Lec5: Knowledge and logic reasoning 1: Logical Agent

Outline

- Logical agent
- Logical language
  - Propositional logic
  - First order logic (mainly in the next lecture)
- How to reason with rules and facts?

Logical agent

- Use a knowledge base to keep track of things
- Can TELL it facts&rules, or ASK for answer
- For example:
  - TELL: 2 friends are either both smokers or non-smokers (rule)
  - TELL: smoking causes cancer (rule)
  - TELL: Anna and Bob are friends (fact)
  - TELL: Bob is a smoker (fact)
  - ASK: Will Anna have cancer?

Knowledge-Based Agents

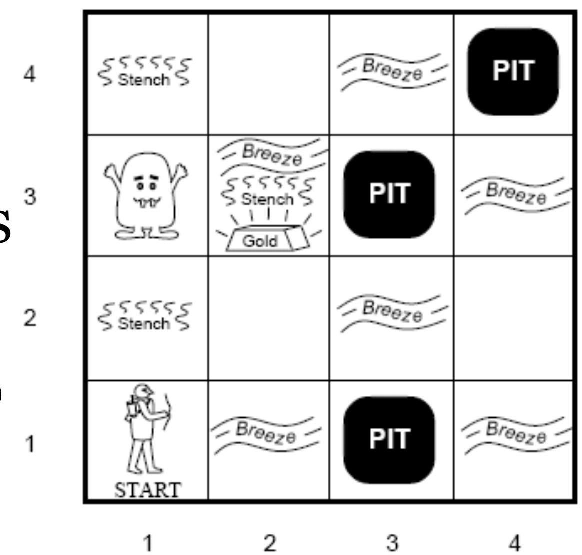
- KB = knowledge base
  - A set of sentences or facts
  - e.g., a set of statements in a logic language
  - v.s. database (facts only)

- Inference
  - Deriving new sentences from old
  - e.g., using a set of logical statements to infer new ones

- A simple model for reasoning
  - Agent is told or perceives new evidence
    - E.g., A is true
  - Agent then infers new facts to add to the KB
    - E.g., KB = { A -> (B OR C) }, then given A and not C we can infer that B is true
    - B is now added to the KB even though it was not explicitly asserted, i.e., the agent inferred B

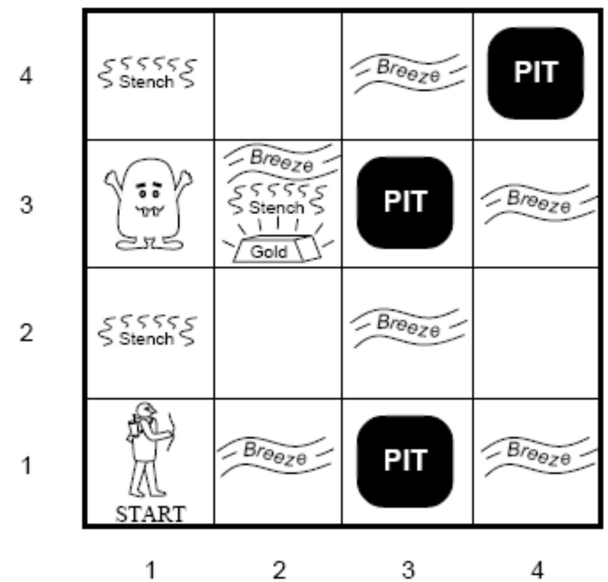
Wumpus World

- Environment
  - Cave of 4×4
  - Agent enters in [1,1]
  - 16 rooms
    - Wumpus: A deadly beast who kills anyone entering his room.
    - Pits: Bottomless pits that will trap you forever.
    - Gold



Wumpus World

- Agents Sensors:
  - Stench next to Wumpus
  - Breeze next to pit
  - Glitter in square with gold
  - Bump when agent moves into a wall
  - Scream from wumpus when killed



