Class Logistics

- Instructor: Emma Brunskill
- TA: Christoph Dann
- Time: Monday/Wednesday 1:30-2:50pm
- Website: http://www.cs.cmu.edu/~ebrun/15889e/index.html
- We will be using Piazza for class discussions and communication: please use this to pose all standard questions
- Office hours will be announced
Prerequisites

• Assume basic familiarity with probability, machine learning, sequential decision making under uncertainty and programming

• It is useful but not required to have taken one or more of: Machine Learning, Stat Techniques in Robotics, Graduate AI.

• Enthusiasm and creativity are required!
Class Requirements & Policy

• Grading
  • Homeworks (30%)
  • Midterm (20%)
  • Final project (40%)
  • Participation (10%)

• Late policy
  • 4 late days to use without penalty on homeworks only across the semester. See website for full details.

• Collaboration: unless otherwise specified, written homeworks can be discussed with others but must be written up individually. You must write the names of the other students you collaborated with on your homework.
Reinforcement Learning

Learn a behavior strategy (policy) that maximizes the long term sum of rewards in an unknown & stochastic environment
RL Examples: Intelligent Tutoring Systems
RL Examples: Robotics
RL Examples: Playing Atari

Image from David Silver
RL Examples: Healthcare decision support
Go through background knowledge check
Why is RL Different Than Other AI and Machine Learning?

optimization +
RL: Designer Choices
RL: Designer Choices

- Representation (how represent the world and the space of actions/interventions, and feedback signal/ reward)
- Algorithm for learning
- Objective function
- Evaluation
Common Restrictions / Constraints

- Computation time
Common Restrictions / Constraints

- Computation time
- Data available
- Restricted in way can act (policy class, constraints on which actions can take in states)
- Online vs offline
- Do we get to choose how to act or does someone else (an expert, semi-expert, offpolicy/onpolicy learning…)

Carnegie Mellon University 15
Desirable Properties in a RL Algorithm?
Desirable Properties in a RL Algorithm?

Convergence
Consistency
Small generalization error
Small estimation error
Small approximation error
High learning speed
Safety
Broad Classes of RL Approaches

- Value-Based
- Value Function
- Model-Based
- Model-Free
- Actor Critic
- Policy-Based
- Policy

Image from David Silver
3 Important Challenges in Real Life RL

1. From Old Data to Future Decisions
2. Quickly Learning to Act Well: Highly Sample Efficient RL
3. Beyond Expectation: Safety & Risk Sensitive RL

→ Most of class will focus on these 3 topics
# Reasoning Under Uncertainty

<table>
<thead>
<tr>
<th>Learn model of outcomes</th>
<th>Multi-armed bandits</th>
<th>Reinforcement Learning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Given model of stochastic outcomes</td>
<td>Decision theory</td>
<td>Markov Decision Processes</td>
</tr>
<tr>
<td>Actions Don’t Change State of the World</td>
<td></td>
<td>Actions Change State of the World</td>
</tr>
</tbody>
</table>

Carnegie Mellon University
Markov Decision Processes
MDP is a tuple \((S,A,P,R,\gamma)\)

- Set of states \(S\)
- Start state \(s_0\)
- Set of actions \(A\)
- Transitions \(P(s'|s,a)\) (or \(T(s,a,s')\))
- Rewards \(R(s,a,s')\) (or \(R(s)\) or \(R(s,a)\))
- Discount \(\gamma\)
- Policy = Choice of action for each state
- Utility / Value = sum of (discounted) rewards

Slide adapted from Klein and Abbeel
• Value of a Policy

\[ V^\pi(s) = \sum_{s' \in S} p(s' \mid s, \pi(s)) \left[ R(s, \pi(s), s') + \gamma V^\pi(s') \right] \]

\[ Q^\pi(s, a) = \sum_{s' \in S} p(s' \mid s, a) \left[ R(s, a, s') + \gamma V^\pi(s') \right] \]

• Optimal Value & Optimal Policy

\[ V^*(s_i) = \max_a \left( \sum_{s_j \in S} p(s_j \mid s_i, a) \left[ R(s, \pi(s), s') + \gamma V^*(s_j) \right] \right) \]

\[ = \max_a Q^*(s, a) \]

\[ \pi^*(s) = \arg\max_a Q^*(s, a) \]
Bellman Equation

\[ V^*(s_i) = \max_a \left( \sum_{s_j \in S} p(s_j | s_i, a) \left[ R(s, \pi(s), s') + \gamma V^*(s_j) \right] \right) \]

- Holds for \( V^* \)
- Inspires an update rule
Value Iteration

1. Initialize $V_1(s_i)$ for all states $s_i$

2. $k=2$

3. While $k < \text{desired horizon}$ or (if infinite horizon) values have converged
   - For all $s$,
     \[
     V_k(s_i) = \max_a \left( \sum_{s_j \in S} p(s_j \mid s_i, a) \left[ R(s, \pi(s), s') + \gamma V_{k-1}(s_j) \right] \right)
     \]
     \[
     \pi_k(s_i) = \arg\max_a \left( \sum_{s_j \in S} p(s_j \mid s_i, a) \left[ R(s, \pi(s), s') + \gamma V_{k-1}(s_j) \right] \right)
     \]
Will Value Iteration Converge?

• Yes, if discount factor is < 1 or end up in a terminal state with probability 1

• Bellman equation is a contraction

• If apply it to two different value functions, distance between value functions shrinks after apply Bellman equation to each
Bellman Operator is a Contraction

\[ \| V - V' \| = \text{Infinity norm} \]

(find max diff
Over all states)

\[
\| BV - BV' \| = \max_a \left[ R(s, a) + \gamma \sum_{s_j \in S} p(s_j | s_i, a) V(s_j) \right] \\
- \max_{a'} \left[ R(s, a') - \gamma \sum_{s_j \in S} p(s_j | s_i, a') V'(s_j) \right]
\]

\[
\leq \max_a \left[ R(s, a) + \gamma \sum_{s_j \in S} p(s_j | s_i, a) V(s_j) - R(s, a) + \gamma \sum_{s_j \in S} p(s_j | s_i, a) V'(s_j) \right]
\]

\[
\leq \gamma \max_a \left[ \sum_{s_j \in S} p(s_j | s_i, a) V(s_j) - \sum_{s_j \in S} p(s_j | s_i, a) V'(s_j) \right]
\]

\[
= \gamma \max_a \left[ \sum_{s_j \in S} p(s_j | s_i, a) (V(s_j) - V'(s_j)) \right]
\]

\[
\leq \gamma \max_{a, s_i} \sum_{s_j \in S} p(s_j | s_i, a) |V(s_j) - V'(s_j)|
\]

\[
\leq \gamma \max_{a, s_i} \sum_{s_j \in S} p(s_j | s_i, a) \| V - V' \|
\]

\[
= \gamma \| V - V' \|
\]
Properties of Contraction

- Only has 1 fixed point
  - If had two, then would not get closer when apply contraction function, violating definition of contraction
- When apply contraction function to any argument, value must get closer to fixed point
  - Fixed point doesn’t move
  - Repeated function applications yield fixed point
Value Iteration Converges

- If discount factor < 1
- Bellman is a contraction
- Value iteration converges to unique solution which is optimal value function