

INTRODUCTION TO MARKOV DECISION PROCESSES

Balázs Csanád Csáji

Research Fellow, The University of Melbourne

Signals & Systems Colloquium, 29 April 2010

Department of Electrical and Electronic Engineering, The University of Melbourne

Overview

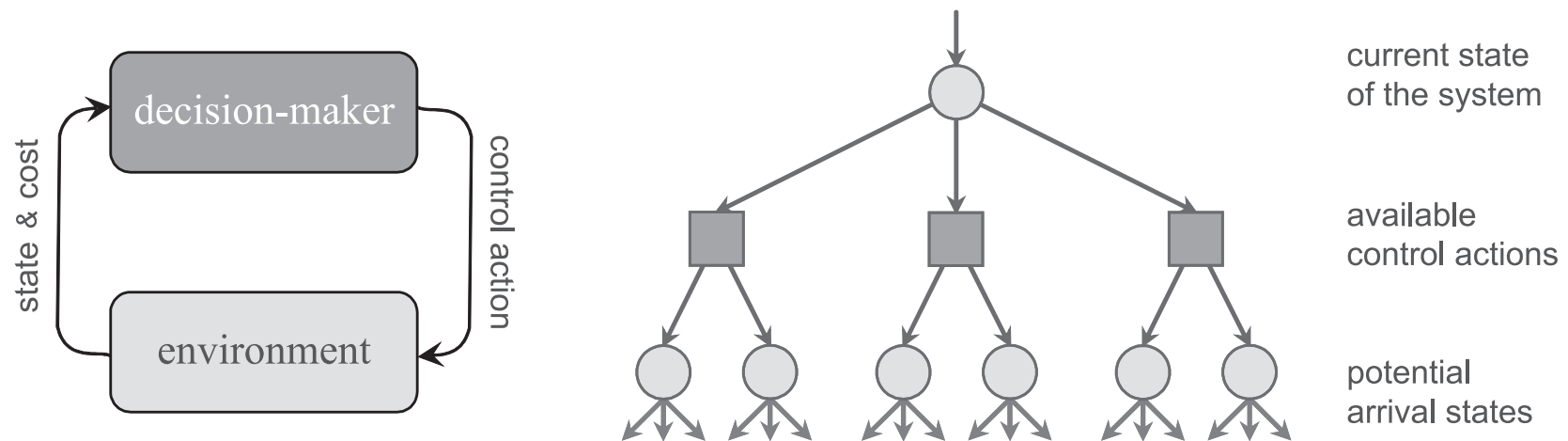
- PART I. **Introduction**
(Motivations, Applications and Prerequisites)
- PART II. **The Basic Problem**
(Policies, Value Functions and Optimality Equations)
- PART III. **Solution Methods**
(Successive Approximations, Direct Policy Search, and Linear Programming)
- PART IV. **Generalizations**
(Unbounded Costs, Partially Observable Problems, and General Measurable Spaces)
- PART V. **Summary and Literature**

PART I.

Introduction

Motivation: Reinforcement Learning

- **Reinforcement learning** (RL) is a computational approach to learn from the interaction with an environment based on feedbacks, e.g., rewards.
- An interpretation: consider an **agent** acting in an uncertain environment and receiving information on the actual states and immediate costs.
- The aim is to **learn** an efficient **behavior** (control policy), such that applying this strategy minimizes the expected costs in the long run.



