# Introduction to Dynamic Programming

Soo Min Kwon

Department of Electrical and Computer Engineering
Rutgers, The State University of New Jersey

September 10, 2021

# Outline

# Deterministic problem setting

A deterministic DP problem involves a discrete-time dynamic system that evolves over finite steps $N$ and is of the form

$$x_{k+1} = f_k(x_k, u_k), \quad k = 0, 1, ..., N-1, \tag{1}$$

where

- $k$ is the time index
- $x_k$ is the state of the system (an element of some space)
- $u_k$ is the control or decision variable, to be selected at time $k$ from some given set $U_k(x_k)$ that depends on $x_k$
- $f_k$ is a function of $(x_k, u_k)$ that describes how the state is updated from time $k$ to $k+1$
- $N$ is the horizon or number of times that control is applied

# Deterministic problem setting

Some more notation:

- *state space* is the set of all possible $x_k$ at time $k$
- *control space* is the set of all possible $u_k$ at time $k$
- $g_k(x_k, u_k)$ is the cost function incurred at time $k$ that takes real number values
- $g_N(x_N)$ is a terminal cost incurred at the end of the process

# Example of deterministic DP

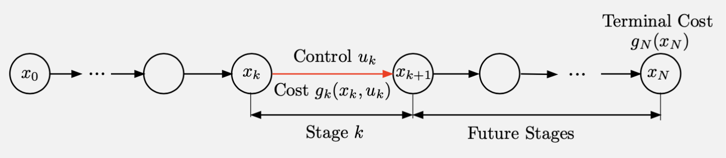



Figure: From state $x_k$, we can move to the next state under some nonrandom control $u_k$ according to $x_{k+1} = f_k(x_k, u_k)$

# Deterministic DP objective

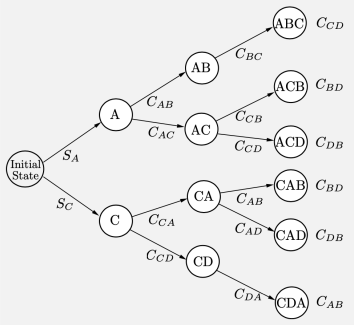
- For a given initial state $x_0$, the total cost of control sequence $\{u_0, ..., u_{N-1}\}$ is

$$J(x_0; u_0, ..., u_{N-1}) = g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k)) \qquad (2)$$

- We want to minimize this total cost over all control sequences to obtain the optimal value as a function of $x_0$:

$$J^*(x_0) = \min_{u_k \in U_k(x_k)} J(x_0; u_0, ..., u_{N-1}) \qquad (3)$$

# Example of a discrete control problem




- Example of a combinatorial finite-state, finite-horizon optimal control problem
- Goal is to produce a certain product from an initial state by performing a combination of operations A, B, C, and D
- Operation B can only be performed after A, and D can be performed only after C

# Principle of Optimality

The DP algorithm is based on the *principle of optimality*:

Let $\{u_0^*, ..., u_k^*, u_{k+1}^*, ..., u_{N-1}^*\}$ be an optimal control sequence, which determines the corresponding optimal state sequence, $\{x_1^*, ..., x_{N-1}^*\}$. Consider the subproblem in which we start at $x_k^*$ at time $k$ and wish to minimize the "cost-to-go" from time $k$ to $N$,

$$g_k(x_k^*, u_k) + \sum_{m=k+1}^{N-1} g_m(x_m, u_m) + g_N(x_N),$$

over $\{u_k, ..., u_{N-1}\}$ with $u_m \in U_m(x_m)$, $m = k, ..., N-1$. Then the truncated optimal control sequence $\{u_k^*, ..., u_{N-1}^*\}$ is optimal for the subproblem.

# Intuition behind the principle of optimality

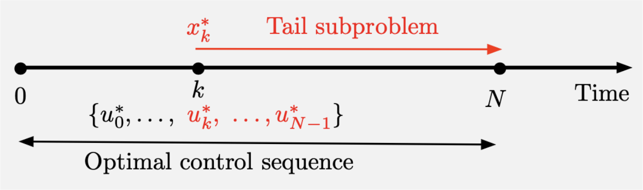
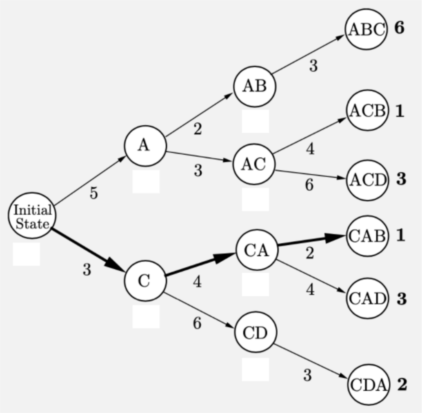



Figure: The tail $\{u_k^*, ..., u_{N-1}^*\}$ of an optimal control sequence $\{u_0^*, ..., u_k^*, u_{k+1}^*, ..., u_{N-1}^*\}$ is optimal for the tail subproblem that starts at $x_k^*$

- **Simple analogy:** If the fastest route from Los Angeles to Boston passes through Chicago, the principle of optimality states that the Chicago to Boston portion of the route is also the fastest for a trip that starts in Chicago and ends in Boston.

