Chapter 11

A SENSITIVITY VIEW OF MARKOV DECISION PROCESSES AND REINFORCEMENT LEARNING

Xi-Ren Cao*

Department of Electrical and Electronic Engineering
The Hong Kong University of Science and Technology
Clear Water Bay, Kowloon, Hong Kong

Abstract The goals of perturbation analysis (PA), Markov decision processes (MDPs), and reinforcement learning (RL) are common: to make decisions to improve the system performance based on the information obtained by analyzing the current system behavior. In this paper, we study the relations among these closely related fields. We show that MDP solutions can be derived naturally from performance sensitivity analysis provided by PA. Performance potential plays an important role in both PA and MDPs; it also offers a clear intuitive interpretation for many results. Reinforcement learning, TD(\lambda), neuro-dynamic programming, etc, are efficient ways of estimating the performance potentials and related quantities based on sample paths. This new view of PA, MDPs and RL leads to the gradient-based policy iteration method that can be applied to some nonstandard optimization problems such as those with correlated actions. Sample path-based approaches are also discussed.

Key words: Potentials, Poisson equations, gradient-based policy iteration, Q-learning, TD(\lambda)

*Supported in part by a grant from Hong Kong UGC. Tel: (852) 2358-7048 Fax: (852) 2358-1485 Email: eecao@ee.ust.hk
1. Introduction

Perturbation analysis (PA) [4, 8, 13, 17] provides performance sensitivities for a discrete event dynamic system (DEDS) by analyzing the dynamical behavior of a single sample path of the DEDS. Performance optimization can be achieved by combining PA with stochastic approximation methods. Markov decision process (MDP) [1, 21, 22, 23] is a general model for performance optimization of DEDSs. Policy iteration, the basic approach in MDPs, can be implemented based on sample paths. The goal of reinforcement learning (RL) [1, 3, 24, 25] is to learn how to make decisions to improve a system's performance by observing its behavior. In this paper, we study the relations among these closely related fields. We show that MDP solutions can be derived naturally from performance sensitivity analysis provided by PA, and that reinforcement learning, TD(\(\lambda\)), neuro-dynamic programming, etc, are sample-path-based efficient ways of estimating the performance potentials and related quantities, which are crucial elements in PA, MDPs and other optimization approaches. This sensitivity-based view of MDPs leads to a new approach, the gradient-based policy iteration, to MDPs with correlated actions at different states.

We first briefly review the basic principles. We show that the effect of a small change in a system parameter on the system performance equals the sum of the effects of all the perturbations (small or large) induced by this small change in the parameter. The effect of a perturbation in a DEDS on its long term performance can be measured quantitatively. From this basic principle, the performance sensitivities can be derived [4, 8].

The above principle can be applied to Markov processes to obtain performance sensitivity [8]. Two types of sensitivities are studied: sensitivities with respect to discrete and continuous parameters. In the discrete case, the difference of the performance of the Markov processes under two different policies is found. In the continuous case, the policies are randomized, with a parameter \(\delta\) indicating the probability of applying one policy and \(1 - \delta\) the probability of the other, the derivative of the performance with respect to \(\delta\) is found; this is the derivative along the direction from one policy to the other in the policy space. With the discrete sensitivity formula, the standard policy iteration optimization algorithm can be very easily derived. Compared with the continuous sensitivity formula, it is clear that at each step the policy iteration algorithm simply chooses the "steepest" direction (with the largest absolute value of the directional derivative) to go for its next policy [5].
The concept of performance potentials plays a significant role in sensitivity analysis of Markov processes and MDPs. Just as the potential energy in physics, only the relative values (i.e., the differences) of the performance potentials at different states are meaningful, i.e., the potentials are determined only up to an additive constant. Realization factor is defined as the difference between the potentials at two states \( i \) and \( j \), \( g(i) - g(j) \), which measures the effort of a jump from state \( j \) to state \( i \) on the long term (average or discounted) performance. This offers an intuitive explanation for performance sensitivity and MDPs (in view of discrete sensitivity). The performance potential at state \( i \), \( g(i) \), can be approximated by the mean sum of the performance at the first \( N \) states on a sample path starting from state \( i \), and hence it can be estimated online. A number of other single sample path-based estimation algorithms for potentials have been derived.

With the potentials estimated, performance derivatives (i.e., PA) can be obtained and policy iteration (i.e., MDPs) can be implemented based on a single sample path. The single sample path-based approach is practically important because it can be applied online to real engineering systems; in most cases, it does not require the system parameters to be completely known (see the example in Section 5). There are two ways to achieve the optimal performance with this approach: by perturbation analysis using performance derivatives, or by policy iteration, both can be implemented online. Stochastic approximation methods have to be used in these two cases to improve the convergence speeds and to reduce stochastic errors. For details, see Section 5 and [12, 19].

Inspired by the potential-based sensitivity view of MDPs and the single sample path-based approach, a number of new research topics emerge. First, we observe that policy iteration can be implemented by using the performance derivatives. This leads to a new approach, the gradient-based policy iteration, to MDPs with correlated actions at different states. This gradient-based policy iteration can be applied to problems such as MDPs with hidden state components and the distributed control of MDPs, the details are reported in a forthcoming paper [10]. The other topics include the time aggregation of MDPs [9], which was first used in [26] for perturbation analysis. When the number of controllable states is small, this approach may save computations, storage spaces, and/or the number of transitions in sample-path-based approaches.

We first study MDPs with long term average costs. In Section 2, we start with a brief introduction to PA. In Section 2.1, we apply the basic ideas of PA to Markov chains to obtain the performance gradients. Performance potentials are introduced here. In Section 2.2, the results
are extended to discrete performance sensitivities, i.e., the difference of the performance measures of any two policies. In Section 3, we show that policy iteration in MDP can be easily derived from the discrete sensitivity formula. Gradient-based policy iteration is discussed in Section 3.2. In Section 4, we extend the results to MDPs with discounted performance criteria. All the results for long term average MDPs can be viewed as special cases when the discount factor $\alpha$ goes to one. Thus, potential theory provides a unified framework to MDPs with both long term average and discounted performance criteria. In Section 5, some sample path-based estimation algorithms for potentials are derived. An example is given to illustrate the possible advantages of applying the sample-path-based optimization approach. In Section 6 we give a brief review of Q-learning, $TD(\lambda)$, and neuro-dynamic programming, which provide efficient ways in estimating the potentials and Q-factors, the latter are used in place of potentials when the system structure is completely unknown. In Section 7, we give a brief summary as a conclusion.

