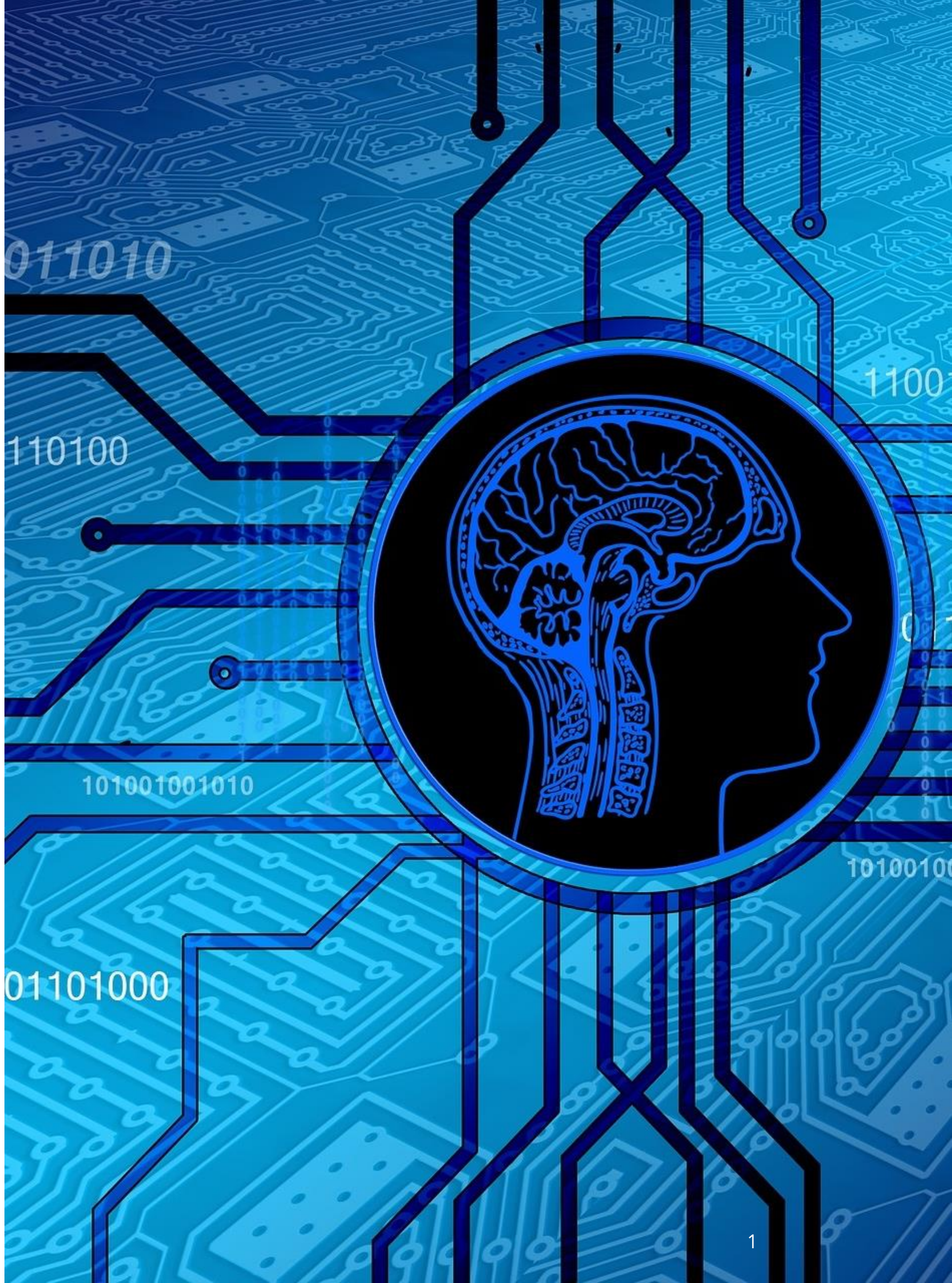
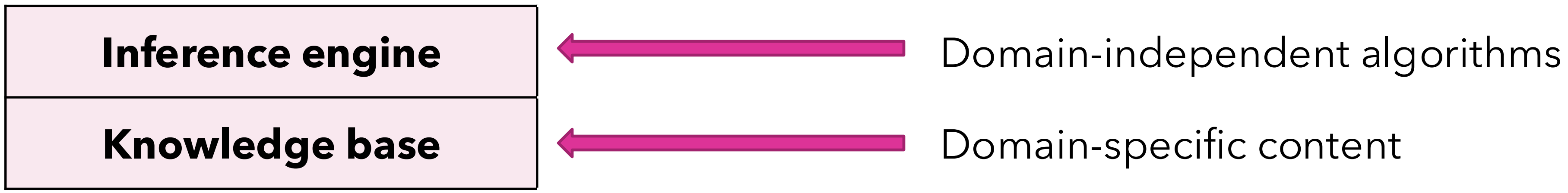
Logical Agents

Informatics 2D: Reasoning and Agents
Lecture 9



Knowledge bases



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

Declarative approach to building an agent (or other system):

- `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

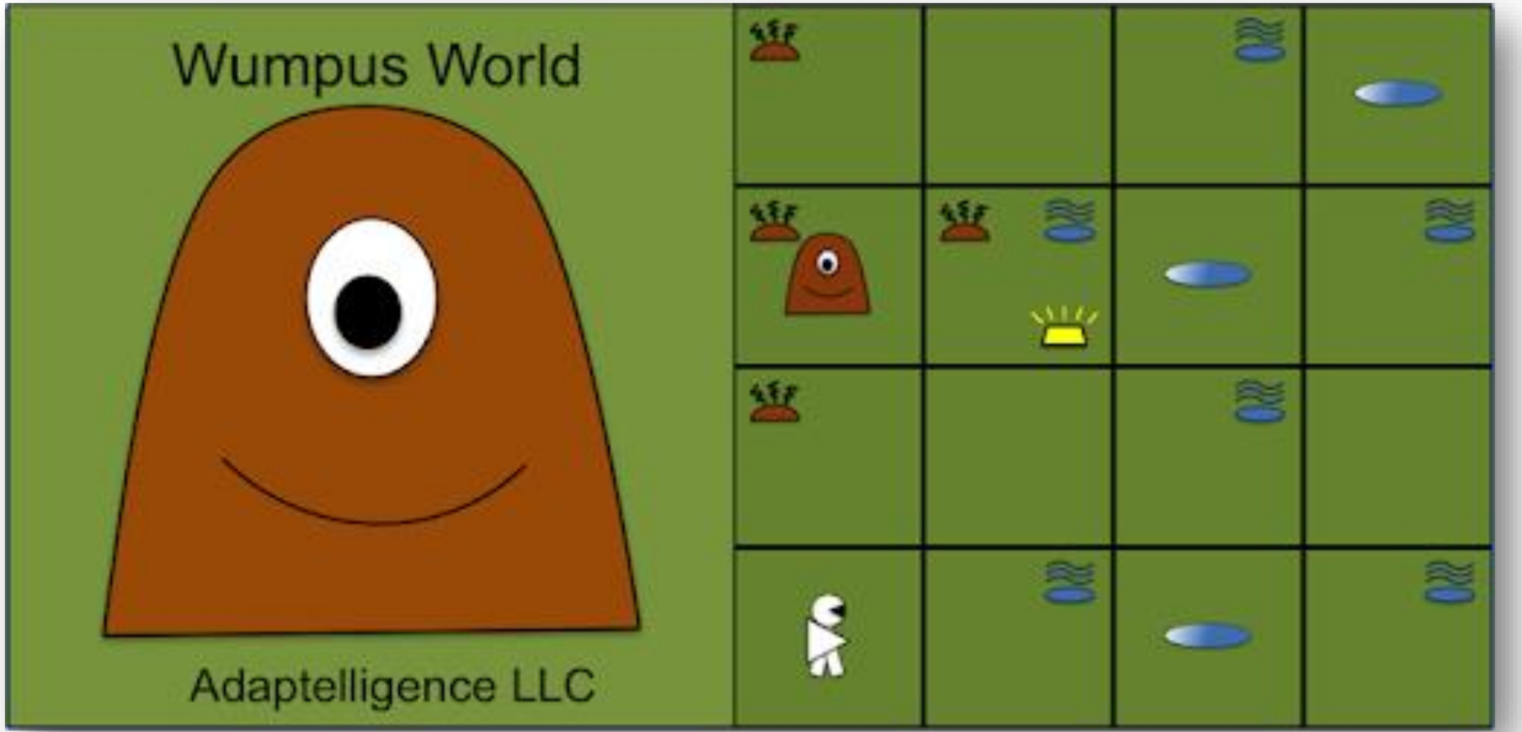
i.e., what they know, regardless of how implemented

