Dynamic Programming

CMPT 419/983

Mo Chen

SFU Computing Science

7/10/2019

Dynamic Programming Outline

- Continuous time Hamilton-Jacobi equation
- Continuous linear quadratic regulator

Optimal Control: Types of Solutions

$$\underset{u(\cdot)}{\text{minimize}}\ l\big(T, x(T)\big) + \int_0^T c(x(t), u(t), t)dt$$

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

- Open-loop control
  - Scalable, but errors will add up
- Closed-loop control
  - Find $u(t, x)$ for $t \in [0, T]$, $x \in \mathbb{R}^n$
  - 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

Final cost

Running cost

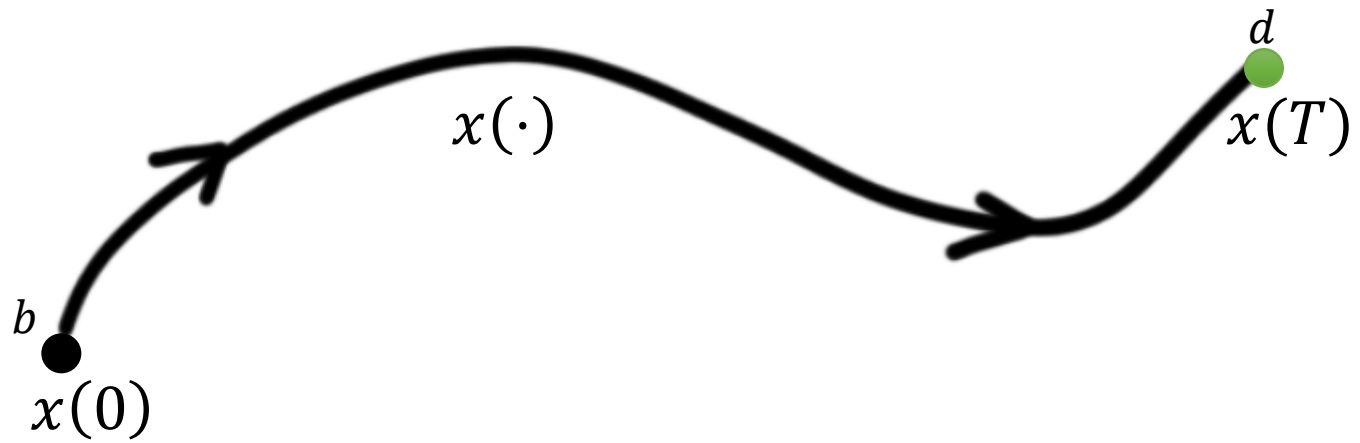
$$\underset{u(\cdot)}{\text{minimize}} \; l\big(T, x(T)\big) + \int_0^T c\big(x(t), u(t), t\big)dt$$

Cost functional, $J\big(x(\cdot), u(\cdot)\big)$

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

Dynamic model

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

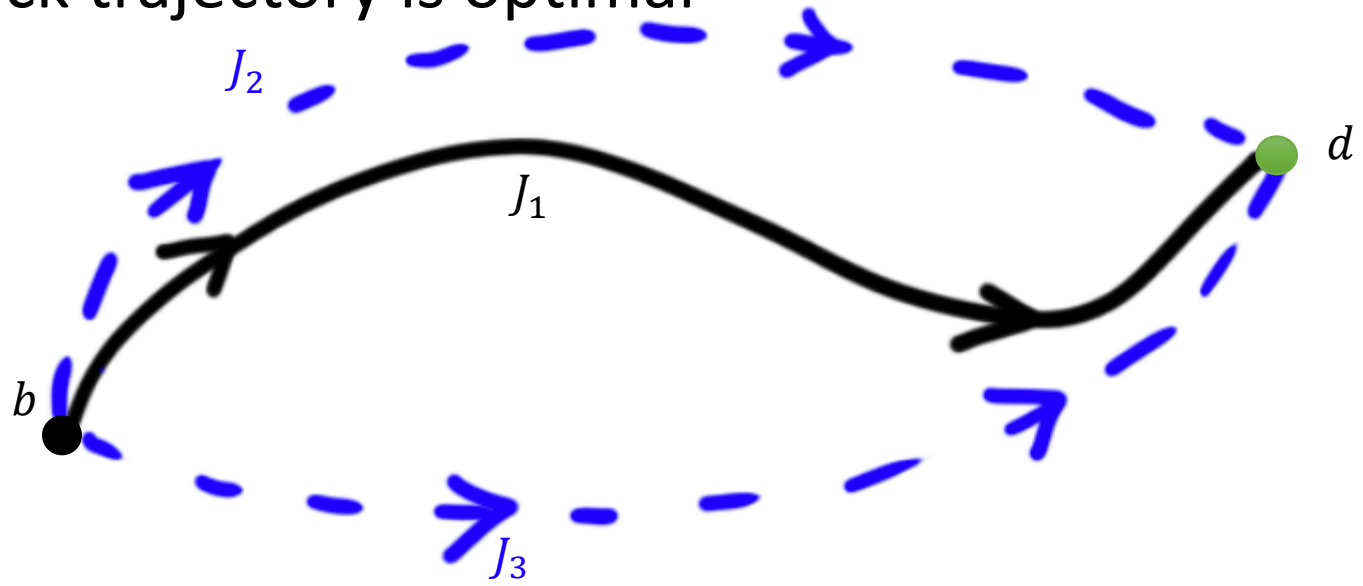


Dynamic Programming

- 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



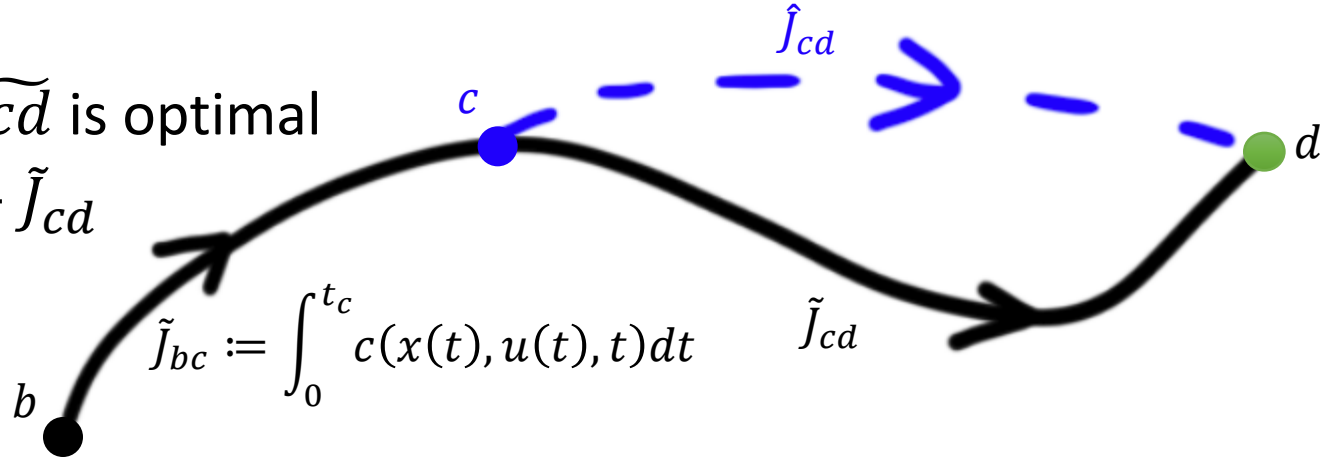
  - Then $J_1 \leq J_2, J_3$
  - Let $J_{bd}^* = J_1$

