ECE276B: Planning & Learning in Robotics
Lecture 4: The Dynamic Programming Algorithm

Instructor:

Nikolay Atanasov: natanasov@ucsd.edu

Teaching Assistant:

Hanwen Cao: h1cao@ucsd.edu

UC San Diego

JACOBS SCHOOL OF ENGINEERING

Electrical and Computer Engineering

Dynamic Programming

- **Control policy**: a function π that maps a time step $t \in \mathbb{N}$ and a state $\mathbf{x} \in \mathcal{X}$ to a feasible control input $\mathbf{u} \in \mathcal{U}$

- **Value function** $V_t^\pi(\mathbf{x})$: expected long-term cost starting in state $\mathbf{x}$ at time t and following policy π

- **Objective**: construct an optimal control policy:

$$\pi^* = \arg\min_\pi V_0^\pi(\mathbf{x}_0)$$

- **Dynamic Programming Algorithm** obtains an optimal control policy given an MDP model
  - Idea: uses the value function to structure the search for good policies
  - Generality: can handle non-convex and non-linear problems
  - Complexity: polynomial in the number of states and actions
  - Efficiency: much more efficient than a brute-force approach evaluating all possible strategies

Principle of Optimality

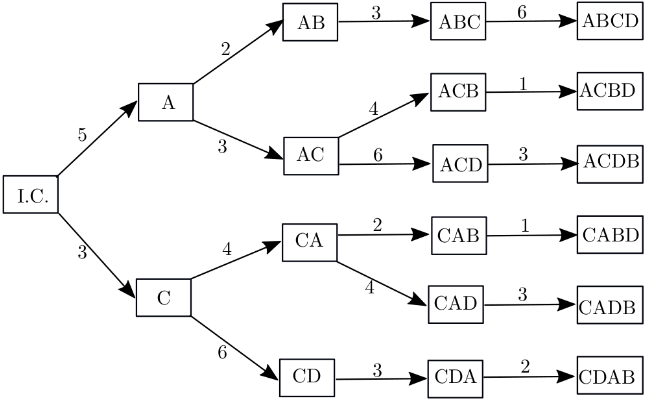
- Let $\pi^*_{0:T-1}$ be an optimal control policy
- Consider a **subproblem**, minimizing the value at state $\mathbf{x}$ at time t:

$$V_t^{\pi}(\mathbf{x}) = \mathbb{E}_{\mathbf{x}_{t+1:T}}\left[\gamma^{T-t}\mathfrak{q}(\mathbf{x}_T) + \sum_{\tau=t}^{T-1}\gamma^{\tau-t}\ell(\mathbf{x}_\tau, \pi_\tau(\mathbf{x}_\tau)) \,\middle|\, \mathbf{x}_t = \mathbf{x}\right]$$

- **Principle of optimality**: the truncated policy $\pi^*_{t:T-1}$ is optimal for the subproblem $\min_\pi V_t^\pi(\mathbf{x})$ at time t
- **Intuition**: Suppose $\pi^*_{t:T-1}$ were not optimal for the subproblem. Then, there would exist a policy yielding a lower cost on at least some portion of the state space.

Example: Deterministic Scheduling Problem

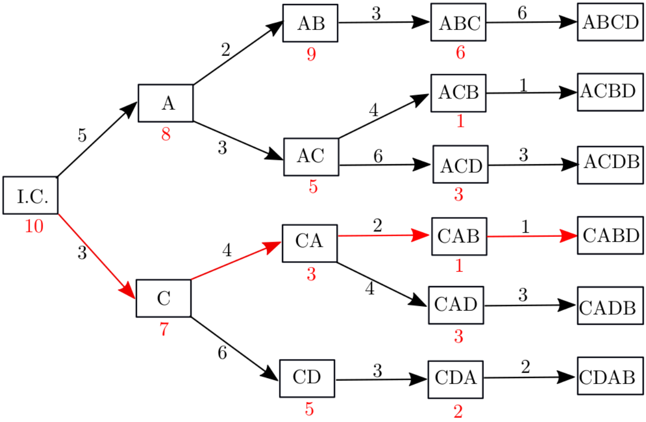
- Consider a deterministic scheduling problem where 4 operations A, B, C, D are used to produce a product
- Rules: Operation A must occur before B, and C before D
- Cost: there is a transition cost between each two operations:



Example: Deterministic Scheduling Problem

- Dynamic programming is applied backwards in time. First, construct an optimal solution at the last stage and then work backwards.

