Discrete versus Continuous Markov Decision Processes

Ashwin Rao

ICME, Stanford University

January 23, 2020

"Discrete or Continuous" in States, Actions or Time Steps

- When we say Discrete or Continuous MDP, we could be talking of:
  - States
  - Actions
  - Time Steps
- Basic Case: Finite in States & Actions, Discrete in Time Steps
- Classical Dynamic Programming (DP) algorithms cover this case
- DP algorithms sweep through all States, consider all State Transitions
- Updates a table mapping each State to its Value Function (VF)
- We call these (Policy Iteration, Value Iteration) as tabular algorithms
- Policy Improvement sweeps though all Actions ($\arg\max_a Q(s, a)$)

States: Value Function Approx and Sampling/Simulations

- Let's first consider State Space in real-world problems
- Real-world problems suffer from the two so-called "Curses":
    - Curse of Dimensionality (CD): Multi-Dim/Continuous State Space
    - Curse of Modeling (CM): Transition Probabilities/Rewards too complex
- CD leads us to Value Function Approximation (eg: Deep Networks)
- CD and CM can be cured with Sampling/Simulations
- RL algorithms can be employed either with Actual Experiences or with Sampling/Simulations
- Simulation Model is often a feasible alternative to Probability Model
- Besides, we don't need the precise model dynamics as long as we have a good approximation for the Value Function

Multi-Dimensional/Continuous Actions

- How to improve Policy if Action Space is Multi-Dim/Continuous?
- $\pi'(s) = \arg\max_a Q(s, a)$ cannot be done as a sweep over actions
- Instead, we perform an (unconstrained) optimization over a on $Q(s, a)$
- For analytical DP solution, write out Bellman Optimality Equation and assume a functional form for the Optimal Value Function (with unknown parameters)
- Then take partial derivatives of the expression within max with respect to the dimensions of Action Space, and set to 0
- This gives us the optimal actions in terms of the state and the Optimal VF parameters
- Substituting the optimal actions in the Bellman Optimality Equation gives us a recursive expression for the Optimal VF parameters
- Use boundary condition for Optimal VF to solve for the parameters
- This gives us the Optimal VF and the Optimal Policy
- If we are restricted to doing RL $\Rightarrow$ Policy Gradient Algorithms

Continuous in States, Actions, Time Steps

- Optimal VF expressed in terms of state dimensions s_t and time t
- In continuous time, we can write Optimal VF as a differential dV^*

$$\max_a \mathbb{E}[dV^*(t, s_t) + R(t, s_t, a_t) \cdot dt] = 0$$

  where $R(t, s_t, a_t)$ is the Reward per unit time
- This is called the Hamilton-Jacobi-Bellman (HJB) equation
- dV^* is expanded as Taylor series in terms of t and s_t (involving partial derivatives of V^* w.r.t. t and s_t)
- This is Ito's Lemma if dynamics for s_t based on Brownian motion
- We eliminate randomness from the expression due to the $\mathbb{E}$ operation
- Let resultant expression (involving partials w.r.t. t, s_t) be $\phi(t, s_t, a_t)$

$$\max_a \phi(t, s_t, a_t) = 0$$

Continuous in States and Actions and Time Steps

- Setting partial derivatives of ϕ w.r.t. a_t to 0 gives optimal a_t^*
- a_t^* is now in terms of partial derivatives of V^* w.r.t. t and s_t
- Substituting a_t^* in ϕ gives:

$$\phi(t, s_t, a_t^*) = 0$$

- This is a partial differential equation for V^* in terms of t and s_t
- Boundary condition for PDE obtained from terminal Reward
- We would typically solve this PDE numerically
- If we seek an analytic solution, use Boundary condition to make a smart guess for functional form of V^* in terms of t and s_t
- This would lead us to an ODE whose solution provides V^* as well as a_t^* in terms of t and s_t