Week 10: Logical Agents

Dr Ammar Masood

Department of Cyber Security,

Air University Islamabad

Table of Contents

Logical Agents

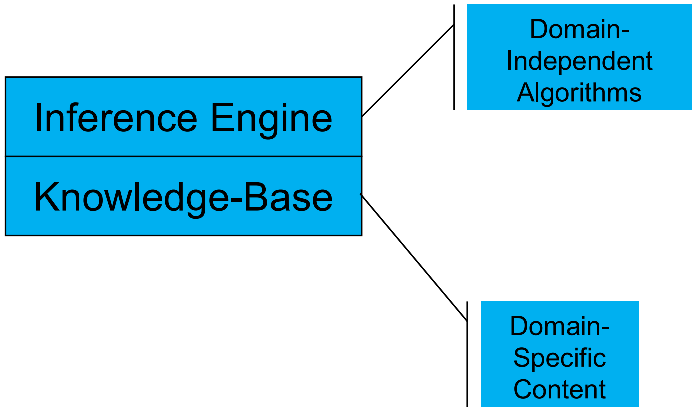
- Humans can know “things” and “reason”
  - Representation: How are the things stored?
  - Reasoning: How is the knowledge used?
    - To solve a problem…
    - To generate more knowledge…
- Knowledge and reasoning are important to artificial agents because they enable successful behaviors difficult to achieve otherwise
  - Useful in partially observable environments
- Can benefit from knowledge in very general forms, combining and recombining information

Knowledge-Based Agents

- Central component of a Knowledge-Based Agent is a ***Knowledge-Base***
  - A set of sentences in a formal language
    - Sentences are expressed using a knowledge representation language
- Two generic functions:
  - TELL - add new sentences (facts) to the KB
    - “Tell it what it needs to know”
  - ASK - query what is known from the KB
    - “Ask what to do next”

Knowledge-Based Agents

- The agent must be able to:
  - Represent states and actions
  - Incorporate new percepts
  - Update internal representations of the world
  - Deduce hidden properties of the world
  - Deduce appropriate actions



Knowledge-Based Agents

function KB-AGENT(*percept*) **returns** an *action*
  persistent: *KB*, a knowledge base
            t, a counter, initially 0, indicating time

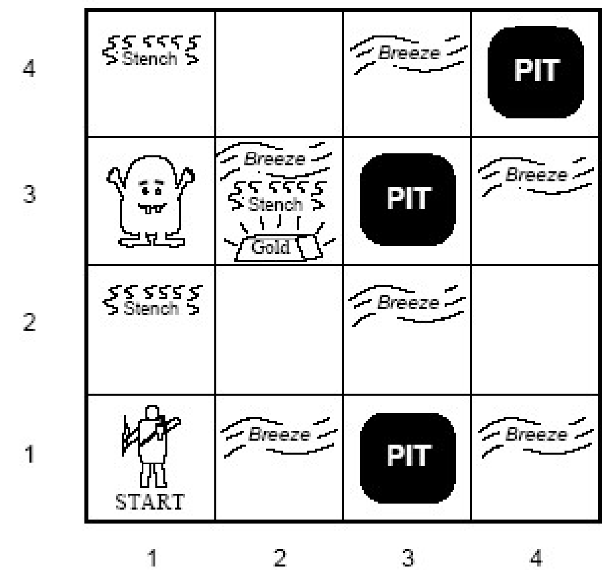
  TELL(*KB*, MAKE-PERCEPT-SENTENCE(*percept*, *t*))
  action ← ASK(*KB*, MAKE-ACTION-QUERY(*t*))
  TELL(*KB*, MAKE-ACTION-SENTENCE(*action*, *t*))
  t ← *t* + 1
  return *action*

- MAKE-PERCEPT-SENTENCE constructs a sentence asserting that the agent perceived the given percept at the given time.
- MAKE-ACTION-QUERY constructs a sentence that asks what action should be done at the current time.
- Finally, MAKE-ACTION-SENTENCE constructs a sentence asserting that the chosen action was executed.

Wumpus World

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

- Environment
  - 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



- Sensors: Stench, Breeze, Glitter, Bump, Scream
- Actuators: Left turn, Right turn, Forward, Grab, Release, Shoot

Wumpus World

- Characterization of Wumpus World
  - Observable
    - partial, only local perception
  - Deterministic
    - Yes, outcomes are specified
  - Episodic
    - No, sequential at the level of actions
  - Static
    - Yes, Wumpus and pits do not move
  - Discrete
    - Yes
  - Single Agent
    - Yes

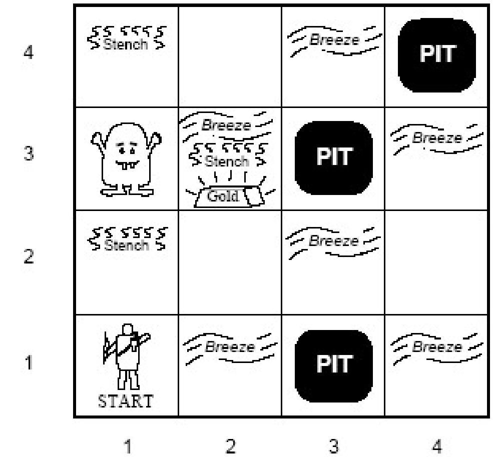
Wumpus World continued

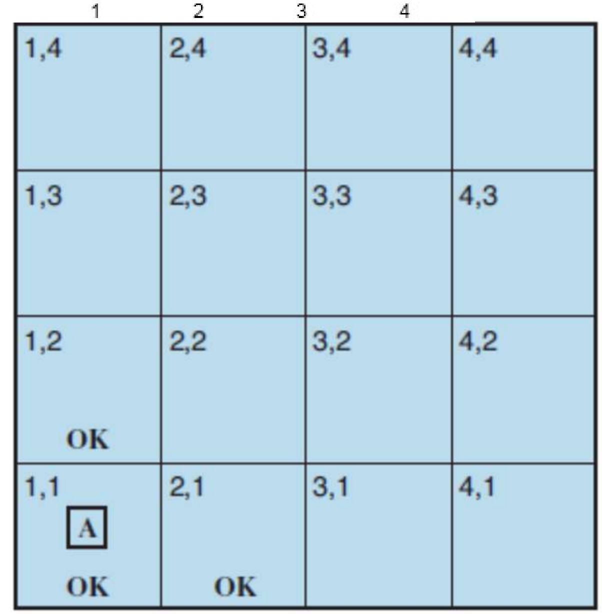
- Main difficulty: Agent doesn't know the configuration
- Reason about configuration
- Knowledge evolves as new percepts arrive and actions are taken.

