Topics in Reinforcement Learning: Rollout and Approximate Policy Iteration

ASU, CSE 691, Spring 2020

Dimitri P. Bertsekas
dbertsek@asu.edu

Lecture 2
Outline

1. Review of Exact Deterministic DP Algorithm
2. Examples: Finite-State/Discrete/Combinatorial DP Problems
3. Stochastic DP Algorithm
4. Infinite Horizon - Briefly
5. Problem Formulations and Examples
Finite Horizon Deterministic Problem

- System
  \[ x_{k+1} = f_k(x_k, u_k), \quad k = 0, 1, \ldots, N - 1 \]
  where \( x_k \): State, \( u_k \): Control chosen from some set \( U_k(x_k) \)

- Cost function:
  \[
  g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k)
  \]

- For given initial state \( x_0 \), minimize over control sequences \( \{u_0, \ldots, u_{N-1}\} \)
  \[
  J(x_0; u_0, \ldots, u_{N-1}) = g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k)
  \]

- Optimal cost function \( J^*(x_0) = \min_{u_k \in U_k(x_k)} \sum_{k=0}^{N-1} J(x_0; u_0, \ldots, u_{N-1}) \)
DP Algorithm: Solving Progressively Longer Tail Subproblems

Go backward to compute the optimal costs $J^*_k(x_k)$ of the $x_k$-tail subproblems

Start with

$$J^*_N(x_N) = g_N(x_N), \quad \text{for all } x_N,$$

and for $k = 0, \ldots, N - 1$, let

$$J^*_k(x_k) = \min_{u_k \in U_k(x_k)} \left[ g_k(x_k, u_k) + J^*_{k+1}(f_k(x_k, u_k)) \right], \quad \text{for all } x_k.$$

Then optimal cost $J^*(x_0)$ is obtained at the last step: $J^*_0(x_0) = J^*(x_0)$.

Go forward to construct optimal control sequence $\{u_0^*, \ldots, u_{N-1}^*\}$

Start with

$$u_0^* \in \arg \min_{u_0 \in U_0(x_0)} \left[ g_0(x_0, u_0) + J^*_1(f_0(x_0, u_0)) \right], \quad x_1^* = f_0(x_0, u_0^*).$$

Sequentially, going forward, for $k = 1, 2, \ldots, N - 1$, set

$$u_k^* \in \arg \min_{u_k \in U_k(x_k^*)} \left[ g_k(x_k^*, u_k) + J^*_{k+1}(f_k(x_k^*, u_k)) \right], \quad x_{k+1}^* = f_k(x_k^*, u_k^*).$$

Approximation in value space approach: We replace $J^*_k$ with an approximation $\tilde{J}_k$. 
Finite-State Problems: Shortest Path View

- Nodes correspond to states $x_k$
- Arcs correspond to state-control pairs $(x_k, u_k)$
- An arc $(x_k, u_k)$ has start and end nodes $x_k$ and $x_{k+1} = f_k(x_k, u_k)$
- An arc $(x_k, u_k)$ has a cost $g_k(x_k, u_k)$. The cost to optimize is the sum of the arc costs from the initial node $s$ to the terminal node $t$.
- **The problem is equivalent to finding a minimum cost/shortest path from $s$ to $t$.**
Discrete-State Deterministic Scheduling Example

Find optimal sequence of operations A, B, C, D (A must precede B and C must precede D)

DP Problem Formulation

- States: Partial schedules; Controls: Stage 0, 1, and 2 decisions; Cost data shown along the arcs
- Recall the DP idea: Break down the problem into smaller pieces (tail subproblems)
- Start from the last decision and go backwards
Solve the stage 2 subproblems (using the terminal costs - in red)

At each state of stage 2, we record the optimal cost-to-go and the optimal decision
Solve the stage 1 subproblems (using the optimal costs of stage 2 subproblems - in purple)

At each state of stage 1, we record the optimal cost-to-go and the optimal decision.
Solve the stage 0 subproblem (using the optimal costs of stage 1 subproblems - in orange)

