Cooperating Intelligent Systems

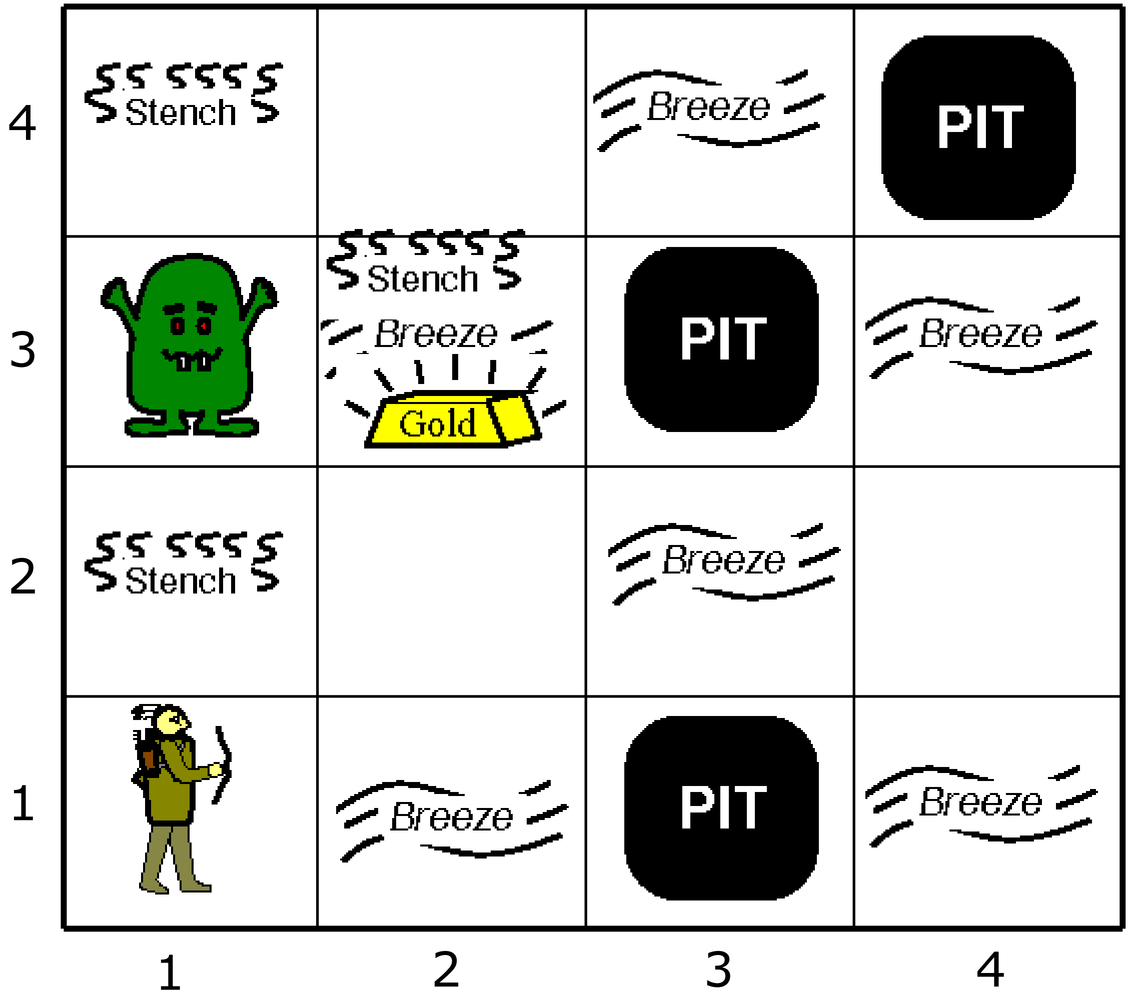
Logical agents
Chapter 7, AIMA

This presentation owes some to V. Pavlovic @ Rutgers and D. Byron @ OSU

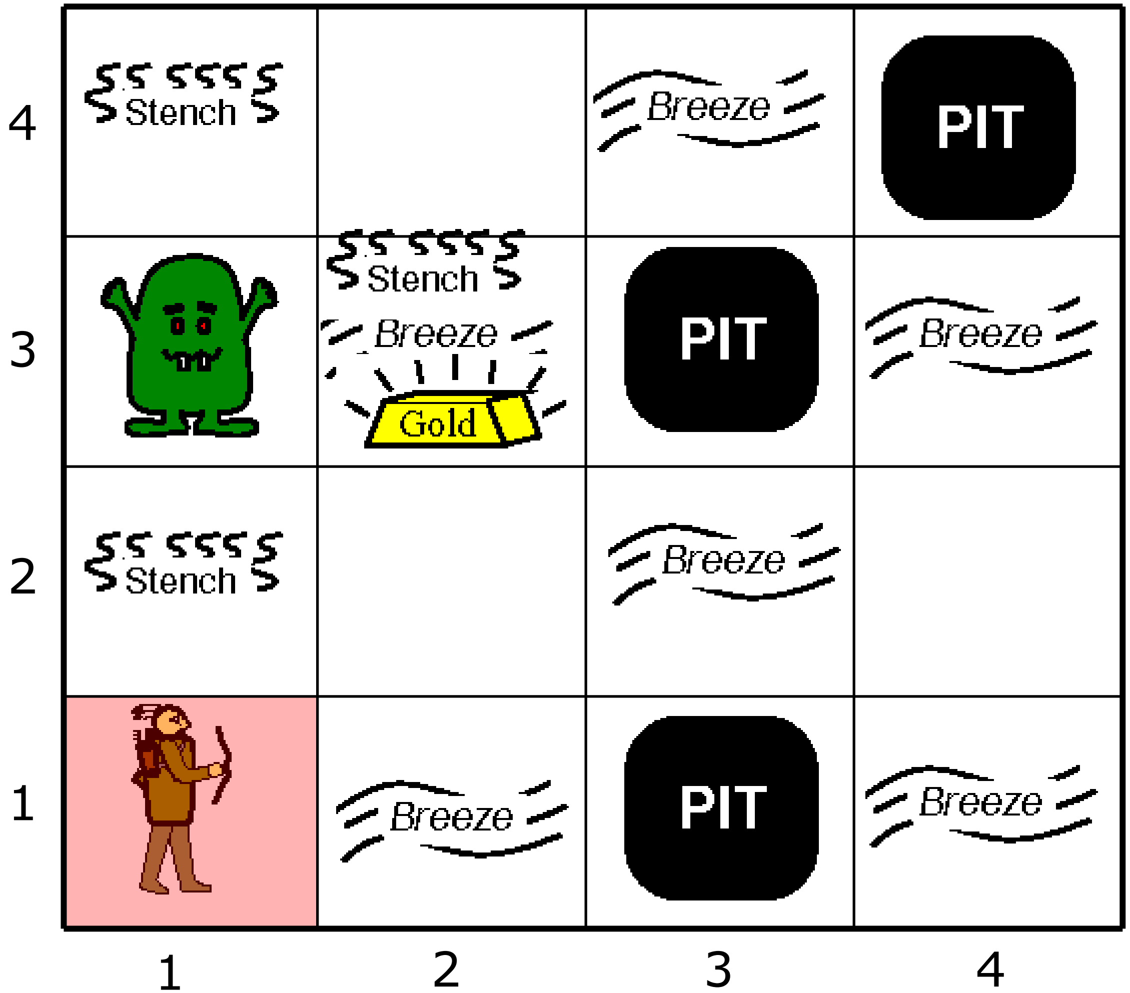
Motivation for Knowledge Representation

- Search algorithms discussed previously can be called *meta-programming*
  - but it still is programming
  - the code needs to be specialised for every concrete application according
- We need something more general
  - specify only the rules of the game
  - use „out-of-the-box" *reasoning engine*

The Wumpus World

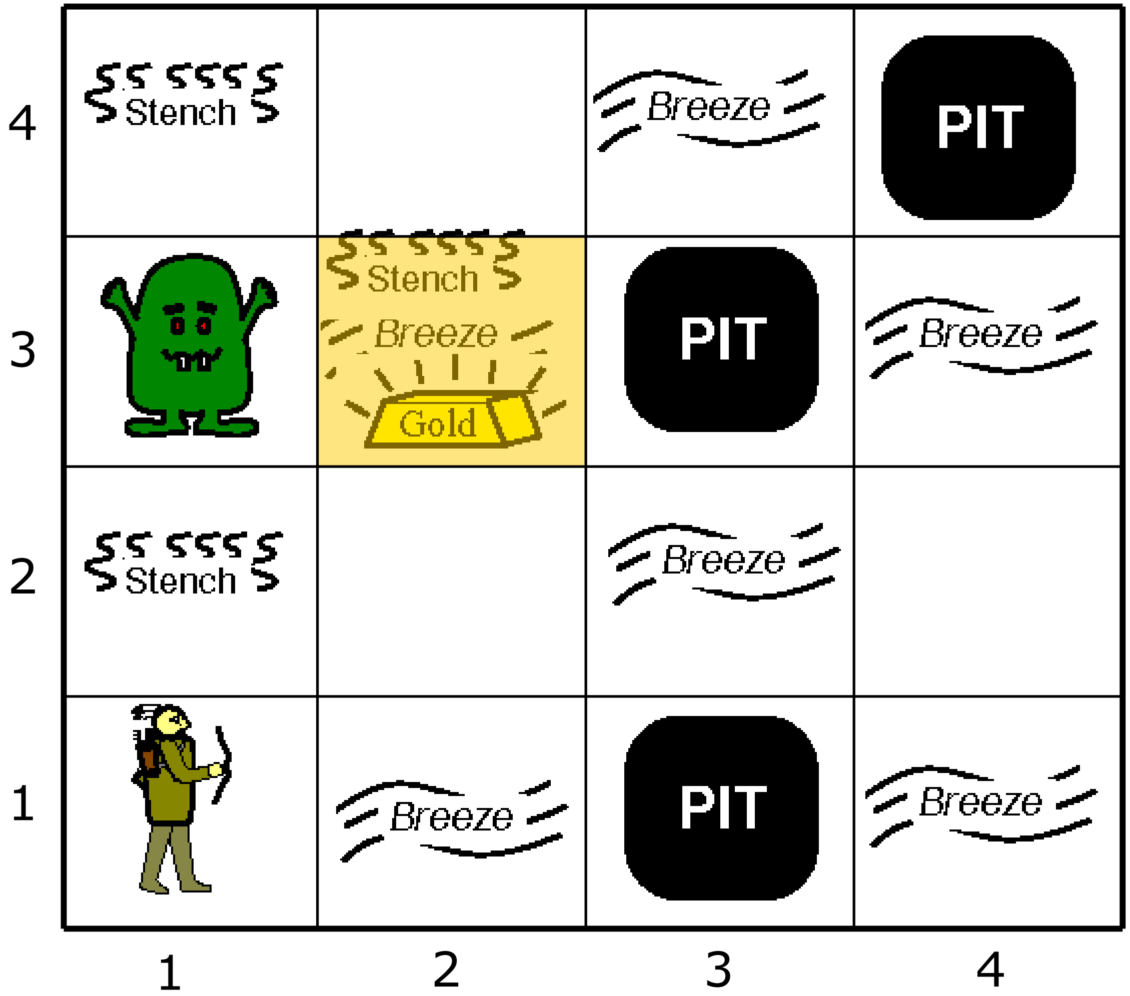


The Wumpus World



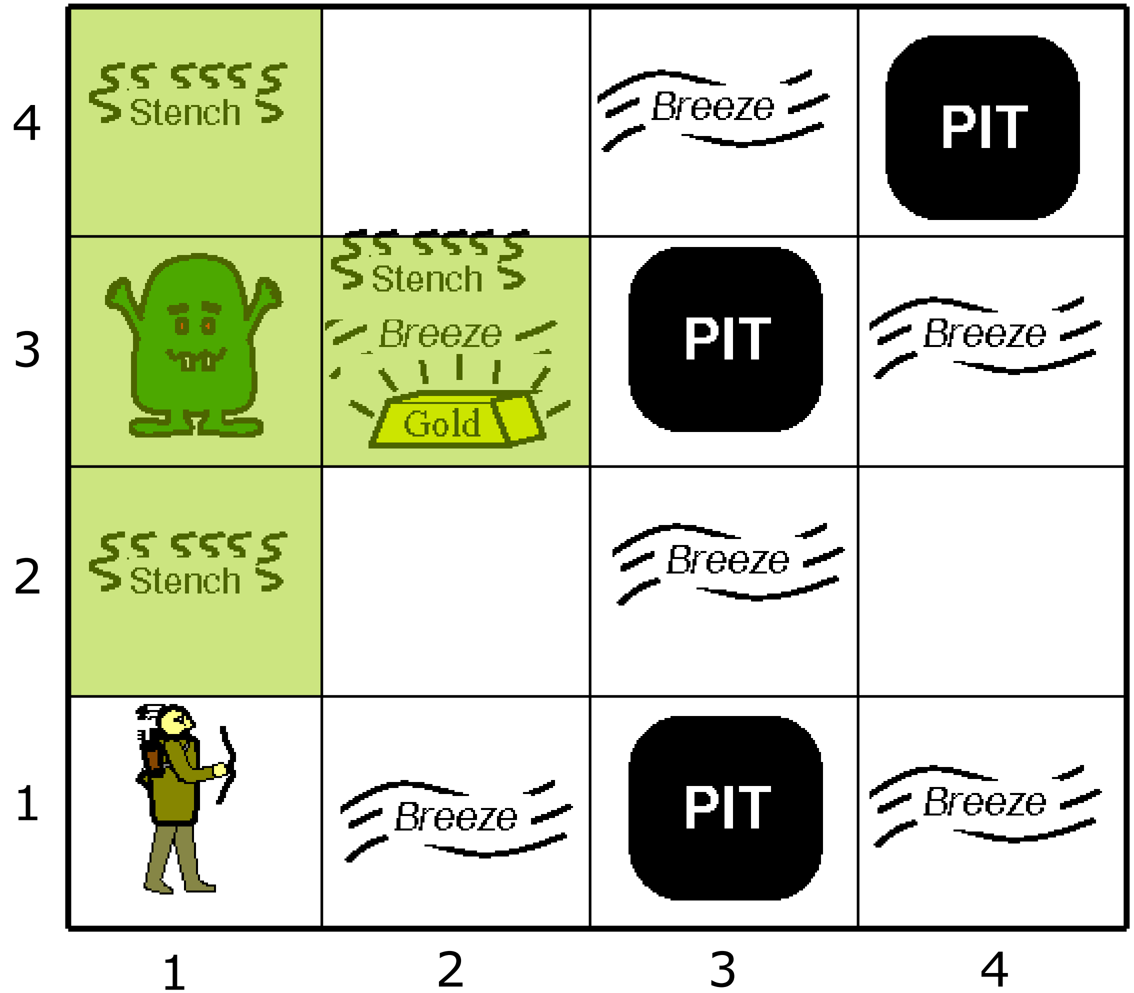
Start position = (1,1)
Always safe

The Wumpus World



Goal: Get the gold

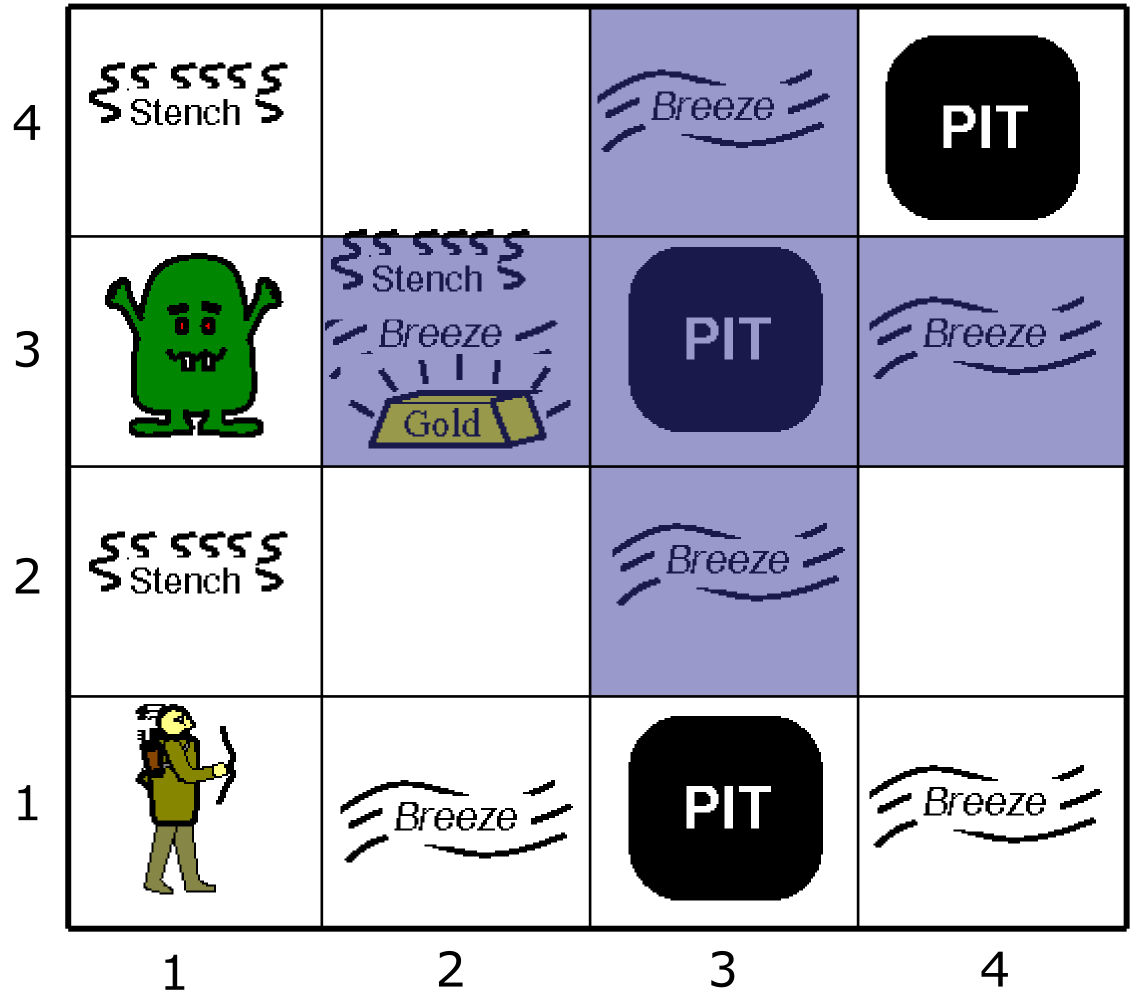
The Wumpus World



The environment is static: the Wumpus doesn't move around

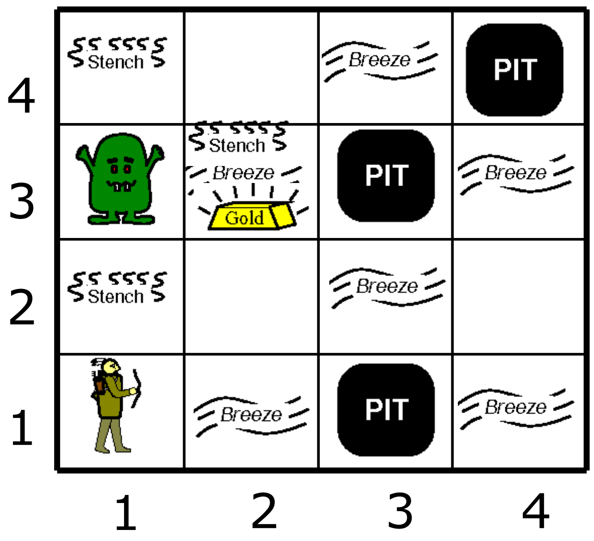
Problem 1: Big, hairy, smelly, dangerous Wumpus. Will eat you if you run into it, but you can smell it a block away.

