Dynamic Programming and Reinforcement Learning

Infinite Time Markov Decision Processes (Week 2a)

Rainer Schlosser, Alexander Kastius

Hasso Plattner Institute (EPIC)

April 19, 2021

Outline

- Questions?
- Today: Infinite Horizon Problems

  Problem Examples

  Value Iteration

  Policy Iteration

Recap: Last Week

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

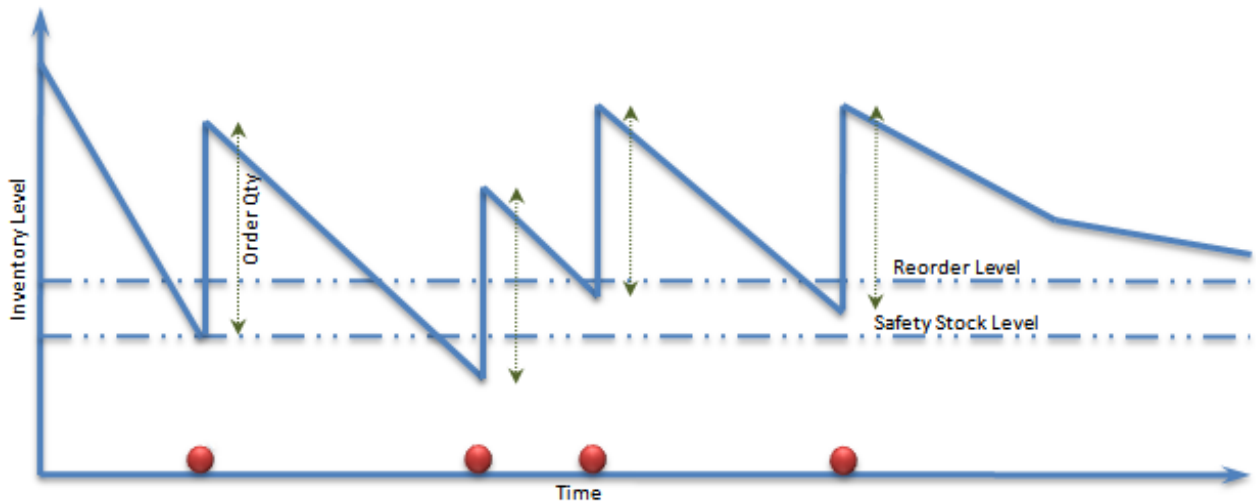
Solving MDP Problems

- Continuous Time Problems & Control Theory (not in focus)
- Discrete Time MDP Problems with **Finite Horizon** (last week)
  - *Time-dependent* Framework, Terminal Condition/Reward
  - Bellman Equation
  - Optimal numerical solutions via Backward induction
- Discrete Time MDP Problems with **Infinite Horizon** (today)
  - *Time-independent* Framework, Bellman Equation
  - Optimal numerical solutions via Value & Policy Iteration
  - Basis for **Reinforcement Learning**

Classification (Infinite Horizon Problems)

Example	Objective	State	Action	Dynamic	Payments
Inventory Mgmt.	min costs	#items	#order	order/holding	entry-sales
Advertising	max profits	image	#advertise	campaigns	effect-forget
Used Cars	min costs	age	replace(y/n)	buy/repair	aging/faults
Agriculture/Forestry					
Durable Products					
Chess/Go/Tetris					
Circular Economy					

Example Problem (Inventory Management)



Example Problem (Inventory Management)

- Problem context: Sell and order items over time
- Time Horizon: Infinite
- Demand: Stochastic (where price is fixed)
- Action: Replenish your inventory from time to time
- Rewards: Order cost vs. inventory holding costs
- Goal: Maximize expected discounted future profits

- How to find an optimal order policy?

Example MDP (Inventory Management)

- Framework: $t = 0,1,2,...,\infty$ — Discrete time periods
- State: $s \in S$ — Number of items left
- Actions: $a \in A$ — Number of ordered items (replenish)
- Events: $i \in I$, $P(i,a,s)$ — Demand i (e.g., 0,1,2,3 with prob. 1/4 each)
- Rewards: $r = r(i,a,s)$ $:= p \cdot \min(i,s) - c \cdot a$ $-h \cdot s - 1_{\{a>0\}} \cdot f$ — Revenue – Order Cost – Holding Cost; e.g., for given price p, variable order cost c, holding h, and fixed order costs f
- New State: $s \to s' = \Gamma(i,a,s)$ — Old – Sold + Replenish (end of period)
- Initial State: $s_0 \in S$ — Initial items in t=0

Sequence of Events (Infinite 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)

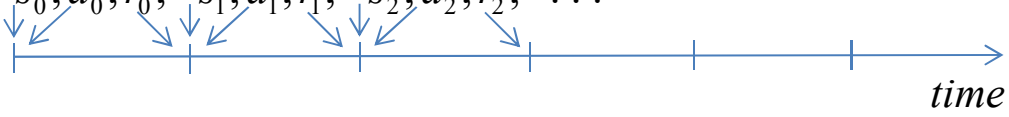
t=1 observe realized new state s_1 after period (0,1) / the beginning of period (1,2)

choose/play action a_1 for period (1,2)

observe realized reward r_1 of period (1,2)

. . .

