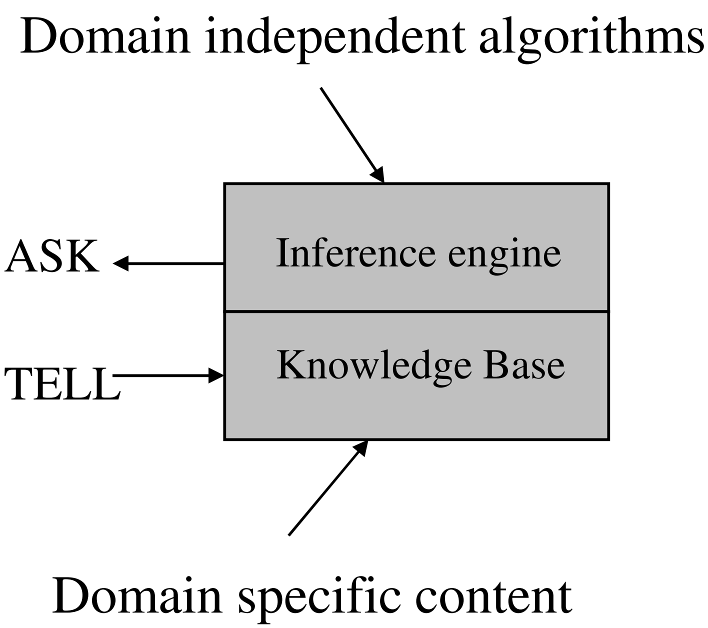
Knowledge and Reasoning

- Knowledge representation
- Wumpus world example
- Logic in general: models and entailment
- Propositional (Boolean) logic
- Normal forms
- Equivalence, validity, satisfiability
- Inference in propositional logic
  - Forward and Backward Chaining
  - Resolution

Knowledge-Based Agent



- Agent that uses **prior** or **acquired** knowledge to achieve its goals
  - Can make more efficient decisions
  - Can make informed decisions
- Knowledge Base (KB): contains a set of representations of facts about the Agent's environment
- Each representation is called a **sentence**
- Use some **knowledge representation language**, to TELL it what to know e.g., (temperature 72F)
- ASK agent to query what to do
- Agent can use inference to deduce new facts from TELLed facts

Knowledge bases

- DECLARATIVE approach to building an agent (or other system):
  - Tell it what it needs to know
  - Not PROCEDURAL – which is the alternative approach
- 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

Generic knowledge-based agent

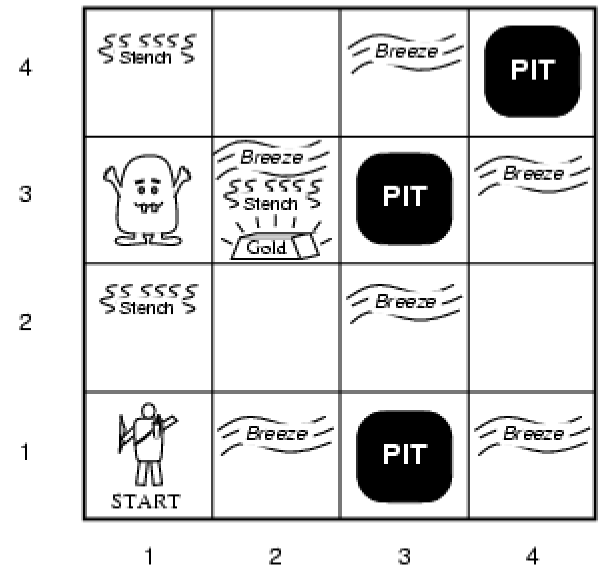
```
function KB-AGENT(percept) returns an action
   static: KB, a knowledge base
           t, a counter, initially 0, indicating time

   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

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



- **Sensors:** Stench, Breeze, Glitter, Bump, Scream
- **Actuators:** Left turn, Right turn, Forward, Grab, Release, Shoot

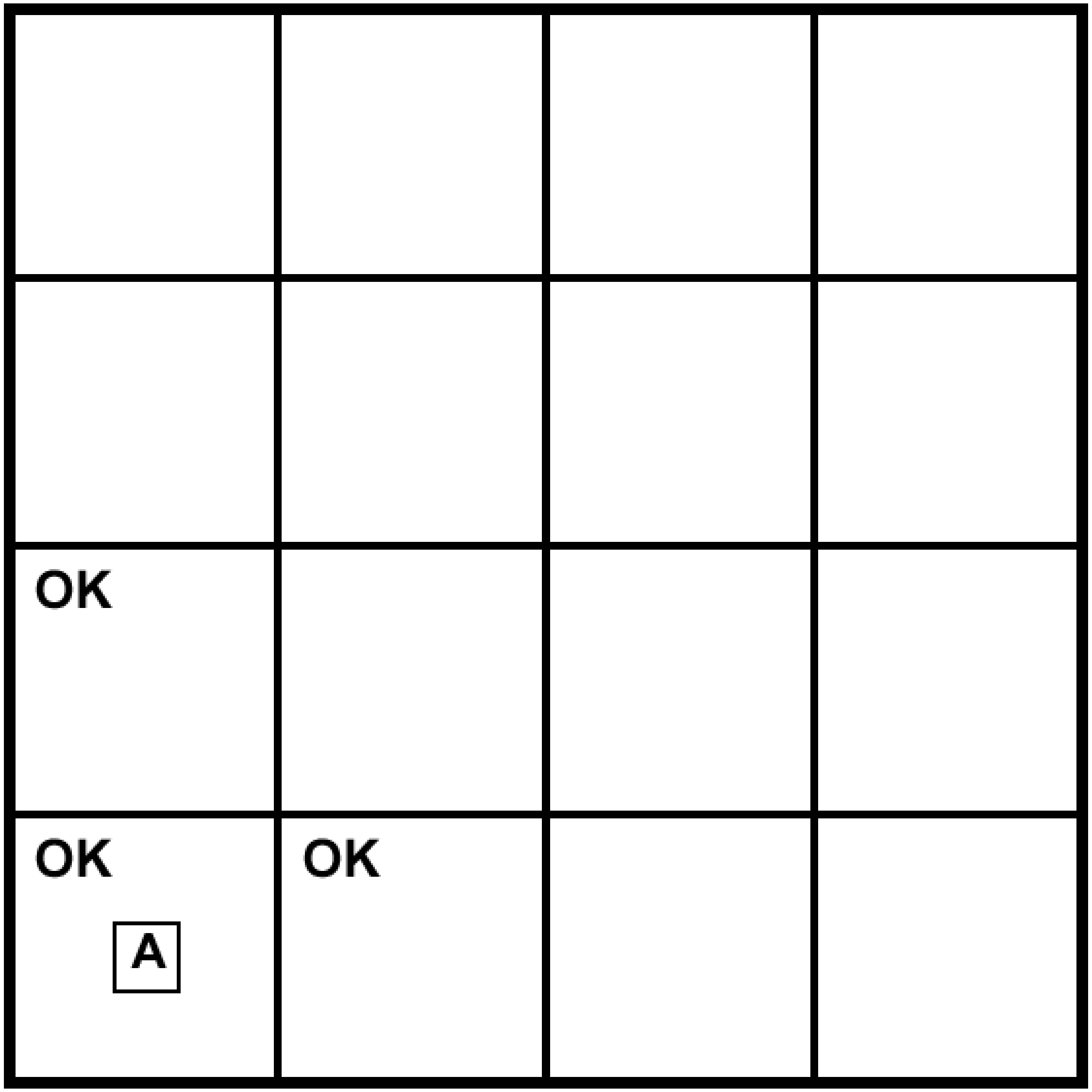
Wumpus world characterization

- Deterministic?
- Accessible?
- Static?
- Discrete?
- Episodic?

Wumpus world characterization

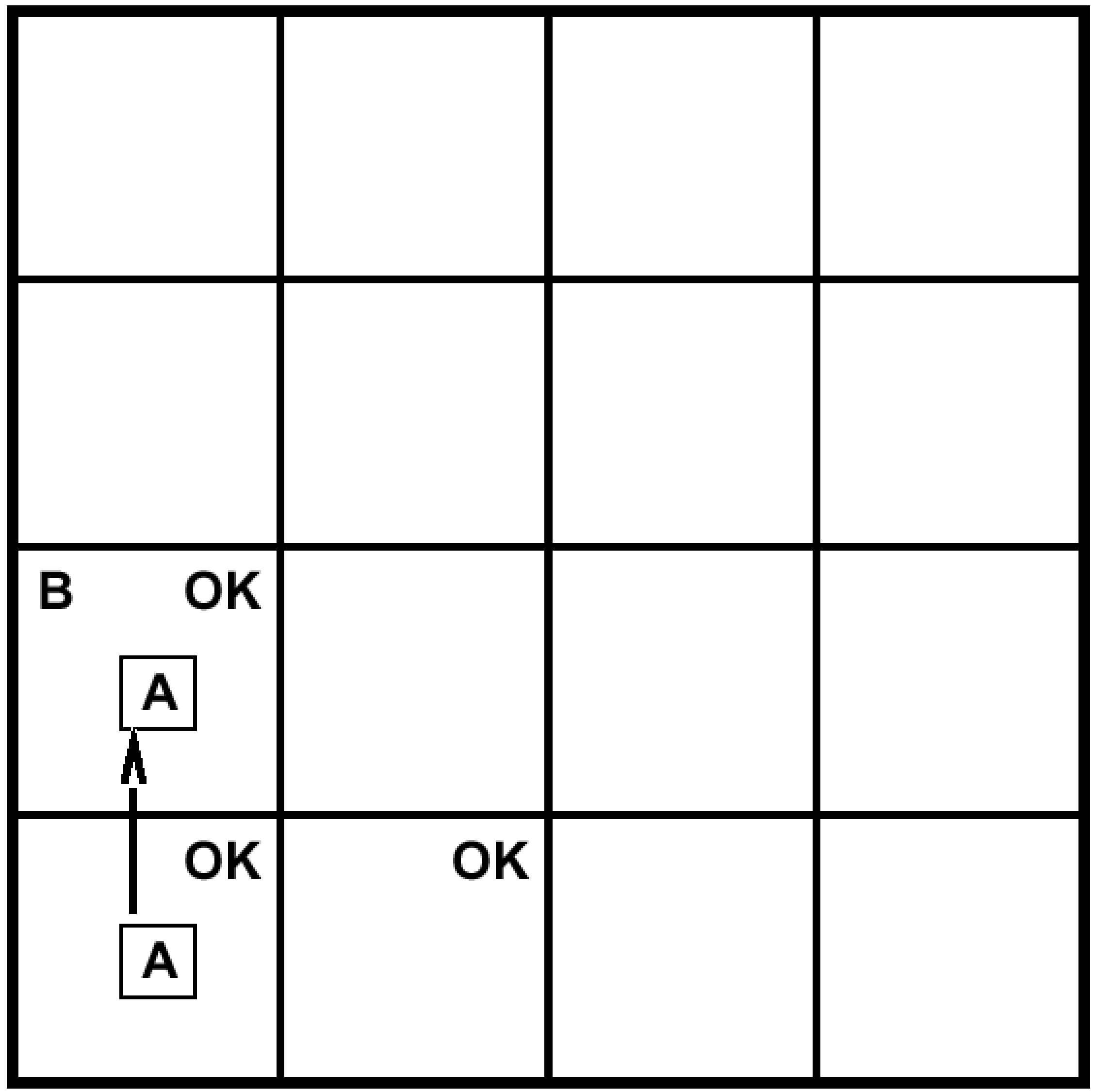
• Deterministic?	Yes – outcome exactly specified.
