CS 188: Artificial Intelligence

Introduction

Instructor: Angela Liu, Yanlai Yang
University of California, Berkeley
(slides adapted from Dan Klein, Pieter Abbeel, Stuart Russell, Anca Dragan, et al)
Sci-Fi AI?
IBM's Watson Jeopardy Computer Shuts Down Humans in Final Game

DAILY NEWS 9 March 2016

Sili

‘I’m in shock!’ How world’s best human

Blizzard will show off Google's Deepmind AI in StarCraft 2 later this week

By Andy Chalk 4 hours ago

Google and Blizzard launched the artificial intelligence project in 2016.
CS 188: Artificial Intelligence

Logistics

Instructors: Angela Liu, Yanlai Yang
University of California, Berkeley

(slides adapted from Dan Klein, Pieter Abbeel, Anca Dragan, et al)
# Course Staff

## Instructors

<table>
<thead>
<tr>
<th>Instructor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Angela Liu</td>
</tr>
<tr>
<td>Yanlai Yang</td>
</tr>
<tr>
<td>Arvind Rajaraman</td>
</tr>
<tr>
<td>Zhuang Liu</td>
</tr>
<tr>
<td>Neil Thomas</td>
</tr>
<tr>
<td>Sid Iju</td>
</tr>
<tr>
<td>Joy Liu</td>
</tr>
</tbody>
</table>

## GSIs

<table>
<thead>
<tr>
<th>GSI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cham Yao</td>
</tr>
<tr>
<td>Andrew Wang</td>
</tr>
<tr>
<td>Perry Dong</td>
</tr>
<tr>
<td>Weijia Zeng</td>
</tr>
<tr>
<td>Xiangwei Kong</td>
</tr>
</tbody>
</table>
Course Information

- **Website**: [http://inst.eecs.berkeley.edu/~cs188/su22/](http://inst.eecs.berkeley.edu/~cs188/su22/)
  - Tentative schedule, lecture slides and notes, course policies, etc.

- **Piazza**: [https://piazza.com/berkeley/summer2022/cs188](https://piazza.com/berkeley/summer2022/cs188)
  - Announcements, Q&A for any course-related questions
  - Private posts: personal questions

- **Gradescope**: [https://www.gradescope.com/courses/399015](https://www.gradescope.com/courses/399015), entry code **DJ55BE**
  - All assignment submissions and grades

- **Staff email**: [cs188@berkeley.edu](mailto:cs188@berkeley.edu)
  - Personal questions for the head TAs and Instructors
  - Individual staff emails can also be found on the website
Prerequisites

- Programming experience - CS 61A and CS 61B
  - Coding/debugging Python, data structures, search strategies (BFS/DFS/A*)
  - Project 0 – Python setup + short coding diagnostic

- Math – CS 70
  - Discrete probability, counting, derivatives
  - Homework 0 – probability diagnostic questions

Project 0 and Homework 0 are not graded. Due dates are a recommendation and you will not be penalized for submitting late (or not at all).

However, we do highly recommend that you complete them!
Course Format – Direct Instructions

- Lectures
  - Live lecture in Lewis 100; lecture recording will be posted afterwards

- Discussion Sections
  - Twice a week: either M+W (regular) or T+Th (exam prep)
  - Discussion recordings and solutions will be posted after the last section

- Office Hours
  - 1-2 TAs helping with course-related conceptual, hw, or proj questions
  - Hybrid but *in-person students will be prioritized on the queue*

- Homework/Project Parties
  - Work with other students and TAs on the current hw or proj!
Assignments:
- Electronic Homework (10%)
- Written Homework (10%)
- Projects (25%)

Exams:
- Midterm (20%)
- Final (35%)

This course is not curved, grade bins on the website.
Course Format - Assessments

- Midterm: 20%, Final 35%

- Times
  - Midterm: Friday 07/15, 7-9 PM Pacific Time
  - Final: Friday 08/12, 7-10 PM Pacific Time

No alternative times will be provided.

- In-person by default. Remote exam requests will be approved on a case-by-case basis for extraneous situations. (More info provided a week before the exam.)
Some Historical Statistics

- Homework and projects: work alone/together, iterate/learn till you nailed it

- Exams: assessment
Course Principles

- Diversity Statement
- A Note on Student Wellness
- Academic Integrity
Course Principles

- Diversity Statement
  - We believe in the crucial importance of creating a learning environment that is welcoming and respectful to students of all backgrounds.
  - We expect all students to understand that not everyone comes from the same background, and no one should make unwarranted assumptions about anyone based on stereotypes. Mutual understanding and respect is extremely important towards building an inclusive atmosphere.
A Note on Student Wellness

- We understand that computer science courses at UC Berkeley are rigorous and mentally demanding, and CS 188 is no exception.
- If you ever feel overwhelmed by the course, please email the instructors directly, and we will set up a one-on-one meeting with you to identify your options.
- If you would like to seek professional help, you may find a list of resources here: https://sa.berkeley.edu/conduct/resources/wellness

Extensions Policy

- All <= 3 day extension requests for any assignment will be automatically granted, no questions asked.
- For extensions of > 3 days, please email cs188@berkeley.edu
Course Principles

- **Academic Integrity Policy**
  - No consultation or collaboration for Exams
  - You should acknowledge all collaborators and sources on Written HW and projects
    - Accessing or viewing homework/project solutions online before they are officially released is strictly prohibited.
    - Posting solutions online (ex. public github project repo) is also considered cheating and a copyright violation
  - We have a zero-tolerance policy towards academic misconduct
    - We will forward all suspicious cases to the Center of Student Conduct, and recommend **immediate failure (F)** if the involved individuals are found guilty.
Today

