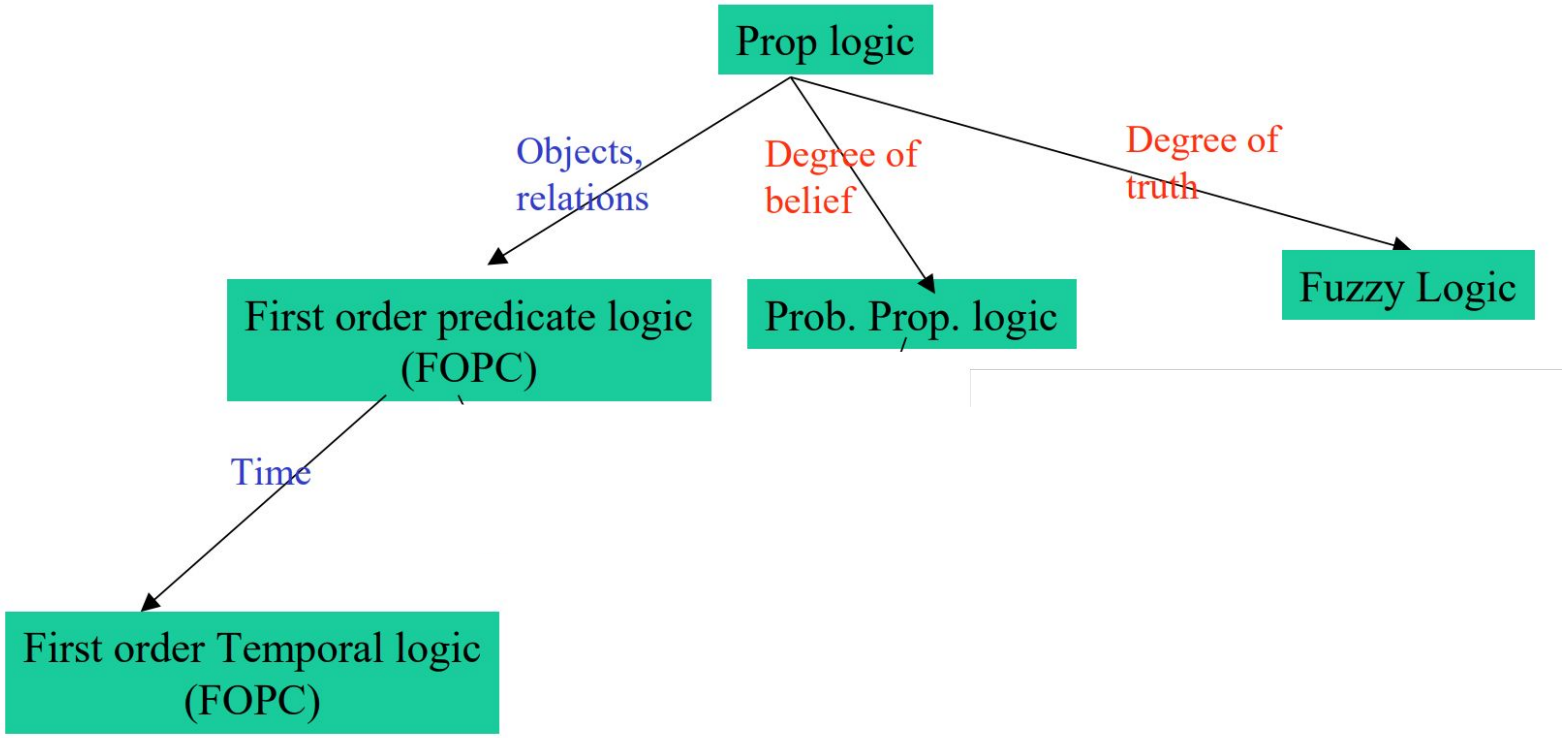
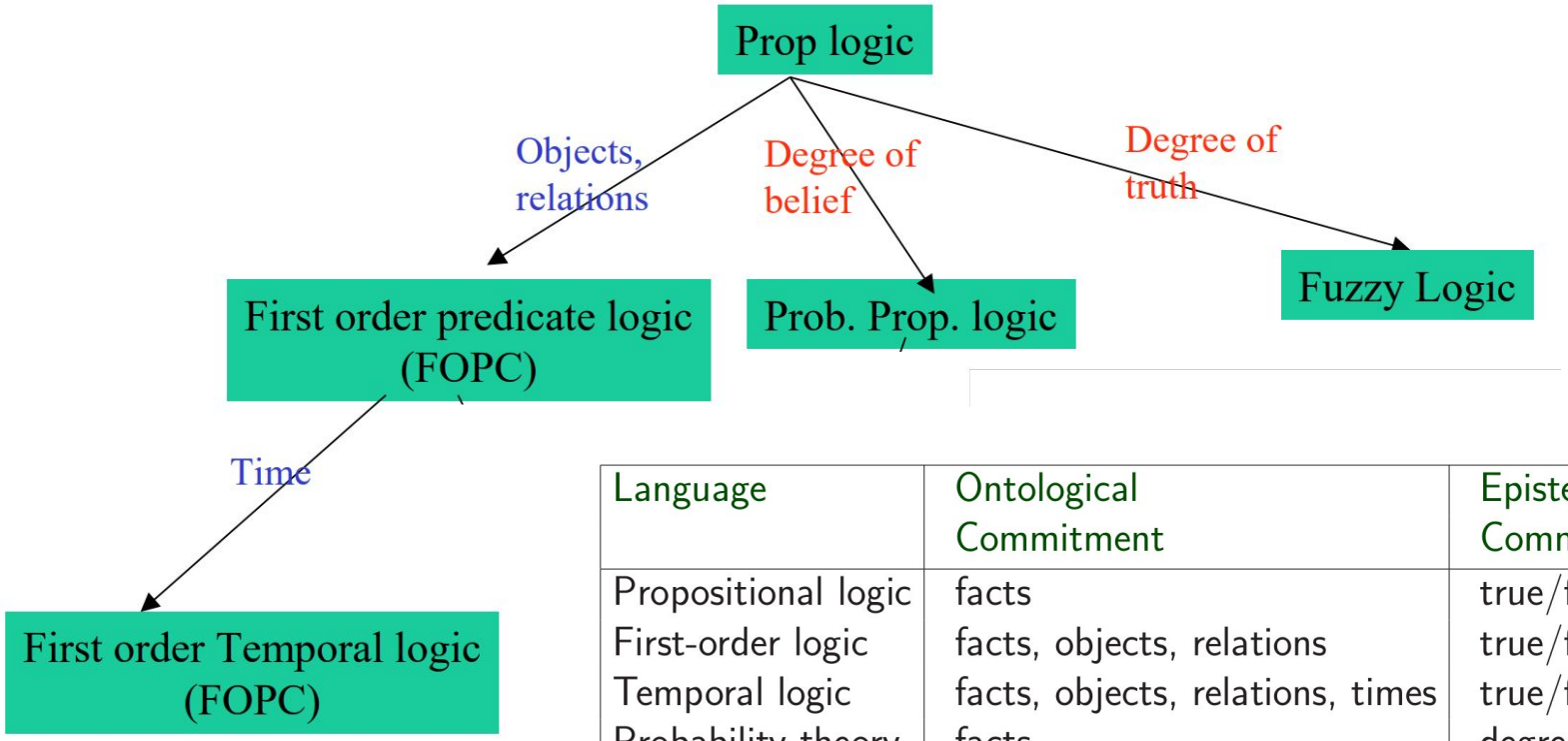
Introduction to Knowledge Representation

KR Languages



KR Languages



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
Fuzzy logic	facts + degree of truth	known interval value

- **Ontology**: a set of concepts and categories in a subject area (What entities are modeled).
- **Epistemology** is the theory of knowledge (What an agent believes about entities).

