数理解析研究所講究録 1461 巻 2005 年 202-213

A structured pattern matrix algorithm for multichain Markov decision processes

宮崎大学・教育文化学部　伊喜　哲一郎 (Tetsuichiro IKI)
Faculty of Education and Culture, Miyazaki University
東京電機大学・情報環境学部　堀口　正之 (Masayuki HORIGUCHI)
School of Information Environment, Tokyo Denki University
千葉大学・教育学部　蔵野　正美 (Masami KURANO)
Faculty of Education, Chiba University

Abstract

In this paper, we are concerned with a new algorithm for multichain finite state Markov decision processes which finds an average optimal policy through the decomposition of the state space into some communicating classes and a transient class. For each communicating class, a relatively optimal policy is found, which is used to find an optimal policy by applying the value iteration algorithm. Using a pattern matrix determining the behaviour pattern of the decision process, the decomposition of the state space is effectively done, so that the proposed algorithm simplifies the structured one given by the excellent Leizarowitz's paper (Mathematics of Operations Research, **28**, 2003, 553–586). Also, a numerical example is given to comprehend the algorithm.

Key words: multichain Markov decision processes, structured algorithm, communicating class, transient class, value iteration.

1 Introduction and notation

In this paper, we are concerned with the structured pattern matrix algorithm for multichain finite state Markov decision processes (MDPs) with an average reward criterion. The efficient algorithm for finding an optimal policy in average reward MDPs have been studied by many authors (cf. [5, 8, 11, 12, 14]). For the unichain or communicating case, the optimal policy can be found by solving a single optimality equation (cf. [11]). However, for the multichain case, the optimal policy is characterized by two equations, called the multichain optimality equations, which are solved, for example, by linear programming (cf. [6, 11]) or policy iteration algorithms (cf. [7, 8]).

Recently, Leizarowitz[10] have developed the new algorithm for multichain average reward MDPs, in which the problem is broken into several communicating models and a transient model and for each model, an optimal policy is found by solving the corresponding single optimality equation. The objective of this paper is to propose a structured algorithm for the multichain finite state MDPs which simplify the Leizarowitz's suggestive algorithm by use of pattern matrix representation of transition probabilities in MDPs and the idea of the value iteration algorithm (cf. [5, 11]). That is, by the convenience and simplicity of pattern matrix, we can easily classify the set of states into

several maximum communicating classes and a transient class and easily find the maximum sub-MDPs. In addition, the finding of an optimal policy is done by applying the value iteration algorithm.

In the reminder of this section, we will define finite state MDPs to be examined and describe the basic results for the communicating case.

We define a controlled dynamic system with a finite state space denoted by $S = \{1, 2, \ldots, N\}$. Associated with each state $i \in S$ is a non-empty finite set $A(i)$ of available actions. When the system is in state $i \in S$ and action $a \in A(i)$ is taken, then the system moves to a new state $j \in S$ with probability $q_{ij}(a)$ where $\sum_{j \in S} q_{ij}(a) = 1$ for all $i \in S$ and $a \in A(i)$ and the reward $r(i, a)$ is earned. The process is repeated from the new state $j \in S$. This structure is called a Markov decision process, denoted by an MDP $\Gamma = (S, \{A(i) : i \in S\}, Q, r)$, where $Q = (q_{ij}(a) : i, j \in S, a \in A(i))$ and $r = r(i, a) \in \mathbb{R}$ for all $i \in S$ and $a \in A(i)$ and $\mathbb{R}$ is the set of all real numbers. The set of admissible state-action pairs will be denoted by

$$\mathbb{K} = \{(i, a) : i \in S, a \in A(i)\}.$$

The sample space is the product space $\Omega = \mathbb{K}^\infty$ such that the projection (X_t, Δ_t) on the t-th factor describes the state and action at the t-th time of the process ($t \geqq 0$). A policy $\pi = (\pi_0, \pi_1, \ldots)$ is a sequence of conditional probabilities with $\pi_t(A(x_t)|x_0, a_0, \ldots, x_t) = 1$ for all histories $(x_0, a_0, \ldots, x_t) \in \mathbb{K}^t S (t \geqq 0)$ where $\mathbb{K}^0 S = S$. The set of all policies is denoted by Π. A policy $\pi = (\pi_0, \pi_1, \ldots)$ is called randomized stationary if a conditional probability $\gamma = (\gamma(\cdot|i) : i \in S)$ given S for which $\pi(\cdot|x_0, a_0, \ldots, x_t) = \gamma(\cdot|x_t)$ for all $t \geqq 0$ and $(x_0, a_0, \ldots, x_t) \in \mathbb{K}^t S$. Such a policy is simply denoted by γ. We denote by F the set of functions on S with $f(i) \in A(i)$ for all $i \in S$. A randomized stationary policy γ is called stationary if there exists a function $f \in F$ such that $\gamma(\{f(i)\}|i) = 1$ for all $i \in S$, which is denoted simply by f. For each $\pi \in \Pi$, starting state $X_0 = i$, the probability measure $P_\pi(\cdot|X_0 = i)$ on Ω is defined in an usual way. The problem we are concerned with is the maximization of the long-run expected average reward per unit time, which is defined by