The Wumpus World



Problem 2: Big, bottomless pits where you fall down.
You can feel the breeze when you are near them.

The Wumpus World



PEAS description

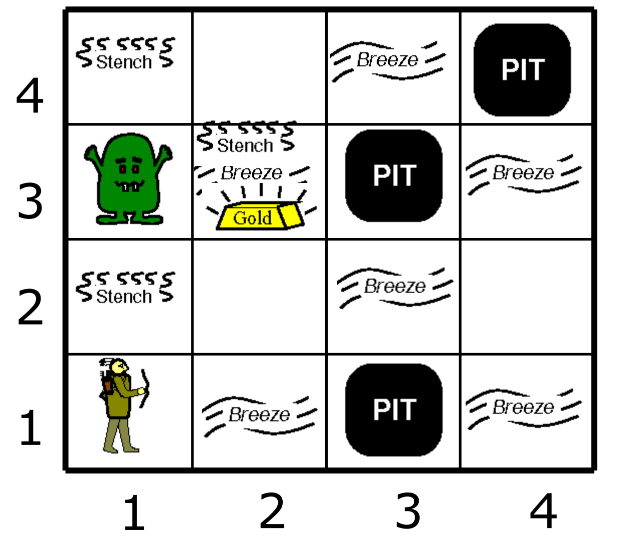
Performance measure:
- +1000 for gold
- -1000 for being eaten or falling down pit
- -1 for each action
- -10 for using the arrow

Environment:
4×4 grid of ”rooms”, each ”room” can be empty, with gold, occupied by Wumpus, or with a pit.

Acuators:
Move forward, turn left 90°, turn right 90°
Grab, shoot

Sensors:
Olfactory – stench from Wumpus
Touch – breeze (pits) & hardness (wall)
Vision – see gold
Auditory – hear Wumpus scream when killed

The Wumpus World



PEAS description

Performance measure:

+1000 for gold
-1000 for being eaten or falling down pit
-1 for each action
-10 for using the arrow

Environment:

4×4 grid of "rooms", each "room" can be empty, with gold, occupied by Wumpus, or with a pit.

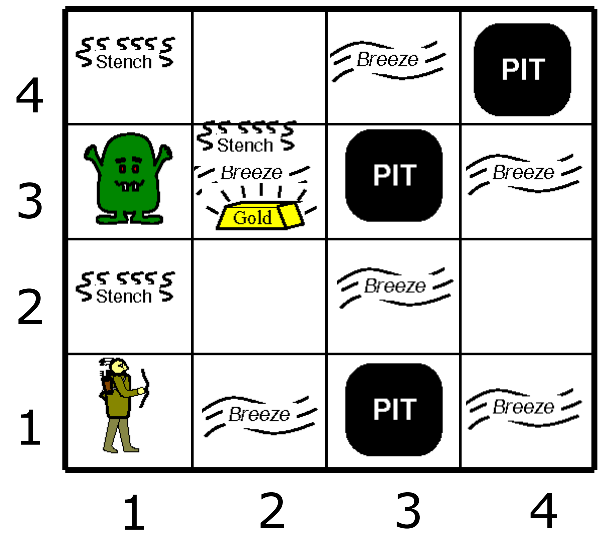
Acuators:

Move forward, turn left 90°, turn right 90°
Grab, shoot

Sensors:

Olfactory – stench from Wumpus
Touch – breeze (pits) & hardness (wall)
Vision – see gold
Auditory – hear Wumpus scream when killed

The Wumpus World



PEAS description

Performance measure:
+1000 for gold
-1000 for being eaten or falling down pit
-1 for each action
-10 for using the arrow

Environment:
4×4 grid of ”rooms”, each ”room” can be empty, with gold, occupied by Wumpus, or with a pit.

Acuators:
Move forward, turn left 90°, turn right 90°
Grab, shoot

Sensors:
Olfactory – stench from Wumpus
Touch – breeze (pits) & hardness (wall)
Vision – see gold
Auditory – hear Wumpus scream when killed

$$\boldsymbol{\alpha} = \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \\ \alpha_5 \end{pmatrix} = \begin{pmatrix} \textit{forward} \\ \textit{turn left} \\ \textit{turn right} \\ \textit{grab} \\ \textit{shoot} \end{pmatrix}, \alpha_i \in \{0,1\}$$

The Wumpus World

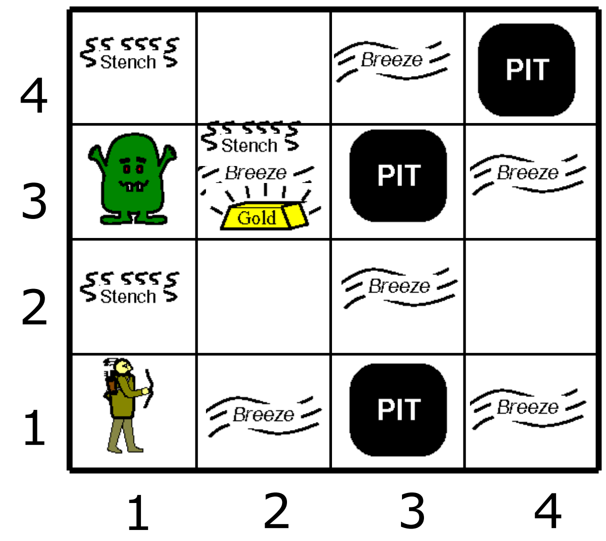
PEAS description

Performance measure:
+1000 for gold
-1000 for being eaten or falling down pit
-1 for each action
-10 for using the arrow

Environment:
4×4 grid of ”rooms”, each ”room” can be empty, with gold, occupied by Wumpus, or with a pit.

Actuators:
Move forward, turn left 90°, turn right 90°
Grab, shoot

Sensors:
Olfactory – stench from Wumpus
Touch – breeze (pits) & hardness (wall)
Vision – see gold
Auditory – hear Wumpus scream when killed

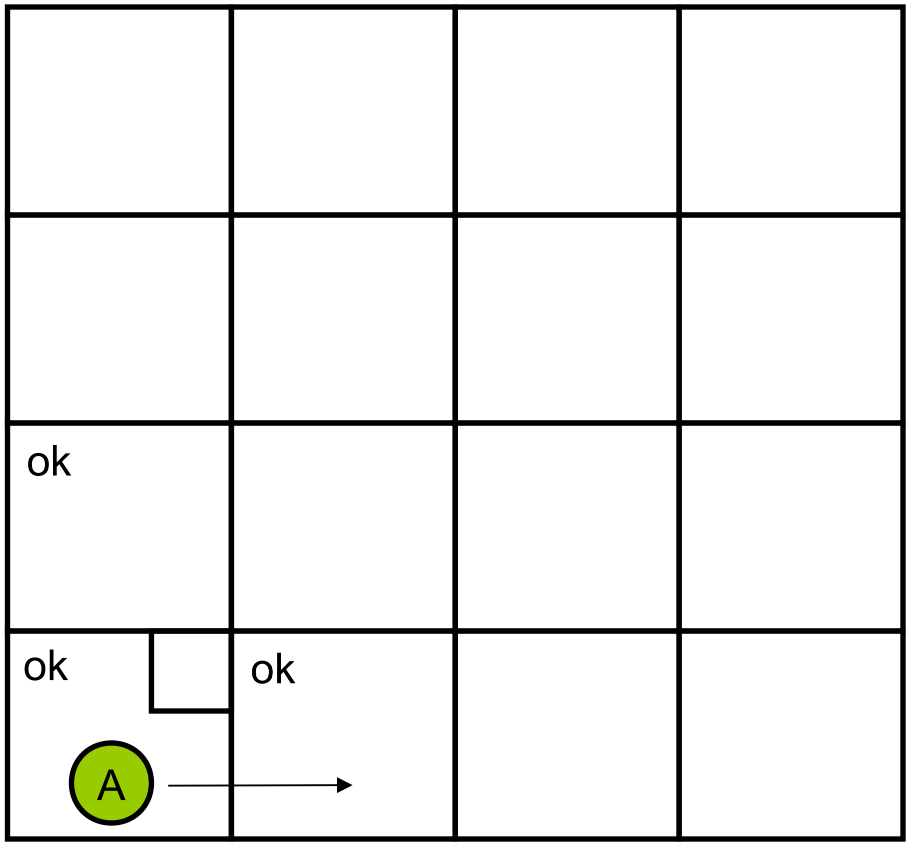


$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{pmatrix} = \begin{pmatrix} stench \\ breeze \\ wall \\ glitter \\ scream \end{pmatrix}, x_i \in \{0,1\}$$

Exploring the Wumpus world

$$\mathbf{x}_{1,1} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Agent senses nothing
(no breeze, no smell,..)

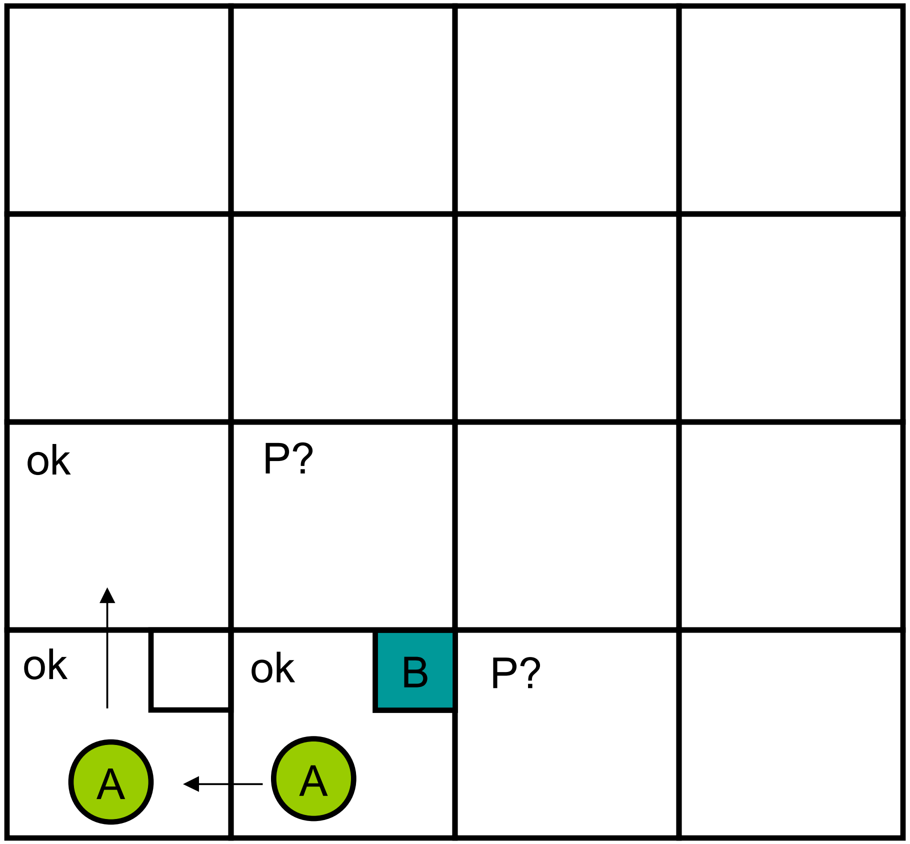


Slide adapted from V. Pavlovic

Exploring the Wumpus world

$$\mathbf{x}_{1,2} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

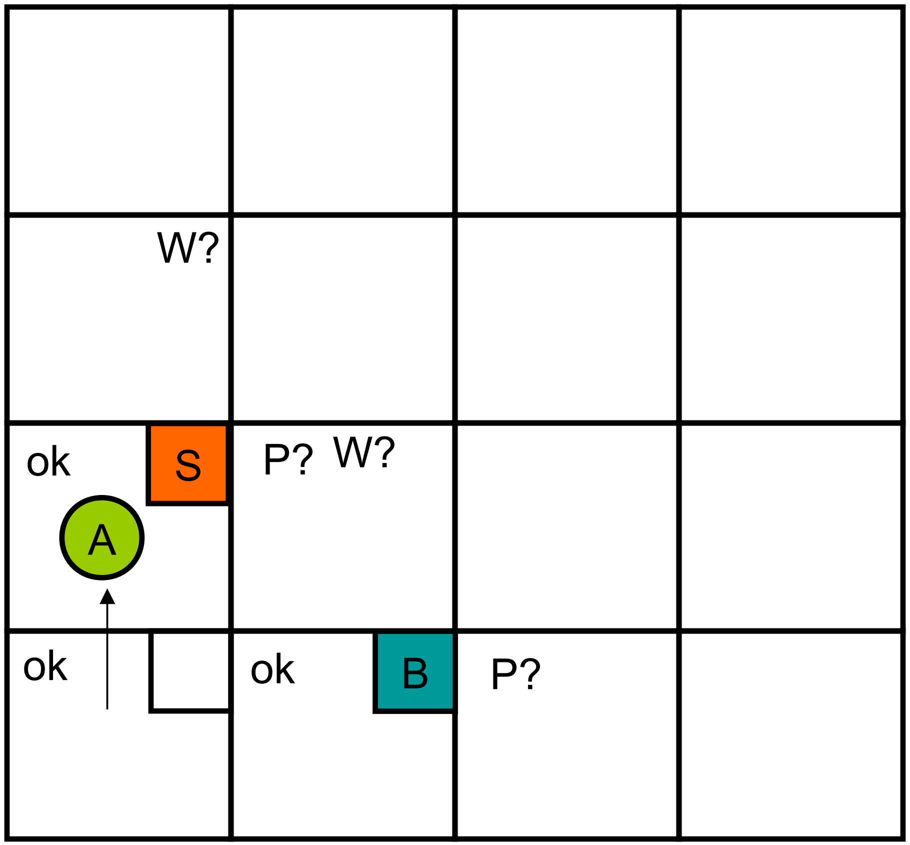
Agent feels a breeze



Slide adapted from V. Pavlovic

Exploring the Wumpus world

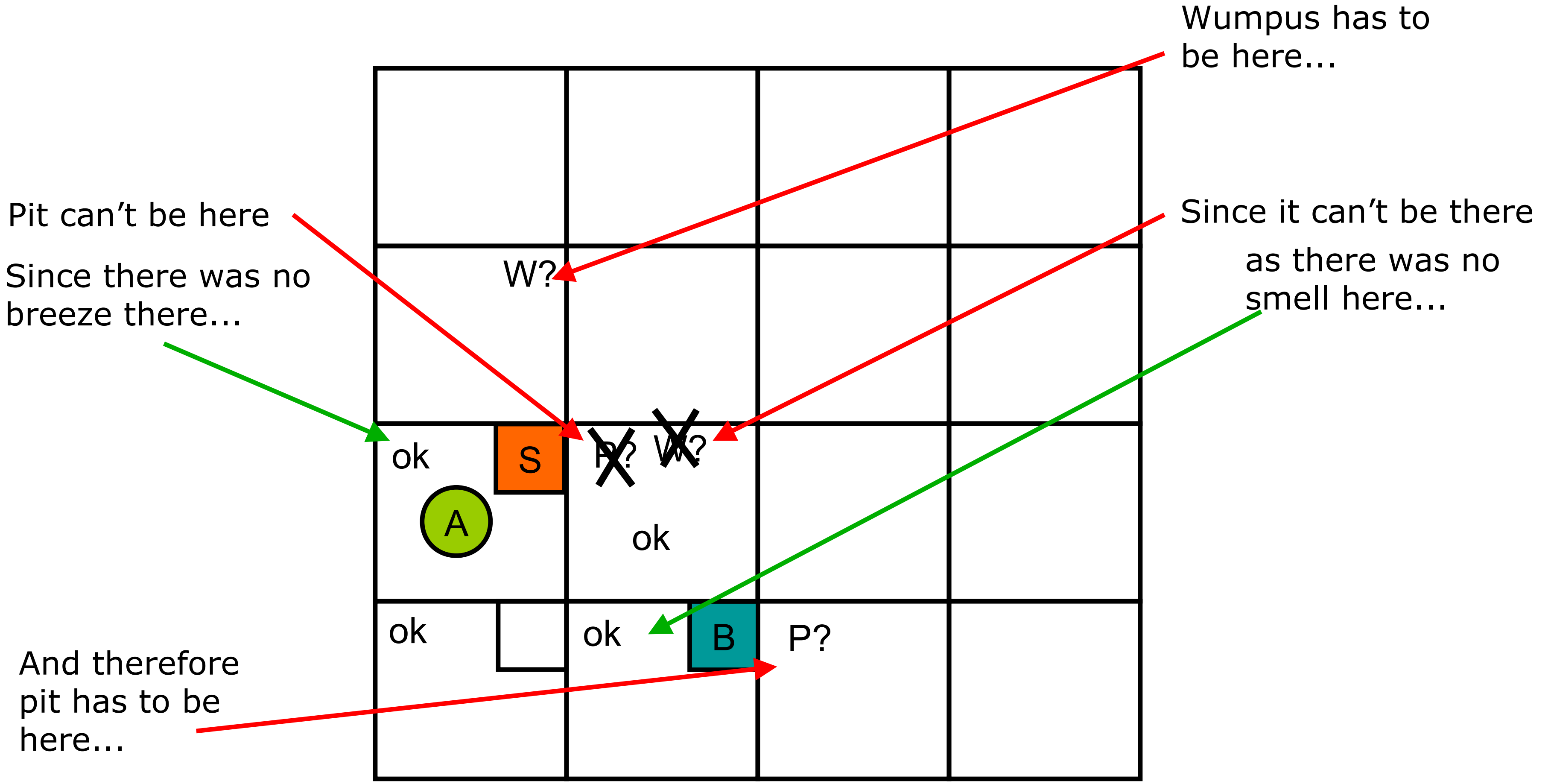
Agent feels a foul smell



$$\mathbf{x}_{2,1} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Slide adapted from V. Pavlovic

