TECHNISCHE UNIVERSITÄT DARMSTADT

Fachbereich Informatik

hessian.AI

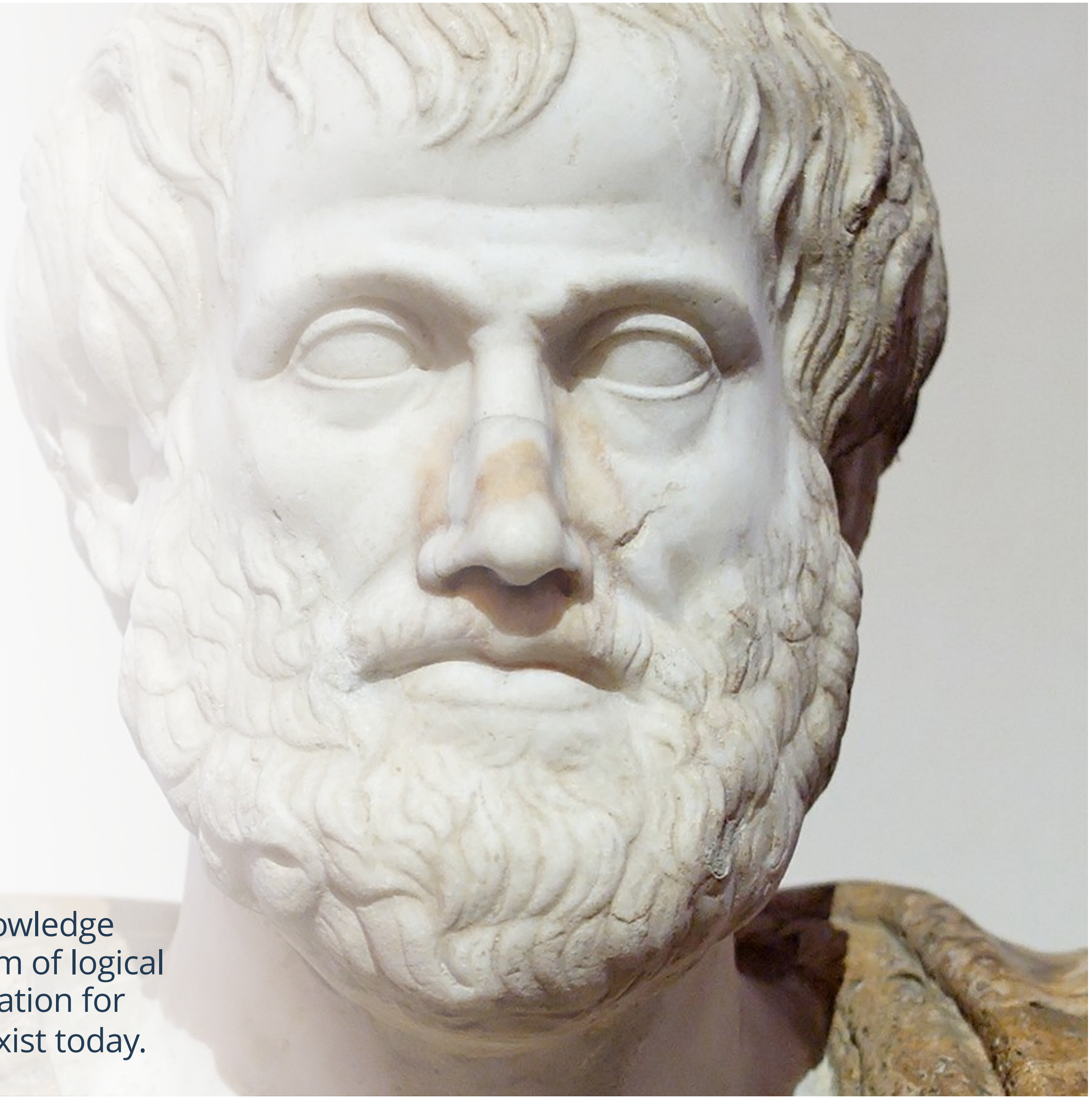
$$\begin{array}{r l} & p \\ & p \rightarrow q \\ \hline \therefore & q \end{array}$$

AI101

Lecture 6

Logic and AI 1: Propositional Logic

Aristotle was the first to classify areas of human knowledge into distinct disciplines and to create a formal system of logical reasoning. His discoveries and works laid the foundation for philosophy, science, and other fields of study that exist today.



Recap

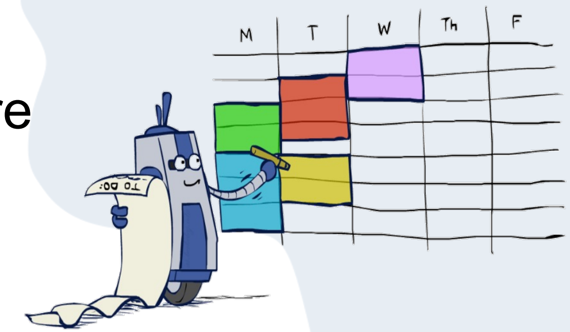
Constraint Satisfaction Problems (CSPs)

- CSPs are a special kind of search problem:
  - States defined by values of a fixed set of variables
  - Goal test defined by constraints on variable values
- Backtracking = depth-first search with one variable assigned per node
- Variable ordering and value selection heuristics help significantly
- Forward checking prevents assignments that guarantee later failure
- Constraint propagation (e.g., arc consistency) does additional work
  - To constrain values and detect inconsistencies
- The CSP representation allows analysis of problem structure
- Tree-structured CSPs can be solved in linear time

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

5	3	4	6	7	8	9	1	2
6	7	2	1	9	5	3	4	8
1	9	8	3	4	2	5	6	7
8	5	9	7	6	1	4	2	3
4	2	6	8	5	3	7	9	1
7	1	3	9	2	4	8	5	6
9	6	1	5	3	7	2	8	4
2	8	7	4	1	9	6	3	5
3	4	5	2	8	6	1	7	9

```
   SEND
+  MORE
--------
  MONEY
```



Search algorithms and "Understanding"

Search algorithms generate successors and evaluate them, but do not "understand" much about the setting

Example question: is it possible for a chess player to have 8 pawns and 2 queens?