$$\psi(i, \pi) = \liminf_{T \to \infty} \frac{1}{T} E_\pi \left(\sum_{t=0}^{T-1} r(X_t, \Delta_t) \,\middle|\, X_0 = i \right), \tag{1.1}$$

where $E_\pi(\cdot|X_0 = i)$ is the expectation w.r.t. $P_\pi(\cdot|X_0 = i)$.

A policy $\pi^* \in \Pi$ satisfying that

$$\psi(i, \pi^*) = \sup_{\pi \in \Pi} \psi(i, \pi) := \psi^*(i) \quad \text{for all } i \in S \tag{1.2}$$

is called to be average optimal or simply optimal.

The structured algorithm treated with in this paper is based on a communicating model introduced by Bather[1]. We say that the MDP Γ is communicating if there exists a randomized stationary policy $\gamma = (\gamma(\cdot|i) : i \in S)$ satisfying that the transition matrix $Q(\gamma)$ induced by γ defines a irreducible Markov chain where $Q(\gamma) = (q_{ij}(\gamma))$ with $q_{ij}(\gamma) = \sum_{a \in A(i)} q_{ij}(a)\gamma(a|i)$ for all $i, j \in S$. The following fact is well known (cf. [1, 11]).

Lemma 1.1. *For the communicating MDP* Γ, *there exists a constant* g *and a vector* $v = (v_i : i \in S)$ *such that*

$$v_i + g = \max_{a \in A(i)} \{r(i,a) + \sum_{j \in S} q_{ij}(a) v_j\} \quad (i \in S) \tag{1.3}$$

and any stationary policy f^* *such that* f^* *attains the maximum of (1.3) is optimal with* $g = \psi(i, f^*) = \sup_{\pi \in \Pi} \psi(i, \pi)$ *for all* $i \in S$.

We note that the algorithm for finding an optimal policy in Lemma 1.1 are given as the value iteration or policy iteration algorithms (cf. [11]).

In Section 2, the methods of classifying the set of states by use of the corresponding pattern matrices is proposed, which is used to find a optimal policy by the value iteration algorithm in Section 3. In Section 4, a numerical example is given to comprehend our structured algorithm.

2 Pattern matrices and state classification

In this section, we define a pattern matrix for the transition matrix of the MDPs which is used to classify the state space and find the maximum sub-MDPs.

For a non-empty subset D of S, if, for each $i \in D$, there exists a non-empty subset $\overline{A}(i) \subset A(i)$ with $\sum_{j \in D} q_{ij}(a) = 1$ for all $a \in \overline{A}(i)$, we can consider the sub-MDP with the restricted state space D and available action space $\overline{A}(i)$ for $i \in D$, which is denoted by $\overline{\Gamma} = (D, \{\overline{A}(i) : i \in D\}, Q_D, r_D)$ where Q_D and r_D are restriction of Q and r on $\{(i,a) : i \in D, a \in \overline{A}(i)\}$. For any sub-MDP $\overline{\Gamma} = (D, \{\overline{A}(i) : i \in D\}, Q_D, r_D)$, a non-empty subset $\overline{D}$ of D is a communicating class in $\overline{\Gamma}$ if $\overline{D}$ is closed, that is, $\sum_{j \in \overline{D}} q_{ij}(a) = 1$ for all $i \in \overline{D}$ and $a \in \overline{A}(i)$ and the sub-MDP $(\overline{D}, \{\overline{A}(i) : i \in \overline{D}\}, Q_{\overline{D}}, r_{\overline{D}})$ is communicating. Also, $\overline{D}$ is maximum communicating class if it is not strictly contained in any communicating class.

For any positive integer l, let $\mathbb{C}^l$ and $\mathbb{C}^{l \times l}$ be the sets of l-dimensional row vectors and $l \times l$ matrices with $\{0,1\}$-valued elements, respectively. The sum $(+)$ and product $(\cdot)$ operators among elements in $\mathbb{C}^l$ and $\mathbb{C}^{l \times l}$ is defined by $a + b = \max\{a, b\}$ and $a \cdot b = \min\{a, b\}$ for $a, b \in \{0, 1\}$.