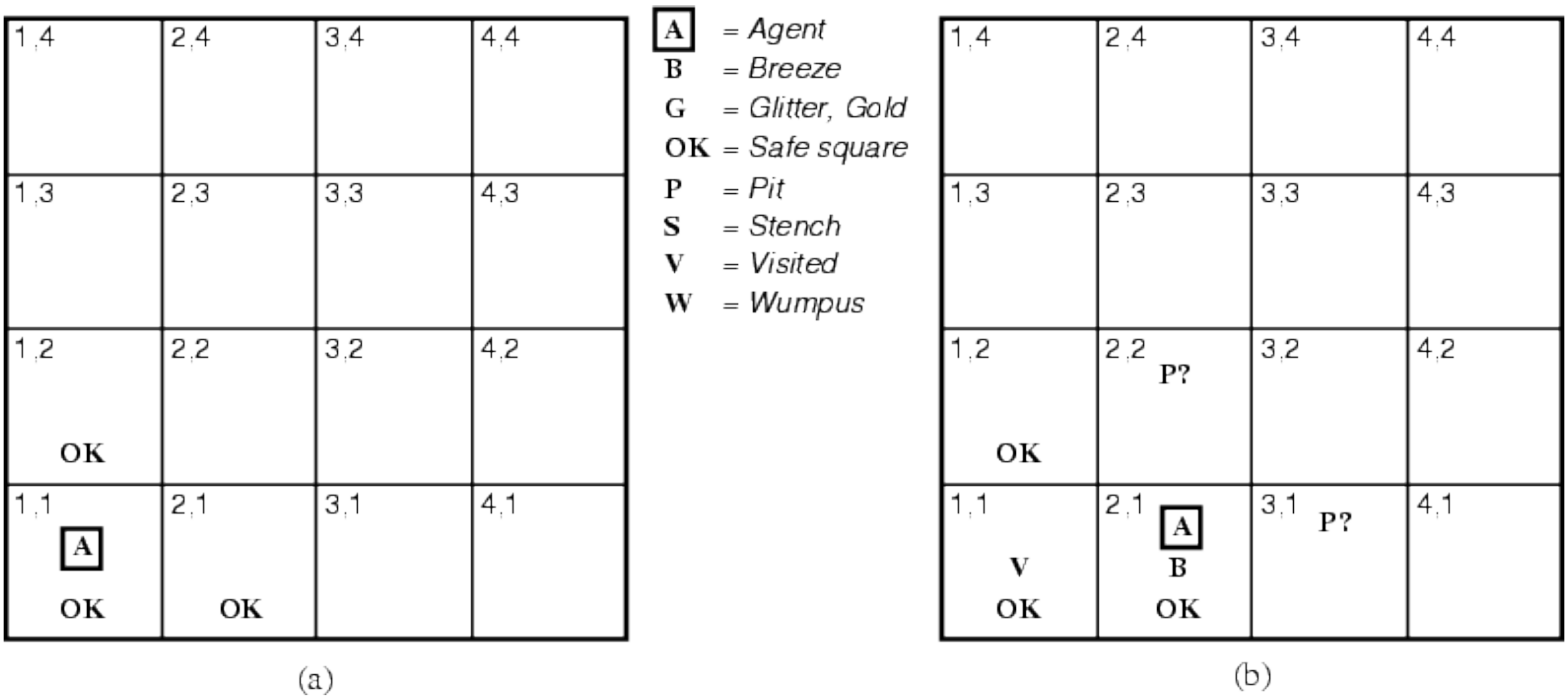
- Agents actions
  - Agent can move forward, turn left or turn right
  - Shoot, one shot

Wumpus world: formulation

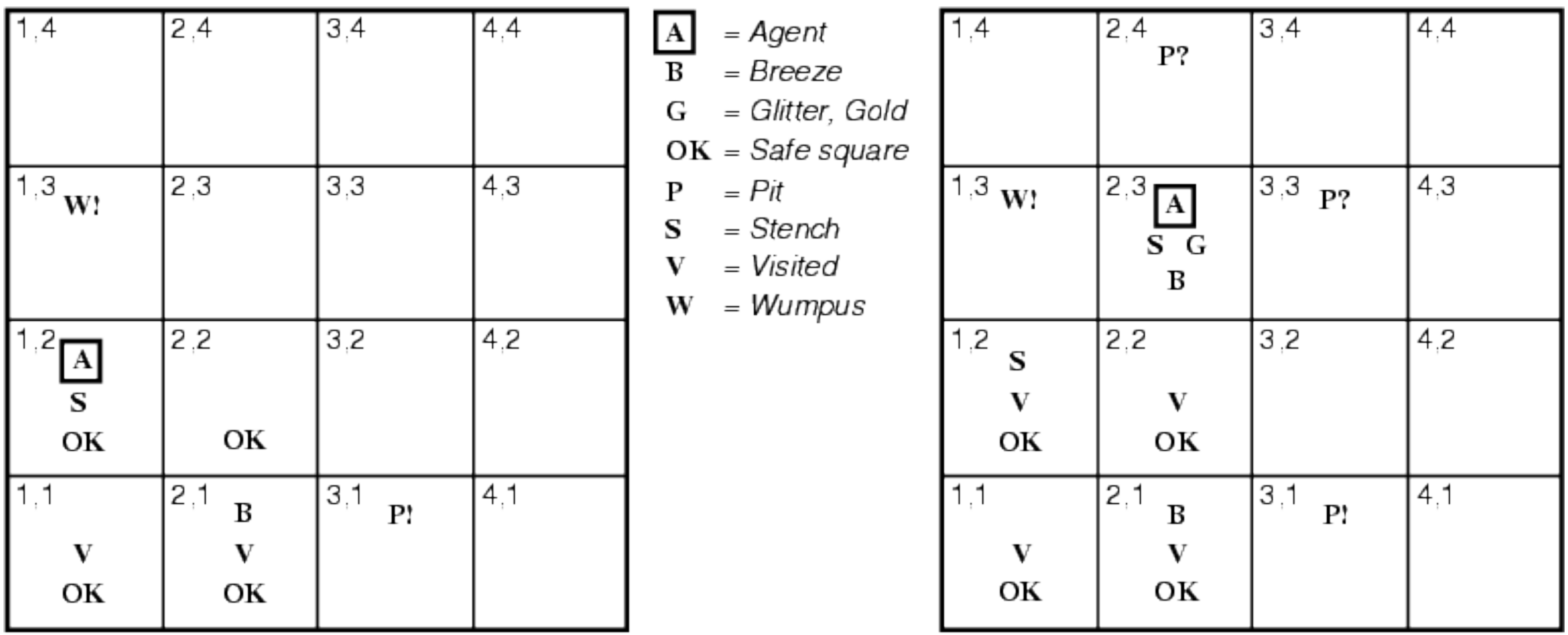
PEAS:

- **Performance measure**: +1000 for walk out w/ gold;-1000 for dying; -1 for each action, -10 for arrow
- **Environment**: 4×4 grid. Agent starts at [1,1], gold and pits randomly distributed etc
- **Actuator**: Agent can move up, down, left or right
- **Sensors**: {[Smell, Breeze, Glitter, Bump, Scream]}

The Wumpus agent’s first step



Later



1,4
2,4
3,4
4,4
1,3
W!
2,3
3,3
4,3
1,2
A
S
OK
2,2
OK
3,2
4,2
1,1
V
OK
2,1
B
V
OK
3,1
P!
4,1
A = Agent
B = Breeze
G = Glitter, Gold
OK = Safe square
P = Pit
S = Stench
V = Visited
W = Wumpus
1,4
2,4
P?
3,4
4,4
1,3
W!
2,3
A
S G
B
3,3
P?
4,3
1,2
S
V
OK
2,2
V
OK
3,2
4,2
1,1
V
OK
2,1
B
V
OK
3,1
P!
4,1

(a) (b)

What is a logical language?

- A formal language
  - KB = set of sentences
- Syntax
  - what sentences are legal (well-formed)
  - E.g., arithmetic
    - e.g. x+4=6 ✓; 4x=6+ ✗
    - X+2 >= y ✓ ; +x2y ✗
- Semantics
  - Loose meaning: the interpretation of each sentence
  - More precisely:
    - the rules for determining the truth of each sentence wrt to each possible world
  - e.g,
    - X+2 = y is true in a world where x=7 and y =9
    - X+2 = y is false in a world where x=7 and y =1
  - Note: standard logic – each sentence is T of F wrt eachworld
    - Fuzzy logic – allows for degrees of truth.

Logic --- Entailment

- Entailment is when a sentence follows another
- We say α entails β, written as $\alpha \vDash \beta$, if and only if (iff), in every model where α is true, β is also true.
- Examples
  1. $(X=0) \vDash (XY=0)$
  2. $(A = True) \vDash (A \vee B)$
  3. $(A \wedge B) \vDash (A \vee B)$

Propositional logic (a simple logic language)

- A branch of logic, with other names:
  - sentential calculus, sentential logic, or sometimes zeroth-order logic
- **Propositions**: special sentences that are either true or false (but not both)
- Which one below is a proposition?
  1. "Grass is green"
  2. "2+5=3"
  3. "Close the door"
  4. "Is it hot outside?"
  5. "x is greater than 2", where x is a variable representing a number
- Logical **connectives** (or **operator**)
  1. ¬ (not) ~
  2. ^ (And) & *
  3. v (or) | +
  4. => (implies) ->
  5. < => (if and only if) <->

Propositional logic: Syntax

- Propositional logic is the simplest logic – illustrates basic ideas
- Atomic sentences = single proposition symbols
  - E.g., P, Q, R
  - Special cases: True = always true, False = always false
- Complex sentences:
  - 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: Semantics

Each model/world specifies true or false for each proposition symbol

E.g. $P_{1,2}$ $P_{2,2}$ $P_{3,1}$

false 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 **every** sentence, e.g.,

