

Dynamic Programming and Reinforcement Learning

Finite Time Markov Decision Processes (Week 1b)

Rainer Schlosser, Alexander Kastius

Hasso Plattner Institute (EPIC)

April 15, 2021

Outline

- Questions?
- Today: Getting familiar with MDPs
 Finite Horizon Problems
 Dynamic Programming
 Bellman Equation & Value Function

Goals for Today

- Understand: States, Actions, Rewards, State Transitions
- Learn: Finite Horizon MDPs
- Learn: Concept of Expected Future Rewards
- Evaluate: Policies and their Performance
- Learn: Solve Problems using DP (Backward Induction)

What are Finite Horizon MDP Problems?

- We seek to control a dynamic system over a finite time
- We consider a finite *sequence* of decisions
- The system evolves according to a certain (time-dependent) dynamic
- The decisions are supposed to be chosen such that a certain objective is optimized (expected rewards)
- Find the right balance between short and long-term effects

Example Decision Problems with Finite Horizon

Examples with Finite Horizon

- Drinking at a Party
- Exam Preparation
- Eating Cake
- Selling Christmas Trees
- Selling Airline Tickets
- Accommodation Services
- Perishable Products, Fashion, etc.

Task: Describe & Classify

- Goal/Objective
- State of the System
- Actions
- Dynamic of the System
- Revenues/Costs
- Finite/Infinite Horizon
- Stochastic Components



Classification (Finite Horizon Problems)

Example	Objective	State	Action	Dynamic	Payments
Drinking at Party	max fun	%	#beer	impact-rehab	fun/money
Exam Preparation	max mark/effort	#learned	#learn	learn-forget	effort, mark
Eating Cake	max utility	%cake	#eat	outflow	utility

Christmas Trees

Airline Tickets

Accommodation

Fashion Items

General Problem Components (Finite Horizon)

- What do you want to optimize (e.g., expected rewards) (Objective)
- Define the state of your system (State)
- Define the set of possible actions (time+state dependent) (Actions)
- Quantify event probabilities (time+state+action dep.) (Dynamics)
- Define rewards (time+state+action+event dep.) (Rewards)
- Define state transitions (time+state+action+event dep.) (Transitions)
- What happens at the end of the time horizon? (state dep.) (Final Rewards)

Solving Finite Horizon MDPs via Dynamic Programming

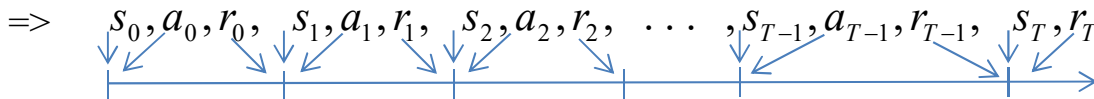
- **Continuous** Time Problems & Control Theory (not in focus)
 - Hamilton-Jacobi-Bellman Equation
 - Solve (Partial) Differential Equations
 - Allows for Structural and Analytical Results
 - See also: Pontryagin's maximum principle / Hamiltonian
 - See also: Differential Games & Stochastic Extensions
- **Discrete** Time Problems (today)
 - Bellman Equation, Backward Induction
 - Optimal numerical solutions

Basic Notation (Discrete Time Models)

- Framework: $t = 0, 1, 2, \dots, T$ Discrete time periods
- State: $s_t \in S$ One- or multi-dimensional
- Actions: $a_t \in A$ One- or multi-dimensional
- Events: $i_t \in I, P_t(i, a, s)$ Probability of event i in $(t, t+1)$ under a in s
- Rewards: $r_t = r_t(i, a, s)$ Realized reward in $(t, t+1)$ under i, a in s
- New State: $s_t \rightarrow s_{t+1} = \Gamma_t(i, a, s)$ State transition (under i, a in s)
- Initial State: $s_0 \in S$ State in $t=0$

Sequence of Events (Finite Horizon)