Examples of reasoning

- The agent's initial knowledge base contains the rules of the environment; in particular, it knows that it is in [1,1] and that [1,1] is a safe square; we denote that with an "A" and "OK," respectively, in square [1,1].

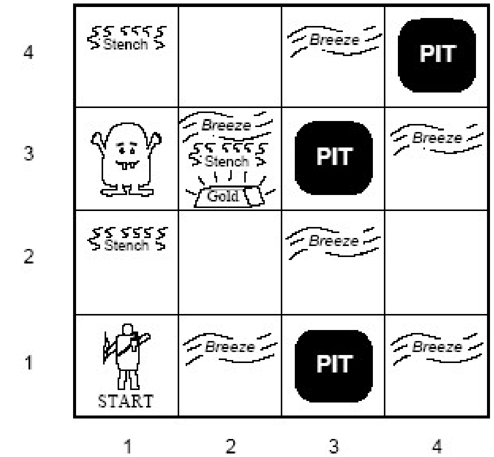
- The first percept is *[None,None,None,None,None],* from which the agent can conclude that its neighboring squares, [1,2] and [2,1], are free of dangers—they are OK

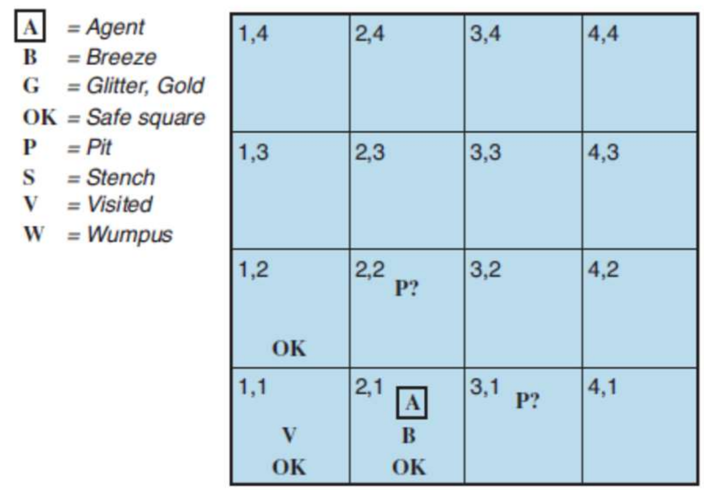




Examples of reasoning

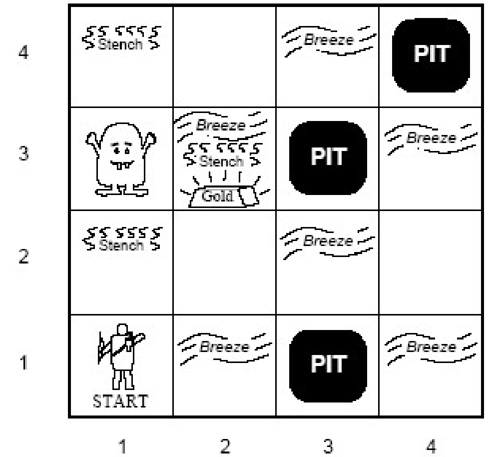
- The agent perceives a breeze (denoted by “B”) in [2,1], so there must be a pit in a neighboring square. The pit cannot be in [1,1], by the rules of the game, so there must be a pit in [2,2] or [3,1] or both.
- The notation “P?” indicates a possible pit in those squares. At this point, there is only one known square that is OK and that has not yet been visited.
- So the prudent agent will turn around, go back to [1,1], and then proceed to [1,2].

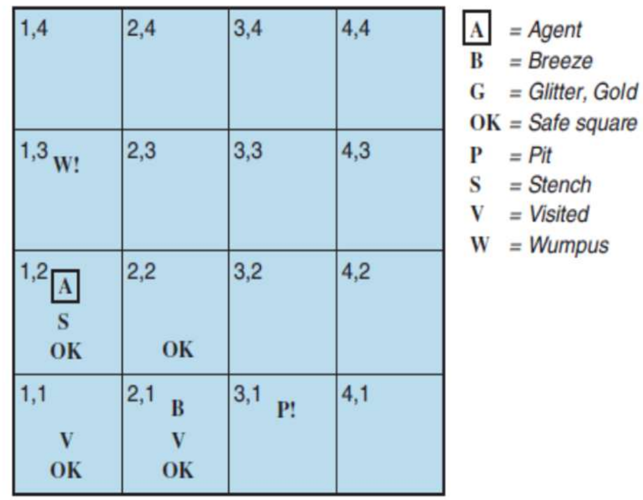




Examples of reasoning

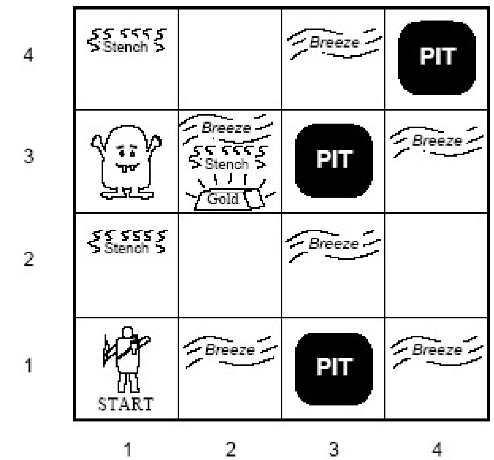
- The agent perceives a stench in [1,2], resulting in the shown state of knowledge .
- The stench in [1,2] means that there must be a wumpus nearby. But the Wumpus cannot be in [1,1], by the rules of the game, and it cannot be in [2,2] (or the agent would have detected a stench when it was in [2,1]).
- Therefore, the agent can infer that the Wumpus is in [1,3]. The notation **W!** indicates this inference.
- Moreover, the lack of a breeze in [1,2] implies that there is no pit in [2,2]. Yet the agent has already inferred that there must be a pit in either [2,2] or [3,1], so this means it must be in [3,1].

