Reinforcement Learning

Lecture 2:
Deterministic Decision Processes

Yishay Mansour, Tel-Aviv University

Lecture 2: outline

- Deterministic Model
- Finite Horizon
  - Shortest paths
  - Dynamic Programing
- Policies
  - History
  - Stochastic
- Average rewards
  - Min Mean cost cycle

Deterministic Decision Processes

- Motivating examples:
  - Routing in a network
  - Scheduling jobs on machines
  - Robot arm

Deterministic Model

- Discrete time dynamic system
  - $x_{t+1} = f_t(x_t, u_t)$
    - t is the time index
  - $x_t \in X_t$ the state variable
    - X_t is the set of possible states at time *t*
  - $u_t \in U_t$ the control variable
    - U_t is the set of possible control at time *t*.
  - $f_t: X_t \times U_t \rightarrow X_{t+1}$ transition function
    - Defines the dynamics

Model

- Horizon H
  - Can be finite or infinite
- Control variable
  - Can depend on the state
  - $u_t \in U_t(x_t) \subset U_t$
- Time invariant system
  - For all t: $X_t = X; U_t = U; f_t = f$

Observations and Rewards

- Observation:
  - $y_t = Obs_t(x_t, u_t)$
    - Fully observable: $y_t = x_t$
- Costs / Rewards
  - Per state action (x_t, u_t)
    - Costs $C(x_t, u_t)$
    - Rewards $R(x_t, u_t) = -C(x_t, u_t)$

Linear Dynamics

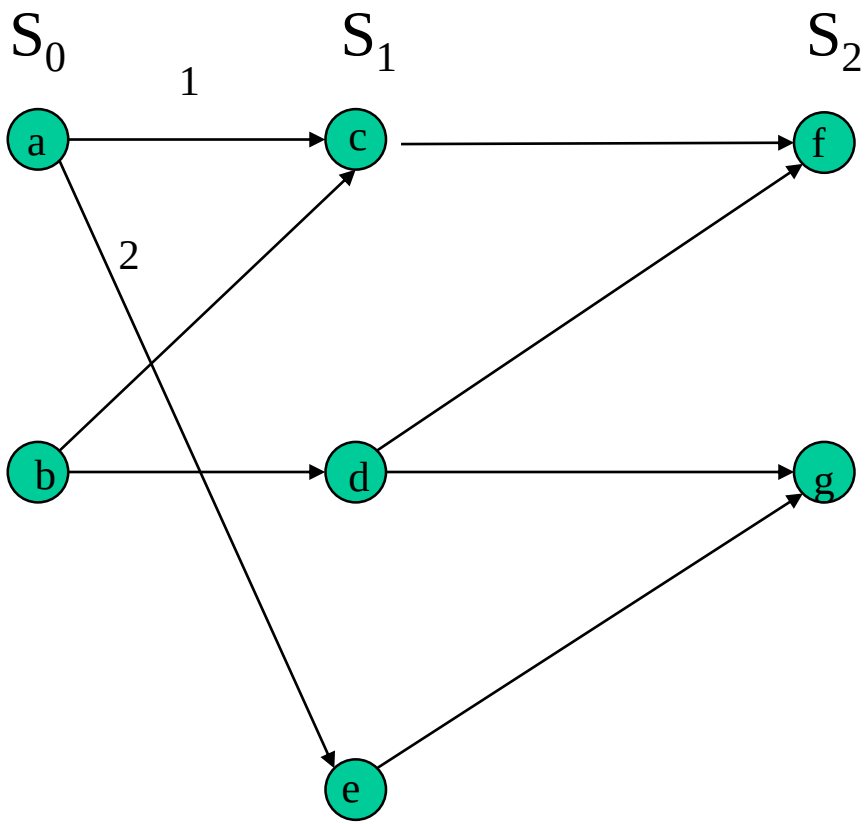
- Continuous state and action
  - $x_t \in \mathbb{R}^n; u_t \in \mathbb{R}^m$
- Linear dynamics
  - $x_{t+1} = Ax_t + Bu_t$
  - Matrices $A \in \mathbb{R}^{n \times n}; B \in \mathbb{R}^{n \times m}$
- Quadratic cost
  - $C(x_t, u_t) = x_t^\top Q x_t + u_t^\top R u_t$
- More in the end of the course (LQR)

Finite Models

- Finite number:
  - States S_t
    - rather than X_t
  - Actions A_t
    - rather than U_t

- Deterministic Decision Process
  - Directed graph
  - Nodes are states
  - Edges are actions

