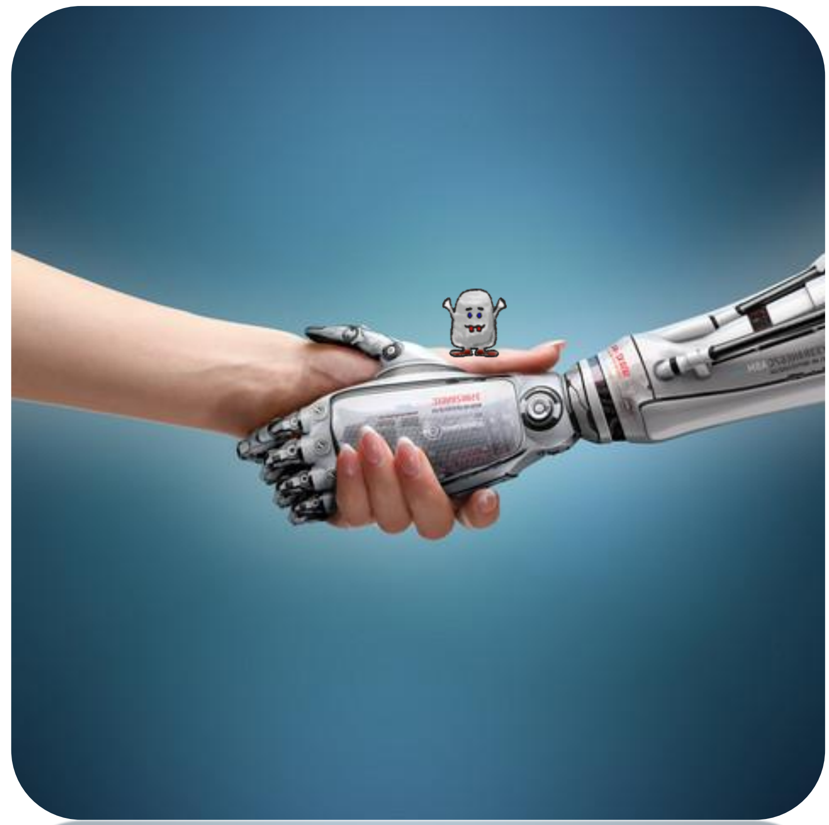
Zurich University
of Applied Sciences

Artificial Intelligence
V06a: Knowledge, reasoning & logic

Knowledge representation with logic
From propositional to first-order logic



Based on material by
- Stuart Russell, UC Berkeley
- Kevin Leyton-Brown, U British Columbia

Zurich University of Applied Sciences and Arts
InIT Institute of Applied Information Technology (stdm)

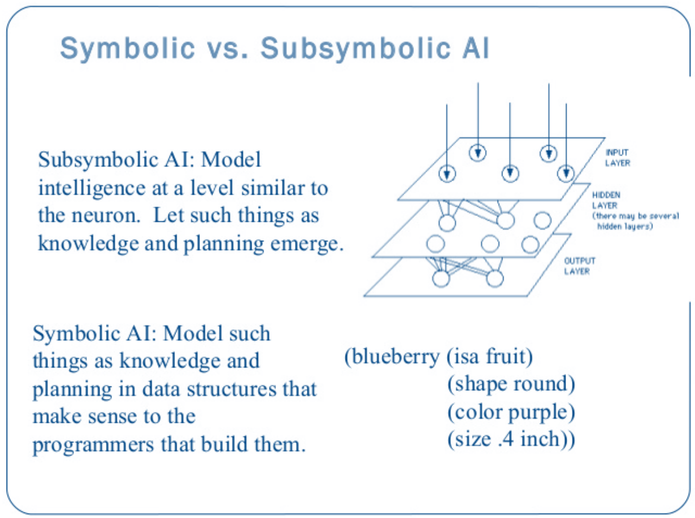
Educational objectives

- **Remember** the **syntax** & **semantics** of predicate- and first-order **logic**
- **Explain** **how** the **representation of knowledge** with formal languages (e.g., logic) **facilitates reasoning**
- **Solve** logic **exercises** using pen & paper

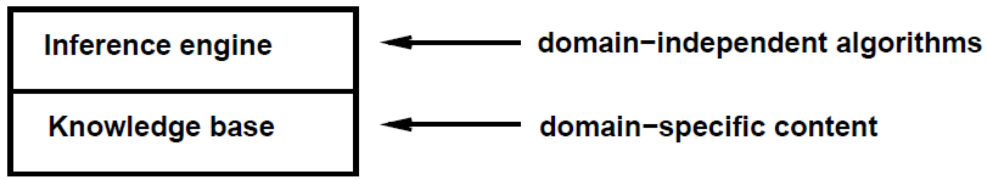
“In which we design agents that can form representations of the world, use a process of inference to derive new representations about the world, and use these new representations to deduce what to do.”

➔ Reading: AIMA, ch. 7 (ch. 7-7.3; 7.7; 8-8.1 covered here)
(ch. 7.4; 7.8; 8.2 is related material)

1. KNOWLEDGE REPRESENTATION WITH LOGIC



Knowledge bases



Knowledge base (**KB**): a set of **sentences** in a **formal language**

- The 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)
- **Two views** of an agent (regardless of approach)
  → At the **knowledge level**: i.e., what they know, regardless of how implemented
  → At the **implementation level**: i.e., data structures in KB and algorithms manipulating them

```
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
```

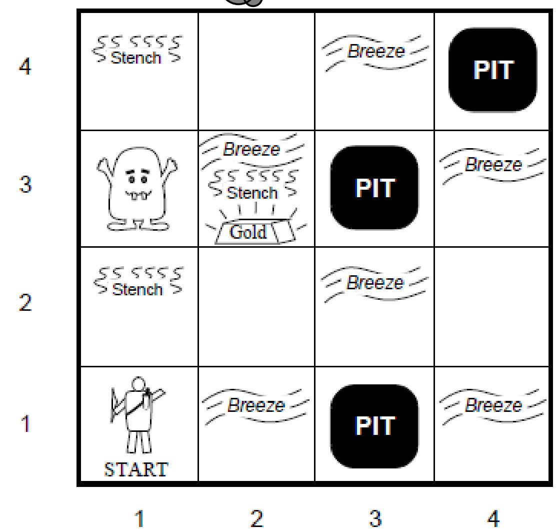
The agent must be able to

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

Example: The wumpus world
PEAS description

Performance measure
- gold $+1000$, death -1000, -1 per step, -10 for using 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



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

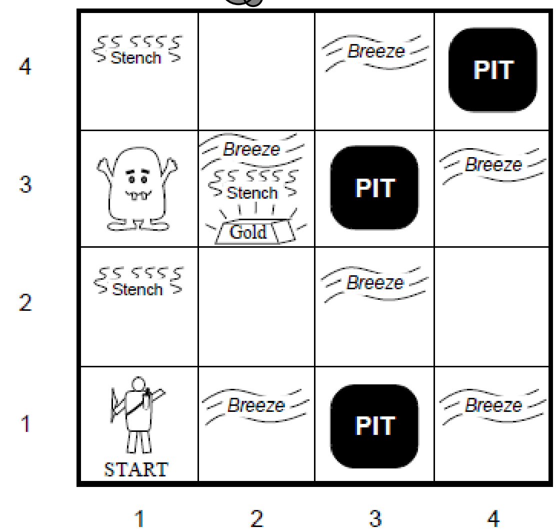
Sensors
- Breeze, Glitter, Smell

Wumpus world characterization
- Observable?
- Deterministic?
- Episodic?
- Static?
- Discrete?
- Single-agent?

