Artificial Intelligence

Augustin Cosse.

Fall 2020

October 13, 2020

So far

- Simple Reflex, Random agents, Utility based, Goal based
- Improvement through Search Methods (uninformed (DFS, BFS), informed (BS, A*)).

Knowledge Based Agent

- An agent that has goals and search for solutions towards the goals can do better than one that just reacts to the environment
- So far we focused on the search, leaving aside the methods for generating states and actions
- We will now improve the agents by endowing them with a general ability for logical reasoning
- A (Logical) Knowledge Based (KB) agent begins with some knowledge of the world and of its own actions.
- It then uses logical reasoning to maintain a description of the world as it acquires new percepts, and from its representation of the world, it deduces a course of actions that will help it achieve its goals

Knowledge Based Agent

- Today we will start by discussing a simple language for expressing knowledge and show how it can be used to draw conclusions about the environment and decide which action to take.
- We will then augment the language to make it capable of expressing knowledge about more complex worlds

Knowledge Based Agent

- A (logical) knowledge-based agent needs to keep track of several things:
  - The current state of the world
  - How to infer unseen properties of the world from percepts
  - How the world evolves over time
  - What it wants to achieve
  - What its own actions do in various circumstances

Knowledge Based Agent

- The fundamental element of a (logical) knowledge based agent is its Knowledge Base (KB)
- The Knowledge Base is a set of representations of facts about the world. Each individual representation is called a sentence.
- The sentences are expressed in a language called a knowledge representation language.
- We will consider two basic functions when implementing a Knowledge based agent:
  - A function TELL (used to add new sentences to the knowledge base)
  - A function ASK (used to query what is known)

Knowledge Based Agent

- When one asks a question to the knowledge base, the answer should follow from what has been stored in this knowledge base previously. Determining what follows from what has been stored in the Knowledge base previously is done through inference mechanisms.

- Each time an agent is called, it tells the Knowledge base what it perceives and it asks the knowledge base what action it should select

```
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
```

Figure 6.1 A generic knowledge-based agent.

Knowledge Based Agent

- The details of the representation language are hidden in the functions "Make-Percept-Sentence", "Make-Action-Query" and "Make-Action-Sentence"
  - "Make-Percept-Sentence" takes a percept and a time, and returns a sentence encoding the fact that the agent perceived the percept at the given time
  - "Make-Action-Query" takes a time and input and returns a sentence corresponding to asking what action should be performed.
  - "Make-Action-Sentence" updates the knowledge base following the choice of the action.

Knowledge Based Agent

- The Knowledge based agent is different from the agent that maintains an internal state (which also has ASK and TELL functions) from that is relies on a three level implementation (first level corresponding to an internal state representation)

Knowledge Based Agent

- The knowledge level or epistemological level. This is the most abstract level. This level only describes the agent in terms of what it knows (no logical details). E.g. an autonomous agent could be said to know that there is a path between Paris and Berlin.
- The logical level (how information is stored by the sentences and the language). This is the level at which the knowledge is encoded into sentences. An autonomous cab could for example have the logical sentence "Links(path, Paris, Marseille)"
- The implementation level. This is the level at which the logical sentences are encoded numerically. E.g. the sentence "Links(path, Paris, Marseille)" could be represented by value 1 in the matrix of all possible paths. It could also be encoded simply by the string "Links(path, Paris, Marseille)".

Knowledge Based Agent

- Implementation is important for efficiency but is irrelevant to the logical and knowledge levels.
- The agent initial program is designed by adding one by one the sentences that represent the designer's knowledge of the environment.
- This idea is known as Declarative system building.
- Declarative knowledge is one type of knowledge that can be possessed by an agent. We specify the logical mechanisms but not the control flow.
- Another type of knowledge that can be encoded in the agent is procedural knowledge which is knowing how to perform a particular task.

The Wumpus World

- In order to clarify what we mean by logical reasoning, let us consider a simple environment known as the Wumpus world. In this environment, the agent starts from the lower left corner and its task it to discover the gold, then return to the (1,1) cell

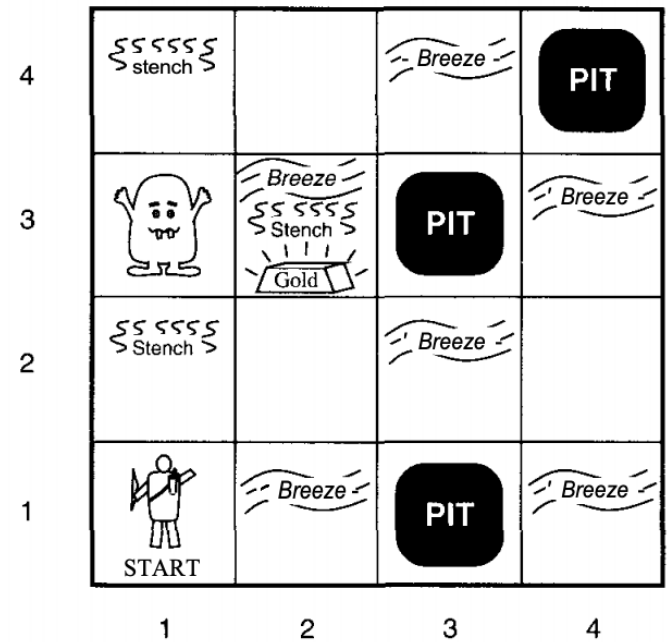


Figure 6.2 A typical wumpus world.

The Wumpus World

- The Wumpus world provides plenty of motivation for logical reasoning.
- Somewhere in the environment is the Wumpus, a ghost that eats anyone who enters its room
- Some cells contain bottomless pits that will trap anyone stepping on those cells

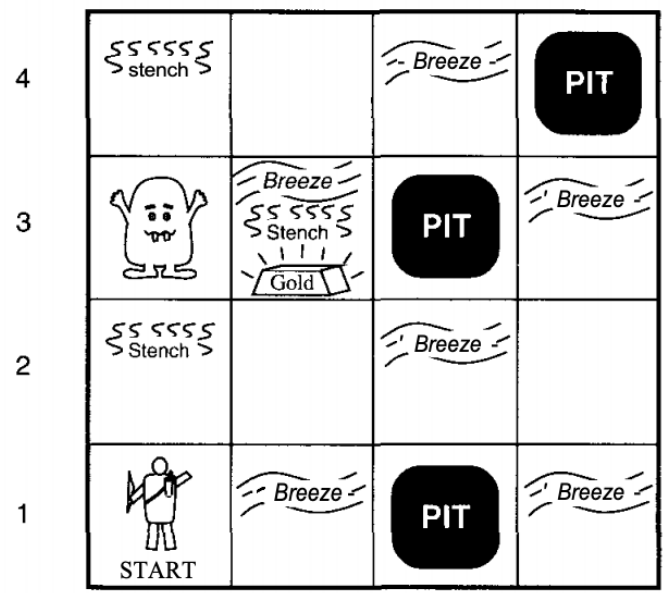


Figure 6.2 A typical wumpus world.

The Wumpus World

- The reward is represented by the gold
- The Wumpus world is represented by a grid of squares surrounded by walls

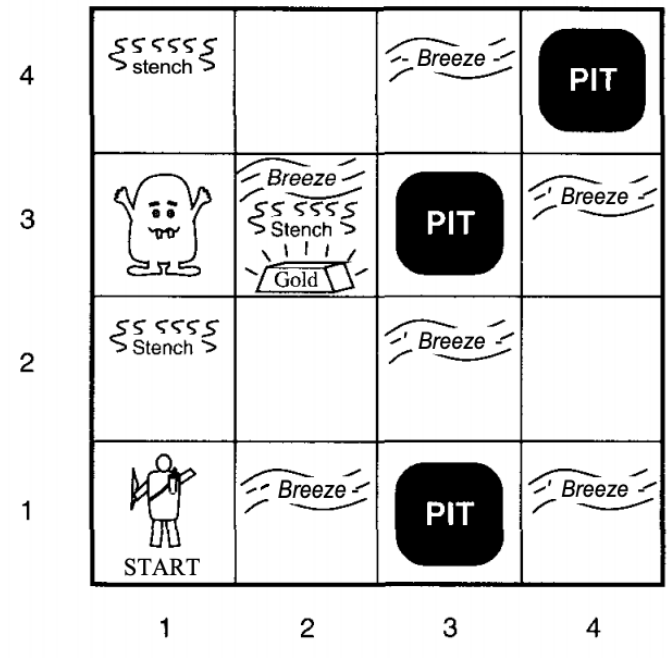


Figure 6.2 A typical wumpus world.