Applications

Some **applications** of Markov decision processes:

- Optimal Stopping
- Routing
- Maintenance and Repair
- Dispatching & Scheduling
- Inventory Control
- Optimal Control of Queues
- Strategic Asset Pricing
- Dynamic Options
- Insurance Risk Management
- Robot Control
- Logic Games
- Communication Networks
- Dynamic Channel Allocation
- Power Grid Management
- Supply-Chain Management
- Stochastic Resource Allocation
- Sequential Clinical Trials
- PageRank Optimization

Reminder: Markov Chains

- Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space with a nondecreasing family of σ -algebras, called a **filtration**, $\mathcal{F}_0 \subseteq \mathcal{F}_1 \subseteq \dots \subseteq \mathcal{F}$.
- Note that \mathcal{F}_t can be interpreted as the information available at time t .
- A stochastic sequence $X = (X_i, \mathcal{F}_i)_{i=0}^{\infty}$ is a **Markov chain** (w.r.t. \mathbb{P}) if

$$\mathbb{P}(X_i \in A \mid \mathcal{F}_j) = \mathbb{P}(X_i \in A \mid X_j),$$

for all $0 \leq j \leq i$ and $A \in \mathcal{B}(\mathbb{X})$ (\mathbb{P} -a.s.), where \mathbb{X} is the **state space** of the process and $\mathcal{B}(\cdot)$ denotes a σ -algebra.

- For example, $\mathbb{X} = \mathbb{R}$ and $\mathcal{B}(\mathbb{X})$ denotes the Borel measurable sets.
- For **countable** state spaces, for example $\mathbb{X} \subseteq \mathbb{Q}^d$, the σ -algebra $\mathcal{B}(\mathbb{X})$ will be assumed to be the set of all subsets of \mathbb{X} .

Countable State Spaces

- Henceforth we assume that \mathbb{X} is **countable** and $\mathcal{B}(\mathbb{X}) = \mathcal{P}(\mathbb{X}) (= 2^{\mathbb{X}})$.
- We say that $\lambda = (\lambda_x : x \in \mathbb{X})$ is a **distribution** if $\lambda_x \geq 0$ for all x and

$$\sum_{x \in \mathbb{X}} \lambda_x = 1.$$

- A (potentially infinite) matrix $P = (p_{xy} : x, y \in \mathbb{X})$ is **stochastic** if its each row $(p_{xy} : y \in \mathbb{X})$ is a distribution.
- A sequence of discrete random variables $(X_i)_{i=0}^{\infty}$ is a (homogeneous) **Markov chain** with **initial distribution** λ and **transition matrix** P if
 - X_0 has distribution λ ;
 - For all $i \geq 0$, conditional on $X_i = x$, X_{i+1} has distribution $(p_{xy} : y \in \mathbb{X})$ and is independent of X_0, \dots, X_{i-1} .

Transition Probabilities

- We can easily extend the matrix **multiplication** to the general case:

$$(\lambda P)_y \triangleq \sum_{x \in \mathbb{X}} \lambda_x p_{xy}, \quad (PQ)_{xz} \triangleq \sum_{y \in \mathbb{X}} p_{xy} q_{yz},$$

for all (potentially infinite) vector λ and matrices P, Q .

- The (generalized) **identity** matrix is denoted by I , where $I_{xy} = \delta_{xy}$.
- Matrix **powers** can be defined as usual, $P^0 \triangleq I$ and $P^{n+1} \triangleq P^n P$.
- If $(X_i)_{i=0}^{\infty}$ is a (discrete, homogeneous) **Markov chain** then

$$- \mathbb{P}(X_n = x) = (\lambda P^n)_x,$$

$$- \mathbb{P}(X_{m+n} = y \mid X_m = x) = (P^n)_{xy},$$

where λ is its initial distribution and P is its transition matrix.

PART II.

The Basic Model of Markov Decision Processes

Markov Decision Processes

A (homogeneous, discrete, observable) **Markov decision process** (MDP) is a stochastic system characterized by a 5-tuple $\mathcal{M} = \langle \mathbb{X}, \mathbb{A}, \mathcal{A}, p, g \rangle$, where:

- \mathbb{X} is a countable set of discrete **states**,
- \mathbb{A} is a countable set of control **actions**,
- $\mathcal{A} : \mathbb{X} \rightarrow \mathcal{P}(\mathbb{A})$ is an **action constraint** function,
 $\mathcal{A}(x)$ denotes the finite set of allowed actions in state x ,
- $p : \mathbb{X} \times \mathbb{A} \rightarrow \Delta(\mathbb{X})$ is the **transition** function, $p(y \mid x, a)$ denotes the probability of arriving at state y after taking action $a \in \mathcal{A}(x)$ in a state x ,
- $g : \mathbb{X} \times \mathbb{A} \rightarrow \mathbb{R}$ is an **immediate cost** (or reward) function.