- Search algorithm could search through tons of states to see if this ever happens, but…

Logic and AI — Propositional Logic
Goals and Outline

Can we make our agent smarter with logical reasoning?

Would like our AI to have knowledge about the world, and logically draw conclusions from it

Today

Logical Agents

Propositional Logic

- Syntax
- Semantics
- Resolution

Recap

A Short Definition of Syntax and Semantic

Syntax

It is the sequence of a specific language which should be followed in order to form a sentence. Syntax is the representation of a language and is related to grammar and structure.

Semantic

The sentence or the syntax which a logic follows should be meaningful. Semantics defines the sense of the sentence which relates to the real world.

Example: Programming (in C)

Syntax: Describe structural rules, i.e. separate statements with a semi-colon, …

- If it compiles, the syntax is most likely correct

Semantics: Does the program work? Does it makes sense?

Logic and AI
What is Logic

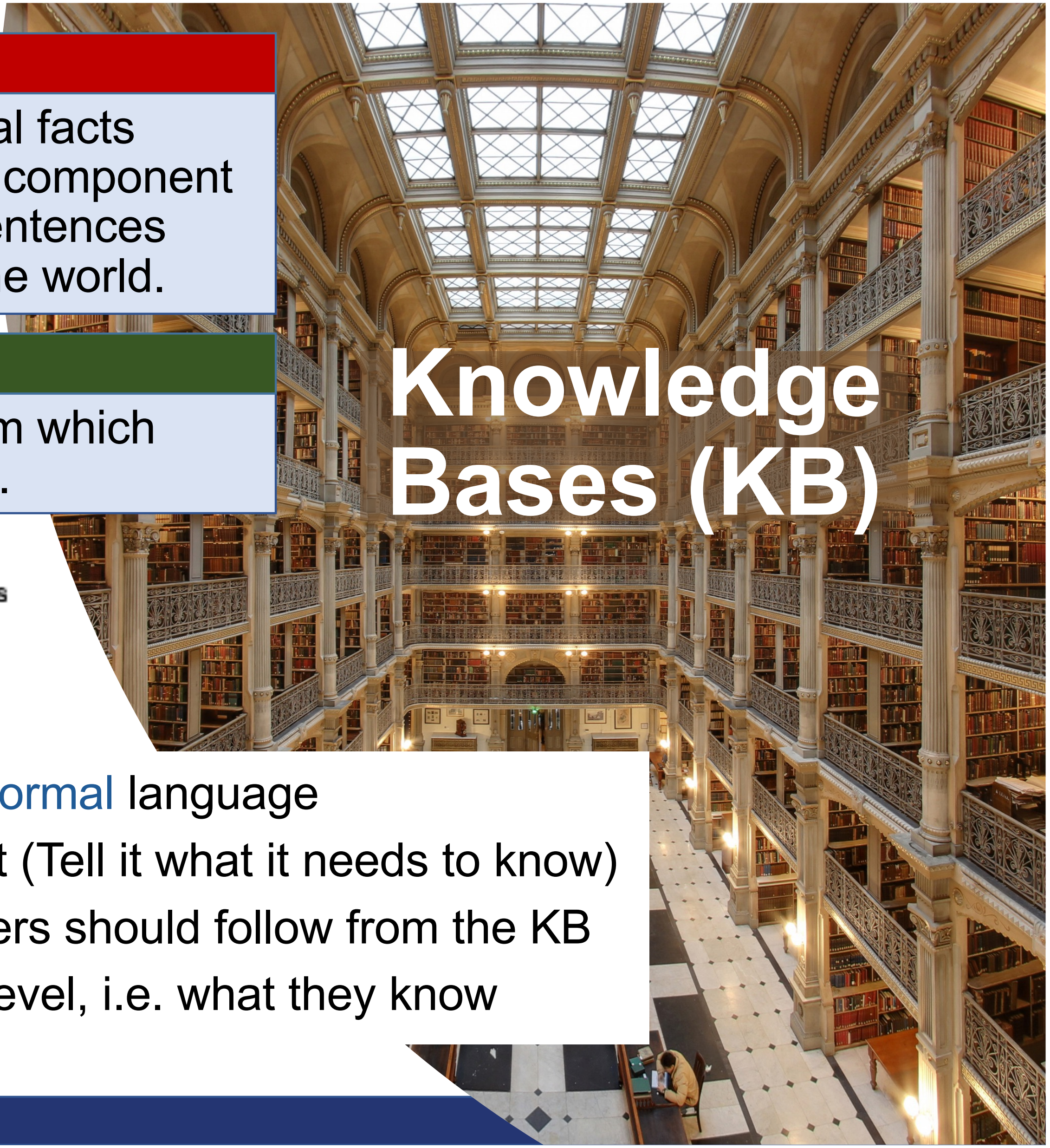
Logic

Logic is the key behind any (formal) knowledge. It allows a person to filter the necessary information from the bulk and draw a conclusion. In artificial intelligence, the representation of knowledge is done via logics.

Wikipedia: Logic is the study of correct reasoning. It includes both formal and informal logic. Formal logic is the science of deductively valid inferences or of logical truths. It is a formal science investigating how conclusions follow from premises in a topic-neutral way. When used as a countable noun, the term "a logic" refers to a logical formal system that articulates a proof system. Formal logic contrasts with informal logic, which is associated with informal fallacies, critical thinking, and argumentation theory.

$$\begin{array}{l} p \\ p \rightarrow q \\ \hline \therefore q \end{array}$$

Knowledge Bases (KB)

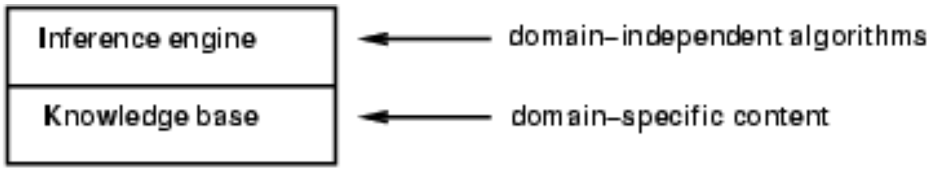


Knowledge Base (KB)

A knowledge base (KB) represents the actual facts which exist in the real world. It is the central component of a knowledge-based agent. It is a set of sentences which describes the information related to the world.

