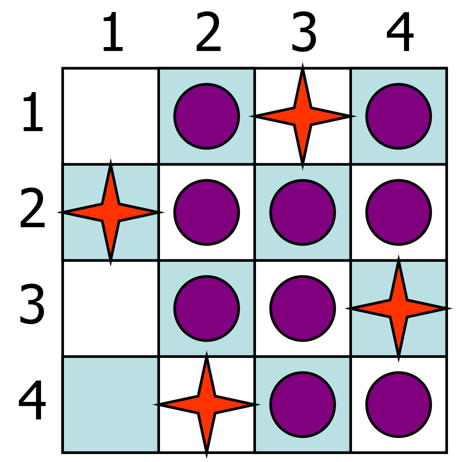
Propositional Logic

CPSC 470 – Artificial Intelligence

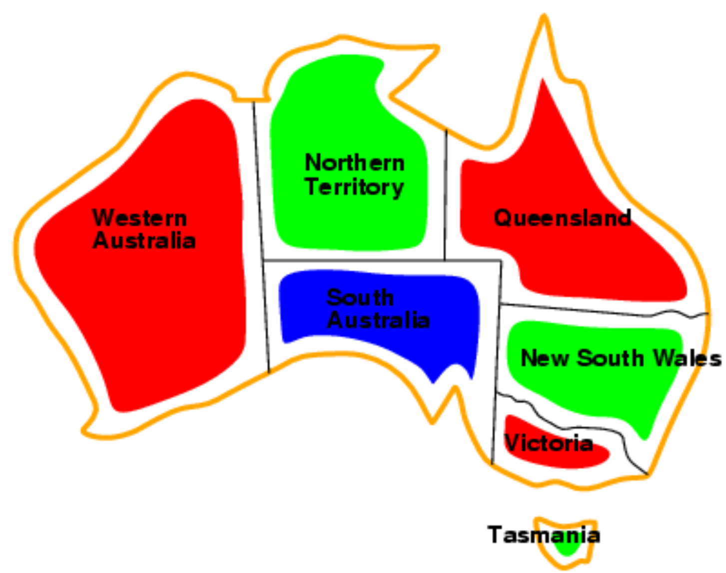
Brian Scassellati

Constraint Satisfaction Problems

```
  T W O
+ T W O
-------
F O U R
```



4						8		5
	3							
			7					
	2						6	
				8		4		
	4			1				
			6		3		7	
5		3	2		1			
1		4						



