# 6.231 DYNAMIC PROGRAMMING

# LECTURE 2

# LECTURE OUTLINE

- The basic problem
- Principle of optimality
- DP example: Deterministic problem
- DP example: Stochastic problem
- The general DP algorithm
- State augmentation

# BASIC PROBLEM

- System $x_{k+1} = f_k(x_k, u_k, w_k)$, $k = 0, \ldots, N-1$
- Control constraints $u_k \in U_k(x_k)$
- Probability distribution $P_k(\cdot \mid x_k, u_k)$ of $w_k$
- Policies $\pi = \{\mu_0, \ldots, \mu_{N-1}\}$, where $\mu_k$ maps states $x_k$ into controls $u_k = \mu_k(x_k)$ and is such that $\mu_k(x_k) \in U_k(x_k)$ for all $x_k$
- Expected cost of $\pi$ starting at $x_0$ is

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

- Optimal cost function

$$J^*(x_0) = \min_\pi J_\pi(x_0)$$

- Optimal policy $\pi^*$ is one that satisfies

$$J_{\pi^*}(x_0) = J^*(x_0)$$

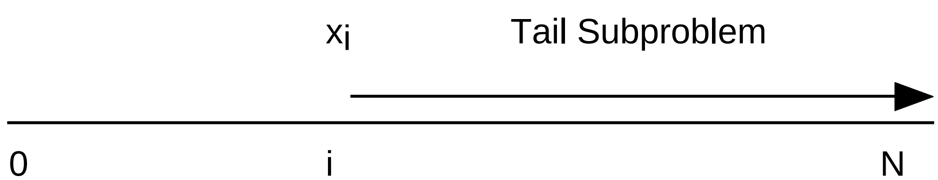
# PRINCIPLE OF OPTIMALITY

- Let $\pi^* = \{\mu_0^*, \mu_1^*, \ldots, \mu_{N-1}^*\}$ be optimal policy

- Consider the "tail subproblem" whereby we are at $x_i$ at time $i$ and wish to minimize the "cost-to-go" from time $i$ to time $N$

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

and the "tail policy" $\{\mu_i^*, \mu_{i+1}^*, \ldots, \mu_{N-1}^*\}$



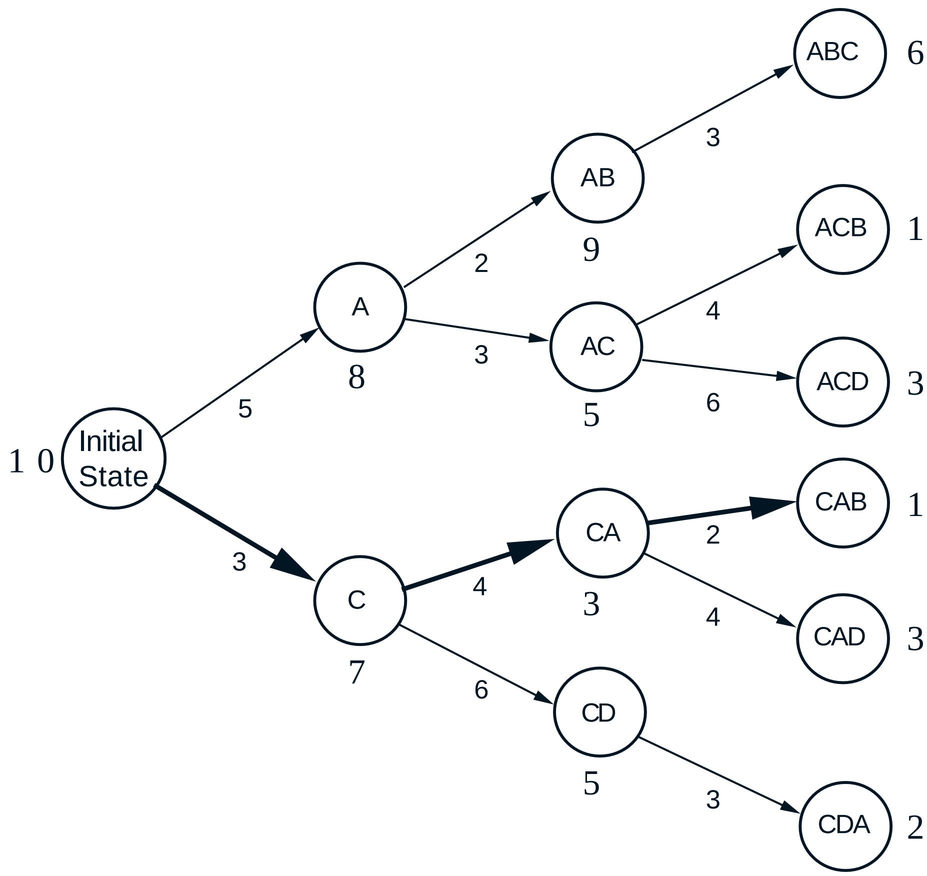

- *Principle of optimality*: The tail policy is optimal for the tail subproblem (optimization of the future does not depend on what we did in the past)

