CS3243: Introduction to Artificial Intelligence

Semester 2, 2017/2018
Teaching Staff

• Lecturer: Yair Zick

  Email: zick@comp.nus.edu.sg
  Website: http://www.comp.nus.edu.sg/~zick
  Office: COM2-02-60

  Consultation hours: By appointment

Research: Algorithmic Game Theory, Computational Fair Division, Algorithmic Fairness/Accountability/Transparency
Teaching Staff

- TAs:
  - Ma Xiao (e0204987@u.nus.edu)
  - Nguyen Ta-Duy (a0112066@u.nus.edu)
  - Sakryukin Andrey (e0146324@u.nus.edu)
  - Strobel Martin (e0267912@u.nus.edu)
  - Wang Danding (e0028729@u.nus.edu)

Consultation hours: By appointment
Teaching Resources: IVLE

http://ivle.nus.edu.sg/

- Lesson Plan
- Lectures, Tutorials, Supplementary Materials, Homework
- Discussion forum
  - Any questions related to the course should be raised on this forum
  - Emails to me will be considered public unless otherwise specified
- Announcements
- Homework submissions
- Webcasts
A ‘Tasting Menu’ of AI

<table>
<thead>
<tr>
<th>Foundational concepts of AI</th>
<th>Who?</th>
</tr>
</thead>
<tbody>
<tr>
<td>• search</td>
<td>• Undergraduates</td>
</tr>
<tr>
<td>• game playing</td>
<td>• beginning graduate students.</td>
</tr>
<tr>
<td>• logic</td>
<td>• CS orientation, or by permission.</td>
</tr>
<tr>
<td>• uncertainty</td>
<td></td>
</tr>
<tr>
<td>• probabilistic reasoning</td>
<td></td>
</tr>
<tr>
<td>• machine learning.</td>
<td></td>
</tr>
</tbody>
</table>
Beyond CS3243

Machine Learning
- CS3244
- CS5242
- CS5339
- CS5340
- CS5344

Search & Planning
- CS4246
- CS5338, TBA

Logic
- CS4248
- CS6207
- CS4244

... And more!
- CS4261, TBA
- CS6208
- CS6281
Readings

**Textbook:**


- We will not cover entire book! But it makes for an interesting read...
Syllabus

• Introduction and Agents (chapters 1, 2)
• Search (chapters 3, 4, 5, 6)
• Logic (chapters 7, 8, 9)
• Uncertainty (chapters 13, 14)
• Machine Learning (chapter 18)
## Assessment Overview

<table>
<thead>
<tr>
<th>What</th>
<th>When</th>
<th>Grade Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Midterm Exam (during lecture, NO make-up)</td>
<td>5 March 2018</td>
<td>20%</td>
</tr>
<tr>
<td>Final Exam</td>
<td>9 May 2018 (afternoon)</td>
<td>50%</td>
</tr>
<tr>
<td>Term Project</td>
<td>TBA</td>
<td>25%</td>
</tr>
<tr>
<td>Tutorials + Attendance</td>
<td>-</td>
<td>5%</td>
</tr>
</tbody>
</table>
Freedom of Information Rule

• Collaboration is acceptable and encouraged

• You must always write the name(s) of your collaborators on your assignment.

• You will be assessed for the parts for which you claim is your own contribution.
On Collaboration

• You are free to meet with fellow student(s) and discuss assignments.

• Writing on a board or shared piece of paper is acceptable during the meeting; however, you **may not take any written (electronic or otherwise) record away from the meeting.**

• Do not solve assignment immediately after discussion; wait a while, **ensure you can reconstruct solution by yourself!**
Introduction
AIMA Chapter 1
What is AI?
Is this AI?