- What is artificial intelligence?
- Where are we and how did we get here?
- How do we think about the design of AI systems?
A (Short) History of AI

Demo: HISTORY – MT1950.wmv
A short prehistory of AI

- **Prehistory:**
  - **Philosophy** (reasoning, planning, learning, science, automation)
  - **Mathematics** (logic, probability, optimization)
  - **Neuroscience** (neurons, adaptation)
  - **Economics** (rationality, game theory)
  - **Control theory** (feedback)
  - **Psychology** (learning, cognitive models)
  - **Linguistics** (grammars, formal representation of meaning)

- **Near miss (1842):**
  - Babbage design for universal machine
  - Lovelace: “a thinking machine” for “all subjects in the universe.”
“An attempt will be made to find how to make machines use language, form abstractions and concepts, solve kinds of problems now reserved for humans, and improve themselves. We think that a significant advance can be made if we work on it together for a summer.”

John McCarthy and Claude Shannon
Dartmouth Workshop Proposal
A (Short) History of AI

- **1940-1950: Early days**
  - 1943: McCulloch & Pitts: Boolean circuit model of brain
  - 1950: Turing's “Computing Machinery and Intelligence”

- **1950—70: Excitement: Look, Ma, no hands!**
  - 1950s: Early AI programs: chess, checkers (RL), theorem proving
  - 1956: Dartmouth meeting: “Artificial Intelligence” adopted
  - 1965: Robinson's complete algorithm for logical reasoning

- **1970—90: Knowledge-based approaches**
  - 1969—79: Early development of knowledge-based systems
  - 1980—88: Expert systems industry booms

- **1990—2012: Statistical approaches + subfield expertise**
  - Resurgence of probability, focus on uncertainty
  - General increase in technical depth
  - Agents and learning systems... “AI Spring”?

- **2012—___: Excitement: Look, Ma, no hands again?**
  - Big data, big compute, deep learning
  - AI used in many industries
AI as Designing Rational Agents

- An **agent** is an entity that *perceives* and *acts*.
- A **rational agent** selects actions that maximize its expected *utility*.
- Characteristics of the **sensors, actuators, and environment** dictate techniques for selecting rational actions.
- **This course** is about:
  - General AI techniques for many problem types
  - Learning to choose and apply the technique appropriate for each problem

Pac-Man is a registered trademark of Namco-Bandai Games, used here for educational purposes
Agents and environments

- An agent **perceives** its environment through **sensors** and **acts** upon it through **actuators** (or **effectors**, depending on whom you ask)
- The **agent function** maps percept sequences to actions
- It is generated by an **agent program** running on a **machine**
A human agent in Pacman
The task environment - PEAS

- **Performance measure**
  - -1 per step; +10 food; +500 win; -500 die; +200 hit scared ghost

- **Environment**
  - Pacman dynamics (incl ghost behavior)

- **Actuators**
  - Left Right Up Down or NSEW

- **Sensors**
  - Entire state is visible (except power pellet duration)
PEAS: Automated taxi

- **Performance measure**
  - Income, happy customer, vehicle costs, fines, insurance premiums

- **Environment**
  - US streets, other drivers, customers, weather, police...

- **Actuators**
  - Steering, brake, gas, display/speaker

- **Sensors**
  - Camera, radar, accelerometer, engine sensors, microphone, GPS

PEAS: Medical diagnosis system

- **Performance measure**
  - Patient health, cost, reputation

- **Environment**
  - Patients, medical staff, insurers, courts

- **Actuators**
  - Screen display, email

- **Sensors**
  - Keyboard/mouse
## Environment types

<table>
<thead>
<tr>
<th>Feature</th>
<th>Pacman</th>
<th>Backgammon</th>
<th>Diagnosis</th>
<th>Taxi</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fully or partially observable</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Single-agent or multiagent</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deterministic or stochastic</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Static or dynamic</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Discrete or continuous</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Known physics?</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Known perf. measure?</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Agent design

- The environment type largely determines the agent design
  - Partially observable => agent requires memory (internal state)
  - Stochastic => agent may have to prepare for contingencies
  - Multi-agent => agent may need to behave randomly
  - Static => agent has time to compute a rational decision
  - Continuous time => continuously operating controller
  - Unknown physics => need for exploration
  - Unknown perf. measure => observe/interact with human principal
Simple reflex agents
Pacman *agent program* in Python

class GoWestAgent(Agent):

    def getAction(self, percept):
        if Directions.WEST in percept.getLegalPacmanActions():
            return Directions.WEST
        else:
            return Directions.STOP
Eat adjacent dot, if any
Eat adjacent dot, if any
Pacman agent contd.

- Can we (in principle) extend this reflex agent to behave well in all standard Pacman environments?
  - No – Pacman is not quite fully observable (power pellet duration)
  - Otherwise, yes – we can (*in principle*) make a lookup table.....
  - *How large would it be?*
Model-based agents
Goal-based agents
Spectrum of representations

(a) Atomic
(b) Factored
(c) Structured
Outline of the course

- Atomic
- Factored
- Structured
- Deterministic
- Stochastic
- Known
- Unknown
- Search
- CSPs
- MDPs
- Bayes nets
- RL
- First-order logic
- Deterministic
- Stochastic
- Known
- Unknown
- Atomic
- Factored
- Structured