Dynamic programming

G. Ferrari Trecate

Dipartimento di Ingegneria Industriale e dell'Informazione
Università degli Studi di Pavia

Industrial Automation

Introduction

Dynamic Programming (DP) is a method for solving management problems composed by N consecutive decisions

Quantitative models for sequences of actions are automata

Def. An automaton is a discrete-time dynamical system

$$x_{k+1} = f(x_k, u_k) \qquad k = 0, 1, \dots \qquad \text{(DYN)}$$

where the state x_k and the input u_k verify constraints

$$x_k \in S, \quad u_k \in U(x_k) \subseteq C \qquad \text{(V)}$$

and S and C are finite sets. $U(x_k)$ is the (state-dependent) set of admissible inputs.

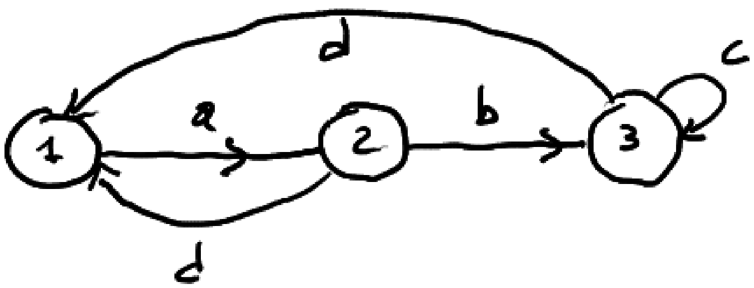
Example: bolt making machine

States : 1 = off , 2 = starting, 3 = on $\longrightarrow$ $S = \{1, 2, 3\}$

Input commands: a = turn on b = warm up c = bolt production

d = turn off $\longrightarrow$ $C = \{a, b, c, d\}$

Automaton



Graphic definition of f and $U(x)$

$k = 0, 1, 2, \ldots$: event number

$U(1) = \{a\}$, $U(2) = \{d, b\}$, $U(3) = \{c, d\}$

$f(x, u)$ given by the table

	a	b	c	d
1	2			
2		3		1
3			3	1

$\rightarrow$ empty entries correspond to unfeasible (state, input) pairs

Finite-time optimal control

Problem OC. For given

- $N \in \mathbb{N}$
- stage costs $g_k(x_k, u_k) \in \mathbb{R}$, $k = 0, \dots, N-1$
- final cost $g_N(x_N) \in \mathbb{R}$

solve the problem

$$J(x_0) = \min_{u_0, \dots, u_{N-1}} g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k)$$

subject to the constraints (DYN) and (V)

Notation

$$J(x_0) = \min_{u_0, \dots, u_{N-1}} g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k)$$

- $\mathcal{U}^*(x_0) = \{u_k^*\}_{k=0}^{N-1}$ optimal control variables, i.e. the minimizers
- The optimal state evolves according to

$$x_0 \to x_1^* = f(x_0, u_0^*) \to x_2^* = f(x_1^*, u_1^*) \to \dots$$

$\hookrightarrow \mathcal{X}^*(x_0) = \{x_k^*\}_{k=1}^{N}$

- N is the control horizon

Remarks

$$J(x_0) = \min_{u_0,\dots,u_{N-1}} g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k)$$

- Stage and final costs encode desired properties of the optimal decision sequence. For instance if

$$g_k(i, \bar{u}) \gg g_k(s, u), \quad \forall s \in S,\ s \neq i,\ \forall u \in U(s)$$

it is unlikely that $x_k^* = i$ and $u_k^* = \bar{u}$

- $V(x_0, \{u_k\}_{k=0}^{N-1}) = g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k)$ is the cost of a given input sequence starting from x_0

- u_k = consecutive decisions. $V(\cdot,\cdot)$ = criterion for taking optimal decisions (e.g. economical)

Properties of OC problems

Problem. How to compute a solution to OC in an efficient way?

Idea: break OC into a sequence of simpler sub-problems

Bellman's optimality principle. For a given $x_0 \in S$, let $u_0^*, \dots, u_{N-1}^*$ be optimal inputs and x_ℓ^* be the optimal state at time ℓ, $0 < \ell < N-1$.

