CSE 411: Machine Learning

Reinforcement Learning

Dr Muhammad Abul Hasan

Department of Computer Science and Engineering
Green University of Bangladesh
muhammad.hasan@cse.green.edu.bd

Fall 2023

Outline

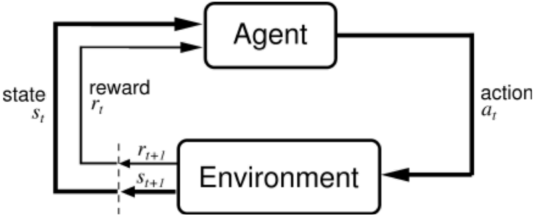
“ *Learning never exhausts the mind.*

– Leonardo da Vinci ”

Next Up ...

Reinforcement Learning

- Basic Idea:
  - Receive feedback in the form of **rewards**
  - Agent's utility is defined by the reward function
  - Must (learn to) act so as to **maximize expected rewards**

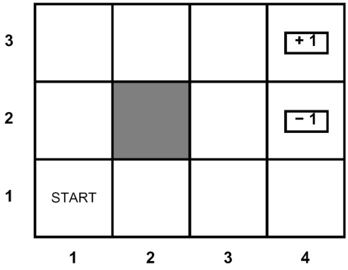


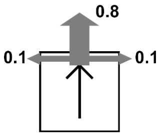
Reinforcement Learning

- RL algorithms attempt to find a policy for maximizing cumulative reward for the agent over the course of the problem.
- Typically represented by a **Markov Decision Process**
- RL differs from supervised learning in that correct input/output pairs are never presented, nor sub-optimal actions explicitly corrected.

Grid World

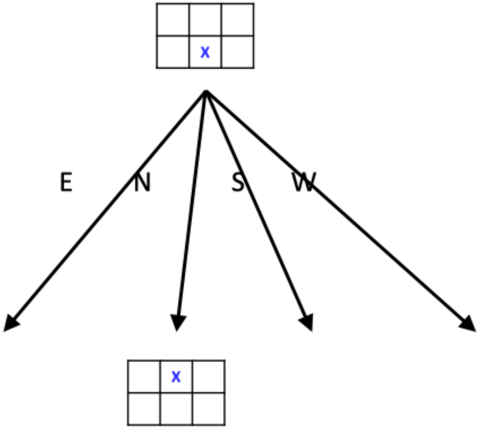
- The agent lives in a grid
- Walls block the agent's path
- The agent's actions do not always go as planned:
  - 80% of the time, the action North takes the agent North (if there is no wall there)
  - 10% of the time, North takes the agent West; 10% East
  - If there is a wall in the direction the agent would have been taken, the agent stays put
- Small "living" reward each step
- Big rewards come at the end
- Goal: maximize sum of rewards*



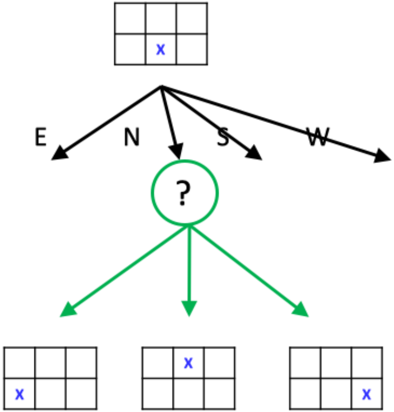


Grid Futures

- Deterministic Grid World:



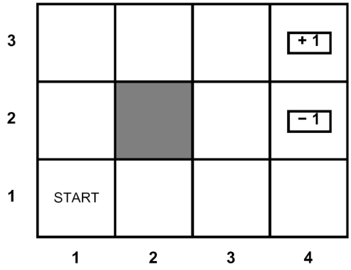
- Stochastic Grid World

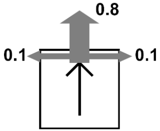


Next Up ...

