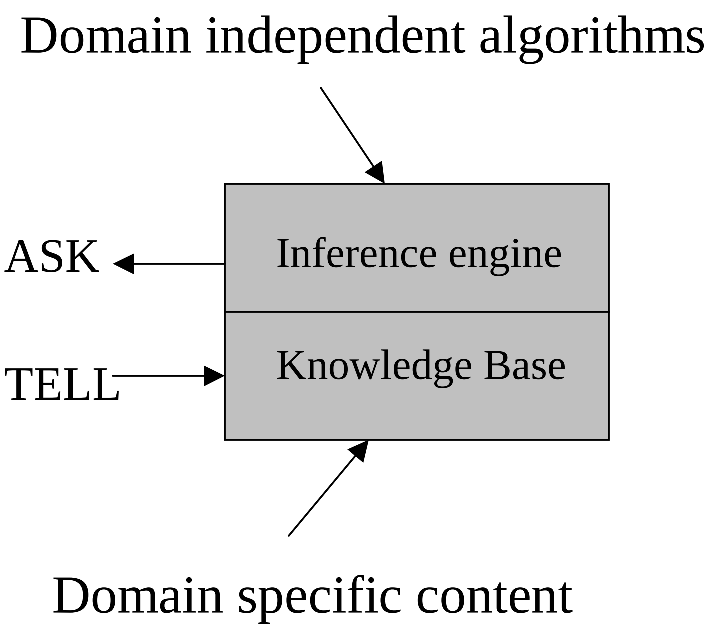
# Knowledge and reasoning – second part

- Knowledge representation
- Logic and representation
- Propositional (Boolean) logic
- Normal forms
- Inference in propositional logic
- Wumpus world example

# 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

# Generic knowledge-based agent

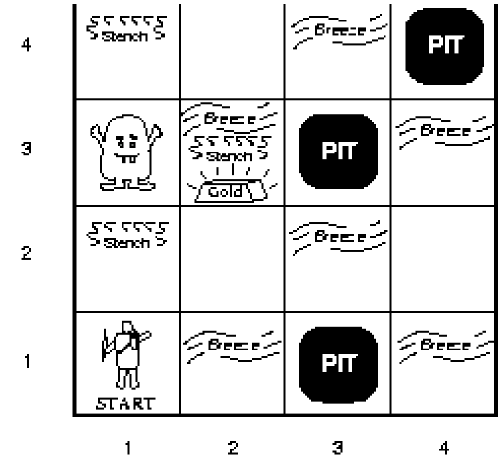
```
function KB-AGENT(percept) returns an action
   static: KB, a knowledge base
           t, a counter, initially 0, indicating time

   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
```

1. TELL KB what was perceived
   Uses a KRL to insert new sentences, representations of facts, into KB

2. ASK KB what to do.
   Uses logical reasoning to examine actions and select best.

# Wumpus world example




Percepts Breeze, Glitter, Smell

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

Goals Get gold back to start
without entering pit or wumpus square

Environment

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

# Wumpus world characterization

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

# Wumpus world characterization

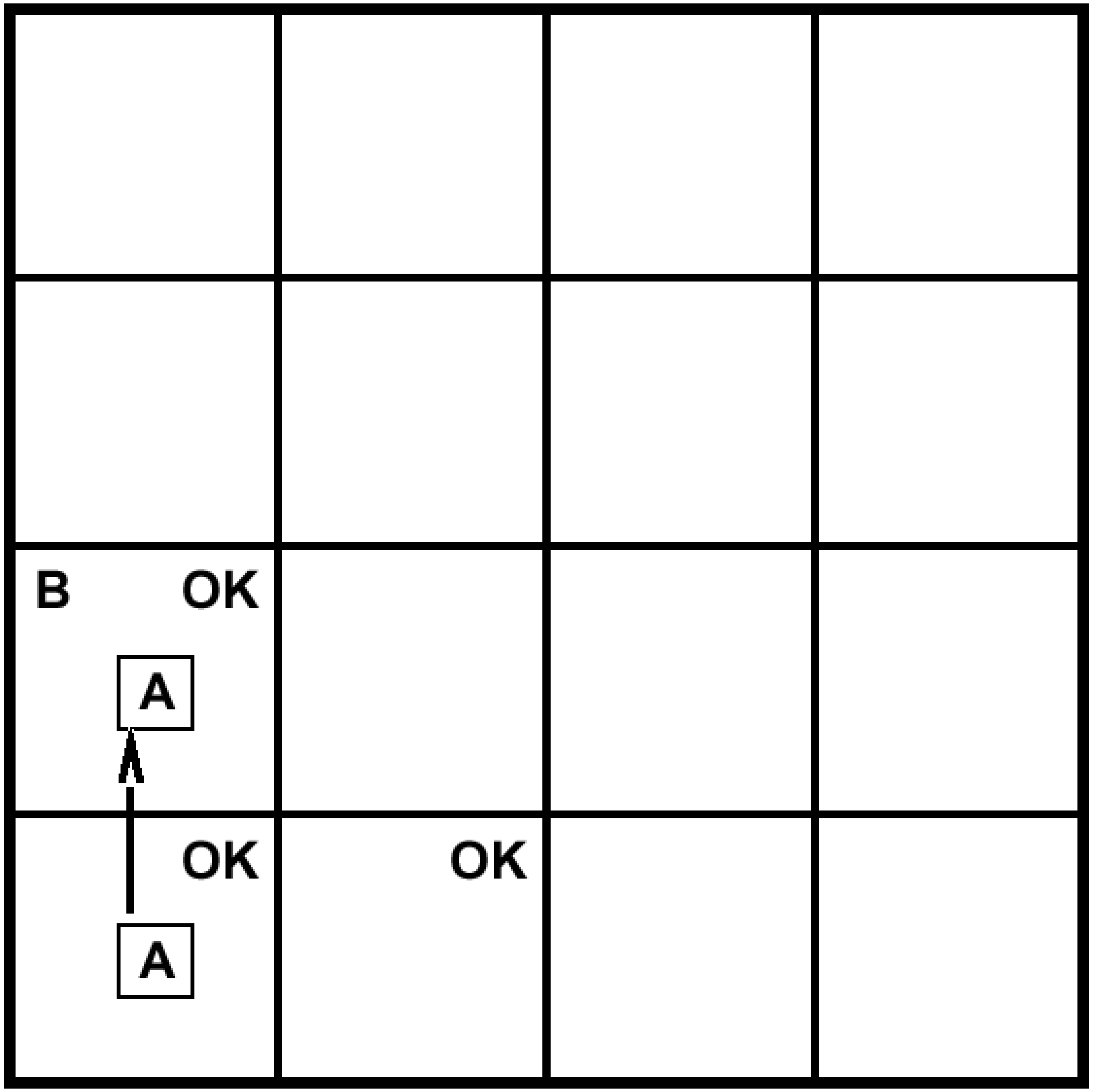
| | |
|---|---|
| • Deterministic? | Yes – outcome exactly specified. |
| • Accessible? | No – only local perception. |
| • Static? | Yes – Wumpus and pits do not move. |
| • Discrete? | Yes |
| • Episodic? | (Yes) – because static. |

