Artificial Intelligence

Jeff Clune
Assistant Professor
Evolving Artificial Intelligence Laboratory

University of Wyoming
Meet Your Classmates

• Pair up, introduce each other
  • Something interesting about them
  • Previous experience with AI
AI Challenge One

• Due Sept 13th (this Sunday before sleep)
Pen and Paper

• Start to bring them please
  • Lots of in-class problems
Environments/Problems/Domains

• Fully observable vs. partially observable
  • **Fully observable**: agents sensors can see all relevant info
    - no need to keep internal state
    - examples of fully observable problems?
  • **Partially observable**:
    - what are different reasons why world may not be observable?
    - examples of partially observable problems?
    - Is chess fully observable?
    - What about the castle rule? “Can only occur if neither piece is moved…”
      - What if I mark each piece as moved?
      - What if I list each move that has occurred?
    - Would that make every environment fully observable?
Environments/Problems/Domains

• Single agent vs. multi-agent
  • examples of single?
  • multi?
Environments/Problems/Domains

• Single agent vs. multi-agent
  • examples of single?
  • multi?
    - competitive
    - cooperative
Environments/Problems/Domains

- Deterministic vs. stochastic
Environments/Problems/Domains

• Episodic vs. sequential
  • Episodic: sense & act, repeat, but episode(t) is independent of episode (t-1)
    - examples?
Environments/Problems/Domains

• Episodic vs. sequential
  • Episodic: sense & act, repeat, but episode(t) is independent of episode (t-1)
    - examples?
      - classification tasks
Environments/Problems/Domains

• Episodic vs. sequential
  • Episodic: sense & act, repeat, but episode(t) is independent of episode (t-1)
    - examples?
      - classification tasks
  • Sequential
    - current action can influence all future actions
    - which is harder?
Environments/Problems/Domains

• Static vs. Dynamic
  • Dynamic: environment can change while agent is deliberating
    - If you choose not to decide (or think too long), you still have made a choice!
  • Examples of both?
Environments/Problems/Domains

• Discrete vs. Continuous
  • states, time, percepts, actions
    - separate time steps
    - integers vs. floats
Environments/Problems/Domains

• Known vs. Unknown
  • Known: agent knows the laws of physics ahead of time
    - or the rules of the game...how the system works
Environments/Problems/Domains

- Hardest
  - Partially observable
  - multiagent
  - stochastic
  - sequential
  - dynamic
  - continuous
  - unknown
Environments/Problems/Domains

• Hardest
  • Partially observable or fully?
  • single or multi agent?
  • deterministic or stochastic?
  • sequential or episodic?
  • dynamic or static?
  • discrete or continuous?
  • known or unknown?

What is homework one?
Possible Agents: Lookup Table

- If percepts are:
  
- Then

- aka “State Machine”
Possible Agents: Lookup Table

- **State space:** all of the possible agent situations
- **Chess:** $\sim 10^{150}$
- **Any robot with camera:** $\sim 10^{250,000,000,000}$
  - video at 27 mb/second
    - 30 frames per second, 640x480 pixels, 24 bits of color info
- **Number of atoms in observable universe:** $\sim 10^{50}$
Possible agents: Lookup Table

• Such large lookup tables are not going to work
  • can’t store them, learn them, etc.

• Key challenge for AI:
  • make small programs perform as well as optimal/good, vast, lookup tables
Possible Agents: Simple Reflex Agent

- Other end of the spectrum:
  - open-loop controller
    - steadfastly refuses to alter its plan despite what’s happening in reality
    - (i.e. ignores/has no percepts)
Possible Agents: Simple Reflex Agent

• Simple reflex agent
  • responds to current precepts
  • very limited
    - why? can you think of examples where you need more than the current percepts?
Possible Agents: Simple Reflex Agent

• Simple reflex agent
  • responds to current precepts
  • very limited
    - only works optimally if environment is fully observable
Possible Agents: Model-based Reflex Agent

• Best way to handle partial observability?
  • Remember important things that you can’t perceive now
  • I.e. build a model of the world
    - internal states that represent external states
Possible Agents: Model-based Reflex Agent

• Best way to handle partial observability?
  • Remember important things that you can’t perceive now
  • I.e. build a model of the world
    - internal states that represent external states

• To build a model
  • must guess how the world changes when you can’t see it
  • must guess how the agent’s actions change the world
Goal-based Agents

• Have goal; can figure out autonomously how to accomplish it

• The more abstract the goal can be specified, the better
  • Imagine hand coding every decision a robot makes to deliver a pizza vs. saying “deliver this pizza to 300 Water St.”
Utility-based Agents

• Have many goals
  • deliver pizza
  • fast
  • don’t crash
  • etc.

• Best way to combine these different factors into one score?
Utility-based Agents

- Have many goals
  - deliver pizza
  - fast
  - don’t crash
  - etc.

- Given uncertainty (stochasticity):
  - Maximize expected utility (aka expected value)
Pen and Paper

• Start to bring them please
  • Lots of in-class problems
Utility-based Agents

• Have many goals
  • deliver pizza
  • fast
  • don’t crash
  • etc.

• Given uncertainty (stochasticity):
  • Maximize expected utility (aka expected value)
Learning Agents

• Instead of programming each decision, they learn
  • there is usually an implicit or explicit utility function

• To key parts of an agent
  • Select actions to perform
  • Learn from what happened
    - requires knowing what’s good/bad
      - either a reward signal
      - or a internal “critic”
Exploration vs. Exploitation

• Two-armed bandit problem
  • Let’s play
    - Arm 1: payoff = ????
    - Arm 2: payoff = ????
    - Goal: a policy that maximized expected value over N pulls
    - Problem version 1: payoffs don’t change
      - your policy?
    - Problem version 2: payoffs change
      - your policy?
Exploration vs. Exploitation

- **Exploitation**: Doing the best you can given your current knowledge.
- **Exploration**: Trying things that are less-optimal (according to your current model) in order to improve the model.
  - examples from real life?
Abstraction

• Including only what’s relevant
  • e.g. in chess, the color of the pieces is not important