The research of perturbation analysis initiated with Prof. Ho's insightful and creative works in late 70's and early 80's [15, 16, 17, 18]. This paper aims at positioning this research area in the global picture of system optimization. We dedicate this survey on the recent development in this fascinating area to Prof. Y. C. Ho on the occasion of his retirement from teaching.

## 2. Perturbation Analysis of Markov Chains

The basic principle for PA can be explained by using the concept of perturbation realization. A small change in a system parameter (such as the mean service time of a server) induces a series of (small or large) changes on a sample path; each change is called a perturbation. The average effect of each perturbation on the system performance can be precisely measured. The total effect of the small change in the parameter on the system performance can then be calculated by adding together the average effects of all the perturbations induced by the parameter change. The sensitivity of the performance with respect to the parameter can then be determined. The idea has been successfully applied to performance sensitivity analysis of queueing networks [18] as well as Markov chains [8, 5].

### 2.1 Performance Sensitivity of Markov Chains: the Continuous Case

Consider an irreducible and aperiodic Markov chain $X = \{X_n : n \geq 0\}$ on a finite state space $S = \{1, 2, \cdots, M\}$ with transition probability
matrix $P = [p(i, j)] \in [0, 1]^{M \times M}$. Let $\pi = (\pi_1, \ldots, \pi_M)$ be the vector representing its steady-state probabilities, and $f = (f_1, f_2, \ldots, f_M)^T$ be the performance vector, where "$T$" represents transpose. We have $Pe = e$, where $e = (1, 1, \ldots, 1)^T$ is an $M$-dimensional vector whose all components equal 1, and $\pi = \pi P$. The performance measure is the long term average defined as

$$
\eta = E_\pi(f) = \sum_{i=1}^{M} \pi_i f_i = \pi^T f = \lim_{L \to \infty} \frac{1}{L} \sum_{l=0}^{L-1} f(X_l) = \lim_{L \to \infty} \frac{F_L}{L}, \quad (1)
$$

where

$$
F_L = \sum_{i=0}^{L-1} f(X_i).
$$

Let $P'$ be another irreducible transition probability matrix on the same state space. Suppose $P$ changes to $P(\delta) = P + \delta Q = \delta P' + (1-\delta) P$, with $\delta > 0$, $Q = P' - P = [q(i, j)]$. We have $Qe = 0$. The performance measure will change to $\eta(\delta) = \eta + \Delta \eta$. The derivative of $\eta$ in the direction of $Q$ is defined as $\frac{d\eta}{d\delta} = \lim_{\delta \to 0} \frac{\Delta \eta}{\delta}$.

In this system, a perturbation means that the system is perturbed from one state $i$ to another state $j$. For example, consider the case where $q(k, i) = -\frac{1}{2}$, $q(k, j) = \frac{1}{2}$, and $q(k, l) = 0$ for all $l \neq i, j$. Suppose that in the original sample path the system is in state $k$ and jumps to state $i$, then in the perturbed path it may jump to state $j$ instead. Thus, we study two independent Markov chains $X = \{X_n; n \geq 0\}$ and $X' = \{X'_n; n \geq 0\}$ with $X_0 = i$ and $X'_0 = j$; both of them have the same transition matrix $P$. The realization factor is defined as [8]:

$$
d(i, j) = \lim_{L \to \infty} E \left[ \sum_{l=0}^{L-1} (f(X'_l) - f(X_l)) \mid X_0 = i, X'_0 = j \right], \quad (2)
$$

Thus, $d(i, j)$ represents the average effect of a jump from $i$ to $j$ on $F_L$ in (1).

If $P$ is irreducible, then with probability one the two sample paths of $X$ and $X'$ will merge together. That is, there is a random number $L^*$ such that $X_{L^*} = X'_{L^*}$. Therefore, from the Markov property, (2) becomes

$$
d(i, j) = E \left[ \sum_{l=0}^{L^*-1} (f(X'_l) - f(X_l)) \mid X_0 = i, X'_0 = j \right], \quad i, j = 1, \ldots, M. \quad (3)
$$
The matrix $D \in \mathcal{R}^{M \times M}$, with $d(i,j)$ as its $(i,j)$th element, is called a realization matrix. From (3), we can prove that $D$ satisfies the Lyapunov equation [8]

$$D - PD^T = F,$$

(4)

where $F = fe^T - ef^T$.

(2) indicates that every visit to state $i$ contributes to $F_L$ on the average by the amount of

$$g(i) \approx E\left[\sum_{l=0}^{L-1} f(X_l)|X_0 = i\right],$$

(5)

and only the difference between different $g(i)$s are important. Now we consider a sample path consisting of $L$ transitions. Among these transitions, on the average there are $L\pi_i$ transitions at which the system are at state $i$. After being at state $i$, the system jumps to state $j$ on the average $L\pi_ip(i,j)$ times. If the transition probability matrix $P$ changes to $P(\delta) = P + \delta Q$, then the change in the number of visits to state $j$ after being at state $i$ is $L\pi_iq(i,j)\delta = L\pi_i[p'(i,j) - p(i,j)]\delta$. This contributes a change of $\{L\pi_i[p'(i,j) - p(i,j)]\delta\}g(i)$ to $F_L$. Thus, the total change in $F_L$ due to the change of $P$ to $P(\delta)$ is

$$\Delta F_L = \sum_{i,j=1}^{M} L\pi_i[p'(i,j) - p(i,j)]\delta g(i) = \pi [P' - P]g\delta L = \pi Qg\delta L,$$

where $g = (g(1), \cdots, g(M))^T$. Finally, we have

$$\frac{dn}{d\delta} = \lim_{\delta \to 0} \frac{1}{\delta} \Delta F_L L = \pi Qg = \pi (P' - P)g.$$

