Dynamic Programming & Optimal Control

Lecture 7
Shortest Path Problems and Deterministic Finite State Systems

Fall 2023

Prof. Raffaello D'Andrea

ETH Zurich

Learning Objectives

Topic: Shortest Path Problems and Deterministic Finite State Systems

Objectives

- You know the *DP algorithm* for *deterministic, finite-state systems*.
- You understand which problems can be solved by the *forward DP algorithm*.
- You are able to convert a *shortest path problem* to a *DP problem* and vice versa.
- You understand the *Viterbi algorithm* used for estimating the most likely sequence of hidden states.
- You are able to draw the *Trellis diagram* for a given problem.

Outline

Shortest Path Problems and Deterministic Finite State Systems

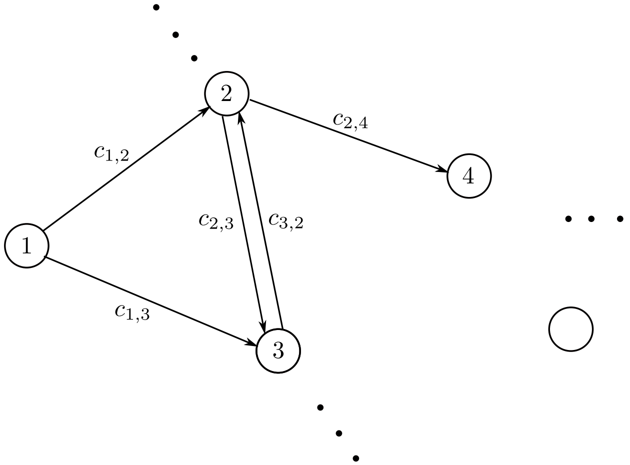
- The Shortest Path (SP) Problem
- Deterministic Finite State (DFS) Problem
- DFS to SP
- SP to DFS
- Hidden Markov Models and the Viterbi Algorithm
- Additional reading material

Graphs

A graph 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.



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 $\mathbb{Q}_{\mathsf{S},\mathsf{T}}$.

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

Find a path Q^* from node $\mathsf{S} \in \mathcal{V}$ to $\mathsf{T} \in \mathcal{V}$ that has the smallest length, i.e.:

$$Q^* = \underset{Q \in \mathbb{Q}_{\mathsf{S},\mathsf{T}}}{\arg\min}\, J_Q.$$

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

Assumption 7.1: No negative cycles
For all $i \in \mathcal{V}$ and for all $Q \in \mathbb{Q}_{i,i}$, $J_Q \geq 0$.

Assumption 7.1 implies $c_{i,i} \geq 0$ for all $i \in \mathcal{V}$.

Outline

Deterministic Finite State (DFS) Problem (1/2)

One method to find the SP in a graph is to map the SP problem to that of a deterministic, finite state system, on which we can apply standard DPA.

Consider the standard problem seen in Lecture 2 where there are no disturbances w_k and the state space $\mathcal{S}_k$ is a finite set for all $k = 0, ..., N$.

Dynamics

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

Cost Function

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

Deterministic Finite State (DFS) Problem (2/2)

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:

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

Nonetheless, this problem can be solved using the DPA.

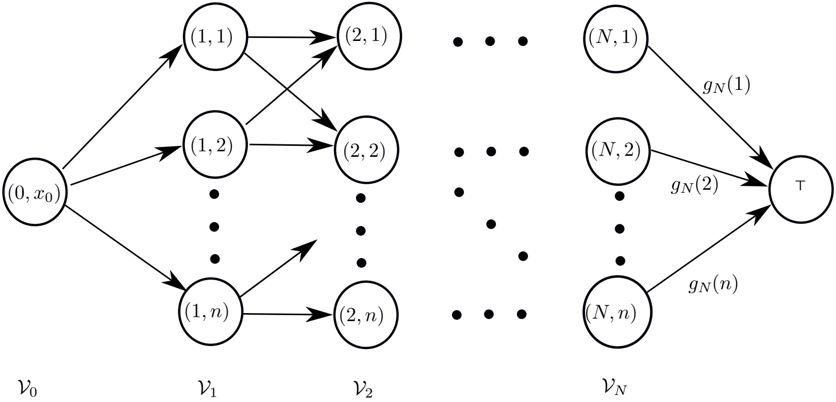
Outline

Shortest Path Problems and Deterministic Finite State Systems

- The Shortest Path (SP) Problem
- Deterministic Finite State (DFS) Problem
- DFS to SP
- SP to DFS
- Hidden Markov Models and the Viterbi Algorithm
- Additional reading material

DFS to SP (1/3)

We construct the graph of the associated problem. Without loss of generality, $\mathcal{S}_k = \{1, ..., n\}$ for $k = 1, ..., N$. Each node (k, x_k) in the graph represents the state $x_k \in \mathcal{S}_k$ at the k^{th} time period.