- DP first solves ALL tail subroblems of final stage

- At the generic step, it solves ALL tail subproblems of a given time length, using the solution of the tail subproblems of shorter time length

# DETERMINISTIC SCHEDULING EXAMPLE

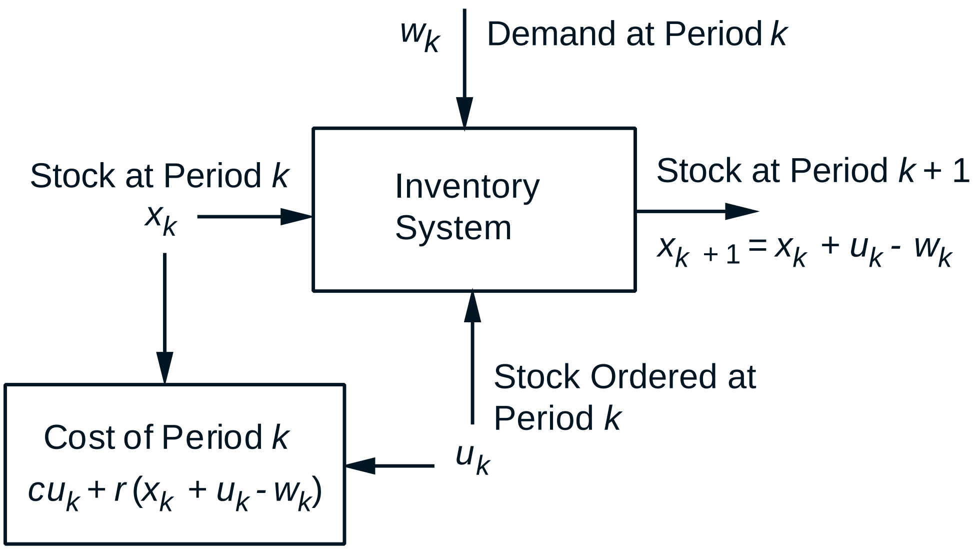
- Find optimal sequence of operations A, B, C, D (A must precede B and C must precede D)




- Start from the last tail subproblem and go backwards

- At each state-time pair, we record the optimal cost-to-go and the optimal decision

# STOCHASTIC INVENTORY EXAMPLE




- Tail Subproblems of Length 1:

$$J_{N-1}(x_{N-1}) = \min_{u_{N-1} \geq 0} \underset{w_{N-1}}{E} \Big\{ cu_{N-1} + r(x_{N-1} + u_{N-1} - w_{N-1}) \Big\}$$

- Tail Subproblems of Length $N-k$:

$$J_k(x_k) = \min_{u_k \geq 0} \underset{w_k}{E} \Big\{ cu_k + r(x_k + u_k - w_k) + J_{k+1}(x_k + u_k - w_k) \Big\}$$

- $J_0(x_0)$ is opt. cost of initial state $x_0$

# DP ALGORITHM

- Start with

$$J_N(x_N) = g_N(x_N),$$

and go backwards using

$$J_k(x_k) = \min_{u_k \in U_k(x_k)} \mathop{E}_{w_k} \Big\{ g_k(x_k, u_k, w_k) + J_{k+1}\big(f_k(x_k, u_k, w_k)\big) \Big\}, \quad k = 0, 1, \ldots, N-1.$$

- Then $J_0(x_0)$, generated at the last step, is equal to the optimal cost $J^*(x_0)$. Also, the policy

$$\pi^* = \{\mu_0^*, \ldots, \mu_{N-1}^*\}$$

where $\mu_k^*(x_k)$ minimizes in the right side above for each $x_k$ and $k$, is optimal

- Justification: Proof by induction that $J_k(x_k)$ is equal to $J_k^*(x_k)$, defined as the optimal cost of the tail subproblem that starts at time $k$ at state $x_k$

- Note:
  - ALL the tail subproblems are solved (in addition to the original problem)
  - Intensive computational requirements