Exploring the Wumpus world

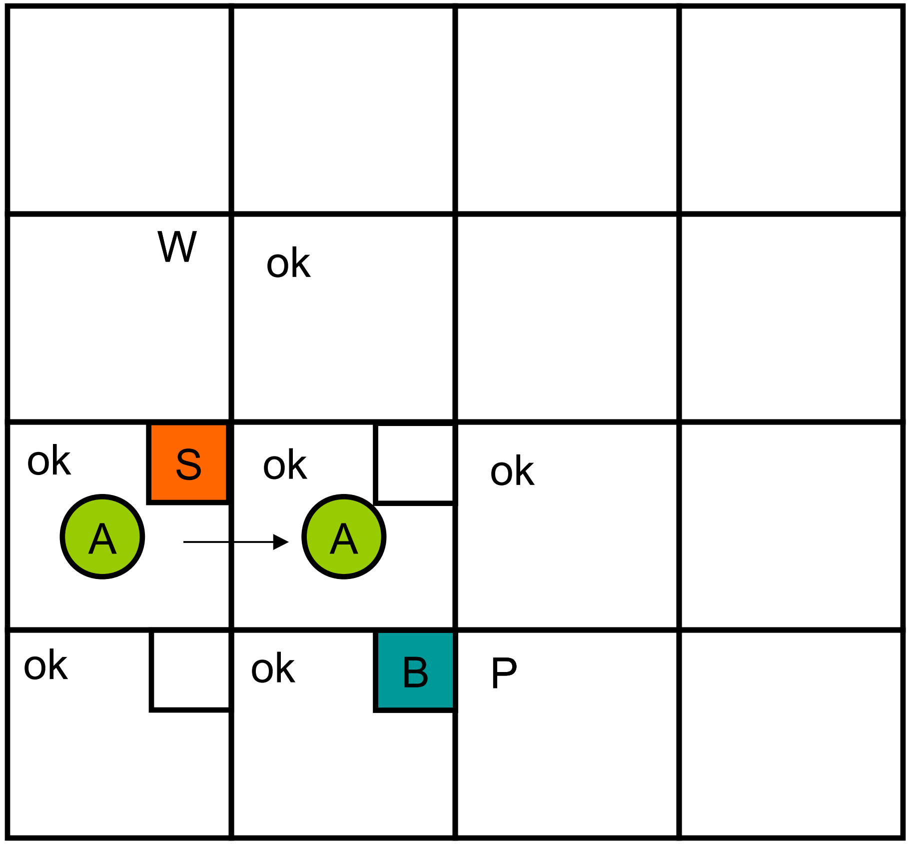


Slide adapted from V. Pavlovic

Exploring the Wumpus world

Agent senses nothing
(no breeze, no smell,..)

$$\mathbf{x}_{2,2} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

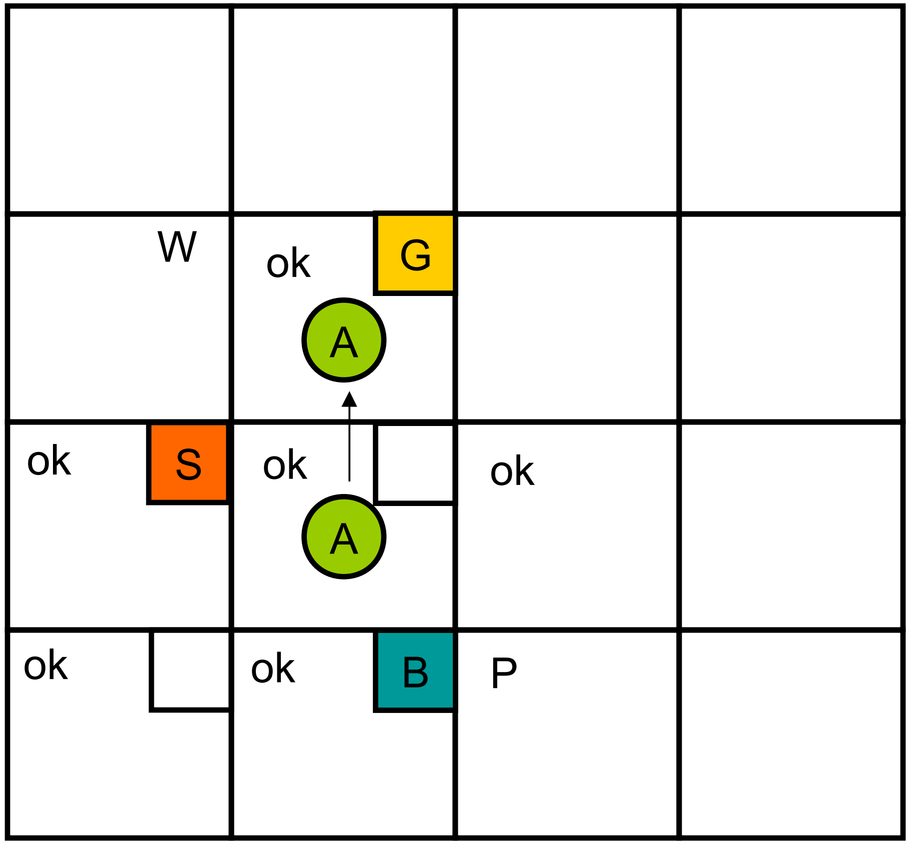


Slide adapted from V. Pavlovic

Exploring the Wumpus world

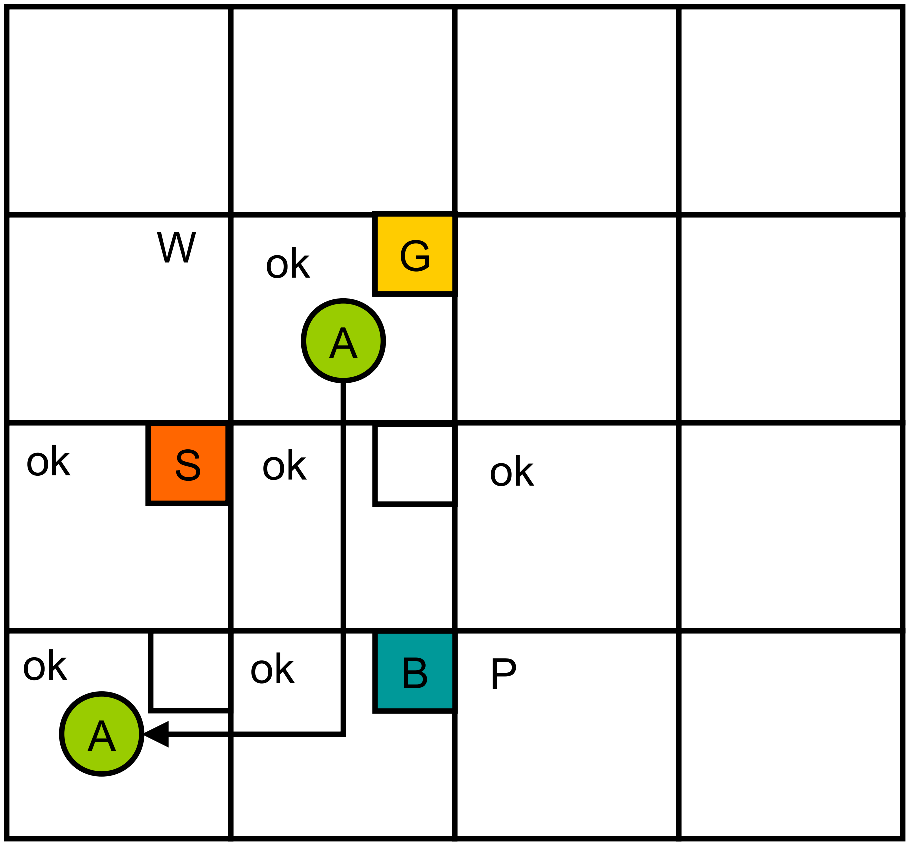
Agent senses breeze, smell, and sees gold!

$$\mathbf{x} = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 1 \\ 0 \end{pmatrix}$$



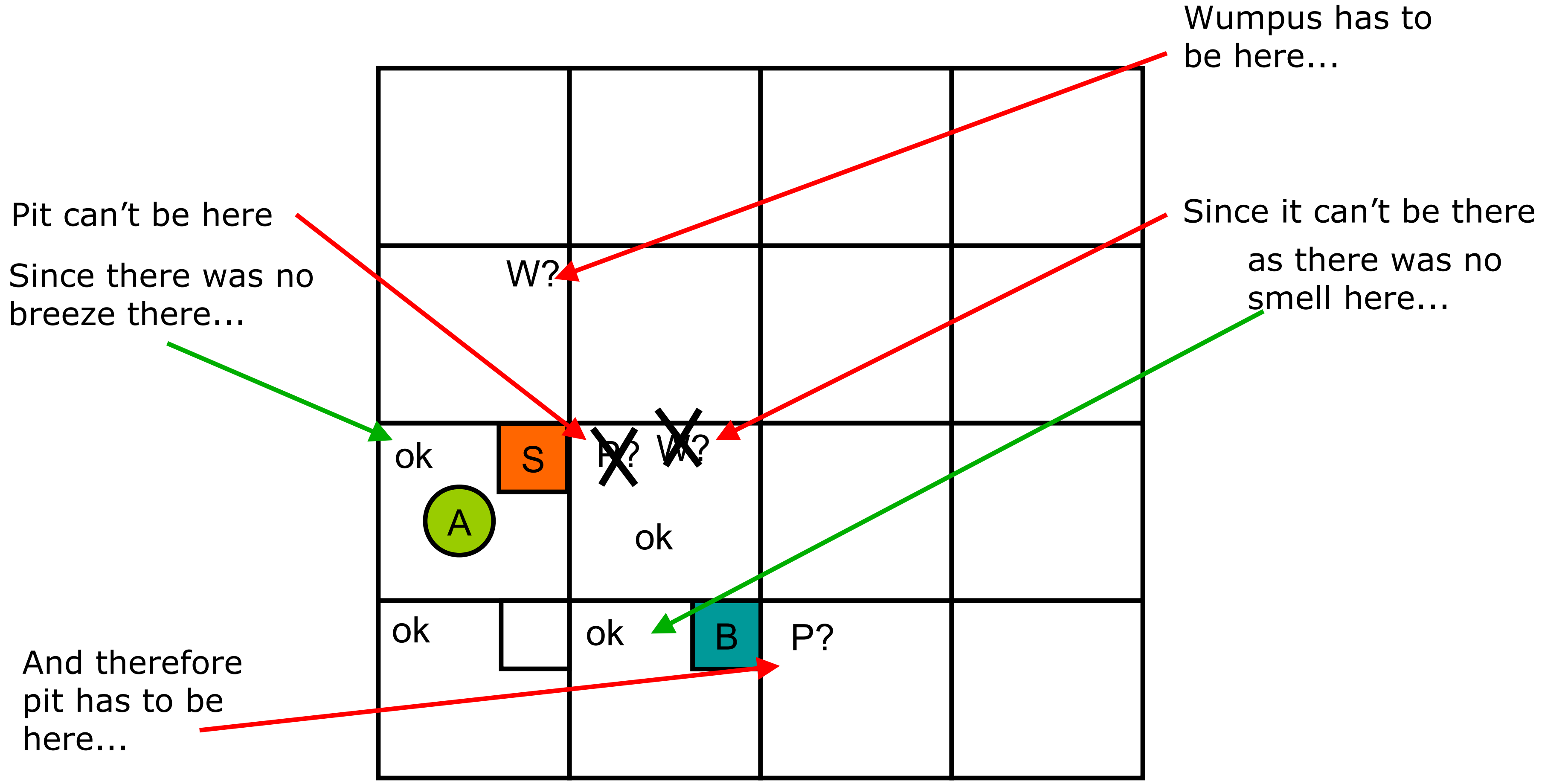
Slide adapted from V. Pavlovic

Exploring the Wumpus world



Grab the gold and get out!

Exploring the Wumpus world



How do we automate this kind of reasoning?
How can we make computers figure it out on their own?

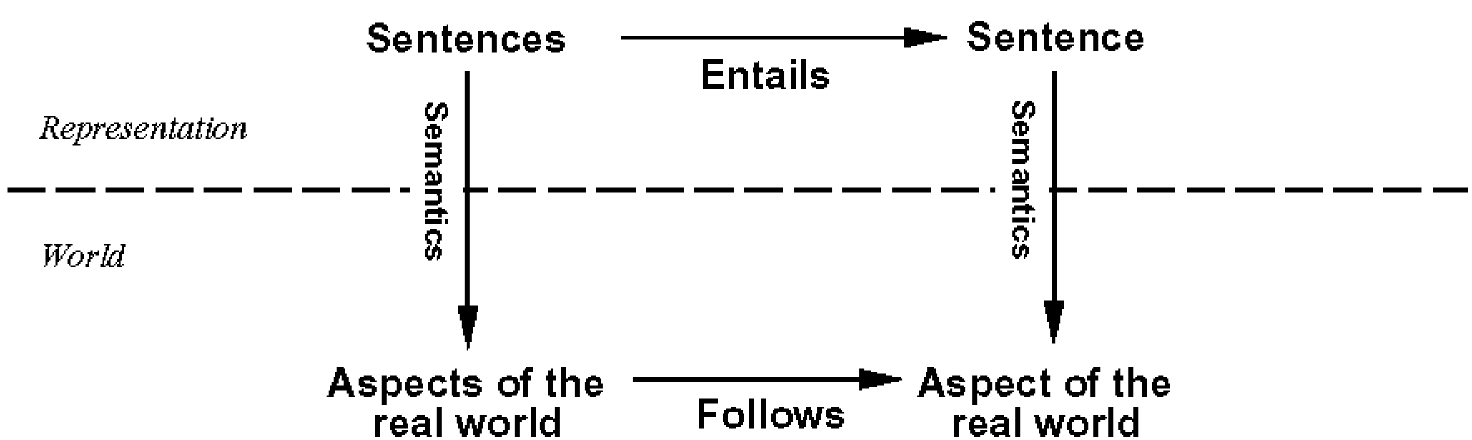
Slide adapted from V. Pavlovic

Logic

Logic is a formal language for representing information in such a way that conclusions can be drawn

A logic has

- **Syntax** that specifies symbols in the language and how they can be combined to form *sentences*
- **Semantics** that specifies what facts in the world these sentences refer to and assigns *truth values* to them based on their meaning in the world.
- **Inference procedure**, a mechanical method for computing (deriving) new (true) sentences from existing (known) sentences.



Entailment

$$A \vDash B$$

The sentence A *entails* the sentence B

- If A is true, then B must also be true
- B is a ”logical consequence” of A

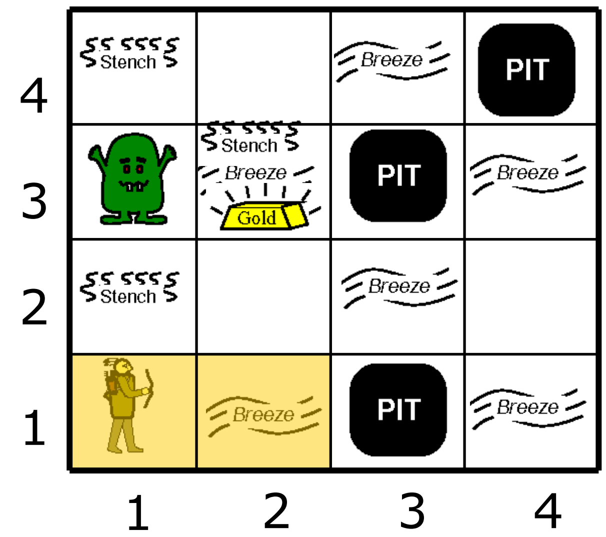
Let’s explore this concept a bit...

Example: Wumpus entailment

Agent's knowledge base (KB) after having visited (1,1) and (1,2):

1) The rules of the game (PEAS)
2) Nothing in (1,1)
3) Breeze in (1,2)

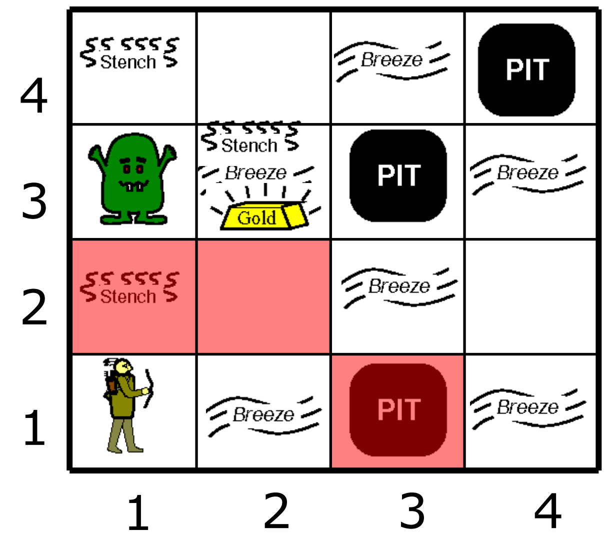
Which models (states of the world) match these observations?



$$\mathbf{x}_{1,1} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \quad \mathbf{x}_{1,2} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Example: Wumpus entailment

We only care about neighboring rooms, i.e. {(2,1),(2,2),(1,3)}. We can't know anything about the other rooms.



We care about pits, because we have detected a breeze. We don't want to fall down a pit.

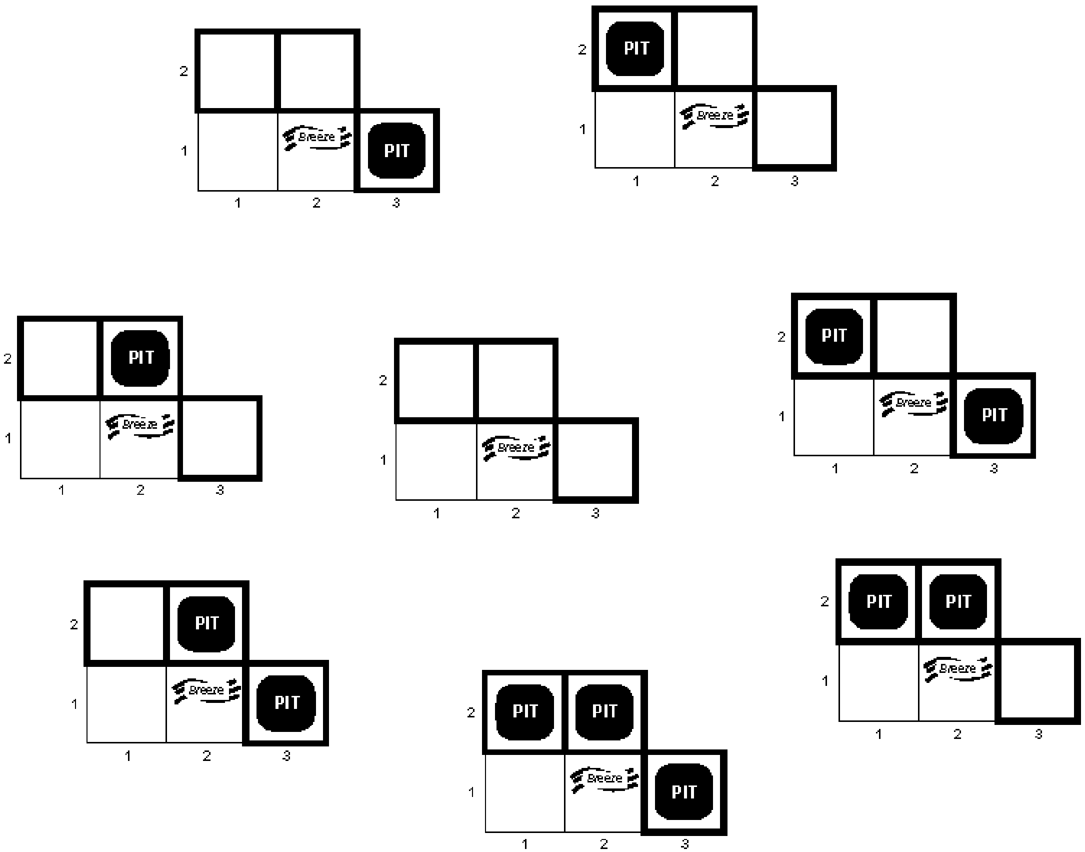
There are $2^3=8$ possible arrangements of {pit, no pit} in the three neighboring rooms.

