Knowledge based agents
A knowledge-based agent

- A knowledge-based agent includes a knowledge base and an inference system.

- **Knowledge base (KB)**
  - A set of assertions (*sentences*) about the world,
  - expressed in a *knowledge representation language*.
  - Observations can be added as new sentences.
  - Queries about what *follows* from the KB
  - May initially contain some *background knowledge*

- **Inference**: deriving new sentences from old ones.
A knowledge-based agent (2)

The agent operates as follows:
- It TELLs the knowledge base what it perceives.
- It ASKs the knowledge base what action it should perform.
- It performs the chosen action.
Declarative approach

- Knowledge-based agents follow a **declarative approach** to system building
  - Can be built simply by TELLing what it needs to know.
- In contrast a **procedural approach** encodes desired behaviors directly as program code.
Architecture of a knowledge-based agent

- **Knowledge Level.**
  - The most abstract level: describe agent by saying what it knows.
  - Example: A taxi agent might know that the Golden Gate Bridge connects San Francisco with the Marin County.

- **Logical Level.**
  - The level at which the knowledge is encoded into sentences.
  - Example: `Links(GoldenGateBridge, SanFrancisco, MarinCounty)`.

- **Implementation Level.**
  - The physical representation of the sentences in the logical level.
  - Example: `'(links goldengatebridge sanfrancisco marincounty)'`
Example: The Wumpus World environment

- The Wumpus computer game
  - The agent explores a cave consisting of rooms connected by passageways.
  - Lurking somewhere in the cave is the Wumpus, a beast that eats any agent that enters its room.
  - Some rooms contain bottomless pits that trap any agent that wanders into the room.
  - Occasionally, there is a heap of gold in a room.
  - The goal is to collect the gold and exit the world without being eaten.
A typical Wumpus world

- The agent always starts in the field [1,1].
- The task of the agent is to find the gold, return to the field [1,1] and climb out of the cave.
Agent in a Wumpus world: Percepts

- The agent perceives
  - a stench in the square containing the wumpus and in the adjacent squares (not diagonally)
  - a breeze in the squares adjacent to a pit
  - a glitter in the square where the gold is
  - a bump, if it walks into a wall
  - a woeful scream everywhere in the cave, if the wumpus is killed

- The percepts are given as a five-symbol list.
  - Example: [Stench, Breeze, None, None, None]
Agent in a Wumpus world: actions

- **go forward**
- **turn right (90°)**
- **turn left (90°)**
- **grab**: Pick up an object in the same square as the agent.
- **shoot**: Fire an arrow in a straight line in the direction the agent is facing (arrow continues until it either hits and kills the wumpus or hits the outer wall). Agent has only 1 arrow, so only the 1st *Shoot* action has any effect.
- **climb** is used to leave the cave (only effective in the start square).
- **die**: This action automatically and irretrievably happens if the agent enters a square with a pit or a live wumpus.
Wumpus goal

- The agent’s goal is to find the gold and bring it back to the start square as quickly as possible, without getting killed
Exercise: make the agent find the gold

- Start at [1,1]
- Move to [2,1]: perceive a breeze (PIT either on [2,2] or [3,1])
- Go back to [1,1]
- ?
The Wumpus agent’s first step

(a) 1.4  2.4  3.4  4.4
    1.3  2.3  3.3  4.3
    1.2  2.2  3.2  4.2
    1.1  2.1  3.1  4.1
    OK   OK   OK   OK

(b) 1.4  2.2  3.4  4.4
    1.3  2.3  3.3  4.3
    1.2  2.2  3.2  4.2
    1.1  2.1  3.1  4.1
    OK   OK   OK   -W

Legend:
A = Agent
B = Broozo
G = Glitter, Gold
OK = Safe square
P = Pit
S = Stench
V = Visited
W = Wumpus
Later