# Exploring a Wumpus world

| | | | |
|---|---|---|---|
| | | | |
| | | | |
| OK | | | |
| OK<br>$\boxed{A}$ | OK | | |

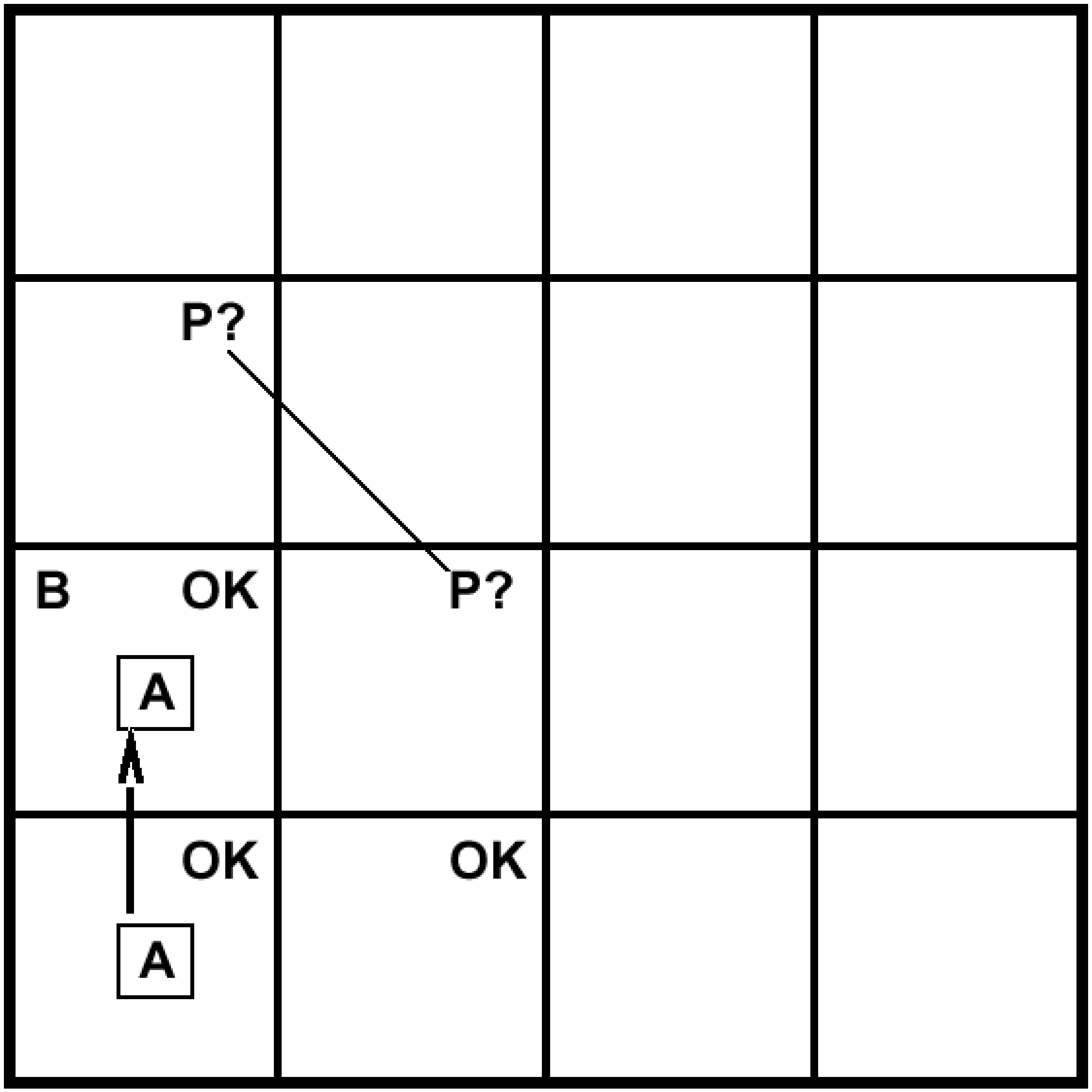
A= Agent
B= Breeze
S= Smell
P= Pit
W= Wumpus
OK = Safe
V = Visited
G = Glitter

