CONTROLLED MARKOV CHAINS

Carlos F. G. Bispo, Pedro U. Lima

Instituto Superior Técnico (IST)

Instituto de Sistemas e Robótica (ISR)

Av.Rovisco Pais, 1

1049-001 Lisboa

PORTUGAL

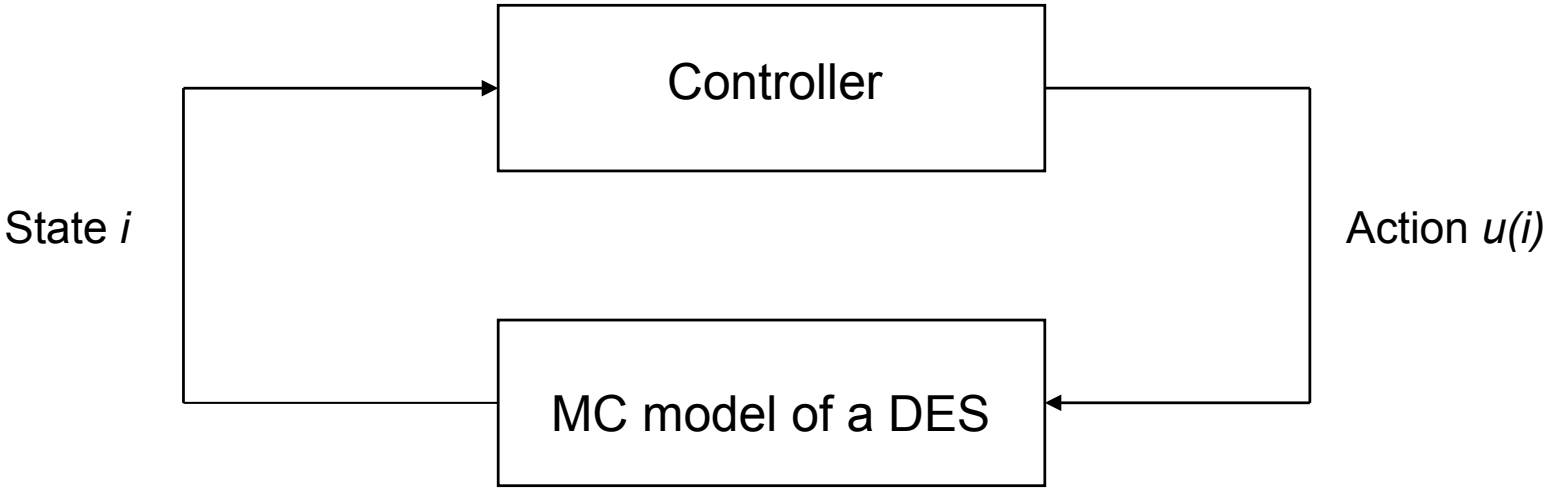
Carlos Bispo, March 2001

Revised by Pedro U. Lima, December 2002

CONTROLLED MARKOV CHAINS
Outline

- Introduction
- The Nature of "Control" in Markov Chains
- Markov Decision Processes
- Solving Markov Decision Problems
  - Basic Dynamic Programming
  - DP & the Optimality Equation
  - Extensions to Unbounded and Undiscounted Costs
  - Optimization of the Average Cost Criterion
- Control of Queuing Systems – basic concepts
- Conclusions

INTRODUCTION



Goal: attain the “best” possible performance for the system.

THE NATURE OF "CONTROL"

- Our analysis of Markov chains so far considered transition probabilities to be fixed;
- We are going to explore the possibility of taking action to control these probabilities;
- What is meant by controlling a Markov chain?

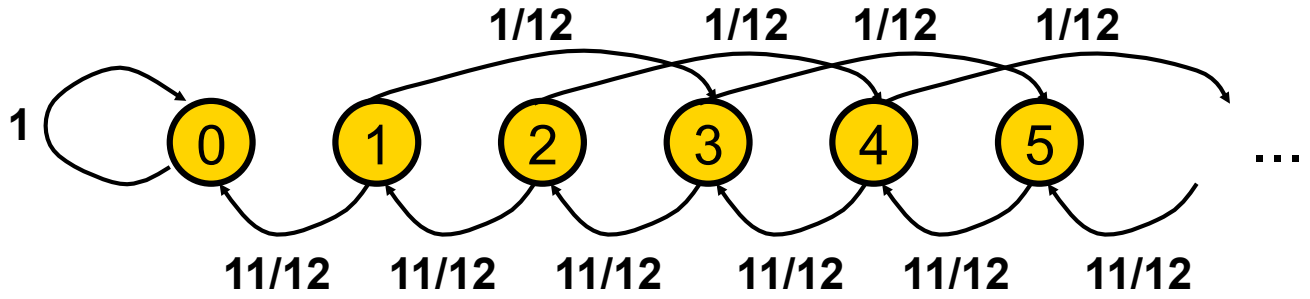
- Example

  Suppose you are a player on a gambling process. At any given point in time you have *i* USD. If you bet 1 unit you either lose it with probability 11/12 or gain 2 units more with probability 1/12.

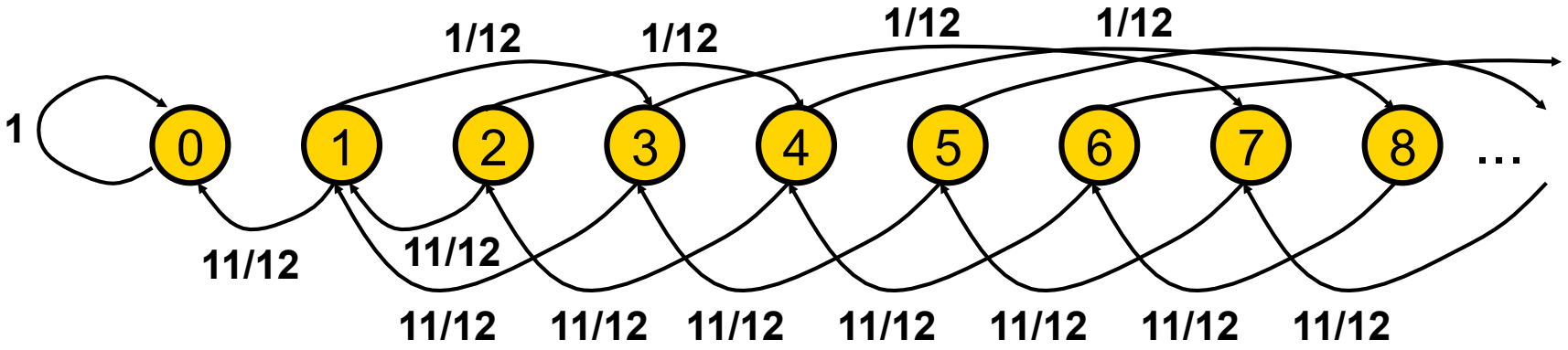
  How much should you bet in order to
  - stay in game?
  - achieve some amount?
  - etc.?

THE NATURE OF "CONTROL"

Policy #1 - One dollar a turn



Policy #2 - Two dollars a turn when above 2 in hand



MARKOV DECISION PROCESSES

- Let a Discrete Event System have state space *X* and assume we can observe all state transitions.

- To introduce the MDP related with this DES we need to specify three ingredients:
  - *Control actions* taken when a state transition takes place;
  - *Cost* associated with such actions;
  - *Transition probabilities* which may depend on the control actions.