Consider the optimization problem

$$J_\ell(x_\ell) = \min_{u_\ell, \dots, u_{N-1}} g_N(x_N) + \sum_{k=\ell}^{N-1} g_k(x_k, u_k) \qquad (CTG)$$

equipped with constraints (DYN) and (V). If $x_\ell = x_\ell^*$, then the solution to (CTG) is $u_\ell = u_\ell^*, u_{\ell+1} = u_{\ell+1}^*, \dots, u_{N-1} = u_{N-1}^*$.

Remarks on Bellman's principle

- $J_\ell(x_\ell)$ is the cost-to-go. It is the "queue" of (oc) from ℓ to N
- Intuition: you have computed the following optimal route from the city x_0

If one starts in the city x_ℓ^*, the optimal route is still given by roads $u_\ell^*, \dots, u_{N-1}^*$

Dynamic programming

Idea. Compute functions $J_{N-1}(x_{N-1})$, $J_{N-2}(x_{N-2})$, ..., $J_0(x_0)$

backwards $\Rightarrow$ $J_0(x_0) = J(x_0)$ by Bellman's principle!

Drawback: one has to store the values $J_\ell(x_\ell)$, $\forall x_\ell \in S$

Theorem. The optimal cost $J(x_0)$ is equal to $J_0(x_0)$ obtained at the end of the following iterations

$$J_N(x_N) = g_N(x_N) \qquad \text{(A)}$$

$$J_k(x_k) = \min_{u_k \in U(x_k)} g_k(x_k, u_k) + J_{k+1}(f_k(x_k, u_k)), \quad k = N-1, N-2, \dots, 0 \qquad \text{(B)}$$

Moreover, define

$$\mu_k(x_k) = \arg\min_{u_k \in U(x_k)} g_k(x_k, u_k) + J_{k+1}(f(x_k, u_k))$$

$$x^*_{k+1} = f(x^*_k, \mu_k(x^*_k)), \quad x^*_0 = x_0$$

Then, $u^*_0 = \mu_0(x_0)$, $u^*_1 = \mu_1(x^*_1)$, ..., $u^*_{N-1} = \mu_{N-1}(x^*_{N-1})$ are the optimizers of problem OC

Remarks

$$J_N(x_N) = g_N(x_N) \qquad \text{(A)}$$

$$J_k(x_k) = \min_{u_k \in U(x_k)} g_k(x_k, u_k) + J_{k+1}\left(f_k(x_k, u_k)\right), \quad k = N-1, N-2, \dots, 0 \qquad \text{(B)}$$

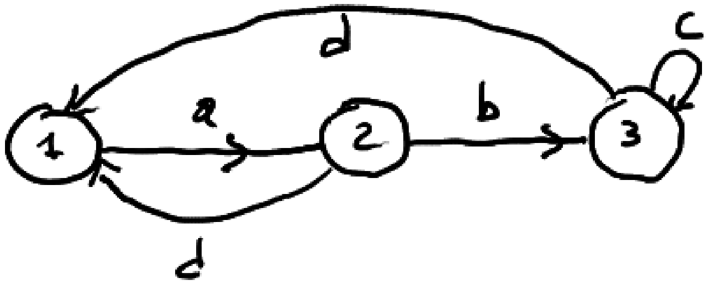
- (A) and (B) are the Dynamic Programming (DP) algorithm
- (B): Bellman's iterations → Optimize over a single input
- $\mu_k(\cdot)$ = state-feedback control law
- The problem of taking N optimal decision has been split into the problem of taking a sequence of atomic decisions + the computation of the CTG

Generalizations

DP can be also applied for solving

- Optimal control problems for continuous-state systems
- OC problems in presence of deterministic or stochastic disturbances

Example



Solve the OC problem

$$J(x_0) = \min_{u_0, u_1, u_2} g_3(x_3) + \sum_{k=0}^{2} g_k(x_k, u_k)$$

where $g_0 = g_1 = g_2$ are defined by the table

$g_{0,1,2}$	a	b	c	d
1	1			
2		1		2
3			0.5	2

and the final cost is $g_3(x) = \begin{cases} 1 & x = 1 \\ 0.5 & x = 2 \\ 0 & x = 3 \end{cases}$