# Exploring a Wumpus world



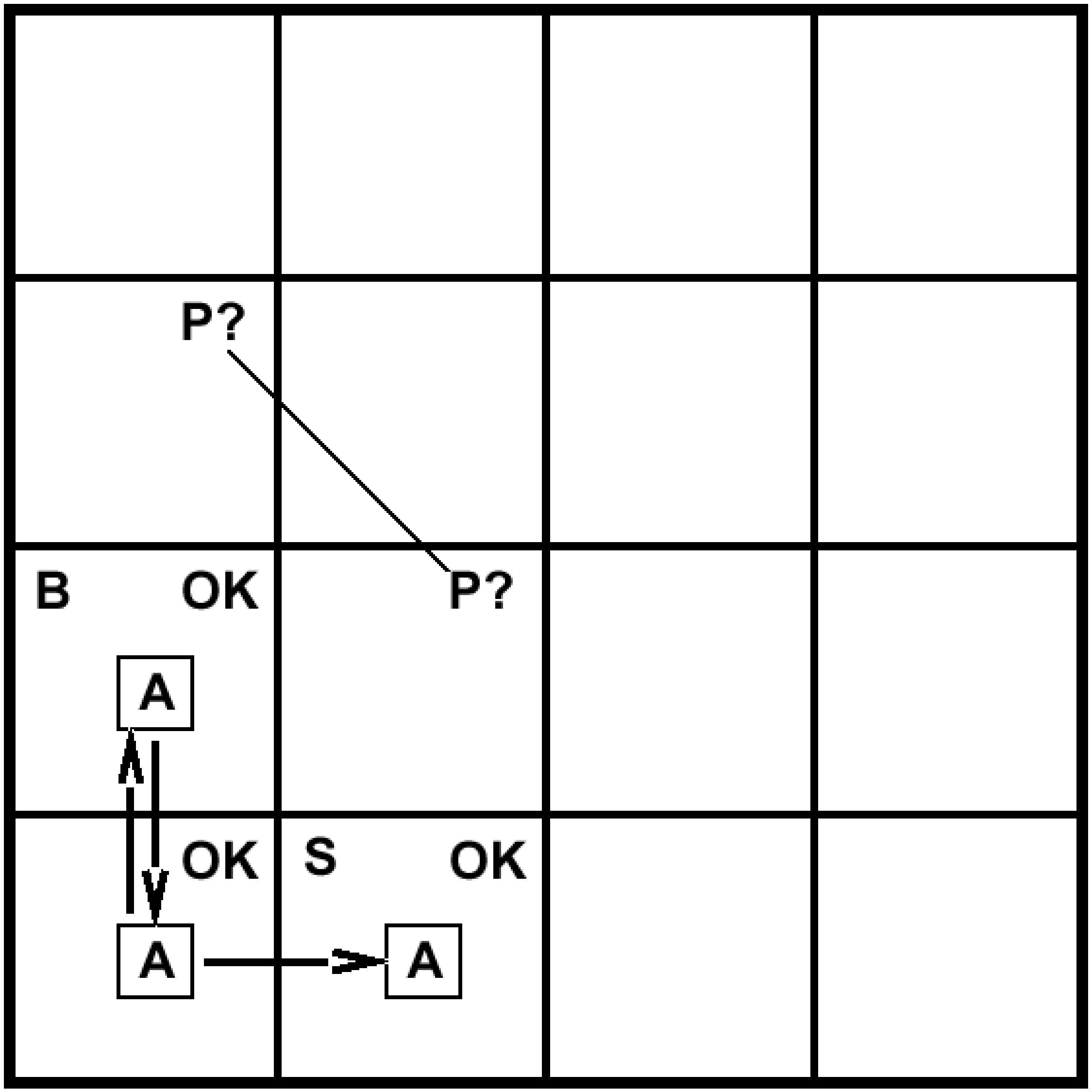

A= Agent
B= Breeze
S= Smell
P= Pit
W= Wumpus
OK = Safe
V = Visited
G = Glitter

# Exploring a Wumpus world



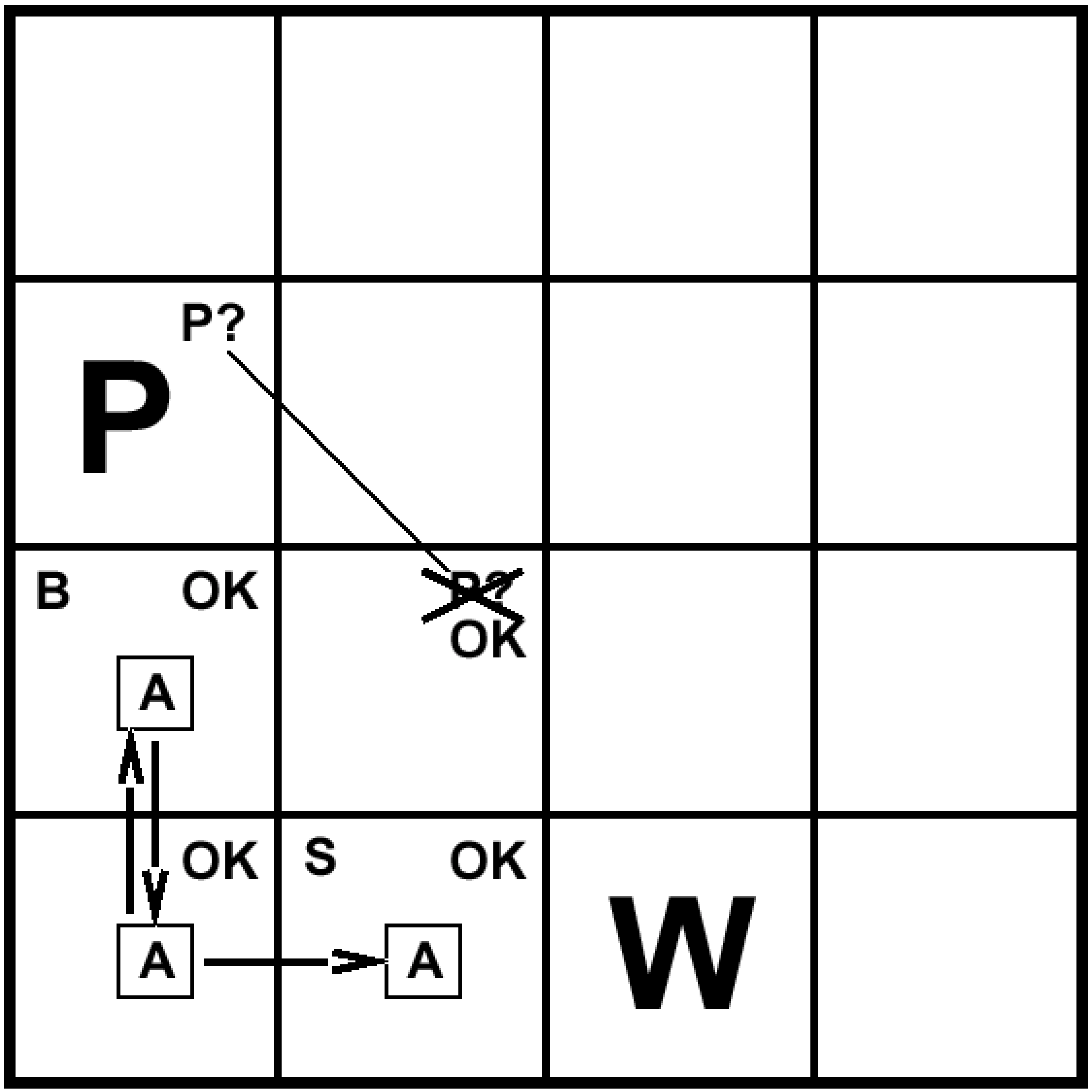

A= Agent
B= Breeze
S= Smell
P= Pit
W= Wumpus
OK = Safe
V = Visited
G = Glitter

# Exploring a Wumpus world



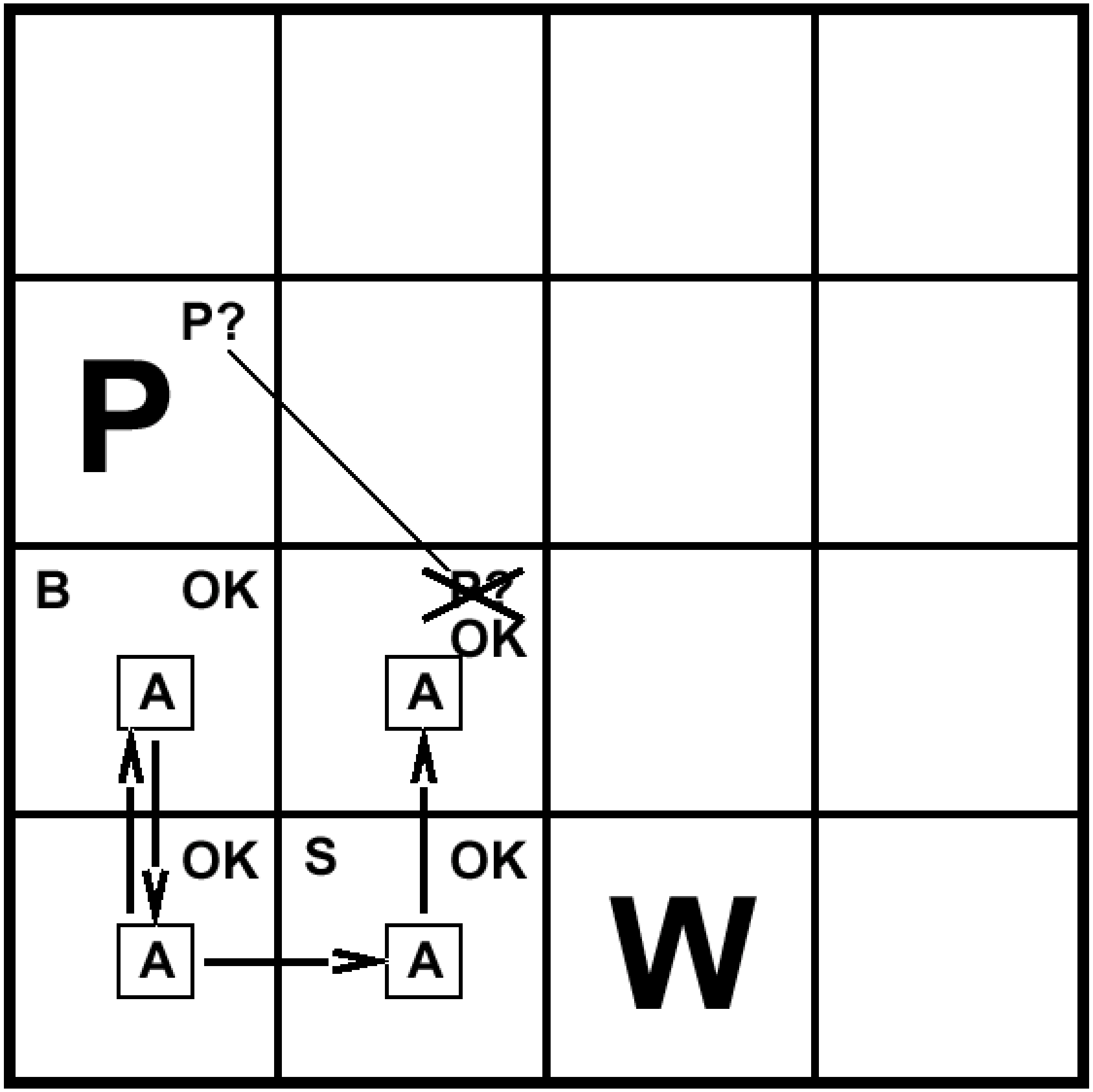

A= Agent