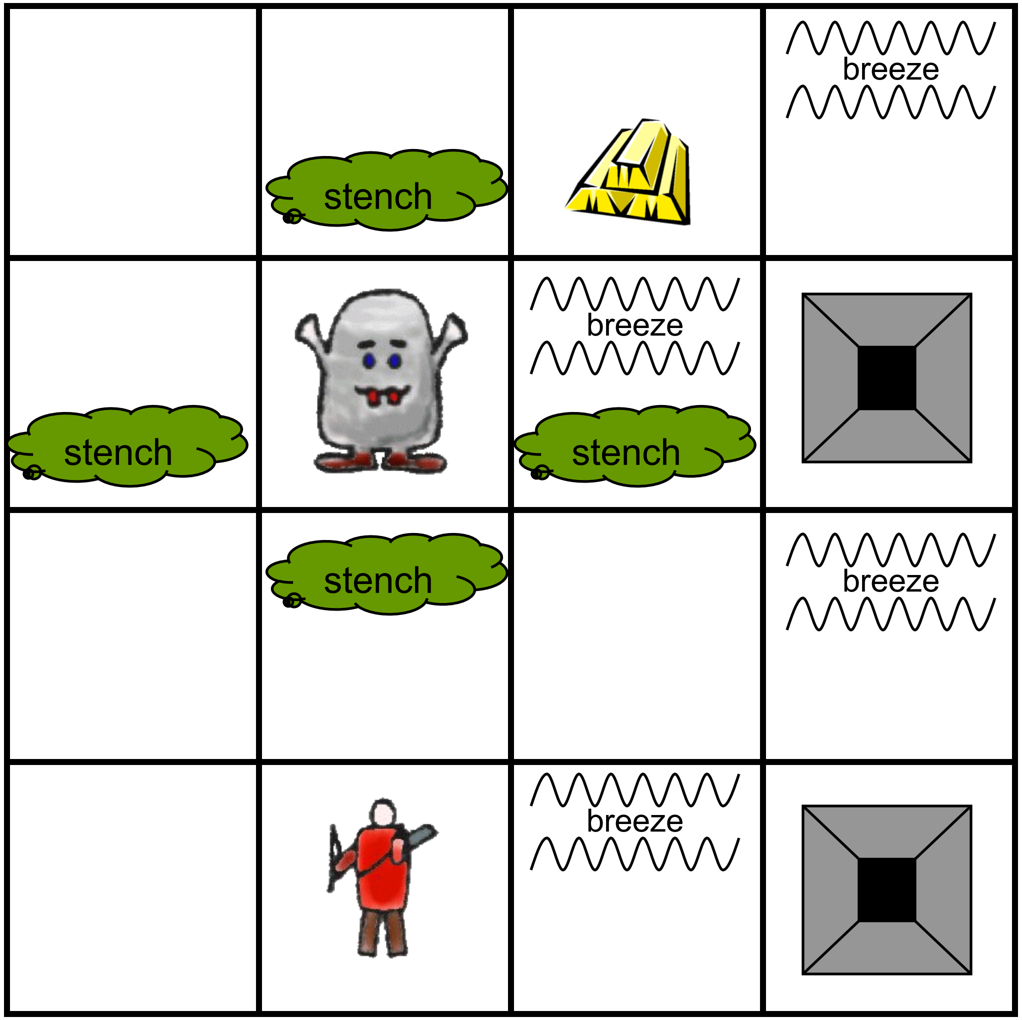
World Characterization

	Search	**CSP**
Fully Observable	Yes	Yes
Deterministic	Yes	Yes
Episodic	No	No
Static	Yes	Yes
Discrete	Yes	Mostly

World Characterization

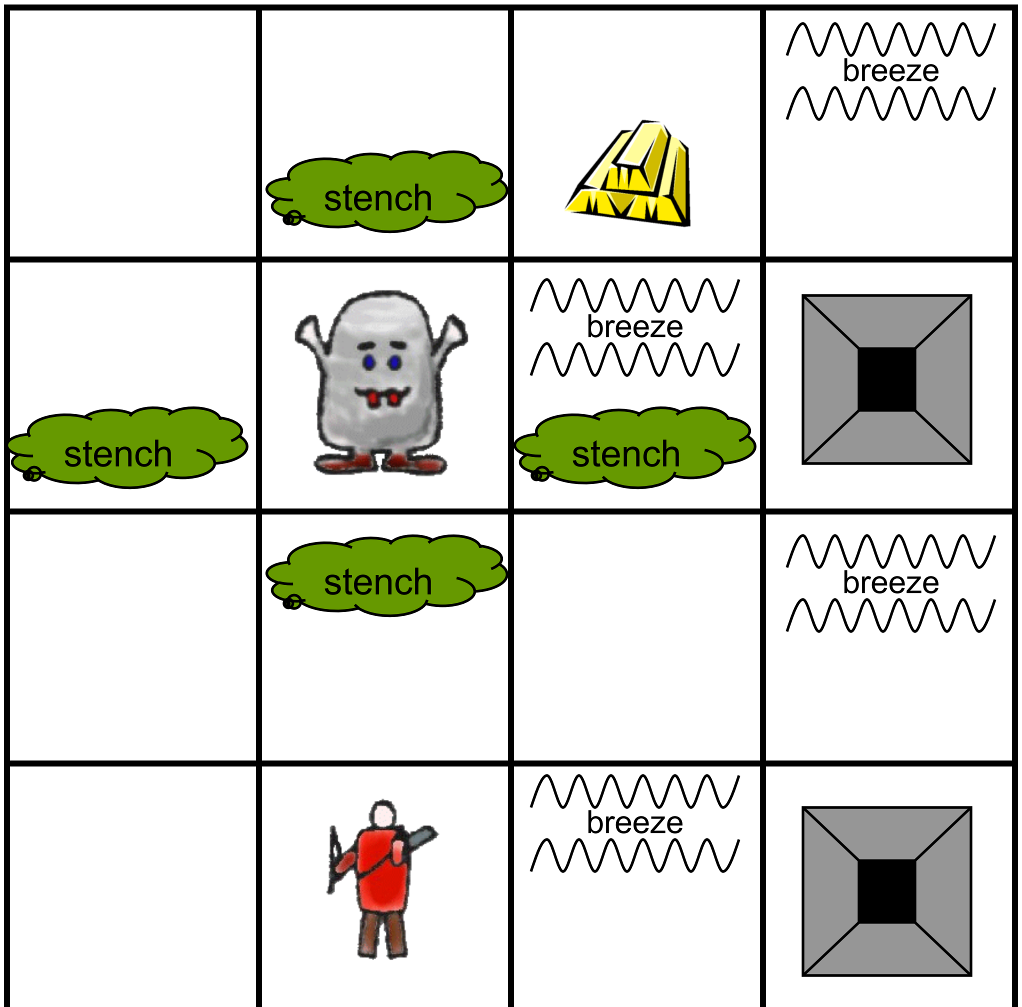
	Search	**CSP**	**Today**
Fully Observable	Yes	Yes	No
Deterministic	Yes	Yes	Yes
Episodic	No	No	No
Static	Yes	Yes	Yes
Discrete	Yes	Mostly	Yes