=> $s_0, a_0, r_0, \; s_1, a_1, r_1, \; s_2, a_2, r_2, \; \ldots$



Simulation of a Given Policy

- Assume a (stationary/time-indep.) order strategy: $\pi(s) = \textit{if } s < 5 \textit{ then } 12 \textit{ else } 0$
- Parameters: $p = 10, c = 2, h = 0.5, f = 20$ Demand: $P(i) = 1/4,\ i = 0,1,2,3$

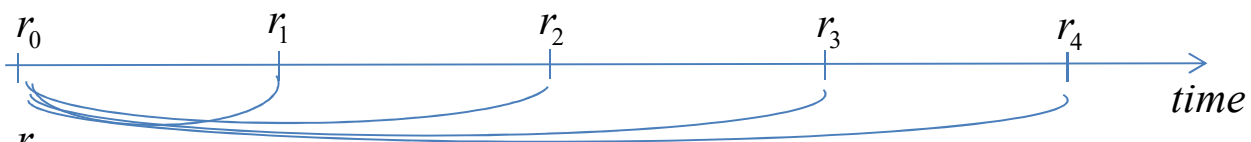
period	state	action	demand (event)	sales revenue	order cost	holding costs	reward	new state
0	5	0	1	10	0	-2.5	7.5	4
1	4	12	0	0	-24-20	-2	-46	16
2	16	0	2	20	0	-8	12	14
3	14	0	1	10	0	-7	-7	13
...								

- What is the tong-term performance of $\pi(s)$?

Discounting

- **Idea**: If delayed rewards are worth less, we use a penalty factor $\gamma < 1$ for each period to measure the value they have for us "now"

value of future rewards when being in $t = 0$?



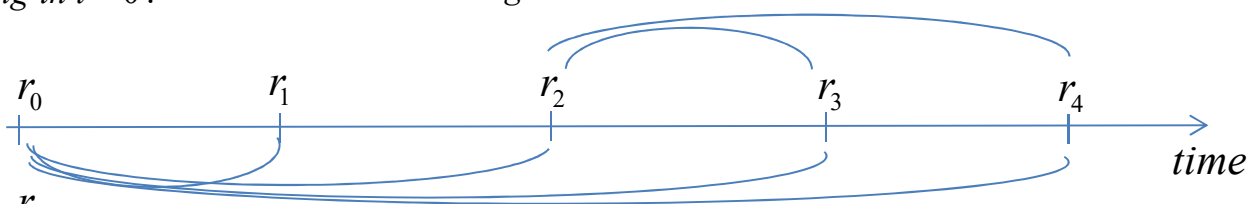
$$r_0 + \gamma \cdot r_1 + \gamma^2 \cdot r_2 + \gamma^3 \cdot r_3 + \gamma^4 \cdot r_4$$

Discounting

- **Idea**: If delayed rewards are worth less, we use a penalty factor $\gamma < 1$ for each period to measure the value they have for us "now"

value of future rewards when being in $t = 0$?

value of future rewards when being in $t = 2$?



$$r_0 + \gamma \cdot r_1 + \gamma^2 \cdot r_2 + \gamma^3 \cdot r_3 + \gamma^4 \cdot r_4$$

$$r_2 + \gamma \cdot r_3 + \gamma^2 \cdot r_4$$

Expected Discounted Future Rewards (Infinite Horizon)

- *Random* discounted reward stream: $r_0, \gamma \cdot r_1, \gamma^2 \cdot r_2, \gamma^3 \cdot r_3, \ldots$ with $\gamma \in [0,1)$
- Exp. disc. future rewards from t=0 on (discounted on t=0 using $\pi(s)$):

$$V_0^{(\pi)}(s) = E\left(\sum_{t=0}^{\infty} \gamma^t \cdot r_t \middle| s_0 = s, \pi \right)$$

Expected Discounted Future Rewards (Infinite Horizon)

- *Random* discounted reward stream: $r_0, \gamma \cdot r_1, \gamma^2 \cdot r_2, \gamma^3 \cdot r_3, \ldots$ with $\gamma \in [0,1)$
- Exp. disc. future rewards from t=0 on (discounted on t=0 using $\pi(s)$):

$$V_0^{(\pi)}(s) = E\left(\sum_{t=0}^{\infty} \gamma^t \cdot r_t \middle| s_0 = s, \pi \right)$$

Exp. disc. future rewards from t=3 on (discounted on *initial* time t=0):

$$\tilde{V}_3^{(\pi)}(s) = E\left(\sum_{t=3}^{\infty} \gamma^t \cdot r_t \middle| s_3 = s, \pi \right)$$

Exp. disc. future rewards from t=5 on **(discounted on *current* time t=3)**:

$$V_3^{(\pi)}(s) = E\left(\sum_{t=3}^{\infty} \gamma^{t-3} \cdot r_t \middle| s_3 = s, \pi \right) = E\left(\sum_{t=0}^{\infty} \gamma^t \cdot r_{t+3} \middle| s_3 = s, \pi \right)$$