Zurich University of Applied Sciences

Example: The wumpus world

PEAS description



Performance measure

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

Actuators

- Left turn, Right turn,
- Forward, Grab, Release, Shoot

Sensors

- Breeze, Glitter, Smell

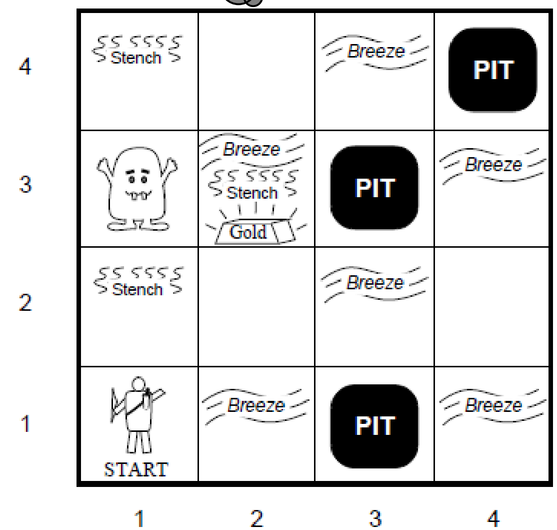
Wumpus world characterization

- Observable? No (only local perception)
- Deterministic?
- Episodic?
- Static?
- Discrete?
- Single-agent?

Example: The wumpus world
PEAS description

Performance measure
- gold $+1000$, death -1000, -1 per step, -10 for using 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



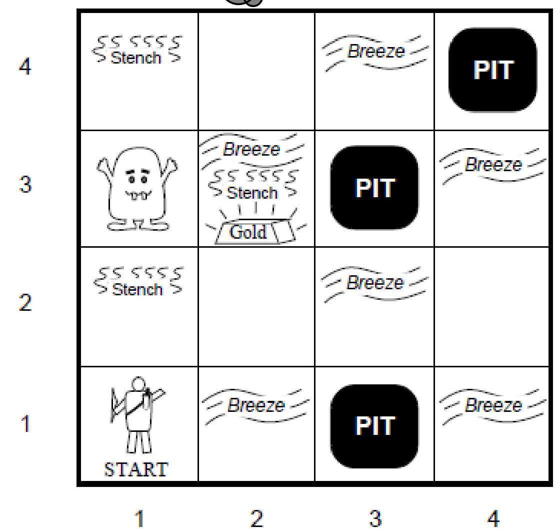
Actuators
- Left turn, Right turn,
- Forward, Grab, Release, Shoot

Sensors
- Breeze, Glitter, Smell

Wumpus world characterization
- Observable? No (only local perception)
- Deterministic? Yes (outcomes exactly specified)
- Episodic?
- Static?
- Discrete?
- Single-agent?

Example: The wumpus world
PEAS description

Performance measure
- gold $+1000$, death -1000, -1 per step, -10 for using 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

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

Sensors
- Breeze, Glitter, Smell

Wumpus world characterization
- Observable? No (only local perception)
- Deterministic? Yes (outcomes exactly specified)
- Episodic? No (sequential at the level of actions)
- Static?
- Discrete?
- Single-agent?

Example: The wumpus world

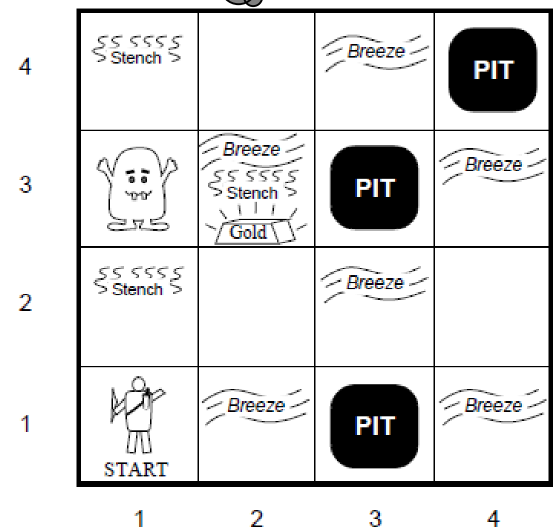
PEAS description

Performance measure
- gold $+1000$, death -1000, -1 per step, -10 for using 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

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

Sensors
- Breeze, Glitter, Smell



Wumpus world characterization
- Observable? No (only local perception)
- Deterministic? Yes (outcomes exactly specified)
- Episodic? No (sequential at the level of actions)
- Static? Yes (wumpus and pits do not move)
- Discrete?
- Single-agent?

Example: The wumpus world

PEAS description

Performance measure

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

Environment

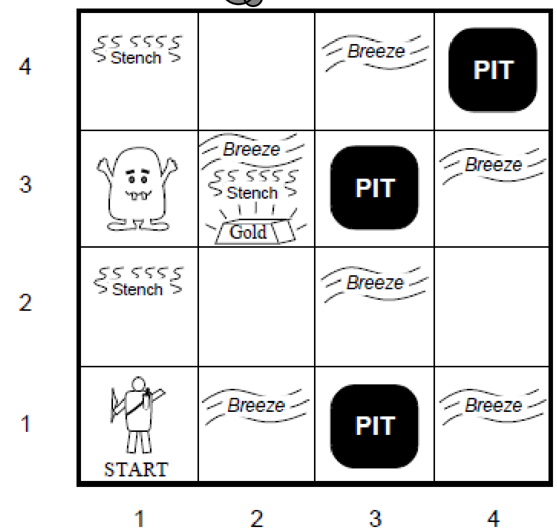
- Squares adjacent to **wumpus** are **smelly**
- Squares adjacent to **pit** are **breezy**
- **Glitter** iff **gold** is in the same square
- **Shooting** kills **wumpus** if you are facing it
- Shooting uses up the only **arrow**
- Grabbing picks up gold if in same square
- Releasing drops the gold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

Actuators

- Left turn, Right turn,
- Forward, Grab, Release, Shoot

Sensors

- Breeze, Glitter, Smell



Wumpus world characterization

- Observable? No (only local perception)
- Deterministic? Yes (outcomes exactly specified)
- Episodic? No (sequential at the level of actions)
- Static? Yes (wumpus and pits do not move)
- Discrete? Yes
- Single-agent?

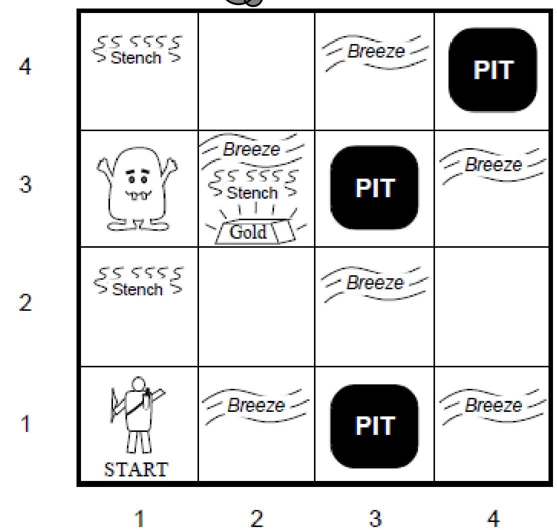
Example: The wumpus world
PEAS description

Performance measure
- gold $+1000$, death -1000, -1 per step, -10 for using 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