The Wumpus World



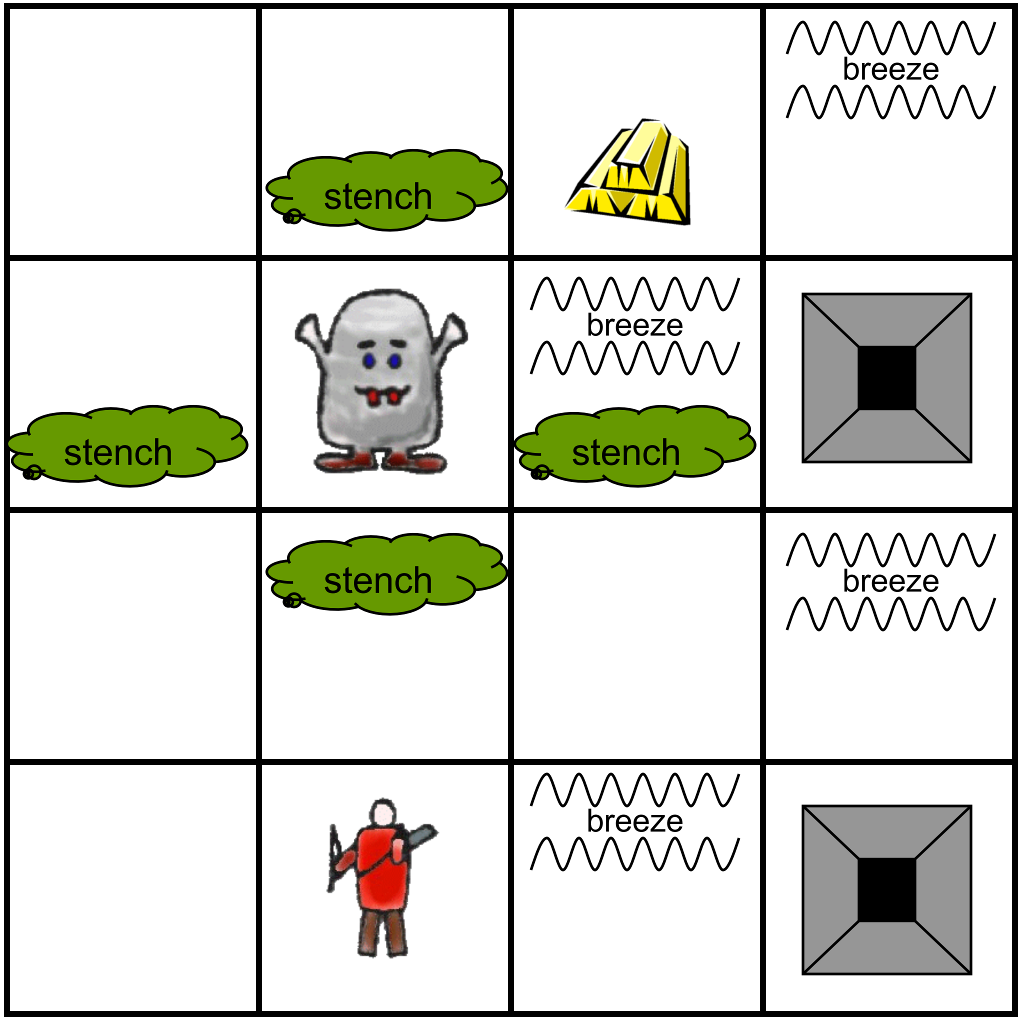
- Grid-like world
- Noble hero
- Horrible wumpus
- Bottomless pits
- Gold
- Breeze
- Stench