Control Actions

- When a new state is entered, a *control action u* is selected from a known set of possible control actions denoted by *U*.

- There is a cost associated with the selection of a control action *u* at state *i*, denoted by *C(i, u)*.

  C(i, u) will be assumed as bounded, that is,

  $$0 \leq C(i, u) \leq K$$

- The rule based on which control actions are chosen is called a *policy* denoted as π.
  - Can be quite arbitrary

MARKOV DECISION PROCESSES

Control Actions (continued)

- We will limit our analysis to a class of policies where
  - a control action is not chosen at random;
  - the control action chosen when the state is *i* depends only on *i*.

stationary policies

- Under a stationary policy, a control action is a mapping from the state set *X* to the set *U*.
  - That is, it is a function of the form *u(i)*, $i \in X$.

- Sometimes, not all control actions in *U* may be allowable when the state is *i*.
  - U_i denotes the subset of *U* containing all *admissible* actions at state *i*.

MARKOV DECISION PROCESSES

Control Actions (continued)

- Given that $u(i)$ has been chosen at state i, the next state is selected according to transition probabilities $p_{ij}[u(i)]$, which depend on the value of i alone.

- We assume that, for any state i, the holding time is exponentially distributed with rate parameter $\Lambda(i)$.

- Once we know $p_{ij}[u(i)]$ and $\Lambda(i)$ we have completely specified a Markov Chain model.

  The only novelty here is the fact that the transition probabilities depend on the particular policy we wish to adopt.

Cost Criteria

- Total Expected Cost over a Finite Horizon

$$V_\pi(x_0) = E_\pi\left[\int_0^T C\left[X(t),u(t)\right]dt\right]$$

- Total Expected Discounted Cost over an Infinite Horizon

$$V_\pi(x_0) = E_\pi\left[\int_0^\infty e^{-\beta t} C\left[X(t),u(t)\right]dt\right]$$

- Total Expected (Undiscounted) Cost over an Infinite Horizon

$$V_\pi(x_0) = E_\pi\left[\int_0^\infty C\left[X(t),u(t)\right]dt\right]$$

- Expected Average Cost

$$V_\pi(x_0) = \lim_{T\to\infty}\frac{1}{T} E_\pi\left[\int_0^T C\left[X(t),u(t)\right]dt\right]$$

Problem of interest

Determine a policy π to minimize

$$V_\pi(x_0) = E_\pi\left[\int_0^\infty e^{-\beta t} C\left[X(t), u(t)\right] dt\right]$$

Total Expected Discounted Cost over an Infinite Horizon

Uniformization

- Let a continuous time Markov Chain be described by

the matrix **Q** – *Transition Rate Matrix*.

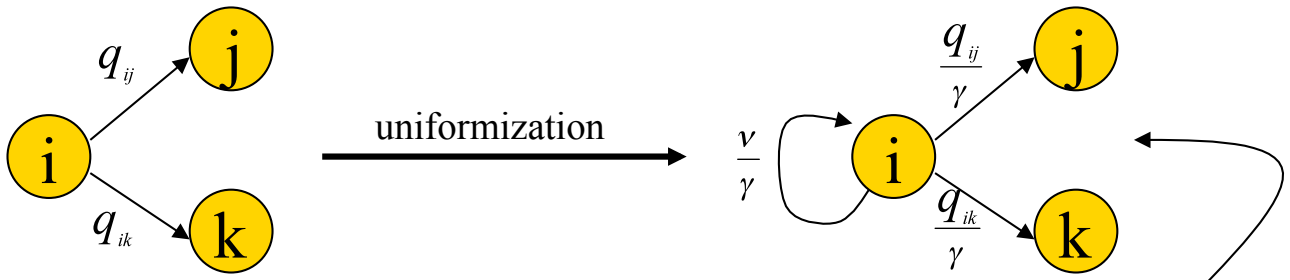
- q_{ij} designates the instantaneous rate at which the chain jumps to state j when at state i.

- It is known that $q_{ii} = -\Lambda(i)$ and $q_{ij} = \lambda_{ij}$

- We can define the probability of jumping to state j when at state i by $P_{ij} = - q_{ij} / q_{ii}$ for $j \neq i$.

  There are no transitions from i to i in a continuous time chain.

Uniformization (continued)



$$\gamma = q_{ij} + q_{ik} + \nu$$

In general, let γ be an upper bound on all $\Lambda(i) = -q_{ii}$

The uniformized transition probabilities are defined as

CTMC ⇒ DTMC

$$P_{ij}^{U} = \begin{cases} \dfrac{\Lambda(i)}{\gamma} P_{ij} & \text{if } i \neq j \\ 1 - \dfrac{\Lambda(i)}{\gamma} & \text{if } i = j \end{cases}$$

- Uniformization (continued)

Q.: *What is the effect of uniformization on the cost functions*?

- Let's see what happens with the total expected discounted cost over an infinite horizon...
- Let T_0, T_1, ..., T_k, ... be the time instants when state transitions occur (including the fictitious ones). $T_0 = 0$ by convention.

$$E_\pi\left[\int_0^\infty e^{-\beta t} C\left[X(t),u(t)\right]dt\right] = \sum_{k=0}^{\infty} E_\pi\left[\int_{T_k}^{T_{k+1}} e^{-\beta t} C\left[X(t),u(t)\right]dt\right]$$

Given that cost is incurred when a control action is decided and this is done at time T_k, it turns out that

$$E_\pi\left[\int_0^\infty e^{-\beta t} C\left[X(t),u(t)\right]dt\right] = \sum_{k=0}^{\infty} E_\pi\left[\int_{T_k}^{T_{k+1}} e^{-\beta t} dt\right] E_\pi\left[C(X_k,u_k)\right]$$

Uniformization (continued)

After some trivial manipulation and taking into account that $T_k+1 = T_k + V_{k+1}$ and that V_{k+1} is exponentially distributed with parameter γ, it can be shown that

$$E_\pi\left[\int_0^\infty e^{-\beta t} C\left[X(t),u(t)\right]dt\right] = \frac{1}{\beta+\gamma} E_\pi\left[\sum_{k=0}^{\infty} \alpha^k C(X_k,u_k)\right]$$

Where

- $\alpha = \gamma/(\beta+\gamma)$ is the new discount factor – for a DTMC.
- $C(i,u)/(\beta+\gamma)$ is the single stage cost whenever we enter state i and choose control action u (dictated by policy π)

The Basic Markov Decision Problem

- Limit ourselves to discrete-time Markov chains.
  - If a continuous-time model with discount factor β is of interest, then it is uniformized with rate γ.
  - In the resulting discrete-time model we use a new discount factor, $\alpha = \gamma/(\beta+\gamma)$, and replace the original costs $C(i, u)$ by $C(i, u)/(\beta+\gamma)$.

- We assume that at every state transition a cost $C(i, u)$ is incurred, where i is the state entered and u a control action selected from a set of admissible actions U_i.

- Under a stationary policy π, the control u depends only on the state i.

The Basic Markov Decision Problem (continued)

- The next state is determined according to transition probabilities $p_{ij}(u)$.
- We assume that a discount factor α, $0 < \alpha < 1$, is given, as well as the initial state is specified.
- We then define the cost criterion

$$V_\pi(i) = E_\pi\left[\sum_{k=0}^{\infty} \alpha^k C(X_k, u_k)\right]$$

which is the total expected discounted cost accumulated over an infinite horizon, given the initial state *i*.

