Dynamic Programming Lecture #2

Outline:

- Deterministic DP
- Illustrations
- Extensions

Deterministic DP Framework

- System:

$$x_{k+1} = f_k(x_k, u_k)$$

  - State: $x_k \in S_k$
  - Control (decision): $u_k \in U_k(x_k)$

- Objective:

$$\begin{aligned} J^*(x_0) &= \min g_0(x_0, u_0) + \ldots + g_{N-1}(x_{N-1}, u_{N-1}) + g_N(x_N) \\ &= \min g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k) \end{aligned}$$

  - $g_k(x_k, u_k) \stackrel{\text{def}}{=}$ stage cost
  - $g_N(x_N) \stackrel{\text{def}}{=}$ terminal cost
  - $[0, \ldots, N] \stackrel{\text{def}}{=}$ (finite) horizon

- Issue: Minimize over what?

  - Open-loop viewpoint: Given x_0, produce $\{u_0^*, \ldots, u_{N-1}^*\}$.
  - Feedback viewpoint: Produce $\{\mu_0^*, \ldots, \mu_{N-1}^*\}$ and implement

$$u_k = \mu_k(x_k), \quad \text{where } \mu_k : S_k \to U_k(x_k)$$

  - In case of no uncertainty (randomness, disturbances, etc), two are SAME.

Deterministic DP Framework, cont (2)

- Example:

$$x_{k+1} = \frac{1}{2}x_k + u_k, \quad x_0 = 1$$

  - Compare:

$$u_0 = 1/2, u_1 = 1/2, \ldots, u_{N-1} = 1/2$$
$$u_k = \mu_k(x_k) = (1/2)x_k$$

  - BOTH lead to $x_1, \ldots, x_N = 1$ if $x_0 = 1$
  - DIFFERENT trajectories if $x_0 \neq 1$, eg

$$\{2, 3/2, 5/4, 9/8, 17/16, \ldots\} \quad \text{vs} \quad \{2, 2, 2, \ldots\}$$

- We will use POLICIES in anticipation of future discussion.
- Principle of optimality:
  - Full problem:

$$x_{k+1} = f_k(x_k, u_k)$$
$$\min_{\mu_0,\ldots,\mu_{N-1}} g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, \mu_k(x_k))$$
$$\text{where} \quad \mu_k : S_k \to U_k(x_k)$$

  - Partial problem:

$$\min_{\mu_k,\ldots,\mu_{N-1}} g_N(x_N) + \sum_{\tilde{k}=k}^{N-1} g_{\tilde{k}}(x_{\tilde{k}}, \mu_{\tilde{k}}(x_{\tilde{k}}))$$

  - If $\{\mu_0^*, \ldots, \mu_{N-1}^*\}$ is optimal for full problem, then the "tail" $\{\mu_k^*, \ldots, \mu_{N-1}^*\}$ must be optimal for for partial problem.

DP Algorithm

- Theorem: Define

$$J_N(x_N) = g_N(x_N)$$
$$J_k(x_k) = \min_{u_k \in U_k(x_k)} g_k(x_k, u_k) + J_{k+1}(f_k(x_k, u_k))$$

  Then

  - $J^*(x_0) = J_0(x_0)$
  - $\mu_k^*(x_k) = \arg\min_{u_k \in U_k(x_k)} g_k(x_k, u_k) + J_{k+1}(f_k(x_k, u_k))$

- Comments:

  - Algorithm produces optimal cost AND policies.
  - One multistage problem converted into several 1-stage problem.
  - Associate $J_k(x_k)$ as "optimal cost-to-go" starting from x_k, i.e., partial solution:

$$J_k(x_k) = \min_{\mu_k, \ldots, \mu_{N-1}} g_N(x_N) + \sum_{\tilde{k}=k}^{N-1} g_{\tilde{k}}(x_{\tilde{k}}, \mu_{\tilde{k}}(x_{\tilde{k}}))$$