The Wumpus World

- In the square containing the Wumpus and in the directly (not diagonally) adjacent squares, 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

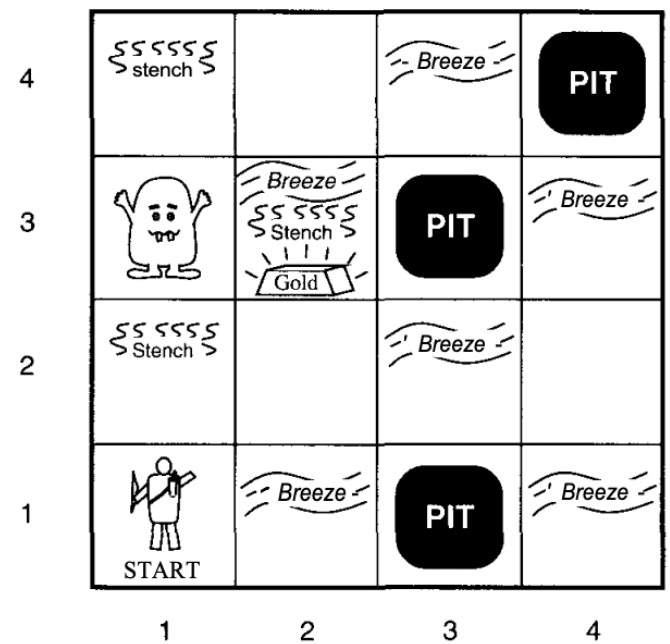


Figure 6.2 A typical wumpus world.

The Wumpus World

- When the agent walks into a wall, it will perceive a bump.
- When the Wumpus is killed, it gives out a scream that can be perceived anywhere in the cave

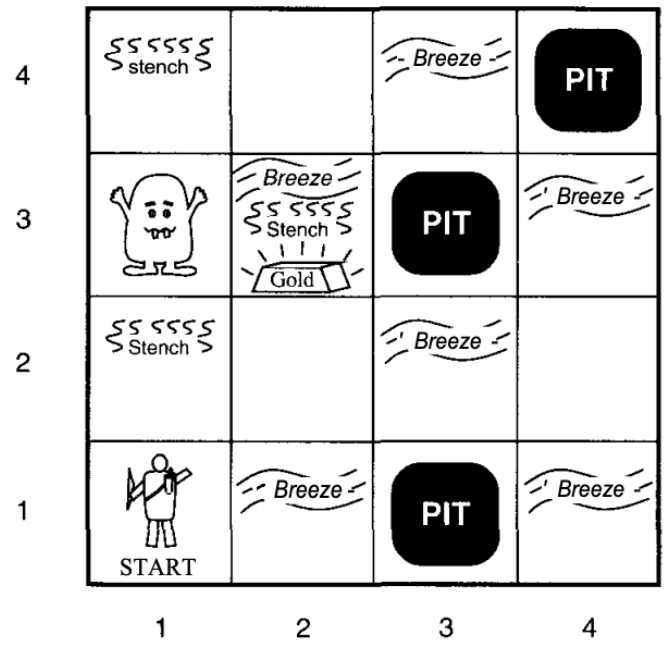


Figure 6.2 A typical wumpus world.

The Wumpus World

- The percepts will be given to the agent in the form of a list of 5 symbols: If there is a stench, a breeze, a glitter but no bump and no scream, the agent will get the percept [*Stench*, *Breeze*, *Glitter*, *None*, *None*]. The agent does not perceive its own location

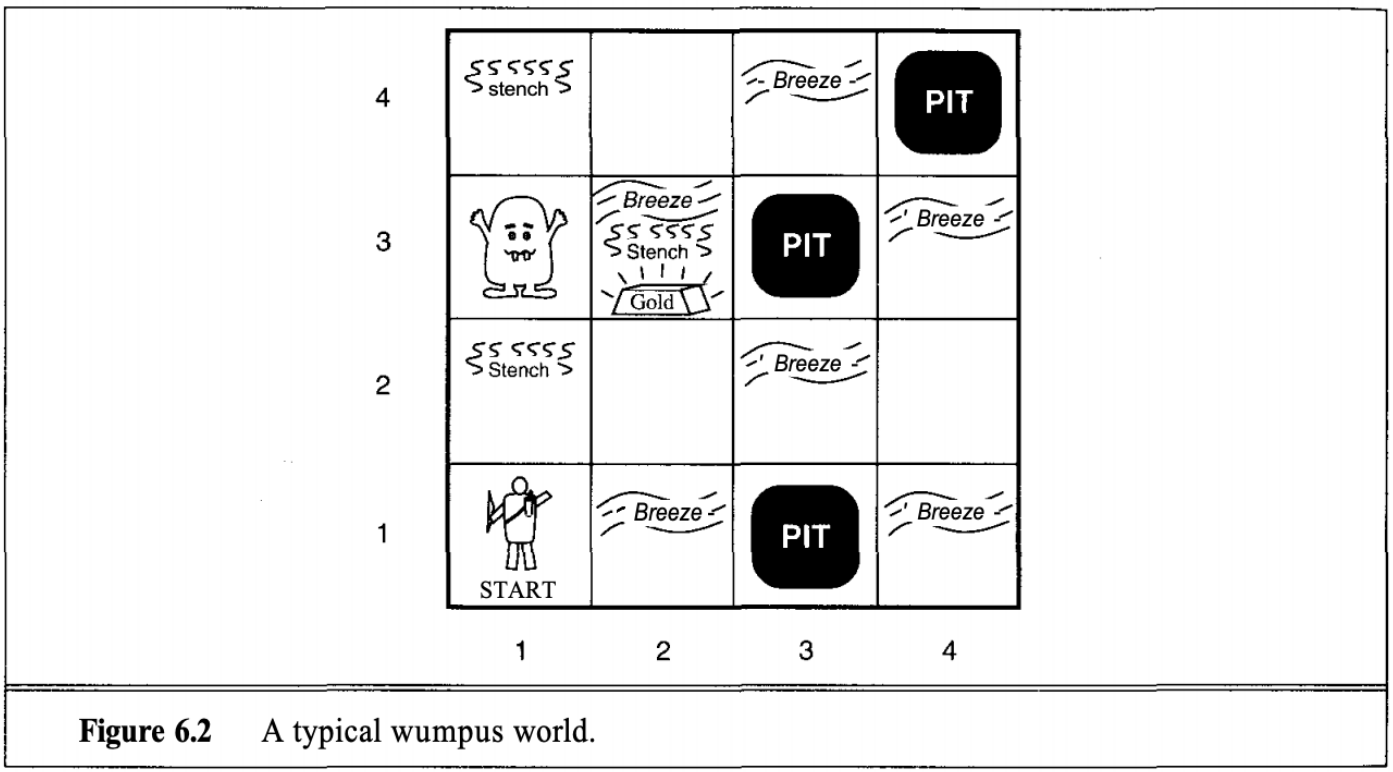


Figure 6.2 A typical wumpus world.

The Wumpus World

- As in the vacuum environment, there are various actions, including: go forward, turn right by 90 degrees, turn left by 90 degrees, grab an object, fire an arrow in a straight line, in the direction the agent is facing (the arrow then continues until it either hits a wall or kill the Wumpus). The action 'climb' can be used to leave the cave

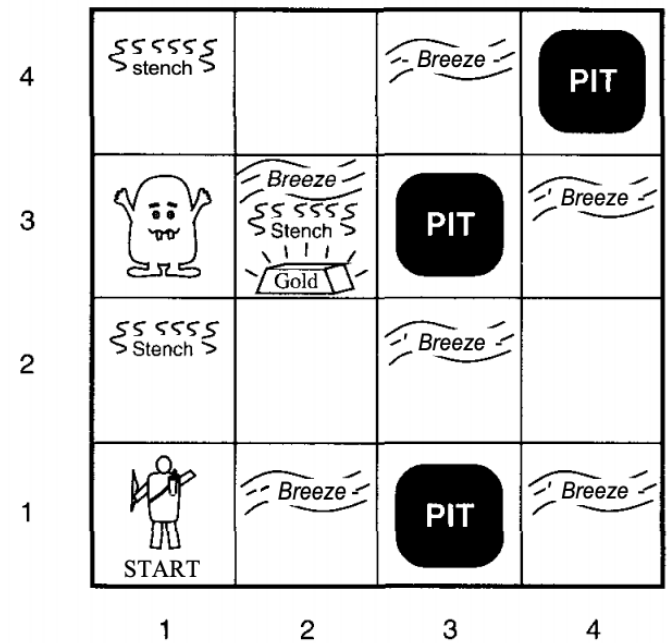


Figure 6.2 A typical wumpus world.