- The optimal cost-to-go at each state of the scheduling problem is denoted with red text below the state:



The Dynamic Programming Algorithm

Algorithm 1 Dynamic Programming

Input: MDP $(\mathcal{X}, \mathcal{U}, p_0, p_f, T, \ell, \mathfrak{q}, \gamma)$

$V_T(\mathbf{x}) = \mathfrak{q}(\mathbf{x}), \quad \forall \mathbf{x} \in \mathcal{X}$

for $t = (T-1) \dots 0$ **do**

$\quad Q_t(\mathbf{x}, \mathbf{u}) = \ell(\mathbf{x}, \mathbf{u}) + \gamma \mathbb{E}_{\mathbf{x}' \sim p_f(\cdot|\mathbf{x}, \mathbf{u})} [V_{t+1}(\mathbf{x}')], \quad \forall \mathbf{x} \in \mathcal{X}, \mathbf{u} \in \mathcal{U}(\mathbf{x})$

$\quad V_t(\mathbf{x}) = \min_{\mathbf{u} \in \mathcal{U}(\mathbf{x})} Q_t(\mathbf{x}, \mathbf{u}), \quad \forall \mathbf{x} \in \mathcal{X}$

$\quad \pi_t(\mathbf{x}) = \arg\min_{\mathbf{u} \in \mathcal{U}(\mathbf{x})} Q_t(\mathbf{x}, \mathbf{u}), \quad \forall \mathbf{x} \in \mathcal{X}$

return policy $\pi_{0:T-1}$ and value function V_0

The expected value function at $\mathbf{x}' \sim p_f(\cdot|\mathbf{x}, \mathbf{u})$ is:

- Continuous $\mathcal{X}$: $\mathbb{E}_{\mathbf{x}' \sim p_f(\cdot|\mathbf{x}, \mathbf{u})} \left[V_{t+1}(\mathbf{x}') \right] = \int V_{t+1}(\mathbf{x}') p_f(\mathbf{x}' \mid \mathbf{x}, \mathbf{u}) d\mathbf{x}'$
- Discrete $\mathcal{X}$: $\mathbb{E}_{\mathbf{x}' \sim p_f(\cdot|\mathbf{x}, \mathbf{u})} \left[V_{t+1}(\mathbf{x}') \right] = \sum_{\mathbf{x}' \in \mathcal{X}} V_{t+1}(\mathbf{x}') p_f(\mathbf{x}' \mid \mathbf{x}, \mathbf{u})$

The Dynamic Programming Algorithm

- At each step, all possible states $\mathbf{x} \in \mathcal{X}$ are considered because we do not know a priori which states will be visited
- This point-wise optimization at each $\mathbf{x} \in \mathcal{X}$ is what gives us a policy $\pi_t(\mathbf{x})$, i.e., a function specifying a control input for **every** state $\mathbf{x} \in \mathcal{X}$
- Consider a discrete-space example with $|\mathcal{X}| = 10$ states, $|\mathcal{U}| = 10$ control inputs, planning horizon $T = 4$, and given x_0:
  - There are $|\mathcal{U}|^T = 10^4$ different open-loop strategies
  - There are $|\mathcal{U}|^{|\mathcal{X}|(T-1)+1} = 10^{31}$ different closed-loop strategies
  - For each stage t and each state $\mathbf{x}$, the DP algorithm goes through the $|\mathcal{U}|$ control inputs to determine the optimal input. In total, there are $|\mathcal{U}||\mathcal{X}|(T-1) + |\mathcal{U}| = 310$ such operations.

Dynamic Programming Optimality

Theorem: Optimality of the DP Algorithm

The policy $\pi_{0:T-1}$ and value function V_0 returned by the DP algorithm are optimal for the finite-horizon optimal control problem.

- **Proof**:
  - Let $V_t^*(\mathbf{x})$ be the optimal cost for the $(T-t)$-stage problem that starts at time t in state $\mathbf{x}$.
  - Proceed by induction
  - **Base-case**: $V_T^*(\mathbf{x}) = \mathfrak{q}(\mathbf{x}) = V_T(\mathbf{x})$
  - **Hypothesis**: Assume that for $t+1$, $V_{t+1}^*(\mathbf{x}) = V_{t+1}(\mathbf{x})$ for all $\mathbf{x} \in \mathcal{X}$
  - **Induction**: Show that $V_t^*(\mathbf{x}) = V_t(\mathbf{x})$ for all $\mathbf{x} \in \mathcal{X}$

