

---

*INF5390 – Kunstig intelligens*

# **Logical Agents**

Roar Fjellheim

---

# Outline

---

- Knowledge-based agents
- The Wumpus world
- Knowledge representation
- Logical reasoning
- Propositional logic
- Wumpus agent
- Summary

AIMA Chapter 7: Logical Agents

# Knowledge-based agents

---

- *Knowledge-based* agents are able to:
  - ✓ Maintain a description of the environment
  - ✓ Deduce a course of action that will achieve goals
- Knowledge-based agents have:
  - ✓ A **knowledge base**
  - ✓ **Logical reasoning** abilities
- The performance of a knowledge-based agent is determined by its knowledge base

# Knowledge bases

---

- A *knowledge base* is a set of representations of facts about the world, called *sentences*, expressed in a *knowledge representation language*
- Knowledge base (KB) interface
  - $\text{TELL}(\text{KB}, \text{fact})$  - Add a new fact
  - $\text{fact} \Leftarrow \text{ASK}(\text{KB}, \text{query})$  - Retrieves a fact
  - $\text{RETRACT}(\text{KB}, \text{fact})$  - Removes a fact
- A knowledge-base agent can be built by TELLing it what it needs to know (*declarative* approach)
- The agent can be used to solve problems by ASKing questions

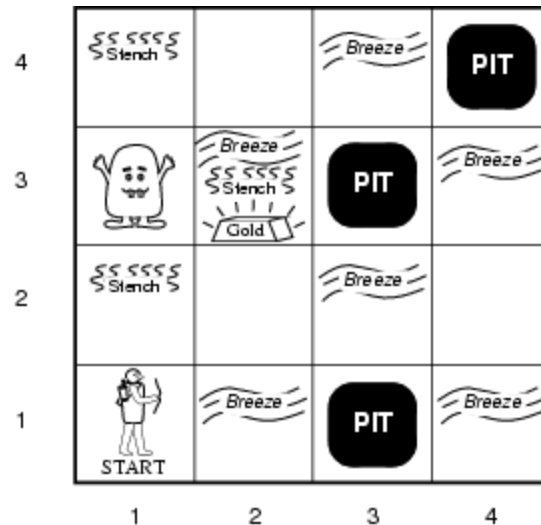
# Generic knowledge-based agent

---

```
function KB-AGENT(percept) returns an action  
persistent: KB, the agent's 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
```

# Example - The Wumpus world

- Wumpus
  - ✓ Stench
- Pits
  - ✓ Breeze
- Gold
  - ✓ Glitter
- Agent
  - ✓ Move
  - ✓ Shoot



- Performance
  - ✓ +1000 for gold, -1000 fall in pit/eaten, -1 for action, -10 using arrow
- Environment
  - ✓ 4x4 grid, start in [1,1]
- Actuators
  - ✓ Move forward, turn left 90°, turn right 90°, grab gold, shoot (one) arrow
- Sensors
  - ✓ [Stench?, Breeze?, Glitter?, Wall?, Killed?]

- Agent goal: Find gold and not get killed!
- Play at: [http://mostplays.com/play/Wumpus\\_World\\_1585](http://mostplays.com/play/Wumpus_World_1585)

# Classification of the Wumpus world

---

- Partially observable
  - √ Some aspects not directly observable, e.g. position of Wumpus
- Single-agent
  - √ Self
- Deterministic
  - √ Next state given by current state and action
- Sequential
  - √ Reward may require many steps
- Static
  - √ Wumpus does not move
- Discrete
  - √ Everything discretized
- Known
  - √ Effect of actions known