Inference Engine

It is the engine of a knowledge-based system which allows to infer new knowledge in the system.



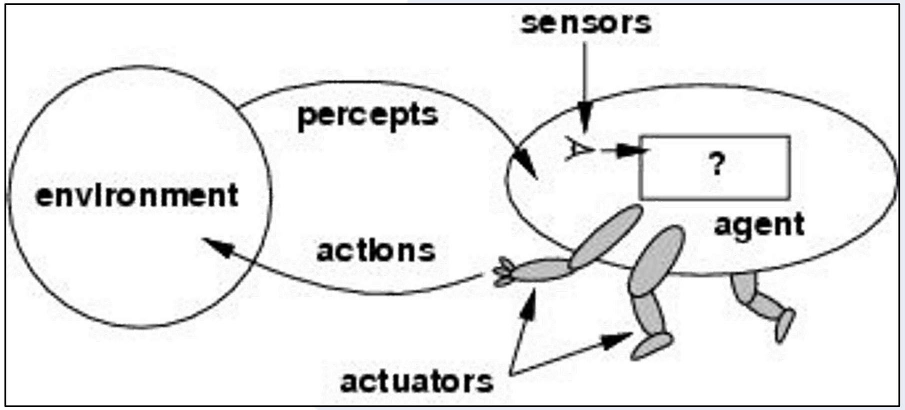
- Knowledge base = set of sentences in a formal language
- Declarative approach to building an agent (Tell it what it needs to know)
- Then it can “Ask” itself what to do - answers should follow from the KB
- Agents can be viewed at the knowledge level, i.e. what they know

Image: Wikipedia, Peabody Library (Baltimore)

Agents
Intelligent Agents

Intelligent Agents

An intelligent agent is a goal-directed agent. It perceives its environment through its sensors using the observations and built-in knowledge, acts upon the environment through its actuators.



Recap: An agent can be viewed as anything that perceives its environment through sensors and acts upon that environment through actuators.

Example: human beings perceive their surroundings through their sensory organs known as sensors and take actions using their hands, legs, etc., known as actuators.

Agents
Knowledge-based Agents

```
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
```

A knowledge-based agent must be able to:

- Represent states, actions, etc...
- Incorporate new percepts
- Update internal representations of the world
- Deduce hidden properties of the world
- Deduce appropriate actions

Logic and AI
Roommate Story

You roommate comes home; he/she/they is completely wet

You know the following things:

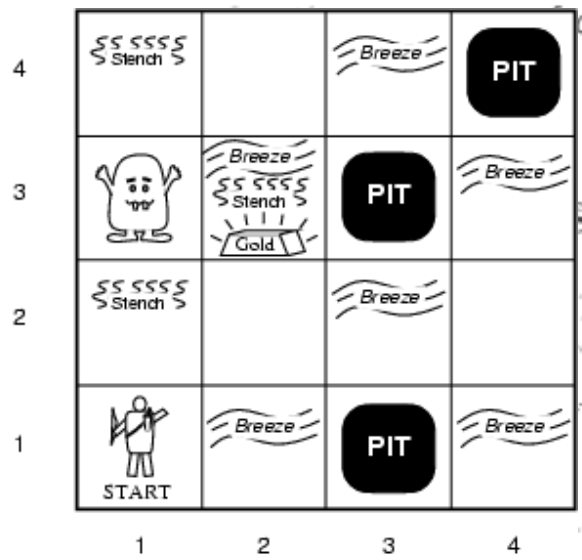
- Your roommate is wet
- If your roommate is wet, it is because of rain, sprinklers, or both
- If your roommate is wet because of sprinklers, the sprinklers must be on
- If your roommate is wet because of rain, your roommate must not be carrying the umbrella
- The umbrella is not in the umbrella holder
- If the umbrella is not in the umbrella holder, either you must be carrying the umbrella, or your roommate must be carrying the umbrella
- You are not carrying the umbrella

Can you conclude that the sprinklers are on?

Can AI conclude that the sprinklers are on?

WikiMedia Commons, A man walks through heavy rain under an umbrella in Kigali, Rwanda. Emmanuel Kwizera.jpg

Logic and AI
Wumpus



Performance measure

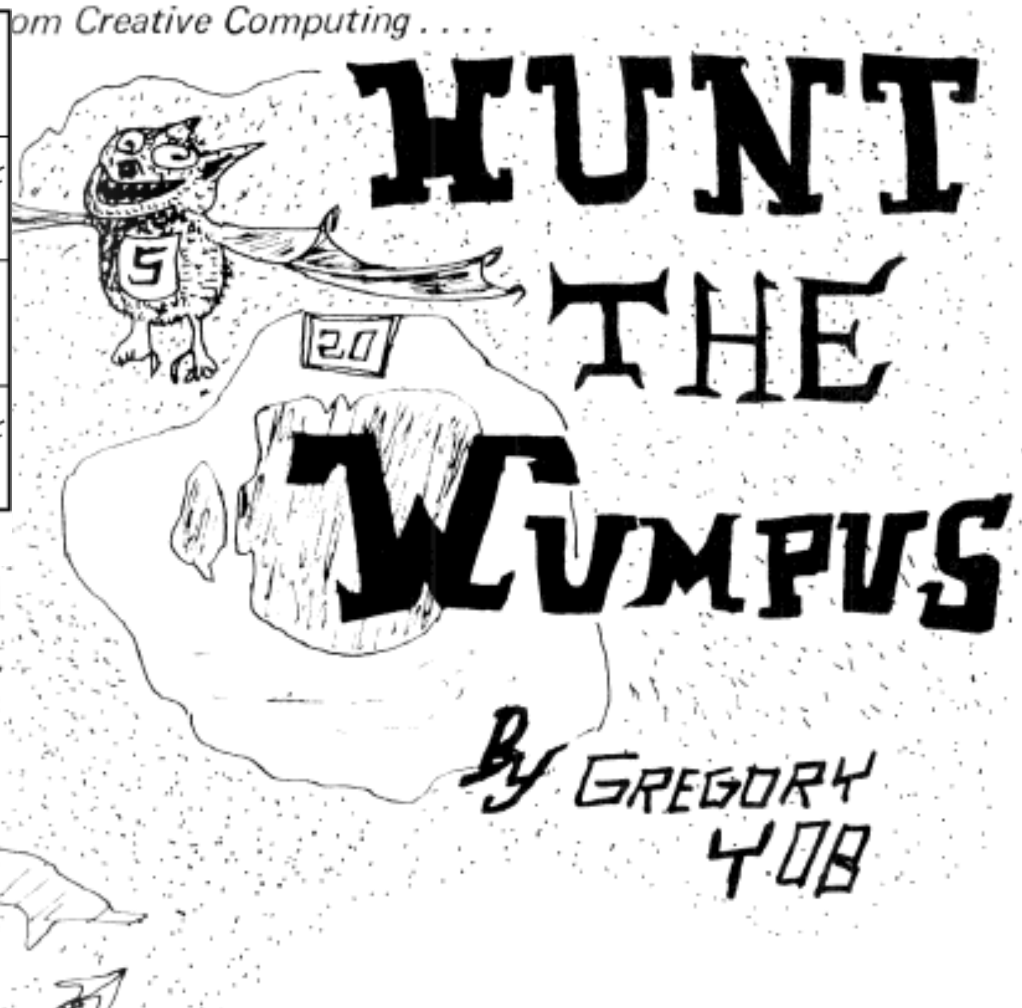
- gold +1000, death -1000
- -1 per step
- -10 for using the arrow