Principle of Optimality

- Optimal cost: J^*_{bd}

Principle of Optimality

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

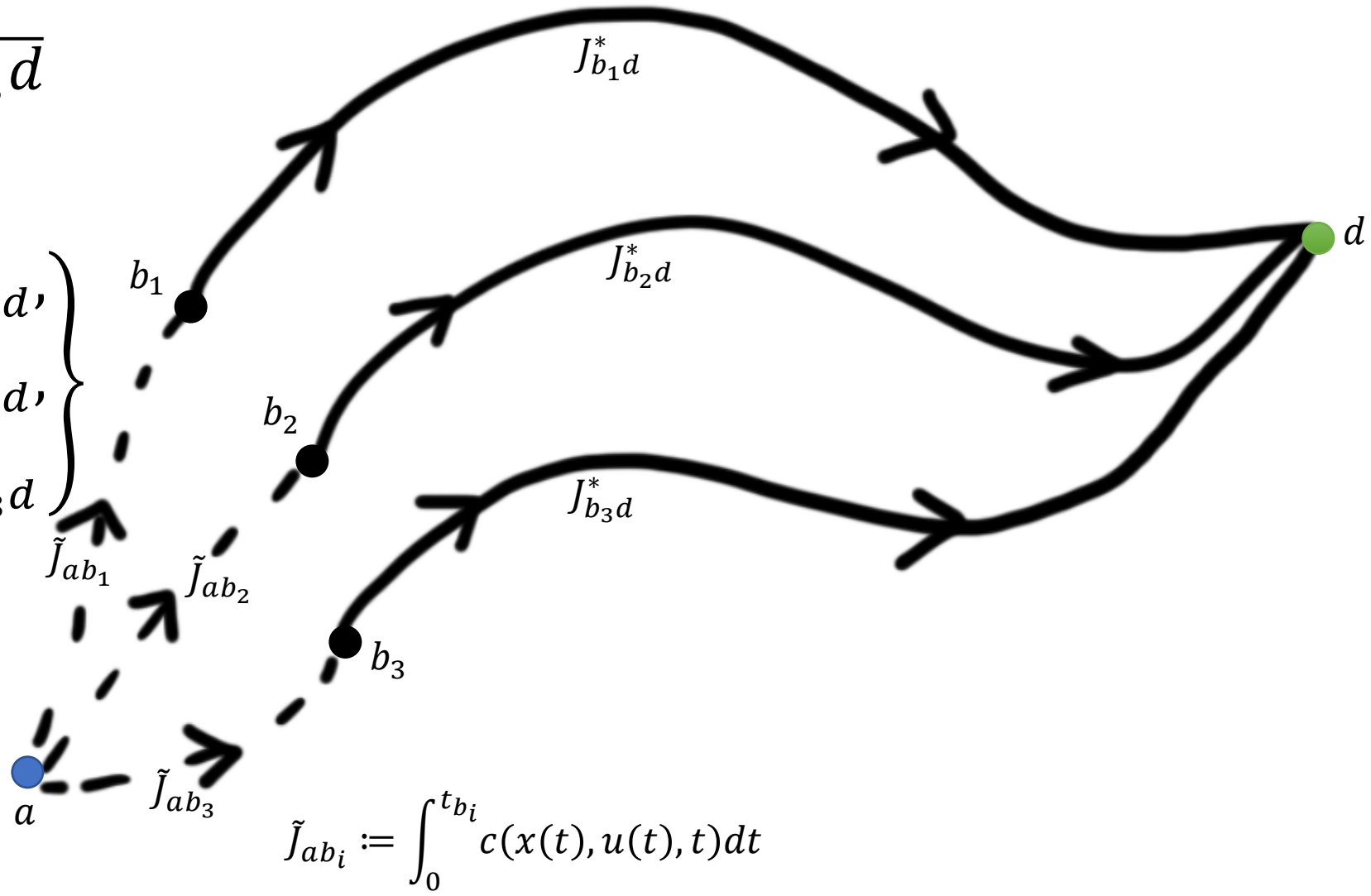


- Proof:
  - Suppose not, then there is some other path from c to d with cost $\hat{J}_{cd}$ such that $\hat{J}_{cd} < \tilde{J}_{cd}$
  - This means $J^*_{bd} = \tilde{J}_{bc} + \tilde{J}_{cd} > \tilde{J}_{bc} + \hat{J}_{cd}$
  - Therefore, the original trajectory $\overline{bd}$ is not optimal ← contradiction!