- $t=0$ start in state s_0 at the beginning of period (0,1)
 choose/play action a_0 for period (0,1)
 observe realized reward r_0 of period (0,1)
- $t=1$ observe realized new state s_1 after period (0,1) / the beginning of period (1,2)
 choose/play action a_1 for period (1,2)
 observe realized reward r_1 of period (1,2)
- ...
- $t=T$ observe realized new state s_T after period (T-1,T) / end of time horizon
 no further action
 observe realized terminal reward $r_T(s_T)$ (cf. state-dependent salvage value)



Markov Property

- The distribution of rewards and state transitions do only depend on the current state and action – not the history to get there.
- This can formally be expressed as:

$$P(s_{t+1} | s_t, a_t) = P(s_{t+1} | s_0, \dots, s_t, a_0, \dots, a_t) \quad \forall t = 0, 1, \dots, T-1$$

$$P(r_{t+1} | s_t, a_t) = P(r_{t+1} | s_0, \dots, s_t, a_0, \dots, a_t) \quad \forall t = 0, 1, \dots, T-1$$

Policies

- A non-anticipating (Markov) *policy* $\pi_t(s_t)$ determines an action a_t to be chosen/played in state s_t at time t , for all $s_t \in S$, $t = 0, \dots, T - 1$.
- A policy is usually deterministic, i.e., a unique action is chosen.
- A policy can also be randomized (mixed), i.e., an action is chosen according to a certain probability distribution within the set A .
- Note, already the number of potential det. policies is large, cf. $|A|^{|T \cdot |S|}$.

Could you Simulate Test Examples?

- Finite Horizon Problems
 - Eating cake (deterministic utility)
 - **Selling Airline Tickets (stochastic demand)**
- Describe: Horizon, states, actions, dynamics, rewards, transitions
- Evaluate rewards and state realizations of a certain Markov Policy (which contains “*what to do in which state at which point in time*”)
- Simulate multiple runs (and policies) & evaluate **average total rewards**

Example Problem (Selling Airline Tickets)



Example Problem (Selling Airline Tickets)

- Problem context: Sell items (N tickets) over time
- Time Horizon: Finite (T)
- Action: Offer price
- Demand: Stochastic (Price and Time-dependent)
- Rewards: Sales revenues & final rewards (salvage value)
- Goal: Maximize expected total profits
- How to find an optimal pricing policy?

Example MDP (Selling Airline Tickets)

- Framework: $t = 0, 1, 2, \dots, T$ Time Periods
- State: $s_t \in S := \{0, 1, \dots, N\}$ Items left
- Actions: $a_t \in A := \{5, 10, \dots, 400\}$ Price
- Events: $i_t \in I := \{0, 1\}$ with probabilities Demand
 $P_t(1, a, s) := (1 - a / 400) \cdot (1 + t) / T$ $P_t(0, a, s) = 1 - P_t(1, a, s)$
- Rewards: $r_t = r(i, a, s) := a \cdot \min(i, s)$ Revenue
- New State: $s_t \rightarrow s_{t+1} = \Gamma(i_t, a_t, s_t) := \max(0, s_t - i_t)$ Old – Sold
- Initial State: $s_0 \in S$ Initial items N
- Final Reward: $r_T(s) := f \cdot s$ with $f = 10$ Weight for Freight

Simulation of a Given Policy

- Assume (dynamic) pricing strategy $\pi_t(s) = 250$ (e.g., static price)
- Parameters: $T = 100$, $N = 50$, $f = 10$

time	state	action	sales	reward (revenue)	accum. revenue	new state
0	50	250	0	0	0	50
1	50	250	1	250	250	49
2	49	250	0	0	250	49
3	49	250	1	250	500	48
...						
50	5	/	/	50	11,300	/

- Expected performance of $\pi(s)$?
- What is the **best possible** performance & an **optimal** policy??

