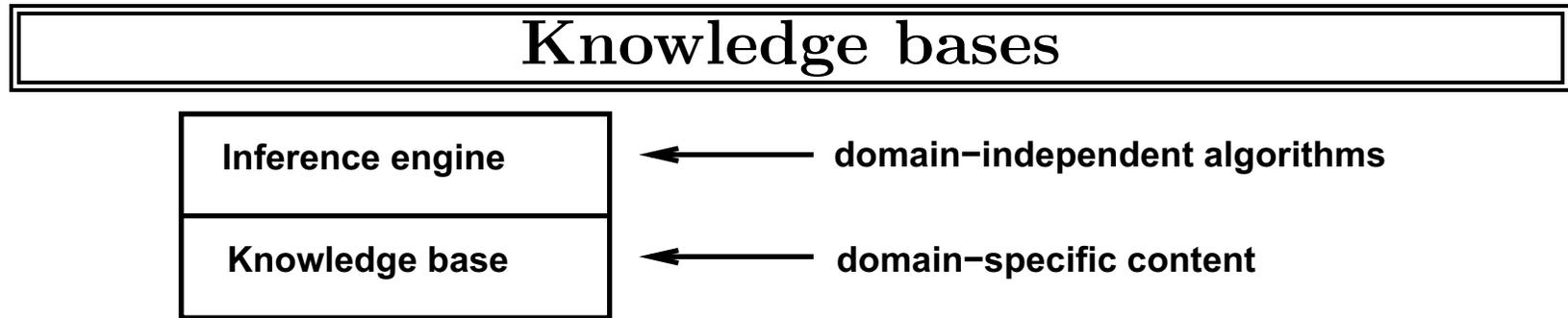


LOGICAL AGENTS

CHAPTER 7

Outline

- ◇ Knowledge-based agents
- ◇ Wumpus world
- ◇ Logic in general—models and entailment
- ◇ Propositional (Boolean) logic
- ◇ Equivalence, validity, satisfiability
- ◇ Inference rules and theorem proving
 - forward chaining
 - backward chaining
 - resolution



Knowledge base = 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

Or at the **implementation level**

i.e., data structures in KB and algorithms that manipulate them

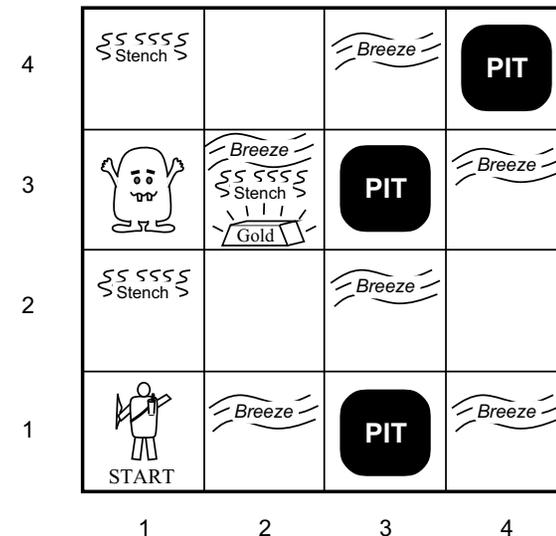
Wumpus World PEAS description

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



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

Sensors Breeze, Glitter, Smell

Wumpus world characterization

Observable??

Wumpus world characterization

Observable?? No—only **local** perception

Deterministic??

Wumpus world characterization

Observable?? No—only **local** perception

Deterministic?? Yes—outcomes exactly specified

Episodic??

Wumpus world characterization

Observable?? No—only **local** perception

Deterministic?? Yes—outcomes exactly specified

Episodic?? No—sequential at the level of actions

Static??

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??

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??