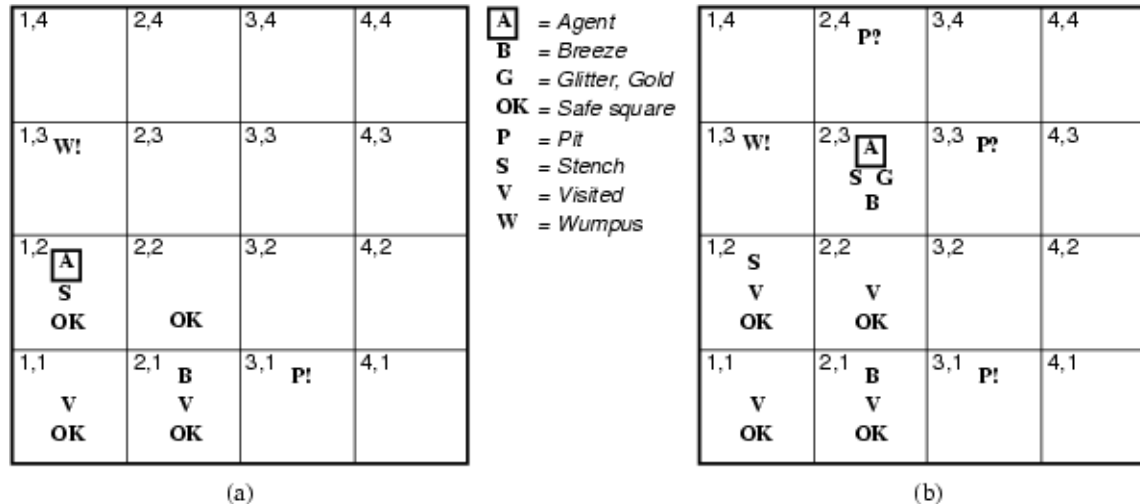
# Exploring the Wumpus world



- Initial percept: [None, None, None, None, None]
- Deduction: [1,2] and [2,1] are OK
- Action: Move right
- Second percept: [None, Breeze, None, None, None]
- Deduction: Pit in [2,2] or [3,1]
- Action sequence: Turn back and go to [1,2]



# Exploring the Wumpus world (cont.)



- Fourth percept: [Stench, None, None, None, None]
  - Deduction: Wumpus must be in [1,3], pit in [3,1]
  - Action: Move right, etc.
- 
- What does an intelligent agent need to know and how can it reason to succeed in the Wumpus world?

# Requirements for knowledge representation

---

- Knowledge representation languages should be:
  - ✓ Expressive and concise
  - ✓ Unambiguous and independent of context
  - ✓ Able to express incomplete knowledge
  - ✓ Effective inference of new knowledge
- Existing languages not suited:
  - ✓ Programming languages - *precise* descriptions and recipes for machines
  - ✓ Natural languages - *flexible* communication between humans
- In AI, logic is used for knowledge representation

# Knowledge representation languages

---

- Syntax
  - ✓ How are legal sentences in the language composed
- Semantics
  - ✓ What do the sentences *mean*
  - ✓ What is the *truth* of every sentence with respect to each *possible world* (also called a *model*)
- Entailment
  - ✓ The fact that sentences *logically follow* from other sentences
- Inference
  - ✓ How to derive new sentences that are entailed by known ones

# Logical entailment

---

- Logical *entailment* between two sentences

$$\alpha \models \beta$$

means that  $\beta$  *follows logically* from  $\alpha$ : in every *model* (possible world) in which  $\alpha$  is true,  $\beta$  is also true

- We can also say that an entire KB (all sentences in the knowledge base) entails a sentence

$$\text{KB} \models \beta$$

$\beta$  *follows logically* from KB: in every *model* (possible world) in which KB is true,  $\beta$  is also true