<table>
<thead>
<tr>
<th></th>
<th>1.1</th>
<th>1.3</th>
<th>1.4</th>
<th>2.1</th>
<th>2.3</th>
<th>3.1</th>
<th>3.3</th>
<th>4.1</th>
<th>4.3</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>A</strong></td>
<td>V</td>
<td>2.1</td>
<td>B</td>
<td>V</td>
<td>V</td>
<td>2.2</td>
<td>S</td>
<td>2.2</td>
<td>OK</td>
</tr>
<tr>
<td><strong>S</strong></td>
<td>OK</td>
<td>2.2</td>
<td>¬W</td>
<td>¬P</td>
<td>OK</td>
<td>3.2</td>
<td>V</td>
<td>OK</td>
<td>OK</td>
</tr>
<tr>
<td><strong>G</strong></td>
<td>OK</td>
<td>3.1</td>
<td>P!</td>
<td>¬W</td>
<td>OK</td>
<td>4.1</td>
<td>OK</td>
<td>OK</td>
<td>OK</td>
</tr>
</tbody>
</table>

**Legend:**
- **A** = Agent
- **B** = Breeze
- **G** = Glitter, Gold
- **OK** = Safe square
- **P** = Pit
- **S** = Stench
- **V** = Visited
- **W** = Wumpus

(b)
Wumpuses online

Logics are formal languages for representing information such that conclusions can be drawn.

A knowledge representation language is defined by:

- its **syntax**, which specifies all valid sentences in the language
- its **semantics**, which defines the “meaning” or truth of each sentence
Semantics maps sentences in logic to facts in the world. The property of one fact following from another is mirrored by the property of one sentence being entailed by another.
Entailment and derivation

**Entailment: KB |= Q**
- The needle (Q) being in the haystack
- Q is entailed by KB if and only if the conclusion is true in every logically possible world in which all the premises in KB are true.

**Derivation: KB |- Q**
- Finding the needle (Q) in the haystack (KB)
- We can derive Q from KB if there is a proof consisting of a sequence of valid **inference** steps starting from the premises in KB and resulting in Q
Inference

- $KB \vdash_i \alpha = \text{sentence } \alpha \text{ can be derived from } KB \text{ by procedure } i$
- **Soundness**: derivations produce only entailed sentences
- **Completeness**: derivations can produce all entailed sentences
Entailment in the wumpus world

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

Consider possible models for $KB$ assuming only pits

3 Boolean choices $\Rightarrow$ 8 possible models
Wumpus models
Wumpus models

- $KB = \text{wumpus-world rules} + \text{observations}$
- $KB = \text{wumpus-world rules + observations}$
- $\alpha_1 = "[1,2] \text{ is safe"}$, $KB \models \alpha_1$, proved by model checking
Wumpus models

- $KB = \text{wumpus-world rules} + \text{observations}$
- $KB = \text{wumpus-world rules + observations}$
- $\alpha_2 = "[2,2] \text{ is safe}"$, $KB \models \alpha_2$
Propositional Logic
Propositional logic

- Propositional logic is the simplest logic – illustrates basic ideas
- Syntax - the proposition symbols $P_1$, $P_2$ are sentences
  - $\neg S$ (negation), $S_1 \land S_2$ (conjunction), $S_1 \lor S_2$ (disjunction), $S_1 \Rightarrow S_2$ (implication), $S_1 \iff S_2$ (biconditional)
- Semantics
  - $\neg S$ is true iff $S$ is false
  - $S_1 \land S_2$ is true iff $S_1$ is true and $S_2$ is true
  - $S_1 \lor S_2$ 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
  - $S_1 \iff S_2$ is true iff $S_1 \Rightarrow S_2$ is true and $S_2 \Rightarrow S_1$ is true
- Wumpus world sentences
  - $P_{i,j}$ true if pit in [i, j]; $B_{i,j}$ true if breeze in [i, j]
    - $\neg P_{1,1}$; $\neg B_{1,1}$; $B_{2,1}$
  - "Pits cause breezes in adjacent squares"
    - $B_{1,1} \iff (P_{1,2} \lor P_{2,1})$
    - $B_{2,1} \iff (P_{1,1} \lor P_{2,2} \lor P_{3,1})
Propositional logic: inference