Environment

- Squares adjacent to wumpus are smelly
- Squares adjacent to pit are breezy
- Glitter iff gold is in the same square
- Shooting kills wumpus if you are facing it
- Shooting uses up the only arrow
- Grabbing picks up gold if in same square
- Releasing drops the gold in same square

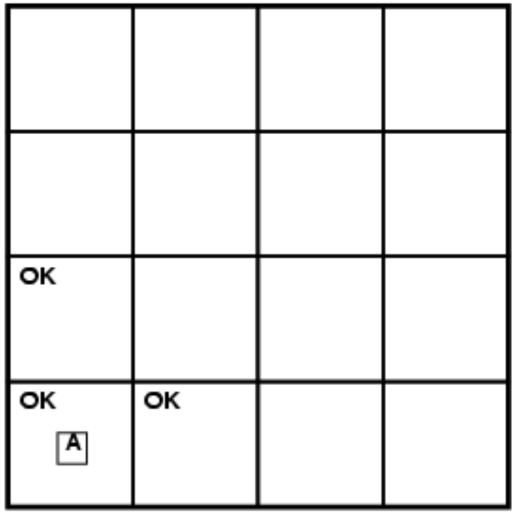
Sensors: Stench, Breeze, Glitter, Bump, Scream

Actuators: Left turn, Right turn, Forward, Grab, Release, Shoot



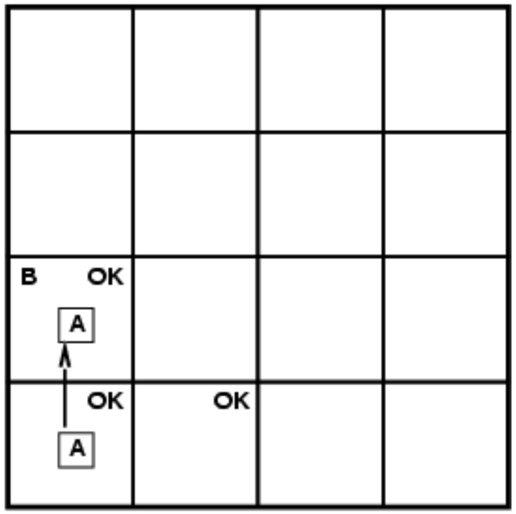
According to Wikipedia, “Hunt the Wumpus” has been cited as an early example of the survival horror genre and was listed in 2012 on Time's All-Time 100 greatest video games list. The Wumpus monster has appeared in several forms in media since 1973, including other video games, a novella, and Magic: The Gathering cards

Logic and AI
Exploring the Wumpus World



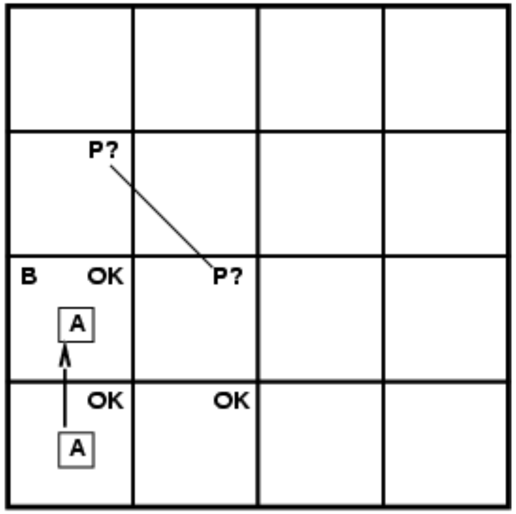
Sensors: Stench, Breeze, Glitter, Bump, Scream

Logic and AI
Exploring the Wumpus World



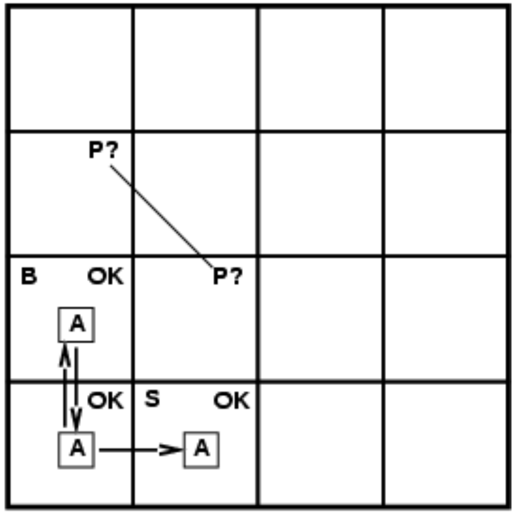
Sensors: Stench, Breeze, Glitter, Bump, Scream

