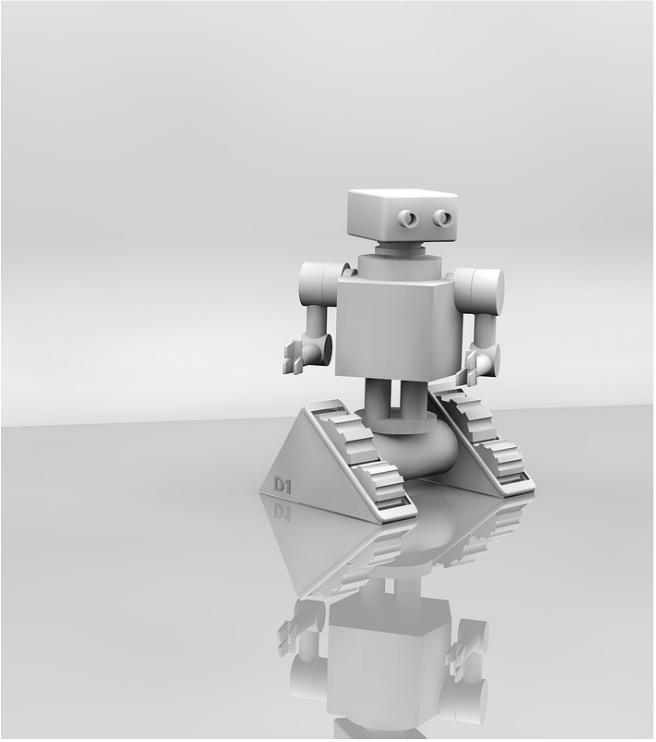


CPSC 481 Artificial Intelligence

Dr. Mira Kim

Mira.kim@fullerton.edu

What we will cover this week

- Propositional logic

Imperative/procedural programming

- An approach to programming where the program is a sequence of statements
- C++, Python, …
- Imperative programming focuses on describing *how* a program operates

Declarative programming

Inference engine — Domain-independent algorithms

Knowledge base (KB) — Domain-specific content (facts)

- Knowledge base = set of sentences in a formal language
- Declarative approach to building an agent:
  - `Tell` it what it needs to know
- Then it can `Ask` itself what to do - answers should follow from the KB

Some Knowledge Representation Languages

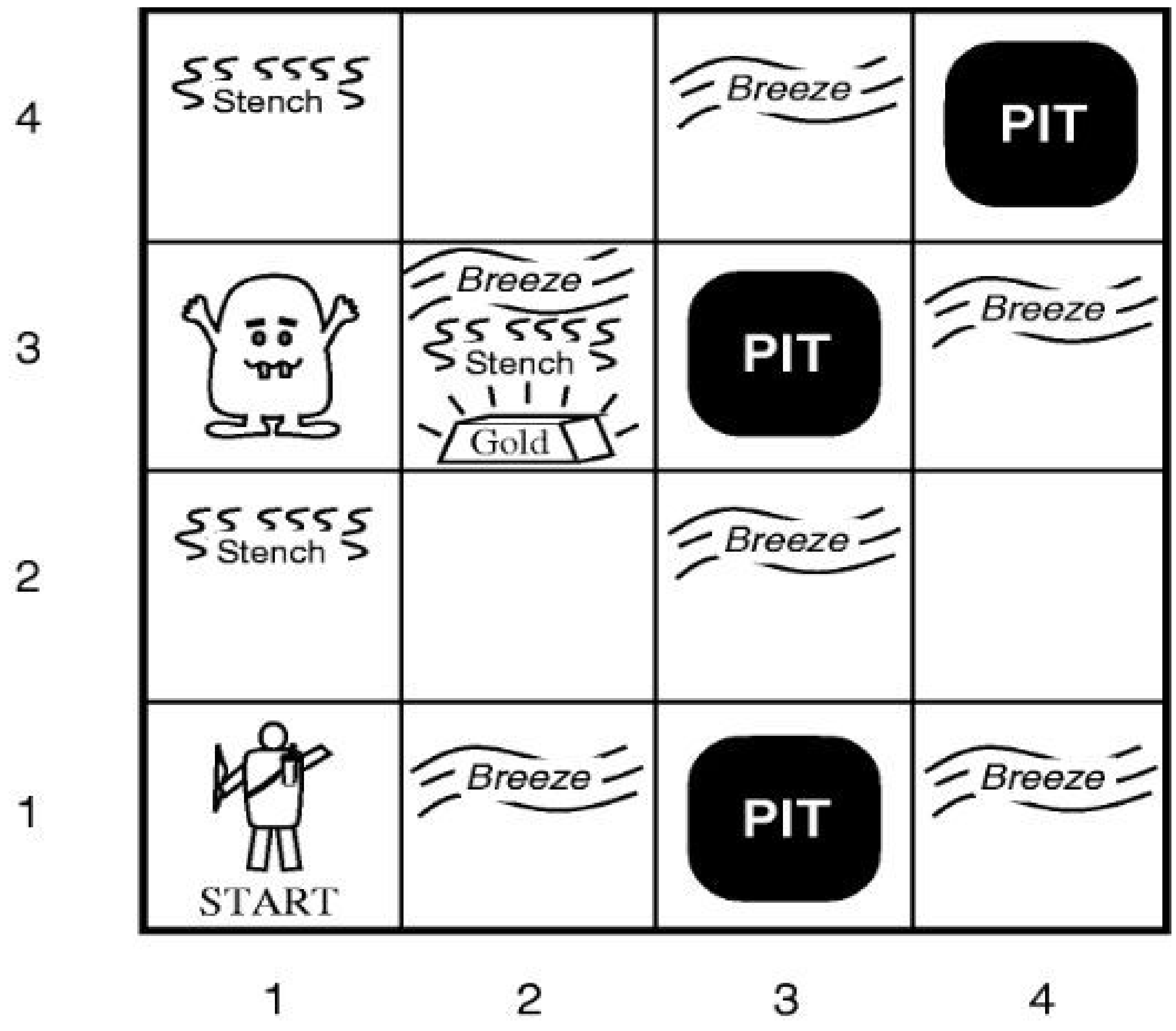
- Propositional Logic
- Predicate Calculus
- Frame Systems
- Influence Diagrams
- Semantic Networks
- Nonmonotonic Logic
- Concept Description Languages
- Rules with Certainty Factors
- Bayesian Networks