Logics

A logic is a triple $\langle \mathcal{L}, \mathcal{S}, \mathcal{R} \rangle$ where

- $\mathcal{L}$, the logic's language, is a class of sentences described by a formal grammar
- $\mathcal{S}$, the logic's semantics is a formal specification of how to assign *meaning* in the "real world" to the elements of $\mathcal{L}$
- $\mathcal{R}$, the logic's inference system, is a set of formal derivation *rules* over $\mathcal{L}$

Logics

A logic is a triple $\langle \mathcal{L}, \mathcal{S}, \mathcal{R} \rangle$ where

- $\mathcal{L}$, the logic's language, is a class of sentences described by a formal grammar
- $\mathcal{S}$, the logic's semantics is a formal specification of how to assign *meaning* in the "real world" to the elements of $\mathcal{L}$
- $\mathcal{R}$, the logic's inference system, is a set of formal derivation *rules* over $\mathcal{L}$

There are several logics: propositional, first-order, higher-order, modal, temporal, intuitionistic, linear, equational, non-monotonic, fuzzy, ...

We will concentrate on propositional logic and first-order logic

Conversion to CNF

Ex.: $A \Leftrightarrow (B \vee C)$

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

$$(A \Rightarrow (B \vee C)) \wedge ((B \vee C) \Rightarrow A)$$

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

$$(\neg A \vee B \vee C) \wedge (\neg(B \vee C) \vee A)$$

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

$$(\neg A \vee B \vee C) \wedge ((\neg B \wedge \neg C) \vee A)$$

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

$$(\neg A \vee B \vee C) \wedge (\neg B \vee A) \wedge (\neg C \vee A)$$

Logical Equivalence

Two sentences φ_1 and φ_2 are logically equivalent, written

$$\varphi_1 \equiv \varphi_2$$

if $\varphi_1 \models \varphi_2$ and $\varphi_2 \models \varphi_1$

Note:

- $\varphi_1 \equiv \varphi_2$ if and only if every interpretation assigns the same Boolean value to φ_1 and φ_2

Inference Systems for Propositional Logic

An inference system $\mathcal{I}$ for PL is a procedure that

given a set $\Gamma = \{\alpha_1, \ldots, \alpha_m\}$ of sentences and a sentence φ,
may reply "yes", "no", or run forever

If $\mathcal{I}$ replies positively to input (Γ, φ), we say that Γ derives φ in $\mathcal{I}$ (or, $\mathcal{I}$ derives φ from Γ, or, φ derives from Γ in $\mathcal{I}$) and write

$$\Gamma \vdash_{\mathcal{I}} \varphi$$

Intuitively, $\mathcal{I}$ should be such that it replies "yes" on input (Γ, φ)
only if φ is in fact entailed by Γ

Inference systems for PL

Divided into (roughly) two kinds:

Rule-based

- Sound generation of new sentences from old
- Proof = a sequence of inference rule applications
  Can use inference rules as operators as in a standard search procedures
- Typically require translation of sentences into some normal form

Model-based

- Truth table enumeration (always exponential in n)

Rule-Based Inference in PL

An inference system in Propositional Logic can also be specified as a set $\mathcal{R}$ of inference (or derivation) rules

- Each rule is just a *pattern* premises/conclusion
- A rule applies to Γ and derives φ if
  - some of the sentences in Γ match with the premises of the rule and
  - φ matches with the conclusion
- A rule is sound it the set of its premises entails its conclusion

Some Inference Rules

- And-Introduction

$$\frac{\alpha \quad \beta}{\alpha \wedge \beta}$$

- And-Elimination

$$\frac{\alpha \wedge \beta}{\alpha} \qquad \frac{\alpha \wedge \beta}{\beta}$$

- Or-Introduction

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

Some Inference Rules (cont'd)

- Implication-Elimination (aka Modus Ponens)

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

- Unit Resolution

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

- Resolution

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

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

Soundness

- An inference algorithm that derives only entailed sentences is called sound or truth- preserving.
- Soundness is a good thing!
- An unsound inference procedure essentially makes things up as it goes along (discovery of nonexistent)

if $\Gamma \vdash_{\mathcal{I}} \varphi$ then $\Gamma \models \varphi$

Soundness

Implication-Elimination (aka Modus Ponens)

$$\frac{\alpha \Rightarrow \beta \quad \alpha}{\beta}$$

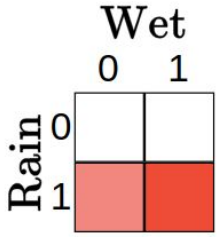
Is $\dfrac{\mathbf{Rain}, \quad \mathbf{Rain} \rightarrow \mathbf{Wet}}{\mathbf{Wet}}$ (Modus ponens) sound?

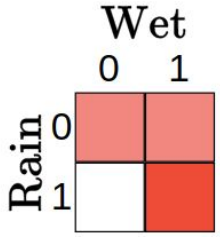
Soundness

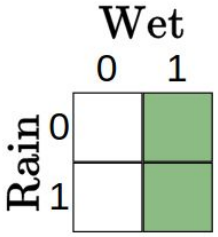
Implication-Elimination (aka Modus Ponens)

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

Is $\dfrac{\mathbf{Rain}, \qquad \mathbf{Rain} \rightarrow \mathbf{Wet}}{\mathbf{Wet}}$ (Modus ponens) sound?







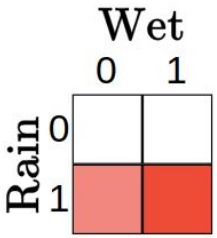
https://stanford-cs221.github.io/winter2021-extra/modules/logic-conclusion/inference-rules.pdf

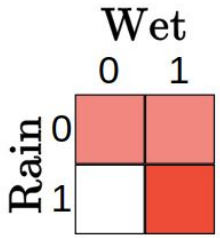
Soundness

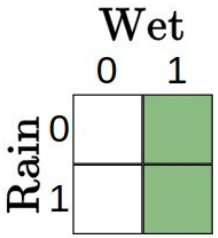
Implication-Elimination (aka Modus Ponens)

$$\frac{\alpha \Rightarrow \beta \quad \alpha}{\beta}$$

Is $\dfrac{\textbf{Rain}, \quad \textbf{Rain} \rightarrow \textbf{Wet}}{\textbf{Wet}}$ (Modus ponens) sound?







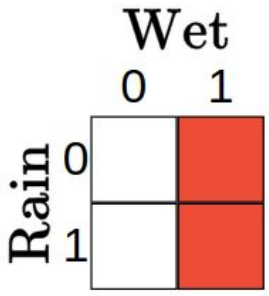
Sound!

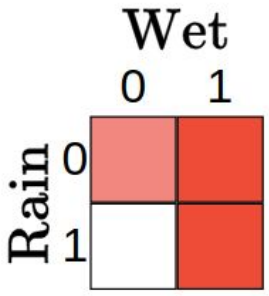
Soundness

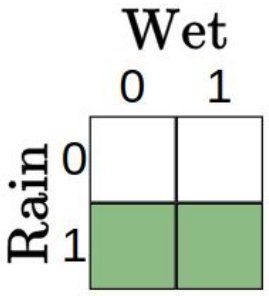
Is $\dfrac{\mathbf{Wet}, \quad \mathbf{Rain} \rightarrow \mathbf{Wet}}{\mathbf{Rain}}$ sound?

Soundness

Is $\dfrac{\mathbf{Wet}, \quad \mathbf{Rain} \rightarrow \mathbf{Wet}}{\mathbf{Rain}}$ sound?

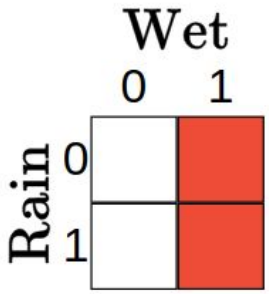


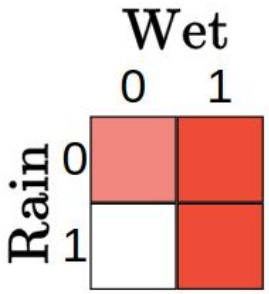


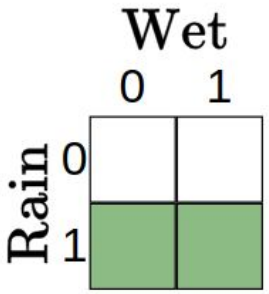


Soundness

Is $\dfrac{\mathbf{Wet}, \quad \mathbf{Rain} \rightarrow \mathbf{Wet}}{\mathbf{Rain}}$ sound?







