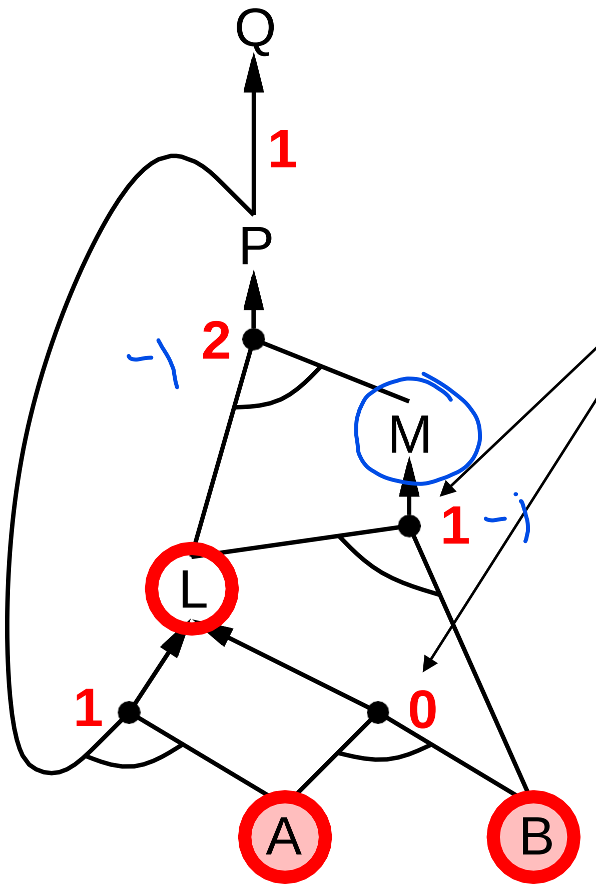
We will start with parameter A as we know it is true, then we will remove it from all relations.

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

Then we will reduce all numbers from their relation by one

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 −1 −1
B −1 −1



Next, we will go to parameter B as we know it is true, then we will remove it from all relations.

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

Then we will reduce all numbers from their relation by one

Forward chaining example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

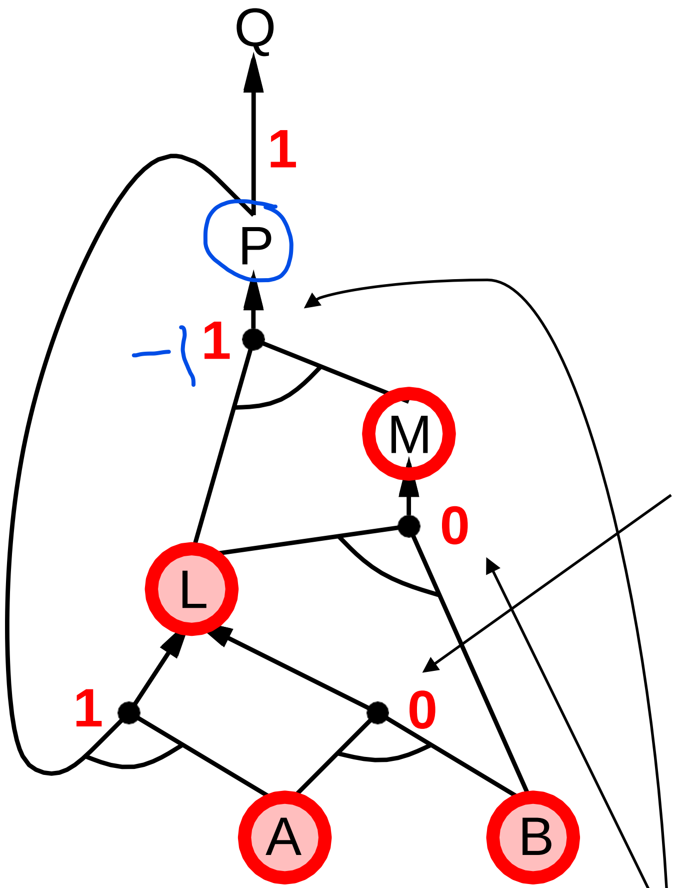
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



When you get any parameter has it relation equal to (0) then it also will be true.

Here, L parameter has (0) relation, then it is true

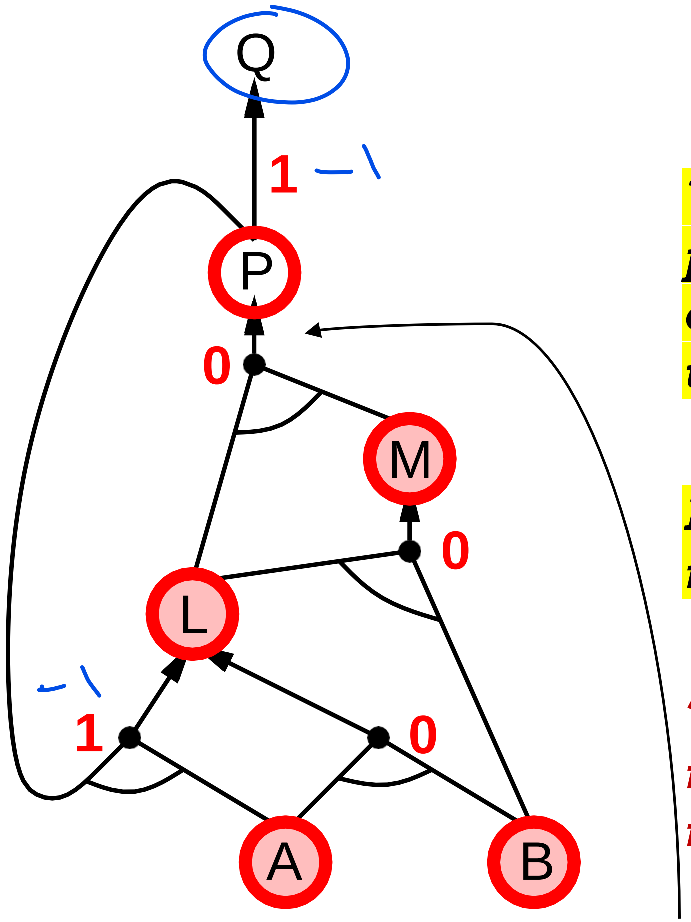
Then we will reduce all numbers from their relation by one