A simple knowledge-based agent

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

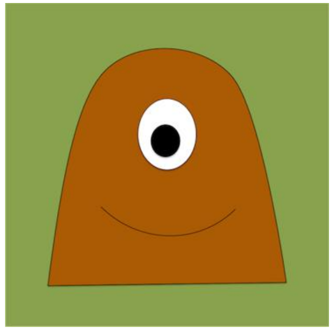
```
function KB-AGENT(percept) returns an action
  persistent: KB, a knowledge base
              t, a counter, initially 0, indicating time

  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
```



Wumpus World

https://en.wikipedia.org/wiki/Hunt_the_Wumpus

Wumpus World

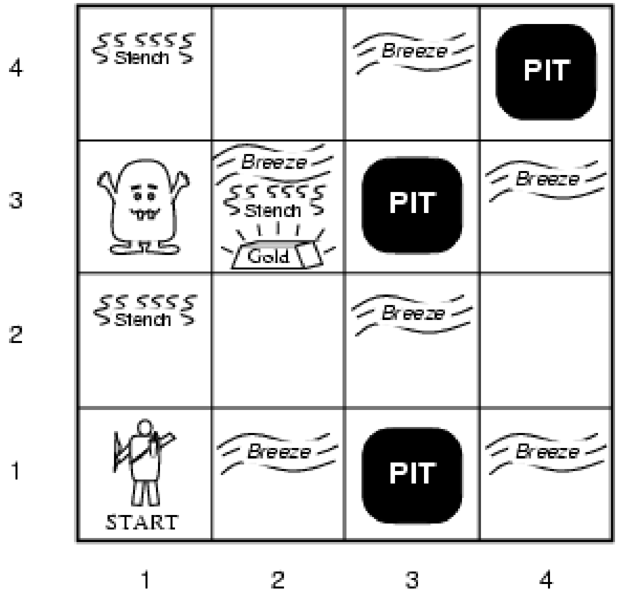


Performance measure
- Climb with the gold +1000, death –1000, -1 per step, -10 for using the arrow

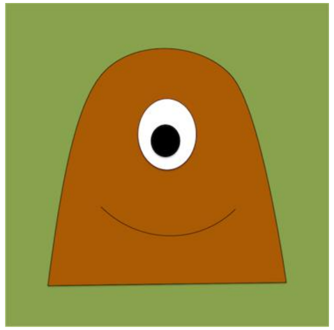
Actuators: Left turn, Right turn, Forward, Grab, Shoot, Climb

Environment: 4x4 grid, agent starts in [1,1]

Sensors: Stench, Breeze, Glitter, Bump, Scream
- Squares adjacent to wumpus are smelly
- Squares adjacent to pits are breezy
- Glitter iff gold is in the same square
- When the agent walks into a wall, it will perceive bump
- When the wumpus is killed, it will scream

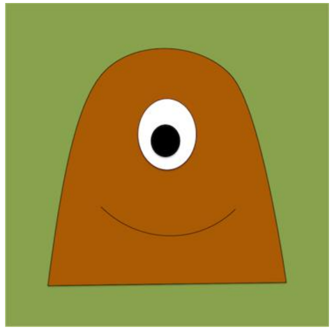


Wumpus World Environment Characterization



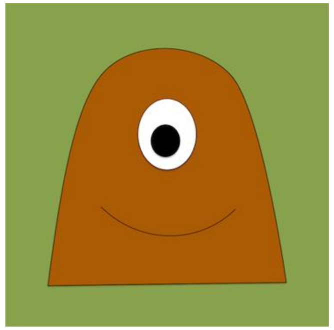
Observable	
Deterministic	
Episodic	
Static	
Discrete	
Single-agent	

Wumpus World Environment Characterization



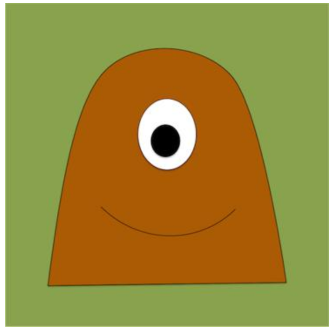
Observable	• No – only local perception
Deterministic	
Episodic	
Static	
Discrete	
Single-agent	

Wumpus World Environment Characterization



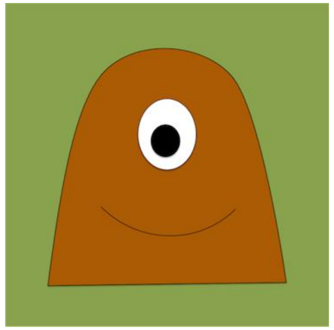
Observable	• No – only local perception
Deterministic	• Yes – outcomes exactly specified
Episodic	
Static	
Discrete	
Single-agent	

Wumpus World Environment Characterization



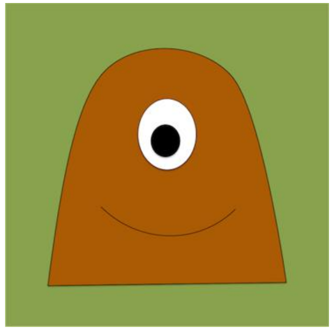
Observable	• No – only local perception
Deterministic	• Yes – outcomes exactly specified
Episodic	• No – sequential at the level of actions
Static	
Discrete	
Single-agent	

Wumpus World Environment Characterization



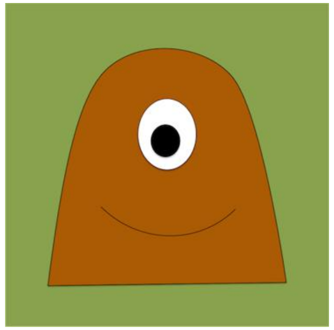
Observable	• No – only local perception
Deterministic	• Yes – outcomes exactly specified
Episodic	• No – sequential at the level of actions
Static	• Yes – Wumpus and Pits do not move
Discrete	
Single-agent	

Wumpus World Environment Characterization



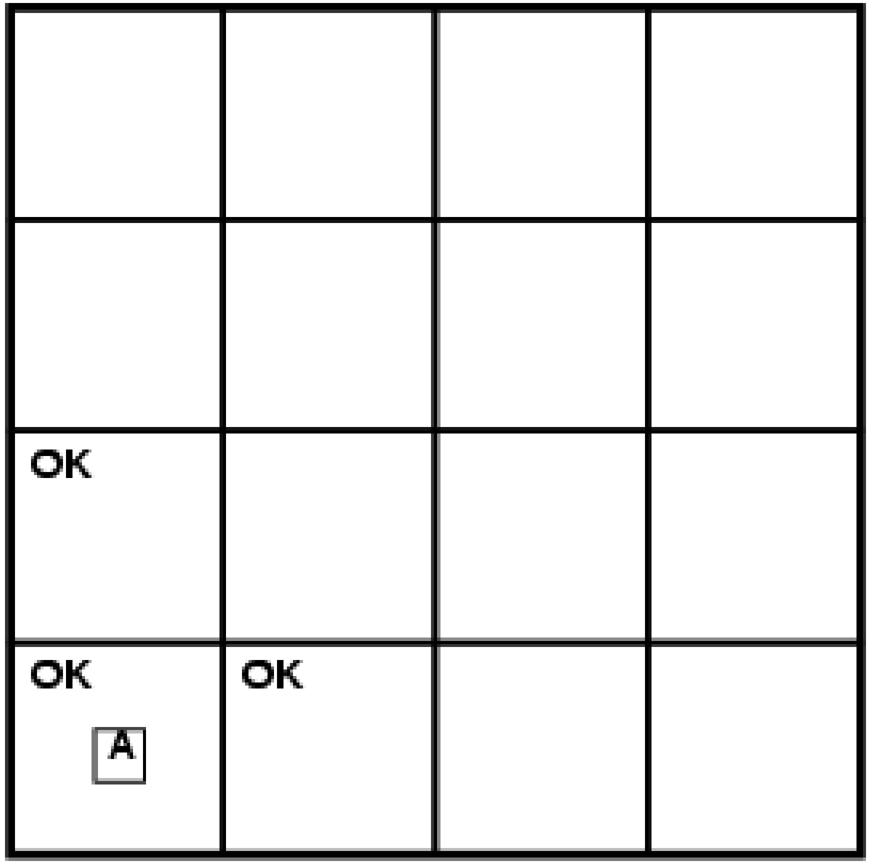
Observable	• No – only local perception
Deterministic	• Yes – outcomes exactly specified
Episodic	• No – sequential at the level of actions
Static	• Yes – Wumpus and Pits do not move
Discrete	• Yes
Single-agent	