Markov Decision Processes

- An MDP is defined by:
  - A set of states $s \in S$
  - A set of actions $a \in A$
  - A transition function $T(s, a, s')$
    - Prob that a from s leads to s'
    - i.e., $P(s'|s, a)$
    - Also called the model
  - A reward function $R(s, a, s')$
    - Sometimes just $R(s)$ or $R(s')$
  - A **start** state (or distribution)
  - Maybe a **terminal** state
  - A discount factor: γ
- MDPs are a family of non-deterministic search problems.





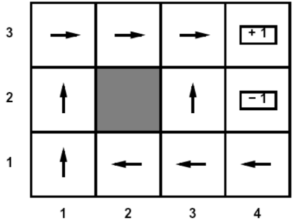
What is Markov about MDPs?

- "Markov" generally means that given the present state, the future and the past are independent
- For Markov decision processes, "Markov" means:

$$P(s_{t+1} = s'|S_t = s_t, A_t = a_t, St - 1 = s_{t-1}, A_{t-1} = a_{t-1}, \ldots, S_0 = s_0)$$
$$\equiv P(s_{t+1} = s'|S_t = s_t, A_t = a_t)$$

Solving MDPs

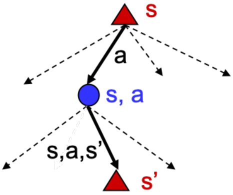
- In deterministic single-agent search problems, want an optimal **plan**, or sequence of actions, from start to a goal
- In an MDP, we want an optimal **policy** $\pi^* : S \to A$
  - A policy π gives an action for each state
  - An optimal policy maximizes expected utility if followed
  - Defines a reflex agent



Optimal policy when $R(s, a, s') = -0.03$ for all non-terminals s

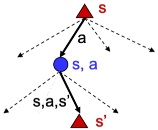
Recap: Defining MDPs

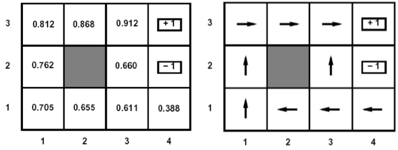
- Markov decision processes:
  - States S
  - Start state $s0$
  - Actions A
  - Transitions $P(s'|s,a)$ (or $T(s,a,s')$)
  - Rewards $R(s,a,s')$ (and discount γ)
- MDP quantities so far:
  - Policy = Choice of action for each state
  - Utility (or return) = sum of discounted rewards



Optimal Utilities

- Fundamental operation: compute the values (optimal expectimax utilities) of states s
- Why? Optimal values define optimal policies!
- Define the value of a state $s : V*(s) =$ expected utility starting in s and acting optimally
- Define the value of a q-state $(s, a) : Q*(s, a)$ $=$ expected utility starting in s, taking action a and thereafter acting optimally
- Define the optimal policy: $\pi*(s) =$ optimal action from state s





Value Iteration

- Idea:
  - Start with $V_0^*(s) = 0$, which we know is right (why?)
  - Given V_i^*, calculate the values for all states for depth $i + 1$:

$$V_{i+1} \leftarrow \max_a \sum_{s'} T(s, a, s')[R(s, a, s') + \gamma V_i(s')]$$

  - This is called a **value update** or **Bellman update**
  - Repeat until convergence
- Theorem: will converge to unique optimal values
  - Basic idea: approximations get refined towards optimal values
  - Policy may converge long before values do

Example: Value Iteration

V_1

	1	2	3	4
3	0	0	0	+1
2	0		0	−1
1	0	0	0	0

V_2

	1	2	3	4
3	0	0	0.72	+1
2	0		0	−1
1	0	0	0	0

- Information propagates outward from terminal states and eventually all states have correct value estimates

Example: Value Iteration

V_2

3	0	0	0.72	+1
2	0		0	-1
1	0	0	0	0
	1	2	3	4

V_3

3	0	0.52	0.78	+1
2	0		0.43	-1
1	0	0	0	0
	1	2	3	4

- Information propagates outward from terminal states and eventually all states have correct value estimates

Discounted Rewards

- Rewards in the future are worth less than an immediate reward
  - Because of uncertainty: who knows if/when you're going to get to that reward state

Thank You!