

# Logical Agents. Propositional Logic and Inference.

## Chapter 7

Sept 22, 2008

COMP 131, Lecture 6

1

## Announcements

- A0 graded. Answers available in class.
- A2 Question 4 has been revised. Available on course webpage.
- A2 now due on Monday Sept 29<sup>th</sup>.

Sept 22, 2008

COMP 131, Lecture 6

2

## Last time: Adversarial search

- A game is defined by the **initial state**, the legal actions, a **terminal test** (when the game is over), and a **utility function** that applies to terminal states.
- Games can be **zero-sum** (when A's win is equal to B's loss) or **non-zero-sum**, can have **perfect** or **imperfect information**, can involve **turn-taking** or continuous play
- The **minimax** algorithm solves 2-player zero-sum games of perfect information optimally by a depth-first enumeration of the **game tree**, choosing actions that alternately maximize and minimize the state value
- For games with an element of chance, **expected value** is used in **chance nodes** as their **evaluation function**
- The **alpha-beta pruning** algorithm computes the same moves as **minimax** but doesn't expand provably irrelevant subtrees

Sept 22, 2008

COMP 131, Lecture 6

3

## Today

- Knowledge-based agents
  - can form representations of the world
  - can reason about the world and what to do based on those representations
- The Wumpus world
  - illustrates logic in general
- Propositional logic
- Equivalence, validity, satisfiability
- Reasoning by inference
- Logical circuits

Sept 22, 2008

COMP 131, Lecture 6

4

## Knowledge-based agents

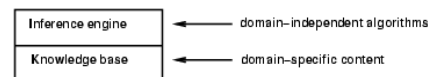
- Our intelligent activities can be abstracted:
  - we know things
  - we can reason about things we know
- Knowledge representation: a formal language to codify things we (or an agent) know
- Reasoning: updating our knowledge representation according to general purpose rules
- E.g., infer a patient diagnosis from test results, infer an action sequence from task description, infer a decision based on data

Sept 22, 2008

COMP 131, Lecture 6

5

## 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

Sept 22, 2008

COMP 131, Lecture 6

6

## CSP a form of knowledge rep.

- But some problems are not easily formulated as CSPs
  - what's this patient's diagnosis?
  - should I carry an umbrella today?
  - should this applicant get a line of credit?

Sept 22, 2008

COMP 131, Lecture 6

7

## A simple knowledge-based agent

```

function KB-AGENT(percept) returns an action
  static: KB, a knowledge base
         t, a counter, initially 0, indicating time
  TELL(KB, MAKE-PERCEPT-SENTENCE(percept, t))
  action ← ASK(KB, MAKE-ACTION-QUERY(t))
  TELL(KB, MAKE-ACTION-SENTENCE(action, t))
  t ← t + 1
  return action
    
```

- The agent must be able to:
  - represent states, actions, etc.
  - incorporate new percepts
  - update internal representations of the world
  - deduce hidden properties of the world
  - deduce appropriate actions

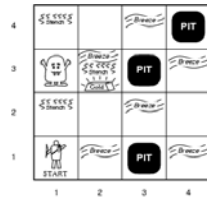
Sept 22, 2008

COMP 131, Lecture 6

8

## Wumpus World PEAS description

- **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
- **Sensors:** Stench, Breeze, Glitter, Bump, Scream
- **Actuators:** Left turn, Right turn, Forward, Grab, Release, Shoot



Sept 22, 2008

COMP 131, Lecture 6

9

## Wumpus world properties

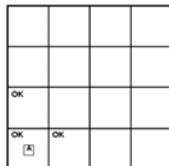
- **Fully Observable?**
  - No – only local perception
- **Deterministic?**
  - Yes – outcomes exactly specified
- **Episodic?**
  - No – sequential at the level of actions
- **Static?**
  - Yes – Wumpus and Pits do not move
- **Discrete?**
  - Yes
- **Single agent?**
  - Yes – Wumpus is essentially a natural feature