- Proof:

  - Assume J_{k+1} is optimal cost-to-go, i.e., solution to subproblem.
  - $J_N(x_N) = g_N(x_N)$ is indeed optimal cost-to-go.

$$\begin{aligned}
&\min_{\mu_k, \ldots, \mu_{N-1}} g_N(x_N) + \sum_{\tilde{k}=k}^{N-1} g_{\tilde{k}}(x_{\tilde{k}}, \mu_{\tilde{k}}(x_{\tilde{k}})) \\
&\quad = \min_{\mu_k} \left(g_k(x_k, \mu_k(x_k)) + \min_{\mu_{k+1}, \ldots, \mu_{N-1}} \left(g_N(x_N) + \sum_{\tilde{k}=k+1}^{N-1} g_{\tilde{k}}(x_{\tilde{k}}, \mu_{\tilde{k}}(x_{\tilde{k}})) \right) \right) \\
&\quad = \min_{\mu_k} g_k(x_k, \mu_k(x_k)) + J_{k+1}(x_{k+1}) \\
&\quad = \min_{u_k \in U_k(x_k)} g_k(x_k, u_k) + J_{k+1}(f(x_k, u_k)) \\
&\quad \stackrel{\text{def}}{=} J_k(x_k)
\end{aligned}$$

Formulation Examples

- Shortest path:

$$x_{k+1} = u_k$$
$$g_k(x_k, u_k) = \ell^k_{x_k u_k}$$
$$g_N(x_N) = 0$$

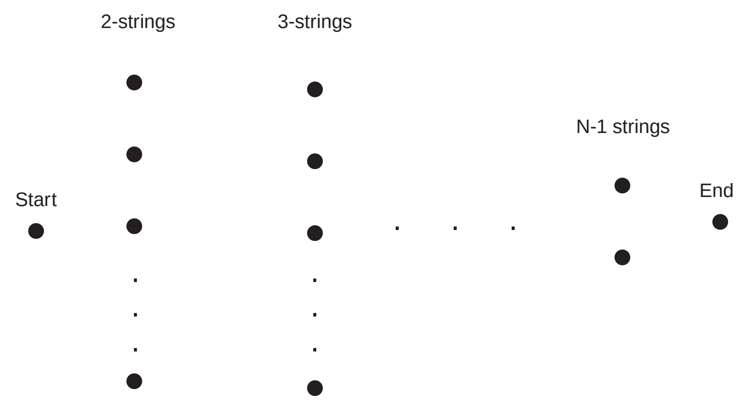
  - $U_k(x_k)$ defines connectivity
  - Or can set $\ell^k_{x_k u_k} = \infty$ for illegal jumps

- Can exploit shortest path algorithms (tailored DP).
- Example: Matrix multiplication (problem 1.16).
  - Cost to multiply $A \times B$: # rows$(A)\times$ # cols$(A)\times$ # cols(B)
  - Consider

$$\begin{aligned} M_1 M_2 M_3 &= (1 \times 10)(10 \times 1)(1 \times 10) \\ &= (\, M_1 M_2 \,)\, M_3 = 20 \text{ multiplications} \\ &= M_1 \,(\, M_2 M_3 \,) = 200 \text{ multiplications} \end{aligned}$$

  - What is optimal sequence to multiply $M = M_1 M_2 \ldots M_N$?
  - Two formulations:
    1. Split products allowed:
$$M = (M_1 M_2)(M_3 M_4)$$
    2. Split products not allowed:
$$M = M_1((M_2 M_3) M_4)$$

