Lecture 10 of 42

Introduction to Knowledge Representation and Logic

William H. Hsu
Department of Computing and Information Sciences, KSU

KSOL course page: http://snipurl.com/v9v3
Course web site: http://www.kddresearch.org/Courses/CIS730
Instructor home page: http://www.cis.ksu.edu/~bhsu

Reading for Next Class:

Lecture Outline

- Reading for Next Class: Sections 7.5 – 7.7 (p. 211 – 232), R&N 2e
  - Propositional calculus (aka propositional logic)
    - Syntax and semantics
    - Proof rules
    - Properties of sentences: entailment and provability
    - Properties of proof rules: soundness and completeness
  - Elements of logic: ontology and epistemology
- Last Class: Game Trees, Search Concluded
  - Minimax with alpha-beta (α - β) pruning
  - Expectiminimax: dealing with nondeterminism and imperfect information
  - “Averaging over clairvoyance” and when/why it fails
  - Quiescence and the horizon effect
- Today: Intro to KR and Logic, Sections 7.1 – 7.4 (p. 194 – 210), R&N 2e
  - Wumpus world and need for knowledge representation
  - Syntax and (possible worlds) semantics of logic
- Coming Week: Propositional and First-Order Logic (7.5 – 9.1)
Games:
Review

- Games
- Perfect play
  - minimax decisions
  - $\alpha$/$\beta$ pruning
- Resource limits and approximate evaluation
- Games of chance
- Games of imperfect information

Nondeterministic Games:
Review

In nondeterministic games, chance introduced by dice, card-shuffling

Simplified example with coin-flipping:

```
          MAX
            ▲
              ▼
                ▼
                ▼
                ▼
            0.5  0.5
            ▼  ▼
              ▼
              ▼
            0.5  0.5
            ▼  ▼
              ▼
              ▼
              ▼
            0.5  0.5
            ▼  ▼
              ▼
            0.5  0.5
            ▼  ▼
              ▼
              ▼
              ▼
            0.5  0.5
            ▼  ▼
              ▼
              ▼
              ▼
            0.5  0.5
            ▼  ▼
              ▼
              ▼
              ▼
            0.5  0.5
            ▼  ▼
              ▼
              ▼
              ▼
            0.5  0.5
            ▼  ▼
              ▼
              ▼
              ▼
            0.5  0.5
            ▼  ▼
              ▼
              ▼
              ▼
            0.5  0.5
            ▼  ▼
              ▼
              ▼
              ▼
            0.5  0.5
            ▼  ▼
              ▼
              ▼
              ▼
            0.5  0.5
            ▼  ▼
              ▼
              ▼
              ▼
            0.5  0.5
            ▼  ▼
              ▼
              ▼
              ▼
            0.5  0.5
            ▼  ▼
              ▼
              ▼
              ▼
            0.5  0.5
            ▼  ▼
              ▼
              ▼
              ▼
            0.5  0.5
            ▼  ▼
              ▼
              ▼
              ▼
            0.5  0.5
            ▼  ▼
              ▼
              ▼
              ▼
            0.5  0.5
            ▼  ▼
              ▼
              ▼
              ▼
            0.5  0.5
            ▼  ▼
              ▼
              ▼
              ▼
            0.5  0.5
            ▼  ▼
              ▼
              ▼
              ▼
            0.5  0.5
            ▼  ▼
              ▼
              ▼
              ▼
            0.5  0.5
            ▼  ▼
              ▼
              ▼
              ▼
            0.5  0.5
            ▼  ▼
              ▼
              ▼
              ▼
            0.5  0.5
            ▼  ▼
              ▼
              ▼
              ▼
```
Commonsense Example — Statement:

Review

Day 1
Road A leads to a small heap of gold pieces
Road B leads to a fork:
- take the left fork and you’ll find a mound of jewels;
- take the right fork and you’ll be run over by a bus.

Day 2
Road A leads to a small heap of gold pieces
Road B leads to a fork:
- take the left fork and you’ll be run over by a bus;
- take the right fork and you’ll find a mound of jewels.

Day 3
Road A leads to a small heap of gold pieces
Road B leads to a fork:
- guess correctly and you’ll find a mound of jewels;
- guess incorrectly and you’ll be run over by a bus.