Graphical description



Horizon $H = 2$

$S_0 = \{a, b\}$

$S_1 = \{c, d, e\}$

$S_2 = \{f, g\}$

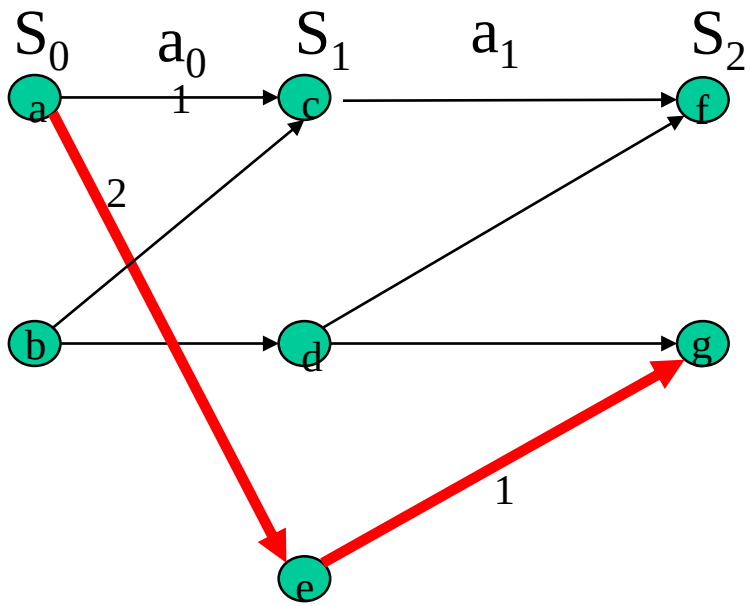
$A_0(a) = \{1,2\}$

$f_0(a, 1) = c$

Feasible paths

- Feasible Path
  - Sequence of states and actions
    - Nodes and edges

      $(s_0, a_0, s_1, a_1, \dots, s_T)$
  - $a_t \in A_t(s_t)$
  - $s_{t+1} = f_t(s_t, a_t)$
- Example: (a,2,e,1,g)



Finite Horizon

- The return function:
  - Given $path = (s_0, a_0, s_1, a_1, \dots, s_T)$
  - $J(path) = \sum_{t=0}^{T-1} c_t(s_t, a_t) + c_T(s_T)$
- Simplest *return* objective
  - Cost to go
- Similar formulation for rewards
  - Simply $r_t(s_t, a_t) = -c_t(s_t, a_t)$

Optimal Path

- Optimal path
  - Given a start state s_0
  - Feasible path from s_0
    - Length T
  - Minimizes cost to go
- $OptPath = \min_{path_T \in PATHS_T} J(path_T)$

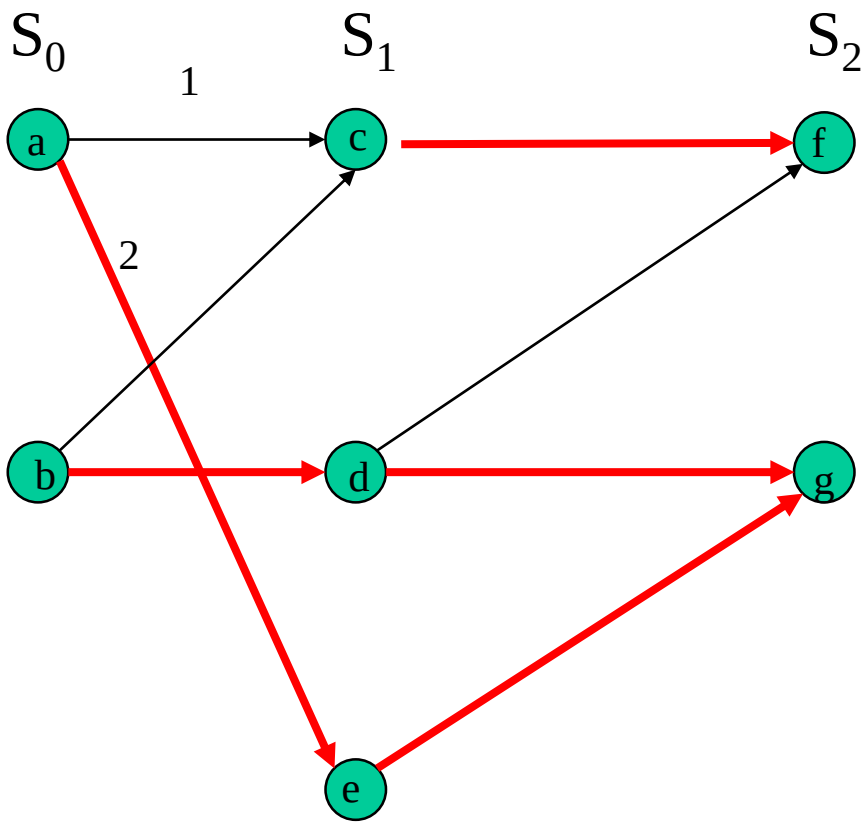
Control policies

- Control strategy
  - Maps histories to distribution over actions
- Control policy
  - Maps states to distribution over actions
- Stochastic vs deterministic
  - Distribution over actions versus a single action
- Paths versus policies