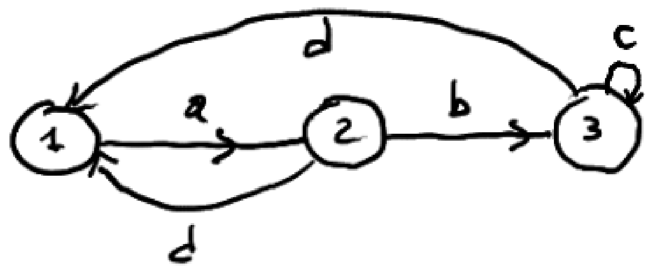
Rmk. Stage and final costs penalize the state OFF

- Set $J_3(x_3) = g_3(x_3)$
- Bellman's iteration

$$J_2(x_2) = \min_{u_2 \in U(x_2)} g_2(x_2, u_2) + J_3(f(x_2, u_2))$$

$$J_2(x_2) = \begin{cases} 1 + 0.5 = 1.5 & x_2 = 1 \\ \min\{\underbrace{1+0}_{\text{input "b"}}, \underbrace{2+1}_{\text{input "d"}}\} = 1 & x_2 = 2 \\ \min\{\underbrace{0.5+0}_{\text{input "c"}}, \underbrace{2+1}_{\text{input "d"}}\} = 0.5 & x_2 = 3 \end{cases}$$

$$\mu_2(x_2) = \begin{cases} a & x_2 = 1 \\ b & x_2 = 2 \\ c & x_2 = 3 \end{cases}$$



$g_{0,1,2}$	a	b	c	d
1	1			
2		1		2
3			0.5	2

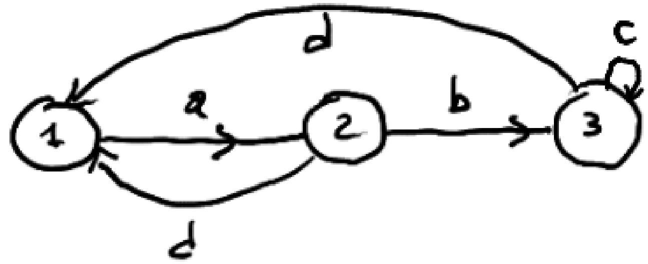
$$g_3(x) = \begin{cases} 1 & x = 1 \\ 0.5 & x = 2 \\ 0 & x = 3 \end{cases}$$

- Bellman's iteration

$$J_1(x_1) = \min_{u_1 \in U(x_1)} g_1(x_1, u_1) + J_2(f(x_1, u_1))$$

$$J_1(x_1) = \begin{cases} 1 + J_2(2) = 2 & x_1 = 1 \\ \min\{\underbrace{1 + J_2(3)}_{\text{input "b"}}, \underbrace{2 + J_2(1)}_{\text{input "d"}}\} = 1.5 & x_1 = 2 \\ \min\{\underbrace{0.5 + J_2(3)}_{\text{input "c"}}, \underbrace{2 + J_2(1)}_{\text{input "d"}}\} = 1 & x_1 = 3 \end{cases}$$

$$\mu_1(x_1) = \begin{cases} a & x_1 = 1 \\ b & x_1 = 2 \\ c & x_1 = 3 \end{cases}$$



$g_{0,1,2}$	a	b	c	d
1	1			
2		1		2
3			0.5	2

$$J_2(x_2) = \begin{cases} 1.5 & x_2 = 1 \\ 1 & x_2 = 2 \\ 0.5 & x_2 = 3 \end{cases}$$

- Bellman's iteration

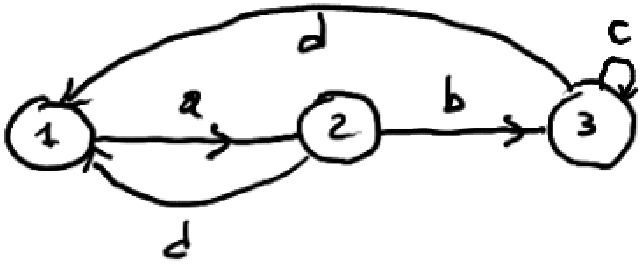
$$J_0(x_0) = \min_{u_0 \in U(x_0)} g_0(x_0, u_0) + J_1(f(x_0, u_0))$$

$$J_0(x_0) = \begin{cases} 1 + J_1(2) = 2.5 & x_0 = 1 \\ \min\{ \underbrace{1 + J_1(3)}_{\text{input "b"}}, \underbrace{2 + J_1(1)}_{\text{input "d"}} \} = 2 & x_0 = 2 \\ \min\{ \underbrace{0.5 + J_1(3)}_{\text{input "c"}}, \underbrace{2 + J_1(1)}_{\text{input "d"}} \} = 1.5 & x_0 = 3 \end{cases}$$