- translate complex sentences in most common languages
- beat human players in Go, chess and poker
- answer simple spoken queries, hold simple conversations
- retrieve relevant queries instantly
- navigate through disaster zone, find injured persons and call for help.
- Recognize images of dogs and cats
- Fold laundry and clean the house
- Diagnose disease
Think like a human
Act like a human
Think rationally
Act rationally
Philosophy
• Ethics
• Logic
• Learning
• Rationality
• Theory of the Mind

Computer Science
• Theory of Computing
• Hardware
• Control Theory
• Dynamic Systems

Mathematics
• Formal representation
• Probability
• Statistics

Economics
• Game theory
• Decision theory
• Fair Division
• Utility theory

Psychology
• Perception and motor control
• Experiments

Linguistics
• Knowledge representation
• Grammar
Abridged History of AI

1943
McCulloch & Pitts: Boolean circuit model of brain

1950
Turing’s “Computing Machinery and Intelligence”

1950s
Early AI programs,

1956
Dartmouth meeting: the term “Artificial Intelligence” is adopted

1952–69
Look, Ma, no hands! “A computer could never do X…” Show solution to X.
Abridged History of AI

1965
- Robinson's complete algorithm for logical reasoning

1966–73
- AI discovers computational complexity
- Neural network research nearly disappears

1969–79
- Early knowledge-based systems
Abridged History of AI

1980– AI becomes an industry

1986– Neural networks return to popularity

1987– AI becomes a science

1995– The emergence of intelligent agents

2008– Widespread use of deep neural networks
AI is Getting Better at Gameplay

“A computer once beat me at chess, but it was no match for me in kickboxing” – Emo Philips

<table>
<thead>
<tr>
<th>Year</th>
<th>Game</th>
<th>Program</th>
<th>Developer</th>
<th>Techniques</th>
</tr>
</thead>
<tbody>
<tr>
<td>1994</td>
<td>Checkers</td>
<td>Chinook</td>
<td>U. Alberta</td>
<td>Rule Based + search</td>
</tr>
<tr>
<td>1997</td>
<td>Chess</td>
<td>Deep Blue</td>
<td>IBM</td>
<td>Search + randomization</td>
</tr>
<tr>
<td>2008</td>
<td>Limit Texas Hold’em</td>
<td>Polaris (Cepheus 2015)</td>
<td>U. Alberta</td>
<td>Agent based modeling, game theory</td>
</tr>
<tr>
<td>2011</td>
<td>Jeopardy</td>
<td>Watson</td>
<td>IBM</td>
<td>NLP, Information retrieval, data analytics</td>
</tr>
<tr>
<td>2015</td>
<td>No Limit Texas Hold’em</td>
<td>Claudico (later Libratus)</td>
<td>Carnegie Mellon Univ.</td>
<td>Game Theory, Reinforcement Learning</td>
</tr>
<tr>
<td>2016</td>
<td>Atari Games</td>
<td>DeepMind</td>
<td>Google</td>
<td>Deep Learning</td>
</tr>
<tr>
<td>2016</td>
<td>Go</td>
<td>AlphaGo</td>
<td>Google</td>
<td>Deep Learning, search</td>
</tr>
</tbody>
</table>
AI is Getting Better at Gameplay

Deepmind + Blizzard released an API for designing AI playing SC II: fun idea for FYP!
Acting Humanly: Turing Test

• Turing (1950). Computing Machinery and Intelligence: “Can machines think?” → “Can machines behave intelligently?”

• Operational test for intelligent behavior: The Imitation Game
Human Thinking: Cognitive Modeling

• 1960s “cognitive revolution”: information-processing psychology (materialistic view of the mind)

• How does the brain process information?
  • Validation? Requires
    (1) Predicting and testing behavior of human subjects, or
    (2) Direct identification from neurological data

• Both approaches (roughly, Cognitive Science and Cognitive Neuroscience) are now distinct from AI
Rational Thought: “Laws of Thought”

• Aristotle: how do we correctly argue/logically think (precursor to mathematical reasoning)

• Problems:
  • Can all intelligent behavior can be captured by logical rules?
  • A logical solution in principle does not translate to practice – computational issues!
Acting Rationally: Rational Agent