Note that $\Delta(\mathbb{X})$ is the set of all probability distributions on \mathbb{X} .

Control Policies

The **policy** defines which action to take depending on the history:

$$\pi_n : \mathbb{X} \times (\mathbb{A} \times \mathbb{X})^n \rightarrow \Delta(\mathbb{A}).$$

Unless indicated otherwise, we consider **stationary, Markov** policies:

- A **deterministic** (stationary, Markov) policy is $\pi : \mathbb{X} \rightarrow \mathbb{A}$.
- A **randomized** (stationary, Markov) policy is $\pi : \mathbb{X} \rightarrow \Delta(\mathbb{A})$.

Each policy induces a (stochastic) **transition matrix** on the state space:

$$(P_\pi)_{xy} \triangleq \sum_{a \in \mathcal{A}(x)} p(y | x, a) \pi(x, a).$$

The initial distribution of the states $\lambda \in \Delta(\mathbb{X})$, the transition probabilities p , together with a policy π define a homogeneous **Markov chain** on \mathbb{X} .

Performance Measures: Total Cost

- We aim at finding a policy that minimizes the costs in the long run.
- A common performance measure is the **expected discounted cost**.
- In this case, the **value function** of a policy $V^\pi : \mathbb{X} \rightarrow \mathbb{R}$ is defined as

$$V^\pi(x) \triangleq \mathbb{E}_\pi \left[\sum_{n=0}^{\infty} \beta^n g(X_n, A_n) \mid X_0 = x \right],$$

where $\beta \in [0, 1)$ is a **discount factor** and A_n, X_n are random variables with $A_n \sim \pi(X_n)$ and $X_{n+1} \sim p(X_n, A_n)$ (“ \sim ” = “has distribution”).

- Function V^π is **well-defined** (and finite) if, e.g., the immediate-costs are **bounded**: for all $x \in \mathbb{X}$ and $a \in \mathcal{A}(x)$, we have $|g(x, a)| \leq C$.
- In this latter case, for all state x , we have $|V^\pi(x)| \leq C/(1 - \beta)$.

Performance Measures: Ergodic Cost

- An alternative performance measure is the **expected ergodic cost**; in this case, the **value function** of a policy $W^\pi : \mathbb{X} \rightarrow \mathbb{R}$ is defined as

$$W^\pi(x) \triangleq \limsup_{k \rightarrow \infty} \frac{1}{k} \mathbb{E}_\pi \left[\sum_{n=0}^{k-1} g(X_n, A_n) \mid X_0 = x \right].$$

where, as previously, A_n, X_n are random variables representing the state and action at time n , with $A_n \sim \pi(X_n)$ and $X_{n+1} \sim p(X_n, A_n)$.

- If the state space \mathbb{X} is finite or the costs are bounded, W is finite.
- If \mathbb{X} is **infinite**, an optimal stationary Markov policy may not exist.
- In many applications the average cost is independent of the **initial state**, namely, $W^\pi(x)$ is constant (e.g., for finite, irreducible chains).

Optimal Solutions

- In general, we search for a solution that has minimal cost over all policies, with respect to a given **performance measure** $\mu(x, \pi)$.
- For example, $\mu(x, \pi) = V^\pi(x)$ or $\mu(x, \pi) = W^\pi(x)$.
- The **optimal value function** is defined as

$$\mu^*(x) \triangleq \inf_{\pi \in \Pi} \mu(x, \pi),$$

for all state $x \in \mathbb{X}$, where Π denotes the set of all admissible policies.

