Dynamic programming

CMPT 882

Feb. 15

Outline

- Principle of optimality

- Discrete Hamilton-Jacobi equation
- Discrete LQR

- Continuous Hamilton-Jacobi equation
- Continuous LQR

Optimal Control: Types of Solutions

$$\underset{u(\cdot)}{\text{minimize}} \; l\left(x\left(t_f\right), t_f\right) + \int_0^{t_f} c(x(t), u(t), t) dt$$

$$\text{subject to } \dot{x}(t) = f\left(x(t), u(t)\right)$$
$$x(t) \in \mathbb{R}^n, u(t) \in \mathbb{R}^m, x(0) = x_0$$

- Open-loop control
  - Scalable, but errors will add up
- Closed-loop control
  - Find $u(t, x)$ for $t \in \left[0, t_f\right]$, $x \in \mathbb{R}^n$
  - Not scalable, but robust
  - "Special" techniques needed (eg. Reinforcement learning) for large n
- Receding horizon control:
  - Has features of both open- and closed-loop control

Optimal Control Problem

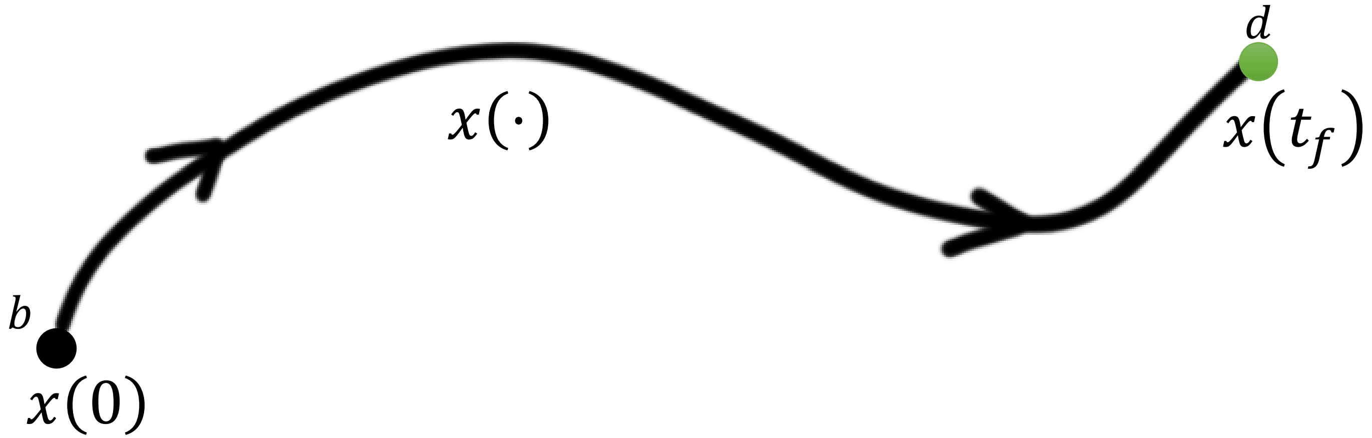
$$\underset{u(\cdot)}{\text{minimize}} \; \overbrace{l\left(x(t_f), t_f\right)}^{\text{Final cost}} + \overbrace{\int_0^{t_f} c(x(t), u(t), t)\,dt}^{\text{Running cost}}$$

Cost functional, $J\left(x(\cdot), u(\cdot)\right)$

$$\text{subject to } \dot{x}(t) = f\left(x(t), u(t)\right)$$

Dynamic model

$$x(t) \in \mathbb{R}^n, u(t) \in \mathbb{R}^m, x(0) = x_0$$

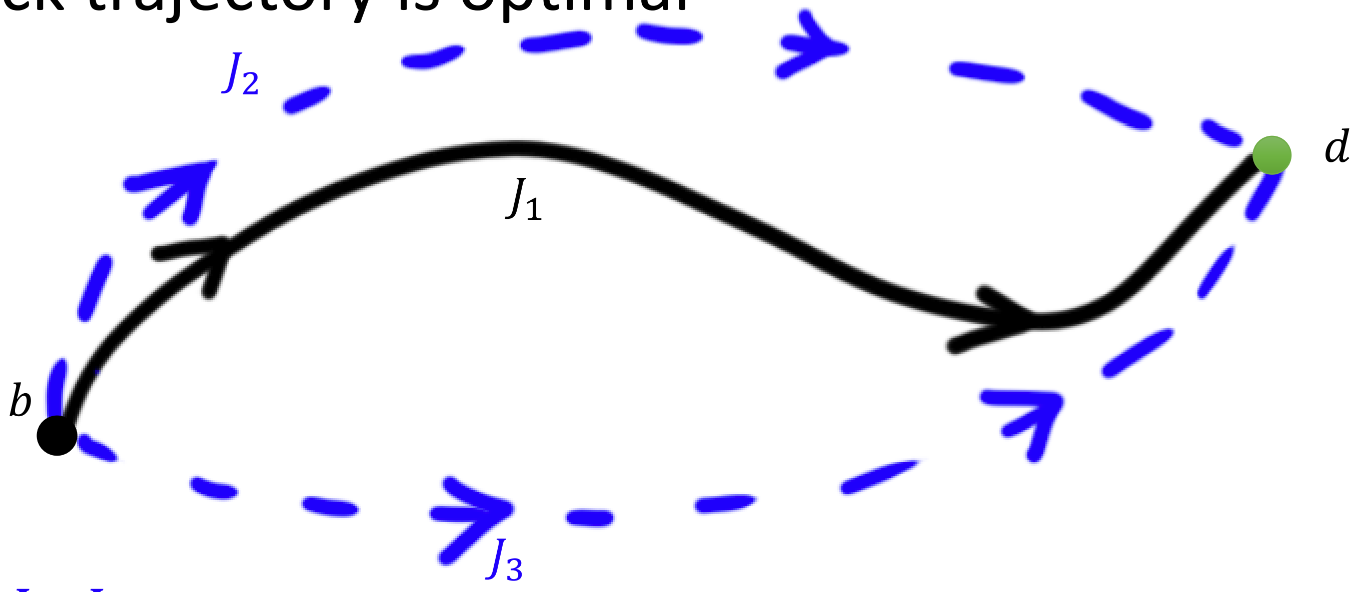


Dynamic Programming

- Pros
  - Globally optimal solutions
  - Closed-loop (state feedback) control: $u = u(t, x)$
    - More robust
- Cons
  - Poor scalability except special cases