$B \wedge L \Rightarrow M$

$L \wedge M \Rightarrow P$

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



When you get any parameter has it relation equal to (0) then it also will be true.

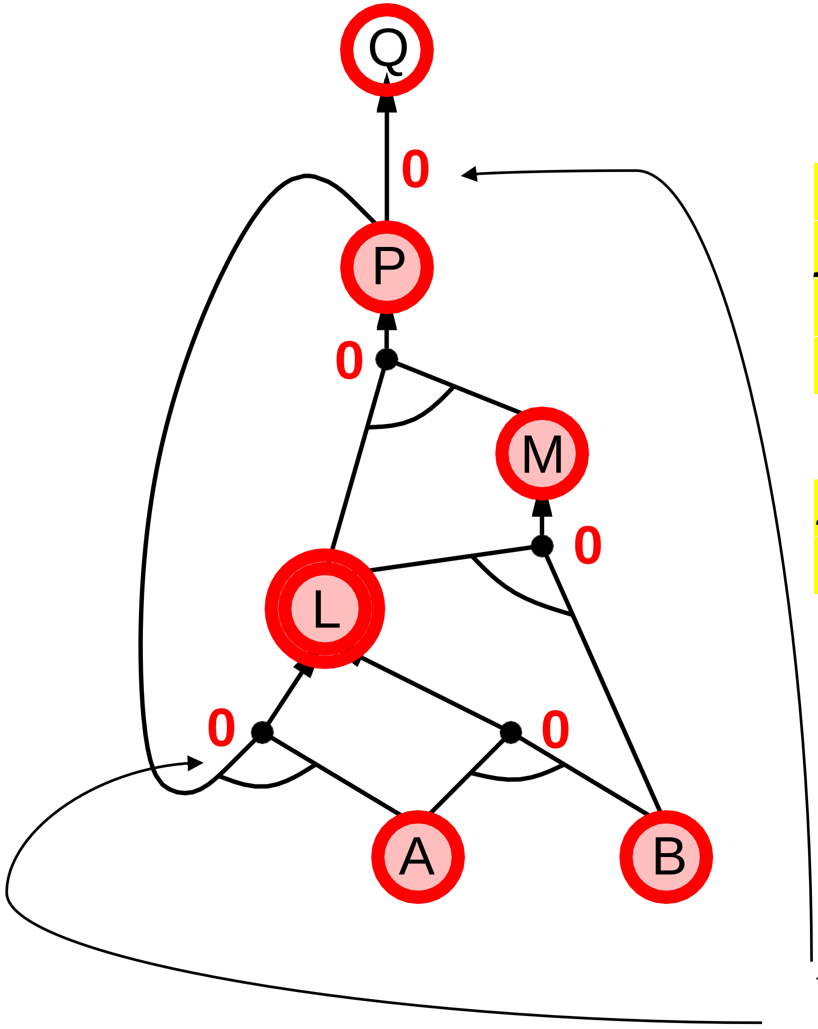
Here, M parameter has (0) relation, then it is true

Then we will reduce all numbers from their relation by one

$L \wedge M \Rightarrow P$

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



When you get any parameter has it relation equal to (0) then it also will be true.

Here, P parameter has (0) relation, then it is true

Then we will reduce all numbers from their relation by one

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

Forward chaining example

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

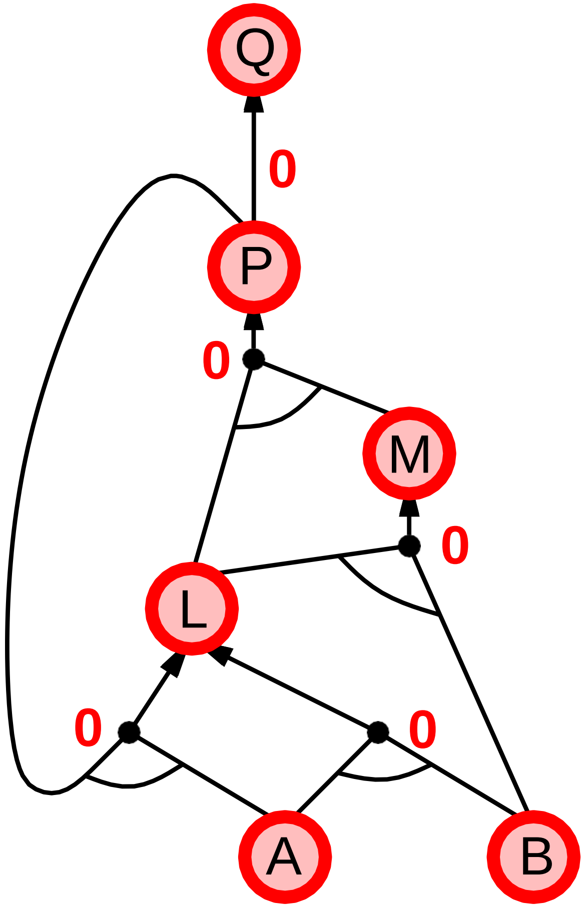
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