For each $i \in S$ and $a \in A$, we define $m_i(a) = (m_{i1}(a), m_{i2}(a), \ldots, m_{iN}(a)) \in \mathbb{C}^N$ by

$$m_{ij}(a) = \begin{cases} 1 & \text{if } q_{ij}(a) > 0, \\ 0 & \text{if } q_{ij}(a) = 0, \end{cases} \quad (j \in S). \tag{2.1}$$

For a sub-MDP $\overline{\Gamma} = (D, \{\overline{A}(i) : i \in D\}, Q_D, r_D)$ with $D = \{i_1, i_2, \ldots, i_l\}$, we put $m_i(\overline{\Gamma}) = (m_{ii_1}(\overline{\Gamma}), m_{ii_2}(\overline{\Gamma}),$
$\ldots, m_{ii_l}(\overline{\Gamma})) = \sum_{a \in \overline{A}(i)} m_i(a|\overline{\Gamma})$ $(i \in D)$, where $m_i(a|\overline{\Gamma}) = (m_{ii_1}(a), m_{ii_2}(a), \ldots, m_{ii_l}(a))$ $(i \in D, a \in \overline{A}(i))$. Using $m_i(\overline{\Gamma})$ $(i \in D)$, we define a pattern matrix $M(\overline{\Gamma})$ for the sub-MDP

$\overline{\Gamma}$ by

$$M(\overline{\Gamma}) = \begin{pmatrix} m_{i_1}(\overline{\Gamma}) \\ \vdots \\ m_{i_l}(\overline{\Gamma}) \end{pmatrix} \in \mathbb{C}^{l\times l}. \tag{2.2}$$

We note that for $i, j \in D$, $m_{ij}(\overline{\Gamma}) = 1$ means that there exists $a \in \overline{A}(i)$ with $q_{ij}(a) > 0$.

The pattern matrix defined above is called a communication matrix (cf. [10]). However, since $M(\overline{\Gamma})$ determines the behaviour pattern of the sub-MDP $\overline{\Gamma}$, we call it a pattern matrix in this paper. For the pattern matrix $M(\overline{\Gamma})$, we define $\overline{M}(\overline{\Gamma}) \in \mathbb{C}^{l\times l}$ by

$$\overline{M}(\overline{\Gamma}) = \sum_{k=1}^{N} M(\overline{\Gamma})^k. \tag{2.3}$$

Then, we have the following.

Lemma 2.1. *For a non-empty subset $\overline{D}$ of D, $\overline{D}$ is a communicating class if and only if, for each $i \in \overline{D}$,*

$$\overline{m}_{ij}(\overline{\Gamma}) = \begin{cases} 1 & \text{if } j \in \overline{D}, \\ 0 & \text{if } j \notin \overline{D}, \end{cases}$$

where $\overline{m}_{ij}(\overline{\Gamma})$ is the (i,j) element of $\overline{M}(\overline{\Gamma})$ and $\overline{M}(\overline{\Gamma}) = (\overline{m}_{ij}(\overline{\Gamma}))$.

Proof. See Appendix.∎

By reordering the states in D, $\overline{M}(\overline{\Gamma})$ can be transformed to the standard form:

$$\overline{M}(\overline{\Gamma}) = \begin{pmatrix} E_1 & & & \\ & E_2 & & 0 \\ & & \ddots & \\ & 0 & & E_d \\ \hline & R & & K \end{pmatrix} \quad (d \geqq 1), \tag{2.4}$$

where E_j is a square matrix whose elements are all 1 ($1 \leqq j \leqq d$) and $[R\ K]$ does not include a sub-matrix having the above form.

For any sets U, V, if $U \cap V = \emptyset$, we write the union $U \cup V$ by $U + V$.

Here, we get the classification of the state space.

Theorem 2.1. *For a sub-MDP $\overline{\Gamma} = (D, \{\overline{A}(i) : i \in D\}, Q_D, r_D)$, the state space D is classified as follows:*

$$D = U_1 + U_2 + \cdots + U_d + L \quad (d \geqq 1), \tag{2.5}$$

where U_j is a communicating class for $\overline{\Gamma}$ ($1 \leqq j \leqq d$) and L is not closed.

Proof. See Appendix.∎

The algorithm of obtaining the state-decomposition (2.5) through (2.4) by use of the pattern matrix will be called **Algorithm A**.

When the not-closed class L in (2.5) is non-empty, we go on with the state classification of L by finding the maximum sub-MDP. The basic idea of the following algorithm is the same as in [10], called **Algorithm B** hereafter.