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

Syntax v.s. Semantics

- **Syntax** defines allowable sentences
  - Atomic sentence: a single proposition symbol. P, Q
  - Literal: atomic sentence or negated atomic sentence. P, ¬ P
  - Complex sentence: build from simpler sentence(s) using parentheses and/or logical connectives. ((P<=> Q) v R)

- **Semantics** defines the rules for determining the truth of a sentence w.r.t. a model

  1. ¬ P is true iff P is false (in m)
  2. P^Q is true iff both P and Q are true (in m)   AND (Conjunction)
  3. P v Q is true iff either P or Q is true (in m)   OR (Disjunction)
  4. P => Q is true unless P is true and Q is false (in m). ¬ P v Q
  5. P < => Q is true iff P and Q are both true or both false (in m). In other words, P < => Q is true whenever both P => Q and Q => P are true.

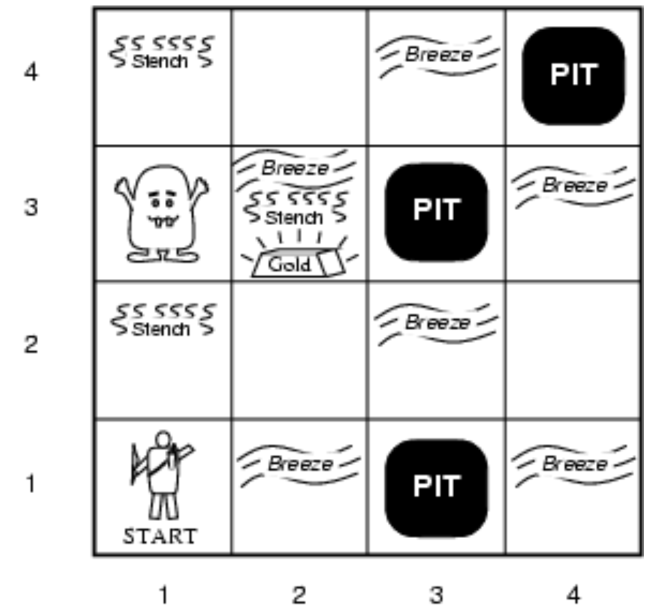
Wumpus world sentences

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].

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

$\neg B_{1,1}$

$B_{2,1}$



- "Pits cause breezes in adjacent squares"

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

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

- KB can be expressed as the conjunction of all of these sentences

- Note that these sentences are rather long-winded!
  - E.g., breeze "rule" must be stated explicitly for each square
  - First-order logic will allow us to define more general patterns.

More on Possible Worlds

- m is a model of a sentence α if α is true in m
- $M(\alpha)$ is the set of all models of α
- Possible worlds ~ models
  - Possible worlds: potentially real environments
  - Models: mathematical abstractions that establish the truth or falsity of every sentence
- Example:
  - x + y = 4, where x = #men, y = #women
  - Possible models = all possible assignments of integers to x and y.
  - For CSPs, possible model = complete assignment of values to variables.