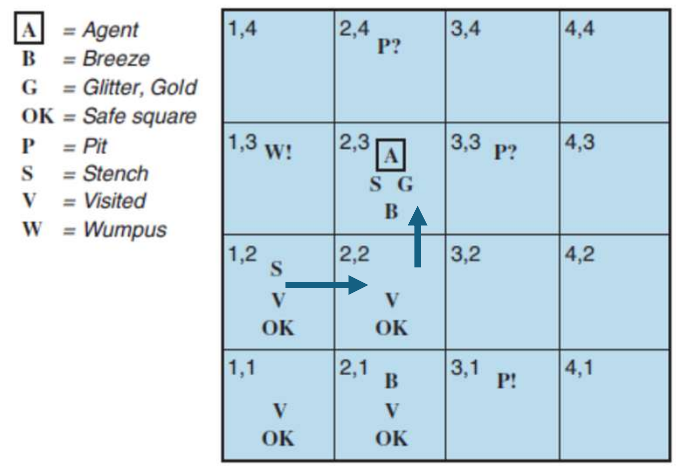




Examples of reasoning

- The agent has now proved to itself that there is neither a pit nor a wumpus in [2,2], so it is OK to move there.
- Then the agent turns and moves to [2,3],
- In [2,3], the agent detects a glitter, so it should grab the gold and then return home.





Logic

- Knowledge bases consist of sentences in a formal language
  - Syntax
    - Sentences are well formed
  - Semantics
    - The “meaning” of the sentence
    - ***The truth of each sentence with respect to each possible world (model)***

- Example:

  x + 2 >= y is a sentence

  x2 + y > is not a sentence

  x + 2 >= y is true iff x + 2 is no less than y

  x + 2 >= y is true in a world where x = 7, y=1

  x + 2 >= y is false in world where x = 0, y =6

Logic

- ***Entailment*** means that one thing follows logically from another

  $\alpha \models \beta$

- $\alpha \models \beta$ if in every model in which α is true, β is also true

- if α is true, then β must be true

- the truth of β is “contained” in the truth of α

Logic

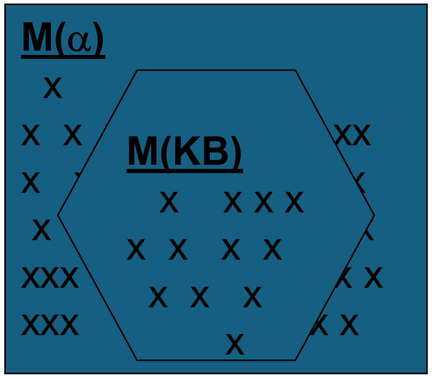
- Example:
  - α: "It is raining."
  - β: "The ground is getting wet."
  - In this case, α |= β. If it is raining, then it *must* be true that the ground is getting wet. There's no scenario where it's raining, and the ground remains completely dry.

- Example:

  $x + y = 4$ entails $4 = x + y$

Logic

- A model is a formally structured world with respect to which truth can be evaluated
  - M is a model of sentence α if α is true in m
  - **model** is a truth assignment in which the sentence α evaluates to **true**.



- Then KB $\models \alpha$ if $M(KB) \subseteq M(\alpha)$

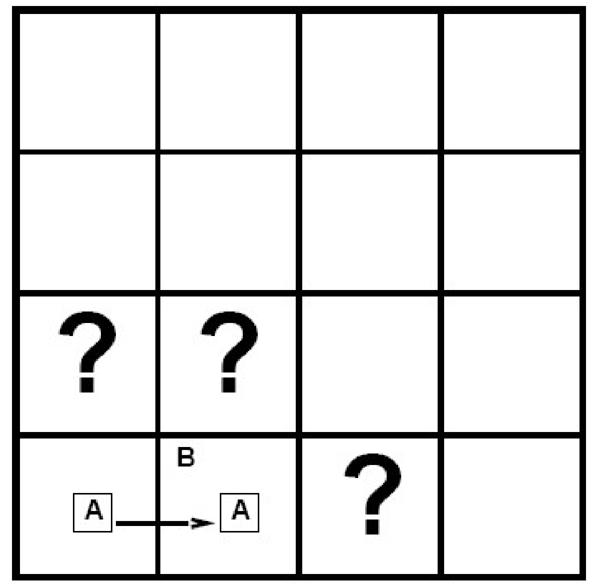
A	B	C	$A \wedge B$	$A \wedge C$	$B \wedge C$
F	F	F	F	F	F
F	F	T	F	F	F
F	T	F	F	F	F
F	T	T	F	F	T
T	F	F	F	F	F
T	F	T	F	T	F
T	T	F	T	F	F
T	T	T	T	T	T

$A \wedge C$, C does not entail $B \wedge C$

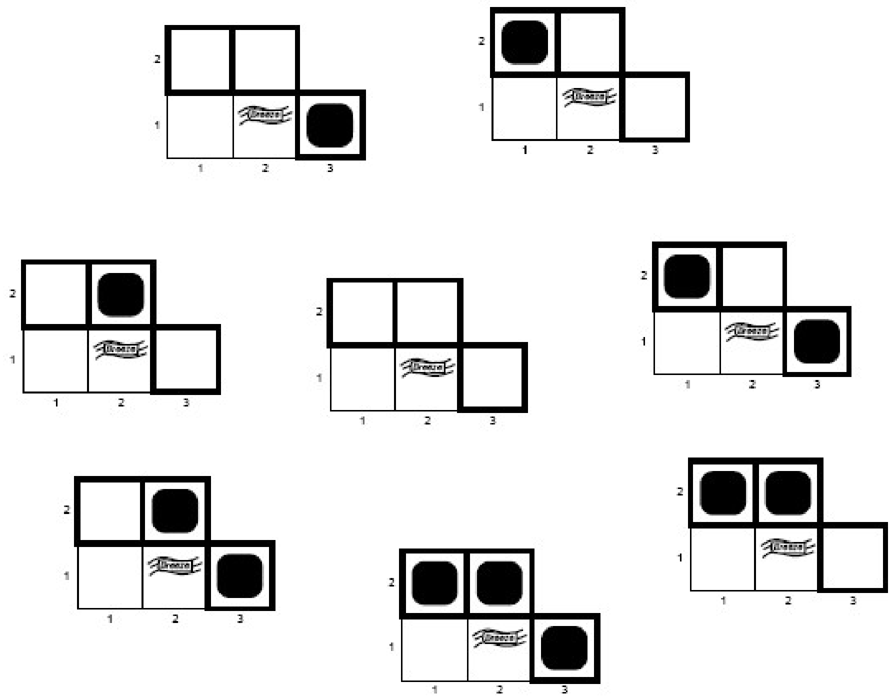
A,B, Entails $A \wedge B$

Logic

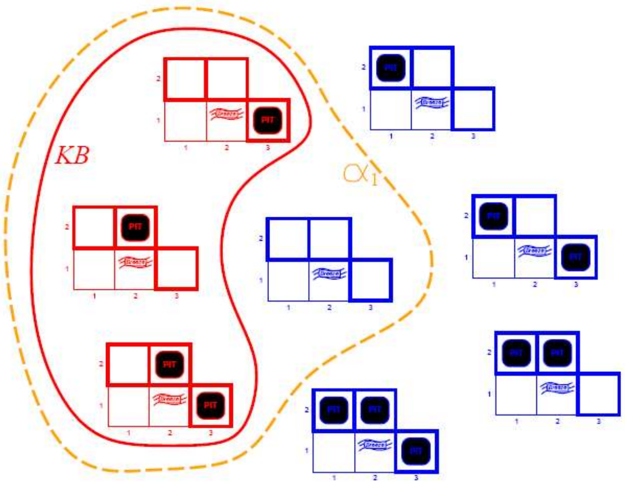
- Entailment in Wumpus World
- Situation after detecting nothing in [1,1], moving right, breeze in [2,1]
- Consider possible models for ? assuming only pits
- 3 Boolean choices => 8 possible models