Algorithm B:

1. Set $K_1 = L$.
2. Suppose that $\{K_i : 1 \leqq i \leqq n\}$ with $K_1 \supsetneqq K_2 \supsetneqq \cdots \supsetneqq K_n$ $(K_n \neq \emptyset)$ is given. Then, set $V = S - K_n$ and each $i \in K_n$, set $A_1(i) = A(i) - T(i)$, where $T(i) = \{a \in A(i) | \sum_{j \in V} m_{ij}(a) = 1\}$. Set $K_{n+1} = \{i \in K_n | A_1(i) \neq \emptyset\}$.
3. The following three cases happen:

Case 1: $K_{n+1} \neq \emptyset$ and $K_n \supsetneqq K_{n+1}$.
For this case, put $n = n + 1$ and go to Step 2 with $\{K_i : 1 \leqq i \leqq n + 1\}$.

Case 2: $K_{n+1} = K_n$.
For this case, set $\overline{D} := K_{n+1}$. Then, $\overline{\Gamma} = \{\overline{D}, \{A_1(i) : i \in \overline{D}\}, Q_{\overline{D}}, r_{\overline{D}}\}$ is a maximum sub-MDP in L. Apply **Algorithm A** for this sub-MDP $\overline{\Gamma}$.

Case 3: $K_{n+1} = \emptyset$. *Stop the algorithm.*
For this case, the set L does not include any sub-MDP and it holds that

$$\max_{i \in L, \pi \in \Pi} P_\pi(X_n \in L | X_0 = i) < 1. \tag{2.6}$$

Starting with the MDP $\Gamma = (S, \{A(i) : i \in S\}, Q, r)$ given firstly, we apply **Algorithm A** and **B** iteratively, so that we get the following.

Theorem 2.2. *Let $\Gamma = (S, \{A(i) : i \in S\}, Q, r)$ be the fixed MDP. Then, there exists a sequence of sub-MDPs*

$$\Gamma_k = (S_k, \{A_k(i) : i \in S_k\}, Q_{S_k}, r_{S_k}) \quad (k = 0, 1, \ldots, n^*)$$

satisfying the following (i)-(ii).

(i) $S_0 = S \supsetneqq S_1 \supsetneqq \cdots \supsetneqq S_{n^*}$

(ii) The state space S is decomposed to:

$$S = U_0 + U_1 + \cdots + U_{n^*} + L, \tag{2.7}$$

where

$$U_k = U_{k1} + U_{k2} + \cdots + U_{kl_k} \quad (0 \leqq k \leqq n^*), \tag{2.8}$$

U_{kj} $(1 \leqq j \leqq l_k)$ is a maximum communicating class (m.c.c.) for the sub-MDP Γ_k, and L is a transient class, that is, for any $i \in L$ and $a \in A(i)$, there exists an integer $n \geqq 1$ such that

$$\max_{i \in L, \pi \in \Pi} P_\pi(X_n \in L | X_0 = i) < 1. \tag{2.9}$$

Applying Lemma 1.1 to each communicating sub-MDP $\Gamma_{kj} = (U_{kj}, \{A_k(i) : i \in U_{kj}\}, Q_{U_{kj}}, r_{U_{kj}})$, we get an optimal stationary policy f_{kj} and an optimal average reward g_{kj} for sub-MDP Γ_{kj}, called relatively o.p. and relatively a.r., respectively, which is summarized in Table 2.1.

m.c.c.	$U_{01}, \ldots, U_{0l_0}$	$U_{11}, \ldots, U_{1l_1}$	$\cdots$	$U_{n^*1}, \ldots, U_{n^*l_{n^*}}$
relatively o.p.	$f_{01}, \ldots, f_{0l_0}$	$f_{11}, \ldots, f_{1l_1}$	$\cdots$	$f_{n^*1}, \ldots, f_{n^*l_{n^*}}$
relatively a.r.	$g_{01}, \ldots, g_{0l_0}$	$g_{11}, \ldots, g_{1l_1}$	$\cdots$	$g_{n^*1}, \ldots, g_{n^*l_{n^*}}$

Table 2.1: Relatively o.p. and a.r.

We note that a stationary policy f_{0j} is absolutely optimal on U_{0j} $(1 \leqq j \leqq l_0)$ because optimization is done in the MDP Γ.

3 Algorithm for obtaining an optimal policy

In this section, we give a value iterative algorithm based on data in Table 2.1 to find an (absolutely) optimal policy for the MDP Γ.