Truth tables for connectives

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

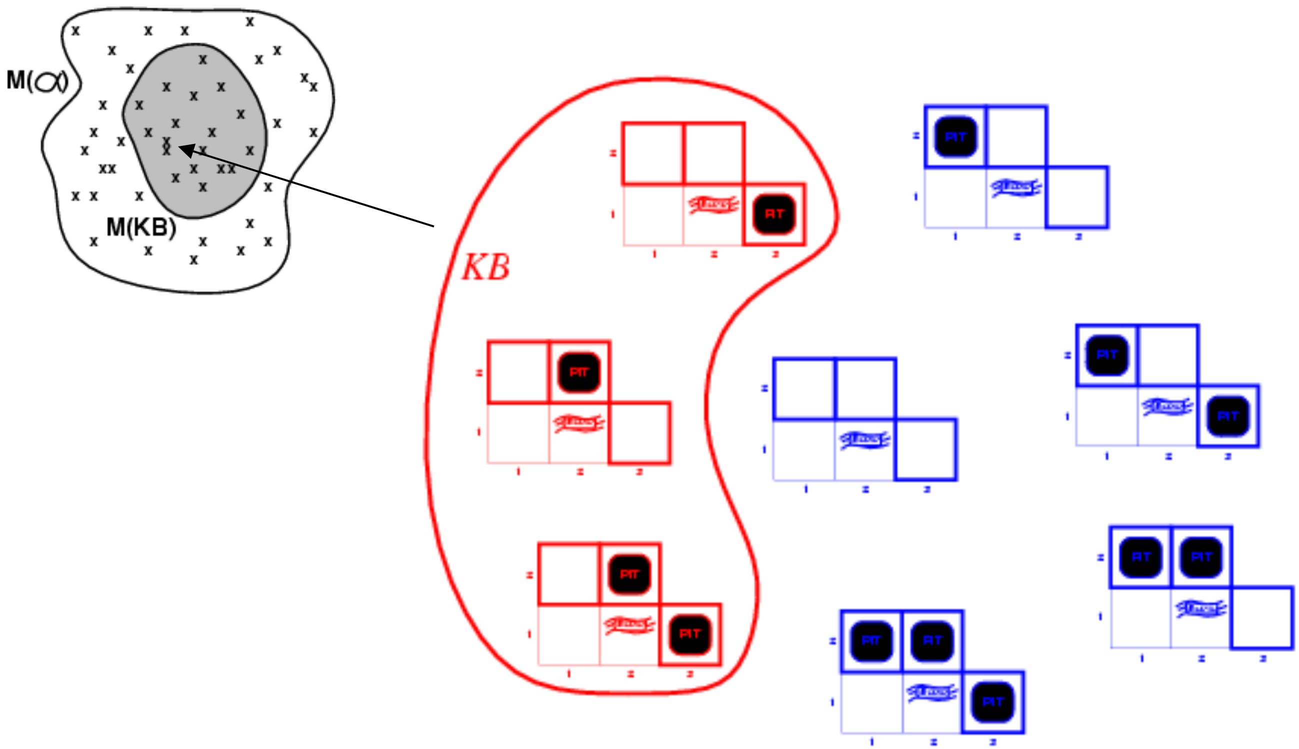
Truth tables for connectives

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

Implication is always true
when the premise is false

Why? P=>Q means "if P is true then I am claiming that Q is true otherwise no claim"
Only way for this to be false is if P is true and Q is false

Wumpus models



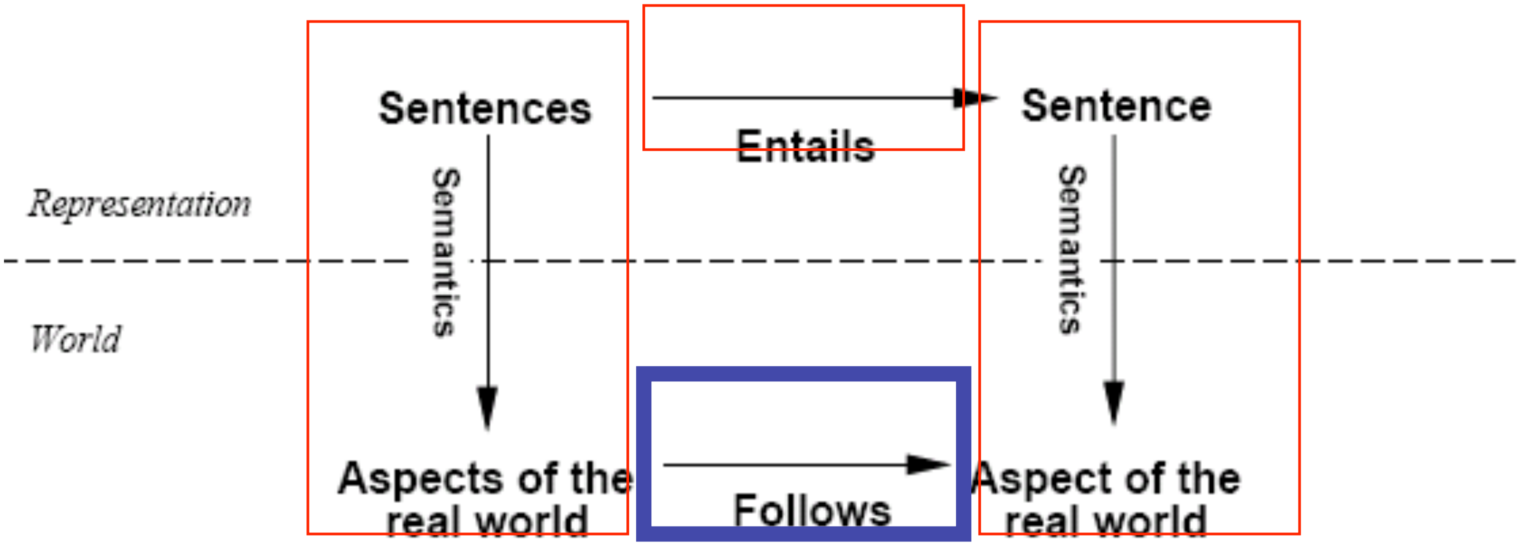
- *KB* = all possible wumpus-worlds consistent with the observations and the "physics" of the Wumpus world.