B= Breeze
S= Smell
P= Pit
W= Wumpus
OK = Safe
V = Visited
G = Glitter

# Exploring a Wumpus world



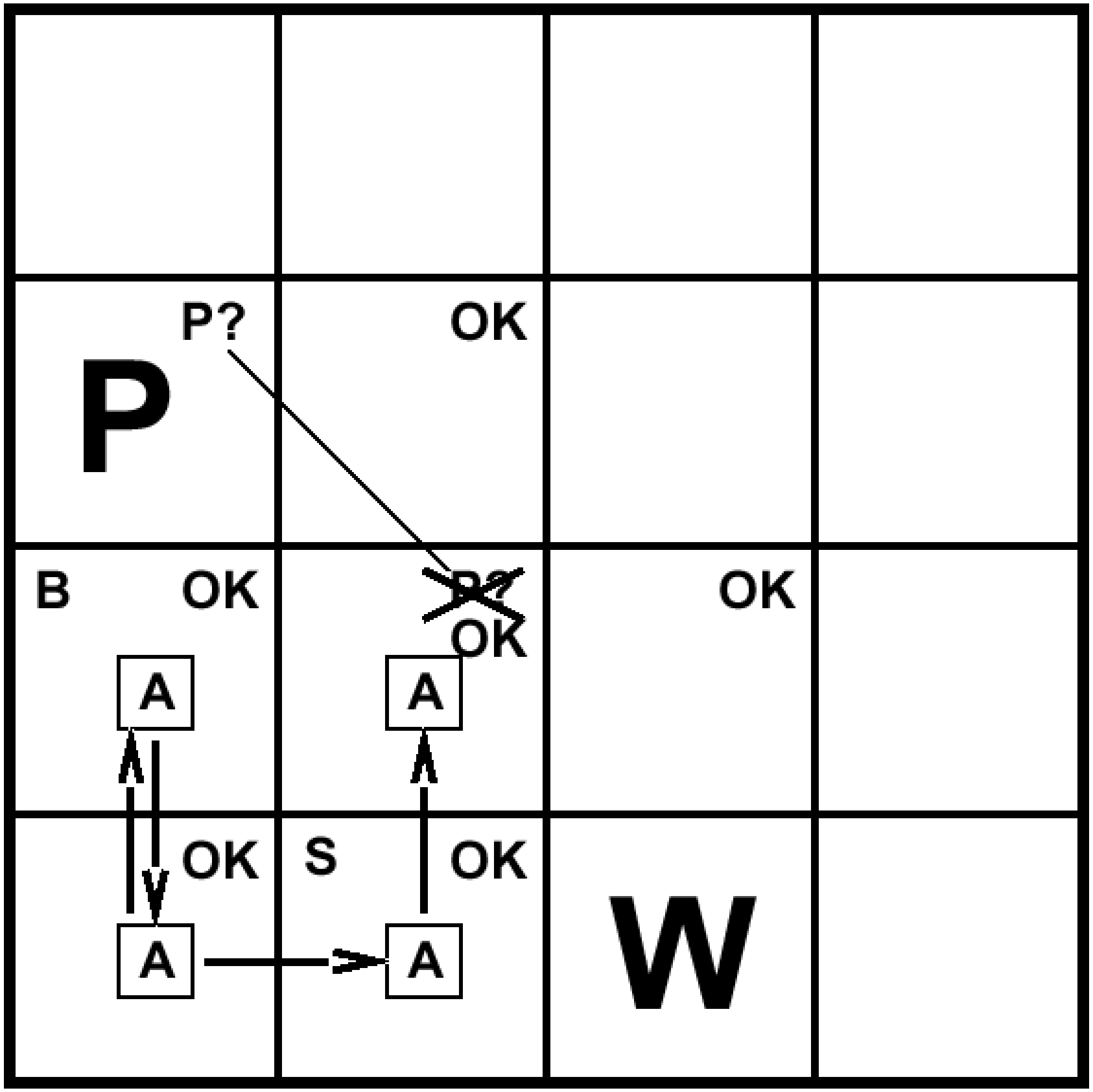

A= Agent
B= Breeze
S= Smell
P= Pit
W= Wumpus
OK = Safe
V = Visited
G = Glitter

# Exploring a Wumpus world



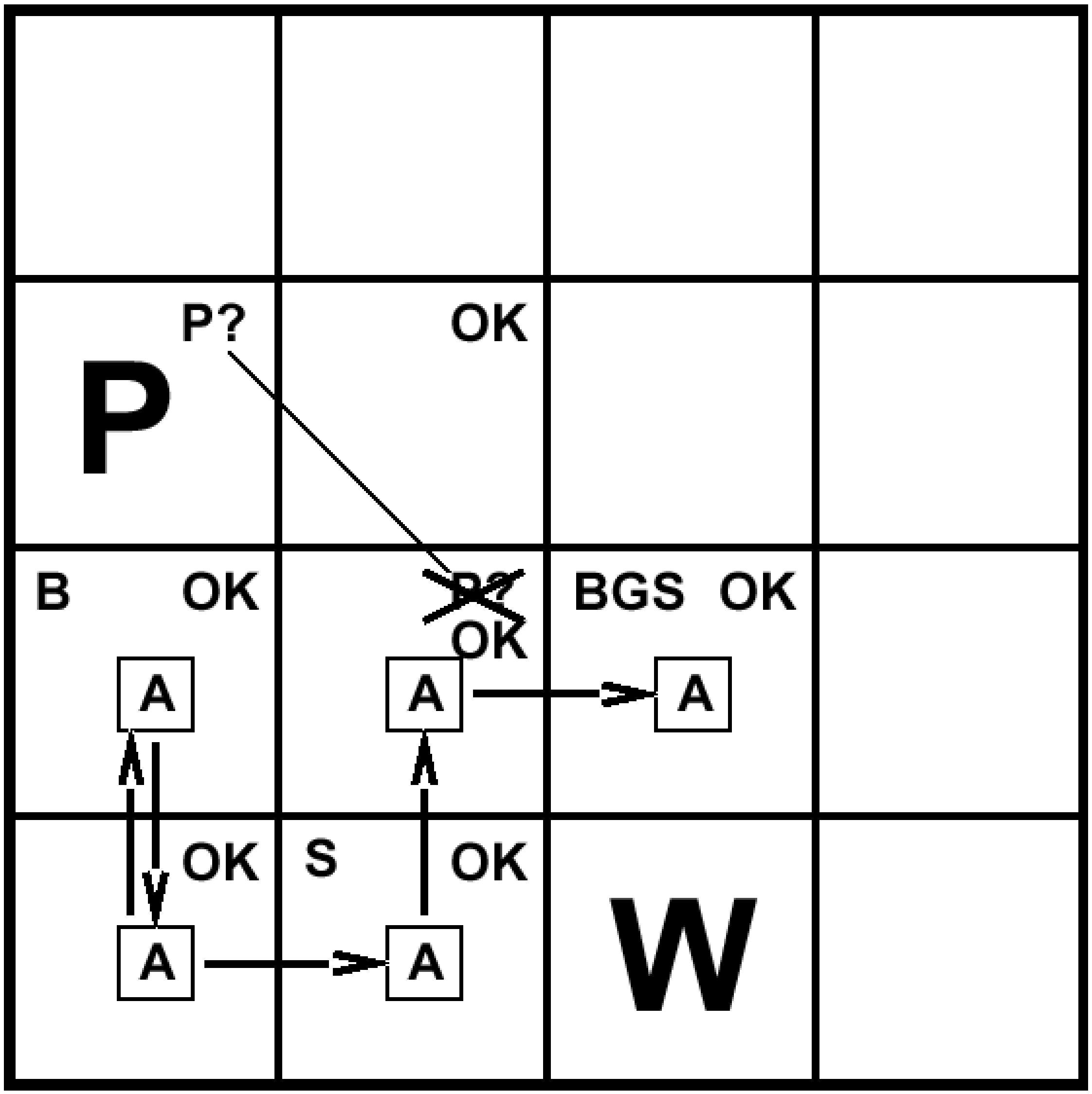

A= Agent
B= Breeze
S= Smell
P= Pit
W= Wumpus
OK = Safe
V = Visited
G = Glitter