Possible conclusions:

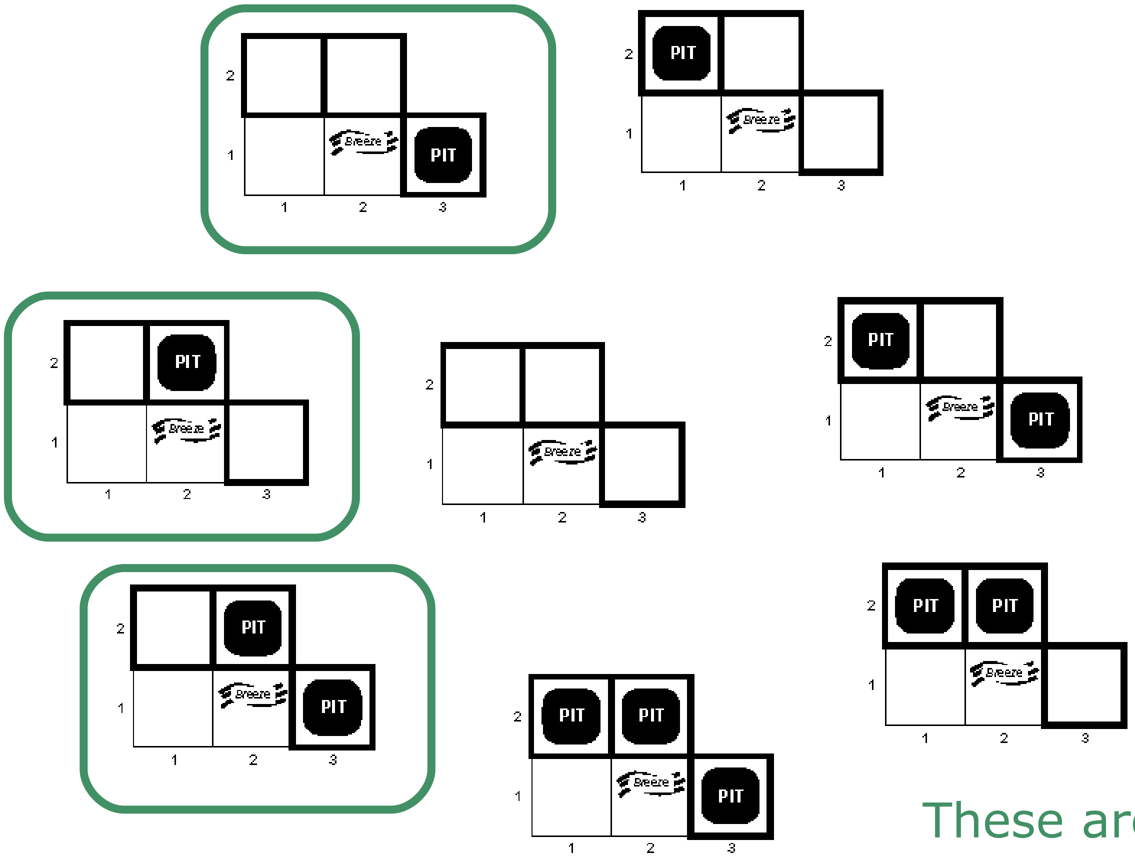
α_1 : There is no pit in (2,1)

α_2 : There is no pit in (2,2)

α_3 : There is no pit in (1,3)

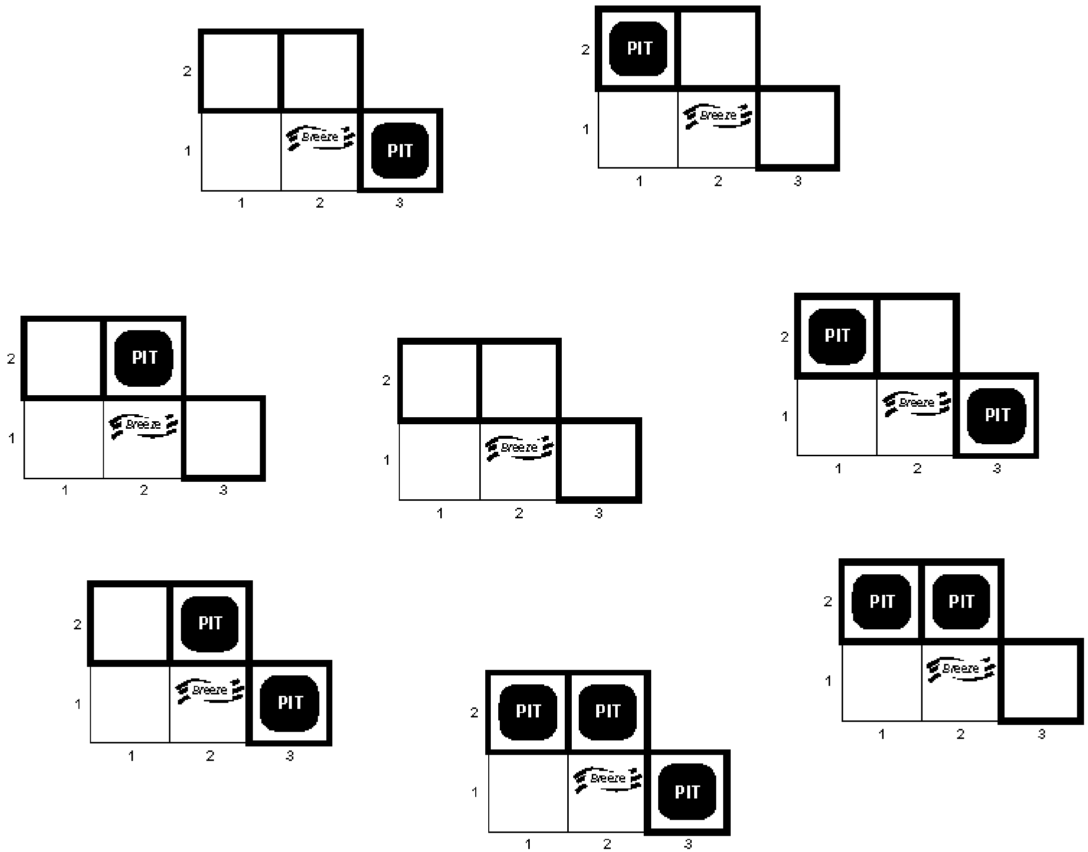


The eight possible situations...



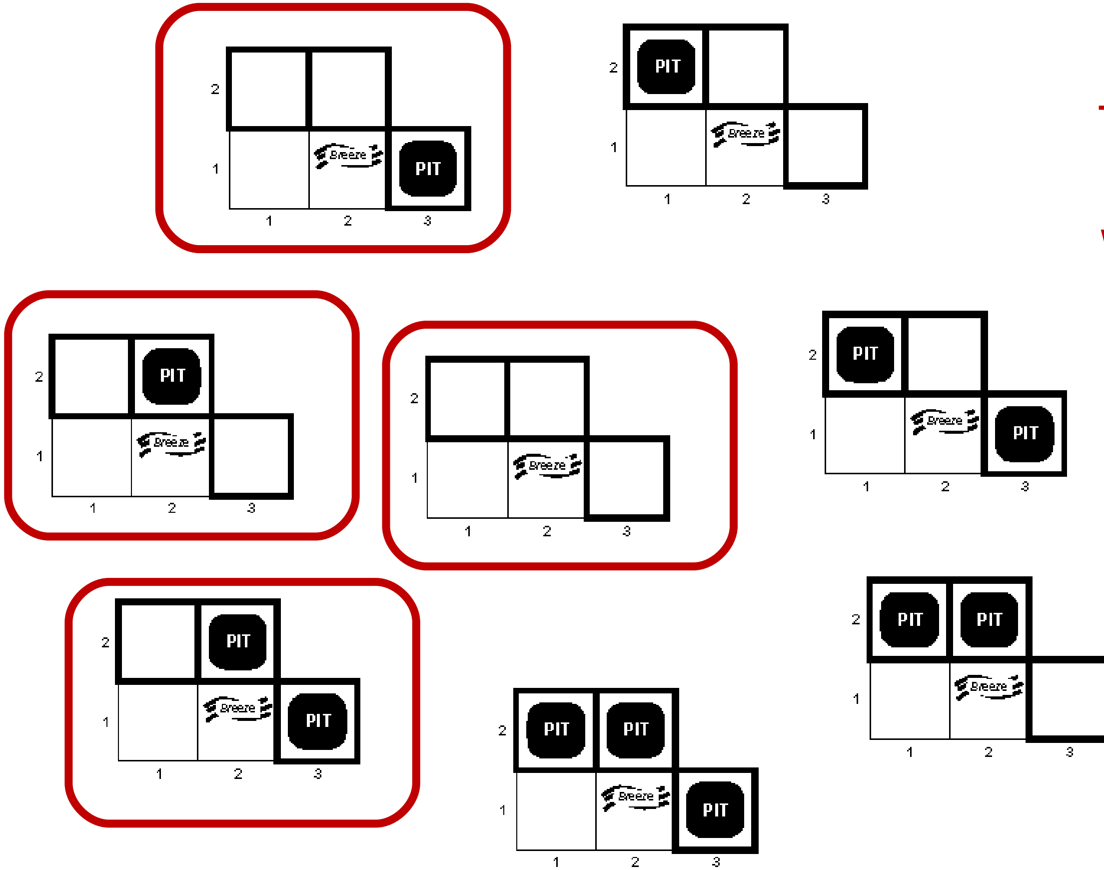
The eight possible situations...

These are the ones that agree with our Knowledge Base (KB), i.e. the rules of the game and our observations.



α_1 : There is no pit in (2,1)

...let's explore this conclusion



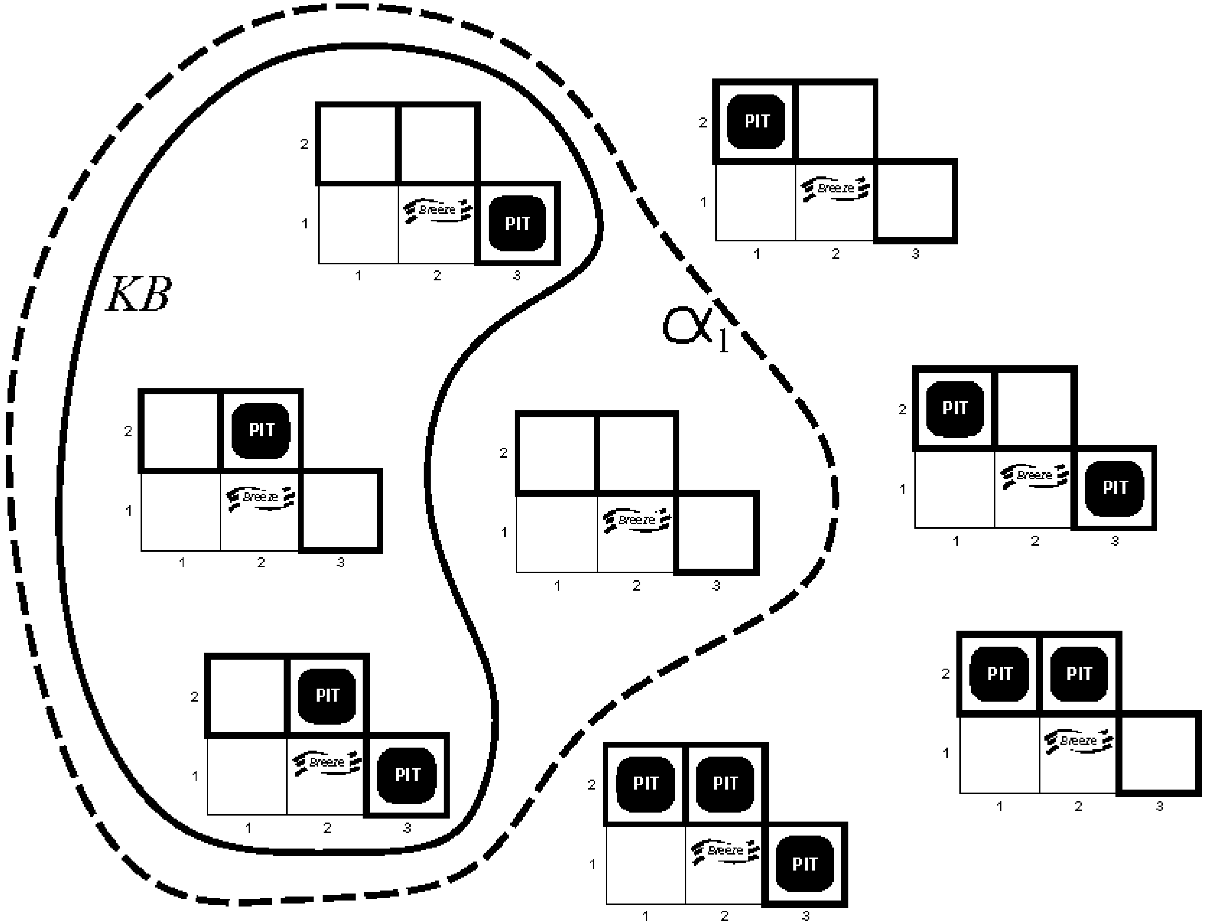
These are the situations where α_1 is true.

α_1 : There is no pit in (2,1)

...let’s explore this conclusion

If KB is true, then α_1 is also true.
KB entails α_1.

$$KB \vDash \alpha_1$$



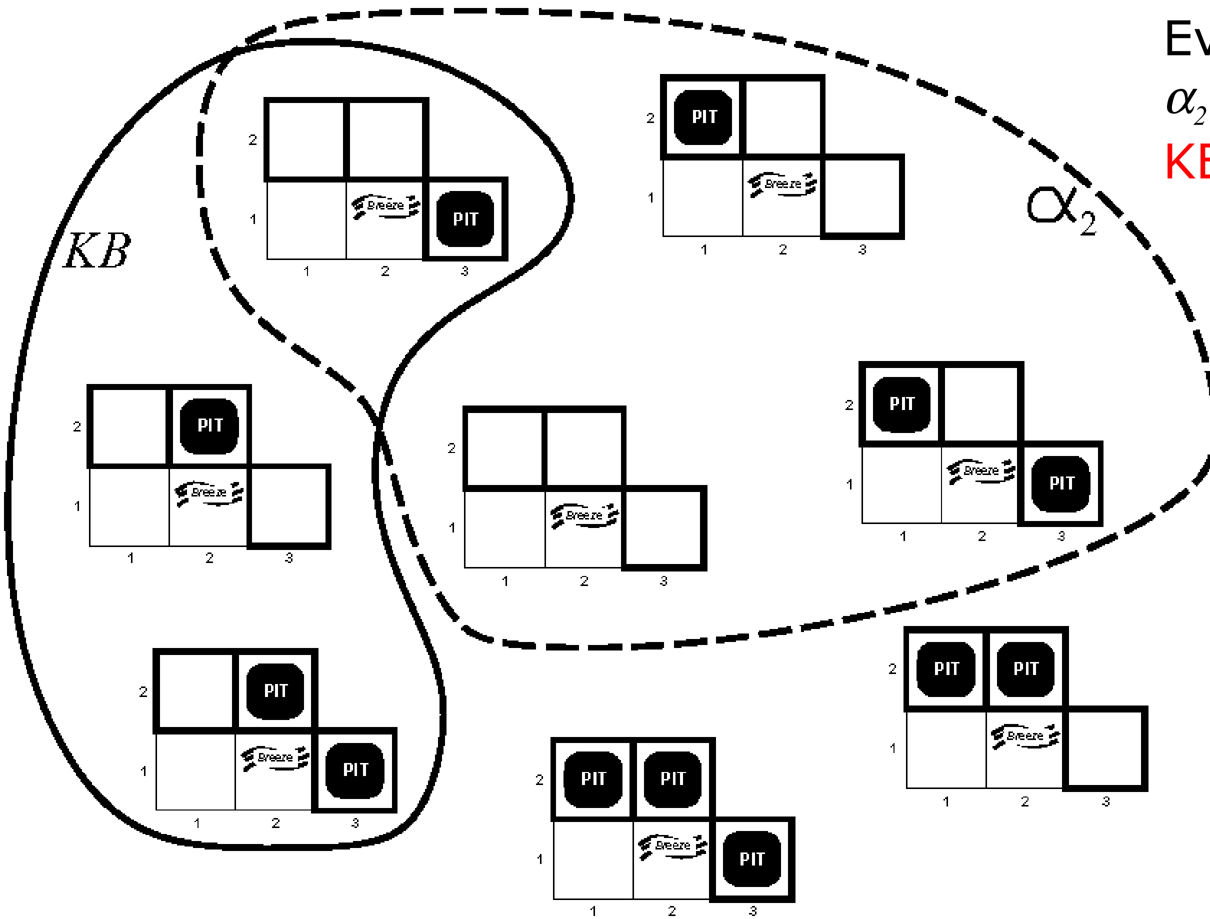
α_1 : There is no pit in (2,1)

KB = The set of models that agrees with the knowledge base (the observed facts) [The KB is true in these models]

α_1 = The set of models that agrees with conclusion α_1 [conclusion α_1 is true in these models]

Even if KB is true,
α_2 can be false.