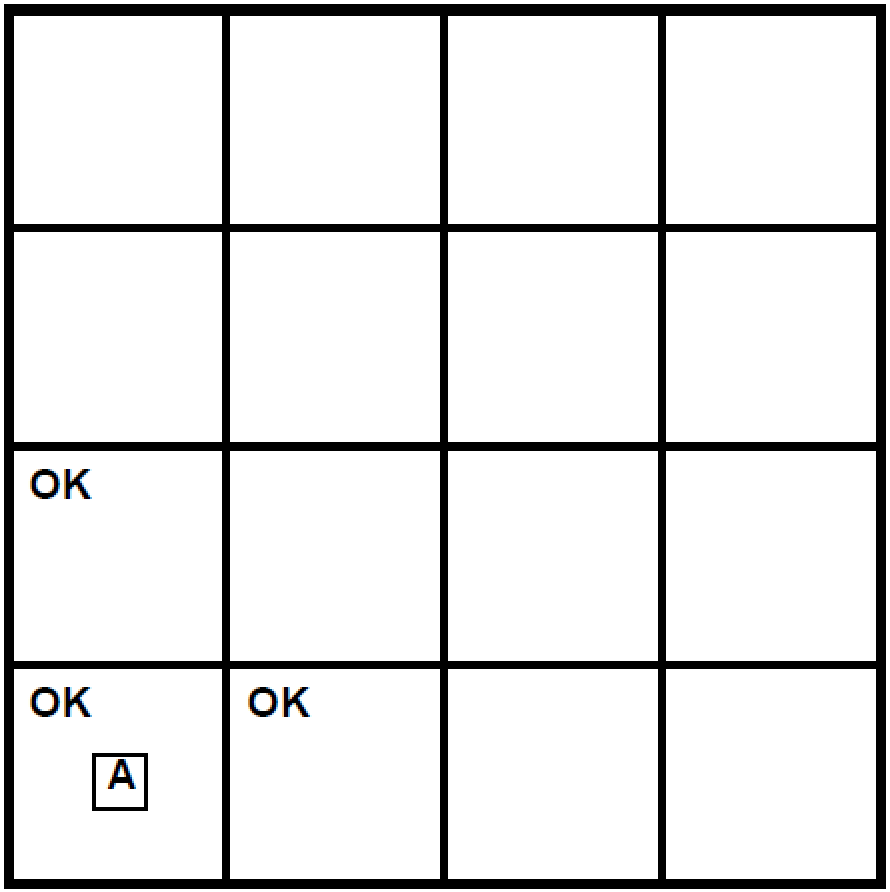
Actuators
- Left turn, Right turn,
- Forward, Grab, Release, Shoot

Sensors
- Breeze, Glitter, Smell



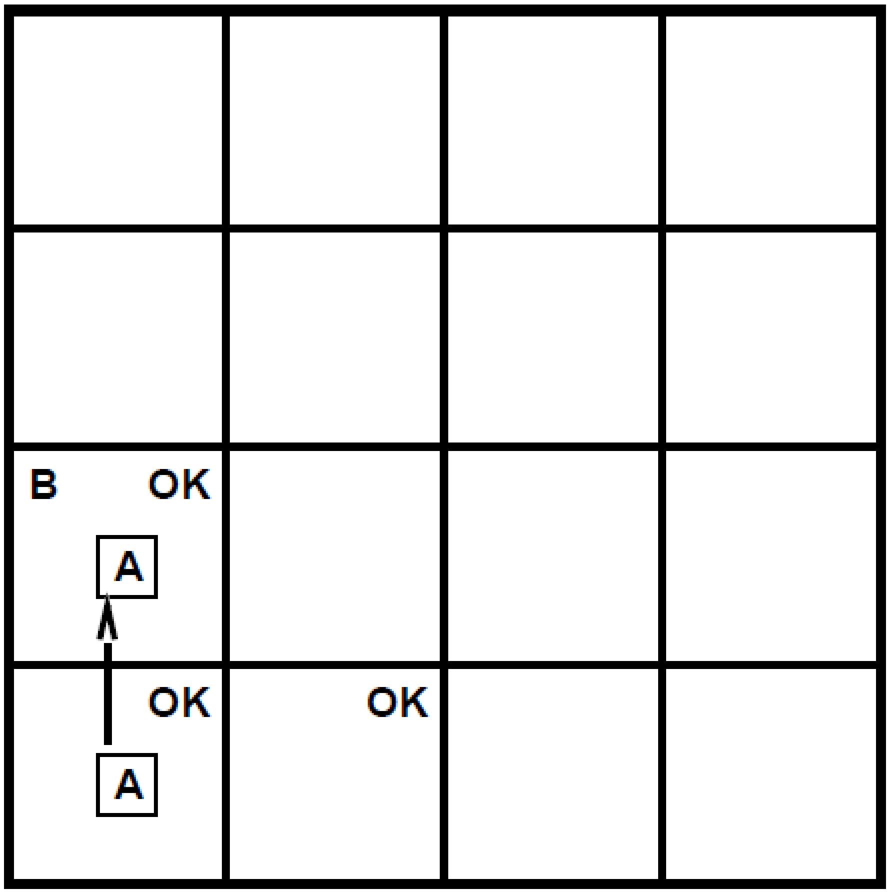
Wumpus world characterization
- Observable? No (only local perception)
- Deterministic? Yes (outcomes exactly specified)
- Episodic? No (sequential at the level of actions)
- Static? Yes (wumpus and pits do not move)
- Discrete? Yes
- Single-agent? Yes (wumpus is a natural feature)

Exploring a wumpus world



Start (square 1/1)

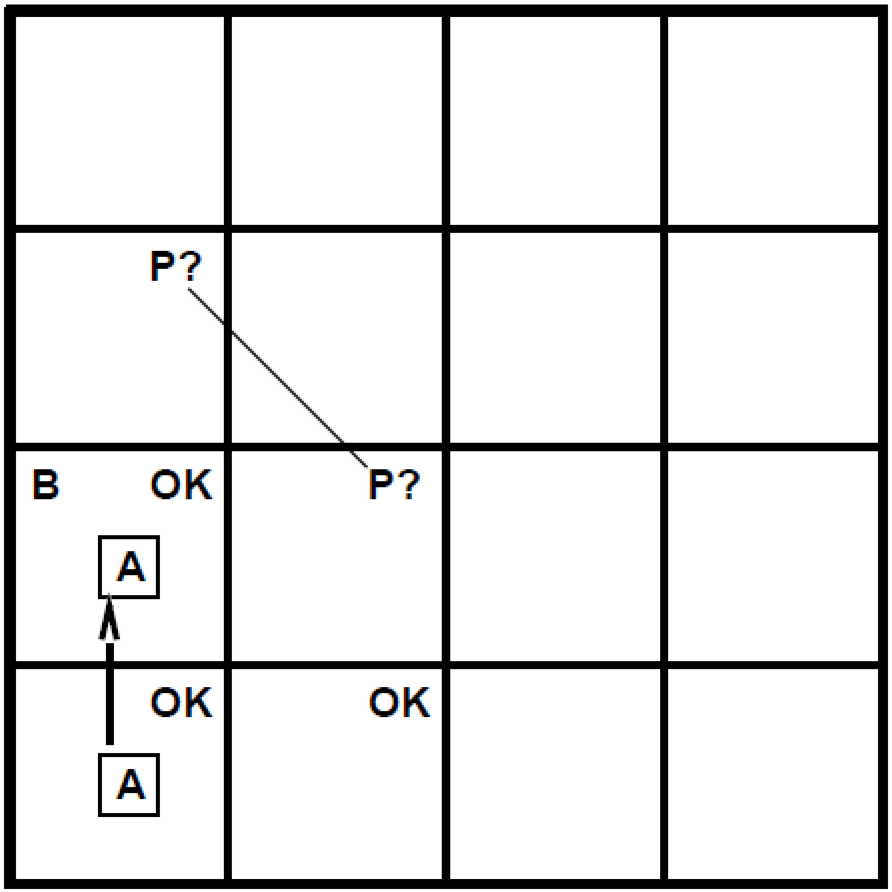
Exploring a wumpus world



Start (square 1/1)