# Exploring a Wumpus world



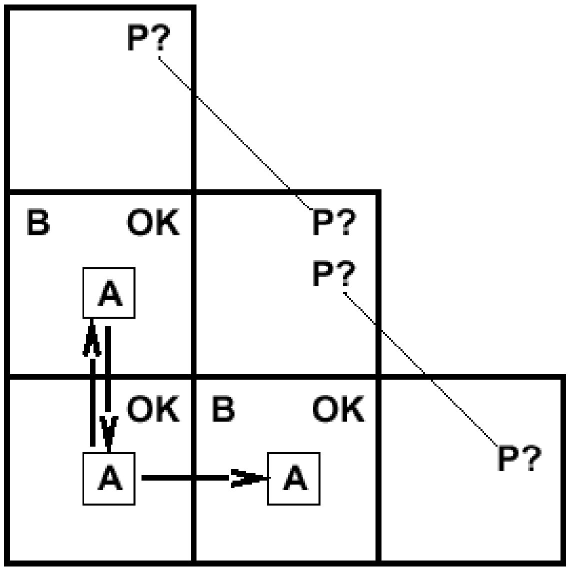

A= Agent
B= Breeze
S= Smell
P= Pit
W= Wumpus
OK = Safe
V = Visited
G = Glitter

# Exploring a Wumpus world



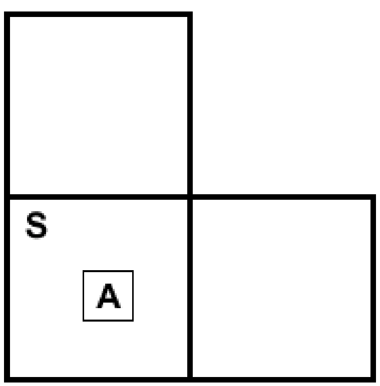

A= Agent
B= Breeze
S= Smell
P= Pit
W= Wumpus
OK = Safe
V = Visited
G = Glitter

# Other tight spots




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

