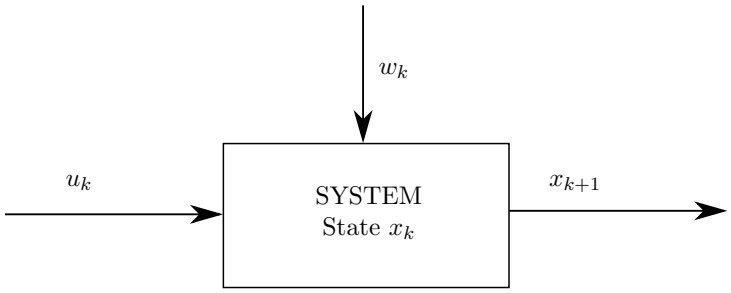
1. Introduction to Dynamic Programming

1.1 Problem Statement

Given a model of how a dynamic system evolves and a direct measurement of its state, apply a control input to the system so that a given cost is minimized.

Dynamics



$$x_{k+1} = f_k\left(x_k, u_k, w_k\right), \quad k = 0, 1, ..., N-1 \tag{1.1}$$

where

- k: discrete time index, or stage;
- N: given time horizon;
- $x_k \in \mathcal{S}_k$: system state vector at time k, can be measured at time k. $\mathcal{S}_k$ is the allowable set of states, which can be a function of time;
- $u_k \in \mathcal{U}_k(x_k)$: control input vector at time k. $\mathcal{U}_k(x_k)$ is the allowable set of control inputs, which can be a function of state and time;
- w_k: disturbance vector at time k, a random variable (see Appendix). It is assumed that w_k is conditionally independent with all prior variables $x_l,\ u_l,\ w_l,\ l < k$, given x_k and u_k. Furthermore, it is assumed that the conditional probability distribution of w_k given x_k and u_k is known;
- $f_k\left(\cdot, \cdot, \cdot\right)$: function capturing system evolution at time k.

Date compiled: September 22, 2020

Cost Function

We will consider the following *scalar-valued additive* cost function:

$$\underbrace{g_N(x_N)}_{\text{terminal cost}} + \underbrace{\sum_{k=0}^{N-1} \underbrace{g_k(x_k, u_k, w_k)}_{\text{stage cost}}}_{\text{accumulated cost}} \tag{1.2}$$

Example 1: Inventory Control
We are keeping an item stocked in a warehouse. If there is too little, we will run out of it and lose sales (not preferred). If there is too much, there will be more cost of storage and misuse of capital (not preferred). We will model this scenario as a discrete time system:

- $x_k \in \mathcal{S}_k = \mathbb{R}$: stock available in the warehouse at the beginning of the k^{th} time period.
- $u_k \in \mathcal{U}_k(x_k) = \mathbb{R}_{\geq 0}$: stock ordered and immediately delivered at the beginning of the k^{th} time period (supply).
- w_k: demand during the k^{th} time period, with some given probability distribution.
- *Dynamics*: $x_{k+1} = x_k + u_k - w_k$. We assume that excess demand is back-logged, which corresponds to negative x_k.
- *Cost function*:

  $$R(x_N) + \sum_{k=0}^{N-1} r(x_k) + cu_k - pw_k$$

  where
  - pw_k: revenue
  - cu_k: cost of items;
  - $r(x_k)$: cost associated with too much stock or negative stock;
  - $R(x_N)$: terminal cost; cost associated with stock left at the end which we can't sell, or demand we can't meet;

△

Expected Cost

Let $X_1 := (x_1, \ldots, x_N)$, $U_0 := (u_0, \ldots, u_{N-1})$, and $W_0 := (w_0, \ldots, w_{N-1})$. In general, given x_0, X_1, U_0 and W_0 are all random variables due to the disturbances w_k and the dynamic coupling (1.1). For example, the state x_1 is a random variable with *probability density function* (PDF, see Appendix) defined through the system equations (1.1), the control input u_0, and the random variable w_0; x_2 is then a random variable as well, as it is a function of the random variables x_1, w_1, and so on. The control inputs can either be fixed and thus deterministic, or a function of the state and thus also random. The cost function (1.2) is thus a random variable; a convenient metric for optimization is taking its expected value to yield the expected cost of starting at an initial state x_0, that is,

$$\underset{(X_1,U_0,W_0|x_0)}{\mathrm{E}}\left[g_N(x_N)+\sum_{k=0}^{N-1} g_k(x_k,u_k,w_k)\right] \text{ subject to (1.1)},$$

where $\mathrm{E}[\cdot]$ is the *expectation operator*, see Appendix. Note: the underset on $\mathrm{E}[\cdot]$ is sometimes omitted for brevity. The expected cost can further be simplified as we will see in Section 1.2 depending on the employed control strategy.

1.2 Open Loop and Closed Loop Control

There are two different control methodologies: open loop, where all the control inputs are determined at once at time 0, and closed loop, where the control inputs are determined in a "just-in-time fashion", depending on the measured state x_k at time k.

1.2.1 Open Loop Control

Given an initial state x_0 and a set of control inputs $\bar{U}_0 := (\bar{u}_0, \ldots, \bar{u}_{N-1})$ that is fixed (we use the bar to emphasize that the control inputs are fixed), the cost (1.2) becomes

$$g_N(x_N)+\sum_{k=0}^{N-1} g_k(x_k,\bar{u}_k,w_k)\,, \tag{1.3}$$

and is thus a function of the random variables x_k and w_k. We therefore wish to optimize the following expected cost

$$\underset{(X_1,W_0|x_0)}{\mathrm{E}}\left[g_N(x_N)+\sum_{k=0}^{N-1} g_k(x_k,\bar{u}_k,w_k)\right] \tag{1.4}$$

subject to the dynamics given by (1.1) (with $u_k = \bar{u}_k$).
Summarizing, the open loop control problem is thus the following: at time $k = 0$, given x_0, find $\bar{U}_0$ that minimizes the expected open loop cost (1.4). This is called *open loop* control because measurements of the state are not used to calculate the control inputs.

1.2.2 Closed Loop Control

Let $\mu_k(\cdot)$ map state x_k to control input u_k:

$$u_k = \mu_k(x_k)\,,\ \ u_k \in \mathcal{U}_k(x_k)\ \forall x_k \in \mathcal{S}_k,\ k = 0,\ldots,N-1 \tag{1.5}$$

and define

$$\pi := (\mu_0(\cdot)\,, \mu_1(\cdot)\,, \ldots, \mu_{N-1}(\cdot))\,,$$

where π is called an *admissible policy*. Given an initial state x_0, the states $x_1, \ldots, x_N$, the control inputs $u_1, \ldots u_{N-1}$ and the disturbances $w_0, \ldots, w_{N-1}$ are random variables with PDFs defined through the system equations (1.1) and the state feedback equations (1.5).
The cost (1.2) therefore becomes

$$g_N(x_N)+\sum_{k=0}^{N-1} g_k(x_k,\mu_k(x_k),w_k)\,, \tag{1.6}$$

a function of the random variables $x_1, \ldots, x_N$ and $w_0, \ldots, w_{N-1}$. As a result, we define, for any $\mathrm{x} \in \mathcal{S}_0$, the *expected closed loop cost* associated with an admissible policy π to be

$$J_\pi(\mathrm{x}) := \underset{(X_1, W_0 | x_0 = \mathrm{x})}{\mathrm{E}} \left[g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, \mu_k(x_k), w_k) \right] \text{ subject to (1.1), (1.5).}$$

Let Π denote the set of all admissible policies. Then π^* is called an *optimal policy* if

$$J_{\pi^*}(\mathrm{x}) \leq J_\pi(\mathrm{x}) \qquad \forall \pi \in \Pi, \forall \mathrm{x} \in \mathcal{S}_0.$$

The *optimal cost* is defined as $J^*(\mathrm{x}) := J_{\pi^*}(\mathrm{x})$. $J^*(\cdot)$ is thus a function that maps initial states to optimal costs. The closed loop control problem aims at finding π^*.

Example 2: Inventory Control
For the inventory control problem described in Example 1, an intuitive example of an admissible policy is

$$\mu_k(x_k) = \begin{cases} s_k - x_k, & \text{if } x_k < s_k \\ 0, & \text{otherwise} \end{cases} \quad \forall k$$

where s_k is some predefined, potentially time varying threshold. △

1.2.3 Performance

It is clear that open loop control can never give better performance than closed loop control: a special case of closed loop control is to simply disregard state information, and thus open loop control is a special case of closed loop control. In the absence of disturbances w_k, the two give *theoretically* the same performance. In particular, in the absence of disturbances, $x_1, \ldots, x_N$ can be calculated from x_0 and $u_0, \ldots, u_{N-1}$, and there is thus no need to measure the state, and the optimal sequence of control inputs can be determined as soon as x_0 is known.
In practice, even when there are no disturbances, closed loop control will give better performance than open loop control for the following reasons:

- x_0 is often not known precisely and may also be random;
- the $f_k(\cdot, \cdot, \cdot)$ are often not known precisely;
- models are often only approximations of reality.

However, when a system is well-behaved and a good model for it exists, open loop control is a viable strategy, especially for short time horizons.

1.2.4 Computation

In terms of computation, open loop control is typically much less demanding than closed loop control. Consider, for example, a system with N_x distinct states and N_u distinct control inputs. There are a total of N_u^N different open loop strategies and $N_u(N_u^{N_x})^{N-1} = N_u^{N_x(N-1)+1}$ different closed loop strategies. There are thus many more closed loop strategies than open loop ones. As an example, let's take $N_u = 10$, $N_x = 10$, and $N = 4$. The number of open loop strategies is 10^4. The number of closed loop strategies is 10^{31}, which is almost 10 orders of magnitude larger than the number of stars in the observable universe!

1.3 Discrete State and Finite State Problems

When state x_k takes on discrete values, or is finite in size, it is often more convenient to express the dynamics in terms of *transition probabilities*

$$P_{ij}(u,k) := \Pr\left(x_{k+1} = j \mid x_k = i, u_k = u\right) = p_{x_{k+1}|x_k,u_k}(j|i,u)$$

where $p_{x_{k+1}|x_k,u_k}(\cdot\,|\,\cdot,\,\cdot)$ denotes the PDF of x_{k+1} given x_k and u_k.
This is equivalent to the dynamics:

$$x_{k+1} = w_k$$

where w_k has the following probability distribution:

$$p_{w_k|x_k,u_k}(j|i,u) = P_{ij}(u,k).$$

Conversely, given $x_{k+1} = f_k\left(x_k, u_k, w_k\right)$ and $p_{w_k|x_k,u_k}(\cdot\,|\,\cdot,\,\cdot)$, then

$$P_{ij}(u,k) = \sum_{\{\bar{w}_k | f_k(i,u,\bar{w}_k)=j\}} p_{w_k|x_k,u_k}(\bar{w}_k|i,u)$$

that is, $P_{ij}(u,k)$ is equal to the sum over the probabilities of all possible disturbances $\bar{w}_k$ that get us to state j from state i using control u at time k.

Example 3: Optimizing chess match strategy
Consider a two game chess match with an opponent. Our objective is to come up with a strategy that maximizes the chance of winning the match. Each game can have one of two outcomes: 1) Win/Lose: 1 point for the winner, 0 for the loser; 2) Tie: 0.5 points for each player. In addition, if at the end of two games the score is equal, the players keep on playing new games until one wins, and thereby wins the match (also known as sudden death).
There are two possible playing styles for our player: timid and bold. When playing timid, our player ties with probability p_d and loses with probability $(1-p_d)$. When playing bold, our player wins with probability p_w and loses with probability $(1-p_w)$. We also assume that $p_d > p_w$, a necessary condition for this problem to make sense.
We will model this as a finite state problem:

- The *state* x_k is a two dimensional vector with the score of each player after the k^{th} game, where the first entry denotes the score of our player and the second the score of the opponent.
- The *control inputs* u_k are the two playing styles: timid and bold.
- The *disturbance* w_k is the score of the next game x_{k+1}.
- *Dynamics:* since it doesn't make sense to play timid if the game goes into sudden death, the problem is a two-stage finite state problem. We can construct a *Transition Probability Graph*, which can then be used to deduce $P_{ij}(u,k)$ to express the dynamics. Fig. 1.1 and 1.2 show all possible outcomes after the first and second game, respectively.
- *Cost*: we want to maximize the probability of winning, P_{win}. Therefore, the cost is $-P_{\text{win}}$, which is to be minimized. This is equivalent to the standard form

$$g_2(x_2) + \sum_{k=0}^{1} g_k(x_k, u_k, w_k)$$

where

$$g_k(x_k, u_k, w_k) = 0, \quad \forall k \in \{0, 1\}$$

$$g_2(x_2) = \begin{cases} -1 & \text{if } x_2 = (\frac{3}{2}, \frac{1}{2}) \text{ or } (2, 0), \\ -p_w & \text{if } x_2 = (1, 1), \\ 0 & \text{if } x_2 = (\frac{1}{2}, \frac{3}{2}) \text{ or } (0, 2). \end{cases}$$

Transition Probability Graph:

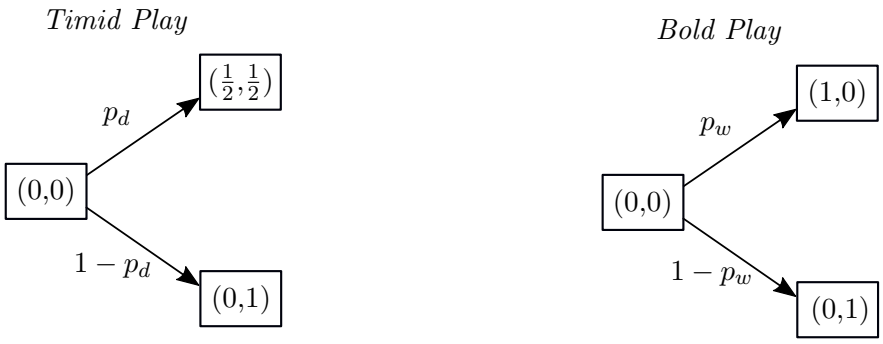


Figure 1.1: First game

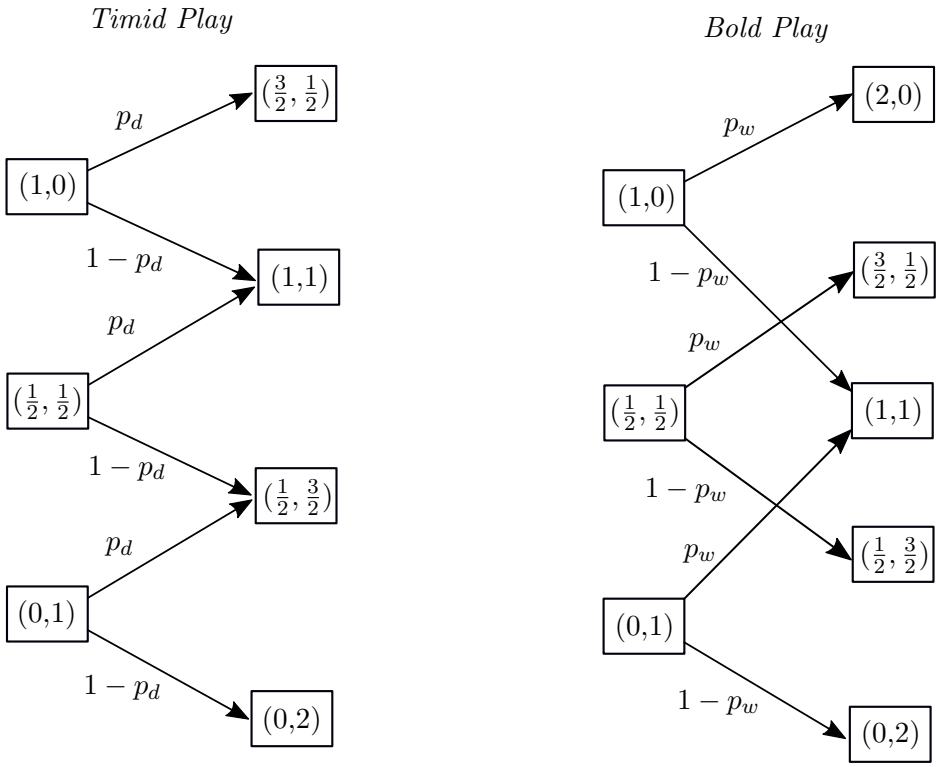


Figure 1.2: Second game

Now let's look at the expected cost under open loop and closed loop strategies:

Open Loop Strategy: There are 4 possibilities:

1) Play timid in first 2 games: $P_{\text{win}} = p_d^2 p_w$

2) Play bold in first 2 games: $P_{\text{win}} = p_w^2 + p_w(1-p_w)p_w + (1-p_w)p_w p_w = p_w^2(3-2p_w)$

3) Play bold in first, timid in second game: $P_{\text{win}} = p_w p_d + p_w(1-p_d)p_w$

4) Play timid in first, bold in second game: $P_{\text{win}} = p_d p_w + (1-p_d)p_w^2$

Since $p_d^2 p_w \le p_d p_w \le p_d p_w + (1-p_d)p_w^2$, clearly 1) is not the optimal open loop strategy. The best achievable winning probability P^*_{win} is:

$$\begin{aligned} P^*_{\text{win}} &= \max\{\overbrace{p_w^2(3-2p_w)}^{2)}, \overbrace{p_d p_w + (1-p_d)p_w^2}^{3) \text{ or } 4)}\} \\ &= p_w^2 + \max\{2(p_w^2 - p_w^3),\ p_d p_w - p_d p_w^2\} \\ &= p_w^2 + \max\{2p_w(1-p_w)p_w,\ p_w(1-p_w)p_d\} \\ &= p_w^2 + p_w(1-p_w)\max\{2p_w,\ p_d\} \end{aligned}$$

If $p_d > 2p_w$, then 3) and 4) are the best open loop strategies, otherwise 2) is the best open loop strategy.

- For $p_w = 0.45$ and $p_d = 0.9$, $P^*_{\text{win}} = 0.43$.
- For $p_w = 0.5$ and $p_d = 1.0$, $P^*_{\text{win}} = 0.5$.

It can also be shown that, in the open loop case, if $p_w \le 0.5$ then $P^*_{\text{win}} \le 0.5$.

Closed Loop Strategy: There are 8 admissible policies. Let's consider one possible policy: play timid iff player is winning (in Lecture 3 we will show that this strategy is indeed the optimal policy). Fig. 1.3 shows the corresponding transition probabilities under this specific policy. Then, the associated probability of winning P_{win} is

$$p_d p_w + p_w((1-p_d)p_w + p_w(1-p_w)) = p_w^2(2-p_w) + p_w(1-p_w)p_d$$

- For $p_w = 0.45$ and $p_d = 0.9$, $P_{\text{win}} = 0.54$
- For $p_w = 0.5$ and $p_d = 1.0$, $P_{\text{win}} = 0.625$

Note that in the closed loop case we can achieve a winning probability larger than 0.5 even when p_w is less than 0.5.

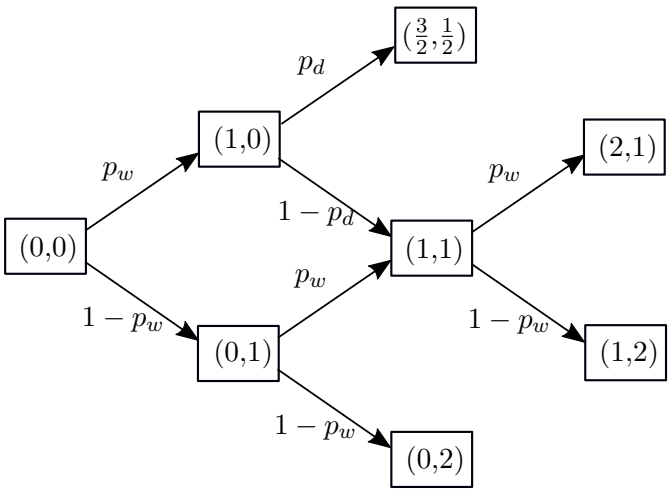


Figure 1.3: Closed loop strategy

△

2. The Dynamic Programming Algorithm

Last lecture we introduced the concepts of closed loop and open loop control and demonstrated the superiority of the closed loop approach. Now we will introduce the Dynamic Programming Algorithm (DPA), which generates an optimal closed loop controller that minimizes the expected closed loop cost. The algorithm is general enough to handle non-convex and non-linear problems, and is much more efficient than the brute-force approach of evaluating all possible strategies.

2.1 The Standard Problem Formulation

Recall the standard problem formulation for closed loop control from the previous lecture:

Dynamics

- The state evolution is governed by

$$x_{k+1} = f_k(x_k, u_k, w_k), \quad k = 0, 1, 2, \ldots, N-1, \tag{2.1}$$

  where $x_k \in \mathcal{S}_k$, $u_k \in \mathcal{U}_k(x_k)$, and $w_k \sim p_{w_k|x_k,u_k}$. Furthermore, $p_{w_k|x_k,u_k,*} = p_{w_k|x_k,u_k}$, $\forall * \in \{x_l, u_l, w_l \mid l < k\}$ (i.e. the conditional independence assumption from the previous lecture).

- The control inputs u_k are generated by an admissible policy $\pi \in \Pi$:

$$\pi = (\mu_0(\cdot), \mu_1(\cdot), \ldots, \mu_{N-1}(\cdot))$$

  such that

$$u_k = \mu_k(x_k), \; u_k \in \mathcal{U}_k(x_k), \; k = 0, \ldots, N-1. \tag{2.2}$$

Expected Cost

Given $\mathrm{x} \in \mathcal{S}_0$, the expected closed loop cost of starting at $x_0 = \mathrm{x}$ associated with policy π is:

$$J_\pi(\mathrm{x}) = \underset{(X_1, W_0 | x_0 = \mathrm{x})}{\mathrm{E}} \left[g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, \mu_k(x_k), w_k) \right] \quad \text{subject to (2.1), (2.2),}$$

where $X_1 := (x_1, \ldots, x_N)$ and $W_0 := (w_0, \ldots, w_{N-1})$.

Objective

Construct an optimal policy π^* such that for all $\mathrm{x} \in \mathcal{S}_0$,

$$\pi^* = \arg\min_{\pi \in \Pi} J_\pi(\mathrm{x}).$$

2.2 Principle of Optimality

The heart of the DPA is based on the following simple idea known as the *Principle of Optimality*: Let $\pi^* = (\mu_0^*(\cdot), \mu_1^*(\cdot), \ldots, \mu_{N-1}^*(\cdot))$ be an optimal policy. Consider the subproblem whereby we are at $\mathrm{x} \in \mathcal{S}_i$ at time i and we want to minimize:

$$\underset{(X_{i+1}, W_i | x_i = \mathrm{x})}{\mathrm{E}} \left[g_N(x_N) + \sum_{k=i}^{N-1} g_k(x_k, \mu_k(x_k), w_k) \right] \text{ subject to (2.1), (2.2),}$$

where $X_{i+1} := (x_{i+1}, \ldots, x_N)$ and $W_i := (w_i, \ldots, w_{N-1})$. Then the truncated policy $\left(\mu_i^*(\cdot), \mu_{i+1}^*(\cdot), \ldots, \mu_{N-1}^*(\cdot)\right)$ is optimal for this problem.

Intuition: if it weren't, we would have a policy yielding a lower cost on at least some portion of the state space, which contradicts that π^* was an optimal policy. We will prove this later in this lecture.

Before stating the DPA, the following example illustrates how the algorithm is applied: first construct an optimal solution at the last stage, then work backwards, sequentially.

Example 1: Deterministic Scheduling Problem
Consider a deterministic scheduling problem, where 4 operations A, B, C, D are used to produce a certain product. Operation A must occur before B, and C before D. Between each two operations there is a transition cost. Fig. 8.2 illustrates the transition dynamics and the transition costs. The solution can be found in Fig. 2.2.

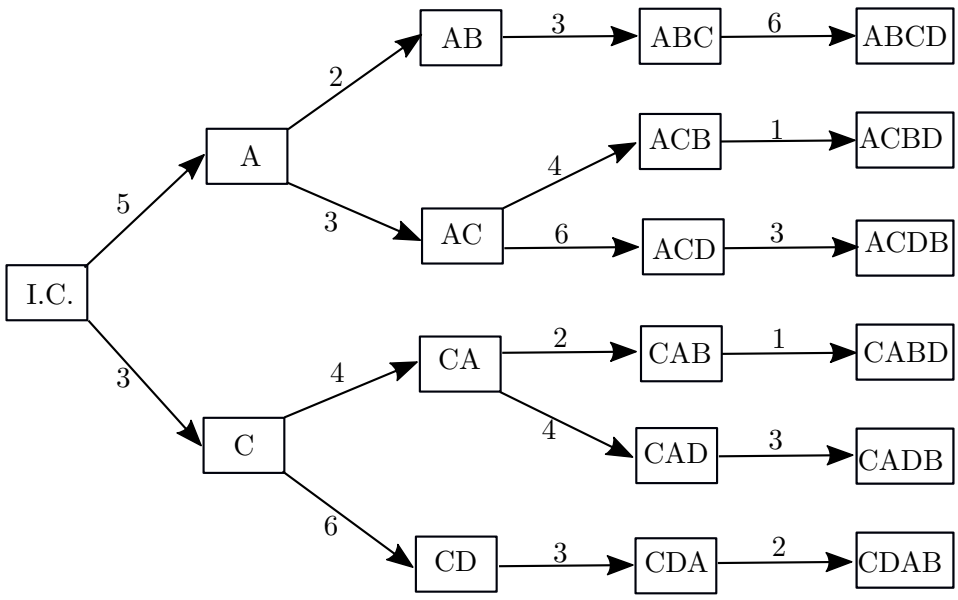


Figure 2.1: The number on each arc denotes the transition cost between two operations, and is given. The state is the cumulative operations performed and their order.

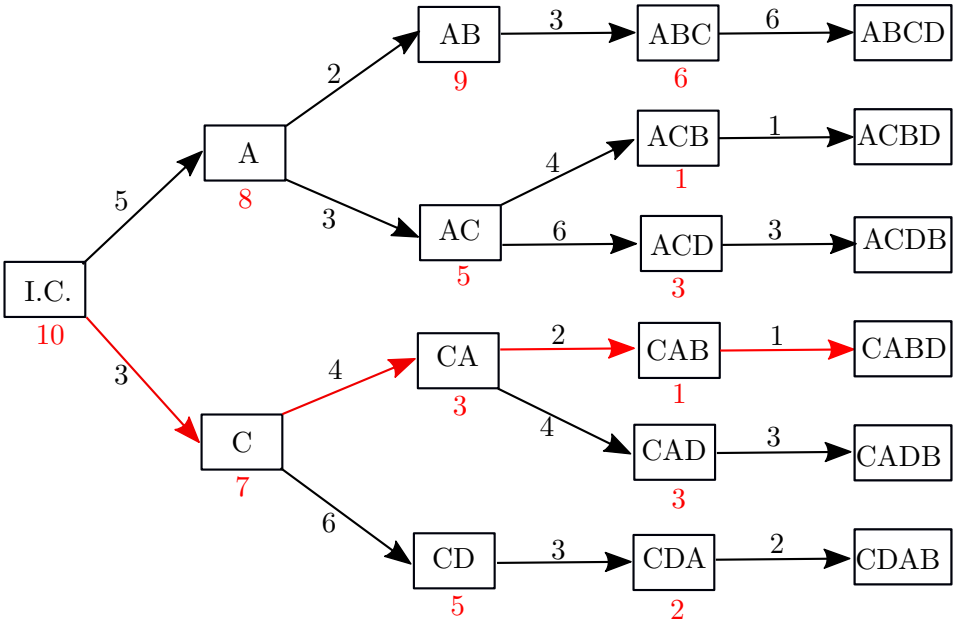


Figure 2.2: The optimal cost to go at each state is stated with red text below the corresponding state. The optimal solution is denoted with the red line.