- Model checking: Can *check* entailment by reviewing all possible models

# Model checking – Wumpus example

---

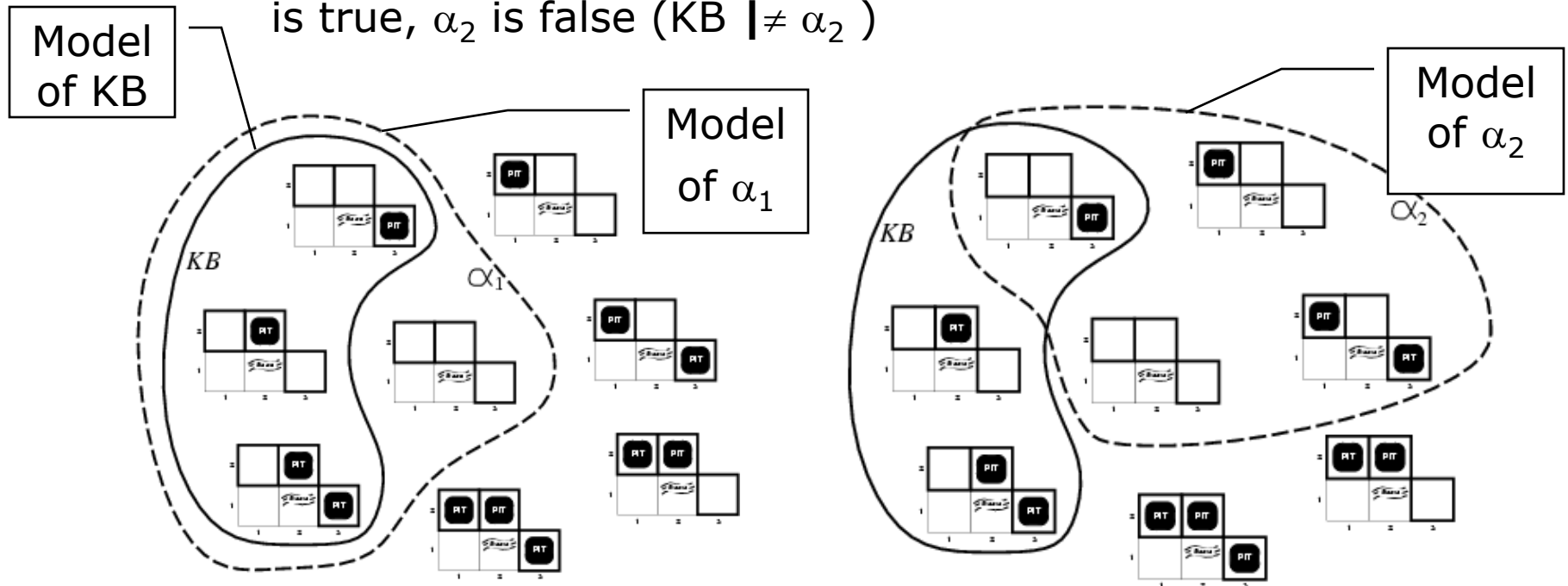
- Agent is in [2,1] and has detected a breeze
- Agent wants to know: Pits in [1,2], [2,2] and [3,1]?
- Each square may have a pit or not, i.e. there are  $2^3 = 8$  models
- KB is false in any model that contradicts what the agent knows
- Only three models in which the KB is true (next slide)

1,4	2,4	3,4	4,4
1,3	2,3	3,3	4,3
1,2 OK	2,2 P?	3,2	4,2
1,1 V OK	2,1 <input type="checkbox"/> A B OK	3,1 P?	4,1

(b)

# Model checking example (cont.)

- Check two conclusions
  - ✓  $\alpha_1 = \text{No pit in } [1,2]$  – *True* since every model where KB is true,  $\alpha_1$  is also true ( $\text{KB} \models \alpha_1$ )
  - ✓  $\alpha_2 = \text{No pit in } [2,2]$  – *False* since for some models where KB is true,  $\alpha_2$  is false ( $\text{KB} \not\models \alpha_2$ )



# Logical inference

---

- Logical *inference* between two sentences

$$\alpha \vdash \beta$$

means that  $\beta$  *can be derived* from  $\alpha$  by following an *inference algorithm* (can also say  $\text{KB} \vdash \beta$ )

- Model checking is an example of an inference algorithm

# Inference and entailment

---

- “Entailment is like the needle being in the haystack; inference is like finding it”
- *Sound* inference: The inference algorithm only derives entailed sentences
  - ✓ Required property
  - ✓ Model checking is sound
- *Complete* inference: The inference algorithm can derive any entailed sentence
  - ✓ Desirable property
  - ✓ Not always possible



# Propositional and first-order logic

---

- Propositional logic
  - ✓ Symbols represent true or false facts
  - ✓ More complex sentences can be constructed with Boolean connectives (and, or, ..)
- First-order logic
  - ✓ Symbols represent objects and predicates on objects
  - ✓ More complex sentences can be constructed with connectives and quantifiers (for-all, there-is, ..)
- In AI, both propositional and first-order logic are heavily used

# Propositional logic - syntax

---

$Sentence \rightarrow AtomicSentence \mid ComplexSentence$

$AtomicSentence \rightarrow True \mid False$

$\mid P \mid Q \mid R$

$ComplexSentence \rightarrow (Sentence)$

$\mid Sentence \text{ Connective } Sentence$

$\mid \neg Sentence$

$Connective \rightarrow \wedge \mid \vee \mid \Leftrightarrow \mid \Rightarrow$

# Logical connectives

---

- $\wedge$  (and)
  - ✓ Conjunction  $P \wedge Q$
- $\vee$  (or)
  - ✓ Disjunction  $P \vee Q$
- $\neg$  (not)
  - ✓ Negation  $\neg P$
- $\Leftrightarrow$  (equivalent)
  - ✓ Equivalence  $(P \wedge Q) \Leftrightarrow (Q \wedge P)$
- $\Rightarrow$  (implies)
  - ✓ Implication  $(P \wedge Q) \Rightarrow R$