Finally, the parameter Q also will be true

Then

$P \Rightarrow Q$

Is true

Proof of completeness

FC derives every atomic sentence that is entailed by KB

1. FC reaches a fixed point where no new atomic sentences are derived

2. Consider the final state as a model m, assigning true/false to symbols

3. Every clause in the original KB is true in m
   Proof: Suppose a clause $a_1 \wedge \ldots \wedge a_k \Rightarrow b$ is false in m Then $a_1 \wedge \ldots \wedge a_k$ is true in m and b is false in m Therefore the algorithm has not reached a fixed point!

4. Hence m is a model of KB

5. If $KB \models q$, q is true in every model of KB, including m

General idea: construct any model of KB by sound inference, check a

Backward Chaining

https://www.youtube.com/watch?v=EZJs6w2YFRM

From min 11:48

Backward chaining

SL

Idea: work backwards from the query q:

to prove q by BC,

check if q is known already, or

prove by BC all premises of some rule concluding q

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

Avoid repeated work: check if new subgoal

1) has already been proved true, or
2) has already failed

Backward chaining example

SL

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

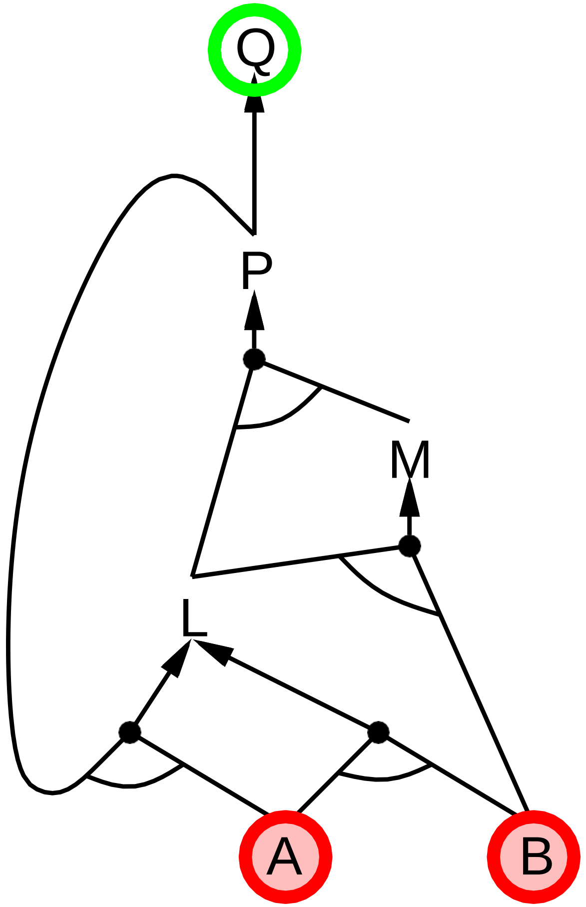
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



Backward chaining example SL

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

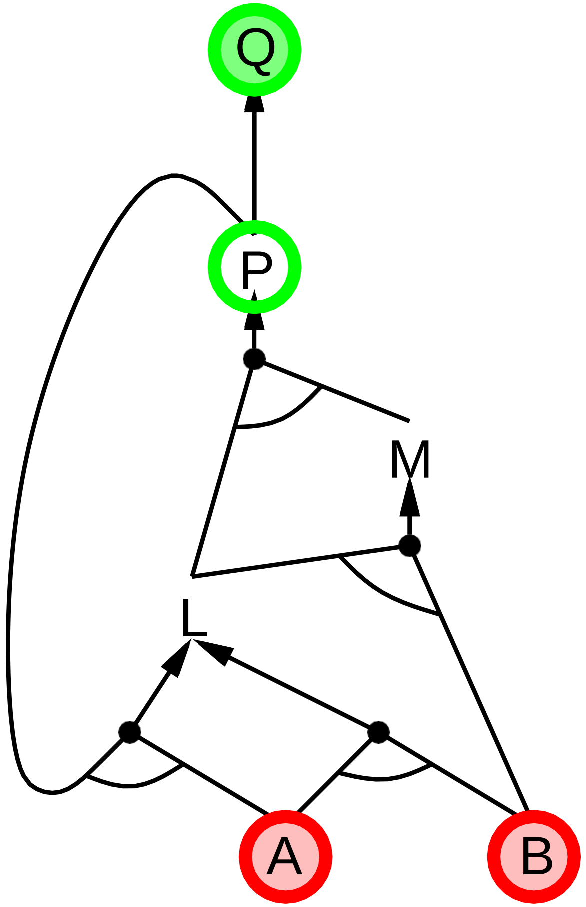
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