Listing of possible worlds for the Wumpus KB

α_1 = ”square [1,2] is safe”.
KB = detect nothing in [1,1], detect breeze in [2,1]

$B_{1,1}$	$B_{2,1}$	$P_{1,1}$	$P_{1,2}$	$P_{2,1}$	$P_{2,2}$	$P_{3,1}$	KB	α_1
false	*false*	*false*	*false*	*false*	*false*	*false*	*false*	*true*
false	*false*	*false*	*false*	*false*	*false*	*true*	*false*	*true*
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
false	*true*	*false*	*false*	*false*	*false*	*false*	*false*	*true*
false	*true*	*false*	*false*	*false*	*false*	*true*	<u>*true*</u>	<u>*true*</u>
false	*true*	*false*	*false*	*false*	*true*	*false*	<u>*true*</u>	<u>*true*</u>
false	*true*	*false*	*false*	*false*	*true*	*true*	<u>*true*</u>	<u>*true*</u>
false	*true*	*false*	*false*	*true*	*false*	*false*	*false*	*true*
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
true	*true*	*true*	*true*	*true*	*true*	*true*	*false*	*false*

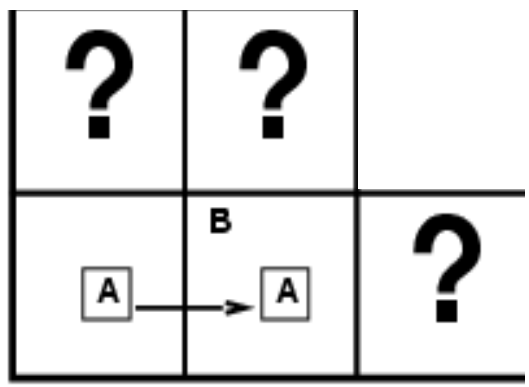
Schematic perspective



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.*

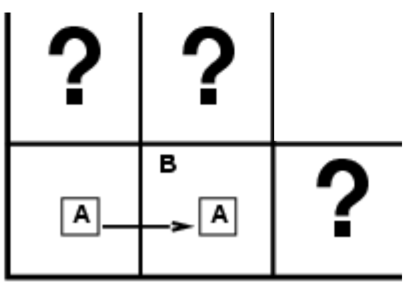
Entailment in the wumpus world

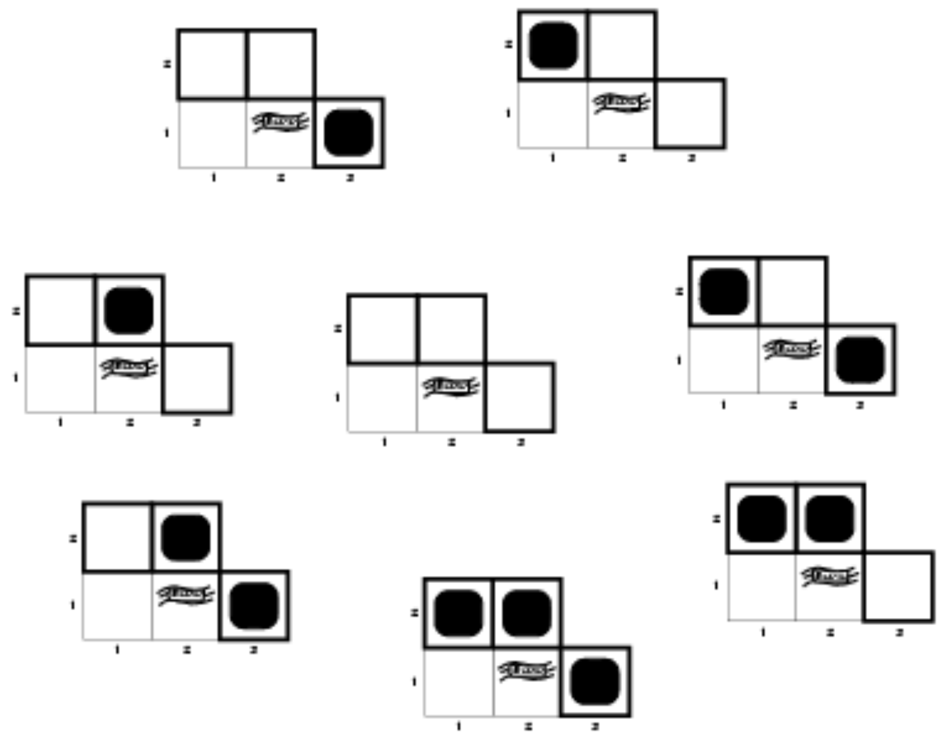
- Consider possible models for *KB* assuming only pits and a reduced Wumpus world
- Situation after detecting nothing in [1,1], moving right, detecting breeze in [2,1]



Wumpus models

All possible models in this reduced Wumpus world.