Some Knowledge Representation Languages

- All popular knowledge representation systems are a subset of
  - Logic
    - Either Propositional Logic
    - Or Predicate Calculus
  - Probability Theory
    - E.g.,: Bayesian networks

Wumpus world

- 4x4 grid world
- In the squares adjacent to the wumpus, you will get a stench
- In the square adjacent to a pit, you will feel a breeze
- In the square where the gold is, you will see a glitter
- You die if you enter a square containing a pit or a wumpus
- You can move one step in any direction
- Start from (1,1)
- Goal:
    - Move through the grid to get the gold without getting killed (by either the wumpus or pit)



Stench
Breeze
PIT
Breeze
Stench
Gold
PIT
Breeze
Stench
Breeze
Breeze
PIT
Breeze
START
1
2
3
4
1
2
3
4

Wumpus world logic

- if you sense a stench, then you knows the wumpus must be in the front or left or right square.
- if you feel a breeze, then it knows the PIT must be in the front or left or right square.
- if no stench and no breeze, all directly adjacent squares are safe.

Wumpus world characterization

Fully Observable No – only local perception

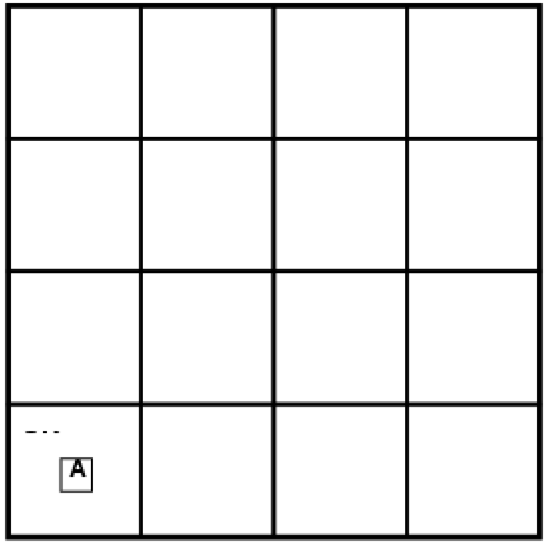
Deterministic Yes – outcomes exactly specified