Logic

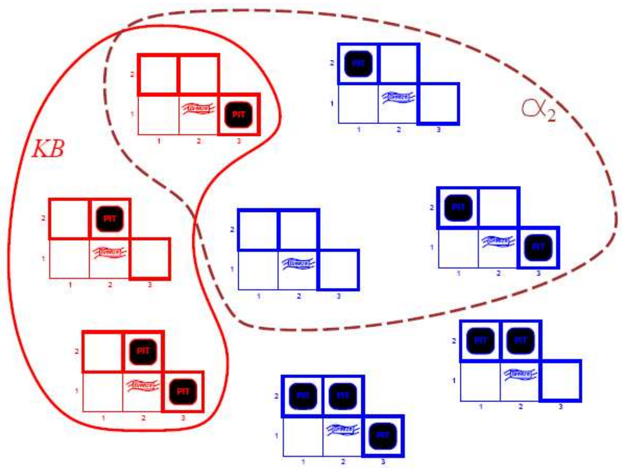


Logic



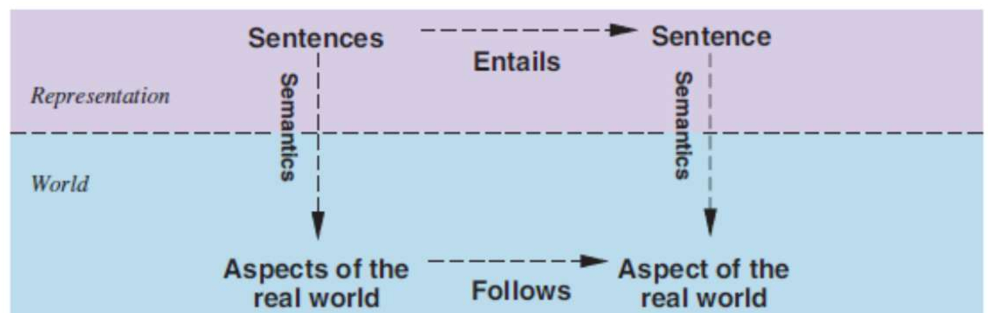
- KB = wumpus world rules + observations
- a_1 = "[1,2] is safe", KB |= a_1, proved by model checking

Logic



- KB = wumpus world rules + observations

- a_2 = "[2,2] is safe", KB ¬|= a_2 proved by model checking



Logic

- ***<u>Inference</u>*** is the process of deriving a specific sentence from a KB (where the sentence must be entailed by the KB)
  - KB $\vdash_i$ *a* = sentence *a* can be derived from KB by procedure i
- "KB's are a haystack"
  - Entailment = needle in haystack
  - Inference = finding it

Logic

- Soundness
  - i is sound if…
  - whenever KB $|-_i$ α is true, KB $|=$ α is true
- Completeness
  - i is complete if
  - whenever KB $|=$ α is true, KB $|-_i$ α is true
- 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
- We will define a logic (first-order logic) which is expressive enough to say almost anything of interest, and for which there exists a sound and complete inference procedure.

Propositional Logic

- AKA Boolean Logic
- False and True
- Proposition symbols P1, P2, etc are sentences

- NOT: If S1 is a sentence, then $\neg$S1 is a sentence (negation)

- AND: If S1, S2 are sentences, then S1 $\wedge$ S2 is a sentence (conjunction)

- OR: If S1, S2 are sentences, then S1 $\vee$ S2 is a sentence (disjunction)

- IMPLIES: If S1, S2 are sentences, then S1 $\Rightarrow$ S2 is a sentence (implication)

- IFF: If S1, S2 are sentences, then S1 $\Leftrightarrow$ S2 is a sentence (biconditional)

Propositional Logic

P	Q	¬P	P∧Q	P∨Q	P⇒Q ¬P∨Q	P⇔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*

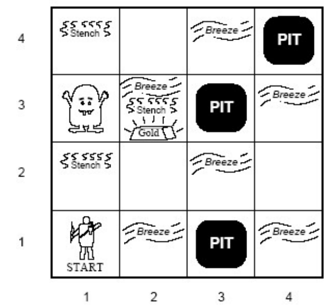
OPERATOR PRECEDENCE : $\neg, \wedge, \vee, \Rightarrow, \Leftrightarrow$

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"



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

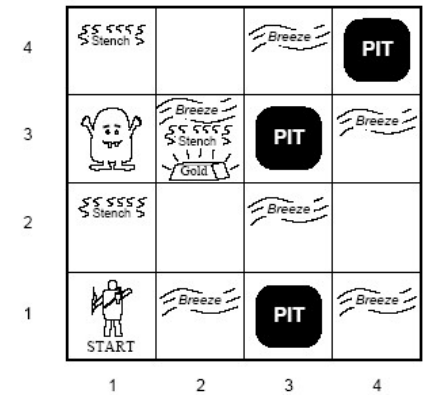
$$B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$$

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

A Simple Knowledge Base Sentences

$P_{x,y}$ is true if there is a pit in $[x, y]$.
$W_{x,y}$ is true if there is a wumpus in $[x, y]$, dead or alive.
$B_{x,y}$ is true if there is a breeze in $[x, y]$.
$S_{x,y}$ is true if there is a stench in $[x, y]$.
$L_{x,y}$ is true if the agent is in location $[x, y]$.

$$R_1 : \quad \neg P_{1,1} .$$
$$R_2 : \quad B_{1,1} \quad \Leftrightarrow \quad (P_{1,2} \vee P_{2,1}) .$$
$$R_3 : \quad B_{2,1} \quad \Leftrightarrow \quad (P_{1,1} \vee P_{2,2} \vee P_{3,1}) .$$
$$R_4 : \quad \neg B_{1,1} .$$
$$R_5 : \quad B_{2,1} .$$



Truth Table Approach

$B_{1,1}$	$B_{2,1}$	$P_{1,1}$	$P_{1,2}$	$P_{2,1}$	$P_{2,2}$	$P_{3,1}$	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*

Figure 7.9 A truth table constructed for the knowledge base given in the text. KB is true if R_1 through R_5 are true, which occurs in just 3 of the 128 rows (the ones underlined in the right-hand column). In all 3 rows, $P_{1,2}$ is false, so there is no pit in [1,2]. On the other hand, there might (or might not) be a pit in [2,2].