Recursion for Future Rewards (Infinite Horizon)

- *Random* discounted reward stream: $r_0, \gamma \cdot r_1, \gamma^2 \cdot r_2, \gamma^3 \cdot r_3, \ldots$ with $\gamma \in [0,1)$
- **Recursion** for expected future rewards from time *t* on, $s \in S$:

$$V_t^{(\pi)}(s) = E\left(\sum_{k=t}^{\infty} \gamma^{k-t} \cdot r_k \middle| s_t = s, \pi \right) = E\left(\gamma^{t-t} \cdot r_t + \sum_{k=t+1}^{\infty} \gamma^{k-t} \cdot r_k \middle| s_t = s, \pi \right)$$

$$= E\left(r_t + \gamma \cdot \sum_{k=t+1}^{\infty} \gamma^{k-t-1} \cdot r_k \middle| s_t = s, \pi \right)$$

rewards now + from t+1 on?

Recursion for Future Rewards (Infinite Horizon)

- *Random* discounted reward stream: $r_0, \gamma \cdot r_1, \gamma^2 \cdot r_2, \gamma^3 \cdot r_3, \ldots$ with $\gamma \in [0,1)$
- **Recursion** for expected future rewards from time *t* on, $s \in S$:

$$V_t^{(\pi)}(s) = E\left(\sum_{k=t}^{\infty} \gamma^{k-t} \cdot r_k \middle| s_t = s, \pi \right) = E\left(\gamma^{t-t} \cdot r_t + \sum_{k=t+1}^{\infty} \gamma^{k-t} \cdot r_k \middle| s_t = s, \pi \right)$$

$$= E\left(r_t + \gamma \cdot \sum_{k=t+1}^{\infty} \gamma^{k-t-1} \cdot r_k \middle| s_t = s, \pi \right)$$ rewards now + from t+1 on?

$$= E\left(r_t + \gamma \cdot \underbrace{E\left(\sum_{k=t+1}^{\infty} \gamma^{k-(t+1)} \cdot r_k \middle| s_{t+1} = s' \right)}_{=V_{t+1}^{(\pi)}(s')} \middle| s_t = s, \pi \right)$$ yes :-)

Time Independence of Future Rewards (Infinite Horizon)

- Assume a given *policy* $\pi_t(s_t) = \pi(s_t)$, **which does not depend on time**
- Random discounted reward stream: $r_0, \gamma \cdot r_1, \gamma^2 \cdot r_2, \gamma^3 \cdot r_3, \ldots$ with $\gamma \in [0,1)$
- Expected future rewards from time t on (discounted on t), $s \in S$:

$$V_0^{(\pi)}(s) = E\left(\sum_{k=0}^{\infty} \gamma^k \cdot r_k \middle| s_0 = s, a_k = \pi(s_k) \right)$$

$$V_t^{(\pi)}(s) = V_0^{(\pi)}(s)? \quad \text{for all } t = 0,1,2,\ldots \quad ?$$

Time Independence of Future Rewards (Infinite Horizon)

- Assume a given *policy* $\pi_t(s_t) = \pi(s_t)$, **which does not depend on time**
- Random discounted reward stream: $r_0, \gamma \cdot r_1, \gamma^2 \cdot r_2, \gamma^3 \cdot r_3, \ldots$ with $\gamma \in [0,1)$
- Expected future rewards from time *t* on (discounted on *t*), $s \in S$:

$$V_0^{(\pi)}(s) = E\left(\sum_{k=0}^{\infty} \gamma^k \cdot r_k \middle| s_0 = s, a_k = \pi(s_k) \right)$$

$$V_t^{(\pi)}(s) = E\left(\sum_{k=t}^{\infty} \gamma^{k-t} \cdot r_k \middle| s_t = s, a_k = \pi(s_k) \right) = V_0^{(\pi)}(s) \quad \text{for all } t = 0,1,2,\ldots$$

- Same state, same actions, same expected reward stream, same discounting:

=> The value of "*being in a certain state*" is **time-independent** (cf. $V^{(\pi)}(s)$).

Problem Formulation (Infinite Horizon)

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

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

  where states evolve (time-independently) according to $s \to s' = \Gamma(i, a, s)$.

- How to solve such problems?
- Will the *value function* or the *optimal policy* be **time-dependent**?

Solution Approach (Dynamic Programming)

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

  "sell items (from any time t on, disc. on t) starting in state s"?

- Answer: That's easy $V(s)$!

Solution Approach (Dynamic Programming)

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

  "sell items (from any time t on, disc. on t) starting in state s"?

- Answer: That's easy $V(s)$!

- We can assume V is independent of time and satisfies the Bellman equation!

- We don't know the "Value Function V", but V is determined by:

  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 & transition dynamics within one period.
  What can happen during one time interval (under action a)?