Static Yes – Wumpus and Pits do not move

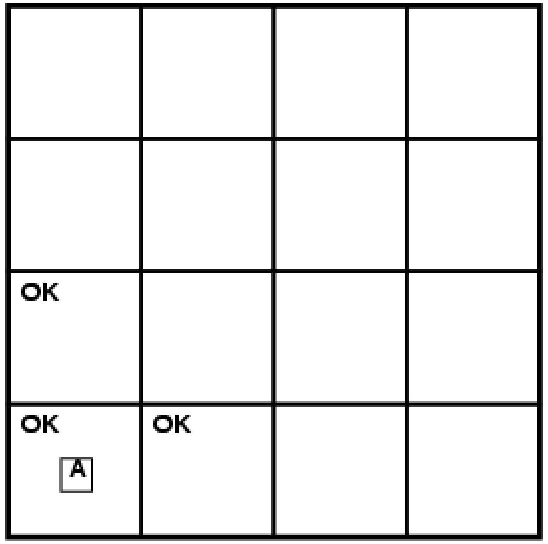
Discrete Yes

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

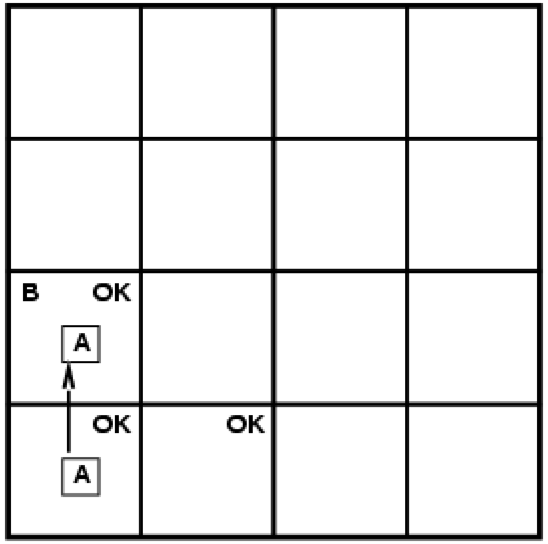
Exploring a wumpus world



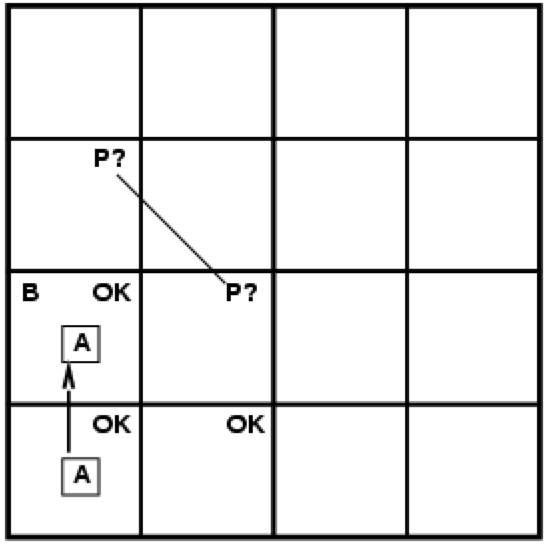
Exploring a wumpus world



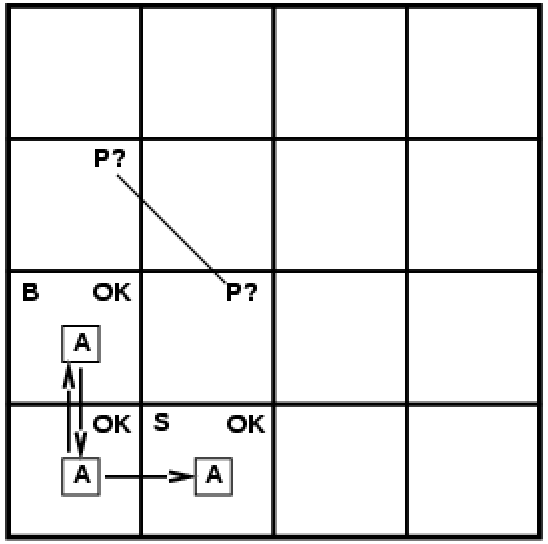
Exploring a wumpus world



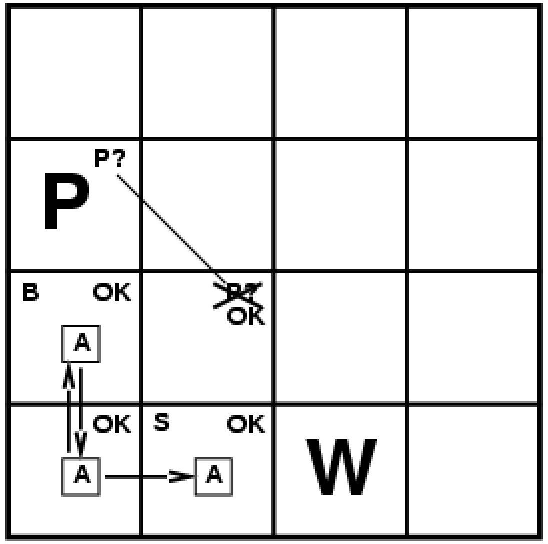
Exploring a wumpus world



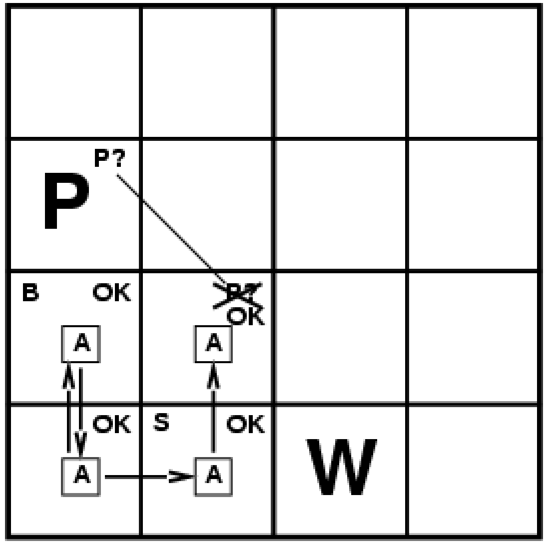
Exploring a wumpus world



Exploring a wumpus world

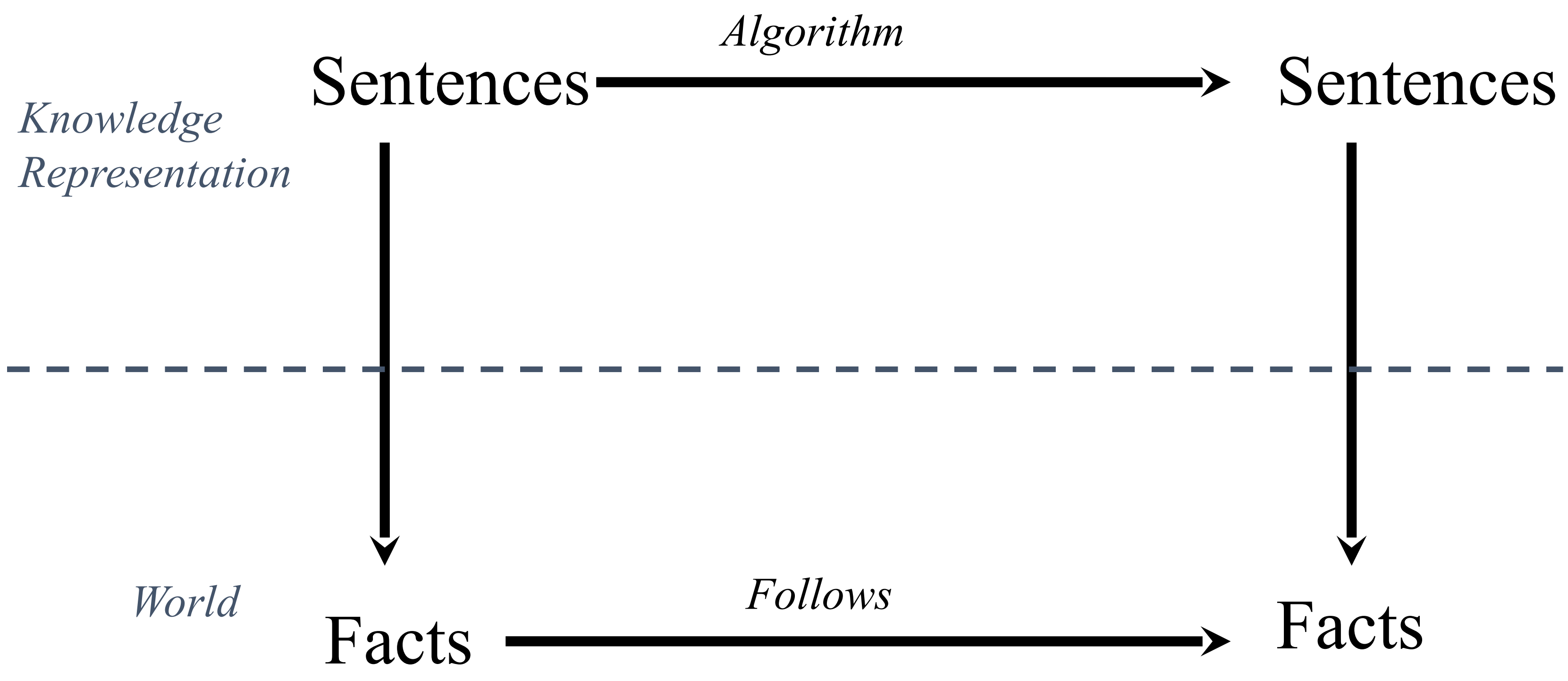


Exploring a wumpus world



Basic Idea of Logic

- By starting with true assumptions, you can deduce true conclusions.



Propositional logic

- Symbolic logic for manipulating propositions
  - Can be classified as either TRUE or FALSE
- **Logical constants**: true, false
- **Propositional symbols**: P, Q, S, ... (**atomic sentences**)
- Wrapping **parentheses**: (…)
- Sentences are combined by **connectives**:

  $\wedge$ ...and [conjunction]

  $\vee$ ...or [disjunction]

  $\rightarrow$...implies [implication / conditional]

  $\leftrightarrow$ ..is equivalent [biconditional]

  ~ ...not [negation]
- **Literal**: atomic sentence or negated atomic sentence

Examples

- P
- ~Q
- $Q \rightarrow P$
- $(P \wedge Q) \rightarrow R$

Syntax

- A sentence is defined as follows:
  - A symbol is a sentence
  - If S is a sentence, then ~S is a sentence
  - If S is a sentence, then (S) is a sentence
  - If S and T are sentences, then ($S \vee T$), ($S \wedge T$), ($S \rightarrow T$), and ($S \leftrightarrow T$) are sentences
  - A sentence results from a finite number of applications of the above rules
- Well formed formula, WFF

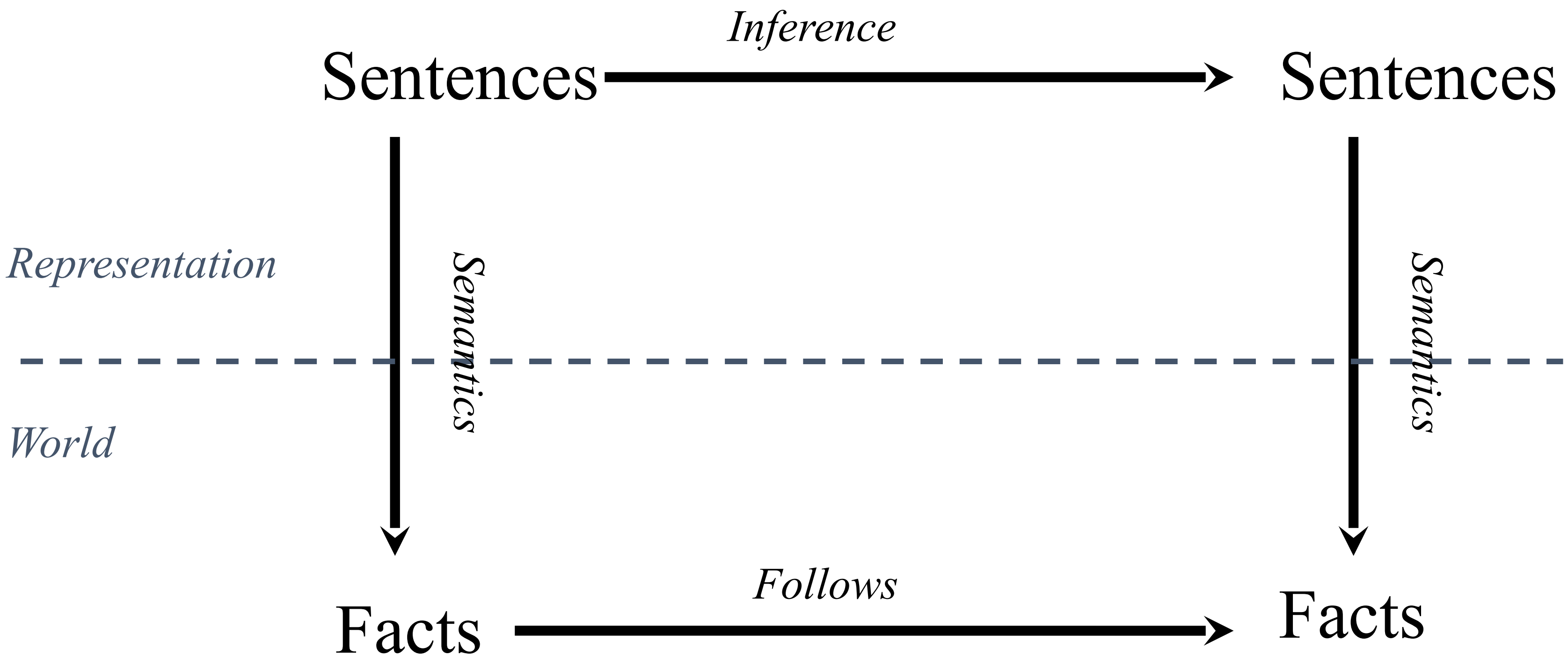
Semantics

- User defines the set of propositional symbols: P, Q, …
- User defines the **semantics** (meaning) of each propositional symbol:
- P means “It is hot.”
- Q means “It is humid.”
- R means “It is raining.”
- $(P \wedge Q) \rightarrow R$

