CS 1571 Introduction to AI
Lecture 9

Propositional logic.

Milos Hauskrecht
milos@cs.pitt.edu
5329 Sennott Square

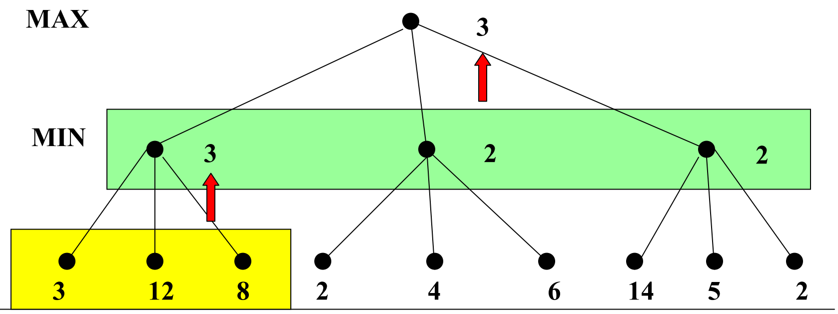
Administration

- **PS–3 due today** Report
  - Programs through ftp
- **PS-4 is out**
  - on the course web page
  - due next week on Tuesday, September 31, 2003

Minimax algorithm

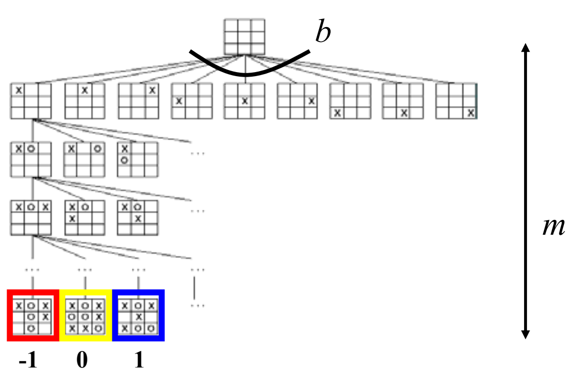
How to deal with the contingency problem?

- Assuming that the opponent is rational and always optimizes its behavior (opposite to us) we consider **the best opponent's response**
- Then the **minimax algorithm** determines the best move



Complexity of the minimax algorithm

- We need to explore the complete game tree before making the decision



Complexity:

$O(b^m)$

- Impossible for large games
  - Chess: 35 operators, game can have 50 or more moves

Solution to the complexity problem

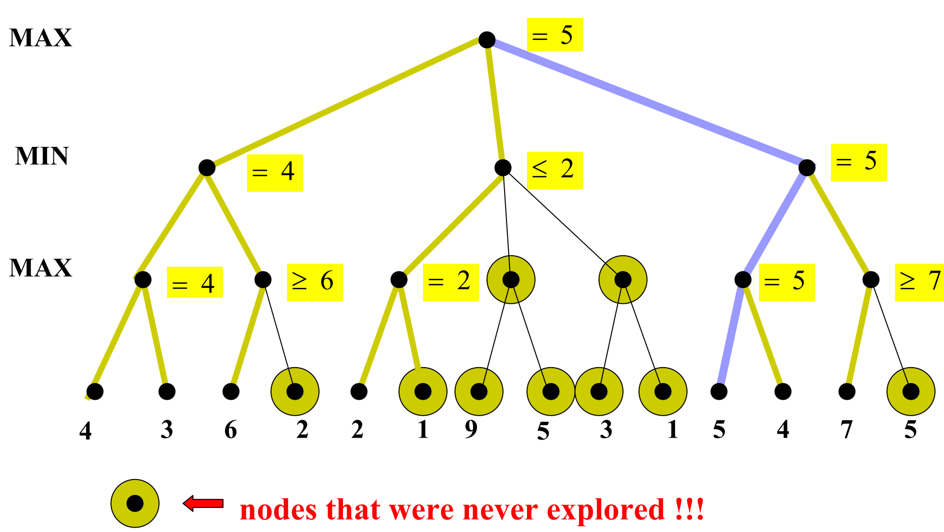
Two solutions:

1. **Dynamic pruning of redundant branches** of the search tree
   - identify provably suboptimal branch of the search tree even before it is fully explored
   - Eliminate the suboptimal branch

   Procedure: Alpha-Beta pruning

2. **Early cutoff of the search tree**
   - uses imperfect minimax value estimate of non-terminal states (positions)