(6)

(5) is only an approximation. When $L \to \infty$, this expression is unbounded. However, because $Qe = 0$, we have $Q(g + ce) = Qg$ for any real constant $c$. Thus, the sensitivity equation (6) holds when $g$ is replaced by $g + ce$. (This confirms the fact that $g$ is only important up to an additive constant.) Therefore, we may add a constant $-L\eta$ to (5) and obtain

$$g(i) \approx E\left[\sum_{l=0}^{L-1} [f(X_l) - \eta]|X_0 = i\right].$$

Letting $L \to \infty$, we obtain the formal definition of the performance potential at state $i$:

$$g(i) = \lim_{L \to \infty} E\left[\sum_{l=0}^{L-1} [f(X_l) - \eta]|X_0 = i\right],$$

(7)
which can be proved to be finite for ergodic chains. Again, the above discussion only provides an intuitive explanation; for a rigorous proof, see [8].

We have \( d(i,j) = g(i) - g(j) \) and \( D = ge^T - eg^T \). From this, we have

\[
\frac{d\eta}{d\delta} = \pi Q D^T \pi^T. \tag{8}
\]

From (7), it is easy to prove that the potential vector \( g \) satisfies the Poisson equation

\[
(I - P + e\pi)g = f. \tag{9}
\]

### 2.2 Performance Sensitivity of Markov Chains: the Discrete Case

In sensitivity analysis with discrete parameters, we wish to obtain the difference of the performance measures of two Markov chains with transition probability matrices \( P \) and \( P' \), respectively, both are ergodic on the same state space. We use prime "'" to denote the values associated with \( P' \).

First, multiplying both side of the Poisson equation with \( \pi \) on the left, we get

\[
\pi g = \pi f = \eta. \tag{10}
\]

Next, multiplying both side of the Poisson equation with \( \pi' \) on the left yields

\[
\pi' Q g = \pi'(P' - P) g = \pi'(I - P) g = \pi' f - \pi g = \pi' f - \eta.
\]

That is,

\[
\eta' - \eta = \pi' Q g. \tag{11}
\]

(11) can be viewed as the discrete version of the performance sensitivity. Note that the discrete sensitivity (11) can be obtained from its continuous counterpart (6) by replacing \( \pi \) with \( \pi' \). This feature serves as the basis for the gradient-base policy iteration for Markov decision processes (MDPs) proposed later in Section 3.2.

For more general cases, we assume that the performance function also changes from \( f \) to \( f' \). Let \( h = f' - f \). It is easy to check that

\[
\eta' - \eta = \pi'(Q g + h). \tag{12}
\]

For continuous sensitivity, we set \( f(\delta) = f + \delta h \). Together with \( P(\delta) = P + \delta Q \), we have

\[
\frac{d\eta}{d\delta} = \lim_{\delta \to 0} \frac{1}{\delta} \Delta F L \Delta = \pi(Q g + h). \tag{13}
\]
3. Markov Decision Processes

3.1 Policy Iteration

To know the exact value of the performance difference from (12), one needs to know \( \pi' \). On the other hand, if \( \pi' \) is known, one can get \( \eta' \) directly by \( \pi' f \). In addition, it is impossible to calculate \( \pi' \) for all the policies since the policy space is usually very large. Fortunately, since \( \pi' > 0 \) (componentwisely), we can always determine which action is better at each state by using (12). Thus, we can perform policy iteration without knowing the exact value of the performance difference. This leads to the following discussion.

For two \( M \)-dimensional vectors \( a \) and \( b \), we define \( a = b \) if \( a(i) = b(i) \) for all \( i = 1, 2, \cdots, M \); \( a \leq b \) if \( a(i) < b(i) \) or \( a(i) = b(i) \) for all \( i = 1, 2, \cdots, M \); \( a < b \) if \( a(i) < b(i) \) for all \( i = 1, 2, \cdots, M \); and \( a \preceq b \) if \( a(i) < b(i) \) for at least one \( i \), and \( a(j) = b(j) \) for other components. The relation \( \leq \) includes \( = \), \( \preceq \), and \( < \). Similar definitions are used for the relations \( > \), \( \succeq \), and \( > \).

Next, we note that \( \pi'(i) > 0 \) for all \( i = 1, 2, \cdots, M \). Thus, from (12), we know that if \( Qg + h = (P' - P)g + (f' - f) \succeq 0 \) then \( \eta' - \eta > 0 \). From (12) and the fact \( \pi' > 0 \), the proof of the following lemma is straightforward.

**Lemma 7.** If \( Pg + f \preceq P'g + f' \), then \( \eta < \eta' \).

It is interesting to note that in the lemma, we use only the potentials with one Markov chain, i.e., \( g \). Thus, to compare the performance measures under two policies, only the potentials with one policy is needed.

In an MDP, at any transition instant \( n \geq 0 \) of a Markov chain \( X = \{X_n, n \geq 0\} \), an action is chosen from an action space \( \mathcal{A} \) and is applied to the Markov chain. We assume that the number of actions is finite, and we only consider stationary policies. The actions that are available for \( i \in S \) form a nonempty subset \( A(i) \subseteq \mathcal{A} \). A stationary policy is a mapping \( \mathcal{L} : S \rightarrow \mathcal{A} \), i.e., for any state \( i \), \( \mathcal{L} \) specifies an action \( \mathcal{L}(i) \in A(i) \). Let \( \mathcal{E} \) be the policy space. If action \( \alpha \) is taken at state \( i \), then the state transition probabilities at state \( i \) are denoted as \( p^\alpha(i, j), j = 1, 2, \cdots, M \). With a policy \( \mathcal{L} \), the Markov process evolves according to the transition matrix \( P^\mathcal{L} = [p^\mathcal{L}(i, j)]_{i=1}^{M}_{j=1} \). We use the superscript \( *^\mathcal{L} \) to denote the quantities associated with policy \( \mathcal{L} \).