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

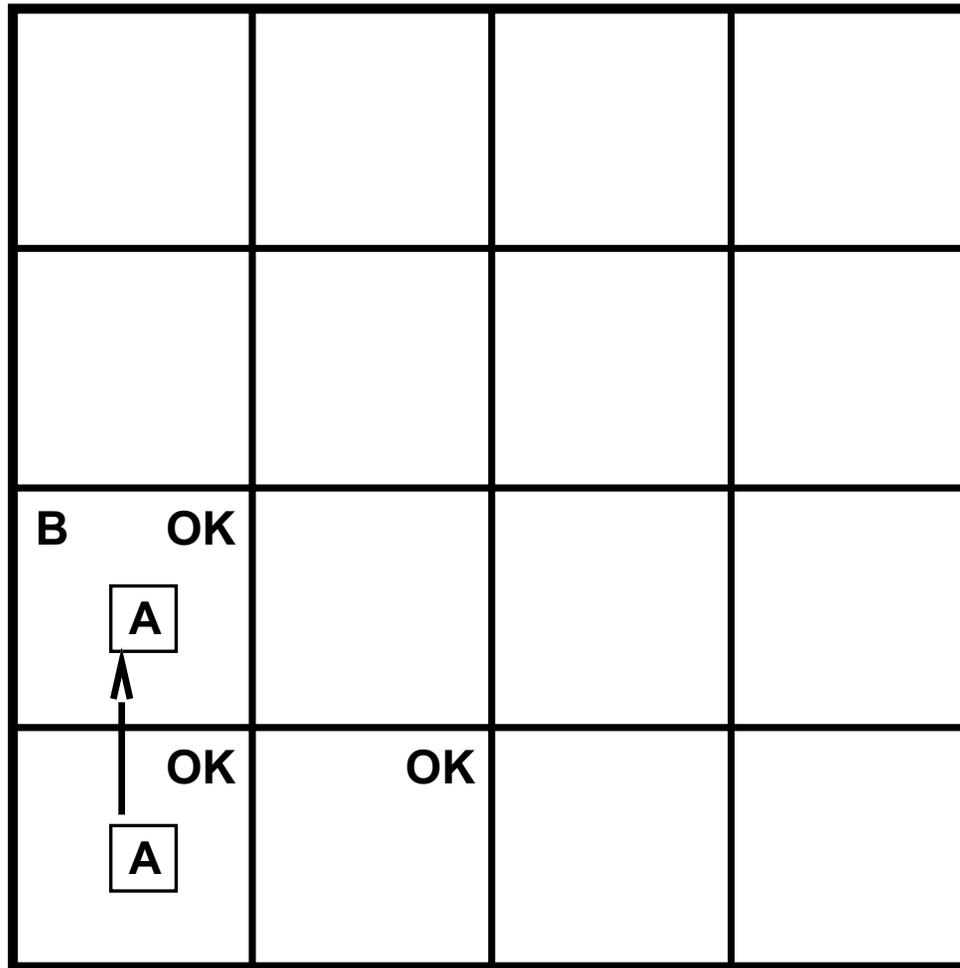
Discrete?? Yes

Single-agent?? Yes—Wumpus is essentially a natural feature

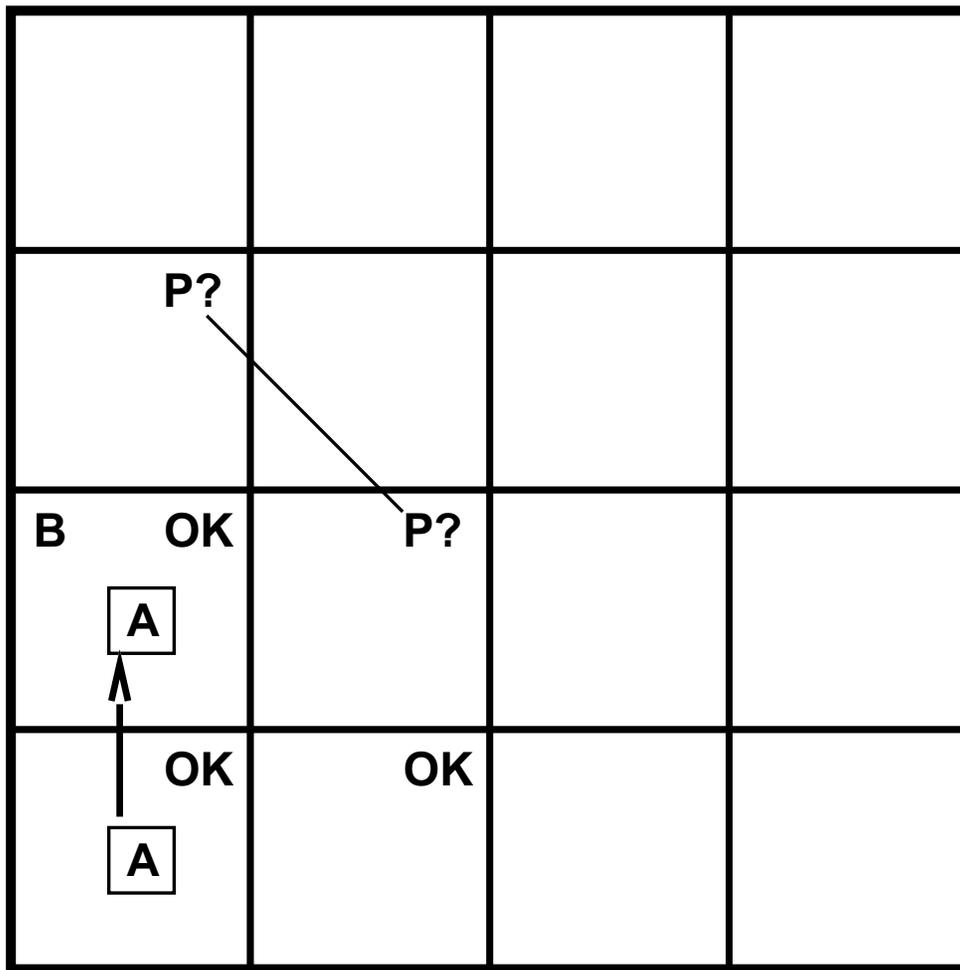
Exploring a wumpus world

| | | | |
|---|----|--|--|
| | | | |
| | | | |
| OK | | | |
| OK A | OK | | |

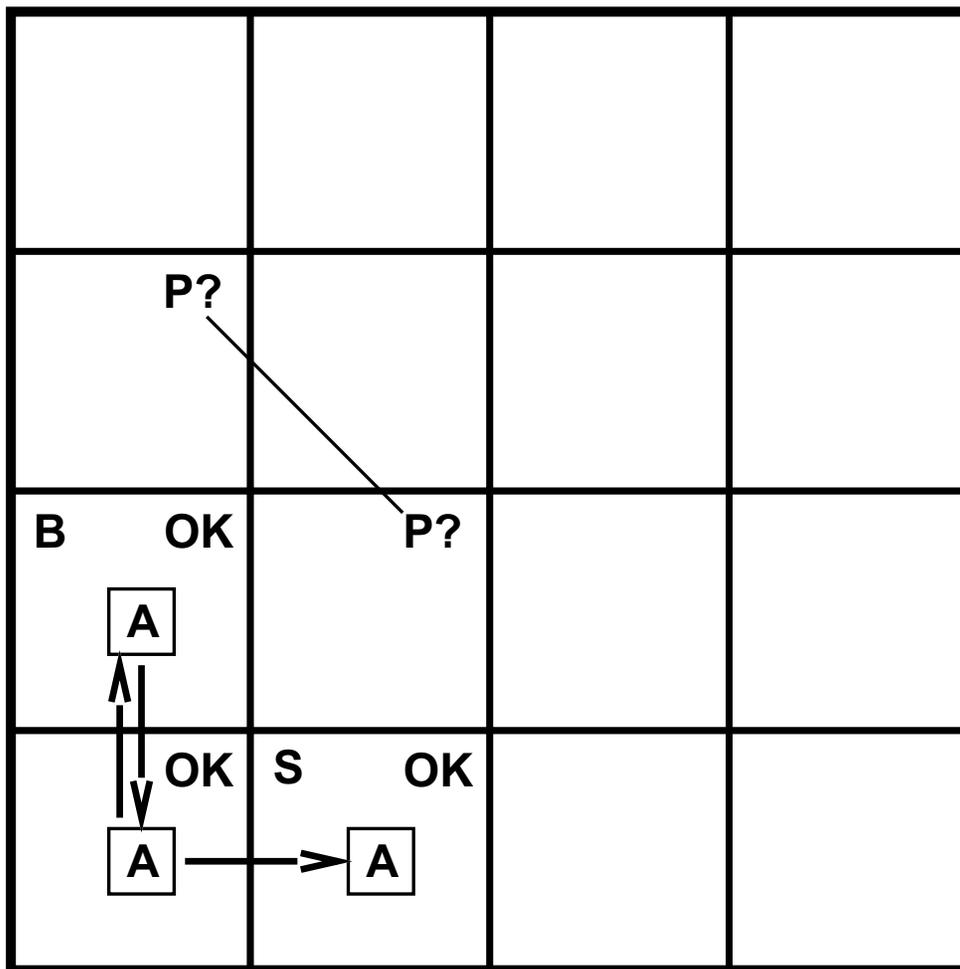
Exploring a wumpus world



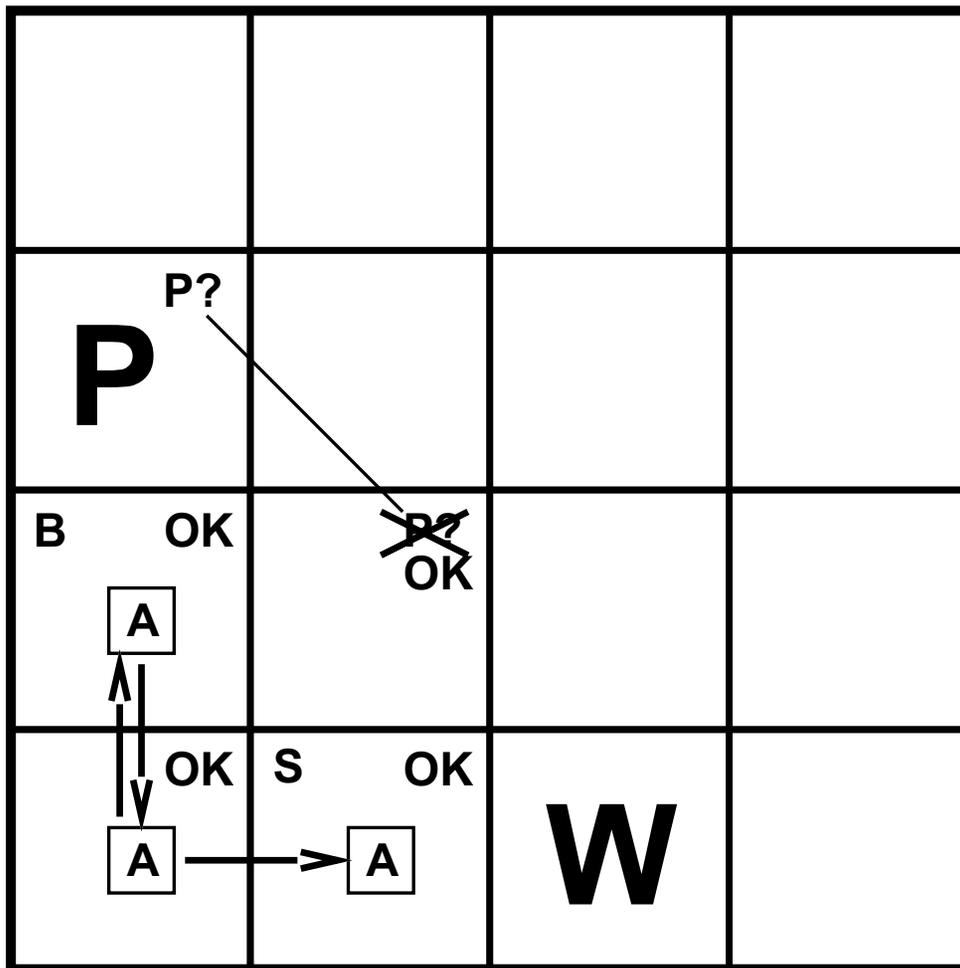
Exploring a wumpus world