Alpha beta pruning. Example



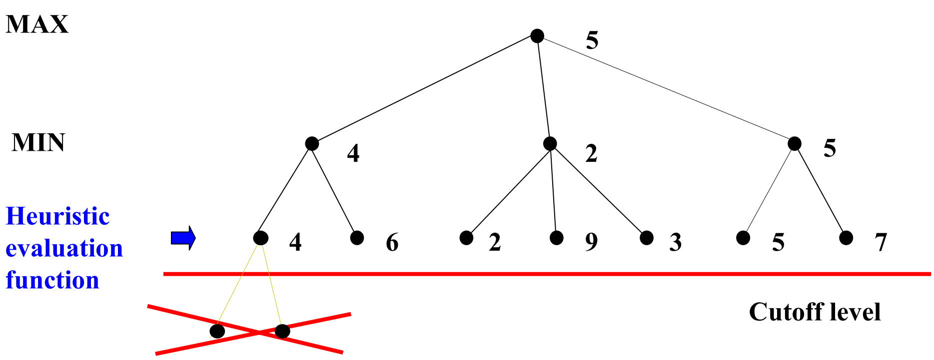
Alpha-Beta pruning

```
function MAX-VALUE(state, game, α, β) returns the minimax value of state
   inputs: state, current state in game
           game, game description
           α, the best score for MAX along the path to state
           β, the best score for MIN along the path to state

   if  GOAL-TEST(state) then return EVAL(state)
   for each s in SUCCESSORS(state) do
      α ← MAX(α, MIN-VALUE(s, game, α, β))
      if α ≥ β then return β
   end
   return α
```

```
function MIN-VALUE(state, game, α, β) returns the minimax value of state

   if  GOAL-TEST(state) then return EVAL(state)
   for each s in SUCCESSORS(state) do
      β ← MIN(β, MAX-VALUE(s, game, α, β))
      if β ≤ α then return α
   end
   return β
```


Using minimax value estimates

- **Idea:**
  - Cutoff the search tree before the terminal state is reached
  - Use imperfect estimate of the minimax value at the leaves
    - Evaluation function



Design of evaluation functions

- **Heuristic estimate** of the value for a sub-tree
- **Examples of a heuristic functions**:
  - **Material advantage in chess, checkers**
    - Gives a value to every piece on the board, its position and combines them
  - More general **feature-based evaluation function**
    - Typically a linear evaluation function:

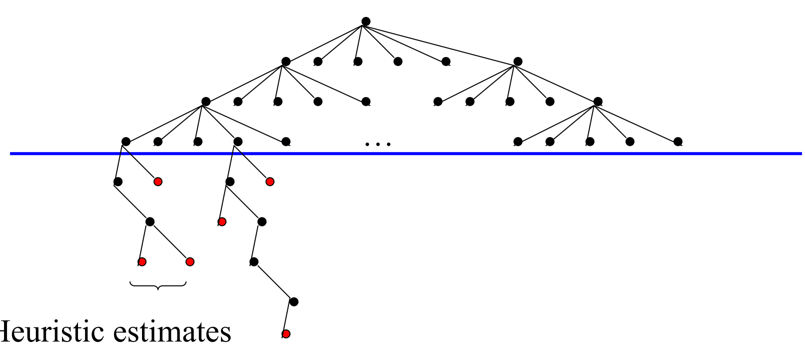
$$f(s) = f_1(s)w_1 + f_2(s)w_2 + \dots f_k(s)w_k$$

$f_i(s)$ - a feature of a state s

w_i - feature weight

Further extensions to real games

- Restricted set of moves to be considered under **the cutoff level** to reduce branching and improve the evaluation function
  - E.g., consider only the capture moves in chess



Heuristic estimates

Propositional logic

Knowledge-based agent

Knowledge base
Inference engine

- **Knowledge base (KB):**
  - A set of sentences that describe facts about the world in some formal (representational) language
  - **Domain specific**
- **Inference engine:**
  - A set of procedures that work upon the representational language and can infer new facts or answer KB queries
  - **Domain independent**

Example: MYCIN

- MYCIN: an expert system for diagnosis of bacterial infections
- **Knowledge base** represents
  - Facts about a specific patient case
  - Rules describing relations between entities in the bacterial infection domain

If	1. The stain of the organism is gram-positive, and
	2. The morphology of the organism is coccus, and
	3. The growth conformation of the organism is chains
Then	the identity of the organism is streptococcus

- **Inference engine**:
  - manipulates the facts and known relations to answer diagnostic queries (consistent with findings and rules)

Knowledge representation

- The **objective of knowledge representation** is to express the knowledge about the world in a computer-tractable form

- Key aspects of knowledge representation languages:
  - **Syntax:** describes how sentences are formed in the language
  - **Semantics:** describes the meaning of sentences, what is it the sentence refers to in the real world
  - **Computational aspect:** describes how sentences and objects are manipulated in concordance with semantic conventions

Many KB systems rely on some variant of logic

Logic

A formal language for expressing knowledge and ways of reasoning.