Logic and AI
Exploring the Wumpus World



Sensors: Stench, Breeze, Glitter, Bump, Scream

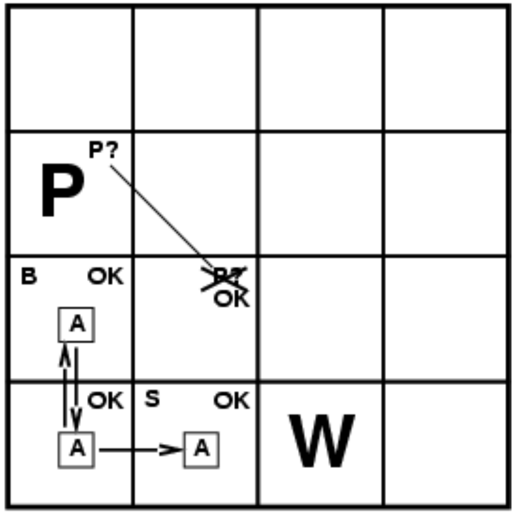
Logic and AI
Exploring the Wumpus World



Sensors: Stench, Breeze, Glitter, Bump, Scream

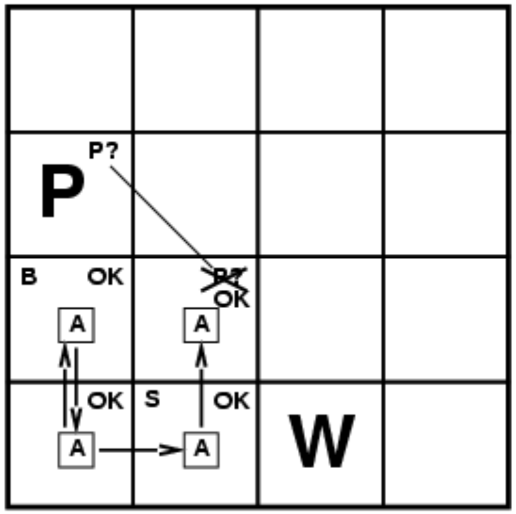
Logic and AI
Exploring the Wumpus World

Reasoning
turns
observation
into knoweldge



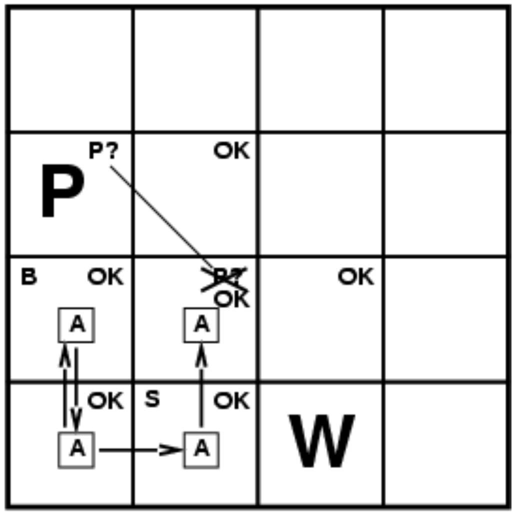
Sensors: Stench, Breeze, Glitter, Bump, Scream

Logic and AI
Exploring the Wumpus World



Sensors: Stench, Breeze, Glitter, Bump, Scream

Logic and AI
Exploring the Wumpus World



Sensors: Stench, Breeze, Glitter, Bump, Scream

Syntax of Propositional Logic

What do well-formed sentences in the knowledge base look like?

We use Backus-Naur form (BNF) for that

(Remember your APL/AFE lecture)

- *Symbol* → *P, Q, R, …, RoommateWet, …*
- *Sentence* → *True* | *False* | *Symbol* |
  NOT(Sentence) |
  (Sentence AND Sentence) |
  (Sentence OR Sentence) |
  (Sentence => Sentence)

Even if the parentheses are not on every slide, formally they should always be there

Knowledge Base
Example: Roommate Story

- RoommateWet
  - RoommateWet => (RoommateWetBecauseOfRain OR RoommateWetBecauseOfSprinklers)
  - RoommateWetBecauseOfSprinklers => SprinklersOn
  - RoommateWetBecauseOfRain => NOT(RoommateCarryingUmbrella)
- UmbrellaGone
  - UmbrellaGone => (YouCarryingUmbrella OR RoommateCarryingUmbrella)
  - NOT(YouCarryingUmbrella)

Semantics

Interpretation specifies which of the proposition symbols are true and which are false

Given a interpretation, I should be able to tell you whether a sentence is true or false.

Truth table defines semantics of operators:

a	b	NOT(a)	a AND b	a OR b	a => b
false	false	true	false	false	true
false	true	true	false	true	true
true	false	false	false	true	false
true	true	false	true	true	true

A model (of a set of sentences) is an interpretation in which all the sentences are true.

Caveats

TwoIsAnEvenNumber OR ThreeIsAnOddNumber

is true (not exclusive OR)

TwoIsAnOddNumber => ThreeIsAnEvenNumber

is true (if the left side is false it’s always true)

All of this is assuming those symbols are assigned their natural values…

Tautologies

Tautology

A sentence is a **tautology** if it is true for any setting of its propositional symbols

P	Q	P OR Q	NOT(P) AND NOT(Q)	(P OR Q) OR (NOT(P) AND NOT(Q))
false	false	false	true	true
false	true	true	false	true
true	false	true	false	true
true	true	true	false	true

(P OR Q) OR (NOT(P) AND NOT(Q)) is a tautology

Tautologies
Is This a Tautology?

Tautology

A sentence is a **tautology** if it is true for any setting of its propositional symbols

Is this a tautology?