Control Policies and Strategies

- SD: Stationary Deterministic
  - $\pi: S \to A$
- MD: Markov Deterministic
  - $\pi: S \times T \to A$
- HD: History Deterministic
  - $\pi: \mathbb{H} \to A$
- SS: Stationary Stochastic
  - $\pi: S \to \Delta(A)$
- MS: Markov Stochastic
  - $\pi: S \times T \to \Delta(A)$
- HS: History Stochastic
  - $\pi: \mathbb{H} \to \Delta(A)$
- Optimal control:
  - Minimizes cost
  - Maximizes reward

Control policies versus path



Finite Horizon

- Evaluation of a general policy π
  - History-dependent Stochastic (HS)
- Induces distribution over paths
  - Possibly very complicated interaction
- Focus on what is needed for evaluation
  - $\Pr[s_t = s, a_t = a]$

Finite Horizon

- The expected cost to go

$$E[J(path_T)] = \sum_{t=1}^{T-1} E[c_t(s_t, a_t)] + E[c_T(s_T)]$$

  - Note that s_t and a_t are random variables
    - depend on the history $(s_0, a_0, \dots, s_{t-1}, a_{t-1})$

Finite Horizon: Expected cost

- Expectation:
  $\rho_t(s,a) = E_{h_{t-1}} \Pr[(s_t = s, a_t = a | h_{t-1}]$
- Path:
  - $\sum_{t=0}^{T-1} \sum_{a \in A_t; s \in S_t} c_t(s,a) \quad \rho_t(s,a)$
- Termination:
  - $\sum_{s \in S_T} c_T(s) \rho_T(s)$

Lecture 2: outline

- Deterministic Model
- Finite Horizon
  - Shortest paths
  - Dynamic Programing
- Policies
  - History
  - Stochastic
- Average rewards
  - Min Mean cost cycle

History versus Markovian

- Theorem
  - For every strategy $\pi \in HS$
  - There exists a policy $\pi' \in MS$
  - Such that

$$E^{\pi}[J(path)] = E^{\pi'}[J(path')]$$

History versus Markovian

- Proof:
  - Given π we define π'
  - For every $s \in S_t$ define $\pi'_t(\cdot\,|s)$ as follows
    - $E_{h_{t-1}}[\Pr[a_t = a|s_t = s, h_{t-1}]] = \pi'_t(a|s)$
  - Note:
    - π' is Markovian
    - $\rho_t(s_t = s, a_t = a)$ is identical for π and π'
- QED

Stochastic versus Deterministic Policies

- Theorem
  - For every policy $\pi \in MS$
  - There exists a policy $\pi' \in MD$
  - Such that

$$E_{\pi}[J(path)] \geq J(path')$$

where $path'$ is generated by π'

Stochastic versus Deterministic Policies

- Proof:
  - The proof is by induction
  - Inductive claim:
    - For any policy π which is deterministic in $[t+1, T]$
    - There is a policy π' which is deterministic in $[t, T]$

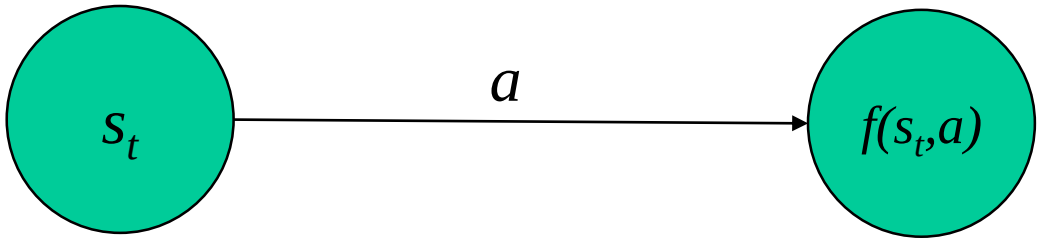
    such that
    - $E^{\pi}[J(path)] \geq E^{\pi'}[J(path')]$
  - Base $t = T$

Stochastic versus Deterministic Policies

- Assume policy π is deterministic in $[t+1, T]$
  - For every state $s_{t+1} \in S_{t+1}$ define
  - $J(s_{t+1}) = J\big(path(s_{t+1}, \dots, s_T)\big)$
    - $path(s_{t+1}, \dots, s_T)$ is a deterministic path of π
  - Need to define π'