Unsound!

Completeness

- An inference algorithm is complete if it can derive any sentence that is entailed.
- For some KBs, the number of sentences can be infinite
- Can't exhaustively check all of them, need to rely on proving completeness

$$\text{if } \Gamma \models \varphi \text{ then } \Gamma \vdash_{\mathcal{I}} \varphi$$

Completeness

Example: Modus ponens is incomplete

Setup:

$$\mathrm{KB} = \{\ \text{Rain},\ \text{Rain} \vee \text{Snow} \rightarrow \text{Wet}\ \}$$

$$\alpha = \text{Wet}$$

$$\text{Rules} = \left\{ \frac{f, \quad f \rightarrow g}{g} \right\} \text{ (Modus ponens)}$$

Semantically: $\mathrm{KB} \models \alpha$ (α is entailed).

Syntactically: $\mathrm{KB} \nvdash \alpha$ (can't derive α).

Completeness

Example: Modus ponens is incomplete

Setup:

$$\mathrm{KB} = \{\ \text{Rain},\ \text{Rain} \vee \text{Snow} \rightarrow \text{Wet}\ \}$$

$$\alpha = \text{Wet}$$

$$\text{Rules} = \left\{ \frac{f,\quad f \rightarrow g}{g} \right\} \text{ (Modus ponens)}$$

Semantically: $\mathrm{KB} \models \alpha$ (α is entailed).
Syntactically: $\mathrm{KB} \nvdash \alpha$ (can't derive α).

Incomplete!

Fixing Completeness

Option 1: Restrict the allowed set of formulas

propositional logic

propositional logic with only Horn clauses

Option 2: Use more powerful inference rules

Modus ponens

resolution

Modus ponens

Inference rule:

Definition: Modus ponens

$$\frac{p_1, \quad \cdots \quad , p_k, \quad (p_1 \wedge \cdots \wedge p_k) \rightarrow q}{q}$$

$$\frac{\text{Wet}, \quad \text{Weekday}, \quad \text{Wet} \wedge \text{Weekday} \rightarrow \text{Traffic}}{\text{Traffic}}$$

Completeness Modus ponens

Theorem: Modus ponens on Horn clauses

Modus ponens is **complete** with respect to Horn clauses:

- Suppose KB contains only Horn clauses and p is an entailed propositional symbol.
- Then applying modus ponens will derive p.

$\mathrm{KB} \models p$ (entailment) is the same as $\mathrm{KB} \vdash p$ (derivation)!

Resolution Rule is Sound?

Example: the Resolution rule $\dfrac{\alpha \vee \beta, \quad \neg\beta \vee \gamma}{\alpha \vee \gamma}$

	α	β	γ	$\neg\beta$	$\alpha \vee \beta$	$\neg\beta \vee \gamma$	$\alpha \vee \gamma$
1.	*false*	*false*	*false*	*true*	*false*	*true*	*false*
2.	*false*	*false*	*true*	*true*	*false*	*true*	*true*
3.	*false*	*true*	*false*	*false*	*true*	*false*	*false*
4.	*false*	*true*	*true*	*false*	*true*	*true*	*true*
5.	*true*	*false*	*false*	*true*	*true*	*true*	*true*
6.	*true*	*false*	*true*	*true*	*true*	*true*	*true*
7.	*true*	*true*	*false*	*false*	*true*	*false*	*true*
8.	*true*	*true*	*true*	*false*	*true*	*true*	*true*

Resolution Rule is Sound?

Example: the Resolution rule $\dfrac{\alpha \vee \beta, \quad \neg\beta \vee \gamma}{\alpha \vee \gamma}$

	α	β	γ	$\neg\beta$	$\alpha \vee \beta$	$\neg\beta \vee \gamma$	$\alpha \vee \gamma$
1.	*false*	*false*	*false*	*true*	*false*	*true*	*false*
2.	*false*	*false*	*true*	*true*	*false*	*true*	*true*
3.	*false*	*true*	*false*	*false*	*true*	*false*	*false*
4.	*false*	*true*	*true*	*false*	*true*	*true*	*true*
5.	*true*	*false*	*false*	*true*	*true*	*true*	*true*
6.	*true*	*false*	*true*	*true*	*true*	*true*	*true*
7.	*true*	*true*	*false*	*false*	*true*	*false*	*true*
8.	*true*	*true*	*true*	*false*	*true*	*true*	*true*

All the interpretations that satisfy both $\alpha \vee \beta$ and $\neg\beta \vee \gamma$ (4,5,6,8) satisfy $\alpha \vee \gamma$ as well

Complete?

Resolution for inference system

Resolution is a more efficient inference system (in practice)

The main reason is that

- it requires formulas to be in a special form: CNF
- it has only one inference rule

Resolution

Literal: prop. symbol (P) or negated prop. symbol ($\neg P$)

Clause: set of literals $\{l_1, \ldots, l_k\}$ (understood as $l_1 \vee \cdots \vee l_k$)

Conjunctive Normal Form: set of clauses $\{C_1, \ldots, C_n\}$ (understood as $C_1 \wedge \cdots \wedge C_n$)

Resolution

Literal: prop. symbol (P) or negated prop. symbol ($\neg P$)

Clause: set of literals $\{l_1, \ldots, l_k\}$ (understood as $l_1 \vee \cdots \vee l_k$)

Conjunctive Normal Form: set of clauses $\{C_1, \ldots, C_n\}$ (understood as $C_1 \wedge \cdots \wedge C_n$)

Resolution rule for CNF:

$$\frac{l_1 \vee \cdots \vee l_k \vee P \qquad \neg P \vee m_1 \vee \cdots \vee m_n}{l_1 \vee \cdots \vee l_k \vee m_1 \vee \cdots \vee m_n}$$

E.g.,

$$\frac{P \vee Q \qquad \neg Q}{P} \qquad \frac{P \vee Q \qquad R \vee \neg Q \vee \neg S}{P \vee R \vee \neg S} \qquad \frac{P \vee Q \qquad \neg Q \vee P \vee R}{P \vee R}$$

Resolution

Literal: prop. symbol (P) or negated prop. symbol ($\neg P$)

Clause: set of literals $\{l_1, \ldots, l_k\}$ (understood as $l_1 \vee \cdots \vee l_k$)

Conjunctive Normal Form: set of clauses $\{C_1, \ldots, C_n\}$ (understood as $C_1 \wedge \cdots \wedge C_n$)

Resolution rule for CNF:

$$\frac{l_1 \vee \cdots \vee l_k \vee P \qquad \neg P \vee m_1 \vee \cdots \vee m_n}{l_1 \vee \cdots \vee l_k \vee m_1 \vee \cdots \vee m_n}$$

E.g.,

$$\frac{P \vee Q \qquad \neg Q}{P} \qquad \frac{P \vee Q \qquad R \vee \neg Q \vee \neg S}{P \vee R \vee \neg S} \qquad \frac{P \vee Q \qquad \neg Q \vee P \vee R}{P \vee R}$$

Resolution is sound and complete for CNF KBs

Resolution: example

Proof by contradiction: show $KB \models \alpha$ by showing $KB \wedge \neg\alpha$ unsatisfiable.
Do the latter by deriving **False** from CNF of $KB \wedge \neg\alpha$

KB = $\{A \rightarrow (B \vee C), A\}$

B ? C ?