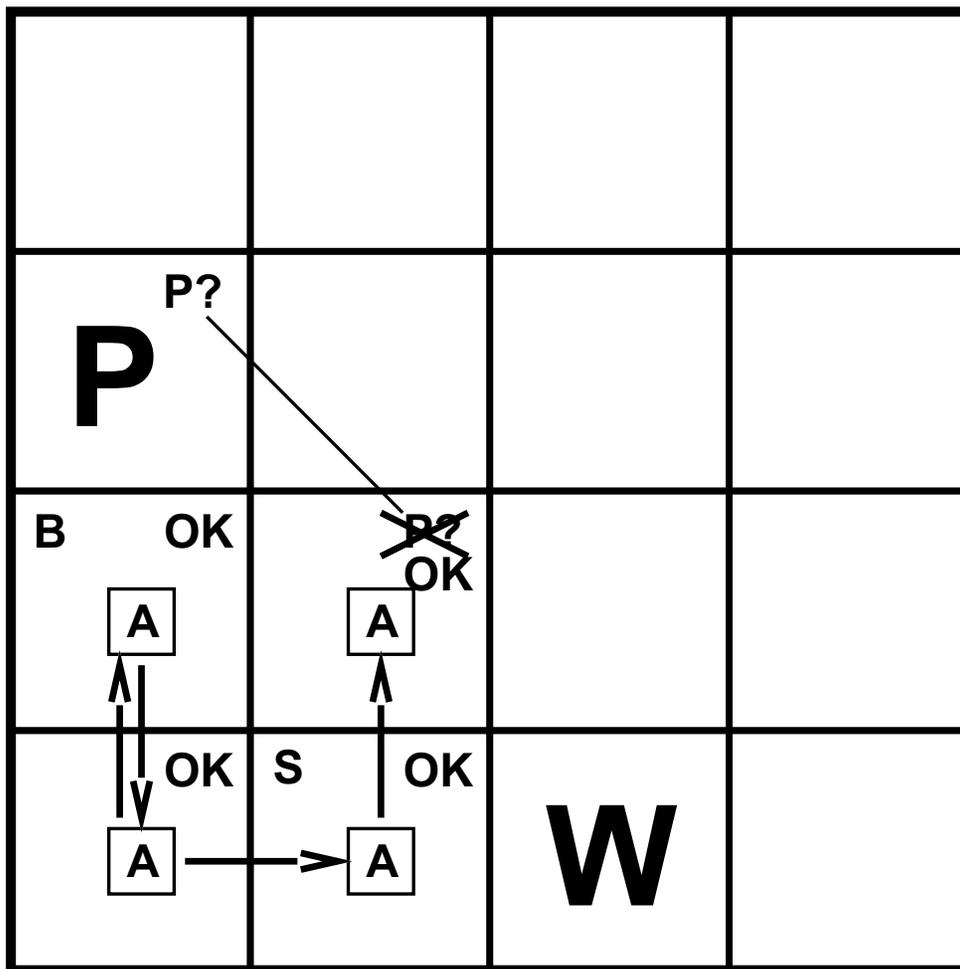
Exploring a wumpus world



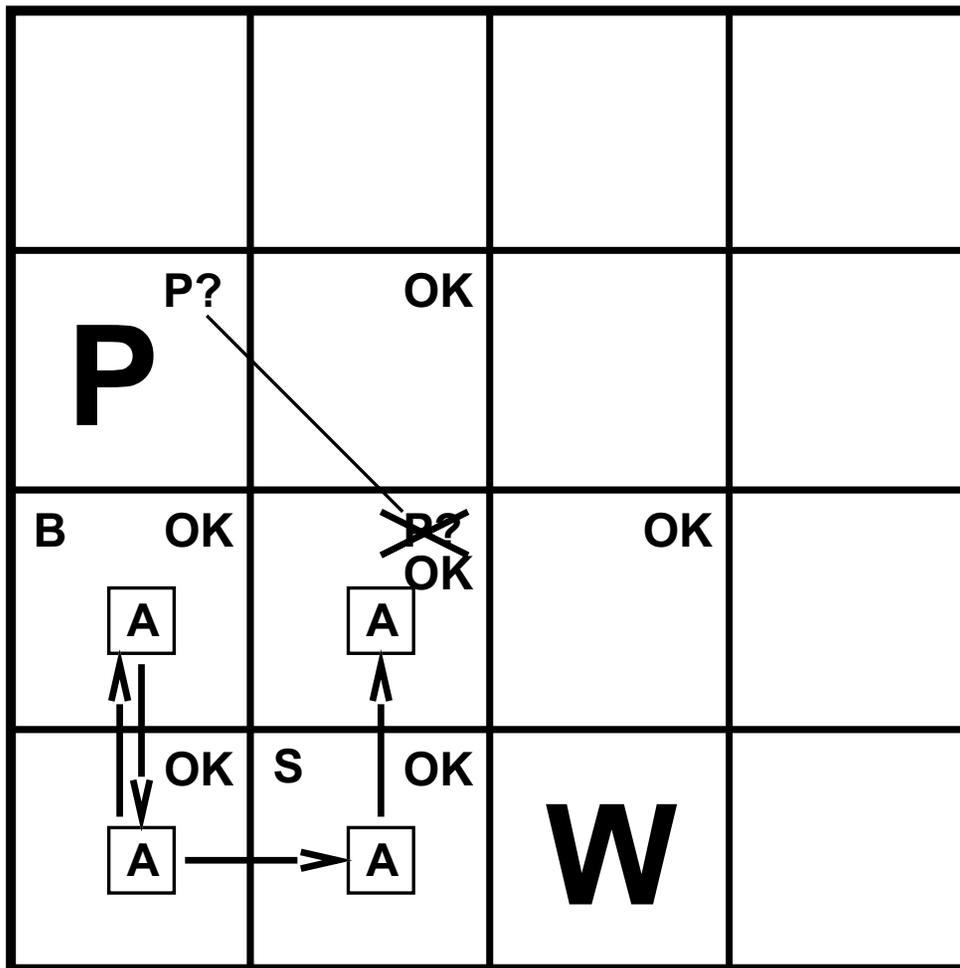
Exploring a wumpus world



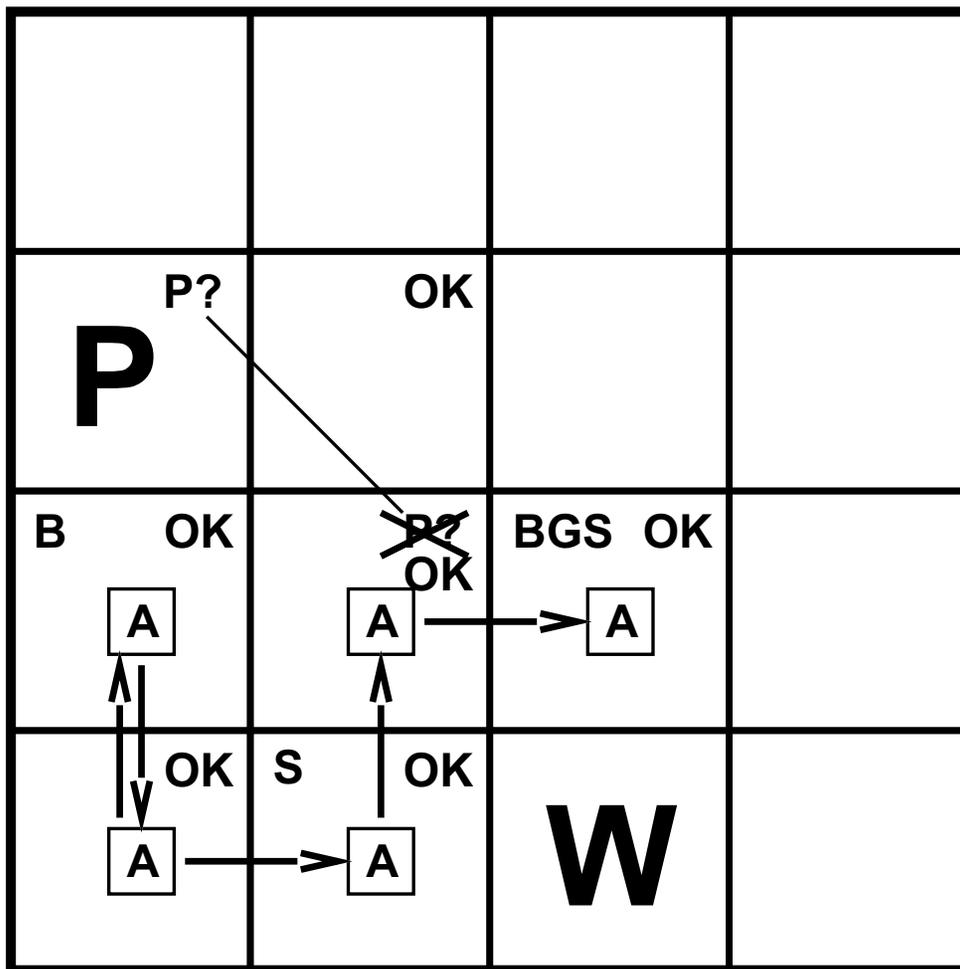
Exploring a wumpus world



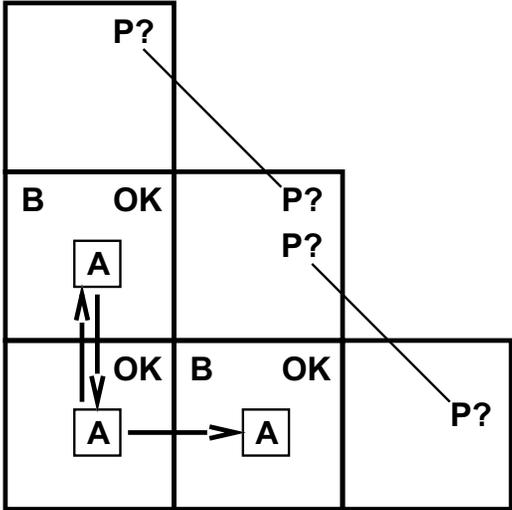
Exploring a wumpus world



Exploring a wumpus world



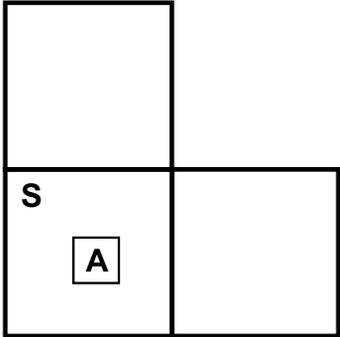
Other tight spots



Breeze in (1,2) and (2,1)
 \Rightarrow no safe actions

Assuming pits uniformly distributed,
 (2,2) has pit w/ prob 0.86, vs. 0.31

Smell in (1,1)
 \Rightarrow cannot move



Can use a strategy of **coercion**:
 shoot straight ahead
 wumpus was there \Rightarrow dead \Rightarrow safe
 wumpus wasn't there \Rightarrow safe

Logic in general

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

Syntax defines the sentences in the language

Semantics define the “meaning” of sentences; i.e., define truth of a sentence in a world

E.g., the language of arithmetic

$x + 2 \geq y$ is a sentence; $x^2 + y >$ 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 $x = 0, y = 6$

Entailment

Entailment means that one thing **follows from** another:

$$KB \models \alpha$$

Knowledge base KB entails sentence α
if and only if
 α is true in all worlds where KB is true

E.g., the KB containing “the Giants won” and “the Reds won”
entails “Either the Giants won or the Reds won”

E.g., $x + y = 4$ entails $4 = x + y$

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

Note: brains process **syntax** (of some sort)

Inference

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

Consequences of KB are a haystack; α is a needle.

Entailment = needle in haystack; inference = finding it

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 $KB \vdash_i \alpha$

Preview: we will define a logic (first-order logic) which is expressive enough to say almost anything of interest, and for which there exists a sound and complete inference procedure.