Stochastic versus Deterministic Policies

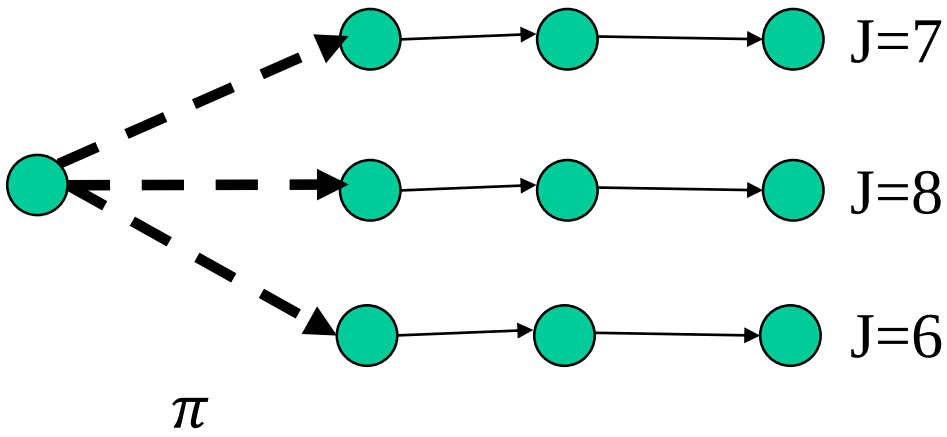
- Define policy π'
  - For every state $s_t \in S_t$ define:
  - $\pi'_t(a_t|s_t) = \arg\min_{a \in A_t} J(f_t(s_t, a))$



Stochastic versus Deterministic Policies

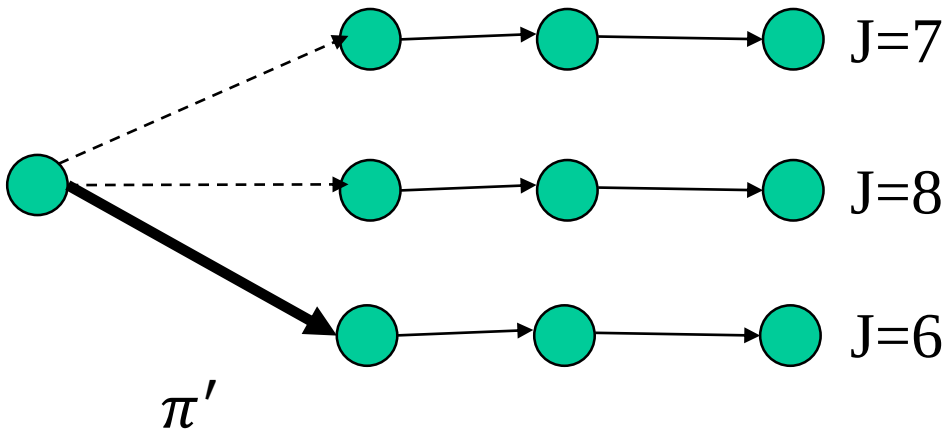
- Analysis:
  - Until time t the policies are identical
  - The definition of π' can only decrease the cost until the end.
  - $E^{\pi}[J(s_t, \dots, s_T)] = E^{\pi} E_{a_t}[J(f_t(s_t, a_t), \dots, s_T)]$
    $\geq E^{\pi} \min_a [J(f_t(s_t, a), \dots, s_T)] = E^{\pi'}[J(s_t, \dots, s_T)]$
- QED

Example



$$E^{\pi}[J(path)] = 7$$

Example



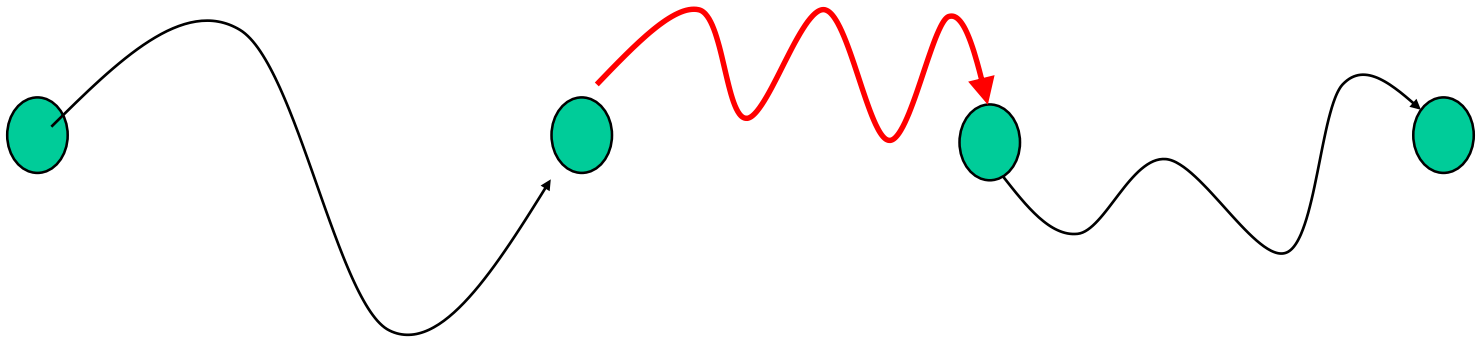
$$E^{\pi'}[J(path)] = 6$$

Optimality Criteria

- Corollary:
  - For any Deterministic Decision Process
  - For the finite horizon objective
  - There exists a Deterministic Markovian policy
    - namely, a path
  - Which minimizes the cost to go
- Remark:
  - Similar result for stochastic processes