Resolution: example

Proof by contradiction: show $KB \models \alpha$ by showing $KB \wedge \neg\alpha$ unsatisfiable.
Do the latter by deriving **False** from CNF of $KB \wedge \neg\alpha$

Knowledge base (is it satisfiable?):

$$\mathrm{KB} = \{A \rightarrow (B \vee C), A, \neg B, \neg C\}$$

Convert to CNF:

$$\mathrm{KB} = \{\neg A \vee B \vee C, A, \neg B, \neg C\}$$

Resolution: example

Proof by contradiction: show $KB \models \alpha$ by showing $KB \wedge \neg\alpha$ unsatisfiable.
Do the latter by deriving **False** from CNF of $KB \wedge \neg\alpha$

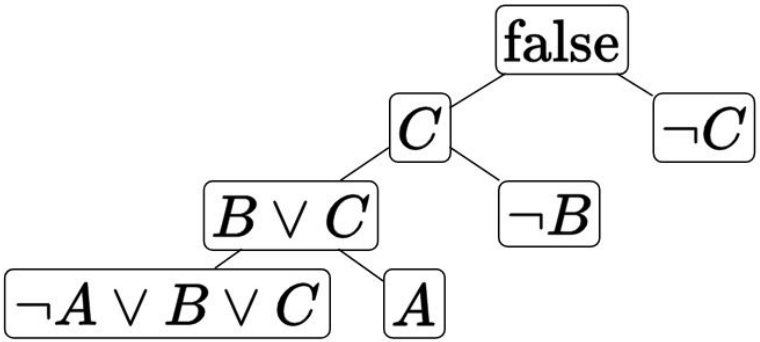
Knowledge base (is it satisfiable?):

$$\mathrm{KB} = \{A \rightarrow (B \vee C), A, \neg B, \neg C\}$$

Convert to CNF:

$$\mathrm{KB} = \{\neg A \vee B \vee C, A, \neg B, \neg C\}$$

Repeatedly apply **resolution** rule:



Resolution: example

Proof by contradiction: show $KB \models \alpha$ by showing $KB \wedge \neg\alpha$ unsatisfiable.
Do the latter by deriving **False** from CNF of $KB \wedge \neg\alpha$

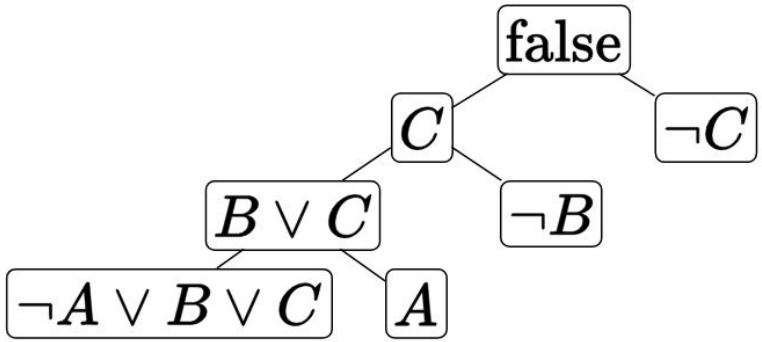
Knowledge base (is it satisfiable?):

$$\mathrm{KB} = \{A \rightarrow (B \vee C), A, \neg B, \neg C\}$$

Convert to CNF:

$$\mathrm{KB} = \{\neg A \vee B \vee C, A, \neg B, \neg C\}$$

Repeatedly apply **resolution** rule:



Unsatisfiable!

Time complexity

Modus ponens inference rule

$$\frac{p_1, \quad \cdots \quad , p_k, \quad (p_1 \wedge \cdots \wedge p_k) \rightarrow q}{q}$$

- Each rule application adds clause with one propositional symbol

$\Rightarrow$ linear time

Resolution inference rule

$$\frac{f_1 \vee \cdots \vee f_n \vee p, \quad \neg p \vee g_1 \vee \cdots \vee g_m}{f_1 \vee \cdots \vee f_n \vee g_1 \vee \cdots \vee g_m}$$

- Each rule application adds clause with many propositional symbols

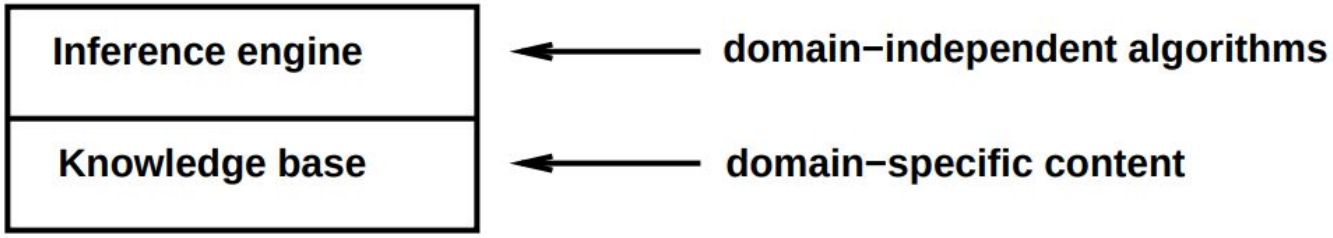
$\Rightarrow$ exponential time

Modus ponens & Resolution

Horn clauses	any clauses
modus ponens	resolution
linear time	exponential time
less expressive	more expressive

Simple Knowledge based Agent

Knowledge bases



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

A simple knowledge-based agent

```
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
```

Figure 7.1 A generic knowledge-based agent. Given a percept, the agent adds the percept to its knowledge base, asks the knowledge base for the best action, and tells the knowledge base that it has in fact taken that action.

An Example: The Wumpus World!

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

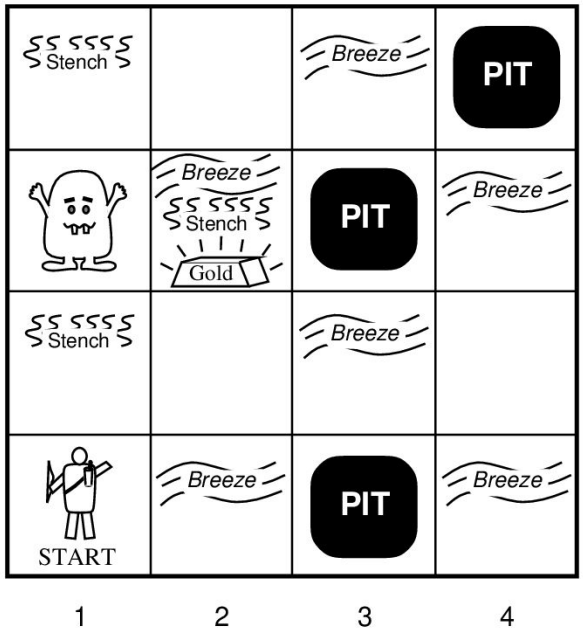
Shooting uses up the only arrow

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?

Deterministic?

Episodic?

Static?

Discrete?

Single-agent?

Wumpus World Characterization

Observable? Partially—only local perception

Deterministic?

Episodic?

Static?

Discrete?

Single-agent?

Wumpus World Characterization

Observable? Partially—only local perception

Deterministic? Yes—outcomes exactly specified

Episodic?

Static?

Discrete?

Single-agent?

Wumpus World Characterization

Observable? Partially—only local perception

Deterministic? Yes—outcomes exactly specified

Episodic? No—sequential at the level of actions

Static?

Discrete?

Single-agent?

Wumpus World Characterization

Observable? Partially—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 Characterization

Observable? Partially—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? Partially—only local perception