KB does not entail α_2

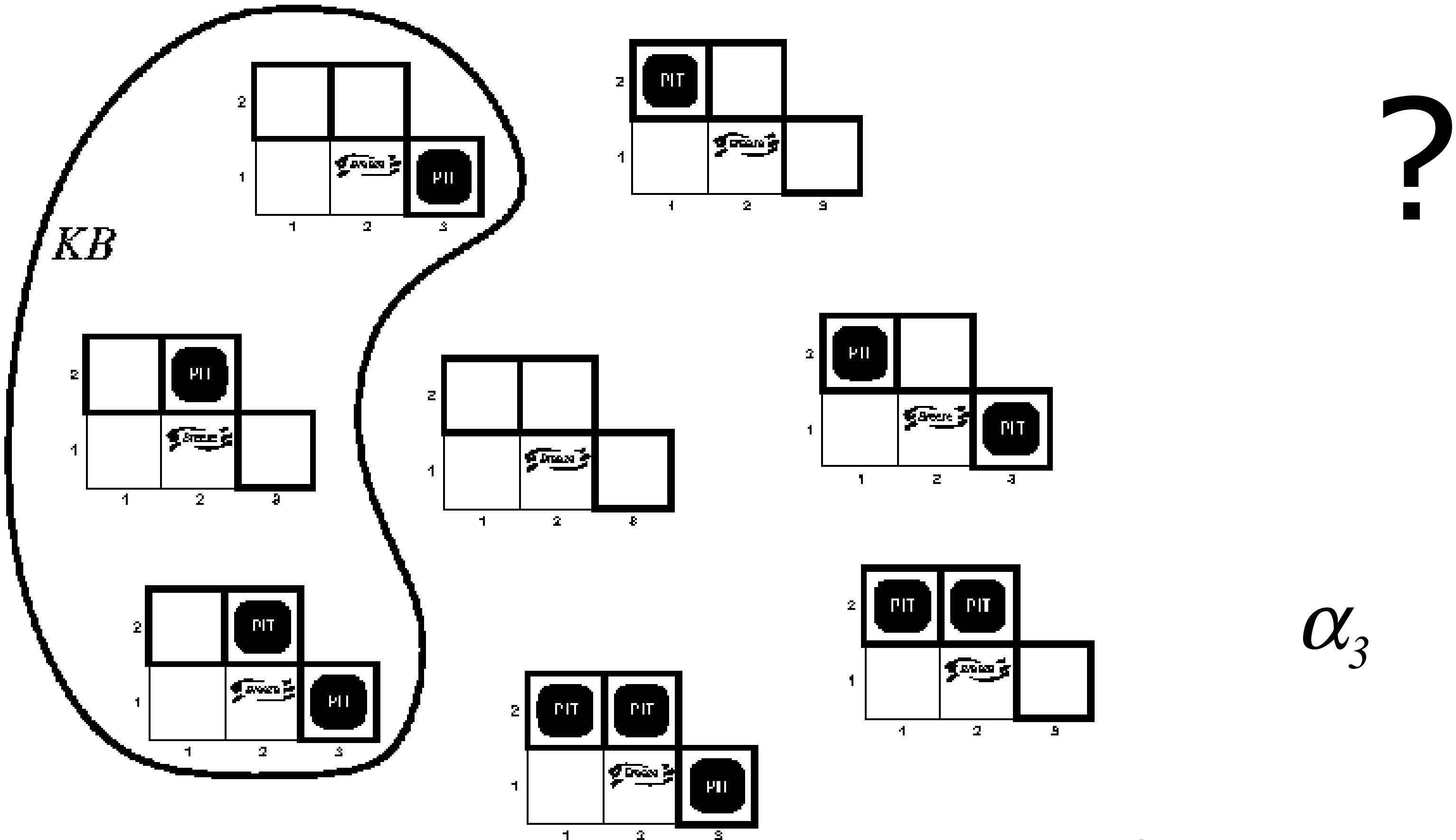


$$KB \nvDash \alpha_2$$

α_2 : There is no pit in (2,2)

KB = The set of models that agrees with the knowledge base (the observed facts) [The KB is true in these models]

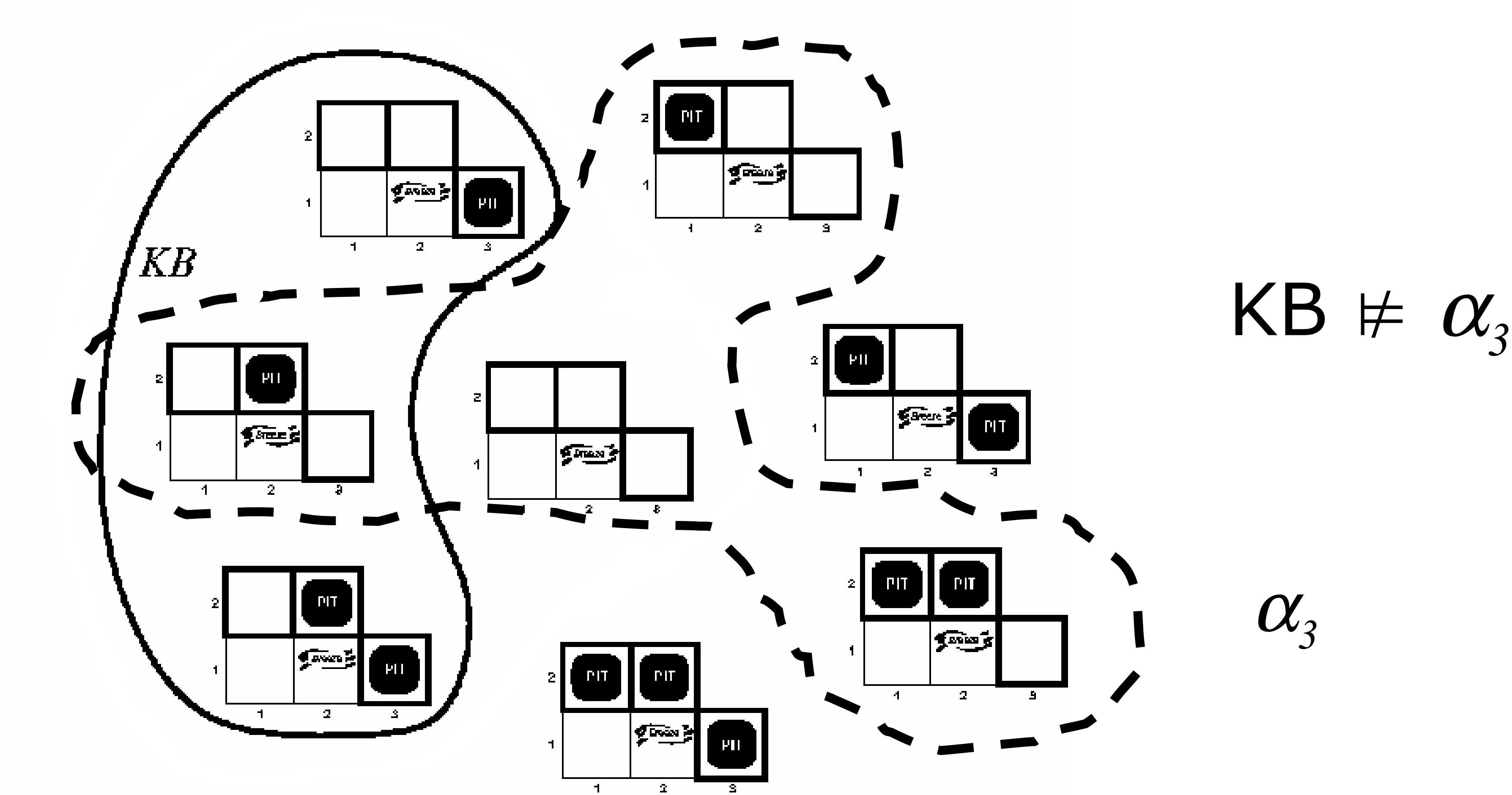
α_2 = The set of models that agrees with conclusion α_2 [conclusion α_2 is true in these models]



α_3 : There is no pit in (1,3)

KB = The set of models that agrees with the knowledge base (the observed facts) [The KB is true in these models]

α_3 = The set of models that agrees with conclusion α_3 [conclusion α_3 is true in these models]



α_3 : There is no pit in (1,3)

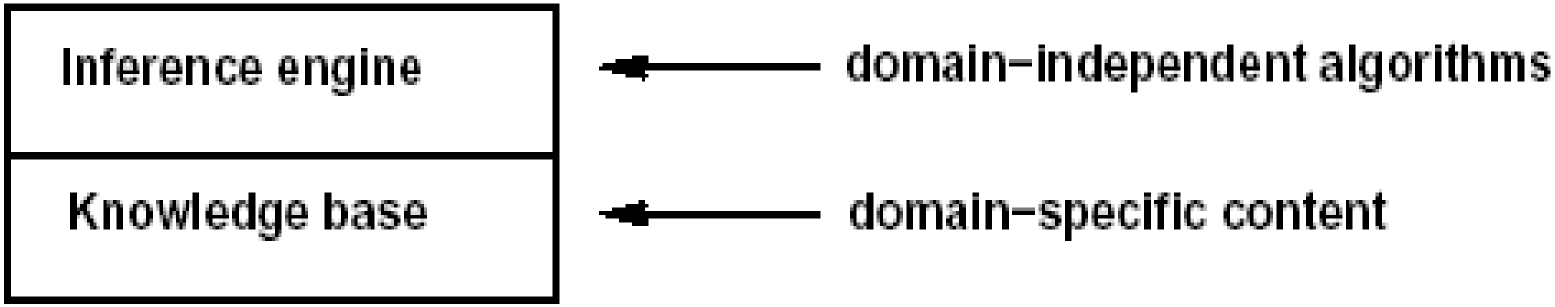
KB = The set of models that agrees with the knowledge base (the observed facts) [The KB is true in these models]

α_3 = The set of models that agrees with conclusion α_3 [conclusion α_3 is true in these models]

Inference engine

- We need an algorithm that produces the entailed conclusions automatically
  - for any user-defined Knowledge Base
- Entailment is the most important property in logic
  - most interesting things can be expressed using entailment
- We will call this algorithm, and it's implementation, an *inference engine*

Inference engine



$$\text{KB} \vdash_i \text{A}$$

"A is derived from KB by inference engine i"

- **Truth-preserving:** i only derives entailed sentences
- **Complete:** i derives all entailed sentences

We want inference engines that are both truth-preserving and complete

Propositional (boolean) logic
Syntax

Atomic sentence = a single propositional symbol
e.g. P, Q, P_{13}, W_{31}, G_{32}, T, F

Pit in room (1,3)

room (3,1)

Complex sentence = combination of simple sentences using *connectives*

$\neg$ (not) negation

$\wedge$ (and) conjunction

$\vee$ (or) disjunction

$\Rightarrow$ (implies) implication

$\Leftrightarrow$ (iff = if and only if) biconditional/logical equality

$P_{13} \wedge W_{31}$

$W_{31} \vee \neg W_{31}$

$W_{31} \Rightarrow S_{32}$

Precedence: $\neg, \wedge, \vee, \Rightarrow, \Leftrightarrow$

Propositional (boolean) logic
Semantics

Semantics: The rules for whether a sentence is true or false

- T (true) is true in every model
- F (false) is false in every model
- The truth values for other proposition symbols are specified in the model

Atomic sentences

- Truth values for complex sentences are specified according to the definitions of connectives
  - using a *truth table*

Boolean truth table

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
False	False					
False	True					
True	False					
True	True					

Please complete this table...

Boolean truth table

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

Boolean truth table

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

Not P is the opposite of P

Boolean truth table

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

$P \wedge Q$ is true only when both P and Q are true

Boolean truth table

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

$P \vee Q$ is true when either P or Q is true

Boolean truth table

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

$P \Rightarrow Q$: If P is true then we claim that Q is true, otherwise we make no claim

Boolean truth table

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

$P \Leftrightarrow Q$ is true when the truth values for P and Q are identical

Boolean truth table

P	Q	$P \oplus Q$
False	False	
False	True	
True	False	
True	True	

The exlusive or (XOR) is different from the OR

Boolean truth table

P	Q	P⊕Q
False	False	False
False	True	True
True	False	True
True	True	False

The exlusive or (XOR) is different from the OR

Example: Wumpus KB

Interesting sentences [tell us what is in neighbour squares]

Knowledge base

R_1: $\neg P_{11}$

R_2: $\neg B_{11}$

R_3: $\neg W_{11}$

R_4: $\neg S_{11}$

R_5: $\neg G_{11}$

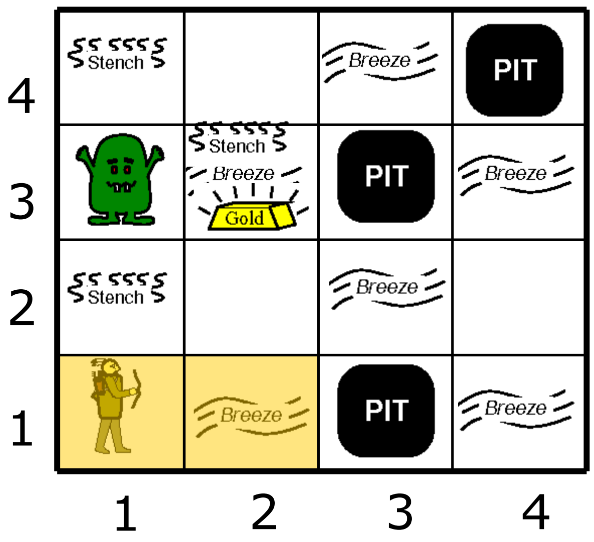
R_6: B_{12}

R_7: $\neg P_{12}$

R_8: $\neg S_{12}$

R_9: $\neg W_{12}$

R_{10}: $\neg G_{12}$



1. Nothing in (1,1)
2. Breeze in (1,2)

$$KB = R_1 \wedge R_2 \wedge R_3 \wedge R_4 \wedge R_5 \wedge R_6 \wedge R_7 \wedge R_8 \wedge R_9 \wedge R_{10}$$

Plus the rules of the game

Example: Wumpus KB

We infer this from the rules of the game

<u>Knowledge base</u>

R_1: $\neg P_{11}$

R_2: $\neg B_{11} \Leftrightarrow \boxed{\neg(P_{21} \vee P_{12})}$

R_3: $\neg W_{11}$

R_4: $\neg S_{11} \Leftrightarrow \boxed{\neg(W_{21} \vee W_{12})}$

R_5: $\neg G_{11}$

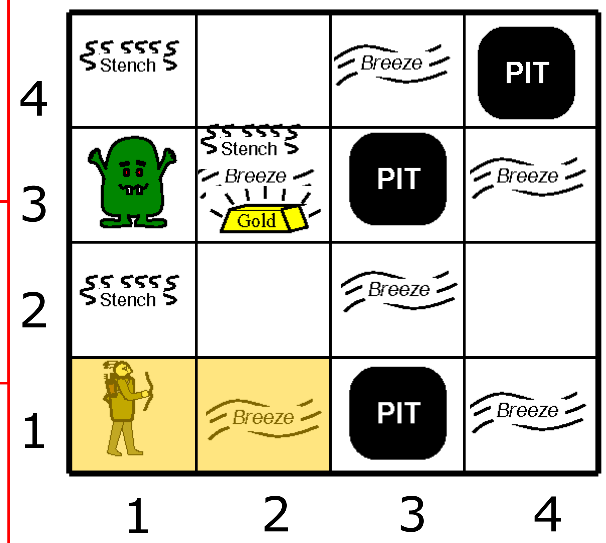
R_6: $B_{12} \Leftrightarrow \boxed{(P_{11} \vee P_{22} \vee P_{13})}$

R_7: $\neg P_{12}$

R_8: $\neg S_{12} \Leftrightarrow \boxed{\neg(W_{11} \vee W_{21} \vee W_{13})}$

R_9: $\neg W_{12}$ (already in R_4)

R_{10}: $\neg G_{12}$



1. Nothing in (1,1)
2. Breeze in (1,2)

$KB = R_1 \wedge R_2 \wedge R_3 \wedge R_4 \wedge R_5 \wedge R_6 \wedge R_7 \wedge R_8 \wedge R_9 \wedge R_{10}$

Plus the rules of the game

Inference by enumerating models

What is in squares (1,3), (2,1), and (2,2)?

#	W_{21}	W_{22}	W_{13}	P_{21}	P_{22}	P_{13}	R_2	R_4	R_6	R_8
1	0	0	0	0	0	0	1	1	0	1
2	0	0	0	0	0	1	1	1	1	1
3	0	0	0	0	1	0	1	1	1	1
4	0	0	0	0	1	1	1	1	1	1
5	0	0	0	1	0	0	0	1	0	1
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
63	0	1	1	1	1	1	0	1	1	0
64	1	1	1	1	1	1	0	0	1	0

KB true

KB
(interesting sentences)

We have 6 interesting sentences: W_{21}, W_{22}, W_{13}, P_{21}, P_{22}, P_{13} : 2^6 = 64 comb.

Inference by enumerating models

What is in squares (1,3), (2,1), and (2,2)?

#	W_{21}	W_{22}	W_{13}	P_{21}	P_{22}	P_{13}	R_2	R_4	R_6	R_8
1	0	0	0	0	0	0	1	1	0	1
2	0	0	0	0	0	1	1	1	1	1
3	0	0	0	0	1	0	1	1	1	1
4	0	0	0	0	1	1	1	1	1	1
5	0	0	0	1	0	0	0	1	0	1
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
63	0	1	1	1	1	1	0	1	1	0
64	1	1	1	1	1	1	0	0	1	0

KB true

What do we deduce from this?

Inference by enumerating models

What is in squares (1,3), (2,1), and (2,2)?

#	W_{21}	W_{22}	W_{13}	P_{21}	P_{22}	P_{13}	R_2	R_4	R_6	R_8
1	0	0	0	0	0	0	1	1	0	1
2	0	0	0	0	0	1	1	1	1	1
3	0	0	0	0	1	0	1	1	1	1
4	0	0	0	0	1	1	1	1	1	1
5	0	0	0	1	0	0	0	1	0	1
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
63	0	1	1	1	1	1	0	1	1	0
64	1	1	1	1	1	1	0	0	1	0

KB true

$$KB \models \neg W_{21} \wedge \neg W_{22} \wedge \neg W_{13} \wedge \neg P_{21}$$

Inference by enumerating models

- Can be implemented as a depth-first search on a constraint graph
  - with backtracking
- Time complexity ~ $O(2^n)$
  where n is the number of relevant symbols
- Space complexity ~ $O(n)$

Not very efficient...

Although computers are good with long sequences of 0s and 1s

Some more definitions

Equivalence:

$A \equiv B$ iff $A \vDash B$ and $B \vDash A$

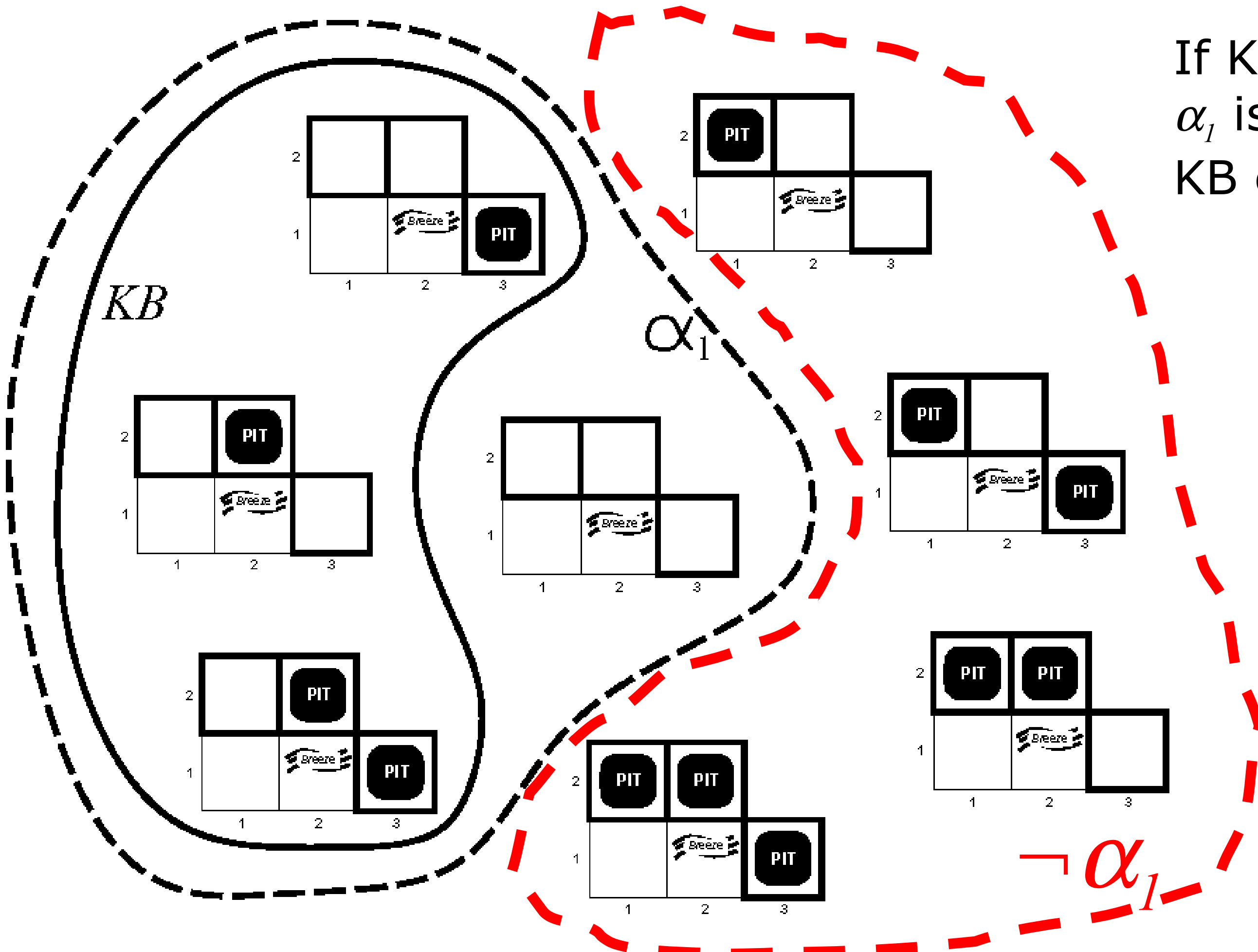
Validity: A valid sentence is true in all models (a tautology)

$A \vDash B$ iff $(A \Rightarrow B)$ is valid

Satisfiability: A sentence is satisfiable if it is true in some model

$A \vDash B$ iff $(A \wedge \neg B)$ is unsatisfiable

Let's explore *satisfiability* first...