- The stage 0 subproblem is the entire problem
- The optimal value of the stage 0 subproblem is the optimal cost $J^*(\text{initial state})$
- Construct the optimal sequence going forward
Minimize $G(u)$ subject to $u \in U$

- Assume that each solution $u$ has $N$ components: $u = (u_0, \ldots, u_{N-1})$
- View the components as the controls of $N$ stages
- Define $x_k = (u_0, \ldots, u_{k-1})$, $k = 1, \ldots, N$, and introduce artificial start state $x_0 = s$
- Define just terminal cost as $G(u)$; all other costs are 0

This formulation typically makes little sense for exact DP, but often makes a lot of sense for approximate DP/approximation in value space
Stochastic DP Problems - Perfect State Observation

Random Transition
\[ x_{k+1} = f_k(x_k, u_k, w_k) \]

Random Cost
\[ g_k(x_k, u_k, w_k) \]

- System \( x_{k+1} = f_k(x_k, u_k, w_k) \) with random "disturbance" \( w_k \) (e.g., physical noise, market uncertainties, demand for inventory, unpredictable breakdowns, etc)

- Cost function:
\[
E \left\{ g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k, w_k) \right\}
\]

- Policies \( \pi = \{\mu_0, \ldots, \mu_{N-1}\} \), where \( \mu_k \) is a "closed-loop control law" or "feedback policy"/a function of \( x_k \). Specifies control \( u_k = \mu_k(x_k) \) to apply when at \( x_k \).

- For given initial state \( x_0 \), minimize over all \( \pi = \{\mu_0, \ldots, \mu_{N-1}\} \) the cost
\[
J_\pi(x_0) = E \left\{ g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, \mu_k(x_k), w_k) \right\}
\]

- Optimal cost function \( J^*(x_0) = \min_\pi J_\pi(x_0) \)
The Stochastic DP Algorithm

Produces the optimal costs $J_k^*(x_k)$ of the tail subproblems that start at $x_k$

Start with $J_N^*(x_N) = g_N(x_N)$, and for $k = 0, \ldots, N - 1$, let

$$J_k^*(x_k) = \min_{u_k \in U_k(x_k)} E \left\{ g_k(x_k, u_k, w_k) + J_{k+1}^* (f_k(x_k, u_k, w_k)) \right\}, \quad \text{for all } x_k.$$ 

- The optimal cost $J^*(x_0)$ is obtained at the last step: $J_0^*(x_0) = J^*(x_0)$.
- The optimal control function $\mu_k^*$ is constructed simultaneously with $J_k^*$, and consists of the minimizing $u_k^* = \mu_k^*(x_k)$ above.

Online implementation of the optimal policy, given $J_1^*, \ldots, J_{N-1}^*$

Sequentially, going forward, for $k = 0, 1, \ldots, N - 1$, observe $x_k$ and apply

$$u_k^* \in \arg \min_{u_k \in U_k(x_k)} E \left\{ g_k(x_k, u_k, w_k) + J_{k+1}^* (f_k(x_k, u_k, w_k)) \right\}.$$ 

Issues: Need to compute $J_{k+1}^*$ (possibly off-line), compute expectation for each $u_k$, minimize over all $u_k$

Approximation in value space: Use $\tilde{J}_k$ in place of $J_k^*$; approximate $E\{\cdot\}$ and $\min_{u_k}$. 