Actions in the Wumpus World



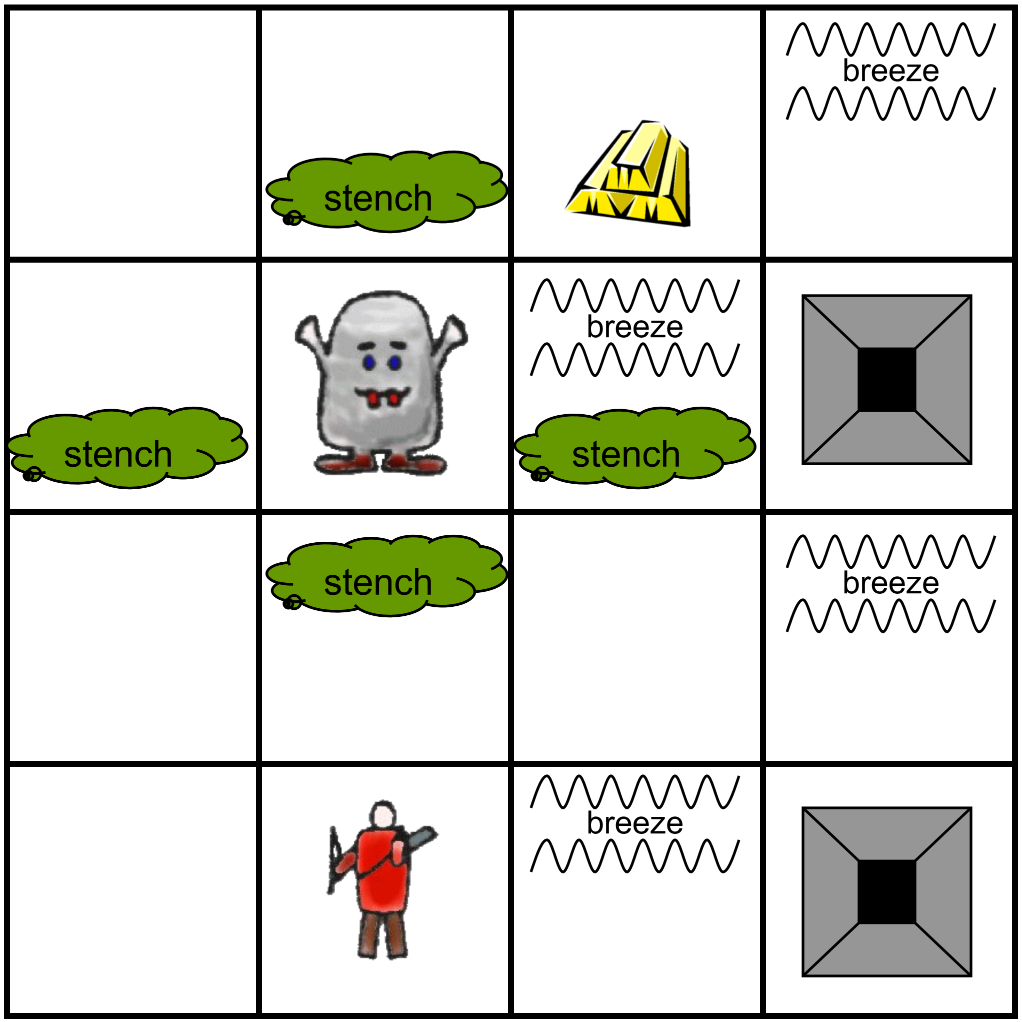
- Goals:
  - find the gold
  - kill the wumpus
  - go home
- Actions
  - Move N,S,E,W
  - Grab
  - Shoot(N,S,E,W)
    - Only one arrow!