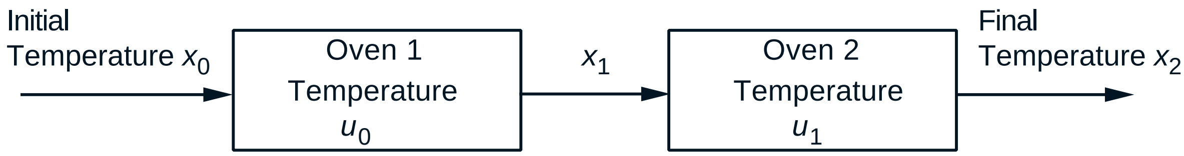
# PROOF OF THE INDUCTION STEP

- Let $\pi_k = \{\mu_k, \mu_{k+1}, \ldots, \mu_{N-1}\}$ denote a tail policy from time $k$ onward
- Assume that $J_{k+1}(x_{k+1}) = J^*_{k+1}(x_{k+1})$. Then

$$
\begin{aligned}
J^*_k(x_k) = & \min_{(\mu_k,\pi_{k+1})} \mathop{E}_{w_k,\ldots,w_{N-1}} \Bigg\{ g_k\big(x_k, \mu_k(x_k), w_k\big) \\
& \qquad + g_N(x_N) + \sum_{i=k+1}^{N-1} g_i\big(x_i, \mu_i(x_i), w_i\big) \Bigg\}
\end{aligned}
$$

$$
\begin{aligned}
&= \min_{\mu_k} \mathop{E}_{w_k} \Bigg\{ g_k\big(x_k, \mu_k(x_k), w_k\big) \\
&\quad + \min_{\pi_{k+1}} \Bigg[ \mathop{E}_{w_{k+1},\ldots,w_{N-1}} \Bigg\{ g_N(x_N) + \sum_{i=k+1}^{N-1} g_i\big(x_i, \mu_i(x_i), w_i\big) \Bigg\} \Bigg] \Bigg\} \\
&= \min_{\mu_k} \mathop{E}_{w_k} \Big\{ g_k\big(x_k, \mu_k(x_k), w_k\big) + J^*_{k+1}\Big(f_k\big(x_k, \mu_k(x_k), w_k\big)\Big) \Big\} \\
&= \min_{\mu_k} \mathop{E}_{w_k} \Big\{ g_k\big(x_k, \mu_k(x_k), w_k\big) + J_{k+1}\Big(f_k\big(x_k, \mu_k(x_k), w_k\big)\Big) \Big\} \\
&= \min_{u_k \in U_k(x_k)} \mathop{E}_{w_k} \Big\{ g_k(x_k, u_k, w_k) + J_{k+1}\big(f_k(x_k, u_k, w_k)\big) \Big\} \\
&= J_k(x_k)
\end{aligned}
$$

# LINEAR-QUADRATIC ANALYTICAL EXAMPLE




- System

$$x_{k+1} = (1-a)x_k + au_k, \qquad k = 0, 1,$$

where $a$ is given scalar from the interval $(0,1)$

- Cost

$$r(x_2 - T)^2 + u_0^2 + u_1^2$$

where $r$ is given positive scalar

- DP Algorithm:

$$J_2(x_2) = r(x_2 - T)^2$$

$$J_1(x_1) = \min_{u_1} \left[ u_1^2 + r\Big((1-a)x_1 + au_1 - T\Big)^2 \right]$$

$$J_0(x_0) = \min_{u_0} \Big[ u_0^2 + J_1\big((1-a)x_0 + au_0\big) \Big]$$

# STATE AUGMENTATION

- When assumptions of the basic problem are violated (e.g., disturbances are correlated, cost is nonadditive, etc) reformulate/augment the state
- DP algorithm still applies, but the problem gets BIGGER
- Example: Time lags

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

- Introduce additional state variable $y_k = x_{k-1}$. New system takes the form

$$\begin{pmatrix} x_{k+1} \\ y_{k+1} \end{pmatrix} = \begin{pmatrix} f_k(x_k, y_k, u_k, w_k) \\ x_k \end{pmatrix}$$

View $\tilde{x}_k = (x_k, y_k)$ as the new state.

- DP algorithm for the reformulated problem:

$$J_k(x_k, x_{k-1}) = \min_{u_k \in U_k(x_k)} \mathop{E}_{w_k} \Big\{ g_k(x_k, u_k, w_k) + J_{k+1}\big(f_k(x_k, x_{k-1}, u_k, w_k), x_k\big) \Big\}$$

6.231 Dynamic Programming and Stochastic Control
Fall 2015