Wumpus World Environment Characterization

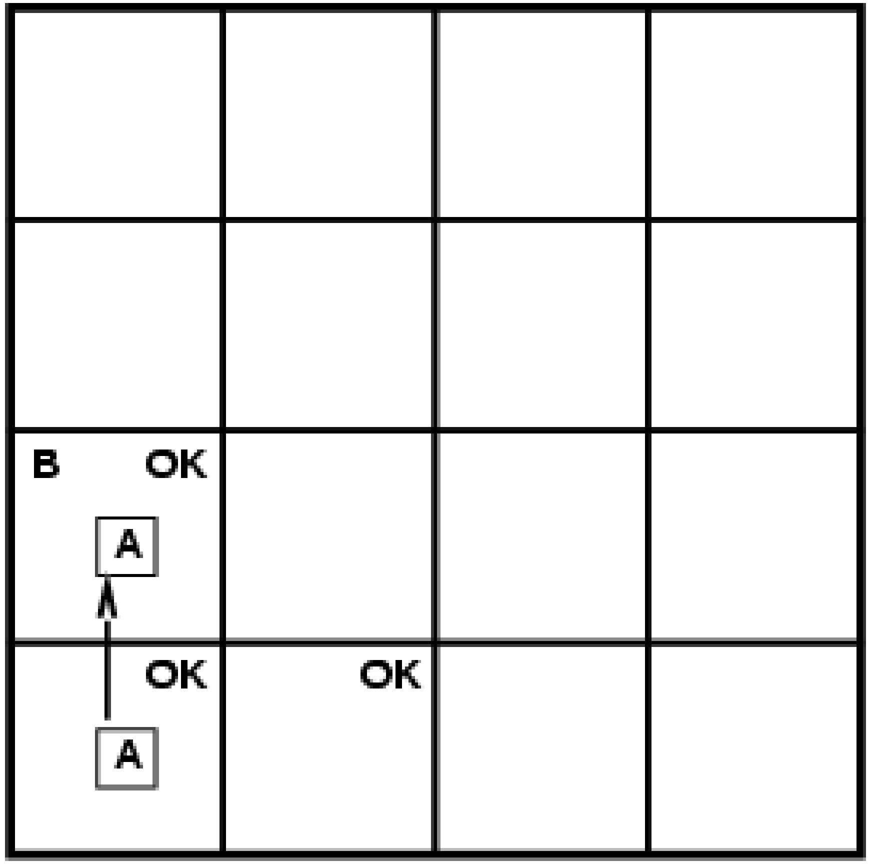


Observable	• No – only local perception
Deterministic	• Yes – outcomes exactly specified
Episodic	• No – sequential at the level of actions
Static	• Yes – Wumpus and Pits do not move
Discrete	• Yes
Single-agent	• Yes – Wumpus is not moving