• Accessible?	No – not fully observable, only local perception.
• Static?	Yes – Wumpus and pits do not move.
• Discrete?	Yes
• Episodic?	(Yes) – because static.

Exploring a Wumpus world



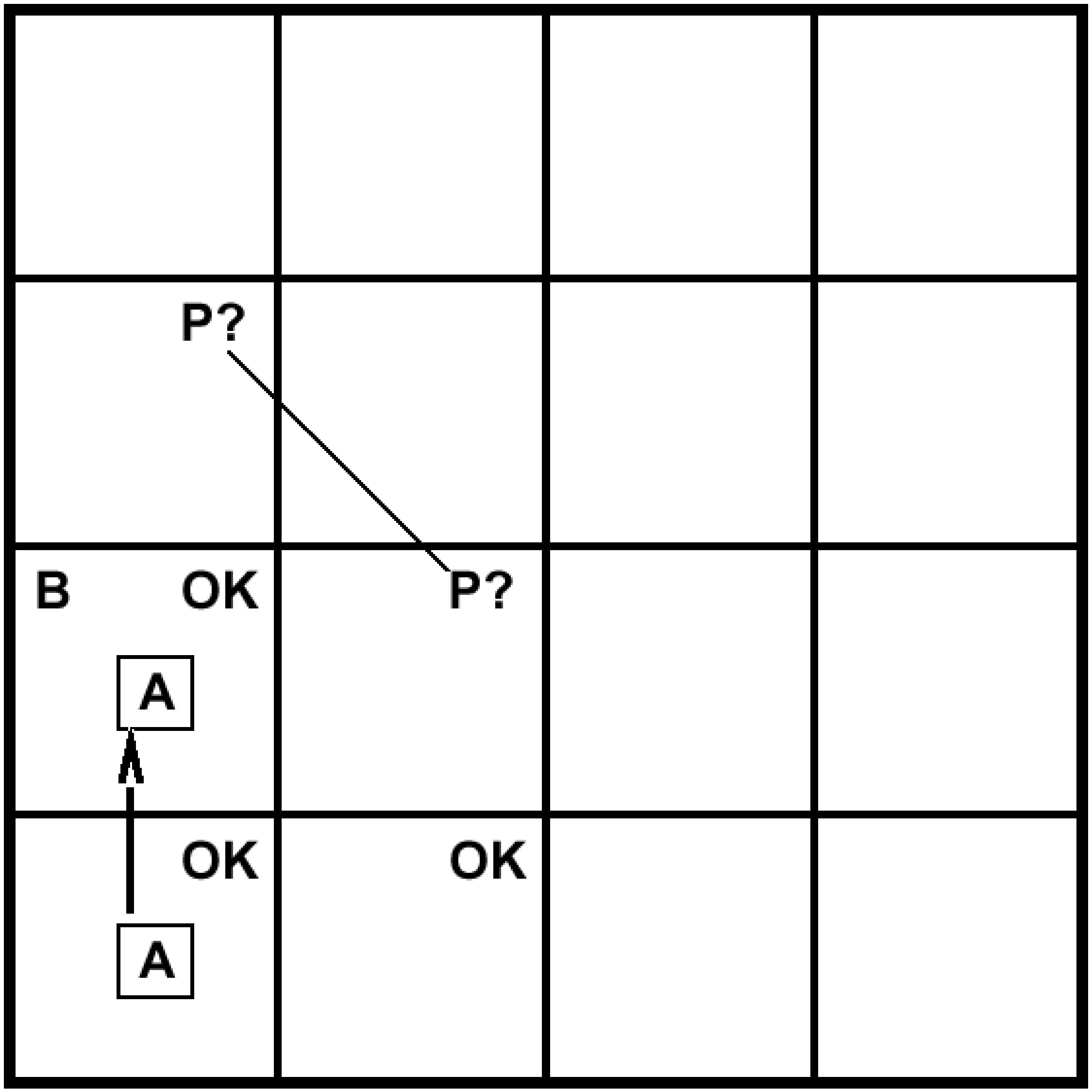
A= Agent
B= Breeze
S= Smell
P= Pit
W= Wumpus
OK = Safe
V = Visited
G = Glitter

Exploring a Wumpus world



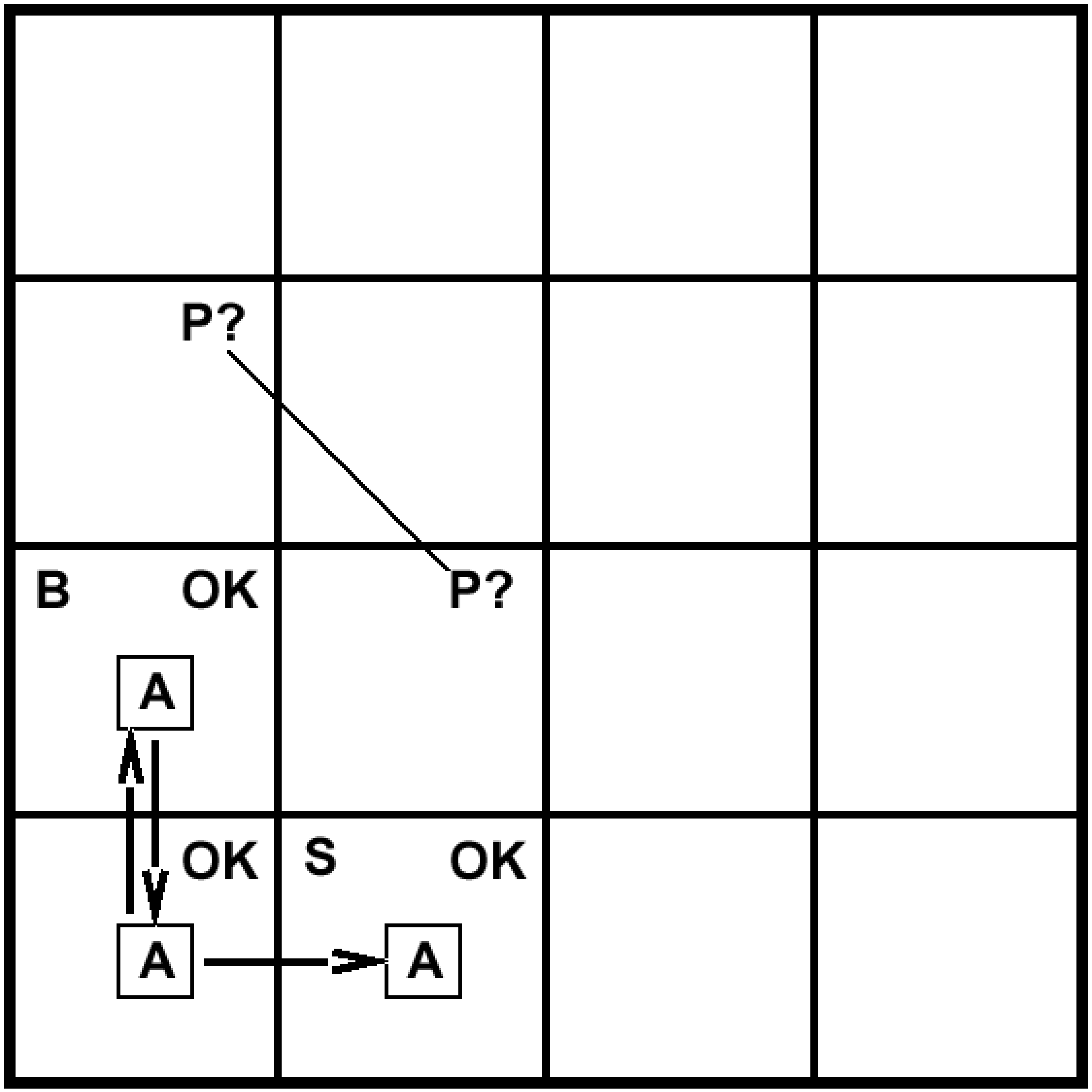
A= Agent
B= Breeze
S= Smell
P= Pit
W= Wumpus
OK = Safe
V = Visited
G = Glitter

Exploring a Wumpus world



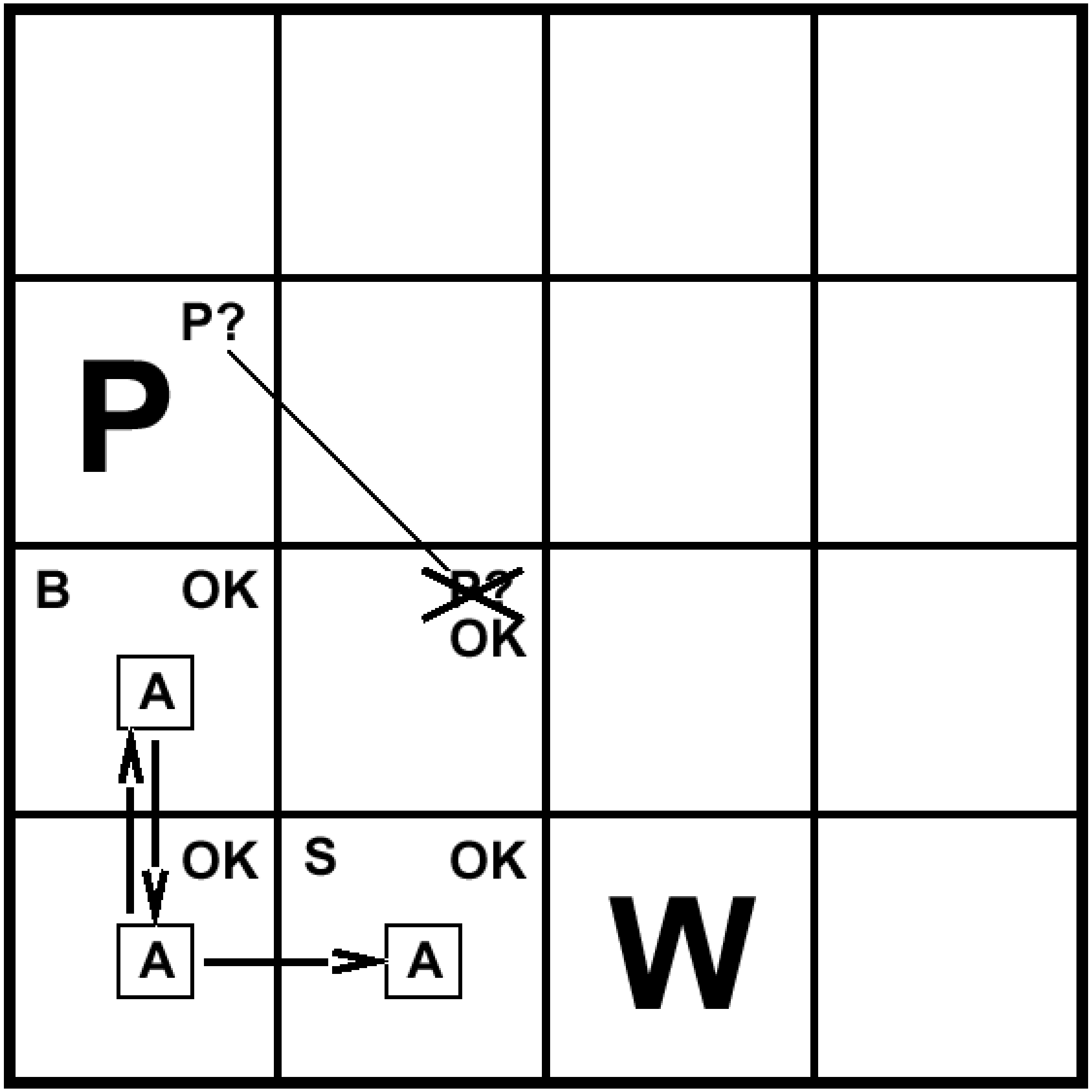
A= Agent
B= Breeze
S= Smell
P= Pit
W= Wumpus
OK = Safe
V = Visited
G = Glitter

Exploring a Wumpus world



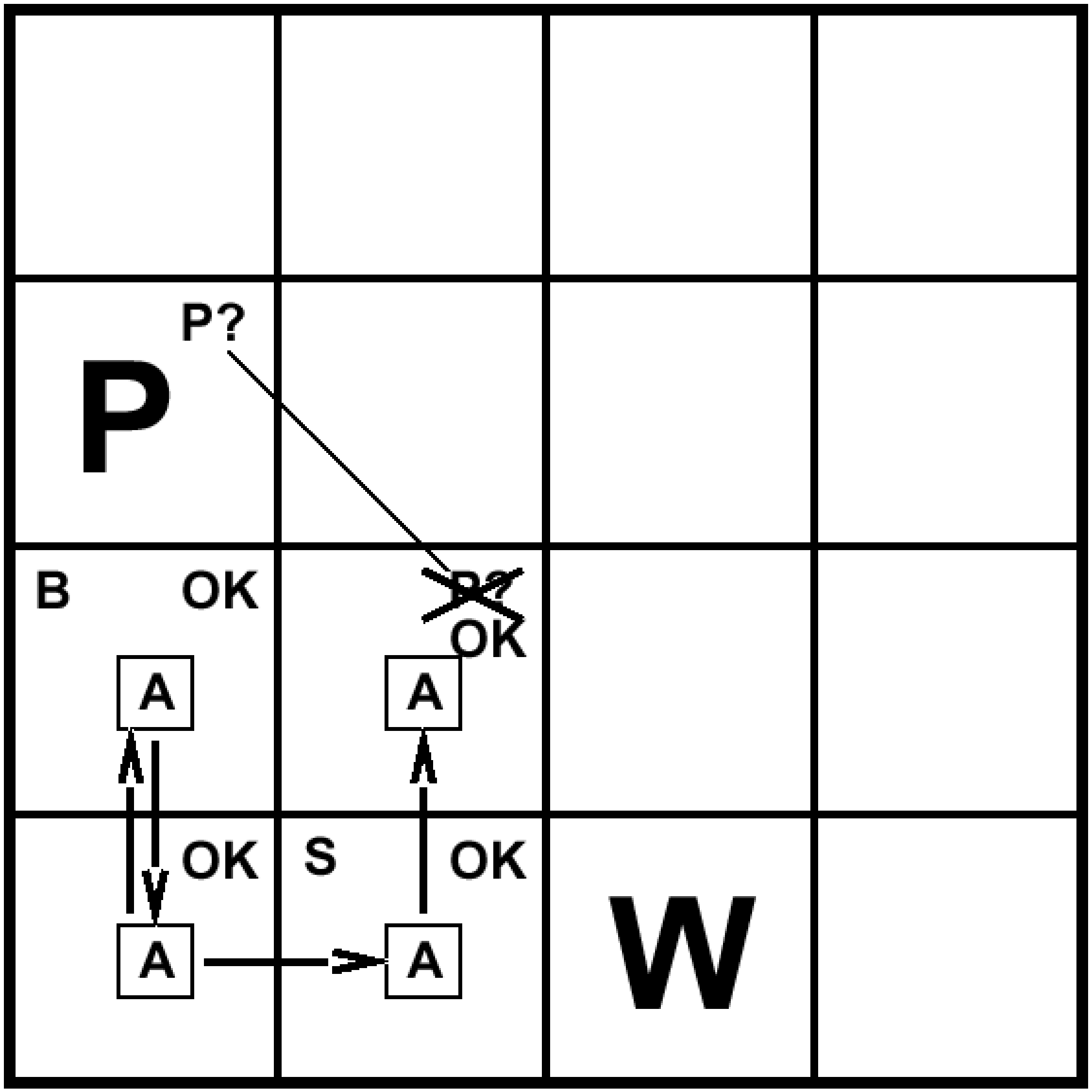
A= Agent
B= Breeze
S= Smell
P= Pit
W= Wumpus
OK = Safe
V = Visited
G = Glitter

Exploring a Wumpus world



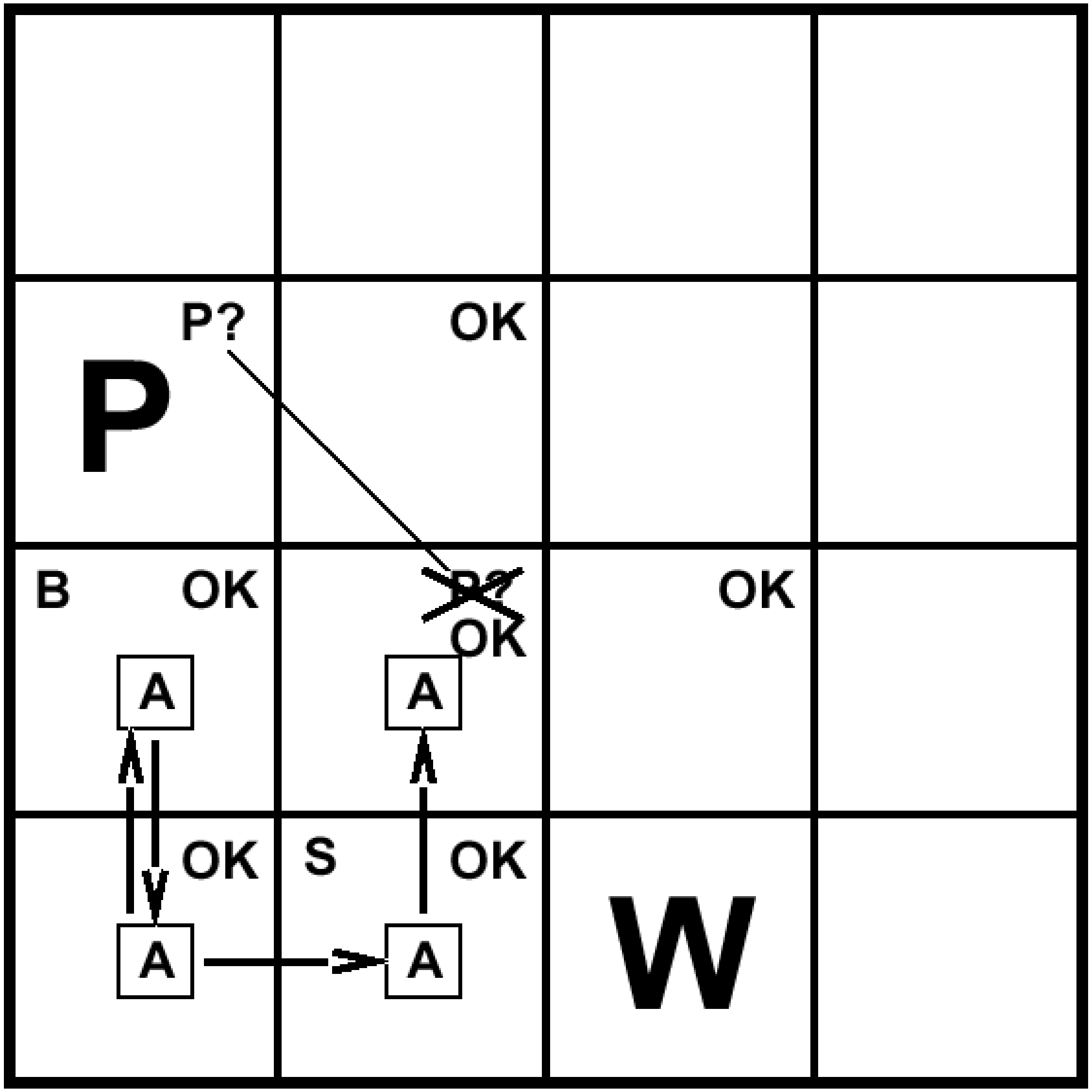