Proof of Dynamic Programming Optimality

$$
\begin{aligned}
V_t^*(\mathbf{x}_t) &= \min_{\pi_{t:T-1}} \mathbb{E}_{\mathbf{x}_{t+1:T}|\mathbf{x}_t}\left[\gamma^{T-t}\mathfrak{q}(\mathbf{x}_T) + \sum_{\tau=t}^{T-1}\gamma^{\tau-t}\ell(\mathbf{x}_\tau,\pi_\tau(\mathbf{x}_\tau))\right] \\
&= \min_{\pi_{t:T-1}} \mathbb{E}_{\mathbf{x}_{t+1:T}|\mathbf{x}_t}\left[\ell(\mathbf{x}_t,\pi_t(\mathbf{x}_t)) + \gamma^{T-t}\mathfrak{q}(\mathbf{x}_T) + \sum_{\tau=t+1}^{T-1}\gamma^{\tau-t}\ell(\mathbf{x}_\tau,\pi_\tau(\mathbf{x}_\tau))\right] \\
&\overset{(1)}{=} \min_{\pi_{t:T-1}} \ell(\mathbf{x}_t,\pi_t(\mathbf{x}_t)) + \mathbb{E}_{\mathbf{x}_{t+1:T}|\mathbf{x}_t}\left[\gamma^{T-t}\mathfrak{q}(\mathbf{x}_T) + \sum_{\tau=t+1}^{T-1}\gamma^{\tau-t}\ell(\mathbf{x}_\tau,\pi_\tau(\mathbf{x}_\tau))\right] \\
&\overset{(2)}{=} \min_{\pi_{t:T-1}} \ell(\mathbf{x}_t,\pi_t(\mathbf{x}_t)) + \gamma\mathbb{E}_{\mathbf{x}_{t+1}|\mathbf{x}_t}\left[\mathbb{E}_{\mathbf{x}_{t+2:T}|\mathbf{x}_{t+1}}\left[\gamma^{T-t-1}\mathfrak{q}(\mathbf{x}_T) + \sum_{\tau=t+1}^{T-1}\gamma^{\tau-t-1}\ell(\mathbf{x}_\tau,\pi_\tau(\mathbf{x}_\tau))\right]\right] \\
&\overset{(3)}{=} \min_{\pi_t}\left\{\ell(\mathbf{x}_t,\pi_t(\mathbf{x}_t)) + \gamma\mathbb{E}_{\mathbf{x}_{t+1}|\mathbf{x}_t}\left[\min_{\pi_{t+1:T-1}}\mathbb{E}_{\mathbf{x}_{t+2:T}|\mathbf{x}_{t+1}}\left[\gamma^{T-t-1}\mathfrak{q}(\mathbf{x}_T) + \sum_{\tau=t+1}^{T-1}\gamma^{\tau-t-1}\ell(\mathbf{x}_\tau,\pi_\tau(\mathbf{x}_\tau))\right]\right]\right\} \\
&\overset{(4)}{=} \min_{\pi_t}\left\{\ell(\mathbf{x}_t,\pi_t(\mathbf{x}_t)) + \gamma\mathbb{E}_{\mathbf{x}_{t+1}\sim p_f(\cdot|\mathbf{x}_t,\pi_t(\mathbf{x}_t))}\left[V_{t+1}^*(\mathbf{x}_{t+1})\right]\right\} \\
&\overset{(5)}{=} \min_{\mathbf{u}_t\in\mathcal{U}(\mathbf{x}_t)}\left\{\ell(\mathbf{x}_t,\mathbf{u}_t) + \gamma\mathbb{E}_{\mathbf{x}_{t+1}\sim p_f(\cdot|\mathbf{x}_t,\mathbf{u}_t)}\left[V_{t+1}(\mathbf{x}_{t+1})\right]\right\} \\
&= V_t(\mathbf{x}_t), \quad \forall \mathbf{x}_t \in \mathcal{X}
\end{aligned}
$$

Proof of Dynamic Programming Optimality

(1) Since $\ell(\mathbf{x}_t, \pi_t(\mathbf{x}_t))$ is not a function of $\mathbf{x}_{t+1:T}$

(2) Using conditional probability $p(\mathbf{x}_{t+1:T}|\mathbf{x}_t) = p(\mathbf{x}_{t+2:T}|\mathbf{x}_{t+1}, \mathbf{x}_t)p(\mathbf{x}_{t+1}|\mathbf{x}_t)$ and the Markov assumption

(3) The minimization can be split since the term $\ell(\mathbf{x}_t, \pi_t(\mathbf{x}_t))$ does not depend on $\pi_{t+1:T-1}$. The expectation $\mathbb{E}_{\mathbf{x}_{t+1}|\mathbf{x}_t}$ and $\min_{\pi_{t+1:T}}$ can be exchanged since the functions $\pi_{t+1:T-1}$ make the cost small for all initial conditions., i.e., independently of $\mathbf{x}_{t+1}$.