The Wumpus World

- As indicated before, the agent dies if it enters the Wumpus cell or a pit cell.
- 1000 points are awarded for climbing out of the cave with the gold. And there is a 1 point penalty for each action taken and a 10,000 points penalty for getting killed

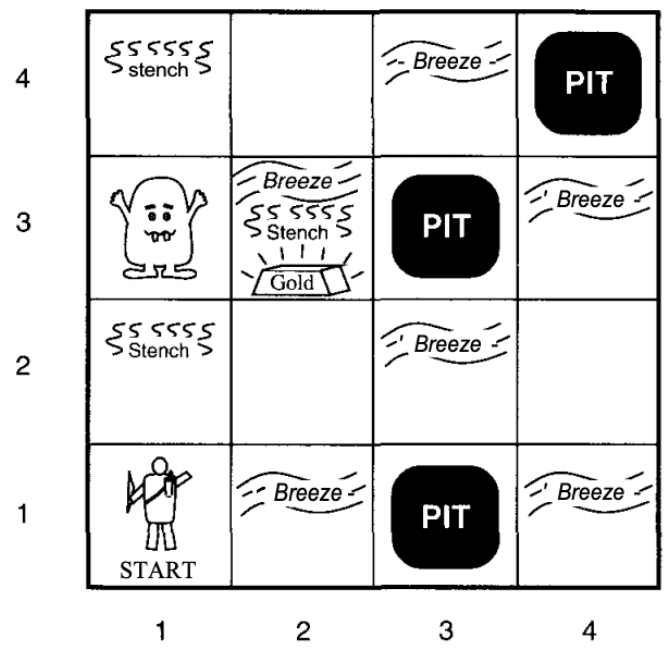


Figure 6.2 A typical wumpus world.

The Wumpus World

- We will now discuss why successful agents should have some form of logical reasoning ability
- The first information that the agent gets is that there is no stench or breeze in cell $(1, 1)$. From this, it can infer that $(1, 2)$ and $(2, 1)$ are free of dangers.

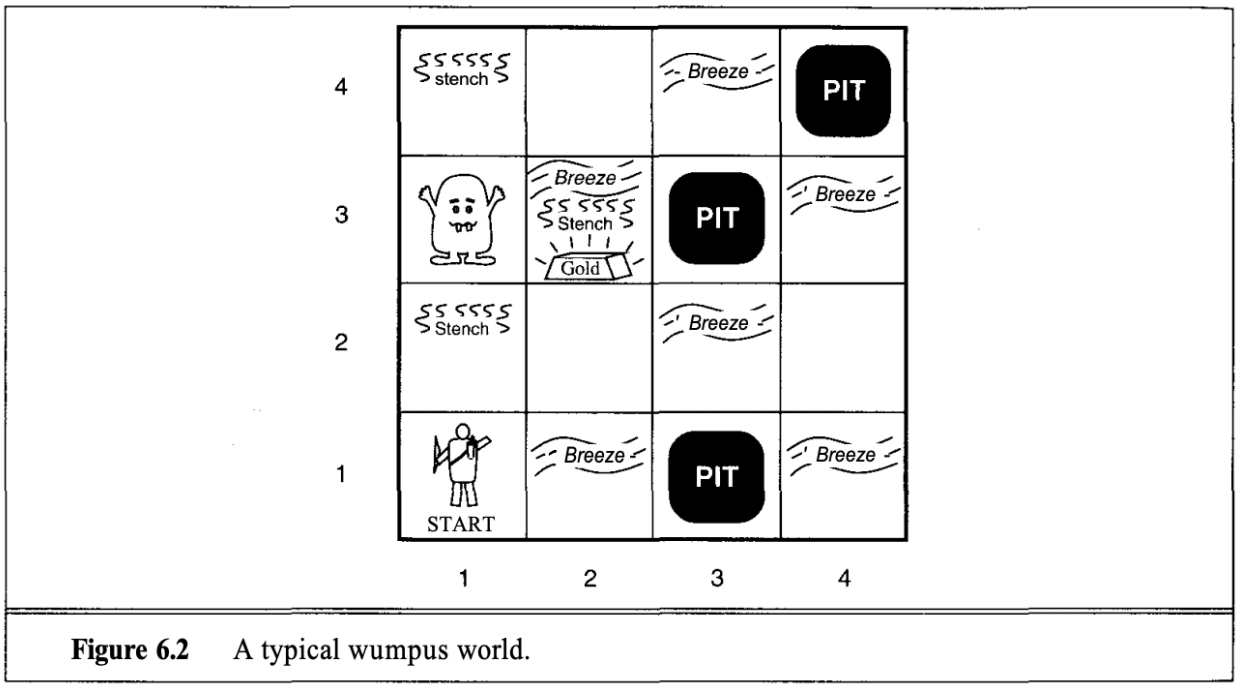


Figure 6.2 A typical wumpus world.

The Wumpus World

- The agent can then mark those cells as 'OK'
- From the fact that the agent is still alive, it can infer that $(1, 1)$ is also OK
- A cautious agent should only move to a cell that it knows is safe

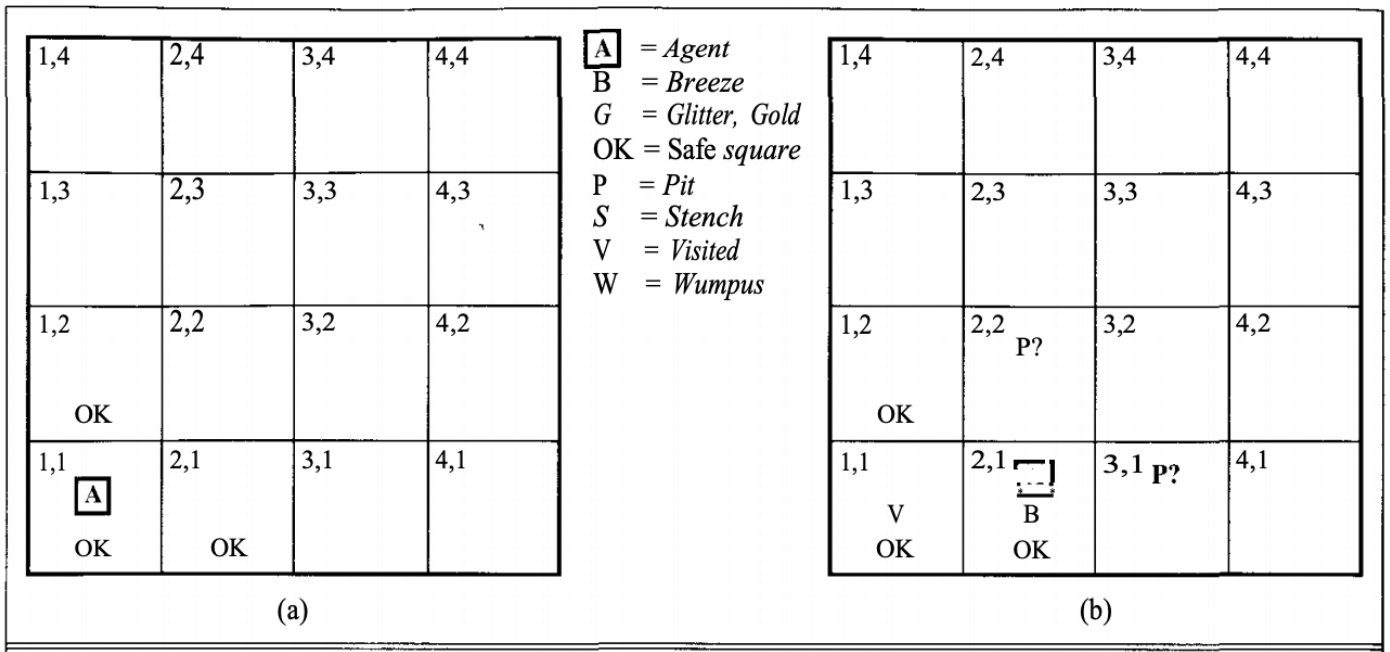


Figure 6.3 The first step taken by the agent in the wumpus world. (a) The initial situation, after percept [*None*,*None*,*None*,*None*,*None*]. (b) After one move, with percept [*None*, *Breeze*,*None*,*None*,*None*].