△

2.3 The Dynamic Programming Algorithm

Recall that the optimal cost is defined[1] as $J^*(\mathrm{x}) := J_{\pi^*}(\mathrm{x})$. The following theorem states the DPA which computes the optimal cost and an associated policy.

Theorem 2.1. *For any initial state* $\mathrm{x} \in \mathcal{S}_0$*, the optimal cost* $J^*(\mathrm{x})$ *is equal to* $J_0(\mathrm{x})$ *given by the following recursive algorithm:*

Initialization

$$J_N(\mathrm{x}) := g_N(\mathrm{x}), \qquad \forall \mathrm{x} \in \mathcal{S}_N$$

Recursion

$$J_k(\mathrm{x}) := \min_{\mathrm{u} \in \mathcal{U}_k(\mathrm{x})} \mathrm{E}_{(w_k | x_k = \mathrm{x}, u_k = \mathrm{u})} \Big[g_k(x_k, u_k, w_k) + J_{k+1}(f_k(x_k, u_k, w_k)) \Big], \qquad \forall \mathrm{x} \in \mathcal{S}_k, \ k = N-1, \ldots, 0.$$

Furthermore, if for each k *and* $\mathrm{x} \in \mathcal{S}_k$, $\mathrm{u}^* =: \mu_k^*(\mathrm{x})$ *minimizes the recursion equation, the policy* $\pi^* = (\mu_0^*(\cdot), \mu_1^*(\cdot), \ldots, \mu_{N-1}^*(\cdot))$ *is optimal.*

[1] Strictly speaking, π^* may not exist even though $J^*(\mathrm{x})$ exists. In that case, $J^*(\mathrm{x}) = \inf_{\pi \in \Pi} J_\pi(\mathrm{x})$, and the objective then becomes to construct a policy which results in an expected closed loop cost that comes to within some pre-specified tolerance of the optimal cost.

Comments:

- For each recursion step, we have to perform the optimization over all possible values of $\mathrm{x} \in \mathcal{S}_k$, since we don't know a-priori which states will be visited.
- This point-wise optimization in $\mathrm{x} \in \mathcal{S}_k$ is what gives us $\mu_k^*(\cdot)$.
- $J_k(\mathrm{x})$ is often called *cost-to-go at state* $\mathrm{x} \in \mathcal{S}_k$, $J_k(\cdot)$ the *cost-to-go function*.

Proof. We define the auxiliary functions

$$l_k(X_k, W_k, \pi_k) := g_N(x_N) + \sum_{i=k}^{N-1} g_i(x_i, \mu_i(x_i), w_i), \quad k = 0, 1, \dots N-1,$$
$$l_N(x_N) := g_N(x_N),$$

where $X_k = (x_k, \dots, x_N)$, $W_k = (w_k, \dots, w_{N-1})$, and $\pi_k := (\mu_k(\cdot), \dots, \mu_{N-1}(\cdot))$. These functions satisfy the following recursion

$$l_k(X_k, W_k, \pi_k) = g_k(x_k, \mu_k(x_k), w_k) + l_{k+1}(X_{k+1}, W_{k+1}, \pi_{k+1}), \tag{2.3}$$

for all $k = 0, 1, \dots N-1$.
Given these auxiliary functions, the optimal cost for the $(N-k)$-stage problem that starts at time k and state x can be introduced as

$$J_k^*(\mathrm{x}) := \min_{\pi_k} \underset{(X_{k+1}, W_k | x_k = \mathrm{x})}{\mathrm{E}} [\, l_k(X_k, W_k, \pi_k) \,] \quad \text{subject to (2.1), (2.2).} \tag{2.4}$$

We proceed via induction, namely, we will show that for all k, $J_k^*(\cdot)$ is equal to $J_k(\cdot)$ generated by the DPA, which gives us the desired result when $k = 0$.
The terminal case is satisfied by definition:

$$J_N^*(\mathrm{x}) = g_N(\mathrm{x}) = J_N(\mathrm{x}), \quad \forall \mathrm{x} \in \mathcal{S}_N.$$

Now by induction, assume that for $k+1$, $J_{k+1}^*(\mathrm{x}) = J_{k+1}(\mathrm{x})$ for all $\mathrm{x} \in \mathcal{S}_{k+1}$. Then,

$$\begin{aligned}
J_k^*(\mathrm{x}) &\overset{1}{=} \min_{\pi_k} \underset{(X_{k+1}, W_k | x_k = \mathrm{x})}{\mathrm{E}} [g_k(x_k, \mu_k(x_k), w_k) + l_{k+1}(X_{k+1}, W_{k+1}, \pi_{k+1})] \\
&\overset{2}{=} \min_{\pi_k} \underset{(w_k | x_k = \mathrm{x})}{\mathrm{E}} \left[\underset{(X_{k+1}, W_{k+1} | x_k = \mathrm{x}, w_k)}{\mathrm{E}} [g_k(x_k, \mu_k(x_k), w_k) + l_{k+1}(X_{k+1}, W_{k+1}, \pi_{k+1})] \right] \\
&\overset{3}{=} \min_{\pi_k} \underset{(w_k | x_k = \mathrm{x})}{\mathrm{E}} \left[g_k(x_k, \mu_k(x_k), w_k) + \underset{(X_{k+1}, W_{k+1} | x_k = \mathrm{x}, w_k)}{\mathrm{E}} [l_{k+1}(X_{k+1}, W_{k+1}, \pi_{k+1})] \right] \\
&\overset{4}{=} \min_{\pi_k} \underset{(w_k | x_k = \mathrm{x})}{\mathrm{E}} \left[g_k(x_k, \mu_k(x_k), w_k) + \underset{(X_{k+2}, W_{k+1} | x_{k+1} = f_k(\mathrm{x}, \mu_k(\mathrm{x}), w_k))}{\mathrm{E}} [l_{k+1}(X_{k+1}, W_{k+1}, \pi_{k+1})] \right] \\
&\overset{5}{=} \min_{\mu_k(\mathrm{x})} \underset{(w_k | x_k = \mathrm{x})}{\mathrm{E}} \left[g_k(x_k, \mu_k(x_k), w_k) + \min_{\pi_{k+1}} \underset{(X_{k+2}, W_{k+1} | x_{k+1} = f_k(\mathrm{x}, \mu_k(\mathrm{x}), w_k))}{\mathrm{E}} [l_{k+1}(X_{k+1}, W_{k+1}, \pi_{k+1})] \right] \\
&\overset{6}{=} \min_{\mu_k(\mathrm{x})} \underset{(w_k | x_k = \mathrm{x})}{\mathrm{E}} \left[g_k(x_k, \mu_k(x_k), w_k) + J_{k+1}^*(f_k(x_k, \mu_k(x_k), w_k)) \right] \\
&\overset{7}{=} \min_{\mu_k(\mathrm{x})} \underset{(w_k | x_k = \mathrm{x})}{\mathrm{E}} \left[g_k(x_k, \mu_k(x_k), w_k) + J_{k+1}(f_k(x_k, \mu_k(x_k), w_k)) \right] \\
&\overset{8}{=} \min_{\mathrm{u} \in \mathcal{U}_k(\mathrm{x})} \underset{(w_k | x_k = \mathrm{x}, u_k = \mathrm{u})}{\mathrm{E}} \left[g_k(x_k, u_k, w_k) + J_{k+1}(f_k(x_k, u_k, w_k)) \right] = J_k(\mathrm{x}), \quad \forall \mathrm{x} \in \mathcal{S}_k.
\end{aligned}$$

Note that all the minimizations and expectations are subject to (2.1) and (2.2), as usual.

Steps:

1. Using the relation given by (2.3).
2. Using the property $\underset{ab}{\mathrm{E}}\left[q(a,b)\right] = \underset{b}{\mathrm{E}}\left[\underset{a|b}{\mathrm{E}}\left[q(a,b)\right]\right]$ (see Appendix Example 1).
3. Since $g_k(x_k, \mu_k(x_k), w_k)$ is not a function of the random variables X_{k+1}, W_{k+1}.
4. Using the relation

$$p_{X_{k+1},W_{k+1}|x_k,w_k}(\bar{x}_{k+1},\ldots,\bar{x}_N,\bar{W}|\bar{x}_k,\bar{w}_k) = \begin{cases} p_{X_{k+2},W_{k+1}|x_{k+1}}(\bar{x}_{k+2},\ldots,\bar{x}_N,\bar{W}|\bar{x}_{k+1}) & \text{if } \bar{x}_{k+1} = f_k(\bar{x}_k,\mu_k(\bar{x}_k),\bar{w}_k) \\ 0 & \text{otherwise,} \end{cases}$$

   which is proved in the Appendix.
5. By noting that $g_k(x_k, \mu_k(x_k), w_k)$ is not dependent on π_{k+1}.
6. By definition of the expected cost-to-go given by (2.4).
7. By induction hypothesis.
8. By using the fact that $u_k = \mu_k(x_k)$, which implies

$$p_{w_k|x_k}(\bar{w}|\bar{x}_k) = \sum_{\bar{u}_k \in \mathcal{U}_k(\bar{x}_k)} p_{w_k|x_k,u_k}(\bar{w}|\bar{x}_k,\bar{u}_k) \underbrace{p_{u_k|x_k}(\bar{u}_k|\bar{x}_k)}_{\text{1 for } \bar{u}_k = \mu_k(\bar{x}_k),\ 0 \text{ otherwise}} = p_{w_k|x_k,u_k}(\bar{w}|\bar{x}_k,\mu_k(\bar{x}_k)).$$

□

Discussion:

- Steps 1 to 6 (inclusive) is the *Principle of Optimality.*
- Referring back to Section 1.2.4, consider an N-stage finite-state-system with N_x distinct states and N_u distinct control inputs. For each stage k and each state x_k, we have to go through N_u control inputs to determine the associated optimal input. In total, we must then have to do $N_u N_x(N-1) + N_u$ such operations. Referring to the example with $N_x = 10$, $N_u = 10$, and $N = 4$, this would be just 310 operations instead of going through all 10^{31} policies.

2.4 Is Expected Value a Good Choice for the Cost?

Although the expected value of the cost is a useful metric in the sense that it is convenient and often makes sense, it does not take higher order statistics (e.g. variance) into account, as demonstrated in Example 2.

Example 2: Expected Value vs. Variance
Consider the two PDFs in Fig. 2.3 where

$$a = \frac{4(L-1)}{(2L-1)}, \quad b = \frac{1}{(L-1)(2L-1)}$$

and note the following:

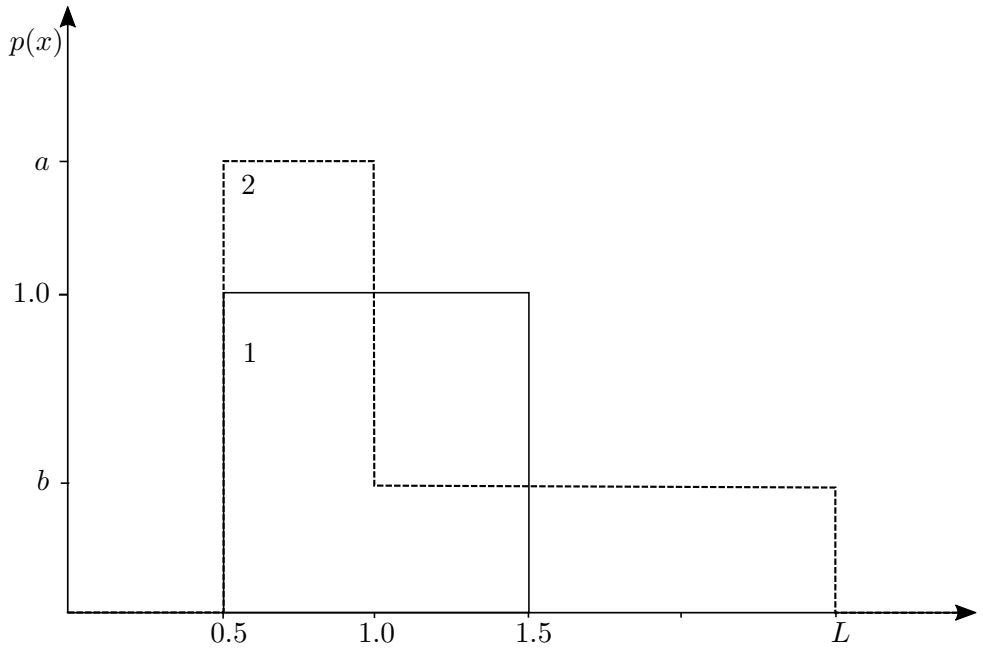


Figure 2.3: Two PDFs.

- Both are valid probability distributions:

$$p(x) \geq 0 \quad \text{and} \quad \int_{-\infty}^{\infty} p(x)\,dx = 1.$$

- Both have the same mean (expected value):

$$\int_{-\infty}^{\infty} xp(x)\,dx = 1.$$

- The variance of distribution 1 is:

$$\mathrm{Var}(x) = \mathrm{E}(x^2) - \mathrm{E}(x)^2 = \int_{0.5}^{1.5} x^2 dx - 1 = \frac{x^3}{3}\Big|_{0.5}^{1.5} - 1 = \frac{1}{12}.$$

- The variance of distribution 2, as $L \to \infty$ is:

$$\begin{aligned}\mathrm{Var}(x) &= \int_{0.5}^{1.0} ax^2 dx + \int_{1}^{L} bx^2 dx - 1 = \frac{4(L-1)}{(2L-1)}\frac{x^3}{3}\Big|_{0.5}^{1.0} + \frac{1}{(L-1)(2L-1)}\frac{x^3}{3}\Big|_{1.0}^{L} - 1 \\ &= \frac{2L^3 + 7L^2 - 14L + 5}{12L^2 - 18L + 6} - 1 \to \infty.\end{aligned}$$

This means that the variance for probability distribution 2 can be made arbitrary large, hence it would be preferable to have the first scenario even though both have the same expected value. Note that if we put variance into the cost function, the problem becomes much more complicated and we cannot simply apply DPA as described in Sec. 2.3 anymore. △

3. The Dynamic Programming Algorithm (cont'd)

Last lecture we introduced the DPA. In this lecture, we first apply the DPA to the chess match example, and then show how to deal with problems that do not match the standard form outlined in Section 2.1.

Example 1: Chess match strategy (revisited)

Consider a two game chess match with an opponent. Our objective is to develop a strategy that maximizes the chance of winning the match. Each game can have one of two outcomes: 1) Win/Lose: 1 point for the winner, 0 for the loser; 2) Draw: 0.5 points for each player. In addition, if at the end of two games the score is equal, the players keep on playing new games until one wins, and thereby wins the match (also known as sudden death).
There are two possible playing styles for our player: timid and bold. When playing timid, our player draws with probability p_d and loses with probability $(1 - p_d)$. When playing bold, our player wins with probability p_w and loses with probability $(1 - p_w)$. We also assume that $p_d > p_w$, a necessary condition for this problem to make sense.

We want to find a control policy that maximizes the probability of winning the match. We will solve this using the DPA (replacing min with max).

- The *state* x_k is the difference between our player's score and the opponent's score at the end of game k. That is, $x_0 = 0, x_1 \in \mathcal{S}_1 = \{-1, 0, 1\}, x_2 \in \mathcal{S}_2 = \{-2, -1, 0, 1, 2\}$.
- The *control inputs* u_k are the two playing styles, that is $u_k \in \mathcal{U} = \{\text{timid}, \text{bold}\}$.
- *Dynamics:* model as a finite state system using transition probabilities (see Section 1.3):

  $$x_{k+1} = w_k, \quad k = 0, 1$$

  where

  $$\begin{aligned} &\Pr(w_k = x_k | u_k = \text{timid}) = p_d \\ &\Pr(w_k = x_k - 1 | u_k = \text{timid}) = 1 - p_d \\ &\Pr(w_k = x_k + 1 | u_k = \text{bold}) = p_w \\ &\Pr(w_k = x_k - 1 | u_k = \text{bold}) = 1 - p_w \end{aligned}$$

  with $p_d > p_w$.
- *Cost*: we want to maximize the probability of winning. This is equivalent to the standard form

  $$g_2(x_2) + \sum_{k=0}^{1} g_k(x_k, u_k, w_k)$$

where

$$g_k(x_k, u_k, w_k) = 0, \quad \forall k \in \{0, 1\}$$
$$g_2(x_2) = \begin{cases} 1 & \text{if } x_2 > 0 \\ p_w & \text{if } x_2 = 0 \\ 0 & \text{if } x_2 < 0 \end{cases}$$

To see that the expected cost is equal to the probability of winning P_{win}, let

$$q_+ := \Pr(x_2 > 0), \quad q_0 := \Pr(x_2 = 0) \quad q_- := \Pr(x_2 < 0).$$

The probability of winning is

$$P_{\text{win}} = q_+ + q_0 p_w$$

and the expected value of the cost is

$$\mathrm{E}[g_2(x_2)] = q_+ \cdot 1 + q_0 \cdot p_w + q_- \cdot 0 = q_+ + q_0 p_w.$$

Now apply DPA:
Initialization:

$$J_2(\mathrm{x}) = \begin{cases} 1 & \text{if } \mathrm{x} > 0 \\ p_w & \text{if } \mathrm{x} = 0 \\ 0 & \text{if } \mathrm{x} < 0 \end{cases}$$

Recursion:

$$\begin{aligned} J_k(\mathrm{x}) &= \max_{\mathrm{u} \in \mathcal{U}} \mathop{\mathrm{E}}_{(w_k | x_k = \mathrm{x}, u_k = \mathrm{u})} [g_k(x_k, u_k, w_k) + J_{k+1}(x_{k+1})], \quad \forall \mathrm{x} \in \mathcal{S}_k, \forall k \in \{0, 1\} \\ &= \max_{\mathrm{u} \in \mathcal{U}} \mathop{\mathrm{E}}_{(w_k | x_k = \mathrm{x}, u_k = \mathrm{u})} [J_{k+1}(w_k)] \\ &= \max \left\{ \underbrace{p_d J_{k+1}(\mathrm{x}) + (1 - p_d) J_{k+1}(\mathrm{x} - 1)}_{\text{timid}}, \underbrace{p_w J_{k+1}(\mathrm{x} + 1) + (1 - p_w) J_{k+1}(\mathrm{x} - 1)}_{\text{bold}} \right\}. \end{aligned}$$

Henceforth the first entry of the maximum will denote the cost associated with timid play and the second with bold play.
$\underline{k = 1}$:

$$J_1(\mathrm{x}) = \max \{ p_d J_2(\mathrm{x}) + (1 - p_d) J_2(\mathrm{x} - 1), \; p_w J_2(\mathrm{x} + 1) + (1 - p_w) J_2(\mathrm{x} - 1) \}$$

- $x_1 = 1$:

  $$J_1(1) = \max \{ p_d + (1 - p_d) p_w, \; p_w + (1 - p_w) p_w \}$$

  Comparing the two entries yields:

  $$\begin{aligned} &(p_d + (1 - p_d) p_w) - (p_w + (1 - p_w) p_w) \\ &= (p_d - p_w)(1 - p_w) \; > 0 \text{ (since } p_d > p_w) \end{aligned}$$

  Therefore $\mu_1^*(1) = \text{timid}$ and $J_1(1) = p_d + (1 - p_d) p_w$.

- $x_1 = 0$:

$$J_1(0) = \max\{p_d p_w + (1-p_d)\cdot 0,\ p_w + (1-p_w)\cdot 0\} = \max\{p_d p_w,\ p_w\}$$

Therefore $\mu_1^*(0) =$ bold and $J_1(0) = p_w$.

- $x_1 = -1$:

$$J_1(-1) = \max\{p_d \cdot 0 + (1-p_d)\cdot 0,\ p_w p_w + (1-p_w)\cdot 0\} = \max\left\{0,\ p_w^2\right\}$$

Therefore $\mu_1^*(-1) =$ bold and $J_1(-1) = p_w^2$.

$\underline{k=0}$:

$$J_0(\mathrm{x}) = \max\{p_d J_1(\mathrm{x}) + (1-p_d)J_1(\mathrm{x}-1)\,,\ p_w J_1(\mathrm{x}+1) + (1-p_w)J_1(\mathrm{x}-1)\}$$

- $x_0 = 0$:

$$\begin{aligned} J_0(0) &= \max\{p_d J_1(0) + (1-p_d)J_1(-1)\,,\ p_w J_1(1) + (1-p_w)J_1(-1)\} \\ &= \max\left\{p_d p_w + (1-p_d)p_w^2,\ p_w(p_d + (1-p_d)p_w) + (1-p_w)p_w^2\right\} \\ &= \max\left\{p_d p_w + (1-p_d)p_w^2,\ p_d p_w + (1-p_d)p_w^2 + (1-p_w)p_w^2\right\} \end{aligned}$$

Therefore $\mu_0^*(0) =$ bold and $J_0(0) = p_d p_w + (1-p_d)p_w^2 + (1-p_w)p_w^2$.

Optimal match strategy: Play timid iff ahead in the score. △

3.1 Converting non-standard problems to the standard form

At first glance, the class of systems of our standard problem formulation in Section 1.1 may seem limiting but is in fact general enough to handle other types of problems via state-augmentation. We include some examples below.

3.1.1 Time Lags

Assume the dynamics have the following form:

$$x_{k+1} = f_k\left(x_k, x_{k-1}, u_k, u_{k-1}, w_k\right)$$

- Let $y_k := x_{k-1}$, $s_k := u_{k-1}$, and the augmented state vector $\tilde{x}_k := (x_k, y_k, s_k)$.
- The dynamics of the augmented state then become

$$\tilde{x}_{k+1} = \begin{bmatrix} x_{k+1} \\ y_{k+1} \\ s_{k+1} \end{bmatrix} = \begin{bmatrix} f_k\left(x_k, y_k, u_k, s_k, w_k\right) \\ x_k \\ u_k \end{bmatrix} =: \tilde{f}_k\left(\tilde{x}_k, u_k, w_k\right)$$

which now matches the standard form. Note that this procedure works for an arbitrary number of time lags.

3.1.2 Correlated Disturbances

Disturbances w_k that are correlated across time (colored noise) can commonly be modeled as the output of a linear system driven by independent random variables as follows:

$$\begin{aligned} w_k &= C_k y_{k+1} \\ y_{k+1} &= A_k y_k + \xi_k \end{aligned}$$

where A_k, C_k are given and $\xi_k,\ k = 0, \ldots, N-1$, are independent random variables.

- Let the augmented state vector $\tilde{x}_k := (x_k, y_k)$. Note that now y_k must be observed at time k, which can be done using a state estimator.
- The dynamics of the augmented state then become

  $$\tilde{x}_{k+1} = \begin{bmatrix} x_{k+1} \\ y_{k+1} \end{bmatrix} = \begin{bmatrix} f_k\left(x_k, u_k, C_k(A_k y_k + \xi_k)\right) \\ A_k y_k + \xi_k \end{bmatrix} =: \tilde{f}_k\left(\tilde{x}_k, u_k, \xi_k\right)$$

  which now matches the standard form.

3.1.3 Forecasts

Consider the case where at each time period we have access to a forecast that reveals the probability distribution of w_k, and possibly of future disturbances.
For example, assume that w_k is independent of x_k and u_k. At the beginning of each period k, we receive a prediction y_k (forecast) that w_k will attain a probability distribution out of a given finite collection of distributions $\left\{p_{w_k|y_k}(\cdot|1), p_{w_k|y_k}(\cdot|2), ..., p_{w_k|y_k}(\cdot|m)\right\}$. In particular, we receive a forecast that $y_k = i$ and thus $p_{w_k|y_k}(\cdot|i)$ is used to generate w_k. Furthermore, the forecast itself has a given *a-priori* probability distribution, namely,

$$y_{k+1} = \xi_k,$$

where the ξ_k are independent random variables taking value $i \in \{1, 2, ..., m\}$ with probability $p_{\xi_k}(i)$.

- Let the augmented state vector $\tilde{x}_k := (x_k, y_k)$. Since the forecast y_k is known at time k, we still have perfect state information.
- We define our new disturbance as $\tilde{w}_k := (w_k, \xi_k)$, with probability distribution

  $$\begin{aligned} p(\tilde{w}_k|\tilde{x}_k, u_k) &= p(w_k, \xi_k|x_k, y_k, u_k) \\ &= p(w_k|x_k, y_k, u_k, \xi_k)\, p(\xi_k|x_k, y_k, u_k) \\ &= p(w_k|y_k)\, p(\xi_k)\,. \end{aligned}$$

  Note that w_k depends only on $\tilde{x}_k$ (in particular y_k), and ξ_k does not depend on anything.
- The dynamics therefore become

  $$\tilde{x}_{k+1} = \begin{bmatrix} x_{k+1} \\ y_{k+1} \end{bmatrix} = \begin{bmatrix} f_k\left(x_k, u_k, w_k\right) \\ \xi_k \end{bmatrix} =: \tilde{f}_k\left(\tilde{x}_k, u_k, \tilde{w}_k\right).$$

  which now matches the standard form.
- The associated DPA becomes:

Initialization

$$J_N(\tilde{\mathrm{x}}) = J_N(\mathrm{x}, \mathrm{y}) = g_N(\mathrm{x}), \quad \mathrm{x} \in \mathcal{S}_N,\ \mathrm{y} \in \{1, \ldots, m\}$$

Recursion

$$\begin{aligned}
J_k(\tilde{\mathrm{x}}) &= J_k(\mathrm{x}, \mathrm{y}) \\
&= \min_{\mathrm{u} \in \mathcal{U}_k(x_k)} \mathop{\mathrm{E}}_{(\tilde{w}_k | \tilde{x}_k = \tilde{\mathrm{x}}, u_k = \mathrm{u})} \Big[g_k(x_k, u_k, w_k) + J_{k+1}(f_k(x_k, u_k, w_k), \xi_k) \Big] \\
&\overset{1}{=} \min_{\mathrm{u} \in \mathcal{U}_k(x_k)} \mathop{\mathrm{E}}_{(w_k | y_k = \mathrm{y})} \left[\mathop{\mathrm{E}}_{\xi_k} \Big[g_k(\mathrm{x}, \mathrm{u}, w_k) + J_{k+1}(f_k(\mathrm{x}, \mathrm{u}, w_k), \xi_k) \Big] \right] \\
&\overset{2}{=} \min_{\mathrm{u} \in \mathcal{U}_k(x_k)} \mathop{\mathrm{E}}_{(w_k | y_k = \mathrm{y})} \Big[g_k(\mathrm{x}, \mathrm{u}, w_k) + \mathop{\mathrm{E}}_{\xi_k} [J_{k+1}(f_k(\mathrm{x}, \mathrm{u}, w_k), \xi_k)] \Big] \\
&= \min_{\mathrm{u} \in \mathcal{U}_k(x_k)} \mathop{\mathrm{E}}_{(w_k | y_k = \mathrm{y})} \left[g_k(\mathrm{x}, \mathrm{u}, w_k) + \sum_{i=1}^{m} p_{\xi_k}(i)\, J_{k+1}(f_k(\mathrm{x}, \mathrm{u}, w_k), i) \right] \\
&\qquad\qquad \forall \mathrm{x} \in \mathcal{S}_k,\ \forall \mathrm{y} \in \{1, \ldots, m\},\ \forall k = N-1, \ldots, 0.
\end{aligned}$$

Steps:

1. Using $p(\tilde{w}_k | \tilde{x}_k, u_k) = p(w_k | y_k)\, p(\xi_k)$.
2. Since $g_k(x_k, u_k, w_k)$ is not a function of the random variable ξ_k.