Optimal Trajectory

- Suppose black trajectory is optimal



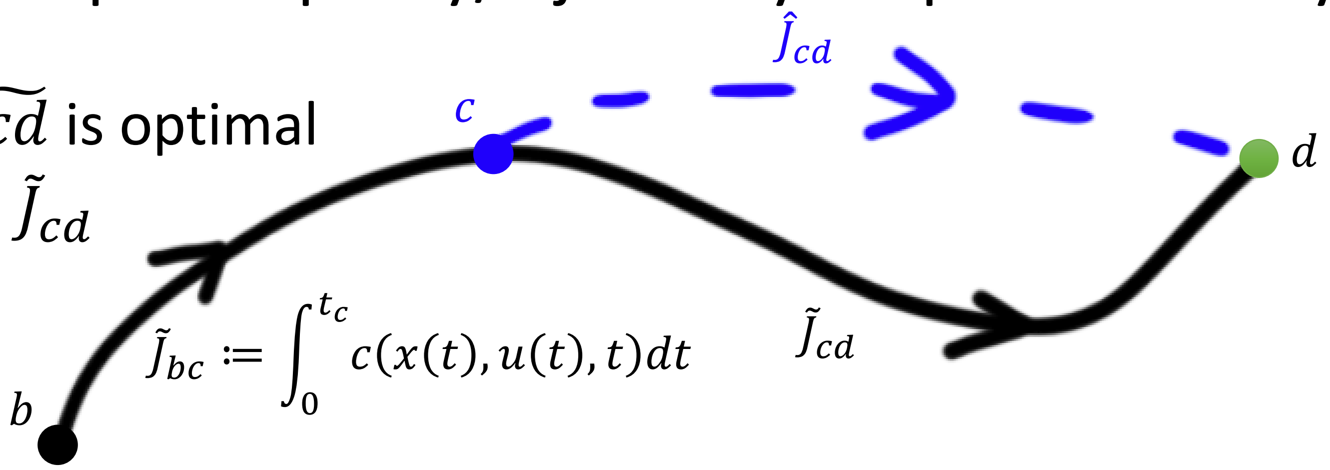
- Then $J_1 \leq J_2, J_3$
- Let $J_{bd}^{*} = J_1$

Principle of Optimality

- Optimal cost: J^*_{bd}

Principle of Optimality

- Any truncated optimal policy/trjaectory is optimal for any “tail” subproblem
  - Black path $\widetilde{cd}$ is optimal
  - $J^*_{bd} = \tilde{J}_{bc} + \tilde{J}_{cd}$

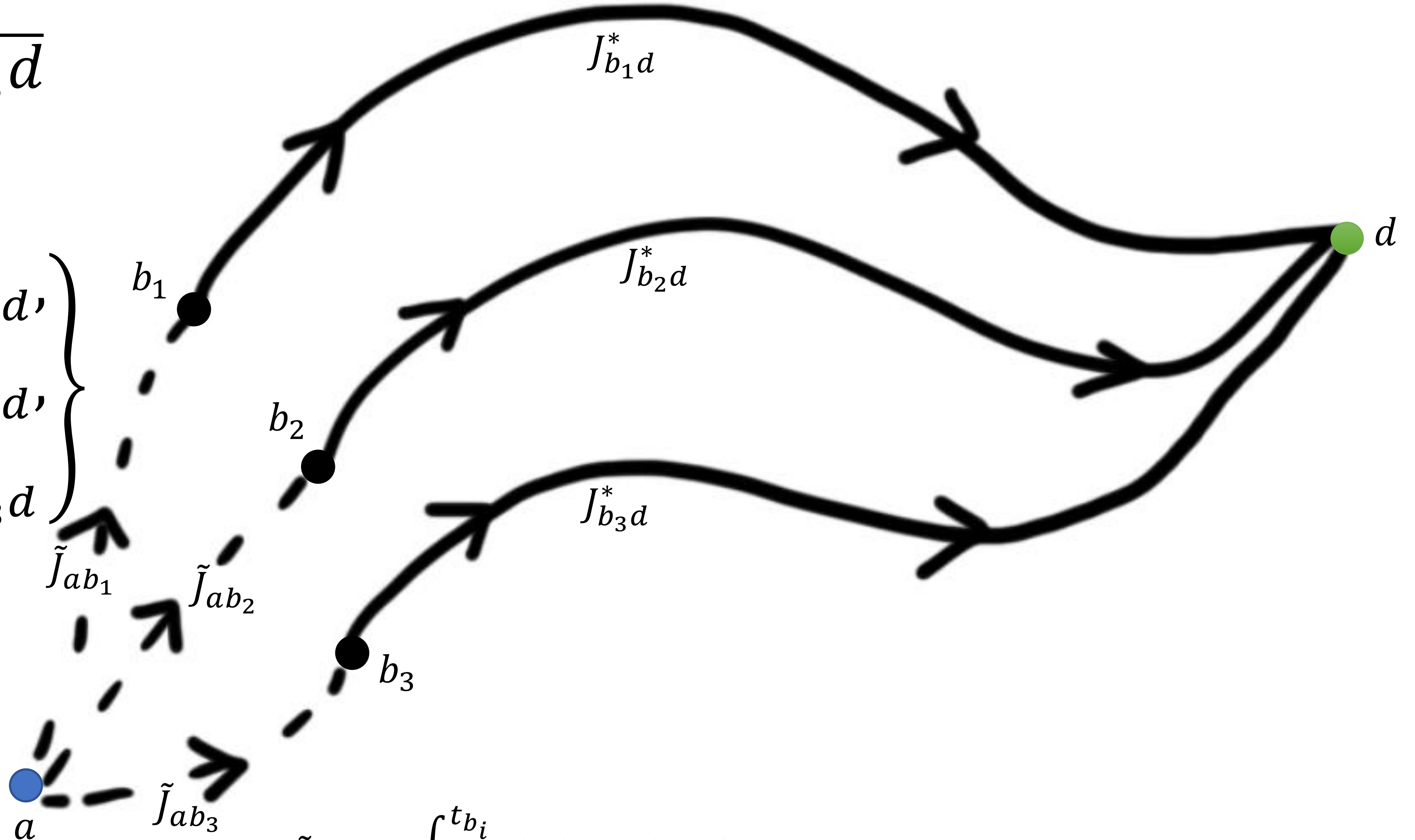


- Proof:
  - Suppose not, then there is some other path from c to d with cost $\hat{J}_{cd}$ such that $\hat{J}_{cd} < \tilde{J}_{cd}$
  - This means $J^*_{bd} = \tilde{J}_{bc} + \tilde{J}_{cd} > \underline{\tilde{J}_{bc} + \hat{J}_{cd}}$
  - Therefore, the original trajectory $\overline{bd}$ is not optimal ← contradiction!

