Logical Agents

1. Knowledge based agents

In AI, knowledge based use a process of reasoning over an internal representation of knowledge to agents to decide what actions to take.

A Knowledge base is a set of sentences. Each sentence is expressed in a language called a knowledge representation language and represents some assertion about the world. When the sentence is taken as being given without being derived from other sentences, we call it an **axiom**.

There must be a way to add new sentences to the knowledge base and a way to query what is known. The standard names for these operations are TELL and ASK, respectively. Both operations may involve inference—that is, deriving new sentences from old.

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

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

The sentences in the knowledge bases are expressed according to the syntax of the representation language, which specifies all the sentences that are well formed.

The notion of syntax is clear enough in ordinary arithmetic: "x+y = 4" is a **well-formed sentence**, whereas "x4y+ =" is not.

A logic must also define the semantics, or meaning, of sentences. The semantics defines the truth of each sentence with respect to each possible world.

For example, the semantics world for arithmetic specifies that the sentence "x+y=4" is true in a world where x is 2 and y is 2, but false in a world where x is 1 and y is 1.

In standard logics, every sentence must be either true or false in each possible world—there is no "in between."

If a sentence α is true in model m, we say that m satisfies α or sometimes m is a model of α. We use the notation M(α) to mean the set of all models of α.

2. Logical Reasoning

This involves the relation of logical entailment between sentences—the idea that a sentence follows logically from another sentence.

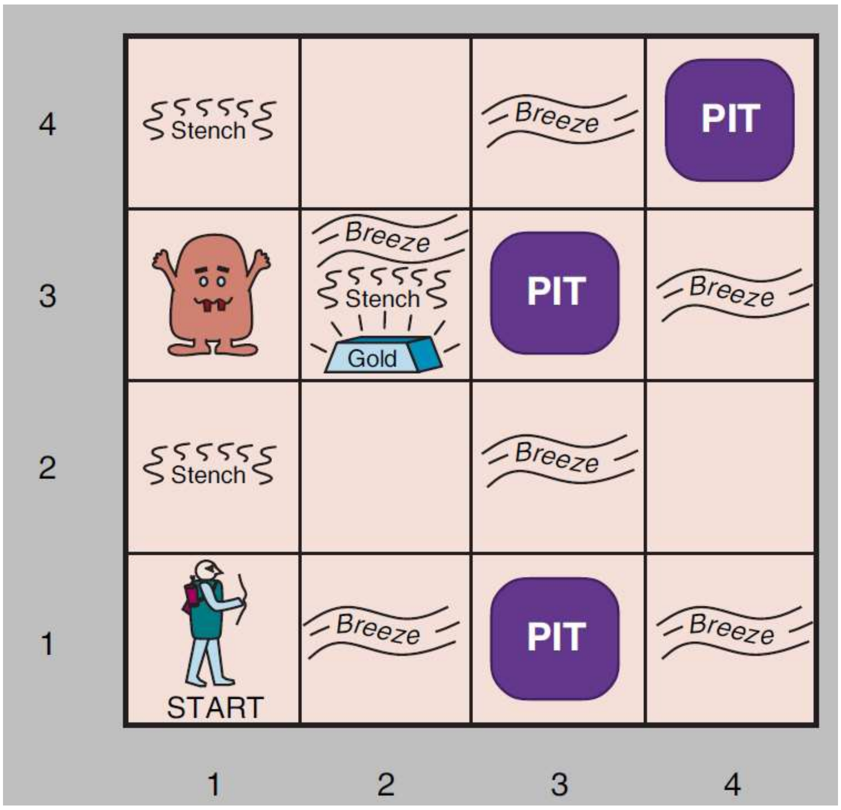
In mathematical notation, we write $\alpha \vDash \beta$ to mean that the sentence α entails the sentence β.

The formal definition of entailment is this: $\alpha \models \beta$ if and only if, in every model in which α is true, β is also true.

Using the notation just introduced, we can write $\alpha \models \beta$ if and only if $M(\alpha) \subseteq M(\beta)$.

3. Wumpus World

A sample wumpus world is shown in below.



The precise definition of the task environment is given by the PEAS description:

Performance measure: +1000 for climbing out of the cave with the gold, –1000 for falling into a pit or being eaten by the wumpus, –1 for each action taken, and –10 for using up the arrow. The game ends either when the agent dies or when the agent climbs out of the cave.