$\parallel$

$J(x_0)$

$$\mu_0(x_0) = \begin{cases} a & x_0 = 1 \\ b & x_0 = 2 \\ c & x_0 = 3 \end{cases}$$



$g_{0,1,2}$	a	b	c	d
1	1			
2		1		2
3			0.5	2

$$J_1(x_1) = \begin{cases} 2 & x_1 = 1 \\ 1.5 & x_1 = 2 \\ 1 & x_1 = 3 \end{cases}$$

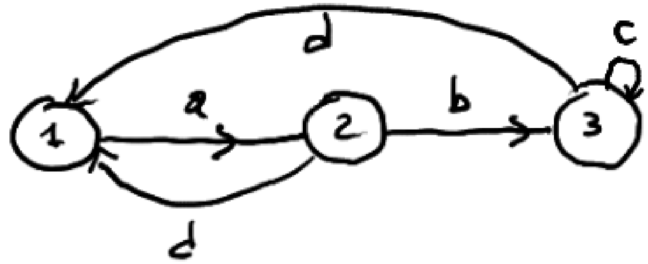
From $x_0 = 1$, the optimal sequence is

$$u_0^* = \mu_0(x_0) = a \rightarrow x_1^* = 2$$

$$u_1^* = \mu_1(x_1^*) = b \rightarrow x_2^* = 3$$

$$u_2^* = \mu_2(x_2^*) = c \rightarrow x_3^* = 3$$

and the optimal cost is $J_0(x_0) = 2.5$



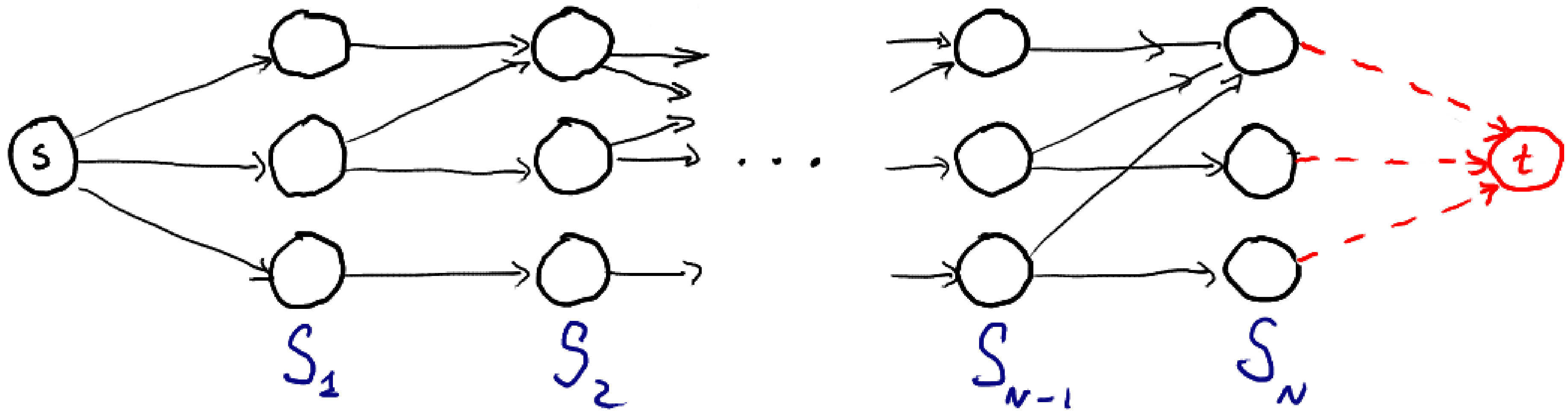
$$\mu_0(x_0) = \begin{cases} a & x_0 = 1 \\ b & x_0 = 2 \\ c & x_0 = 3 \end{cases}$$

$$\mu_1(x_1) = \begin{cases} a & x_1 = 1 \\ b & x_1 = 2 \\ c & x_1 = 3 \end{cases}$$

$$\mu_2(x_2) = \begin{cases} a & x_2 = 1 \\ b & x_2 = 2 \\ c & x_2 = 3 \end{cases}$$