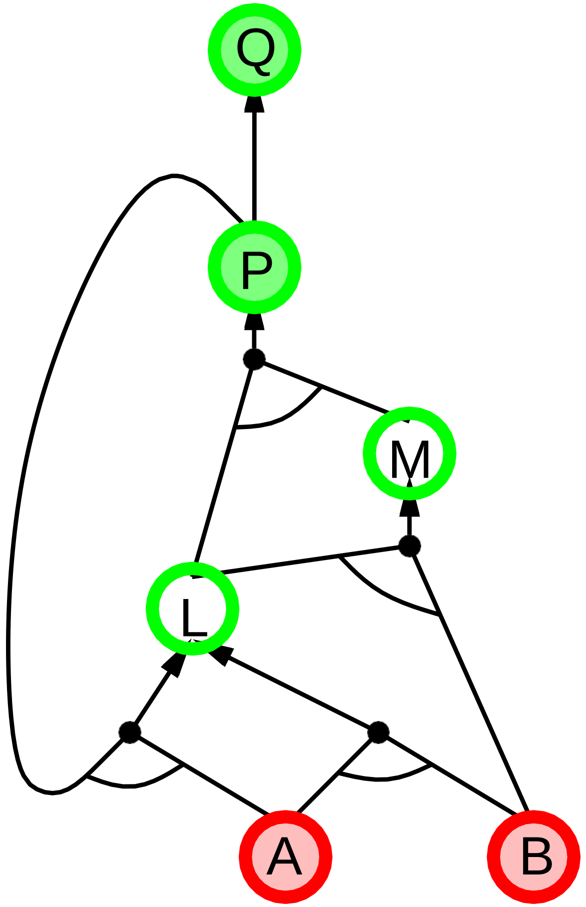
B



Backward chaining example

SL

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



Backward chaining example

SL

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

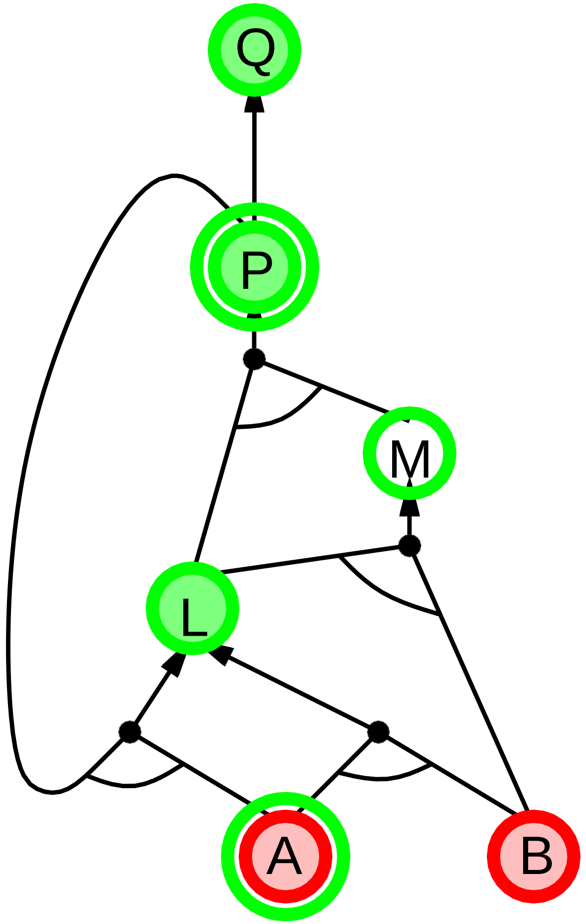
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