- **KB consists of sentences R_1 thru R_5**
- **$R_1 \wedge R_2 \wedge R_3 \wedge R_4 \wedge R_5$**

Truth Table Approach

```
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 ← a list of the proposition symbols in KB and α
  return TT-CHECK-ALL(KB, α, symbols, { })

function TT-CHECK-ALL(KB, α, symbols, model) returns true or false
  if EMPTY?(symbols) then
     if PL-TRUE?(KB, model) then return PL-TRUE?(α, model)
     else return true       //  when KB is false, always return true
  else
     P ← FIRST(symbols)
     rest ← REST(symbols)
     return (TT-CHECK-ALL(KB, α, rest, model ∪ {P = true})
             and
             TT-CHECK-ALL(KB, α, rest, model ∪ {P = false }))
```

- Depth-first enumeration of all models is sound and complete
- For *n* symbols, time complexity is $O(2^n)$, space complexity is $O(n)$

Inference Approach

- To prove that **KB |= α** Start from KB
- Infer new sentences that are true from existing KB sentences
- Repeat till alpha is proved (inferred true) or no more sentences can be proved

Equivalence, Validity, Satisfiability

A sentence is valid if it is true in all models. For example, the sentence P∨¬P is valid.

Two sentences are logically equivalent iff true in same models:

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

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

Equivalence, Validity, Satisfiability

- A sentence is valid if it is true in all models
  - e.g. True, $A \lor \neg A$, $A \Rightarrow A$, $(A \land (A \Rightarrow B)) \Rightarrow B$
- Validity is connected to inference via the Deduction Theorem
  - KB |- α iff (KB $\Rightarrow \alpha$) is valid
- A sentence is satisfiable if it is True in some model
  - e.g. $A \lor B$,
- A sentence is unstatisfiable if it is True in no models
  - e.g. $A \land \neg A$
- Satisfiability is connected to inference via the following
  - KB |= α iff (KB $\land \neg\alpha$) is unsatisfiable
  - proof by contradiction

Reasoning Patterns

- Inference Rules
  - Patterns of inference that can be applied to derive chains of conclusions that lead to the desired goal.

- Modus Ponens
  - Given: $S1 \Rightarrow S2$ and S1, derive S2

- And-Elimination
  - Given: $S1 \wedge S2$, derive S1
  - Given: $S1 \wedge S2$, derive S2

- DeMorgan's Law
  - Given: $\neg(A \vee B)$ derive $\neg A \wedge \neg B$
  - Given: $\neg(A \wedge B)$ derive $\neg A \vee \neg B$

Reasoning Patterns

- And Elimination

$$\frac{\alpha \wedge \beta}{\alpha}$$

- From a conjunction, any of the conjuncts can be inferred

- (WumpusAhead $\wedge$ WumpusAlive), WumpusAlive can be inferred

- Modus Ponens

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

- Whenever sentences of the form $\alpha \Rightarrow \beta$ and α are given, then sentence β can be inferred

- (WumpusAhead $\wedge$ WumpusAlive) $\Rightarrow$ Shoot and (WumpusAhead $\wedge$ WumpusAlive), Shoot can be inferred

Example Proof By Deduction

- Knowledge

S1: $B_{22} \Leftrightarrow (P_{21} \vee P_{23} \vee P_{12} \vee P_{32})$ *rule*

S2: $\neg B_{22}$ *observation*

- Inferences

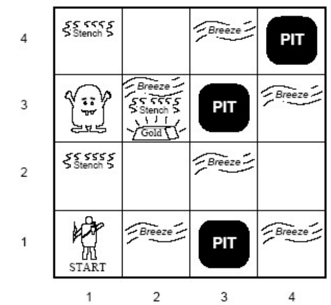
S3: $(B_{22} \Rightarrow (P_{21} \vee P_{23} \vee P_{12} \vee P_{32})) \wedge$
$((P_{21} \vee P_{23} \vee P_{12} \vee P_{32}) \Rightarrow B_{22})$ *[S1,bi elim]*

S4: $((P_{21} \vee P_{23} \vee P_{12} \vee P_{32}) \Rightarrow B_{22})$ *[S3, and elim]* $\dfrac{\alpha \wedge \beta}{\alpha}$

S5: $(\neg B_{22} \Rightarrow \neg(P_{21} \vee P_{23} \vee P_{12} \vee P_{32}))$ *[contrapos]*

S6: $\neg(P_{21} \vee P_{23} \vee P_{12} \vee P_{32})$ $\dfrac{\alpha \Rightarrow \beta, \quad \alpha}{\beta}$ *[S2,S6, MP]*

S7: $\neg P_{21} \wedge \neg P_{23} \wedge \neg P_{12} \wedge \neg P_{32}$ *[S6, DeMorg]*



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 $\vee$
$\neg(\neg\alpha) \equiv \alpha$ double-negation elimination
$(\alpha \Rightarrow \beta) \equiv (\neg\beta \Rightarrow \neg\alpha)$ contraposition
$(\alpha \Rightarrow \beta) \equiv (\neg\alpha \vee \beta)$ implication elimination
$(\alpha \Leftrightarrow \beta) \equiv ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha))$ 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 $\wedge$

Evaluation of Deductive Inference

- Sound
  - Yes, because the inference rules themselves are sound. (This can be proven using a truth table argument).
- Complete
  - If we allow all possible inference rules, we're searching in an infinite space, hence not complete
  - If we limit inference rules, we run the risk of leaving out the necessary one…
- Monotonic
  - If we have a proof, adding information to the DB will not invalidate the proof

Resolution Approach

- Resolution allows a complete inference mechanism (search-based) using only one rule of inference
- Resolution rule:
  - Given: $P_1 \vee P_2 \vee P_3 \ldots \vee P_n$, and $\neg P_1 \vee Q_1 \ldots \vee Q_m$
  - Conclude: $P_2 \vee P_3 \ldots \vee P_n \vee Q_1 \ldots \vee Q_m$
    Complementary literals P_1 and $\neg P_1$ "cancel out"