(P => Q) OR (Q => P)

Logical equivalences

Logical Equivalence

Two sentences are **logically equivalent** if they have the same truth value for every setting of their propositional variables

P	Q	P OR Q	NOT(NOT(P) AND NOT(Q))
false	false	false	false
false	true	true	true
true	false	true	true
true	true	true	true

(P OR Q) OR (NOT(P) AND NOT(Q)) is a logically equivalent

Famous Logical Equivalences

They can be used for rewriting and simplifying rules

(a OR b) ≡ (b OR a) *commutatitvity*

(a AND b) ≡ (b AND a) *commutatitvity*

((a AND b) AND c) ≡ (a AND (b AND c)) *associativity*

((a OR b) OR c) ≡ (a OR (b OR c)) *associativity*

NOT(NOT(a)) ≡ a *double-negation elimination*

(a => b) ≡ (NOT(b) => NOT(a)) *contraposition*

(a => b) ≡ (NOT(a) OR b) *implication elimination*

NOT(a AND b) ≡ (NOT(a) OR NOT(b)) *De Morgan*

NOT(a OR b) ≡ (NOT(a) AND NOT(b)) *De Morgan*

(a AND (b OR c)) ≡ ((a AND b) OR (a AND c)) *distributitivity*

(a OR (b AND c)) ≡ ((a OR b) AND (a OR c)) *distributitivity*

Tautologies
Is this a Tautology?

(P => Q) OR (Q => P)

- (not(P) OR Q) OR (not(Q) or P)
- not(P) OR Q OR not(Q) or P
- (not(P) OR P) OR (not(Q) or Q)
- (true) OR (true)
- true

Inference / Entailment

- We have a knowledge base of things that we know are true
  - RoommateWetBecauseOfSprinklers
  - RoommateWetBecauseOfSprinklers => SprinklersOn

Can we conclude that SprinklersOn?

- We say SprinklersOn is entailed by the knowledge base if, for every setting (models) of the propositional variables for which the knowledge base is true, SprinklersOn is also true

RWBOS	SprinklersOn	Knowledge base
false	false	false
false	true	false
true	false	false
true	true	true

Inference / Entailment

Simple Algorithm for Inference

Want to find out if sentence a is entailed by knowledge base…

Idea:

Go through the possible settings of the propositional variables,

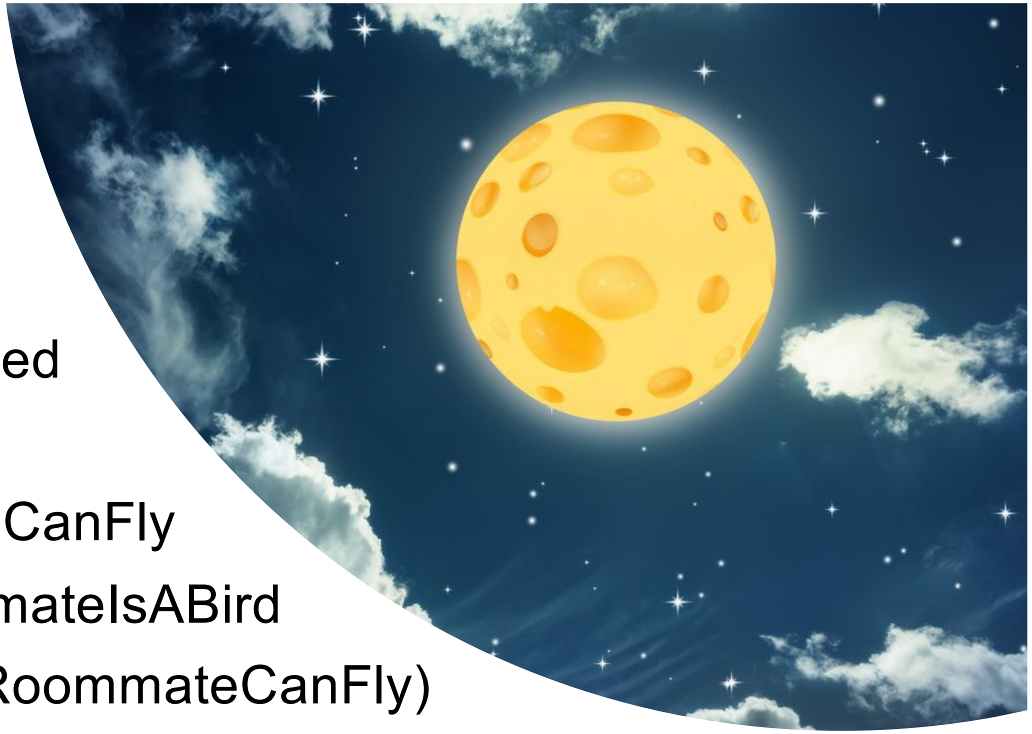
- If knowledge base is true and a is false, return false
    - Else: Return true

Problem:

Not very efficient:

- Number of settings: $2^{\#propositional\,variables}$

https://www.eater.com/2015/9/17/9341079/investigation-is-the-moon-made-of-cheese



Consistency in Knowledge bases

Suppose we were careless in how we specified our knowledge base:

PetOfRoommateIsABird => PetOfRoommateCanFly

PetOfRoommateIsAPenguin => PetOfRoommateIsABird

PetOfRoommateIsAPenguin => NOT(PetOfRoommateCanFly)

PetOfRoommateIsAPenguin

Problem: It entails both PetOfRoommateCanFly and NOT(PetOfRoommateCanFly)

→ Therefore, technically, this knowledge base implies anything:

→ The Moon Is Made Of Cheese

Important Message:
Make sure that your (logical) knowledge base is consistent!

The Moon is Made of Cheese
The Principle of Non-Contradiction

PetOfRoommateCanFly AND NOT(PetOfRoommateCanFly)

PetOfRoommateCanFly, NOT(PetOfRoommateCanFly)

Now, “a true statement OR anything else” is always true, hence

- PetOfRoommateCanFly OR MoonMadeOfCheese

Therefore, MoonMadeOfCheese has to be true.

Please note that you can really put anything there!

So, we justify the Aristotelian claim that “there cannot be contradictions” (The Principle of Non-Contradiction)

The Law of Non-Contradiction

A cannot be not-A

Two contradictory statements cannot both be true at the same time and in the same way.