For most stable systems in real world applications, the Markov chains under any policy are recurrent. For example, in any stable communication systems, from any state it is always possible to reach the null state where there are no packets in the system. Therefore, in this paper we assume that the Markov chain is recurrent under any policy. In addition,
for simplicity we assume that it is aperiodic; although all the results in this paper hold for periodic chains if we replace the steady state value by the corresponding time average value [23]. Therefore, the Markov chains considered are ergodic [11]. This corresponds to the problems classified in [23] as the recurrent case.

The steady-state probabilities corresponding to policy $\mathcal{L}$ is denoted as a vector $\pi^\mathcal{L} = (\pi^\mathcal{L}(1), \cdots, \pi^\mathcal{L}(M))$. Suppose that at each stage with state $i$ and control action $\alpha \in A(i)$, a cost $f(i, \alpha) = f(i, \mathcal{L}(i))$ is incurred. The long term expected value of the average cost per stage corresponding to policy $\mathcal{L}$ is then

$$\eta^\mathcal{L} = \lim_{L \to \infty} \frac{1}{L} E\{\sum_{i=0}^{L-1} f[X_i, \mathcal{L}(X_i)]\},$$

For ergodic chains, the above limit exists and does not depend on the initial state. Our objective is to minimize this average cost per stage over the policy space $S$, i.e., to obtain $\min_{L \in T} \eta^\mathcal{L}$.

Define $f^\mathcal{L} = (f[1, \mathcal{L}(1)], \cdots, f[M, \mathcal{L}(M)])^T$. (9) and (10) becomes

$$(I - P^\mathcal{L} + e\pi^\mathcal{L})g^\mathcal{L} = f^\mathcal{L}, \tag{14}$$

and

$$\pi^\mathcal{L}g^\mathcal{L} = \pi^\mathcal{L}f^\mathcal{L}.$$

The following optimality theorem follows almost immediately from Lemma 7, which is derived directly from the sensitivity formula (12).

**Theorem 1** A policy $\mathcal{L}$ is optimal if and only if

$$P^\mathcal{L}g^\mathcal{L} + f^\mathcal{L} \leq P^{\mathcal{L}'}g^\mathcal{L} + f^{\mathcal{L}'} \tag{15}$$

for all $\mathcal{L}' \in \mathcal{E}$.

The optimality condition (15) is, of course, equivalent to the other conditions in the literature. To see this, we rewrite (14) in the following form:

$$\eta^\mathcal{L}e + g^\mathcal{L} = f^\mathcal{L} + P^\mathcal{L}g^\mathcal{L}. \tag{16}$$

Then Theorem 1 becomes: A policy $\mathcal{L}$ is optimal if and only if

$$\eta^\mathcal{L}e + g^\mathcal{L} = \min_{\mathcal{L}' \in \mathcal{E}} \{P^{\mathcal{L}'}g^\mathcal{L} + f^{\mathcal{L}'}\}. \tag{17}$$

The minimum is taken for every component of the vector. (17) is the optimality equation, or the Bellman equation. From (17), $g^\mathcal{L}$ is equivalent
to the “differential” or “relative cost vector” in [1], or the “bias” in [23]. In our approach, \( \gamma \) is directly related to the long term expected performance, and many results, such as the existence and the uniqueness of the solution to Equation (16), the optimality condition (17), and the convergence of the optimal algorithms, become almost obvious. In addition, as shown in the Section 5, \( g^L \) can be easily estimated based on a single sample path. This is an important feature which allows optimization algorithms to be implemented online for real world systems.

Policy iteration algorithms for determining the optimal policy can be easily developed by combining Lemma 7 and Theorem 1. Roughly speaking, at the \( k \)th step with policy \( \mathcal{L}_k \), we set the policy for the next step (the \((k+1)\)th step) as \( \mathcal{L}_{k+1} = \arg\{\min[\mathcal{P}^L g^L_k + f^L \} \), with \( g^L_k \) being the solution to the Poisson equation for \( \mathcal{P}^L \). Lemma 7 implies that performance usually improves at each iteration. Theorem 1 shows that the minimum is reached when no performance improvement can be achieved. We shall not state the details here because they are standard.

Finally, since \( \mathcal{P}^L_k \) and \( f^L_k \) are fixed, we note that \((\mathcal{P}^L - \mathcal{P}^L_k)g^L_k + (f^L - f^L_k)\) takes the minimal value (component wise) at \( \mathcal{P}^L_{k+1} \). Thus, because \( \pi^L_k > 0 \), from (13) the performance derivative also reaches minimum at \( \mathcal{P}^L_{k+1} \) (i.e., the largest absolute value, since the derivative is negative). Therefore, we conclude that policy iteration in fact chooses the steepest direction to go in the policy space.

### 3.2 Gradient-Based Policy Iteration

Because in (11) \( \pi' \) is unknown, one has to compare the effect of the actions on the performance state by state. Therefore, the policy iteration described in Section 3.1 can be used only for problems in which actions at difference state are independent. That is, the approach does not apply to systems in which actions at different states may be related. On the other hand, the performance gradient (6) or (13) does not depend on \( \pi' \), and all the information except \( P' \) can be obtained from the current policy \( P \). Thus, principally one can solve for \( \pi \) and determine the smallest \( \pi Qg \) to implement policy iteration. This is the gradient-based policy iteration method, which can be used to solve the optimization problem for systems where the actions at different states are correlated.

For example, consider a simple system where the actions at states 1 and 2 must be the same. In the gradient-based policy iteration, we simply choose the action that minimizes \( \pi(1) \sum_{j=1}^{M} p^\alpha(1,j)g(j) + \pi(2) \sum_{j=1}^{M} p^\alpha(2,j)g(j) \) as the action for both states 1 and 2 in the next iteration. More details are reported in a recent paper [10].
4. Problems with Discounted Performance Criteria

In this section, we extend the above results to MDPs with discounted performance criteria [7]. We shall see, with performance potentials, both the average and the discounted cost problems can be solved with the same framework, and the average cost problem can be treated as a special case when the discount factor $\alpha$ goes to 1.