Applying the Principle of Optimality

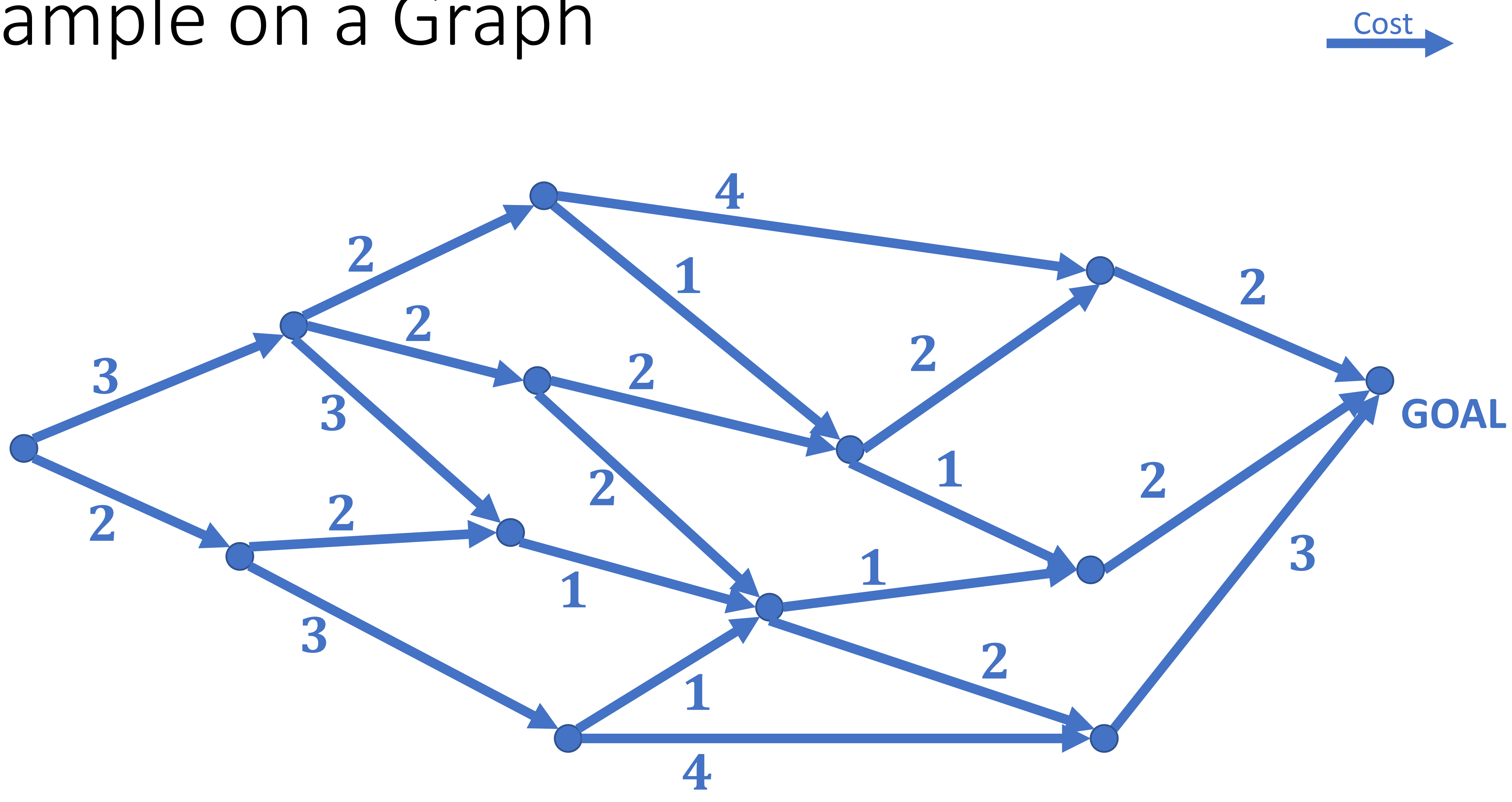
- Suppose $\overline{b_1 d}, \overline{b_2 d}, \overline{b_3 d}$ are optimal
- Then,

$$J_{ad}^* = \min \begin{Bmatrix} \tilde{J}_{ab_1} + J_{b_1 d}^*, \\ \tilde{J}_{ab_2} + J_{b_2 d}^*, \\ \tilde{J}_{ab_3} + J_{b_3 d}^* \end{Bmatrix}$$