DFS to SP (2/3)

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 S and an artificial terminal node T is added to handle the final stage, such that the arc lengths to 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.

The objective of SP is to determine a path from node S to T that has the smallest length, which corresponds to finding the optimal state sequence (also the optimal control sequence since there is no disturbance) that has the smallest total cost.

DFS to SP (3/3)

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

$$\mathcal{V} := \left(\bigcup_{k=0}^{N} \mathcal{V}_k \right) \cup \{\mathsf{T}\},$$

where,

$$\begin{gathered} \mathcal{V}_0 := \{(0, x_0)\} \\ \mathcal{V}_k := \{(k, x_k) | x_k \in \mathcal{S}_k\},\ k = 1, \dots, N, \\ \mathsf{S} := (0, x_0). \end{gathered}$$

The weighted edge space is then:

$$\mathcal{C} :=$$

$$\left\{ ((k, x_k), (k+1, x_{k+1}), c) \middle| \begin{aligned} &(k, x_k) \in \mathcal{V}_k \\ &(k+1, x_{k+1}) \in \mathcal{V}_{k+1} \\ &c = \min_{\{u_k \in \mathcal{U}_k(x_k) | x_{k+1} = f_k(x_k, u_k)\}} g_k(x_k, u_k) \\ &k \in \{0, \dots, N-1\} \end{aligned} \right\} \bigcup$$

$$\left\{ \left((N, x_N),\ \mathsf{T},\ g_N(x_N) \right) \middle| \ (N, x_N) \in \mathcal{V}_N \right\}.$$

Outline

Shortest Path Problems and Deterministic Finite State Systems

- The Shortest Path (SP) Problem
- Deterministic Finite State (DFS) Problem
- DFS to SP
- SP to DFS
- Hidden Markov Models and the Viterbi Algorithm
- Additional reading material

SP to DFS (1/3)

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 $\mathrm{S} \in \mathcal{V}$ to $\mathrm{T} \in \mathcal{V}$.

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 (see next slide).

SP to DFS (2/3)

- The state space is:

$$\mathcal{S}_k := \mathcal{V} \backslash \{\mathsf{T}\} \text{ for } k = 1, ..., N-1, \mathcal{S}_N := \{\mathsf{T}\} \text{ and } \mathcal{S}_0 := \{\mathsf{S}\}.$$

- The control space is:

$$\mathcal{U}_k := \mathcal{V} \backslash \{\mathsf{T}\} \text{ for } k = 0, ..., N-2, \text{ and } \mathcal{U}_{N-1} := \{\mathsf{T}\}.$$

- 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:

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

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

SP to DFS (3/3)

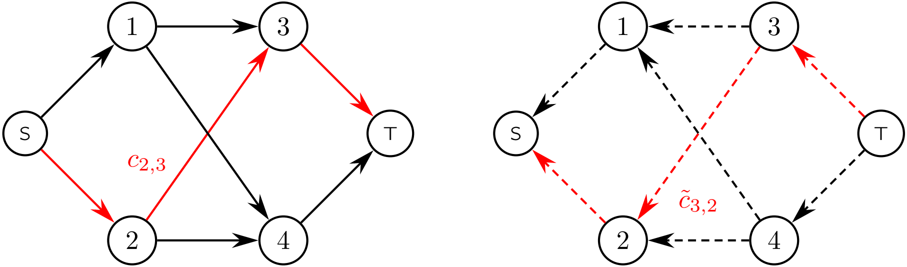
We can solve the DFS problem using DPA, where $J_k(i)$ is the optimal cost of getting from node i to node T in $N-k = |\mathcal{V}|-1-k$ moves:

$$\begin{aligned}
J_N(\mathsf{T}) &= g_N(\mathsf{T}) = 0, \\
J_k(i) &= \min_{u \in \mathcal{U}_k} \left(g_k(i,u) + J_{k+1}(u) \right), \quad \forall i \in \mathcal{S}_k,\ k = N-1, \ldots, 0, \\
\Rightarrow J_{N-1}(i) &= c_{i,T}, \quad \forall i \in \mathcal{V} \backslash \{\mathsf{T}\}, \\
J_k(i) &= \min_{j \in \mathcal{V} \backslash \{\mathsf{T}\}} \left(c_{i,j} + J_{k+1}(j) \right), \quad \forall i \in \mathcal{V} \backslash \{\mathsf{T}\},\ k = N-2, \ldots, 1, \\
J_0(\mathsf{S}) &= \min_{j \in \mathcal{V} \backslash \{\mathsf{T}\}} \left(c_{\mathsf{S},j} + J_1(j) \right).
\end{aligned}$$

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 (1/2)



Observation: 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.