Let $f(i), i \in S$, be a performance function and $\alpha, 0 < \alpha \leq 1$, be a discount factor. For $0 < \alpha < 1$, the performance cost is defined as a column vector $\eta_\alpha = (\eta_\alpha(1), \eta_\alpha(2), \ldots, \eta_\alpha(M))^T$ with

$$\eta_\alpha(i) = (1 - \alpha)E \{ \sum_{n=0}^{\infty} \alpha^n f(X_n) | X_0 = i \}. \quad (18)$$

The factor $(1 - \alpha)$ in (18) is used to obtain the continuity of $\eta_\alpha$ at $\alpha = 1$.

In fact, we define

$$\eta_1 = \lim_{\alpha \to 1^-} \eta_\alpha. \quad (19)$$

It is proved in (20) and (24) that the above limit exists.

**Lemma 8** $\eta_1 = \eta e$ with $\eta$ being the average-cost performance:

$$\eta = \pi f = \lim_{N \to \infty} \{ E[\frac{1}{N} \sum_{n=0}^{N-1} f(X_n)] \}.$$

**Proof.** In a matrix form, we have

$$\eta_\alpha = (1 - \alpha) \sum_{n=0}^{\infty} \alpha^n P^n f = (1 - \alpha)(I - \alpha P)^{-1} f, \quad 0 < \alpha < 1. \quad (20)$$

The second equation in (20) holds because for $0 < \alpha < 1$ all the eigenvalues of $\alpha P$ are within the unit circle [2]. Next, it is easy to verify

$$(I - \alpha P)^{-1} = (I - \alpha P + \alpha e\pi)^{-1} + \frac{\alpha}{1 - \alpha} e\pi, \quad \alpha < 1. \quad (21)$$

Thus,

$$\lim_{\alpha \to 1^-} (1 - \alpha)(I - \alpha P)^{-1} = e\pi. \quad (22)$$

The lemma then follows directly from (20) and (22). \[ \square \]

Similar to (9), the $\alpha$-Poisson equation is defined as

$$(I - \alpha P + \alpha e\pi)g_\alpha = f. \quad (23)$$
$g_\alpha$ is called the $\alpha$-potential, which is the same as what is defined in the classical potential theory for $0 < \alpha < 1$ (see (30) and [11]). Moreover, (21) and (20) leads to

$$\eta_\alpha = (1 - \alpha)g_\alpha + \alpha \eta e.$$  \hspace{1cm} (24)

It becomes obvious that (19) does converge.

When $\alpha = 1$, it is the standard Poisson equation. From (23), we have

$$g_\alpha = (I - \alpha P + \alpha e \pi)^{-1} f$$

$$= \left\{ \sum_{n=0}^{\infty} \alpha^n (P - e \pi)^n \right\} f$$

$$= \left\{ I + \left( \sum_{n=1}^{\infty} \alpha^n (P^n - e \pi) \right) \right\} f, \hspace{1cm} 0 < \alpha \leq 1.$$  

In particular,

$$g_1 = (I - P + e \pi)^{-1} f = \{ I + \sum_{n=1}^{\infty} (P^n - e \pi) \} f.$$  

This is the same as the performance potentials defined for the average cost MDPs. We have

$$\lim_{\alpha \to 1^-} g_\alpha = g_1,$$

$$\pi g_\alpha = \pi f.$$  

Now suppose that $P$ changes to $P'$, then $\pi$ and $\eta$ will change to $\pi'$ and $\eta'$, respectively. Let $Q = P' - P$. From (20), we get

$$\eta'_\alpha - \eta_\alpha = \alpha (P' \eta'_\alpha - P \eta_\alpha) = \alpha (P' - P) \eta_\alpha + \alpha P' (\eta'_\alpha - \eta_\alpha).$$  

This leads to

$$\eta'_\alpha - \eta_\alpha = \alpha (I - \alpha P')^{-1} Q \eta_\alpha.$$  

Substituting (24) into the right-hand side of the above equation and noting that $Q e = 0$, we obtain

$$\eta'_\alpha - \eta_\alpha = \alpha (1 - \alpha) (I - \alpha P')^{-1} Q g_\alpha, \hspace{1cm} 0 < \alpha < 1,$$  \hspace{1cm} (25)

Letting $\alpha \to 1^-$, we obtain (11) as a special case.

Similar to Lemma 7, if we assume that $f$ also changes to $f'$, then we have

Lemma 9 If $\alpha P' g_\alpha + f' \leq \alpha P g_\alpha + f$, then $\eta'_\alpha < \eta_\alpha, 0 < \alpha \leq 1.$
Proof. For $0 < \alpha < 1$, we have
\[(I - \alpha P')^{-1} = I + \alpha P' + \alpha^2 P'^2 + \ldots.\]

Since the Markov chain is positive recurrent, every item in $(I - \alpha P')^{-1}$ is positive. The lemma for $0 < \alpha < 1$ follows directly from (25); for $\alpha = 1$, it is the same as Lemma 7.

Now, policy iteration algorithms for discounted cost can be derived easily. At the $k$th step, it simply chooses the policy that minimizes $\alpha P^k g^k + f^k$, componentwisely, as the policy for the next step.

Next, we study the performance sensitivity. Let $P(\delta) = P + Q \delta$. Thus, $P(1) = P'$, $P(0) = P$. Then from (25), we have
\[
\frac{d\eta_\alpha}{d\delta} = \alpha(1 - \alpha)(I - \alpha P)^{-1}Qg_\alpha, \quad 0 < \alpha < 1. \tag{26}
\]

From (25) and (26), it is clear that policy iteration for discounted problems also chooses the next policy along the steepest direction. We have
\[
\frac{d\eta}{d\delta} = e \frac{d\eta}{d\delta} = \lim_{\alpha \to 1} \frac{d\eta_\alpha}{d\delta}.
\]

5. Single Sample Path-Based Implementations

5.1 Estimating Performance Potentials

In this section, we present some sample-path-based estimation algorithms for potentials and realization factors [5] [6].