Deterministic? Yes—outcomes exactly specified

Episodic? No—sequential at the level of actions

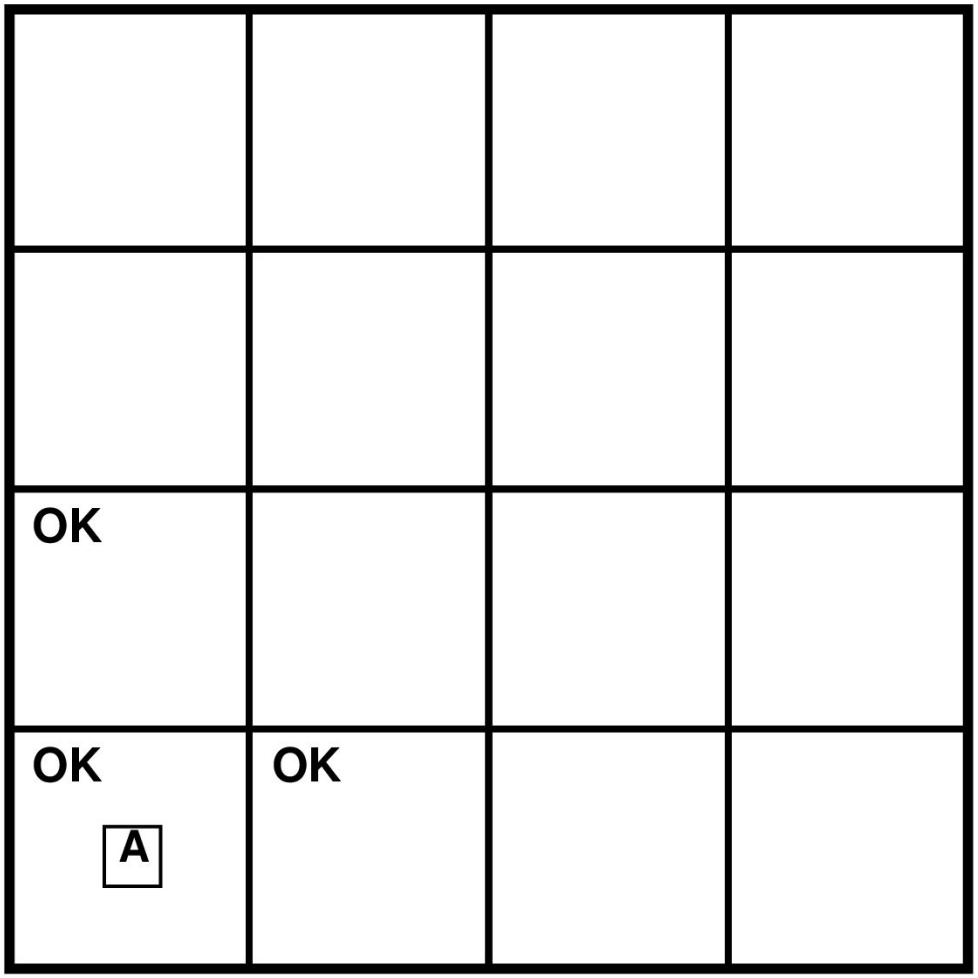
Static? Yes—Wumpus and Pits do not move

Discrete? Yes

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

Exploring a Wumpus World

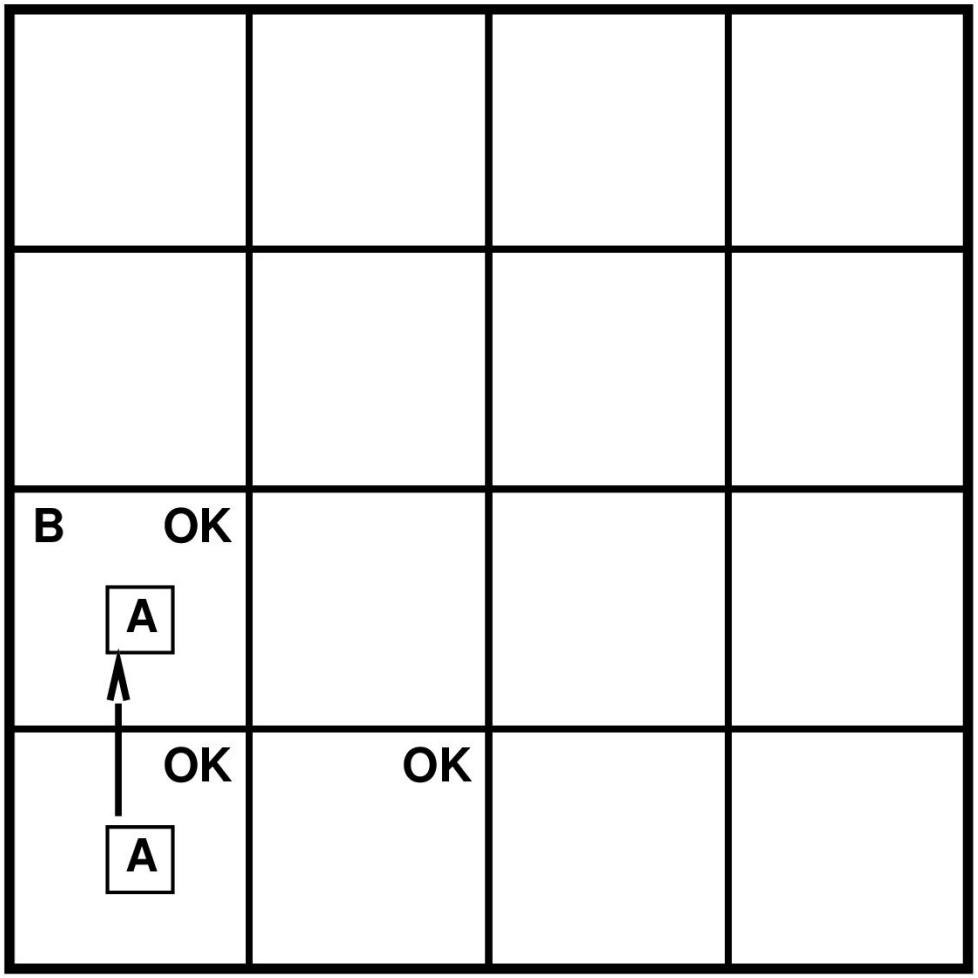
[1,2] and [2,1] OK



A: Agent; B: Breeze; G: Glitter; S: Stench
OK: safe square; W: Wumpus; P: Pit; BGS: bag of gold

Exploring a Wumpus World

Agent moves to [2,1]

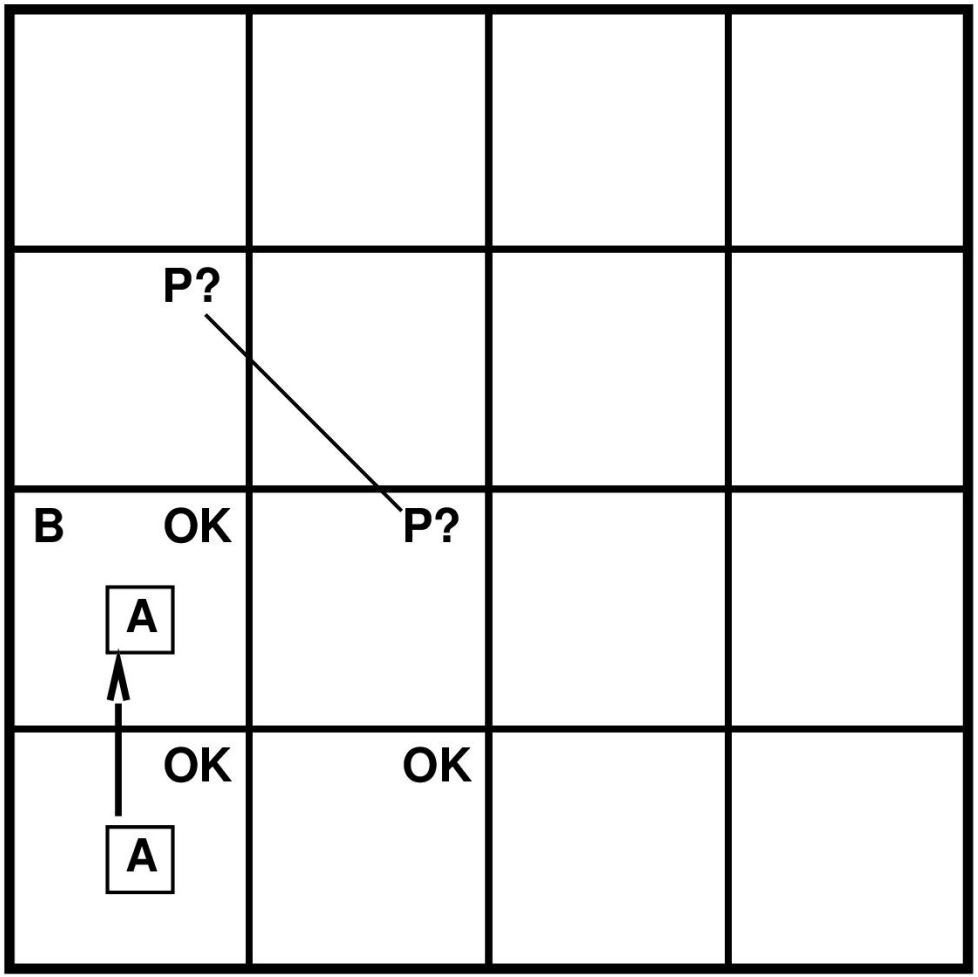


A: Agent; B: Breeze; G: Glitter; S: Stench
OK: safe square; W: Wumpus; P: Pit; BGS: bag of gold