Applying the Principle of Optimality

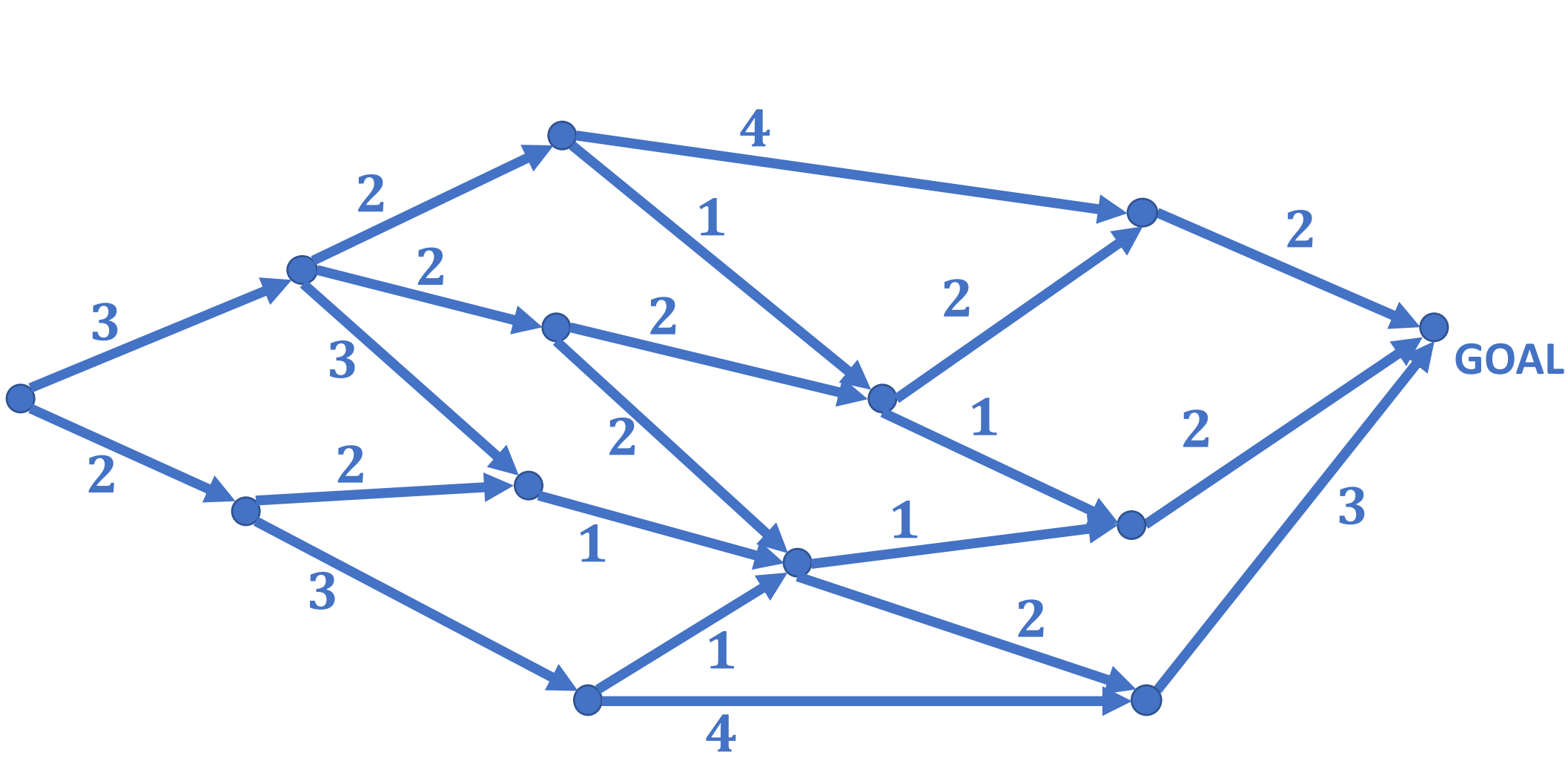
- Suppose $\overline{b_1 d}, \overline{b_2 d}, \overline{b_3 d}$ are optimal
- Then,