The Wumpus World

- Let us suppose the agent moves to $(2, 1)$. We then get the map 6.3b below
- The agent detects a breeze in $(2, 1)$ so it knows there must be a pit in a neighboring square, either $(2, 2)$ or $(3, 1)$. The notation $P?$ indicates the possibility of such a pit.
- The pit cannot be in $(1, 1)$ because the agent was already there and did not fall

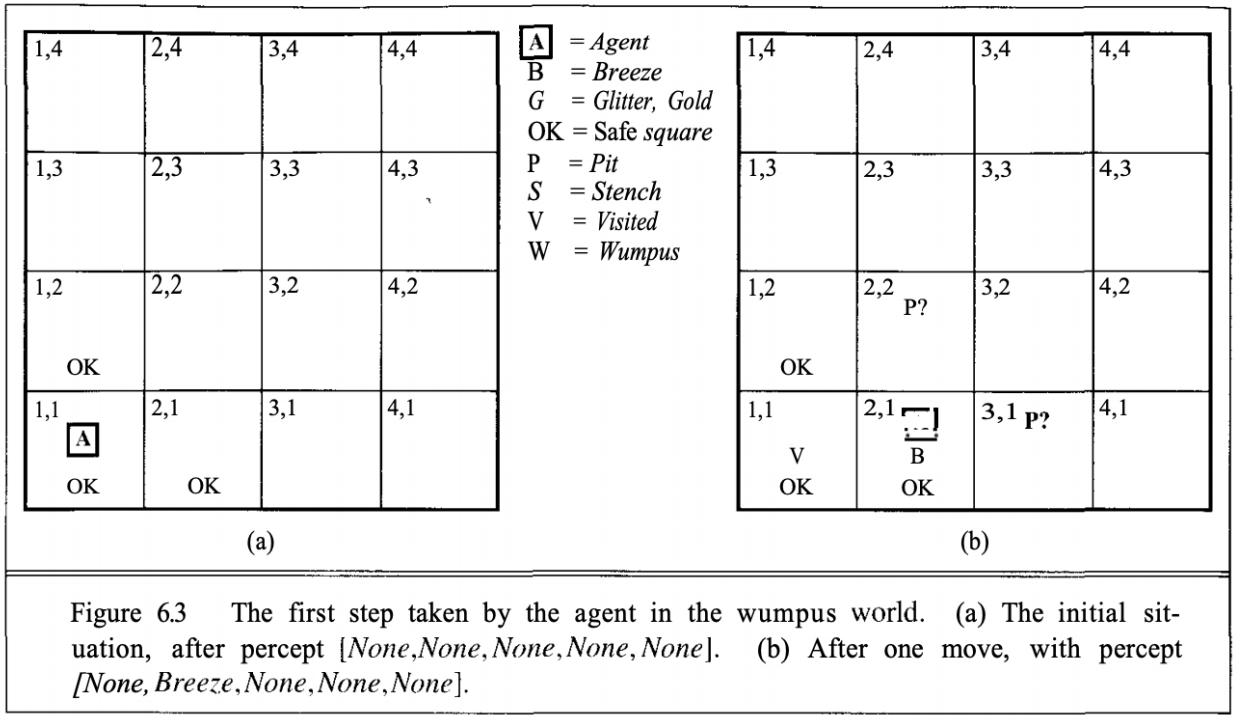


Figure 6.3 The first step taken by the agent in the wumpus world. (a) The initial situation, after percept [*None*,*None*,*None*,*None*,*None*]. (b) After one move, with percept [*None*, *Breeze*,*None*,*None*,*None*].

The Wumpus World

- At this point, there is only one known square that has not been visited and is OK.
- The agent will thus turn around, go back to $(1, 1)$ and then proceed to $(1, 2)$

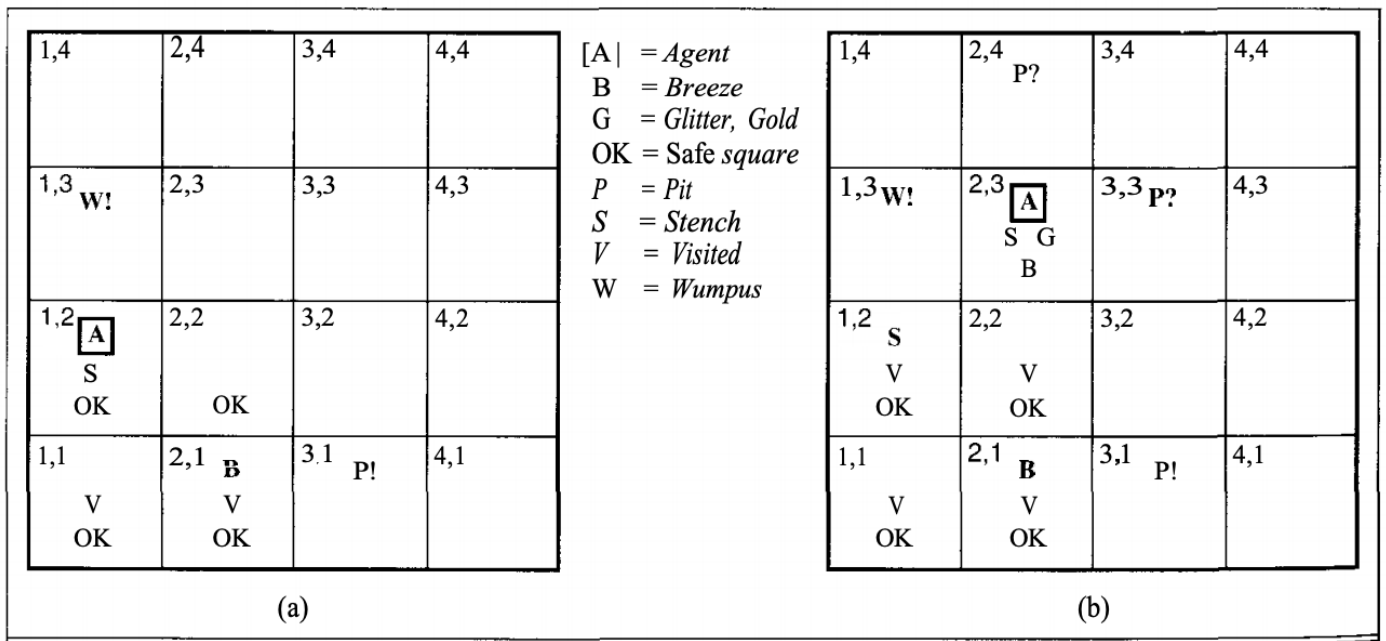


Figure 6.4 Two later stages in the progress of the agent. (a) After the third move, with percept [*Stench*, *None*, *None*, *None*, *None*]. (b) After the fifth move, with percept [*Stench*, *Breeze*, *Glitter*, *None*, *None*].

The Wumpus World

- The agent detects a stench in $(1, 2)$ which means that there must be a Wumpus nearby but the Wumpus cannot be in $(1, 1)$ and it cannot be in $(2, 2)$ (or the agent would have detected a stench when it was in $(2, 1)$)
- The agent can thus infer that the Wumpus is in $(1, 3)$ (notation $W!$ in the figures below)

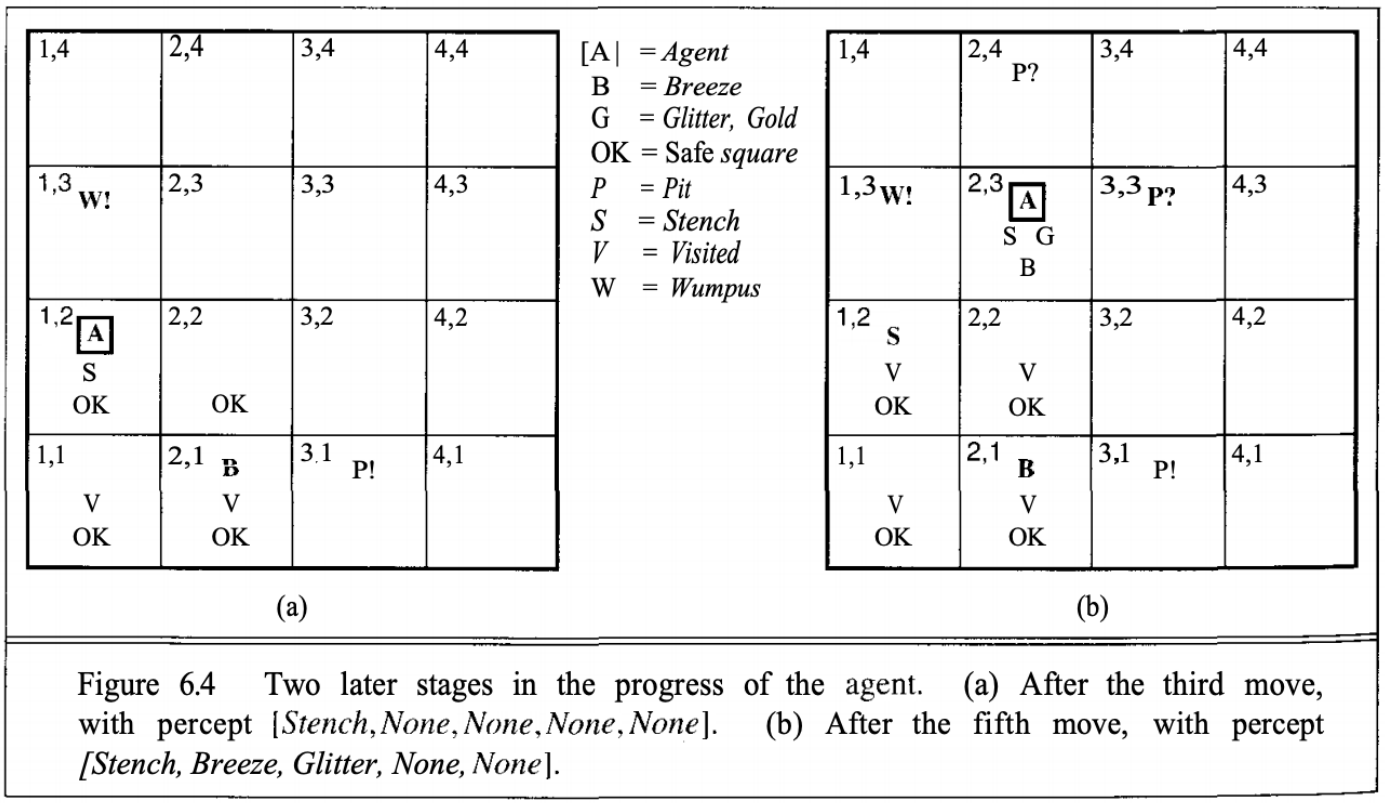


Figure 6.4 Two later stages in the progress of the agent. (a) After the third move, with percept [*Stench*, *None*, *None*, *None*, *None*]. (b) After the fifth move, with percept [*Stench*, *Breeze*, *Glitter*, *None*, *None*].