Sept 22, 2008

COMP 131, Lecture 6

10

## Exploring a wumpus world

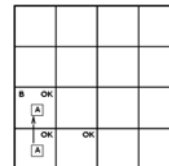


Sept 22, 2008

COMP 131, Lecture 6

11

## Exploring a wumpus world

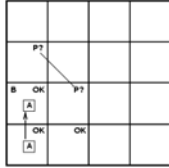


Sept 22, 2008

COMP 131, Lecture 6

12

## Exploring a wumpus world

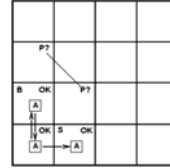


Sept 22, 2008

COMP 131, Lecture 6

13

## Exploring a wumpus world

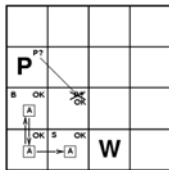


Sept 22, 2008

COMP 131, Lecture 6

14

## Exploring a wumpus world

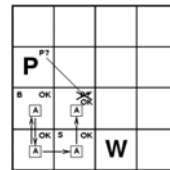


Sept 22, 2008

COMP 131, Lecture 6

15

## Exploring a wumpus world

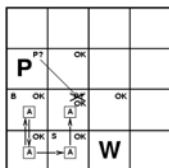


Sept 22, 2008

COMP 131, Lecture 6

16

## Exploring a wumpus world

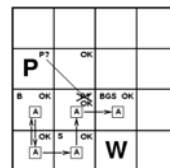


Sept 22, 2008

COMP 131, Lecture 6

17

## Exploring a wumpus world



Sept 22, 2008

COMP 131, Lecture 6

18

## What is a logic?

- **Logics** are formal languages for representing information such that conclusions can be drawn
  - **syntax**: defines the acceptable (legal) sentences in the language
  - **semantics**: define the "meaning" of legal sentences
  - **proof system**: a way to derive new legal sentences
- Why proofs?
  - multiple percepts to conclusions about the world
  - current state and operator to conclusions about properties of next state

Sept 22, 2008

COMP 131, Lecture 6

19

## Syntax

- **KB** = set of sentences
  - e.g., in arithmetic
    - $x + y = 4$  is a well-formed (syntactically correct) sentence
    - $x \ 2 \ y + =$  is not
  - compare to English
    - "The cat jumped over the counter." is well-formed
    - "Jumped the the cat counter over." is not
    - "Colorless green ideas sleep furiously." is well-formed

Sept 22, 2008

COMP 131, Lecture 6

20

## Propositional logic syntax

- A sentence of propositional logic is a well formed formula (wff):
  - the **literal** *true* or *false* is a sentence
  - a propositional variable (e.g., P, Q, R...) is a sentence
  - if *a* and *b* are sentences, then so are
    - $(a)$ ,  $\neg a$ ,  $a \wedge b$ ,  $a \vee b$ ,  $a \Rightarrow b$ ,  $a \Leftrightarrow b$
  - nothing else is a sentence

Sept 22, 2008

COMP 131, Lecture 6

21

## Precedence – shorthand

$\neg$	$\wedge$	$\vee$	$\Rightarrow$	$\Leftrightarrow$
--------	----------	--------	---------------	-------------------

highest

lowest

- $A \vee B \wedge \neg C$  is equivalent to  $A \vee (B \wedge (\neg C))$
- $A \wedge B \Rightarrow C \vee D$  is equivalent to  $(A \wedge B) \Rightarrow (C \vee D)$
- Allows to drop parentheses

Sept 22, 2008

COMP 131, Lecture 6

22

## Semantics

- **Semantics** defines the meaning of legal sentences
  - in logic, the semantics define the **truth** of each sentence with respect to each **possible world (model, interpretation)**
- E.g.,  $x + y = 4$  is true in a world where *x* is 2 and *y* is 2, but false in a world where *x* is 1 and *y* is 2
- Every sentence must be true or false in every possible world, there is no in between

Sept 22, 2008

COMP 131, Lecture 6

23

## Propositional logic semantics

- Meaning of a sentence is its truth value **{t, f}**
- Rules of semantics:
  - *true* has value **t** in all models
  - *false* has value **f** in all models
  - $\neg a$  has value **t** in models where *a* has value **f** (negation)
  - $a \wedge b$  has value **t** in models where both *a* and *b* have value **t** (conjunction)
  - $a \vee b$  has value **t** in models where *a* or *b* or both have value **t** (disjunction)