Exploring a Wumpus World

perceives a breeze
⇒ Pit in [3,1] or [2,2]

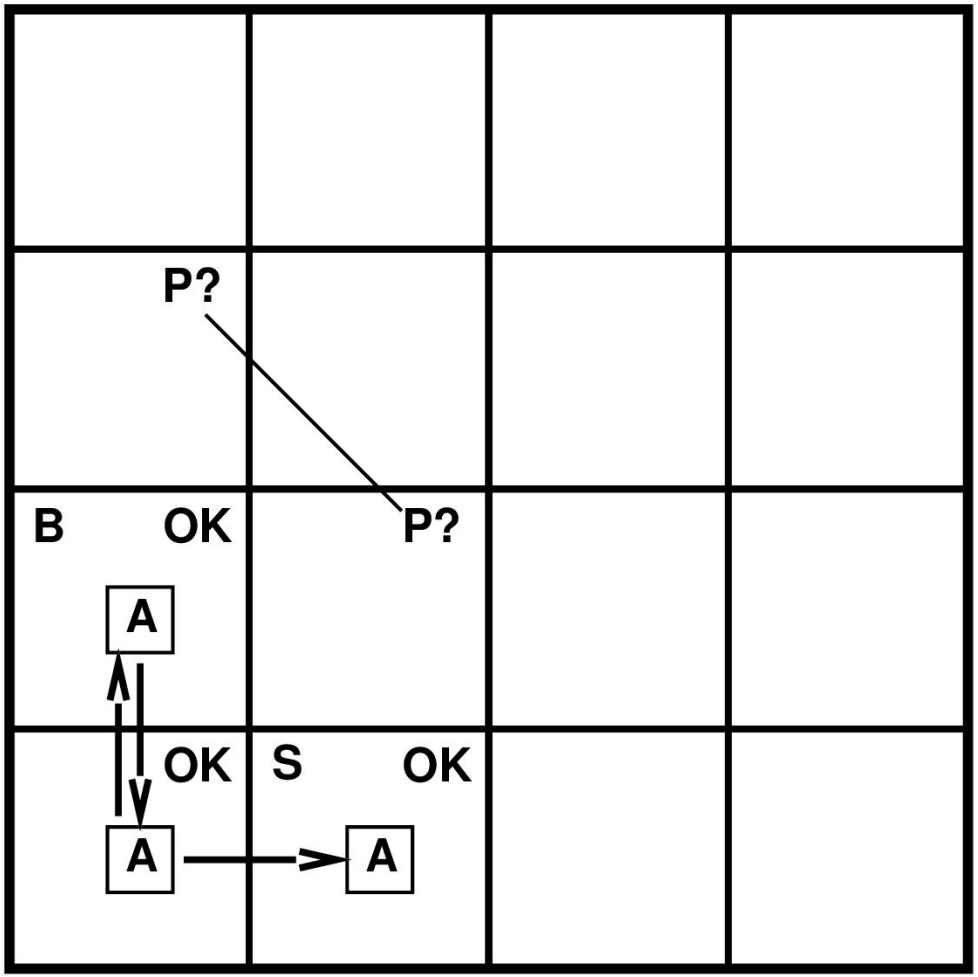
perceives no stench
⇒ no Wumpus in
[3,1], [2,2]



A: Agent; B: Breeze; G: Glitter; S: Stench
OK: safe square; W: Wumpus; P: Pit; BGS: bag of gold

Exploring a Wumpus World

Agent moves to [1,1]-[1,2]

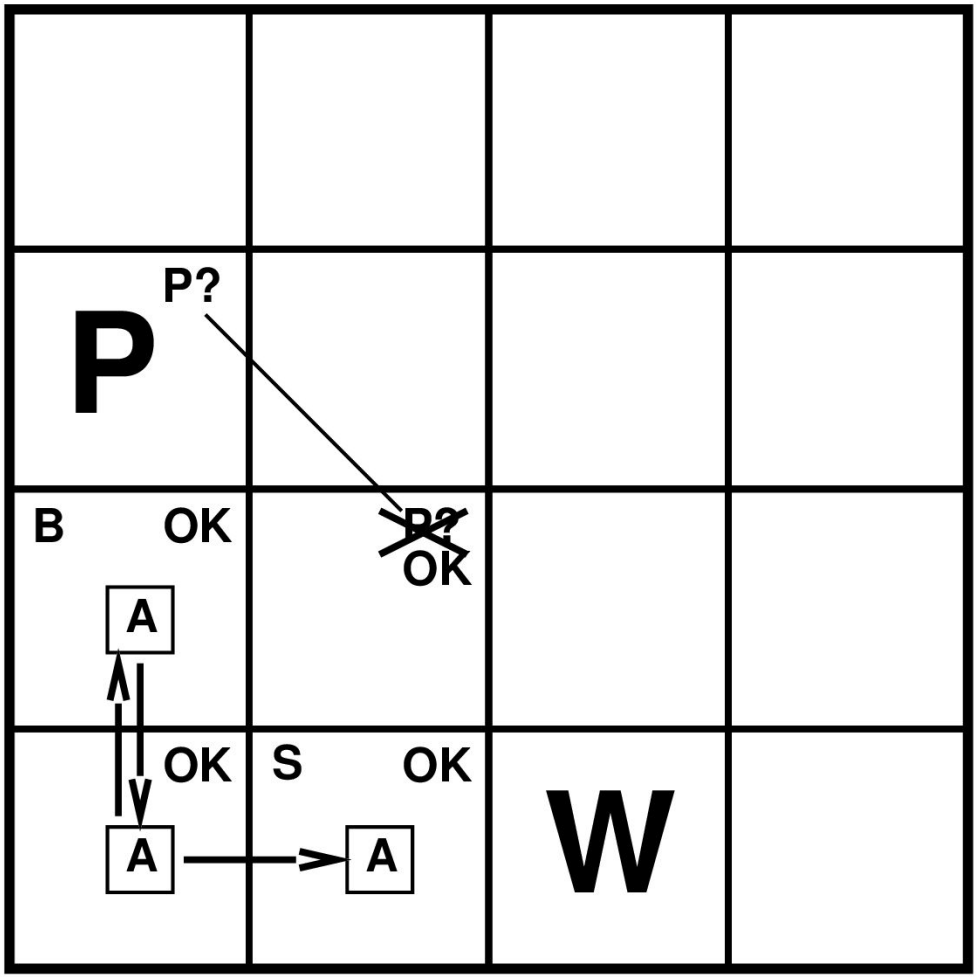


A: Agent; B: Breeze; G: Glitter; S: Stench
OK: safe square; W: Wumpus; P: Pit; BGS: bag of gold

Exploring a Wumpus World

perceives no breeze
⇒ no Pit in [1,3], [2,2]
⇒ [2,2] OK
⇒ pit in [3,1]