Forward DP Algorithm (2/2)

We can now apply the DPA:

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

Let $J_l^{\mathsf{F}} := J_{N-l}$. 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).

Outline

Shortest Path Problems and Deterministic Finite State Systems

- The Shortest Path (SP) Problem
- Deterministic Finite State (DFS) Problem
- DFS to SP
- SP to DFS
- Hidden Markov Models and the Viterbi Algorithm
- Additional reading material

Hidden Markov Models (1/3)

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

Dynamics

Consider the finite state, time-invariant system

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

where $\mathcal{S} = \{1, \ldots, n\}$ is a finite set and the distribution $p_{w|x}$ is given.

The initial state x_0 is not known but its distribution p_{x_0} is.

This system is also known as a *Markov chain*.

Hidden Markov Models (2/3)

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

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

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*.

Such Markov chains 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.

Hidden Markov Models (3/3)

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

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

The Viterbi Algorithm (1/3)

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 $p(X_0, Z_1)$.

It can be shown (see Lecture Notes) that :

$$\begin{aligned}
p(X_0, Z_1) &= p(X_1, Z_1|x_0)\, p(x_0) \\
&= p(X_2, Z_2|x_0, x_1, z_1)\, p(z_1, x_1|x_0)\, p(x_0) \\
&= p(X_2, Z_2|x_0, x_1, z_1)\, p(z_1|x_1, x_0)\, p(x_1|x_0)\, p(x_0) \\
&= p(X_2, Z_2|x_0, x_1, z_1)\, M_{x_0 x_1}(z_1) P_{x_0 x_1} p(x_0) \\
&= \cdots \\
&= p(x_0) \prod_{k=1}^{N} P_{x_{k-1} x_k} M_{x_{k-1} x_k}(z_k).
\end{aligned}$$

The Viterbi Algorithm (2/3)

$$p(X_0, Z_1) = p(x_0) \prod_{k=1}^{N} P_{x_{k-1}x_k} M_{x_{k-1}x_k}(z_k)$$

Probabilities in the product are non-negative, thus maximizing $p(X_0, Z_1)$ is equivalent to:

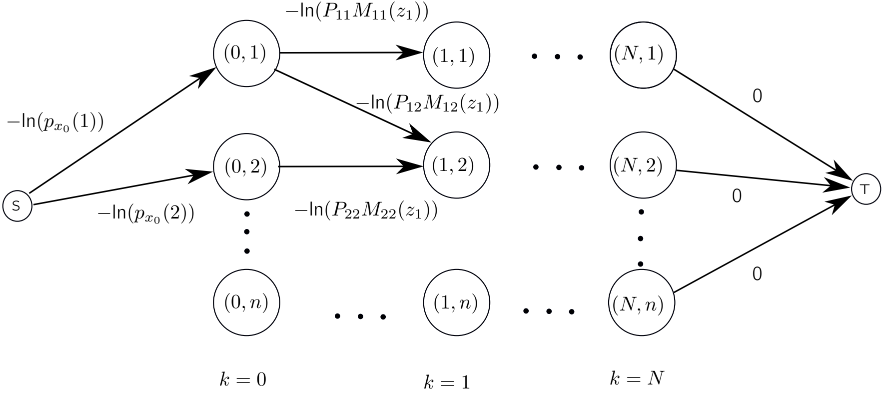
$$\underset{X_0}{\text{minimize}} \left(c_{\mathsf{S},(0,x_0)} + \sum_{k=1}^{N} c_{(k-1,x_{k-1}),(k,x_k)} \right),$$

where

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

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

The Viterbi Algorithm (3/3)



Outline

Additional reading material

Shortest Path Algorithms are among the most widely applied algorithms in practice. Some (non-exhaustive) examples:

1. In map applications (e.g., Google maps or robotic navigation) we look for the shortest route between two locations.
2. In computer networks we look for the most efficient way to dispatch packets from a starting node (e.g., the web server) to a terminal node (e.g., our web browser).
3. In social network and recommendation systems, we are interested in the "closeness" between two users or between a user and a topic to recommend.

We encourage you to think about every-day instances of SPPs!