Problem Formulation (Finite Horizon)

- Find a *Markov policy* $\pi = \pi_t(s)$ that maximizes total expected (discounted) future rewards, $0 \leq \gamma \leq 1$:

$$\max_{\pi} E \left[\sum_{t=0}^T \underbrace{\gamma^t}_{\text{discount factor}} \cdot \left(\sum_{i_t \in I_t} \underbrace{P_t(i_t, a_t, s_t)}_{\text{probability for event } i_t \text{ under action } a_t \text{ in state } s_t} \cdot \underbrace{r_t(i_t, a_t, s_t)}_{\text{reward for event } i_t \text{ under action } a_t \text{ in state } s_t} \right) \middle| \underbrace{s_0}_{\text{initial state}} \right],$$

where states evolve according to $s_t \rightarrow s_{t+1} = \Gamma_t(i_t, a_t, s_t)$

- How to solve such problems? Answer: Dynamic Programming

Expected Future Rewards (Finite Horizon)

- Assume a given *policy* $\pi = \pi_t(s_t)$
- *Random* reward stream: $r_0, r_1, r_2, r_3, r_4, \dots, r_{T-1}, r_T$ (finite horizon)
- Expected future rewards . . .

. . . from $t=0$ on:
$$V_0^{(\pi)}(s) = E \left(\sum_{t=0, \dots, T} r_t \mid s_0 = s, a_t = \pi_t(s_t) \right)$$

. . . from time t on:
$$V_t^{(\pi)}(s) = E \left(\sum_{t=3, \dots, T} r_t \mid s_t = s, a_t = \pi_t(s_t) \right)$$

- $V_t^{(\pi)}(s)$ describes “*the value of being in a certain state s at time t* ”
for a given policy π , $s \in S$, $t = 0, \dots, T$.

Recursion for Expected Future Rewards (Finite T)

- *Random* reward stream: $r_0, r_1, r_2, r_3, r_4, \dots, r_{T-1}, r_T$ (finite horizon)
- Recursion for expected future rewards from time t on, $s \in S$:

$$\begin{aligned}
 V_t^{(\pi)}(s) &= E \left(\sum_{k=t, \dots, T} r_k \middle| s_t = s, \pi \right) = E \left(r_t + \sum_{k=t+1, \dots, T} r_k \middle| s_t = s, \pi \right) \\
 &= E \left(r_t + \underbrace{E \left(\sum_{k=t+1, \dots, T} r_k \middle| s_{t+1} = s', \pi \right)}_{?} \middle| s_t = s, \pi \right)
 \end{aligned}$$

Recursion for Expected Future Rewards (Finite T)

- *Random* reward stream: $r_0, r_1, r_2, r_3, r_4, \dots, r_{T-1}, r_T$ (finite horizon)
- Recursion for expected future rewards from time t on, $s \in S$:

$$\begin{aligned}
 V_t^{(\pi)}(s) &= E\left(\sum_{k=t, \dots, T} r_k \middle| s_t = s, \pi\right) = E\left(r_t + \sum_{k=t+1, \dots, T} r_k \middle| s_t = s, \pi\right) \\
 &= E\left(r_t + \underbrace{E\left(\sum_{k=t+1, \dots, T} r_k \middle| s_{t+1} = s', \pi\right)}_{?} \middle| s_t = s, \pi\right) \\
 &= E\left(r_t + V_{t+1}^{(\pi)}(s_{t+1}) \middle| s_t = s, \pi\right) \quad \text{i.e. rewards now + from t+1 on}
 \end{aligned}$$

Solution Approach (Dynamic Programming)

- What is the **best expected value** of having the chance to . . .

“sell items from time t on being in state s ”?

- Answer: That’s easy $V_t(s)$! ?????

Solution Approach (Dynamic Programming)

- What is the **best expected value** of having the chance to . . .

“sell items from time t on being in state s ”?

- Answer: That’s easy $V_t(s)!$?????
- We have renamed the problem. Awesome. But - that’s a solution approach!
- We don’t know the “**Value Function V** ”, but V has to satisfy the relation:

Value (state today) = Best expected (profit today + Value (state tomorrow))

Solution Approach (Dynamic Programming)

- *Value (state today) = Best expected (profit today + Value (state tomorrow))*
- Idea: Consider potential events & transitions within one period.

What can happen during one time interval (under action a)?

state in t	event	reward	state in $t+1$	probability
s	0	$r(0, a, s)$	$\Gamma(0, a, s)$	$P_t(0, a, s)$
	1	$r(1, a, s)$	$\Gamma(1, a, s)$	$P_t(1, a, s)$
	2	$r(2, a, s)$	$\Gamma(2, a, s)$	$P_t(2, a, s)$

- What does that mean for the **value** of state s at time t , i.e., $V_t(s)$?