Aristotle

Reasoning Patterns and Modus Ponens

Obtain new sentences directly from some other sentences in knowledge base according to **reasoning patterns:**

- All of the logical equivalences from before give us reasoning patterns

Modus Ponens:

- Another reasoning pattern
- If we have sentences a and $a \Rightarrow b$, we can correctly conclude the new sentence b

Reasoning Patterns
Proof that the sprinklers are on

Knowledge Base

1) RoommateWet
2) RoommateWet => (RoommateWetBecauseOfRain OR RoommateWetBecauseOfSprinklers)
3) RoommateWetBecauseOfSprinklers => SprinklersOn
4) RoommateWetBecauseOfRain => NOT(RoommateCarryingUmbrella)
5) UmbrellaGone
6) UmbrellaGone => (YouCarryingUmbrella OR RoommateCarryingUmbrella)
7) NOT(YouCarryingUmbrella)

Rephrasing/Reasoning

8) YouCarryingUmbrella OR RoommateCarryingUmbrella *(modus ponens on 5 and 6)*
9) NOT(YouCarryingUmbrella) => RoommateCarryingUmbrella *(equivalent to 8)*
10) RoommateCarryingUmbrella *(modus ponens on 7 and 9)*
11) NOT(NOT(RoommateCarryingUmbrella) *(equivalent to 10)*
12) NOT(NOT(RoommateCarryingUmbrella)) => NOT(RoommateWetBecauseOfRain) *(equivalent to 4 by contraposition)*
13) NOT(RoommateWetBecauseOfRain) *(modus ponens on 11 and 12)*
14) RoommateWetBecauseOfRain OR RoommateWetBecauseOfSprinklers *(modus ponens on 1 and 2)*
15) NOT(RoommateWetBecauseOfRain) => RoommateWetBecauseOfSprinklers *(equivalent to 14)*
16) RoommateWetBecauseOfSprinklers *(modus ponens on 13 and 15)*
17) SprinklersOn *(modus ponens on 16 and 3)*

Reasoning about Penguins
Back to the Moon

1) PetOfRoommateIsABird => PetOfRoommateCanFly
2) PetOfRoommateIsAPenguin => PetOfRoommateIsABird
3) PetOfRoommateIsAPenguin => NOT(PetOfRoommateCanFly)
4) PetOfRoommateIsAPenguin
5) PetOfRoommateIsABird *(modus ponens on 4 and 2)*
6) PetOfRoommateCanFly *(modus ponens on 5 and 1)*
7) NOT(PetOfRoommateCanFly) *(modus ponens on 4 and 3)*
8) NOT(PetOfRoommateCanFly) => FALSE *(equivalent to 6)*
9) FALSE *(modus ponens on 7 and 8)*
10) FALSE => TheMoonIsMadeOfCheese *(tautology)*
11) TheMoonIsMadeOfCheese *(modus ponens on 9 and 10)*

Conjunctive Normal Form (CNF)
Getting more Systematic

Any KB can be written as a single formula in **Conjunctive Normal Form (CNF)**

- CNF formula: (… OR … OR …) AND (… OR …) AND …
- … can be a symbol x, or NOT(x) (these are called **Literals**)
- Multiple facts in knowledge base are effectively ANDed together

Example

RoommateWet => (RoommateWetBecauseOfRain OR RoommateWetBecauseOfSprinklers)

becomes

(NOT(RoommateWet) OR RoommateWetBecauseOfRain OR RoommateWetBecauseOfSprinklers)

Conjunctive normal form (CNF)
Converting the Roommate Story Problem to CNF

RoommateWet
- RoommateWet

RoommateWet => (RoommateWetBecauseOfRain OR RoommateWetBecauseOfSprinklers)
- NOT(RoommateWet) OR RoommateWetBecauseOfRain OR RoommateWetBecauseOfSprinklers

RoommateWetBecauseOfSprinklers => SprinklersOn
- NOT(RoommateWetBecauseOfSprinklers) OR SprinklersOn

RoommateWetBecauseOfRain => NOT(RoommateCarryingUmbrella)
- NOT(RoommateWetBecauseOfRain) OR NOT(RoommateCarryingUmbrella)

UmbrellaGone
- UmbrellaGone

UmbrellaGone => (YouCarryingUmbrella OR RoommateCarryingUmbrella)
- NOT(UmbrellaGone) OR YouCarryingUmbrella OR RoommateCarryingUmbrella

NOT(YouCarryingUmbrella)
- NOT(YouCarryingUmbrella)

Unit resolution

Now that we have a normal form, we can implement general reasoning patterns for this normal form. One of them is

If we have

- l_1 OR l_2 OR … OR l_k

and

- NOT(l_i)

we can conclude

- l_1 OR l_2 OR … l_{i-1} OR l_{i+1} OR … OR l_k

This is modus ponens

Unit resolution

Applying Resolution to the Roommate Story Problem

1) RoommateWet
2) NOT(RoommateWet) OR RoommateWetBecauseOfRain OR RoommateWetBecauseOfSprinklers
3) NOT(RoommateWetBecauseOfSprinklers) OR SprinklersOn
4) NOT(RoommateWetBecauseOfRain) OR NOT(RoommateCarryingUmbrella)
5) UmbrellaGone
6) NOT(UmbrellaGone) OR YouCarryingUmbrella OR RoommateCarryingUmbrella
7) NOT(YouCarryingUmbrella)

Knowledge Base

Unit Resolution

8) NOT(UmbrellaGone) OR RoommateCarryingUmbrella (6,7)
9) RoommateCarryingUmbrella (5,8)
10) NOT(RoommateWetBecauseOfRain) (4,9)
11) NOT(RoommateWet) OR RoommateWetBecauseOfSprinklers (2,10)
12) RoommateWetBecauseOfSprinklers (1,11)
13) SprinklersOn (3,12)

Limitations of unit resolution

Unfortunately, unit resolution is not enough!

- P OR Q
- NOT(P) OR Q

Can we conclude Q?