First, from (7), we can choose a fixed $L$ and use $g_L(i) = E[\sum_{t=0}^{L-1} f(X_t) | X_0 = i] - L \eta$ as an estimate of $g(i)$. Define $\epsilon_i(x) = 1$, if $x = i$, and $\epsilon_i(x) = 0$, otherwise. Given a long sample path $(X_0, X_1, \ldots, X_K)$, we can prove
\[
g_L(i) = \lim_{K \to \infty} \left\{ \frac{\sum_{k=0}^{K-L+1} \epsilon_i(X_k) [\sum_{j=0}^{L-1} f(X_{k+j})]}{\sum_{k=0}^{K-L+1} \epsilon_i(X_k)} - \frac{L}{K} \sum_{k=0}^{K-1} f(X_k) \right\}, \tag{27}
\]
w.p.1.

For realization factor, algorithms can be developed from (3). However, such algorithms require two sample paths, one starting with $X_0 = i$ and the other with $X'_0 = j$. Another algorithm overcomes this difficulty and is based on a finite sample path. On a Markov chain $X = \{X_n, n \geq 0\}$ with $X_0 = i$, we define $L_i(j) = \min\{n: n \geq 0, X_n = j\}$; i.e., at $n = L_i(j)$, the Markov chain reaches state $j$ for the first time. We have
\( E[L_i(j)|X_0 = i] < \infty \) \[11\], and from \[8\]

\[
d(j, i) = E\{ \sum_{l=0}^{L_i(j)-1} [f(X_n) - \eta]|X_0 = i]. \tag{28}
\]

(28) relates \( d(i, j) \) to a finite portion of the sample paths of \( X \). To develop an algorithm based on (28), we define \( u_0 = 0 \), and \( u_{k+1} = \min\{n : n > u_k, X_n = i\}, k \geq 0 \), where \( i \) is a fixed state. \( u_k, k \geq 0 \) are regenerative points. For any \( j \neq i \), define \( v_k(j) = \min\{n : u_{k+1} > n > u_k, X_n = j\} \) and \( \chi_k(j) = 1 \), if \( \{u_{k+1} > n > u_k, X_n = j\} \neq \emptyset \); and \( \chi_k(j) = 0 \), otherwise. From (28), we have

\[
d(i, j) = \lim_{K \to \infty} \frac{1}{\sum_{k=0}^{K-1} \chi_k(j)} \left\{ \left[ \sum_{k=0}^{K-1} \chi_k(j) \sum_{n=v_k(j)}^{u_{k+1}-1} f(X_n) \right] - \left[ \sum_{k=0}^{K-1} \chi_k(j)[u_{k+1} - v_k(j)] \right] \eta \right\}, \ w.p.1 \tag{29}
\]

where \( \eta \) can be simply estimated by

\[
\eta = \lim_{N \to \infty} \frac{1}{N} \sum_{n=0}^{N-1} f(X_n), \quad w.p.1.
\]

Given a sample path of a system run under the current policy, we can estimate the potentials using (27) or (29). Then the performance derivatives in a continuous parameter space can be obtained by (6) or (13). The derivatives can be used together with stochastic approximation techniques to reach the optimal performance. Algorithms and proofs of convergence and other results in this direction is reported in [19]. We can also use the potentials estimated on a sample path to implement policy iteration. When the new policy is found, we run the system under the new policy for another sample path and update the policy again until some criterion is met. We shall refer to this procedure as the \textit{sample path-based MDP algorithm}. Stochastic approximation techniques can also be used in this procedure to provide fast algorithms. Research in this direction is reported in [12]. The main advantage of the sample path-based approach is that it can be applied online to real systems without estimating the system parameters, and the inversion of large matrices is avoided. In addition, the sample path-based approach offers flexibility in obtaining sub-optimal policies with less computation. For more details, see [6] and the example in Section 5.2.
The above discussion applies to discounted MDPs as well. In particular, ignoring the constant term, the \( \alpha \) potential can be written as

\[
g_\alpha(i) = E\{\sum_{n=0}^{\infty} \alpha^n f(X_n)|X_0 = i\}, \quad 0 < \alpha < 1. \tag{30}
\]

This is the same as the \( \alpha \) potential defined in the standard potential theory [11]. When \( \alpha = 1 \), the above expression goes to infinity. Thus, (7) can be viewed as the extension of the \( \alpha \) potential to the case of \( \alpha = 1 \).

### 5.2 An Example

We use an example to illustrate the application of the single sample path-based approach.

**Example 1** A manufacturing system consists of two machines and \( N \) pieces of works, which are circulating between the two machines, as shown in Figure 11.1. Each work piece has to undertake three consecutive operations at machine 1; thus machine 1 is illustrated by three circles in the figure. The service rates of these three operations are \( \lambda_1 \), \( \lambda_2 \), and \( \lambda_3 \), respectively. Machine 2 has only one operation with service rate \( \lambda_4 \). The system can be modeled as a Markov process with state denoted as \((n, i)\), where \( n \) is the number of pieces in machine 1 and \( i = 1, 2, 3 \) denotes the operation that the piece at machine 1 is undertaking. A work piece, after the completion of service at machine 1, goes to machine 2 with probability \( p^\alpha(n) \) and feeds back to machine 1 with probability \( 1 - p^\alpha(n) \). The subscript "\( \alpha \)" represents an action with \( \alpha \in A \). For any \( \alpha \in A \) and \( n = 1, 2, \cdots, N \), \( p^\alpha(n) \in [0, 1] \). The performance to be optimized is the weighted sum of the two machines' throughputs. (The cost function \( f \) does not depend on the actions.) Some transition probabilities of the Markov chain embedded at the operation completion times are (for \( 0 < n < N \)):

\[
p[(n, 1), (n + 1, 1)] = \frac{\lambda_4}{\lambda_1 + \lambda_4},
\]

\[
p[(n, 1), (n, 2)] = \frac{\lambda_1}{\lambda_1 + \lambda_4},
\]

\[
p^\alpha[(n, 3), (n + 1, 3)] = \frac{\lambda_3}{\lambda_3 + \lambda_4},
\]

\[
p^\alpha[(n, 3), (n - 1, 1)] = \frac{\lambda_3}{\lambda_3 + \lambda_4} p^\alpha(n),
\]

\[
p^\alpha[(n, 3), (n, 1)] = \frac{\lambda_3}{\lambda_3 + \lambda_4} [1 - p^\alpha(n)],
\]
Other nonzero transition probabilities have a similar form.