The Wumpus World

- Moreover, the lack of a Breeze percept in $(1, 2)$ means that there must be a pit in $(3, 1)$
- The reasoning is that no breeze in $(1, 2)$ means that there can be no pit in $(2, 2)$ but we have already inferred that there must be a pit in either $(2, 2)$ or $(3, 1)$ so it must be in $(3, 1)$.

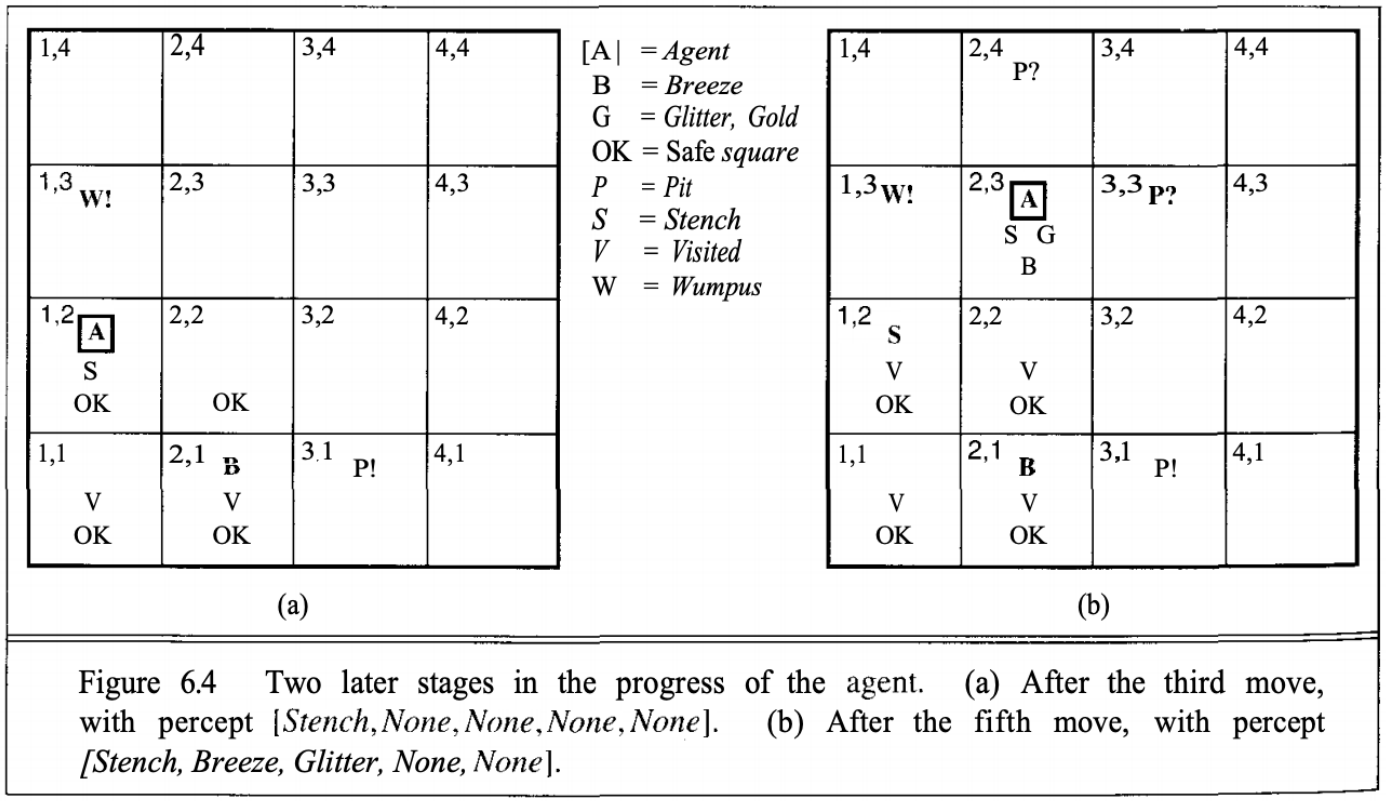


Figure 6.4 Two later stages in the progress of the agent. (a) After the third move, with percept [*Stench*, *None*, *None*, *None*, *None*]. (b) After the fifth move, with percept [*Stench*, *Breeze*, *Glitter*, *None*, *None*].

The Wumpus World

- What we are doing here is a fairly difficult example of inference because it relies on combining knowledge gained at different times in different places and relies on the lack of a percept to make a crucial step
- Inference is beyond the ability of most animals but is typically the kind of reasoning that an agent does
- We will now formally describe how such inference can be encoded in the computer

Representation, reasoning and logic

- The idea of Knowledge representation is to express knowledge in a computer tractable form. A knowledge representation language is defined by two aspects
  - The syntax which describes the possible configurations that groups of words can take.
  - The semantics indicates the facts of the world to which the sentences refer. Without the semantics, a sentence is just an arrangement of symbols. With the semantics, the sentence makes a claim about the world.

Representation, reasoning and logic

- As an example, consider arithmetic. The syntax of the language of arithmetic expressions says that if x, and y are representing some numbers, then we can build the sentence $x > y$.
- The semantics of the language says that $x > y$ is true when x is strictly greater than y
- Provided that syntax and semantics are defined precisely, we can call the language a logic

Representation, reasoning and logic

- Semantics determines the facts to which a given sentence refers. Facts are part of the world whereas their representation must be encoded in a way that can be stored within the agent.
- The world cannot be stored on a computer (nor can we put it inside a human) so we must operate on representations of facts rather than on the facts themselves.

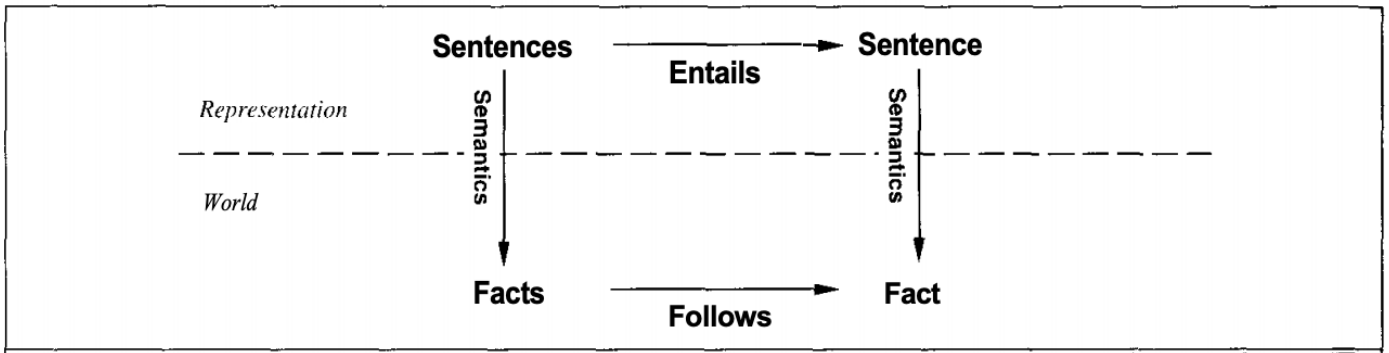


Figure 6.5 The connection between sentences and facts is provided by the semantics of the language. The property of one fact following from some other facts is mirrored by the property of one sentence being entailed by some other sentences. Logical inference generates new sentences that are entailed by existing sentences.