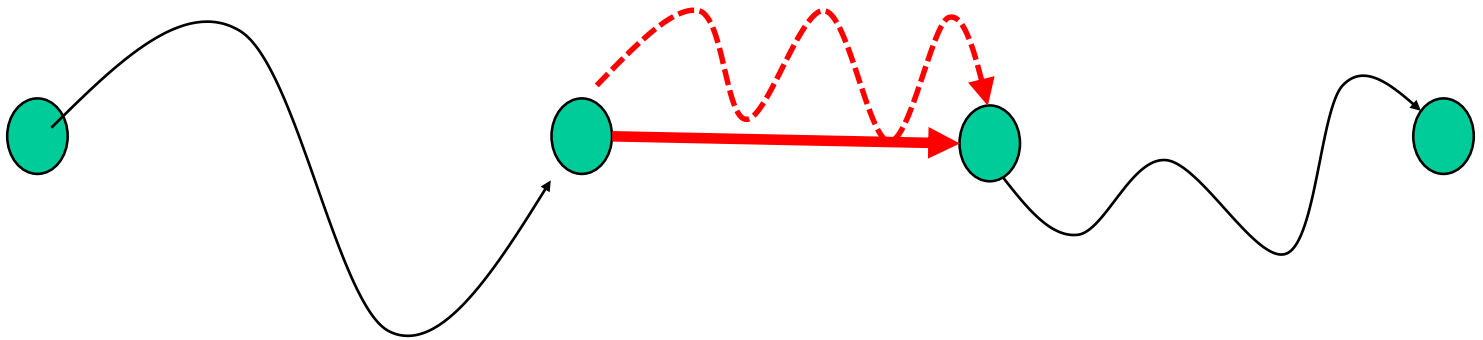
Finite Horizon: Dynamic Programing

- Theorem [Bellman's optimality]
  - For deterministic decision process:
  - Any sub-path of an optimal path is optimal



Finite Horizon: Dynamic Programing

- Proof: Assume a sub-path sub-optimal
  - Replace it, and improve.
- QED



Dynamic program: Finite Horizon

- Initialize: $J_T(s) = c_T(s) \ \forall s \in S$
- Backward recursion, for $t = T-1, \dots, 0$
  - $J_t(s) = \min_{a \in A_t} \{c_t(s,a) + J_{t+1}\big(f_t(s,a)\big)\}$
- Optimal policy: any $\pi^* = (\pi_t^*)$ s.t.
  - $\pi_t^*(s) \in \min_{a \in A_t} \arg\{c_t(s,a) + J_{t+1}(f_t(s,a))\}$

Dynamic program: Finite Horizon

- Bellman Optimality
  - $J_t(s) = \min_a \{c_t(s,a) + J_{t+1}(f_t(s,a))\}$
    - One edge look ahead
  - Value Iteration

Dynamic program: Finite Horizon

- Proposition:
  - The control policy π^* is optimal for the T-horizon problem
  - $J_0(s)$ is the optimal T-horizon return
    - $J_0(s) = \min_{\pi} J_0(\pi; s)$