(General) Resolution

if we have

l_1 OR l_2 OR … OR l_k

and

m_1 OR m_2 OR … OR m_n

where for some i,j, l_i = NOT(m_j)

we can conclude

l_1 OR l_2 OR … l_{i-1} OR l_{i+1} OR … OR l_k OR m_1 OR m_2 OR … OR m_{j-1} OR m_{j+1} OR … OR m_n

Same literal may appear multiple times; remove those

(General) Resolution

Applying Resolution to the Roommate Story Problem

1) RoommateWet
2) NOT(RoommateWet) OR RoommateWetBecauseOfRain OR RoommateWetBecauseOfSprinklers
3) NOT(RoommateWetBecauseOfSprinklers) OR SprinklersOn
4) NOT(RoommateWetBecauseOfRain) OR NOT(RoommateCarryingUmbrella)
5) UmbrellaGone
6) NOT(UmbrellaGone) OR YouCarryingUmbrella OR RoommateCarryingUmbrella
7) NOT(YouCarryingUmbrella)

Knowledge Base

Resolution

8) NOT(RoommateWet) OR RoommateWetBecauseOfRain OR SprinklersOn (2,3)
9) NOT(RoommateCarryingUmbrella) OR NOT(RoommateWet) OR SprinklersOn (4,8)
10) NOT(UmbrellaGone) OR YouCarryingUmbrella OR NOT(RoommateWet) OR SprinklersOn (6,9)
11) YouCarryingUmbrella OR NOT(RoommateWet) OR SprinklersOn (5,10)
12) NOT(RoommateWet) OR SprinklersOn (7,11)
13) SprinklersOn (1,12)

How to use Resolution for Systematic Inference?

Satisfiable
There exists a model that makes the modified knowledge base (KB) true, i.e., the modified knowledge base is consistent

Strategy: if we want to see if sentence a is entailed, add NOT(a) to the knowledge base and see if it becomes inconsistent (we can derive a contradiction)

→ CNF formula for modified knowledge base is satisfiable if and only if sentence a is not entailed

Resolution Algorithm

Given formula in conjunctive normal form, in 4 steps:

Repeat:

1. **Find** two clauses with complementary literals
2. **Apply** resolution
3. **Add** resulting clause (if not already there)
4. **Test**, if it results in the empty clause, formula is not satisfiable

Resolution Algorithm
Example

Our knowledge base:

- 1) RoommateWetBecauseOfSprinklers,
- 2) NOT(RoommateWetBecauseOfSprinklers) OR SprinklersOn

Can we infer SprinklersOn?

We add:

- 3) NOT(SprinklersOn)

From 2) and 3), get

- 4) NOT(RoommateWetBecauseOfSprinklers)

From 4) and 1), get empty clause

Horn Clauses
The special case

Horn Clauses

Horn Clauses are implications with only positive literals.

$$X_1 \text{ AND } X_2 \text{ AND } X_4 \rightarrow X_3 \text{ AND } X_6$$
$$\text{TRUE} \rightarrow X_1$$

- Try to figure out whether some X_j is entailed
- Simply follow the implications (Modus Ponens) as far as you can, see if you can reach X_j
- X_j is entailed if and only if it can be reached
- Can implement this more efficiently by maintaining, for each implication, a count of how many of the left-hand side variables have been reached

Limitations of Propositional Logic

Some English statements are hard to model in propositional logic:

“If your roommate is wet because of rain, your roommate must not be carrying any umbrella”

Pathetic attempt at modeling this:

RoommateWetBecauseOfRain =>
(NOT(RoommateCarryingUmbrella0) AND
NOT(RoommateCarryingUmbrella1) AND
NOT(RoommateCarryingUmbrella2) AND …)

Limitations of Propositional Logic

- No notion of objects
- No notion of relations among objects

- RoommateCarryingUmbrella is instructive to us, suggesting that
    - there is an object we call Roomate
    - there is an object we cal Umbrella
    - there is a relationship called Carrying between these objects

- Formally, none of this meaning is there

  Might as well have replaced “RoommateCarryingUmbrella” by “P”?

Summary
Propositional Logic

- Syntax & Semantics
- Wumpus
- Equivalence of logical statements
- Consistency
- Satisfiable
- Clausal Normal Form
- (Unit) Resolution
- Horn clauses

You should be able to:

- Translate English 2 Logic and vv
- Rewrite logical statements keeping semantics
- Convert to CNF
- Prove statements using resolution

Next Week: Logic and AI 2 – First Order Logic

Wumpus in propositional logic

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].

$\neg P_{1,1}$

$\neg B_{1,1}$

$B_{2,1}$

"Pits cause breezes in adjacent squares"

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

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

Wumpus in propositional logic

$B_{1,1}$	$B_{2,1}$	$P_{1,1}$	$P_{1,2}$	$P_{2,1}$	$P_{2,2}$	$P_{3,1}$	KB	α_1
$false$	$false$	$false$	$false$	$false$	$false$	$false$	$false$	$true$
$false$	$false$	$false$	$false$	$false$	$false$	$true$	$false$	$true$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$false$	$true$	$false$	$false$	$false$	$false$	$false$	$false$	$true$
$false$	$true$	$false$	$false$	$false$	$false$	$true$	$\underline{true}$	$\underline{true}$
$false$	$true$	$false$	$false$	$false$	$true$	$false$	$\underline{true}$	$\underline{true}$
$false$	$true$	$false$	$false$	$false$	$true$	$true$	$\underline{true}$	$\underline{true}$
$false$	$true$	$false$	$false$	$true$	$false$	$false$	$false$	$true$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$true$	$true$	$true$	$true$	$true$	$true$	$true$	$false$	$false$

Wumpus in propositional logic

A wumpus-world agent using propositional logic:

$\neg P_{1,1}$
$\neg W_{1,1}$
$B_{x,y} \Leftrightarrow (P_{x,y+1} \vee P_{x,y-1} \vee P_{x+1,y} \vee P_{x-1,y})$
$S_{x,y} \Leftrightarrow (W_{x,y+1} \vee W_{x,y-1} \vee W_{x+1,y} \vee W_{x-1,y})$
$W_{1,1} \vee W_{1,2} \vee \ldots \vee W_{4,4}$
$\neg W_{1,1} \vee \neg W_{1,2}$
$\neg W_{1,1} \vee \neg W_{1,3}$
…

$\Rightarrow$ 64 distinct proposition symbols, 155 sentences