  “If it is hot and humid, then it is raining”
- $Q \rightarrow P$

  “If it is humid, then it is hot”

- ***Syntax***: which arrangements of symbols are *legal sentences*
  - “Well-formed formulae”
- ***Semantics***: what the symbols *mean* in the world
  - (Mapping between symbols and worlds)



Truth tables

A	~A
T	F
F	T

A	B	A ˅ B
T	T	T
T	F	T
F	T	T
F	F	F

A	B	A ˄ B
T	T	T
T	F	F
F	T	F
F	F	F

A	B	A → B
T	T	T
T	F	F
F	T	T
F	F	T

Truth value of a sentence

- Given the truth values of all symbols in a sentence, it can be "evaluated" to determine its **truth value** (True or False).
- Truth table
  - P
  - $\sim P$
  - $P \wedge Q$
  - $Q \rightarrow P$
  - $(P \wedge Q) \rightarrow R$

Knowledgebase (KB) with Propositional Logic

- KB contains a set of propositional logic formulae that are known to be true
  - The premises
- Question?
  - Are there other formulae that are also true given this specific KB?

Logical Entailment

- **Entailment: KB $\models$ Q**
  - Q is *entailed* by KB if and only if :
    - the conclusion is true for every possible world in which all the premises are true.

Entailment and derivation

- **Entailment: KB $\models$ Q**
  - Q is *entailed* by KB if and only if :
    - the conclusion is true for every possible world in which all the premises are true.
- **Derivation: KB $\vdash$ Q**
  - We can derive Q from KB if there is a *proof* consisting of a sequence of valid inference steps starting from the premises in KB and resulting in Q
  - An algorithm

Two important properties for inference

Soundness: If KB $\vdash$ Q then KB $\models$ Q

- If Q is derived from a set of sentences KB using a given set of rules of inference, then Q is entailed by KB.
- Hence, inference produces only real entailments, or any sentence that follows deductively from the premises is valid.

Completeness: If KB $\models$ Q then KB $\vdash$ Q

- If Q is entailed by a set of sentences KB, then Q can be derived from KB using the rules of inference.
- Hence, inference produces all entailments, or all valid sentences can be proved from the premises.

Proof methods

- Proof methods divide into (roughly) two kinds:
- Application of inference rules
- Generate new sentences from old
  - Proof = a sequence of inference rule applications
    Can use inference rules as operators in a standard search algorithm
- Model checking
- truth table enumeration
- improved backtracking, e.g., Davis-Putnam-Logemann-Loveland (DPLL)
  - heuristic search in model space (sound but incomplete)
  - e.g., hill-climbing algorithms

“Proofs” using a truth table

- If we have a finite number of premises, then we can build a truth table
- Exhaustively test every possible “world”
- Check
  - for every case where all premises are true,
  - is the conclusion is also true?

"Proofs" using a truth table

- Premises (KB)
  1) Q
  2) $(P \wedge Q) \rightarrow R$
  3) $Q \rightarrow P$
- Does R follow?
- Yes

A	B	$A \rightarrow B$
T	T	T
T	F	F
F	T	T
F	F	T

P	Q	R	Q	$(P \wedge Q) \rightarrow R$	$Q \rightarrow P$	R (conclusion)
T	**T**	**T**	**T**	**T**	**T**	**T**
T	T	F	T	F	T	F
T	F	T	F	T	T	T
T	F	F	F	T	T	F
F	T	T	T	T	F	T
F	T	F	T	T	F	F
F	F	T	F	T	T	T
F	F	F	F	T	T	F

Check

for every case where all premises are true

"Proofs" using a truth table

- Premises (KB)
  1) Q
  2) (P ^ Q) → R
  3) Q → P
- Does R follow?
- Yes

P	Q *	R	(P ^ Q) → R *	Q → P *	R (concl)
True	**T**	**T**	**T**	**T**	**T** (all prem true)
T	T	F	F	T	F
T	F	T	T	T	T
T	F	F	T	T	F
F	T	T	T	F	T
F	T	F	T	F	F
F	F	T	T	T	T
F	F	F	T	T	F

A	B	A → B
T	T	T
T	F	F
F	T	T
F	F	T

Check

for every case where all premises are true

In-class exercise

- Given
  - $P \rightarrow (Q \rightarrow R)$
  - Q
- Does this follow?
  - $P \rightarrow R$
- Use Truth table

“Proofs” using a truth table

- Can we prove something does *not* follow?