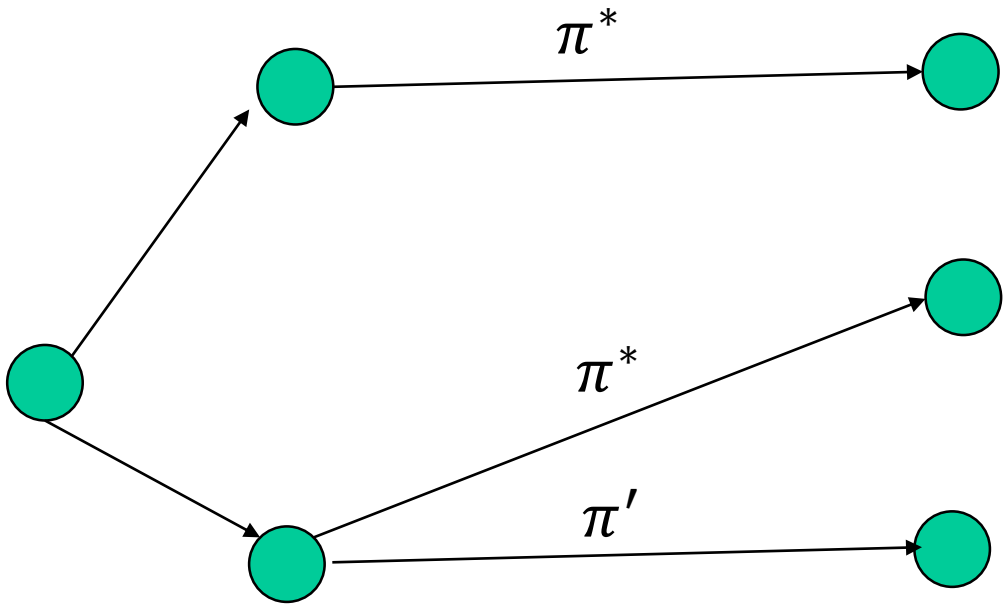
Dynamic program: Finite Horizon

- Proof:
  - Inductive claim:
    - For any s the path from s define by π^* is the minimum cost path of length T-t
    - The value of $J_t(s)$ is the min cost from s
  - The proof is by a backward induction from T
  - Basis: $t=T$. From the initialization.

Dynamic program: Finite Horizon

- Assume the inductive claim holds for *t+1*
  - prove for *t*.
- For contradiction assume there is a lower cost path from s.
  - Path by π^* is $(s, s_{t+1}, \ldots, s_T)$
  - Let $(s, s'_{t+1}, \ldots, s'_T)$ be an alterative path by π'
  - Let $(s'_{t+1}, \hat{s}_{t+2}, \ldots, \hat{s}_T)$ be the path from s'_{t+1} of π^*

Dynamic program: Finite Horizon



J_1

J_2

J_3

$J_1 \leq J_2$

$J_2 \leq J_3$

QED

Computing shortest paths

- Bellman Ford
  - Handles also negative weights
  - No negative cycle
- Dijkstra
  - Only non-negative weights
- Both single source or destination
  - Source is the start state
  - Destination is a “goal state”

Lecture 2: outline

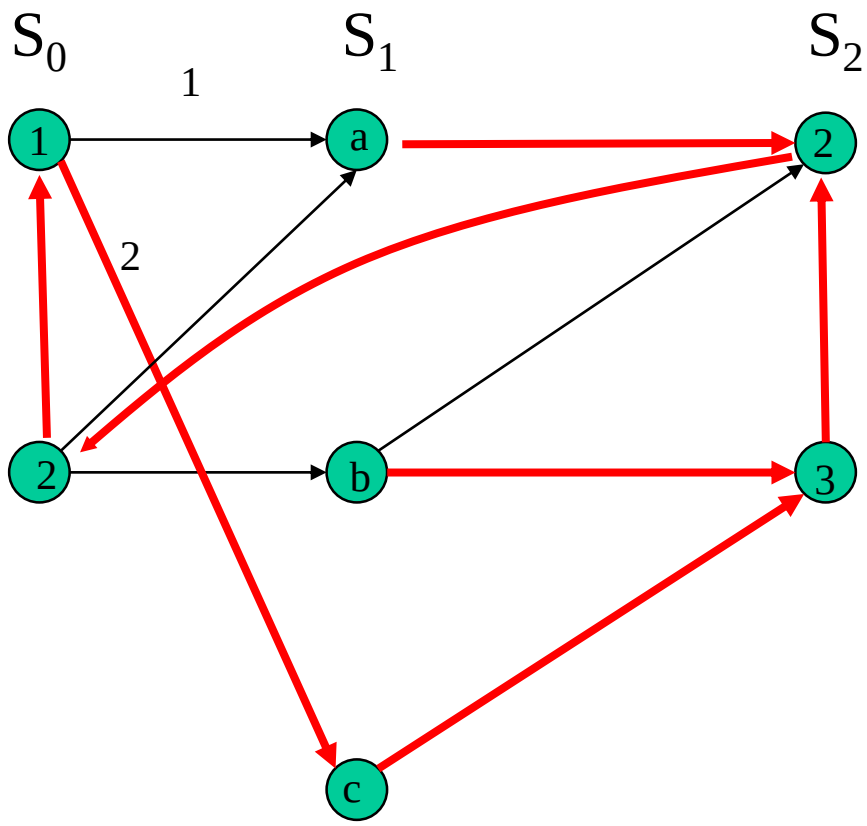
- Deterministic Model
- Finite Horizon
  - Shortest paths
  - Dynamic Programing
- Policies
  - History
  - Stochastic
- Average rewards
  - Min Mean cost cycle

Average Cost

- Definition:
  - $\lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} c_t(s_t,a_t)$
- Intuition:
  - Long term cost
  - Any initial finite segment not important.
- Model (DDP):
  - Directed graph
    - Strongly connected
  - Nodes = states
  - Edges = actions