- Resolution inference rule (for CNF, conjunction of disjunctions):

$$\frac{l_i \vee \ldots \vee l_k, \qquad\qquad m_1 \vee \ldots \vee m_n}{l_i \vee \ldots \vee l_{i-1} \vee l_{i+1} \vee \ldots \vee l_k \vee m_1 \vee \ldots \vee m_{j-1} \vee m_{j+1} \vee \ldots \vee m_n}$$

Resolution in Wumpus World

- There is a pit at 2,1 or 2,3 or 1,2 or 3,2
  - $P_{21} \vee P_{23} \vee P_{12} \vee P_{32}$
- There is no pit at 2,1
  - $\neg P_{21}$
- Therefore (by resolution) the pit must be at 2,3 or 1,2 or 3,2
  - $P_{23} \vee P_{12} \vee P_{32}$

Proof using Resolution

- To prove a fact α, That is, to show that KB |= α, we show that (KB∧ ¬α) is unsatisfiable. We do this by proving a contradiction.
- First, (KB∧¬α) is converted into Conjunctive Normal Form (CNF). Then, the resolution rule is applied to the resulting clauses.
  - conjunction of clauses (clauses include disjunctions of literals)

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

Disjunctive normal form (DNF)

  - Disjunction of terms (terms include conjunction of literals)

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

- 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.
- 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; or,
  - Two clauses resolve to yield the empty clause, in which case KB entails α.

CNF Example

1. $B_{22} \Leftrightarrow (P_{21} \vee P_{23} \vee P_{12} \vee P_{32})$

2. Eliminate $\Leftrightarrow$, replacing with two implications
   $(B_{22} \Rightarrow (P_{21} \vee P_{23} \vee P_{12} \vee P_{32})) \wedge ((P_{21} \vee P_{23} \vee P_{12} \vee P_{32}) \Rightarrow B_{22})$

3. Replace implication $(A \Rightarrow B)$ by $\neg A \vee B$
   $(\neg B_{22} \vee (P_{21} \vee P_{23} \vee P_{12} \vee P_{32})) \wedge (\neg(P_{21} \vee P_{23} \vee P_{12} \vee P_{32}) \vee B_{22})$

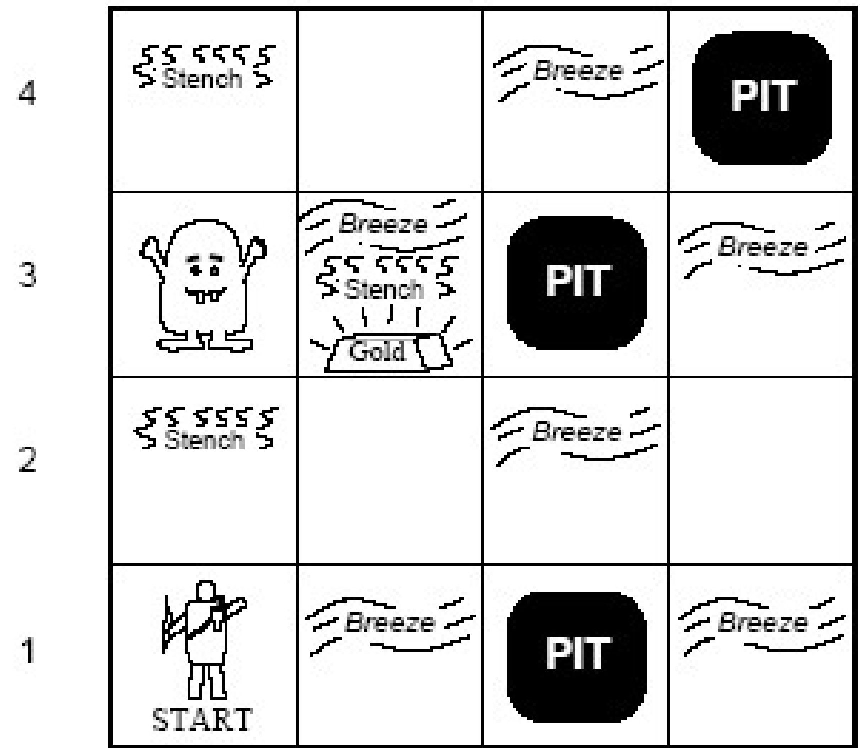
4. Move $\neg$ "inwards" (unnecessary parens removed)
   $(\neg B_{22} \vee P_{21} \vee P_{23} \vee P_{12} \vee P_{32}) \wedge ((\neg P_{21} \wedge \neg P_{23} \wedge \neg P_{12} \wedge \neg P_{32}) \vee B_{22})$

5. Distributive Law
   $(\neg B_{22} \vee P_{21} \vee P_{23} \vee P_{12} \vee P_{32}) \wedge (\neg P_{21} \vee B_{22}) \wedge (\neg P_{23} \vee B_{22}) \wedge (\neg P_{12} \vee B_{22}) \wedge (\neg P_{32} \vee B_{22})$

(Final result has 5 clauses)

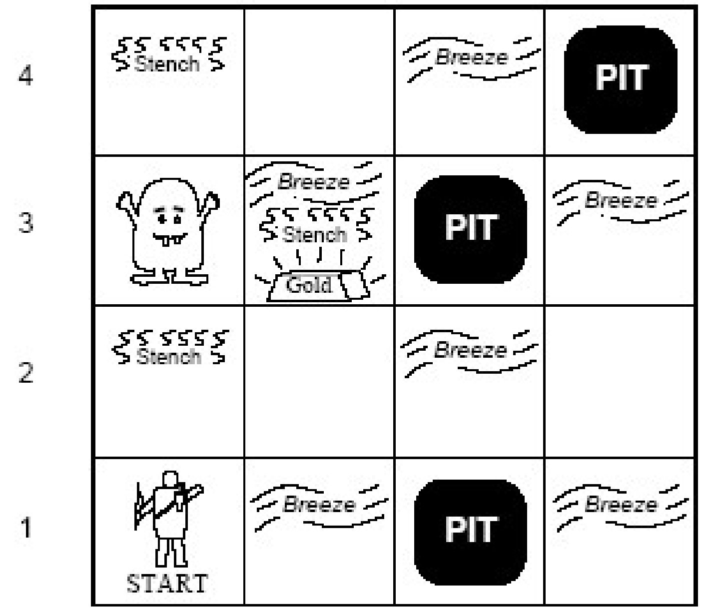
Simple Resolution Example

- When the agent is in 1,1, there is no breeze, so there can be no pits in neighboring squares
- Percept: $\neg$B11
- Prove: $\neg$P12