4. Infinite Horizon Problems

In the next three lectures, we will look at how to solve the standard problem that we have been considering as the time horizon N goes to infinity. We consider the time-invariant setup as follows:

Dynamics

- The state evolution is governed by the time-invariant system:

$$x_{k+1} = f(x_k, u_k, w_k), \quad x_k \in \mathcal{S},\ u_k \in \mathcal{U}(x_k),\ w_k \sim p_{w|x,u}, \quad k = 0, \ldots, N-1. \tag{4.1}$$

  It is assumed that w_k is conditionally independent with all prior variables x_l, u_l, w_l, $l < k$, given x_k and u_k. Note that the PDF of w_k given x_k, u_k is time-invariant.

- As usual, the control inputs u_k are generated by an admissible policy $\pi \in \Pi$:

$$\pi = (\mu_0(\cdot), \mu_1(\cdot), \ldots, \mu_{N-1}(\cdot))$$

  such that

$$u_k = \mu_k(x_k), \quad u_k \in \mathcal{U}(x_k),\ \forall x_k \in \mathcal{S},\ \forall k. \tag{4.2}$$

Cost

Let the cost function be a sum of time-invariant stage costs, namely:

$$\sum_{k=0}^{N-1} g(x_k, u_k, w_k).$$

Given $\mathrm{x} \in \mathcal{S}$, the expected closed loop cost of starting at x associated with policy $\pi \in \Pi$ is then

$$J_\pi(\mathrm{x}) = \mathop{\mathrm{E}}_{(X_1, W_0 | x_0 = \mathrm{x})} \left[\sum_{k=0}^{N-1} g(x_k, \mu_k(x_k), w_k) \right] \quad \text{subject to (4.1) and (4.2)} \tag{4.3}$$

where $X_1 := (x_1, \ldots, x_N)$ and $W_0 := (w_0, \ldots, w_{N-1})$.

Objective

Construct an optimal policy π^* with associated optimal cost $J^*(\mathrm{x}) = J_{\pi^*}(\mathrm{x})$ such that for all $\mathrm{x} \in \mathcal{S}$,

$$\pi^* = \arg\min_{\pi \in \Pi} J_\pi(\mathrm{x}).$$

As we shall see, as the time horizon N goes to infinity, the complexity of the solution collapses. The intuition is the following: since the dynamics and stage cost are time-invariant, and the

time horizon is infinite, the cost-to-go is itself time-invariant. We will now write down the DPA and explore what happens as N goes to infinity:

$$J_N(\mathrm{x}) = 0, \quad \forall \mathrm{x} \in \mathcal{S}.$$

$$J_k(\mathrm{x}) = \min_{\mathrm{u} \in \mathcal{U}(\mathrm{x})} \mathop{\mathrm{E}}_{(w|x=\mathrm{x}, u=\mathrm{u})} \left[g(x,u,w) + J_{k+1}(f(x,u,w)) \right], \quad \forall \mathrm{x} \in \mathcal{S}, \quad \forall k = N-1, ..., 0.$$

Let $l := N - k$ and $V_l(\cdot) := J_{N-l}(\cdot)$. Then the above becomes

$$V_0(\mathrm{x}) = 0, \quad \forall \mathrm{x} \in \mathcal{S}.$$

$$V_l(\mathrm{x}) = \min_{\mathrm{u} \in \mathcal{U}(\mathrm{x})} \mathop{\mathrm{E}}_{(w|x=\mathrm{x}, u=\mathrm{u})} \left[g(x,u,w) + V_{l-1}(f(x,u,w)) \right], \quad \forall \mathrm{x} \in \mathcal{S}, \quad \forall l = 1, ..., N. \tag{4.4}$$

Now assume that for each $\mathrm{x} \in \mathcal{S}$, the sequence $V_l(\mathrm{x})$ converges to some value $J(\mathrm{x})$ as N approaches infinity. Then, (4.4) becomes

$$J(\mathrm{x}) = \min_{\mathrm{u} \in \mathcal{U}(\mathrm{x})} \mathop{\mathrm{E}}_{(w|x=\mathrm{x}, u=\mathrm{u})} \left[g(x,u,w) + J(f(x,u,w)) \right], \quad \forall \mathrm{x} \in \mathcal{S}, \tag{BE}$$

which is known as the **Bellman Equation** (BE). Assuming this convergence, $J(\mathrm{x})$ is the optimal cost-to-go $J^*(\mathrm{x})$. This suggests that the optimal policy is time-invariant, or *stationary*, and the point-wise minimization for every $\mathrm{x} \in \mathcal{S}$ yields that policy.

The BE may seem simple but it has to be solved for all $\mathrm{x} \in \mathcal{S}$ simultaneously. We can do this analytically only for a very small set of problems (for example, the Linear Quadratic Regulator (LQR) problem). In this lecture, we will be looking at a specific problem structure under which this convergence assumption is true.

4.1 The Stochastic Shortest Path (SSP) Problem

We now consider a subclass of problems for which solving the BE yields the optimal cost-to-go and an optimal stationary policy.

Dynamics

Consider the finite state, time-invariant system

$$\begin{aligned} &x_{k+1} = w_k, \qquad x_k \in \mathcal{S}, \\ &\Pr(w_k = j | x_k = i, u_k = \mathrm{u}) = P_{ij}(\mathrm{u}), \qquad \mathrm{u} \in \mathcal{U}(i), \end{aligned} \tag{4.5}$$

where $\mathcal{S}$ is a finite set and $\mathcal{U}(\mathrm{x})$ is a finite set for all $\mathrm{x} \in \mathcal{S}$. Note the transition probabilities are time-invariant as opposed to the general time-varying case of Section 1.3.

Cost

Given an initial state $i \in \mathcal{S}$, the expected closed loop cost of starting at i associated with policy π becomes

$$J_\pi(i) = \mathop{\mathrm{E}}_{(X_1, W_0 | x_0 = i)} \left[\sum_{k=0}^{N-1} g(x_k, \mu_k(x_k), w_k) \right] \quad \text{subject to (4.5) and (4.2).} \tag{4.6}$$

Assumption 4.1. *There exists a cost-free termination state, which we designate as state* 0*. In particular, there are* $n+1$ *states with* $\mathcal{S} = \{0, 1, \dots, n\}$*, where*

$$P_{00}(u) = 1 \text{ and } g(0, \mathrm{u}, 0) = 0, \quad \forall \mathrm{u} \in \mathcal{U}(0).$$

Assumption 4.1 is used to make the expected cost meaningful. We can think of the termination state as a destination state where we can land and stop accumulating cost.

Objective

As before, we want to construct an optimal policy π^* such that for all $i \in \mathcal{S}$,

$$\pi^* = \arg\min_{\pi \in \Pi} J_\pi(i),$$

and explore what happens as the time horizon N goes to infinity.

4.2 Theorem: SSP and BE

For simplicity, and with a slight abuse of notation, we let a stationary policy $\pi = (\mu(\cdot), \mu(\cdot), \dots)$ be represented by μ. A stationary policy μ is said to be *proper* if, when using this policy, there exists an integer m such that

$$\Pr(x_m = 0 | x_0 = i) > 0 \quad \text{subject to (4.5) and (4.2)}, \forall i \in \mathcal{S}.$$

A stationary policy that is not proper is said to be *improper*.

Assumption 4.2. *There exists at least one proper policy* $\mu \in \Pi$*. Furthermore, for every improper policy* μ'*, the corresponding cost function* $J_{\mu'}(i)$ *is infinity for at least one state* $i \in \mathcal{S}$*.*

Assumption 4.2 is required for there to exist a unique solution to the BE for the SSP problem, which will then be the optimal cost. It ensures that a policy exists for which the probability of reaching the termination state goes to one as the time horizon N goes to infinity. It also ensures that the policies for which this does not occur incur infinite cost, which ensures there are no non-positive cycles; this will be further discussed in future lectures on Shortest Path Problems.

Theorem 4.1. *Under assumptions 4.1 and 4.2, the following are true for the SSP:*

1) Given any initial conditions $V_0(1), \ldots, V_0(n)$*, the sequence* $V_l(i)$ *generated by the iteration*

$$V_{l+1}(i) = \min_{\mathrm{u}\in\mathcal{U}(i)} \left(q(i,\mathrm{u}) + \sum_{j=1}^{n} P_{ij}(\mathrm{u}) V_l(j) \right), \quad \forall\, i \in \mathcal{S}^+ \tag{4.7}$$

where $\mathcal{S}^+ := \mathcal{S}\backslash\{0\}$ *and*

$$q(i,\mathrm{u}) := \mathop{\mathrm{E}}_{(w|x=i,u=\mathrm{u})} \left[g(x,u,w) \right],$$

converges to the optimal cost $J^*(i)$ *for all* $i \in \mathcal{S}^+$*;*

2) The optimal costs satisfy the Bellman Equation (BE)*:*

$$J^*(i) = \min_{\mathrm{u}\in\mathcal{U}(i)} \left(q(i,\mathrm{u}) + \sum_{j=1}^{n} P_{ij}(\mathrm{u}) J^*(j) \right) \quad \forall\, i \in \mathcal{S}^+;$$

3) The solution to the BE is unique;

4) The minimizing u *for each* $i \in \mathcal{S}^+$ *of the BE gives an optimal policy, which is proper.*

We will not provide a rigorous proof. Instead, an intuition will be given under the stronger Assumption 4.3 (below) in place of Assumption 4.2:

Assumption 4.3. *There exists an integer* m *such that for* ***any*** *admissible policy* $\pi \in \Pi$*:*

$$\Pr\left(x_m = 0 | x_0 = i\right) > 0 \quad \textit{subject to } (4.5) \textit{ and } (4.2), \forall i \in \mathcal{S},$$

i.e. there is a positive probability that the termination state will be reached for any admissible policy regardless of the initial state.

Intuition 1):
Let $V_0(0) = 0$. For fixed N, consider the following expected cost starting at $x_0 = i$ for all $i \in \mathcal{S}$:

$$J_\pi(i) = \mathop{\mathrm{E}}_{(X_1,W_0|x_0=i)} \left[\sum_{k=0}^{N-1} g(x_k, \mu_k(x_k), w_k) + V_0(x_N) \right] \quad \text{subject to (4.5) and (4.2)}, \tag{4.8}$$

where $V_0(x_N)$ from 1) can be interpreted as a terminal cost. As N goes to infinity, the probability that the termination state 0 is reached approaches one for all policies by Assumption 4.3. Thus $V_0(x_N)$ does not influence (4.8) in the limit, since x_N will be 0 with probability 1, and $V_0(0) = 0$. Therefore, (4.8) is equivalent to (4.6) in the limit.
The corresponding DPA is then:

$$J_N(i) = V_0(i), \quad \forall i \in \mathcal{S}.$$

$$\begin{aligned} J_k(i) &= \min_{\mathrm{u}} \mathop{\mathrm{E}}_{(w|x=i,u=\mathrm{u})} \left[g(x,u,w) + J_{k+1}(w) \right] \\ &= \min_{\mathrm{u}} \left(q(i,\mathrm{u}) + \sum_{j=0}^{n} P_{ij}(\mathrm{u}) J_{k+1}(j) \right), \quad \forall i \in \mathcal{S}, \quad k = N-1, ..., 0 \end{aligned}$$

$$= \min_{\mathrm{u}} \left(q(i,\mathrm{u}) + \sum_{j=1}^{n} P_{ij}(\mathrm{u}) J_{k+1}(j) \right), \quad \forall i \in \mathcal{S}^+, \quad k = N-1, ..., 0. \tag{4.9}$$

Note that the last step comes from the fact that $q(0,\mathrm{u}) = 0$ and $P_{0j}(\mathrm{u}) = 0$ for all $j \in \mathcal{S}^+$ by Assumption 4.1, and we have initialized $J_N(0) = V_0(0) = 0$. Therefore, $J_k(0) = 0$ for $k = 0, ..., N$. $J_0(i)$ in (4.9) is the optimal cost for the cost function (4.8).
Let $l := N - k$ and $V_l(\cdot) := J_{N-l}(\cdot)$. Then the DPA recursion above becomes

$$V_l(i) = \min_{\mathrm{u}} \left(q(i,\mathrm{u}) + \sum_{j=1}^{n} P_{ij}(\mathrm{u}) V_{l-1}(j) \right), \quad \forall i \in \mathcal{S}^+, \quad l = 1, ..., N. \tag{4.10}$$

which is precisely the iteration in 1). Furthermore, $V_N(i) = J_0(i)$ is the optimal cost for the cost function (4.8) and the optimal cost for (4.6) in the limit. Because of Assumption 4.3, it can be shown that $V_l(i)$ converges, and we thus have $\lim_{l\to\infty} V_l(i) = J^*(i)$.

Intuition 2)
It follows from taking limits of both sides of (4.7) in 1).

Intuition 3)
Let $J_0(1), \ldots, J_0(n)$ and $\bar{J}_0(1), \ldots, \bar{J}_0(n)$ be two different solutions of Bellman's Equation. If we use both solutions as initial conditions for iteration (4.7) in 1), they both converge after 1 iteration in the DP recursion. This leads to two different optimal costs which is a contradiction.

Intuition 4)
Let μ be the stationary policy inferred from the minimizing u in 2). We then have

$$\begin{aligned} J^*(i) &= \min_{\mathrm{u}\in\mathcal{U}(i)} \left(q(i,\mathrm{u}) + \sum_{j=1}^{n} P_{ij}(\mathrm{u}) J^*(j) \right) \\ &= \left(q(i,\mu(i)) + \sum_{j=1}^{n} P_{ij}(\mu(i)) J^*(j) \right) \quad \forall\, i \in \mathcal{S}^+ \end{aligned} \tag{4.11}$$

Consider the modified problem where $\tilde{\mathcal{U}}(i) = \{\mu(i)\}$ for all $i \in \mathcal{S}$, then by 1) with arbitrary initialization we converge to the optimal cost J_μ for the modified problem. The BE for the modified problem has the following form

$$J_\mu(i) = \left(q(i,\mu(i)) + \sum_{j=1}^{n} P_{ij}(\mu(i)) J_\mu(j) \right) \quad \forall\, i \in \mathcal{S}^+ \tag{4.12}$$

Note that we do not need the minimization here since there is only one choice in the allowable control space.
By 3), (4.12) has a unique solution and (4.11) is the same as (4.12). Thus $J_\mu(i)$ is equal to $J^*(i)$ for all $i \in \mathcal{S}^+$. μ therefore incurs the optimal cost $J^*(i)$ and is an optimal policy.

5. Solving the Bellman Equation

In the next two lectures, we will look at several methods to solve Bellman's Equation (BE) for the stochastic shortest path problem: Value Iteration, Policy Iteration and Linear Programming.

5.1 Value Iteration (VI)

Value iteration simply applies the DP recursion introduced in Theorem 4.1 1):

$$V_{l+1}(i) = \min_{u \in \mathcal{U}(i)} \left(q(i,u) + \sum_{j=1}^{n} P_{ij}(u) V_l(j) \right), \quad \forall i \in \mathcal{S}^+$$

with arbitrary initialization $V_0(i)$ for all $i \in \mathcal{S}^+$, until it converges. For brevity we define $\mathcal{S}^+ := \mathcal{S} \backslash \{0\}$.
Generally, value iteration requires an *infinite* number of iterations to converge to the optimum $J^*(\cdot)$. In practice, we usually define a threshold for $\|V_{l+1}(i) - V_l(i)\|$, $\forall i \in \mathcal{S}^+$, at which the algorithm terminates.

5.2 Policy Iteration (PI)

Policy iteration, as its name indicates, iterates over policies instead of values (costs). We now introduce a theorem before we present the PI algorithm.

Theorem 5.1. *Under Assumption 4.1, for any proper policy $\mu \in \Pi$, the associated cost vector $J_\mu := (J_\mu(1), ..., J_\mu(n))$ is the unique solution of*

$$J_\mu(i) = q(i, \mu(i)) + \sum_{j=1}^{n} P_{ij}(\mu(i)) J_\mu(j), \quad \forall i \in \mathcal{S}^+. \tag{5.1}$$

Furthermore, given any initial conditions V_0, the sequence V_l generated by the recursion:

$$V_{l+1}(i) = q(i, \mu(i)) + \sum_{j=1}^{n} P_{ij}(\mu(i)) V_l(j), \quad \forall i \in \mathcal{S}^+ \tag{5.2}$$

converges to J_μ.

Proof. Note that this theorem is a special case of Theorem 4.1: consider a modified problem where the only allowable control at state i is $\mu(i)$. Thus $\Pi = \{\mu\}$, containing only the proper policy under consideration, thereby satisfying Assumption 4.2. Then, invoke Theorem 4.1 1), 2), and 3) to yield Theorem 5.1. □

Now we state the PI algorithm and prove that it will yield the optimal cost and an optimal policy.

Policy Iteration

Initialization: Initialize with a proper policy $\mu^0 \in \Pi$.

Stage 1 (*Policy Evaluation*): Given a policy μ^h, solve for the corresponding cost J_{μ^h} by solving the linear system of equations

$$J_{\mu^h}(i) = q\Big(i, \mu^h(i)\Big) + \sum_{j=1}^{n} P_{ij}(\mu^h(i)) J_{\mu^h}(j)\,,\ \forall i \in \mathcal{S}^+. \tag{5.3}$$

Stage 2 (*Policy Improvement*): Obtain a new stationary policy μ^{h+1} as follows

$$\mu^{h+1}(i) = \arg\min_{u \in \mathcal{U}(i)} \left(q(i,u) + \sum_{j=1}^{n} P_{ij}(u) J_{\mu^h}(j) \right),\ \forall i \in \mathcal{S}^+. \tag{5.4}$$

Iterate between Stage 1 and 2 until $J_{\mu^{h+1}}(i) = J_{\mu^h}(i)$ for all $i \in \mathcal{S}^+$.

Theorem 5.2. *Under Assumptions 4.1 and 4.2, the policy iteration converges to an optimal policy after a finite number of steps.*

Proof. For a fixed h and a proper policy μ^h, consider the following auxiliary recursion in l:

$$\begin{aligned} V_0(i) &= J_{\mu^h}(i) \\ V_{l+1}(i) &= q\Big(i, \mu^{h+1}(i)\Big) + \sum_{j=1}^{n} P_{ij}(\mu^{h+1}(i)) V_l(j)\,,\ \forall i \in \mathcal{S}^+. \end{aligned} \tag{5.5}$$

Then, for all $i \in \mathcal{S}^+$:

$$\begin{aligned} J_{\mu^h}(i) &= V_0(i) \\ &\overset{1}{=} q\Big(i, \mu^h(i)\Big) + \sum_{j=1}^{n} P_{ij}(\mu^h(i)) V_0(j) \\ &\overset{2}{\geq} q\Big(i, \mu^{h+1}(i)\Big) + \sum_{j=1}^{n} P_{ij}(\mu^{h+1}(i)) V_0(j) \\ &\overset{3}{=} V_1(i) \\ &\overset{4}{\geq} q\Big(i, \mu^{h+1}(i)\Big) + \sum_{j=1}^{n} P_{ij}(\mu^{h+1}(i)) V_1(j) \\ &\overset{3}{=} V_2(i) \end{aligned}$$

Steps:

1. By (5.1) in Theorem 5.1 applied to policy μ^h.
2. By (5.4).
3. By (5.5).

4. Since $V_1(i) \leq V_0(i)$ for all $i \in \mathcal{S}^+$ (from steps 1 to 3).

Therefore,

$$V_0(i) \geq V_1(i) \geq V_2(i) \geq ... \geq V_l(i), \quad \forall i \in \mathcal{S}^+ \tag{5.6}$$

Note that the policy μ^{h+1} is also proper: by contradiction, assume μ^{h+1} is improper, thus by Assumption 4.2, $J_{\mu^{h+1}}(i)$ goes to infinity for at least one state i as the time horizon goes to infinity. Furthermore, note that (5.5) is the DP recursion after the index substitution $l := N-k$ for the following expected cost function

$$\begin{aligned}\tilde{J}_{\mu^{h+1}}(i) &= \underset{(X_1,W_0|x_0=i)}{\mathrm{E}}\left[\sum_{k=0}^{N-1} g\Big(x_k, \mu^{h+1}(x_k), w_k\Big) + J_{\mu^h}(x_N)\right] \\ &= J_{\mu^{h+1}}(i) + \underset{(X_1,W_0|x_0=i)}{\mathrm{E}}\left[J_{\mu^h}(x_N)\right]\end{aligned} \tag{5.7}$$

subject to the dynamics with the control space constrained to $\mathcal{U}(i) = \{\mu^{h+1}(i)\}$, thus $V_N(i) = \tilde{J}_{\mu^{h+1}}(i)$ for all $i \in \mathcal{S}$. Since $J_{\mu^{h+1}}(i)$ goes to infinity for some state i as N goes to infinity, the finite expected value of the terminal costs $J_{\mu^h}(x_N)$ become inconsequential in (5.7) for that state. Thus $V_l(i)$ will approach $J_{\mu^{h+1}}(i)$ in the limit and is infinite, which contradicts (5.6). Therefore, μ^{h+1} is proper.

The above implies that (5.3) always has a unique solution by Theorem 5.1. Furthermore, as $l \to \infty$, $V_l \to J_{\mu^{h+1}}$ and therefore $J_{\mu^h}(i) \geq J_{\mu^{h+1}}(i)$ for all $i \in \mathcal{S}^+$. Since the number of stationary policies is finite, we will eventually have $J_{\mu^h} = J_{\mu^{h+1}}$ for some finite h.

Once we have converged, it follows from **Stage 2** that

$$J_{\mu^{h+1}}(i) = J_{\mu^h}(i) = \min_{u \in \mathcal{U}(i)} \left(q(i,u) + \sum_{j=1}^{n} P_{ij}(u) J_{\mu^h}(j) \right), \ i \in \mathcal{S}^+$$

This is the BE and thus we have converged to an optimal policy $\mu^* = \mu^h$ and the optimal cost $J^* = J_{\mu^h}$. □

5.3 Analogy and Comparison between VI and PI

At first glance, PI looks very different from VI, but actually they share a lot in common. Let us rewrite the iteration in VI:

Stage 1 (*Value Update*):

$$V_{l+1}(i) = q\Big(i, \mu^l(i)\Big) + \sum_{j=1}^{n} P_{ij}(\mu^l(i)) V_l(j), \quad \forall i \in \mathcal{S}^+$$

Stage 2 (*Policy Improvement*):

$$\mu^{l+1}(i) = \arg\min_{u \in \mathcal{U}(i)} \left(q(i,u) + \sum_{j=1}^{n} P_{ij}(u) V_{l+1}(j) \right), \quad \forall i \in \mathcal{S}^+$$

PI and VI differ in their respective Stage 1: PI performs a policy evaluation, which solves a system of linear equations and is equivalent to running the value update of VI an infinite number of times (Theorem 5.1).

Computational Complexity

Let p denote the maximum size of the control space $\mathcal{U}(i)$ for all $i \in \mathcal{S}^+$.

- Complexity of PI:
  - Stage 1 of PI involves solving a system of n linear equations in n unknowns, which has a computational complexity of $\mathcal{O}(n^3)$.
  - Stage 2 involves n minimizations over p possible control inputs, and evaluating the sum takes n steps. Thus the complexity is $\mathcal{O}(n^2 p)$.

  This yields a total complexity of $\mathcal{O}(n^2(n+p))$ *at each iteration.*
- Complexity of VI: it involves n minimizations over p possible control inputs, and evaluating the sum takes n steps. Thus the complexity is $\mathcal{O}(n^2 p)$ *at each iteration.*
- Thus at each iteration, PI is more computationally expensive than VI. But remember: theoretically it takes an infinite number iterations for VI to converge, whereas with PI, in the worst case the number of iterations is p^n (though in practice PI terminates much sooner).

5.4 Variants of VI and PI

Gauss-Seidel Update

In practice, VI would be implemented as follows:

For $i = 1$ to n:

$$\bar{V}(i) \leftarrow \min_{u \in \mathcal{U}(i)} \left(q(i,u) + \sum_{j=1}^{n} P_{ij}(u) V(j) \right),$$

For $i = 1$ to n:

$$V(i) \leftarrow \bar{V}(i).$$

that is, the current iteration's cost has to be updated for all states simultaneously, and the previous iteration's cost vector has to be stored.
We can use a **Gauss-Seidel Update** which updates the V in place as follows:

For $i = 1$ to n:

$$V(i) \leftarrow \min_{u \in \mathcal{U}(i)} \left(q(i,u) + \sum_{j=1}^{n} P_{ij}(u) V(j) \right).$$

Here we iterate one state at a time while incorporating into the computation the interim results. The Gauss-Seidel Update is a generic technique for solving iterative equations. This method usually leads to faster convergence than the ordinary VI and requires less memory.

Asynchronous Policy Iteration

Under mild conditions[1], all combinations of the following steps will converge:

[1] Essentially, the value and policy updates are executed an infinite number of times for all states.

- Any number of value updates in between policy updates;
- Any number of states updated at each value update;
- Any number of states updated at each policy update.