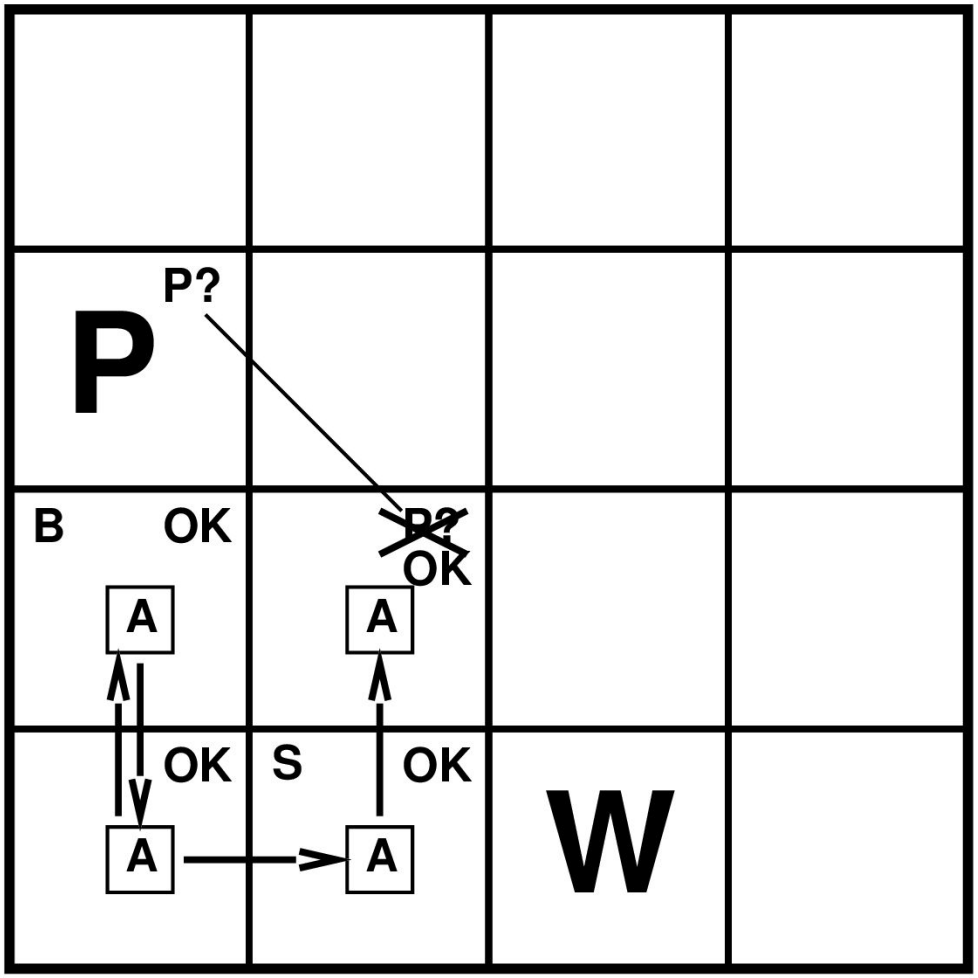
perceives a stench
⇒ Wumpus in [1,3]!



A: Agent; B: Breeze; G: Glitter; S: Stench
OK: safe square; W: Wumpus; P: Pit; BGS: bag of gold

Exploring a Wumpus World

Agent moves to [2,2]
perceives no breeze
perceives no stench

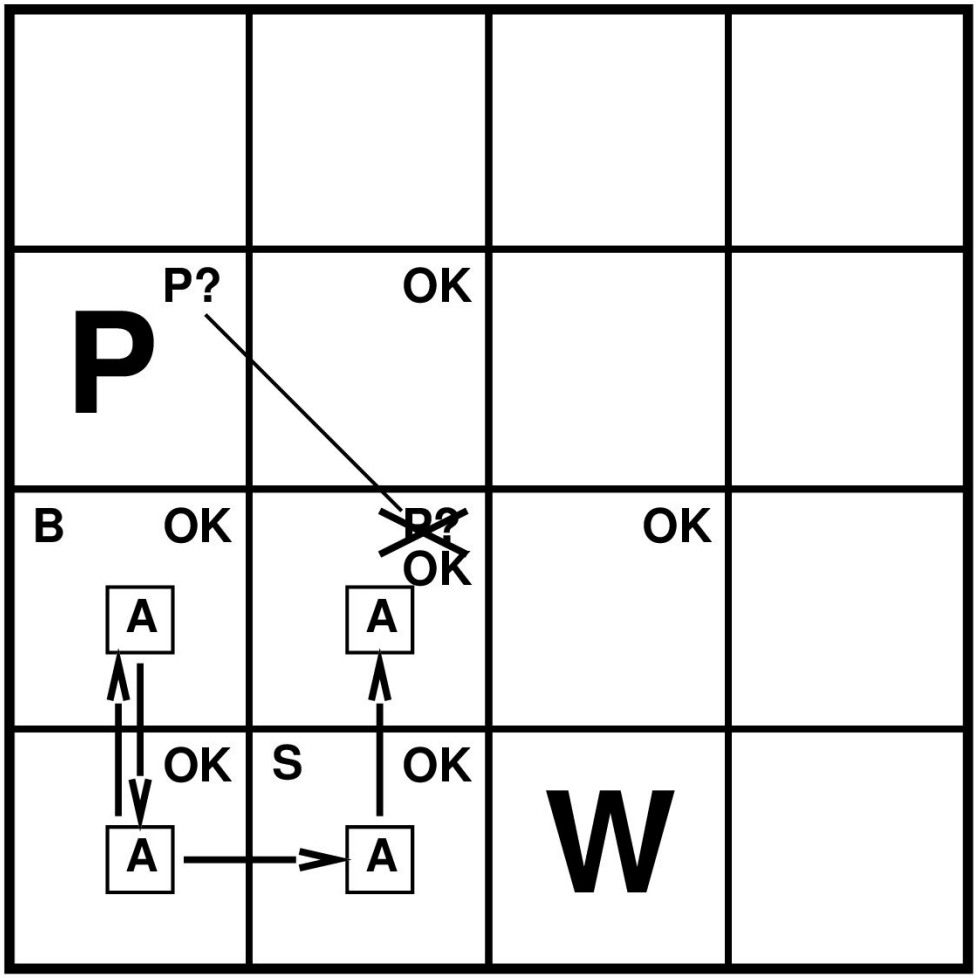


A: Agent; B: Breeze; G: Glitter; S: Stench
OK: safe square; W: Wumpus; P: Pit; BGS: bag of gold

Exploring a Wumpus World

perceives no breeze
⇒ no pit in [3,2], [2,3]

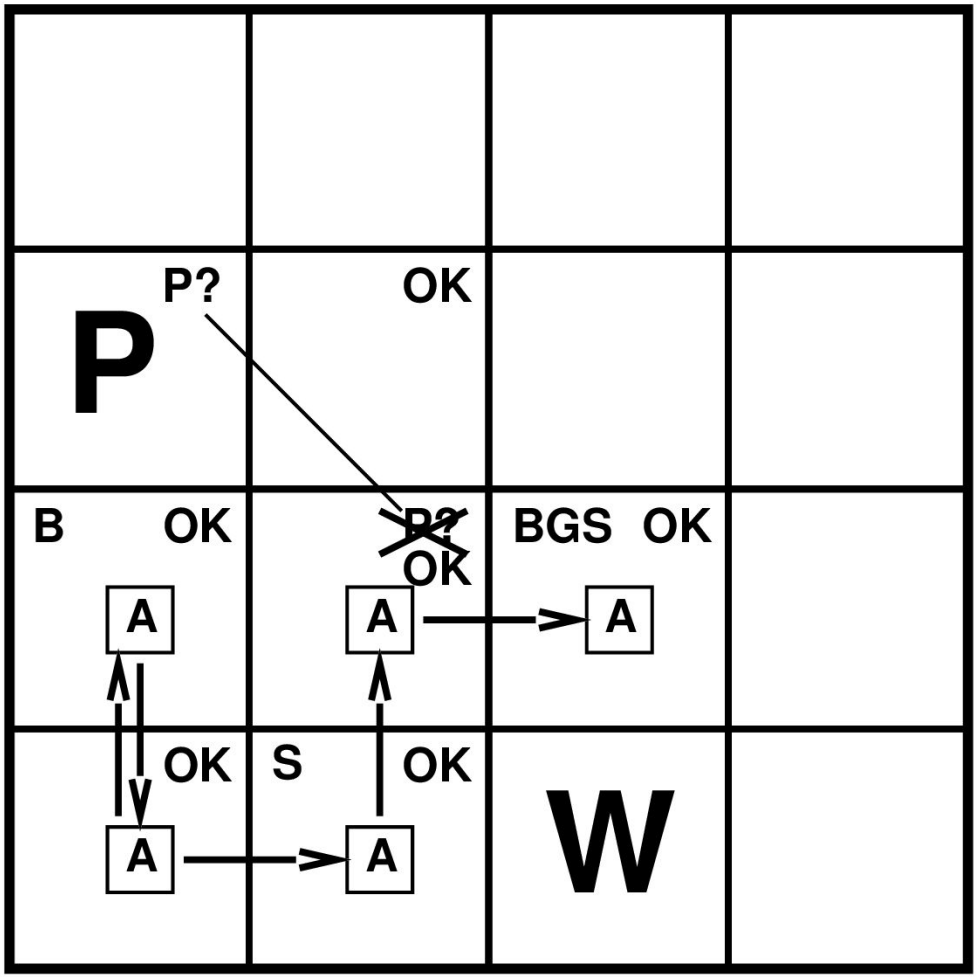
perceives no stench
⇒ no Wumpus in [3,2], [2,3]
⇒ [3,2] and [2,3] OK