Backward chaining example

SL

$P \Rightarrow Q$

$L \land M \Rightarrow P$

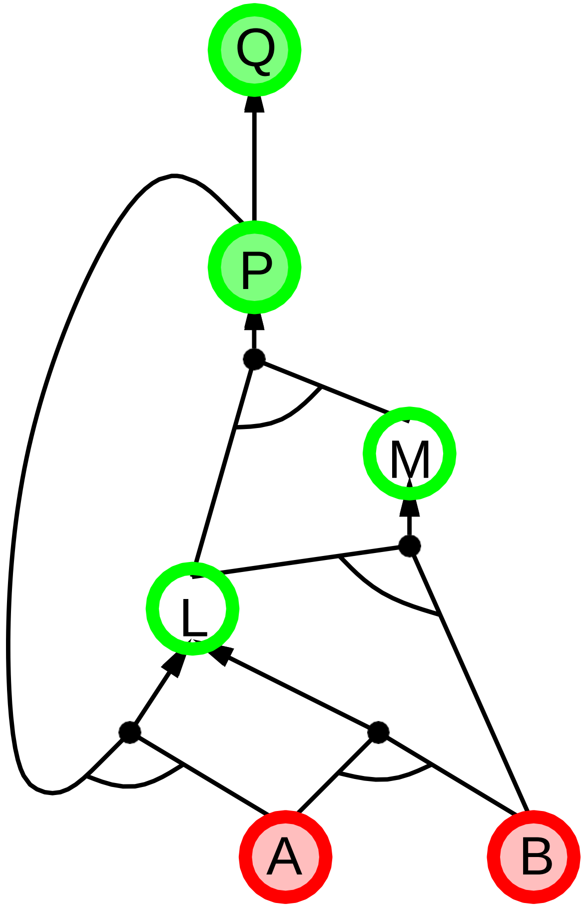
$B \land L \Rightarrow M$

$A \land P \Rightarrow L$

$A \land B \Rightarrow L$

A

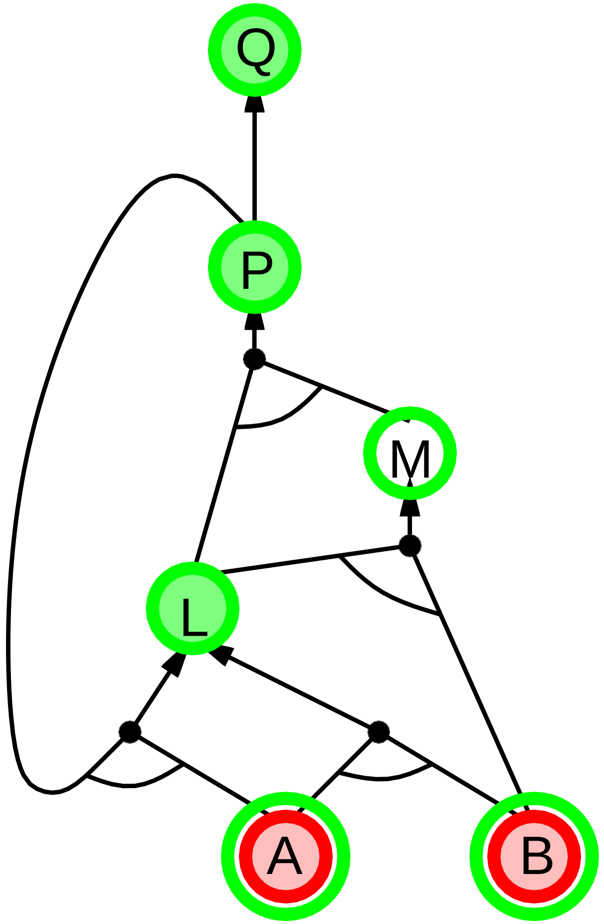
B



Backward chaining example

SL

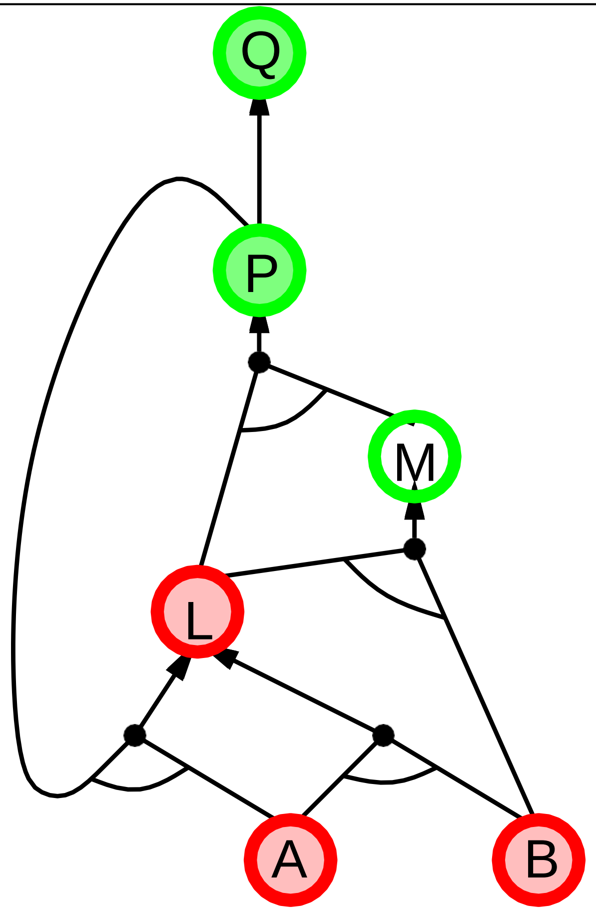
$$
\begin{aligned}
&P \Rightarrow Q \\
&L \wedge M \Rightarrow P \\
&B \wedge L \Rightarrow M \\
&A \wedge P \Rightarrow L \\
&A \wedge B \Rightarrow L \\
&A \\
&B
\end{aligned}
$$



Backward chaining example

SL

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



Backward chaining example

SL

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

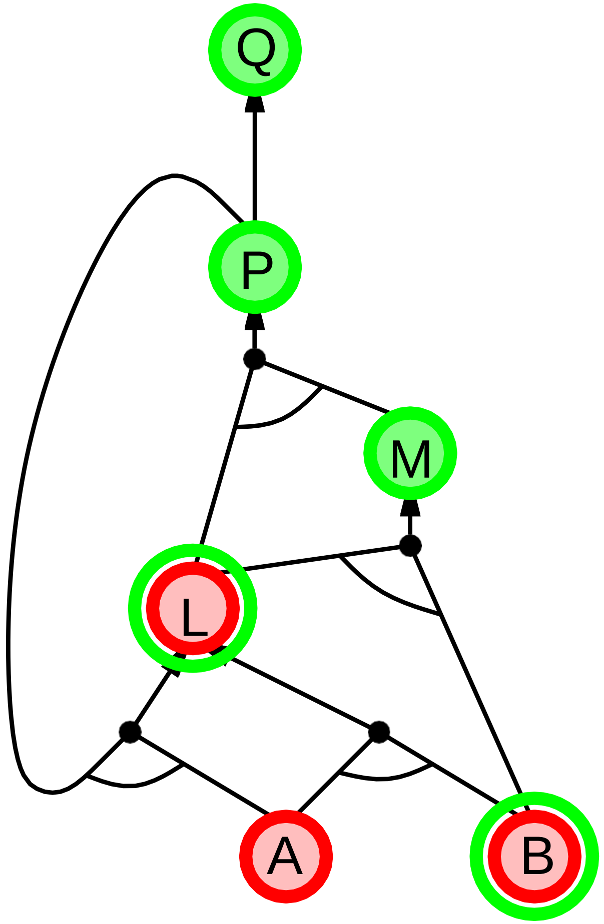
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