OC problems as shortest path problems

N-step evolution of an automaton as a directed network

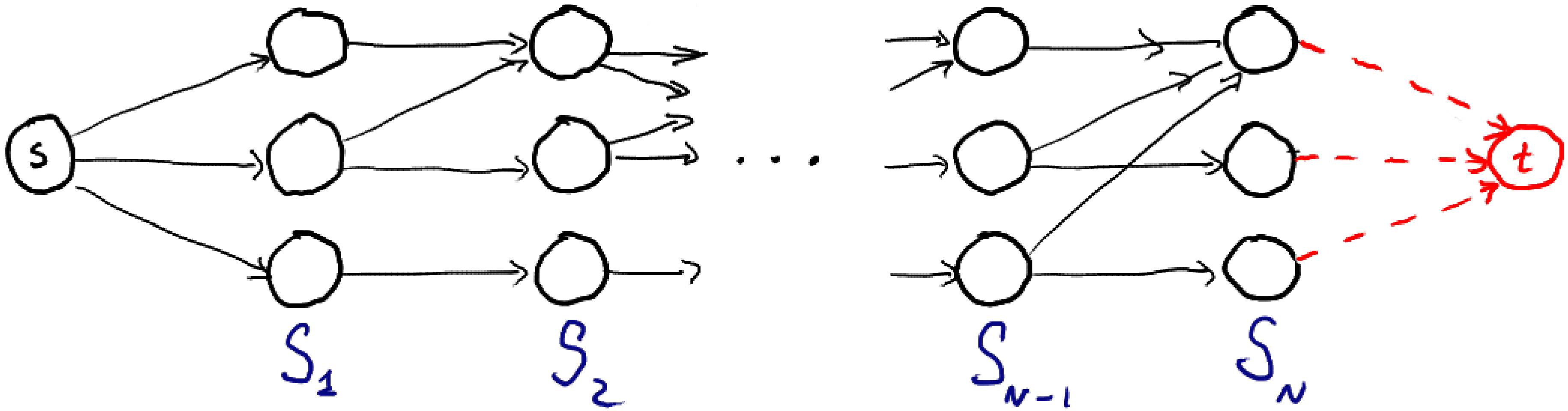


Vertices. Initial state s. S_k are copies of the state set S. Terminal dummy node t.

Edges. For all pairs (i, u), $i \in S_k$, $u \in U(i)$, create an edge from i to $j = f(i, u) \in S_{k+1}$ with weight $g_k(i, u)$.

Create then dummy edges from all $i \in S_N$ to t with weight $g_N(i)$.

OC problems as shortest path problems



Rmk. Starting the automaton with $x_0 = s$ and applying $u_0, \dots, u_{N-1}$ generates a path $x_0, x_1, \dots, x_N, t$ with cost

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

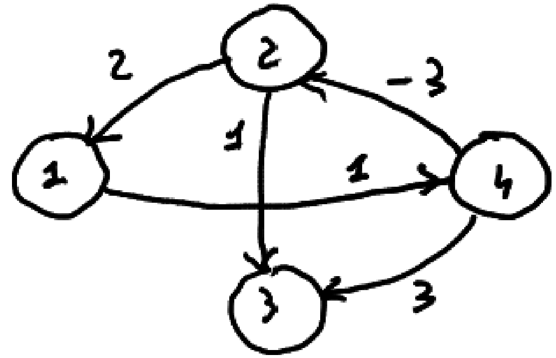
↳ For $x_0 = s$ one can solve the OC problem computing a shortest path from s to t

Shortest path problems as OC problems

Let $G = (V, E, c)$ be a directed network with

- $c(i,j) = +\infty, \quad \forall (i,j) \notin E$
- without cycles with strictly negative cost

↳ a shortest path has at most $N = |V|$ edges



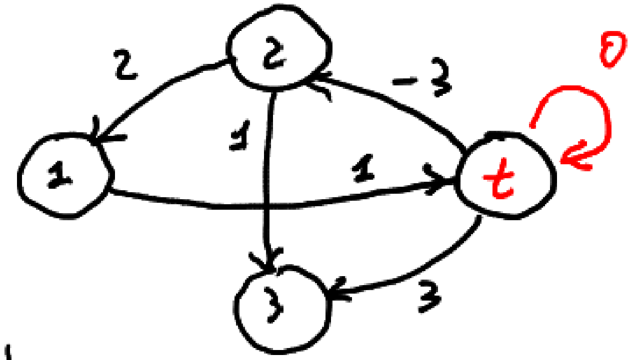
Problem SP: compute a shortest path from all $i \in V$ to a given terminal vertex t

Problem SP1: compute a shortest path of N edges from all $i \in V$ to a given terminal vertex t in the network $\bar{G}$ obtained by adding to G an edge (t,t) with zero cost

↳ Problems SP and SP1 do coincide!