Sept 22, 2008

COMP 131, Lecture 6

24

## Some important shorthand

- $a \Rightarrow b$  is equivalent (by definition) to  $\neg a \vee b$ 
  - conditional, implication
- $a \Leftrightarrow b$  is equivalent (by definition) to  $(a \Rightarrow b) \wedge (b \Rightarrow a)$ 
  - biconditional, equivalence

Sept 22, 2008

COMP 131, Lecture 6

25

## Truth tables

- Truth tables enumerate every possible model
- Conditional and biconditional on blackboard

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

Sept 22, 2008

COMP 131, Lecture 6

26

## Proof system: entailment

- **Entailment** means that one thing **follows from** another:
 
$$KB \models \alpha$$
- Knowledge base  $KB$  entails sentence  $\alpha$  if and only if  $\alpha$  is true in all worlds where  $KB$  is true
  - $x+y = 4$  entails  $4 = x+y$
  - “It is Monday” and “It is raining” entail “It is Monday”
  - entailment is a relationship between sentences (i.e., **syntax**) that is based on **semantics**

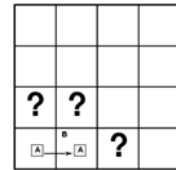
Sept 22, 2008

COMP 131, Lecture 6

27

## Entailment in the wumpus world

Situation after detecting nothing in [1,1], moving right, breeze in [2,1]



Consider possible models for  $KB$  assuming only pits

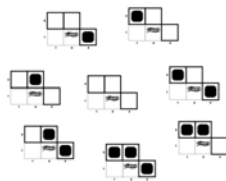
3 Boolean choices (pit or no pit)  $\Rightarrow$  8 possible models

Sept 22, 2008

COMP 131, Lecture 6

28

## Wumpus models



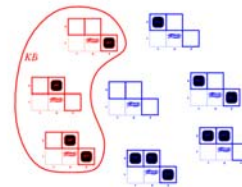
Sept 22, 2008

COMP 131, Lecture 6

29

## Wumpus models

red = true



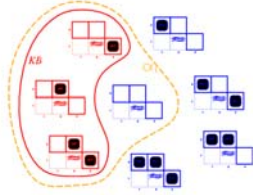
- $KB$  = wumpus-world rules + observations

Sept 22, 2008

COMP 131, Lecture 6

30

## Wumpus models



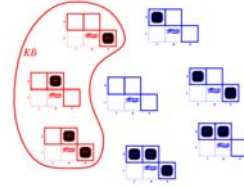
- $KB$  = wumpus-world rules + observations
- $\alpha_1 = "[1,2]$  has no pit",  $KB \models \alpha_1$ , proved by model checking

Sept 22, 2008

COMP 131, Lecture 6

31

## Wumpus models



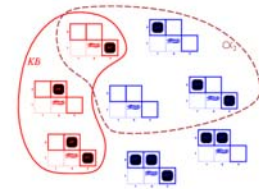
- $KB$  = wumpus-world rules + observations

Sept 22, 2008

COMP 131, Lecture 6

32

## Wumpus models



- $KB$  = wumpus-world rules + observations
- $\alpha_2 = "[2,2]$  is safe",  $KB \not\models \alpha_2$

Sept 22, 2008

COMP 131, Lecture 6

33

## Inference

- $KB \vdash_i \alpha$  = sentence  $\alpha$  can be derived (inferred) from  $KB$  by procedure  $i$
- **Soundness:**  $i$  is sound if whenever  $KB \vdash_i \alpha$ , it is also true that  $KB \models \alpha$
- **Completeness:**  $i$  is complete if whenever  $KB \models \alpha$ , it is also true that  $KB \vdash_i \alpha$
- That is, the procedure will answer any question whose answer follows from what is known by the  $KB$ .

Sept 22, 2008

COMP 131, Lecture 6

34

## Wumpus world sentences

Let  $P_{i,j}$  be true if there is a pit in  $[i, j]$ .

Let  $B_{i,j}$  be true if there is a breeze in  $[i, j]$ .