Let $K_L := \{(i,a) | i \in L \text{ and } a \in A(i)\}$ and $B(L) = \{v | v : L \to \mathbb{R}\}$. For any function d on K_L, the map $T_d : B(L) \to B(L)$ will be defined as

$$T_d v(i) = \max_{a \in A(i)} \{d(i,a) + \sum_{j \in L} q_{ij}(a) v(j)\} \quad (i \in L). \tag{3.1}$$

We need the following lemma to treat with the transient class L in Theorem 2.2.

Lemma 3.1. *(cf. [3]) The following holds:*

(i) The map T_d is an n-step contraction, that is, $\|T_d^n v - T_d^n v'\| \leqq \beta \|v - v'\|$ for some $0 < \beta < 1$ and $v, v' \in B(L)$, where n is as that in (2.9).

(ii) T_d is monotone, that is, if $v \leqq v', T_d v \leqq T_d v'$.

(iii) Denote by $v\{d\}$ a unique fixed point of T_d. Then, if $d \leqq d', v\{d\} \leqq v\{d'\}$.

Proof. See Appendix.∎

Let $\mathcal{K} = \{(s,j) | 0 \leqq s \leqq n^*, 1 \leqq j \leqq l_s\}$ where n^* and l_s are given in Theorem 2.2. For $D \subset S$, let $q_i(D|a) := \sum_{j \in D} q_{ij}(a)$. In the ensuring discussion, we give the value iteration algorithm, called **Algorithm C**, for finding an optimal policy starting with data in Table 2.1.

Algorithm C.

1. Set $n = 1, g_{sj}(1) = g_{sj}$ for $(s,j) \in \mathcal{K}$ and $g_i(1) = v\{d_1\}(i)$ for $i \in L$, where $d_1(i,a) = \sum_{(s,j) \in \mathcal{K}} q_i(U_{sj}|a) g_{sj}(1)$.

2. Suppose that $\{g_{sj}(n) : (s,j) \in \mathcal{K}\}$ and $\{g_i(n) : i \in L\}$ are given $(n \geqq 1)$. Let for $i \in S - L$,

$$g_i = \max_{a \in A(i)} \{d_n(i,a) + \sum_{j \in L} q_{ij}(a) g_j(n)\}, \tag{3.2}$$

where $d_n(i,a) = \sum_{(s,j)\in\mathcal{K}} q_i(U_{sj}|a) g_{sj}(n)$.
Let $g_{sj}(n+1) = \max_{i \in U_{sj}} g_i$ and $g_i(n+1) = v\{d_n\}(i)$ for $i \in L$.

3. Let $n = n+1$ and go to Step 2.

Concerning with **Algorithm C**, we have the following.

Lemma 3.2. *In* **Algorithm C**, *we have:*

(i) It holds that

$$g_{sj}(n+1) \geqq g_{sj}(n) \text{ for } (s,j) \in \mathcal{K}, \text{ and} \tag{3.3}$$

$$g_i(n+1) \geqq g_i(n) \text{ for } i \in L. \tag{3.4}$$

(ii) The **Algorithm C** *converges, i.e., when* $n \to \infty$ $g_{sj}(n) \to \overline{g}_{sj}$ *for* $(s,j) \in \mathcal{K}$ *and* $g_i(n) \to \overline{g}_i$ *for* $i \in L$.

Proof. See Appendix.∎

For $\overline{g}_{sj}$ and $\overline{g}_i$ in Lemma 3.2, we have the following.

$$\overline{g}_{sj} = \max_{a \in A(i)} \{d(i,a) + \sum_{j \in L} q_{ij}(a) \overline{g}_j\} \quad \text{for } i \in U_{sj} \text{ and } (s,j) \in \mathcal{K}, \tag{3.5}$$

and

$$\overline{g}_i = \max_{a \in A(i)} \{d(i,a) + \sum_{j \in L} q_{ij}(a) \overline{g}_j\} \quad \text{for } i \in L, \tag{3.6}$$

where $d(i,a) = \sum_{(s,j)\in\mathcal{K}} q_i(U_{sj}|a) \overline{g}_{sj}$.