- Move forward ➔ sense breeze

Exploring a wumpus world

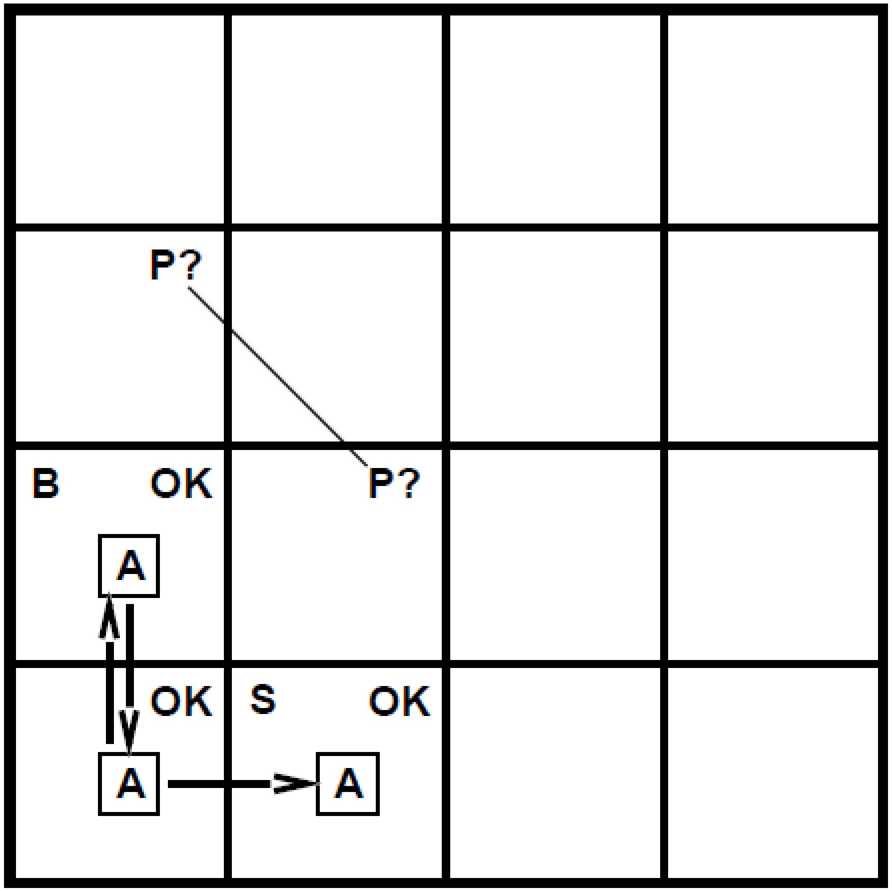


Start (square 1/1)

- Move forward ➔ sense breeze
- Infer possible pits (because of breeze)

“Squares adjacent to pit are breezy”

Exploring a wumpus world

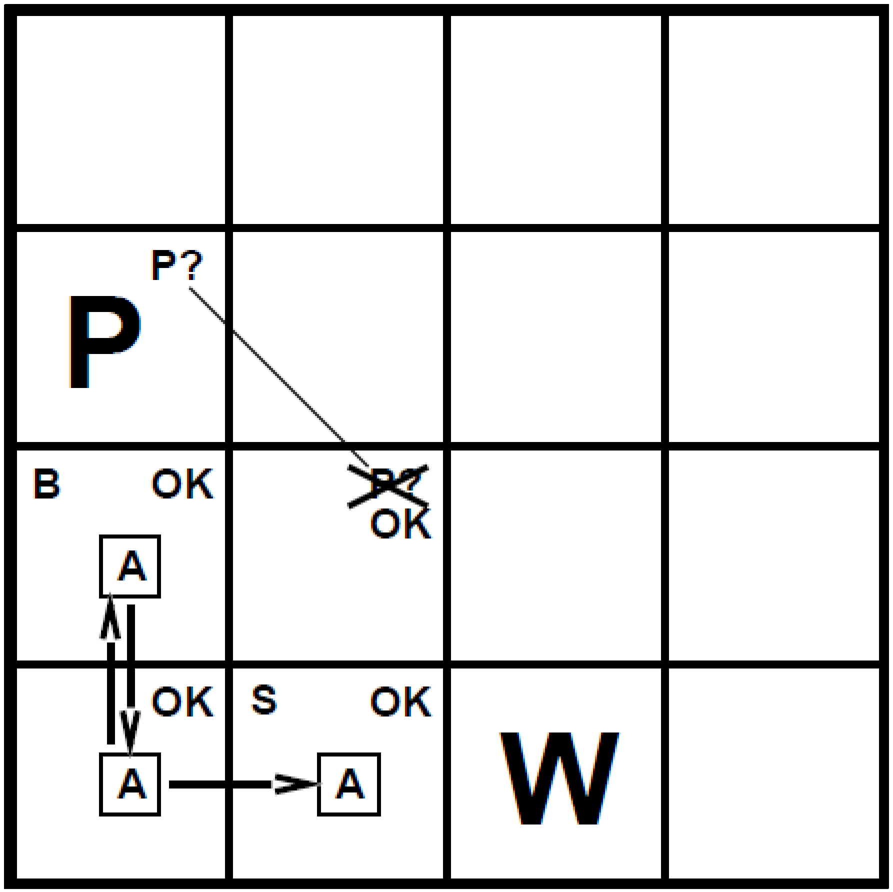


Start (square 1/1)

- Move forward ➔ sense breeze
- Infer possible pits (because of breeze)
- Move to 1/2 (row/col) ➔ sense stench

"Squares adjacent to pit are breezy"

Exploring a wumpus world

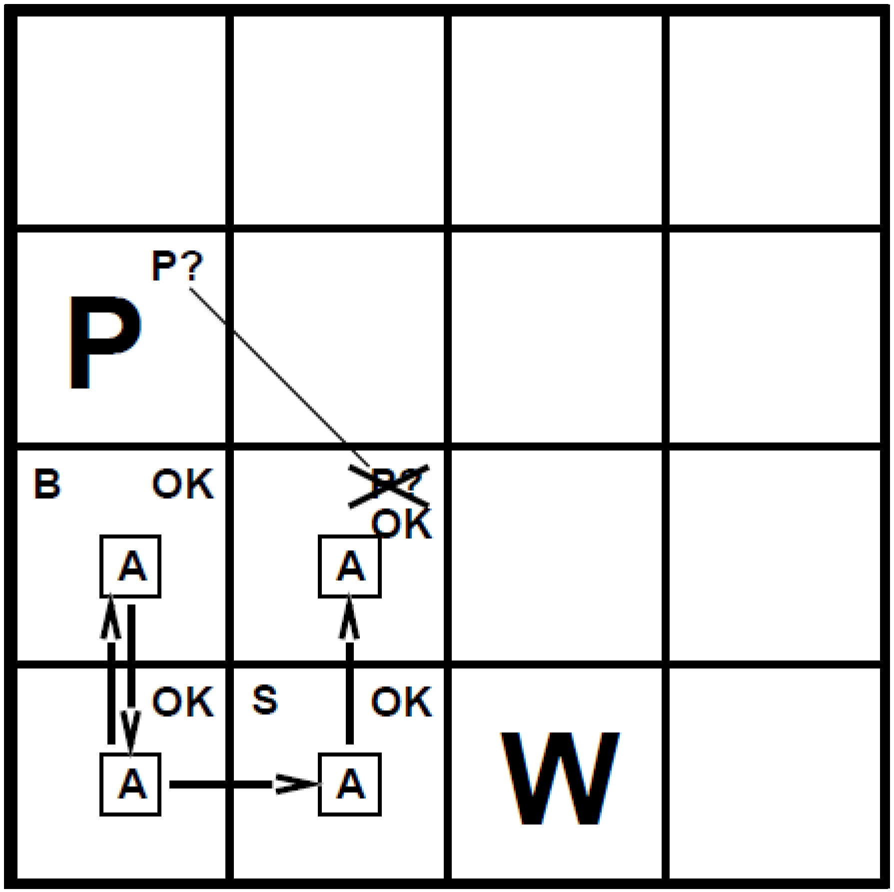


Start (square 1/1)

"*Squares adjacent to pit are breezy*"

- Move forward ➔ sense breeze
- Infer possible pits (because of breeze)
- Move to 1/2 (row/col) ➔ sense stench
- Infer pit, wumpus (and no pit in 2/2)