# Propositional logic - semantics

---

- Semantics defines the rules for determining the truth of a sentence with respect to a certain model
- A model in propositional logic fixes the truth (true or false) of every propositional symbol
- The truth of a complex sentence is given by the truth value of its parts and the connectives used

# Truth table for logical connectives

---

$P$	$Q$	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
<i>False</i>	<i>False</i>	<i>True</i>	<i>False</i>	<i>False</i>	<i>True</i>	<i>True</i>
<i>False</i>	<i>True</i>	<i>True</i>	<i>False</i>	<i>True</i>	<i>True</i>	<i>False</i>
<i>True</i>	<i>False</i>	<i>False</i>	<i>False</i>	<i>True</i>	<i>False</i>	<i>False</i>
<i>True</i>	<i>True</i>	<i>False</i>	<i>True</i>	<i>True</i>	<i>True</i>	<i>True</i>

# A simple KB for the Wumpus world

---

- Propositional symbols
  - ✓  $P_{i,j}$  is true if there is a pit in  $[i,j]$
  - ✓  $B_{i,j}$  is true if there is a breeze in  $[i,j]$
- The KB contains the following sentences (and more)
  - ✓ R1:  $\neg P_{1,1}$  – No pit in  $[1,1]$
  - ✓ R2:  $B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$  – Breezy iff a pit in next cell
  - ✓ R3:  $B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$  – Same, etc.
  - ✓ R4:  $\neg B_{1,1}$  – No breeze in  $[1,1]$
  - ✓ R5:  $B_{2,1}$  – Breeze in  $[2,1]$

# Propositional inference by checking

---

- Construct truth table with one row for each combination of truth values of the propositional symbols in the sentence
- Calculate truth value of the KB sentences for each row; if all are true, the row is a model of the KB
- All other sentences that are true in the same rows as the KB is true, are entailed by the KB

# Wumpus - Model checking by truth table

<b>P[1,2]</b>	<b>P[2,2]</b>	<b>P[3,1]</b>	<b>KB</b>	<b><math>\alpha_1 = \text{Not P[1,2]}</math></b>	<b><math>\alpha_2 = \text{Not P[2,2]}</math></b>
T	T	T	F	F	F
T	T	F	F	F	F
T	F	T	F	F	T
T	F	F	F	F	T
F	T	T	<b>T</b>	T	F
F	T	F	<b>T</b>	T	F
F	F	T	<b>T</b>	T	T
F	F	F	F	T	T



# Complexity of propositional inference

---

- The checking algorithm is sound and complete, but
  - ✓ Time complexity is  $O(2^n)$
  - ✓ Space complexity is  $O(n)$
- All known inference algorithms for propositional logic have worst-case complexity that is exponential in the size of inputs

# Equivalence, validity and satisfiability

---

- Two sentences are *equivalent* if they are true in the same models
- A sentence is *valid* (necessarily true, tautological) if it is true in all possible models
- A sentence is *satisfiable* if it true in some model
- A sentence that is not satisfiable is *unsatisfiable* (contradictory)

# Inference by applying rules

---

- An *inference rule* is a standard pattern of inference that can be applied to drive chains of conclusions leading to goal
- A sequence of applications of inference rules is called a *proof*
- Searching for proofs is similar (or in some cases identical) to problem-solving by search
- Using rules for inference is an alternative to inference by model checking

# Inference rules $\frac{\alpha}{\beta}$ for propositional logic

---

- Modus ponens  $\frac{\alpha \Rightarrow \beta, \alpha}{\beta}$
- And-Elimination  $\frac{\alpha_1 \wedge \alpha_2 \wedge \mathbf{K} \wedge \alpha_2}{\alpha_1}$
- Unit Resolution  $\frac{\alpha \vee \beta, \neg\beta}{\alpha}$
- Etc.