1) $P \rightarrow Q$
2) $\sim Q \rightarrow R$
3) R

- Does P follow?
- No

P	Q	R *	P→Q *	~Q→R *	P
True	T	T	T	T	T
True	T	F	T	T	True
T	F	T	F	T	T
T	F	F	F	F	T
F	T	T	T	T	F (concl is FALSE)
F	T	F	T	T	F
F	F	T	T	T	F(concl is FALSE)
F	F	F	T	F	F

Inference by enumeration (truth tables)

Depth-first enumeration of all models is sound and complete

-

```
function TT-ENTAILS?(KB, α) returns true or false
   symbols ← a list of the proposition symbols in KB and α
   return TT-CHECK-ALL(KB, α, symbols, [])

function TT-CHECK-ALL(KB, α, symbols, model) returns true or false
   if EMPTY?(symbols) then
      if PL-TRUE?(KB, model) then return PL-TRUE?(α, model)
      else return true
   else do
      P ← FIRST(symbols); rest ← REST(symbols)
      return TT-CHECK-ALL(KB, α, rest, EXTEND(P, true, model)) and
             TT-CHECK-ALL(KB, α, rest, EXTEND(P, false, model))
```

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

Rules of inference

- **Logical inference** is used to create new sentences that logically follow from a given set of sentences (KB).
- An inference rule is **sound** if every sentence X produced by an inference rule operating on a KB logically follows from the KB.
- The inference rule does not create any contradictions

Sound (correct) rules of inference

RULE	**PREMISE**	**CONCLUSION**
Modus Ponens	A, A $\rightarrow$ B	B
AND Introduction	A, B	A $\wedge$ B
AND Elimination	A $\wedge$ B	A
Double Negation	~~A	A
Resolution	A $\vee$ B, ~B $\vee$ C	A $\vee$ C
Unit resolution	A $\vee$ B, ~B	A

- Each can be shown to be sound/correct using a truth table

Modus ponens is sound

A	B	$A \rightarrow B$
True	True	True
True	False	False
False	True	True
False	False	True

Premise: A, $A \rightarrow B$

Conclusion: B

conclusion is true whenever the premise is true

Proofs

- A **proof** is a sequence of sentences, where each sentence is either a premise or a sentence derived from earlier sentences in the proof by one of the rules of inference.
- The last sentence is the **theorem** (also called goal or query) that we want to prove.

Example of a proof

- Premises
  1) Q
  2) $(P \wedge Q) \rightarrow R$
  3) $Q \rightarrow P$
- How to prove?
  - R

Example of a proof

1. Q
   - Premise 1
2. $Q \rightarrow P$
   - Premise 3
3. P
   - Modus ponens on 1 and 2
4. $(P \wedge Q) \rightarrow R$
   - Premise 2
5. $(P \wedge Q)$
   - AND introduction on 1 and 3
6. R
   - Modus ponens on 4 and 5

Resolution algorithm

- Disadvantages of our derivation method
  - Which rule of inference to apply?
  - Apply to which sentences?
- Needed: an algorithm – can be executed by a computer
- Alan Robinson, 1965

Resolution rule

RULE	PREMISE	CONCLUSION
~~Modus Ponens~~	~~$A, A \rightarrow B$~~	~~B~~
~~AND Introduction~~	~~A, B~~	~~$A \wedge B$~~
~~AND Elimination~~	~~$A \wedge B$~~	~~A~~
~~Double Negation~~	~~$\sim\sim A$~~	~~A~~
Resolution	$A \vee B, \sim B \vee C$	$A \vee C$
Unit resolution	$A \vee B, \sim B$	A

This single rule is sufficient

Applying the resolution rule

- First, Convert to **Conjunctive Normal Form (CNF)**
  - CNF: Knowledgebase (KB) is a *conjunction* of *disjunctions*
    - AND of clauses (a conjunction of clauses)
    - Each clause is an OR of literals (a disjunction of literals)
    - A literal is either a propositional variable or its negation
  - Example: $(A \vee B) \wedge (A \vee \sim C) \wedge (\sim B \vee \sim C)$
- KB can then be represented as a list of *conjunctions*:
  1. $A \vee B$
  2. $A \vee \sim C$
  3. $\sim B \vee \sim C$

Convert to CNF

1. Eliminate

2. Eliminate
   - with
3. Move ~ (negation) inwards
   - DeMorgan's laws and double negation
4. Distribute over
   - Distributive property

CNF: Eliminate

- is equivalent to

CNF: Dealing with parentheses

- DeMorgan's laws
  - $\sim(A \vee B) \equiv \sim A \wedge \sim B$
  - $\sim(A \wedge B) \equiv \sim A \vee \sim B$
- Distributive property
  - $(A \wedge B) \vee C \equiv (A \vee C) \wedge (B \vee C)$

Examples: convert to CNF

- ~(~P $\vee$ Q)
- ~(~P $\vee$ Q) $\vee$ R
- (P $\rightarrow$ Q) $\rightarrow$ R

Examples: convert to CNF

- ~(~P ˅ Q)
  - ~~P ˄ ~Q
  - P ˄ ~Q
    1. P
    2. ~Q
- ~(~P ˅ Q) ˅ R
  - (P ˄ ~Q) ˅ R
  - (P ˅ R) ˄ (~Q ˅ R)
    1. P ˅ R
    2. ~Q ˅ R
- (P → Q) → R

In-class exercise

- Convert the following to CNF
  - $(P \rightarrow Q) \rightarrow R$

Proof by Resolution Refutation

- Does Premise (KB) $\rightarrow$ Conclusion (α)?

1. Convert all premise sentences (KB) to CNF
2. Add the *negated* conclusion
3. Repeatedly apply rule of resolution until
   - Derive FALSE (contradiction): Conclusion is **valid**
   - Can’t apply any more: Conclusion **cannot be proved**

- Proof by contradiction

Example 1

- Premise:

  $P \vee Q$

  $P \rightarrow R$

  $Q \rightarrow R$

- Prove:

  R

1. $P \vee Q$
2. $\sim P \vee R$
3. $\sim Q \vee R$
4. $\sim R$
5. $Q \vee R$
   1. Resolution on 1,2
6. R
   1. Resolution on 3,5
7. FALSE
   1. Resolution on 4,6

Conclusion follows

In-class exercise

- Given
  - $P \rightarrow (Q \rightarrow R)$
  - Q
- Does this follow?
  - $P \rightarrow R$
- Use
  1. Resolution