That is, the procedure will answer any question whose answer follows from what is known by the KB .

Propositional logic: Syntax

Propositional logic is the simplest logic—illustrates basic ideas

The proposition symbols P_1, P_2 etc are 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. $P_{1,2}$ $P_{2,2}$ $P_{3,1}$
true true false

(With these symbols, 8 possible models, can be enumerated automatically.)

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 true iff | S_1 | is false or S_2 is true |
| i.e., is false iff | S_1 | is true and S_2 is false |
| $S_1 \Leftrightarrow S_2$ is true iff | $S_1 \Rightarrow S_2$ | is true and $S_2 \Rightarrow S_1$ is true |

Simple recursive process evaluates an arbitrary sentence, e.g.,

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

Truth tables for 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> |

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 $[i, j]$.

$$\neg P_{1,1}$$

$$\neg B_{1,1}$$

$$B_{2,1}$$

“Pits cause breezes in adjacent squares”

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 $[i, j]$.

$$\neg P_{1,1}$$

$$\neg B_{1,1}$$

$$B_{2,1}$$

“Pits cause breezes in adjacent squares”

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

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

“A square is breezy **if and only if** there is an adjacent pit”

Truth tables for inference

| $B_{1,1}$ | $B_{2,1}$ | $P_{1,1}$ | $P_{1,2}$ | $P_{2,1}$ | $P_{2,2}$ | $P_{3,1}$ | R_1 | R_2 | R_3 | R_4 | R_5 | KB |
|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------------|
| <i>false</i> | <i>true</i> | <i>true</i> | <i>true</i> | <i>true</i> | <i>false</i> | <i>false</i> |
| <i>false</i> | <i>false</i> | <i>false</i> | <i>false</i> | <i>false</i> | <i>false</i> | <i>true</i> | <i>true</i> | <i>true</i> | <i>false</i> | <i>true</i> | <i>false</i> | <i>false</i> |
| \vdots |
| <i>false</i> | <i>true</i> | <i>false</i> | <i>false</i> | <i>false</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>false</i> | <i>true</i> | <i>false</i> | <i>false</i> | <i>false</i> | <i>false</i> | <i>true</i> | <i>true</i> | <i>true</i> | <i>true</i> | <i>true</i> | <i>true</i> | <u><i>true</i></u> |
| <i>false</i> | <i>true</i> | <i>false</i> | <i>false</i> | <i>false</i> | <i>true</i> | <i>false</i> | <i>true</i> | <i>true</i> | <i>true</i> | <i>true</i> | <i>true</i> | <u><i>true</i></u> |
| <i>false</i> | <i>true</i> | <i>false</i> | <i>false</i> | <i>false</i> | <i>true</i> | <u><i>true</i></u> |
| <i>false</i> | <i>true</i> | <i>false</i> | <i>false</i> | <i>true</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> |
| \vdots |
| <i>true</i> | <i>false</i> | <i>true</i> | <i>true</i> | <i>false</i> | <i>true</i> | <i>false</i> |

Enumerate rows (different assignments to symbols),
 if **KB** is true in row, check that α is too

Logical equivalence

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

$\alpha \equiv \beta$ if and only if $\alpha \models \beta$ and $\beta \models \alpha$

| | |
|--|--|
| $(\alpha \wedge \beta) \equiv (\beta \wedge \alpha)$ | commutativity of \wedge |
| $(\alpha \vee \beta) \equiv (\beta \vee \alpha)$ | commutativity of \vee |
| $((\alpha \wedge \beta) \wedge \gamma) \equiv (\alpha \wedge (\beta \wedge \gamma))$ | associativity of \wedge |
| $((\alpha \vee \beta) \vee \gamma) \equiv (\alpha \vee (\beta \vee \gamma))$ | associativity of \vee |
| $\neg(\neg\alpha) \equiv \alpha$ | double-negation elimination |
| $(\alpha \Rightarrow \beta) \equiv (\neg\beta \Rightarrow \neg\alpha)$ | contraposition |
| $(\alpha \Rightarrow \beta) \equiv (\neg\alpha \vee \beta)$ | implication elimination |
| $(\alpha \Leftrightarrow \beta) \equiv ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha))$ | biconditional elimination |
| $\neg(\alpha \wedge \beta) \equiv (\neg\alpha \vee \neg\beta)$ | De Morgan |
| $\neg(\alpha \vee \beta) \equiv (\neg\alpha \wedge \neg\beta)$ | De Morgan |
| $(\alpha \wedge (\beta \vee \gamma)) \equiv ((\alpha \wedge \beta) \vee (\alpha \wedge \gamma))$ | distributivity of \wedge over \vee |
| $(\alpha \vee (\beta \wedge \gamma)) \equiv ((\alpha \vee \beta) \wedge (\alpha \vee \gamma))$ | distributivity of \vee over \wedge |