A= Agent
B= Breeze
S= Smell
P= Pit
W= Wumpus
OK = Safe
V = Visited
G = Glitter

Exploring a Wumpus world



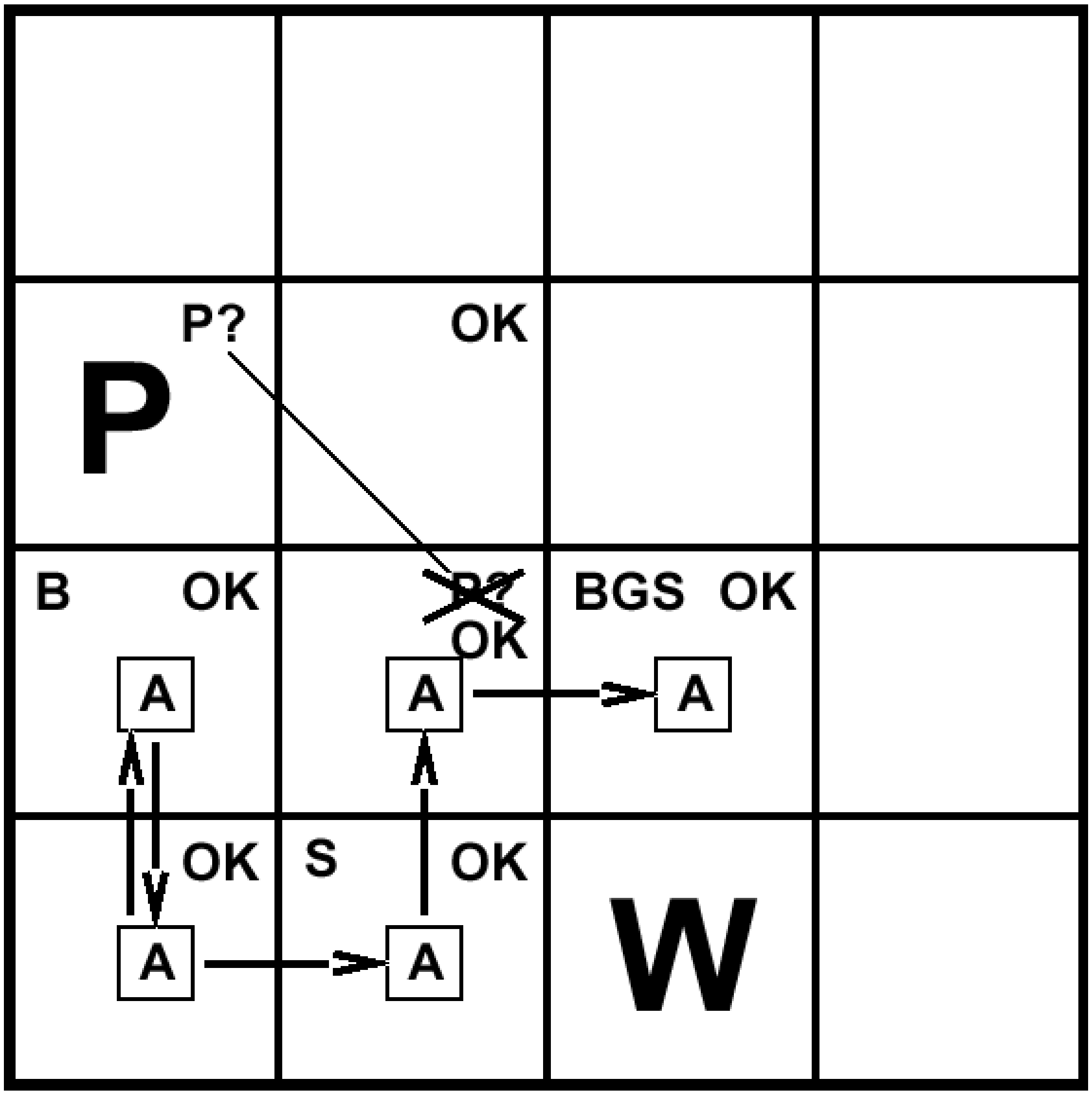
A= Agent
B= Breeze
S= Smell
P= Pit
W= Wumpus
OK = Safe
V = Visited
G = Glitter

Exploring a Wumpus world



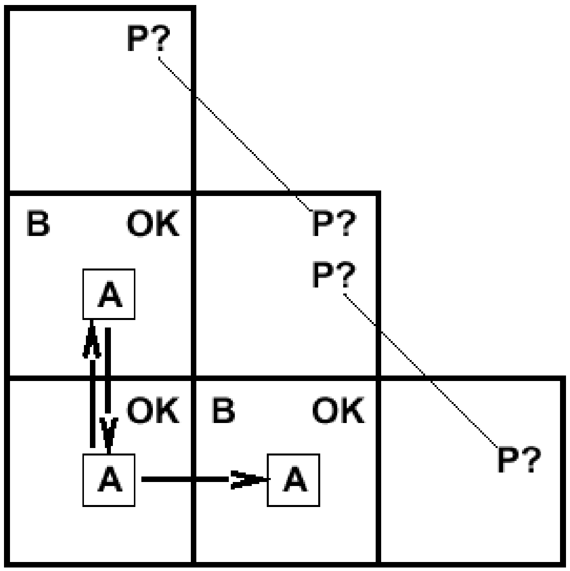
A= Agent
B= Breeze
S= Smell
P= Pit
W= Wumpus
OK = Safe
V = Visited
G = Glitter

Exploring a Wumpus world



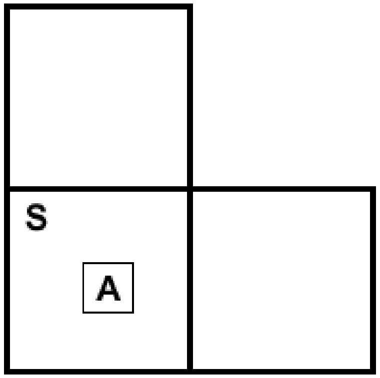
A= Agent
B= Breeze
S= Smell
P= Pit
W= Wumpus
OK = Safe
V = Visited
G = Glitter

Other tight spots



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

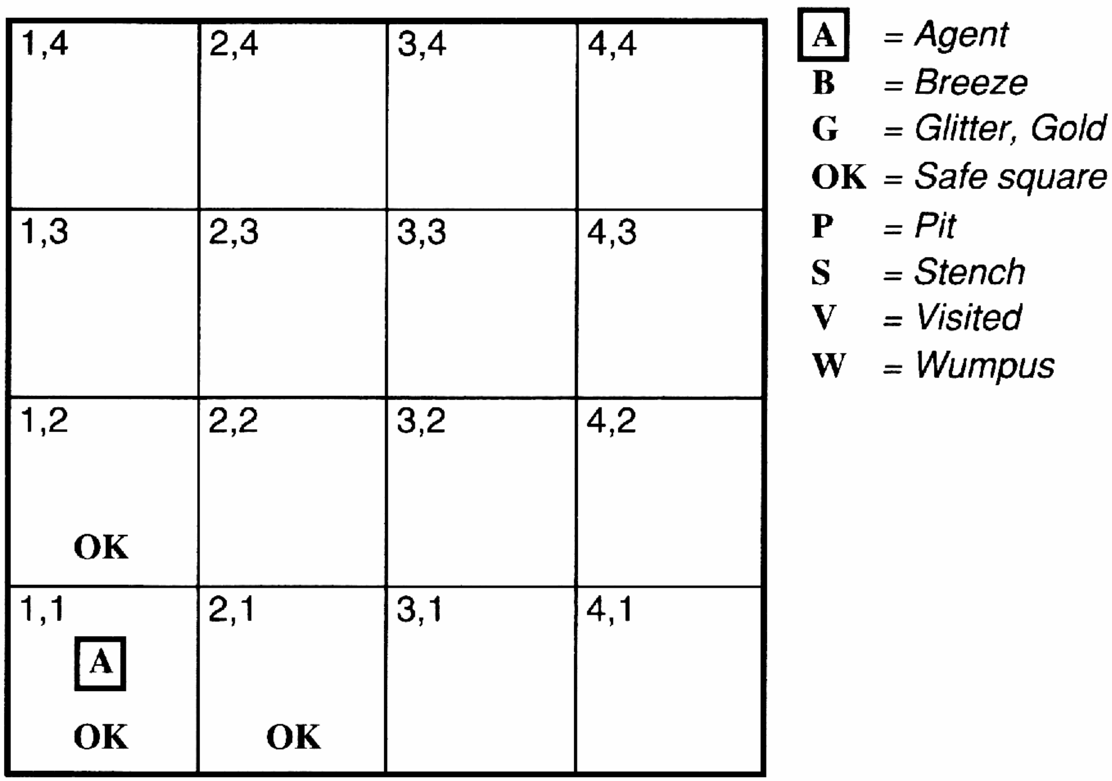
Assuming pits uniformly distributed,
(2,2) is most likely to have a pit

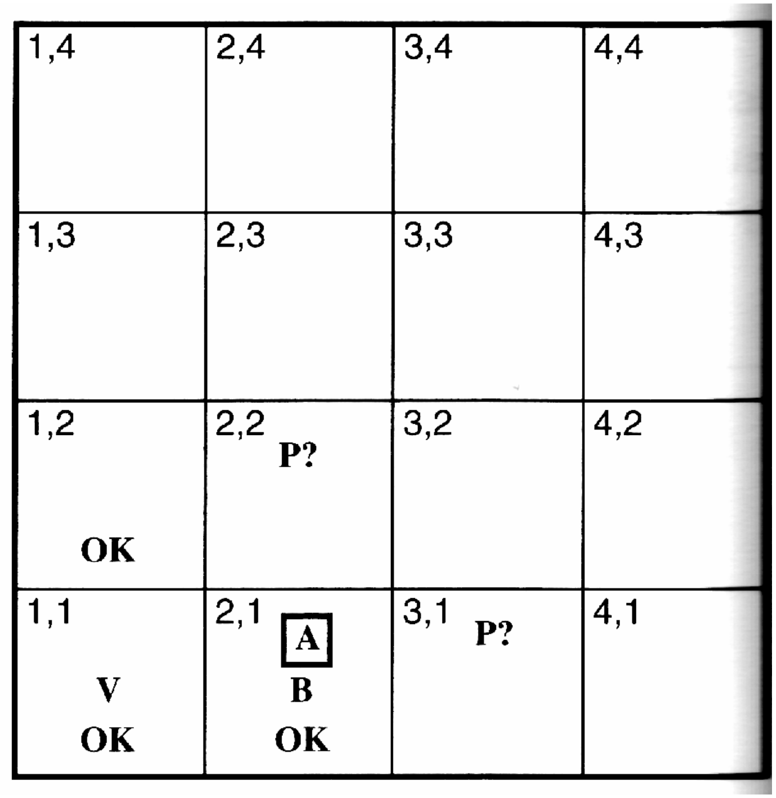


Smell in (1,1)
$\Rightarrow$ cannot move