Representation, reasoning and logic

- We want to generate sentences that are necessarily true given that the old sentences are true.
- The relation between sentences is called entailment (i.e. "if A is true, then B is necessarily true")

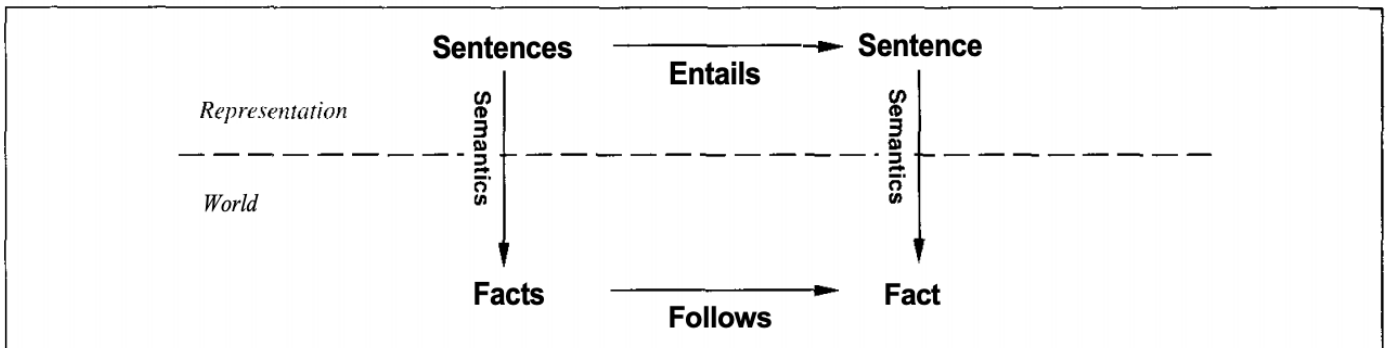


Figure 6.5 The connection between sentences and facts is provided by the semantics of the language. The property of one fact following from some other facts is mirrored by the property of one sentence being entailed by some other sentences. Logical inference generates new sentences that are entailed by existing sentences.

Representation, reasoning and logic

- The relation of entailment between a knowledge base KB and a particular sentence *a* is pronounced "KB entails *a*" and is written as $KB \models a$
- An inference procedure can do one of two things:
  - Given a knowledge base KB, it can generate new sentences that purport to be entailed by KB
  - Or, given a knowledge base KB, and another sentence *a*, it can report whether or not *a* is entailed by KB
- An inference procedure that generates only entailed sentences is called sound or truth-preserving.
- An inference procedure i can be described by the sentences it can derive. if i can derive *a* from KB, we write $KB \vdash_i a$ ("*a* is derived from KB by i").

Representation, reasoning and logic

- The record of operations of a sound inference procedure is called a proof
- Entailment is like the needle in a haystack, proof is like finding it. For a real haystack, systematic examination can always decide whether the needle is in the haystack
- This is the question of completeness. An inference procedure is complete if it can find a proof for any sentence that is entailed
- For many knowledge bases, the haystack of consequences is infinite and completeness becomes an important issue.

Representation, reasoning and logic

- The key to sound inference is to have the inference steps respect the semantics of the sentences they operate upon
- Given a Knowledge Base KB, the inference steps should only derive new sentences that represent facts that follow from the facts represented by KB
- This idea is encoded by what is known as a proof theory of the language. The proof theory specifies the steps that are sound.

Representation, reasoning and logic

- Consider the simple sentence $E = mc^2$.
  - The syntax here allows two expressions to be connected by the '=' sign, or multiple expressions to be concatenated with each other
  - The semantics says that the two expressions on each side of the equal sign refer to the same quantity and that the concatenation of two expressions refers to the quantity that is the product of the quantities referred to by each expression,..
  - From the semantics, we can show that a new sentence can be generated by, for example, concatenating the same expression to both side of the equal sign as in $ET = mc^2 T$.

Logic

- Recall that a logic consists of the following elements:
  - A formal system, consisting of (a) the syntax, which describes how to make sentences and (b) the semantics of the language which indicates how sentences are related to state affairs.
  - The Proof Theory. A set of rule for deducing the entailment of a set of sentences.
- In this course, we willl study two logical languages: Propositional Logic (or Boolean Logic) and First order logic (first order predicate calculus with equality)

Logic

- In Propositional Logic (PL), symbols represent whole propositions (facts). E.g. 'Paris and Marseille are connected by a road' is represented by one symbol.
- Proposition symbols can be combined using Boolean/Logical connectives to generate sentences with more complex meanings.
- First order logic (FOL) on the other hand commits to the representation of the world by means of objects and predicates on those objects (properties of objects or relations between those objects) as well as connectives and quantifiers

Logic

- First order Logic seems to be able to capture a good deal of what we know about the world
- Logics are often described and distinguished based on their Ontological and Epistemological commitments.
  - Ontological commitments = what the language assumes regarding the nature of reality. E.g. PL assume that the world is made up of fact that are truth or not. FOL assumes more: that the world consists of objects with certain relations among them that hold or do not hold.
  - Epistemological commitments = possible states of knownledge that a language allows with respect to facts. In both PL and FOL, a sentences represents a fact and the agent either believes the sentence to be true, false, or is unable to conclude. These logics thus have 3 possible states of belief. Systems using probability theory on the other hand can have any degree of belief.

Examples of Languages

Language	Ontological Commitment (What exists in the world)	Epistemological Commitment (What an agent believes about facts)
Propositional logic	facts	true/false/unknown
First-order logic	facts, objects, relations	true/false/unknown
Temporal logic	facts, objects, relations, times	true/false/unknown
Probability theory	facts	degree of belief 0...1
Fuzzy logic	degree of truth	degree of belief 0...1

Figure 6.7 Formal languages and their ontological and epistemological commitments.

Propositional logic

- The Syntax of Propositional logic is simple. The language consists of the following elements:
  - Logical constants: True and False
  - Proposition symbols such as Q and P for example
  - Logical connectives: $\wedge, \vee, \Leftrightarrow, \Rightarrow, \neg$
- A sentence can be formed by combining simpler sentences with one of the five logical connectives

Propositional logic, logical connectives

- $\wedge$ (and). A sentence whose main connective is $\wedge$ such as $P \wedge (Q \vee R)$ is called a conjunction. Its parts are the conjunts (you can remember that the $\wedge$ looks like a A for And)
- $\vee$ (or) A sentence using $\vee$ such as $A \vee (P \wedge Q)$ is called a disjunction of the disjuncts A and $P \wedge Q$. (Historically, the $\vee$ comes from the latin 'vel' which means 'or')
- $\Rightarrow$ (imply). A sentence such as $(P \wedge Q) \Rightarrow R$ is called an Implication (or conditional). Its Premise or antecedent is $P \wedge Q$ and its conclusion or consequent is R. Implications are also known as Rules or if-then statements. The implication symbol is sometimes also written $\supset$
- $\Leftrightarrow$ (equivalent). The sentence $(P \wedge Q) \Leftrightarrow (Q \wedge P)$ is an equivalence (also known as biconditional)
- $\neg$ (not). A sentence such as $\neg P$ is called the negation of P. All the other connectives combine two sentences in one. $\neg$ is the only connective that can operate on a single sentence.

Propositional logic

- The grammar of propositional logic introduces two types of sentences: atomic sentences which consist of a single symbol (e.g. P) and complex sentences which contain connectives or parentheses, e.g. $(P \wedge Q)$.
- The Grammar is ambiguous. As an example, a sentence such as $P \wedge Q \wedge R$ could be interpreted as either $(P \wedge Q) \wedge R$ or $P \wedge (Q \wedge R)$. This is similar to the ambiguity that can be found in arithmetic expressions such as $P + Q \times R$
- The way to resolve this ambiguity is similar to arithmetic, we pick an order of precedence for the operators
- In Propositional Logic, the order of precedence is (from highest to lowest, L to R): $\neg$, $\wedge$, $\vee$, $\Rightarrow$, $\Leftrightarrow$.
- The sentence $\neg P \vee Q \wedge R \Rightarrow S$ is thus equivalent to $((\neg P) \vee (Q \wedge R)) \Rightarrow S$.

Propositional logic: Semantics

- The semantics of Propositional Logic is quite straightforward. It is defined by specifying the interpretation of the symbols and constants as well as the meaning of logical connectives
- A symbol can represent whatever you want. For example, the interpretation of P might be that Paris is the capital city of France, or that there is bridge between San Francisco and Oakland.
- With logical constants there is no choice. The sentence 'True' has as its interpretation the way the world is (the facts that hold true) and the sentence 'False' has as its interpretation the way the world is not.
- The meaning of a complex sentence is defined from the meaning of its parts (we say that the language is compositional).