Proof by Resolution Refutation

- Sound
  - The answer is always correct
- Complete
  - It always generates an answer

In-class exercise

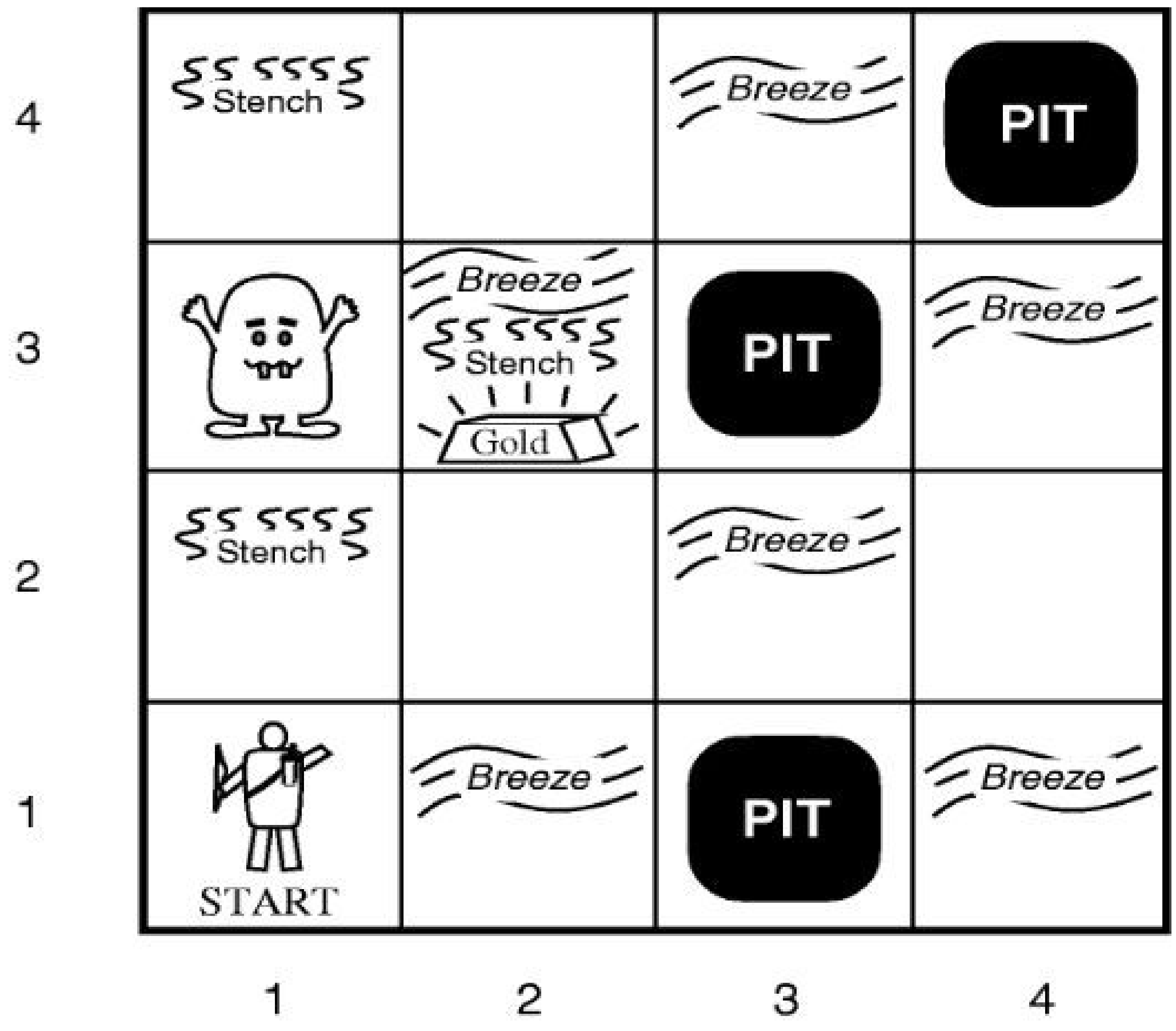
- Given
  - $P \rightarrow (Q \rightarrow R)$
  - Q
- Does this follow?
  - R
- Use
  1. Resolution

Efficient proofs

- Not every application of resolution is needed
- Unit preference
  - prefer a resolution step involving an unit clause (clause with one literal)
  - Produces a shorter clause
- Set of support
  - Choose a resolution involving the negated goal or any clause derived from the negated goal

Wumpus world

- 4x4 grid world
- In the squares adjacent to the wumpus, you will get a stench
- In the square adjacent to a pit, you will feel a breeze
- In the square where the gold is, you will see a glitter
- You die if you enter a square containing a pit or a wumpus
- You can move one step in any direction
- Start from (1,1)
- Goal:
  - Move through the grid to get the gold without getting killed (by either the wumpus or pit)



4
3
2
1
Stench
Breeze
PIT
Breeze
Stench
Gold
PIT
Breeze
Stench
Breeze
START
Breeze
PIT
Breeze
1
2
3
4

Wumpus world logic

- if you sense a stench, then you knows the wumpus must be in the front or left or right square.
- if you feel a breeze, then it knows the PIT must be in the front or left or right square.
- if no stench and no breeze, all directly adjacent squares are safe.

Wumpus world logic

- Develop a propositional logic system to decide where to move
- What symbols?