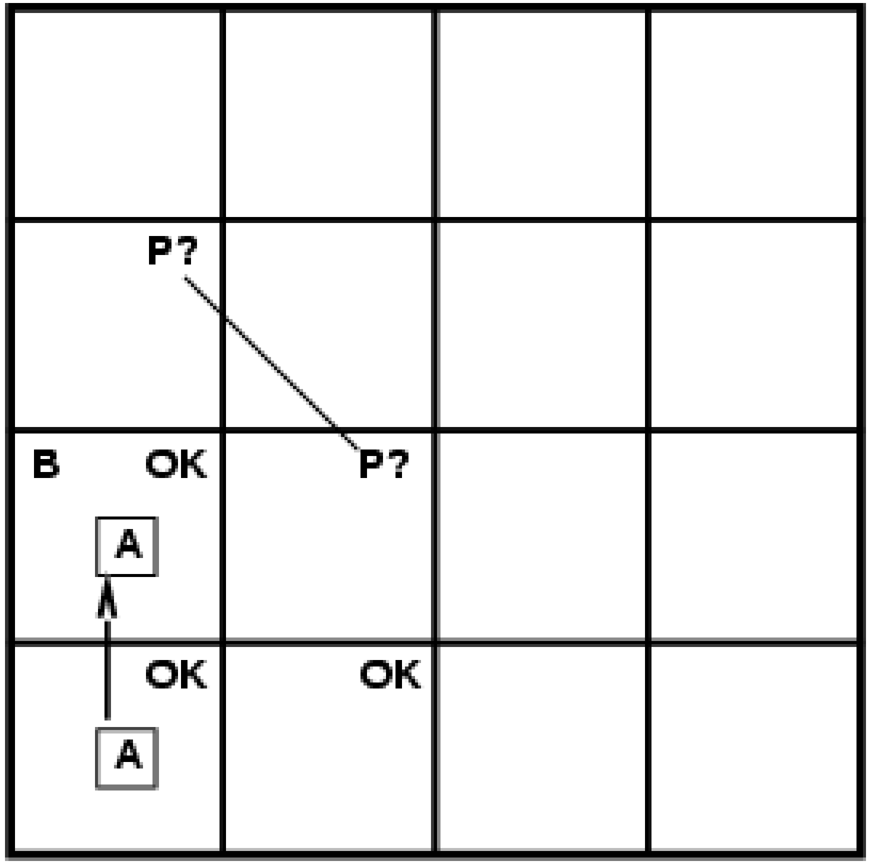
Exploring a wumpus world



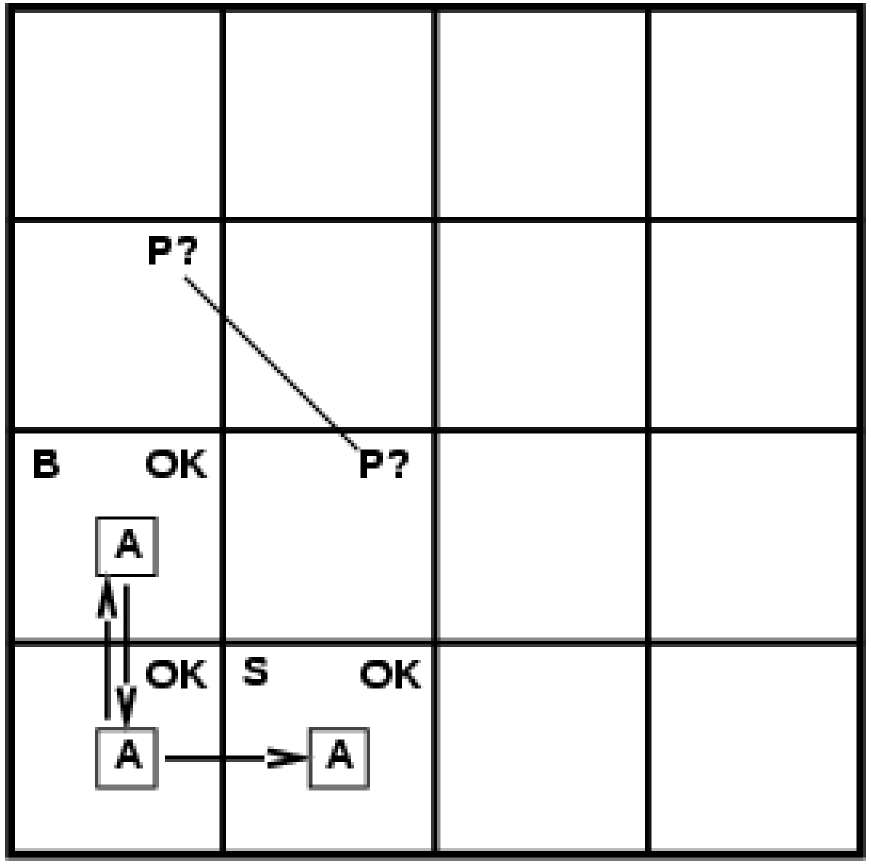
Exploring a wumpus world



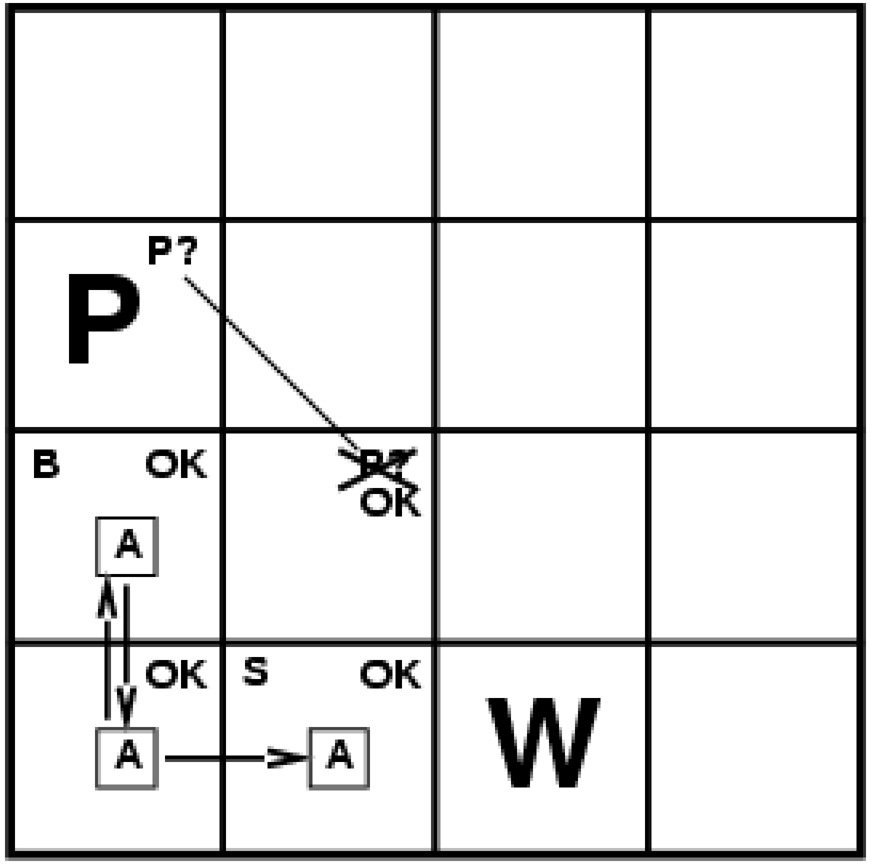
Exploring a wumpus world



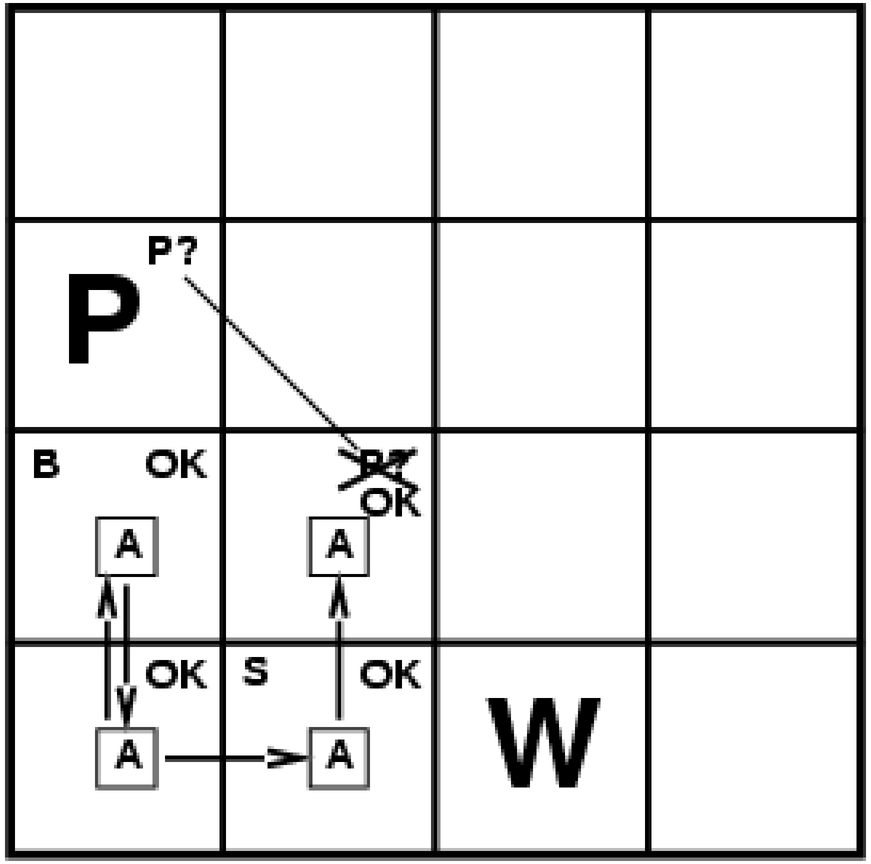
Exploring a wumpus world



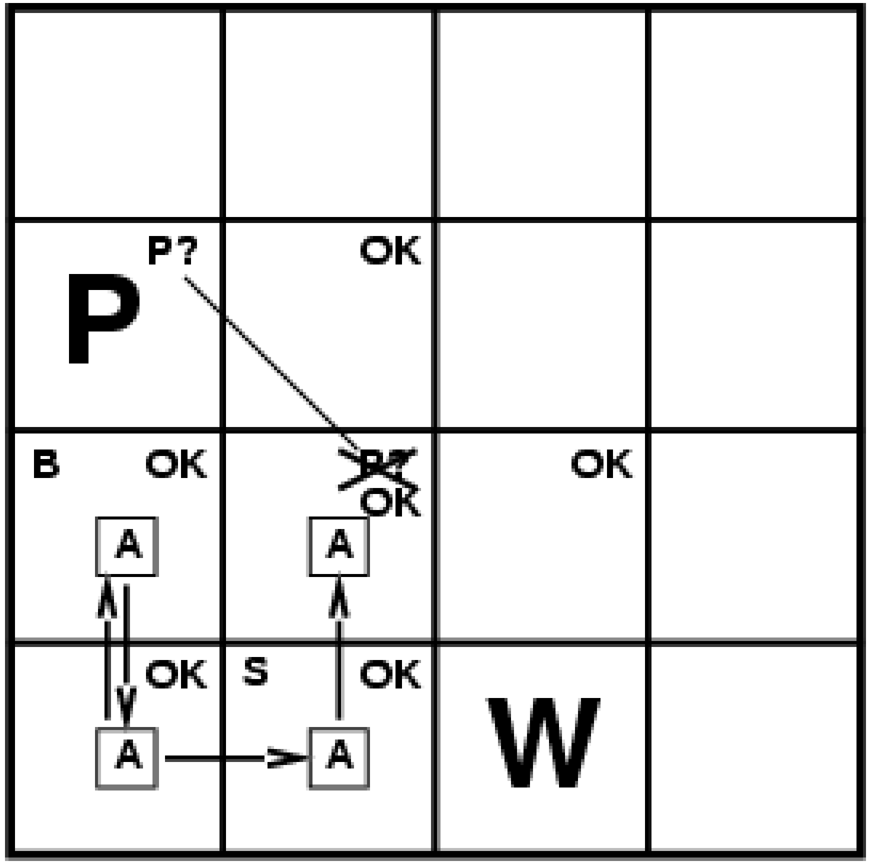
Exploring a wumpus world



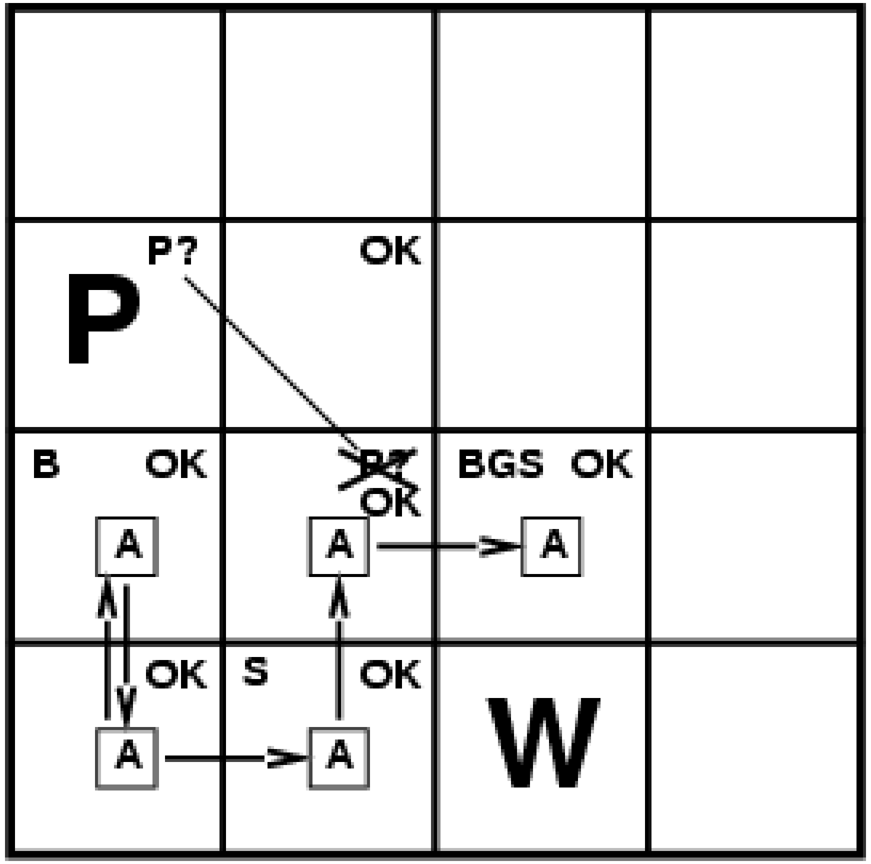
Exploring a wumpus world



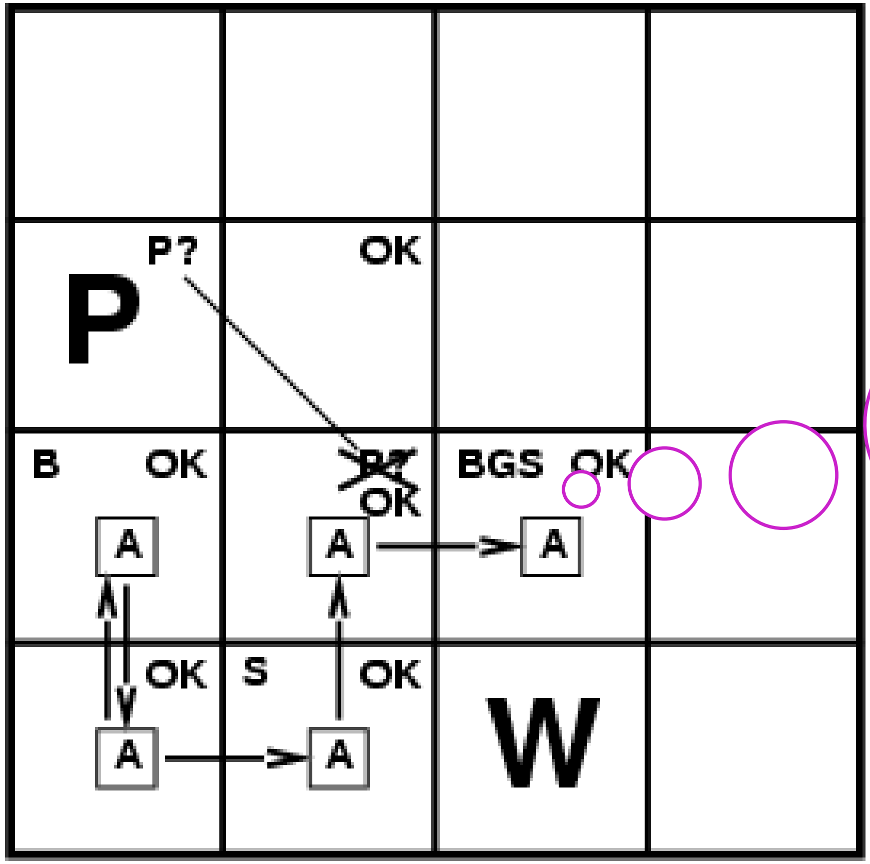
Exploring a wumpus world



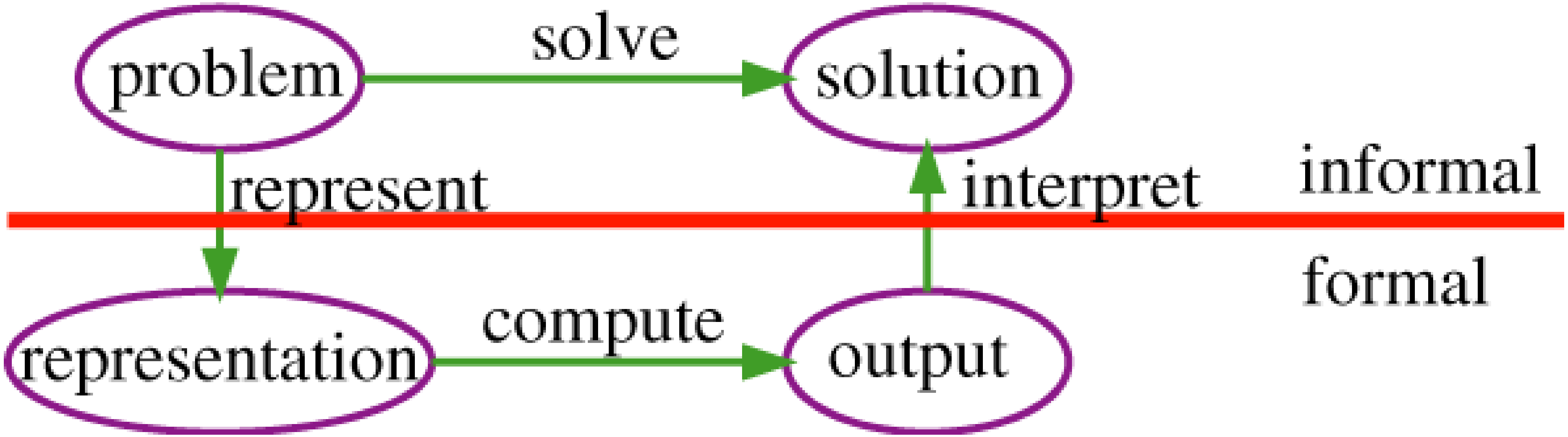
Exploring a wumpus world



Exploring a wumpus world



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


problem
solve
solution
represent
interpret
informal
formal
representation
compute
output

Logic

Logic in general

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

Syntax defines the sentences in the language

Semantics defines the *meaning* of sentences; define truth of a sentence in a world

e.g., the language of arithmetic

Syntax	**Semantics**