# Solving the previous example




Solution:

- Solve the tail subproblem of length 2, then 3, then 4
- Note the shortest path from the initial state to the terminal state

# DP algorithm for deterministic finite horizon problems

With the previous example, we can now construct the DP algorithm:

Start with

$$J_N^*(x_N) = g_N(x_N), \quad \text{for all } x_N, \tag{4}$$

and for $k = 0, ..., N-1$, let

$$J_k^*(x_k) = \min_{u_k \in U_k(x_k)} [g_k(x_k, u_k) + J_{k+1}^*(f_k(x_k, u_k))], \tag{5}$$

for all $x_k$.

- The algorithm is solving every tail subproblem by constructing functions $J_N^*(x_N), ..., J_0^*(x_0)$
- The last step gives us the optimal cost $J^*(x_0)$

# Constructing the optimal control sequence

With the previous algorithm, we can obtain the functions $J_0^*, ..., J_N^*$ and solve for the optimal control sequence:

**Construction of Optimal Control Sequence** $\{u_0^*, \ldots, u_{N-1}^*\}$

Set

$$u_0^* \in \arg \min_{u_0 \in U_0(x_0)} \Big[ g_0(x_0, u_0) + J_1^*\big(f_0(x_0, u_0)\big) \Big],$$

and

$$x_1^* = f_0(x_0, u_0^*).$$

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

$$u_k^* \in \arg \min_{u_k \in U_k(x_k^*)} \Big[ g_k(x_k^*, u_k) + J_{k+1}^*\big(f_k(x_k^*, u_k)\big) \Big], \tag{1.7}$$

and

$$x_{k+1}^* = f_k(x_k^*, u_k^*). \tag{1.8}$$

Figure: Construction of Optimal Control Sequence

# Approximation in value space

- Of course, in practice, computing $J_k^*(x_k)$ can be expensive
- Instead, find an approximation $\tilde{J}_k$ for a suboptimal solution

**Approximation in Value Space - Use of $\tilde{J}_k$ in Place of $J_k^*$**

Start with

$$\tilde{u}_0 \in \arg \min_{u_0 \in U_0(x_0)} \Big[ g_0(x_0, u_0) + \tilde{J}_1\big(f_0(x_0, u_0)\big) \Big],$$

and set

$$\tilde{x}_1 = f_0(x_0, \tilde{u}_0).$$

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

$$\tilde{u}_k \in \arg \min_{u_k \in U_k(\tilde{x}_k)} \Big[ g_k(\tilde{x}_k, u_k) + \tilde{J}_{k+1}\big(f_k(\tilde{x}_k, u_k)\big) \Big], \tag{1.9}$$

and

$$\tilde{x}_{k+1} = f_k(\tilde{x}_k, \tilde{u}_k). \tag{1.10}$$

Figure: Approximation in Value Space

# Q-Factors

The minimization expression of equation (1.9) is the **Q-factor** of $(x_k, u_k)$:

$$\tilde{Q}_k(x_k, u_k) = g_k(x_k, u_k) + \tilde{J}_{k+1}(f_k(x_k, u_k)) \tag{6}$$

# Stochastic dynamic programming

- Stochastic DP problems have the form

$$x_{k+1} = f_k(x_k, u_k, w_k), \quad k = 0, 1, ..., N-1, \tag{7}$$

  where $w_k$ is a "disturbance" (e.g. physical noise, market uncertainties, demand for inventory) characterized by a probability distribution $P_k(\cdot | x_k, u_k)$ that may depend on $x_k, u_k$, but not on prior disturbances
- An **important** difference is that we optimize not over the control sequences, but rather over *policies*

$$\pi = \{\mu_0, ..., \mu_{N-1}\},$$

  where $u_k = \mu_k(x_k)$.

# Stochastic dynamic programming

- In the presence of uncertainty, optimizing over the policies can give us improved costs, as they allow choices that use knowledge of $x_k$
- Given an initial state $x_0$ and policies $\pi = \{\mu_0, ..., \mu_{N-1}\}$, the expected cost of $\pi$ starting at $x_0$ is

$$J_\pi(x_0) = E\left\{g_N(x_N) + \sum_{k=0}^{N-1} g_k\big(x_k, \mu_k(x_k), w_k\big)\right\}$$

- An optimal policy $\pi^*$ is one that minimizes the cost

$$J_{\pi^*}(x_0) = \min_{\pi \in \Pi} J_\pi(x_0)$$

# DP algorithm for stochastic finite horizon problems

Similar to that of the deterministic algorithm except with random variables:

**DP Algorithm for Stochastic Finite Horizon Problems**

Start with

$$J_N^*(x_N) = g_N(x_N), \tag{1.12}$$

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

$$J_k^*(x_k) = \min_{u_k \in U_k(x_k)} E\Big\{ g_k(x_k, u_k, w_k) + J_{k+1}^*\big(f_k(x_k, u_k, w_k)\big)\Big\}. \tag{1.13}$$

If $u_k^* = \mu_k^*(x_k)$ minimizes the right side of this equation for each $x_k$ and $k$, the policy $\pi^* = \{\mu_0^*, \dots, \mu_{N-1}^*\}$ is optimal.