- (1)-(3) is the *principle of optimality*

(4) By definition of $V^*_{t+1}(\cdot)$ and the motion model $\mathbf{x}_{t+1} \sim p_f(\cdot \mid \mathbf{x}_t, \mathbf{u}_t)$

(5) By the induction hypothesis

Example: Chess Strategy Optimization

- State: $x_t \in \mathcal{X} := \{-2, -1, 0, 1, 2\}$ – the difference between our and the opponent's score at the end of game t
- Input: $u_t \in \mathcal{U} := \{timid, bold\}$
- Dynamics: with $p_d > p_w$:

$$\begin{aligned} p_f(x_{t+1} = x_t \mid u_t = timid, x_t) &= p_d \\ p_f(x_{t+1} = x_t - 1 \mid u_t = timid, x_t) &= 1 - p_d \\ p_f(x_{t+1} = x_t + 1 \mid u_t = bold, x_t) &= p_w \\ p_f(x_{t+1} = x_t - 1 \mid u_t = bold, x_t) &= 1 - p_w \end{aligned}$$

- Cost: $V_t(x_t) = \mathbb{E}\left[\mathfrak{q}(x_2) + \sum_{\tau=t}^{1} \underbrace{\ell(x_\tau, u_\tau)}_{=0}\right]$ with

$$\mathfrak{q}(x) = \begin{cases} -1 & \text{if } x > 0 \\ -p_w & \text{if } x = 0 \\ 0 & \text{if } x < 0 \end{cases}$$

Example: Chess Strategy Optimization

- Initialize: $V_2(x_2) = \begin{cases} -1 & \text{if } x_2 > 0 \\ -p_w & \text{if } x_2 = 0 \\ 0 & \text{if } x_2 < 0 \end{cases}$
- Recursion: for all $x_t \in \mathcal{X}$ and $t = 1, 0$:

$$V_t(x_t) = \min_{u_t \in \mathcal{U}} \left\{ \ell(x_t, u_t) + \mathbb{E}_{x_{t+1}|x_t, u_t} \left[V_{t+1}(x_{t+1}) \right] \right\}$$

$$= \min \left\{ \underbrace{p_d V_{t+1}(x_t) + (1 - p_d) V_{t+1}(x_t - 1)}_{\text{timid}}, \underbrace{p_w V_{t+1}(x_t + 1) + (1 - p_w) V_{t+1}(x_t - 1)}_{\text{bold}} \right\}$$

Example: Chess Strategy Optimization

- $x_1 = 1$:

$$\begin{aligned} V_1(1) &= -\max\{p_d + (1-p_d)p_w, p_w + (1-p_w)p_w\} \xlongequal[p_d > p_w]{\text{since}} \\ &= -p_d - (1-p_d)p_w \\ \pi_1^*(1) &= \mathit{timid} \end{aligned}$$

- $x_1 = 0$:

$$\begin{aligned} V_1(0) &= -\max\{p_d p_w + (1-p_d)0, p_w + (1-p_w)0\} = -p_w \\ \pi_1^*(0) &= \mathit{bold} \end{aligned}$$

- $x_1 = -1$:

$$\begin{aligned} V_1(-1) &= -\max\{p_d 0 + (1-p_d)0, p_w p_w + (1-p_w)0\} = -p_w^2 \\ \pi_1^*(-1) &= \mathit{bold} \end{aligned}$$

Example: Chess Strategy Optimization

- $x_0 = 0$:

$$\begin{aligned} V_0(0) &= -\max\{p_d V_1(0) + (1-p_d)V_1(-1), p_w V_1(1) + (1-p_w)V_1(-1)\} \\ &= -\max\left\{p_d p_w + (1-p_d)p_w^2, p_w(p_d + (1-p_d)p_w) + (1-p_w)p_w^2\right\} \\ &= -p_d p_w - (1-p_d)p_w^2 - (1-p_w)p_w^2 \end{aligned}$$

$$\pi_0^*(0) = bold$$

- As before, the optimal strategy is to play timid iff ahead in the score.