Resolution example

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

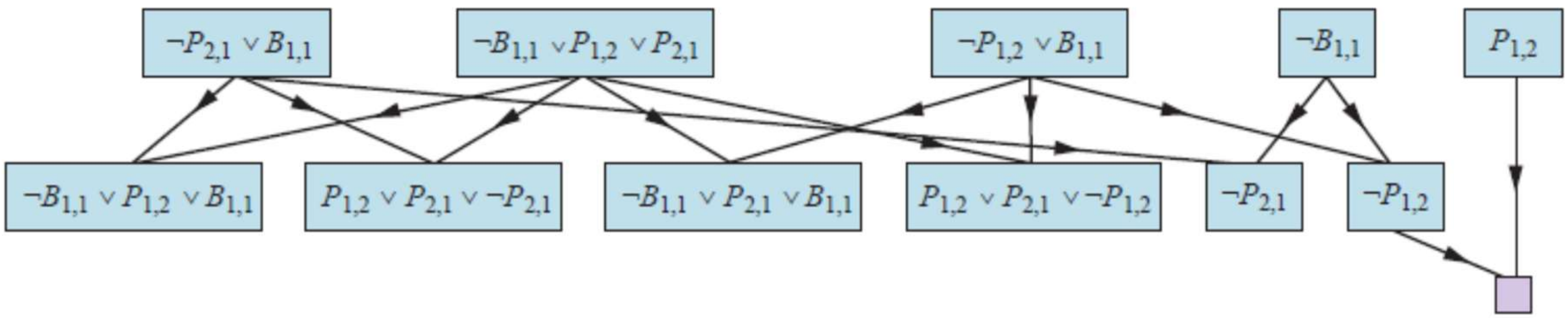


You can resolve only one pair of complementary literals at a time. For example, we can resolve P and $\neg P$ to deduce

$$\frac{P \vee \neg Q \vee R, \qquad \neg P \vee Q}{\neg Q \vee Q \vee R},$$

Resolution example

- $KB = (B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})) \wedge \neg B_{1,1}$
- $CNF = (\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg B_{1,1}) \wedge (\neg P_{1,2} \vee B_{1,1}) \wedge (\neg P_{2,1} \vee B_{1,1})$
- $\alpha = \neg P_{1,2}$ **We show that (KB∧ ¬α) is unsatisfiable**



- 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.
- 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; or,
  - Two clauses resolve to yield the empty clause, in which case KB entails α.

Evaluation of Resolution

- Resolution is sound
  - Because the resolution rule is true in all cases
- Resolution is complete
  - Provided a complete search method is used to find the proof, if a proof can be found it will
  - Note: you must know what you're trying to prove in order to prove it!
- Resolution is exponential
  - The number of clauses that we must search grows exponentially…

Horn Clauses

- A Horn Clause is a CNF clause with exactly one positive literal
  - The positive literal is called the head
  - The negative literals are called the body
  - $(\neg L_{1,1} \vee \neg \mathit{Breeze} \vee B_{1,1})$ can be written as the implication $(L1,1 \wedge \mathit{Breeze}) \Rightarrow B_{1,1}$.
- Horn Clauses form the basis of forward and backward chaining
- The Prolog language is based on Horn Clauses
- Deciding entailment with Horn Clauses is *linear in the size of the knowledge base*

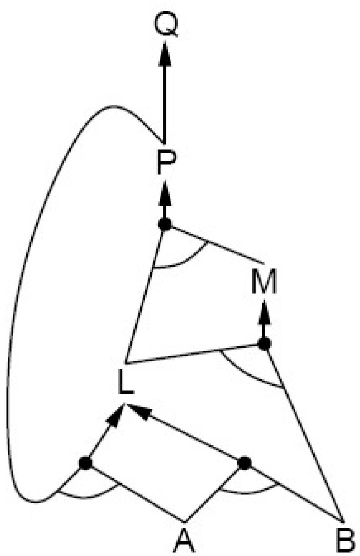
Reasoning with Horn Clauses

- Forward Chaining
  - For each new piece of data, generate all new facts, until the desired fact is generated
  - Data-directed reasoning
- Backward Chaining
  - To prove the goal, find a clause that contains the goal as its head, and prove the body recursively
  - (Backtrack when you chose the wrong clause)
  - Goal-directed reasoning

Forward Chaining

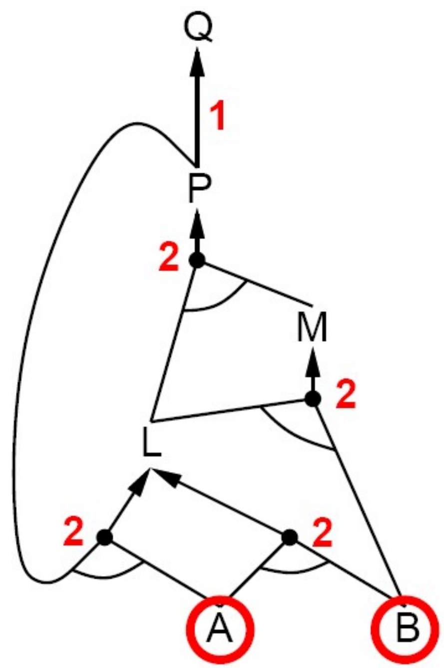
- AND-OR Graph
  - multiple links joined by an arc indicate conjunction – every link must be proved
  - multiple links without an arc indicate disjunction – any link can be proved