Can use a strategy of <u>coercion</u>:
shoot straight ahead
wumpus was there $\Rightarrow$ dead $\Rightarrow$ safe
wumpus wasn't there $\Rightarrow$ safe

Another example solution





No perception → 1,2 and 2,1 OK

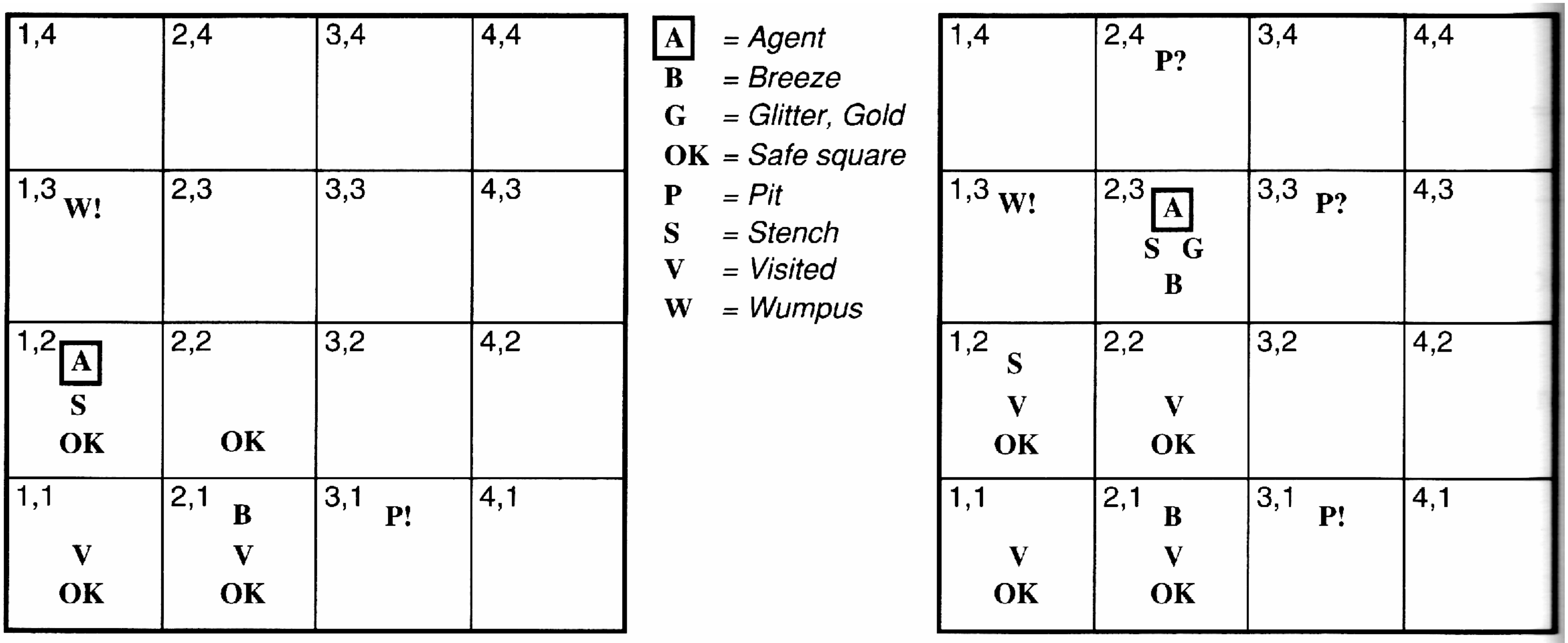
Move to 2,1

B in 2,1 → 2,2 or 3,1 P?

1,1 V → no P in 1,1

Move to 1,2 (only option)

Example solution



S in 1,2 → 1,3 or 2,2 has W

No S in 2,1 → 1,3 W

B in 2,1 and No B in 1,2 → 3,1 P

Translating Facts into Propositional Sentences

- Propositional calculus has ONLY symbols in the language
- Each symbol has 2 values ➔ true and false
- This severely limits what you can model
- Often clumsy to model big knowledge bases (KB) because you need to specify a very large number of facts
- Remember that EVERY FACT has to be explicitly modeled
- For example:
  - Wumpus is in location 2,1 ➔ W2,1 = true
  - But, you may also have to generate W1,1 = false, W1,3 = false, etc. to cover all the locations where there is no wumpus.
  - There is a smell in locations 1,2; 2,1; 3,2; and 2,3

    S1,2 = true; S2,1 = true; S3,2 = true; and S2,3 = true; all other locations have no smell, so they must be modeled explicitly as well:

    e.g., S1,3 = false
  - Note that all the above are just symbols – if you have 100 pieces of data, your KB will have 100 variables

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; $x2 + y >$ 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 $x = 0,\ y = 6$

Types of logic

Logics are characterized by what they commit to as "primitives"

Ontological commitment: what exists—facts? objects? time? beliefs?

Epistemological commitment: what states of knowledge?