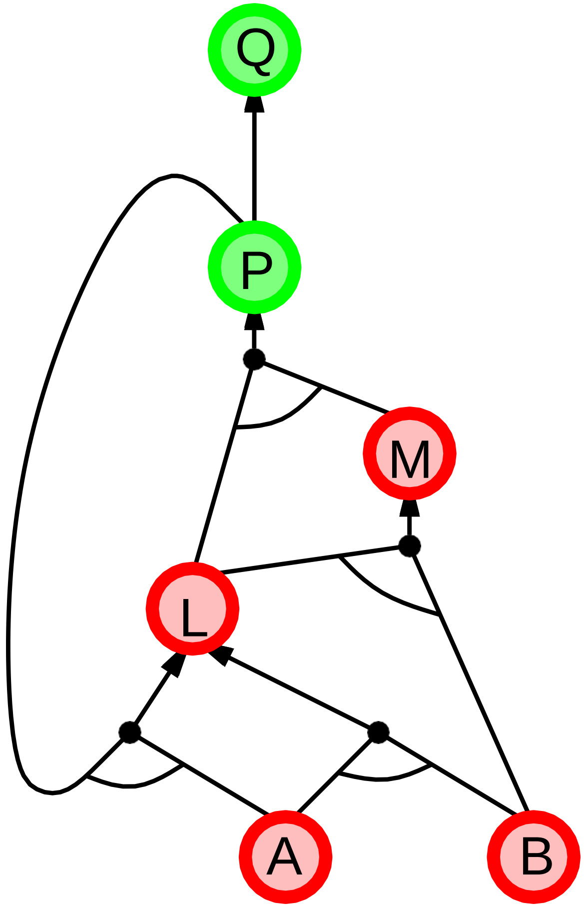
B



Backward chaining example

SL

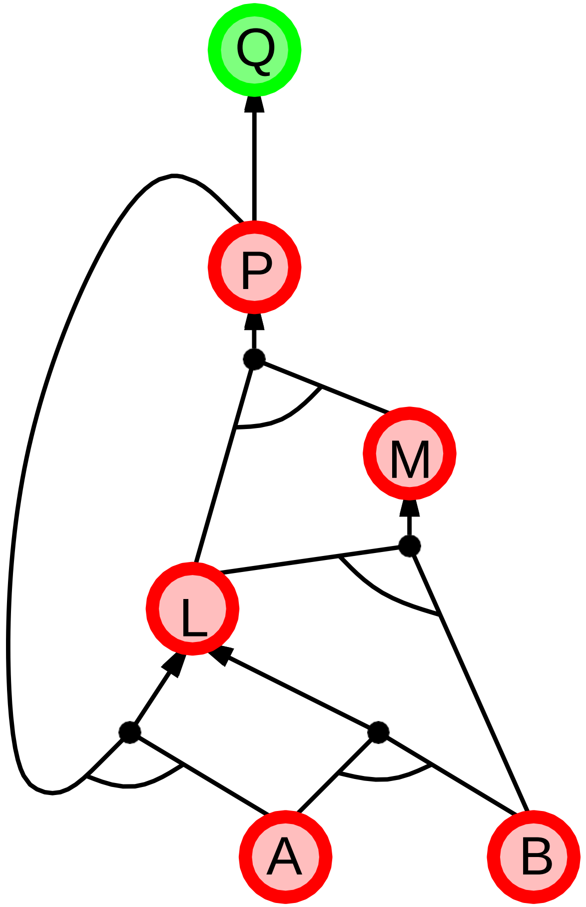
$$P \Rightarrow Q$$
$$L \wedge M \Rightarrow P$$
$$B \wedge L \Rightarrow M$$
$$A \wedge P \Rightarrow L$$
$$A \wedge B \Rightarrow L$$
$$A$$
$$B$$



Backward chaining example

SL

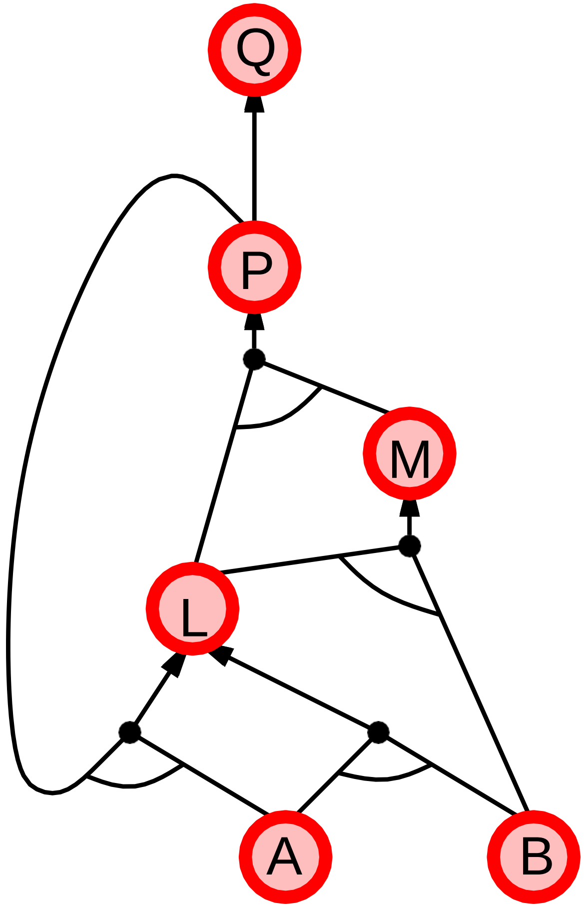
$$P \Rightarrow Q$$
$$L \wedge M \Rightarrow P$$
$$B \wedge L \Rightarrow M$$
$$A \wedge P \Rightarrow L$$
$$A \wedge B \Rightarrow L$$
$$A$$
$$B$$