$x+2 \geq y$ is a sentence	$x+2 \geq y$ is true iff the number $x+2$ is no less than the number y
x2+y > {} is not a sentence	$x+2 \geq y$ is true in a world where $x = 7$, $y = 1$
	$x+2 \geq y$ is false in a world where $x = 0$, $y = 6$

Entailment

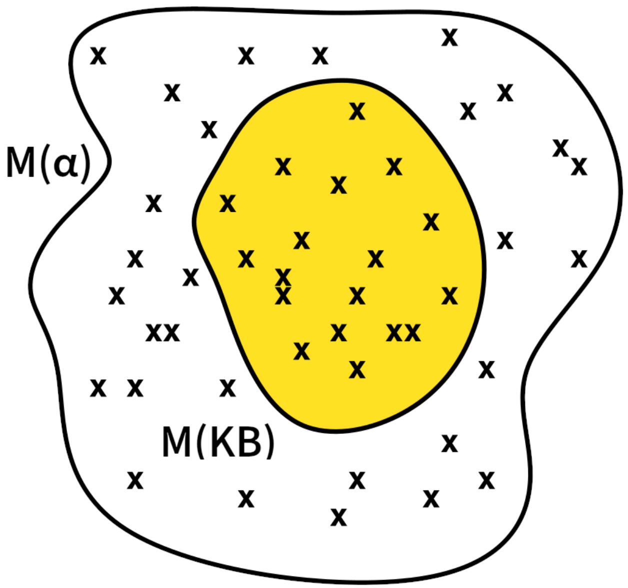
- Entailment means that one thing follows from another:

$$KB \models \alpha$$

- Knowledge base *KB* entails sentence α iff α is true in all worlds where *KB* is true
  - e.g., x+y = 4 entails 4 = x+y
  - e.g., the KB containing "*Celtic won*" and "*Hearts won*" entails "*Either Celtic won or Hearts won*"
- Entailment is a relationship between sentences (*syntax*) that is based on *semantics*

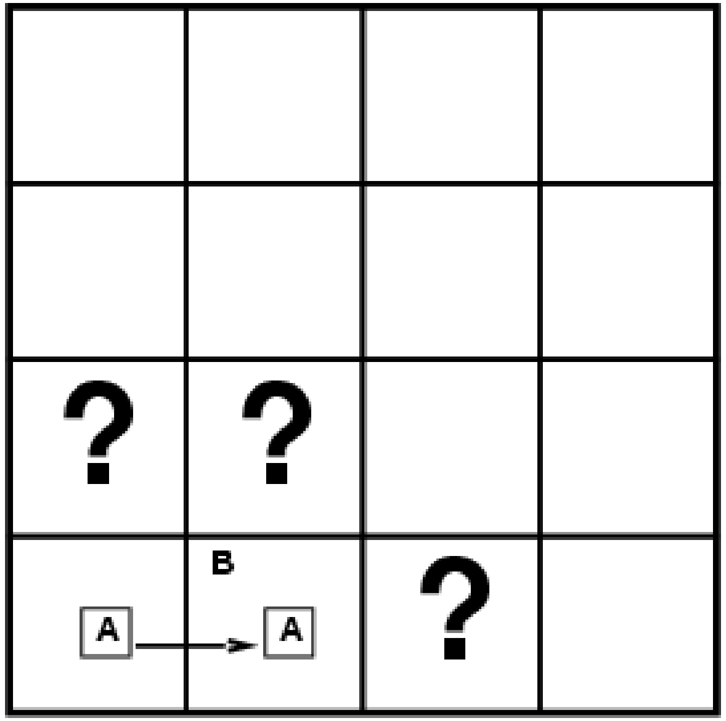
Models

- Logicians typically think in terms of models that are formally structured worlds with respect to which truth can be evaluated
- We say *m* is a model of a sentence α if α is true in *m*.
- *M(α)* is the set of all models of α.
- KB ⊨ α iff *M(KB)* ⊆ *M(α)*
- The *stricter* an assertion, the fewer models it has.