Logic is defined by**:**

- **A set of sentences**
  - A sentence is constructed from a set of primitives according to syntax rules.
- **A set of interpretations**
  - An interpretation gives a semantic to primitives. It associates primitives with values.
- **The valuation (meaning) function** V
  - Assigns a value (typically the truth value) to a given sentence under some interpretation

$$V : \text{sentence} \times \text{interpretation} \rightarrow \{True, False\}$$

Types of logic

- Different types of logic exist:
  - **Propositional logic**
  - First-order logic
  - Temporal logic
  - Numerical constraints logic
  - Map-coloring logic

In the following:

- **Propositional logic.**
  - Formal language for making symbolic inferences
  - Foundations of **propositional logic**: **George Boole** (1854)

Propositional logic. Syntax

Syntax:

- **Symbols (alphabet)** in P:
  - **Constants**: True, False
  - **A set of propositional variables** (propositional symbols):
    Examples: $P, Q, R, \ldots$ or statements like:
    Light in the room is on,
    It rains outside, etc.
  - **A set of logical connectives**:
    $\neg, \wedge, \vee, \Rightarrow, \Leftrightarrow$
- **Sentences**
  - **Build from symbols**

Propositional logic. Syntax

Sentences in the propositional logic:

- **Atomic sentences:**
  - **Constructed from constants and propositional symbols**
  - True, False are (atomic) sentences
  - P, Q or *Light in the room is on, It rains outside* are (atomic) sentences
- **Composite sentences:**
  - **Constructed from valid sentences via logical connectives**
  - If A, B are sentences then
    $\neg A \quad (A \wedge B) \quad (A \vee B) \quad (A \Rightarrow B) \quad (A \Leftrightarrow B)$
    or $(A \vee B) \wedge (A \vee \neg B)$
    are sentences

Propositional logic. Semantics.

The semantic gives the meaning to sentences.

In the propositional logic the semantics is defined by:

1. **Interpretation of propositional symbols and constants**
   – Semantics of atomic sentences
2. **Through the meaning of connectives**
   – Meaning (semantics) of composite sentences

Semantic: interpretations

A **propositional symbol** (an atomic sentence) can stand for an arbitrary fact (statement) about the world

Examples: "*Light in the room is on*",
"*It rains outside",* etc.

An **interpretation:**

- maps symbols to one of the two values: ***True (T),*** or ***False (F)***,
- the value depends on the world we want to describe

World 1:

I: *Light in the room is on ->* ***True,*** *It rains outside ->* ***False***

World 2:

I': *Light in the room is on ->* ***False,*** *It rains outside ->* ***False***

Semantics: symbols and constants

- The **meaning (truth)** of the propositional symbol for a **specific interpretation** is given by its interpretation

  V(Light in the room is on, **I***)* = ***True***
  V(Light in the room is on, **I'***)* = ***False***

- **The meaning (truth) of constants:**
  – ***True*** and ***False*** constants are always (under any interpretation) assigned the corresponding ***True,False*** value

V(True, **I***)* = ***True***

V(False, **I***)* = ***False***

} For any interpretation **I**

Semantics: composite sentences.

- **The meaning (truth value) of complex propositional sentences.**
  – Determined using the "standard" rules for combining logical sentences:

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

Some definitions

- A **model (in logic):** An interpretation is **a model for a set of sentences** if it assigns true to each sentence in the set.
- A sentence is **satisfiable** if it has a model;
  - There is at least one interpretation under which the sentence can evaluate to ***True***.
- A sentence is **valid** if it is ***True*** in all interpretations
  - i.e., if its negation is **not satisfiable** (leads to contradiction)