- For $\varepsilon \geq 0$, a policy is called **ε -optimal**, if $\mu(x, \pi) \leq \mu^*(x) + \varepsilon$, for all state $x \in \mathbb{X}$. A 0-optimal policy is called **optimal**.
- Henceforth, we apply the discounted cost criterion, $\mu(x, \pi) = V^\pi(x)$.

Optimality Operators

- Let $\mathcal{B}(\mathbb{X}, \mathbb{R})$ denote the set of all **bounded** real-valued functions on \mathbb{X} with bound $\|g\| / (1 - \beta)$. It contains the value functions of all policies.
- For a function $f \in \mathcal{B}(\mathbb{X}, \mathbb{R})$ we consider two **cost operators**:

$$(P_a f)(x) \triangleq \mathbb{E} [f(X_1) \mid X_0 = x, A_0 = a],$$

$$(T_a f)(x) \triangleq g(x, a) + \beta(P_a f)(x).$$

- The **optimality operators** are defined for all state $x \in \mathbb{X}$ by:

$$(P f)(x) \triangleq \inf_{a \in \mathcal{A}(x)} (P_a f)(x),$$

$$(T f)(x) \triangleq \inf_{a \in \mathcal{A}(x)} (T_a f)(x).$$

- Operators T_a and T are often called the **Bellman operators**.

Optimality Equations

- The Bellman **optimality equation** for the discounted cost problem is:

$$V(x) = (T V)(x),$$

for all state $x \in \mathbb{X}$. Its solution V^* is the optimal value function.

- It can be proven that T is a **contraction** with **Lipschitz constant** β :

$$\|T f_1 - T f_2\| \leq \beta \|f_1 - f_2\|.$$

for all $f_1, f_2 \in \mathcal{B}(\mathbb{X}, \mathbb{R})$, where $\|\cdot\|$ denotes supremum norm.

- Using the Banach fixed point theorem we get that T has a **unique fixed point**. Thus, all optimal policies share the same value function.
- Moreover, T is **monotone**, namely, $f_1 \leq f_2 \Rightarrow T f_1 \leq T f_2$.

Greedy Policies

- For **countable** state spaces the Bellman operator takes the form:

$$(TV)(x) = \min_{a \in \mathcal{A}(x)} \left[g(x, a) + \beta \sum_{y \in \mathbb{X}} p(y | x, a) V(y) \right].$$

- Given a value function V , the **greedy** policy (w.r.t. V) is defined as

$$\pi(x) \in \arg \min_{a \in \mathcal{A}(x)} \left[g(x, a) + \beta \sum_{y \in \mathbb{X}} p(y | x, a) V(y) \right].$$

- If $V \in \mathcal{B}(\mathbb{X}, \mathbb{R})$ and π is a greedy policy with respect to V , then

$$\|V^* - V^\pi\| \leq \frac{2\beta}{1 - \beta} \|V - V^*\|,$$

- Thus, good **approximations** to V^* directly lead to efficient policies.

PART III.

Solution Methods

Three Ways

The three classical ways of solving MDPs are:

1. **Successive approximations**: iteratively computing a value function that approximates V^* . For example, value iteration and Q-learning.
2. **Direct policy search**: searching for an optimal control policy in the space of policies. For example, policy iteration and policy gradient methods.
3. **Linear programming**: finding the optimal value function as a solution of a static optimization problem with linear objective function and constraints.

TYPE 1 SOLUTIONS

Successive Approximations
(Value Function Based Methods)

Value Iteration

- Since T is a contraction, the **recursive sequence**

$$V_{n+1} = T V_n$$

converges to V^* for any initial value function $V_0 \in \mathcal{B}(\mathbb{X}, \mathbb{R})$.

- This method is called **value iteration**. Its main **problems** are:
 1. If \mathbb{X} is very large $\Rightarrow T V$ cannot be computed for all states at once.
 2. The transition probabilities $p(y | x, a)$ are often not known in advance.
 3. If \mathbb{X} is very large $\Rightarrow V$ cannot be directly stored in the memory.
- A possible solution to problem 1 is **asynchronous** value iteration:

$$\tilde{V}_{n+1}(x) = (T \tilde{V}_n)(x)$$
 is updated only for $x \in \mathbb{X}_{n+1} \subseteq \mathbb{X}$.
- \tilde{V}_n converges to V^* if each state is updated infinite many times.