Exploring a wumpus world

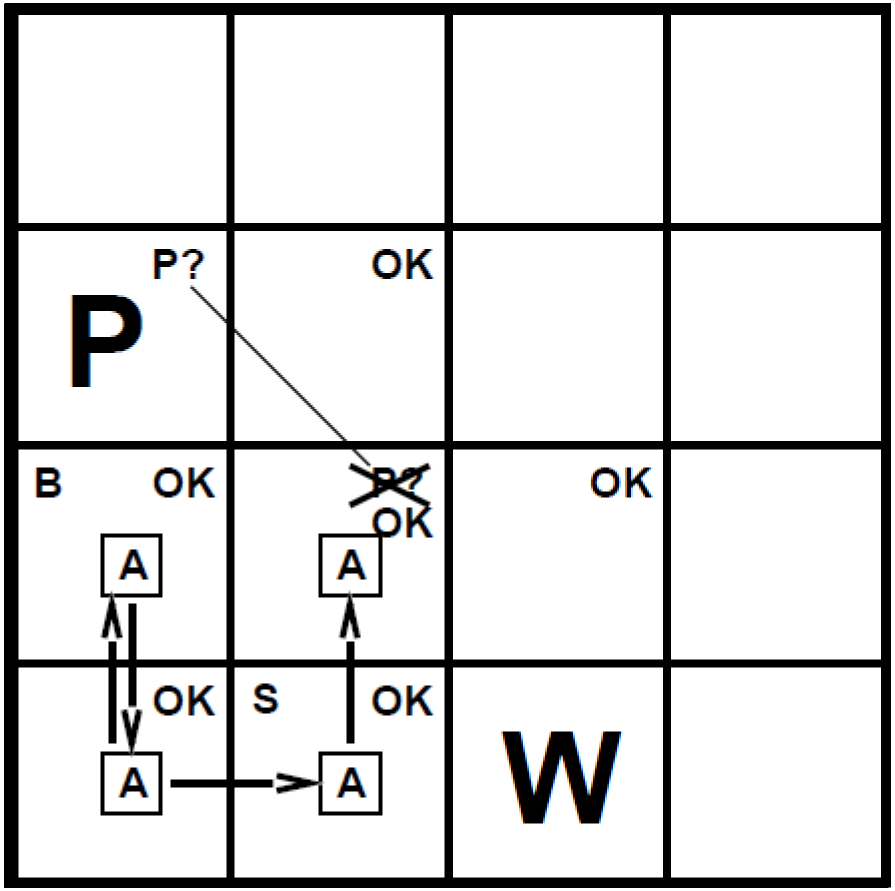


Start (square 1/1)

"Squares adjacent to pit are breezy"

- Move forward ➔ sense breeze
- Infer possible pits (because of breeze)
- Move to 1/2 (row/col) ➔ sense stench
- Infer pit, wumpus (and no pit in 2/2)
- Move to 2/2 (only save square)

Exploring a wumpus world

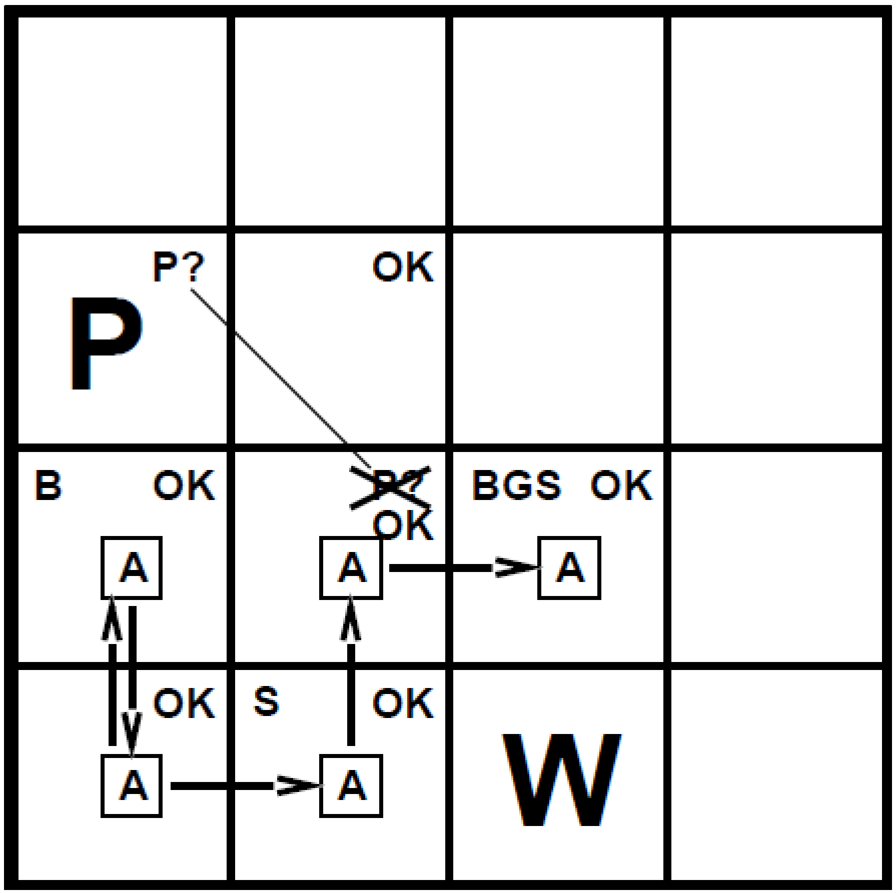


Start (square 1/1)

"*Squares adjacent to pit are breezy*"

- Move forward ➔ sense breeze
- Infer possible pits (because of breeze)
- Move to 1/2 (row/col) ➔ sense stench
- Infer pit, wumpus (and no pit in 2/2)
- Move to 2/2 (only save square)
- Sense nothing ➔ 2/3 and 3/2 are ok

Exploring a wumpus world



Start (square 1/1)

"Squares adjacent to pit are breezy"

- Move forward ➔ sense breeze
- Infer possible pits (because of breeze)
- Move to 1/2 (row/col) ➔ sense stench
- Infer pit, wumpus (and no pit in 2/2)
- Move to 2/2 (only save square)
- Sense nothing ➔ 2/3 and 3/2 are ok
- Move to 2/3 ➔ sense breeze, stench, glitter

➔ What next?

Logic in general

Logics are formal languages for representing information

- …such that **conclusions** can be drawn
- Syntax defines the “structure” of sentences in the language
- Semantics defines the “meaning" of sentences (i.e. **truth** of a sentence in a **world**/model)

Model: a formally structured possible world with respect to which truth can be evaluated.
➔ We say “m **is a model of** a sentence α” or “m **satisfies** α” if α is true in m
(i.e., m instantiates all variables in α such that α is true)

Example: the language of **arithmetic**

- $x + 2 \geq y$ is a sentence; $x2 + y >$ is not a sentence
- $x + 2 \geq y$ is true iff the number $x + 2$ is no less than the number y
- $x + 2 \geq y$ is true in a world where $x = 7; y = 1$
- $x + 2 \geq y$ is false in a world where $x = 0; y = 6$

Key concept 1: Entailment
(logical consequence, DE “semantische Implikation”)

$KB \vDash \alpha$

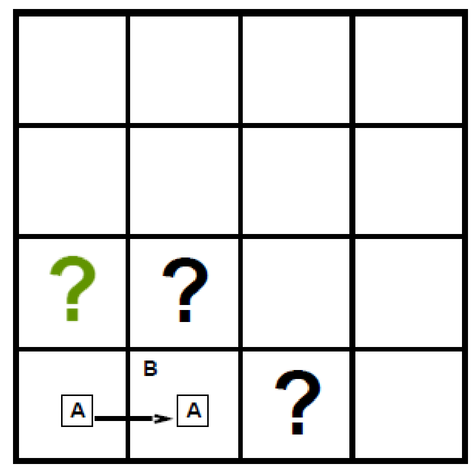
- Intuitively, entailment means that **one thing follows from another:** *“from KB I know that α”*
- Formally, *“knowledge base KB entails sentence α iff α is true in all worlds where KB is true”*
- Examples:
  - A KB containing *“FCZ won”* and *“YB won“* entails *“Either FCZ won or the Young Boys won“*
  - $x + y = 4$ entails $4 = x + y$

➔ Entailment is a **relationship between** sentences (i.e. **syntax**) that is **based on semantics**

Example: Entailment in the wumpus world

- Figure: the situation after detecting nothing in 1/1 → moving right → breeze
- Is α_1 *(“no pit in 2/1”) true, given the KB* (wumpus-world rules & percept)?