The Wumpus World



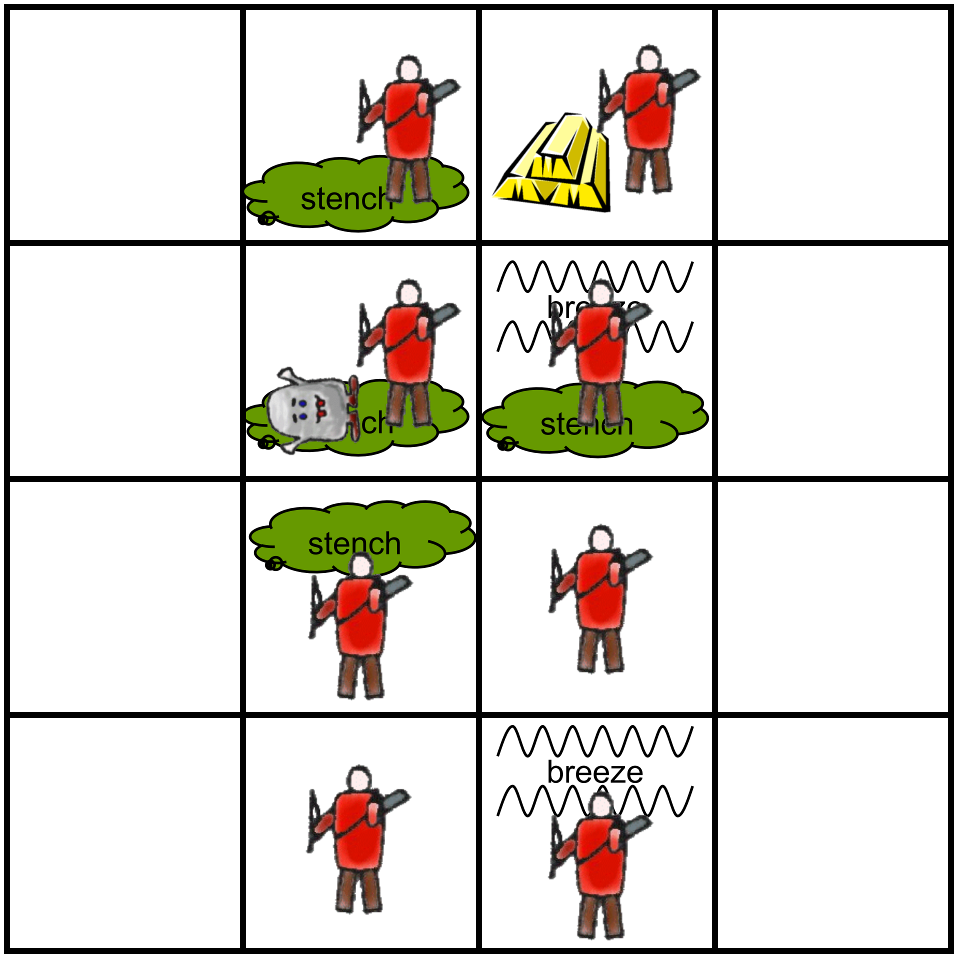
- If we had complete knowledge of the world, then we could simply build a search tree
- What if our perceptions are limited?

Incomplete Knowledge of the World



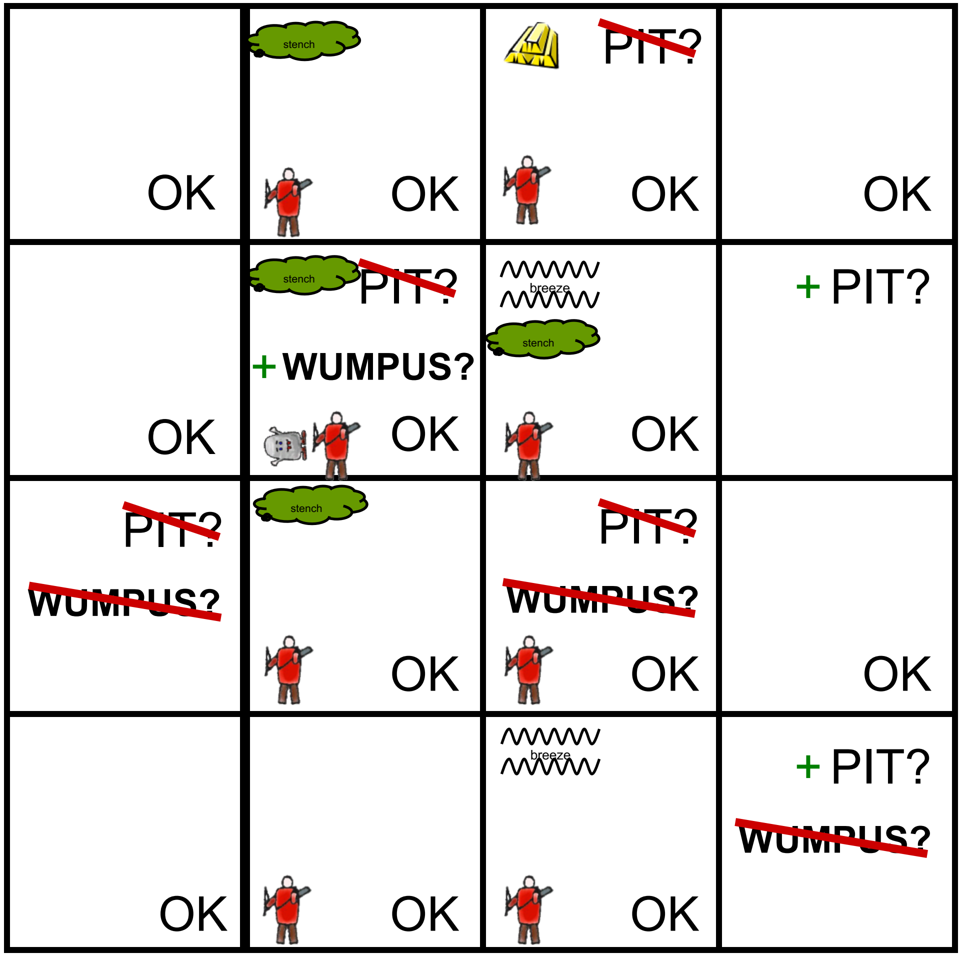
- Agent's percepts:
  - Stench
  - Breeze
  - Glitter
  - Bump
  - Scream
- Other than the agent, the world is static