Commonsense Example — Analysis:

Review

* Intuition that the value of an action is the average of its values in all actual states is WRONG

With partial observability, value of an action depends on the information state or belief state the agent is in

Can generate and search a tree of information states

Leads to rational behaviors such as
- Acting to obtain information
- Signalling to one’s partner
- Acting randomly to minimize information disclosure

Learning = Improving with Experience at Some Task
- Improve over task $T$,
- with respect to performance measure $P$,
- based on experience $E$.

Example: Learning to Play Checkers
- $T$: play games of checkers
- $P$: percent of games won in tournament
- $E$: opportunity to play against self

Refining the Problem Specification: Issues
- What experience?
- What exactly should be learned?
- How shall it be represented?
- What specific algorithm to learn it?

Defining the Problem Milieu
- Performance element: How shall the results of learning be applied?
- How shall performance element be evaluated? Learning system?

Type of Training Experience
- Direct or indirect?
- Teacher or not?
- Knowledge about the game (e.g., openings/endgames)?

Problem: Is Training Experience Representative (of Performance Goal)?

Software Design
- Assumptions of the learning system: legal move generator exists
- Software requirements: generator, evaluator(s), parametric target function

Choosing a Target Function
- $ChooseMove: Board \rightarrow Move$ (action selection function, or policy)
- $V: Board \rightarrow R$ (board evaluation function)
- Ideal target $V$; approximated target $\hat{V}$
- Goal of learning process: operational description (approximation) of $V$

Adapted from materials © 1997 T. M. Mitchell. Reused with permission.
Learning to Play Checkers: Representation of Evaluation Function

- **Possible Definition**
  - If $b$ is a final board state that is won, then $V(b) = 100$
  - If $b$ is a final board state that is lost, then $V(b) = -100$
  - If $b$ is a final board state that is drawn, then $V(b) = 0$
  - If $b$ is not a final board state in the game, then $V(b) = V(b')$ where $b'$ is best final board state according to Minimax (optimal play to end of game)
  - Correct values, but not operational

- **Choosing Representation for Target Function**
  - Collection of rules?
  - Neural network?
  - Polynomial function (e.g., linear, quadratic combination) of board features?
  - Other?

- **Representation for Learned Function**
  - $\hat{V}(b) = w_b + w_{bp}(b) + w_{br}(b) + w_{bk}(b) + w_{bk}'(b) + w_{bk}'(b) + w_{br}(b) + w_{br}(b)$
  - $bp/rp = \text{number of black/red pieces; bk/rk = number of black/red kings; bt/rt = number of black/red pieces threatened (can be taken next turn)}$

Obtaining Training Examples

- **One Rule For Estimating Training Values:**
  - $V_{\text{train}}(b) \leftarrow \hat{V}(\text{Successor}(b))$

Choose Weight Tuning Rule

- **Least Mean Square (LMS) weight update rule:**
  - REPEAT
    - Select a training example $b$ at random
    - Compute the $\text{error}(b)$ for this training example
    - For each board feature $f_i$, update weight $w_i$ as follows:
      $$\text{error}(b) = V_{\text{train}}(b) - \hat{V}(b)$$
      $$w_i \leftarrow w_i - c \cdot f_i \cdot \text{error}(b)$$
where $c$ is small, constant factor to adjust learning rate

Adapted from materials © 1997 T. M. Mitchell. Reused with permission.
Learning to Play Checkers: Design Choices

- **Determine Type of Training Experience**
  - Games against experts
  - Games against self
  - Table of correct moves

- **Determine Target Function**
  - Board → move
  - Board → value

- **Determine Representation of Learned Function**
  - Polynomial
  - Linear function of six features
  - Artificial neural network

- **Determine Learning Algorithm**
  - Gradient descent
  - Linear programming

- Completed Design

Section III: Knowledge and Reasoning

Adapted from materials © 1997 T. M. Mitchell. Reused with permission.
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 KB
- Agents can be viewed at the **knowledge level**
  - i.e., what they know, regardless of how implemented
- Or at the **implementation level**
  - i.e., data structures in KB and algorithms that manipulate them
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