Q-learning

- **Simulation** based methods can handle unknown transition probabilities.
- The **action-value function** of a policy $Q^\pi : \mathbb{X} \times \mathbb{A} \rightarrow \mathbb{R}$ is

$$Q^\pi(x, a) = \mathbb{E} \left[\sum_{t=0}^{\infty} \beta^t g(X_t, A_t) \mid X_0 = x, A_0 = a \right].$$

- The **optimal** action-value function, which uniquely exists, is Q^* .
- The one-step version of Watkins' **Q-learning** rule is as follows.

$$Q_{n+1}(x, a) = (1 - \gamma_n(x, a))Q_n(x, a) + \gamma_n(x, a) \beta \min_{B \in \mathcal{A}(Y)} Q_n(Y, B),$$

where $\gamma_n \in (0, 1)$ is the **learning rate**, e.g., $\gamma_n(x, a) = 1/n$, and Y is a random variable generated from $(x, a) \in \mathbb{X} \times \mathbb{A}$ by simulation.

Q-learning

- Q-learning is **off-policy**: it can work independently of the applied policy.
- Assume that the **learning rates** satisfy for all state x and action a (w.p.1):

$$\sum_{n=0}^{\infty} \gamma_t(x, a) < \infty, \quad \sum_{n=0}^{\infty} \gamma_t^2(x, a) = \infty.$$

- Under this assumption and if all state-action pairs are continue to update, the sequence Q_t **converges** to Q^* almost surely for all Q_0 .
- Policy π is called **soft** if $\forall x \in \mathbb{X} : \forall a \in \mathcal{A}(x) : \pi(x, a) > 0$, e.g.,

$$\pi_n(x, a) \triangleq \frac{\exp(-Q_n(x, a)/\tau)}{\sum_{b \in \mathcal{A}(x)} \exp(-Q_n(x, b)/\tau)},$$

where $\tau > 0$ is the **Boltzmann** temperature. It is a **semi-greedy** policy.

Stochastic Iterative Algorithms

- Many learning and optimization methods can be written in a general form as a **stochastic iterative algorithm**. More precisely, for all $z \in \mathcal{Z}$ as

$$f_{t+1}(z) = (1 - \gamma_t(z))f_t(z) + \gamma_t(z)((K f_t)(z) + W_t(z)),$$

where f_t is a (generalized) value function, operator K acts on value functions, γ_t denotes random stepsizes and W_t is a noise parameter.

- **Approximate dynamic programming** methods often take the form

$$\Phi(r_{t+1}) = \Pi((1 - \gamma_t) \Phi(r_t) + \gamma_t(K_t(\Phi(r_t)) + W_t)),$$

where $r_t \in \Theta$ is a **parameter**, Θ is the parameter space, e.g., $\Theta \subseteq \mathbb{R}^d$, $\Phi : \Theta \rightarrow \mathcal{F}$ is an approximation function where $\mathcal{F} \subseteq \mathcal{V}$ is a Hilbert space of functions. Function $\Pi : \mathcal{V} \rightarrow \mathcal{F}$ is a **projection** mapping.

TYPE 2 SOLUTIONS

Direct Policy Search

Policy Iteration

- Let π_1 be a policy with value function V^{π_1} . Let π_2 be the greedy policy w.r.t V^{π_1} , then $V^{\pi_2} \leq V^{\pi_1}$ and $V^{\pi_2} = V^{\pi_1} \Rightarrow$ they are both optimal.
- The method of **policy iteration** works by iteratively **evaluating** the policy then **improving** it by the greedy policy with respect to the evaluation.