5.5 Connections to Linear Algebra

In Theorem 5.1 (and in Stage 1 of PI), we solve a linear system of equations that has the following form:

$$J = G + PJ$$

with $J := J_\mu \in \mathbb{R}^n$ whose i-th entry represents the cost-to-go at the corresponding state $i \in \mathcal{S}^+$, $G := (q(1, \mu(1)), \ldots, q(n, \mu(n))) \in \mathbb{R}^n$ is the stage cost vector, and $P \in \mathbb{R}^{n \times n}$ is the probability transition matrix under the proper policy μ, whose $(i,j)^{\text{th}}$ entry is $P_{ij}(\mu(i))$. We can rewrite this as

$$(I - P)J = G$$

It is easy to see that there exists a unique solution for J if and only if $(I - P)$ is invertible. We will now show that $(I - P)$ is guaranteed to be invertible when the policy is proper. Note that $(I - P)$ is invertible if and only if P does not have eigenvalue(s) at 1. Furthermore, the (i,j)-th entry of P^2 is

$$\begin{aligned}
&\sum_{\xi \in \mathcal{S}^+} P_{i\xi}(\mu(i)) P_{\xi j}(\mu(\xi)) \\
&= \sum_{\xi \in \mathcal{S}^+} \Pr(x_2 = j | x_1 = \xi) \Pr(x_1 = \xi | x_0 = i) \\
&\overset{1}{=} \sum_{\xi \in \mathcal{S}} \Pr(x_2 = j | x_1 = \xi) \Pr(x_1 = \xi | x_0 = i) \\
&\overset{2}{=} \sum_{\xi \in \mathcal{S}} \Pr(x_2 = j | x_1 = \xi, x_0 = i) \Pr(x_1 = \xi | x_0 = i) \\
&\overset{3}{=} \sum_{\xi \in \mathcal{S}} \Pr(x_2 = j, x_1 = \xi | x_0 = i) \\
&\overset{4}{=} \Pr(x_2 = j | x_0 = i)
\end{aligned}$$

where the above are subject to $u_k = \mu(x_k)$.
Steps:

1. Since $\Pr(x_2 = j | x_1 = 0) = 0$ for all $j \in \mathcal{S}^+$.
2. Recall the dynamics are $x_{k+1} = w_k$, and by our system assumption, w_k is conditionally independent of all prior states, disturbances and inputs.
3. Use the conditioning rule on x_1.
4. Use the sum rule.

If we continue on, the (i,j)-th entry of P^N is $\Pr(x_N = j | x_0 = i)$. When the policy μ is proper, $\Pr(x_N = j | x_0 = i)$ goes to 0 as N goes to infinity for all $i, j \in \mathcal{S}^+$. Thus the entire matrix P^N vanishes as N approaches infinity, which means that all eigenvalues of P must have modulus less than 1. Therefore, there is no eigenvalue of P that is 1, and so the inverse of $(I - P)$ exists if the policy is proper.
Furthermore, note that

$$\begin{aligned}
&(I - P)(I + P + P^2 + P^3 + ...) \\
&= (I + P + P^2 + P^3 + ...) - (P + P^2 + P^3 + P^4 + ...) \\
&= I
\end{aligned}$$

since $P^N \to 0$ as $N \to \infty$ for proper policies. Thus,

$$(I - P)^{-1} = \sum_{k=0}^{\infty} P^k.$$

From the iteration (5.2) in Theorem 5.1, we have the sequence:

$$\begin{aligned}
J_1 &= G + PJ_0 \\
J_2 &= G + PJ_1 = G + PG + P^2 J_0 \\
&\vdots \\
J_N &= (I + P + P^2 + P^3 + ... + P^{N-1})G + P^N J_0
\end{aligned}$$

which by the above, gives $J_N \to (I - P)^{-1} G$ as $N \to \infty$, and is thus consistent with the claim of Theorem 5.1.

5.6 Linear Programming (LP)

Recall VI:

$$V_{l+1}(i) = \min_{u \in \mathcal{U}(i)} \left(q(i,u) + \sum_{j=1}^{n} P_{ij}(u) V_l(j) \right), \quad \forall i \in \mathcal{S}^+. \tag{5.8}$$

Suppose that we use value iteration to generate a sequence of vectors V_l starting with an initial vector V_0 that satisfies

$$V_0(i) \leq \min_{u \in \mathcal{U}(i)} \left(q(i,u) + \sum_{j=1}^{n} P_{ij}(u) V_0(j) \right), \quad \forall i \in \mathcal{S}^+, \tag{5.9}$$

i.e.

$$V_0(i) \leq \left(q(i,u) + \sum_{j=1}^{n} P_{ij}(u) V_0(j) \right), \quad \forall u \in \mathcal{U}(i),\ \forall i \in \mathcal{S}^+. \tag{5.10}$$

Then by (5.8) we have

$$V_0(i) \leq V_1(i), \quad \forall i \in \mathcal{S}^+$$

Therefore,

$$\begin{aligned} V_2(i) &= \min_{u \in \mathcal{U}(i)} \left(q(i,u) + \sum_{j=1}^{n} P_{ij}(u) V_1(j) \right) \\ &\geq \min_{u \in \mathcal{U}(i)} \left(q(i,u) + \sum_{j=1}^{n} P_{ij}(u) V_0(j) \right) \\ &= V_1(i), \quad \forall i \in \mathcal{S}^+ \end{aligned}$$

This leads to

$$V_{l+1}(i) \geq V_l(i) \quad \forall i \in \mathcal{S}^+, \forall l.$$

By VI we know that $V_l(i)$ converges to $J^*(i)$ as l goes to infinity. We thus have

$$J^*(i) \geq V_0(i), \quad \forall i \in \mathcal{S}^+, \tag{5.11}$$

where (5.11) is valid for **any** V_0 that satisfies (5.10). Since J^* also satisfies (5.9) with equality (Bellman Equation), J^* is the "largest" V_0 that satisfies the constraint (5.10). We can write this as an optimization problem as in Theorem 5.2 below.

Theorem 5.2. *The solution to the optimization problem*

$$\begin{aligned} \underset{V}{\text{maximize}} \quad & \sum_{i \in \mathcal{S}^+} V(i) \\ \text{subject to} \quad & V(i) \leq \left(q(i,u) + \sum_{j=1}^{n} P_{ij}(u) V(j) \right), \quad \forall u \in \mathcal{U}(i),\ \forall i \in \mathcal{S}^+. \end{aligned} \tag{5.12}$$

also solves the Bellman Equation to yield the optimal cost J^* for the SSP problem.

Note that in the optimization problem, both the objective function and the constraints are linear in V. This is known as a *Linear Program*. In general, Linear Programs can be solved efficiently and can handle millions of variables, with many mature solvers available (CPLEX, CVXGEN, CVXPY, FORCES Pro, linprog in Matlab).

Proof. Let V^* be the solution to the linear program and thus satisfies the inequality constraint (5.12). We know from (5.11) that the inequality constraint implies that $V^*(i) \leq J^*(i)$, for all $i \in \mathcal{S}^+$. By contradiction, assume that $V^* \neq J^*$, thus there exists a state $\bar{i} \in \mathcal{S}^+$ such that

$$V^*(\bar{i}) < J^*(\bar{i}).$$

Thus,

$$\sum_{i \in \mathcal{S}^+} V^*(i) < \sum_{i \in \mathcal{S}^+} J^*(i).$$

But since J^* also satisfies the constraint (5.12) as it solves the BE, V^* is not the solution to the linear program, which is a contradiction. □

6. Discounted Problems

We now consider a class of infinite horizon problems called discounted problems, where future stage costs are discounted exponentially, and there is no assumption of a termination state. We will show that this is equivalent to an associated stochastic shortest path problem.

Dynamics

$$\begin{aligned} &\tilde{x}_{k+1} = \tilde{w}_k, \quad \tilde{x}_k \in \mathcal{S}^+,\ k = 0, \ldots, N-1 \\ &p_{\tilde{w}|\tilde{x},\tilde{u}}(j|i,u) = \tilde{P}_{ij}(u), \quad u \in \tilde{\mathcal{U}}(\tilde{x}_k) \end{aligned} \tag{6.1}$$

where $\mathcal{S}^+ = \{1, \ldots, n\}$ is a finite set and $\tilde{\mathcal{U}}(x)$ is a finite set for all $x \in \mathcal{S}^+$, and as usual, the $\tilde{w}_k$ are independent of all previous variables when conditioned on $\tilde{x}_k, \tilde{u}_k$. Note that there is

no explicit termination state required, and thus Assumptions 4.1 and 4.2 do not necessarily hold.

As usual, the control inputs $\tilde{u}_k$ are generated by an admissible policy $\tilde{\pi} \in \tilde{\Pi}$:

$$\tilde{\pi} = (\tilde{\mu}_0(\cdot)\,, \tilde{\mu}_1(\cdot)\,, \ldots, \tilde{\mu}_{N-1}(\cdot))$$

such that

$$\tilde{u}_k = \tilde{\mu}_k(\tilde{x}_k)\,, \;\; \tilde{u}_k \in \tilde{\mathcal{U}}(\tilde{x}_k),\; \forall \tilde{x}_k \in \mathcal{S}^+,\; k = 0, \ldots, N-1. \tag{6.2}$$

Cost

Given an initial state $i \in \mathcal{S}^+$, the expected closed loop cost of starting at i associated with policy $\tilde{\pi} \in \tilde{\Pi}$ is

$$\tilde{J}_{\tilde{\pi}}(i) = \underset{(\tilde{X}_1, \tilde{W}_0 | \tilde{x}_0 = i)}{\mathrm{E}} \left[\sum_{k=0}^{N-1} \alpha^k \tilde{g}(\tilde{x}_k, \tilde{\mu}_k(\tilde{x}_k)\,, \tilde{w}_k) \right] \quad \text{subject to (6.1) and (6.2)} \tag{6.3}$$

where $\tilde{X}_1 := (\tilde{x}_1, \ldots, \tilde{x}_N)$, $\tilde{W}_0 := (\tilde{w}_0, \ldots, \tilde{w}_{N-1})$, and $\alpha \in (0, 1)$ is called the *discount factor*.

Objective

Construct an optimal policy $\tilde{\pi}^*$ with associated optimal cost $\tilde{J}^*(i) = \tilde{J}_{\tilde{\pi}^*}(i)$ such that for all $i \in \mathcal{S}^+$,

$$\tilde{\pi}^* = \underset{\tilde{\pi} \in \tilde{\Pi}}{\arg\min}\, J_{\tilde{\pi}}(i)\,,$$

and explore what happens as the time horizon N goes to infinity.

We now define an auxiliary stochastic shortest path problem and show that it is equivalent to the discounted problem presented above.

Auxiliary stochastic shortest path problem

We will drop the tilde notation to denote quantities related to the auxiliary problem. For example, $\mathcal{S}$ denotes the state space of the auxiliary SSP.

- *State*: $x_k \in \mathcal{S} = \mathcal{S}^+ \cup \{0\} = \{0, 1, \ldots, n\}$, where 0 is a virtual terminal state.
- *Control*: $u_k \in \mathcal{U}(x_k),\; \forall x_k \in \mathcal{S}$ where

  $$\mathcal{U}(x_k) := \tilde{\mathcal{U}}(x_k)\; \forall x_k \in \mathcal{S}^+,$$
  $$\mathcal{U}(0) := \{\texttt{stay}\}$$

  where `stay` is a virtual control action that is applied when the state is the virtual termination state. The control inputs u_k are generated by an admissible policy $\pi \in \Pi$:

  $$\pi = (\mu_0(\cdot)\,, \mu_1(\cdot)\,, \ldots, \mu_{N-1}(\cdot))$$

  such that

  $$u_k = \mu_k(x_k)\,, \;\; u_k \in \mathcal{U}(x_k),\; \forall x_k \in \mathcal{S},\; \forall k. \tag{6.4}$$

- *Dynamics*:

$$x_{k+1} = w_k, \quad x_k \in \mathcal{S} \tag{6.5}$$

where the transition probabilities are generated from

$$\begin{aligned}
p_{w|x,u}(j|i,u) &= P_{ij}(u) := \alpha \tilde{P}_{ij}(u), \quad u \in \mathcal{U}(i), \quad \forall i,j \in \mathcal{S}^+ \\
p_{w|x,u}(0|i,u) &= P_{i0}(u) := 1-\alpha, \quad u \in \mathcal{U}(i), \quad \forall i \in \mathcal{S}^+ \\
p_{w|x,u}(j|0,u) &= P_{0j}(u) := 0, \quad u = \texttt{stay}, \quad \forall j \in \mathcal{S}^+ \\
p_{w|x,u}(0|0,u) &= P_{00}(u) := 1, \quad u = \texttt{stay}.
\end{aligned}$$

Note that this is a valid transition probability distribution since for any $i \in \mathcal{S}^+$, and for any $u \in \mathcal{U}(i)$,

$$\sum_{j \in \mathcal{S}} P_{ij}(u) = \sum_{j \in \mathcal{S}^+} \alpha \tilde{P}_{ij}(u) + P_{i0}(u) = \alpha \cdot 1 + (1-\alpha) = 1,$$

and for $i = 0$, $u = \texttt{stay}$,

$$\sum_{j \in \mathcal{S}} P_{0j}(u) = \sum_{j \in \mathcal{S}^+} P_{0j}(u) + P_{00}(u) = 0 + 1 = 1.$$

It is clear that since $\alpha < 1$, we have a non-zero probability to go to state 0, regardless of the control input applied and the initial state. Hence both Assumptions 4.1 and 4.2 are satisfied.

- *Cost*: The stage costs are defined as

$$\begin{aligned}
g(x_k, u_k, w_k) &= \alpha^{-1} \tilde{g}(x_k, u_k, w_k), \quad \forall u_k \in \mathcal{U}(x_k), \ \forall x_k, w_k \in \mathcal{S}^+ \\
g(x_k, u_k, 0) &= 0, \quad \forall u_k \in \mathcal{U}(x_k), \ \forall x_k \in \mathcal{S}^+, \\
g(0, \texttt{stay}, 0) &= 0.
\end{aligned}$$

The total expected closed loop cost starting at $r \in \mathcal{S}$ associated with policy $\pi \in \Pi$ is

$$J_\pi(r) = \mathop{\mathrm{E}}_{(X_1, W_0 | x_0 = r)} \left[\sum_{k=0}^{N-1} g(x_k, \mu_k(x_k), w_k) \right] \quad \text{subject to (6.4) and (6.5)}, \tag{6.6}$$

where $X_1 := (x_1, \ldots, x_N)$, $W_0 := (w_0, \ldots, w_{N-1})$.

Note that there is a one-to-one mapping between a policy π of the auxiliary problem to a policy $\tilde{\pi}$ of the discounted problem: the feedback law of the auxiliary problem just trivially assigns $\mu_k(0) = \texttt{stay}$, and for the rest of the states $x_k \in \mathcal{S}$ they remain the same.
Next, we show that the cost $J_\pi(i) = \tilde{J}_{\tilde{\pi}}(i), \forall i \in \mathcal{S}^+$, where π is the mapping of $\tilde{\pi}$.
First note that for any $k > 0$ and for any $i, j, r \in \mathcal{S}$,

$$p_{x_k, w_k | x_0}(i, j | r) = p_{w_k | x_k, x_0}(j | i, r) p_{x_k | x_0}(i | r) = p_{w_k | x_k, x_0}(j | i, r) p_{w_{k-1} | x_0}(i | r),$$

which follows from (6.5). We further note that (6.4) and the independence assumption on w_k implies

$$p_{w_k | x_k, x_0}(j | i, r) = p_{w|x,u}(j | i, \mu_k(i)),$$

which can be seen by applying the total probability theorem over u_k and using $u_k = \mu_k(x_k)$. Combining these results and applying the total probability theorem over x_{k-1} results in

$$\begin{aligned} p_{x_k,w_k|x_0}(i,j|r) &= \sum_{\bar{x}_{k-1}\in\mathcal{S}} p_{w|x,u}(j|i,\mu_k(i)) p_{x_{k-1},w_{k-1}|x_0}(\bar{x}_{k-1},i|r) \\ &= \sum_{\bar{x}_{k-1}\in\mathcal{S}} \sum_{\bar{x}_{k-2}\in\mathcal{S}} p_{w|x,u}(j|i,\mu_k(i)) p_{w|x,u}(i|\bar{x}_{k-1},\mu_{k-1}(\bar{x}_{k-1}))\cdot \\ &\qquad\qquad p_{x_{k-2},w_{k-2}|x_0}(\bar{x}_{k-2},\bar{x}_{k-1}|r). \end{aligned}$$

Continuing the recursion leads to

$$\begin{aligned} p_{x_k,w_k|x_0}(i,j|r) = &\sum_{\bar{x}_{k-1}\in\mathcal{S}} \sum_{\bar{x}_{k-2}\in\mathcal{S}} \cdots \sum_{\bar{x}_1\in\mathcal{S}} p_{w|x,u}(j|i,\mu_k(i)) p_{w|x,u}(i|\bar{x}_{k-1},\mu_{k-1}(\bar{x}_{k-1})) \\ &p_{w|x,u}(\bar{x}_{k-1}|\bar{x}_{k-2},\mu_{k-2}(\bar{x}_{k-2}))\ldots p_{w|x,u}(\bar{x}_1|r,\mu_0(r)), \end{aligned}$$

which holds for all $i,j,r\in\mathcal{S}$. Applying the same argument to $\tilde{x}_k$ and $\tilde{w}_k$ yields

$$\begin{aligned} p_{\tilde{x}_k,\tilde{w}_k|\tilde{x}_0}(i,j|r) = &\sum_{\bar{x}_{k-1}\in\mathcal{S}^+} \sum_{\bar{x}_{k-2}\in\mathcal{S}^+} \cdots \sum_{\bar{x}_1\in\mathcal{S}^+} p_{\tilde{w}|\tilde{x},\tilde{u}}(j|i,\tilde{\mu}_k(i)) p_{\tilde{w}|\tilde{x},\tilde{u}}(i|\bar{x}_{k-1},\tilde{\mu}_{k-1}(\bar{x}_{k-1})) \\ &p_{\tilde{w}|\tilde{x},\tilde{u}}(\bar{x}_{k-1}|\bar{x}_{k-2},\tilde{\mu}_{k-2}(\bar{x}_{k-2}))\ldots p_{\tilde{w}|\tilde{x},\tilde{u}}(\bar{x}_1|r,\tilde{\mu}_0(r)), \end{aligned}$$

for all $i,j,r\in\mathcal{S}^+$.
In the special case that $i\neq 0$ and $j\neq 0$ we have that $p_{w|x,u}(j|i,\mu_k(i)) = \alpha p_{\tilde{w}|\tilde{x},\tilde{u}}(j|i,\tilde{\mu}_k(i))$ for all $i,j\in\mathcal{S}^+$. This implies that

$$\begin{aligned} p_{x_k,w_k|x_0}(i,j|r) =&\alpha^{k+1} \sum_{\bar{x}_{k-1}\in\mathcal{S}^+} \sum_{\bar{x}_{k-2}\in\mathcal{S}^+} \cdots \sum_{\bar{x}_1\in\mathcal{S}^+} p_{\tilde{w}|\tilde{x},\tilde{u}}(j|i,\tilde{\mu}_k(i)) p_{\tilde{w}|\tilde{x},\tilde{u}}(i|\bar{x}_{k-1},\tilde{\mu}_{k-1}(\bar{x}_{k-1})) \\ &p_{\tilde{w}|\tilde{x},\tilde{u}}(\bar{x}_{k-1}|\bar{x}_{k-2},\tilde{\mu}_{k-2}(\bar{x}_{k-2}))\ldots p_{\tilde{w}|\tilde{x},\tilde{u}}(\bar{x}_1|r,\tilde{\mu}_0(r)) \\ =&\alpha^{k+1} p_{\tilde{x}_k,\tilde{w}_k|\tilde{x}_0}(i,j|r) \end{aligned}$$

for any $r\neq 0$, since $x_k\neq 0$ implies by our assumption that all $x_l\neq 0$ for all $l<k$.
Consequently, due to the fact that $g(0,\mu_k(0),0) = 0$ and $g(i,\mu_k(i),0) = 0$,

$$\begin{aligned} \underset{(X_1,W_0|x_0=r)}{\mathrm{E}}\left[g(x_k,\mu_k(x_k),w_k)\right] &= \sum_{i\in\mathcal{S}}\sum_{j\in\mathcal{S}} g(i,\mu_k(i),j) p_{x_k,w_k|x_0}(i,j|r) \\ &= \sum_{i\in\mathcal{S}^+}\sum_{j\in\mathcal{S}^+} g(i,\mu_k(i),j) p_{x_k,w_k|x_0}(i,j|r) \\ &= \sum_{i\in\mathcal{S}^+}\sum_{j\in\mathcal{S}^+} \alpha^{k+1} g(i,\mu_k(i),j) p_{\tilde{x}_k,\tilde{w}_k|\tilde{x}_0}(i,j|r) \\ &= \sum_{i\in\mathcal{S}^+}\sum_{j\in\mathcal{S}^+} \alpha^{k} \tilde{g}(i,\mu_k(i),j) p_{\tilde{x}_k,\tilde{w}_k|\tilde{x}_0}(i,j|r) \\ &= \underset{(\tilde{X}_1,\tilde{W}_0|\tilde{x}_0=r)}{\mathrm{E}}\left[\alpha^k \tilde{g}(\tilde{x}_k,\tilde{\mu}_k(\tilde{x}_k),\tilde{w}_k)\right], \end{aligned}$$

provided that $r\neq 0$. This shows that $J_\pi(i) = \tilde{J}_{\tilde{\pi}}(i), \forall i\in\mathcal{S}^+$.
In conclusion, (6.3) is equivalent to (6.6) for any state $i\in\mathcal{S}^+$. This means that the mapping of the policy that minimizes (6.6) minimizes (6.3). By solving the Bellman Equation for the auxiliary problem, we also obtain an optimal policy and optimal cost-to-go for the infinite

horizon discounted problem. Furthermore, from the Bellman Equation for the auxiliary problem we can derive the Bellman Equation for the discounted problem:

$$\begin{aligned} J^*(i) &= \min_{u \in \mathcal{U}(i)} \left(q(i,u) + \sum_{j=1}^{n} P_{ij}(u) J^*(j) \right), \quad \forall i \in \mathcal{S}^+ \\ &= \min_{u \in \tilde{\mathcal{U}}(i)} \left(q(i,u) + \alpha \sum_{j=1}^{n} \tilde{P}_{ij}(u) J^*(j) \right), \quad \forall i \in \mathcal{S}^+, \end{aligned}$$

where

$$q(i,u) = \sum_{j=1}^{n} P_{ij}(u) g(i,u,j) = \sum_{j=1}^{n} (\alpha \tilde{P}_{ij}(u))(\alpha^{-1} \tilde{g}(i,u,j)) = \sum_{j=1}^{n} \tilde{P}_{ij}(u) \tilde{g}(i,u,j).$$

6.1 Connection to Linear Algebra

Recall that in Theorem 5.1 (and in Stage 1 of policy iteration) applied to the augmented SSP, we solve a linear system of equations that has the following form:

$$J = G + PJ$$

with $J := J_\mu \in \mathbb{R}^n$ whose i-th entry represents the cost-to-go at the corresponding state $i \in \mathcal{S}^+$, $G := (q(1, \mu(1)), \ldots, q(n, \mu(n))) \in \mathbb{R}^n$ is the stage cost vector, and $P \in \mathbb{R}^{n \times n}$ is the probability transition matrix under the policy μ, whose (i,j)-th entry is $P_{ij}(\mu(i))$. We have shown that this is equivalent to

$$J = G + \alpha \tilde{P} J$$

where $\tilde{P} \in \mathbb{R}^{n \times n}$ is the probability transition matrix of the discounted problem under the equivalent policy $\tilde{\mu}$, whose (i,j)-th entry is $\tilde{P}_{ij}(\tilde{\mu}(i))$. We can rewrite this as

$$(I - \alpha \tilde{P}) J = G.$$

It can be shown that $\tilde{P}$ has eigenvalues with modulus less than or equal to one. Thus, $(\alpha \tilde{P})^N = \alpha^N \tilde{P}^N \to 0$ as N approaches infinity, which means that all eigenvalues of $\alpha \tilde{P}$ must have modulus strictly less than 1, and so the inverse of $I - \alpha \tilde{P}$ is guaranteed to exist.

7. Shortest Path Problems and Deterministic Finite State Systems

In the next two lectures we will look at shortest path problems, where the objective is to find the shortest path from a start node to an end node. It will be shown that these problems are equivalent to the standard problem we have looked at in Lecture 2 when the system is deterministic and has a finite state space.

7.1 The Shortest Path (SP) Problem

Graph

We consider a graph which is defined by a finite vertex space $\mathcal{V}$ and a weighted edge space $\mathcal{C} := \{(i, j, c_{i,j}) \in \mathcal{V} \times \mathcal{V} \times \mathbb{R} \cup \{\infty\} \mid i, j \in \mathcal{V}\}$, where $c_{i,j}$ denotes the arc length or cost from vertex (or node) i to j. An example can be found in Fig 7.1. If there is no edge from node i to node j, then $c_{i,j} = \infty$.

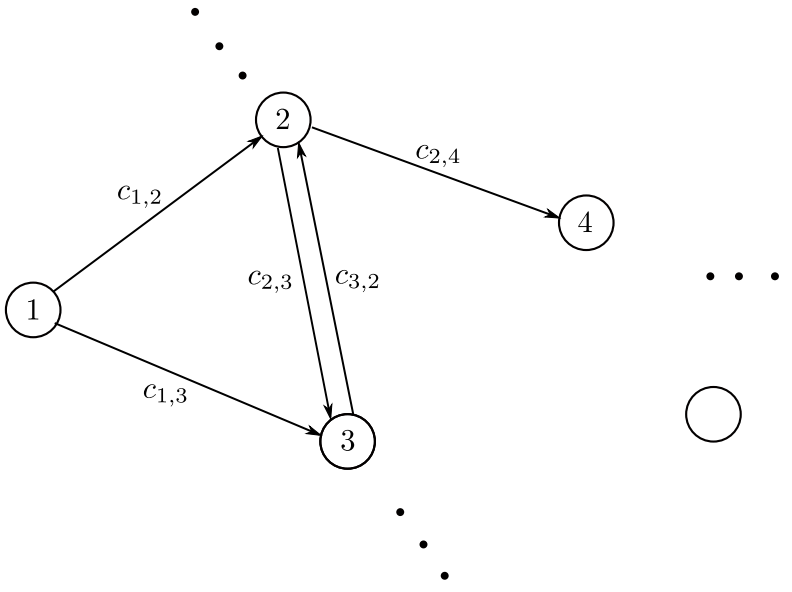


Figure 7.1: Graph with nodes and edges with associated lengths.

Paths

A path is defined as an ordered list of nodes, that is

$$Q := (i_1, i_2, \ldots, i_q),$$

where each element of Q is in $\mathcal{V}$ and q is the number of nodes in path Q.

The set of all paths that start at some node $\mathsf{s} \in \mathcal{V}$ and end at node $\mathsf{t} \in \mathcal{V}$ is denoted by $\mathbb{Q}_{\mathsf{s},\mathsf{t}}$.

Path Length

The path length associated with path Q is the sum of the arc lengths over the path, that is

$$J_Q = \sum_{h=1}^{q-1} c_{i_h, i_{h+1}}.$$

Objective

Determine a path Q^* that has the smallest length from node $\mathsf{s} \in \mathcal{V}$ to $\mathsf{t} \in \mathcal{V}$, that is,

$$Q^* = \arg\min_{Q \in \mathbb{Q}_{\mathsf{s},\mathsf{t}}} J_Q.$$

In order for this problem to make sense, the following assumption must be satisfied:

Assumption 7.1. *For all $i \in \mathcal{V}$ and for all $Q \in \mathbb{Q}_{i,i}$, $J_Q \geq 0$.*

Assumption 7.1 ensures that there are no negative cycles in the graph. Note that this implies $c_{i,i} \geq 0$ for all $i \in \mathcal{V}$.