Balancing Potential Short- and Long-Term Rewards

state in t	event	reward	state in $t+1$	probability
	0	$r(0, a, s)$	$\Gamma(0, a, s)$	$P_t(0, a, s)$
s	1	$r(1, a, s)$	$\Gamma(1, a, s)$	$P_t(1, a, s)$
	2	$r(2, a, s)$	$\Gamma(2, a, s)$	$P_t(2, a, s)$

$$V_t(s) = \max_{\substack{a \in A \\ \text{potential} \\ \text{actions}}} \left\{ \underbrace{P_t(0, a, s)}_{\text{probability not to sell in } (t, t+1)} \cdot \left(\underbrace{r(0, a, s)}_{\text{periods's reward}} + \underbrace{\gamma \cdot V_{t+1}(\Gamma(0, a, s))}_{\text{best disc. exp. future rewards}} \right) \right. \\
 \left. + \underbrace{P_t(1, a, s)}_{\text{probability of 1 sale in } (t, t+1)} \cdot \left(\underbrace{r(1, a, s)}_{\text{period's reward}} + \underbrace{\gamma \cdot V_{t+1}(\Gamma(1, a, s))}_{\text{best disc. exp. future rewards}} \right) + \dots \right\}$$

Bellman Equation (Finite Horizon)

- We obtain the *Bellman Equation*, which **determines** the Value Function:

$$V_t(s) = \max_{\substack{a \in A \\ \text{potential} \\ \text{actions}}} \left\{ \sum_{i \in I} \underbrace{P_t(i, a, s)}_{\text{probability}} \cdot \left(\underbrace{r(i, a, s)}_{\text{today's reward}} + \underbrace{\gamma \cdot V_{t+1}(\Gamma(i, a, s))}_{\text{best disc. exp. future rewards of new state}} \right) \right\}$$

- Ok, but why is that interesting?

Value Function & Optimal Policy

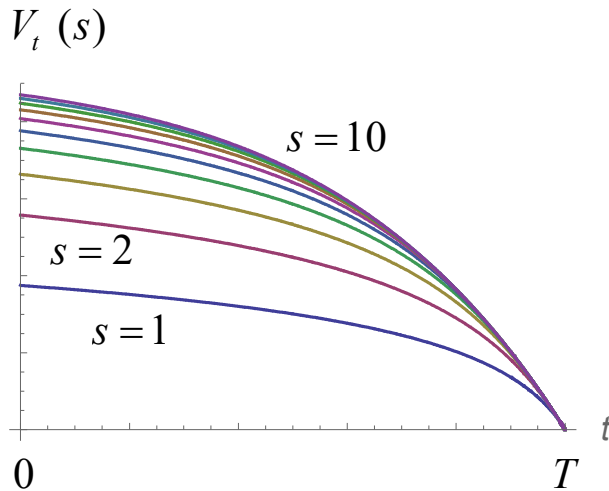
- We obtain the *Bellman Equation*, which **determines** the Value Function:

$$V_t(s) = \max_{\substack{a \in A \\ \text{potential} \\ \text{actions}}} \left\{ \sum_{i \in I} \underbrace{P_t(i, a, s)}_{\text{probability}} \cdot \left(\underbrace{r(i, a, s)}_{\text{today's reward}} + \underbrace{\gamma \cdot V_{t+1}(\Gamma(i, a, s))}_{\text{best disc. exp. future rewards of new state}} \right) \right\}$$