Bertsekas Reinforcement Learning 16 / 27
Linear-quadratic problems involve: multidimensional linear system, quadratic cost, unconstrained controls, independent disturbances

**System:** \( x_{k+1} = (1 - a)x_k + au_k + w_k \) \((w_k \text{ are random, independent, and 0-mean)}\)

**Cost:** \( E\{r(x_N - T)^2 + \sum_{k=0}^{N-1} u_k^2\} \)

**A very favorable structure:** The optimal policy \( \mu^*_k(x_k) \) is a linear function of \( x_k \); it is the same as if \( w_1 \) and \( w_0 \) were set to their expected values \((= 0)\). Can be computed by exact DP

This is called certainty equivalence

Certainty equivalence is a common approximation idea for other problems (replace the original stochastic problem with a deterministic version)
Optimal Q-factors are given by

\[ Q_k^*(x_k, u_k) = E \left\{ g_k(x_k, u_k, w_k) + J_{k+1}^*(f_k(x_k, u_k, w_k)) \right\} \]

They define optimal policies and optimal cost-to-go functions by

\[ \mu_k^*(x_k) \in \arg \min_{u_k \in U_k(x_k)} Q_k^*(x_k, u_k), \quad J_k^*(x_k) = \min_{u_k \in U_k(x_k)} Q_k^*(x_k, u_k) \]

DP algorithm can be written in terms of Q-factors

\[ Q_k^*(x_k, u_k) = E \left\{ g_k(x_k, u_k, w_k) + \min_{u_{k+1}} Q_{k+1}^*(f_k(x_k, u_k, w_k), u_{k+1}) \right\} \]

Some math magic: With \( E\{ \cdot \} \) outside the min, the right side can be approximated by sampling and simulation. (Can be exploited in stochastic iterative algorithms called Q-learning.)

Approximately optimal Q-factors \( \tilde{Q}_k(x_k, u_k) \), define suboptimal policies and suboptimal cost-to-go functions by

\[ \tilde{\mu}_k(x_k) \in \arg \min_{u_k \in U_k(x_k)} \tilde{Q}_k(x_k, u_k), \quad \tilde{J}_k(x_k) = \min_{u_k \in U_k(x_k)} \tilde{Q}_k(x_k, u_k) \]
Infinite Horizon Problems - An Overview

Infinite number of stages, and stationary system and cost

- System \( x_{k+1} = f(x_k, u_k, w_k) \) with state, control, and random disturbance
- Policies \( \pi = \{\mu_0, \mu_1, \ldots\} \) with \( \mu_k(x) \in U(x) \) for all \( x \) and \( k \)
- Special scalar \( \alpha \) with \( 0 < \alpha \leq 1 \). If \( \alpha < 1 \) the problem is called discounted
- Cost of stage \( k \): \( \alpha^k g(x_k, \mu_k(x_k), w_k) \)
- Cost of a policy \( \pi = \{\mu_0, \mu_1, \ldots\} \)

\[
J_\pi(x_0) = \lim_{N \to \infty} E_{w_k} \left\{ \sum_{k=0}^{N-1} \alpha^k g(x_k, \mu_k(x_k), w_k) \right\}
\]

- Optimal cost function \( J^*(x_0) = \min_\pi J_\pi(x_0) \)
- If \( \alpha = 1 \) we assume a special cost-free termination state \( t \). The objective is to reach \( t \) at minimum expected cost. The problem is called stochastic shortest path (SSP) problem
Value iteration (VI): Fix horizon \( N \), let terminal cost be 0

- Let \( V_{N-k}(x) \) be the optimal cost starting at \( x \) with \( k \) stages to go, so
  \[
  V_{N-k}(x) = \min_{u \in U(x)} E_w \left\{ \alpha^{N-k} g(x, u, w) + V_{N-k+1}(f(x, u, w)) \right\}
  \]

- Reverse the time index and divide with \( \alpha^{N-k} \): Define \( J_k(x) = V_{N-k}(x)/\alpha^{N-k} \)
  \[
  J_k(x) = \min_{u \in U(x)} E_w \left\{ g(x, u, w) + \alpha J_{k-1}(f(x, u, w)) \right\}
  \]

- \( J_N(x) \) is equal to \( V_0(x) \), which is the \( N \)-stages optimal cost starting from \( x \)

- Hence, intuitively, VI converges to \( J^* \):
  \[
  J^*(x) = \lim_{N \to \infty} J_N(x), \quad \text{for all states } x \quad (??)
  \]

The following Bellman equation holds: Take the limit in Eq. (VI)

\[
J^*(x) = \min_{u \in U(x)} E_w \left\{ g(x, u, w) + \alpha J^*(f(x, u, w)) \right\}, \quad \text{for all states } x \quad (??)
\]

Optimality condition: Let \( \mu(x) \) attain the min in the Bellman equation for all \( x \)

The policy \( \{\mu, \mu, \ldots\} \) is optimal (??). (This type of policy is called stationary.)
How do we Formulate DP Problems?