Propositional logic: Semantics

- Each logical connective can be thought of as a function, just as an addition takes two numbers as inputs and returns a number, 'and' is a function that takes two truth values as inputs and returns a truth value.
- One way to define a function is to provide a table that defines the output value for every possible input value
- For most functions (such as addition) this is impossible as there would be an infinite number of entries in the table but in PL, there are only two possible truth values. A logical function with 2 arguments therefore only needs a table with 4 entries

Propositional logic: Semantics

- Such a table is called a 'truth table'

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

Propositional logic: Semantics

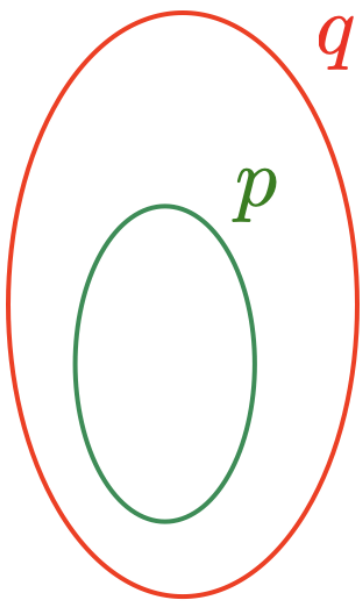
- Truth tables define the semantics for simple sentences such as True $\wedge$ True. More complex sentences, such as $(P \vee Q \wedge \neg S)$ are defined by a process of decomposition. First determine the meaning of $P \wedge Q$ and of $\neg S$, and then combine them using the definition of the $\wedge$ function.

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

- In some ways, the most important connective is probably $\Rightarrow$. Its truth table might seem puzzling at first as it does not fit our understanding of P implies Q or if P then Q. It however makes sense if we think of $P \Rightarrow Q$ as having the meaning 'If P is true, then I can claim that Q is true'
otherwise I make no claim

Propositional logic: Semantics

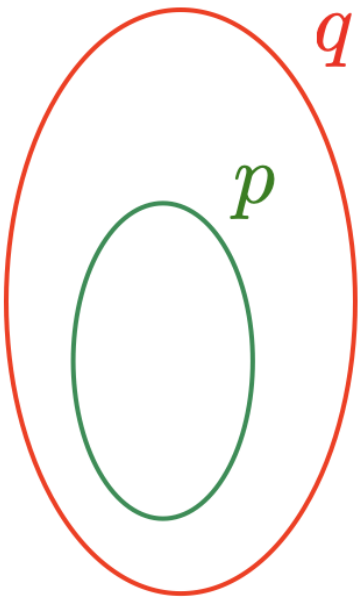
- As an example, think of q as encoding the fact 'A lives in the US', p encodes the fact 'A lives in NY'. There could be people living in the US that do not live in NY.
- Could an element that satisfies p (i.e. lives in NY) not satisfy q (does not live in the US) ? No of course. In this case, the implication takes the value 'False'



p	q	$p \Rightarrow q$
False	False	True
False	True	True
True	False	False
True	True	True

Propositional logic: Semantics

- Propositional logic does not require any relation of causation between p and q. The sentence '5 is odd implies Tokyo is the capital of Japan' would be a true sentence of Propositional logic.
- As we saw below, any implication is true whenever its antecedent is false. The semantic should really be understood as 'If p is true' then I'm claiming that q is true, otherwise I make no claim.



p	q	$p \Rightarrow q$
False	False	True
False	True	True
True	False	False
True	True	True

Propositional logic: Semantics

- Truth tables can also be used to check the validity of a complex inference rule
- Given a sentence, we make a table with one row for each of the possible values of the proposition symbols in the sentence. For each row we can calculate the truth value of the entire sentence.If an inference is true in every row, then the inference is valid.

P	H	$(P \vee H)$	$(P \vee H) \wedge \neg H$	$((P \vee H) \wedge \neg H) \Rightarrow P$
False	False	False	False	True
False	True	True	False	True
True	False	True	True	True
True	True	True	False	True

Propositional logic: Semantics

- Let us go back to the Wumpus and use P to encode the fact that there is a ghost in $(1,3)$ and H to encode the fact that there is a ghost in $(2,2)$. From the table below, if at some point we learn $P \vee H$ and then also learn $\neg H$, then we can use the valid inference rule to conclude that P is true.

P	H	$(P \vee H)$	$(P \vee H) \wedge \neg H$	$((P \vee H) \wedge \neg H) \Rightarrow P$
False	False	False	False	True
False	True	True	False	True
True	False	True	True	True
True	True	True	False	True

Propositional logic: Semantics

- What we just said is particulary important for intelligent agents. If a machine has some premises and a candidate statement (conclusion), it now has a way to determine if the new statement is true.
- It can do this by building a truth table for the sentence *Premises* $\Rightarrow$ *Conclusion* and checking all the rows. If every row is true, then the conclusion is entailed by the premises
- In terms of the environment, that means that the facts encoded in the conclusion follows from the state of affairs represented by the premises

P	H	$(P \vee H)$	$(P \vee H) \wedge \neg H$	$((P \vee H) \wedge \neg H) \Rightarrow P$
False	False	False	False	True
False	True	True	False	True
True	False	True	True	True
True	True	True	False	True

Propositional logic: Semantics

- If there was a configuration where $A \Rightarrow B$ was false, that would mean there is a A and B in the world such that the implication is not possible. I.e. the fact that you are A does not mean that you necessarily are B. The table is used to check that the implication makes sense in the world.

P	H	$(P \vee H)$	$(P \vee H) \wedge \neg H$	$((P \vee H) \wedge \neg H) \Rightarrow P$
False	False	False	False	True
False	True	True	False	True
True	False	True	True	True
True	True	True	False	True

Propositional logic: Semantics

- The machine has no idea what the conclusion means but the user could read the conclusion and use his or her interpretation of the symbols to determine what the conclusion means (in this case that there is a ghost at position $(1, 3)$)

P	H	$(P \vee H)$	$(P \vee H) \wedge \neg H$	$((P \vee H) \wedge \neg H) \Rightarrow P$
False	False	False	False	True
False	True	True	False	True
True	False	True	True	True
True	True	True	False	True

Propositional logic: Semantics

- Usually the sentences originally used to query the knowledge base by the user will refer to a world to which the computer has no independent access
- For this reason, it is essential for the agent to be able to draw conclusions that follow from the premises regardless of the world to which the sentences are intended to refer.

Propositional logic: Model

- Any world in which a sentence is true under a particular interpretation is called a model for that sentence under that particular interpretation
- As an example, the world below is a model of the sentence $S_{1,2}$ under the interpretation that this sentence means there is a stench at position $(1, 2)$. There are many other models of this sentence (we could generate new worlds with pitts and ghosts in different locations)

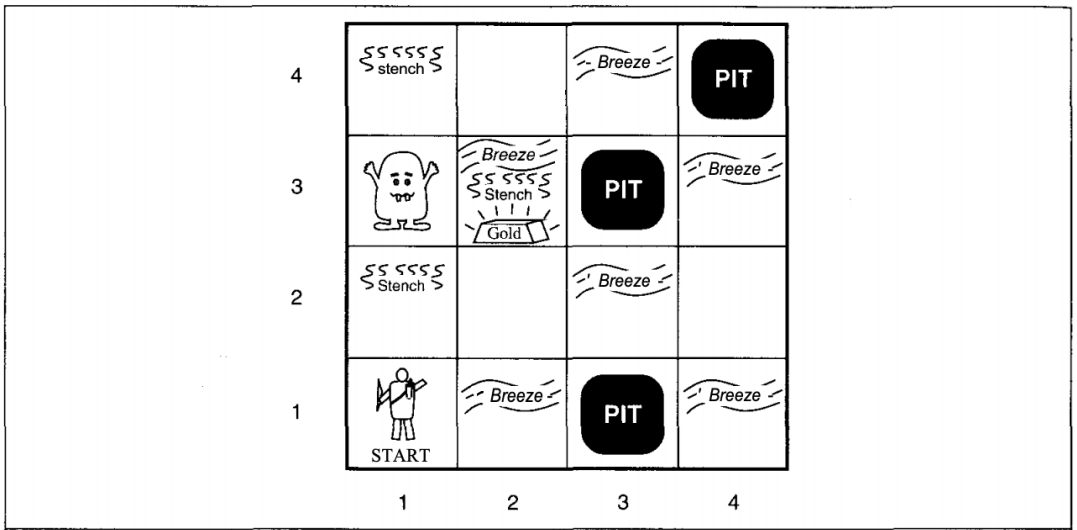


Figure 6.2 A typical wumpus world.

Propositional logic: Model

- Models are important in logic because we could restate the entailment property as 'A sentence '*a*' is entailed by a knowledge base KB if the models of KB are all models of A'. That is if KB is true, A must necessarily be true.