• **Rational** behavior: doing the “right thing”

• What is the “right thing” to do? Expected to achieve best outcome
  
  • Best for whom?
  • What are we optimizing?
  • What information is available?
  • Unintended effects

• Break through wall to get a cup of coffee
  
  • Prescribe high doses of opiates to depressed patient
  • Kill human who tries to deactivate robot
Rational Agents

• An **agent** is an entity that perceives and acts

• **This course:** designing rational agents

• Abstractly, an agent is a function from percept histories to actions, i.e., \( f : P^* \rightarrow A \)

• We seek the best-performing agent for a certain task; must consider computation limits!

  design best **program**

  given resources
Intelligent Agents

AIMA Chapter 2
Agents

• Anything that can be viewed as **perceiving** its **environment through sensors**; **acting** upon that environment through **actuators**

• Human agent: eyes, ears, skin etc. are sensors; hands, legs, mouth, and other body parts are actuators

• Robotic agent: cameras and laser range finders for sensors; various motors for actuators
• The agent function maps from percept histories/sequences to actions, i.e., $f : P^* \rightarrow A$

• The agent program runs on the physical architecture to perform $f$

agent = architecture + program
Vacuum-Cleaner World

- Perceps: location and status, e.g., \([A, \text{Dirty}]\)
- Actions: Left, Right, Suck, NoOp
## Vacuum-Cleaner Agent Function

<table>
<thead>
<tr>
<th>Percept Sequence</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>[A, Clean]</td>
<td>Right</td>
</tr>
<tr>
<td>[A, Dirty]</td>
<td>Suck</td>
</tr>
<tr>
<td>[B, Clean]</td>
<td>Left</td>
</tr>
<tr>
<td>[B, Dirty]</td>
<td>Suck</td>
</tr>
<tr>
<td>[A, Clean], [A, Clean]</td>
<td>Right</td>
</tr>
<tr>
<td>[A, Clean], [A, Dirty]</td>
<td>Suck</td>
</tr>
</tbody>
</table>
Rational Agents

- An agent should strive to “do the right thing”, based on what it can perceive and the actions it can perform. The right action: maximize agent success.

- **Performance measure**: objective criterion for measuring success of an agent's behavior

- **Vacuum-cleaner agent:**
  - amount of dirt cleaned
  - time taken
  - electricity consumed
  - noise generated

Perhaps a bit of everything?
Rational Agents

• Rational Agent:
  • For each possible percept sequence, select an action that is expected to maximize its performance measure...
  • given the evidence provided by the percept sequence and whatever built-in knowledge the agent has.
Rational Agents

• Rationality ≠ omniscience (all-knowing with infinite knowledge)

• Agents can perform actions that help them gather useful information (exploration)

• An agent is autonomous if its behavior is determined by its own experience (with ability to learn and adapt)
Specifying Task Environment: PEAS

- PEAS: Performance measure, Environment, Actuators, Sensors
- Must first specify the setting for intelligent agent design
- Consider, e.g., the task of designing an automated taxi driver:
  - Performance measure
  - Environment
  - Actuators
  - Sensors
Specifying Task Environment: PEAS Automated Taxi

Performance Measure
- Safe
- Fast
- Legal
- Comfort
- Revenue

Environment
- Roads
- Other traffic
- Pedestrians
- Customers

Actuators
- Steering wheel
- Accelerator
- Brake
- Signal
- Horn

Sensors
- Camera
- Sonar
- Speedometer
- GPS
- Engine sensors
Specifying Task Environment: PEAS
Part Picking Robot

Performance Measure
- % parts in correct bins

Environment
- Conveyor belt
- parts
- bins

Actuators
- Jointed arm
- hand

Sensors
- Camera
- joint angle sensors
Specifying Task Environment: PEAS
Medical Diagnosis System

Performance measure
- Healthy patient
- cost
- lawsuits

Environment
- Patient
- hospital
- staff

Actuators
- Screen display (questions, tests, diagnoses, treatments, referrals)

Sensors
- Keyboard
- Medical Readings
- Medical History
Specifying Task Environment: PEAS
Interactive English Tutor

**Performance measure**
- Student's score on test

**Environment**
- Set of students
- Testing agency
- Chat platform

**Actuators**
- Screen display (exercises, suggestions, corrections)

**Sensors**
- Keyboard entry
Properties of Task Environments

Fully observable (vs. partially observable):
- Sensors provide access to the complete state of the environment at each point in time.