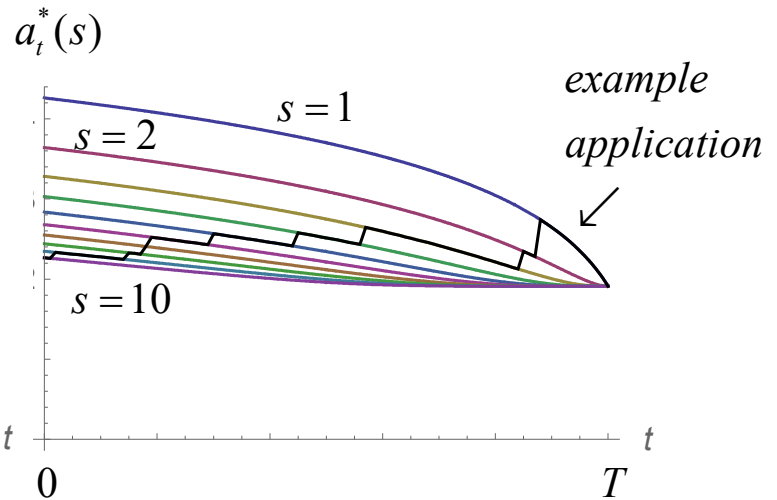
- Ok, but why is that interesting?
- Answer: Because $a_t^*(s) = \arg \max_{a \in A} \{ \dots \}$ is the *optimal policy*.
- Ok! Now, we just need to compute the Value Function!

Value Function & Optimal Policy (Illustration)

Value Function



Pricing Policy



Backward Induction for Discrete Finite Horizon MDPs

- Starting with the terminal condition $V_T(s) := r_T(s)$ at the horizon T
we can compute the value function *recursively* $\forall s \in S_t, t = 0, 1, \dots, T-1$:

$$V_t(s) = \max_{\substack{a \in A_t(s) \\ \text{potential} \\ \text{actions}}} \left\{ \sum_{i \in I_t} \underbrace{P_t(i, a, s)}_{\text{probability}} \cdot \left(\underbrace{r_t(i, a, s)}_{\text{today's reward}} + \underbrace{\gamma \cdot V_{t+1}(\Gamma_t(i, a, s))}_{\text{best disc. exp. future rewards of new state}} \right) \right\}$$

- The optimal strategy $a_t^*(s)$, $t = 0, 1, \dots, T-1$, $s \in S$,
is determined by the *arg max* of the value function
- The approach is general applicable & optimal for finite horizon problems
The numerical complexity increases with T , $|S|$, $|A|$, and $|I|$

Backward Induction Tabular Schema (Airline Example)

		<i>time / periods</i>			
		0		$T - 1$	T
<i>all states</i> $s \in S$	$s = 4$				$V_T(4) = r_T(4) = 0$
	$s = 3$				$V_T(3) = r_T(3) = 0$
	$s = 2$				$V_T(2) = r_T(2) = 0$
	$s = 1$				$V_T(1) = r_T(1) = 0$
	$s = 0$				$V_T(0) = r_T(0) = 0$

Backward Induction Tabular Schema (Airline Example)

		<i>time / periods</i>			
		0		$T-1$	T
<i>all states</i> $s \in S$	$s = 4$			$P_{T-1}(0, a, 4)$ $P_{T-1}(1, a, 4)$	$V_T(4) = r_T(4) = 0$
	$s = 3$	$P_{T-1}(0, a, 4) \cdot (r_{T-1}(0, a, 4) + \gamma \cdot V_T(4))$ $+ P_{T-1}(1, a, 4) \cdot (r_{T-1}(1, a, 4) + \gamma \cdot V_T(3))$			$V_T(3) = r_T(3) = 0$
	$s = 2$				$V_T(2) = r_T(2) = 0$
	$s = 1$				$V_T(1) = r_T(1) = 0$
	$s = 0$				$V_T(0) = r_T(0) = 0$

Backward Induction Tabular Schema (Airline Example)

time / periods

		0			$T-1$	T
<i>all states</i> $s \in S$	$s = 4$				$P_{T-1}(0, a^*, 4)$ $P_{T-1}(1, a^*, 4)$	$V_T(4) = r_T(4) = 0$
	$s = 3$					$V_T(3) = r_T(3) = 0$
	$s = 2$					$V_T(2) = r_T(2) = 0$
	$s = 1$					$V_T(1) = r_T(1) = 0$
	$s = 0$					$V_T(0) = r_T(0) = 0$

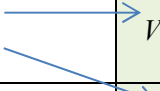
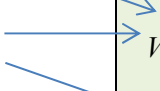
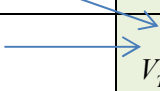
Backward Induction Tabular Schema (Airline Example)