➔ See next slide



Entailment in the wumpus world, contd.

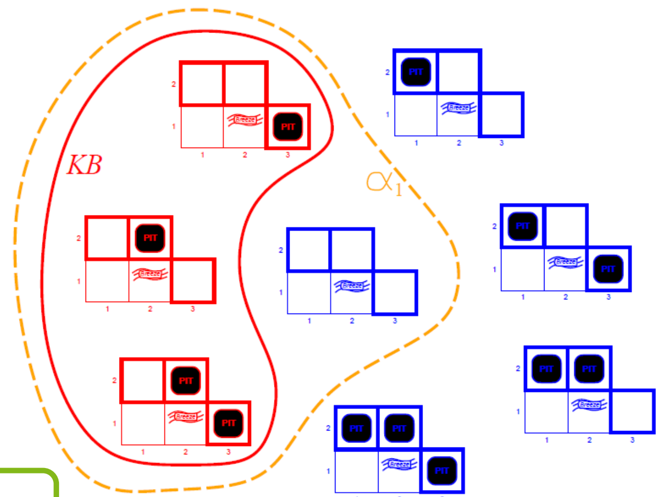
Consider possible models for all "?", assuming only pits
(3 Boolean choices give 2^3 possible models)

All possible models for the presence of pits in 2/1, 2/2 and 1/3:
- Red: all models compliant with the KB
  (KB := wumpus-world rules + percepts)
- Blue: all models where KB is false
- Yellow: all models for sentence
  α_1 =*"no pit in 2/1"*

➔ $KB \vDash \alpha_1$
(the sentence, not all possible models for it!)

➔ i.e., KB is a stronger assertion than α_1
(ruling out more possible worlds/models)

Formal: $\alpha \vDash \beta \ iif \ M(\alpha) \subseteq M(\beta)$
(with $M(\alpha)$ being the set of all models of α)



Possible models for KB $\subseteq$ *possible models for* α_1

➔ $KB \vDash \alpha_1$

Key concept 2: Inference

$KB \vDash_i \alpha$

- Meaning: sentence α can be derived from KB **by procedure i**
- Intuition:
  - Consequences of KB are a haystack
  - α is a needle
  - Entailment says: *“needle is in haystack”*
  - Inference: finding it

Desirable properties of i

- Soundness: i is sound if whenever $KB \vDash_i \alpha$, it is also true that $KB \vDash \alpha$
- Completeness: i is complete if whenever $KB \vDash \alpha$, it is also true that $KB \vDash_i \alpha$

Preview

- We’ll define: a logic (first-order definite clauses) expressive enough to say almost anything storable in a RDBMS, and a sound and complete inference procedure (forward chaining)
- That is: The procedure will **answer any question** whose answer follows from what is known by KB

2. FROM PROPOSITIONAL TO FIRST-ORDER LOGIC

Propositional logic
(DE “Aussagenlogik”)

Reasoning over (unrelated) **facts**
- The **simplest of all logics** to illustrate basic ideas

Syntax
- If S is a sentence, $\neg S$ is a sentence (negation)
- If S_1 and S_2 are sentences, $S_1 \wedge S_2$ is a sentence (conjunction, “and”)
- If S_1 and S_2 are sentences, $S_1 \vee S_2$ is a sentence (disjunction, “or”)
- If S_1 and S_2 are sentences, $S_1 \Rightarrow S_2$ is a sentence (implication)
- If S_1 and S_2 are sentences, $S_1 \Leftrightarrow S_2$ is a sentence (biconditional)

Semantics (**rules for evaluating truth** with respect to a model m)

$\neg S$	is true iff	S	is false		
$S_1 \wedge S_2$	is true iff	S_1	is true **and**	S_2	is true
$S_1 \vee S_2$	is true iff	S_1	is true **or**	S_2	is true
$S_1 \Rightarrow S_2$	is false iff	S_1	is true **and**	S_2	is false
$S_1 \Leftrightarrow S_2$	is true iff	$S_1 \Rightarrow S_2$	is true **and**	$S_2 \Rightarrow S_1$	is true

iff

The logical implication $S_1 \Rightarrow S_2$ (a.k.a. rule: “S_2 *if* S_1 *is true*”) shows paradox behavior when interpreted in a colloquial way:
- *“**if** I teach AI **then** the earth is a sphere”* is formally true regardless of meaning.

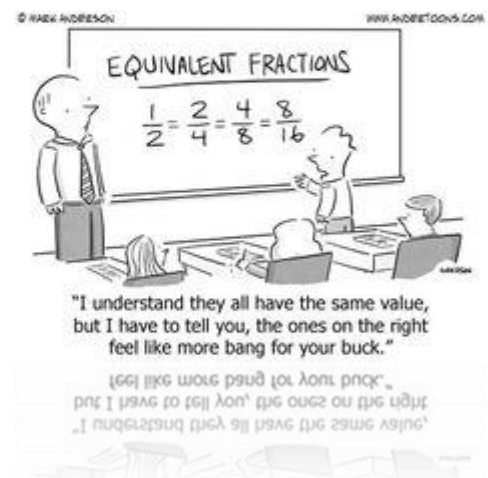
But the definition makes sense:
- *“**if** it is raining **then** the street gets wet”* has to be true (as a rule) regardless of if it is raining (there might be other reasons for a wet street).

See it as if saying *“**if** S1 is true **then** I claim S2 to be true as well; **else**, I make no claim”*.

Logical equivalence
i.e., rules to manipulate sentences of logic

Two sentences are logically equivalent iff true in same models:
- $\alpha \equiv \beta$ if and only if $\alpha \vDash \beta$ and $\beta \vDash \alpha$

$(\alpha \land \beta) \equiv (\beta \land \alpha)$	commutativity of ∧
$(\alpha \lor \beta) \equiv (\beta \lor \alpha)$	commutativity of ∨
$((\alpha \land \beta) \land \gamma) \equiv (\alpha \land (\beta \land \gamma))$	associativity of ∧
$((\alpha \lor \beta) \lor \gamma) \equiv (\alpha \lor (\beta \lor \gamma))$	associativity of ∨
$\neg(\neg\alpha) \equiv \alpha$	double-negation elimination
$(\alpha \Rightarrow \beta) \equiv (\neg\beta \Rightarrow \neg\alpha)$	contraposition
$(\alpha \Rightarrow \beta) \equiv (\neg\alpha \lor \beta)$	implication elimination
$(\alpha \Leftrightarrow \beta) \equiv ((\alpha \Rightarrow \beta) \land (\beta \Rightarrow \alpha))$	biconditional elimination
$\neg(\alpha \land \beta) \equiv (\neg\alpha \lor \neg\beta)$	De Morgan
$\neg(\alpha \lor \beta) \equiv (\neg\alpha \land \neg\beta)$	De Morgan
$(\alpha \land (\beta \lor \gamma)) \equiv ((\alpha \land \beta) \lor (\alpha \land \gamma))$	distributivity of ∧ over ∨
$(\alpha \lor (\beta \land \gamma)) \equiv ((\alpha \lor \beta) \land (\alpha \lor \gamma))$	distributivity of ∧ over ∨