Our First Wumpus Hunt



stench	breeze	glitter	bump	scream	
No	No	No	No	No	**South**
No	No	No	Yes	No	**East**
No	Yes	No	No	No	**West**
No	No	No	No	No	**North**
Yes	No	No	No	No	**East**
No	No	No	No	No	**North**
Yes	Yes	No	No	No	**Shoot(W)**
Yes	Yes	No	No	Yes	**West**
Yes	No	No	No	No	**North**
Yes	No	No	No	No	**East**
No	No	Yes	No	No	**Grab**

Annotated Wumpus Hunt



stench	breeze	glitter	bump	scream	
No	No	No	No	No	**South**
No	No	No	Yes	No	**East**
No	Yes	No	No	No	**West**
No	No	No	No	No	**North**
Yes	No	No	No	No	**East**
No	No	No	No	No	**North**
Yes	Yes	No	No	No	**Shoot(W)**
Yes	Yes	No	No	Yes	**West**
Yes	No	No	No	No	**North**
Yes	No	No	No	No	**East**
No	No	Yes	No	No	**Grab**

Today we will see how to build an agent that can perform this reasoning

Representing Beliefs

- In most programming languages, it is easy to specify statements like this…
  - *There is a pit in square [3,1]*
- But it is difficult to specify statements like these…
  - *There is a pit in either square [3,1] or [2,2]*
  - *There is no wumpus in square [2,2]*
  - *Because there was no breeze in square [1,2], there is a pit in square [3,1]*
- Require an agent that can represent this knowledge and perform the reasoning to infer new conclusions

Components of a Logic

- A formal system for representing the state of affairs
  - A sentence is a representation of a fact about the world
  - A syntax that describes how to make sentences
  - A semantics that gives constraints on how sentences relate to the state of affairs
  - A proof theory – a set of rules for deducing the entailments of a set of sentences

Entailment means that one thing follows from another

Properties of Logical Inference

- Inference is complete if it can find a proof for any sentence that is entailed
- A sentence is valid or necessarily true if and only if it is true under all possible interpretations in all possible worlds

  There is a stench in [1,1] or there is not a stench in [1,1]
- A sentence is satisfiable if and only if there is some interpretation in some world for which it is true

  There is a wumpus at [1,1]

Types of Commitment

Language	Ontological Commitment (What exists in the world)	Epistemological Commitment (What an agent believes about facts)
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 $0 \ldots 1$

- We make assumptions about
  - the world (ontological commitments)
  - the beliefs that an agent can hold (epistemological commitments)

Propositional Logic Syntax

- Basic Units (sentences)
  - *True* and *False*
  - Propositions *P, Q, …*
- Connectives

$P \wedge Q$ and (conjunction)

Returns true if both P and Q are true

$P \vee Q$ or (disjunction)

Returns true if either P or Q is true

$P \Rightarrow Q$ implication

If P is true then Q is also true

$P \Leftrightarrow Q$ equivalence

P is true exactly when Q is true

$\neg P$ negation

Returns true when P is false

Propositional Logic Grammar

- BNF (Backus-Naur form) for PL Grammar:

$$
\begin{aligned}
Sentence &\rightarrow AtomicSentence \mid ComplexSentence \\
AtomicSentence &\rightarrow True \mid False \mid P \mid Q \mid \ldots \\
ComplexSentence &\rightarrow (Sentence) \mid \\
&\qquad Sentence\ Connective\ Sentence \mid \\
&\qquad \neg Sentence \\
Connective &\rightarrow \wedge \mid \vee \mid \Rightarrow \mid \Leftrightarrow
\end{aligned}
$$

- Also require an order of precedence

From highest to lowest: $\neg \quad \wedge \quad \vee \quad \Rightarrow \quad \Leftrightarrow$

Propositional Logic Semantics

- Propositions can have any semantic meaning:

  P = "Paris is the capital of France"

  Q = "The wumpus is dead"

  R = "Bill Gates is the US President"

- Compound functions can be derived from a 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*

Validity and Inference

$$((P \vee H) \wedge \neg H) \;\Rightarrow\; P$$

P	H	$P \vee H$	$(P \vee H) \wedge \neg H$	$((P \vee H) \wedge \neg H) \Rightarrow P$
False	*False*			
False	*True*			
True	*False*			
True	*True*			

- Truth tables can also be used to test validity of a sentence
- Remember to read implications as conditionals:

  $P \Rightarrow Q$ is read as "if P then Q"

Inference Rules for Propositional Logic

- Modus Ponens (Implication-Elimination)
  - From an implication and its premise, infer conclusion

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

- And-Elimination
  - From a conjunction, you can infer any conjunct

$$\frac{\alpha_1 \wedge \alpha_2 \wedge \alpha_3 \wedge ... \wedge \alpha_n}{\alpha_i}$$

Inference Rules for Propositional Logic

- And-Introduction
  - From a list of sentences, you can infer the conjunct

$$\frac{\alpha_1, \alpha_2, \alpha_3, \ldots \alpha_n}{\alpha_1 \wedge \alpha_2 \wedge \ldots \wedge \alpha_n}$$

- Or-Introduction
  - From a sentence, infer its disjunction with anything

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

Inference Rules for Propositional Logic

- Double-Negative Elimination
  - From a double negation, infer the positive sentence

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

- Unit Resolution
  - From a disjunction in which one is false, then you can infer the other is true

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

Inference Rules for Propositional Logic

- Resolution
  - Since beta cannot be both true and false, one of the disjuncts must be true

$$\frac{\alpha \vee \beta \;, \neg\beta \vee \gamma}{\alpha \vee \gamma}$$

  - Implication is transitive

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

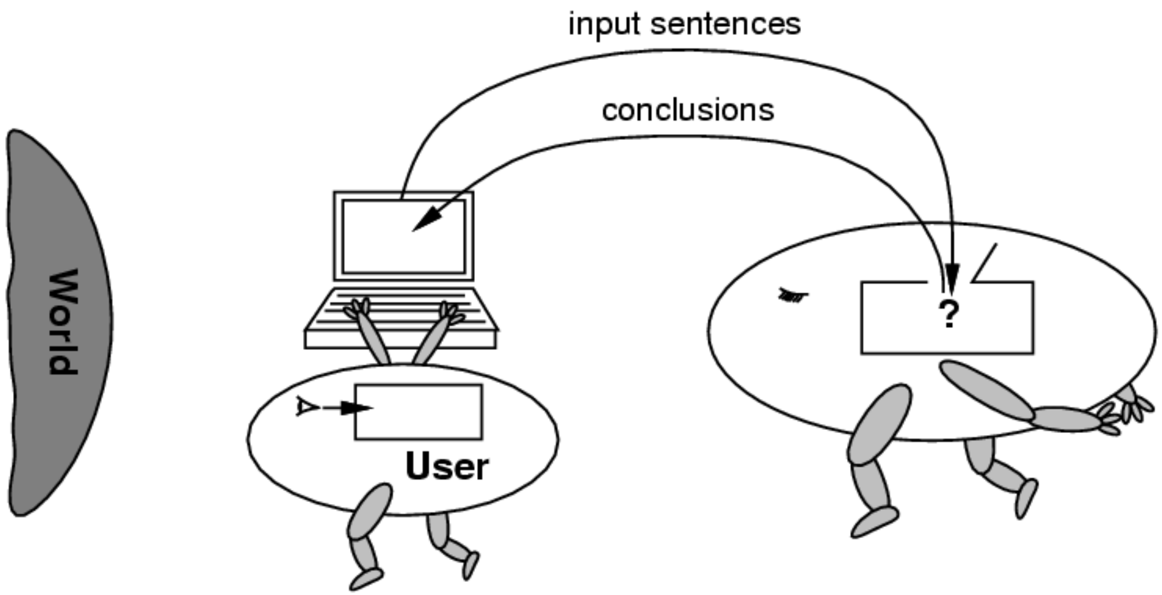
Truth Table for Resolution

α	β	γ	$\alpha \vee \beta$	$\neg\beta \vee \gamma$	$\alpha \vee \gamma$
False	*False*	*False*	*False*	*True*	*False*
False	*False*	*True*	*False*	*True*	*True*
False	*True*	*False*	*True*	*False*	*False*
False	*True*	*True*	*True*	*True*	*True*
True	*False*	*False*	*True*	*True*	*True*
True	*False*	*True*	*True*	*True*	*True*
True	*True*	*False*	*True*	*False*	*True*
True	*True*	*True*	*True*	*True*	*True*

- Truth tables can also be used to verify the inference rules

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

Logical Agents



- Input sentences can come from the user perceiving the world, or from a machine-readable representation of the world
- Infer new statements about the world that are valid

An Agent for the Wumpus World

- Convert perceptions into sentences:

  "In square [1,1], there is no breeze and no stench" … becomes…

  $\neg B_{11} \wedge \neg S_{11}$

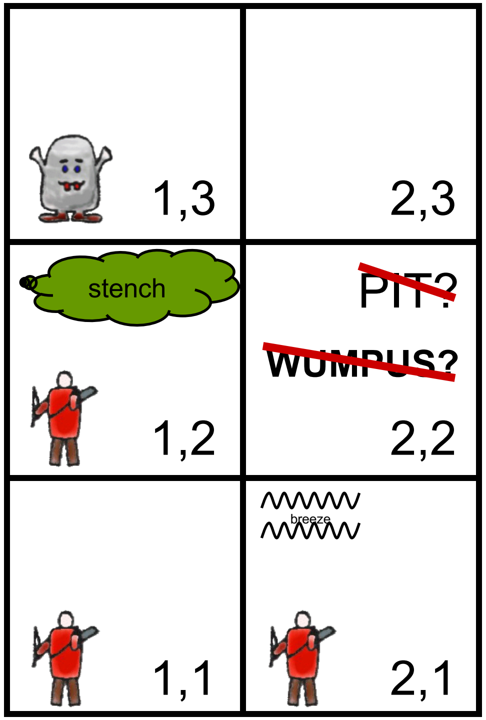
- Start with some knowledge of the world (in the form of rules)

$$R1 : \neg S_{11} \Rightarrow \neg W_{11} \wedge \neg W_{12} \wedge \neg W_{21}$$

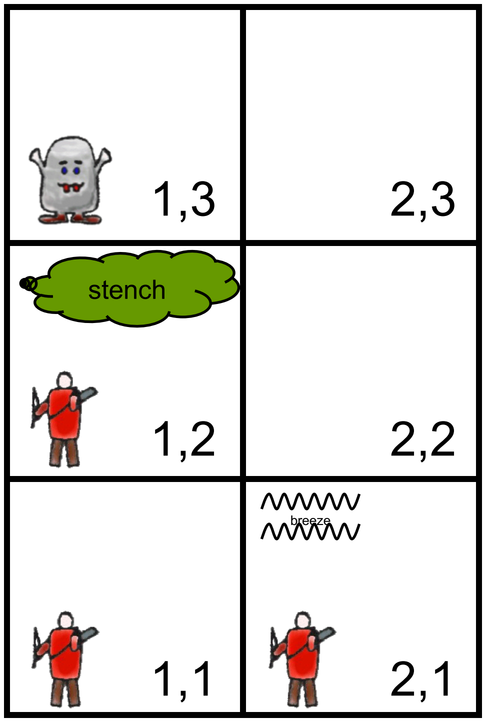
$$R2 : \neg S_{21} \Rightarrow \neg W_{11} \wedge \neg W_{21} \wedge \neg W_{22}$$

….

$$R4 : S_{12} \Rightarrow W_{13} \vee W_{12} \vee W_{22} \vee W_{11}$$



Finding the Wumpus



Percepts:

$\neg S_{11}$

$\neg S_{21}$

S_{12}

1. Apply modus ponens and and-elimination to $\neg S_{11} \Rightarrow \neg W_{11} \wedge \neg W_{12} \wedge \neg W_{21}$ to get

   $\neg \mathbf{W_{11}} \quad \neg \mathbf{W_{12}} \quad \neg \mathbf{W_{21}}$

2. Apply modus ponens and and-elimination to $\neg S_{21} \Rightarrow \neg W_{11} \wedge \neg W_{21} \wedge \neg W_{22}$ to get

   $\neg \mathbf{W_{22}} \quad \neg \mathbf{W_{21}} \quad \neg \mathbf{W_{31}}$

3. Apply modus ponens to

   $S_{12} \Rightarrow W_{13} \vee W_{12} \vee W_{22} \vee W_{11}$ to get

   $\mathbf{W_{13}} \vee \mathbf{W_{12}} \vee \mathbf{W_{22}} \vee \mathbf{W_{11}}$

4. Apply unit resolution to #3 and #1

   $\mathbf{W_{13}} \vee \mathbf{W_{22}}$

5. Apply unit resolution to #4 and #2

   $\mathbf{W_{13}}$

The wumpus is in square [1,3]!!!

Problems with Propositional Logic

- Too many propositions!
  - How can you encode a rule such as “don’t go forward if the wumpus is in front of you”?
  - In propositional logic, this takes (16 squares * 4 orientations) = 64 rules!
- Truth tables become unwieldy quickly
  - Size of the truth table is 2^n where n is the number of propositional symbols

More Problems with Propositional Logic

- No good way to represent changes in the world
  - How do you encode the location of the agent?
- What kinds of practical applications is this good for?
  - Relatively little

Coming Up…

- More powerful logic!
  - First-order logic (also known as First Order Predicate Calculus)