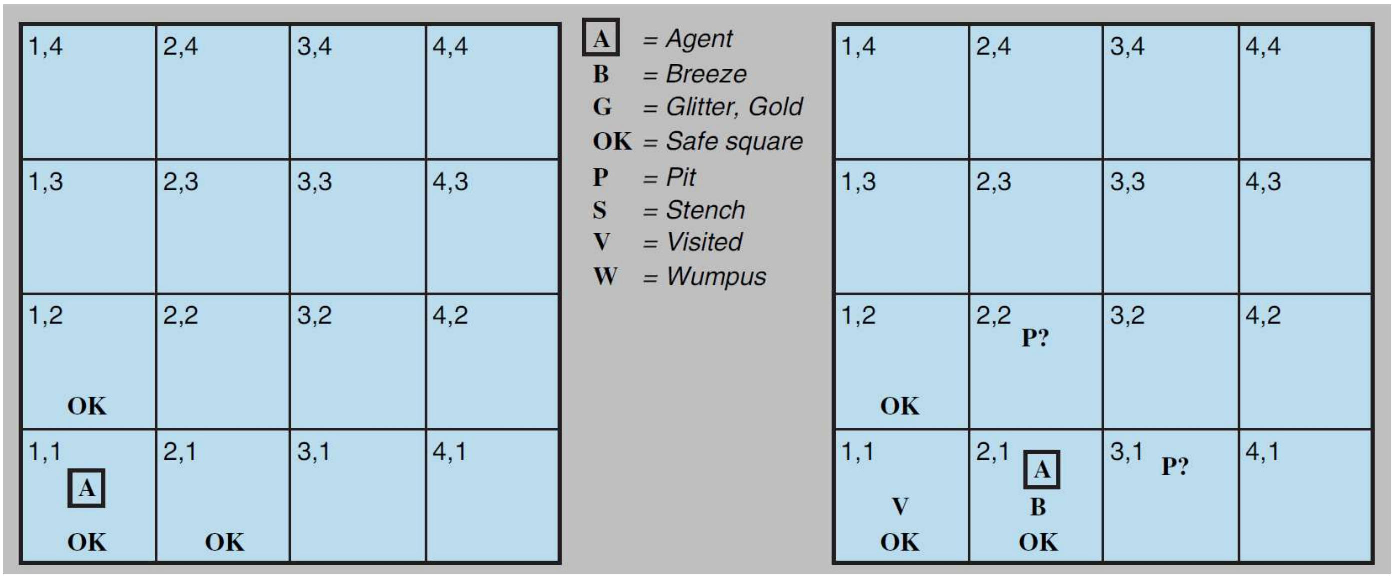
Environment: A 4X4 grid of rooms, with walls surrounding the grid. The agent always starts in the square labeled [1, 1], facing to the east. The locations of the gold and the wumpus are chosen randomly, with a uniform distribution, from the squares other than the start square. In addition, each square other than the start can be a pit, with probability 0.2.

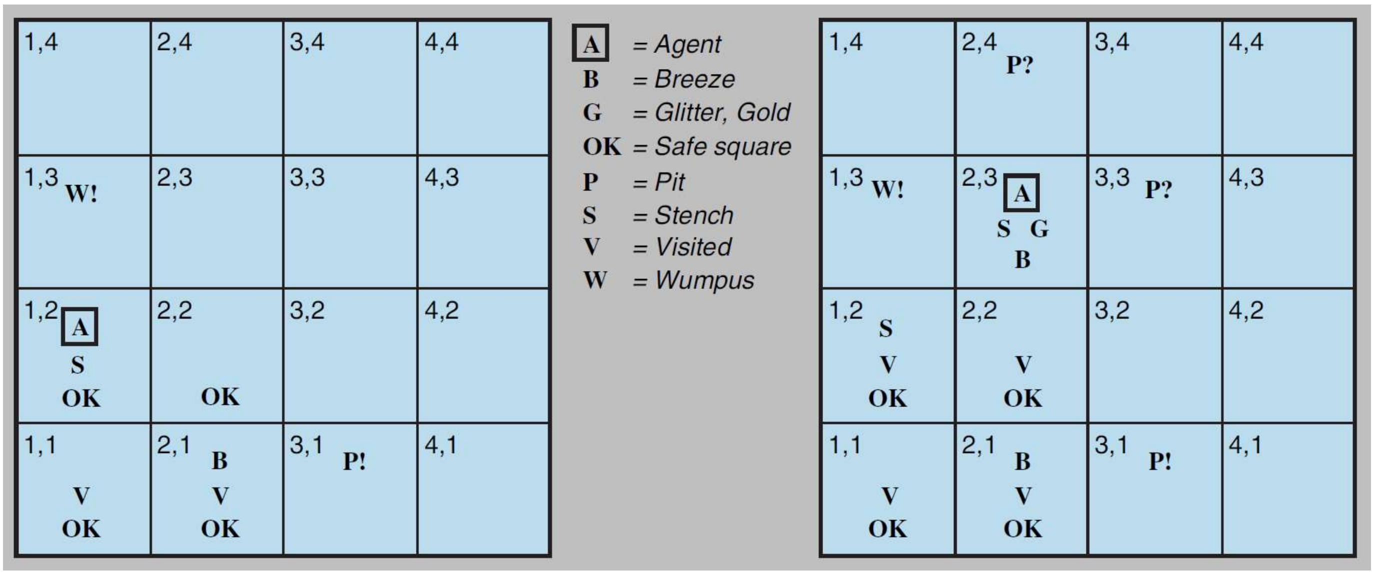
Actuators: The agent can move Forward, TurnLeft by 90°, or TurnRight by 90°. The agent dies a miserable death if it enters a square containing a pit or a live wumpus. (It is safe, albeit smelly, to enter a square with a dead wumpus.) If an agent tries to move forward and bumps into a wall, then the agent does not move. The action Grab can be used to pick up the gold if it is in the same square as the agent. The action Shoot can be used to fire an arrow in a straight line in the direction the agent is facing. The arrow continues until it either hits (and hence kills) the wumpus or hits a wall. The agent has only one arrow, so only the first Shoot action has any effect. Finally, the action Climb can be used to climb out of the cave, but only from square [1,1].