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

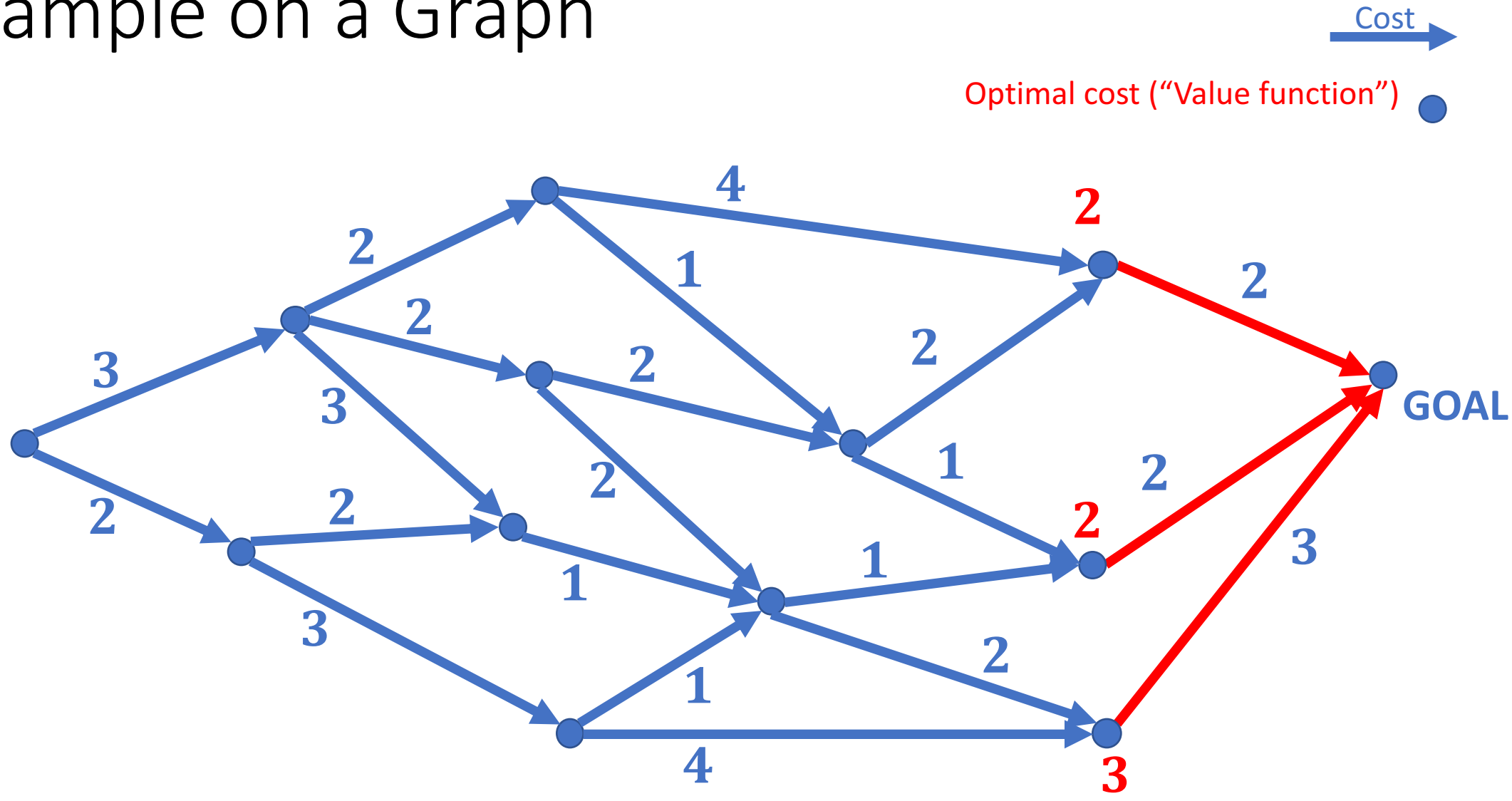


$$\tilde{J}_{ab_i} \coloneqq \int_0^{t_{b_i}} c(x(t), u(t), t) dt$$

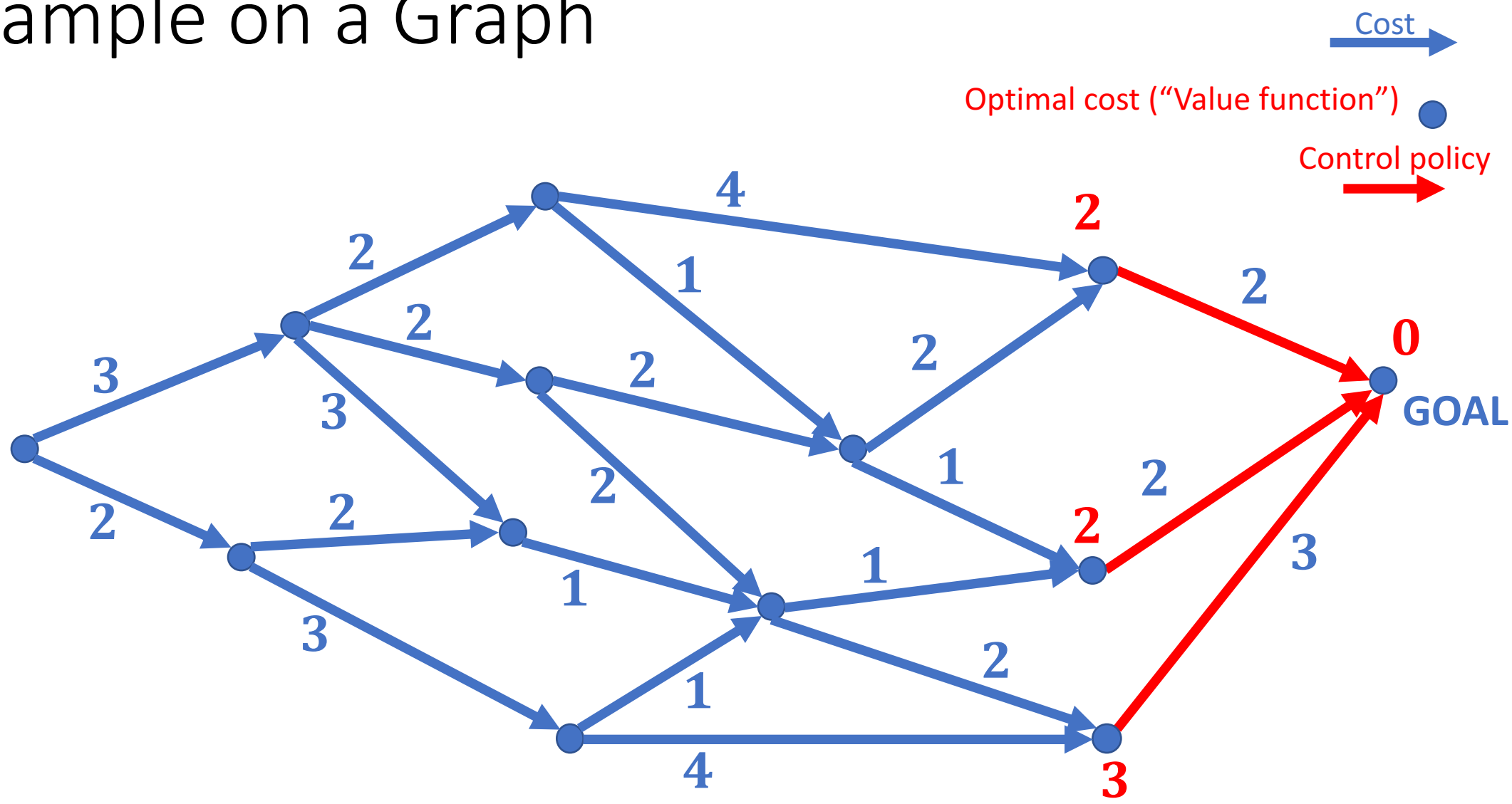
Example on a Graph



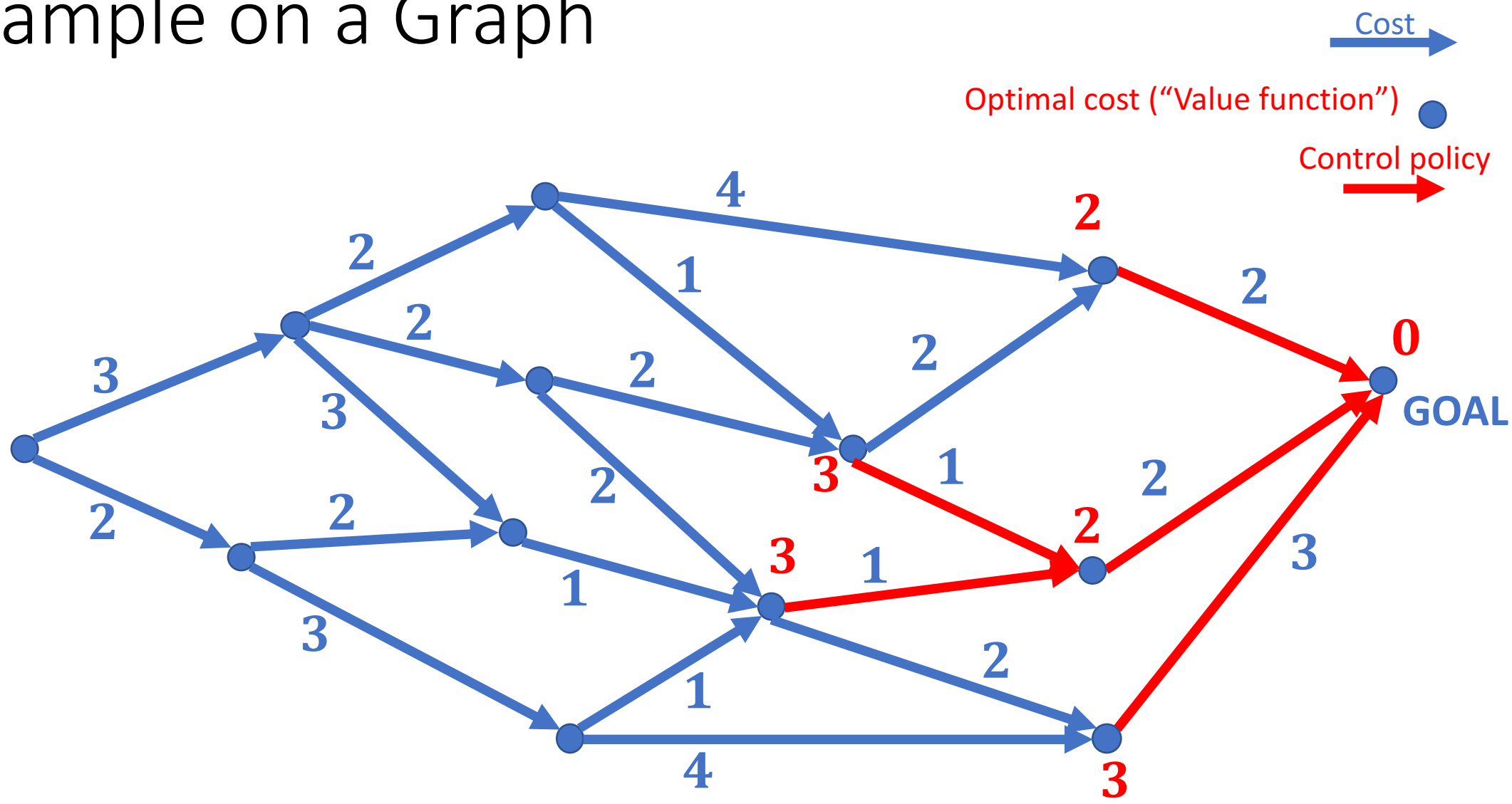
Example on a Graph



