Reinforcement Learning

Approximate Dynamic Programming/ Neuro Dynamic Programming

Sanjit Kaul

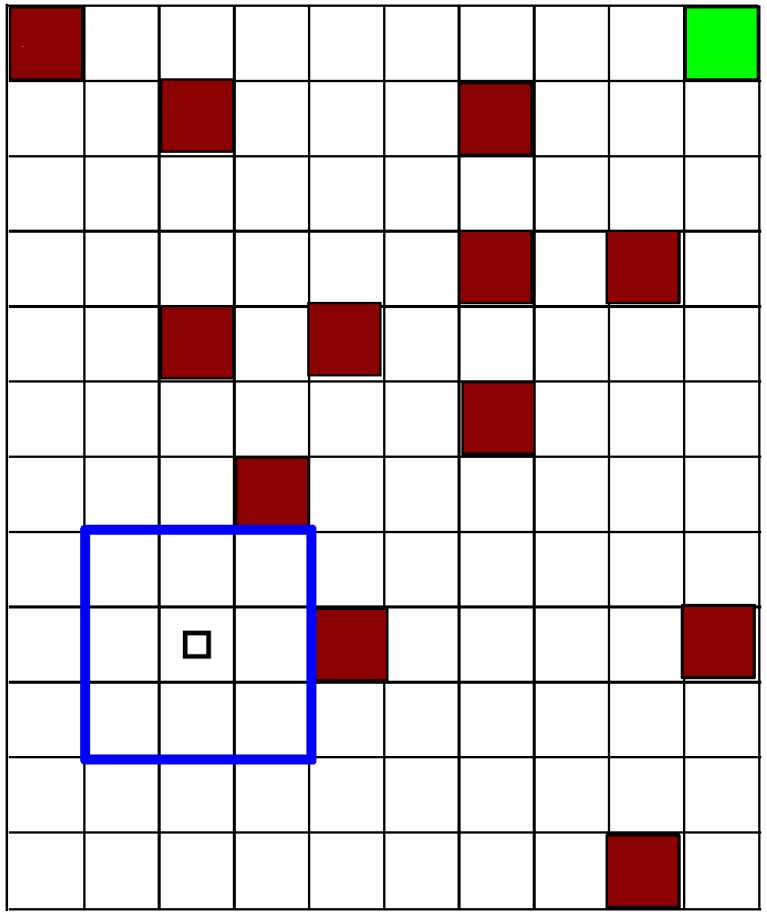
Mathematical Games

- 1961: Matchbox Educable Naughts and Crosses Engine (MENACE)
- Teaching a machine to play Naughts and Crosses (Tic-Tac-Toe)
  - Donald Michie, Biologist at University of Edinburgh, used 304 matchboxes
- One matchbox per tic tac toe position
  - Each matchbox contains multiple beads of different colors
    - Each color corresponding to a possible move
  - Shake the matchbox to randomly draw a bead
  - Remove (reduce in number) beads that led to a defeat in a game
  - Increase beads that led to a victory/ draw
- More details
  - http://cs.williams.edu/~freund/cs136-073/GardnerHexapawn.pdf