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

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

Balancing Potential Short- and Long-Term Rewards

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

$$V(s) = \underbrace{\max_{a \in A}}_{\substack{\text{potential} \\ \text{actions}}} \left\{ \underbrace{P(0,a,s)}_{\text{probability not to sell}} \cdot \left(\underbrace{r(0,a,s)}_{\text{today's reward}} + \underbrace{\gamma \cdot V\left(\Gamma(0,a,s)\right)}_{\text{best disc. exp. future rewards}} \right) \right.$$

$$\left. + \underbrace{P(1,a,s)}_{\text{probability of demand}=1} \cdot \left(\underbrace{r(1,a,s)}_{\text{today's reward}} + \underbrace{\gamma \cdot V\left(\Gamma(1,a,s)\right)}_{\text{best disc. exp. future rewards}} \right) + \ldots \right\}$$

Bellman Equation (Infinite Horizon)

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

$$V(s) = \underbrace{\max_{a \in A}}_{\substack{\text{potential} \\ \text{actions}}} \left\{ \sum_{i \in I} \underbrace{P(i,a,s)}_{\text{probability}} \cdot \left(\underbrace{r(i,a,s)}_{\text{today's reward}} + \underbrace{\gamma \cdot V\left(\Gamma(i,a,s)\right)}_{\text{best disc. exp. future rewards of new state}} \right) \right\}$$

- Does it reveal optimal policies?

Value Function & Optimal Policy

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

$$V(s) = \underbrace{\max_{a \in A}}_{\substack{\text{potential} \\ \text{actions}}} \left\{ \sum_{i \in I} \underbrace{P(i,a,s)}_{\text{probability}} \cdot \left(\underbrace{r(i,a,s)}_{\text{today's reward}} + \underbrace{\gamma \cdot V\left(\Gamma(i,a,s)\right)}_{\text{best disc. exp. future rewards of new state}} \right) \right\}$$

- Does it reveal optimal policies?

- Yes, $a^*(s) = \arg\max_{a \in A} \{...\}$ is the *optimal policy.*

- But, how can we compute the Value Function? By *backward induction*?

Value Iteration for Infinite Horizon MDPs

- We want to determine the values $V^*(s)$ that solve the equation system

$$V^*(s) = \max_{a \in A} \left\{ \sum_{i \in I} P(i,a,s) \cdot \left(r(i,a,s) + \gamma \cdot V^* \left(\Gamma(i,a,s) \right) \right) \right\}$$

- **Value Iteration**: Use the "Finite horizon" *backward induction* approach

$$V_t(s) = \max_{a \in A} \left\{ \sum_{i \in I} P(i,a,s) \cdot \left(r(i,a,s) + \gamma \cdot V_{t+1} \left(\Gamma(i,a,s) \right) \right) \right\}, \; V_T(s) = r_T(s) = 0$$

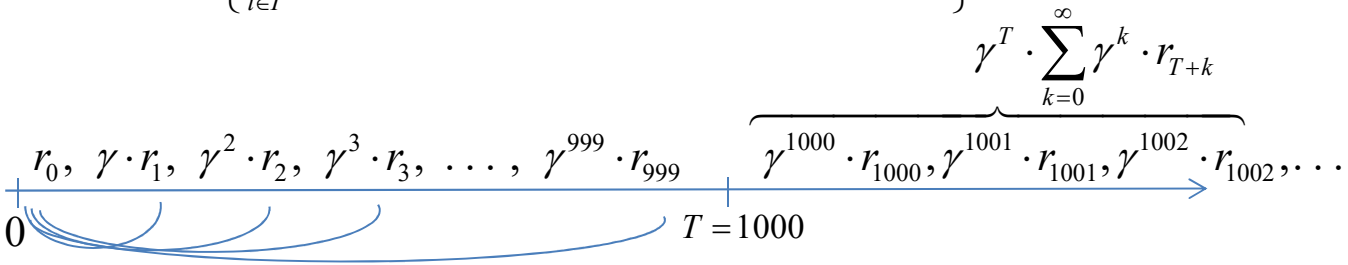
Value Iteration for Infinite Horizon MDPs

- We want to determine the values $V^*(s)$ that solve the equation system

$$V^*(s) = \max_{a \in A} \left\{ \sum_{i \in I} P(i,a,s) \cdot \left(r(i,a,s) + \gamma \cdot V^* \left(\Gamma(i,a,s) \right) \right) \right\}$$

- **Value Iteration**: Use the "Finite horizon" *backward induction* approach

$$V_t(s) = \max_{a \in A} \left\{ \sum_{i \in I} P(i,a,s) \cdot \left(r(i,a,s) + \gamma \cdot V_{t+1} \left(\Gamma(i,a,s) \right) \right) \right\}, \; V_T(s) = r_T(s) = 0$$