Deterministic (vs. stochastic)
- The next state of the environment is completely determined by the current state and the action executed by the agent.

Episodic (vs. sequential)
- The agent’s experience is divided into atomic “episodes” (each episode consists of the agent perceiving and then performing a single action)
- The choice of action in each episode does not depend on actions in past episodes.
Properties of Task Environments

- **Static (vs. dynamic)**
  - The environment is unchanged while an agent is deliberating.

- **Discrete (vs. continuous)**
  - A finite number of distinct states, percepts, and actions.

- **Single agent (vs. multi-agent)**
  - An agent operating by itself in an environment.
## Properties of Task Environments

<table>
<thead>
<tr>
<th>Task Environment</th>
<th>Crossword puzzle</th>
<th>Part-picking robot</th>
<th>Taxi driving</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fully observable</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Deterministic</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Episodic</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Static</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Discrete</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Single agent</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

Properties of task environment largely determine agent design. The real world is (naturally) partially observable, stochastic, sequential, dynamic, continuous, multi-agent.
The Boston Dynamics Robots
Agent Functions and Programs

• An agent is completely specified by the agent function mapping percept sequences to actions

• One agent function (or a small equivalence class) is rational

• Aim: Find a way to implement the rational agent function concisely
Table-Lookup Agent

function TABLE-DRIVEN-AGENT(percept) returns action
  static: percepts, a sequence, initially empty
table, a table of actions, indexed by percept sequences, fully specified

append percept to the end of percepts
action ← LOOKUP(percepts, table)
return action

• Drawbacks:
  • Huge table to store
  • Take a long time to build the table
  • No autonomy: impossible to learn all correct table entries from experience
  • No guidance on filling in the correct table entries
function Reflex-Vacuum-Agent([location, status]) returns an action
    if status = Dirty then return Suck
    else if location = A then return Right
    else if location = B then return Left
Agent Types

• Four basic types in order of increasing generality:
  • Simple reflex agent
  • Model-based reflex agent
  • Goal-based agent
  • Utility-based agent
Simple Reflex Agent

Agent

Condition-action rules

Sensors

What the world is like now

What action I should do now

Actuators
Model-Based Reflex Agent

- **Agent**
  - State
  - How the world evolves
  - What my actions do
  - Condition-action rules
  - What the world is like now
  - What action I should do now

- **Environment**
  - Sensors
  - Actuators
Goal-Based Agent

Agent

State

Goals

What the world is like now

What my actions do

How the world evolves

What it will be like if I do action $A$

What action I should do now

Sensors

Actuators

Environment
Utility-Based Agent

- How the world evolves
- What my actions do
- What the world is like now
- What it will be like if I do action A
- How happy I will be in such a state
- What action I should do now
Learning Agent

Performance standard

Agent

Critic

feedback

learning goals

Problem generator

changes

knowledge

Sensors

Performance element

Actuators

Environment

Learning element
Learning Agent

- Performance element: selects the external actions
- Learning element: improves agent to perform better
- Critic: provides feedback on how well the agent is doing
- Problem generator: suggests explorative actions that will lead to new, informative (but not necessarily better) experiences
Exploitation vs. Exploration

• An agent operating in the real world must often choose between:
  • maximizing its expected utility according to its current knowledge about the world; and
  • trying to learn more about the world since this may improve its future gains.

• This problem is known as the trade-off between \textit{exploitation} and \textit{exploration}.