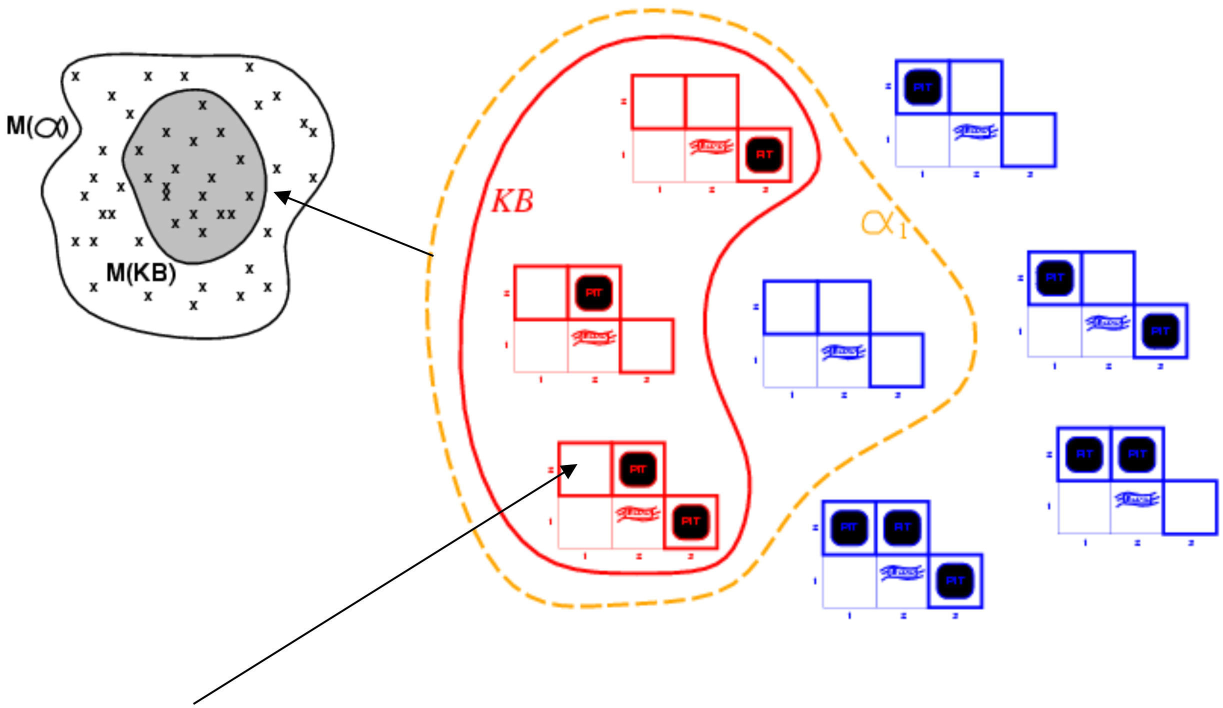




Inferring conclusions

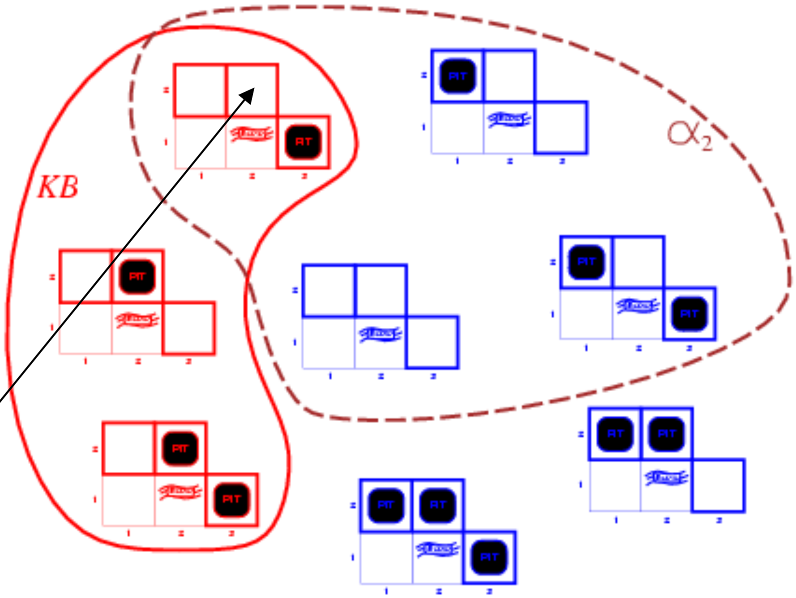
- Consider 2 possible conclusions given a KB
  - α_1 = "[1,2] is safe"
  - α_2 = "[2,2] is safe“
- One possible inference procedure
  - Start with KB
  - Model-checking
    - Check if KB $\models \alpha$ by checking if in all possible models where KB is true that α is also true
- Comments:
  - Model-checking enumerates all possible worlds
    - Only works on finite domains, will suffer from exponential growth of possible models

Wumpus models



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

Wumpus models



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

- There are some models entailed by KB where α_2 is false.