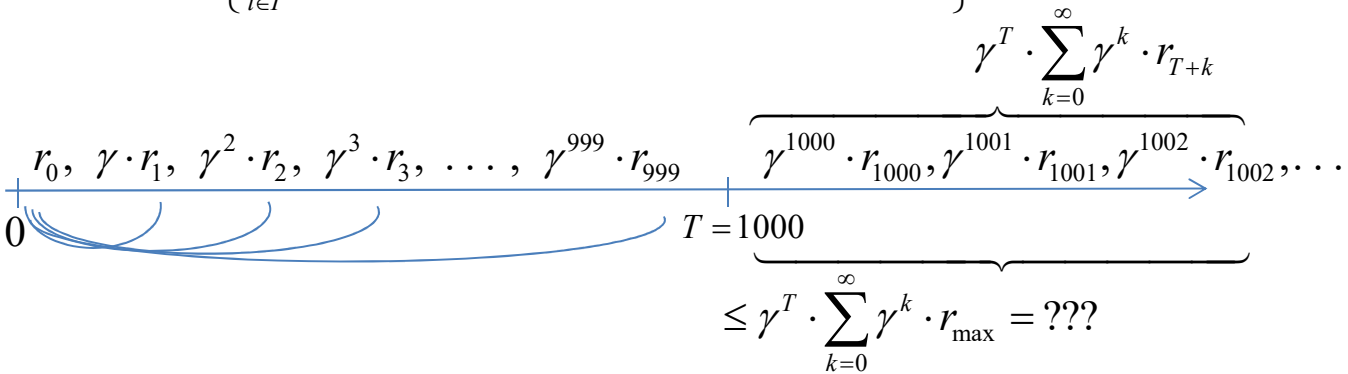
Value Iteration for Infinite Horizon MDPs

- We want to determine the values $V^*(s)$ that solve the equation system

$$V^*(s) = \max_{a \in A} \left\{ \sum_{i \in I} P(i,a,s) \cdot \left(r(i,a,s) + \gamma \cdot V^* \left(\Gamma(i,a,s) \right) \right) \right\}$$

- **Value Iteration**: Use the "Finite horizon" *backward induction* approach

$$V_t(s) = \max_{a \in A} \left\{ \sum_{i \in I} P(i,a,s) \cdot \left(r(i,a,s) + \gamma \cdot V_{t+1} \left(\Gamma(i,a,s) \right) \right) \right\}, \; V_T(s) = r_T(s) = 0$$



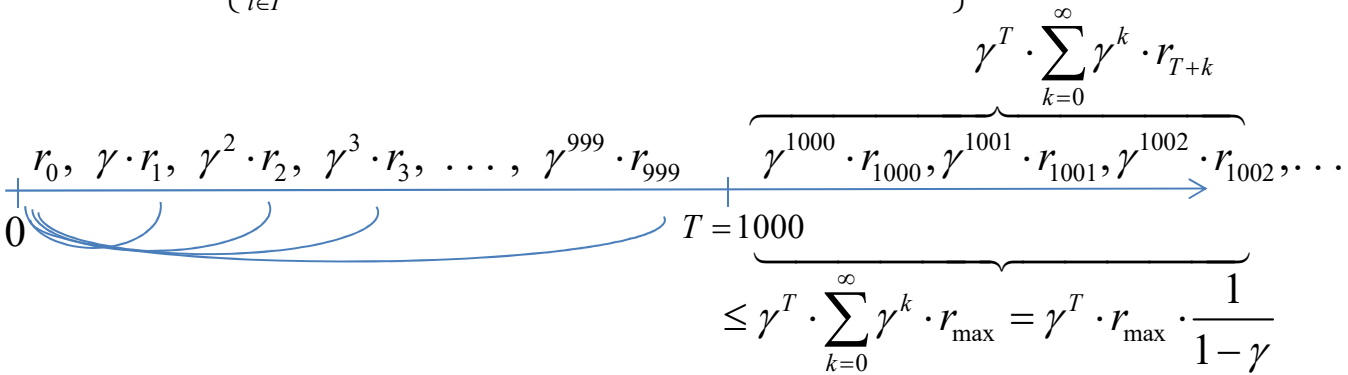
Value Iteration for Infinite Horizon MDPs

- We want to determine the values $V^*(s)$ that solve the equation system

$$V^*(s) = \max_{a \in A} \left\{ \sum_{i \in I} P(i,a,s) \cdot \left(r(i,a,s) + \gamma \cdot V^* \left(\Gamma(i,a,s) \right) \right) \right\}$$

- **Value Iteration**: Use the "Finite horizon" *backward induction* approach

$$V_t(s) = \max_{a \in A} \left\{ \sum_{i \in I} P(i,a,s) \cdot \left(r(i,a,s) + \gamma \cdot V_{t+1} \left(\Gamma(i,a,s) \right) \right) \right\}, \; V_T(s) = r_T(s) = 0$$



Value Iteration for Infinite Horizon MDPs

- We want to determine the values $V^*(s)$ that solve the equation system

$$V^*(s) = \max_{a \in A} \left\{ \sum_{i \in I} P(i,a,s) \cdot \left(r(i,a,s) + \gamma \cdot V^* \left(\Gamma(i,a,s) \right) \right) \right\}$$

- **Value Iteration**: Use the "Finite horizon" *backward induction* approach

$$V_t(s) = \max_{a \in A} \left\{ \sum_{i \in I} P(i,a,s) \cdot \left(r(i,a,s) + \gamma \cdot V_{t+1} \left(\Gamma(i,a,s) \right) \right) \right\}, \; V_T(s) = r_T(s) = 0$$