Let f^* be any stationary policy such that

$$f^*(i) = \begin{cases} f_{0j}(i) & \text{for } i \in U_{0j} \ (1 \leqq j \leqq l_0) \\ \text{any maximizer in (3.5) and (3.6)} & \text{for } i \in S_1. \end{cases} \tag{3.7}$$

Theorem 3.1. *The stationary policy* f^* *is optimal and*

$$\psi(i, f^*) = \overline{g}_{sj} \quad \text{for } i \in U_{sj}, (s,j) \in \mathcal{K}, \text{ and } \psi(i, f^*) = \overline{g}_i \quad \text{for } i \in L.$$

Proof. See Appendix.∎

The dynamic programming (DP) algorithm associated with MDPs has to compute a suitably defined value function on the state space, which gives rise to the difficulties called "curse of dimensionality" when applying the DP algorithm to the problem involving very large state space (cf. [2, 4]). In the structured algorithm developed in this section, the state space may be partitioned into several classes of a small number of states. This may be a way to overcome the difficulties mentioned above. The availability of our algorithm depends deeply on the structure of the model. So that it may have the possibility of reducing the amount of computation and the number of steps required in the computation for a simply structured model, whose studies will be in our future works.

4 A numerical example

In this section, we give a numerical example to comprehend our structured algorithm for the multichain case.

Let us find the optimality policy for the eight-state MDP with $S = \{1, 2, 3, 4, 5, 6, 7, 8\}$ and $A(1) = \{1, 2\}, A(2) = \{1\}, A(3) = \{1, 2, 3\}, A(4) = \{1, 2\}, A(5) = \{1, 2\}, A(6) = \{1, 2, 3\}, A(7) = \{1, 2\}$ and $A(8) = \{1, 2, 3\}$, whose transition probability matrix and rewards are given in Table 4.1.

state	action	transition probabilities $q_{ij}(a)$								reward
i	$a \in A(i)$	$j=1$	$j=2$	$j=3$	$j=4$	$j=5$	$j=6$	$j=7$	$j=8$	$r(i,a)$
1	1	0	1/2	1/4	0	0	1/4	0	0	9
	2	1/8	0	0	1/4	0	1/4	1/8	1/4	10
2	1	0	7/10	0	3/10	0	0	0	0	11
3	1	0	0	1/2	0	0	1/4	0	1/4	5
	2	0	0	1/8	0	0	1/8	0	3/4	2
	3	0	0	3/16	0	0	3/4	0	1/16	2.5
4	1	0	3/5	0	2/5	0	0	0	0	7
	2	0	2/5	0	3/5	0	0	0	0	8
5	1	0	0	0	0	1/2	0	1/2	0	12
	2	1/8	1/2	0	0	1/8	1/8	1/8	0	2
6	1	0	0	1/4	0	0	1/2	0	1/4	6
	2	0	0	1/16	0	0	3/16	0	3/4	0.75
	3	0	0	5/8	0	0	1/4	0	1/8	2.25
7	1	1/4	0	0	0	1/4	0	1/2	0	6
	2	1/4	0	0	1/8	1/4	0	1/4	1/8	7
	3	0	0	0	0	1/4	0	3/4	0	10
8	1	0	0	1/2	0	0	1/2	0	0	14
	2	0	0	1/16	0	0	1/16	0	7/8	13

Table 4.1: A numerical example

First we apply **Algorithm A** to the above multichain MDP. The pattern matrix

$M = (m_{ij})$ in (2.2) and $\overline{M} = (\overline{m}_{ij})$ in (2.3) are easily computed, which are shown as follows.

$$M = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \end{array} \begin{array}{c} \begin{array}{cccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \end{array} \\ \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \end{pmatrix} \end{array}, \quad \overline{M} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \end{array} \begin{array}{c} \begin{array}{cccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \end{array} \\ \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \end{pmatrix} \end{array}.$$

By reordering of the states, $\overline{M}$ is transformed as follows (abusing the notation).

$$\overline{M} = \begin{array}{c} \\ 3 \\ 6 \\ 8 \\ 2 \\ 4 \\ 1 \\ 5 \\ 7 \end{array} \begin{array}{c} \begin{array}{cccccccc} 3 & 6 & 8 & 2 & 4 & 1 & 5 & 7 \end{array} \\ \left(\begin{array}{ccc:cc:ccc} 1 & 1 & 1 & & & & & \\ 1 & 1 & 1 & & 0 & & & \\ 1 & 1 & 1 & & & & & \\ \hdashline & & & 1 & 1 & & & \\ & 0 & & 1 & 1 & & 0 & \\ \hdashline 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{array}\right) \end{array},$$

Observing the above, we have

$$S = U_{01} + U_{02} + L,$$

where $U_{01} = \{3, 6, 8\}, U_{02} = \{2, 4\}, L = \{1, 5, 7\}$.