Shortest path problems as OC problems

Definition of the automaton



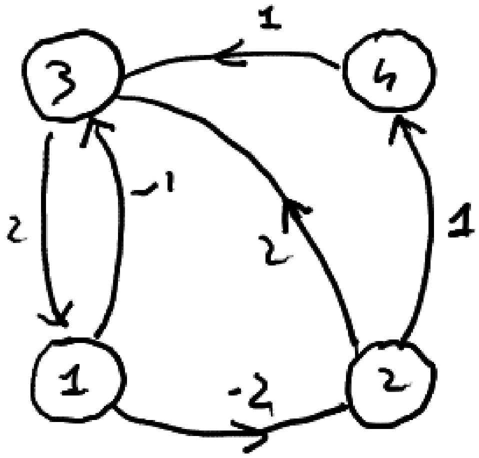
- States: $S = V$
- Admissible inputs: $U(i) = \{(i,j) \in E\}$
- Dynamics: $x_{k+1} = f(x_k, u_k)$ where $f(i, (i,j)) = j$

Rmk. The automaton is a path generator for the graph

OC problem

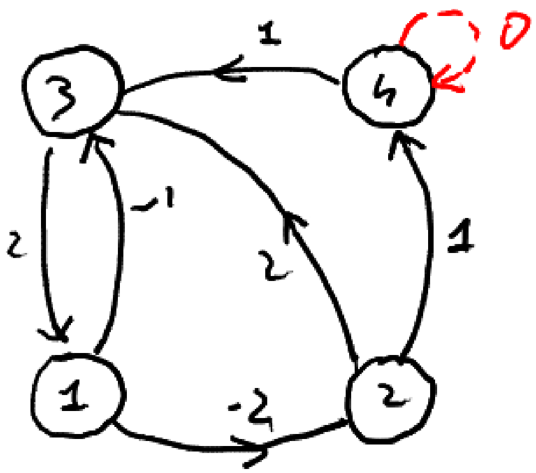
SP1 is equivalent to solve an OC problem with control horizon $N = |V|$ and costs $g_k(i,u) = g(i,(i,j)) = c(i,j)$, $g_N(i) = c(i,t)$.

Example



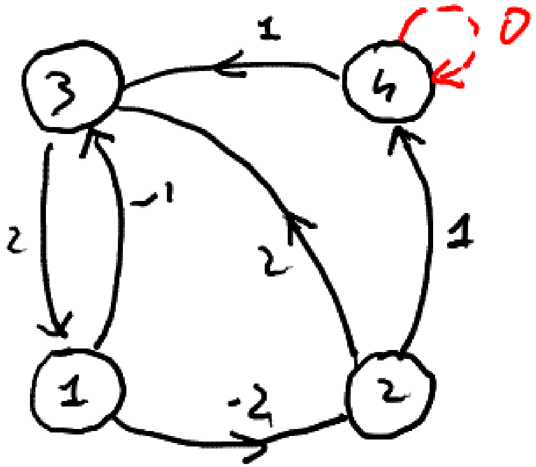
Compute all shortest paths ending in vertex 4.

Provide a shortest path from 3 to 4



- No cycles with cost < 0
- Set $N = |V| = 4$, $S = V$

$$g_4(x) = c(x,4) = \begin{cases} \infty & x=1 \\ 1 & x=2 \\ \infty & x=3 \\ 0 & x=4 \end{cases}$$



- Input set $C = E$
- Stage costs $g_k(x_k, u_k)$, $k = 0, 1, 2, 3$ identical and given by the table

$x \backslash u$	(1,3)	(3,1)	(1,2)	(2,3)	(2,4)	(4,3)	(4,4)
1	-1		-2				
2				2	1		
3		2					
4						1	0