The Optimization Problem

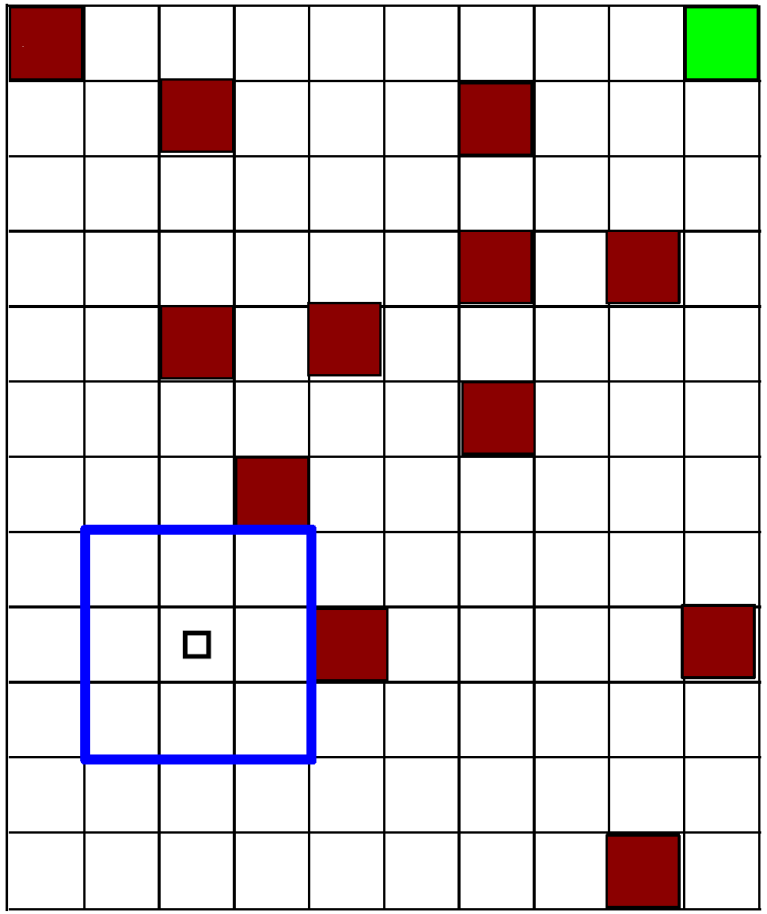
- Stochastic and Multi-stage (multi-time-step)
  - NOT one shot optimization
    - One shot: A priori deciding that you will drive in the right most lane
    - Multi-Stage: Change lane while you drive based on your perceived lane occupancy
      - Beneficial as the lane conditions are stochastic (not deterministic)
      - Multi-stage allows adapting to the current conditions
- Examples
  - Artificial Intelligence
    - Play games like humans
  - Operations Research
    - Inventory control
  - Optimal Control
    - Path planning
    - Network optimization

The Grid World



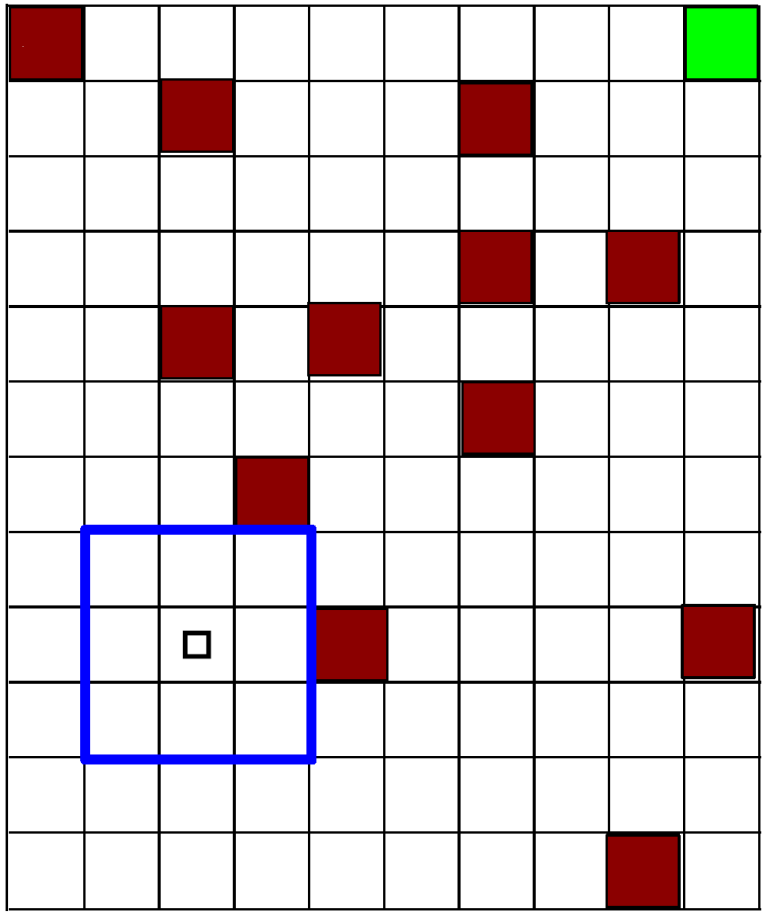
- State, Agent, Actions, Reward, Environment
- At every time-step/stage Agent can choose from a known set of *actions* in a state
  - State is a perfect summary the current environment.
  - All that the agent needs at a given time to choose the best possible action. No additional information can help, given the state.
  - Agent observes state and chooses its action.
  - State is *Markov*
    - *Often, an agent can't see the state and has access to observations (the state is partially observable or agent has imperfect state information)*
- Reward (negative Costs)/ Costs (negative rewards)
  - Agent receives reward after every action
- The received reward and next state is in general random (described by a PMF/PDF)