With further assumptions on the problem data, the SP problem can be solved directly using efficient label correcting methods as will be discussed next lecture. Another method, which will be shown in Section 7.3, is to map the SP problem to that of a deterministic, finite state system (reviewed in Section 7.2) on which we can apply standard DPA that we are familiar with.

7.2 Deterministic Finite State (DFS) Problem

Consider the standard problem in Section 1.1 where there are no disturbances w_k and the state space $\mathcal{S}_k$ is a finite set for all $k = 0, ..., N$:

Dynamics

$$x_{k+1} = f_k(x_k, u_k), \quad x_k \in \mathcal{S}_k,\ k = 0, \ldots, N,\ u_k \in \mathcal{U}_k(x_k),\ k = 0, \ldots, N-1 \tag{7.1}$$

Cost Function

$$g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k) \tag{7.2}$$

Objective

Since the problem is deterministic, feedback control results in no advantage in terms of cost reduction. Therefore, given $x_0 \in \mathcal{S}_0$, we want to construct an optimal control sequence $(u_0, \ldots, u_{N-1})$ that minimizes the cost function (7.2). Nonetheless, this problem can be solved using the DPA.

7.3 Equivalence of SP and DFS

7.3.1 DFS to SP

We construct the graph of the associated problem. Every state $x_k \in \mathcal{S}_k$ at each stage k is represented by a node in the graph. The given x_0 is designated as the starting node $\mathtt{s}$ and an artificial terminal node $\mathtt{t}$ is added to handle the final stage, such that the arc lengths to $\mathtt{t}$ are simply the terminal costs of the DFS. The arc length between any two nodes is the (smallest) stage cost between the corresponding time-state pairs and is infinite if there is no control action that links them.

To be precise, the vertex space is the union of all stage and state pairs, that is,

$$\begin{aligned}
\mathcal{V} &:= \left(\bigcup_{k=0}^{N} \mathcal{V}_k \right) \cup \{\mathtt{t}\}, \text{ where} \\
\mathcal{V}_0 &:= \{(0, x_0)\} \\
\mathcal{V}_k &:= \{(k, x_k) | x_k \in \mathcal{S}_k\}, \ k = 1, \ldots, N,
\end{aligned}$$

and $\mathtt{s} := (0, x_0)$.

The weighted edge space is then

$$\begin{aligned}
\mathcal{C} := & \\
& \left\{ \big((k, x_k), (k+1, x_{k+1}), c\big) \,\middle|\, \begin{array}{l} (k, x_k) \in \mathcal{V}_k \\ (k+1, x_{k+1}) \in \mathcal{V}_{k+1} \\ c = \min\limits_{\{u \in \mathcal{U}_k(x_k) | x_{k+1} = f_k(x_k, u)\}} g_k(x_k, u) \\ k \in \{0, \ldots, N-1\} \end{array} \right\} \bigcup \\
& \quad \big\{ \big((N, x_N),\, \mathtt{t},\, g_N(x_N)\big) \,\big|\, (N, x_N) \in \mathcal{V}_N \big\}.
\end{aligned}$$

Note that the min operator is subject to the control space constraint and the dynamics (to ensure that a control input can link the respective states). If the constraints cannot be satisfied, the min returns ∞. Furthermore, other unconnected edges have infinite arc length.
This conversion can be visualized in Fig. 7.2.

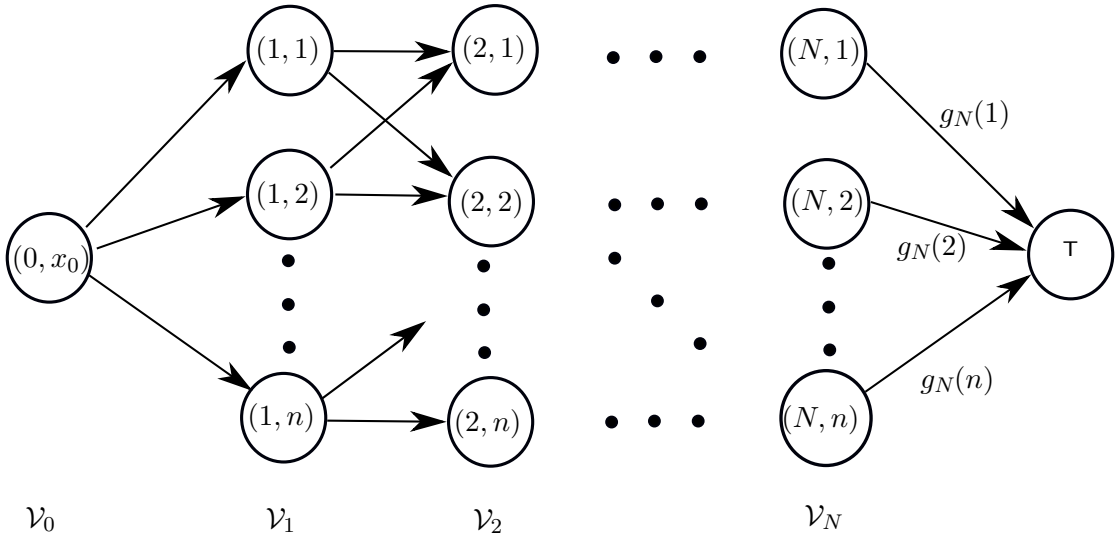


Figure 7.2: The graph corresponding to DFS. Without loss of generality, $\mathcal{S}_k = \{1, ..., n\}$ for $k = 1, ..., N$.

7.3.2 SP to DFS

Consider a graph with a vertex space $\mathcal{V}$ and a weighted edge space $\mathcal{C}$. We want to find a shortest path from any node $\mathsf{s} \in \mathcal{V}$ to $\mathsf{T} \in \mathcal{V}$. Since we exclude the possibility that a cycle has negative cost, it is clear that an optimal path need not have more than $|\mathcal{V}|$ elements, where $|\mathcal{V}|$ denotes the number of elements (the cardinality) of set $\mathcal{V}$. Assume $c_{i,i} = 0$ for all $i \in \mathcal{V}$, which allows for degenerate moves, that is, the steps where we don't move. We can formulate this problem as an $N := |\mathcal{V}| - 1$ stage DFS:

- State space: $\mathcal{S}_k := \mathcal{V} \backslash \{\mathsf{T}\}$ for $k = 1, ..., N-1$, $\mathcal{S}_N := \{\mathsf{T}\}$ and $\mathcal{S}_0 := \{\mathsf{s}\}$.
- Control space: $\mathcal{U}_k := \mathcal{V} \backslash \{\mathsf{T}\}$ for $k = 0, ..., N-2$, and $\mathcal{U}_{N-1} := \{\mathsf{T}\}$
- The dynamics are

$$x_{k+1} = u_k, \quad u_k \in \mathcal{U}_k, \ k = 0, \ldots, N-1$$

- The stage cost functions are

$$g_k(x_k, u_k) := c_{x_k, u_k}, \quad k = 0, ..., N-1$$
$$g_N(\mathsf{T}) := 0$$

  Note that stage costs can be infinite, which corresponds to the case that there is no edge between the nodes.

We can solve the above DFS using DPA, where $J_k(i)$ is the optimal cost of getting from node i to node T in $N - k = |\mathcal{V}| - 1 - k$ moves:

$$J_N(\mathsf{T}) = g_N(\mathsf{T}) = 0,$$
$$J_k(i) = \min_{u \in \mathcal{U}_k} \left(g_k(i, u) + J_{k+1}(u) \right), \quad \forall i \in \mathcal{S}_k, \ k = N-1, \ldots, 0$$
$$\Rightarrow J_{N-1}(i) = c_{i,T}, \quad \forall i \in \mathcal{V} \backslash \{\mathsf{T}\}$$

$$J_k(i) = \min_{j \in \mathcal{V}\backslash\{\mathsf{T}\}} \left(c_{i,j} + J_{k+1}(j)\right), \quad \forall i \in \mathcal{V}\backslash\{\mathsf{T}\},\ k = N-2, \ldots, 1$$
$$J_0(\mathsf{S}) = \min_{j \in \mathcal{V}\backslash\{\mathsf{T}\}} \left(c_{\mathsf{S},j} + J_1(j)\right)$$

Remarks:

- $J_0(\mathsf{S}) = J_{Q^*}$, where the path Q^* can be inferred from the minimizations after the removal of degenerate moves.
- We can terminate the algorithm early if $J_k(i) = J_{k+1}(i)$ for all $i \in \mathcal{V}\backslash\{\mathsf{T}\}$.

Forward DP Algorithm

From Fig. 7.1, it can be seen that shortest path problem is symmetric: an optimal path from S to T is also a shortest path from T to S, where all the arcs are "flipped", that is, the direction of each arc is reversed and its length remains the same.
We can therefore formulate the DPA for the "reverse" SP problem as follows. First we define an auxiliary SP problem. In particular, let

$$\tilde{c}_{j,i} := c_{i,j}, \quad \forall (i,j,c_{i,j}) \in \mathcal{C}$$

be the weighted edges of the auxiliary problem. The objective is to find the optimal path from T to S.
We can now apply the DP as in Section 7.3.2:

$$\begin{aligned}
J_{N-1}(j) &= \tilde{c}_{j,\mathsf{S}} = c_{\mathsf{S},j}, \quad \forall j \in \mathcal{V}\backslash\{\mathsf{s}\} \\
J_k(j) &= \min_{i \in \mathcal{V}\backslash\{\mathsf{S}\}} \left(\tilde{c}_{j,i} + J_{k+1}(i)\right), \quad \forall j \in \mathcal{V}\backslash\{\mathsf{s}\},\ k = N-2, \ldots, 1 \\
&= \min_{i \in \mathcal{V}\backslash\{\mathsf{S}\}} \left(c_{i,j} + J_{k+1}(i)\right), \quad \forall j \in \mathcal{V}\backslash\{\mathsf{s}\},\ k = N-2, \ldots, 1 \\
J_0(\mathsf{T}) &= \min_{i \in \mathcal{V}\backslash\{\mathsf{S}\}} \left(c_{i,\mathsf{T}} + J_1(i)\right).
\end{aligned}$$

Let $l := N - k$ and $J_l^{\mathsf{F}} := J_{N-l}$. Then the above become

$$\begin{aligned}
J_1^{\mathsf{F}}(j) &= c_{\mathsf{S},j}, \quad \forall j \in \mathcal{V}\backslash\{\mathsf{s}\} \\
J_l^{\mathsf{F}}(j) &= \min_{i \in \mathcal{V}\backslash\{\mathsf{S}\}} \left(c_{i,j} + J_{l-1}^{\mathsf{F}}(i)\right), \quad \forall j \in \mathcal{V}\backslash\{\mathsf{s}\},\ l = 2, \ldots, N-1 \\
J_N^{\mathsf{F}}(\mathsf{T}) &= \min_{i \in \mathcal{V}\backslash\{\mathsf{S}\}} \left(c_{i,\mathsf{T}} + J_{N-1}^{\mathsf{F}}(i)\right),
\end{aligned}$$

where $J_l^{\mathsf{F}}(j)$ is the optimal *cost-to-arrive* to node j from node S in l moves (in the original SP problem).

7.4 Hidden Markov Models and the Viterbi Algorithm

We now consider an estimation problem and show how it can be converted to an SP problem.

Dynamics

Consider the finite state, time-invariant[1] system

$$\begin{aligned} &x_{k+1} = w_k, \qquad x_k \in \mathcal{S}, \\ &P_{ij} := p_{w|x}(j|i)\,, \quad \forall i, j \in \mathcal{S} \end{aligned} \tag{7.3}$$

where $\mathcal{S} = \{1, \ldots, n\}$ is a finite set and the distribution $p_{w|x}$ is given. Furthermore, the initial state x_0 is not known but its distribution p_{x_0} is. This system is also known as a *Markov chain*.

Measurement Model

When a state transition occurs, the states before and after the transition are unknown (or "hidden") to us, but we obtain an observation z that relates to that transition, that is

$$M_{ij}(z) := p_{z|x,w}(z|i,j)\,, \quad z \in \mathcal{Z} \tag{7.4}$$

where $\mathcal{Z}$ is the measurement space, and the time-invariant distribution $p_{z|x,w}$ is given and is known as the *likelihood function*. Furthermore, Markov chains whose state transitions are imperfectly observed according to (7.4) are called *Hidden Markov Models*. We assume independent observations, that is, observation z_k is conditionally independent with all prior variables x_l, $l < k-1$ and z_l, $l < k$, given x_{k-1}, x_k.

Objective

Let $Z_i := (z_i, \ldots, z_N)$ and $X_i := (x_i, \ldots, x_N)$. Given a measurement sequence $Z_1 = (z_1, \ldots, z_N)$, we want to find the "most likely" state trajectory $X_0 = (x_0, \ldots, x_N)$. In particular, solve for a maximum a posteriori estimate $\hat{X}_0 := (\hat{x}_0, \ldots, \hat{x}_N)$ where

$$\hat{X}_0 = \arg\max_{X_0} p(X_0|Z_1)\,.$$

By the conditioning rule, $p(X_0, Z_1) = p(X_0|Z_1)\,p(Z_1)$. Since $p(Z_1)$ is fixed and non-negative, maximizing $p(X_0|Z_1)$ is equivalent to maximizing $p(X_0, Z_1)$. Note that

$$\begin{aligned} p(X_0, Z_1) &= p(x_0, X_1, Z_1) \\ &\overset{1}{=} p(X_1, Z_1|x_0)\,p(x_0) \\ &\overset{2}{=} p(X_2, Z_2|x_0, x_1, z_1)\,p(z_1, x_1|x_0)\,p(x_0) \\ &\overset{3}{=} p(X_2, Z_2|x_0, x_1, z_1)\,p(z_1|x_1, x_0)\,p(x_1|x_0)\,p(x_0) \\ &\overset{4}{=} p(X_2, Z_2|x_0, x_1, z_1)\,M_{x_0x_1}(z_1)P_{x_0x_1}p(x_0) \\ &\overset{5}{=} p(X_3, Z_3|x_0, x_1, z_1, x_2, z_2)\,p(z_2, x_2|z_1, x_1, x_0)\,M_{x_0x_1}(z_1)P_{x_0x_1}p(x_0) \\ &\overset{5}{=} p(X_3, Z_3|x_0, x_1, z_1, x_2, z_2)\,p(z_2|x_2, z_1, x_1, x_0)\,p(x_2|z_1, x_1, x_0)\,M_{x_0x_1}(z_1)P_{x_0x_1}p(x_0) \\ &\overset{6}{=} p(X_3, Z_3|x_0, x_1, z_1, x_2, z_2)\,M_{x_1x_2}(z_2)P_{x_1x_2}M_{x_0x_1}(z_1)P_{x_0x_1}p(x_0) \\ &\;\;\vdots \\ &= p(x_0)\prod_{k=1}^{N} P_{x_{k-1}x_k}M_{x_{k-1}x_k}(z_k) \end{aligned}$$

Steps:

[1] Can be easily extended to time-varying distributions as well.

1. Use conditioning rule on x_0
2. Use conditioning rule on x_1, z_1
3. Use conditioning rule on x_1
4. By (7.3) and (7.4)
5. Apply step 2 to step 3 on x_2 and z_2
6. z_2 is conditionally independent with prior states and measurements given x_2, x_1, (7.3) and (7.4).

Since the probabilities in the product are non-negative, maximizing $p(X_0, Z_1)$ is equivalent to

$$\underset{X_0}{\text{minimize}} \left(c_{\mathsf{S},(0,x_0)} + \sum_{k=1}^{N} c_{(k-1,x_{k-1}),(k,x_k)} \right) \tag{7.5}$$

where

$$c_{\mathsf{S},(0,x_0)} = \begin{cases} -\ln(p(x_0)) & \text{if } p(x_0) > 0 \\ \infty & \text{if } p(x_0) = 0 \end{cases}$$

$$c_{(k-1,x_{k-1}),(k,x_k)} = \begin{cases} -\ln(P_{x_{k-1}x_k} M_{x_{k-1}x_k}(z_k)) & \text{if } P_{x_{k-1}x_k} M_{x_{k-1}x_k}(z_k) > 0 \\ \infty & \text{if } P_{x_{k-1}x_k} M_{x_{k-1}x_k}(z_k) = 0 \end{cases}$$

We can now construct a graph of state-time pairs (similar to Section 7.3.1), with artificial starting node s and terminal node t, and the arc costs inferred from (7.5) (see Fig. 7.3).

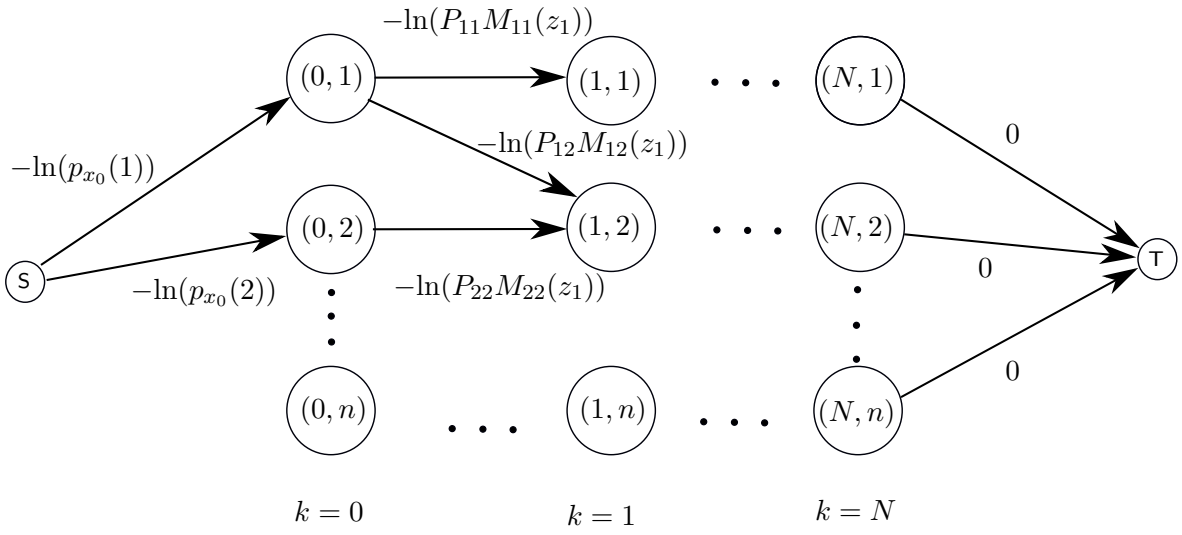


Figure 7.3: State estimation of a hidden Markov model viewed as a SP problem from s to t.

In practice, the shortest path is constructed sequentially by forward DP as costs to arrive. Therefore the optimal state estimates can be computed in real-time.

8. Shortest Path Algorithms

In this lecture we will look at more efficient approaches to solving shortest path (SP) problems. In SP problems, we are interested in reaching a destination node from a starting node in the shortest way possible, whereas the Dynamic Programming Algorithm (DPA) approach we saw last lecture solves the shortest path[1] from every node to the destination node.

8.1 Label correcting methods

We present a general algorithm used to solve SP problems called the Label Correcting Algorithm (LCA). The key idea is to progressively discover paths from s to other nodes $i \in \mathcal{V}$ and to keep track of the associated lengths to get to i in a variable d_i, called the *label* of i. It also makes use of a set OPEN which is a current list of nodes that could potentially be a part of the shortest path to T. We assume that $c_{i,j} \geq 0$ for all $(i, j, c_{i,j}) \in \mathcal{C}$, which ensures that Assumption 7.1 (no negative-cycles) is satisfied. The algorithm is described below and represented visually in Fig. 8.1.

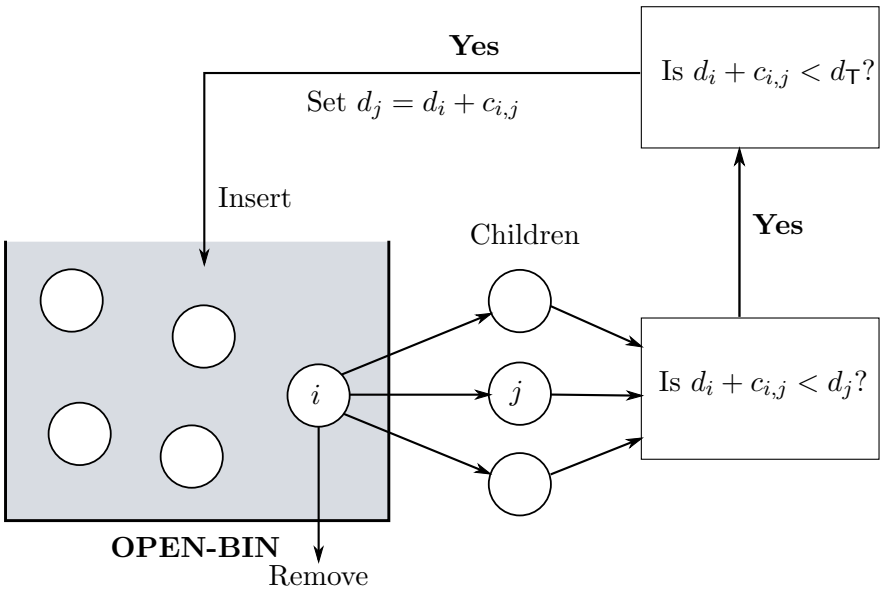


Figure 8.1: Label correcting algorithm. A node j is called a child of node i if there is an edge (i, j) connecting i with j.

[1]Note that "length" is synonymous with "cost" and "shortest path" is synonymous with "smallest cost". We use the two interchangeably, and both are used in the literature.

STEP 0: Place node s in OPEN, set $d_{\mathsf{S}} = 0$ and $d_j = \infty \; \forall j \in \mathcal{V} \backslash \{\mathsf{s}\}$.

STEP 1: Remove a node i from OPEN and execute step 2 for all children j of i (that is, for all nodes $j \in \mathcal{V}$ with $c_{i,j} < \infty$).

STEP 2: If $(d_i + c_{i,j}) < d_j$ *and* $(d_i + c_{i,j}) < d_{\mathsf{T}}$, set $d_j = d_i + c_{i,j}$ and set i to be the parent of j in the path from s to T. If $j \neq \mathsf{T}$, put j in OPEN (it may be already there).

STEP 3: If OPEN is empty, we are done. Else go to **STEP 1**.

Theorem 8.1. *If there exists at least one finite cost path from s to T, then the Label Correcting Algorithm (LCA) terminates with $d_{\mathsf{T}} = J_{Q^*}$. Otherwise, the LCA terminates with $d_{\mathsf{T}} = \infty$.*

Proof. We prove this in three steps: first we show that the LCA terminates in a finite number of steps; then we show that it terminates with $d_{\mathsf{T}} = \infty$ if there is no finite cost path from s to T; finally we prove that the algorithm terminates with $d_{\mathsf{T}} = J_{Q^*}$ if there exists at least one finite cost path.

1. The LCA always terminates: each time a node j enters the OPEN list, its label is decreased and becomes equal to the length of some path from s to j. Furthermore, the number of distinct lengths of paths from s to j that are smaller than any given number is finite (since by Assumption 7.1 the graph does not contain negative cycles). Therefore, there can only be a finite number of label reductions for each node j and hence, since the LCA removes nodes in Step 1, the algorithm will eventually terminate.

2. Note that a node $i \in \mathcal{V}$ is in OPEN only if there is a finite cost path from s to i. Suppose there is no finite cost path from s to T. Then, for any node i in OPEN, $c_{i,\mathsf{T}}$ cannot be finite, because if it were, then there would be a finite cost path from s to T. Thus by the LCA Step 2, d_{T} is never updated and remains at ∞.

3. Let $Q^* = (\mathsf{s}, i_1, i_2, \ldots, i_{q-2}, \mathsf{T}) \in \mathbb{Q}_{\mathsf{S},\mathsf{T}}$ be a shortest path from s to T with length J_{Q^*}.

   Note that by the principle of optimality, $Q^*_m = (\mathsf{s}, i_1, \ldots, i_m)$ is the shortest path from s to i_m, $m = 1, \ldots, q-2$, with length $J_{Q^*_m}$.

   Assume by way of contradiction that upon termination, $d_{\mathsf{T}} > J_{Q^*}$. Then, $d_{\mathsf{T}} > J_{Q^*_m}, m = 2, \ldots, q-2$ throughout the algorithm (as d_{T} only decreases in the algorithm and every arc length is non-negative).

   This implies that i_{q-2} does not enter OPEN with $d_{i_{q-2}} = J_{Q^*_{q-2}}$ since if it did, then the next time i_{q-2} is removed from OPEN, d_{T} would be updated to J_{Q^*}.

   Similarly, i_{q-3} will not enter OPEN with $d_{i_{q-3}} = J_{Q^*_{q-3}}$.

   Continuing this way, i_1 will not enter OPEN with $d_{i_1} = J_{Q^*_1} = c_{\mathsf{S},i_1}$. But this happens at the first iteration of the algorithm, which is a contradiction. Hence, $d_{\mathsf{T}} = J_{Q^*}$ upon completion of the algorithm.

□

Example 1: Deterministic Scheduling Problem (revisited)
Consider a deterministic scheduling problem, where 4 operations A, B, C, D are used to produce a certain product. Operation A must occur before B, and C before D. Between each two operations there is a transition cost. Fig. 8.2 illustrates the transition dynamics and the transition costs.

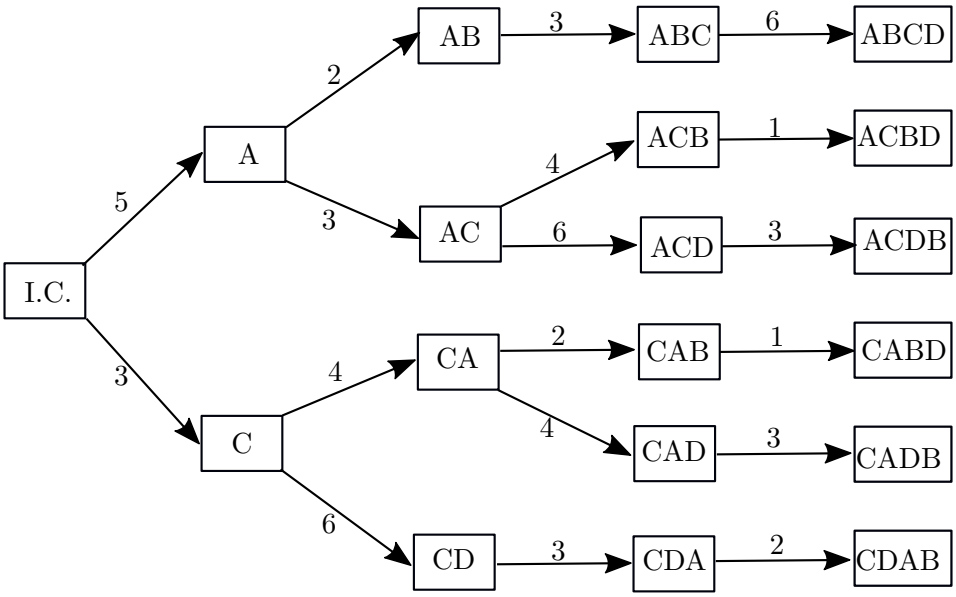


Figure 8.2: State transition diagram for a scheduling problem. The number on each arc denotes the transition cost between two operations. The state is the cumulative operations performed and their order.

