Lecture Overview

- Agent types
- Making all agents learning agents
- First small homework
- I briefly introduce A.I. techniques, I cover them more slowly later in the course.
- Next lecture. Ch3: Intelligent search
- Stupidest agent picture
Performance Measures and Rationality

• Consider a performance measure for:
  – Vacuum world
  – Email harvester
  – Chess
  – Immediate reward versus deferred reward
    • Credit allocation problem

• A rational agent for a given percept sequence maximizes its performance measure.

• This is our definition of A.I., not that the agent thinks/reasons like a human.
Environment Properties - 1

• Variation in these properties are why there are many A.I. techniques

• Fully versus partially observable
  – Agent perceives all information required to make the optimal action. Does not refer to violation of modeling assumptions.
  – Noisy sensors can make an environment partially observable

• Deterministic versus stochastic
  – From the view of the agent
  – Partially observable, stochastic environment = uncertainty
Environment Properties - 2

• Episodal versus Sequential
  – Most interesting environments are sequential
  – Each episode consists of the agent performing one action. If episodes are independent then episodal. ie \( P^*(\text{Action}^t | P^t, P^{t-1}, P^1) = P^*(\text{Action}^t | P^t) \)
Environment Properties - 3

• Static versus Dynamic
  – Semi-dynamic

• Discrete versus continuous
  – Applies to actions, states

• Single agent versus multiple agents
  – Co-operative and competitive agents
Environment Types

Deterministic, accessible $\rightarrow$ single-state problem
Deterministic, inaccessible $\rightarrow$ multiple-state problem

Nondeterministic, inaccessible $\rightarrow$ contingency problem
must use sensors during execution
solution is a tree or policy
often interleave search, execution

Unknown state space $\rightarrow$ exploration problem (“online”)

Each problem type motivates a set of A.I. techniques.
n=8 Queen Problem

Single or multiple state, how to formulate the problem-state space
What algorithm? What information are we not using?
K-Queens Problem as CSP

Assume one queen in each column. Which row does each one go in?

**Variables** $Q_1, Q_2, Q_3, Q_4$

**Domains** $D_i = \{1, 2, 3, 4\}$

**Constraints**

- $Q_i \neq Q_j$ (cannot be in same row)
- $|Q_i - Q_j| \neq |i - j|$ (or same diagonal)

Translate each constraint into set of allowable values for its variables

E.g., values for $(Q_1, Q_2)$ are $(1, 3)$ $(1, 4)$ $(2, 4)$ $(3, 1)$ $(4, 1)$ $(4, 2)$
One or Multiple States Formulation?

- Chess?
- Monopoly?
- Robocup simulation league?
- When do we need multiple states?
- How do we determine the optimal action?
Reflex Agents

- Table driven agent
  - If agent lifetime is $T$ and $n$ possible percepts, table size?
- Write rational agents (?) whose space complexity is far less than the above table
- Reflex agents
  - Memory-less

Each component requires intelligence

```python
function SIMPLE-REFLEX-AGENT(percept) returns action
    static: rules, a set of condition-action rules
    state ← INTERPRET-INPUT(percept)
    rule ← RULE-MATCH(state, rules)
    action ← RULE-ACTION[rule]
    return action
```
Model-based Reflex

• What are we modeling? Why? Why?

```plaintext
function REFLEX-AGENT-WITH-STATE(percept) returns action
static: state, a description of the current world state
rules, a set of condition-action rules

state ← UPDATE-STATE(state, percept)
rule ← RULE-MATCH(state, rules)
action ← RULE-ACTION[rule]
state ← UPDATE-STATE(state, action)
return action
```
Competitive Agents

- Zero sum games

Perfect play for deterministic, perfect-information games

Idea: choose move to position with highest \( \text{minimax value} \) = best achievable payoff against best play

E.g., 2-ply game:

- Rock, scissors, paper example
- How to generalize for non-zero sum games, essentially setting performance measure
An Example: The Taxi Driver Agent

Percepts: input from cameras, speedometer, GPS, sonar, microphone, ...

Actions: steer, accelerate, brake, talk to passenger, horn, ...