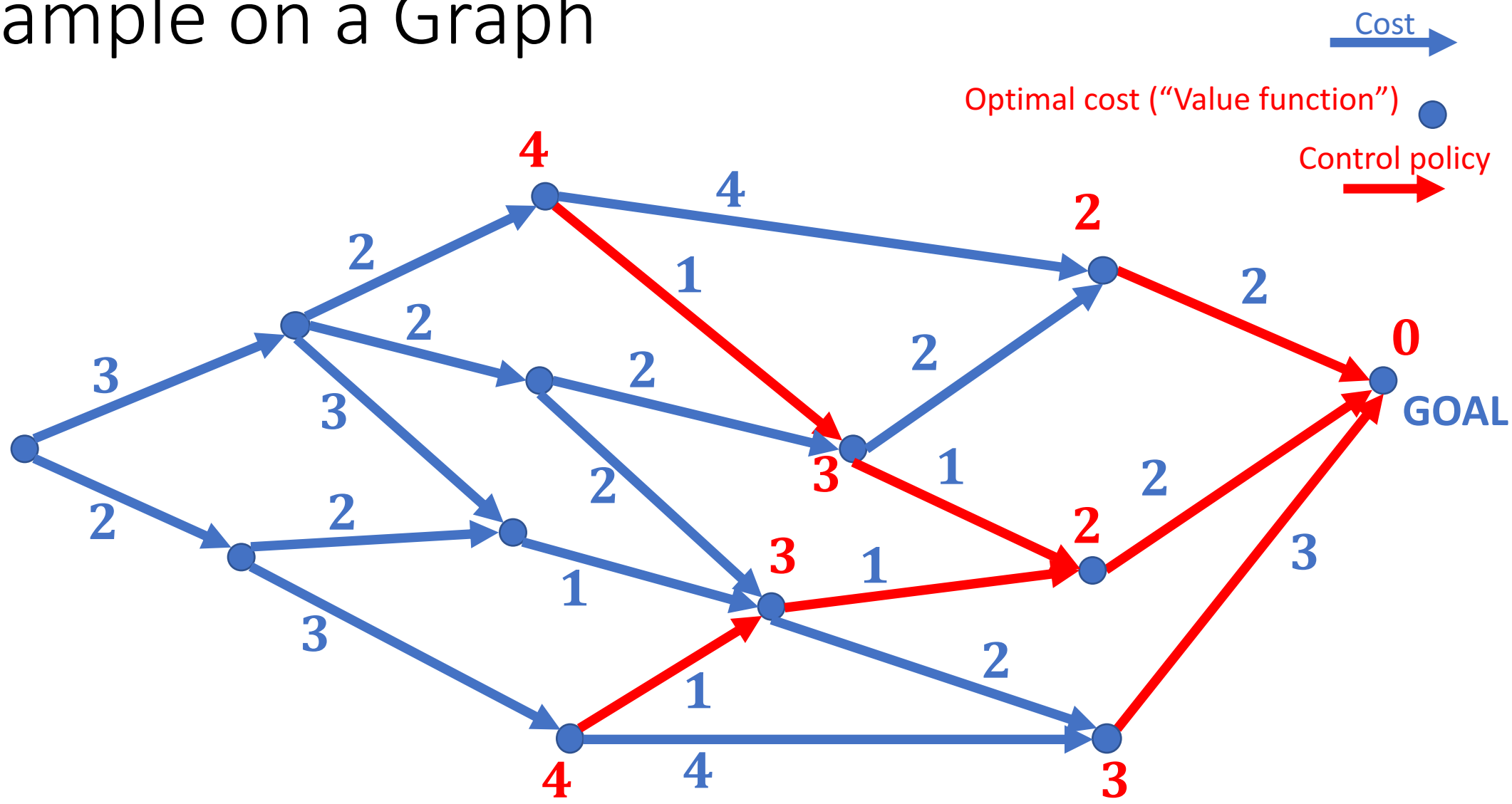
Example on a Graph



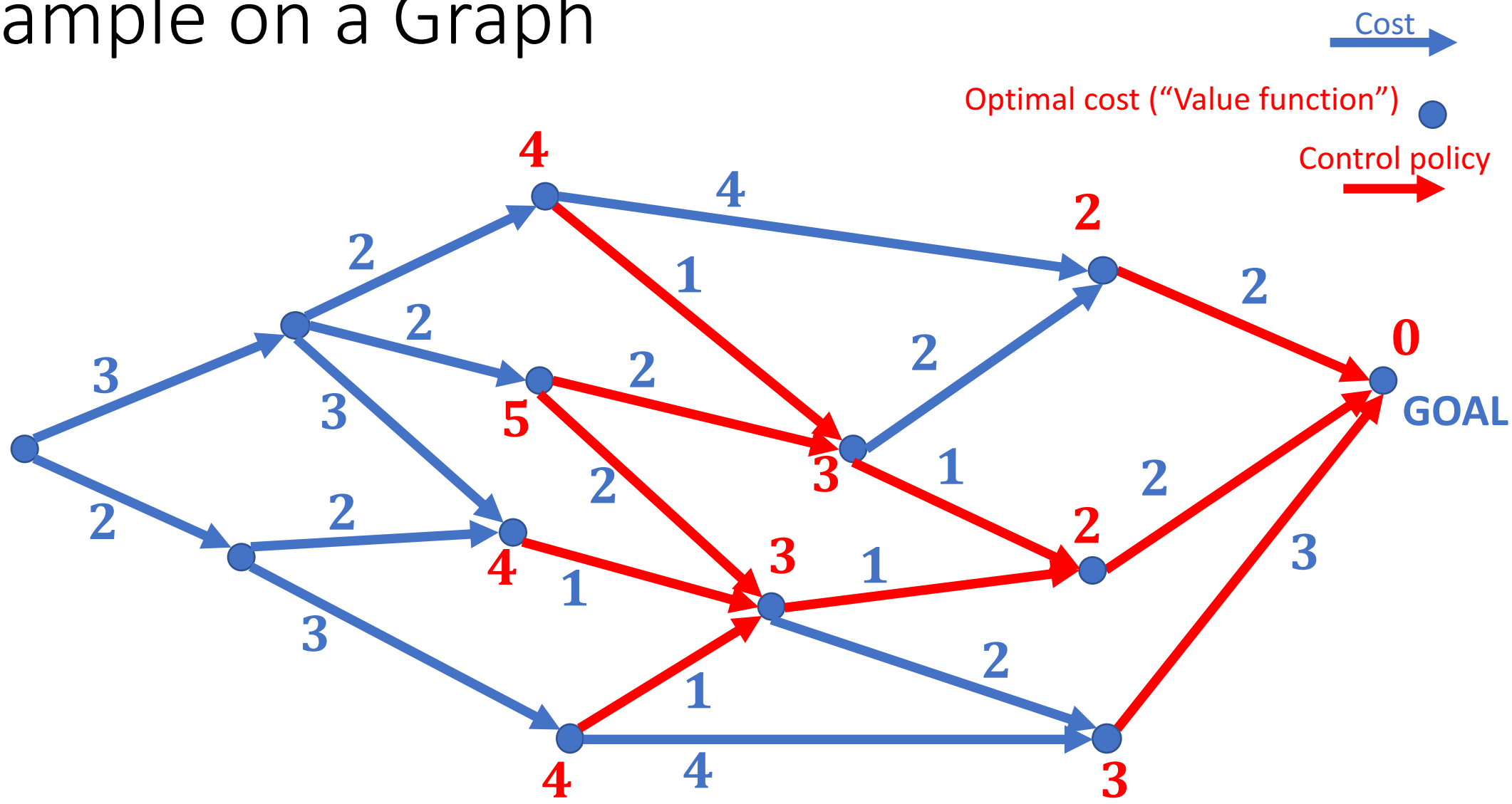
Example on a Graph



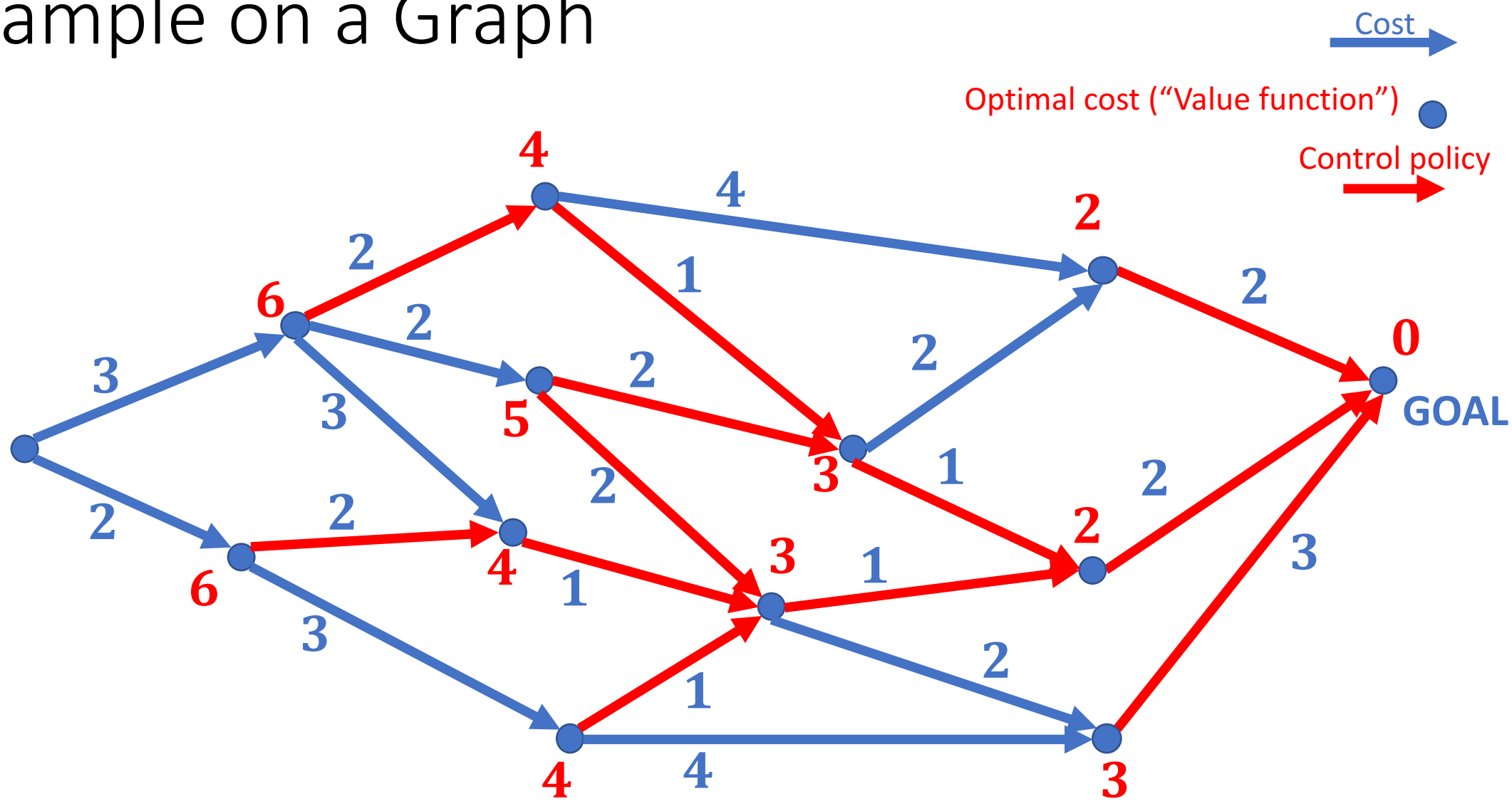
Example on a Graph



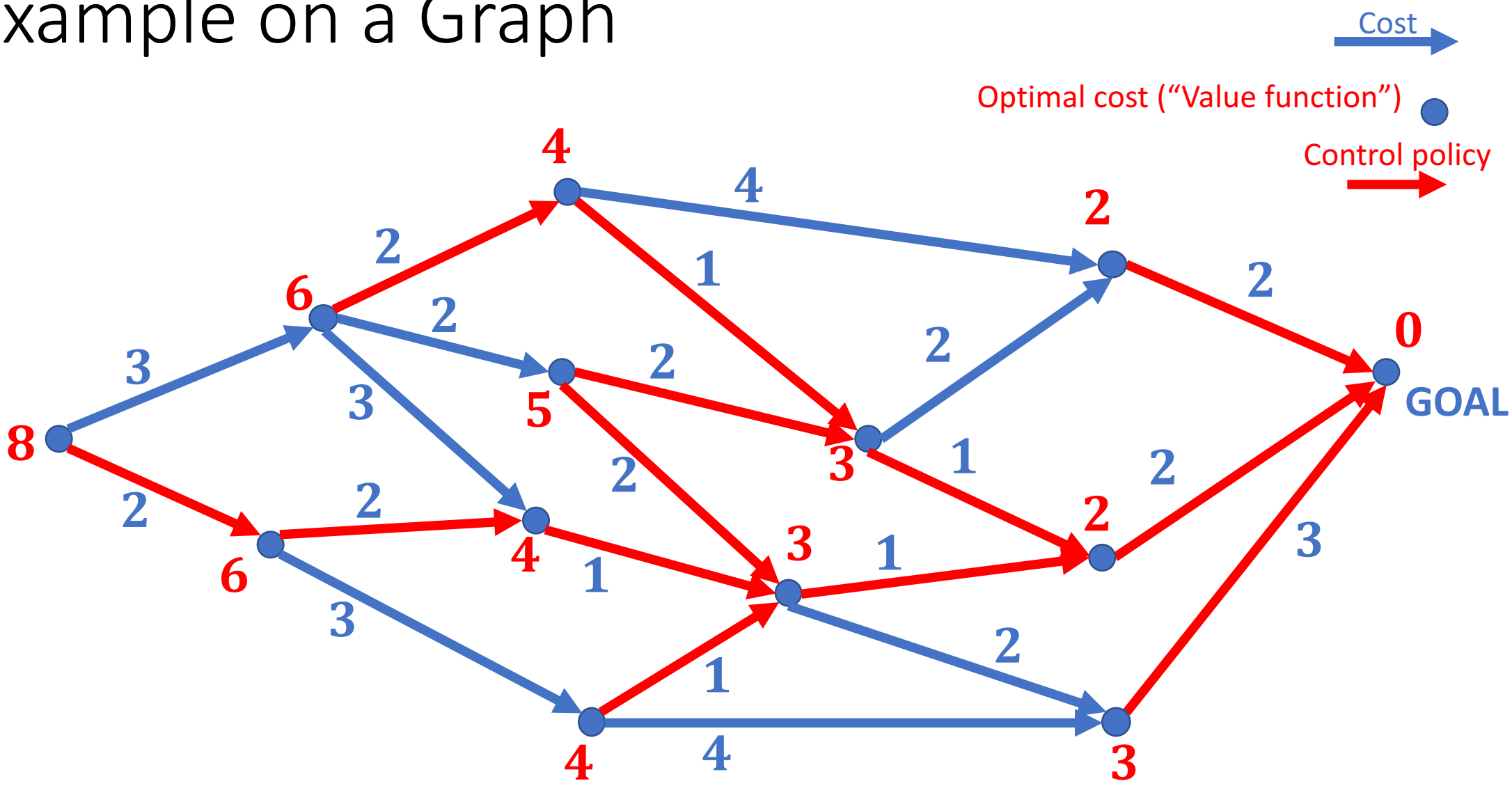
Example on a Graph



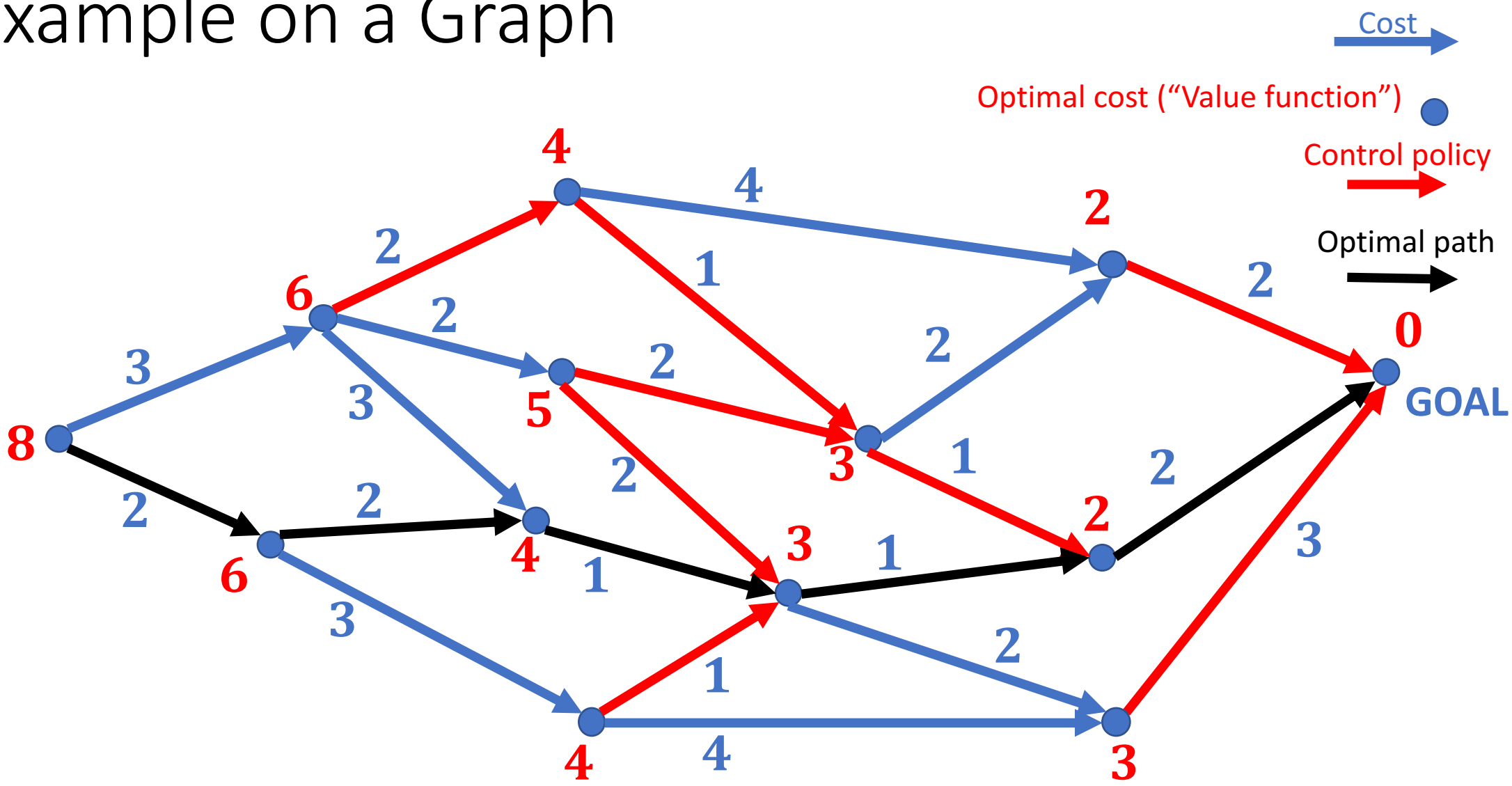
Example on a Graph



Example on a Graph



Example on a Graph



Dynamic Programming: Continuous Time

Final cost

Running cost

$$\underset{u(\cdot)}{\text{minimize}}\ l\big(T, x(T)\big) + \int_0^T c\big(x(t), u(t)\big)dt$$

Cost functional, $J\big(x(\cdot), u(\cdot)\big)$

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

Dynamic model

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

- Let $J(t, x(t)) = l\big(T, x(T)\big) + \int_t^T c\big(x(t), u(t)\big)dt$
  - $V(0, x(0)) = \min_{u(\cdot)} J\big(0, x(0)\big)$ is what we want