Solve the OC problem $J(x_0) = \min_{u_0, \dots, u_3} g_4(x_4) + \sum_{k=0}^{3} g_k(x_k, u_k)$

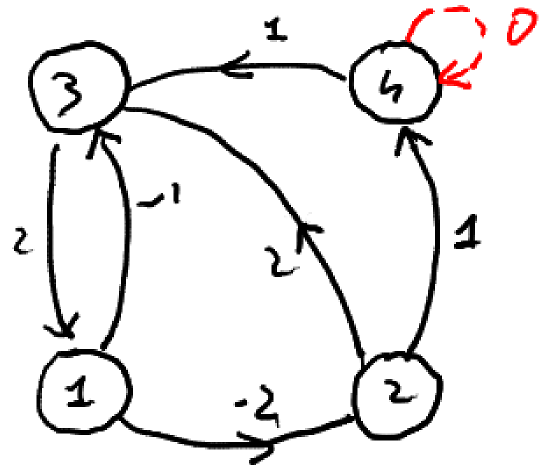
- Set $J_4(x_4) = g_4(x_4)$
- Bellman's iteration

$$J_3(x_3) = \min_{u_3 \in U(x_3)} g_3(x_3, u_3) + J_4(f(x_3, u_3))$$

$$J_3(x_3) = \begin{cases} -2+1 = -1 & x_3 = 1 \\ 1+0 = 1 & x_3 = 2 \\ 2+\infty = \infty & x_3 = 3 \\ 0+0 = 0 & x_3 = 4 \end{cases}$$

$\Rightarrow$ minimal cost from x_3 to 4 in two hops

$$\mu_3(x_3) = \begin{cases} (1,2) & x_3 = 1 \\ (2,4) & x_3 = 2 \\ (3,1) & x_3 = 3 \\ (4,4) & x_3 = 4 \end{cases}$$



$$g_4(x) = \begin{cases} \infty & x = 1 \\ 1 & x = 2 \\ \infty & x = 3 \\ 0 & x = 4 \end{cases}$$

- Bellman's iteration

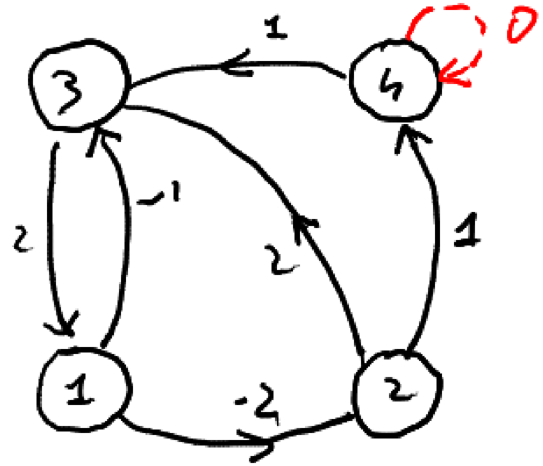
$$J_2(x_2) = \min_{u_2 \in U(x_2)} g_2(x_2, u_2) + J_3(f(x_2, u_2))$$

$$J_2(x_2) = \begin{cases} -2+1 = -1 & x_2 = 1 \\ 1+0 = 1 & x_2 = 2 \\ 2-1 = 1 & x_2 = 3 \\ 0+0 = 0 & x_2 = 4 \end{cases}$$

$\Rightarrow$ minimal cost from x_2 to 4 in three hops

found a finite-cost path from 3 to 4

$$\mu_2(x_2) = \begin{cases} (1,2) & x_2 = 1 \\ (2,4) & x_2 = 2 \\ (3,1) & x_2 = 3 \\ (4,4) & x_2 = 4 \end{cases}$$



$$J_3(x) = \begin{cases} -1 & x = 1 \\ 1 & x = 2 \\ \infty & x = 3 \\ 0 & x = 4 \end{cases}$$