The Grid World



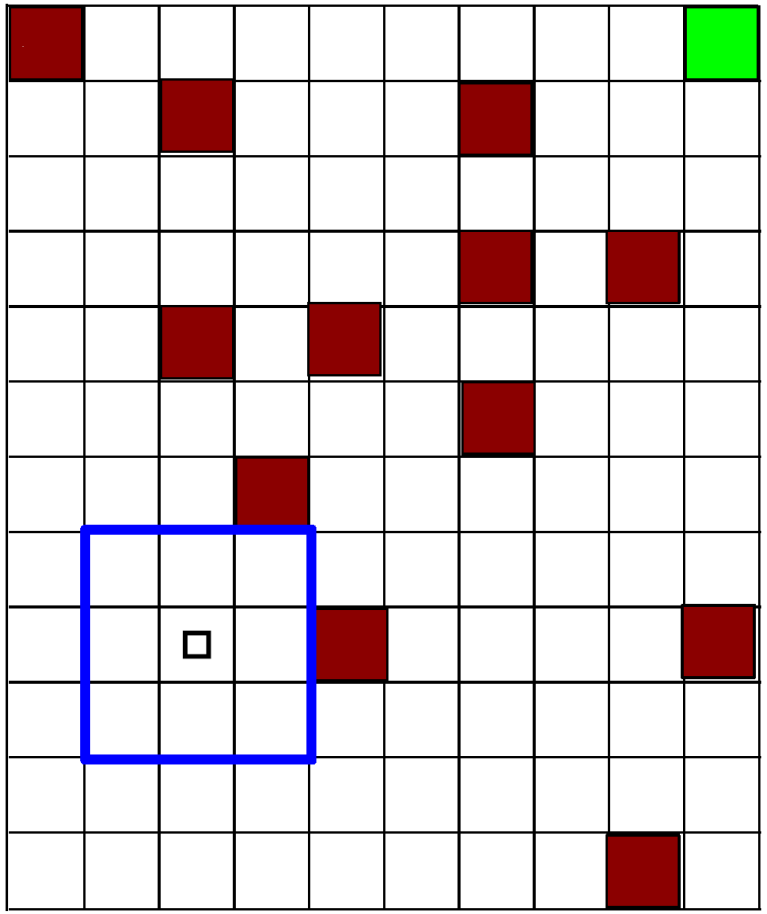
- No sense in optimizing random variables
- Instead optimize expected values of random variables
- Goal: Maximize *expected* sum reward starting in any state
  - Agent wants to perform its assigned task at hand such that sum reward is maximized
- Tasks may be finite time steps long, episodic, infinite
  - Episodic: Task is guaranteed to end in a finite but random number of steps
  - Infinite: Task continues for ever... for an infinite number of time steps
    - What happens to sum reward?
    - Maximize discounted sum reward

The Grid World



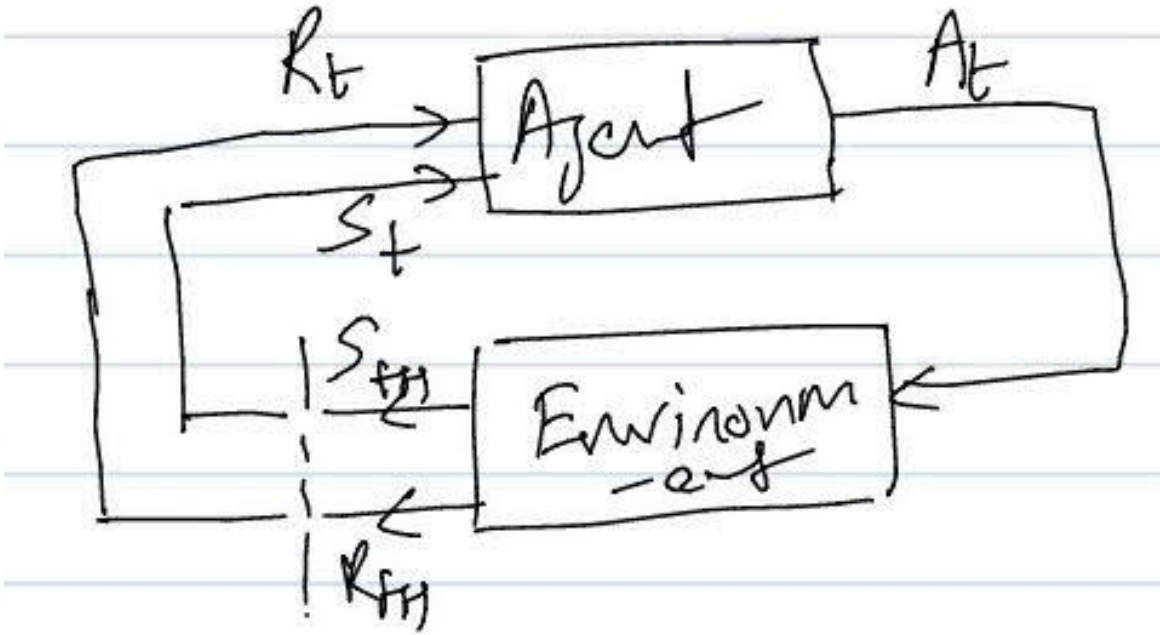
- RL algorithms help the agent learn an optimal map between states and actions
  - The functional map is called a policy
  - At every time step agent gives the observed state to the policy
  - Policy tells the agent the action it must perform
- If using optimal policy, agent maximizes expected sum reward