Entailment in the wumpus world

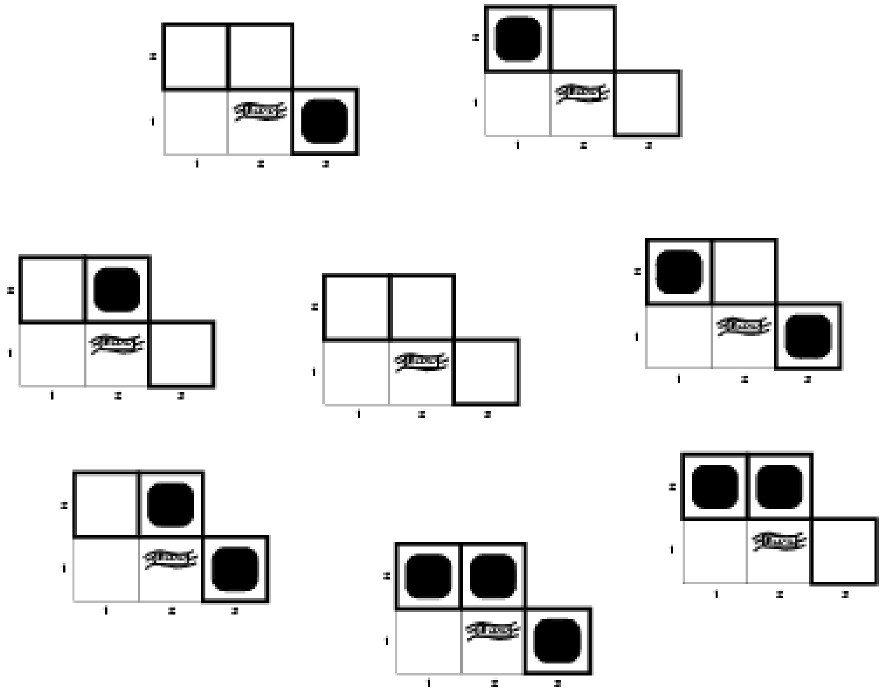
- Situation after detecting nothing in [1,1], moving right, breeze in [2,1]

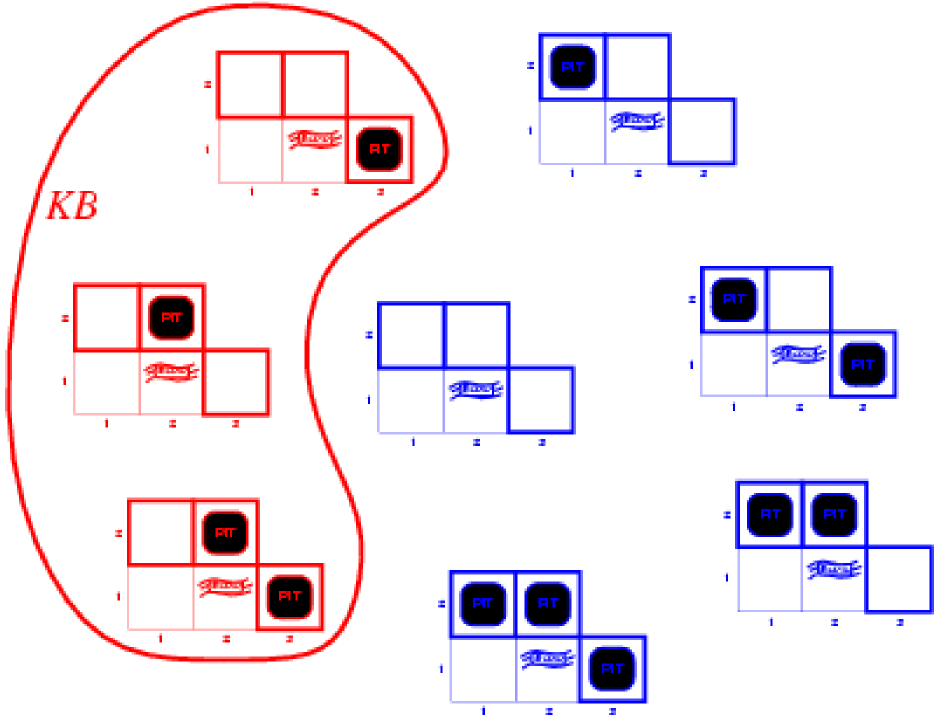


- Possible models for *KB* assuming only pits

  3 Boolean choices → 8 possible models

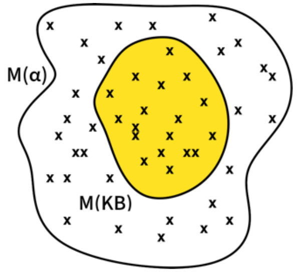
Wumpus models

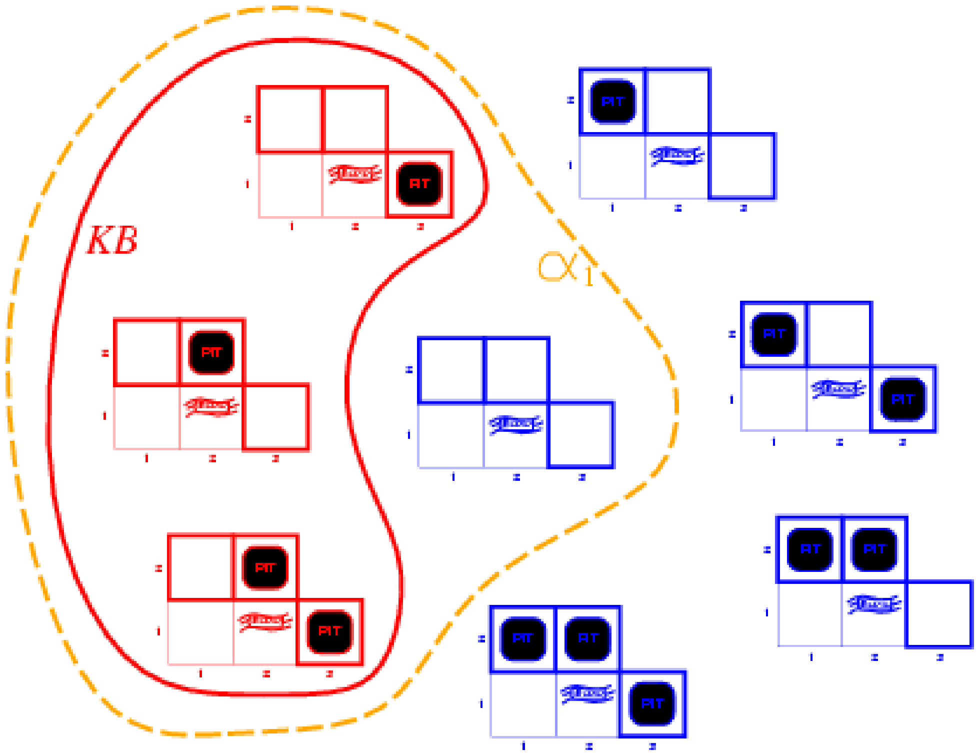




Wumpus models

KB = wumpus-world rules + observations





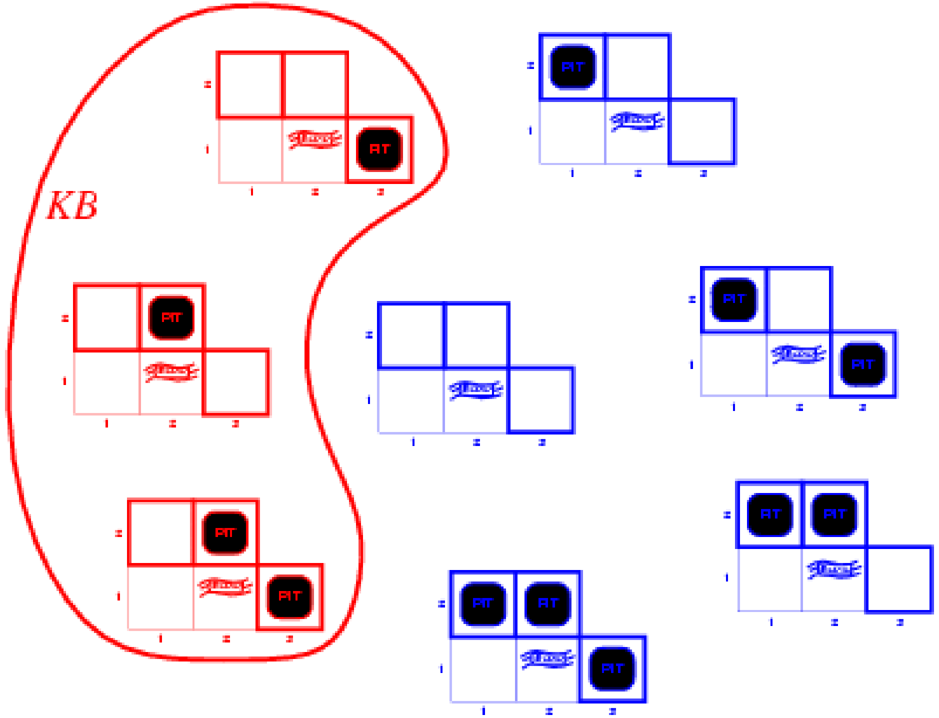
Wumpus models

KB = wumpus-world rules + observations

α_1 = "[1,2] has no pit"