Note: strictly speaking, $V_\pi(i) = E_\pi\left[\sum_{k=0}^{\infty} \alpha^k C(X_k, u_k) \mid X_0 = i\right]$ (not used to simplify notation)

INSTITUTO SUPERIOR TÉCNICO

SOLVING MDPs
The Basics of Dynamic Programming

Discrete-time *deterministic setting*

- At time $k = 0$, the state is x_0.
- Our time horizon consists of N steps, and when we reach state x_N at time $k = N$, we will incur a terminal cost $C(x_N)$.
- At each time step $k = 0, 1, ..., N-1$, we can select a control action $u(x_k)$, where x_k is the state at that time.
- Assume that $u(x_k)$ is chosen from a given set of admissible control actions $U(x_k)$ for that state.
- Depending on the state x_k and the control action selected $u(x_k)$, we incur a cost $C[x_k, u(x_k)]$.
- Then, a state transition occurs according to a state equation of the form

$$\boldsymbol{x_{k+1} = f_k(x_k, u_k)}$$

SOLVING MDPs
The Basics of Dynamic Programming

- A policy π is a sequence $\{u_0, u_1, \ldots, u_{N-1}\}$ of control actions over the time horizon N.
- The optimization problem of interest is to determine a policy $\pi = \{u_0, u_1, \ldots, u_{N-1}\}$ that minimizes the total cost

$$V_\pi(x_0) = C(x_N) + \sum_{k=0}^{N-1} C[x_k, u(x_k)]$$

- We denote such an optimal policy by $\pi^* = \{u^*_0, u^*_1, \ldots, u^*_{N-1}\}$

SOLVING MDPs
The Basics of Dynamic Programming

- Suppose we apply policy π^* and have reached state x_n, and define the cost-to-go

$$V_\pi(x_n) = C(x_N) + \sum_{k=n}^{N-1} C[x_k, u(x_k)], \qquad n = 1,\ldots,N-1$$

- Denote the optimizing policy for this subproblem as

$$\pi^0 = \{u^0_n, u^0_{n+1}, \ldots, u^0_{N-1}\}.$$

- What is the optimal policy for this subproblem?

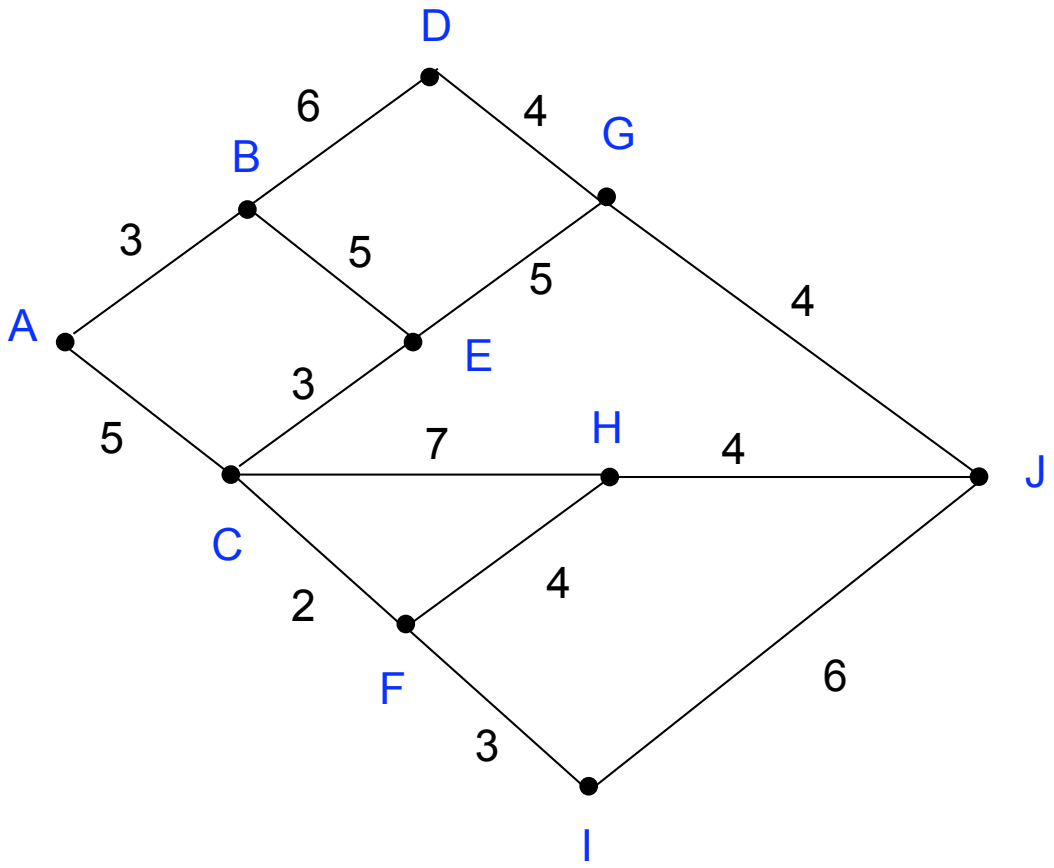
- The answer is based on Bellman's principle of optimality:

i.e., $u^0_n = u^*_n$, $u^0_{n+1} = u^*_{n+1}$, …, $u^0_{N-1} = u^*_{N-1}$.