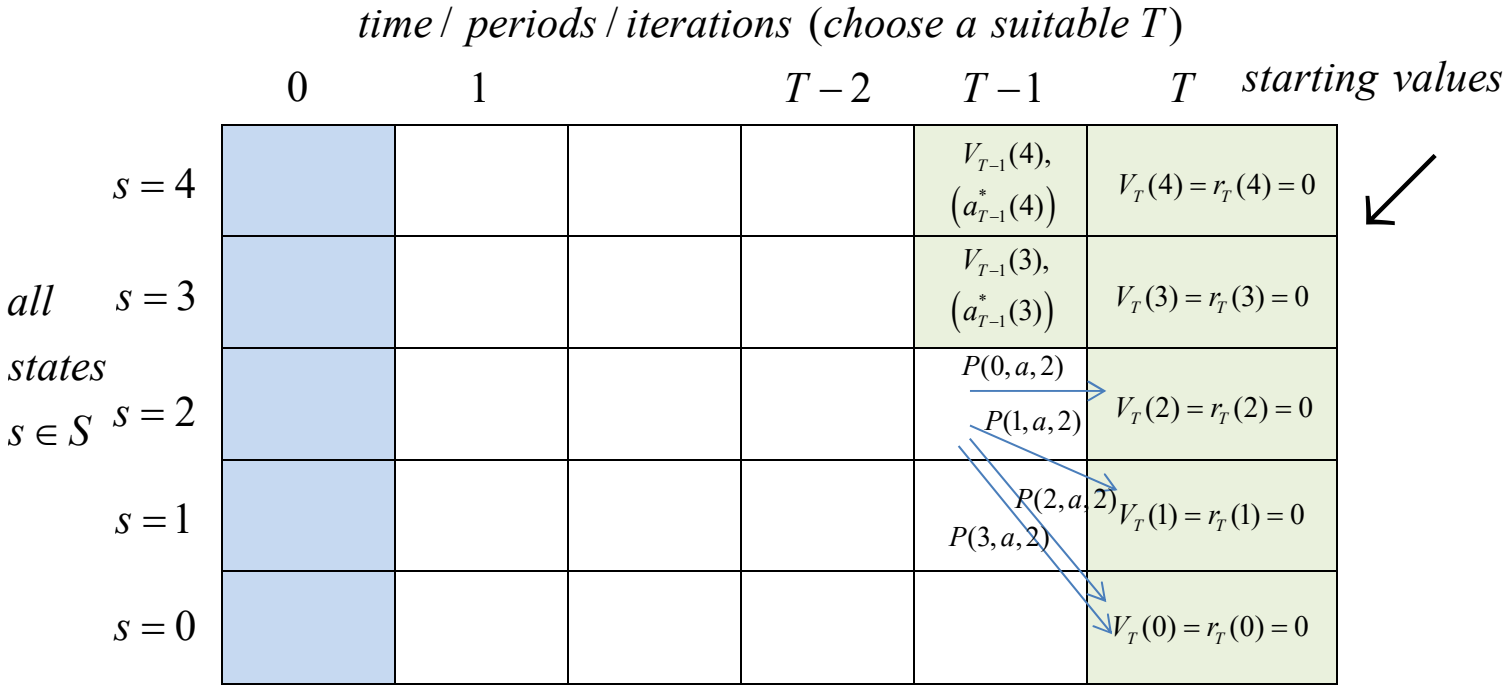
- For "large" T and for any initial $V_T(s)$ the values $V_0(s)$ converge to the exact values $V^*(s)$ with $\left| V_0(s) - V^*(s) \right| \leq \frac{\gamma^T}{1-\gamma} \cdot r_{\max} \xrightarrow{T \to \infty} 0$
- The optimal policy $a^*(s)$, $s \in S$, is determined by the *arg max* of the **last iteration step**, i.e., $a_0(p)$

Value Iteration Tabular Schema (Inventory Example)

time / periods / iterations (*choose a suitable* T)

all states $s \in S$	0	1		$T-2$	$T-1$	T *starting values*
$s=4$						$V_T(4) = r_T(4) = 0$
$s=3$						$V_T(3) = r_T(3) = 0$
$s=2$						$V_T(2) = r_T(2) = 0$
$s=1$						$V_T(1) = r_T(1) = 0$
$s=0$						$V_T(0) = r_T(0) = 0$

Value Iteration Tabular Schema (Inventory Example)



Value Iteration Tabular Schema (Inventory Example)

result / policy — *time / periods / iterations* — *try different starting values*