Backward chaining example

SL

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



Forward vs. backward chaining

SL

FC is data-driven, cf. 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

Propositional logic examples

Propositional logic (PL)

- A proposition (suggestion) is a statement that has a truth value (True or False)

T or F

- **Propositional logic**

?

  - A simple language useful to express key ideas and definitions
  - The user defines a set of propositional symbols, like p and q
  - The user also defines the semantics of each propositional symbol:

p = ___
q = ___

    - p means "It is hot"
    - q means "It is humid"

Propositional Logic: Syntax

- Propositions like "Hassan is sleeping" and "it is humid" are atomic propositions
  - What if we divide these statements?

- Logical connectives are used to represent
  - **AND**: $\wedge$, **OR**: $\vee$, **implication**: $\Rightarrow$, **equivalence**: $\Leftrightarrow$, and **not**: $\neg$

- Statements or sentences in the language are constructed from atomic propositions and logical connectives – parenthesis are also used to enforce grouping
  - $p \wedge \neg q$: Hassan is sleeping **and** it is **not** humid
  - $p \Rightarrow q$: **If** Hassan is sleeping, **then** it is humid

How to express NL in Logic?

- Pick the smallest/atomic statements without **and**, **or**, etc. about which you could answer the question
    - is it true or false?

- Use propositional variables to stand for these statements, and connect them with the relevant logical connectives

Practice example

- If I put a hook in the water and I am patient then I will catch a fish

Practice example

- If I put a hook in the water and I am patient then I will catch a fish

  - A: I put a hook in the water
  - B: I am patient
  - C: I will catch a fish

$$(A \land B) \Rightarrow C$$

Practice example

- If the sun is shining and we win the toss, we will bat first, or if the sun is not shining but we win the toss, then we will not bat first.

Practice example

- If the sun is shining and we win the toss, we will bat first, or if the sun is not shining but we win the toss, then we will not bat first.
  - S: Sun is shining
  - W: We win the toss
  - B: We will bat first

$((S \wedge W) \Rightarrow B) \vee (\neg S \wedge W) \Rightarrow \neg B$

$$((S \wedge W) \Rightarrow B) \vee ((\neg S \wedge W) \Rightarrow \neg B)$$

From logic to NL

p: It is hot

q: It is humid

r: It is raining

$(p \wedge q) \Rightarrow r$

If it is hot & humid then it is raining

$q \Rightarrow p$

If it is humid, then it is hot

From logic to NL

p: It is hot

q: It is humid

r: It is raining

$(p \land q) \Rightarrow r$

If it is hot and humid, then it is raining

$q \Rightarrow p$

If it is humid, then it is hot

Practice example

M: I have money

L: I like the LG-mobile

B: I will buy the LG-mobile

T: I find a touch screen

$(M \wedge L \wedge T) \Rightarrow B$

if I have money

and I like LG-mobile

and I find touch screen

the I will buy LG

Practice example

M: I have money

L: I like the LG-mobile

B: I will buy the LG-mobile

T: I find a touch screen

$(M \wedge L \wedge T) \Rightarrow B$

If I have money and I like the LG-mobile and I find a touch screen, then I will buy the LG-mobile

If I have money and I like the LG-mobile, then I will buy it if I find a touch screen

The same logical formula can be expressed in different ways

Inference: Proof Theory

- **How do we draw new conclusions from existing supplied facts?**
- We can define inference rules, which are guaranteed to give true conclusions given true premises (Laws of thought-based approach in AI)
- One such rule is modus ponens (Symbol for "therefore”. Latin for "**method of affirming**." A rule of inference used to draw logical conclusions, which states that
  - If p is true, and if p implies q, then q is true
  - If A is true and $A \Rightarrow B$ is true, then conclude B is true

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

Inference: Proof Theory

- Let P mean "It is raining", Q mean "I carry my umbrella"
- If we know that P is true and $P \Rightarrow Q$ is true, we can conclude that Q is also true

- **Some expressions are equivalent**
  - E.g., $P \Rightarrow Q$ and $\neg P \vee Q$ are logically equivalent
    - Check with truth tables or just think about them

- Reason why $P \Rightarrow Q$ is not equivalent to $Q \Rightarrow P$

Propositional logic: **limitations**

- All living things will eventually die
- Some mammals can fly
- The above facts cannot be expressed in propositional logic
  - Propositional logic does not deal with general statements, which include quantifiers
  - But quantifications are very common in knowledge/reasoning processes
- Propositional logic lacks expressivity – we need some other logic to represent quantification
  - Predicate logic could serve this purpose

First-order Logic (Predicate logic)