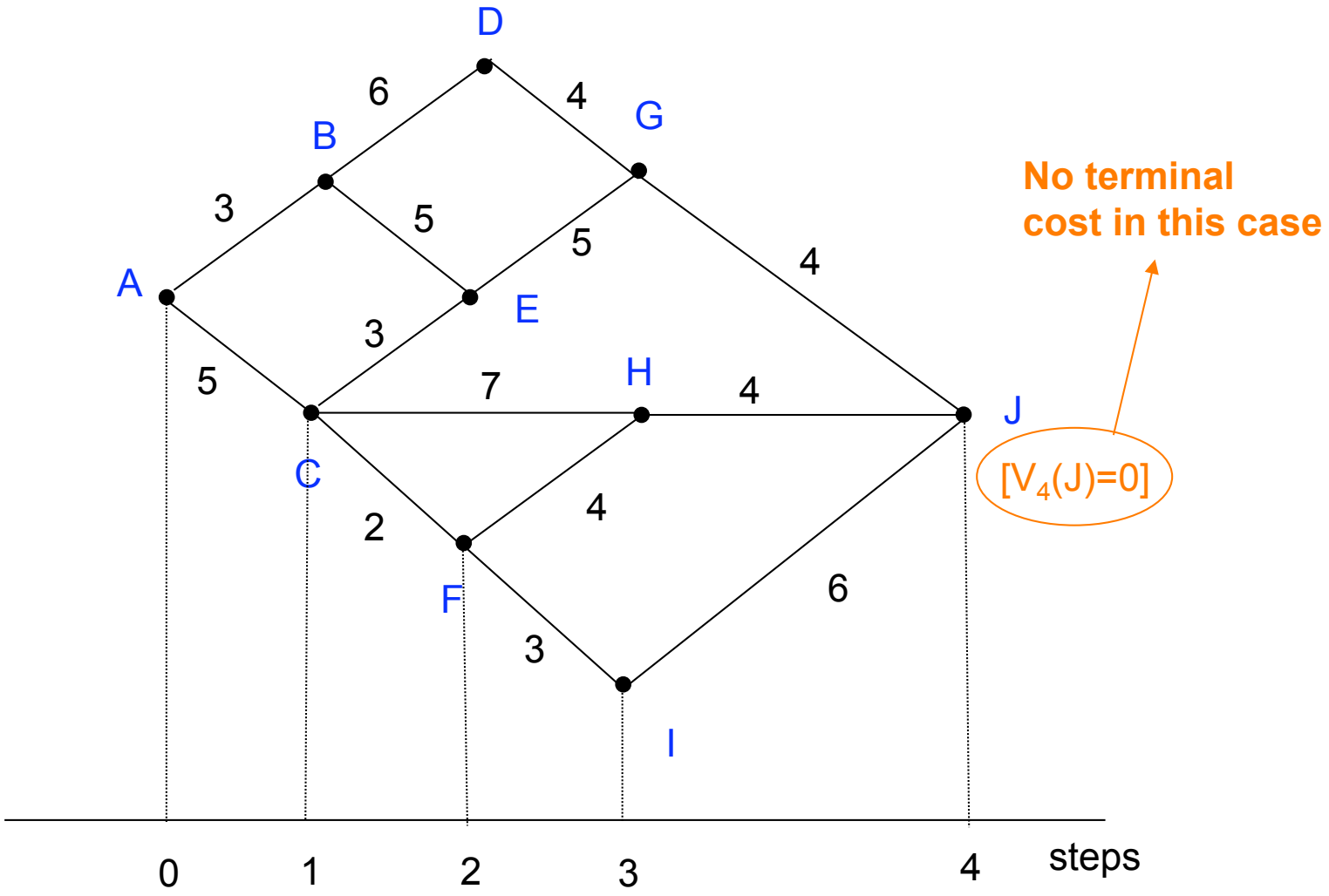
if the policy function is optimal for the finite summation, then it must be the case that whatever the initial state and decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from that first decision (as expressed by the Bellman equation).