On the State



- Suppose the grid is static (occupancy never changes)
- What could be a state?
  - Agent's location on the grid?
  - Entire grid occupancy (obstacle/ free) and the agent's location?
  - Only the occupancy of the cell where the agent is currently?
- What if the grid is just an instance drawn in an IID manner?
- What if the grid summarizes a moving vehicle's view?

At Every Stage/ Time-Step



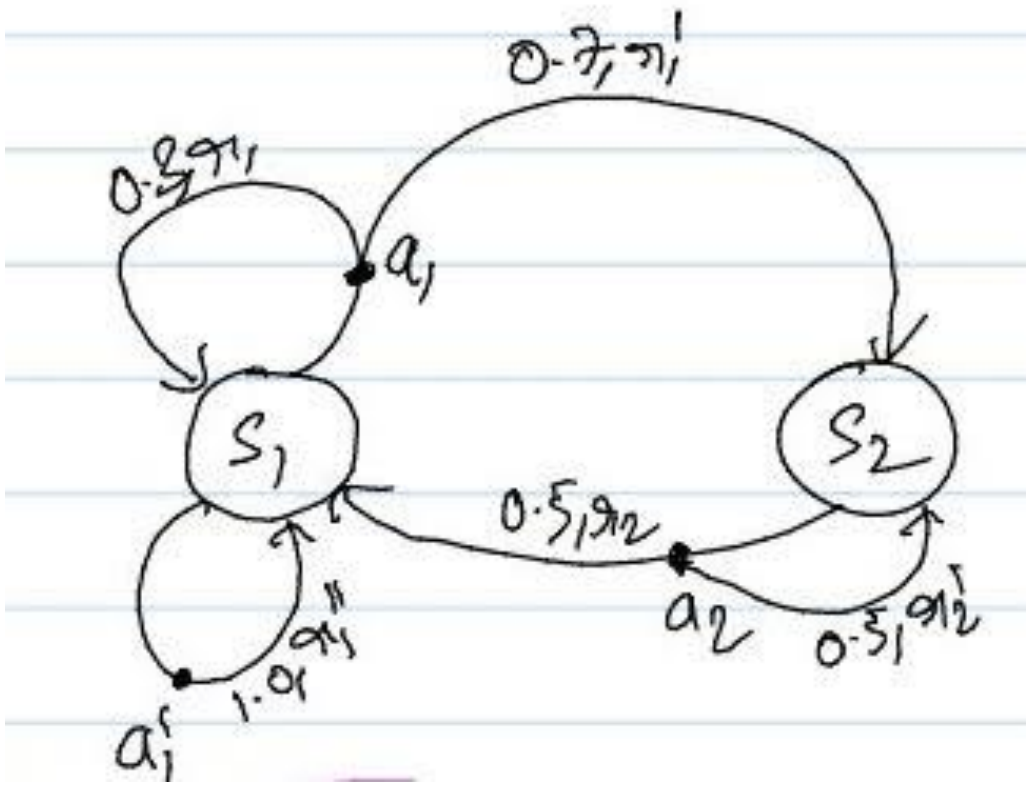
Capturing The Stochastic (Markov) Environment

- Use the so called Markov Decision Process
- Markov
  - The future is entirely determined by the current state and your action
  - Given the current, no amount of information from the past is beneficial
- MDP is just a bunch of probability mass functions (PMFs)

$$P\left[S_{t+1}=s', R_{t+1}=r \,\middle|\, S_t=s, A_t=a\right]$$

One such PMF for every selection of (s,a)

MDP Example (Graphical Representation)



Setting Rewards

- You need to nudge the agent to choose beneficial actions...
  - -1 reward for moving into an occupied cell
  - 0 for moving into a free cell
  - 1 for moving into the goal!