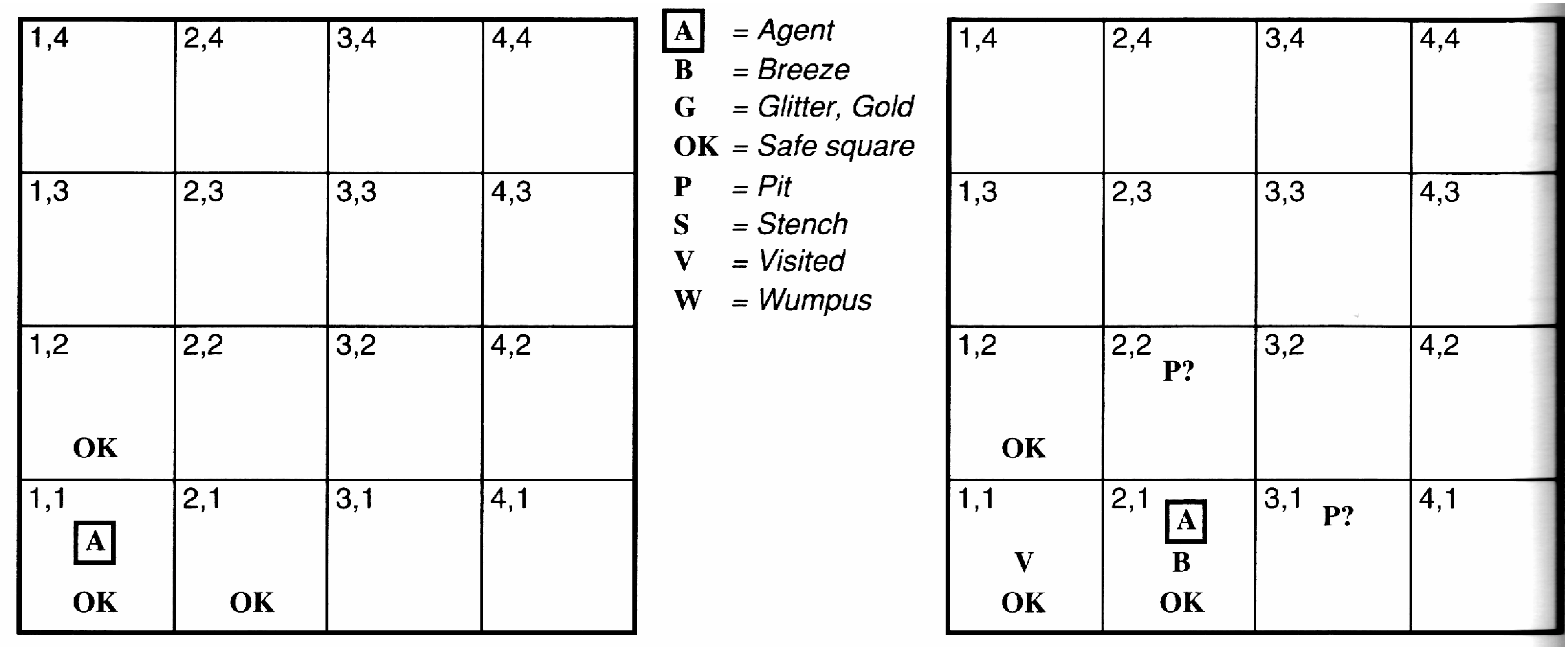
Assuming pits uniformly distributed,
(2,2) is most likely to have a pit




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

Can use a strategy of coercion:
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

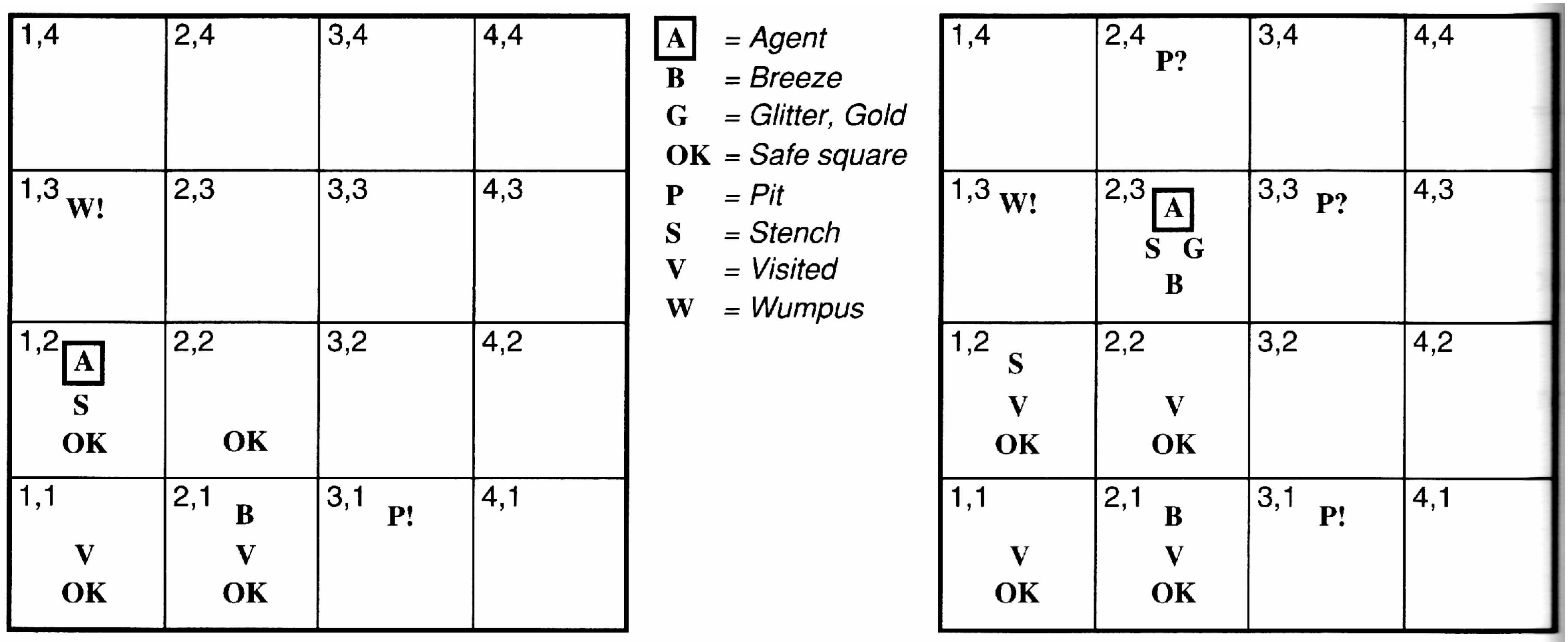
Move to 2,1

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 and No S when in 2,1 → 1,3 or 1,2 has W