Logical inference

- The notion of entailment can be used for inference.
  - Model checking (see wumpus example): enumerate all possible models and check whether α is true.

- If an algorithm only derives entailed sentences it is called *sound* or *truth preserving*.
- A proof system is **sound** if whenever the system derives α from KB, it is also true that KB|= α
  - *E.g., model-checking is sound*

- Completeness : the algorithm can derive any sentence that is entailed.
- A proof system is **complete** if whenever KB|= α, the system derives α from KB.

Inference by enumeration

- We want to see if α is entailed by KB

- Enumeration of all models is sound and complete.

- But…for n symbols, time complexity is $O(2^n)$…

- We need a more efficient way to do inference
  - But worst-case complexity will remain exponential for propositional logic

Logical equivalence

- To manipulate logical sentences we need some rewrite rules.
- Two sentences are logically equivalent iff they are true in same models: $\alpha \equiv \beta$ iff $\alpha \models \beta$ and $\beta \models \alpha$

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

Normal Clausal Form

Eventually we want to prove: $\boxed{\text{Knowledge base KB entails sentence } \alpha}$

We first rewrite into conjunctive normal form (CNF).

A "conjunction of disjunctions"

literals

$$\underbrace{(A \lor \neg B)}_{\text{Clause}} \land \underbrace{(B \lor \neg C \lor \neg D)}_{\text{Clause}}$$

- **Theorem: Any KB can be converted into an equivalent CNF**.
- k-CNF: exactly k literals per clause

Example: Conversion to CNF

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

1. Eliminate $\Leftrightarrow$, replacing $\alpha \Leftrightarrow \beta$ with $(\alpha \Rightarrow \beta)\wedge(\beta \Rightarrow \alpha)$.
   $(B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$

2. Eliminate $\Rightarrow$, replacing $\alpha \Rightarrow \beta$ with $\neg\alpha \vee \beta$.
   $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg(P_{1,2} \vee P_{2,1}) \vee B_{1,1})$

3. Move $\neg$ inwards using de Morgan's rules and double-negation:
   $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge ((\neg P_{1,2} \wedge \neg P_{2,1}) \vee B_{1,1})$

4. Apply distributive law ($\wedge$ over $\vee$) and flatten:
   $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg P_{1,2} \vee B_{1,1}) \wedge (\neg P_{2,1} \vee B_{1,1})$

Horn Clauses

Horn Clause = A clause with at most 1 positive literal.
e.g. $A \vee \neg B \vee \neg C$

- Every Horn clause can be rewritten as an implication with a conjunction of positive literals in the premises and at most a single positive literal as a conclusion.

e.g. $B \wedge C \Rightarrow A$

- 1 positive literal: definite clause

- 0 positive literals: Fact or integrity constraint:
  e.g. $(\neg A \vee \neg B) \equiv (A \wedge B \Rightarrow False)$

- Psychologically natural: a condition implies (causes) a single fact.

- The basis of **logic programming** (the prolog language).

SWI Prolog. Prolog and the Semantic Web. Prolog Applications

Theorem Proving

- Logical equivalence
  - ❖ A≡B   iff A ⊨B and B ⊨ A
- Entail to implication (Deduction theorem)
  - ❖ A ⊨ B iff A => B is true in all models
    - ❖(being true in all models is called valid)
- How to check a sentence (such as A) true or false?
  - – Truth table (if not too many axiom sentences)
  - – Derive it using properties of logical operators
    - Think of ^ as *, v as +, almost everything in ordinary algebra follows (examples)
    - DeMorgan's Laws: ~ (p v q) = ~ p ^ ~ q, ~ (p ^ q) = ~ p v ~ q
    - Transposition: p => q = ~q => ~p
    - Exportation: (p ^ q) => r = p => (q => r)

Resolution

- A v B, ¬ A v C   entails B v C
  - A v B, ¬ A v C means (A v B) ^ (¬ A v C)
- Def in Wikipedia
- Applies only to a special form like above called CNF – Conjunction Normal Form
- All forms in propositional logic can be transformed to CNF
- Examples
- Automation by computers

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.

- The Logic Machine in Isaac Asimov's Foundation Series.

36