If KB is true, then α_1 is also true.
KB entails α_1.

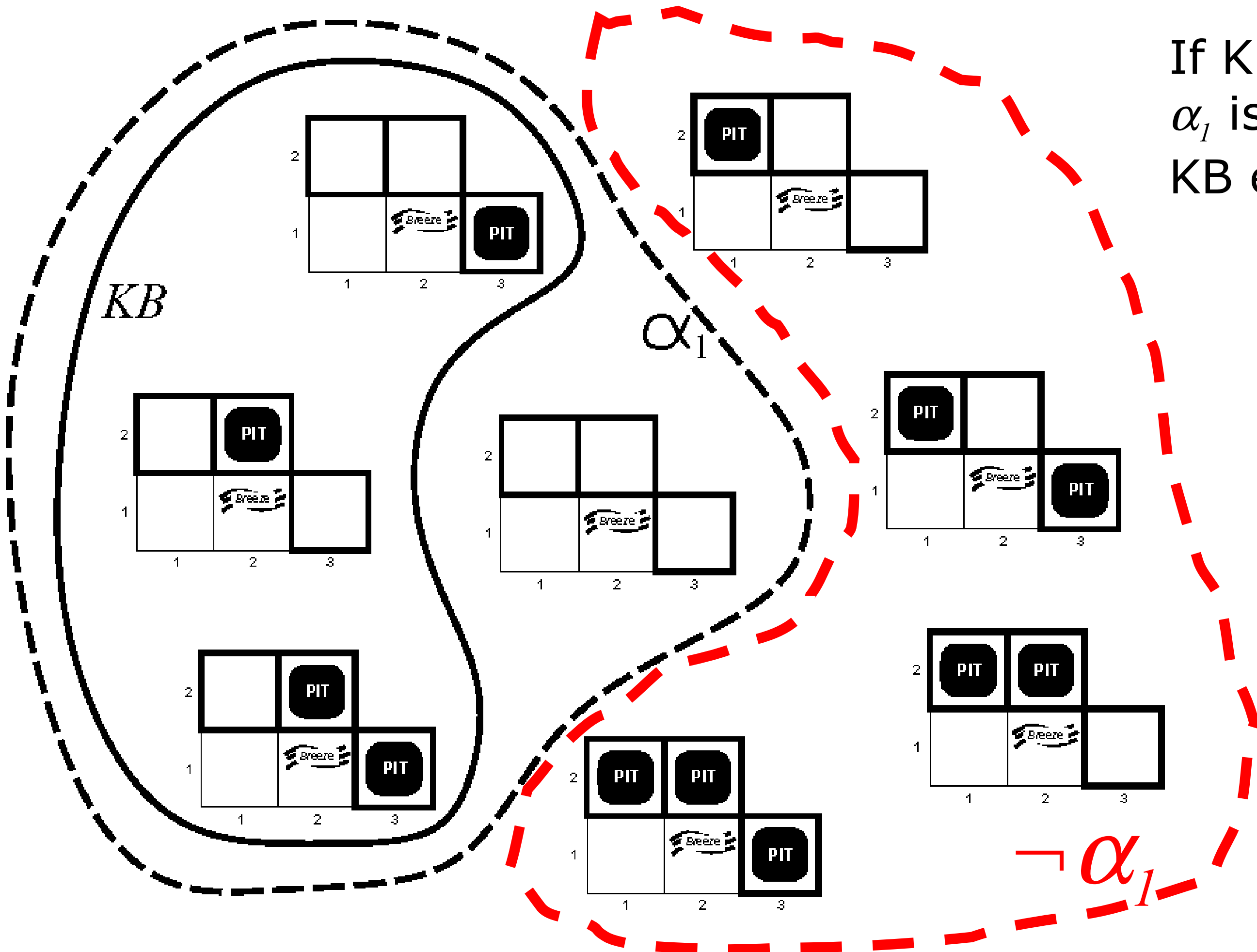
$$KB \models \alpha_1$$

$$KB \subseteq \alpha_1$$

$KB \wedge \neg\alpha_1$ is never true

KB = The set of models that agrees with the knowledge base (the observed facts) [The KB is true in these models]

α_1 = The set of models that agrees with conclusion α_1 [conclusion α_1 is true in these models]



If KB is true, then α_1 is also true.
KB entails α_1.

$$KB \models \alpha_1$$

$$KB \subseteq \alpha_1$$

KB $\wedge$ $\neg\alpha_1$ is unsatisfiable

KB = The set of models that agrees with the knowledge base (the observed facts) [The KB is true in these models]

α_1 = The set of models that agrees with conclusion α_1 [conclusion α_1 is true in these models]

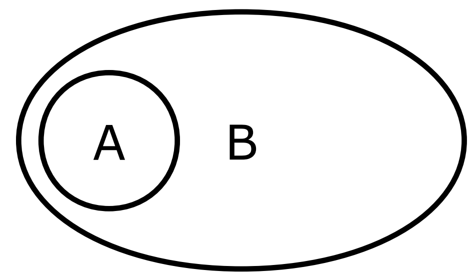
Some more definitions

Equivalence:

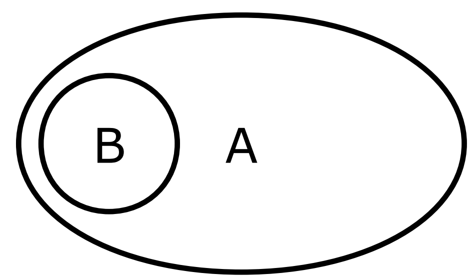
$A \equiv B$ iff $A \models B$ and $B \models A$

For example, $A \equiv \neg(\neg A)$

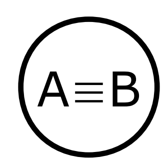
$A \models B$ means that the set of models where A is true is a subset of the models where B is true: $A \subseteq B$



$B \models A$ means that the set of models where B is true is a subset of the models where A is true: $B \subseteq A$



Therefore, the set of models where A is true must be equal to the set of models where B is true: $A \equiv B$

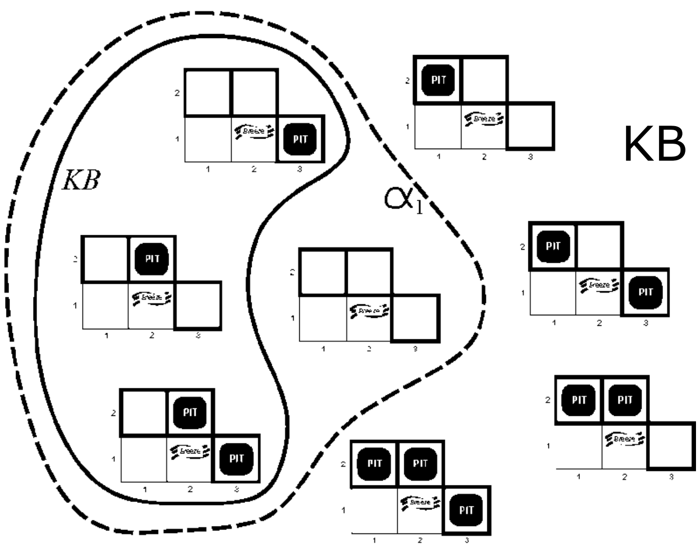


Some more definitions

Validity: A valid sentence is true in all models (a tautology)

For example, $A \vee \neg A$ is valid

$A \models B$ iff $(A \Rightarrow B)$ is valid



$KB \models \alpha_1$

A	B	A⇒B
False	False	True
False	True	True
True	False	False
True	True	True

Logical equivalences

$(A \wedge B) \equiv$	$(B \wedge A)$	$\wedge$ is commutative
$(A \vee B) \equiv$	$(B \vee A)$	$\vee$ is commutative
$((A \wedge B) \wedge C) \equiv$	$(A \wedge (B \wedge C))$	$\wedge$ is associative
$((A \vee B) \vee C) \equiv$	$(A \vee (B \vee C))$	$\vee$ is associative
$\neg(\neg A) \equiv$	A	Double-negation elimination
$(A \Rightarrow B) \equiv$	$(\neg B \Rightarrow \neg A)$	Contraposition
$(A \Rightarrow B) \equiv$	$(\neg A \vee B)$	Implication elimination
$(A \Leftrightarrow B) \equiv$	$((A \Rightarrow B) \wedge (B \Rightarrow A))$	Biconditional elimination
$\neg(A \wedge B) \equiv$	$(\neg A \vee \neg B)$	”De Morgan”
$\neg(A \vee B) \equiv$	$(\neg A \wedge \neg B)$	”De Morgan”
$(A \wedge (B \vee C)) \equiv$	$((A \wedge B) \vee (A \wedge C))$	Distributivity of $\wedge$ over $\vee$
$(A \vee (B \wedge C)) \equiv$	$((A \vee B) \wedge (A \vee C))$	Distributivity of $\vee$ over $\wedge$

Example: DeMorgan

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

A	B	$A \wedge B$	$\neg(A \wedge B)$	$\neg A$	$\neg B$	$\neg A \vee \neg B$
False	False					
False	True					
True	False					
True	True					

Example: DeMorgan

$$\neg(A \land B) \equiv (\neg A \lor \neg B)$$

A	B	$A \land B$	$\neg(A \land B)$	$\neg A$	$\neg B$	$\neg A \lor \neg B$
False	False	False				
False	True	False				
True	False	False				
True	True	True				

Example: DeMorgan

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

A	B	$A \wedge B$	$\neg(A \wedge B)$	$\neg A$	$\neg B$	$\neg A \vee \neg B$
False	False	False	True			
False	True	False	True			
True	False	False	True			
True	True	True	False			

Example: DeMorgan

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

A	B	$A \wedge B$	$\neg(A \wedge B)$	$\neg A$	$\neg B$	$\neg A \vee \neg B$
False	False	False	True	True	True	
False	True	False	True	True	False	
True	False	False	True	False	True	
True	True	True	False	False	False	

Example: DeMorgan

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

A	B	A ∧ B	¬(A ∧ B)	¬A	¬B	¬A ∨ ¬B
False	False	False	True	True	True	True
False	True	False	True	True	False	True
True	False	False	True	False	True	True
True	True	True	False	False	False	False

Logical equivalences

$(A \wedge B) \equiv$	$(B \wedge A)$	$\wedge$ is commutative
$(A \vee B) \equiv$	$(B \vee A)$	$\vee$ is commutative
$((A \wedge B) \wedge C) \equiv$	$(A \wedge (B \wedge C))$	$\wedge$ is associative
$((A \vee B) \vee C) \equiv$	$(A \vee (B \vee C))$	$\vee$ is associative
$\neg(\neg A) \equiv$	A	Double-negation elimination
$(A \Rightarrow B) \equiv$	$(\neg B \Rightarrow \neg A)$	Contraposition
$(A \Rightarrow B) \equiv$	$(\neg A \vee B)$	Implication elimination
$(A \Leftrightarrow B) \equiv$	$((A \Rightarrow B) \wedge (B \Rightarrow A))$	Biconditional elimination
$\neg(A \wedge B) \equiv$	$(\neg A \vee \neg B)$	”De Morgan”
$\neg(A \vee B) \equiv$	$(\neg A \wedge \neg B)$	”De Morgan”
$(A \wedge (B \vee C)) \equiv$	$((A \wedge B) \vee (A \wedge C))$	Distributivity of $\wedge$ over $\vee$
$(A \vee (B \wedge C)) \equiv$	$((A \vee B) \wedge (A \vee C))$	Distributivity of $\vee$ over $\wedge$

Work out some of these on paper for yourself, before we move on...

Inference

- There are two main approaches towards inference:
  - model enumeration
  - inference rules

Slide adapted from D. Byron

Inference rules

- Inference rules are written as

$$\frac{\text{Antecedent}}{\text{Consequent}}$$

If the KB contains the antecedent, you can add the consequent (the KB entails the consequent)

Slide adapted from D. Byron

Inference rules

- Inference rules are written as

$$\frac{\text{Antecedent}}{\text{Consequent}} \qquad \frac{\text{"Before"}}{\text{"After"}}$$

If the KB contains the antecedent, you can add the consequent (the KB entails the consequent)

Commonly used inference rules

Modus Ponens

$$\frac{A \Rightarrow B, A}{B}$$

Modus Tolens

$$\frac{A \Rightarrow B, \neg B}{\neg A}$$

Unit Resolution

$$\frac{A \vee B, \neg B}{A}$$

And Elimination

$$\frac{A \wedge B}{A}$$

Or introduction

$$\frac{A}{A \vee B}$$

And introduction

$$\frac{A, B}{A \wedge B}$$

Slide adapted from D. Byron

Proof for Modus Ponens

$$\frac{A \Rightarrow B, A}{B}$$

	A	B	A ⇒ B
1	False	False	True
2	False	True	True
3	True	False	False
4	True	True	True

Proof for Modus Ponens

$$\frac{A \Rightarrow B, A}{B}$$

	A	B	A ⇒ B
1	False	False	True
2	False	True	True
3	True	False	False
4	True	True	True

These are the cases when A is True

Proof for Modus Ponens

$$\frac{A \Rightarrow B, A}{B}$$

	A	B	A ⇒ B
1	False	False	True
2	False	True	True
3	True	False	False
4	True	True	True

These are the cases when A ⇒ B is True

Proof for Modus Ponens

$$\frac{A \Rightarrow B, A}{B}$$

	A	B	$A \Rightarrow B$
1	False	False	True
2	False	True	True
3	True	False	False
4	True	True	True

This is the case when both A and $A \Rightarrow B$ is True

B is also True here so we can safely add B = True to our KB

Proof for Unit Resolution

$$\frac{A \vee B, \neg B}{A}$$

	A	B	A ∨ B	¬A	¬B
1	False	False	False	True	True
2	False	True	True	True	False
3	True	False	True	False	True
4	True	True	True	False	False

Proof for Unit Resolution

$$\frac{A \vee B, \neg B}{A}$$

	A	B	A ∨ B	¬A	¬B
1	False	False	False	True	True
2	False	True	True	True	False
3	True	False	True	False	True
4	True	True	True	False	False

These are the cases when A ∨ B is True

Proof for Unit Resolution

$$\frac{A \vee B, \neg B}{A}$$

	A	B	A ∨ B	¬A	¬B
1	False	False	False	True	True
2	False	True	True	True	False
3	True	False	True	False	True
4	True	True	True	False	False

These are the cases when ¬B is True

Proof for Unit Resolution

$$\frac{A \vee B, \neg B}{A}$$

	A	B	A ∨ B	¬A	¬B
1	False	False	False	True	True
2	False	True	True	True	False
3	True	False	True	False	True
4	True	True	True	False	False

This is the case when both ¬B and A ∨ B are True

A is also True here so we can safely add A = True to our KB

Commonly used inference rules

Modus Ponens

$$\frac{A \Rightarrow B, A}{B}$$

Modus Tolens

$$\frac{A \Rightarrow B, \neg B}{\neg A}$$

Unit Resolution

$$\frac{A \vee B, \neg B}{A}$$

And Elimination

$$\frac{A \wedge B}{A}$$

Or introduction

$$\frac{A}{A \vee B}$$

And introduction

$$\frac{A, B}{A \wedge B}$$

Slide adapted from D. Byron

Work out these on paper for yourself too

Example: Proof in Wumpus KB

Knowledge base

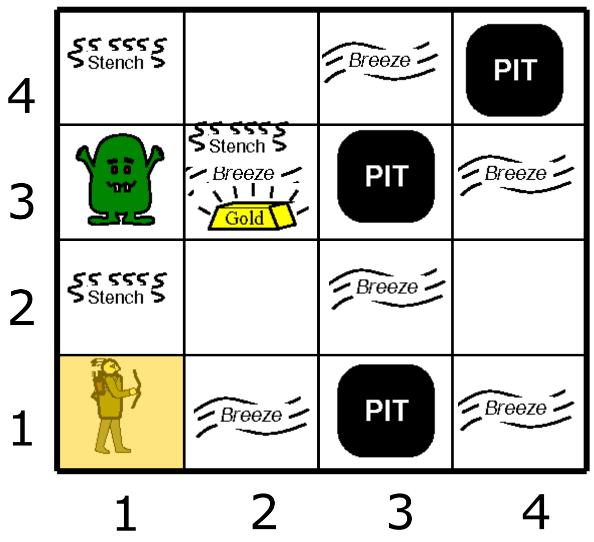
R_1: $\neg P_{11}$

R_2: $\neg B_{11}$

R_3: $\neg W_{11}$

R_4: $\neg S_{11}$

R_5: $\neg G_{11}$



1. Nothing in (1,1)

Proof in Wumpus KB

$B_{11} \Leftrightarrow (P_{12} \vee P_{21})$	Rule of the game
$B_{11} \Rightarrow (P_{12} \vee P_{21}) \wedge (P_{12} \vee P_{21}) \Rightarrow B_{11}$	Biconditional elimination

$$(A \Leftrightarrow B) \equiv ((A \Rightarrow B) \wedge (B \Rightarrow A))$$

Proof in Wumpus KB

$B_{11} \Rightarrow (P_{12} \vee P_{21}) \wedge (P_{12} \vee P_{21}) \Rightarrow B_{11}$	Biconditional elimination
$(P_{12} \vee P_{21}) \Rightarrow B_{11}$	And elimination

$$\frac{A \wedge B}{B}$$

Proof in Wumpus KB

$(P_{12} \lor P_{21}) \Rightarrow B_{11}$	And elimination
$\neg B_{11} \Rightarrow \neg(P_{12} \lor P_{21})$	Contraposition

$$(A \Rightarrow B) \equiv (\neg B \Rightarrow \neg A)$$

Proof in Wumpus KB

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

$\neg B_{11} \Rightarrow \neg(P_{12} \vee P_{21})$	Contraposition
$\neg B_{11} \Rightarrow \neg P_{12} \wedge \neg P_{21}$	”De Morgan”

Proof in Wumpus KB

$B_{11} \Leftrightarrow (P_{12} \vee P_{21})$	Rule of the game
$B_{11} \Rightarrow (P_{12} \vee P_{21}) \wedge (P_{12} \vee P_{21}) \Rightarrow B_{11}$	Biconditional elimination
$(P_{12} \vee P_{21}) \Rightarrow B_{11}$	And elimination
$\neg B_{11} \Rightarrow \neg(P_{12} \vee P_{21})$	Contraposition
$\neg B_{11} \Rightarrow \neg P_{12} \wedge \neg P_{21}$	”De Morgan”

Proof in Wumpus KB

$B_{11} \Leftrightarrow (P_{12} \vee P_{21})$	Rule of the game
$B_{11} \Rightarrow (P_{12} \vee P_{21}) \wedge (P_{12} \vee P_{21}) \Rightarrow B_{11}$	Biconditional elimination
$(P_{12} \vee P_{21}) \Rightarrow B_{11}$	And elimination
$\neg B_{11} \Rightarrow \neg(P_{12} \vee P_{21})$	Contraposition
$\neg B_{11} \Rightarrow \neg P_{12} \wedge \neg P_{21}$	”De Morgan”

Thus, we have proven, in four steps, that no breeze in (1,1) means there can be no pit in either (1,2) or (2,1)

This symbolic inference can be a lot more efficient than naive enumeration of models – if we can apply rules in a good order!