Return: The Sum Reward

Episodic:

$G_t \triangleq R_{t+1} + R_{t+2} + \cdots + R_T$

T is a random variable that is finite w.p. 1.

Continuing Task:

$G_t \triangleq R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots$

$0 \le \gamma < 1$ is the discounting factor

$\gamma = 0 \Rightarrow$ Myopic agent.

$\gamma \to 1 \Rightarrow$ Agent is more farsighted.

- Return Gt obtained by the agent starting at time *t*

Solve Using Dynamic Programming

- In principle, can always find the optimal *solution*
- Large state and action spaces make exact DP computationally infeasible
  - Grid: How big a mapping is the policy function? How many states?
- Models of state evolution may not be known
  - May be easier to simulate than capture in an analytic form
- Approximations and numerical techniques become important

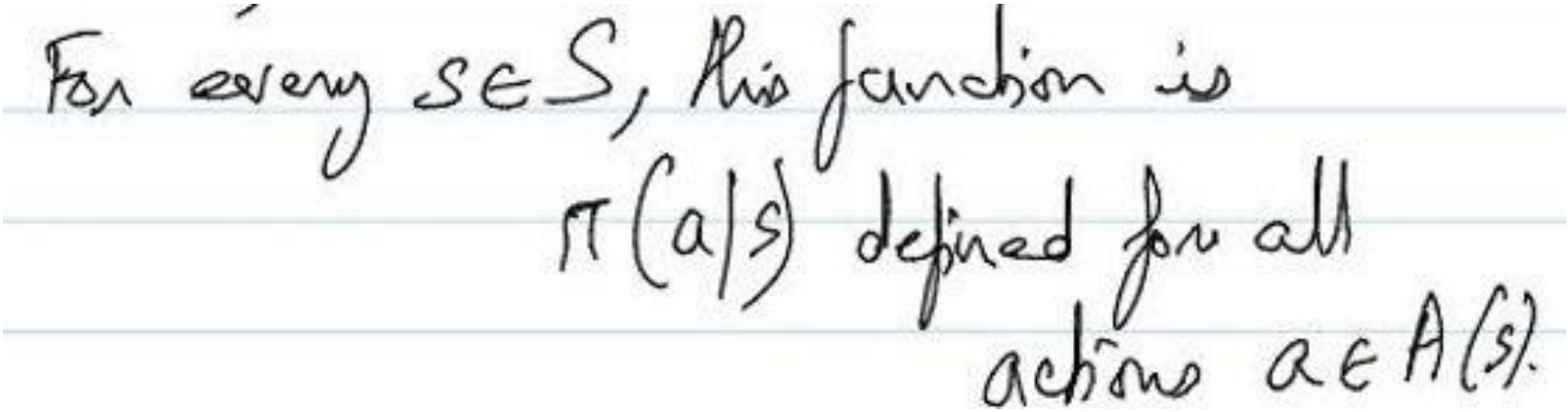
Expected Sum Reward

$$E\left[\sum_{t=1}^{\infty} \gamma^{t-1} R_t\right]$$, where R_t is a random variable that is a function of S_{t-1}, A_{t-1}, S_t.

- The agent could optimize this over a sequence of action a0, a1, a2,…
  - This would be the one-shot optimization way
  - The agent sticks to an a priori chosen sequence of actions
  - Is unable to adapt its actions to what it sees when it interact with the environment
- Our agent must choose a policy (a map between state and actions)

Policy Function

- A conditional PMF over the set of actions conditioned on the state



State-Value and Action-Value Function

$$v_\pi(s) = E_\pi\left[G_t \mid S_t = s\right]$$

$$q_\pi(s,a) = E_\pi\left[G_t \mid S_t = s, A_t = a\right]$$

Relate the two functions?