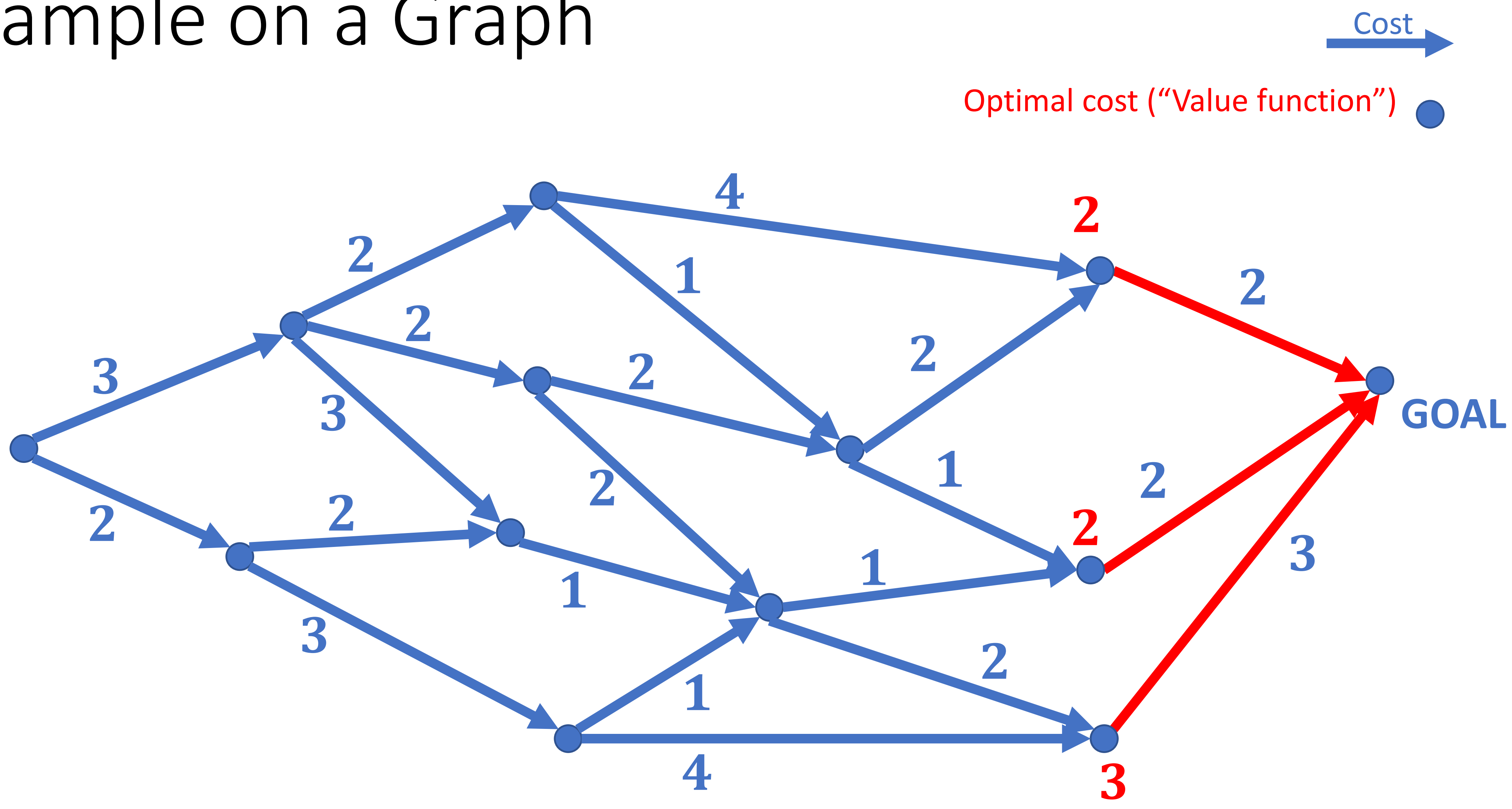


$$\tilde{J}_{ab_i} := \int_0^{t_{b_i}} c(x(t), u(t), t)\, dt$$

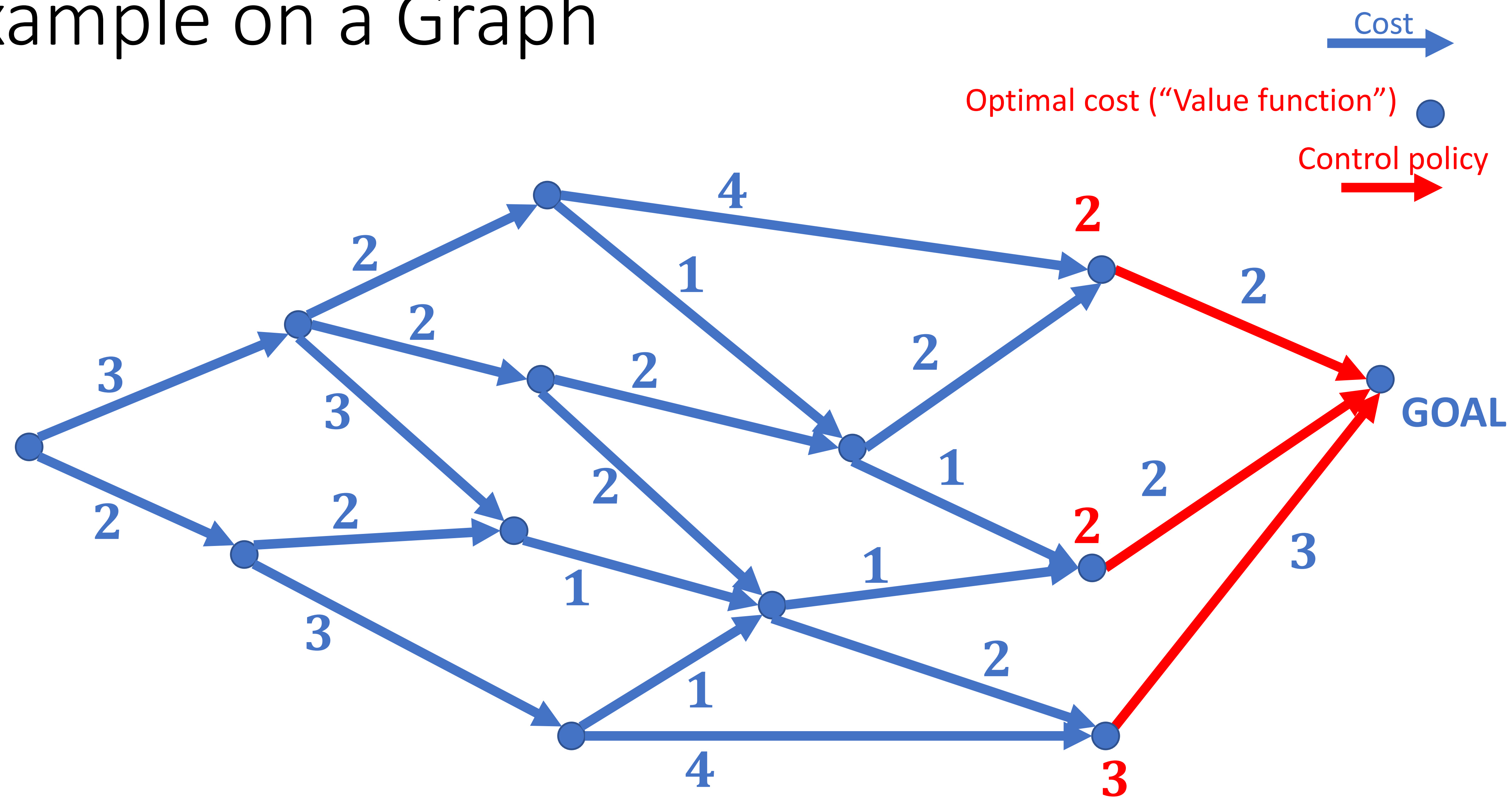
Example on a Graph



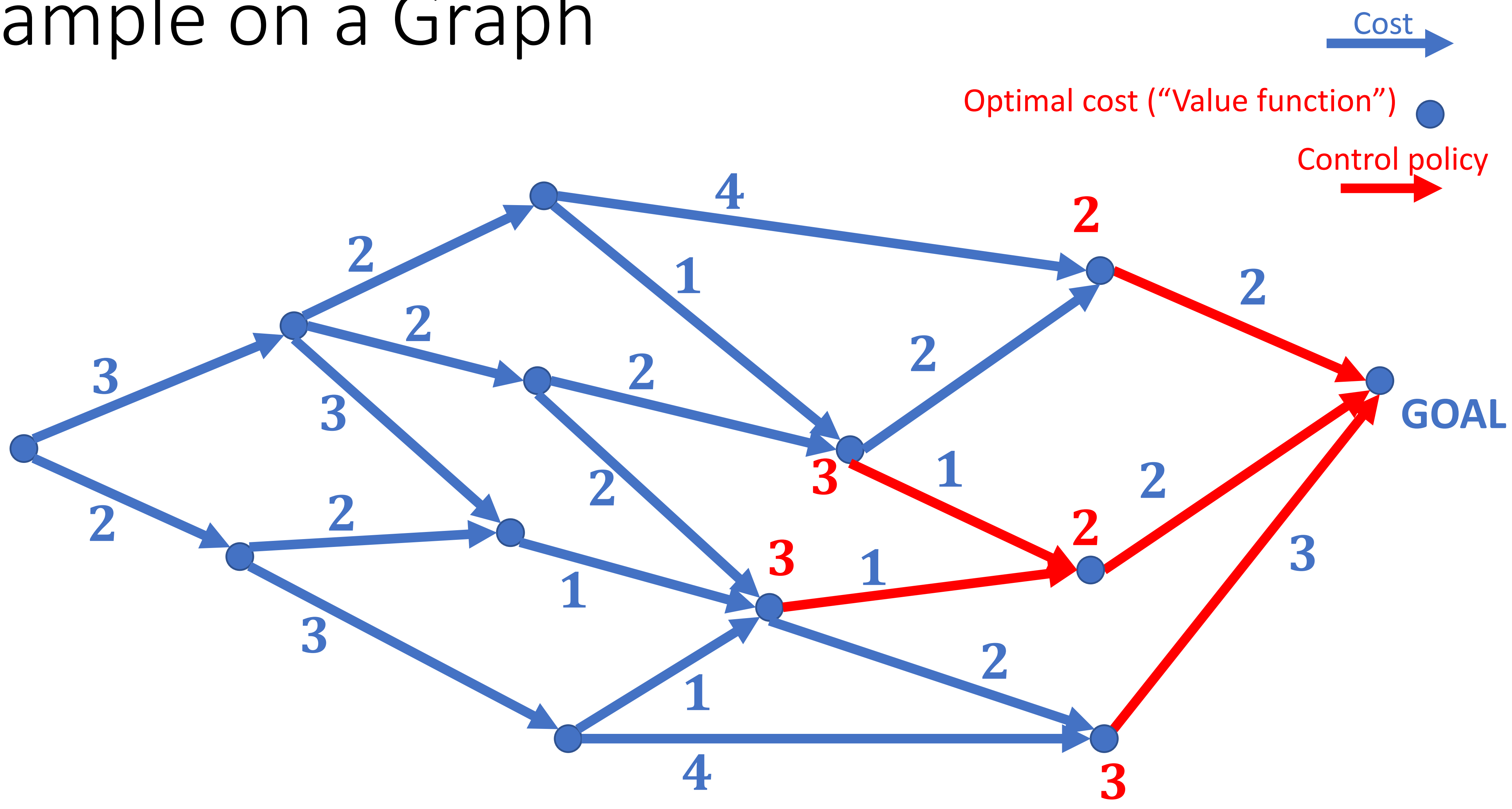
Example on a Graph



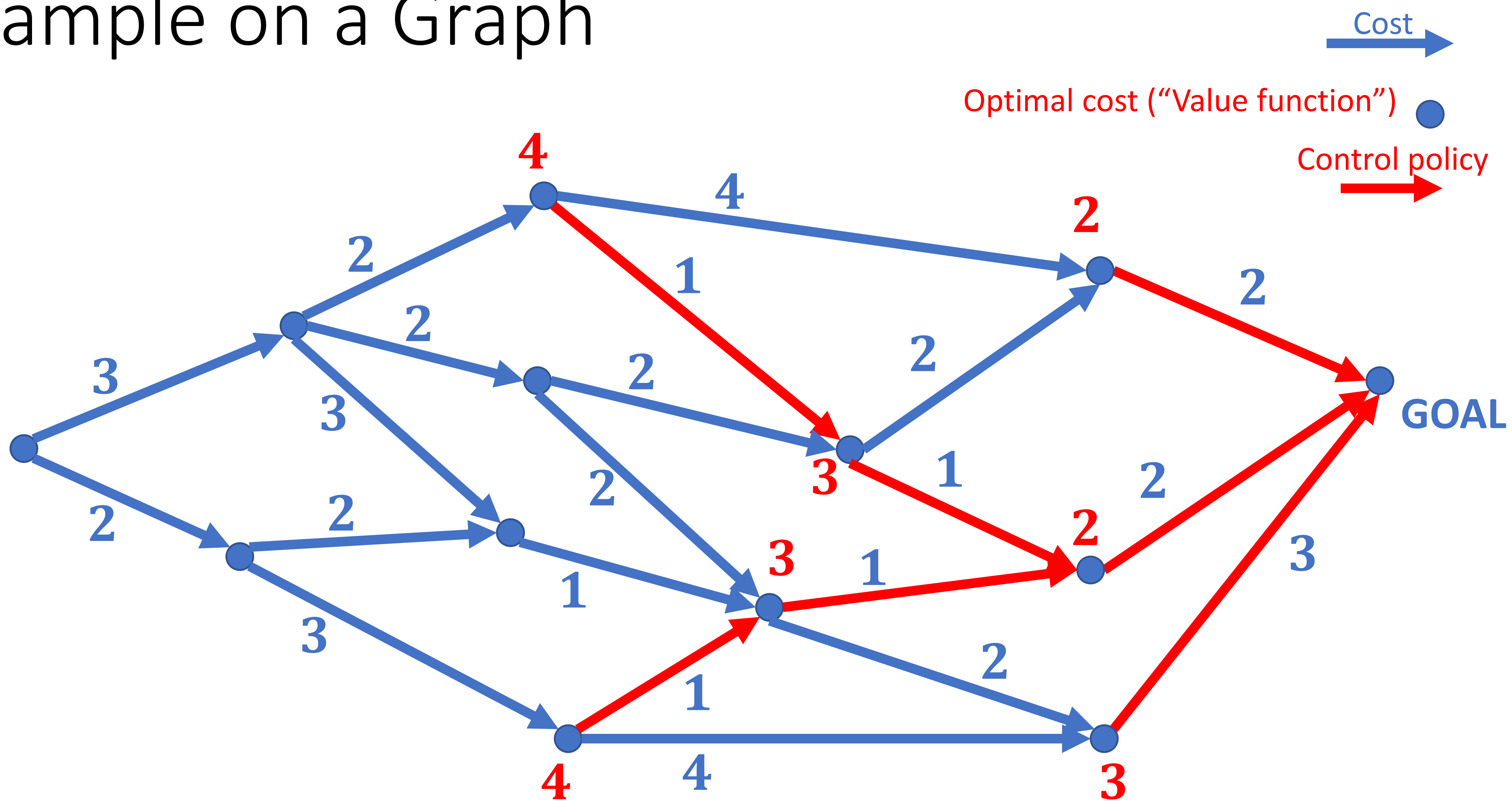
Example on a Graph



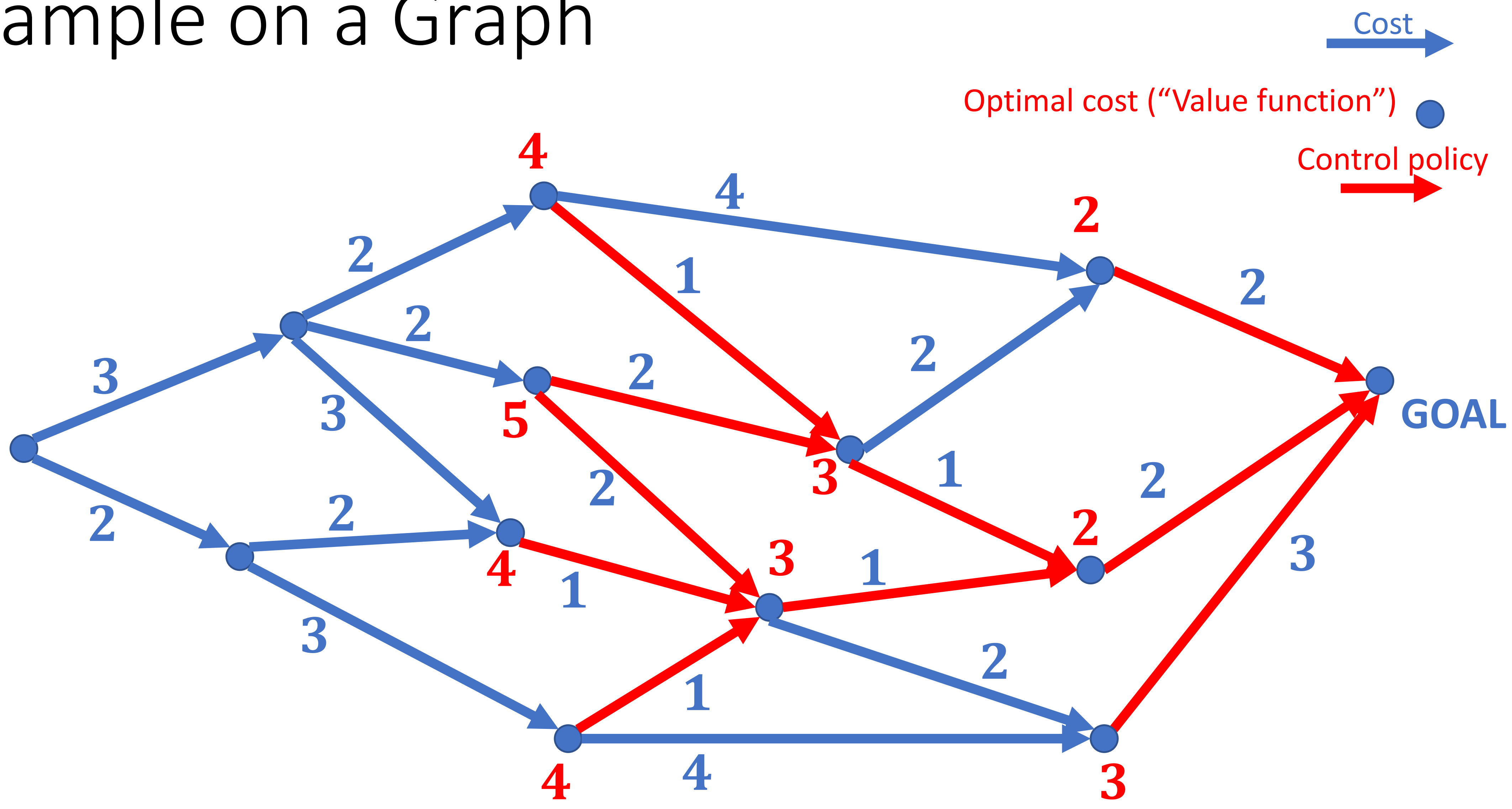
Example on a Graph



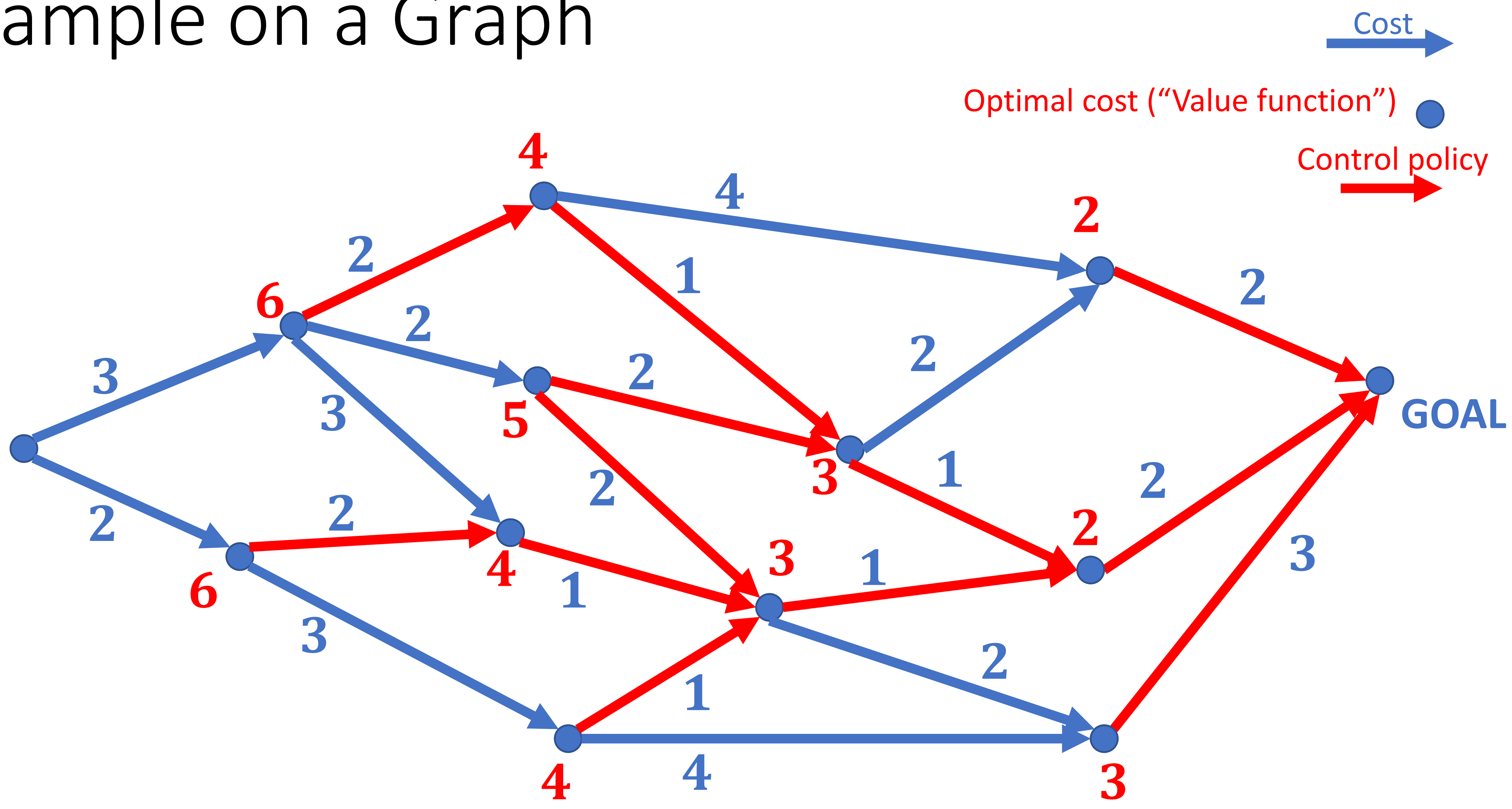
Example on a Graph