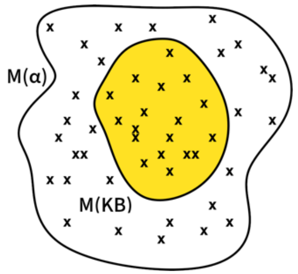
KB ⊨ α_1, proved by model checking

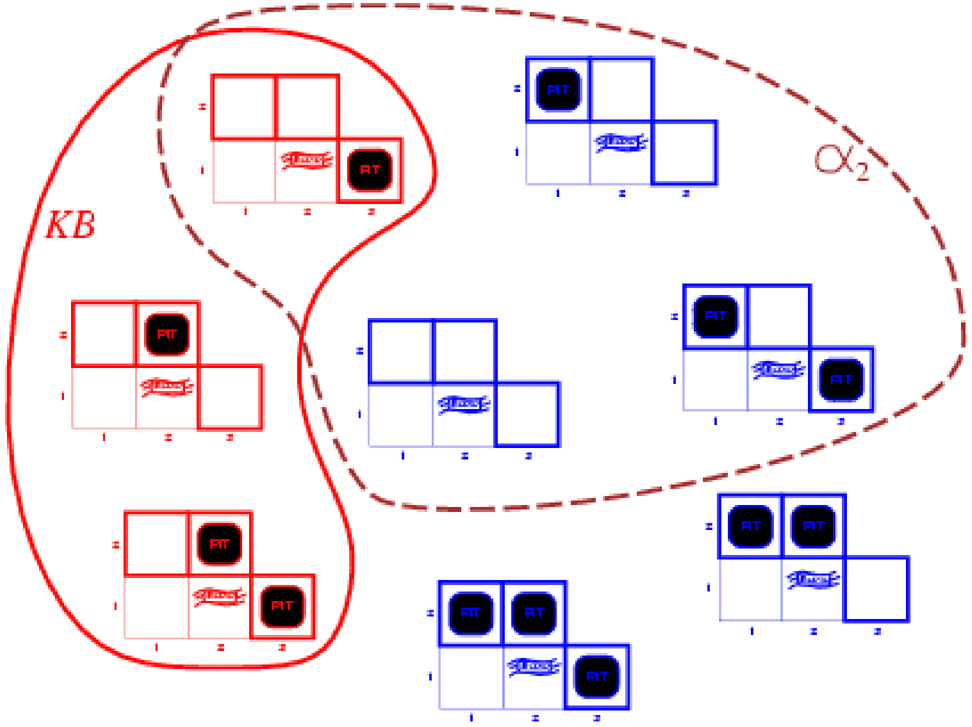
- In every model where KB is true, α_1 is also true



Wumpus models

KB = wumpus-world rules + observations





Wumpus models

KB = wumpus-world rules + observations

α_2 = "[2,2] has no pit"

$KB \not\models \alpha_2$, cannot be proved by model checking

- In some models in which KB is true, α_2 is false

Inference

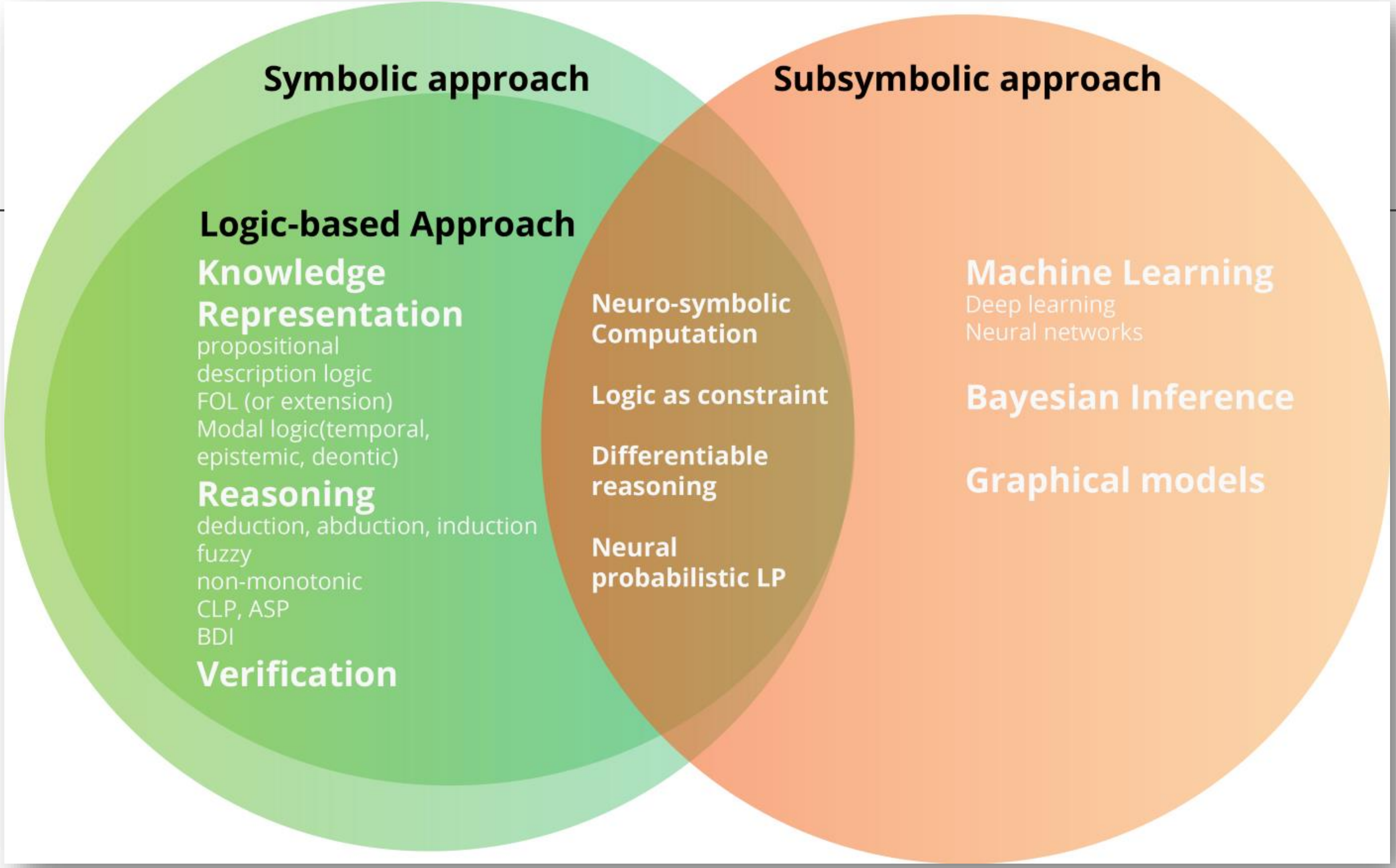
$KB \vdash_i \alpha$ = sentence α can be derived from KB by inference procedure i

Soundness

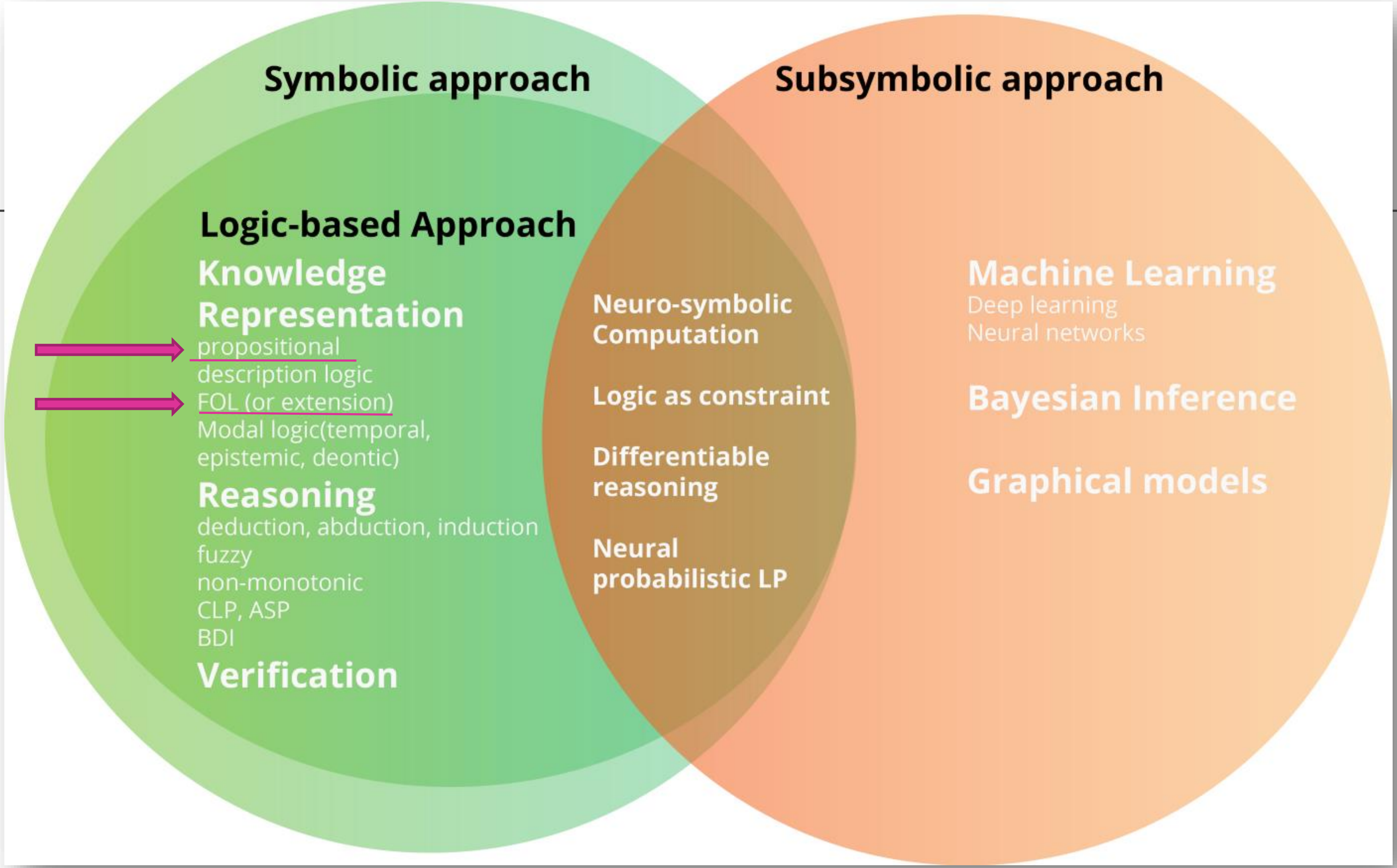
- i is sound if whenever $KB \vdash_i \alpha$, it is also true that $KB \models \alpha$