			Satisfiable sentence	Valid sentence
P	Q	$P \vee Q$	$(P \vee Q) \wedge \neg Q$	$((P \vee Q) \wedge \neg Q) \Rightarrow P$
True	***True***	***True***	***False***	***True***
True	***False***	***True***	***True***	***True***
False	***True***	***True***	***False***	***True***
False	***False***	***False***	***False***	***True***

Some definitions

- A **model (in logic):**
  - An interpretation is **a model for a set of sentences S** if it assigns true to each sentence in the set.
- **Example:**
  - **Assume two sentences:**

    $(P \vee Q) \wedge \neg Q$ and P
  - **Assume an interpretation:**

    I: $P \longrightarrow$ ***False***

    $Q \longrightarrow$ ***False***
  - **Is I a model for** $(P \vee Q) \wedge \neg Q$ and P **?**

Some definitions

- A **model (in logic):**
  - An interpretation is **a model for a set of sentences S** if it assigns true to each sentence in the set.
- **Example:**
  - **Assume two sentences:**

    $(P \vee Q) \wedge \neg Q$ and P
  - **Assume an interpretation:**

    I: $P \longrightarrow$ ***False***

    $Q \longrightarrow$ ***False***
  - **Is I a model for** $(P \vee Q) \wedge \neg Q$ and P **?**

    No !

Some definitions

- A **model (in logic):**
  - An interpretation is **a model for a set of sentences S** if it assigns true to each sentence in the set.
- **Example:**
  - **Assume two sentences:**

    $(P \vee Q) \wedge \neg Q$ and P
  - **Assume an interpretation:**

    I': $P \longrightarrow$ ***True***

    $Q \longrightarrow$ ***False***
  - **I' is a model for** $(P \vee Q) \wedge \neg Q$ and P **!!!**

Some definitions

- A sentence is **satisfiable** if it has a **model**;
  – There is **at least one interpretation** under which the sentence can evaluate to ***True***.
- A sentence is **valid** if it is ***True*** in all interpretations
  – i.e., if its negation is **not satisfiable** (leads to contradiction)

Example:

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

Some definitions

- A sentence is **satisfiable** if it has a **model**;
  – There is **at least one interpretation** under which the sentence can evaluate to ***True***.
- A sentence is **valid** if it is ***True*** in all interpretations
  – i.e., if its negation is **not satisfiable** (leads to contradiction)

Example:

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

Satisfiable, but not valid

Some definitions

- A sentence is **satisfiable** if it has a **model**;
  - There is **at least one interpretation** under which the sentence can evaluate to ***True***.
- A sentence is **valid** if it is ***True*** in all interpretations
  - i.e., if its negation is **not satisfiable** (leads to contradiction)

Example:

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

Some definitions

- A sentence is **satisfiable** if it has a **model**;
  - There is **at least one interpretation** under which the sentence can evaluate to ***True***.
- A sentence is **valid** if it is ***True*** in all interpretations
  - i.e., if its negation is **not satisfiable** (leads to contradiction)

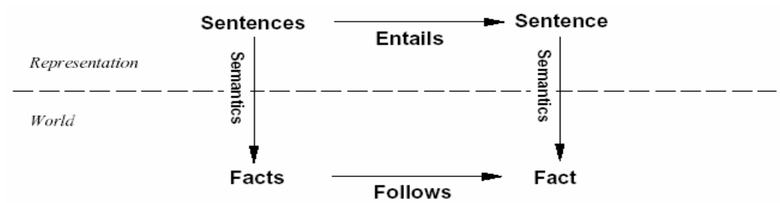
Example:

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

Satisfiable, and valid

Entailment

- **Entailment** reflects the relation of one fact in the world following from the others



- Entailment $KB \models \alpha$
- Knowledge base KB entails sentence α **if and only if** α is true in all worlds where KB is true

Inference.

- **Inference** is a process by which conclusions are reached.
- **Our goal:**
  - We want to implement the inference process on a computer !!!
- Assume an **inference procedure *i*** that
  - derives a sentence α from the KB : $KB \vdash_i \alpha$
- **Important:**
  - **We need to assure that our inference procedure derives correct conclusions**

Sound and complete inference.

Assume an **inference procedure *i*** that

- derives a sentence α from the KB : $KB \vdash_i \alpha$

Properties of the inference procedure:

- **Soundness:** An inference procedure is **sound**

  If $KB \vdash_i \alpha$ then it is true that $KB \models \alpha$

- **Completeness:** An inference procedure is **complete**

  If $KB \models \alpha$ then it is true that $KB \vdash_i \alpha$