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

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

Wumpus World [1]:
Peas Description

Adapted from slides © 2004 S. Russell & P. Norvig. Reused with permission.
Wumpus World \textit{Xtreme!}

- Fully Observable? 
  No – only \textit{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
Exploring Wumpus World

Other Safe Actions

Breeze in (1,2) and (2,1) ⇒ no safe actions
Assuming pits uniformly distributed, (2,2) has pit w/ prob 0.86, vs. 0.31

Smell in (1,1) ⇒ cannot move
Can use a strategy of coercion:
  shoot straight ahead
  wumpus was there ⇒ dead ⇒ safe
  wumpus wasn’t there ⇒ safe

Adapted from slides © 2004 S. Russell & P. Norvig. Reused with permission.
Logic in General

Logics are formal languages for representing information such that conclusions can be drawn.

Syntax defines the sentences in the language.

Semantics define the “meaning” of sentences; i.e., define truth of a sentence in a world.

E.g., the language of arithmetic:

- \( x + 2 \geq y \) is a sentence; \( x^2 + y > \) is not a sentence.
- \( x + 2 \geq y \) is true iff the number \( x + 2 \) is no less than the number \( y \).
- \( x + 2 \geq y \) is true in a world where \( x = 7, y = 1 \).
- \( x + 2 \geq y \) is false in a world where \( x = 0, y = 6 \).

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.

E.g., the KB containing “the Giants won” and “the Reds won” entails “Either the Giants won or the Reds won”.

E.g., \( x + y = 4 \) entails \( 4 = x + y \).

Entailment is a relationship between sentences (i.e., syntax) that is based on semantics.

Note: brains process syntax (of some sort).
**Models**

*(Possible Worlds Semantics)*

Logicians typically think in terms of models, which are formally structured worlds with respect to which truth can be evaluated.

We say \( m \) is a model of a sentence \( \alpha \) if \( \alpha \) is true in \( m \).

\( M(\alpha) \) is the set of all models of \( \alpha \).

Then \( KB \models \alpha \) if and only if \( M(KB) \subseteq M(\alpha) \).

E.g., \( KB = \) Giants won and Reds won

\( \alpha = \) Giants won

---

**Entailment in Wumpus World**

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

Consider possible models for ?s assuming only pits.

3 Boolean choices \( \Rightarrow \) 8 possible models.
Wumpus Models [1]

Wumpus Models [2]

\[ KB = \text{wumpus-world rules + observations} \]
**TERMINOLOGY**

- **Game Trees**
  - *Expectiminimax*: Minimax with alpha-beta ($\alpha - \beta$) pruning and *chance nodes*
  - "Averaging over clairvoyance": expectation applied to hidden info
  - *Quiescence*: state of "calmness" in play

- **PEAS** *(Performance, Environment, Actuators, Sensors)* Specifications

- **Wumpus World**: Toy Domain

- **Intro to Knowledge Representation (KR) and Logic**
  - *Logic*
    - Formal language for representing information
    - Supports reasoning and learning
  - *Sentences*: units of logic

- **Models**: Interpretation *(Denotation, Meaning)* of Sentences
  - *Possible worlds semantics*: assigns sets of models to all sentences
  - *Entailment*: all models of left-hand side (LHS) are models of right (RHS)

- **Next**: *Propositional Logic*

---

**SUMMARY POINTS**

- **Last Class**: Game Trees, Search Concluded
  - Minimax with alpha-beta ($\alpha - \beta$) pruning
  - *Expectiminimax*: dealing with nondeterminism and imperfect information
  - "Averaging over clairvoyance" and when/why it fails
  - *Quiescence* and the horizon effect

- **Today**: Intro to Knowledge Representation (KR) and Logic
  - Logic as formal language
  - Representation: foundation of reasoning and learning
  - Logical entailment

- **Wumpus World**: PEAS Specification and Logical Description

- **Reasoning Examples**

- **Possible Worlds Semantics**: Models and Meaning

- **Next Week**: Propositional and First-Order Logic
  - Propositional logic
  - Resolution theorem proving in propositional logic
  - First-order predicate calculus *(FOPC)* aka first order logic *(FOL)*