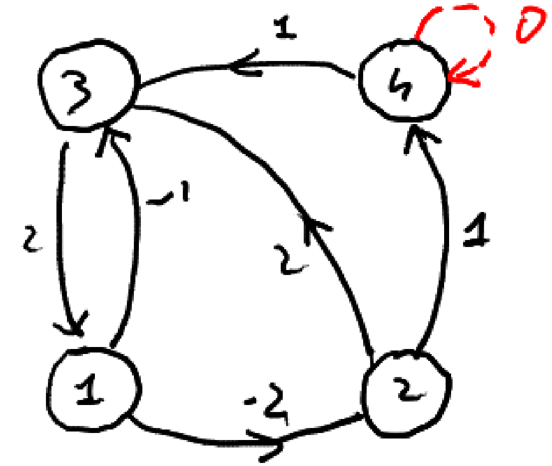
- Bellman's iteration

$$J_1(x_1) = \min_{u_1 \in U(x_1)} g(x_1, u_1) + J_2(f(x_1, u_1))$$

$$J_1(x_1) = \begin{cases} -2+1=-1 & x_1 = 1 \\ 1+0=1 & x_1 = 2 \\ 2+(-1)=1 & x_1 = 3 \\ 0+0=0 & x_1 = 4 \end{cases}$$

$\Rightarrow$ minimal cost from x_1 to 4 in four hops

$$\mu_1(x_1) = \begin{cases} (1,2) & x_1 = 1 \\ (2,4) & x_1 = 2 \\ (3,1) & x_1 = 3 \\ (4,4) & x_1 = 4 \end{cases}$$



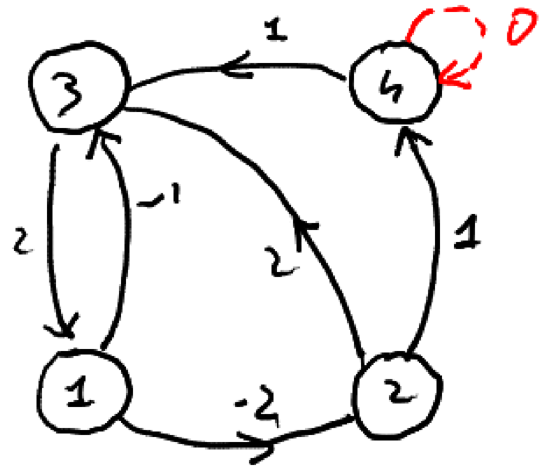
$$J_2(x) = \begin{cases} -1 & x = 1 \\ 1 & x = 2 \\ 1 & x = 3 \\ 0 & x = 4 \end{cases}$$

Rmk. $J_2(x_2) = J_1(x_1)$

↳ Since J_0 depends on J_1 exactly as J_2 depends on J_1, one has $J_0(x_0) = J_1(x_1)$

In general, if all stage costs g_k do not depend on k and there is $\bar{k}$ such that $J_{\bar{k}-1} = J_{\bar{k}}$, then $J_i = J_{\bar{k}}$ and $\mu_i = \mu_{\bar{k}}$ for all $i < \bar{k}$

↳ CTG and control law become stationary



$$J_1(x) = \begin{cases} -1 & x = 1 \\ 1 & x = 2 \\ 1 & x = 3 \\ 0 & x = 4 \end{cases}$$

Shortest path from 3 to 4

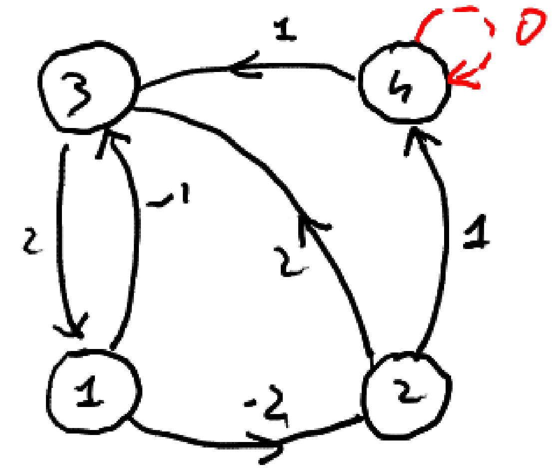
$u_0^* = \mu_0(3) = (3,1) \longrightarrow x_1^* = 1$

$u_1^* = \mu_1(1) = (1,2) \longrightarrow x_2^* = 2$

$u_2^* = \mu_2(2) = (2,4) \longrightarrow x_3^* = 4$

$[\, u_4^* = \mu_4(4) = (4,4) \,]$ → self-loop on terminal node

Cost of the path: $J_0(3) = 1$



$$J_0(x) = \begin{cases} -1 & x = 1 \\ 1 & x = 2 \\ 1 & x = 3 \\ 0 & x = 4 \end{cases}$$