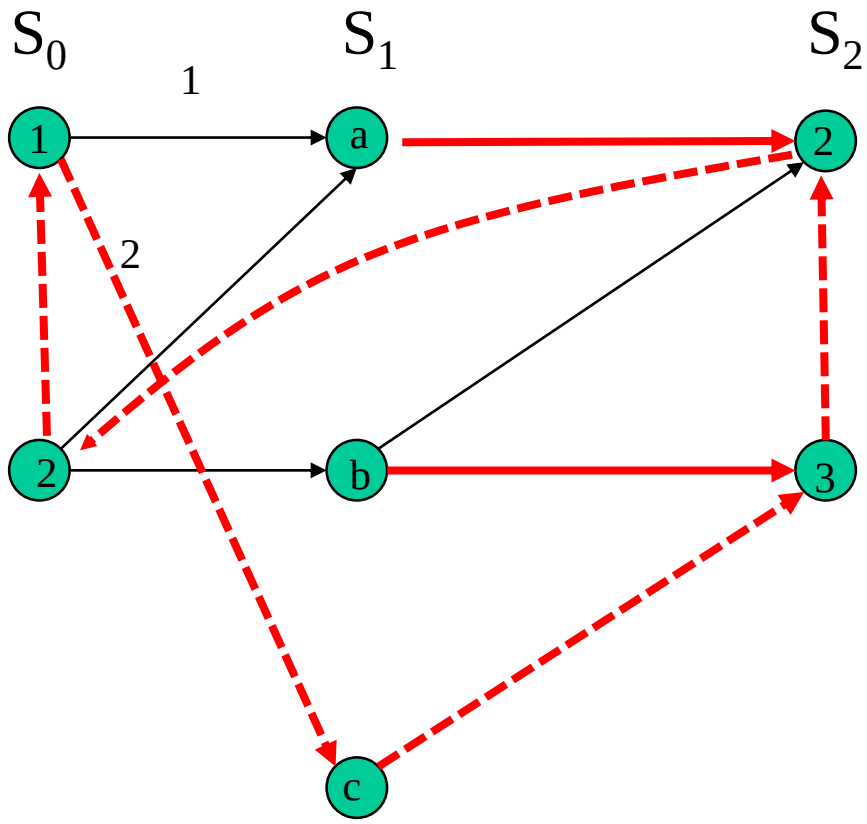
Average Cost

- Consider a directed graph and a policy



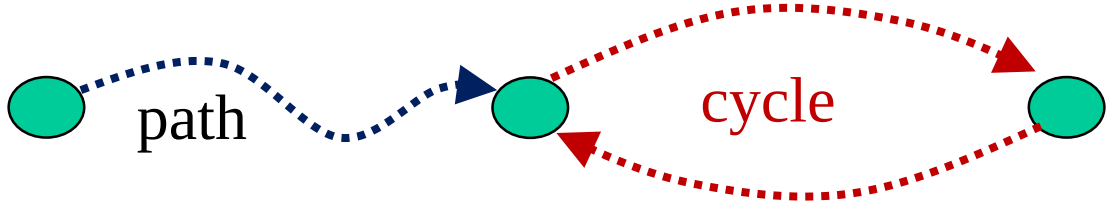
Average Cost

- Every deterministic policy creates a cycle



Average cost: cycle

- Deterministic policy



- The average cost of the policy is the average cost of the cycle!
  - Path has negligible influence.
- Every deterministic policy has a simple cycle.

Average Cost: min cost cycle

- Given a directed graph, let Ω be the collection of cycles
- For each cycle $\omega = (v_1, \ldots, v_k) \in \Omega$ define
  - Length $|\omega| = k$
  - Cost $c(\omega) = \sum_{i=1}^{k} c(v_i, v_{i+1})$
  - $\mu(\omega) = \frac{c(\omega)}{|\omega|}$
- Min cost cycle $\mu^* = \min_{\omega \in \Omega} \mu(\omega)$

Optimal average cost

- Proposition:
  - For any deterministic decision process (DDP):
    - the optimal average cost is μ^*
  - The optimal policy is π_ω
    - cycles on a simple cycle ω with mean cost μ^*

Optimal average cost

- Proof:
  - Let ω be a cycle of average cost μ^*
  - Let π_ω be any policy that:
    - First reaches ω and then cycles ω
    - the average cost of π_ω is μ^*
  - We show that any other strategy has an average cost of at least μ^*

- Assume that there is a strategy π' that has average cost $\mu^* - \varepsilon$
  - Consider a sufficiently long run of π'
    - Fix any realization θ to length $T > \frac{2|S|}{\epsilon}$
  - Consider any simple cycle λ in θ
    - The strategy π' does not to go forever on the cycle!
  - By definition we have $\mu(\lambda) \geq \mu(\omega) = \mu^*$
  - Deleting λ only decrease the average cost of θ