1,2 OK → 1,3 W

No B in 1,2 → 2,2 OK & 3,1 P

## 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

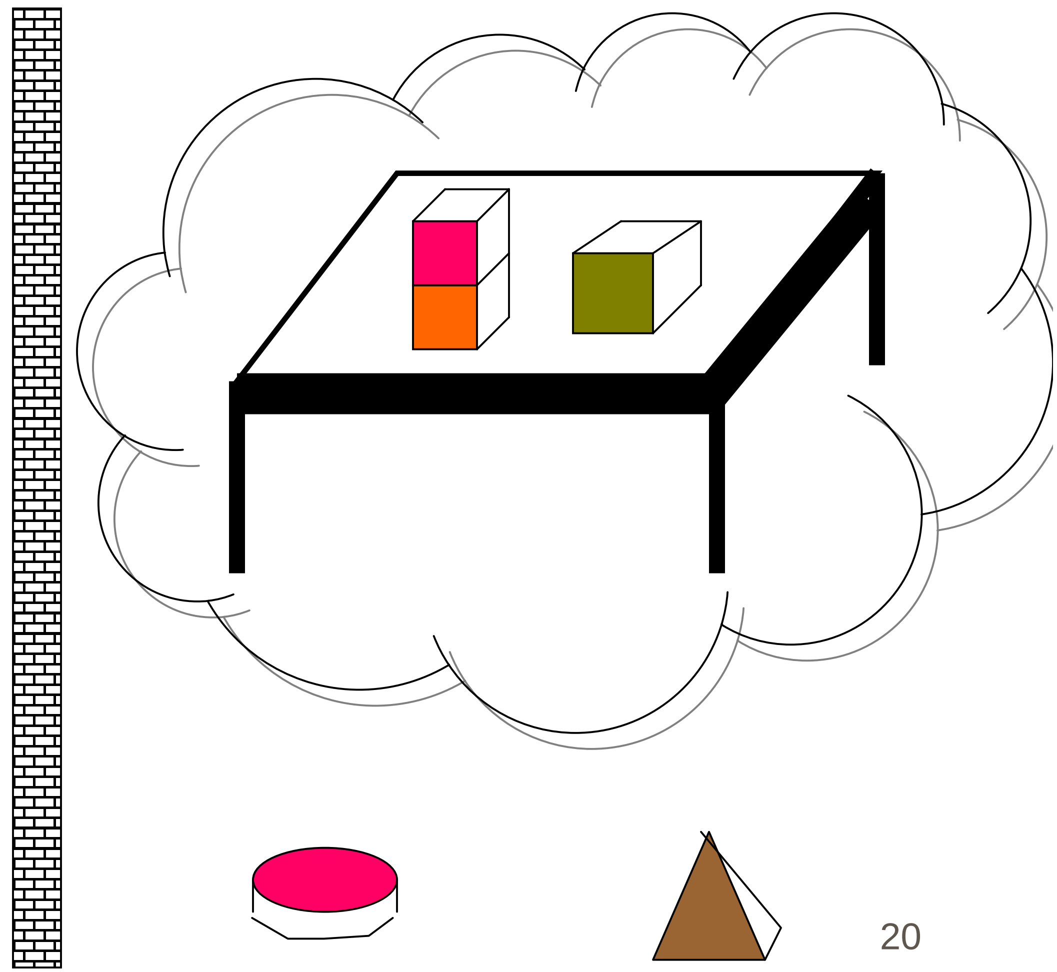
World

+BLOCKA+

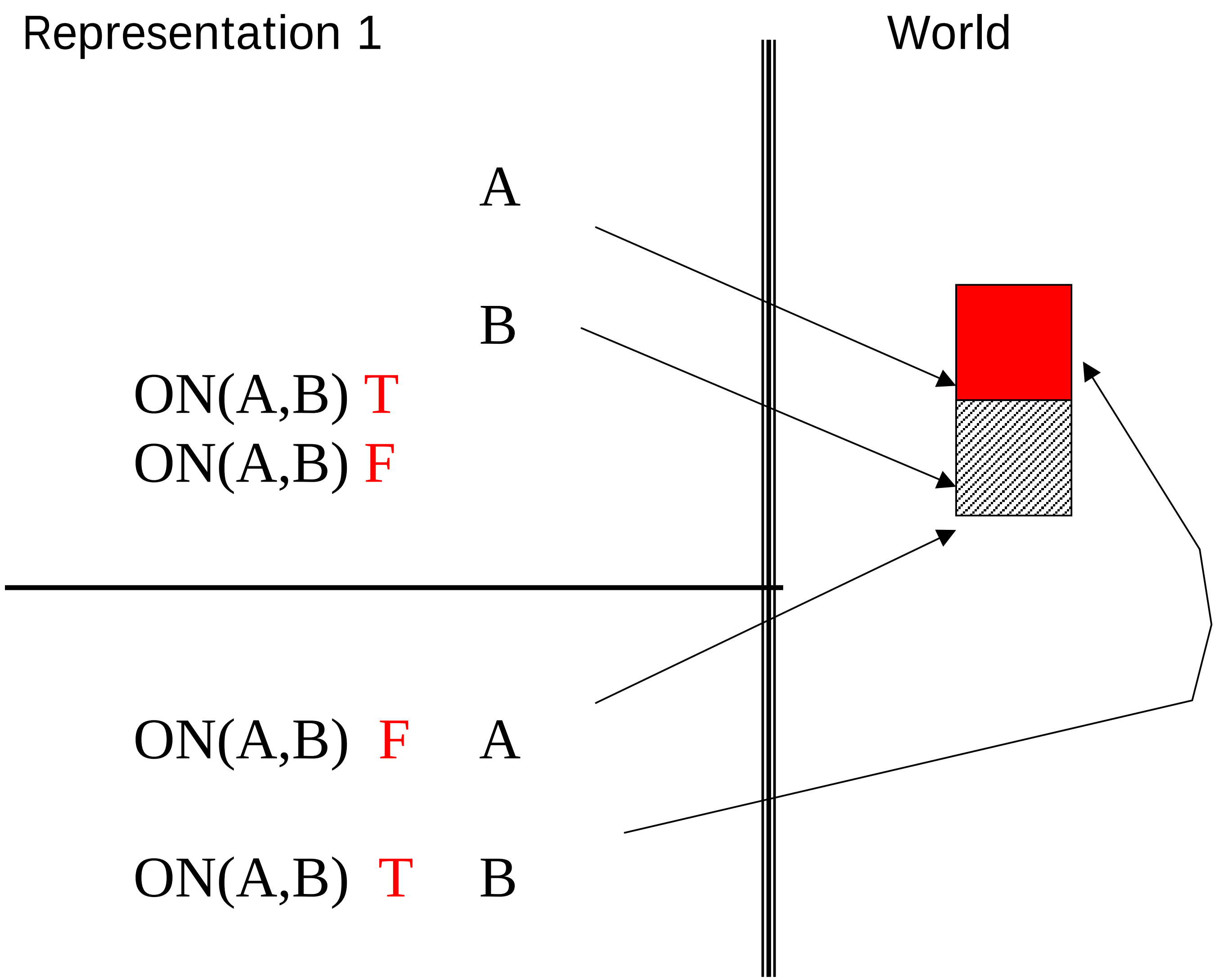
+BLOCKB+

+BLOCKC+

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



## Truth depends on Interpretation



## Entailment

$$KB \models \alpha$$

Knowledge base $KB$ entails sentence $\alpha$
if and only if
$\alpha$ 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"

Entailment is different than inference

## Logic as a representation of the World

$$\begin{tikzcd}
\text{Representation:} & \text{Sentences} \arrow[r, "\text{entails}"] \arrow[d] & \text{Sentence} \arrow[d] \\
\text{Refers to (Semantics)} & & \\
\text{World} & \text{Facts} \arrow[r, "\text{follows}"] & \text{Fact}
\end{tikzcd}$$