Goals: (a) make trip safe, fast, legal, and comfortable; (b) maximize profits.

Environment: roads, other traffic, pedestrians, customers, weather, ...

Performance Measure: Based on one or more of the following goals:

- get to correct destination;
- minimize fuel consumption, wear and tear;
- minimize trip time;
- maximize passenger safety and comfort;
- maximize profits;
- ...
A Reflex Taxi-Driver Agent

- We cannot implement it as a table-lookup: the percepts are too complex.

- But we can abstract some portions of the table by coding common input/output associations.

- We do this with a list of condition/action rules:
  - if car-in-front-is-braking then brake
  - if light-becomes-green then move-forward
  - if intersection-has-stop-sign then stop
Reflex Agent with State

- Often, the agent must remember some of its percepts to take an action.
  \textit{Ex: car in front signals it is turning left.}

- It must also remember which actions it has taken.
  \textit{Ex: loaded/unloaded passenger.}

- In jargon, it must have internal \texttt{state}.

To update its state the agent needs two kinds of \texttt{knowledge}:

1. how the world evolves independently from the agent;
   \textit{Ex: an overtaking car gets closer with time.}

2. how the world is affected by the agent’s actions.
   \textit{Ex: if I turn left, what was to my right is now behind me.}
Goal-based

- Limitation of previous agents
- Typically binary tests
A Goal-based Taxi-Driver Agent

- Knowing about the world is not always enough to decide what to do.
  
  *Ex: what direction do I take at an intersection?*

- The agent needs **goal** information.
  
  *Ex: passenger’s destination*

- Combining goal information with the knowledge of its actions, the agent can choose those actions that will achieve the goal.

- A new kind of decision-making is required (“what-if reasoning”).

- **Search** and **Planning** are devoted to find action sequences that achieve an agent’s goal.
Utility-based

- Can be considered as a continuous goal
- Allows trade-off between competing goals
Utility-based Taxi-Driver Agent

- There may be many ways to get to a destination but some may be better than others.
  
  *Ex: this way is faster/cheaper/more comfortable/*...*

- A particular configuration of the world, a world state, can be assigned a utility value for the agent.

- A sequence of actions is preferred if it leads to a goal state with higher utility value.
Simple Navigation Problem

- Agent percepts: is there 1 metre infront/behind/right/left of me (the agent)
- Agent actions: Forward, back, left, right
- Performance measure: get to the door quickly
- Reflex agent design (states, rules)
- Model based reflex agent design (what to abstract, state, rules)
- Goal agent design (what’s the goal, states, rules)
- Utility agent design (need to change agent to get to door “quickly”)
Learning Agents

• If you have a well understood non-changing environment with time to hand-craft agents …

• Navigation problem
Supervised Learning

Functional Overview – Learning Stage

Examples

<table>
<thead>
<tr>
<th></th>
<th>+</th>
<th>+</th>
<th>-</th>
</tr>
</thead>
<tbody>
<tr>
<td>Case A</td>
<td>30, …, $110K, Default</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Case B</td>
<td>50, …, $110K, NoDefault</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Case C</td>
<td>45, …, $90K, NoDefault</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Case A</td>
<td>32, …, $105K, Default</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Case B</td>
<td>49, …, $82K, NoDefault</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Case C</td>
<td>29, …, $50K, NoDefault</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Age, … income, \{Default | NoDefault\}
Supervised Learning

Functional Overview – Application Stage

Unlabeled Examples

Age, … income, \{**Default | NoDefault**\}
Case zx. 29, …, $113K, ?
Case zy. 42, …, $81K, ?
Case zz. 41, …, $92K, ?

Predictions

zx, Default
zy. NoDefault
zz. NoDefault
Homework #1

• Due next Monday start of class. Hand to TA (Ke Yin)
• Think of a situation where an intelligent agent would be useful
• Map the situation to the PEAS breakdown
• List the six environmental properties
• What type of agent would suffice? Why?
• The term project is to design an intelligent agent. I give some predefined term projects (Rubiks cube, Robocup sim.) but you can choose your own with my approval.