Now, we apply **Algorithm B** to $L = \{1, 5, 7\}$. For $n = 1$, we get the following:

$$K_1 = L = \{1, 5, 7\}, V = S - K_1 = \{2, 3, 4, 6, 8\}, T(1) = \{1, 2\},$$
$$T(5) = \{2\}, T(7) = \{2\}, A_1(1) = \emptyset, A_1(5) = \{1\}, A_1(7) = \{1, 3\}.$$

So that $K_2 = \{5, 7\}$. Thus, we have found the maximum sub-MDP:

$$\Gamma_1 = (S_1 = \{5, 7\}, \{A_1(i) : i \in S_1\}, Q_{S_1}, r_{S_1}).$$

Applying **Algorithm A** to Γ_1, we find that Γ_1 is communicating. In the end, the decomposition of S in (2.7) is shown as: $S = U_{01} + U_{02} + U_{11} + L$ with $L = \{1\}$ and data in Table 2.1 is given in the following Table 4.2.

m.c.c.	$U_{01} = \{3, 6, 8\}$	$U_{02} = \{2, 4\}$	$U_{11} = \{5, 7\}$
relatively o.p.	$f_{01}(3) = f_{01}(6) = f_{01}(8) = 2$	$f_{02}(2) = 1, f_{02}(4) = 2$	$f_{11}(5) = 1, f_{11}(7) = 3$
relatively a.r.	11.333	9.714	10.667

Table 4.2: Relatively o.p. and a.r. for a numerical example

Here, to find an optimal policy, we apply **Algorithm C** to Table 4.2. Since, f_{01} and f_{02} are absolutely optimal, it holds that $g_{01}(n) = 11.333$ and $g_{02}(n) = 9.714$ for all $n \geqq 1$ and $f^*(3) = f^*(6) = f^*(8) = 2, f^*(2) = 1, f^*(4) = 2$. For $n = 1, g_{11}(1) = 10.667$ and $g_1(1) = v\{d_1\}(1) = 10.776$ is obtained as a unique solution of the following equation from Lemma 3.1:

$$g_1(1) = \max\left\{\frac{1}{2}g_{01}(1) + \frac{1}{2}g_{02}(1), \frac{1}{2}g_{01}(1) + \frac{1}{4}g_{02}(1) + \frac{1}{8}g_{11}(1) + \frac{1}{8}g_1(1)\right\}$$
$$= \max\left\{10.524, 9.429 + \frac{1}{8}g_1(1)\right\}.$$

By (3.2), we get that $g_5 = 10.667$ and $g_7 = 10.694$. So, $g_{11}(2) = \max\{g_5, g_7\} = 10.694$ and $g_1(2) = 10.776$. Similarly, we get that $g_{11}(3) = 10.714, g_1(3) = 10.779$ and $g_{11}(4) = 10.731, g_1(4) = 10.782$. We proceed with repeating the step n until $|g_{11}(n) - g_{11}(n+1)| < \epsilon = 10^{-4}$ and $|g_1(n) - g_1(n+1)| < \epsilon$ simultaneously. Then, we find that $\overline{g}_{11} \cong 10.794$ and $\overline{g}_1 \cong 10.794$. From Theorem 3.1, we get optimal policy in other states as $f^*(5) = 1, f^*(7) = 3, f^*(1) = 2$ and the optimal value $\psi^*(3) = \psi^*(6) = \psi^*(8) = 11.333, \psi^*(2) = \psi^*(4) = 9.714, \psi^*(5) = \psi^*(7) \cong 10.794, \psi^*(1) \cong 10.794$.

Appendix

Proof of Lemma 2.1

Let $\overline{D}$ be a communicating class. By the definition, $\overline{D}$ is closed, so that for $i \in \overline{D}$ and $j \notin \overline{D}$, $P_\pi(X_t = j | X_0 = i) = 0$ for all $t \geqq 1$ and $\pi \in \Pi$, which means $\overline{m}_{ij}(\overline{\Gamma}) = 0$. Also, there exists a randomized stationary policy γ such that $Q_{\overline{D}}(\gamma)$ determines a irreducible Markov chain, so that for $i, j \in \overline{D}$, $P_\gamma(X_t = j | X_0 = i) > 0$ for some t $(0 \leqq t \leqq N)$. Thus, $\overline{m}_{ij}(\overline{\Gamma}) = 1$ follows for $i, j \in \overline{D}$. The proof of the "if" part follows obviously.∎

Proof of Theorem 2.1

Let U_j and L be the sets of states corresponding to E_j and K in (2.4), respectively, $(1 \leqq j \leqq d)$. Then, from Lemma 2.1, U_j is a communicating class. If L is closed, $\overline{\overline{\Gamma}} = (L, \{\overline{A}(i) : i \in L, Q_L, r_L)\}$ is a sub-MDP. Applying the above discussion, we get $\overline{M}(\overline{\overline{\Gamma}})$ is rewritten into a matrix which has the same structure as (2.4), which is a contradiction.∎