		<i>time / periods</i>			
		0		$T - 1$	T
<i>all states</i> $s \in S$	$s = 4$			$V_{T-1}(4),$ $a_{T-1}^*(4)$	$V_T(4) = r_T(4) = 0$
	$s = 3$				$V_T(3) = r_T(3) = 0$
	$s = 2$				$V_T(2) = r_T(2) = 0$
	$s = 1$				$V_T(1) = r_T(1) = 0$
	$s = 0$				$V_T(0) = r_T(0) = 0$











Backward Induction Tabular Schema (Airline Example)

		<i>time / periods</i>			
		0		$T - 1$	T
<i>all states</i> $s \in S$	$s = 4$			$V_{T-1}(4),$ $a_{T-1}^*(4)$	$V_T(4) = r_T(4) = 0$
	$s = 3$			$P_{T-1}(0, a, 3)$ $P_{T-1}(1, a, 3)$	$V_T(3) = r_T(3) = 0$
	$s = 2$				$V_T(2) = r_T(2) = 0$
	$s = 1$				$V_T(1) = r_T(1) = 0$
	$s = 0$				$V_T(0) = r_T(0) = 0$

Backward Induction Tabular Schema (Airline Example)

		<i>time / periods</i>			
		0		$T - 1$	T
<i>all states</i> $s \in S$	$s = 4$			$V_{T-1}(4),$ $a_{T-1}^*(4)$	$V_T(4) = r_T(4) = 0$
	$s = 3$			$V_{T-1}(3),$ $a_{T-1}^*(3)$	$V_T(3) = r_T(3) = 0$
	$s = 2$				$V_T(2) = r_T(2) = 0$
	$s = 1$				$V_T(1) = r_T(1) = 0$
	$s = 0$				$V_T(0) = r_T(0) = 0$

Backward Induction Tabular Schema (Airline Example)

		<i>time / periods</i>			
		0	$T-2$	$T-1$	T
<i>all states</i> $s \in S$	$s = 4$			 	$V_{T-1}(4),$ $a_{T-1}^*(4)$ $V_T(4) = r_T(4) = 0$
	$s = 3$			 	$V_{T-1}(3),$ $a_{T-1}^*(3)$ $V_T(3) = r_T(3) = 0$
	$s = 2$			 	$V_{T-1}(2),$ $a_{T-1}^*(2)$ $V_T(2) = r_T(2) = 0$
	$s = 1$			 	$V_{T-1}(1),$ $a_{T-1}^*(1)$ $V_T(1) = r_T(1) = 0$
	$s = 0$			 	$V_{T-1}(0),$ $a_{T-1}^*(0)$ $V_T(0) = r_T(0) = 0$

Backward Induction Tabular Schema (Airline Example)

		<i>time / periods</i>					
		0	1	$T-2$	$T-1$	T	
<i>all states</i> $s \in S$	$s = 4$...	$V_{T-2}(4),$ $a_{T-2}^*(4)$	$V_{T-1}(4),$ $a_{T-1}^*(4)$	$V_T(4) = r_T(4) = 0$
	$s = 3$...	$V_{T-2}(3),$ $a_{T-2}^*(3)$	$V_{T-1}(3),$ $a_{T-1}^*(3)$	$V_T(3) = r_T(3) = 0$
	$s = 2$...	$V_{T-2}(2),$ $a_{T-2}^*(2)$	$V_{T-1}(2),$ $a_{T-1}^*(2)$	$V_T(2) = r_T(2) = 0$
	$s = 1$...	$V_{T-2}(1),$ $a_{T-2}^*(1)$	$V_{T-1}(1),$ $a_{T-1}^*(1)$	$V_T(1) = r_T(1) = 0$
	$s = 0$...	$V_{T-2}(0),$ $a_{T-2}^*(0)$	$V_{T-1}(0),$ $a_{T-1}^*(0)$	$V_T(0) = r_T(0) = 0$

Backward Induction Tabular Schema (Airline Example)