Validity and satisfiability

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

e.g., *True*, $A \vee \neg A$, $A \Rightarrow A$, $(A \wedge (A \Rightarrow B)) \Rightarrow B$

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., $A \vee B$, C

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

e.g., $A \wedge \neg A$

Satisfiability is connected to inference via the following:

$KB \models \alpha$ if and only if $(KB \wedge \neg \alpha)$ is unsatisfiable

i.e., prove α by *reductio ad absurdum*

Proof methods

Proof methods divide into (roughly) two kinds:

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 alg.
- Typically require translation of sentences into a **normal form**

Model checking

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

Forward and backward chaining

Horn Form (restricted)

KB = **conjunction** of **Horn clauses**

Horn clause =

◇ proposition symbol; or

◇ (conjunction of symbols) \Rightarrow symbol

E.g., $C \wedge (B \Rightarrow A) \wedge (C \wedge D \Rightarrow B)$

Modus Ponens (for Horn Form): complete for Horn KBs

$$\frac{\alpha_1, \dots, \alpha_n, \quad \alpha_1 \wedge \dots \wedge \alpha_n \Rightarrow \beta}{\beta}$$

Can be used with **forward chaining** or **backward chaining**.

These algorithms are very natural and run in **linear** time

Forward chaining

Idea: fire any rule whose premises are satisfied in the *KB*,
add its conclusion to the *KB*, until query is found

$$P \Rightarrow Q$$

$$L \wedge M \Rightarrow P$$

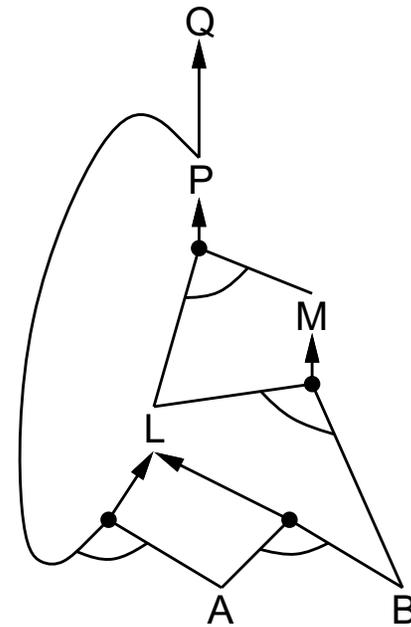
$$B \wedge L \Rightarrow M$$

$$A \wedge P \Rightarrow L$$

$$A \wedge B \Rightarrow L$$

A

B



Backward chaining

Idea: work backwards from the query q :

to prove q by BC,

check if q is known already, or

prove by BC all premises of some rule concluding q

Avoid loops: check if new subgoal is already on the goal stack

Avoid repeated work: check if new subgoal

1) has already been proved true, or

2) has already failed

Forward vs. backward chaining

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

May do lots of work that is irrelevant to the goal

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

Complexity of BC can be **much less** than linear in size of KB

Resolution

Conjunctive Normal Form (CNF—universal)

conjunction of **disjunctions** of **literals**
clauses

E.g., $(A \vee \neg B) \wedge (B \vee \neg C \vee \neg D)$

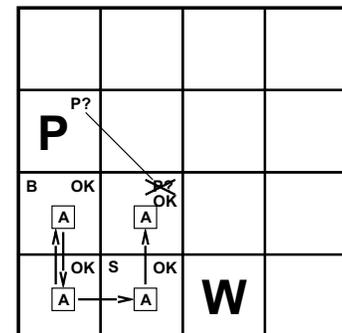
Resolution inference rule (for CNF): complete for propositional logic

$$\frac{l_1 \vee \dots \vee l_k, \quad m_1 \vee \dots \vee m_n}{l_1 \vee \dots \vee l_{i-1} \vee l_{i+1} \vee \dots \vee l_k \vee m_1 \vee \dots \vee m_{j-1} \vee m_{j+1} \vee \dots \vee m_n}$$

where l_i and m_j are complementary literals. E.g.,

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

Resolution is sound and complete for propositional logic



Conversion to CNF

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

1. Eliminate \Leftrightarrow , replacing $\alpha \Leftrightarrow \beta$ with $(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)$.

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

2. Eliminate \Rightarrow , replacing $\alpha \Rightarrow \beta$ with $\neg\alpha \vee \beta$.

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

3. Move \neg inwards using de Morgan's rules and double-negation:

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

4. Apply distributivity law (\vee over \wedge) and flatten:

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

Resolution example

$KB = (B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})) \wedge \neg B_{1,1} \quad \alpha = \neg P_{1,2}$