SL

Predicate Logic SL

- **Syntax**
  - **Terms**: Constants, Variables, Functions
  - **Connectors**: $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$, $\neg$
  - **Quantifiers**: Universal ($\forall$), Existential ($\exists$)
  - **Well-formed formulas**: Atoms (A, B, …), $A \rightarrow B$, $A \leftrightarrow B$, $A \wedge B$, $A \vee B$, $\neg A$, $(\forall x\ A)$, $(\exists x\ A)$

    - ***Example:***

      FGI135 is an airplane: Airplane(FGI135)

      All airplanes fly: $\forall x\ (Airplane(x) \rightarrow Fly(x))$

Predicate Logic SL

- **Semantic:**
  - There is a precise meaning to expressions in predicate logic → determining whether something is true or false
  - $\forall$ x P(x) means that P(x) must be true for every object X in the domain of interest
  - $\exists$ x P(x) means that P(x) must be true for at least one object X in the domain of interest
  - So if we have a domain of interest consisting of just two people, hassan and bilal, and we know that `smart(hassan)` and `smart(bilal)` are true, we can say that $\forall$ `X smart(X)` is true.

Predicate logic SL

- **Predicate**
  - A sentence often has two parts: subject and predicate
    - Subject is what/whom the sentence is about
      - Table is black.
    - Predicate tells something about the subject
      - Table is black.
- Predicate logic allows us to describe properties of objects and/or relations among the objects
  - Table is black: black (table)
  - Hassan is brother of Bilal: brother_of (hassan, bilal)

Predicate Logic

SL

- In predicate logic, the basic unit is a predicate- argument structure, called an atomic sentence:
  - Likes (hassan, chocolate)
  - Tall (ali)
- Arguments can be:
  - constant symbol, such as **ali**
  - variable symbol, such as **X**
  - function expression, e.g., **father_of (hassan)**
- So we can have:
  - likes(X, chocolate)
  - friends(son_of (ali), son_of (omar))
- **Predicate returns truth value; function returns object**

Predicate logic: Syntax SL

- The atomic sentences can be combined using logic connectives (like propositional logic)
  - eating (hassan) $\Rightarrow \neg$ fasting(hassan)
- Sentences can also be formed using quantifiers $\forall$ (form all) and $\exists$ (there exists) to indicate how to treat variables:
  - $\forall$ x lovely (x)
  - $\exists$ x lovely(x)
  - $\exists$ x (goes_to(x, makkah) $\Rightarrow$ muslim(x))

Predicate logic: Syntax SL

- The atomic sentences can be combined using logic connectives (like propositional logic)
  - eating (hassan) ⇒ ¬ fasting(hassan)
- Sentences can also be formed using quantifiers ∀ (form all) and ∃ (there exists) to indicate how to treat variables:
  - ∀ x lovely (x)

    Everything is lovely.
  - ∃ x lovely(x)

    Something is lovely.
  - ∃ x (goes_to(x, makkah) ⇒ muslim(x))

    Only Muslims can go to Makkah.

    Everyone who goes to Makkah is Muslim.

Predicate Logic: Syntax **SL**

- Multiple quantifiers in one sentence, e.g.,
  - $\forall x \, \exists y \; loves(x, y)$
  - $\exists x \, (kind(x) \wedge \forall y \; loves(y,x))$
- Some more examples:
  - Every race has a winner.

    $\forall x \, (race(x) \wedge \exists y \; winner(y, x))$
  - Ahmad likes generous people.

    $\forall x \, ((generous(x) \Rightarrow likes(Ahmad, x))$

Predicate Logic: Syntax SL

- Multiple quantifiers in one sentence, e.g.,
  - $\forall x \, \exists y \; loves(x, y)$

    Everyone loves something.
  - $\exists x \, (kind(x) \wedge \forall y \; loves(y,x))$

    Everyone loves kind people.
- Some more examples:
  - Every race has a winner.

    $\forall x \, (race(x) \wedge \exists y \; winner(y, x))$
  - Ahmad likes generous people.

    $\forall x \, ((generous(x) \Rightarrow likes(Ahmad, x))$

Proof and inference

SL

- We can define inference rules allowing us to say that if certain things are true, certain other things are sure to be true, e.g.

$\exists$ x p(x) $\Rightarrow$ q(x)

p(something)

------------------ (so we can conclude)

q(something)

- This involves matching p(x) against p(something) and binding the variable x to the symbol something

Proof and Inference SL

- What can we conclude from the following?
  - Rule 1: $\exists$ x (pious(x) $\Rightarrow$ prays(x))
  - Rule 2: $\exists$ x (prays(x) $\Rightarrow$ Obeys(x, Allah))
  - Fact 1: pious(Ali)

pious(Ali) ➔

prays(Ali) ➔

Obeys(Ali, Allah)

⦿We conclude that Allah loves Ali.

Summary

- Logical agents apply inference to a knowledge base to derive new information and make decisions
- Basic concepts of logic:
  - syntax: formal structure of sentences
  - semantics: truth of sentences wrt models
  - entailment: necessary truth of one sentence given another
  - inference: deriving sentences from other sentences
- Wumpus world requires the ability to represent partial and negated information, reason by cases, etc.
- Forward, backward chaining are linear-time, complete for Horn clauses.
- Propositional logic lacks expressive power.
- Predicate logic could serve this purpose.

Reference

Book:

Artificial intelligence: a modern approach by Stuart Russel and Peter Norving, fourth edition (Chapters 7 and 8)