- Continue the process, until there are no remaining cycles
  - The remaining length is at most |S|
- The average cost of θ is at least $\mu^* - \frac{|S|}{T}$
  - Costs are in [0,1]
- The average cost of π_ω is at most $\mu^* + \frac{|S|}{T}$
- We have a contradiction for $T > \frac{2|S|}{\epsilon}$
- QED

Min Mean Cost Cycle: Algorithm

- Input
  - Directed strongly connected graph *G(V,E)*
    - with cost $c: E \to \mathbb{R}$, $n = |V|, m = |E|$
- Output:
  - A cycle ω with $\mu(\omega) = \mu^*$
- Efficient Algorithm
  - Computation issue

Min-Mean cost cycle: THM

- Set a root r
- $F_k(v)$ paths of length k from r to v
- $d_k(v) = \min_{P \in F_k(v)} c(P)$
  - If $F_k(v) = \emptyset$ Then $d_k(v) = \infty$
- Theorem

$$\mu^* = \min_{v \in S} \max_{0 \le k < n-1} \left\{ \frac{d_n(v) - d_k(v)}{n-k} \right\}$$

  - Assume $\infty - \infty = \infty$

- Proof: Two cases: $\mu^* = 0$ and $\mu^* > 0$
- Start with $\mu^* = 0$
- There is a cycle of weight zero
  - But no negative cycle
- Sufficient to show:

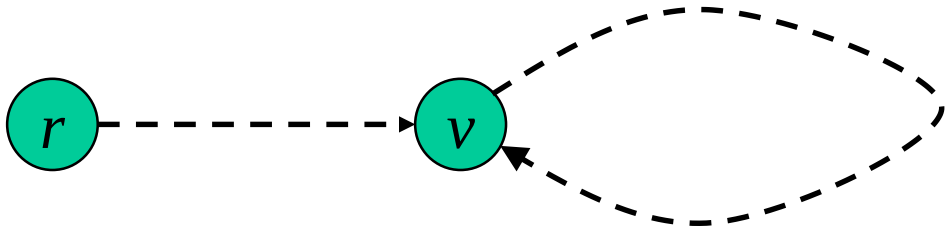
$$0 = \min_{v \in S} \max_{0 \le k < n-1} \{d_n(v) - d_k(v)\}$$

- For every node *v*,
  - there is a simple shortest path of cost *d(v)* from *r*
  - $\max_{0 \le k < n-1} \{d_n(v) - d_k(v)\} = d_n(v) - d(v)$
- Since $d_n(v) \ge d(v)$

$$\min_{v \in V} \{d_n(v) - d(v)\} \ge 0$$

- Need to show for some $v \in V$:
  - $d_n(v) = d(v) \quad \Rightarrow \min_{v \in V} \{d_n(v) - d(v)\} = 0$

- Consider a cycle ω weight zero
  - Let *v* be any node in ω
- Consider a shortest path P from *r* to *v* and then multiple times the cycle ω
  - P is a shortest path to *v*
    - any prefix is also shortest path



- Let P’ be a prefix of length n of P,
  - Let u be the end of P’
  - Path P' and $|P'| = n$
  - $d_n(u) = d(u)$
    - Since u is on a shortest path P
- Therefore

$$\min_{v \in V}\{d_n(v) - d(v)\} = 0$$

- Completes the case $\mu^* = 0$.

- For $\mu^* > 0$
- When we subtract a constant Δ from all costs
  - The min mean cost cycle does not change
  - The new min mean cost cycle $\hat{\mu}^* = \mu^* - \Delta$
- Solution: set $\Delta = \mu^*$
  - There is a cycle of cost zero.
  - Apply the previous proof.
- QED

Min Mean Cycle: Algorithm

- Run "Dynamic Programing"
- Initialize $d_0[r] = 0; d_0[v] = \infty \; \forall v \neq r$
- For $i = 1 \; to \; n$:
  - For every $v \in S$
  - $d_i[v] = \min_{u:(u,v)\in E} \{d_{i-1}[u] + w(u,v)\}$

Min Mean Cycle: Algorithm

- Compute μ^* using the THM:

$$\mu^* = \min_{v \in S} \max_{0 \le k < n-1} \left\{ \frac{d_n(v) - d_k(v)}{n-k} \right\}$$

- Let (v,k) be the minimizing pair
  - P the path of length n from r to v that achieves $d_n(v)$
  - Constructed backtracking the d_i

- Output any cycle ω in P
  - P has some cycle, since has $n+1$ nodes

- Complexity $O(mn)$
  - $n = |V|$ and $m = |E|$

Lecture 2: outline

- Deterministic Model
- Finite Horizon
  - Shortest paths
  - Dynamic Programing
- Policies
  - History
  - Stochastic
- Average rewards
  - Min Mean cost cycle