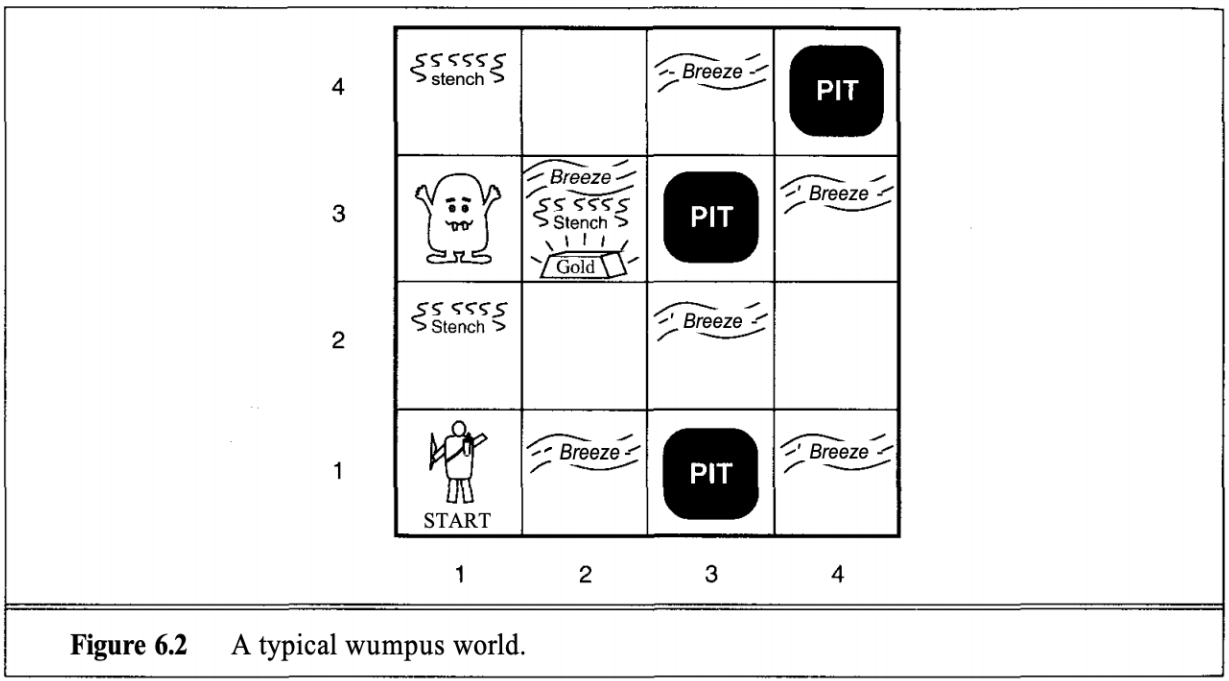


Figure 6.2 A typical wumpus world.

Propositional logic: Model

- Some authors prefer to think of models as mathematical objects (simpler than environments)
- In this case, the models of a sentence are the mappings that make the sentence true

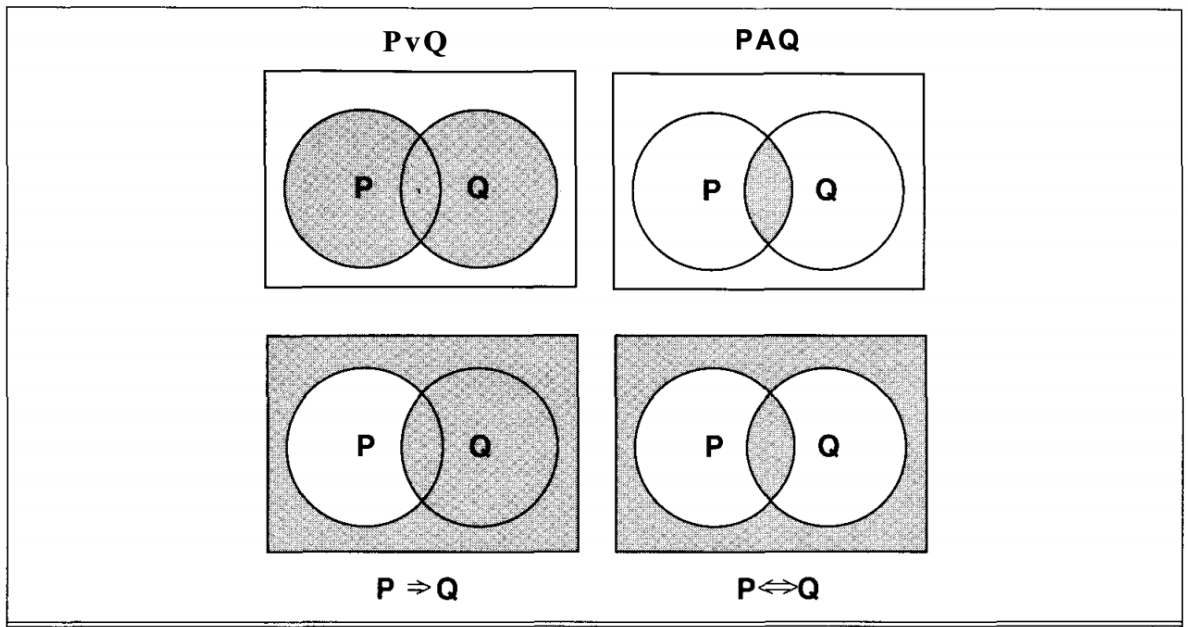


Figure 6.12 Models of complex sentences in terms of the models of their components. In each diagram, the shaded parts correspond to the models of the complex sentence.

Propositional logic: Inference

- The process by which we determine the soundness of an inference sentence through the table can be extended to entire classes of inferences. There are patterns that occu over and over again and the soundness of those patterns can be shown once and for all
- Such patterns can be captured in what is called an inference rule. Once a rule is established, it can be used to make inferences without going through the tedious process of building truth tables
- We have already seen the notation $\alpha \vdash \beta$ meaning that β can be derived from α by inference.
- An alternative notation is $\frac{\alpha}{\beta}$ which emphasizes that this is not a sentence but an inference rule. Whenever something in the KB matches the pattern above the line, the inference rule concludes the sentence below the line.

Propositional logic: Inference

- Modus Ponens or Implication Elimination. From an implication and the premise of the implication, you can infer the conclusion

$$\frac{a \Rightarrow \beta,\ a}{\beta}$$

- And Elimination From a conjunction, you can infer any of the conjuncts

$$\frac{a_1 \wedge a_2 \wedge \ldots \wedge a_n}{a_i}$$

- And-Introduction From a list of sentences, you can infer their conjunction

$$\frac{a_1, a_2, \ldots a_n}{a_1 \wedge a_2 \wedge \ldots \wedge a_n}$$

  Or-Introduction From a sentence, you can infer its disjunction with anything else

$$\frac{a_i}{a_1 \vee a_2 \vee \ldots \vee a_n}$$

Propositional logic: Inference

- Double Negation Elimination From a doubly negated sentence, you can infer a positive sentence

$$\frac{\neg\neg a}{a}$$

- Unit resolution From a disjunction, if one of the disjuncts is false, then you can infer that the other one is true

$$\frac{a \vee \beta,\ \neg\beta}{a}$$

- Resolution. Because a sentence cannot be both true and false, one of the other disjunct must be true in one of the premises Hence

$$\frac{a \vee \beta, \neg\beta \vee \gamma}{a \vee \gamma}$$

Propositional logic: Model

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

α	β	γ	$\alpha \vee \beta$	$\neg\beta \vee \gamma$	$\alpha \vee \gamma$
False	False	False	False	True	False
False	False	True	False	True	True
False	True	False	True	False	False
False	True	True	True	True	True
True	False	False	True	True	True
True	False	True	True	True	True
True	True	False	True	False	True
True	True	True	True	True	True

Propositional inference

- The truth table method of inference is complete because one can always enumerate the 2^n rows of the table for any proof involving n proposition symbols
- The computation time is however exponential in n and therefore impractical.
- One might wonder whether there is a polynomial time proof procedure for propositional logic based on using the inference rules we just defined.
- This problem is one of the first that was checked by Cook (1971) in his theory of NP-completeness. Cook showed that checking a set of sentences for satisfiability is NP-complete, hence unlikely to yield a polynomial time algorithm.

Propositional inference

- Note that this does not mean every instance of propositional inference are going to take time $O(2^n)$.
- In many cases, the proof of a given sentence refers only to a small subset of the KB and can be found fairly quickly.
- The use of inference for logical reasoning relies on a general property of certain logics (such as Propositional and First Order Logic) known as monotonicity.
- A logic is monotonic if when we add some new sentence to the knowledge base, all the sentences entailed by the original KB are still entailed by the larger KB.

Propositional inference

- Mathematically, Monotonicity reads as: If $KB_1 \models a$ then $KB_1 \cup KB_2 \models a$
- Propositional and First Order Logic are monotonic
- Probability theory is not monotonic
- We say that an inference rule is local when its premises need only be compared to a small portion of the KB. An example of a local inference rule is the modus ponens which only requires two premises.