SOLVING MDPs

The Basics of Dynamic Programming

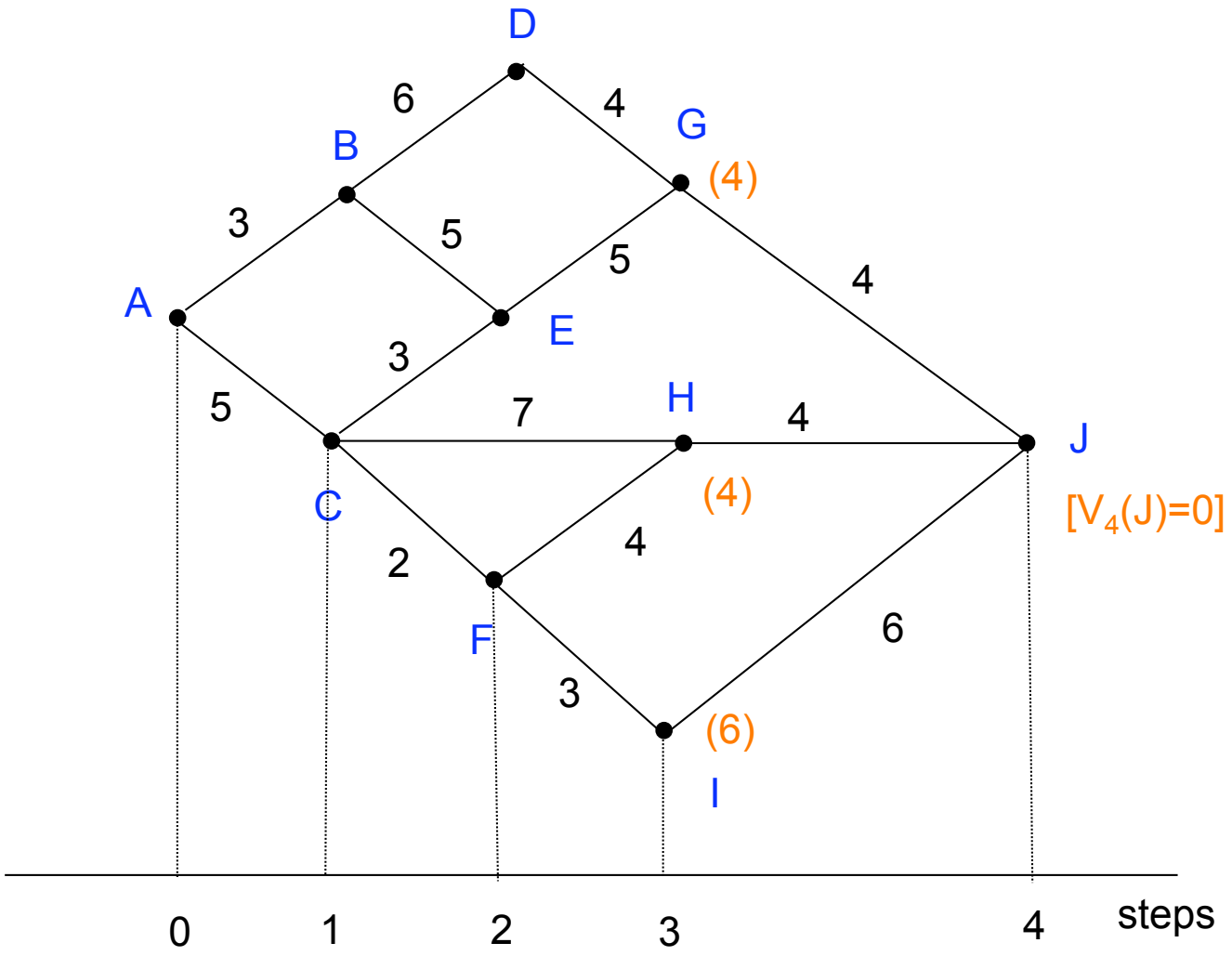


SOLVING MDPs
The Basics of Dynamic Programming



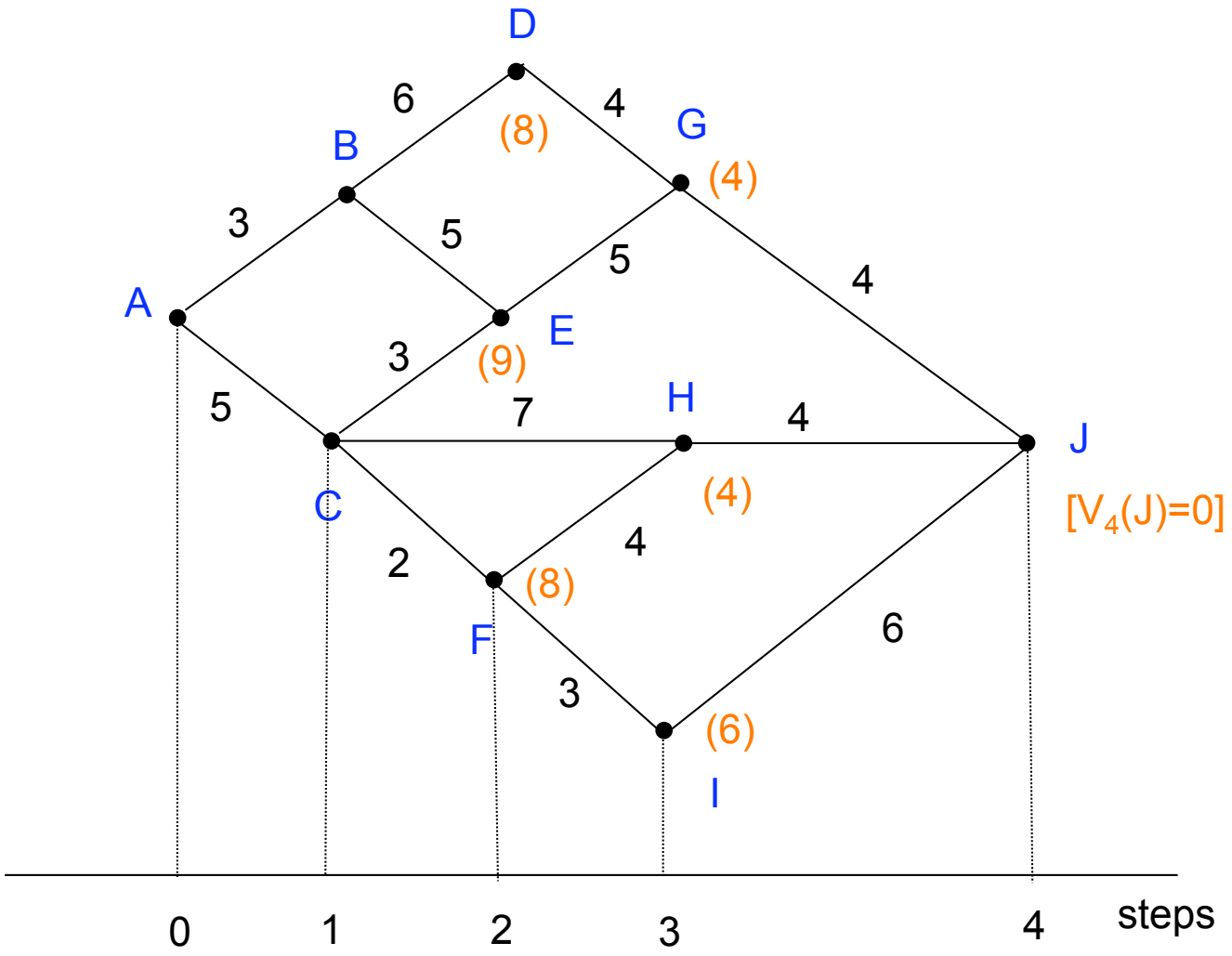
SOLVING MDPs

The Basics of Dynamic Programming



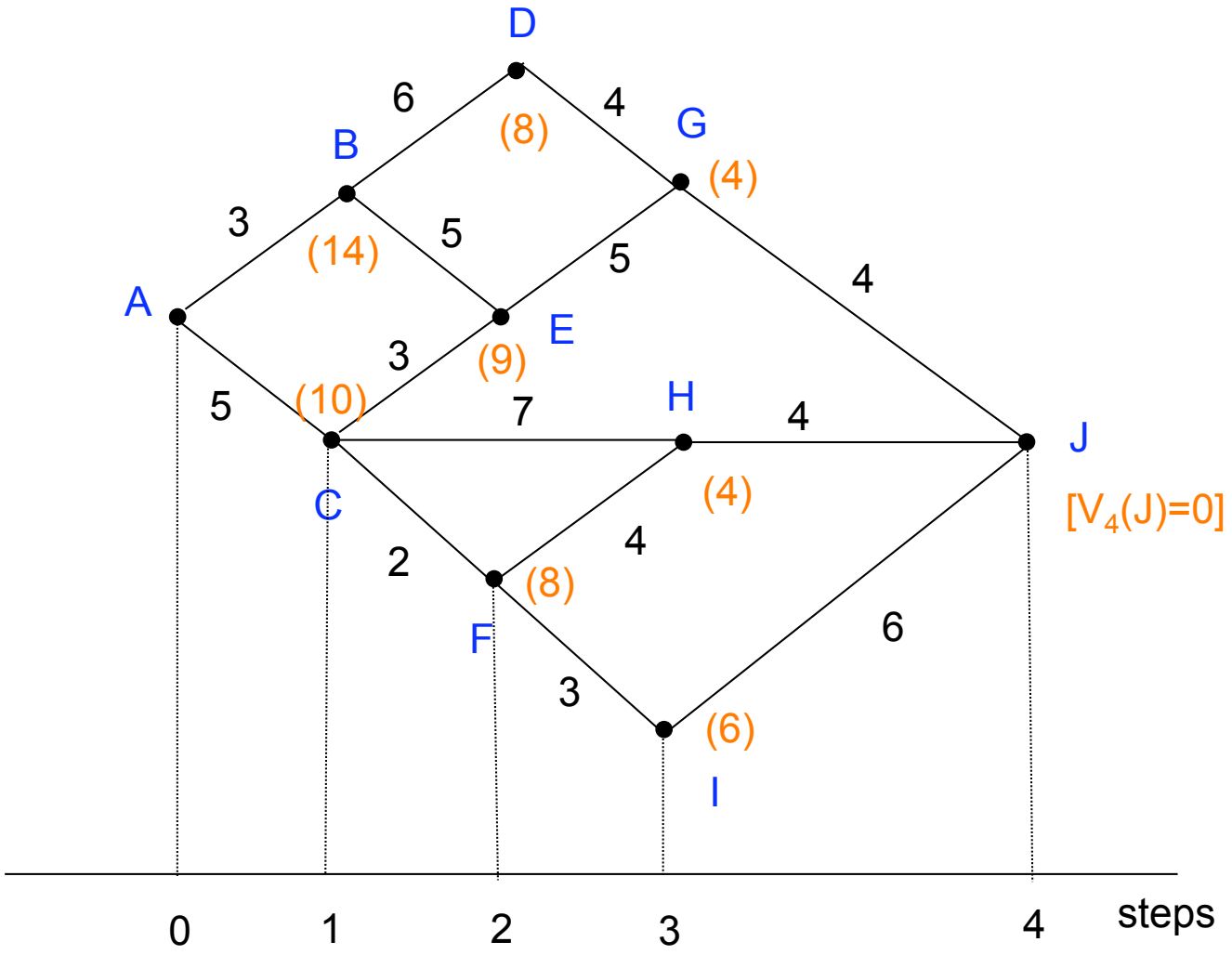
SOLVING MDPs

The Basics of Dynamic Programming



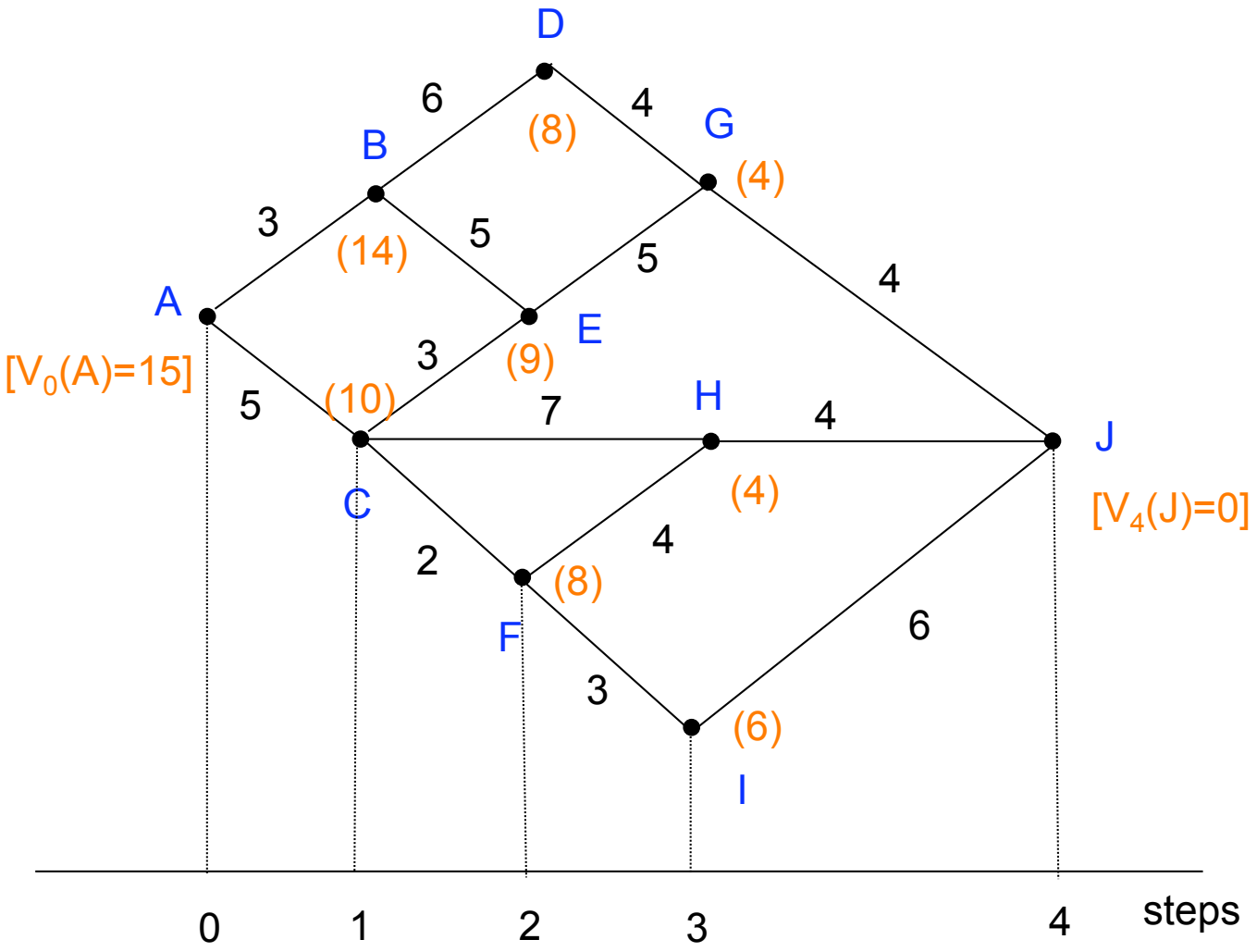
SOLVING MDPs

The Basics of Dynamic Programming



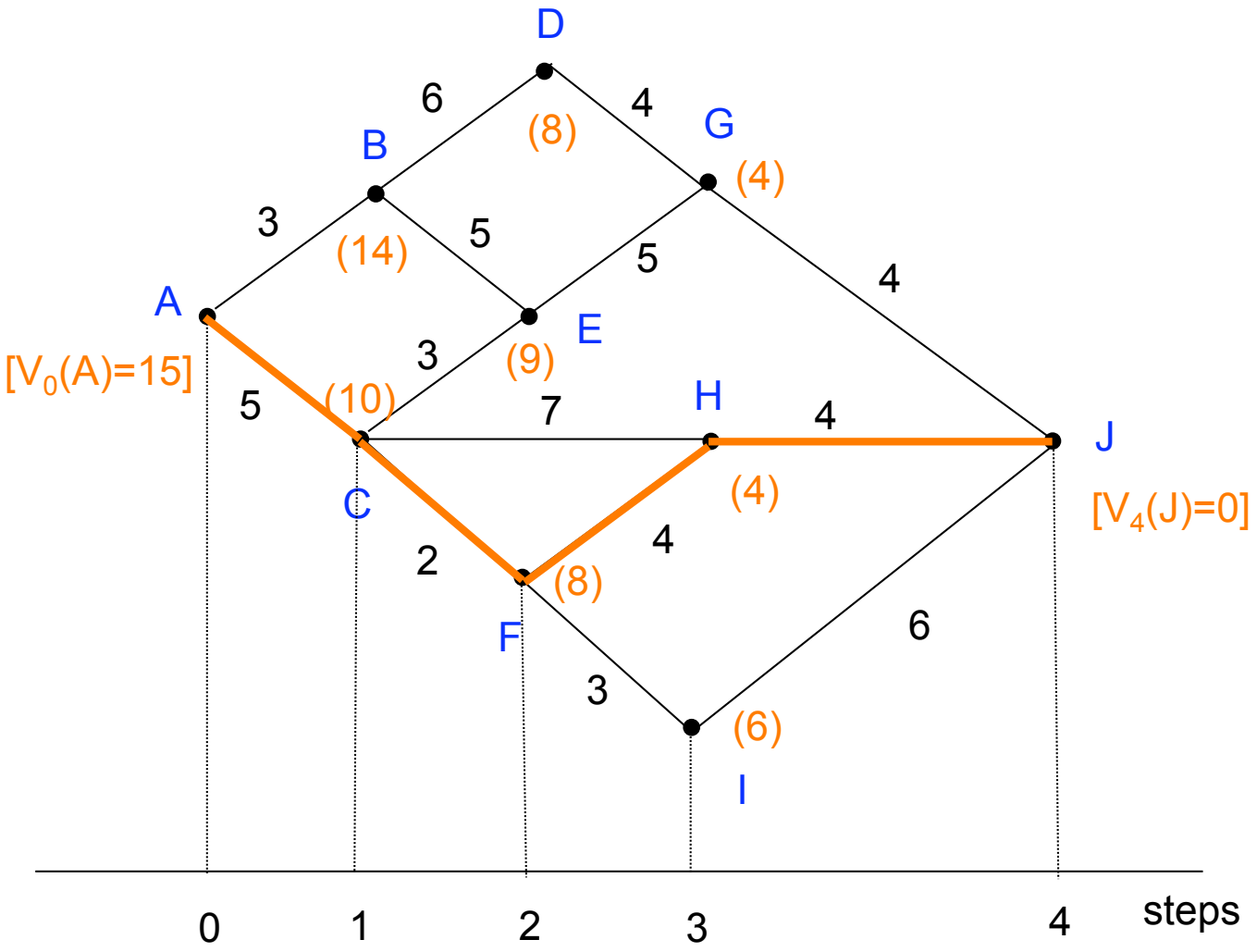
SOLVING MDPs

The Basics of Dynamic Programming



SOLVING MDPs

The Basics of Dynamic Programming



SOLVING MDPs

The Basics of Dynamic Programming

In the previous simple example, each state at one stage only has connections (with a given cost) to states at an upper stage.

Iteration is on the number of stages.

In Markov Chains, each state can be connected by transition probabilities to other states, which in turn can even connect back to it as in a graph.

Therefore, the update of each state value must be done for all states at a time, and the iteration is on the time steps.