1. **Initialize** π_0 arbitrarily and **set** the iteration counter, $n := 0$.
2. **Repeat** (iterative evaluation and improvement)
3. **Evaluate** policy π_n by computing its value function V^{π_n} .
4. V^{π_n} is the solution of the linear system $(I - \beta P_{\pi_n}) x = g_{\pi_n}$.
5. **Improve** the policy by setting π_{n+1} to the greedy policy w.r.t. V^{π_n} .
6. **Increase** the iteration counter, $n := n + 1$.
7. **Until** $\pi_n = \pi_{n-1}$ (until no more improvements are possible)

- If \mathbb{X} is finite, it **terminates** in finite steps with an optimal policy.

Policy Evaluation

- Simulation based methods, e.g., **temporal difference** learning, can evaluate a policy without knowledge of the transition probabilities.
- The update rule of **TD(0)** is defined as

$$V_{n+1}(x_n) = V_n(x_n) + \gamma_n(x_n) \left(g(x_n, a_n) + \beta V_n(x_{n+1}) - V_n(x_n) \right),$$

where the trajectory $x_0, a_0, x_1, a_1, \dots$ is generated by **Monte Carlo** simulation and $\gamma_n(\cdot)$ denote the **learning rate**.

- Assuming the usual properties about $\gamma_n(\cdot)$ and every state is visited infinitely often, V_n **converges** to V^π (w.p.1) starting from $V_0 \equiv 0$.
- The **optimistic** policy iteration with TD(0) converges to V^* (w.p.1).

Policy Gradient

- Assuming that the policy π is **parametrized** by $\theta \in \mathbb{R}^d$: $a \sim \pi_\theta(x, a)$, the **policy gradient** method with **learning rate** γ_n (with $V(\theta) \triangleq V^{\pi_\theta}$)

$$\theta_{n+1} = \theta_n - \gamma_n \nabla_\theta V(\theta)|_{\theta=\theta_n}.$$

- If the gradient estimator is **unbiased** and $\sum_n \gamma_n = \infty$, $\sum_n \gamma_n^2 < \infty$, the convergence to a **local minimum** can be guaranteed.
- In order to estimate the gradient, θ can be **perturbed** $\hat{\theta}_i = \theta_n + \delta\theta_i$.
- In this case, the gradient can be determined by **linear regression**:

$$\nabla_\theta V(\theta)|_{\theta=\theta_n} \approx (\Delta\Theta^T \Delta\Theta)^{-1} \Delta\Theta^T \Delta J,$$

with $\Delta\Theta \triangleq (\delta\theta_1, \dots, \delta\theta_k)^T$ and $\delta J_i \triangleq J(\hat{\theta}_i) - J(\theta_n)$ **rollout** returns form $\Delta J \triangleq (\delta J_1, \dots, \delta J_k)^T$, where k is the **sample size**.

Policy Gradient

- The **advantages** of policy gradient methods are:
 - They allow incorporation of **domain knowledge** in the parametrization.
 - Often significantly **fewer parameters** are enough to represent the optimal policy than the corresponding value function.
 - They are guaranteed to **converge** while successive approximation methods with function approximation often does not.
 - They can handle **continuous** state and action spaces, as well.
- The **disadvantages** of policy gradient methods are:
 - They only converge to a **local optima**, not to a global one.
 - They are difficult to use in **off-policy** settings.
 - The convergence rate is often slow in **discrete** problems.

TYPE 3 SOLUTIONS

Linear Programming

Linear Programming

- We know that the **optimal value function** satisfies $T V^* = V^*$, it is monotone, $V_1 \leq V_2 \Rightarrow T V_1 \leq T V_2$, and $T^n V \rightarrow V^*$ as $n \rightarrow \infty$.
- Therefore, V^* is the **largest** vector satisfying $T V \leq V$.
- The optimal value function, V^* , solves the following **linear program** (LP):

$$\begin{array}{ll}
 \text{maximize} & \sum_{x \in \mathbb{X}} \lambda_x \\
 \text{subject to} & \lambda_x \leq g(x, a) + \beta \sum_{y \in \mathbb{X}} p(y | x, a) \lambda_y
 \end{array}$$

for all state x and action $a \in \mathcal{A}(x)$. If its solution is λ^* , then $\lambda^* = V^*$.