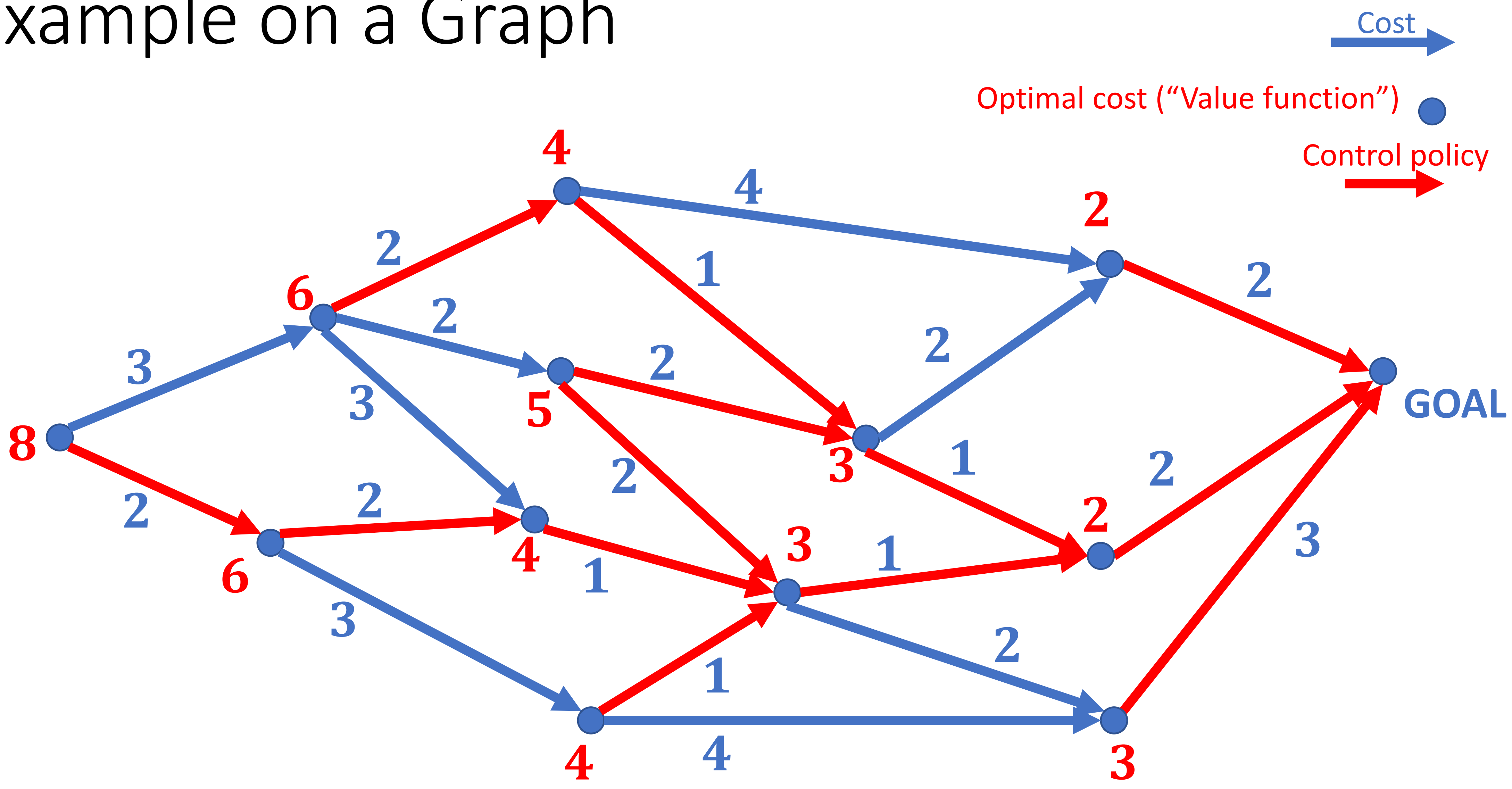
Example on a Graph



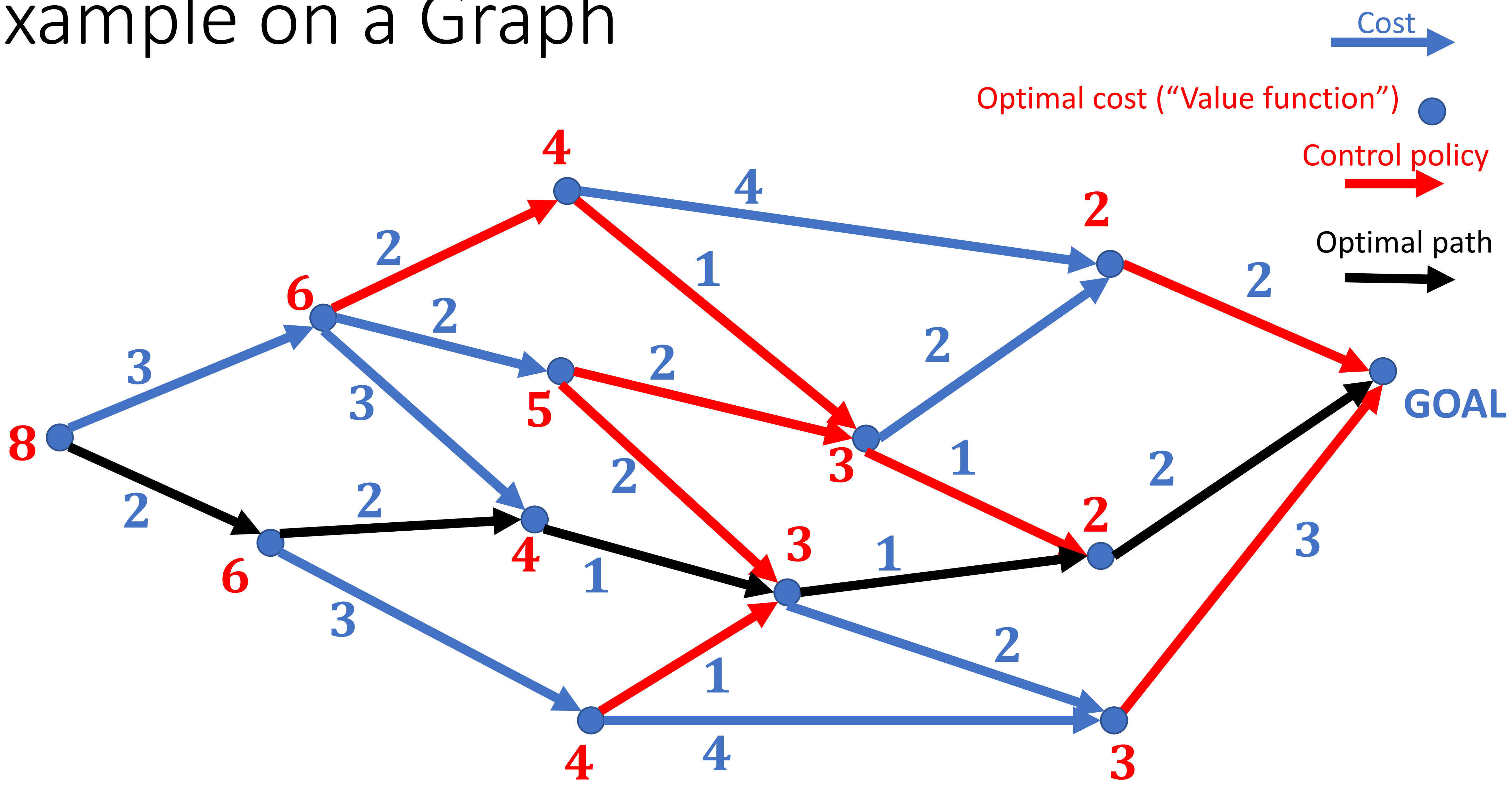
Example on a Graph



Example on a Graph



Example on a Graph

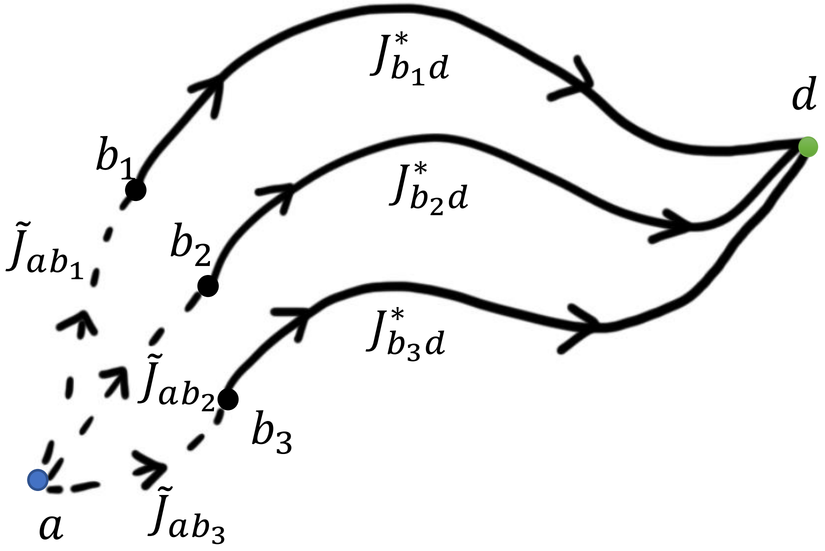


Dynamic programming: discrete time

- Discrete time model: $x_{k+1} = f_d(x_k, u_k),\ u_k \in U(x_k)$
  - May be obtained through discretizing continuous time model (eg. Forward Euler: $x_{k+1} = x_k + \Delta t f(x_k, u_k)$)