SOLVING MDPs
The Basics of Dynamic Programming

- Let $V^*_\pi(x_0)$ denote the optimal cost, attained when the optimal policy is used

$$V^*_\pi(x_0) = \min_\pi \left[C(x_N) + \sum_{k=0}^{N-1} C[x_k, u(x_k)] \right]$$

- Note that

$$V^*_0(x_0) = \min_{\{u_0, u_1, \ldots, u_{N-1}\}} \left[C[x_0, u(x_0)] + C(x_N) + \sum_{k=1}^{N-1} C[x_k, u(x_k)] \right]$$

$$V^*_0(x_0) = \min_{u_0} \left\{ C[x_0, u(x_0)] + \min_{\{u_1, u_2, \ldots, u_{N-1}\}} \left[C(x_N) + \sum_{k=1}^{N-1} C[x_k, u(x_k)] \right] \right\}$$

$$V^*_0(x_0) = \min_{u_0} \left\{ C[x_0, u(x_0)] + V^*_1(f[x_1, u(x_1)]) \right\}$$

SOLVING MDPs
The Basics of Dynamic Programming

In general, we can set the following recursive set of optimization problems

$$V_k(x_k) = \min_{u_k \in U(x_k)} \left[C(x_k, u_k) + V_{k+1}(x_{k+1}) \right] \qquad k = 0, \ldots, N-1$$

with the following boundary condition

$$V_N(x_N) = C(x_N)$$

- The quantity $V_k(x_k)$ is termed the optimal cost-to-go from state x_k in the following sense
  - The controller sees a *single-step* problem with step cost $C(x_k, u_k)$ and terminal cost $V_{k+1}(x_{k+1})$ and selects the control action that solves that problem.
  - $V_{k+1}(x_{k+1})$ has to be known before solving $V_k(x_k)$
  - The equations are solved backward and ultimately $V^*(x_0) = V_0(x_0)$

SOLVING MDPs

Dynamic Programming and the Optimality Equation

For a discrete-time Markov chain (*non-deterministic setting*) we have

$$V_\pi(i) = E_\pi\left[\sum_{k=0}^{\infty} \alpha^k C(X_k, u_k)\right] \qquad (1)$$

which originates the following recursion

$$V_k'(j) = \min_{u \in U_j}\left[\alpha^k C(j,u) + \sum_{\text{all } r} p_{jr}(u) V_{k+1}'(r)\right], \qquad k = 0,\ldots,N-1$$

$$V_N'(j) = 0 \qquad \text{for all } j$$

Dividing both terms by α^k and defining $V''_k = \alpha^{-k} V'_k$, we get

$$V_k''(j) = \min_{u \in U_j}\left[C(j,u) + \alpha \sum_{\text{all } r} p_{jr}(u) V_{k+1}''(r)\right], \qquad k = 0,\ldots,N-1$$

$$V_N''(j) = 0 \qquad \text{for all } j$$

SOLVING MDPs
Dynamic Programming and the Optimality Equation

Finally, defining $V_k = V''_{N-k}$ we get

Recursion is now in the forward direction

$$V_0(j) = 0 \qquad \text{for all } j$$

$$V_{k+1}(j) = \min_{u \in U_j} \left[C(j,u) + \alpha \sum_{\text{all } r} p_{jr}(u) V_k(r) \right], \qquad k = 0,\ldots,N-1$$

- The above establishes an interesting framework for the DP algorithm.

  It clearly defines an operator structure.

$$T[f(j)] = \min_{u \in U_j} \left[C(j,u) + \alpha \sum_{\text{all } r} p_{jr}(u) f(r) \right]$$

$$T^k[f(j)] = T[T^{k-1}[f(j)]] \qquad V_1(j) = T[V_0(j)] \quad V_2(j) = T[T[V_0(j)]] \quad \cdots \quad V_k(j) = T^k[V_0(j)]$$

SOLVING MDPs

Dynamic Programming and the Optimality Equation

With the operator structure above defined, the recursion assumes the following format

$$V_0(j) = 0 \qquad \text{for all } j$$
$$V_k(j) = T^k[V_0(j)], \qquad k = 0,\ldots,N-1 \tag{2}$$

- Property

  If $f(j) = g(j) + \varepsilon$, $T[f(j)] = T[g(j)] + \alpha\varepsilon$

- Consequence

  $T^k[f(j)] = T^k[g(j)] + \alpha^k\varepsilon$, for $k = 1,2,\ldots$

SOLVING MDPs
Dynamic Programming and the Optimality Equation

Lemma 1

Under the assumption $0 \le C(j, u) \le K$ for all states j and control actions $u \in U_j$, the solution $V_N(i)$ of the DP algorithm is such that

$$\lim_{N \to \infty} V_N(i) = V^*(i)$$

Lemma 2

Under the assumption $0 \le C(j, u) \le K$ for all states j and control actions $u \in U_j$, for any bounded function $f(i)$, f: $X \to \Re$, we have

$$\lim_{N \to \infty} T^N[f(i)] = V^*(i)$$

SOLVING MDPs
The Basics of Dynamic Programming

Theorem 1

Under the assumption $0 \leq C(j, u) \leq K$ for all states j and control actions $u \in U_j$, the optimal cost $V^*(i)$ which minimizes (1) satisfies the optimality equation

$$V(i) = \min_{u \in U_i} \left[C(i,u) + \alpha \sum_{\text{all } j} p_{ij}(u) V(j) \right]$$

SOLVING MDPs
The Unbounded and Undiscounted Cases

Extensions to Unbounded and Undiscounted Costs

Theorem 2

Under the assumption $C(j, u) \geq 0$ [or $C(j, u) \leq 0$] for all states j and control actions $u \in U_j$, the optimal cost $V^*(i)$ which minimizes (1) satisfies the optimality equation

$$V(i) = \min_{u \in U_i} \left[C(i,u) + \alpha \sum_{\text{all } j} p_{ij}(u) V(j) \right]$$

where α is no longer constrained to be in the interval (0, 1), but is allowed to take values greater than or equal to 1.

Unlike Theorem 1, we can no longer assert here that V*(i) is the only solution of the optimality equation, although in most cases of practical interest this does not turn out to pose a serious limitation.

INSTITUTO SUPERIOR TÉCNICO

SOLVING MDPs

The Unbounded and Undiscounted Cases

Theorem 3

Assume that the set of control actions U is finite. Then, under the assumption $C(j, u) \geq 0$ for all states j and control actions $u \in U_j$, we have

$$\lim_{N \to \infty} V_N(i) = V^*(i)$$

where $V_N(i)$ is the solution of the DP algorithm (2).