The Resolution rule

An inference algorithm is guaranteed to be complete if it uses the *resolution rule*

$$\frac{A \vee B, \neg B}{A}$$

Unit resolution

$$\frac{A \vee B, \neg B \vee C}{\boxed{A \vee C}}$$

Full resolution

A clause = a disjunction ($\vee$) of literals
A literal = a positive or a negative symbol

The Resolution rule

An inference algorithm is guaranteed to be complete if it uses the *resolution rule*

$$\frac{A_1 \vee A_2 \vee \cdots \vee A_k \vee B, \neg B}{A_1 \vee A_2 \vee \cdots \vee A_k}$$

$$\frac{A_1 \vee A_2 \vee \cdots \vee A_k \vee B, \neg B \vee C_1 \vee C_2 \vee \cdots \vee C_m}{A_1 \vee A_2 \vee \cdots \vee A_k \vee C_1 \vee C_2 \vee \cdots \vee C_m}$$

Note: The resulting clause should only contain one copy of each literal.

Resolution truth table

A	B	$\neg$B	C	A$\vee$B	$\neg$B$\vee$C	A$\vee$C
1	0	1	1	1	1	1
1	1	0	1	1	1	1
0	1	0	1	1	1	1
0	0	1	1	0	1	1
1	0	1	0	1	1	1
1	1	0	0	1	0	1
0	1	0	0	1	0	0
0	0	1	0	0	1	0

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

Resolution truth table

A	B	¬B	C	A∨B	¬B∨C	A∨C
1	0	1	1	1	1	1
1	1	0	1	1	1	1
0	1	0	1	1	1	1
0	0	1	1	0	1	1
1	0	1	0	1	1	1
1	1	0	0	1	0	1
0	1	0	0	1	0	0
0	0	1	0	0	1	0

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

Proof for the resolution rule

Conjunctive normal form (CNF)

- Every sentence of propositional logic is equivalent to a conjunction of clauses
  - a clause is a finite disjunction of literals
  - a literal is an atomic formula or its negation
- Sentences expressed in this way are in *conjunctive normal form* – CNF
  - there is also DNF (disjunctive normal form), i.e. a disjunction of conjunctive clauses
- A sentence with exactly k literals per clause is said to be in k-CNF

This is good, it means we can get far with the resolution inference rule.

Wumpus CNF example

$B_{11} \Leftrightarrow (P_{12} \vee P_{21})$	Rule of the game
$B_{11} \Rightarrow (P_{12} \vee P_{21}) \wedge (P_{12} \vee P_{21}) \Rightarrow B_{11}$	Biconditional elimination
$(\neg B_{11} \vee (P_{12} \vee P_{21})) \wedge (\neg(P_{12} \vee P_{21}) \vee B_{11})$	Implication elimination
$(\neg B_{11} \vee P_{12} \vee P_{21}) \wedge ((\neg P_{12} \wedge \neg P_{21}) \vee B_{11})$	"De Morgan"
$(\neg B_{11} \vee P_{12} \vee P_{21}) \wedge ((\neg P_{12} \vee B_{11}) \wedge (B_{11} \vee \neg P_{21}))$	Distributivity
$(\neg B_{11} \vee P_{12} \vee P_{21}) \wedge (\neg P_{12} \vee B_{11}) \wedge (B_{11} \vee \neg P_{21})$	Voilá – CNF

$(A \Leftrightarrow B) \equiv ((A \Rightarrow B) \wedge (B \Rightarrow A))$

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

$(A \Rightarrow B) \equiv (\neg A \vee B)$

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

Wumpus CNF example

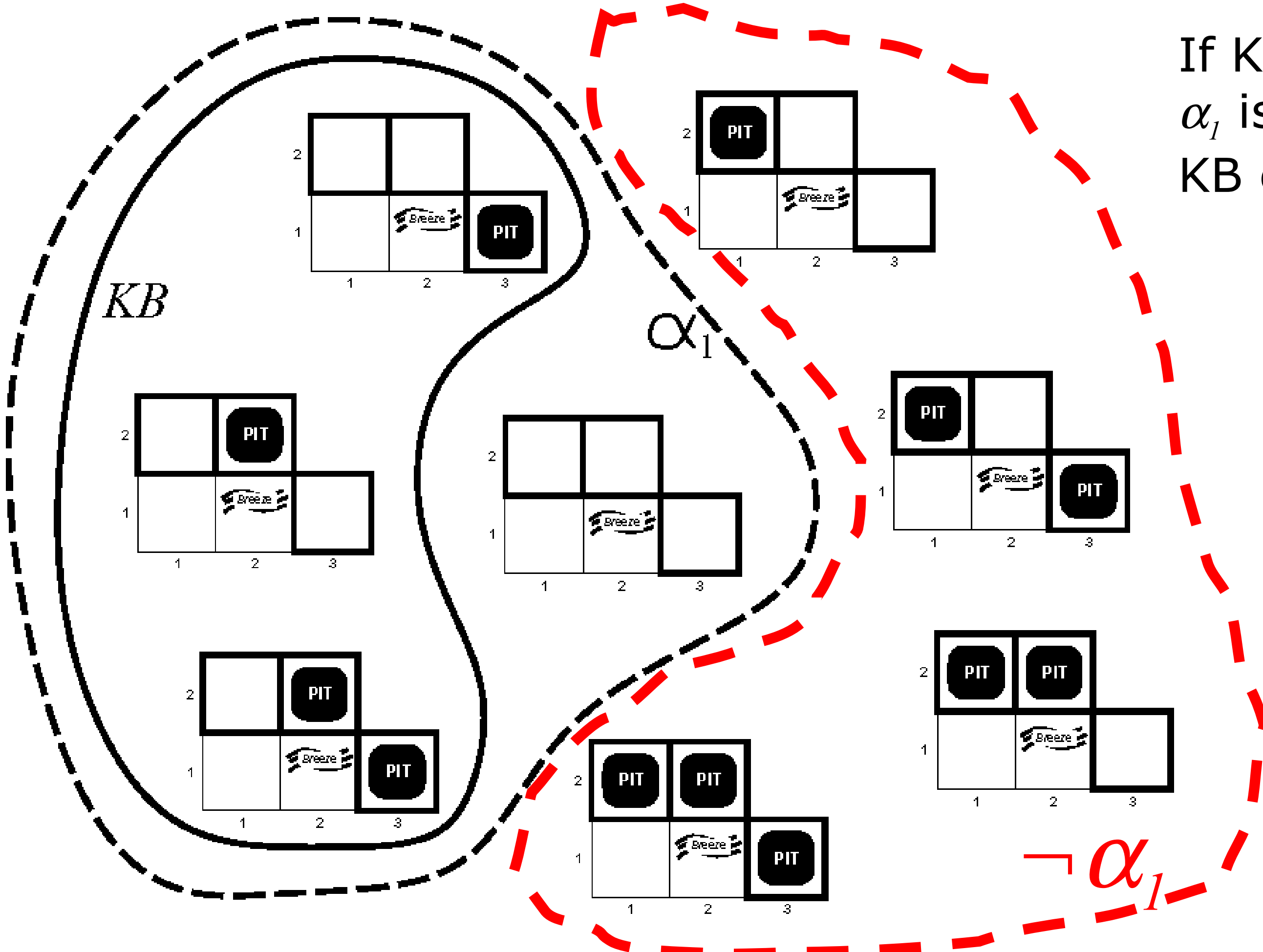
$B_{11} \Leftrightarrow (P_{12} \vee P_{21})$	Rule of the game
$B_{11} \Rightarrow (P_{12} \vee P_{21}) \wedge (P_{12} \vee P_{21}) \Rightarrow B_{11}$	Biconditional elimination
$(\neg B_{11} \vee (P_{12} \vee P_{21})) \wedge (\neg(P_{12} \vee P_{21}) \vee B_{11})$	Implication elimination
$(\neg B_{11} \vee P_{12} \vee P_{21}) \wedge ((\neg P_{12} \wedge \neg P_{21}) \vee B_{11})$	”De Morgan”
$(\neg B_{11} \vee P_{12} \vee P_{21}) \wedge ((\neg P_{12} \vee B_{11}) \wedge (B_{11} \vee \neg P_{21}))$	Distributivity
$(\neg B_{11} \vee P_{12} \vee P_{21}) \wedge (\neg P_{12} \vee B_{11}) \wedge (B_{11} \vee \neg P_{21})$	Voilá – CNF

The **resolution refutation** algorithm

Proves by the principle of contradiction:

Shows that KB $\models \alpha$ by proving that (KB $\wedge \neg\alpha$) is unsatisfiable.

- Convert (KB $\wedge \neg\alpha$) to CNF
- Apply the resolution inference rule repeatedly to the resulting clauses
- Continue until:
  (a) No more clauses can be added, KB $\not\models \alpha$
  (b) The empty clause ($\emptyset$) is produced, KB $\models \alpha$



If KB is true, then α_1 is also true.
KB entails α_1.

$$KB \models \alpha_1$$

$$KB \subseteq \alpha_1$$

KB ∧ $\neg \alpha_1$ never true

KB = The set of models that agrees with the knowledge base (the observed facts) [The KB is true in these models]

α_1 = The set of models that agrees with conclusion α_1 [conclusion α_1 is true in these models]

Wumpus resolution example

$B_{11} \Leftrightarrow (P_{12} \vee P_{21})$	Rule of the game
$(\neg B_{11} \vee P_{12} \vee P_{21}) \wedge (\neg P_{12} \vee B_{11}) \wedge (B_{11} \vee \neg P_{21})$	CNF
$\neg B_{11}$	Observation
$(\neg B_{11} \vee P_{12} \vee P_{21}) \wedge (\neg P_{12} \vee B_{11}) \wedge (B_{11} \vee \neg P_{21}) \wedge \neg B_{11}$	KB in CNF
$\neg P_{21}$	Hypothesis (α)

$$KB \wedge \neg\alpha = (\neg B_{11} \vee P_{12} \vee P_{21}) \wedge (\neg P_{12} \vee B_{11}) \wedge (B_{11} \vee \neg P_{21}) \wedge \neg B_{11} \wedge P_{21}$$

Wumpus resolution example

$$\frac{(B_{11} \lor \neg P_{21}),\ \neg B_{11}}{\boxed{\neg P_{21}}}$$

$$\frac{P_{21},\ \neg P_{21}}{\varnothing}$$

$$\underbrace{KB \land \neg\alpha} = (\neg B_{11}\lor P_{12}\lor P_{21})\land(\neg P_{12}\lor B_{11})\land \boxed{(B_{11}\lor\neg P_{21})\land\neg B_{11}} \land \boxed{P_{21}}$$

Not satisfied, we conclude that KB $\models \alpha$

Completeness of resolution

S = Set of clauses

$RC(S)$ = Resolution closure of S

$RC(S)$ = Set of all clauses that can be derived from S by the resolution inference rule.

$RC(S)$ has finite cardinality (finite number of symbols P_1, P_2, ..., P_k) $\Rightarrow$ Resolution refutation must terminate.

Completeness of resolution

The **ground resolution theorem**

If a set S is unsatisfiable, then $RC(S)$ contains the empty clause $\varnothing$.

Left without proof.

Exercise

Your knowledge base (KB) is this:

$$B$$

$$B \Rightarrow C$$

$$B \wedge C \Rightarrow A$$

Prove, using the resolution refutation algorithm, that A is True

Exercise

Your knowledge base (KB) is this:

KB in CNF

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

Prove, using the resolution refutation algorithm, that A is True

Hypothesis: A is True $\alpha = A$

Exercise

$\text{KB} \wedge \neg\alpha$

$$B$$

$$\neg B \vee C$$

$$\neg\left(B \wedge C\right) \vee A \equiv \neg B \vee \neg C \vee A$$

$$\neg A$$

Exercise

KB $\wedge \neg\alpha$

$$B$$

$$\neg B \vee C$$

$$\neg (B \wedge C) \vee A \equiv \boxed{\neg B \vee \neg C \vee A}$$

$$\boxed{\neg A}$$

$$\frac{\neg B \vee \neg C \vee A, \neg A}{\neg B \vee \neg C}$$

Exercise

KB $\wedge \neg\alpha$

B

$\neg B \vee C$

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

$\neg A$

$\neg B \vee \neg C$

Exercise

$KB \wedge \neg\alpha$

B

$\boxed{\neg B \vee C}$

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

$\neg A$

$\boxed{\neg B \vee \neg C}$

$$\frac{\neg B \vee C, \neg B \vee \neg C}{\neg B}$$

Exercise

KB $\wedge \neg\alpha$

$$B$$

$$\neg B \vee C$$

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

$$\neg A$$

$$\neg B \vee \neg C$$

$$\neg B$$

Exercise

KB $\wedge \neg\alpha$

$$\boxed{B}$$

$$\neg B \vee C$$

$$\frac{B, \neg B}{\emptyset}$$

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

$$\neg A$$

$$\neg B \vee \neg C$$

$$\boxed{\neg B}$$

Exercise

KB $\wedge \neg\alpha$

$\boxed{B}$

$\neg B \vee C$

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

$\neg A$

$\neg B \vee \neg C$

$\boxed{\neg B}$

$$\frac{B, \neg B}{\emptyset}$$

(KB $\wedge \neg\alpha$) is unsatisfiable so α is True.

Exercise

KB $\wedge \neg\alpha$

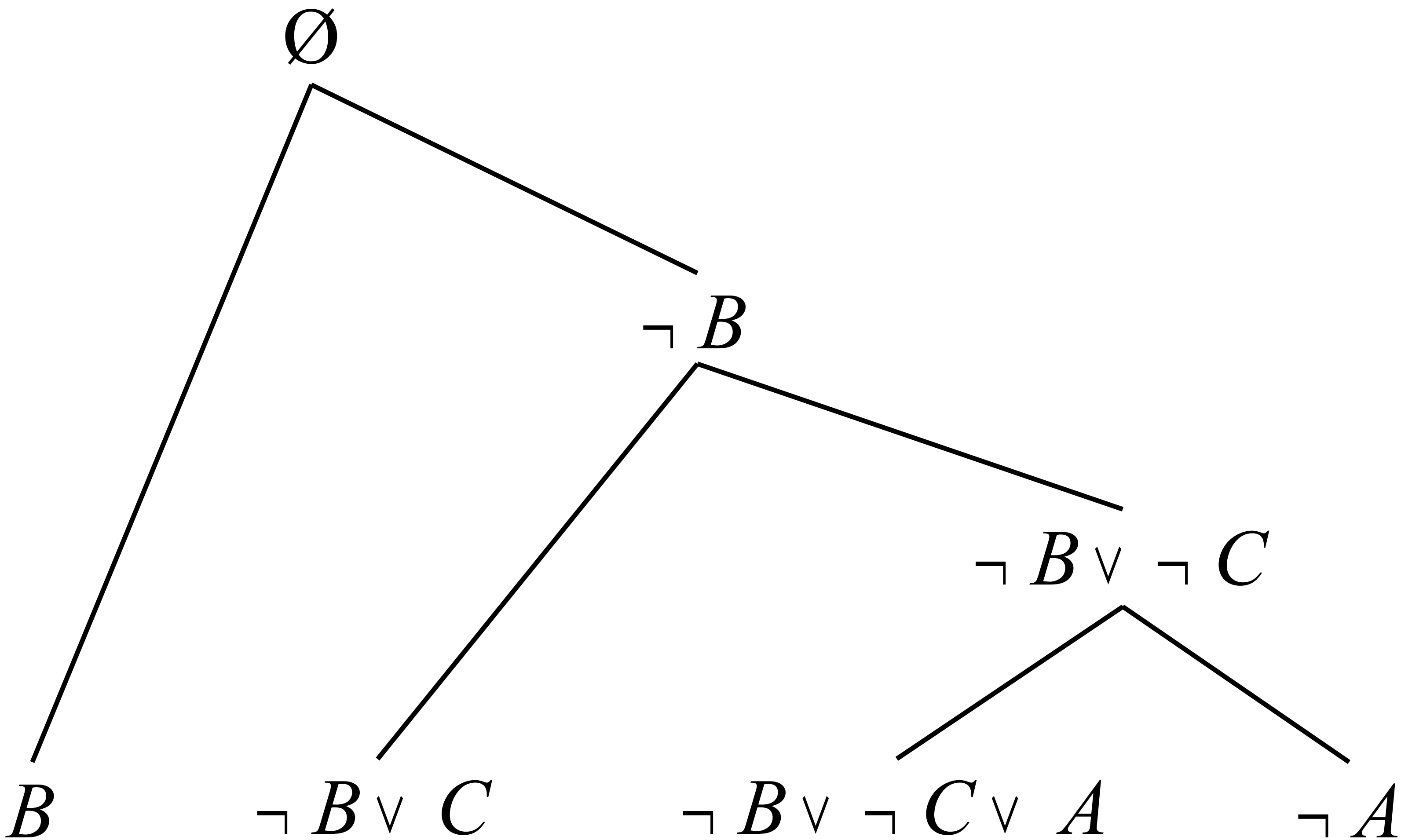
$$B$$

$$\neg B \vee C$$

$$\neg\left(B \wedge C\right) \vee A \equiv \neg B \vee \neg C \vee A$$

$$\neg A$$

We could have illustrated the resolution refutation steps with a graph...