- $KB \models \alpha$

- Different approaches:
  - Model checking – enumerate all models and check that $\alpha=\text{T}$ when $KB=\text{T}$
    - Truth table enumeration (always exponential in $n$)
    - Improved backtracking, e.g., Davis--Putnam-Logemann-Loveland (DPLL)
    - Local heuristic search in model space (sound but incomplete) e.g., min-conflicts-like hill-climbing
  - Application of inference rules to the sentences in $KB$ to construct a proof of the desired sentence.
**Truth tables for inference**

<table>
<thead>
<tr>
<th>( B_{1,1} )</th>
<th>( B_{2,1} )</th>
<th>( P_{1,1} )</th>
<th>( P_{1,2} )</th>
<th>( P_{2,1} )</th>
<th>( P_{2,2} )</th>
<th>( P_{3,1} )</th>
<th>( KB )</th>
<th>( \alpha_1 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
<td>false</td>
<td>false</td>
<td>false</td>
<td>false</td>
<td>false</td>
<td>true</td>
</tr>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
<td>false</td>
<td>false</td>
<td>false</td>
<td>false</td>
<td>false</td>
<td>true</td>
</tr>
<tr>
<td>false</td>
<td>true</td>
<td>false</td>
<td>false</td>
<td>false</td>
<td>false</td>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>false</td>
<td>true</td>
<td>false</td>
<td>false</td>
<td>false</td>
<td>false</td>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>false</td>
<td>true</td>
<td>true</td>
<td>true</td>
<td>true</td>
<td>true</td>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
</tbody>
</table>
Logical equivalence

- Two sentences are logically equivalent iff true in same models: \( \alpha \equiv \beta \) iff \( \alpha \models \beta \) and \( \beta \models \alpha \)

\[
\begin{align*}
(\alpha \land \beta) &\equiv (\beta \land \alpha) \quad \text{commutativity of } \land \\
(\alpha \lor \beta) &\equiv (\beta \lor \alpha) \quad \text{commutativity of } \lor \\
((\alpha \land \beta) \land \gamma) &\equiv (\alpha \land (\beta \land \gamma)) \quad \text{associativity of } \land \\
((\alpha \lor \beta) \lor \gamma) &\equiv (\alpha \lor (\beta \lor \gamma)) \quad \text{associativity of } \lor \\
\neg(\neg\alpha) &\equiv \alpha \quad \text{double-negation elimination} \\
(\alpha \implies \beta) &\equiv (\neg\beta \implies \neg\alpha) \quad \text{contraposition} \\
(\alpha \implies \beta) &\equiv (\neg\alpha \lor \beta) \quad \text{implication elimination} \\
(\alpha \iff \beta) &\equiv ((\alpha \implies \beta) \land (\beta \implies \alpha)) \quad \text{biconditional elimination} \\
\neg(\alpha \land \beta) &\equiv (\neg\alpha \lor \neg\beta) \quad \text{de Morgan} \\
\neg(\alpha \lor \beta) &\equiv (\neg\alpha \land \neg\beta) \quad \text{de Morgan} \\
(\alpha \land (\beta \lor \gamma)) &\equiv ((\alpha \land \beta) \lor (\alpha \land \gamma)) \quad \text{distributivity of } \land \text{ over } \lor \\
(\alpha \lor (\beta \land \gamma)) &\equiv ((\alpha \lor \beta) \land (\alpha \lor \gamma)) \quad \text{distributivity of } \lor \text{ over } \land
\end{align*}
\]
Conjunctive Normal Form (CNF)
conjunction of disjunctions of literals
clauses
E.g., \((A \lor \neg B) \land (B \lor \neg C \lor \neg D)\)