SOLVING MDPs

Optimization of the Average Cost Criterion

The Average Cost Criterion is rewritten for the equivalent DTMC as follows

$$V_\pi(x_0) = \lim_{N\to\infty} \frac{1}{N} E_\pi \left[\sum_{k=0}^{N-1} C[X_k, u_k] \right] \qquad (3)$$

Theorem 4

Suppose there exists a constant ν and a bounded function h(i) which satisfy the following equation

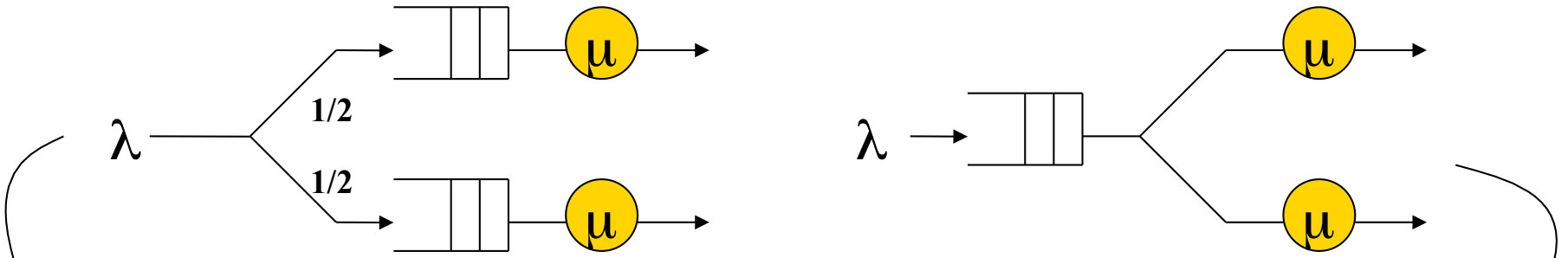
$$\nu + h(i) = \min_{u\in U_i} \left[C(i,u) + \sum_{\text{all } j} p_{ij}(u) h(j) \right] \qquad (4)$$

Then, ν is the optimal cost in (3), that is,

$$\nu = \min_\pi V_\pi(i) = V^*(i)$$

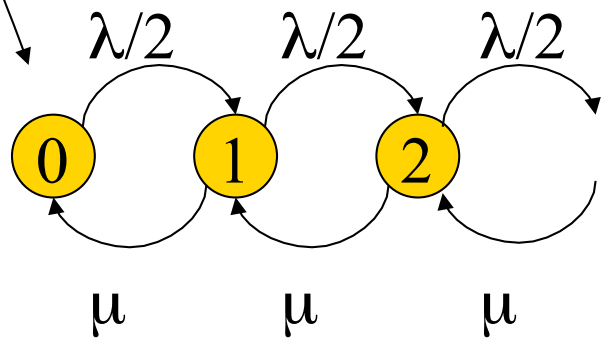
and a stationary policy π^* is optimal if it gives the minimum value in (4) for all states *i*.

CONTROL OF QUEUEING SYSTEMS



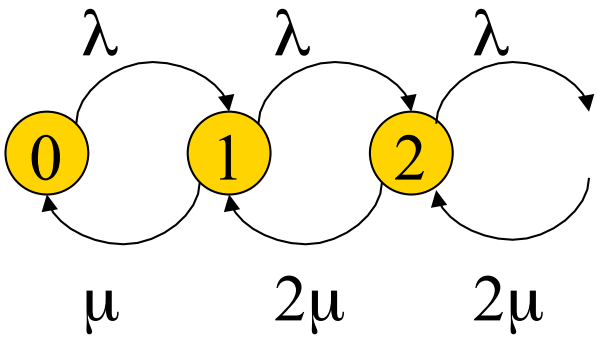
What is the best arrangement?

client immediately routed to a queue with prob 1/2



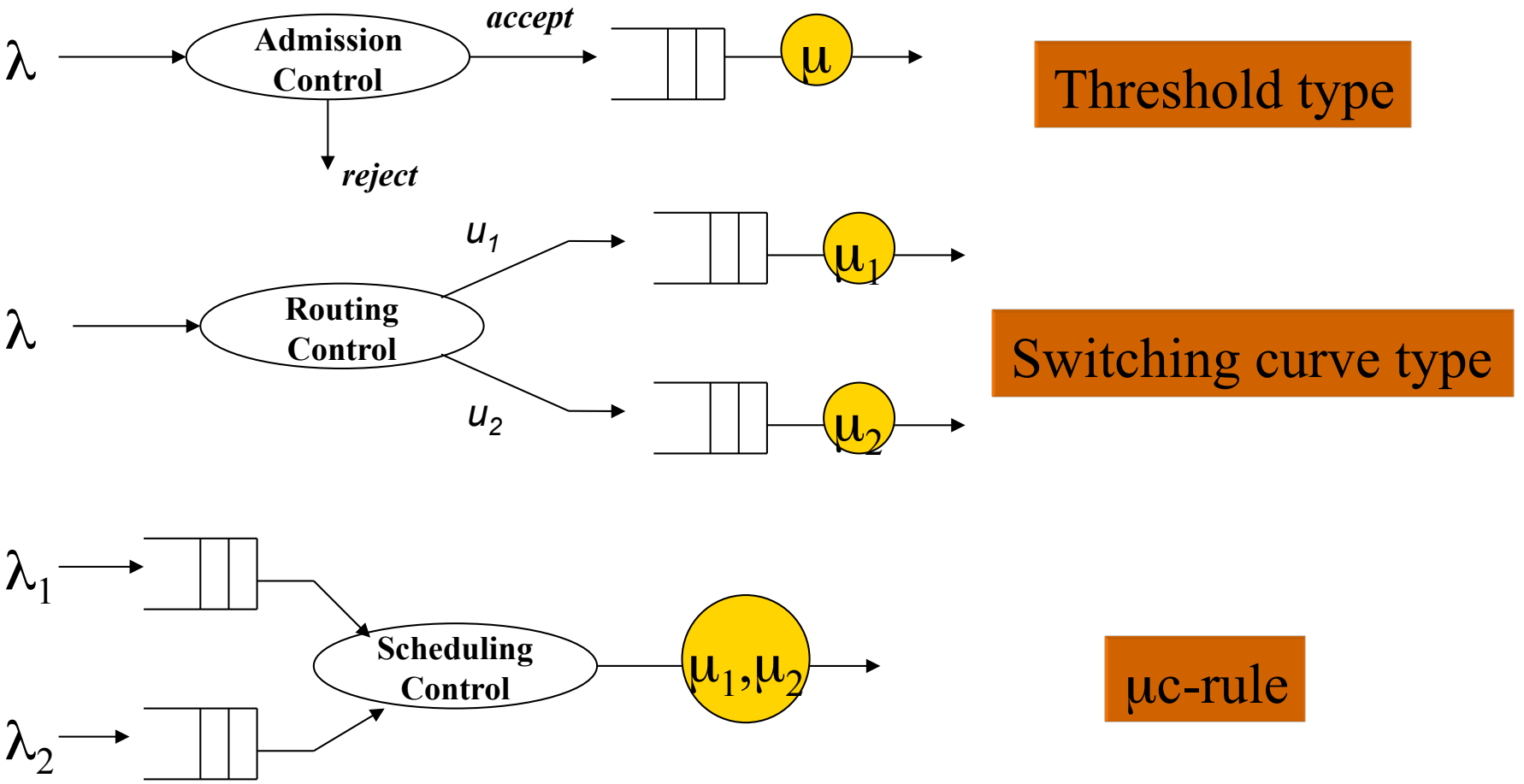
Two queues of this

committing as late as possible to one of the servers



One of this – smaller average system time

CONTROL OF QUEUEING SYSTEMS



INSTITUTO SUPERIOR TÉCNICO

CONTROLLED MARKOV CHAINS

Further reading

- Queuing systems and its control
- Bertsekas, D. P., *Dynamic Programming: Deterministic and Stochastic Models*, Prentice-Hall, Englewood Cliffs, NJ, 1987
- Bertsekas, D. P., *Dynamic Programming and Optimal Control*, Vols 1 and 2, Athena Scientific, Belmont, MA, 1995