Language	Ontological Commitment	Epistemological Commitment
Propositional logic	facts	true/false/unknown
First-order logic	facts, objects, relations	true/false/unknown
Temporal logic	facts, objects, relations, times	true/false/unknown
Probability theory	facts	degree of belief $0 \ldots 1$
Fuzzy logic	degree of truth	degree of belief $0 \ldots 1$

The Semantic Wall

Physical Symbol System

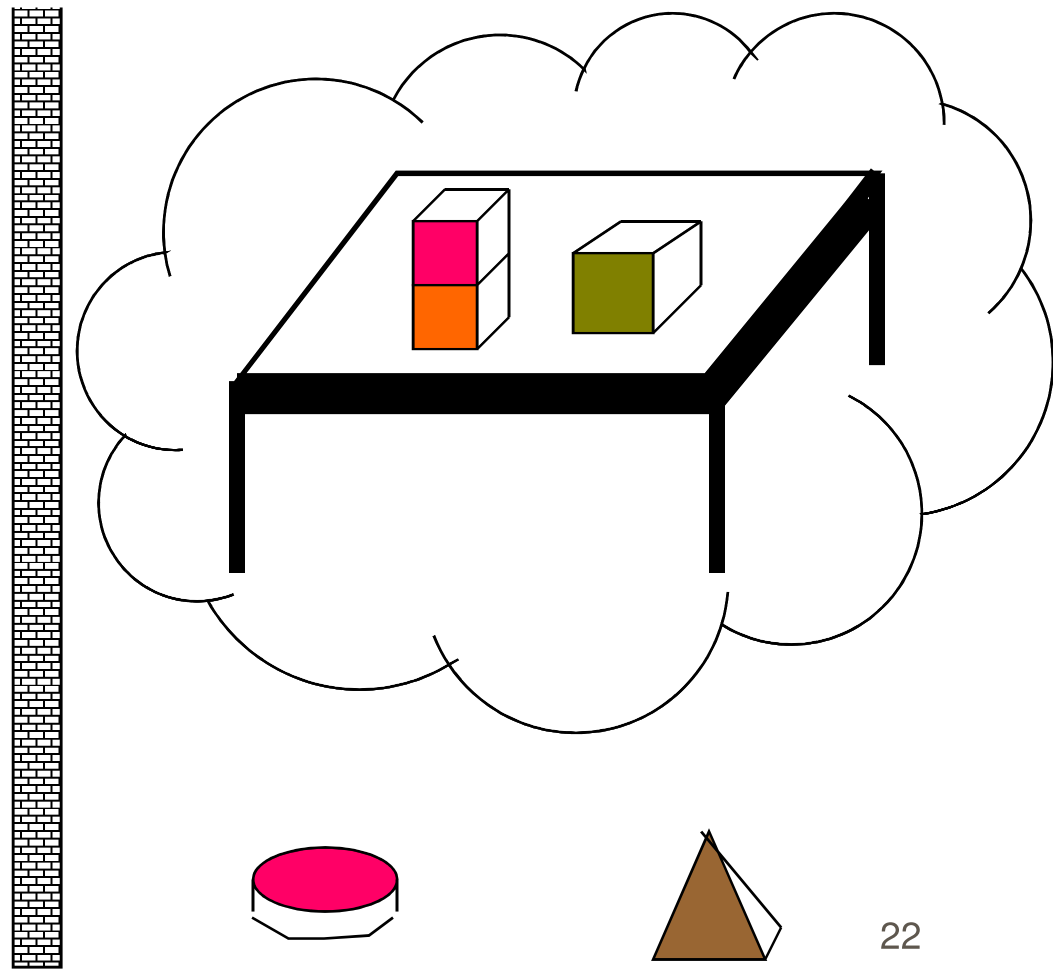
World

+BLOCKA+

+BLOCKB+

+BLOCKC+

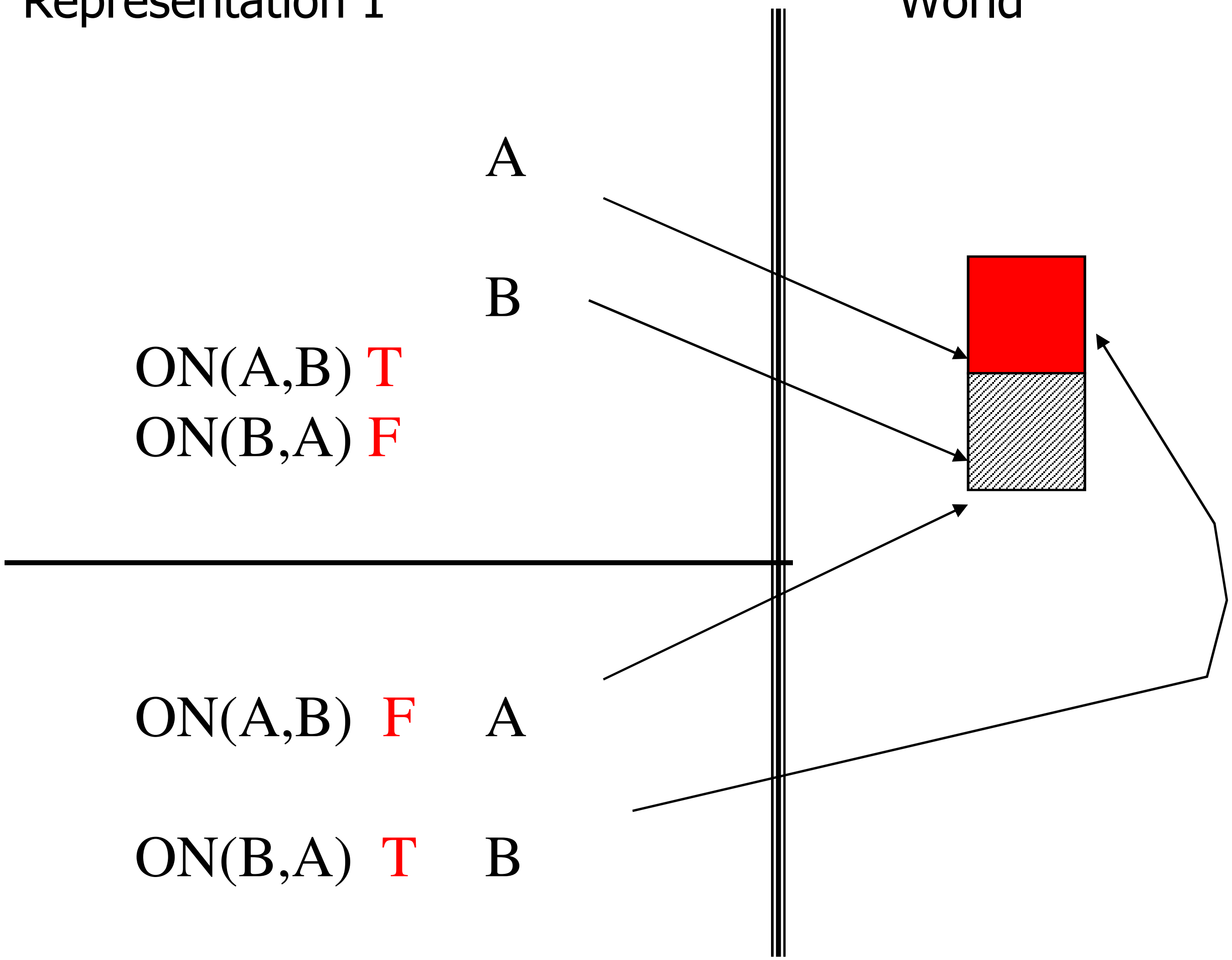
P_1:(IS_ON +BLOCKA+ +BLOCKB+)
P_2:((IS_RED +BLOCKA+)



Truth depends on Interpretation

Representation 1

World



Entailment

- **Entailment** means that truth of one sentence **follows from** the truth of other sentences:

$$KB \models \alpha$$

- 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**

Entailment is different from inference

Logic as a representation of the World

Representation: Sentences $\xrightarrow{\text{entails}}$ Sentence

Refers to
(Semantics)

World: Facts $\xrightarrow{\text{follows}}$ Fact

Models

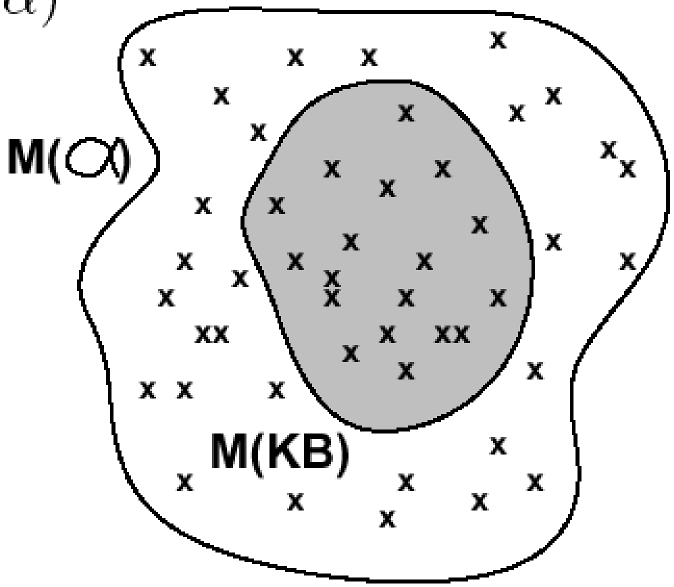
Logicians typically think in terms of <u>models</u>, which are formally structured worlds with respect to which truth can be evaluated

We say m is a <u>model</u> of a sentence α if α is true in m

$M(\alpha)$ is the set of all models of α

Then $KB \models \alpha$ if and only if $M(KB) \subseteq M(\alpha)$

E.g. $KB =$ Giants won and Reds won
$\alpha =$ Giants won

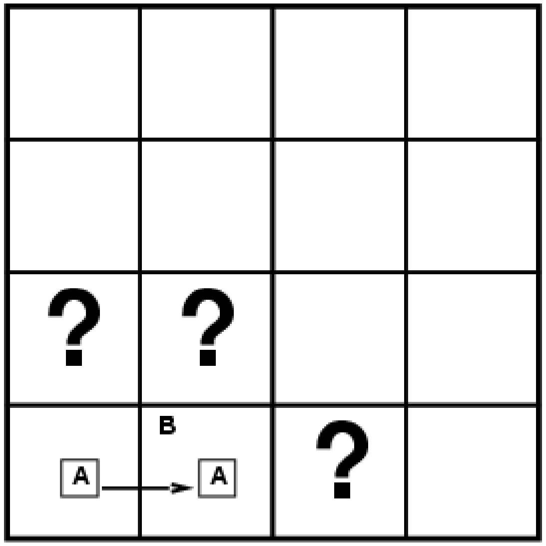


Entailment in the wumpus world

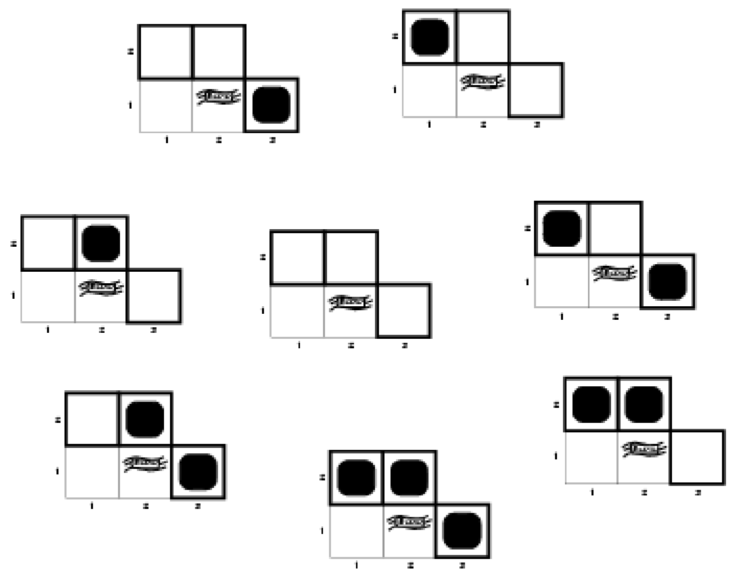
Situation after detecting nothing in [1,1], moving right, breeze in [2,1]

Consider possible models for *KB* assuming only pits

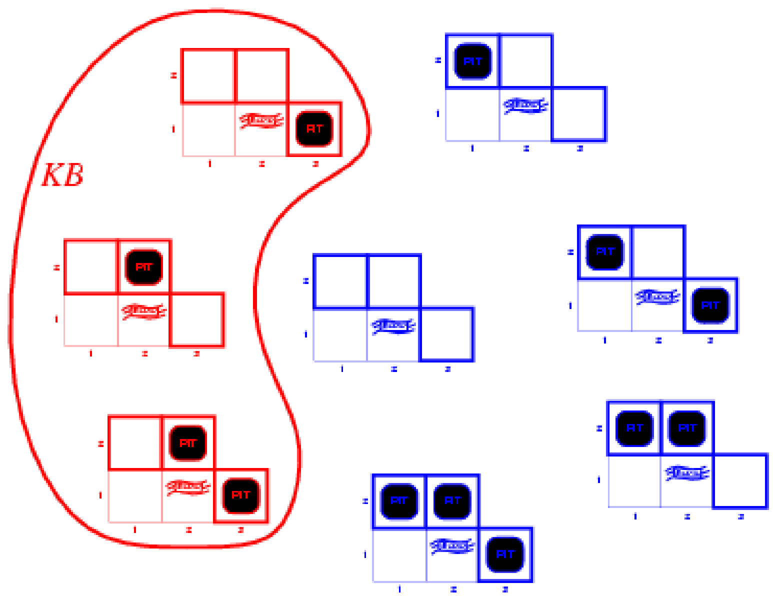
3 Boolean choices $\Rightarrow$ 8 possible models



Wumpus models

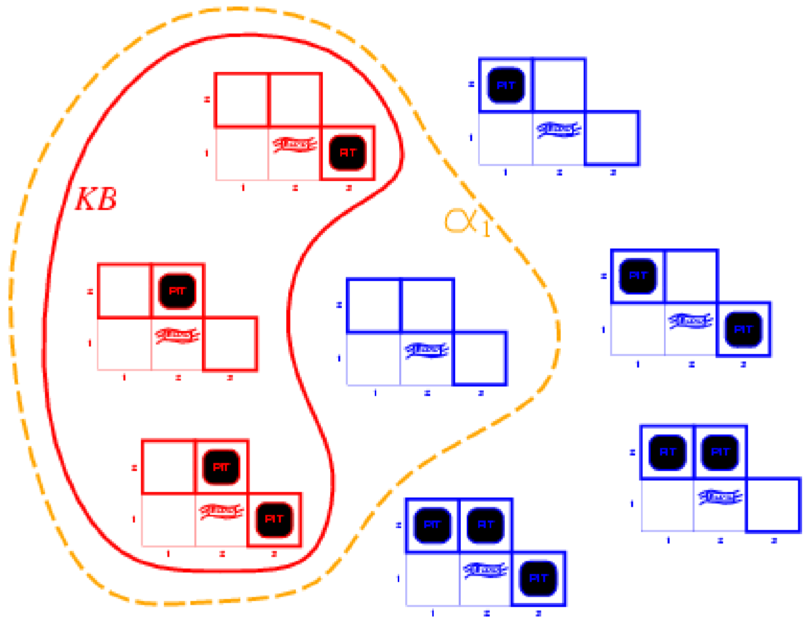


Wumpus models



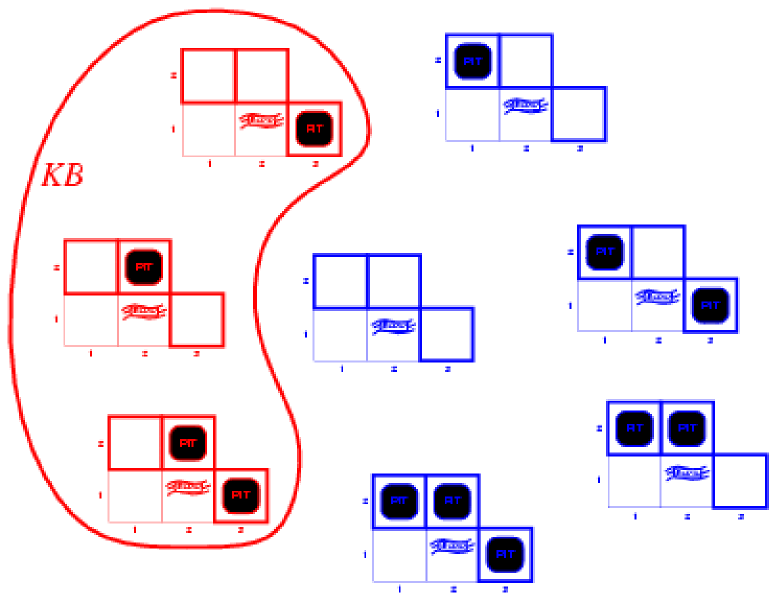
- *KB* = wumpus-world rules + observations

Wumpus models



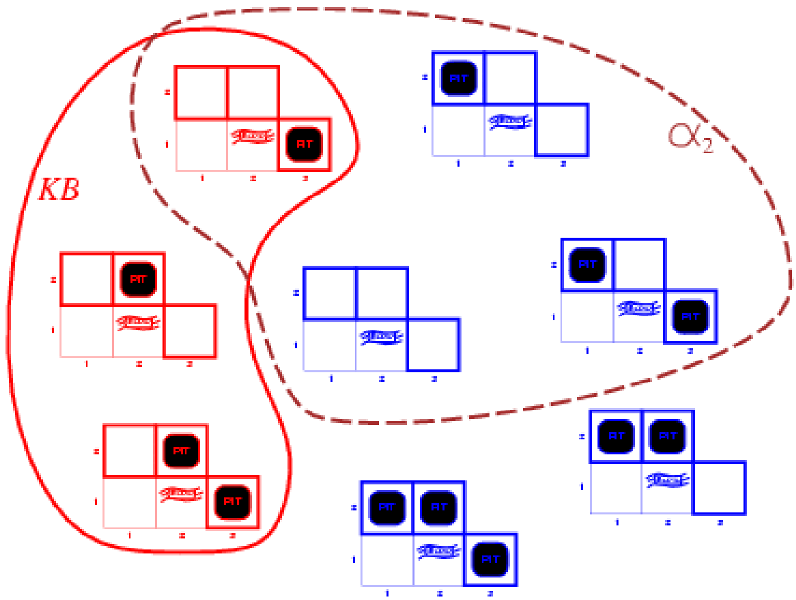
- *KB* = wumpus-world rules + observations
- α_1 = "[1,2] is safe", *KB* $\models \alpha_1$, proved by **model checking**