We can map the deterministic finite-state system depicted in Fig. 8.2 to a shortest path problem on which we can apply the label correcting algorithm. Before we do so, we can simplify the state transition diagram of Fig. 8.2 to that of Fig. 8.3 in order to reduce the number of nodes.

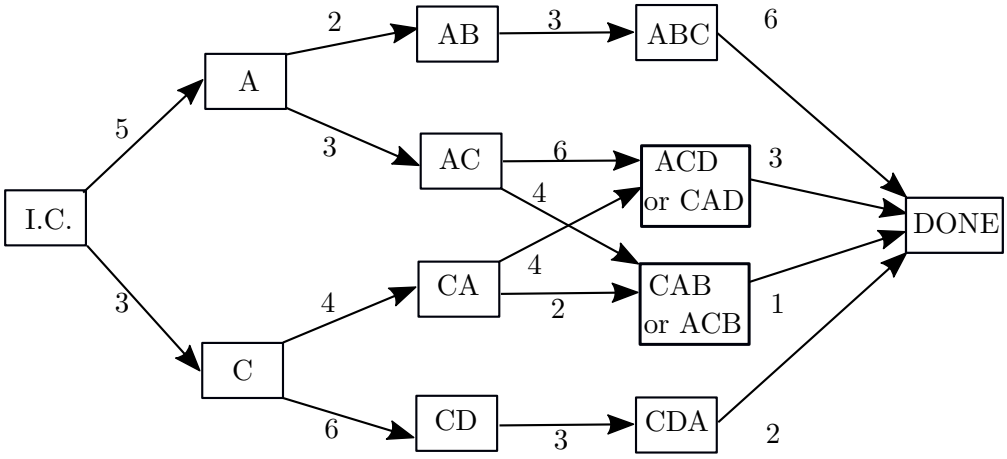


Figure 8.3: Simplified state transition diagram. The number on each arc denotes the transition cost between two operations. For this deterministic finite state system, $N = 4$ with $\mathcal{S}_N = \{\text{DONE}\}$, where DONE represents the completion of the part. Furthermore, $\mathcal{S}_0 = \{\text{I.C.}\}$, $\mathcal{S}_1 = \{\text{A}, \text{C}\}$, $\mathcal{S}_2 = \{\text{AB}, \text{AC}, \text{CA}, \text{CD}\}$, and $\mathcal{S}_3 = \{\text{ABC}, \text{ACD or CAD}, \text{CAB or ACB}, \text{CDA}\}$.

Notice that Fig. 8.3 is already in the form of a SP problem with a clear termination node. See Fig. 8.4, where the states have been enumerated for brevity.

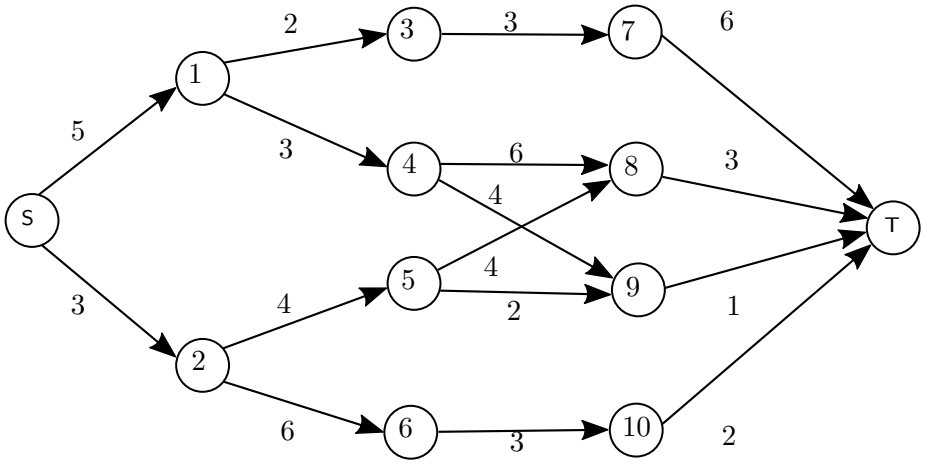


Figure 8.4: Deterministic scheduling problem with enumerated nodes.

We can now apply the LCA on the SP problem depicted in Fig. 8.4:

Iteration	Remove	OPEN	d_{S}	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9	d_{10}	d_{T}
0	–	S	0	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞
1	S	1,2	0	5	3	∞	∞	∞	∞	∞	∞	∞	∞	∞
2	2	1,5,6	0	5	3	∞	∞	7	9	∞	∞	∞	∞	∞
3	6	1,5,10	0	5	3	∞	∞	7	9	∞	∞	∞	12	∞
4	10	1,5	0	5	3	∞	∞	7	9	∞	∞	∞	12	14
5	5	1,8,9	0	5	3	∞	∞	7	9	∞	11	9	12	14
6	9	1,8	0	5	3	∞	∞	7	9	∞	11	9	12	10
7	8	1	0	5	3	∞	∞	7	9	∞	11	9	12	10
8	1	3,4	0	5	3	7	8	7	9	∞	11	9	12	10
9	4	3	0	5	3	7	8	7	9	∞	11	9	12	10
10	3	–	0	5	3	7	8	7	9	∞	11	9	12	10

Keeping track of the parent nodes when a child node is added to open, it can be determined that the optimal path is $(\mathsf{S}, 2, 5, 9, \mathsf{T})$, which corresponds to the state evolution of (C,CA,CAB,CABD), and has a total cost of 10.

△

8.1.1 How to remove nodes from OPEN?

There are various ways to remove nodes from OPEN:

- **Depth-First Search:** This method uses the "last in, first out" strategy, that is, a node is always removed from the top of the OPEN bin and each node entering OPEN is placed at the top. It is often implemented as a stack. This is what we used in Example 1.

- **Breadth-First Search:** This method uses the "first in, first out" strategy; that is, a node is always removed from the top of the OPEN bin and each node entering OPEN is placed at the bottom. It is often implemented as a queue.

- **Best-First Search (Dijkstra's Algorithm):** At each iteration, the node that is removed from OPEN is the node i^* where

  $$d_{i^*} = \min_{i \in \text{OPEN}} d_i,$$

  that is, remove the node that currently has the best label.

8.1.2 A*-algorithm

This algorithm is a modification to the LCA. In particular, in Step 2, we strengthen the requirement of a node j being admitted to OPEN from

$$d_i + c_{i,j} < d_{\mathsf{T}}$$

to

$$d_i + c_{i,j} + h_j < d_{\mathsf{T}}$$

where h_j is some positive lower bound on the cost to get from node j to T. This lower bound can be constructed depending on special knowledge about the problem. This more stringent criterion can reduce the number of iterations required by the LCA.

9. Deterministic Continuous Time Optimal Control and the Hamilton-Jacobi-Bellman Equation

In the next three lectures we will look at continuous-time deterministic optimal control. We begin by deriving the continuous-time analog of the DPA, known as the Hamilton-Jacobi-Bellman equation.

Dynamics

Consider the continuous-time system

$$\dot{x}(t) = f(x(t), u(t)), \qquad 0 \leq t \leq T \tag{9.1}$$

where

- time $t \in \mathbb{R}_{\geq 0}$ and T is the terminal time;
- state $x(t) \in \mathcal{S} := \mathbb{R}^n, \ \forall t \in [0, T]$;
- control $u(t) \in \mathcal{U} \subset \mathbb{R}^m, \ \forall t \in [0, T]$. $\mathcal{U}$ is the control constraint set;
- $f(\cdot, \cdot)$: function capturing system evolution.

Feedback control law

Let $\mu(\cdot, \cdot)$ be an *admissible control law* that maps state $\mathrm{x} \in \mathcal{S}$ at time t to control input $u(t)$:

$$u(t) = \mu(t, \mathrm{x}), \quad u(t) \in \mathcal{U}, \ \forall t \in [0, T], \ \forall \mathrm{x} \in \mathcal{S} \tag{9.2}$$

Furthermore, let Π denote the set of all admissible control laws.

Cost

We consider the following scalar-valued cost function:

$$h(x(T)) + \int_0^T g(x(\tau), u(\tau)) \mathrm{d}\tau \tag{9.3}$$

For an initial time t and state $\mathrm{x} \in \mathcal{S}$, the closed loop cost associated with feedback control law $\mu(\cdot, \cdot) \in \Pi$ is

$$J_\mu(t, \mathrm{x}) := h(x(T)) + \int_t^T g(x(\tau), u(\tau)) \mathrm{d}\tau \quad \text{subject to (10.1), (9.2).} \tag{9.4}$$

Objective

Construct an optimal feedback control law $\mu^* \in \Pi$ such that

$$J_{\mu^*}(0, \mathrm{x}) \leq J_\mu(0, \mathrm{x}), \quad \forall \mu \in \Pi, \forall \mathrm{x} \in \mathcal{S}.$$

The associated closed loop cost $J^*(t, \mathrm{x}) := J_{\mu^*}(t, \mathrm{x})$ is called the *cost-to-go at state* x *and time* t, and $J^*(\cdot, \cdot)$ is the *cost-to-go function* or *value function.*

We further require the following assumption:

Assumption 9.1. *For any admissible control law μ, initial time $t \in [0, T]$ and initial condition $x(t) \in \mathcal{S}$, there exists a unique state trajectory $x(\tau)$ that satisfies*

$$\dot{x}(\tau) = f(x(\tau), u(\tau)), \qquad t \leq \tau \leq T.$$

Assumption 9.1 is required for the problem to be well defined. Ensuring that it is satisfied for a particular problem requires tools from the theory of differential equations, and is beyond the scope of this class.

Example 1: Existence
$\dot{x}(t) = x(t)^2, \quad x(0) = 1$
Solution:

$$x(t) = \frac{1}{1-t}$$
$$\Rightarrow \text{ finite escape time: } x(t) \to \infty \text{ as } t \to 1.$$
$$\Rightarrow \text{ solution does not exist for } T \geq 1.$$

△

Example 2: Uniqueness
$\dot{x}(t) = x(t)^{\frac{1}{3}}, \quad x(0) = 0$
Solution:

$$x(t) = 0 \ \ \forall t$$
$$\text{or } x(t) = \begin{cases} 0 & \text{for } 0 \leq t \leq \tau \\ \left(\frac{2}{3}(t-\tau)\right)^{3/2} & \text{for } t > \tau \end{cases}$$
$$\Rightarrow \text{ infinite number of solutions}$$

△

9.1 The Hamilton-Jacobi-Bellman (HJB) Equation

We now derive a partial differential equation which is satisfied by the cost-to-go function under certain assumptions that we will see later. This equation is the continuous-time analog of the DPA, and will be motivated by applying DPA to a discrete-time approximation of the continuous-time optimal control problem. This is not a rigorous derivation but it does capture the main ideas.

Let us first divide the time horizon $[0, T]$ into N pieces, and define $\delta := \frac{T}{N}$. Furthermore, define $x_k := x(k\delta)$, $u_k := u(k\delta)$ for $k = 0, 1, ..., N$, and approximate the differential equation $\dot{x}(k\delta) = f(x(k\delta), u(k\delta))$ by

$$\frac{x_{k+1} - x_k}{\delta} = f(x_k, u_k), \quad k = 0, 1, ..., N$$

which leads to

$$x_{k+1} = x_k + f(x_k, u_k)\delta, \quad k = 0, 1, ..., N. \tag{9.5}$$

Similarly, cost function (10.2) is approximated by

$$h(x_N) + \sum_{k=0}^{N-1} g(x_k, u_k)\delta.$$

The state space and the control space remain unchanged.
Let $J_k(\mathrm{x})$ be the cost-to-go at stage k and state x for the auxiliary problem and apply the DPA:

$$J_N(\mathrm{x}) = h(\mathrm{x}), \quad \forall \mathrm{x} \in \mathcal{S}$$

$$J_k(\mathrm{x}) = \min_{\mathrm{u} \in \mathcal{U}} \left(\underbrace{g(\mathrm{x}, \mathrm{u})\delta}_{\text{stage cost}} + \underbrace{J_{k+1}(\mathrm{x} + f(\mathrm{x}, \mathrm{u})\delta)}_{\text{cost-to-go at stage } k+1} \right), \quad \forall \mathrm{x} \in \mathcal{S},\ k = N-1, ..., 0. \tag{9.6}$$

We assume $J_k(\mathrm{x}) = J^*(k\delta, \mathrm{x}) + o(\delta)$ for all $\mathrm{x} \in \mathcal{S}$, $k = 0, \ldots, N-1$, where $o(\delta)$ represents second order terms and satisfies $\lim_{\delta \to 0}(o(\delta)/\delta) = 0$. In particular, we assume that the cost-to-go calculated via the DPA converges to the actual cost-to-go of the continuous time problem, as delta approaches zero. Thus for any $\mathrm{u} \in \mathcal{U}$,

$$J_{k+1}(\mathrm{x} + f(\mathrm{x}, \mathrm{u})\delta) = J^*((k+1)\delta,\ \mathrm{x} + f(\mathrm{x}, \mathrm{u})\delta) + o(\delta).$$

Assuming the cost-to-go function $J^*(\cdot, \cdot)$ of the continuous-time formulation is differentiable with respect to t and x, we can express it using a first order Taylor series around $(k\delta, \mathrm{x})$ and evaluate the function at $((k+1)\delta, \mathrm{x} + f(\mathrm{x}, \mathrm{u})\delta)$:

$$J^*((k+1)\delta, \mathrm{x} + f(\mathrm{x}, \mathrm{u})\delta) = J^*(k\delta, \mathrm{x}) + \frac{\partial J^*(k\delta, \mathrm{x})}{\partial t}\delta + \frac{\partial J^*(k\delta, \mathrm{x})}{\partial \mathrm{x}} f(\mathrm{x}, \mathrm{u})\delta + o(\delta)$$

where $\partial J^*(k\delta, \mathrm{x}) / \partial \mathrm{x}$ denotes the Jacobian row vector. Thus with $t = k\delta$, (9.6) becomes

$$\begin{aligned}
J^*(t, \mathrm{x}) &= \min_{\mathrm{u} \in \mathcal{U}} \left(g(\mathrm{x}, \mathrm{u})\delta + J^*(t, \mathrm{x}) + \frac{\partial J^*(t, \mathrm{x})}{\partial t}\delta + \frac{\partial J^*(t, \mathrm{x})}{\partial \mathrm{x}} f(\mathrm{x}, \mathrm{u})\delta + o(\delta) \right) \\
\Leftrightarrow 0 &= \min_{\mathrm{u} \in \mathcal{U}} \left(g(\mathrm{x}, \mathrm{u})\delta + \frac{\partial J^*(t, \mathrm{x})}{\partial t}\delta + \frac{\partial J^*(t, \mathrm{x})}{\partial \mathrm{x}} f(\mathrm{x}, \mathrm{u})\delta + o(\delta) \right) \\
0 &= \min_{\mathrm{u} \in \mathcal{U}} \left(g(\mathrm{x}, \mathrm{u}) + \frac{\partial J^*(t, \mathrm{x})}{\partial t} + \frac{\partial J^*(t, \mathrm{x})}{\partial \mathrm{x}} f(\mathrm{x}, \mathrm{u}) + \frac{o(\delta)}{\delta} \right).
\end{aligned}$$

Taking the limit of the above as $N \to \infty$, or equivalently as $\delta \to 0$, and assuming we can swap the limit and minimization operations, results in:

$$0 = \min_{\mathrm{u} \in \mathcal{U}} \left[g(\mathrm{x}, \mathrm{u}) + \frac{\partial J^*(t, \mathrm{x})}{\partial t} + \frac{\partial J^*(t, \mathrm{x})}{\partial \mathrm{x}} f(\mathrm{x}, \mathrm{u}) \right] \quad \forall t \in [0, T],\ \forall \mathrm{x} \in \mathcal{S}$$

subject to the terminal condition $J^*(T, \mathrm{x}) = h(\mathrm{x})$ for all $\mathrm{x} \in \mathcal{S}$.
The above equation is called the Hamilton-Jacobi-Bellman (HJB) Equation. Note that in the above informal derivation we rely on $J^*(\cdot, \cdot)$ being smooth in x and t.

Example 3:
In this example, we will hand-craft an optimal policy and show that the resulting cost-to-go satisfies the HJB. Consider

$$\dot{x}(t) = u(t), \quad |u(t)| \leq 1, \ 0 \leq t \leq 1$$

with cost function

$$\frac{1}{2}x(1)^2,$$

that is, $h(x(1)) = \frac{1}{2}x(1)^2$ and $g(\mathrm{x}, \mathrm{u}) = 0$ for all $\mathrm{x} \in \mathcal{S}$ and $\mathrm{u} \in \mathcal{U}$.

Since we only care about the square of the terminal state, we can construct a candidate optimal policy that drives the state towards 0 as quickly as possible, and maintaining it at 0 once it is at 0. The corresponding control policy is

$$\begin{aligned} u(t) = \mu(t, \mathrm{x}) &= \begin{cases} -1 & \text{if } \mathrm{x} > 0 \\ 0 & \text{if } \mathrm{x} = 0 \\ 1 & \text{if } \mathrm{x} < 0 \end{cases} \\ &= -\operatorname{sgn}(\mathrm{x}). \end{aligned}$$

For a given initial time t and initial state x, the cost $J_\mu(t, \mathrm{x})$ associated with the above policy is

$$J_\mu(t, \mathrm{x}) = \frac{1}{2}\left(\max\{0, |\mathrm{x}| - (1-t)\}\right)^2.$$

We will verify that this cost function satisfies the HJB and is therefore indeed the cost-to-go function.
For fixed t:

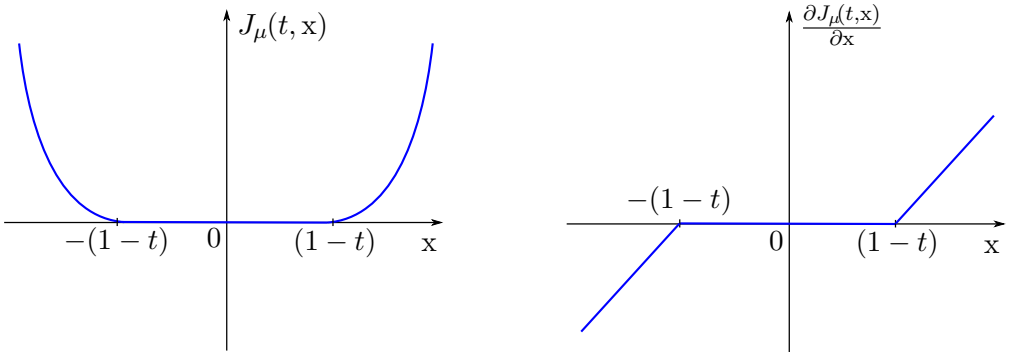


Figure 9.1: Cost function with fixed t and its partial derivative with respect to x

$$\frac{\partial J_\mu(t, \mathrm{x})}{\partial \mathrm{x}} = \operatorname{sgn}(\mathrm{x}) \max\{0, |\mathrm{x}| - (1-t)\}$$

For fixed x:

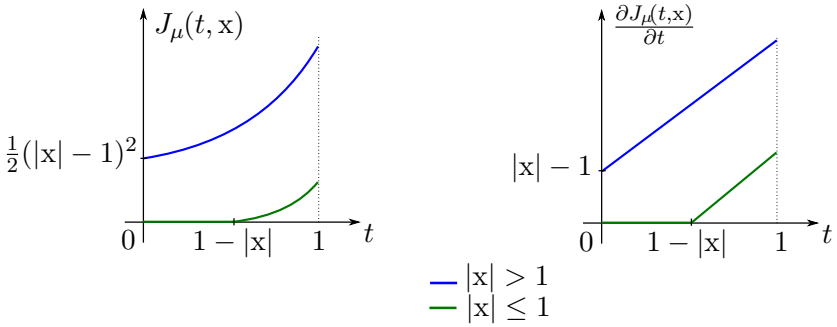


Figure 9.2: Cost function with fixed x and its partial derivative with respect to t

$$\frac{\partial J_\mu(t,x)}{\partial t} = \max\{0, |\mathrm{x}| - (1-t)\}$$

Note that $J_\mu(1,\mathrm{x}) = \frac{1}{2}\mathrm{x}^2$, the boundary condition is satisfied, and furthermore

$$\min_{|\mathrm{u}|\leq 1}\left(\frac{\partial J_\mu(t,\mathrm{x})}{\partial t} + \frac{\partial J_\mu(t,\mathrm{x})}{\partial \mathrm{x}}\mathrm{u}\right) = \min_{|\mathrm{u}|\leq 1}\left((1+\mathrm{sgn}(\mathrm{x})\mathrm{u})\left(\max\{0,|\mathrm{x}|-(1-t)\}\right)\right) = 0$$

where the minimum is attained by $\mathrm{u} = -\mathrm{sgn}(\mathrm{x})$.
Thus $J_\mu(t,\mathrm{x}) = J^*(t,\mathrm{x})$, and $\mu^*(t,\mathrm{x}) = -\mathrm{sgn}(\mathrm{x})$ is an optimal policy.

△

Note that this was a simple example. In general solving the HJB is nontrivial.

Example 4:
We will now look at a problem for which the cost-to-go may not be smooth, and thus we cannot apply the HJB. Consider

$$\dot{x}(t) = x(t)u(t), \quad |u(t)| \leq 1,\ 0 \leq t \leq 1$$

with cost

$$x(1),$$

that is, $h(x(1)) = x(1)$ and $g(\mathrm{x},\mathrm{u}) = 0$ for all $\mathrm{x} \in \mathcal{S}$ and $\mathrm{u} \in \mathcal{U}$.

One can show that an optimal policy is the following:

$$\mu(t,\mathrm{x}) = \begin{cases} -1 & \text{if } \mathrm{x} > 0 \\ 0 & \text{if } \mathrm{x} = 0 \\ 1 & \text{if } \mathrm{x} < 0 \end{cases}$$

and its associated cost-to-go function is:

$$J_\mu(t,\mathrm{x}) = \begin{cases} e^{-1+t}\mathrm{x} & \mathrm{x} > 0 \\ e^{1-t}\mathrm{x} & \mathrm{x} < 0 \\ 0 & \mathrm{x} = 0 \end{cases}$$

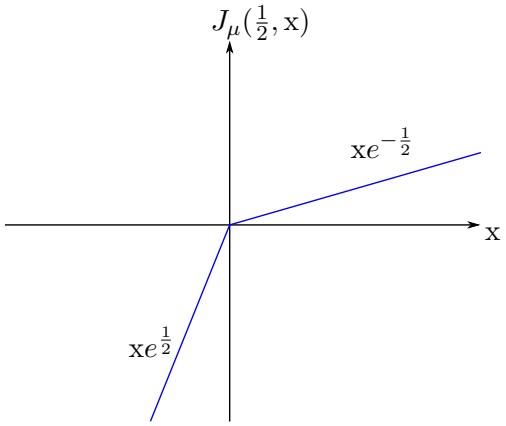


Figure 9.3: Cost function with fixed $t = \frac{1}{2}$.

However, it is clear that the associated cost-to-go function is not differentiable with respect to x at $\mathrm{x} = 0$ (see Figure 9.3), thereby not satisfying the HJB. This illustrates the fact that the HJB is in general not a necessary condition for optimality, but it is sufficient as we will see next[1].

△

9.1.1 Sufficiency of the HJB

Now we prove that HJB is a sufficient condition for the optimal policy and cost-to-go function.

Theorem 9.1. *Suppose $V(t, \mathrm{x})$ is a solution to the HJB equation, that is, V is continuously differentiable in t and x, and is such that*

$$\min_{\mathrm{u} \in \mathcal{U}} \left[g(\mathrm{x}, \mathrm{u}) + \frac{\partial V(t, \mathrm{x})}{\partial t} + \frac{\partial V(t, \mathrm{x})}{\partial \mathrm{x}} f(\mathrm{x}, \mathrm{u}) \right] = 0 \quad \forall \mathrm{x} \in \mathcal{S},\ 0 \le t \le T \tag{9.7}$$
$$\text{subject to } V(T, \mathrm{x}) = h(\mathrm{x}) \quad \forall \mathrm{x} \in \mathcal{S}$$

Suppose also that $\mu(t, \mathrm{x})$ attains the minimum in (9.7) for all t and x. Then under Assumption 9.1, $V(t, \mathrm{x})$ is equal to the cost-to-go function, i.e.

$$V(t, \mathrm{x}) = J^*(t, \mathrm{x}), \quad \forall \mathrm{x} \in \mathcal{S},\ 0 \le t \le T$$

Furthermore, the mapping μ is an optimal feedback law.

Proof. For any initial time $t \in [0, T]$ and any initial condition $x(t) = \mathrm{x}$, $\mathrm{x} \in \mathcal{S}$, let $\hat{u}(\tau) \in \mathcal{U}$ for all $\tau \in [t, T]$ be any admissible control trajectory, and let $\hat{x}(\tau)$ be the corresponding state trajectory where $\hat{x}(\tau)$ is the unique solution to the ODE $\dot{x}(\tau) = f(x(\tau), \hat{u}(\tau))$. From (9.7) we have for all $\tau \in [0, T]$,

$$0 \le g(\hat{x}(\tau), \hat{u}(\tau)) + \left. \frac{\partial V(\tau, \mathrm{x})}{\partial \tau} \right|_{\mathrm{x} = \hat{x}(\tau)} + \left. \frac{\partial V(\tau, \mathrm{x})}{\partial \mathrm{x}} \right|_{\mathrm{x} = \hat{x}(\tau)} f(\hat{x}(\tau), \hat{u}(\tau))$$
$$0 \le g(\hat{x}(\tau), \hat{u}(\tau)) + \frac{\mathrm{d}}{\mathrm{d}\tau} \left(V(\tau, \hat{x}(\tau)) \right),$$

[1] One can address this shortcoming by introducing generalized solutions to the HJB partial differential equation, such as viscosity solutions.

where $\mathrm{d}/\mathrm{d}\tau(\cdot)$ denotes the total derivative with respect to τ. Integrating the above inequality over $\tau \in [t, T]$ yields

$$0 \leq \int_t^T g(\hat{x}(\tau), \hat{u}(\tau))\mathrm{d}\tau + V(T, \hat{x}(T)) - V(t, \mathrm{x})$$
$$V(t, \mathrm{x}) \leq h(\hat{x}(T)) + \int_t^T g(\hat{x}(\tau), \hat{u}(\tau))\mathrm{d}\tau$$

The preceding inequalities become equalities for the minimizing $\mu(\tau, x(\tau))$ of (9.7):

$$V(t, \mathrm{x}) = h(x(T)) + \int_t^T g(x(\tau), \mu(\tau, x(\tau)))\mathrm{d}\tau$$

where $x(\tau)$ is the unique solution to the ODE $\dot{x}(\tau) = f(x(\tau), \mu(\tau, x(\tau))$ with $x(t) = \mathrm{x}$. Thus $V(t, \mathrm{x})$ is the cost-to-go at state x at time t, and $\mu(\tau, x(\tau))$ is an optimal control trajectory. We thus have

$$V(t, \mathrm{x}) = J^*(t, \mathrm{x}), \quad \forall \mathrm{x} \in \mathcal{S}, \forall t \in [0, T]$$

and μ is indeed an optimal feedback law. □

10. Pontryagin's Minimum Principle

The HJB equation provides a lot of information: the optimal cost-to-go and the optimal policy for all time and for all possible states. However, in many cases, we only care about the optimal control trajectory for a specific initial condition. We will see how to exploit the fact that we are asking for much less in order to arrive at simpler conditions for optimality (the Minimum Principle).