- **Resolution** inference rule (for CNF):

  \[
  \ell_i \lor \ldots \lor \ell_k, \quad m_1 \lor \ldots \lor m_n
  \]

  \[
  \ell_i \lor \ldots \lor \ell_{i-1} \lor \ell_{i+1} \lor \ldots \lor \ell_k \lor m_1 \lor \ldots \lor m_{j-1} \lor m_{j+1} \lor \ldots \lor m_n
  \]

  where \(\ell_i\) and \(m_j\) are complementary literals.
E.g., \(P_{1,3} \lor P_{2,2}, \quad \neg P_{2,2}\)

  \[
  P_{1,3}
  \]

- Resolution is sound and complete for propositional logic
Resolution algorithm

- Proof by contradiction, i.e., show $KB \land \lnot \alpha$ unsatisfiable

```plaintext
function PL-RESOLUTION(KB, \alpha) returns true or false

    clauses ← the set of clauses in the CNF representation of $KB \land \lnot \alpha$
    new ← \{
    loop do
        for each $C_i, C_j$ in clauses do
            resolvents ← PL-RESOLVE($C_i, C_j$)
            if resolvents contains the empty clause then return true
            new ← new \cup resolvents
        if new ⊆ clauses then return false
    clauses ← clauses \cup new
```

Resolution example

- $KB = (B_{1,1} \leftrightarrow (P_{1,2} \lor P_{2,1})) \land \neg B_{1,1} \land \neg P_{1,2}$
Forward and backward chaining

- Inference with Horn Clauses ($\text{KB} = \text{conjunction of Horn clauses}$)
  - Body, Head, and Fact
  - $C \land (B \Rightarrow A) \land (C \land D \Rightarrow B)$

- **Forward Chaining**
  - Fire any rule whose premises are satisfied in the $\text{KB}$
  - Add its conclusion to the $\text{KB}$, until query is found

- **Backward Chaining**
  - Work backwards from the query $q$
    - Check if $q$ is known already, or
    - Prove by BC all premises of some rule concluding $q$

- These algorithms are very natural and run in linear time (in the size of the KB)
Forward vs. backward chaining

- FC is **data-driven**, automatic, unconscious processing,
  - e.g., object recognition, routine decisions

- May do lots of work that is irrelevant to the goal

- BC is **goal-driven**, appropriate for problem-solving,
  - e.g., Where are my keys? How do I get into a PhD program?

- Complexity of BC can be **much less** than linear in size of KB
Two families of efficient algorithms for propositional inference:

- **Complete backtracking search algorithms**
  - DPLL algorithm (Davis, Putnam, Logemann, Loveland)

- **Incomplete local search algorithms**
  - WalkSAT algorithm
The DPLL algorithm

- Recursive depth first enumeration of possible models with 3 improvements

  1. Early termination
     A clause is true if any literal is true.
     A sentence is false if any clause is false.

  2. Pure symbol heuristic
     Pure symbol: always appears with the same "sign" in all clauses.
     e.g., In the three clauses (A ∨ ¬B), (¬B ∨ ¬C), (C ∨ A)
     A and B are pure, C is impure.
     Make a pure symbol literal true.

  3. Unit clause heuristic
     Unit clause: only one literal in the clause
     The only literal in a unit clause must be true.
The \texttt{WalkSAT} algorithm

- Incomplete, local search algorithm
- Evaluation function: The min-conflict heuristic of minimizing the number of unsatisfied clauses
- Balance between greediness and randomness
Intelligent agents need knowledge about the world for making good decisions.

The knowledge of an agent is stored in a knowledge base in the form of sentences in a knowledge representation language.

A knowledge-based agent needs a knowledge base and an inference mechanism. It operates by storing sentences in its knowledge base, inferring new sentences with the inference mechanism, and using them to deduce which actions to take.

A representation language is defined by its syntax and semantics, which specify the structure of sentences and how they relate to the facts of the world.

The interpretation of a sentence is the fact to which it refers. If this fact is part of the actual world, then the sentence is true.