$\neg P_{1,1}$

$\neg B_{1,1}$

$B_{2,1}$

- "Pits cause breezes in adjacent squares"

$B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$

$B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$

Sept 22, 2008

COMP 131, Lecture 6

35

## Truth tables for inference

$B_{1,1}$	$B_{2,1}$	$P_{1,1}$	$P_{1,2}$	$P_{2,1}$	$P_{2,2}$	$P_{3,1}$	$KB$	$\alpha_1$
false	false	false	false	false	false	false	false	true
false	false	false	false	false	false	true	false	true
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
false	true	false	false	false	false	false	false	true
false	true	false	false	false	false	true	true	true
false	true	false	false	false	true	true	true	true
false	true	false	false	true	false	false	false	true
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
true	true	true	true	true	true	true	false	false

Sept 22, 2008

COMP 131, Lecture 6

36

## Inference by enumeration

- 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  $O(n)$

Sept 22, 2008

COMP 131, Lecture 6

37

## Logical equivalence

- Two sentences are **logically equivalent** iff true in same models:  $\alpha \equiv \beta$  iff  $\alpha \models \beta$  and  $\beta \models \alpha$

$(\alpha \wedge \beta) \equiv (\beta \wedge \alpha)$  commutativity of  $\wedge$   
 $(\alpha \vee \beta) \equiv (\beta \vee \alpha)$  commutativity of  $\vee$   
 $((\alpha \wedge \beta) \wedge \gamma) \equiv (\alpha \wedge (\beta \wedge \gamma))$  associativity of  $\wedge$   
 $((\alpha \vee \beta) \vee \gamma) \equiv (\alpha \vee (\beta \vee \gamma))$  associativity of  $\vee$   
 $\neg(\neg\alpha) \equiv \alpha$  double-negation elimination  
 $(\alpha \Rightarrow \beta) \equiv (\neg\beta \Rightarrow \neg\alpha)$  contraposition  
 $(\alpha \Rightarrow \beta) \equiv (\neg\alpha \vee \beta)$  implication elimination  
 $(\alpha \Leftrightarrow \beta) \equiv ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha))$  biconditional elimination  
 $\neg(\alpha \wedge \beta) \equiv (\neg\alpha \vee \neg\beta)$  de Morgan  
 $\neg(\alpha \vee \beta) \equiv (\neg\alpha \wedge \neg\beta)$  de Morgan  
 $(\alpha \wedge (\beta \vee \gamma)) \equiv ((\alpha \wedge \beta) \vee (\alpha \wedge \gamma))$  distributivity of  $\wedge$  over  $\vee$   
 $(\alpha \vee (\beta \wedge \gamma)) \equiv ((\alpha \vee \beta) \wedge (\alpha \vee \gamma))$  distributivity of  $\vee$  over  $\wedge$

Sept 22, 2008

COMP 131, Lecture 6

38

## Validity and satisfiability

- A sentence is **valid** if it is true in **all** models,
  - e.g.,  $\text{True}$ ,  $A \vee \neg A$ ,  $A \Rightarrow A$ ,  $(A \wedge (A \Rightarrow B)) \Rightarrow B$
- Validity is connected to inference via the **Deduction Theorem**:
  - $KB \models \alpha$  if and only if  $(KB \Rightarrow \alpha)$  is valid
- A sentence is **satisfiable** if it is true in **some** model
  - e.g.,  $A \vee B$
- A sentence is **unsatisfiable** if it is true in **no** models
  - e.g.,  $A \wedge \neg A$
- Satisfiability is connected to inference via the following:
  - $KB \models \alpha$  if and only if  $(KB \wedge \neg\alpha)$  is unsatisfiable

Sept 22, 2008

COMP 131, Lecture 6

39

## Proof methods

- Proof methods divide into (roughly) two kinds
- Application of inference rules**
  - legitimate (sound) generation of new sentences from old
  - proof** = a sequence of inference rule applications
    - can use inference rules as operators in a standard search algorithm
    - typically require transformation of sentences into a **normal form**
- Model checking**
  - truth table enumeration (always exponential in  $n$ )
  - improved backtracking, e.g., Davis-Putnam-Logemann-Loveland (DPLL)
  - heuristic search in model space (sound but incomplete)
    - e.g., min-conflicts-like hill-climbing algorithms