# Models

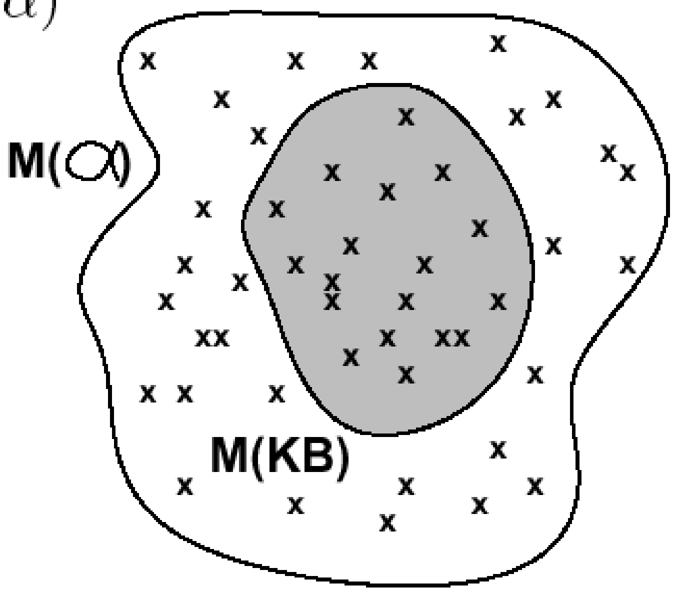
Logicians typically think in terms of <u>models</u>, which are formally structured worlds with respect to which truth can be evaluated

We say $m$ is a <u>model</u> of a sentence $\alpha$ if $\alpha$ is true in $m$

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

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

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



# Inference

$KB \vdash_i \alpha$ = sentence $\alpha$ 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”
- ¬P “P is false” negation
- P V Q “either P is true or Q is true or both” disjunction
- P ^ Q “both P and Q are true” conjunction
- P => Q “if P is true, the Q is true” implication
- P ⇔ 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 |

## 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 | ¬P | ¬Q | P V Q | P ^ Q | P=>Q | P⇔Q |
|---|---|---|---|---|---|---|---|
| T | T | F | F | T | T | T | T |
| T | F | F | T | T | F | F | F |
| F | T | T | F | T | F | T | F |
| F | F | T | T | F | F | T | T |

# Propositional logic: basic manipulation rules

- ¬(¬A) = A — Double negation
- ¬(A ^ B) = (¬A) V (¬B) — Negated "and"
- ¬(A V B) = (¬A) ^ (¬B) — Negated "or"
- A ^ (B V C) = (A ^ B) V (A ^ C) — Distributivity of ^ on V
- A => B = (¬A) V B — by definition
- ¬(A => B) = A ^ (¬B) — using negated or
- A ⇔ B = (A => B) ^ (B => A) — by definition
- ¬(A ⇔ B) = (A ^ (¬B))V(B ^ (¬A)) — using negated and & or
- ...

# Propositional inference: enumeration 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—$\alpha$ must be true wherever $KB$ is tr

| $A$ | $B$ | $C$ | $A \vee C$ | $B \vee \neg C$ | $KB$ | $\alpha$ |
|---|---|---|---|---|---|---|
| $False$ | $False$ | $False$ | | | | |
| $False$ | $False$ | $True$ | | | | |
| $False$ | $True$ | $False$ | | | | |
| $False$ | $True$ | $True$ | | | | |
| $True$ | $False$ | $False$ | | | | |
| $True$ | $False$ | $True$ | | | | |
| $True$ | $True$ | $False$ | | | | |
| $True$ | $True$ | $True$ | | | | |

## Enumeration: Solution

| $A$ | $B$ | $C$ | $A \vee C$ | $B \vee \neg C$ | $KB$ | $\alpha$ |
|---|---|---|---|---|---|---|
| $False$ | $False$ | $False$ | $False$ | $True$ | $False$ | $False$ |
| $False$ | $False$ | $True$ | $True$ | $False$ | $False$ | $False$ |
| $False$ | $True$ | $False$ | $False$ | $True$ | $False$ | $True$ |
| $False$ | $True$ | $True$ | $True$ | $True$ | $True$ | $True$ |
| $True$ | $False$ | $False$ | $True$ | $True$ | $True$ | $True$ |
| $True$ | $False$ | $True$ | $True$ | $False$ | $False$ | $True$ |
| $True$ | $True$ | $False$ | $True$ | $True$ | $True$ | $True$ |
| $True$ | $True$ | $True$ | $True$ | $True$ | $True$ | $True$ |

# 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., $(A \lor \neg B) \land (B \lor \neg C \lor \neg D)$

"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., $(A \land B) \lor (A \land \neg C) \lor (A \land \neg D) \lor (\neg B \land \neg C) \lor (\neg B \land \neg D)$

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

Horn Form (restricted)

*conjunction* of *Horn clauses* (clauses with $\leq 1$ positive literal)

E.g., $(A \lor \neg B) \land (B \lor \neg C \lor \neg D)$

Often written as set of implications:

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

## 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)

## Tautologies

- 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)

## Validity and satisfiability

A sentence is valid if it is true in all models

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

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 $\alpha$ by *reductio ad absurdum*

## Proof methods

Proof methods divide into (roughly) two kinds:

Model checking

- truth table enumeration (sound and complete for propositional)
- heuristic search in model space (sound but incomplete)
  - e.g., the GSAT algorithm (Ex. 6.15)

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.

## Inference Rules

- **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

◇ **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 $\beta$ cannot be both true and false, one of the other disjuncts must be true in one of the premises. Or equivalently, implication is transitive.)

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

## Wumpus world: example

- **Facts:** Percepts inject (TELL) facts into the KB
  - [stench at 1,1 and 2,1] → S1,1 ; S2,1
- **Rules:** if square has no stench then neither the square or adjacent square contain the wumpus
  - R1: $!S1,1 \Rightarrow !W1,1 \wedge !W1,2 \wedge !W2,1$
  - R2: $!S2,1 \Rightarrow !W1,1 \wedge !W2,1 \wedge !W2,2 \wedge !W3,1$
  - ...
- **Inference:**
  - KB contains !S1,1 then using Modus Ponens we infer $!W1,1 \wedge !W1,2 \wedge !W2,1$
  - Using And-Elimination we get: !W1,1 !W1,2 !W2,1
  - ...

## Limitations of Propositional Logic

1. It is too weak, i.e., has very limited expressiveness:

- Each rule has to be represented for each situation: e.g., "don't go forward if the wumpus is in front of you" takes 64 rules

2. It cannot keep track of changes:

- If one needs to track changes, e.g., where the agent has been before then we need a timed-version of each rule. To track 100 steps we'll then need 6400 rules for the previous example.

Its **hard to write and maintain** such a huge rule-base

**Inference becomes intractable**

# 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
- soundess: 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.

Propositional logic suffices for some of these tasks

Truth table method is sound and complete for propositional logic

## Next time

- First-order logic: [AIMA] Chapter 7