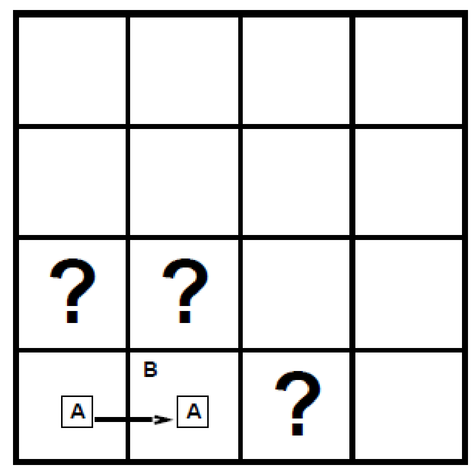


Example: Wumpus world sentences
How logic serves well as a representation language

Notation
- 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

Facts: representing factual knowledge
- $\neg P_{1,1}$
- $\neg B_{1,1}$ For concrete locations – no variables!
- $B_{1,2}$

Rules: representing procedural knowledge
- *“Pits cause breezes in adjacent squares“,*
  → Example for concrete squares: $B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$
- *“A square is breezy if and only if there is an adjacent pit"*
  → Example for concrete squares: $B_{1,2} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{1,3}),\ \ldots$



First-order logic
(FOL, DE "Prädikatenlogik 1. Stufe")

Only in higher-order logics do predicates have other predicates (or functions) as parameters

Pros and cons of propositional logic (as compared to atomic knowledge representation)

- Declarative: pieces of syntax correspond to facts
- Allows partial/disjunctive/negated information (unlike most data structures and databases)
- Compositional: meaning of $B_{1,1} \wedge P_{1,2}$ is derived from meaning of $B_{1,1}$ and of $P_{1,2}$
- Meaning is context-independent (unlike natural language, where meaning depends on context)
- Very limited expressive power (unlike natural language)
  → E.g., cannot say *"pits cause breezes in adjacent squares"* except by **one sentence for each square**!
  → It is useful to view the world as consisting of objects and relationships between them

Much greater expressiveness of FOL (like natural language)

- **Quantifiable variables** over non-logical objects (quantifiers ∀, ∃, ∄)
- **Objects**: people, houses, numbers, theories, Ronald McDonald, colors, soccer matches, wars, centuries, …

  Assert that the relationship exists
- **Relations** (predicates): red, round, bogus, prime, multistoried, brother of, bigger than, inside, part of, has color, occurred after, owns, comes between, …
  - **Functions**: father of, best friend, third inning of, one more than, end of, …

  A function is a relation **with only one "value"** for any given "parameter"/input

Example: Wumpus world sentences in FOL
How logic serves well as a representation language

We keep track of time/situation via quantification over t; in propositional logic, we would need copies of each sentence for each time step.

Perception

- Timestep t is smelly if I perceive a Smell (and whatever else) at t
  $\forall b, g, t \ Percept([Smell, b, g], t) \Rightarrow Smell(t)$
- If I perceive a Glitter at t, I am at the place of the gold
  $\forall s, b, t \ Percept([s, b, Glitter], t) \Rightarrow AtGold(t)$

Reflex

- $\forall t \ AtGold(t) \Rightarrow Action(Grab, t)$

Deduction of hidden properties of a location

- $\forall x, t \ At(Agent, x, t) \land Smell(t) \Rightarrow Smelly(x)$
- A diagnostic rule: $\forall y \ Breezy(y) \Rightarrow \exists x \ Pit(x) \land Adjacent(x, y)$
- A causal rule: $\forall x, y \ Pit(x) \land Adjacent(x, y) \Rightarrow Breezy(y)$
- Definition of the Breezy predicate: $\forall y \ Breezy(y) \Leftrightarrow [\exists x \ Pit(x) \land Adjacent(x, y)]$

Exercise: Pen & paper logic
→ see P03b

Following Russell & Norvig's advice that *"a student of AI must develop a talent for working with logical notation"* [AIMA, p. 290], this is to get you acquainted with formulating and manipulating known facts in logical notation, and to do inference to arrive at new conclusions.

Get together in teams of 2-3 and collectively solve the following exercises from P03b using pen, paper and the previous slides. Distribute the work amongst your group and make sure to explain each result to every group member.

- 1.1 – truth of sentences in propositional logic
- 1.2 – validity & satisfiability in propositional logic
- 1.3 – entailment in the wumpus world
- 2.1 – formulating sentences in first-order logic
- 3.1 – inference in first-order logic

Prepare to explain your findings to the class.

Building a hybrid agent for the wumpus world
Combining (propositional) logic and problem-solving search

1. Start with an **initial KB of atemporal** knowledge (e.g. relating breeziness to presence of pits)

2. At each time step, **add current percept** and temporal axioms (e.g., successor-state axioms)

   State what happens to each fluent (i.e., any aspect of the world that changes) depending on what action is taken

3. Construct a **plan** based on a **hard-coded order of goals** (in decreasing priority):
   a. If glitter: grab the gold → plan a route to initial location → get out of cave
   b. If no such plan: **find save route** to closest unvisited save square **using A***
   c. If no safe square, but arrow: make a safe square by shooting at a possible wumpus location (**determined by asking** $ASK(KB, \neg W_{x,y}) == false$) after going there
   d. If killing wumpus fails: find a square that is not provably unsafe ($ASK(KB, \neg OK^t{}_{x,y}) == false$) and plan to go there
   e. If no such square: mission impossible, find route to get out of cave

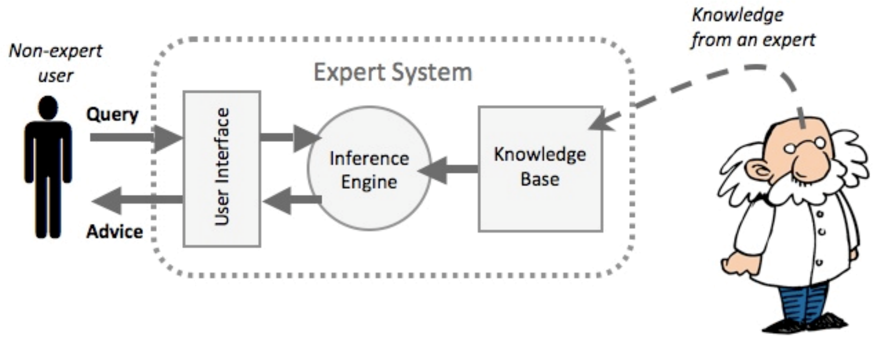
All route planning/finding in the hybrid agent is done using A*

→ See AIMA ch. 7.7.2 and Fig. 7.20

More on logic
→ see remarks on knowledge representation in the appendix

«Just as a student of physics requires some familiarity with mathematics, a ***student of AI*** *must* ***develop*** *a* ***talent for*** *working with* ***logical notation****» (AIMA, p. 290)*

On the importance of logic in AI
- For many years, systems based on **logic dominated AI** (research & successful practice)
- Example applications:
  - Expert systems (e.g. in health & medicine)
  - NASA **spacecraft control** (planning of action sequences, recovery from failures)
  - Electronic **circuit checking** (does it perform the task it is designed for?) and synthesis
  - Automatic **theorem proving**
- They are **still** broadly applied **today** (e.g. in deductive languages like SQL)
- A relevant subfield is logic programming (e.g., Prolog)



Further reading
- Rege, *«Logik Programmierung 1&2» in «Programmiersprachen und -Paradigmen»*
- AIMA ch. 7.4-7.6; 8.2-8.3; 9

Where's the intelligence?
Man vs. machine

- **Logical reasoning** as a higher-order cognitive process **is also applied by humans**

- The undertaking of **reducing intelligent** (human) behavior **to logic has failed**
  → either because of **expressiveness** of the language, or because of computational **intractability**
  → it is doubtful if all reasoning in humans can be reduced to logic, too

- Expert systems based on domain ontologies are **still helpful in** very **specific domains**
- As with humans, symbolic (i.e., logical) reasoning might be a higher-order process **on top of** subsymbolic learning (i.e., machine learning)

Review

- Logical **agents apply inference** to a knowledge base **to derive new information** and make decisions

- **Basic concepts** of logic:
  - **syntax**: formal structure of sentences
  - **semantics**: truth of sentences w.r.t. models
  - **entailment**: necessary truth of one sentence given another
  - **inference**: deriving sentences from other sentences
  - **soundness**: derivations produce only entailed sentences
  - **completeness**: derivations can produce all entailed sentences

- Wumpus world requires the **ability to represent partial and negated information**, reason by cases, etc.