- Strategy:
  - make a "discrete time" argument with Δt
  - Let $\Delta t \to 0$

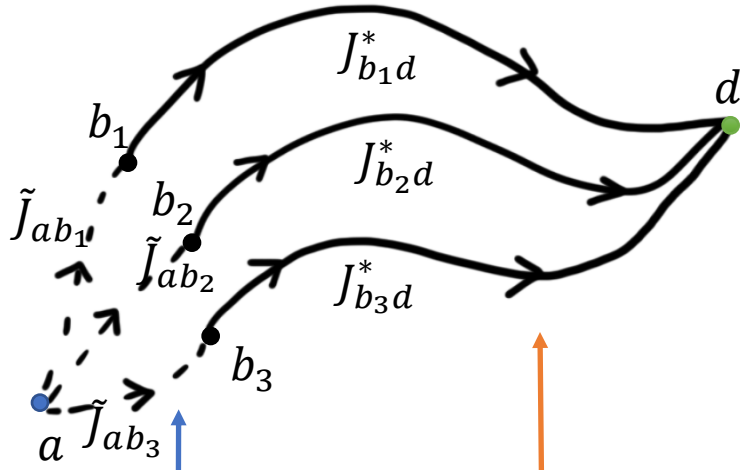
Dynamic Programming: Continuous Time

- Let $J(t,x(t)) = \int_t^T c\big(x(s),u(s)\big)ds + l\big(x(T)\big)$ "Cost to go"

$$V\big(t,x(t)\big) = \min_{u_{[t,T]}(\cdot)} \left[\int_t^T c\big(x(s),u(s)\big)ds + l\big(x(T)\big)\right]$$

Write out time interval explicitly for clarity

"Value function", "$J^*\big(t,x(t)\big)$"



- Dynamic programming principle:

$$V\big(t,x(t)\big) = \min_{u_{[t,t+\delta]}(\cdot)} \left[\int_t^{t+\delta} c\big(x(s),u(s)\big)ds + V(t+\delta, x(t+\delta))\right]$$

  - Approximate integral and Taylor expand $V\big(t+\delta, x(t+\delta)\big)$
  - Derive Hamilton-Jacobi partial differential equation (HJ PDE)

Dynamic Programming: Continuous Time

- Approximations for small δ:

$$V(t,x(t)) = \min_{u_{[t,t+\delta]}(\cdot)} \left[\int_t^{t+\delta} c(x(s),u(s))ds + V(t+\delta, \overbrace{x(t+\delta)}^{x(t)+\delta f(x,u)}) \right]$$

$$c(x(t),u(t))\delta$$

$$V(t,x(t)) + \left(\frac{\partial V}{\partial x}\right)^{\top} \delta f(x(t),u(t)) + \frac{\partial V}{\partial t}\delta$$

  - Omit t dependence…

$$V(t,x) = \min_u \left[c(x,u)\delta + V(t,x) + \left(\frac{\partial V}{\partial x}\right)^{\top} \delta f(x,u) + \frac{\partial V}{\partial t}\delta \right]$$

Assume constant $u_{[t,t+\delta]}$ → Optimization over a vector, not a function!

  - $V(t,x)$ does not depend on u

$$V(t,x) = V(t,x) + \min_u \left[c(x,u)\delta + \left(\frac{\partial V}{\partial x}\right)^{\top} \delta f(x,u) + \frac{\partial V}{\partial t}\delta \right]$$

Dynamic Programming: Continuous Time

- Approximations for small δ:

$$V(t,x(t)) = \min_{u_{[t,t+\delta]}(\cdot)} \left[\int_t^{t+\delta} c(x(s),u(s))ds + V(t+\delta, \overbrace{x(t+\delta)}^{x(t)+\delta f(x,u)}) \right]$$

$$c(x(t),u(t))\delta$$

$$V(t,x(t)) + \left(\frac{\partial V}{\partial x}\right)^{\top} \delta f(x(t),u(t)) + \frac{\partial V}{\partial t}\delta$$

  - Omit t dependence…

$$V(t,x) = \min_u \left[c(x,u)\delta + V(t,x) + \left(\frac{\partial V}{\partial x}\right)^{\top} \delta f(x,u) + \frac{\partial V}{\partial t}\delta \right]$$

Assume constant $u_{[t,t+\delta]}$ → Optimization over a vector, not a function!

  - $V(t,x)$ does not depend on u

$$\cancel{V(t,x)} = \cancel{V(t,x)} + \min_u \left[c(x,u)\delta + \left(\frac{\partial V}{\partial x}\right)^{\top} \delta f(x,u) + \frac{\partial V}{\partial t}\delta \right]$$

Dynamic Programming: Continuous Time

- Approximations for small δ:

$$V(t,x(t)) = \min_{u_{[t,t+\delta]}(\cdot)} \left[\int_t^{t+\delta} c(x(s),u(s))ds + V(t+\delta, \overbrace{x(t+\delta)}^{x(t)+\delta f(x,u)}) \right]$$

$$c(x(t),u(t))\delta$$

$$V(t,x(t)) + \left(\frac{\partial V}{\partial x}\right)^{\top} \delta f(x(t),u(t)) + \frac{\partial V}{\partial t}\delta$$

  - Omit t dependence…

$$V(t,x) = \min_{u} \left[c(x,u)\delta + V(t,x) + \left(\frac{\partial V}{\partial x}\right)^{\top} \delta f(x,u) + \frac{\partial V}{\partial t}\delta \right]$$

Assume constant $u_{[t,t+\delta]}$ → Optimization over a vector, not a function!

  - $V(t,x)$ does not depend on u

$$0 = \frac{\partial V}{\partial t} + \min_{u} \left[c(x,u) + \left(\frac{\partial V}{\partial x}\right)^{\top} f(x,u) \right]$$

Comments

- Hamilton-Jacobi partial differential equation

$$\frac{\partial V}{\partial t} + \min_u \left[c(x,u) + \left(\frac{\partial V}{\partial x}\right)^{\mathsf{T}} f(x,u) \right] = 0, \qquad V(T,x) = l(x)$$

- Terminology:
  - Pre-Hamiltonian: $H(x,u,\lambda) = c(x,u) + \lambda^{\mathsf{T}} f(x,u)$
  - Hamiltonian: $H^*(x,\lambda) = c(x,u^*) + \lambda^{\mathsf{T}} f(x,u^*)$

$$\Rightarrow \frac{\partial V}{\partial t} + H^*(x,\lambda) = 0$$

Comments

- Hamilton-Jacobi partial differential equation

$$\frac{\partial V}{\partial t} + \min_u \left[c(x,u) + \left(\frac{\partial V}{\partial x}\right)^\top f(x,u) \right] = 0, \qquad V(T,x) = l(x)$$

- Minimization over u is typically easy
  - Most systems are control affine: $f(x,u)$ has the form $f(x) + g(x)u$
  - Control constraints are typically "box" constraints, e.g. $|u_i| \leq 1$

- PDE is solved on a grid
  - $x \in \mathbb{R}^n$ means $V(t,x)$ is computed on an $(n+1)$-dimensional grid

- $V(t,x)$is often not differentiable (or continuous)
  - Viscosity solutions
  - Lax Friedrichs numerical method