Wumpus world logic symbols

- : pit in (x,y)
- : wumpus in (x,y)
- : agent perceives breeze in (x,y)
- : agent perceives stench in (x,y)

Wumpus world logic

- Develop a propositional logic system to decide where to move
- What symbols?
- How to represent:
  - S11 = None ⇒ S12 = Safe ∧ S21 = Safe

Inference-based agents in the wumpus world

A wumpus-world agent using propositional logic:

$\neg P_{1,1}$

$\neg W_{1,1}$

Inference-based agents in the wumpus world

A wumpus-world agent using propositional logic:

$\neg P_{1,1}$

$\neg W_{1,1}$

Rules for breeze and stench?

Inference-based agents in the wumpus world

A wumpus-world agent using propositional logic:

$\neg P_{1,1}$

$\neg W_{1,1}$

Rules for breeze and stench:

$B_{x,y} \Leftrightarrow (P_{x,y+1} \vee P_{x,y-1} \vee P_{x+1,y} \vee P_{x-1,y})$

$S_{x,y} \Leftrightarrow (W_{x,y+1} \vee W_{x,y-1} \vee W_{x+1,y} \vee W_{x-1,y})$

Inference-based agents in the wumpus world

A wumpus-world agent using propositional logic:

$\neg P_{1,1}$

$\neg W_{1,1}$

$B_{x,y} \Leftrightarrow (P_{x,y+1} \vee P_{x,y-1} \vee P_{x+1,y} \vee P_{x-1,y})$

$S_{x,y} \Leftrightarrow (W_{x,y+1} \vee W_{x,y-1} \vee W_{x+1,y} \vee W_{x-1,y})$

Exactly one wumpus?

Inference-based agents in the wumpus world

A wumpus-world agent using propositional logic:

$\neg P_{1,1}$

$\neg W_{1,1}$

$B_{x,y} \Leftrightarrow (P_{x,y+1} \vee P_{x,y-1} \vee P_{x+1,y} \vee P_{x-1,y})$

$S_{x,y} \Leftrightarrow (W_{x,y+1} \vee W_{x,y-1} \vee W_{x+1,y} \vee W_{x-1,y})$

Exactly one wumpus:

$W_{1,1} \vee W_{1,2} \vee \ldots \vee W_{4,4}$

$\neg W_{1,1} \vee \neg W_{1,2}$

$\neg W_{1,1} \vee \neg W_{1,3}$

Inference-based agents in the wumpus world

A wumpus-world agent using propositional logic:

$\neg P_{1,1}$

$\neg W_{1,1}$

$B_{x,y} \Leftrightarrow (P_{x,y+1} \vee P_{x,y-1} \vee P_{x+1,y} \vee P_{x-1,y})$

$S_{x,y} \Leftrightarrow (W_{x,y+1} \vee W_{x,y-1} \vee W_{x+1,y} \vee W_{x-1,y})$

$W_{1,1} \vee W_{1,2} \vee \ldots \vee W_{4,4}$

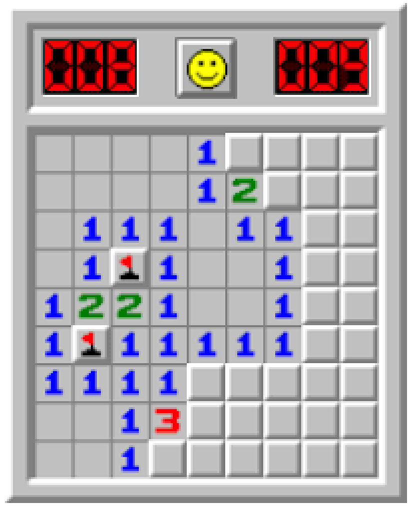
$\neg W_{1,1} \vee \neg W_{1,2}$

$\neg W_{1,1} \vee \neg W_{1,3}$

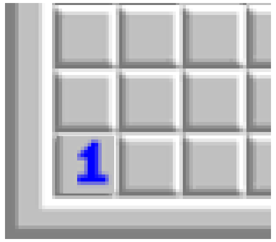
…

$\Rightarrow$ 64 distinct proposition symbols, 155 sentences

Classwork: Minesweeper



- <u>Minesweeper</u> is related to the Wumpus world.
  - The minesweeper world is a rectangular grid with invisible mines scattered around it. Any cell may be probed by the player; instant death follows if a mine is probed. Minesweeper indicates the presence of mines by revealing, in each probed cell, the *number* of mines that are directly or diagonally adjacent. The goal is to probe every unmined cell.
- Let proposition Xij = True (where i=0,1,2,… and j=0,1,2,…) denote that cell (i,j) contains a mine.
- Let probing the ***corner*** cell (0,0) reveal **1** mine in an adjacent cell. How can the assertion that exactly one mine is adjacent to (0,0) be expressed in Propositional logic (as some logical combination of the Xij propositions)?



Limitations of logic

Contradiction in the premise

- Premise:
  P
  ~P
- Prove:
  C

1. P
2. ~P
3. ~C
4. FALSE (resolving 1,2)

Contradiction in the premise

- $P \wedge {\sim}P \rightarrow C$ is valid
- *Any* conclusion can be proved from a contradiction
- Pure logic systems are *brittle*

Expressiveness limitation of propositional logic

- KB contains "physics" sentences for *every single square*
- For *every* location [*x*,*y*]
  - $B_{x,y} \Leftrightarrow (P_{x,y+1} \vee P_{x,y-1} \vee P_{x+1,y} \vee P_{x-1,y})$
- Rapid proliferation of clauses
- Predicate logic introduces *variables* to logic

References

- George F. Luger, Artificial Intelligence: Structures and Strategies for Complex Problem Solving, 6th edition, Addison Wesley, 2009.
    - Section 2.1
- Russel and Norvig, Artificial Intelligence: A Modern Approach, 3rd edition, Prentice Hall, 2010.
    - Section 7.4 Propositional Logic: A Very Simple Logic
    - Section 7.5 Propositional Theorem Proving