- If \mathbb{X} is finite, it can be solved efficiently by, e.g., **interior point** methods. In case of infinite \mathbb{X} , it can be often **approximated** by a finite program.

Computational Complexity

- Assume that the MDP is **finite**, particularly $|\mathbb{X}| = n$ and $|\mathbb{A}| = m$.
- It is **equivalent** with an LP with n variables and $O(nm)$ constraints.
- Thus, it can be solved in **polynomial** time, assuming the Turing model.
- Reducing to an MDP is often useful in **combinatorial optimization**, if we want to show the polynomial computability of a class of problems.
- If the discount factor, β , is **fixed**, value iteration and policy iteration also runs in polynomial time. However, value iteration is not polynomial in β .
- It is not known whether MDPs can be solved in **strongly polynomial** time. (Naturally, it is also not known whether LPs can be solved this way.)

PART IV.

Generalizations

Potential Criticisms

- Some possible **questions** about the presented paradigm:
 1. Isn't the **Markov assumption** too restrictive?
 2. Can this methodology deal with **delayed** costs (or rewards)?
 3. What if the immediate costs are **not bounded**?
 4. Can this theory be extended to system that are **not fully observed**?
 5. What can we do if the state and action spaces are **uncountable**?
- Regarding 1: note that Semi-MDPs can be reduced to MDPs.
- Regarding 2: delayed costs do not introduce change to the theory.
- We are going to investigate 3, 4 and 5 now.

Unbounded Costs

- So far we have assumed that the costs are **bounded**: $\|g\| \leq C$.
- A **generalization**: it is enough to assume that for all state x there is a number C_x and a constant k such that for all policy π , we have

$$\mathbb{E}_\pi [|g(X_{n-1}, A_{n-1})| \mid X_0 = x] \leq C_x n^k.$$

- Under the condition above, the value function of policy π satisfy

$$\mathbb{E}_\pi \left[\sum_{n=0}^{\infty} \beta^n g(X_n, A_n) \mid X_0 = x \right] \leq C_x \sum_{n=0}^{\infty} \beta^n (n+1)^k < \infty,$$

therefore, the value function V^π remains **well-defined** (and finite).

- Alternatively, for most of the theory it is enough if we assume that g is bounded (only) from **below**: $g(x, a) \geq C$ for all state x and action a .

Partial Observability

- A **partially observable** Markov decision process (POMDP) is an MDP where the states cannot be observed directly.
- For POMDPs the definition of MDPs is extended with an **observation set** \mathbb{O} and an **observation probability** function: $q : \mathbb{X} \times \mathbb{A} \rightarrow \Delta(\mathbb{O})$.
- $q(z \mid x, a)$ is the probability of observing z if we take action a in state x .
- The **action constraint** function takes the form: $\mathcal{A} : \mathbb{O} \rightarrow \mathcal{P}(\mathbb{A})$.
- The **policies** also depend on the observations instead states, e.g., a randomized Markov policy takes the form of $\pi : \mathbb{O} \rightarrow \Delta(\mathbb{A})$.
- In general, policies should depend on the whole **history** to be optimal.

Belief States

- **Belief states** are probability distributions over the states, $\mathbb{B} \triangleq \Delta(\mathbb{X})$. They can be interpreted as the agent's **ideas** about the current state.
- Given a belief state b , an action a and an observation z , the new belief state $\tau(b, a, z)$ can be computed by **Bayes rule**:

$$\tau(b, a, z)(y) = \frac{\sum_{x \in \mathbb{X}} p(z, y | x, a) b(x)}{p(z | b, a)},$$

where $p(z, y | x, a) \triangleq q(z | y, a)p(y | x, a)$ and

$$p(z | b, a) \triangleq \sum_{x, y \in \mathbb{X}} p(z, y | x, a) b(x).$$

- They are **sufficient statistics**: an optimal policy can found based on them.