Sept 22, 2008

COMP 131, Lecture 6

40

## Resolution: a single inference rule

**Conjunctive Normal Form (CNF)**  
 conjunction of disjunctions of literals  
 clauses

a KB can be a conjunction of clauses

E.g.,  $(A \vee \neg B) \wedge (B \vee \neg C \vee \neg D)$

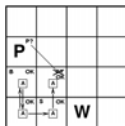
- Resolution inference rule (for CNF):**

$$\frac{\ell_i \vee \dots \vee \ell_{i-1} \vee \ell_{i+1} \vee \dots \vee \ell_k \vee m_1 \vee \dots \vee m_n}{\ell_i \vee \dots \vee \ell_{i-1} \vee \ell_{i+1} \vee \dots \vee \ell_k \vee m_1 \vee \dots \vee m_{j-1} \vee m_{j+1} \vee \dots \vee m_n}$$

where  $\ell_i$  and  $m_j$  are complementary literals.

E.g.,  $P_{1,3} \vee P_{2,2}, \neg P_{2,2}$   
 $P_{1,3}$

- Resolution is sound and complete for propositional logic



Sept 22, 2008

COMP 131, Lecture 6

41

## Resolution

Soundness of resolution inference rule:

$$\frac{\neg(\ell_i \vee \dots \vee \ell_{i-1} \vee \ell_{i+1} \vee \dots \vee \ell_k) \Rightarrow \ell_i}{\neg m_j \Rightarrow (m_1 \vee \dots \vee m_{j-1} \vee m_{j+1} \vee \dots \vee m_n)}$$

$$\neg(\ell_i \vee \dots \vee \ell_{i-1} \vee \ell_{i+1} \vee \dots \vee \ell_k) \Rightarrow (m_1 \vee \dots \vee m_{j-1} \vee m_{j+1} \vee \dots \vee m_n)$$

Sept 22, 2008

COMP 131, Lecture 6

42

## Conversion to CNF

$$B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})\beta$$

1. Eliminate  $\Leftrightarrow$ , replacing  $\alpha \Leftrightarrow \beta$  with  $(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)$ .  
 $(B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$
2. Eliminate  $\Rightarrow$ , replacing  $\alpha \Rightarrow \beta$  with  $\neg\alpha \vee \beta$ .  
 $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg(P_{1,2} \vee P_{2,1}) \vee B_{1,1})$
3. Move  $\neg$  inwards using de Morgan's rules and double-negation:  
 $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge ((\neg P_{1,2} \vee \neg P_{2,1}) \vee B_{1,1})$
4. Apply distributivity law ( $\wedge$  over  $\vee$ ) and flatten:  
 $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg P_{1,2} \vee B_{1,1}) \wedge (\neg P_{2,1} \vee B_{1,1})$

Sept 22, 2008

COMP 131, Lecture 6

43

## Resolution algorithm

- Proof by contradiction, i.e., show  $KB \wedge \neg\alpha$  unsatisfiable

```

function PL-RESOLUTION(KB,  $\alpha$ ) returns true or false
  clauses  $\leftarrow$  the set of clauses in the CNF representation of  $KB \wedge \neg\alpha$ 
  new  $\leftarrow$  { }
  loop do
    for each  $C_i, C_j$  in clauses do
      resolvents  $\leftarrow$  PL-RESOLVE( $C_i, C_j$ )
      if resolvents contains the empty clause then return true
      new  $\leftarrow$  new  $\cup$  resolvents
  if new  $\subseteq$  clauses then return false
  clauses  $\leftarrow$  clauses  $\cup$  new
  
```

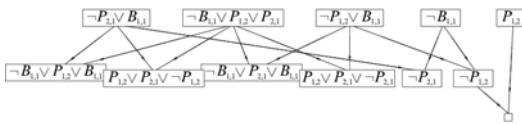
Sept 22, 2008

COMP 131, Lecture 6

44

## Resolution example

- $KB = (B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})) \wedge \neg B_{1,1} \wedge \neg P_{1,2}$



Sept 22, 2008

COMP 131, Lecture 6

45

## We'll pick up next time

Sept 22, 2008

COMP 131, Lecture 6

46