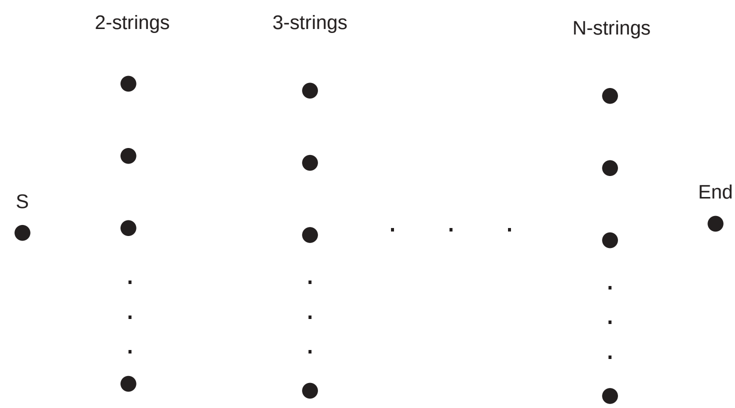
Matrix Example



- Split products not allowed...what is state?
  - 2-strings: $\{(M_1 M_2), (M_2 M_3), \ldots\}$
  - 3-strings: $\{(M_1 M_2 M_3), (M_2 M_3 M_4), \ldots\}$
  - $N-1$-strings: $\{(M_1 \ldots M_{N-1}), (M_2 \ldots M_N)\}$
  - Can define cost of node jump.
  - Not all nodes connected.
- Split products allowed...what is state?

Traveling Salesman

- Given N cities, minimize total distance to visit each city once starting from S and ending at S.



- What is state?
- Example: $\{S, A, B, C, D\}$
  - 2-strings: $\{SA, SB, SC, SD\}$
  - 3-strings: $\{SAB, SAC, SAD, SBA, SBC, SBD, \ldots\}$
  - N-strings: $\{SABC, SABD, SACB, SACD, SBAC, \ldots\}$
- $N!$ different paths

Variations

- Multiplicative positive cost:

$$J = g_0(x_0, \mu_0(x_0)) \times \ldots \times g_N(x_N)$$

  - Since $\log(\cdot)$ is monotonic, equivalent cost:

$$\begin{aligned} \tilde{J} &= \log(g_0(x_0, \mu_0(x_0)) \times \ldots \times g_N(x_N)) \\ &= \log(g_0(x_0, \mu_0(x_0)) + \ldots + \log(g_N(x_N)) \\ &= \tilde{g}_N(x_N) + \sum_{k=0}^{N-1} \tilde{g}(x_k, \mu_k(x_k)) \end{aligned}$$

  - Do not have principle of optimality for nonnegative multiplicative costs!

- History dependent controls: Today's decision constrained by yesterday's decision.
  - What is state?

$$x_{k+1}^{\text{aug}} = \begin{pmatrix} x_{k+1} \\ v_{k+1} \end{pmatrix} = \begin{pmatrix} f_k(x_k, u_k) \\ u_k \end{pmatrix}$$

  - Constraint $u_k \in U(x_k^{\text{aug}}) = U(x_k, v_k) = U(x_k, u_{k-1})$
  - Can similarly augment history dependent (random) disturbances.

Variations, cont (2)

- Terminating processes:
  - Play until end:

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

  OR terminate early:

$$J = T + \sum_{k=0}^{k^*} g_k(x_k, \mu_k(x_k))$$

  - Augment state space: x_T & u_T
  - New state dynamics:

$$\begin{aligned} x_{k+1} &= \tilde{f}_k(x_k, u_k) \\ &= \begin{cases} f_k(x_k, u_k) & x_k \neq x_T \text{ and } u_k \neq u_T; \\ x_T & x_k = x_T \text{ or } u_k = u_T \end{cases} \end{aligned}$$

  - New stage cost:

$$\tilde{g}_k(x_k, u_k) = \begin{cases} g_k(x_k, u_k) & x_k \neq x_T \text{ and } u_k \neq u_T; \\ T & x_k \neq x_T \text{ and } u_k = u_T; \\ 0 & x_k = x_T \end{cases}$$

  - New terminal cost:

$$\tilde{g}_N(x_N) = \begin{cases} g_N(x_N) & x_N \neq x_T; \\ 0 & x_N = x_T \end{cases}$$