Belief State MDPs

- A POMDP can be **transformed** into a fully observable MDP.
- The new MDP is called the **belief state MDP**. Its **state space** is \mathbb{B} , the **action space** remains \mathbb{A} and the **transition probabilities** are:

$$p(b_2 | b_1, a) = \begin{cases} p(z | b_1, a) & \text{if } b_2 = \tau(b_1, a, z) \text{ for some } z \\ 0 & \text{otherwise} \end{cases}$$

The new **immediate cost** function for all $b \in \mathbb{B}$, $a \in \mathbb{A}$ is given by

$$g(b, a) = \sum_{x \in \mathbb{X}} b(x) g(x, a),$$

consequently, the **optimal value function** of the belief state MDP is

$$\tilde{V}^*(b) = \min_{a \in \mathcal{A}(b)} \left[g(b, a) + \beta \sum_{z \in \mathbb{O}} p(z | b, a) \tilde{V}^*(\tau(b, a, z)) \right].$$

General Measurable Spaces

In case the state \mathbb{X} or the action space \mathbb{A} is **non-discrete**, we need:

- The state space \mathbb{X} is measurable space endowed with a σ -field \mathcal{X} .
- The action space \mathbb{A} is measurable space endowed with a σ -field \mathcal{A} .
- For all state $x \in \mathbb{X}$, the set of allowed actions $\mathcal{A}(x)$ is measurable.
- The cost function g is a measurable function on $(\mathbb{X} \times \mathbb{A}, \mathcal{X} \times \mathcal{A})$.
- $p(\cdot | \cdot)$ is a transition probability from $(\mathbb{X} \times \mathbb{A}, \mathcal{X} \times \mathcal{A})$ to $(\mathbb{X}, \mathcal{X})$:
 - For all x, a : $p(\cdot | x, a)$ is a probability measure on $(\mathbb{X}, \mathcal{X})$ and,
 - For all E : $p(E | \cdot)$ is a measurable function on $(\mathbb{X} \times \mathbb{A}, \mathcal{X} \times \mathcal{A})$.

For **discrete** MDPs, these assumptions always hold, since in that case $\mathcal{X} \triangleq \mathcal{P}(\mathbb{X})(= 2^{\mathbb{X}})$ and $\mathcal{A} \triangleq \mathcal{P}(\mathbb{A})(= 2^{\mathbb{A}})$.

PART V.

Summary and Literature

Summary

- Markov decision processes are **controlled Markov chains** together with an **immediate-cost** function and an **optimization criterion**.
- They have a large number of practical and theoretical **applications**.
- The **basic concepts** are: policy, value function and optimality equations.
- The three classical types of solution methods are:
 1. **Successive Approximations** (e.g., value iteration, Q-learning)
 2. **Direct Policy Search** (e.g., policy iteration, policy gradient)
 3. **Linear Programming** (e.g., interior point, ellipsoid or simplex methods)
- The theory can handle semi-Markov and partially observable problems and can be **generalized** to general measurable state and action spaces.

Recommended Literature

- Puterman M. L.: **Markov Decision Processes**, Wiley-Interscience, 1994
- Bertsekas D. P. & Tsitsiklis J. N.: **Neuro-Dynamic Programming**, Athena Scientific, Belmont, MA, 1996
- Sutton, R. S. & Barto, A. G.: **Reinforcement Learning: An Introduction**, MIT Press, Cambridge, MA, 1998
- Feinberg, A. E. & Shwartz, A. (eds.): **Handbook of Markov Decision Processes**, Kluwer Academic Publishers, 2002
- Bertsekas D. P.: **Dynamic Programming and Optimal Control**, 3rd edition, Vol. I. & II., Athena Scientific, Belmont, MA, 2005 & 2007

Thank you for your attention!

`bcsaji@unimelb.edu.au`