- Model Checking works only with FINITE worlds

Wumpus models



- *KB* = wumpus-world rules + observations

Wumpus models



- *KB* = wumpus-world rules + observations
- α_2 = "[2,2] is safe", KB does not entail α_2
- $KB \not\models \alpha_2$

Inference

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

<u>Soundness</u>: i is sound if

whenever $KB \vdash_i \alpha$, it is also true that $KB \models \alpha$

<u>Completeness</u>: 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.

Basic Symbols

- Expressions only evaluate to either "true" or "false."

• P	"P is true"	
• $\neg P$	"P is false"	negation
• $P \vee Q$	"either P is true or Q is true or both"	disjunction
• $P \wedge Q$	"both P and Q are true"	conjunction
• $P \rightarrow Q$	"if P is true, then Q is true"	implication
• $P \Leftrightarrow Q$	"P and Q are either both true or both false"	equivalence

Propositional logic: syntax

Propositional logic is the simplest logic

The proposition symbols P_1, P_2 etc are sentences

If S is a sentence, $\neg S$ is a sentence

If S_1 and S_2 is a sentence, $S_1 \wedge S_2$ is a sentence

If S_1 and S_2 is a sentence, $S_1 \vee S_2$ is a sentence

If S_1 and S_2 is a sentence, $S_1 \Rightarrow S_2$ is a sentence

If S_1 and S_2 is a sentence, $S_1 \Leftrightarrow S_2$ is a sentence

Propositional logic: semantics

Each model specifies true/false for each proposition symbol

E.g. A B C

$True$ $True$ $False$

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 <u>and</u>	S_2	is true
$S_1 \vee S_2$	is true iff	S_1	is true <u>or</u>	S_2	is true
$S_1 \Rightarrow S_2$	is true iff	S_1	is false <u>or</u>	S_2	is true
i.e.,	is false iff	S_1	is true <u>and</u>	S_2	is false
$S_1 \Leftrightarrow S_2$	is true iff	$S_1 \Rightarrow S_2$	is true <u>and</u>	$S_2 \Rightarrow S_1$	is true

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

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

Truth tables

- Truth value: whether a statement is true or false.
- Truth table: complete list of truth values for a statement given all possible values of the individual atomic expressions.