all states $s \in S$

	0	1	$\cdots$	$T-2$	$T-1$	T
$s=4$	$V_0(4),$ $a_0^*(4)$	$V_1(4),$ $a_1^*(4)$	$\cdots$	$V_{T-2}(4),$ $a_{T-2}^*(4)$	$V_{T-1}(4),$ $a_{T-1}^*(4)$	$V_T(4) = r_T(4) = 0$
$s=3$	$V_0(3),$ $a_0^*(3)$	$V_1(3),$ $a_1^*(3)$	$\cdots$	$V_{T-2}(3),$ $a_{T-2}^*(3)$	$V_{T-1}(3),$ $a_{T-1}^*(3)$	$V_T(3) = r_T(3) = 0$
$s=2$	$V_0(2),$ $a_0^*(2)$	$V_1(2),$ $a_1^*(2)$	$\cdots$	$V_{T-2}(2),$ $a_{T-2}^*(2)$	$V_{T-1}(2),$ $a_{T-1}^*(2)$	$V_T(2) = r_T(2) = 0$
$s=1$	$V_0(1),$ $a_0^*(1)$	$V_1(1),$ $a_1^*(1)$	$\cdots$	$V_{T-2}(1),$ $a_{T-2}^*(1)$	$V_{T-1}(1),$ $a_{T-1}^*(1)$	$V_T(1) = r_T(1) = 0$
$s=0$	$V_0(0),$ $a_0^*(0)$	$V_1(0),$ $a_1^*(0)$	$\cdots$	$V_{T-2}(0),$ $a_{T-2}^*(0)$	$V_{T-1}(0),$ $a_{T-1}^*(0)$	$V_T(0) = r_T(0) = 0$

Policy Iteration for Infinite Horizon MDPs

- We want to determine the values $V^*(s)$ that solve the system above
- **Policy iteration**: Subsequently *evaluate & improve* a policy:

  (1)

  (2)

  (3)

Policy Iteration for Infinite Horizon MDPs

- We want to determine the values $V^*(s)$ that solve the system above
- **Policy iteration**: Subsequently *evaluate & improve* a policy:

(1) Choose any starting policy $\pi(s)$, $s \in S$

(2) **Evaluate** the policy $\pi(s)$ by solving the linear system $\forall s \in S$

$$V^{(\pi)}(s) = \sum_{i \in I} P(i, \pi(s), s) \cdot \left(r(i, \pi(s), s) + \gamma \cdot V^{(\pi)}\left(\Gamma(i, \pi(s), s)\right)\right)$$

e.g., using a simplified Value Iteration (*without max* !) or via LP

(3) **Update** $\pi(s) \leftarrow \arg\max_{a \in A} \left\{ \sum_{i \in I} P(i, a, s) \cdot \left(r(i, a, s) + \gamma \cdot V^{(\pi)}\left(\Gamma(i, a, s)\right)\right)\right\}$

Repeat Step (2) & (3) $\forall s$ until no improvement (=> optimal solution!)

Policy Iteration Schema (Inventory Example)

Choose a starting policy $\pi^{(0)}(s)$, $s \in S$, cf. Step (1).

Compute the value function $V^{(\pi^{(0)})}(s)$ of policy $\pi^{(0)}(s)$, cf. Step (2).
Use $V^{(\pi^{(0)})}(s)$ to improve policy $\pi^{(0)}(s)$ to a new policy $\pi^{(1)}(s)$, cf. Step (3).

Compute the value function $V^{(\pi^{(1)})}(s)$ of policy $\pi^{(1)}(s)$, cf. Step (2).
Use $V^{(\pi^{(1)})}(s)$ to improve policy $\pi^{(1)}(s)$ to a new policy $\pi^{(2)}(s)$, cf. Step (3).

. . .

Stop if after an iteration k we have $\pi^{(k)}(s) = \pi^{(k-1)}(s)$ for all $s \in S$.

Policy Iteration Step (2) via "Value Iteration"

- Assume the current policy $\pi(s)$, $s \in S$
- We want to determine the values $V^{(\pi)}(s)$ that solve the system, cf. Step (2)

$$V^{(\pi)}(s) = \sum_{i \in I} P(i, \pi(s), s) \cdot \left(r(i, \pi(s), s) + \gamma \cdot V^{(\pi)}\left(\Gamma(i, \pi(s), s)\right)\right), \ \forall s \in S$$

- **Iterative solution** (value iteration **with fixed *a***) starting with $V_T^{(\pi)}(s) = 0$:

$$V_t^{(\pi)}(s) = \max_{\substack{a \in A: \\ a = \pi(s)}} \left\{ \sum_{i \in I} P(i, a, s) \cdot \left(r(i, a, s) + \gamma \cdot V_{t+1}^{(\pi)}\left(\Gamma(i, a, s)\right)\right)\right\}$$

Policy Iteration Step (2) via "Value Iteration"

- Assume the current policy $\pi(s)$, $s \in S$
- We want to determine the values $V^{(\pi)}(s)$ that solve the system, cf. Step (2),

$$V^{(\pi)}(s) = \sum_{i \in I} P(i, \pi(s), s) \cdot \left(r(i, \pi(s), s) + \gamma \cdot V^{(\pi)}\left(\Gamma(i, \pi(s), s)\right)\right), \ \forall s \in S$$

- **Iterative solution** (value iteration **with fixed *a***) starting with $V_T^{(\pi)}(s) = 0$:

$$V_t^{(\pi)}(s) = \sum_{i \in I} P(i, \pi(s), s) \cdot \left(r(i, \pi(s), s) + \gamma \cdot V_{t+1}^{(\pi)}\left(\Gamma(i, \pi(s), s)\right)\right)$$

- For "large" T and for any initial $V_T^{(\pi)}(s)$ the values $V_0(s)$ converge to the exact values $V^{(\pi)}(s)$ with $\left|V_0^{(\pi)}(s) - V^{(\pi)}(s)\right| \xrightarrow{T \to \infty} 0$