Proof of Lemma 3.1

By (2.9), (i) clearly holds (cf. [3]). Also, (ii) follows from the definition. Let $d \leqq d'$. Then, we have that

$$v\{d\} = T_d v\{d\} \leqq T_{d'} v\{d\} \leqq (T_{d'})^n v\{d\} \to v\{d'\} \quad \text{as} \quad n \to \infty.$$

This completes the proof of (iii).∎

Proof of Lemma 3.2
Since U_{sj} is a communicating class, by the definition for each $i \in U_{sj}$, there exists $a \in A(i)$ such that $q_i(U_{sj}|a) = 1$, which implies from (3.2) that $g_i \geqq g_{sj}(n)$. So, we get $g_{sj}(n+1) \geqq g_{sj}(n)$. Also, Lemma 3.1 (iii), from $d_n \geqq d_{n-1}$ it follows that $g_i(n+1) = v\{d_n\}(i) \geqq v\{d_{n-1}\}(i) = g_i(n)$ for $i \in L$. Since $\{g_{sj}(n)\}$ and $\{g_i(n)\}$ are bounded, (ii) follows from (i). ∎

Proof of Theorem 3.1
For any stationary policy f, let denote by $Q(f)$ the corresponding transition matrix. The decomposition of the state space S w.r.t. $Q(f)$ will be denoted by

$$S = O_1 + \cdots + O_l + L',$$

where $O_i (i = 1, 2, \ldots, l)$ is an ergodic class and L' is a transient class (cf. [9]). For each O_k, since L is a transient class it is impossible that $O_k \subset L$. Thus, recalling that each U_{sj} is a maximum communicating class, $O_k \subset U_{sj}$ for some $(s, j) \in \mathcal{K}$, which implies $\psi(i, f) \leqq \overline{g}_{sj}$ for $i \in O_k$. Also, by (3.5)–(3.6), $\psi(i, f) \leqq \psi(i, f^*)$ for $i \in L'$. The other half part of theorem holds obviously.∎

References

[1] John Bather. Optimal decision procedures for finite Markov chains. II. Communicating systems. *Advances in Appl. Probability*, 5:521–540, 1973.

[2] Richard Bellman. *Dynamic programming.* Princeton Univeristy Press, Princeton, N. J., 1957.

[3] Eric V. Denardo. Contraction mappings in the theory underlying dynamic programming. *SIAM Rev.*, 9:165–177, 1967.

[4] Eric V. Denardo. *Dynamic programming: models and applications.* Prentice-Hall, Inc., Englewood Cliffs, N. J., 1982.

[5] A. Federgruen and P. J. Schweitzer. Discounted and undiscounted value-iteration in Markov decision problems: a survey. In *Dynamic programming and its applications (Proc. Conf., Univ. British Columbia, Vancouver, B.C., 1977)*, pages 23–52. Academic Press, New York, 1978.

[6] A. Hordijk and L. C. M. Kallenberg. Linear programming and Markov decision chains. *Management Sci.*, 25(4):352–362, 1979/80.

[7] Arie Hordijk and Martin L. Puterman. On the convergence of policy iteration in finite state undiscounted Markov decision processes: the unichain case. *Math. Oper. Res.*, 12(1):163–176, 1987.

[8] Ronald A. Howard. *Dynamic programming and Markov processes.* The Technology Press of M.I.T., Cambridge, Mass., 1960.

[9] John G. Kemeny and J. Laurie Snell. *Finite Markov chains.* The University Series in Undergraduate Mathematics. D. Van Nostrand Co., Inc., Princeton, N.J.-Toronto-London-New York, 1960.

[10] Arie Leizarowitz. An algorithm to identify and compute average optimal policies in multichain Markov decision processes. *Math. Oper. Res.*, 28(3):553–586, 2003.

[11] Martin L. Puterman. *Markov decision processes: discrete stochastic dynamic programming.* John Wiley & Sons Inc., New York, 1994. A Wiley-Interscience Publication.

[12] Paul J. Schweitzer. Iterative solution of the functional equations of undiscounted Markov renewal programming. *J. Math. Anal. Appl.*, 34:495–501, 1971.

[13] E. Seneta. *Nonnegative matrices and Markov chains.* Springer Series in Statistics. Springer-Verlag, New York, second edition, 1981.

[14] D. J. White. Dynamic programming, Markov chains, and the method of successive approximations. *J. Math. Anal. Appl.*, 6:373–376, 1963.