# Ex.: Inference in the Wumpus world

---

- Want to show that there is no pit in [1,2] given R1-R5, i.e.  $\neg P_{1,2}$ 
  - ✓ R6:  $(B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$   
– Biconditional elimination in R2 (def. of equivalence)
  - ✓ R7:  $((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$  – And-Elimination in R6
  - ✓ R8:  $(\neg B_{1,1} \Rightarrow \neg(P_{1,2} \vee P_{2,1}))$  – Equivalence of contrapositives in R7
  - ✓ R9:  $\neg(P_{1,2} \vee P_{2,1})$  – Modus ponens with R8 and R4
  - ✓ R10:  $\neg P_{1,2} \wedge \neg P_{2,1}$  – Morgan's law. Neither [1,2] nor [2,1] contains a pit
- The inference path could be found by *search* as an alternative to model checking

# The resolution rule

---

- Takes two sentences with *complementary* parts and produces a new sentence with the part removed

- Example

$$\frac{P_{1,1} \vee P_{3,1} \quad \neg P_{1,1} \vee \neg P_{2,2}}{P_{3,1} \vee \neg P_{2,2}}$$

- It can be shown that resolution is sound and complete
- Any complete search algorithm, using only resolution as inference rule, can derive any conclusion entailed by any knowledge base in propositional logic

# Searching - forward and backward chaining

---

- Forward chaining
  - ✓ Start with sentences in KB and generate consequences
  - ✓ Uses inference rules in forward direction
  - ✓ Also called *data-driven* procedure
- Backward chaining
  - ✓ Start with goal to be proven and look for premises
  - ✓ Uses inference rules in backward direction
  - ✓ Also called *goal-directed* procedure
- Specialized search algorithms for propositional logic can be very efficient

# Requirements for a Wumpus agent

---

- Able to deduce as far as possible state of the world based on percept history
- Must have complete logical model of effect of its actions
- Able to keep track of world efficiently without going back in all percept history at each step
- Use logical inference to construct plans that are guaranteed to achieve its goals



# Keeping track of state of the world

---

- Knowledge base contains general knowledge of how the world works plus specific percept history
  - ✓ All percept need to be indexed by time  $t$  to capture changes in percepts:  $\neg Stench^3, Stench^4, \dots$
- The term *fluent* is used for any state variable that changes over time
- *Effect axioms* are used to describe outcome of an action at the next time step
- *Frame problem*: Effect axioms fail to state what does *not* change as a result of an action
- Solved by writing *successor-state axioms* for fluents, e.g.
  - ✓  $HaveArrow^{t+1} \Leftrightarrow (HaveArrow^t \wedge \neg Shoot^t)$
- Propositional agent requires *very large number* of rules

# A hybrid Wumpus agent

```
function HYBRID-WUMPUS-AGENT(percept) returns an action  
inputs: percept, a list, [stench, breeze, glitter, bump, scream]  
persistent: KB, knowledge base, initially containing Wumpus “physics”  
t, a counter, initially 0, indicating time; plan, an action sequence, initially empty  
TELL(KB, new percept and temporal “physics” sentences for time t)  
if glitter then // grab the gold, plan safe retreat and climb out  
    plan <= [Grab]+PLAN-ROUTE(current, [1,1], safe)+[Climb]  
if plan is empty then // plan safe route to safe unvisited square  
    plan <= PLAN-ROUTE(current, unvisited  $\cap$  safe, safe)  
if plan is empty and HaveArrowt then // plan move to shoot at Wumpus  
    plan <= PLAN-SHOT(current, possible-wumpus, safe)  
if plan is empty then // go to a square that is not provably unsafe  
    plan <= PLAN-ROUTE(current, unvisited  $\cap$  not-unsafe, safe)  
if plan is empty then // mission impossible, plan retreat and climb out  
    plan <= PLAN-ROUTE(current, [1,1], safe)+[Climb]  
action <= POP(plan)  
TELL(KB, MAKE-ACTION-SENTENCE(action, t))  
t <= t + 1  
return action
```

```
function PLAN-ROUTE(current, goals, allowed)  
returns an action sequence using A* search
```

# Summary

---

- Intelligent agents need *knowledge* stored in a *knowledge base* to reach good decisions
  - A *knowledge-based agent* applies an *inference* mechanism to the KB to reach conclusions
  - Knowledge is expressed in *sentences* in a *knowledge representation language* (with *syntax* and *semantics*)
  - A sentence logically *entails* another sentence if the latter is true in all models where the first is true
  - *Inference* is the process of deriving new sentences from old ones
  - Inference should be *sound* (only true conclusions) and *complete* (all true conclusions)
-

## Summary (cont.)

---

- *Propositional logic* consists of propositional symbols and logical connectives
- *Model-checking* can be used to check propositional entailment
- *Inference rules* can be used to find *proofs*, and *resolution* is a complete and sound rule
- *Fluents* can be used to express values of properties that change over time
- Propositional agents can be efficient for some domains, but do not scale to complex problems
- *Hybrid agents* combine propositional KB and reasoning with problems solving by search