An informal recipe: First define the stages and then the states

Define as state $x_k$ something that summarizes the past for purposes of future optimization, i.e., as long as we know $x_k$, all past information is irrelevant.

Some examples

- In the traveling salesman problem, we need to include all the info (past cities visited) in the state.
- In the linear quadratic problem, when we select the oven temperature $u_k$, the total info available is everything we have seen so far, i.e., the material and oven temperatures $x_0, u_0, x_1, u_1, \ldots, u_{k-1}, x_k$. However, all the useful information at time $k$ is summarized in just $x_k$.
- In partial or imperfect information problems, we use “noisy” measurements for control of some quantity of interest $y_k$ that evolves over time (e.g., the position/velocity vector of a moving object). If $I_k$ is the collection of all measurements up to time $k$, it is correct to use $I_k$ as state.
- It may also be correct to use alternative states; e.g., the conditional probability distribution $P_k(y_k | I_k)$. This is called belief state, and subsumes all the information that is useful for the purposes of control choice.
Problems with a Terminal State: A Parking Example

- Start at spot 0; either park at spot \( k \) with cost \( c(k) \) (if free) or continue; park at garage at cost \( C \) if not earlier.
- Spot \( k \) is free with a priori probability \( p(k) \), and its status is observed upon reaching it.
- How do we formulate the problem as a DP problem?

We have three states. \( F \): current spot is free, \( \overline{F} \): current spot is taken, parked state

\[
J_{N-1}^*(F) = \min \left[ c(N - 1), C \right], \quad J_{N-1}^*(\overline{F}) = C \\
J_k^*(F) = \min \left[ c(k), p(k + 1)J_{k+1}^*(F) + (1 - p(k + 1))J_{k+1}^*(\overline{F}) \right], \quad \text{for } k = 0, \ldots, N - 2 \\
J_k^*(\overline{F}) = p(k + 1)J_{k+1}^*(F) + (1 - p(k + 1))J_{k+1}^*(\overline{F}), \quad \text{for } k = 0, \ldots, N - 2
\]
**More Complex Parking Problems**

- **Bidirectional parking**: We can go back to parking spots we have visited at a cost.
- **More complicated parking lot topologies**.
- **Multiagent versions**: Multiple drivers/autonomous vehicles, “searchers”, etc.
- **“Relatively easy” cases**: The status of already seen spots stays unchanged.

![Parking Diagram](attachment:image.png)
A more complex type of parking example, where taken or free parking spots may free up or get taken, respectively, at the next time step with some probability.

The free/taken state of the spots is “estimated” in a “probabilistic sense” based on the observations (the free/taken status of the spots visited ... when visited).

What should the “state” be? It should summarize all the info needed for the purpose of future optimization.

First candidate for state: The set of all observations so far. Another candidate: The “belief state”, i.e., the conditional probabilities of the free/taken status of all the spots: $p(0), p(1), \ldots, p(N - 1)$.

Generally, partial observation problems (POMDP) can be “solved” by DP with state being the belief state: $P(x_k \mid \text{set of observations up to time } k)$.
About the Next Lecture

We will cover:

- General principles of approximation in value and policy space
- Brief discussion of the problem approximation approach
- Introduction to rollout

CHAPTER 2 OF THE CLASS NOTES POSTED
PLEASE READ AS MUCH OF SECTIONS 2.1, 2.2 AS YOU CAN

1ST HOMEWORK (DUE IN 2 WEEKS) TO BE ANNOUNCED SHORTLY