		<i>time / periods</i>					
		0	1	$T-2$	$T-1$	T	
<i>all</i> <i>states</i> $s \in \mathcal{S}$	$s = 4$	$V_0(4),$ $a_0^*(4)$	$V_1(4),$ $a_1^*(4)$	\dots	$V_{T-2}(4),$ $a_{T-2}^*(4)$	$V_{T-1}(4),$ $a_{T-1}^*(4)$	$V_T(4) = r_T(4) = 0$
	$s = 3$	$V_0(3),$ $a_0^*(3)$	$V_1(3),$ $a_1^*(3)$	\dots	$V_{T-2}(3),$ $a_{T-2}^*(3)$	$V_{T-1}(3),$ $a_{T-1}^*(3)$	$V_T(3) = r_T(3) = 0$
	$s = 2$	$V_0(2),$ $a_0^*(2)$	$V_1(2),$ $a_1^*(2)$	\dots	$V_{T-2}(2),$ $a_{T-2}^*(2)$	$V_{T-1}(2),$ $a_{T-1}^*(2)$	$V_T(2) = r_T(2) = 0$
	$s = 1$	$V_0(1),$ $a_0^*(1)$	$V_1(1),$ $a_1^*(1)$	\dots	$V_{T-2}(1),$ $a_{T-2}^*(1)$	$V_{T-1}(1),$ $a_{T-1}^*(1)$	$V_T(1) = r_T(1) = 0$
	$s = 0$	$V_0(0),$ $a_0^*(0)$	$V_1(0),$ $a_1^*(0)$	\dots	$V_{T-2}(0),$ $a_{T-2}^*(0)$	$V_{T-1}(0),$ $a_{T-1}^*(0)$	$V_T(0) = r_T(0) = 0$

Summary (Solving Discrete Time Finite Horizon MDPs)

Backward Induction

- (+) provides optimal solutions for finite horizon MDPs
- (+) allows for time-dependent frameworks
- (+) general applicable
- (+) numerically simple, no solver needed

- (-) full information required (cf. event & transition probabilities)
- (-) for medium size state spaces, does not scale (curse of dimensionality)

Recall - Questions?

- Markov Policies
- Recursive Concept for Future Rewards
- The Value of “being in a certain state”
- Bellman Equation & Recursive Problem Decomposition
- Backward Induction Solution Approach

Overview

Week	Dates	Topic
1	April 12/15	Introduction + Finite Time Markov Decision Processes (MDP)
2	April 19/22	Infinite Time MDPs + Dynamic Programming (DP) Exercise
3	April 26/29	(Approximate) Dynamic Programming (ADP)
4	May 3/6	Q-Learning (QL)
5	May 10	Deep Q-Networks (DQN) (Thu May 13 “Himmelfahrt”)
6	May 17/20	DQN Extensions
7	May 27	Policy Gradient Algorithms (Mon May 24 “Pfungstmontag”)
8	May 31/June 3	Project Assignments
9	June 7/10	Work on Projects: Input/Support
10	June 14/17	Work on Projects: Input/Support
11	June 21/24	Work on Projects: Input/Support
12	June 28/July 1	Work on Projects: Input/Support
13	July 5/8	Work on Projects: Input/Support
14	July 12/15 Aug 31	Final Presentations Finish Documentation

Exercise (Bonus) Dynamic Programming / Backward Induction

We want to sell event tickets using Dynamic Programming. We seek to find a pricing policy that optimizes expected profits. We have $N=50$ tickets and $T=200$ periods of time. Tickets cannot be sold after T . We do not use a discount factor and there is no salvage value for unsold items. We consider the following demand probabilities, i.e., $P_t(1, a) := (1 - a/400) \cdot (1+t)/T$ and

$$P_t(0, a) := 1 - P_t(1, a), \quad a \in A := \{5, 10, \dots, 400\}, \quad t = 0, 1, \dots, T-1.$$

- (a) Formulate a general model to sell tickets under given N , T , and demand probabilities P_t .
- (b) Solve the given example and output the solution in an appropriate way.
- (c) Simulate 1000 runs of applying the optimal policy over T periods. Show the distribution of realized total profits of these 1000 runs. Compare the mean with the value function.