A: Agent; B: Breeze; G: Glitter; S: Stench
OK: safe square; W: Wumpus; P: Pit; BGS: bag of gold

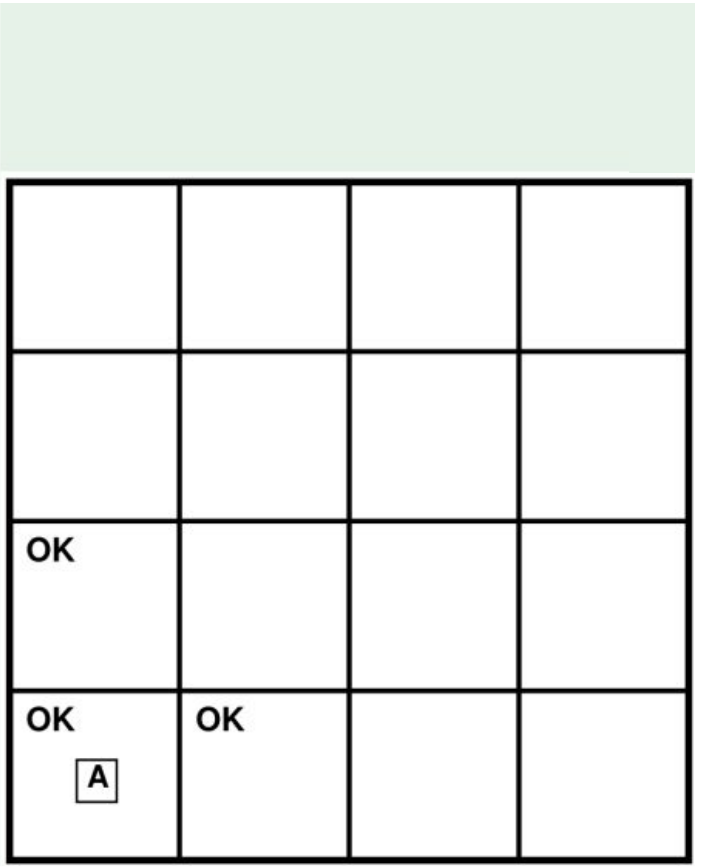
Exploring a Wumpus World

Agent moves to [2,3]
perceives a glitter
⇒ bag of gold!



A: Agent; B: Breeze; G: Glitter; S: Stench
OK: safe square; W: Wumpus; P: Pit; BGS: bag of gold

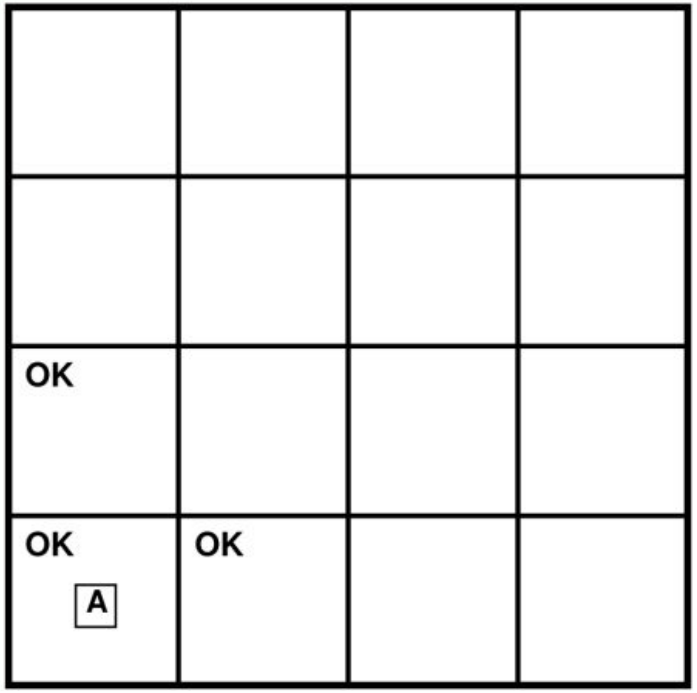
Exploring the Wumpus World



A: Agent; B: Breeze; G: Glitter; S: Stench
OK: safe square; W: Wumpus; P: pit; BGS: glitter, bag of gold

Exploring the Wumpus World

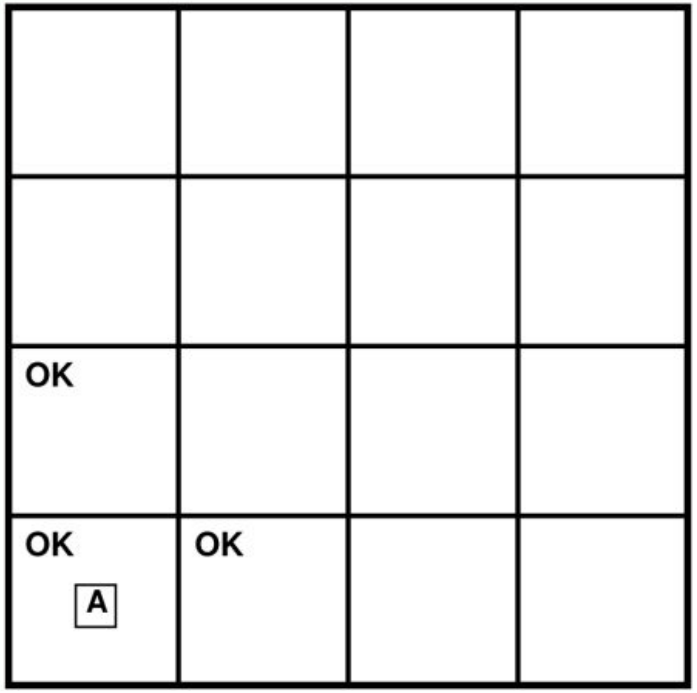
- KB initially contains:
  $\neg P_{[1,1]}, \neg W_{[1,1]}, OK_{[1,1]}$
  $B_{[1,1]} \leftrightarrow (P_{[1,2]} \vee P_{[2,1]})$
  $S_{[1,1]} \leftrightarrow (W_{[1,2]} \vee W_{[2,1]})$
  $OK_{[1,2]} \leftrightarrow (\neg W_{[1,2]} \wedge \neg P_{[2,1]})$
  $OK_{[2,1]} \leftrightarrow (\neg W_{[2,1]} \wedge \neg P_{[1,2]})$
  ...



A: Agent; B: Breeze; G: Glitter; S: Stench
OK: safe square; W: Wumpus; P: pit; BGS: glitter, bag of gold

Exploring the Wumpus World

- KB initially contains:
  $\neg P_{[1,1]}, \neg W_{[1,1]}, OK_{[1,1]}$
  $B_{[1,1]} \leftrightarrow (P_{[1,2]} \vee P_{[2,1]})$
  $S_{[1,1]} \leftrightarrow (W_{[1,2]} \vee W_{[2,1]})$
  $OK_{[1,2]} \leftrightarrow (\neg W_{[1,2]} \wedge \neg P_{[2,1]})$
  $OK_{[2,1]} \leftrightarrow (\neg W_{[2,1]} \wedge \neg P_{[1,2]})$
  ...
- Agent is initially in 1,1
- Percepts (no stench, no breeze):
  $\neg S_{[1,1]}, \neg B_{[1,1]}$



A: Agent; B: Breeze; G: Glitter; S: Stench
OK: safe square; W: Wumpus; P: pit; BGS: glitter, bag of gold