Zurich University
of Applied Sciences

APPENDIX

Building a purely logical agent
Example using propositional logic

1. **Construct a sentence** that includes
   a. $Init^0$: collection of assertions about the initial state
   b. $Transition^1, \ldots, Transition^t$: successor-state axioms for all possible actions at each time up to t
   c. $HaveGold^t \wedge ClimbedOut^t$: the assertion that the goal is reached at time t

2. **Solve with a SAT solver** (see appendix)

3. If satisfiable: extract all **variables representing actions** that are assigned **true** in the model
   ➔ this is the plan

➔ See AIMA ch. 7.7.4

Satisfiability and the SAT problem

A sentence is satisfiable if it is **true in some model**

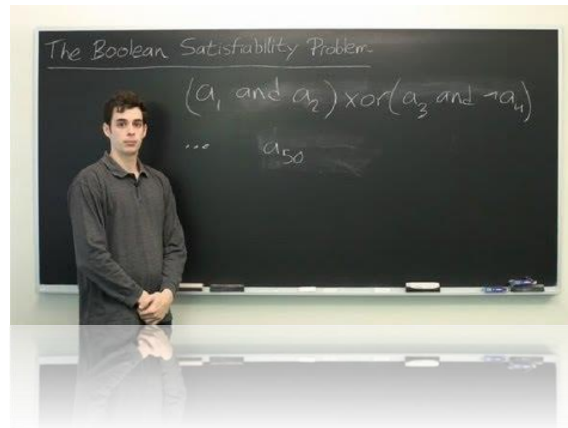
- Examples: $A \lor B$, C

A sentence is unsatisfiable if it is **true in no model**s

- Examples: $A \land \neg A$

Satisfiability is connected to inference via the following:

- $KB \vDash \alpha$ if and only if $(KB \land \neg\alpha)$ is unsatisfiable
  i.e., SAT is used to prove α by reductio ad absurdum



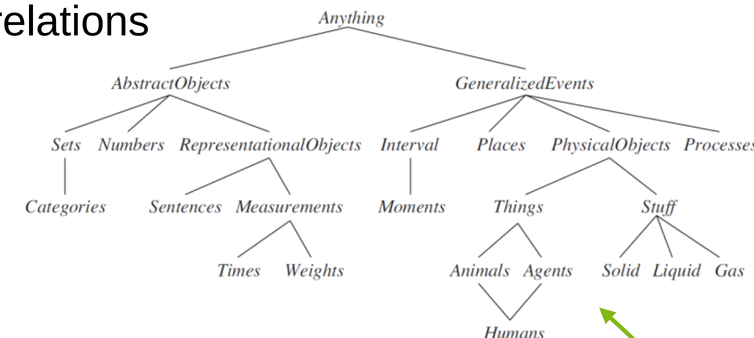
The SAT problem

- Deciding if a sentence in propositional logic is satisfiable (SAT) is the prototypical NP-complete problem (→ see appendix of V03)
- Many computer science problems can be reduced to SAT (e.g., all CSPs of V05)
  ➔ SAT plays in important role in the literature of AI / complexity theory / computer science in general

Practical considerations in building a KB

Building a representation and reasoning system

1. **Begin** with a task **domain**
2. **Decide** on which **objects** (individuals) you want to talk about
   (includes determining correspondence between symbols in the computer and objects/relations in world)
3. **Determine** what **relationships** (predicate symbols) you want to represent
   (includes determining which sentences will be true and which will be false)
4. **Choose symbols** in the computer to denote objects and relations
   (includes deciding which constant denotes which individual)
5. **Tell** the system **knowledge** about the domain (see below)
6. **Ask** the system **questions**



Ontological engineering

- It is unclear (computationally and philosophically) if special-purpose ontologies can be merged into a general-purpose one
- But: Using upper ontologies («world knowledge») connected to task-specific ones is a way that works for many domains (e.g., for a web shopping agent, see AIMA ch. 12.7)
- OWL, the W3C-standardized description logic, is very expressive but still seldom used

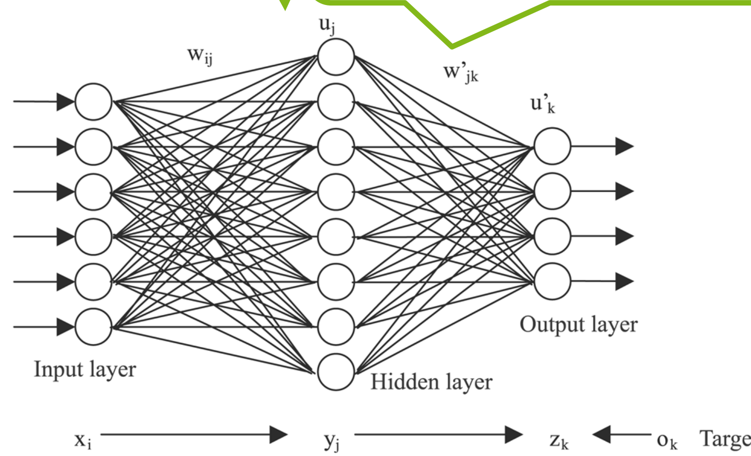
More on knowledge representation (KR)

Remarks

- KR as presented here is **most helpful today** in the context of **ontologies** (expert systems, semantic web)
- There are alternative forms of KR besides formal (deductive) languages, e.g.:
  - Procedural languages (e.g., Python code)
  - Subsymbolic representations (e.g., the weights in a neural network)
  - AI planning uses KR and reasoning in a less formal way (→ see V07)

A neural network encodes all its knowledge (i.e., all it has learned about data) in the weights $w_{i,j}$ and $w'_{j,k}$

The OWL-DL part of the web ontology language refers to a decidable subset of FOL → queries will eventually be answered



Further reading

- AIMA ch. 8.4; 12

Other types of knowledge & their representation
based on Z. Alvi, VU *"Artificial Intelligence CS607"*, lecture 14

- **Procedural knowledge**: Describes **how to do things**, provides a set of directions of how to perform certain tasks, e.g., how to drive a car
- **Declarative knowledge**: It describes **objects, rather than processes**. What is known about a situation, e.g. it is sunny today, cherries are red
- **Meta knowledge**: **Knowledge about knowledge**, e.g., the knowledge that blood pressure is more important for diagnosing a medical condition than eye color.
- **Heuristic knowledge**: (Empirical) **rule-of-thumb**, e.g. if I start seeing shops, I am close to a market
- **Structural knowledge**: Describes **structures and their relationships**, e.g. how the various parts of the car fit together to make a car, or knowledge structures in terms of concepts, sub concepts, and objects

Representations

- **Pictures and symbols**: This is how the earliest humans represented knowledge when sophisticated linguistic systems had not yet evolved
- **Graphs and Networks**:
  → allow **relationships** between entities, e.g., to show family relationships, now we can use a graph
  → May be used to represent procedural knowledge, e.g. how to start a car?
- **Numbers**: Eventually, every representation we use gets translated to numbers in the computers internal representation