Sensors: The agent has five sensors, each of which gives a single bit of information:

- In the squares directly (not diagonally) adjacent to the wumpus, the agent will perceive a Stench.
- In the squares directly adjacent to a pit, the agent will perceive a Breeze.
- In the square where the gold is, the agent will perceive a Glitter.
- When an agent walks into a wall, it will perceive a Bump.
- When the wumpus is killed, it emits a woeful Scream that can be perceived anywhere in the cave.

The **percepts** will be given to the agent program in the form of a list of five symbols; for example, if there is a stench and a breeze, but no glitter, bump, or scream, the agent program will get [Stench, Breeze, None, None, None].





4. Propositional Logic

The Propositional Logic describes the syntax and symantics of the sentences used for knowledge representation.

The syntax of propositional logic defines the allowable sentences. The atomic sentences consist of a single proposition symbol. Each such symbol stands for a proposition that can be true or false.

There are two proposition symbols with fixed meanings: True is the always-true proposition and False is the always-false proposition. Complex sentences are constructed from simpler sentences, using parentheses and operators called logical connectives. There are five connectives in common use:

4.1 Logical connetives in Propositional Logic

- $\neg$ (not). A sentence such as $\neg W_{1,3}$ is called the **negation** of $W_{1,3}$. A **literal** is either an atomic sentence (a **positive literal**) or a negated atomic sentence (a **negative literal**).
- $\wedge$ (and). A sentence whose main connective is $\wedge$, such as $W_{1,3} \wedge P_{3,1}$, is called a **conjunction**; its parts are the **conjuncts**. (The $\wedge$ looks like an "A" for "And.")
- $\vee$ (or). A sentence whose main connective is $\vee$, such as $(W_{1,3} \wedge P_{3,1}) \vee W_{2,2}$, is a **disjunction**; its parts are **disjuncts**—in this example, $(W_{1,3} \wedge P_{3,1})$ and $W_{2,2}$.
- $\Rightarrow$ (implies). A sentence such as $(W_{1,3} \wedge P_{3,1}) \Rightarrow \neg W_{2,2}$ is called an **implication** (or conditional). Its **premise** or **antecedent** is $(W_{1,3} \wedge P_{3,1})$, and its **conclusion** or **consequent** is $\neg W_{2,2}$. Implications are also known as **rules** or **if–then** statements. The implication symbol is sometimes written in other books as $\supset$ or $\rightarrow$.
- $\Leftrightarrow$ (if and only if). The sentence $W_{1,3} \Leftrightarrow \neg W_{2,2}$ is a **biconditional**.

4.2 BNF (Backus Naur Form) grammar of sentences in propositional logic

$$
\begin{aligned}
Sentence &\rightarrow AtomicSentence \mid ComplexSentence \\
AtomicSentence &\rightarrow True \mid False \mid P \mid Q \mid R \mid \ldots \\
ComplexSentence &\rightarrow (\, Sentence \,) \\
&\mid \neg\, Sentence \\
&\mid Sentence \wedge Sentence \\
&\mid Sentence \vee Sentence \\
&\mid Sentence \Rightarrow Sentence \\
&\mid Sentence \Leftrightarrow Sentence \\
\\
\text{OPERATOR PRECEDENCE} &: \neg, \wedge, \vee, \Rightarrow, \Leftrightarrow
\end{aligned}
$$