The transitions from states \((n, 1)\) and \((n, 2)\) do not depend on actions. For state \((n, 3)\), the optimality equation is

\[
\frac{\lambda_4}{\lambda_3 + \lambda_4} g^\alpha(n + 1, 3) + \frac{\lambda_3}{\lambda_3 + \lambda_4} p^\alpha(n) g^\alpha(n - 1, 1) \\
+ \frac{\lambda_3}{\lambda_3 + \lambda_4} [1 - p^\alpha(n)] g^\alpha(n, 1) \\
\leq \frac{\lambda_4}{\lambda_3 + \lambda_4} g^\alpha(n + 1, 3) + \frac{\lambda_3}{\lambda_3 + \lambda_4} p^{\alpha'}(n) g^\alpha(n - 1, 1) \\
+ \frac{\lambda_3}{\lambda_3 + \lambda_4} [1 - p^{\alpha'}(n)] g^\alpha(n, 1),
\]

for all \(\alpha' \in \mathcal{A}\). This is equivalent to

\[
[p^\alpha(n) - p^{\alpha'}(n)] g^\alpha(n - 1, 1) - [p^\alpha(n) - p^{\alpha'}(n)] g^\alpha(n, 1) \leq 0. \tag{31}
\]

In (31), only the action related probabilities are involved; the system parameters \(\lambda_1, \lambda_2, \lambda_3,\) and \(\lambda_4\) do not appear.

In the above example, the service rates govern the evolution of the system, which runs autonomously. The control action can effect only some of the system transitions. The transition probabilities corresponding to the uncontrolled transitions are the same under all policies; they cancel each other in the Bellman equation and hence do not appear in the final form. Since we can estimate \(g^\alpha\)'s on a sample path, we can implement policy iteration without knowing these transition probabilities by observing a sample path.

Many practical systems have the same feature as the above example. Indeed, for many such systems control can be exercised only at a very
limited region (e.g., the admission control can be applied only at the access points of a high-speed communication network); the rest part of the systems simply evolves by its own nature. The dashed box in Figure 11.1 can also be viewed as a machine whose service time has an Erlangian distribution, in such cases the transitions between the three stages are not controllable. This type of service distribution and the more general forms, such as the coxian distribution and the phase-type distributions, are very common in practical systems.

In summary, Example 1 and the above discussion illustrate that the sample-path-based approach has the following advantages.

1 Given a sample path, policy iteration can be implemented without knowing the whole transition matrix; only those items related to control actions have to be known. Matrix inversion is not required.

2 The approach saves memory spaces required for implementing MDPs. Generally speaking, only the $M$ potentials, not the $M \times M$ transition matrix, have to be stored. This can be further reduced when there are some states which cannot be reached by controlled states. As shown in (31), in Example 1 only $g^a(n, 1), n = 0, 1, \ldots, N$, have to be estimated and stored; $g^a(n, 2)$ and $g^a(n, 3), n = 0, 1, \ldots, N$, do not even need to be estimated.

3 Different from the standard computational approach where all the potentials are obtained altogether by a matrix inversion, in the sample path based approach potentials can be estimated one by one. This feature brings in a number of possible applications.

(a) The computational efforts and memory spaces of each iteration may be further reduced at the cost of the convergence rate. The idea is, if the state space is too large, at each iteration we can estimate the potentials of only a subset of the state space and update the actions for the states in this subset. For instance, in Example 1 we may set $0 = n_0 < n_1 < n_2 < \cdots < n_{k-1} < n_k = M$. Then in the $i$th iteration, $i = 1, 2, \ldots, k$, we may estimate $g^a(n, 1)$ only for $n_{i-1} - 1 \leq n \leq n_i, i = 1, 2, \cdots, k$. Of course, it may need more iterations to reach the optimal policy; however, at each iteration the computation and the memory requirements may be reduced to fit the capacity of the available computing equipments. This feature may be important for on-line optimization using special designed hardwares which may have limited capacities. In high speed communication networks, the effect
of a slow convergence rate in terms of iterations may be compensated by the fast speed in system evolution.

(b) For many practical systems, we may have some a priori knowledge about which states are more important than others. Then we can estimate only the potentials for the states that are needed for updating the actions on these important states. This may reduce the computation and memory at the cost of the best performance achieved.

6. **Reinforcement Learning**

As shown in the last section, potentials can be estimated on a single sample path. More efficient algorithms can be developed with stochastic approximation methods. For example, from (7), we have

$$g(i) = E\{\sum_{l=0}^{\infty} [f(X_l) - \eta]|X_0 = i]\}.$$

Given a sample path \(\{X_0, \cdots, X_n, X_{n+1}, \cdots, \}\), at the \(n\)th transition, we have

$$g(X_n) = E\{\sum_{l=0}^{\infty} [f(X_{n+l}) - \eta]|X_n\} = [f(X_n - \eta) + E[g(X_{n+1})]] \quad (32)$$

Applying stochastic approximation methods to (32) leads to the algorithm

$$g(X_n) := g(X_n) + \gamma [f(X_n) - \eta + g(X_{n+1}) - g(X_n)], \quad 0 < \gamma < 1,$$

where \(\gamma\) is the step size. Set

$$d_n = [f(X_n) - \eta + g(X_{n+1}) - g(X_n)].$$

We have

$$g(X_n) := g(X_n) + \gamma d_n. \quad (33)$$

More generally, we have

$$g(X_n) := g(X_n) + \gamma \sum_{m=n}^{\infty} \lambda^{m-n} d_n, \quad 0 \leq \lambda \leq 1. \quad (34)$$