- Optimal cost: $J(x_0) = l(x_N) + \sum_{k=0}^{N-1} c(x_k, u_k)$

- Strategy: start at $k = N$ and work backwards to obtain $J_k(x)$
  - $J_N(x_N) = h_N(x_N)$
  - $J_k(x_k) = \min_{u_k \in U(x_k)} \{c_k(x_k, u_k) + J_{k+1}(f(x_k, u_k))\}$



Example: linear quadratic regulator (LQR)

- Linear discrete time system: $x_{k+1} = Ax_k + Bu_k$
- Quadratic cost: $J(x_0) = \frac{1}{2} x_N^\top L x_N + \frac{1}{2} \sum_{k=0}^{N-1} \left[x_k^\top Q x_k + u_k^\top R u_k \right]$
  - L, Q, R are symmetric positive semidefinite
- Start at $k = N$:

$$J_N^*(x_N) = \frac{1}{2} x_N^\top L x_N$$

- Apply dynamic programming principle:

$$J_{N-1}(x_{N-1}) = \min_{u_{N-1}} \frac{1}{2} \{ x_{N-1}^\top Q x_{N-1} + u_{N-1}^\top R u_{N-1} + x_N^\top L x_N \}$$

$$= \min_{u_{N-1}} \frac{1}{2} \{ x_{N-1}^\top Q x_{N-1} + u_{N-1}^\top R u_{N-1} + (Ax_{N-1} + Bu_{N-1})^\top L (Ax_{N-1} + Bu_{N-1}) \}$$