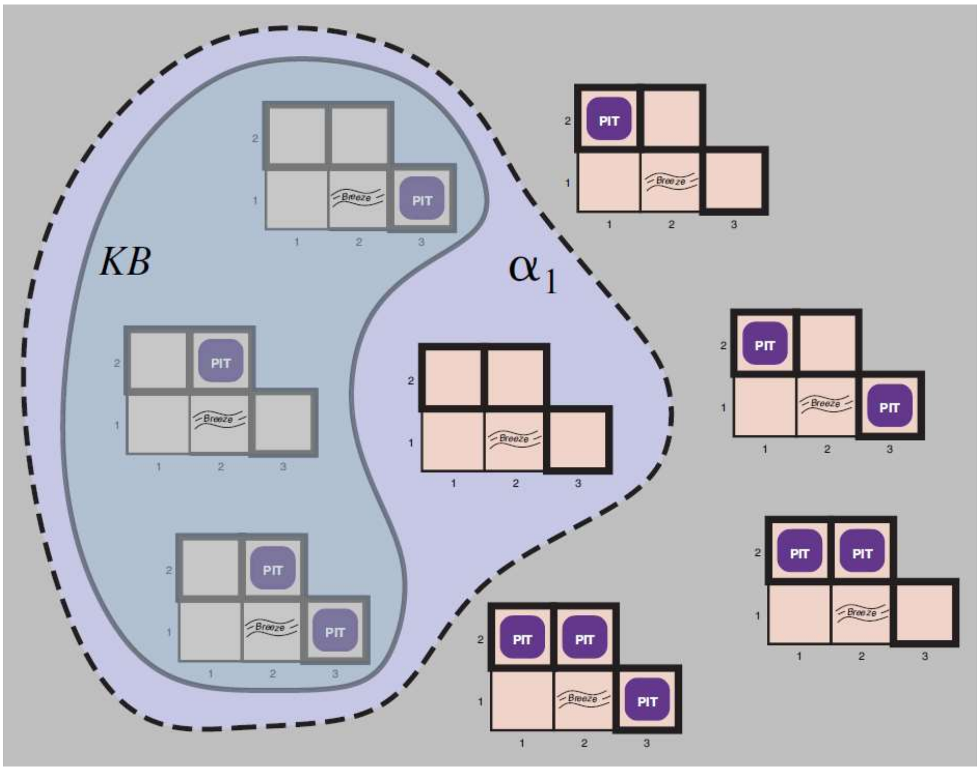
4.3 The semantics

The semantics defines the rules for determining the truth of a sentence with respect to a particular model.

In propositional logic, a model simply sets the truth value—true or false—for every proposition symbol. For example, if the sentences in the knowledge base make use of the proposition symbols $P_{1,2}$, $P_{2,2}$, and $P_{3,1}$, then one possible model is

$$m_1 = \left\{P_{1,2} = false, P_{2,2} = false, P_{3,1} = true\right\}$$

With three proposition symbols, there are $2^3 = 8$ possible models—exactly those depicted below



The semantics for propositional logic must specify how to compute the truth value of any sentence, given a model.

This is done recursively.

All sentences are constructed from atomic sentences and the five connectives; therefore, we need to specify how to compute the truth of atomic sentences and how to compute the truth of sentences formed with each of the five connectives.

Finding truth value of atomic sentences is easy:

- True is true in every model and False is false in every model.

- The truth value of every other proposition symbol must be specified directly in the model. For example, in the model m_1 given earlier, $P_{1,2}$ is false.

For complex sentences, we have five rules, which hold for any subsentences P and Q (atomic or complex) in any model m (here "iff" means "if and only if"):

- $\neg P$ is true iff P is false in m.
- $P \wedge Q$ is true iff both P and Q are true in m.
- $P \vee Q$ is true iff either P or Q is true in m.
- $P \Rightarrow Q$ is true unless P is true and Q is false in m.
- $P \Leftrightarrow Q$ is true iff P and Q are both true or both false in m.

The rules can also be expressed with **truth tables** that specify the truth value of a complex sentence for each possible assignment of truth values to its components. Truth tables for the five connectives are given below

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*

Knowledge Base of Wumpus World using Propositional Logic

We need the following symbols for each [x, y] location:

$P_{x,y}$ 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].

We label each sentence R_i so that we can refer to them:

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

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

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

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

$R_5 : B_{2,1}$

A simple inference procedure

Our goal now is to decide whether KB $\models \alpha$ for some sentence α.

- For example, is $\neg P_{1,2}$ entailed by our KB?
  - Let us use model-checking algorithm for inference.
  - Enumerate the models, and check that α is true in every model in which KB is true.
  - Models are assignments of true or false to every proposition symbol. Returning to our wumpus-world example, the relevant proposition symbols are $B_{1,1}$, $B_{2,1}$, $P_{1,1}$, $P_{1,2}$, $P_{2,1}$, $P_{2,2}$, $P_{3,1}$. With seven symbols, there are 2^7=128 possible models; in three of these, KB is true.
  - In those three models, $\neg P_{1,2}$ is true, hence there is no pit in [1, 2].

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

- For example, is $P_{2,2}$ entailed by our KB?

- $P_{2,2}$ is true in two of the three models and false in one, so we cannot yet tell whether there is a pit in [2, 2].