Completeness

- i is complete if whenever $KB \models \alpha$, it is also true that $KB \vdash_i \alpha$



https://miro.medium.com/max/1400/1*lFbqqQ5UsCtmRrowjthNuA.png



https://miro.medium.com/max/1400/1*lFbqqQ5UsCtmRrowjthNuA.png

Propositional logic

P, Q, R, …

¬

∧

∨

⇒

⇔

Propositional logic

Proposition Symbols

P, Q, R, …	¬	∧
∨	⇒	⇔

Logical Connectives

Propositional logic: Syntax

Propositional logic is the simplest logic – illustrates basic ideas

- The proposition symbols P_1, Q; or True, False etc. are atomic sentences
- 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]
- If S_1 and S_2 are sentences, $S_1 \vee S_2$ is a sentence [disjunction]
- 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]

Propositional logic: Semantics

- Each model specifies true/false for each proposition symbol

e.g. , $P_{1,2}$=false $P_{2,2}$=true $P_{3,1}$=false

- With these symbols, 8 possible models
  - can be enumerated automatically!

Propositional logic: Semantics

- Rules for evaluating truth with respect to a model *m*:

¬S is true iff S is false

S1 ∧ S2 is true iff S1 is true and S2 is true

S1 ∨ S2 is true iff S1 is true or S2 is true

S1 ⇒ S2 is true iff S1 is false or S2 is true

i.e., is false iff S1 is true and S2 is false

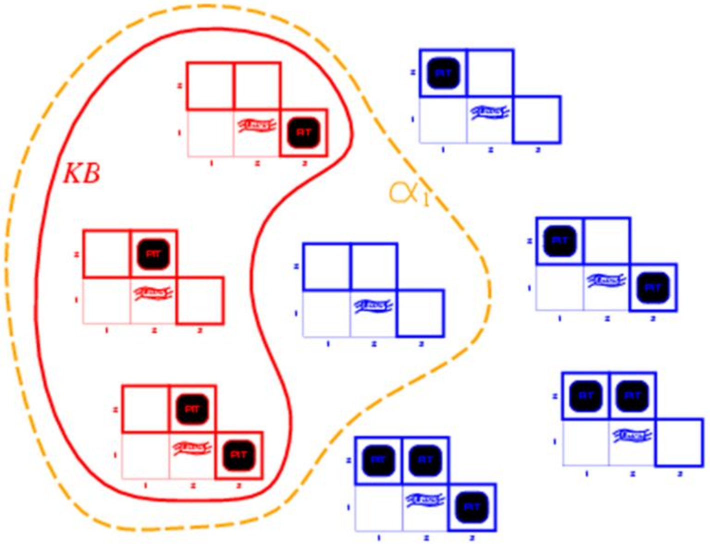
S1 ⇔ S2 is true iff S1 ⇒ S2 is true and S2 ⇒ S1 is true

- Simple recursive process evaluates an arbitrary sentence:

$$\neg P_{1,2} \wedge (P_{2,2} \vee P_{3,1}) = \mathit{true} \wedge (\mathit{true} \vee \mathit{false}) = \mathit{true} \wedge \mathit{true} = \mathit{true}$$

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
$false$	$false$	$true$	$false$	$false$	$true$	$true$
$false$	$true$	$true$	$false$	$true$	$true$	$false$
$true$	$false$	$false$	$false$	$true$	$false$	$false$
$true$	$true$	$false$	$true$	$true$	$true$	$true$

Truth tables for connectives



Wumpus world sentences

- 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].

$$\neg P_{1,1} \quad \neg B_{1,1} \quad B_{2,1}$$

- *"Pits cause breezes in adjacent squares"*

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

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

α_1 = "[1,2] has no pit" ???

$B_{1,1}$	$B_{2,1}$	$P_{1,1}$	$P_{1,2}$	$P_{2,1}$	$P_{2,2}$	$P_{3,1}$	KB	α_1
false	*false*	*false*	*false*	*false*	*false*	*false*	*false*	*true*
false	*false*	*false*	*false*	*false*	*false*	*true*	*false*	*true*
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
true	*true*	*false*	*false*	*false*	*false*	*false*	*false*	*true*
false	*true*	*false*	*false*	*false*	*false*	*true*	*true*	*true*
false	*true*	*false*	*false*	*false*	*true*	*false*	*true*	*true*
false	*true*	*false*	*false*	*false*	*true*	*true*	*true*	*true*
false	*true*	*false*	*false*	*true*	*false*	*false*	*false*	*true*
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
true	*true*	*true*	*true*	*true*	*true*	*true*	*false*	*false*

Truth tables for inference

function TT-ENTAILS?(KB, α) **returns** *true* or *false*
inputs: KB, the knowledge base, a sentence in propositional logic
α, the query, a sentence in propositional logic

symbols ← a list of the proposition symbols in KB and α
return TT-CHECK-ALL($KB, \alpha, symbols, \{\}$)

---

function TT-CHECK-ALL($KB, \alpha, symbols, model$) **returns** *true* or *false*
if EMPTY?(*symbols*) **then**
if PL-TRUE?($KB, model$) **then return** PL-TRUE?($\alpha, model$)
else return *true* // *when KB is false, always return true*
else do
P ← FIRST(*symbols*)
rest ← REST(*symbols*)
return (TT-CHECK-ALL($KB, \alpha, rest, model \cup \{P = true\}$)
and
TT-CHECK-ALL($KB, \alpha, rest, model \cup \{P = false\}$))

Inference by enumeration

- Depth-first enumeration of all models is sound and complete
- **PL-TRUE?**
  - returns true if a sentence holds in a model
- For *n* symbols
  - Time complexity is $O(2^n)$
  - Space complexity is $O(n)$

Logical equivalence

Two sentences are logically equivalent iff true in the same models:

$$\alpha \equiv \beta \text{ iff } \alpha \models \beta \text{ and } \beta \models \alpha$$

$(\alpha \land \beta) \equiv (\beta \land \alpha)$	commutativity of $\land$
$(\alpha \lor \beta) \equiv (\beta \lor \alpha)$	commutativity of $\lor$
$((\alpha \land \beta) \land \gamma) \equiv (\alpha \land (\beta \land \gamma))$	associativity of $\land$
$((\alpha \lor \beta) \lor \gamma) \equiv (\alpha \lor (\beta \lor \gamma))$	associativity of $\lor$
$\neg(\neg\alpha) \equiv \alpha$	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 $\land$ over $\lor$
$(\alpha \lor (\beta \land \gamma)) \equiv ((\alpha \lor \beta) \land (\alpha \lor \gamma))$	distributivity of $\lor$ over $\land$

Validity and Satisfiability

A sentence is **valid** if it is true in *all models*

- *true*, $A \lor \neg A$, $A \Rightarrow A$, $(A \land (A \Rightarrow B)) \Rightarrow B$

Validity is connected to inference via the **Deduction Theorem**

- $KB \models \alpha$ if and only if $(KB \Rightarrow \alpha)$ is valid

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

- e.g., $A \lor B$, C

A sentence is **unsatisfiable** if it is true in *no models*

- e.g., $A \land \neg A$

Satisfiability is connected to inference via the following:

- $KB \models \alpha$ if and only if $(KB \land \neg\alpha)$ is unsatisfiable
- prove α by *reductio ad absurdum*

Propositional Theorem Proving

APPLICATION OF INFERENCE RULES

- Legitimate (sound) generation of new sentences from old
- **Proof** = a sequence of inference rule applications
  - Can use inference rules as operators in a standard search algorithm!
- Typically require transformation of sentences into a **normal form**
- Example: resolution

MODEL CHECKING

- truth table enumeration
  - (always exponential in *n*)
- improved backtracking
  - e.g., DPLL
- heuristic search in model space
  - (sound but incomplete)
  - e.g., min-conflicts-like hill-climbing algorithms

Summary

- 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 wrt 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