  - Simplify and find a pattern

Example: linear quadratic regulator (LQR)

- From previous slide:

$$J_{N-1}(x_{N-1}) = \min_{u_{N-1}} \frac{1}{2}\{x_{N-1}^\top Q x_{N-1} + u_{N-1}^\top R u_{N-1} + (Ax_{N-1} + Bu_{N-1})^\top L(Ax_{N-1} + Bu_{N-1})\}$$

  - Decision variable: u_{N-1}

- Take derivatives to find minimum:

$$\frac{\partial^2 J_{N-1}(x_{N-1})}{\partial u_{N-1}^2} = R + B^\top LB \geq 0$$

  - Positive semidefinite
  - First order condition is sufficient

$$\frac{\partial J_{N-1}(x_{N-1})}{\partial u_{N-1}} = Ru_{N-1} + B^\top L(Ax_{N-1} + Bu_{N-1})$$

  - Set to zero to obtain u_{N-1}^* ➡ $\boxed{u_{N-1}^* = Fx_{N-1}, \text{ where } F = -(R + B^\top LB)^{-1}B^\top LA}$
  - Plug in u_{N-1}^* into $J_{N-1}(x_{N-1})$ ➡ $\boxed{J_{N-1}(x_{N-1}) = \frac{1}{2}x_{N-1}^\top P x_{N-1}, \text{ where } P = Q + F^\top RF + (A + BF)^\top L(A + BF)}$

Example: linear quadratic regulator (LQR)

- Set derivative to zero to obtain control:

$$Ru_{N-1} + B^\top L(Ax_{N-1} + Bu_{N-1}) = 0$$
$$Ru_{N-1} + B^\top LAx_{N-1} + B^\top LBu_{N-1} = 0$$
$$(R + B^\top LB)u_{N-1} + B^\top LAx_{N-1} = 0$$

  - $u_{N-1}^* = Fx_{N-1}$, where $F = -(R + B^\top LB)^{-1}B^\top LA$

- Plug u_{N-1}^* into for J_{N-1}

$$J_{N-1}(x_{N-1}) = \frac{1}{2}\{x_{N-1}^\top Qx_{N-1} + u_{N-1}^{*\top}Ru_{N-1}^* + (Ax_{N-1} + Bu_{N-1}^*)^\top L(Ax_{N-1} + Bu_{N-1}^*)\}$$
$$J_{N-1}(x_{N-1}) = \frac{1}{2}\{x_{N-1}^\top Qx_{N-1} + x_{N-1}^\top F^\top RFx_{N-1} + (Ax_{N-1} + BFx_{N-1})^\top L(Ax_{N-1} + BFx_{N-1})\}$$
$$J_{N-1}(x_{N-1}) = \frac{1}{2}x_{N-1}^\top\big(Q + F^\top RF + (A + BF)^\top L(A + BF)\big)x_{N-1}$$

  - $J_{N-1}(x_{N-1}) = \frac{1}{2}x_{N-1}^\top Px_{N-1}$, where $P = Q + F^\top RF + (A + BF)^\top L(A + BF)$

Example: linear quadratic regulator (LQR)

- From previous slide:

$$J_{N-1}(x_{N-1}) = \min_{u_{N-1}} \frac{1}{2}\{x_{N-1}^\top Q x_{N-1} + u_{N-1}^\top R u_{N-1} + (Ax_{N-1} + Bu_{N-1})^\top L (Ax_{N-1} + Bu_{N-1})\}$$

  - Decision variable: u_{N-1}

- Take derivatives to find minimum:

$$\frac{\partial^2 J_{N-1}(x_{N-1})}{\partial u_{N-1}^2} = R + B^\top L B \geq 0$$

- Positive semidefinite
- First order condition is sufficient

$$\frac{\partial J_{N-1}(x_{N-1})}{\partial u_{N-1}} = R u_{N-1} + B^\top L (Ax_{N-1} + Bu_{N-1})$$

- Set to zero to obtain u_{N-1}^* ➡ $\boxed{u_{N-1}^* = F x_{N-1}, \text{ where } F = -(R + B^\top L B)^{-1} B^\top L A}$
- Plug in u_{N-1}^* into $J_{N-1}(x_{N-1})$ ➡ $\boxed{J_{N-1}(x_{N-1}) = \frac{1}{2} x_{N-1}^\top P x_{N-1}, \text{ where } P = Q + F^\top R F + (A + BF)^\top L (A + BF)}$

Example: linear quadratic regulator (LQR)

- Look for a pattern
- $J_N(x_N) = \frac{1}{2} x_N^\top L x_N$
  - $u_{N-1}^* = F x_{N-1}$, where $F = -(R + B^\top L B)^{-1} B^\top L A$
  - $J_{N-1}(x_{N-1}) = \frac{1}{2} x_{N-1}^\top P x_{N-1}$, where $P = Q + F^\top R F + (A + BF)^\top L (A + BF)$

Example: linear quadratic regulator (LQR)

- Look for a pattern
- $J_N(x_N) = \frac{1}{2} x_N^\top L x_N$
  - $u_{N-1}^* = F x_{N-1}$, where $F = -(R + B^\top L B)^{-1} B^\top L A$
  - $J_{N-1}(x_{N-1}) = \frac{1}{2} x_{N-1}^\top P x_{N-1}$, where $P = Q + F^\top R F + (A + BF)^\top L (A + BF)$

Example: linear quadratic regulator (LQR)

- Look for a pattern
- $J_N(x_N) = \frac{1}{2} x_N^\top P_N x_N$, where $P_N = L$
  - $u_{N-1}^* = F_{N-1} x_{N-1}$, where $F_{N-1} = -(R + B^\top P_N B)^{-1} B^\top P_N A$
  - $J_{N-1}(x_{N-1}) = \frac{1}{2} x_{N-1}^\top P_{N-1} x_{N-1}$, where $P_{N-1} = Q + F_{N-1}^\top R F_{N-1} + (A + B F_{N-1})^\top P_N (A + B F_{N-1})$

Example: linear quadratic regulator (LQR)

- Proceed by induction
- $J_N(x_N) = \frac{1}{2} x_N^\top P_N x_N$, where $P_N = L$
  - $u_k^* = F_k x_k$, where $F_k = -(R + B^\top P_{k+1} B)^{-1} B^\top P_{k+1} A$
  - $J_{N-1}(x_k) = \frac{1}{2} x_k^\top P_k x_k$, where $P_k = Q + F_k^\top R F_k + (A + B F_k)^\top P_{k+1} (A + B F_k)$

Example: linear quadratic regulator (LQR)

- Proceed by induction
- $J_N(x_N) = \frac{1}{2} x_N^\top P_N x_N$, where $P_N = L$
  - $u_k^* = F_k x_k$, where $F_k = -(R + B^\top P_{k+1} B)^{-1} B^\top P_{k+1} A$
  - $J_{N-1}(x_k) = \frac{1}{2} x_k^\top P_k x_k$, where $P_k = Q + F_k^\top R F_k + (A + BF_k)^\top P_{k+1} (A + BF_k)$

- Eventually,
  - $J_0(x_0) = \frac{1}{2} x_0^\top P_0 x_0$

Comments

- No control constraint

- What if there is control constraint?
  - Let controllers saturate
  - Explicitly treat it in the minimization of J

- MATLAB commands
  - Discrete time: `dlqr`
  - Continuous time: `lqr`