Example:

P	Q	P V Q
T	T	T
T	F	T
F	T	T
F	F	F

Truth tables for basic connectives

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

- $P \rightarrow Q$ is the same as (*not* P) *or* Q
- $P \leftrightarrow Q$ is the same as ($P \rightarrow Q$) *and* ($Q \rightarrow P$)

  ((not P) *or* Q) *and* ((not Q) *or* P)

Propositional logic: Logical Equivalence/Manipulation

$$
\begin{aligned}
(\alpha \land \beta) &\equiv (\beta \land \alpha) && \text{commutativity of } \land \\
(\alpha \lor \beta) &\equiv (\beta \lor \alpha) && \text{commutativity of } \lor \\
((\alpha \land \beta) \land \gamma) &\equiv (\alpha \land (\beta \land \gamma)) && \text{associativity of } \land \\
((\alpha \lor \beta) \lor \gamma) &\equiv (\alpha \lor (\beta \lor \gamma)) && \text{associativity of } \lor \\
\neg(\neg\alpha) &\equiv \alpha && \text{double-negation elimination} \\
(\alpha \Rightarrow \beta) &\equiv (\neg\beta \Rightarrow \neg\alpha) && \text{contraposition} \\
(\alpha \Rightarrow \beta) &\equiv (\neg\alpha \lor \beta) && \text{implication elimination} \\
(\alpha \Leftrightarrow \beta) &\equiv ((\alpha \Rightarrow \beta) \land (\beta \Rightarrow \alpha)) && \text{biconditional elimination} \\
\neg(\alpha \land \beta) &\equiv (\neg\alpha \lor \neg\beta) && \text{de Morgan} \\
\neg(\alpha \lor \beta) &\equiv (\neg\alpha \land \neg\beta) && \text{de Morgan} \\
(\alpha \land (\beta \lor \gamma)) &\equiv ((\alpha \land \beta) \lor (\alpha \land \gamma)) && \text{distributivity of } \land \text{ over } \lor \\
(\alpha \lor (\beta \land \gamma)) &\equiv ((\alpha \lor \beta) \land (\alpha \lor \gamma)) && \text{distributivity of } \lor \text{ over } \land
\end{aligned}
$$

Propositional inference: Enumeration / Model Checking Method

Let $\alpha = A \vee B$ and $KB = (A \vee C) \wedge (B \vee \neg C)$

Is it the case that $KB \models \alpha$?
Check all possible models—α must be true wherever KB is tı

A	B	C	$A \vee C$	$B \vee \neg C$	KB	α
$False$	$False$	$False$	$False$	$True$	$False$	$False$
$False$	$False$	$True$	$True$	$False$	$False$	$False$
$False$	$True$	$False$	$False$	$True$	$False$	$True$
$False$	$True$	$True$	$True$	$True$	$True$	$True$
$True$	$False$	$False$	$True$	$True$	$True$	$True$
$True$	$False$	$True$	$True$	$False$	$False$	$True$
$True$	$True$	$False$	$True$	$True$	$True$	$True$
$True$	$True$	$True$	$True$	$True$	$True$	$True$

Conclusion: KB |= α

Enumeration: Solution

A	B	C	$A \vee C$	$B \vee \neg C$	KB	α
$False$	$False$	$False$	$False$	$True$	$False$	$False$
$False$	$False$	$True$	$True$	$False$	$False$	$False$
$False$	$True$	$False$	$False$	$True$	$False$	$True$
$False$	$True$	$True$	$True$	$True$	$True$	$True$
$True$	$False$	$False$	$True$	$True$	$True$	$True$
$True$	$False$	$True$	$True$	$False$	$False$	$True$
$True$	$True$	$False$	$True$	$True$	$True$	$True$
$True$	$True$	$True$	$True$	$True$	$True$	$True$

Conclusion: KB $\models \alpha$

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"

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

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

Truth tables for inference

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

Inference by enumeration

- Depth-first enumeration of all models is sound and complete

```
function TT-ENTAILS?(KB, α) returns true or false
   symbols ← a list of the proposition symbols in KB and α
   return TT-CHECK-ALL(KB, α, symbols, [])
---------------------------------------------------------------
function TT-CHECK-ALL(KB, α, symbols, model) returns true or false
   if EMPTY?(symbols) then
       if PL-TRUE?(KB, model) then return PL-TRUE?(α, model)
       else return true
   else do
       P ← FIRST(symbols); rest ← REST(symbols)
       return TT-CHECK-ALL(KB, α, rest, EXTEND(P, true, model)) and
              TT-CHECK-ALL(KB, α, rest, EXTEND(P, false, model))
```

- For *n* symbols, time complexity is $O(2^n)$, space complexity is $O(n)$

Propositional inference: normal forms

Other approaches to inference use syntactic operations on sentences, often expressed in standardized forms

Conjunctive Normal Form (CNF—universal)

$$\textit{conjunction} \text{ of } \underbrace{\textit{disjunctions} \text{ of } \textit{literals}}_{\textit{clauses}}$$

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

"product of sums of simple variables or negated simple variables"

Disjunctive Normal Form (DNF—universal)

$$\textit{disjunction} \text{ of } \underbrace{\textit{conjunctions} \text{ of } \textit{literals}}_{\textit{terms}}$$

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

"sum of products of simple variables or negated simple variables"

Horn Form (restricted)

$\textit{conjunction}$ of $\textit{Horn clauses}$ (clauses with ≤ 1 positive literal)

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

Often written as set of implications:

$B \Rightarrow A$ and $(C \wedge D) \Rightarrow B$

Deriving expressions from functions

- Given a boolean function in truth table form, find a propositional logic expression for it that uses only V, ^ and ¬.
- **Idea:** We can easily do it by disjoining the "T" rows of the truth table.

Example: XOR function

P	Q	RESULT	
T	T	F	
T	F	T	P ^ (¬Q)
F	T	T	(¬P) ^ Q
F	F	F	

RESULT = (P ^ (¬Q)) V ((¬P) ^ Q)

A more formal approach

- To construct a logical expression in disjunctive normal form from a truth table:

- Build a **"minterm"** for each row of the table, where:

  - For each variable whose value is T in that row, include the variable in the minterm
  - For each variable whose value is F in that row, include the negation of the variable in the minterm
  - Link variables in minterm by conjunctions

- The expression consists of the disjunction of all minterms.

Example: adder with carry

Takes 3 variables in: x, y and ci (carry-in); yields 2 results: sum (s) and carry-out (co). To get you used to other notations, here we assume T = 1, F = 0, V = OR, ^ = AND, ¬ = NOT.

x	y	ci	co	s	
0	0	0	0	0	
0	0	1	0	1	s : NOT x AND NOT y AND ci
0	1	0	0	1	s : NOT x AND y AND NOT ci
0	1	1	1	0	co: NOT x AND y AND ci
1	0	0	0	1	s : x AND NOT y AND NOT ci
1	0	1	1	0	co: x AND NOT y AND ci
1	1	0	1	0	co: x AND y AND NOT ci
1	1	1	1	1	co,s: x AND y AND ci

The logical expression for co is:

(NOT x AND y AND ci) OR (x AND NOT y AND ci) OR
(x AND y AND NOT ci) OR (x AND y AND ci)

The logical expression for s is:

(NOT x AND NOT y AND ci) OR (NOT x AND y AND NOT ci)
OR (x AND NOT y AND NOT ci) OR (x AND y AND ci)

Validity and satisfiability

A sentence is valid if it is true in all models

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

Tautologies / Valid Expressions – true in all models

- Logical expressions that are always true. Can be simplified out.

Examples:

T

T V A

A V (¬A)

¬(A ^ (¬A))

A ⇔ A

((P V Q) ⇔ P) V (¬P ^ Q)

(P ⇔ Q) => (P => Q)

Proof methods

Proof methods divide into (roughly) two kinds:

<u>Model checking</u>

truth table enumeration (sound and complete for propositional)

heuristic search in model space (sound but incomplete)

e.g., the GSAT algorithm (Ex. 6.15)

<u>Application of inference rules</u>

Legitimate (sound) generation of new sentences from old

<u>Proof</u> = a sequence of inference rule applications

Can use inference rules as operators in a standard search alg.

Inference Rules – Part I

◇ **Modus Ponens** or **Implication-Elimination:** (From an implication and the premise of the implication, you can infer the conclusion.)

$$\frac{\alpha \Rightarrow \beta, \qquad \alpha}{\beta}$$

◇ **And-Elimination**: (From a conjunction, you can infer any of the conjuncts.)

$$\frac{\alpha_1 \wedge \alpha_2 \wedge \ldots \wedge \alpha_n}{\alpha_i}$$

◇ **And-Introduction**: (From a list of sentences, you can infer their conjunction.)

$$\frac{\alpha_1, \ \alpha_2, \quad \ldots, \quad \alpha_n}{\alpha_1 \wedge \alpha_2 \wedge \ldots \wedge \alpha_n}$$

◇ **Or-Introduction**: (From a sentence, you can infer its disjunction with anything else at all.)

$$\frac{\alpha_i}{\alpha_1 \vee \alpha_2 \vee \ldots \vee \alpha_n}$$

Inference Rules – Part II

◊ **Double-Negation Elimination**: (From a doubly negated sentence, you can infer a positive sentence.)

$$\frac{\neg\neg\alpha}{\alpha}$$

◊ **Unit Resolution:** (From a disjunction, if one of the disjuncts is false, then you can infer the other one is true.)

$$\frac{\alpha \vee \beta, \qquad \neg\beta}{\alpha}$$

◊ **Resolution**: (This is the most difficult. Because β cannot be both true and false, one of the other disjuncts must be true in one of the premises. Or equivalently, implication is transitive.)

$$\frac{\alpha \vee \beta, \qquad \neg\beta \vee \gamma}{\alpha \vee \gamma} \quad \text{or equivalently} \quad \frac{\neg\alpha \Rightarrow \beta, \qquad \beta \Rightarrow \gamma}{\neg\alpha \Rightarrow \gamma}$$