∅
¬ B
¬ B ∨ ¬ C
B
¬ B ∨ C
¬ B ∨ ¬ C ∨ A
¬ A

Problem with resolution refutation

- It may expand all nodes (all statements) – exponential in both space and time
- Is there not a more efficient way to only expand those nodes (statements) that affect our query?

Horn clauses and forward- backward chaining

- Restricted set of clauses: ***Horn clauses***

  disjunction of literals where <u>at most one is positive</u>, e.g.,

  $(\neg A_1 \vee \neg A_2 \vee \cdots \vee \neg A_k \vee B)$ or $(\neg A_1 \vee \neg A_2 \vee \cdots \vee \neg A_k)$

- Why Horn clauses?
  Every Horn clause can be written as an implication, e.g.,

  $(\neg A_1 \vee \neg A_2 \vee \cdots \vee \neg A_k \vee B) \equiv (A_1 \wedge A_2 \wedge \cdots \wedge A_k) \Rightarrow B$
  $(\neg A_1 \vee \neg A_2 \vee \cdots \vee \neg A_k) \equiv (A_1 \wedge A_2 \wedge \cdots \wedge A_k) \Rightarrow \text{False}$

- Inference in Horn clauses can be done using ***forward-backward*** (F-B) chaining in ***linear time***

Slide adapted from V. Pavlovic

Forward or Backward?

Inference can be run forward or backward

Forward-chaining:

- Use the current facts in the KB to trigger all possible inferences

Backward-chaining:

- Work backward from the query proposition Q
- If a rule has Q as a conclusion, see if antecedents can be found to be true

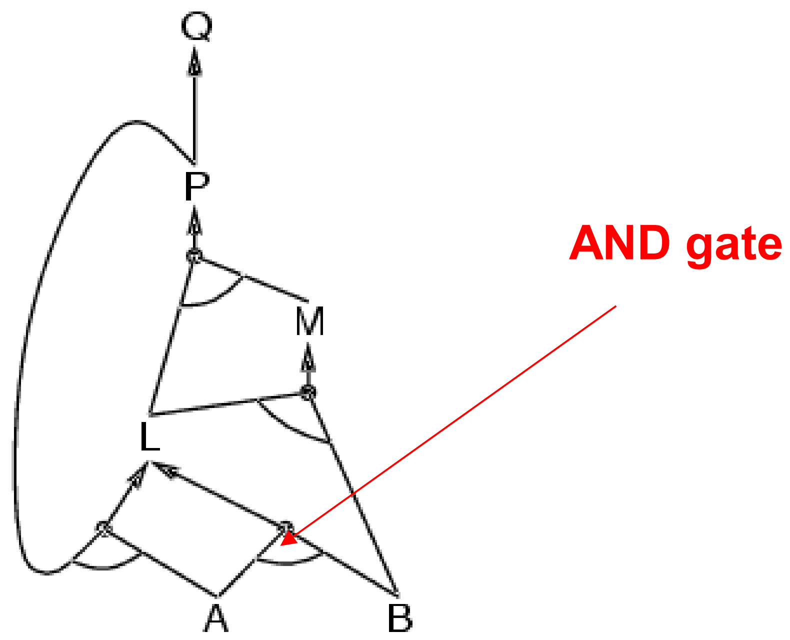
Slide adapted from D. Byron

Example

KB

$$
\begin{aligned}
&P \Rightarrow Q \\
&L \wedge M \Rightarrow P \\
&B \wedge L \Rightarrow M \\
&A \wedge P \Rightarrow L \\
&A \wedge B \Rightarrow L \\
&A \\
&B
\end{aligned}
$$

KB in graph form



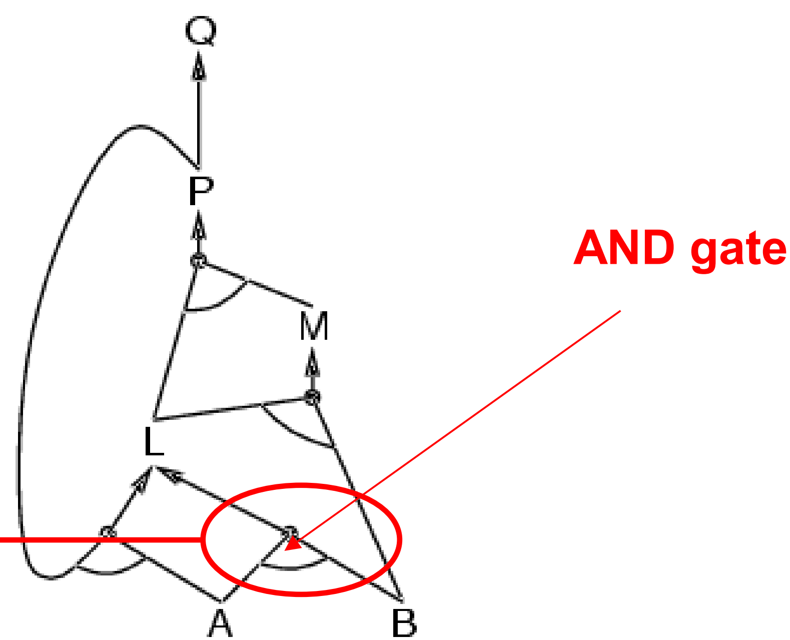
We are going to check if Q is True

Example

KB

$$P \Rightarrow Q$$
$$L \wedge M \Rightarrow P$$
$$B \wedge L \Rightarrow M$$
$$A \wedge P \Rightarrow L$$
$$A \wedge B \Rightarrow L$$
$$A$$
$$B$$

KB in graph form



We are going to check if Q is True

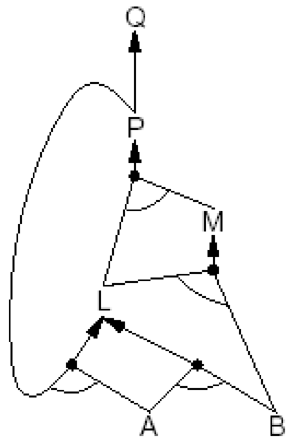
Example of forward chaining

We've proved that Q is true

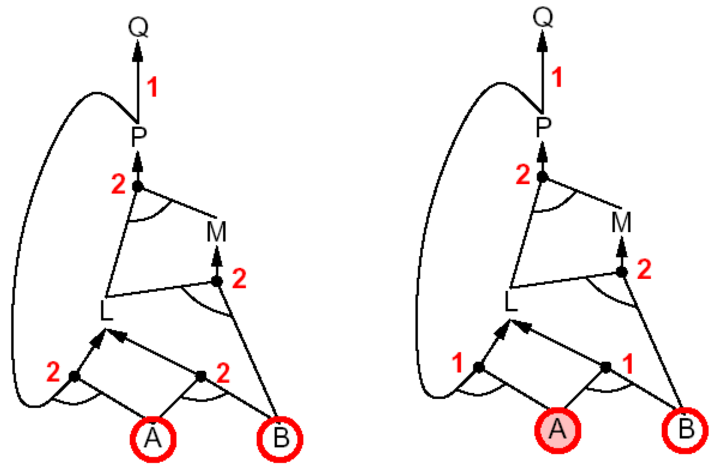
- $P \Rightarrow Q$
- $L \wedge M \Rightarrow P$
- $B \wedge L \Rightarrow M$

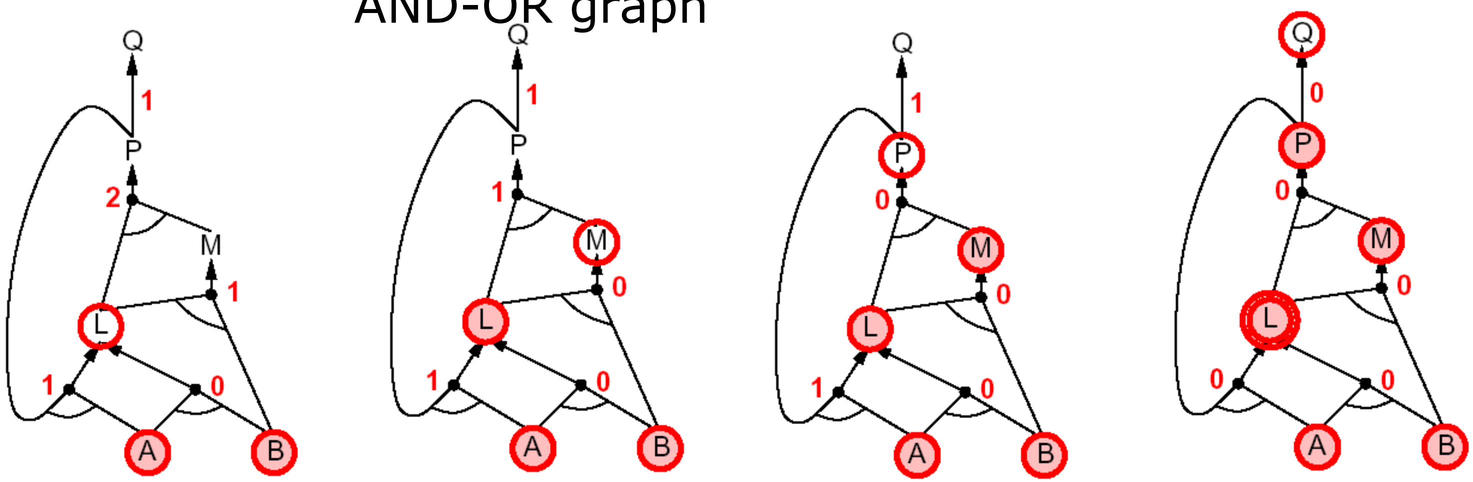
  $A \wedge P \Rightarrow L$
- $A \wedge B \Rightarrow L$
- A
- B

KB



AND-OR graph





Agenda

A
B
L
M
P
Q

Every step is Modus Ponens, e.g. $\dfrac{A \wedge B \Rightarrow L,\; A \wedge B}{L}$

Slide adapted from V. Pavlovic (who borrowed from Lee?)

Example of backward chaining

Query: is Q true

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

$B \wedge L \Rightarrow M$

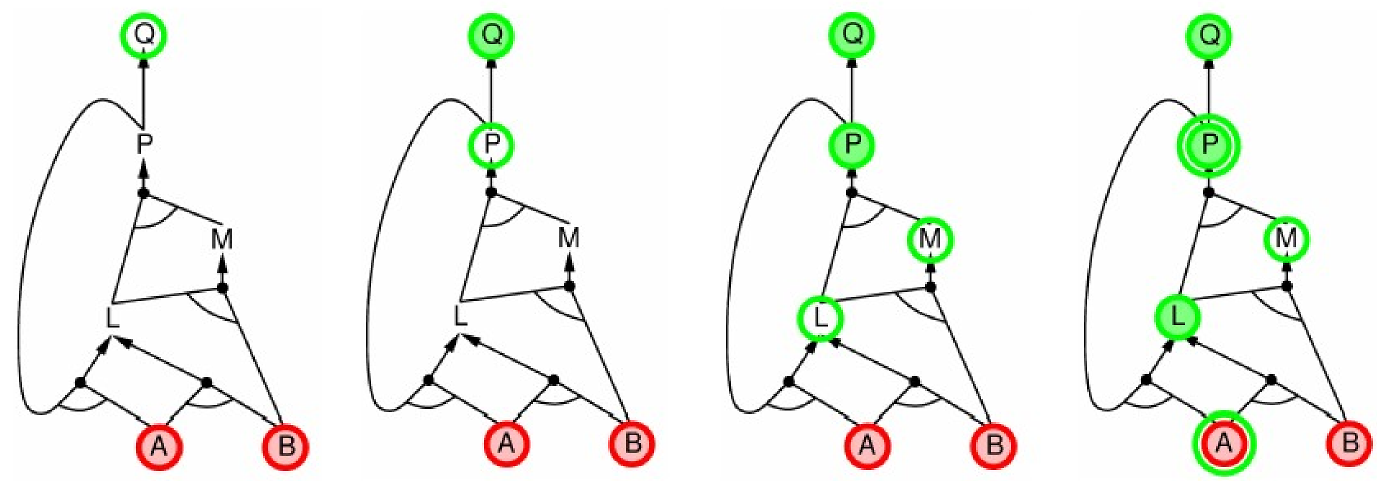
$A \wedge P \Rightarrow L$

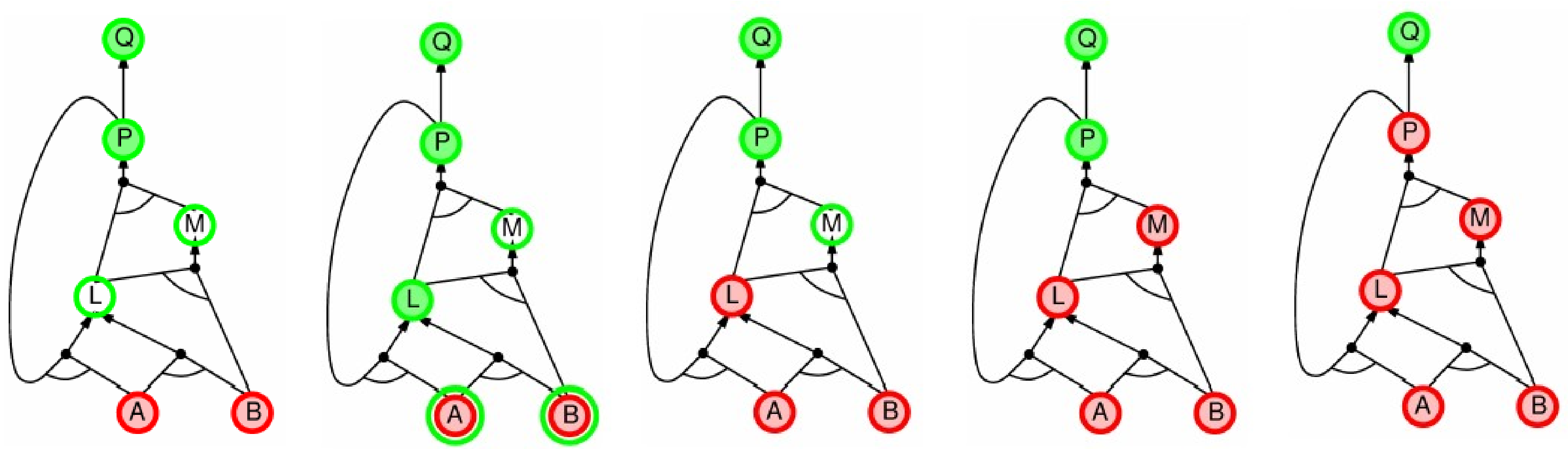
$A \wedge B \Rightarrow L$

A

B

KB



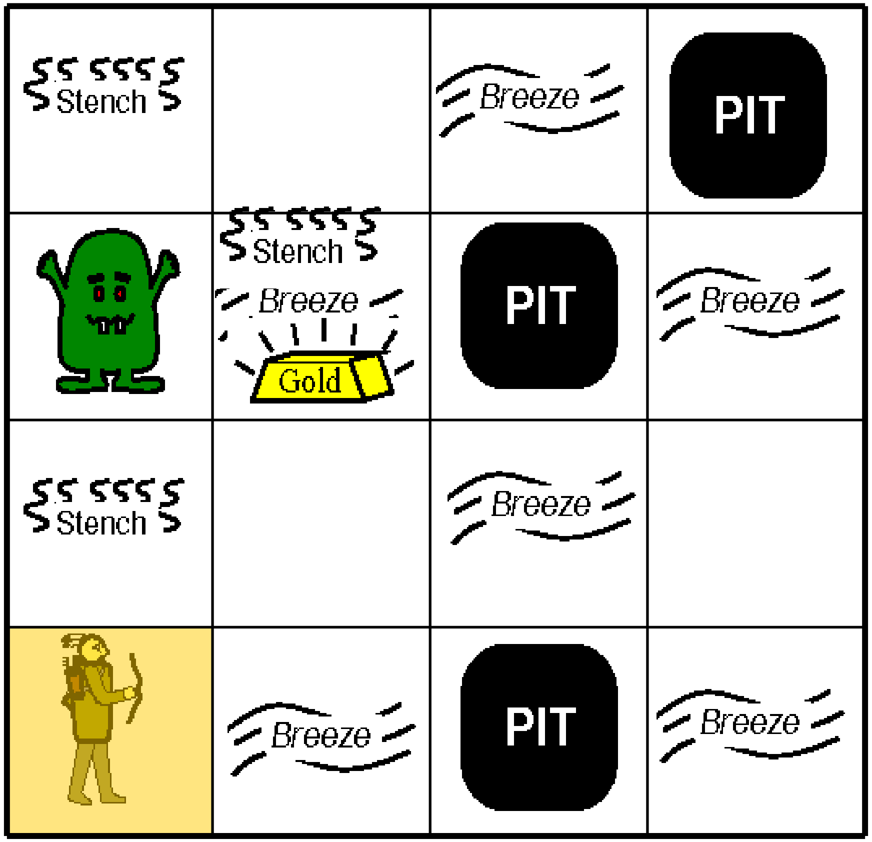


Yes, Q is true

Slide adapted from Lee

Wumpus world revisited

Knowledge base (KB) in initial position (ROG = Rule of the Game)



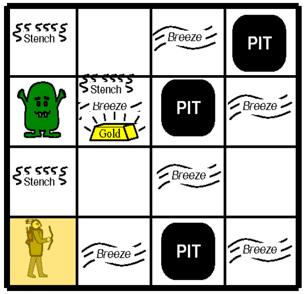
Wumpus world revisited

Knowledge base (KB) in initial position (ROG = Rule of the Game)

1-16	$B_{i,j} \Leftrightarrow (P_{i,j+1} \vee P_{i,j-1} \vee P_{i-1,j} \vee P_{i+1,j})$	ROG: Pits
17-32	$S_{i,j} \Leftrightarrow (W_{i,j+1} \vee W_{i,j-1} \vee W_{i-1,j} \vee W_{i+1,j})$	ROG: Wumpus' odor
33	$(W_{1,1} \vee W_{1,2} \vee W_{1,3} \vee \cdots \vee W_{4,3} \vee W_{4,4})$	ROG: #W ≥ 1
34-153	$\neg(W_{i,j} \wedge W_{k,l})$	ROG: #W ≤ 1
154	$(G_{1,1} \vee G_{1,2} \vee G_{1,3} \vee \cdots \vee G_{4,3} \vee G_{4,4})$	ROG: #G ≥ 1
155-274	$\neg(G_{i,j} \wedge G_{k,l})$	ROG: #G ≤ 1
275	$(\neg B_{11} \wedge \neg W_{11} \wedge \neg G_{11})$	ROG: Start safe

There are 5 "on-states" for every square, {W,P,S,B,G}.
A 4×4 lattice has $16 \times 5 = 80$ distinct symbols.
Enumerating models means going through 2^{80} models!

The physics rules (1-32) are very unsatisfying – no generalization.



Summary

- Knowledge is in the form of sentences in a knowledge representation language.
- The representation language has syntax and semantics.
- Propositional logic consists of
  - proposition symbols
  - logical connectives.
- Inference:
  - Model checking
  - Inference rules (e.g. resolution)
    - Horn clauses