This algorithm is known as TD(\(\lambda\)) [3, 24, 25]. Setting \(\lambda = 0\) (using the convention \(0^0 = 1\)) leads to TD(0) shown in (33). Different variants of TD(\(\lambda\)) exist and convergence can be proved.
When the system structure, i.e., the transition probabilities, \( p(i, j) \)'s, are completely unknown, we cannot apply policy iteration directly by using the potentials. (As shown in Example 1, we need to know at least the transition probabilities related to the actions, i.e., \( p^\alpha(n) \)) to implement policy iteration with estimated potentials.) In this case, we can estimate the Q-factor defined as \([3, 25]\)

\[
Q(i, \alpha) = \left\{ \sum_{j=1}^{M} p^\alpha(i, j)g(j) \right\} + f^\alpha(i) - \eta,
\]

(35)

for every state-action pair \((i, \alpha)\). From Theorem 1 we can choose \( \alpha = \arg\{\min_{\alpha' \in A}[Q(i, \alpha')] \} \) as the action at state \( i \) in the policy iteration. However, there is one problem for this approach: it is impossible to estimate \( Q(i, \alpha') \) on the current sample path with \( \alpha \neq \alpha' \) being the action taken at state \( i \), because the pair \((i, \alpha')\) does not appear on the sample path. One way to obtain the estimation of \( Q(i, \alpha') \), \( \alpha' \neq \alpha \), is to apply the important sampling technique. This requires to know the ratios of \( p^\alpha(i, j) \) and \( p^\alpha(i, j) \). In Example 1, this ratio can be obtained if \( p^\alpha(n) \) and \( p^\alpha(n) \) are known. This shows that policy iteration based on Q-factors is almost the same as that based on potentials.

The value iteration of Q-factor, however, leads to the optimal Q-factor and can be implemented when the system structure is completely unknown. This is briefly explained as follows. From the definition (35), we have

\[
\eta + Q(i, \alpha) = \left\{ \sum_{j=1}^{M} p^\alpha(i, j)g(j) \right\} + f^\alpha(i),
\]

(36)

Taking minimum on both sizes, we get

\[
\eta + \min_{\alpha \in A} Q(i, \alpha) = \min_{\alpha \in A} \left\{ \sum_{j=1}^{M} p^\alpha(i, j)g(j) + f^\alpha(i) \right\},
\]

Comparing with the Bellman equation

\[
\eta + g(i) = \min_{\alpha \in A} \left\{ \sum_{j=1}^{M} p^\alpha(i, j)g(j) + f^\alpha(i) \right\},
\]

we conclude that at the optimal policy \( g(i) = \min_{\alpha \in A} Q(i, \alpha) \). Substituting this into (36), we have at the optimal policy

\[
\eta + Q(i, \alpha) = \left\{ \sum_{j=1}^{M} p^\alpha(i, j)[\min_{\alpha \in A} Q(j, \alpha)] \right\} + f^\alpha(i).
\]

(37)
This is the Bellman equation for Q-factors. Based on (37), the Robbins-Monro stochastic approximation method leads to

\[ Q(i, \alpha) := (1 - \gamma)Q(i, \alpha) + \gamma[\min_{\beta \in A} Q(j, \beta)] + f^{\alpha}(i) - \eta, \quad 0 < \gamma < 1. \]  

(38)

This is the Q-learning algorithm, which can be applied to any sample path: when the system jumps from state \(i\) to \(j\), the Q-factor is updated according to (38). It can be proved that (38) converges to the optimal \(Q^* (i, \alpha)\) and \(\alpha^* = \arg\{\min_{\alpha' \in \mathcal{A}} [Q^* (i, \alpha')]\}\) is the optimal action at state \(i\).

Compared with the approach based on potentials, Q-learning can be applied to systems where the structure is completely unknown. However, from (38) it is clear that Q-learning requires the sample path visit every state-action pair. In addition, the number of Q-factors increases to \(M \times K\) (where \(K\) is the number of possible actions at each state).

In addition to the above approaches, Neuro-Dynamic programming is proposed to overcome the difficulty of the so-called "curse of dimensionality". Roughly speaking, in neuro-dynamic programming, we try to approximate the potential function \(g(i)\) by \(g(i, r)\), with a continuous parameter \(r\). This generally involves with two steps:

1. develop an approximation architecture, e.g., a neuro-network, to represent \(g(i, r)\),

2. find a training algorithm for updating the parameter vector \(r\), based on the information observed on a sample path.

After training, the parameter \(r\) reaches a proper value. The neuro-network will output an approximate value of \(g(i)\) for an input integer \(i\). For details and successful examples, see [3].

7. Discussions and Conclusions

Figure 11.2 illustrates the relation among PA, MDPs, and RL. All these areas are related to performance improvement and optimization. PA provides performance sensitivity with respect to both continuous and discrete parameters. Policy iteration in MDPs can be viewed as a direct consequence of discrete performance sensitivity provided by PA. This new view of PA and MDPs leads to the gradient-based policy iteration method that can be applied to some nonstandard problems such as those with correlated actions at different states. Performance potential plays an important role in both PA and MDPs; it also offers a clear intuitive interpretation for many results. Reinforcement learning, TD(\(\lambda\)), neuro-dynamic programming, etc, are efficient ways of estimating the
PA: provide performance sensitivity based on a sample path

$g(i)$: Pertentials
$Q(i, \alpha)$: Q-factors

Gradient: Policy iter.

MDP: make optimal decision at each system state

Learning: learn how to improve perf. by observing system behavior

Figure 11.2. The Relations Among PA, MDP and RL
performance potentials and related quantities based on sample paths. In particular, Q-learning can be used when the system structure is unknown. The potentials and Q-factors can be implemented on a single sample path; the sample path-based approach leads to on-line optimization schemes that are of practical importance to real world engineering problems such as the optimization of manufacturing and communication systems.

The gradient-based policy iteration method is a new topic for further research. Sample-path-based and Q-learning-type of algorithms can be developed to implement this new policy iteration scheme.

References