10.1 Notation

Let $F(t, \mathrm{x})$ be a continuous, differentiable function. Then,

- The partial derivative of F with respect to its first argument, t, is $\dfrac{\partial F(t, \mathrm{x})}{\partial t}$.
- The partial derivative of F with respect to t when subject to $\mathrm{x} = x(t)$ is

$$\frac{\partial F(t, x(t))}{\partial t} = \left.\frac{\partial F(t, \mathrm{x})}{\partial t}\right|_{\mathrm{x}=x(t)} + \left.\frac{\partial F(t, \mathrm{x})}{\partial \mathrm{x}}\right|_{\mathrm{x}=x(t)} \frac{\partial x(t)}{\partial t}.$$

- The total derivative of F with respect to t when subject to $\mathrm{x} = x(t)$ is

$$\frac{\mathrm{d}F(t, x(t))}{\mathrm{d}t} = \left.\frac{\partial F(t, \mathrm{x})}{\partial t}\right|_{\mathrm{x}=x(t)} + \left.\frac{\partial F(t, \mathrm{x})}{\partial \mathrm{x}}\right|_{\mathrm{x}=x(t)} \frac{\mathrm{d}x(t)}{\mathrm{d}t}$$

  Since $x(t)$ is only a function of t, then

$$\frac{\mathrm{d}F(t, x(t))}{\mathrm{d}t} = \frac{\partial F(t, x(t))}{\partial t}.$$

- We sometimes use the short-hand notation $\left.\dfrac{\partial F(t, \mathrm{x})}{\partial t}\right|_{x(t)} := \left.\dfrac{\partial F(t, \mathrm{x})}{\partial t}\right|_{\mathrm{x}=x(t)}$

Example 1:
Consider $F(t, \mathrm{x}) = t\mathrm{x}$. Then,

$$\frac{\partial F(t, \mathrm{x})}{\partial t} = \mathrm{x}$$
$$\frac{\mathrm{d}F(t, x(t))}{\mathrm{d}t} = x(t) + t\dot{x}(t).$$

△

Lemma 10.1. *Let $F(t, \mathrm{x}, \mathrm{u})$ be a continuously differentiable function of $t \in \mathbb{R}$, $\mathrm{x} \in \mathbb{R}^n$, $\mathrm{u} \in \mathbb{R}^m$ and let $\mathcal{U} \subseteq \mathbb{R}^m$ be a convex set. Furthermore, assume $\mu^*(t, \mathrm{x}) := \arg\min_{\mathrm{u}\in\mathcal{U}} F(t, \mathrm{x}, \mathrm{u})$ exists and is continuously differentiable. Then for all t and x,*

$$\frac{\partial\left(\min\limits_{\mathrm{u}\in\mathcal{U}} F(t,\mathrm{x},\mathrm{u})\right)}{\partial t} = \left.\frac{\partial F(t,\mathrm{x},\mathrm{u})}{\partial t}\right|_{\mathrm{u}=\mu^*(t,\mathrm{x})}$$
$$\frac{\partial\left(\min\limits_{\mathrm{u}\in\mathcal{U}} F(t,\mathrm{x},\mathrm{u})\right)}{\partial \mathrm{x}} = \left.\frac{\partial F(t,\mathrm{x},\mathrm{u})}{\partial \mathrm{x}}\right|_{\mathrm{u}=\mu^*(t,\mathrm{x})}.$$

Proof. We prove this for when $\mathcal{U} = \mathbb{R}^m$. Let $G(t, \mathrm{x}) := \min\limits_{\mathrm{u}\in\mathcal{U}} F(t, \mathrm{x}, \mathrm{u}) = F(t, \mathrm{x}, \mu^*(t, \mathrm{x}))$. Then,

$$\begin{aligned}\frac{\partial G(t,\mathrm{x})}{\partial t} &= \frac{\partial F(t,\mathrm{x},\mu^*(t,\mathrm{x}))}{\partial t}\\ &= \left.\frac{\partial F(t,\mathrm{x},\mathrm{u})}{\partial t}\right|_{\mathrm{u}=\mu^*(t,\mathrm{x})} + \underbrace{\left.\frac{\partial F(t,\mathrm{x},\mathrm{u})}{\partial \mathrm{u}}\right|_{\mathrm{u}=\mu^*(t,\mathrm{x})}}_{=0 \text{ since } \mu^*(t,\mathrm{x}) \text{ minimizes } F(t,\mathrm{x},\mathrm{u})} \frac{\partial \mu^*(t,\mathrm{x})}{\partial t}\end{aligned}$$

Similarly, this can be shown for the partial derivative with respect to x. □

Example 2:
Let $F(t, \mathrm{x}, \mathrm{u}) := (1+t)\mathrm{u}^2 + \mathrm{u}\mathrm{x} + 1, \quad t \geq 0$. Then,

$$\begin{aligned}&\min_{\mathrm{u}\in\mathbb{R}} F(t,\mathrm{x},\mathrm{u}) : 2(1+t)\mathrm{u} + \mathrm{x} = 0, \quad \mathrm{u} = -\frac{\mathrm{x}}{2(1+t)}\\ &\therefore \mu^*(t,\mathrm{x}) = -\frac{\mathrm{x}}{2(1+t)}\\ &\therefore \min_{\mathrm{u}\in\mathbb{R}} F(t,\mathrm{x},\mathrm{u}) = \frac{(1+t)\mathrm{x}^2}{4(1+t)^2} - \frac{\mathrm{x}^2}{2(1+t)} + 1 = -\frac{\mathrm{x}^2}{4(1+t)} + 1\end{aligned}$$

1.
$$\begin{aligned}\frac{\partial\left(\min\limits_{\mathrm{u}\in\mathbb{R}} F(t,\mathrm{x},\mathrm{u})\right)}{\partial t} &= \frac{\partial F(t,\mathrm{x},\mu^*(t,\mathrm{x}))}{\partial t}\\ &= \frac{\mathrm{x}^2}{4(1+t)^2}\end{aligned}$$

And by Lemma 10.1,

$$\begin{aligned}\frac{\partial\left(\min\limits_{\mathrm{u}\in\mathbb{R}} F(t,\mathrm{x},\mathrm{u})\right)}{\partial t} &= \left.\frac{\partial F(t,\mathrm{x},\mathrm{u})}{\partial t}\right|_{\mathrm{u}=\mu^*(t,\mathrm{x})}\\ &= \mathrm{u}^2\big|_{\mathrm{u}=-\frac{\mathrm{x}}{2(1+t)}}\\ &= \frac{\mathrm{x}^2}{4(1+t)^2}\end{aligned}$$

2.

$$\begin{aligned}\frac{\partial\left(\min_{\mathrm{u}\in\mathbb{R}} F(t,\mathrm{x},\mathrm{u})\right)}{\partial \mathrm{x}} &= \frac{\partial F(t,\mathrm{x},\mu^*(t,\mathrm{x}))}{\partial \mathrm{x}} \\ &= -\frac{\mathrm{x}}{2(1+t)}\end{aligned}$$

And by Lemma 10.1,

$$\begin{aligned}\frac{\partial\left(\min_{\mathrm{u}\in\mathbb{R}} F(t,\mathrm{x},\mathrm{u})\right)}{\partial \mathrm{x}} &= \left.\frac{\partial F(t,\mathrm{x},\mathrm{u})}{\partial \mathrm{x}}\right|_{\mathrm{u}=\mu^*(t,\mathrm{x})} \\ &= \mathrm{u}|_{\mathrm{u}=-\frac{\mathrm{x}}{2(1+t)}} \\ &= -\frac{\mathrm{x}}{2(1+t)}\end{aligned}$$

△

10.2 The Minimum Principle

Consider the following continuous-time set-up.

Dynamics

$$\dot{x}(t) = f(x(t), u(t)), \qquad 0 \le t \le T \tag{10.1}$$

where

- time $t \in \mathbb{R}_{\ge 0}$ and T is the terminal time;
- state $x(t) \in \mathcal{S} := \mathbb{R}^n,\ \forall t \in [0,T]$;
- control $u(t) \in \mathcal{U} \subset \mathbb{R}^m,\ \forall t \in [0,T]$. $\mathcal{U}$ is the control constraint set;
- $f(\cdot,\cdot)$: function capturing system evolution that is continuously differentiable in x.

Cost

We consider the following scalar-valued cost function:

$$h(x(T)) + \int_0^T g(x(\tau), u(\tau))\mathrm{d}\tau \tag{10.2}$$

where g and h are both continuously differentiable in x.

Objective

Given an initial condition $x(0) = \mathrm{x} \in \mathcal{S}$, construct an optimal control trajectory $u(t)$ such that (10.2) subject to (10.1) is minimized.

One could potentially solve the above problem using the HJB which gives an optimal policy $\mu^*(t, \mathrm{x})$. The optimal input trajectory can then be inferred from the policy for a given initial condition and the solution to $\dot{x}(t) = f(x(t), \mu^*(t, x(t))$. However, as we have seen, solving the HJB is very difficult in general. The following theorem can be used instead which gives necessary conditions on the optimal trajectory (henceforth the star superscript $\cdot^*$ will be dropped).

Theorem 10.1. *For a given initial condition $x(0) = \mathrm{x} \in \mathcal{S}$, let $u(t)$ be an optimal control trajectory with associated state trajectory $x(t)$ for the system* (10.1)*. Then there exists a trajectory $p(t)$ such that:*

$$\begin{aligned} \dot{p}(t) &= -\left.\frac{\partial H(\mathrm{x},\mathrm{u},\mathrm{p})}{\partial \mathrm{x}}\right|^{\top}_{x(t),u(t),p(t)}, \quad p(T) = \left.\frac{\partial h(\mathrm{x})}{\partial \mathrm{x}}\right|^{\top}_{x(T)} \\ u(t) &= \arg\min_{\mathrm{u}\in\mathcal{U}} H(x(t), \mathrm{u}, p(t)) \\ H(x(t), u(t), p(t)) &= \ \textit{constant} \quad \forall t \in [0, T] \end{aligned}$$

where $H(\mathrm{x},\mathrm{u},\mathrm{p}) := g(\mathrm{x},\mathrm{u}) + \mathrm{p}^{\top} f(\mathrm{x},\mathrm{u})$.

The function $H(\cdot,\cdot,\cdot)$ in the above is referred to as the Hamiltonian function, which comes from Hamilton's Principle of Least Action from mechanics.

Proof. We provide an informal proof which assumes that the cost-to-go $J(t,\mathrm{x})$ is continuously differentiable, the optimal policy $\mu(\cdot,\cdot)$ is continuously differentiable, and $\mathcal{U}$ is convex in order to make use of Lemma 10.1. However, these assumptions are actually not needed in a more formal proof.
With continuously differentiable cost-to-go, the HJB is also a necessary condition for optimality:

$$0 = \min_{\mathrm{u}\in\mathcal{U}} \underbrace{\left(g(\mathrm{x},\mathrm{u}) + \frac{\partial J(t,\mathrm{x})}{\partial t} + \frac{\partial J(t,\mathrm{x})}{\partial \mathrm{x}} f(\mathrm{x},\mathrm{u})\right)}_{=:F(t,\mathrm{x},\mathrm{u})}, \quad \forall t \in [0,T],\ \forall \mathrm{x} \in \mathcal{S} \tag{10.3}$$

$$= \min_{\mathrm{u}\in\mathcal{U}} F(t,\mathrm{x},\mathrm{u}) \tag{10.4}$$

$$J(T,\mathrm{x}) = h(\mathrm{x}) \quad \forall \mathrm{x} \in \mathcal{S} \tag{10.5}$$

with $\mathrm{u} = \mu(t,\mathrm{x})$ the corresponding optimal control strategy.

Now take the partial derivatives of (10.4) with respect to t and x; by Lemma 10.1

$$\begin{aligned} 0 &= \frac{\partial\left(\min_{\mathrm{u}\in\mathcal{U}} F(t,\mathrm{x},\mathrm{u})\right)}{\partial t} = \left.\frac{\partial F(t,\mathrm{x},\mathrm{u})}{\partial t}\right|_{\mu(t,\mathrm{x})} \\ &= \frac{\partial^2 J(t,\mathrm{x})}{\partial t^2} + \frac{\partial^2 J(t,\mathrm{x})}{\partial t \partial \mathrm{x}} f(\mathrm{x},\mu(t,\mathrm{x})), \end{aligned} \tag{10.6}$$

and similarly,

$$\begin{aligned} 0 &= \frac{\partial \left(\min_{\mathrm{u}\in\mathcal{U}} F(t,\mathrm{x},\mathrm{u}) \right)}{\partial \mathrm{x}} = \left.\frac{\partial F(t,\mathrm{x},\mathrm{u})}{\partial \mathrm{x}}\right|_{\mu(t,\mathrm{x})} \\ &= \left.\frac{\partial g(\mathrm{x},\mathrm{u})}{\partial \mathrm{x}}\right|_{\mu(t,\mathrm{x})} + \frac{\partial^2 J(t,\mathrm{x})}{\partial \mathrm{x}\partial t} + f(\mathrm{x},\mu(t,\mathrm{x}))^\top \frac{\partial^2 J(t,\mathrm{x})}{\partial \mathrm{x}^2} + \frac{\partial J(t,\mathrm{x})}{\partial \mathrm{x}} \left.\frac{\partial f(\mathrm{x},\mathrm{u})}{\partial \mathrm{x}}\right|_{\mu(t,\mathrm{x})}. \end{aligned} \tag{10.7}$$

Now consider the specific optimal trajectory $u(t) := \mu(t, x(t))$ where

$$\begin{aligned} \dot{x}(t) &= f(x(t),u(t)) = \left.\frac{\partial H(\mathrm{x},\mathrm{u},\mathrm{p}))}{\partial \mathrm{p}}\right|^\top_{x(t),u(t)} \\ x(0) &= \mathrm{x}. \end{aligned}$$

Along this optimal trajectory, (10.6) becomes

$$\begin{aligned} 0 &= \left.\frac{\partial^2 J(t,\mathrm{x})}{\partial t^2}\right|_{x(t)} + \left.\frac{\partial^2 J(t,\mathrm{x})}{\partial t \partial \mathrm{x}}\right|_{x(t)} \dot{x}(t) \\ &= \frac{\mathrm{d}}{\mathrm{d}t} \left(\underbrace{\left.\frac{\partial J(t,\mathrm{x})}{\partial t}\right|_{x(t)}}_{=:r(t)} \right), \end{aligned} \tag{10.8}$$

and (10.7) becomes

$$\begin{aligned} 0 &= \left.\frac{\partial g(\mathrm{x},\mathrm{u})}{\partial \mathrm{x}}\right|_{x(t),u(t)} + \left.\frac{\partial^2 J(t,\mathrm{x})}{\partial \mathrm{x}\partial t}\right|_{x(t)} + \dot{x}(t)^\top \left.\frac{\partial^2 J(t,\mathrm{x})}{\partial \mathrm{x}^2}\right|_{x(t)} + \left.\frac{\partial J(t,\mathrm{x})}{\partial \mathrm{x}}\right|_{x(t)} \left.\frac{\partial f(\mathrm{x},\mathrm{u})}{\partial \mathrm{x}}\right|_{x(t),u(t)} \\ &= \left.\frac{\partial g(\mathrm{x},\mathrm{u})}{\partial \mathrm{x}}\right|_{x(t),u(t)} + \frac{\mathrm{d}}{\mathrm{d}t} \left(\underbrace{\left.\frac{\partial J(t,\mathrm{x})}{\partial \mathrm{x}}\right|_{x(t)}}_{=:p(t)^\top} \right) + \left.\frac{\partial J(t,\mathrm{x})}{\partial \mathrm{x}}\right|_{x(t)} \left.\frac{\partial f(\mathrm{x},\mathrm{u})}{\partial \mathrm{x}}\right|_{x(t),u(t)}. \end{aligned} \tag{10.9}$$

With $r(t) := \left.\frac{\partial J(t,\mathrm{x})}{\partial t}\right|_{x(t)}$, (10.8) becomes

$$\dot{r}(t) = 0 \qquad \Rightarrow r(t) = \text{ constant } \forall t$$

and with $p(t) := \left.\frac{\partial J(t,\mathrm{x})}{\partial \mathrm{x}}\right|^\top_{x(t)}$, (10.9) becomes

$$\begin{aligned} \dot{p}(t) &= -\left.\frac{\partial g(\mathrm{x},\mathrm{u})}{\partial \mathrm{x}}\right|^\top_{x(t),u(t)} - \left.\frac{\partial f(\mathrm{x},\mathrm{u})}{\partial \mathrm{x}}\right|^\top_{x(t),u(t)} p(t) \\ &= -\left.\frac{\partial H(\mathrm{x},\mathrm{u},\mathrm{p})}{\partial \mathrm{x}}\right|^\top_{x(t),u(t),p(t)} \end{aligned}$$

Taking the partial derivative of the boundary condition (10.5) with respect to x yields

$$\frac{\partial J(T,\mathrm{x})}{\partial \mathrm{x}} = \frac{\partial h(\mathrm{x})}{\partial \mathrm{x}}, \quad \forall \mathrm{x} \in \mathcal{S}$$

and thus

$$p(T) = \left.\frac{\partial J(t,\mathrm{x})}{\partial \mathrm{x}}\right|_{T,x(T)}^{\top} = \left.\frac{\partial h(\mathrm{x})}{\partial \mathrm{x}}\right|_{x(T)}^{\top}.$$

From (10.3), we have

$$\begin{aligned} -\frac{\partial J(t,\mathrm{x})}{\partial t} &= \min_{\mathrm{u}\in\mathcal{U}} \left(g(\mathrm{x},\mathrm{u}) + \frac{\partial J(t,\mathrm{x})}{\partial \mathrm{x}} f(\mathrm{x},\mathrm{u}) \right) \\ &= \min_{\mathrm{u}\in\mathcal{U}} H\left(\mathrm{x},\mathrm{u},\frac{\partial J(t,\mathrm{x})}{\partial \mathrm{x}}^{\top}\right) \end{aligned}$$

which along the optimal trajectory is

$$-r(t) = H(x(t),u(t),p(t))$$

which is constant. Furthermore, note that

$$\begin{aligned} u(t) &= \arg\min_{\mathrm{u}\in\mathcal{U}} F(t,x(t),\mathrm{u}) \\ &= \arg\min_{\mathrm{u}\in\mathcal{U}} \left(g(x(t),\mathrm{u}) + \left.\frac{\partial J(t,\mathrm{x})}{\partial \mathrm{x}}\right|_{x(t)} f(x(t),\mathrm{u}) \right) \\ &= \arg\min_{\mathrm{u}\in\mathcal{U}} \left(g(x(t),\mathrm{u}) + p(t)^{\top} f(x(t),\mathrm{u}) \right) \\ &= \arg\min_{\mathrm{u}\in\mathcal{U}} H(x(t),\mathrm{u},p(t)). \end{aligned}$$

□

Remarks:

- The Minimum Principle requires solving an ODE with split boundary conditions. It is not trivial to solve, but easier than solving a PDE in the HJB.
- The Minimum Principle provides necessary conditions for optimality. If a control trajectory satisfies these conditions, it is not necessarily optimal. Further analysis is needed to guarantee optimality. One method that often works is to prove that an optimal control trajectory exists, and to verify that there is only one control trajectory satisfying the conditions of the Minimum Principle.

Example 3: Resource allocation problem: preparing a Martian base
Consider a problem where a fleet of reconfigurable, general purpose robots is sent to Mars at time 0 to help build a Martian base. They can be used for two things: 1) They can replicate themselves; 2) They can make habitats for human-beings. The number of the robots at time t is denoted by $x(t)$, and the number of habitats by $z(t)$. We want to maximize the size of the Martian base at the terminal time T.
The dynamics are given by

$$\dot{x}(t) = u(t)x(t), \quad x(0) = \mathrm{x} > 0,$$

$$\dot{z}(t) = (1 - u(t))x(t), \quad z(0) = 0,$$
$$0 \leq u(t) \leq 1.$$

where $u(t)$ denotes the fraction of $x(t)$ used to reproduce themselves.
Objective: find a control trajectory $u(t)$ that maximizes $z(T)$.

Solution

Note that the cost function can be written as a function of $x(t)$ and $u(t)$:

$$z(T) = \int_0^T (1 - u(t))x(t)\mathrm{d}t \tag{10.10}$$

and $z(t)$ does not enter into the dynamics of $x(t)$, we can therefore consider $x(t)$ as the only state and $u(t)$ as the control input. The stage cost is then $g(\mathrm{x}, \mathrm{u}) = (1 - \mathrm{u})\mathrm{x}$, the terminal cost is $h(\mathrm{x}) = 0$, and the dynamics function is $f(\mathrm{x}, \mathrm{u}) = \mathrm{ux}$. Thus the Hamiltonian is

$$H(\mathrm{x}, \mathrm{u}, \mathrm{p}) = (1 - \mathrm{u})\mathrm{x} + \mathrm{pux}.$$

Applying Theorem 10.1,

$$\dot{p}(t) = -\left.\frac{\partial H(\mathrm{x}, \mathrm{u}, \mathrm{p})}{\partial \mathrm{x}}\right|^{\top}_{x(t),u(t),p(t)} = -1 + u(t) - p(t)u(t) \tag{10.11}$$
$$p(T) = \left.\frac{\partial h(\mathrm{x})}{\partial \mathrm{x}}\right|^{\top}_{x(T)} = 0$$
$$\dot{x}(t) = x(t)u(t), \quad x(0) = \mathrm{x}$$
$$u(t) = \arg\max_{0 \leq \mathrm{u} \leq 1} H(x(t), \mathrm{u}, p(t)) = \arg\max_{0 \leq \mathrm{u} \leq 1} (x(t) + x(t)(p(t) - 1)\mathrm{u}) \tag{10.12}$$

Since $x(t) > 0$ for $t \in [0, T]$, from (10.12) we can find the following solution[1]

$$u(t) = \begin{cases} 0 & \text{if } p(t) < 1 \\ 1 & \text{if } p(t) \geq 1 \end{cases}$$

We will now work backwards from $t = T$. Since $p(T) = 0$, for t close to T, we have $u(t) = 0$ and therefore (10.11) becomes $\dot{p}(t) = -1$. Therefore at time $t = T - 1$, $p(t) = 1$ and that is when the control input switches to $u(t) = 1$. Thus for $t \leq T - 1$:

$$\dot{p}(t) = -p(t), \quad p(T - 1) = 1$$
$$\Rightarrow p(t) = e^{(T-1)}e^{-t} \tag{10.13}$$

Note that by (10.13) $p(t)$ is bigger than 1 for $t < T - 1$ till time 0, hence we have

$$u(t) = \begin{cases} 1 & \text{if } 0 \leq t \leq T - 1 \\ 0 & \text{if } T - 1 < t \leq T \end{cases}. \tag{10.14}$$

In conclusion, an optimal control trajectory is to use all the robots to replicate themselves from time 0 until $t = T - 1$, and then use all the robots to build habitats. If $T < 1$, then the robots should only build habitats. In general, if the Hamiltonian is linear in u, the maximum or minimum of the Hamiltonian can only be attained on the boundaries of $\mathcal{U}$. The resulting control trajectory is known as bang-bang control. △

[1]Note that when $p(t) = 1$, $u(t)$ can be anything between 0 and 1. It can be shown that this choice does not make a difference in the incurred cost.

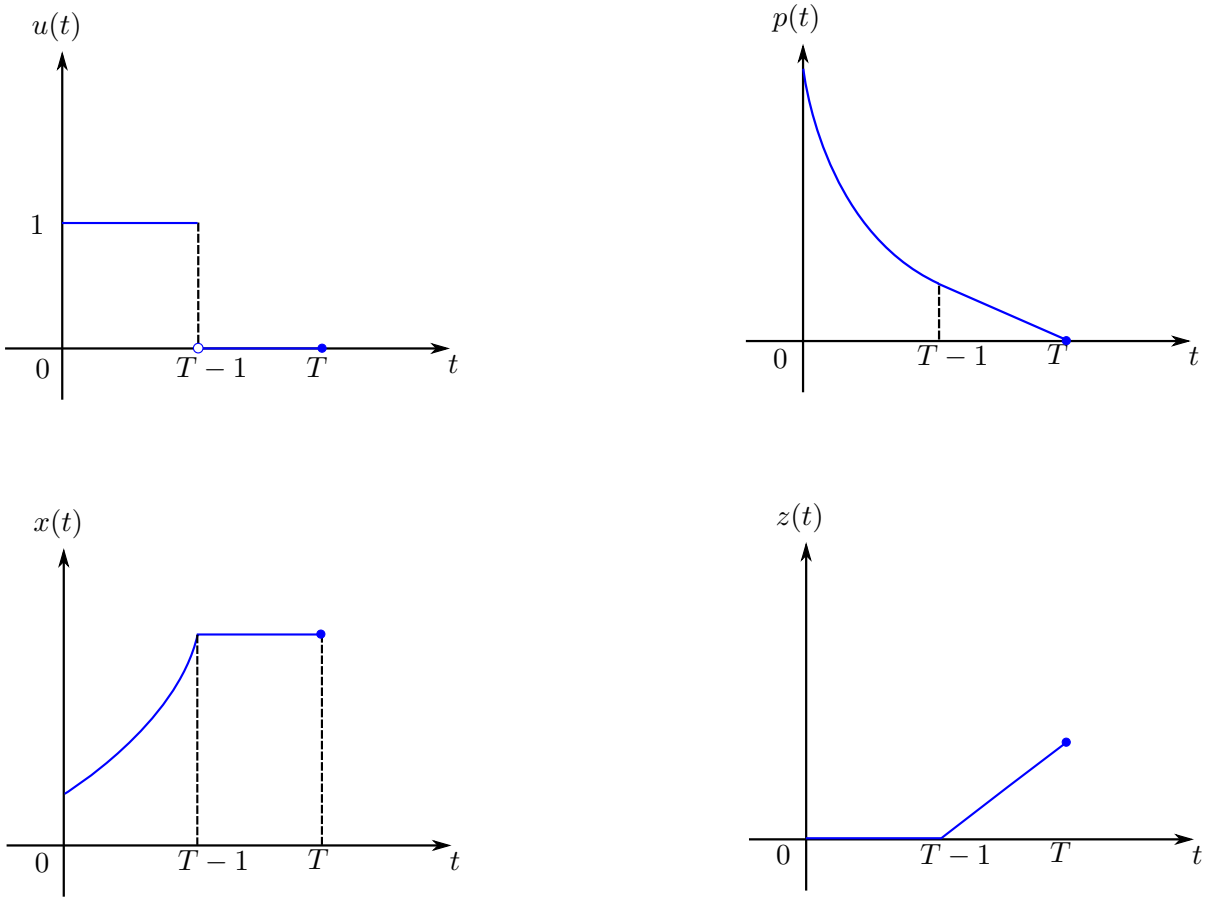


Figure 10.1: Optimal trajectories for the resource allocation example.