Policy Iteration Step (2) via Linear Programming

- Assume a step k's current policy $\pi(s)$, $s \in S$
- We want to determine the values $V^{(\pi)}(s)$ that solve the system, cf. Step (2),

$$V^{(\pi)}(s) = \sum_{i \in I} \underbrace{P(i, \pi(s), s)}_{probability} \cdot \left(\underbrace{r(i, \pi(s), s)}_{period's\ reward} + \underbrace{\gamma \cdot V^{(\pi)}\left(\underbrace{\Gamma(i, \pi(s), s)}_{s'} \right)}_{disc.\ exp.\ future\ profits\ of\ new\ state} \right), \forall s \in S$$

- We have to solve a system of $|S|$ linear equations! (*not in focus*)
- Standard LP solvers can be applied, e.g., Gurobi, Cplex, etc.

Summary (Solving Discrete Time Infinite Horizon MDPs)

Policy Iteration

(+) provides optimal solutions for infinite horizon MDPs
(+) fast convergence (stop if no improvement is possible)
(–) full information required (cf. events & transitions)
(–) only for smaller state spaces

Value Iteration

(+) provides near-optimal solutions for infinite horizon MDPs
(+) numerically simple
(+) fast convergence (if the discount factor is not close to 1)
(–) full information required (cf. events & transitions)
(–) for medium size state spaces, does not scale (curse of dimensionality)

Recall - Questions?

- States, Action, Rewards, Transitions
- Discounted Future Rewards
- Value Function
- Bellman Equation
- Backward Induction
- Value Iteration
- Policy Iteration

Could You Solve Different Test Problems?

- Finite Horizon (*use Backward Induction*)
  - Eating cake (deterministic utility)
  - Selling Airline Tickets (stochastic demand)
- Infinite Horizon (*use Value & Policy Iteration*)
  - Car replacement problem (deterministic costs)
  - Inventory management (stochastic demand)

Problem Embedding (Finite as Infinite Horizon MDP)

- Finite Horizon (time-dependent) with discount $\gamma \leq 1$
  - $r_t = r_t(i,a,s)$ with probabilities $P_t(i,a,s)$, $s \in S$, $t = 0,1,2,...,T$
  - $s_t \rightarrow s_{t+1} = \Gamma_t(i,a,s)$ with $i \in I_t(a,s)$, $a \in A_t(s)$

- Infinite Horizon Value It. Embedding (time-independent) with same $\gamma \leq 1$
  - $\tilde{s}_0 := (0, s_0)$, $\tilde{s} := (t,s)$, $s \in S$, $t = 0,1,2,...,T$
  - $\tilde{r} = \tilde{r}(i,a,\tilde{s}) = \tilde{r}(i,a,t,s) := \text{if } t < T \text{ then } r_t(i,a,s) \text{ else } r_T(s)$
  - $\tilde{P}(i,a,\tilde{s}) := P_t(i,a,s)$, $a \in \tilde{A}(\tilde{s}) := A_t(s)$, $i \in \tilde{I}(a,\tilde{s}) := I_t(a,s)$
  - $\tilde{s} \rightarrow \tilde{s}' = \tilde{\Gamma}(i,a,\tilde{s}) = \tilde{\Gamma}(i,a,t,s) := \left(t+1, \text{ if } t < T \text{ then } \Gamma_t(i,a,s) \text{ else } s\right)$
  - $V_k(t,s) = \max_{a \in \tilde{A}} \left\{ \sum_{i \in \tilde{I}} \tilde{P}(i,a,t,s) \cdot \left(\tilde{r}(i,a,t,s) + \gamma \cdot 1_{\{t<T\}} \cdot V_{k+1}\left(\tilde{\Gamma}_t(i,a,s)\right)\right)\right\}$, i.e. $V_k(t,s) = 0 \ \forall t > T$

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 “Pfingstmontag”)
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	Final Presentations	
	Aug 31	Finish Documentation	

Exercise (Mandatory) Value Iteration & Policy Iteration (until April 28)

Consider the inventory management example. Maximize expected discounted rewards. Find an optimal ordering policy. Items ordered at the beginning of a period are delivered at the beginning of the same period; holding cost only occur for items left at the beginning of the period. If demand exceeds the inventory (at the beginning of the period) then sales are lost, cf. $r(i,a,s) = p \cdot \min(i,s) - c \cdot a - h \cdot s - 1_{\{a>0\}} \cdot f$, where $p = 10, c = 2, h = 0.5, f = 20$. Demand probabilities are $P(i,a,s) = P(i) := 1/4$, $i = 0,1,...,3$. Consider the state space $s \in S = \{0,1,...,50\}$. Feasible ordering decisions are $a \in A := \{0,1,...,50\}$. The discount factor is $\gamma = 0.95$. The initial state is 10 items.

(a) Solve the problem using value iteration.

(b) Solve the problem using policy iteration.

(c) Simulate the optimal policy $\pi(s)$ over runs of 100 periods and accumulate total rewards discounted on *t*=0. Compare the mean with the value function for the initial state (Bonus).