Example: Deterministic Nonlinear System

- Consider a system with state $x_t \in \mathbb{R}$, control $\mathbf{u}_t := [a_t,\ b_t] \in \mathbb{R}^2$ and motion model:

$$x_{t+1} = f(x_t, \mathbf{u}_t) = a_t x_t + b_t$$

- Calculate the optimal value function $V_0^*(x)$ at time $t = 0$ and an optimal policy $\pi_t^*(x)$ for $t \in \{0, 1\}$, which minimize the total cost:

$$x_2 + a_1^2 + a_0^2 + b_1^2 + b_0^2$$

- Planning horizon: $T = 2$
- Terminal cost: $\mathfrak{q}(x) = x$
- Stage cost: $\ell(x, \mathbf{u}) = \|\mathbf{u}\|_2^2 = a^2 + b^2$
- Discount factor: $\gamma = 1$

Example: Deterministic Nonlinear System

- Dynamic programming algorithm at $t = T = 2$:

$$V_2^*(x_2) = \mathfrak{q}(x_2) = x_2, \qquad \forall x_2 \in \mathbb{R}$$

- At $t = 1$:

$$V_1^*(x_1) = \min_{\mathbf{u}_1} \{\ell(x_1, \mathbf{u}_1) + V_2^*(f(x_1, \mathbf{u}_1))\} = \min_{a_1, b_1} \left\{a_1^2 + b_1^2 + a_1 x_1 + b_1\right\}$$

  - Obtain minimum by setting gradient with respect to $\mathbf{u}_1$ to zero:

$$\frac{\partial}{\partial a_1}\left(a_1^2 + b_1^2 + a_1 x_1 + b_1\right) = 2a_1 + x_1 = 0$$
$$\frac{\partial}{\partial b_1}\left(a_1^2 + b_1^2 + a_1 x_1 + b_1\right) = 2b_1 + 1 = 0$$

  leading to $a_1^* = -\frac{1}{2}x_1$ and $b_1^* = -\frac{1}{2}$

  - To confirm this is a minimizer, check that Hessian matrix $\begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$ is positive definite

Example: Deterministic Nonlinear System

- At $t = 1$:
  - Optimal policy at $t = 1$: $\pi_1^*(x_1) = -\frac{1}{2}\begin{bmatrix} x_1 \\ 1 \end{bmatrix}$
  - Substituting the optimal policy into the value function:

$$V_1^*(x_1) = \left(-\frac{1}{2}x_1\right)^2 + \left(-\frac{1}{2}\right)^2 + \left(-\frac{1}{2}x_1\right)x_1 + \left(-\frac{1}{2}\right) = -\frac{1}{4}x_1^2 - \frac{1}{4}$$

- At $t = 0$:

$$\begin{aligned} V_0^*(x_0) &= \min_{\mathbf{u}_0} \{\ell(x_0, \mathbf{u}_0) + V_1^*(f(x_0, \mathbf{u}_0))\} \\ &= \min_{a_0, b_0} \left\{ a_0^2 + b_0^2 - \frac{1}{4}(a_0 x_0 + b_0)^2 - \frac{1}{4} \right\} \\ &= \min_{a_0, b_0} \left\{ \left(1 - \frac{1}{4}x_0^2\right) a_0^2 + \frac{3}{4} b_0^2 - \frac{1}{2} a_0 b_0 x_0 - \frac{1}{4} \right\} \end{aligned}$$

Example: Deterministic Nonlinear System

- At $t = 0$:
  - Obtain minimum by setting gradient with respect to $\mathbf{u}_0$ to zero:

$$\frac{\partial}{\partial a_0}\left(\left(1-\frac{1}{4}x_0^2\right)a_0^2+\frac{3}{4}b_0^2-\frac{1}{2}a_0b_0x_0-\frac{1}{4}\right)=2a_0-\frac{1}{2}a_0x_0^2-\frac{1}{2}b_0x_0=0$$

$$\frac{\partial}{\partial b_0}\left(\left(1-\frac{1}{4}x_0^2\right)a_0^2+\frac{3}{4}b_0^2-\frac{1}{2}a_0b_0x_0-\frac{1}{4}\right)=\frac{3}{2}b_0-\frac{1}{2}a_0x_0=0$$

$$\Rightarrow \qquad \frac{1}{2}\begin{bmatrix}4-x_0^2 & -x_0\\ -x_0 & 3\end{bmatrix}\begin{bmatrix}a_0\\ b_0\end{bmatrix}=\begin{bmatrix}0\\ 0\end{bmatrix}$$

  - For $x_0 \neq \pm\sqrt{3}$, the Hessian matrix $\frac{1}{2}\begin{bmatrix}4-x_0^2 & -x_0\\ -x_0 & 3\end{bmatrix}$ is positive definite and $a_0^* = b_0^* = 0$
  - For $x_0 = \pm\sqrt{3}$, $a_0^* = \pm\sqrt{3}b_0^*$. Hence we can still choose $b_0^* = a_0^* = 0$
  - Optimal policy at $t = 0$: $\pi_0^*(x_0) = \begin{bmatrix}0\\ 0\end{bmatrix}$
  - Substituting the optimal policy into the value function: $V_0^*(x_0) = -\frac{1}{4}$