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



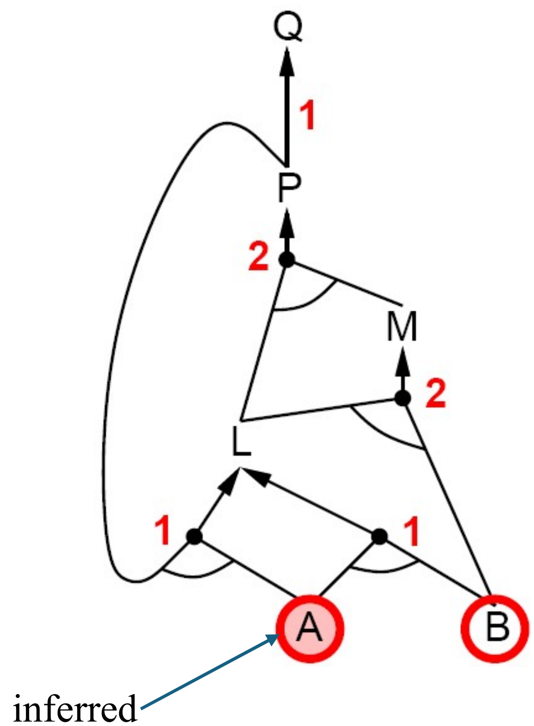
Forward Chaining

Count the number of facts in the antecedent of the rule



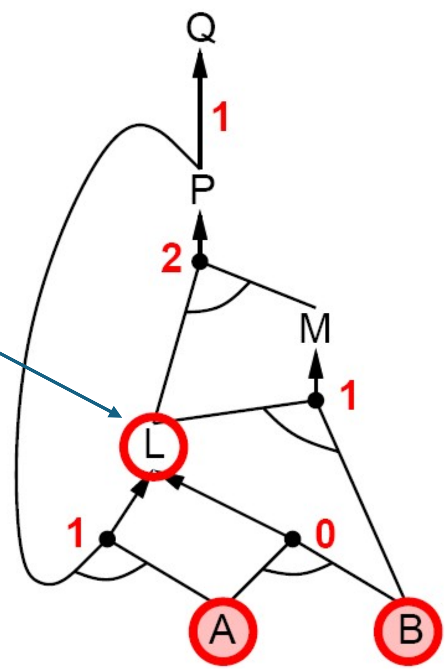
Forward Chaining

Inferred facts decrease the count



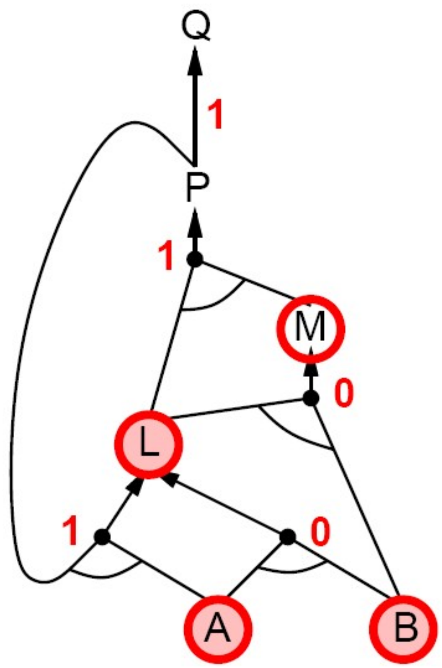
Forward Chaining

New facts can be inferred when the count associated with a rule becomes 0

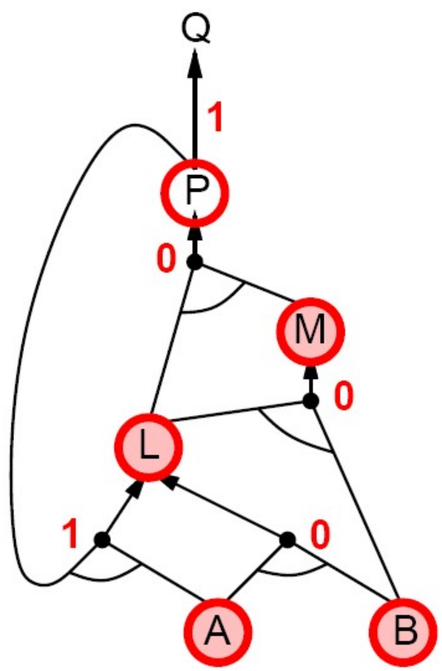


Forward Chaining

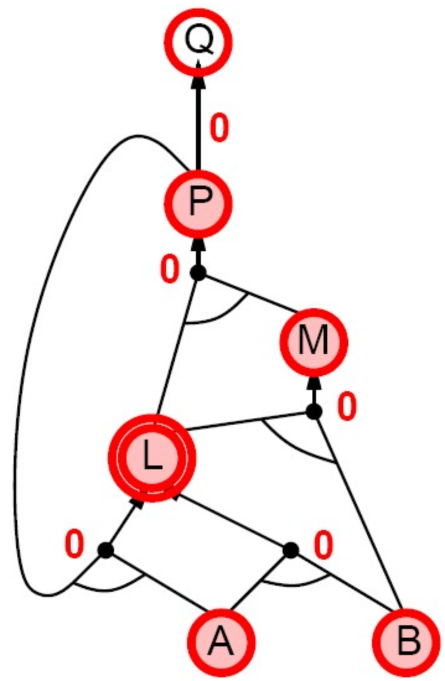
Inferred facts decrease the count



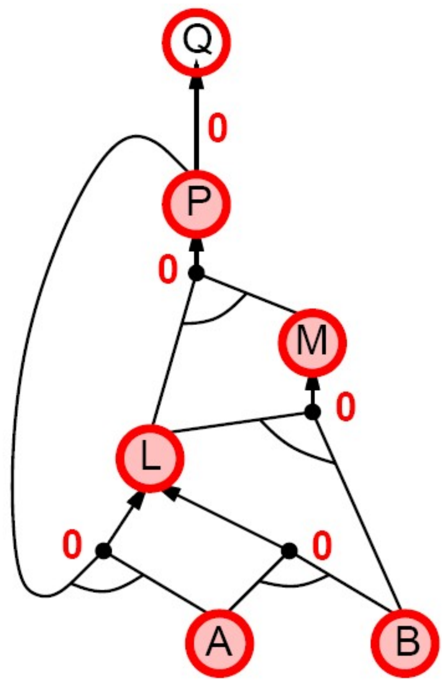
Forward Chaining



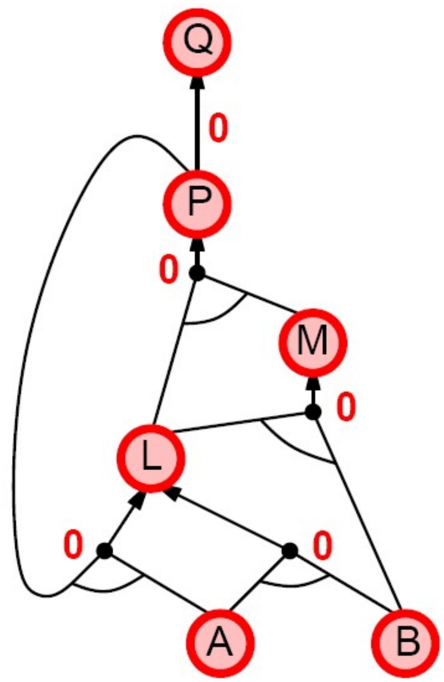
Forward Chaining



Forward Chaining



Forward Chaining



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
  - 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 for propositional logic
  Forward, backward chaining are linear-time, complete for Horn clauses
- Propositional logic lacks expressive power