10.3 Summary

The Minimum Principle is a necessary condition for the optimal trajectory; in particular, it is possible that non-optimal trajectories satisfy the conditions outlined in Theorem 10.1. As we saw last lecture, the HJB is a sufficient condition for optimality; in particular, if a solution satisfies the HJB, then we are guaranteed that it is indeed optimal. This is summarized in Fig. 10.2.

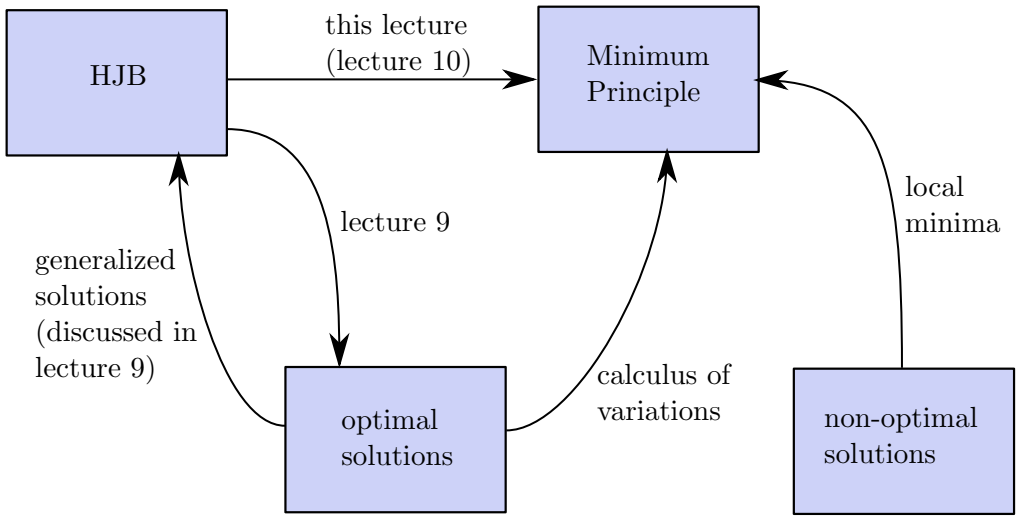


Figure 10.2: Optimal solutions and their relation to the HJB and the Minimum Principle.

11. Pontryagin's Minimum Principle (continued)

We now consider some variations of the continuous-time optimal control problem outlined in Section 10.2 and look at the corresponding adjustments to the Minimum Principle. For brevity, we will drop the $\cdot^*$ notation, as it is clear from context that only optimal trajectories are considered (e.g. $x^*(t)$ will become $x(t)$).

11.1 Fixed Terminal State

Suppose that in addition to the initial state $x(0) = \mathrm{x}_0$, the final state $x(T) = \mathrm{x}_T$ is given. The terminal cost $h(x(T))$ is then not useful as $x(T)$ is fixed. In effect, we have

$$J(T, \mathrm{x}) = \begin{cases} \text{some constant} & \text{if } \mathrm{x} = \mathrm{x}_T \\ \infty & \text{otherwise} \end{cases}$$

Thus $J(T, \mathrm{x})$ is not differentiable in x, and the terminal boundary condition $p(T) = \left.\dfrac{\partial h(\mathrm{x})}{\partial \mathrm{x}}\right|_{x(T)}^{\top}$ for the co-state equation does not hold. However, as compensation, we have n extra boundary conditions from $x(T) = \mathrm{x}_T$. We thus have $2n$ ordinary differential equations (ODEs) and $2n$ boundary conditions:

$$\begin{aligned} &\dot{x}(t) = f(x(t), u(t)), && x(0) = \mathrm{x}_0,\ x(T) = \mathrm{x}_T \\ &\dot{p}(t) = -\left.\frac{\partial H(\mathrm{x}, \mathrm{u}, \mathrm{p})}{\partial \mathrm{x}}\right|_{x(t),u(t),p(t)}^{\top} \end{aligned}$$

If only some of the terminal states are fixed, that is,

$$x_i(T) = \mathrm{x}_{T,i}, \quad \forall i \in I$$

where $x_i(T)$ and $\mathrm{x}_{T,i}$ denote the i^{th} entry of $x(T)$ and x_T, respectively, and I is an index set containing the indices of the fixed terminal states, we have the partial boundary condition

$$p_j(T) = \left.\frac{\partial h(\mathrm{x})}{\partial \mathrm{x}_j}\right|_{x(T)}, \quad \forall j \notin I$$

for the co-state equation. We still have $2n$ ODEs and $2n$ boundary conditions.

Example 1:
Consider the system

$$\dot{x}(t) = u(t), \quad u(t) \in \mathbb{R},\ 0 \le t \le 1,\ x(0) = 0,\ x(1) = 1$$

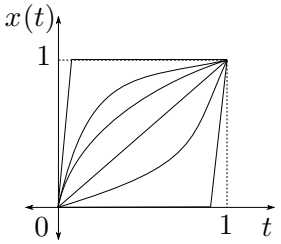


Figure 11.1: Possible state trajectories for Example 1. It is not obvious which one minimizes the cost function.

with cost function

$$\frac{1}{2}\int_0^1 \left(x(t)^2 + u(t)^2\right) \mathrm{d}t.$$

We want $x(t)$ to be small except at the very end to meet the terminal state condition. But this requires large $u(t)$ towards the end and when squared, will incur large cost. This indicates a trade-off in the control and state trajectories.
Applying the Minimum Principle:

$$\begin{aligned}
H(\mathrm{x},\mathrm{u},\mathrm{p}) &= \frac{1}{2}(\mathrm{x}^2+\mathrm{u}^2)+\mathrm{pu}\\
\dot{x}(t) &= u(t)\\
\dot{p}(t) &= -\left.\frac{\partial H(\mathrm{x},\mathrm{u},\mathrm{p})}{\partial \mathrm{x}}\right|^{\top}_{x(t),u(t),p(t)} = -x(t)\\
u(t) &= \arg\min_{\mathrm{u}\in\mathcal{U}} H(x(t),\mathrm{u},p(t)) = \arg\min_{\mathrm{u}\in\mathcal{U}}\left(\frac{1}{2}(x(t)^2+\mathrm{u}^2)+p(t)\mathrm{u}\right)\\
&= -p(t)
\end{aligned}$$

Thus,

$$\begin{aligned}
\dot{x}(t) &= -p(t)\\
\ddot{x}(t) &= -\dot{p}(t) = x(t).
\end{aligned}$$

The general solution to the ODE $\ddot{x}(t) = x(t)$ is $x(t) = Ae^t + Be^{-t}$, where A and B are constants to be determined from the boundary conditions:

$$\begin{aligned}
x(0) = 0 &\Rightarrow A + B = 0\\
x(1) = 1 &\Rightarrow Ae^1 + Be^{-1} = 1.
\end{aligned}$$

Thus,

$$\begin{aligned}
A = \frac{1}{e-e^{-1}},\ B &= -\frac{1}{e-e^{-1}};\\
x(t) &= \frac{e^t - e^{-t}}{e-e^{-1}}.
\end{aligned}$$

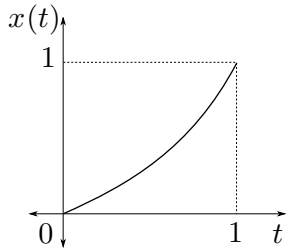


Figure 11.2: The optimal state trajectory for Example 1

△

11.2 Free Initial State

Recall $J(t, \mathrm{x})$ is the cost-to-go at time t and state x. Consider a problem where the initial state $x(0)$ is free and subject to optimization, and we add a cost of $l(x(0))$ to the cost function. For any $x(0)$ we have

$$\text{total cost} = l(x(0)) + J(0, x(0))\,.$$

A necessary condition for the optimal initial state $x(0) = \mathrm{x}_0$ that attains the minimum of the above is

$$\begin{aligned} &\left(\frac{\partial l(\mathrm{x})}{\partial \mathrm{x}} + \frac{\partial J(0,\mathrm{x})}{\partial \mathrm{x}}\right)\Big|_{\mathrm{x}_0} = 0 \\ \Rightarrow\ &\frac{\partial J(0,\mathrm{x})}{\partial \mathrm{x}}\Big|_{\mathrm{x}_0} = -\frac{\partial l(\mathrm{x})}{\partial \mathrm{x}}\Big|_{\mathrm{x}_0}. \end{aligned}$$

Recall that $p(0)$ is defined as $\dfrac{\partial J(0,\mathrm{x})}{\partial \mathrm{x}}\Big|_{x(0)}^{\top}$. Thus with the loss of the initial state boundary condition, we gain the adjoint boundary condition:

$$p(0) = -\frac{\partial l(\mathrm{x})}{\partial \mathrm{x}}\Big|_{x(0)}^{\top}.$$

Similar to Section 11.1, we can also deal with some parts of the initial state being free, while others are not.

Example 2:
Consider the same system as in Example 1, but with free initial state:

$$\dot{x}(t) = u(t), \quad u(t) \in \mathbb{R},\ 0 \le t \le 1,\ x(0) = \text{free},\ x(1) = 1$$

with cost function

$$\frac{1}{2}\int_0^1 \left(x(t)^2 + u(t)^2\right) \mathrm{d}t.$$

In particular, note that the initial state is not penalized, and that $l(x) = 0$ for all x. Here, picking $x(0) = 1$ would result in $u(t) = 0$ for all t, but we accumulate cost due to $x(t)$. On the other hand, if we set $x(0) = 0$, $x(t)$ will contribute to a lower cost but $u(t)$ will add a large cost as the input needs to drive the state to 1 by $t = 1$. Thus in this case there is also a trade-off.

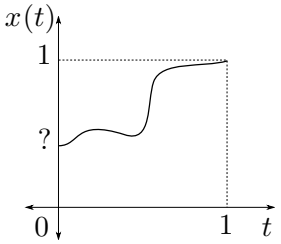


Figure 11.3: A possible state trajectory for Example 2.

Applying the Minimum Principle:

$$\begin{aligned}
H(\mathrm{x},\mathrm{u},\mathrm{p}) &= \frac{1}{2}(\mathrm{x}^2+\mathrm{u}^2)+\mathrm{pu} \\
\dot{x}(t) &= u(t) \\
\dot{p}(t) &= -\left.\frac{\partial H(\mathrm{x},\mathrm{u},\mathrm{p})}{\partial \mathrm{x}}\right|^{\top}_{x(t),u(t),p(t)} = -x(t) \\
u(t) &= \arg\min_{\mathrm{u}\in\mathcal{U}} H(x(t),\mathrm{u},p(t)) = \arg\min_{\mathrm{u}\in\mathcal{U}}\left(\frac{1}{2}(x(t)^2+\mathrm{u}^2)+p(t)\mathrm{u}\right) \\
&= -p(t)
\end{aligned}$$

Thus,

$$\begin{aligned}
\dot{x}(t) &= -p(t) \\
\ddot{x}(t) &= -\dot{p}(t) = x(t).
\end{aligned}$$

Furthermore, $l(\mathrm{x}) = 0$ for all x, thus

$$p(0) = -\left.\frac{\partial l(\mathrm{x})}{\partial \mathrm{x}}\right|^{\top}_{x(0)} = 0.$$

The general solution to the ODE $\ddot{x}(t) = x(t)$ is $x(t) = Ae^t + Be^{-t}$, and

$$p(t) = -\dot{x}(t) = -Ae^t + Be^{-t}$$

where A and B are constants to be determined from the boundary conditions:

$$\begin{aligned}
p(0) = 0 &\Rightarrow -A + B = 0 \\
x(1) = 1 &\Rightarrow Ae^1 + Be^{-1} = 1.
\end{aligned}$$

Thus,

$$\begin{aligned}
A = B &= \frac{1}{e+e^{-1}}; \\
x(t) &= \frac{e^t+e^{-t}}{e+e^{-1}} \\
\Rightarrow x(0) &\approx 0.65
\end{aligned}$$

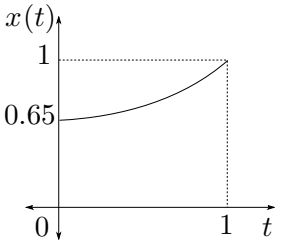


Figure 11.4: The optimal state trajectory for Example 2

△

11.3 Free Terminal Time

Suppose the initial state and/or the terminal state are given, but the terminal time T is free and subject to optimization.
We can compute the total cost for optimal trajectories for various terminal times T, and look for the T that minimizes this cost. This implies that if T is the optimal terminal time, then

$$\left.\frac{\partial J(t,\mathrm{x})}{\partial t}\right|_{0,x(0)} = 0.$$

But recall that on the optimal trajectory,

$$H(x(t),u(t),p(t)) = -\left.\frac{\partial J(t,\mathrm{x})}{\partial t}\right|_{x(t)} = \text{ constant } \forall t$$

Hence, this constant must be 0. In this case we gain an extra degree of freedom with free T, but lose a degree of freedom by the constraint

$$H(x(t),u(t),p(t)) = 0 \qquad \forall t \in [0,T]. \tag{11.1}$$

Example 3:
Consider the following problem setup with free terminal time T:

$$\begin{aligned} \dot{x}(t) &= u(t), \qquad u(t) \in \mathbb{R},\ x(0) = 0,\ x(T) = 1 \\ \text{Cost } &= \int_0^T \left(1 + \frac{1}{2}\left(x(t)^2 + u(t)^2\right)\right) \mathrm{d}t \\ \Rightarrow\ g(\mathrm{x},\mathrm{u}) &= 1 + \frac{1}{2}(\mathrm{x}^2 + \mathrm{u}^2) \end{aligned}$$

Applying the Minimum Principle:

$$\begin{aligned} H(\mathrm{x},\mathrm{u},\mathrm{p}) &= 1 + \frac{1}{2}(\mathrm{x}^2 + \mathrm{u}^2) + \mathrm{pu} \\ \dot{x}(t) &= u(t) \\ \dot{p}(t) &= -\left.\frac{\partial H(\mathrm{x},\mathrm{u},\mathrm{p})}{\partial \mathrm{x}}\right|^{\top}_{x(t),u(t),p(t)} = -x(t) \\ u(t) &= \arg\min_{\mathrm{u}\in\mathcal{U}} H(x(t),\mathrm{u},p(t)) = \arg\min_{\mathrm{u}\in\mathcal{U}} \left(1 + \frac{1}{2}(x(t)^2 + \mathrm{u}^2) + p(t)\mathrm{u}\right) \end{aligned}$$

$$= -p(t)$$

Thus,

$$\dot{x}(t) = -p(t)$$
$$\ddot{x}(t) = -\dot{p}(t) = x(t).$$

The general solution to the ODE $\ddot{x}(t) = x(t)$ is $x(t) = Ae^t + Be^{-t}$. Applying the boundary conditions:

$$x(0) = 0 \Rightarrow A + B = 0$$
$$x(T) = 1 \Rightarrow Ae^T + Be^{-T} = 1.$$

thus yielding

$$A = \frac{1}{e^T - e^{-T}}, B = -\frac{1}{e^T - e^{-T}}$$
$$x(t) = \frac{e^t - e^{-t}}{e^T - e^{-T}}$$
$$p(t) = -\dot{x}(t) = -\frac{e^t + e^{-t}}{e^T - e^{-T}}.$$

We can now apply (11.1):

$$\begin{aligned} 0 &= H(x(t), u(t), p(t)) \\ &= 1 + \frac{1}{2}(x(t)^2 + u(t)^2) + p(t)u(t) \\ &= 1 + \frac{1}{2}(x(t)^2 - p(t)^2) \\ &= 1 + \frac{1}{2}\left(\frac{(e^t - e^{-t})^2 - (e^t + e^{-t})^2}{(e^T - e^{-T})^2}\right) \\ &= 1 - \frac{2}{(e^T - e^{-T})^2} \end{aligned}$$
$$\Rightarrow T \approx 0.66$$

Note that had we not included the 1 in $g(\cdot, \cdot)$ (i.e. use the same $g(\cdot, \cdot)$ as in Example 1), we would get

$$\begin{aligned} 0 &= H(x(t), u(t), p(t)) \\ &= -\frac{2}{(e^T - e^{-T})^2} \end{aligned}$$
$$\Rightarrow T = \infty.$$

△

11.4 Time Varying System and Cost

Consider the case when the system and stage cost vary with time, that is

$$\begin{aligned} \dot{x}(t) &= f(x(t), u(t), t) \\ \text{cost} &= h(x(T)) + \int_0^T g(x(\tau), u(\tau), \tau)\mathrm{d}\tau. \end{aligned} \tag{11.2}$$

We can convert the problem to one involving a time-independent system and cost by introducing an extra state variable $y(t)$ representing time:

$$\begin{aligned} &\dot{y}(t) := 1, \quad y(0) = 0 \\ \Rightarrow\ &y(t) = t. \end{aligned}$$

Now we augment the system (11.2) as follows:

$$\begin{aligned} &z(t) := (x(t), y(t)) \\ &\dot{z}(t) = \begin{bmatrix} f(x(t), u(t), y(t)) \\ 1 \end{bmatrix} =: \bar{f}(z(t), u(t)) \\ &\text{cost} = \underbrace{\bar{h}(z(T))}_{:=h(x(T))} + \int_0^T \underbrace{\bar{g}(z(\tau), u(\tau))}_{:=g(x(\tau),u(\tau),y(\tau))} \, \mathrm{d}\tau. \end{aligned} \tag{11.3}$$

The augmented system (11.3) now fits the standard form we have been looking at. The Hamiltonian for the augmented system (11.3) is

$$\begin{aligned} \bar{H}(\mathrm{z}, \mathrm{u}, \bar{\mathrm{p}}) &= \bar{g}(\mathrm{z}, \mathrm{u}) + \bar{\mathrm{p}}^\top \bar{f}(\mathrm{z}, \mathrm{u}) \\ &= \underbrace{g(\mathrm{x}, \mathrm{u}, \mathrm{y}) + \mathrm{p}^\top f(\mathrm{x}, \mathrm{u}, \mathrm{y})}_{=:H(\mathrm{x},\mathrm{u},\mathrm{p},\mathrm{y})} + \mathrm{q} \\ &= H(\mathrm{x}, \mathrm{u}, \mathrm{p}, \mathrm{y}) + \mathrm{q} \end{aligned}$$

where $\mathrm{z} := (\mathrm{x}, \mathrm{y})$, and $\bar{\mathrm{p}} := (\mathrm{p}, \mathrm{q})$. Let $\bar{p}(t) := (p(t), q(t))$; the corresponding conditions of the Minimum Principle are

$$\begin{aligned} \dot{p}(t) &= -\left.\frac{\partial \bar{H}(\mathrm{z}, \mathrm{u}, \bar{\mathrm{p}})}{\partial \mathrm{x}}\right|^\top_{z(t),u(t),\bar{p}(t)} \\ &= -\left.\frac{\partial H(\mathrm{x}, \mathrm{u}, \mathrm{p}, t)}{\partial \mathrm{x}}\right|^\top_{x(t),u(t),p(t)} \\ \dot{q}(t) &= -\left.\frac{\partial \bar{H}(\mathrm{z}, \mathrm{u}, \bar{\mathrm{p}})}{\partial \mathrm{y}}\right|^\top_{z(t),u(t),\bar{p}(t)} \\ u(t) &= \arg\min_{\mathrm{u} \in \mathcal{U}} \bar{H}(z(t), \mathrm{u}, \bar{p}(t)) \\ &= \arg\min_{\mathrm{u} \in \mathcal{U}} H(x(t), \mathrm{u}, p(t), t) + q(t) \\ &= \arg\min_{\mathrm{u} \in \mathcal{U}} H(x(t), \mathrm{u}, p(t), t) \\ \bar{H}(z(t), u(t), \bar{p}(t)) &= \text{ constant } \Rightarrow H(x(t), u(t), p(t), t) + q(t) = \text{ constant} \end{aligned}$$

Thus, with the Hamiltonian of the original system (11.2)

$$H(\mathrm{x}, \mathrm{u}, \mathrm{p}, t) = g(\mathrm{x}, \mathrm{u}, t) + \mathrm{p}^\top f(\mathrm{x}, \mathrm{u}, t),$$

the conditions of the Minimum Principle are:

$$\begin{aligned} &\dot{p}(t) = -\left.\frac{\partial H(\mathrm{x}, \mathrm{u}, \mathrm{p}, t)}{\partial \mathrm{x}}\right|^\top_{x(t),u(t),p(t)}, \quad p(T) = \left.\frac{\partial h(\mathrm{x})}{\partial \mathrm{x}}\right|^\top_{x(T)} \\ &u(t) = \arg\min_{\mathrm{u} \in \mathcal{U}} H(x(t), \mathrm{u}, p(t)). \end{aligned}$$

Note that the only difference between the time-invariant and the time-varying cases is that the Hamiltonian in the latter one need not be constant along the optimal trajectory.

11.5 Singular Problems

In some cases, the minimum condition

$$u(t) = \arg\min_{\mathrm{u}\in\mathcal{U}} H(x(t), \mathrm{u}, p(t), t) \tag{11.4}$$

is insufficient to determine $u(t)$ for all t, because the values of $x(t)$ and $p(t)$ are such that $H(x(t), \mathrm{u}, p(t))$ is independent of u over a nontrivial interval of time. Such problems are called *singular*. Their optimal trajectories consist of portions, called *regular arcs*, where $u(t)$ can be determined from the minimum condition (11.4), and other portions, called *singular arcs*, which can be determined from the condition that the Hamiltonian is independent of u.

Example 4: Tracking Problem
Consider the system

$$\dot{x}(t) = u(t), \quad 0 \le t \le 1 \tag{11.5}$$

where $|u(t)| \le 1$ and $x(0)$ and $x(1)$ are free. The time-varying cost function is

$$\frac{1}{2}\int_0^1 (x(t) - z(t))^2 \, \mathrm{d}t$$

where $z(t) = 1 - t^2,\ 0 \le t \le 1$.

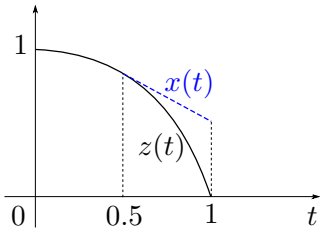


Figure 11.5: An example of a feasible state trajectory for Example 4; it tracks the desired trajectory $z(t)$ until the slope of $z(t)$ becomes less than -1 due to the input constraint.

Solution

We can write down the Hamiltonian

$$H(\mathrm{x}, \mathrm{u}, \mathrm{p}, t) = \frac{1}{2}(\mathrm{x} - z(t))^2 + \mathrm{pu}$$

and the co-state equation

$$\dot{p}(t) = -\left.\frac{\partial H(\mathrm{x}, \mathrm{u}, \mathrm{p}, t)}{\partial \mathrm{x}}\right|_{x(t),u(t),p(t)} = -(x(t) - z(t)) \tag{11.6}$$

with boundary conditions

$$p(0) = 0, \quad p(1) = 0.$$

Furthermore

$$u(t) = \arg\min_{|\mathrm{u}|\le 1} H(x(t), \mathrm{u}, p(t), t) = \arg\min_{|\mathrm{u}|\le 1} \left[\frac{1}{2}(x(t) - z(t))^2 + p(t)\mathrm{u}\right]$$

This leads to

$$u(t) = \begin{cases} -1 & \text{if } p(t) > 0 \\ 1 & \text{if } p(t) < 0 \\ \text{undetermined} & \text{if } p(t) = 0. \end{cases} \tag{11.7}$$

Now we determine the conditions for a singular arc to exist, namely, when $p(t)$ is zero for a non-trivial time interval. This implies that $\dot{p}(t) = 0$ over this time interval, thus by (11.6) we have

$$\begin{aligned} x(t) &= z(t) \\ \dot{x}(t) &= \dot{z}(t) = -2t. \end{aligned}$$

Invoking the system equation (11.5) we obtain the control input for the singular arc

$$u(t) = -2t.$$

Thus (11.7) becomes

$$u(t) = \begin{cases} -1 & \text{if } p(t) > 0 \\ 1 & \text{if } p(t) < 0 \\ -2t & \text{if } p(t) = 0. \end{cases}$$

Since $p(0) = 0$, we assume the state trajectory starts with a singular arc until time t_s. Note that t_s is at most $\frac{1}{2}$, since for $t > \frac{1}{2}$ the control constraint will be violated. The switch to the regular arc must happen at or before time $\frac{1}{2}$. From Figure 11.5 we can see that the control input should switch to $u(t) = -1$. Thus for $0 \le t \le t_s \le \frac{1}{2}$:

$$\begin{aligned} x(t) &= z(t) \\ p(t) &= 0 \end{aligned}$$

and for $t_s < t \le 1$:

$$\begin{aligned} & \dot{x}(t) = -1 \\ \implies\ & x(t) = z(t_s) - (t - t_s) = 1 - t_s^2 - t + t_s \\ & \dot{p}(t) = -(x(t) - z(t)) = -(1 - t_s^2 - t + t_s - 1 + t^2) = t_s^2 - t_s - t^2 + t \end{aligned}$$

From $p(1) = p(t_s) = 0$ we get

$$\begin{aligned} \int_{t_s}^{1} (t_s^2 - t_s - t^2 + t)\,\mathrm{d}t &= 0 \\ \left[t_s^2 t - t_s t - \frac{t^3}{3} + \frac{t^2}{2} \right]_{t_s}^{1} &= 0 \\ t_s^2 - t_s - \frac{1}{3} + \frac{1}{2} - t_s^3 + t_s^2 + \frac{t_s^3}{3} - \frac{t_s^2}{2} &= 0 \\ -4t_s^3 + 9t_s^2 - 6t_s + 1 &= 0 \\ (t_s - 1)(t_s - 1)(1 - 4t_s) &= 0 \end{aligned}$$

Clearly $t_s = 1$ is not possible, whereas $t_s = \frac{1}{4}$ satisfies the switch time constraint. Furthermore, we can verify that with $t_s = \frac{1}{4}$, $p(t) > 0$ for $t_s < t \le 1$, which is consistent with $u(t) = -1$ for $t_s < t \le 1$.

△