Rollout, Approximate Policy Iteration, and Distributed Reinforcement Learning

by

Dimitri P. Bertsekas

Chapter 4
Approximate Policy Iteration for Infinite Horizon Problems

These notes represent “work in progress,” and will be periodically updated. They more than likely contain errors (hopefully not serious ones). Furthermore, the references to the literature are incomplete. Your comments and suggestions to the author at dbertsek@asu.edu are welcome.

February, 2020
4

Approximate Policy Iteration for Infinite Horizon Problems

<table>
<thead>
<tr>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1. Stochastic Shortest Path and Discounted Problems</td>
</tr>
<tr>
<td>4.1.1. Policy Iteration</td>
</tr>
<tr>
<td>4.1.2. Multistep and Optimistic Policy Iteration</td>
</tr>
<tr>
<td>4.2. Approximation in Value Space - Performance Bounds</td>
</tr>
<tr>
<td>4.2.1. Limited Lookahead</td>
</tr>
<tr>
<td>4.2.2. Rollout</td>
</tr>
<tr>
<td>4.2.3. Approximate Policy Iteration</td>
</tr>
<tr>
<td>4.3. Asynchronous Distributed Algorithms in Dynamic Programming</td>
</tr>
<tr>
<td>4.4. Asynchronous Distributed Policy Iteration</td>
</tr>
<tr>
<td>4.5. Approximate Policy Iteration - Partitioned Architectures</td>
</tr>
<tr>
<td>4.6. Approximate Policy Iteration - Partial Observation Problems</td>
</tr>
<tr>
<td>4.7. Notes and Sources</td>
</tr>
</tbody>
</table>
In this chapter, we provide an introduction to the theory of infinite horizon problems. We focus on exact solution by DP methods, leaving the discussion of approximations to subsequent chapters. Infinite horizon problems differ from their finite horizon counterparts in two main respects:

(a) The number of stages is infinite.

(b) The system is stationary, i.e., the system equation, the cost per stage, and the disturbance probability distributions do not change from one stage to the next.

The assumption of an infinite number of stages is hardly ever satisfied in practice, but is a reasonable approximation for problems involving a finite but very large number of stages. The assumption of stationarity is often satisfied in practice, and in other cases it approximates well a situation where the system parameters vary relatively slowly with time. Moreover, nonstationary (including periodic) problems can be converted to stationary ones using a conceptually simple reformulation; see [Ber12], Section 4.6.

Infinite horizon problems give rise to elegant and insightful analysis, and their optimal policies are often simpler than their finite horizon counterparts. For example, optimal policies are typically stationary, i.e., the optimal rule for choosing controls does not change over time.

On the other hand, infinite horizon problems generally require a more sophisticated mathematical treatment. Our discussion will be limited to relatively simple finite-state problems (the theory of problems with an infinite state space is considerably more intricate; see the books [BeS78], [Ber12], [Ber18a]). Still some theoretical results are needed in this chapter. They will be explained intuitively to the extent possible, and their mathematical proofs will be provided in the end-of-chapter appendix.

An Overview of Infinite Horizon Problems

We will focus on two types of infinite horizon problems, where we aim to minimize the total cost over an infinite number of stages, given by

$$ J_\pi(x_0) = \lim_{N \to \infty} \mathbb{E}_{k=0,1,\ldots} \left\{ \sum_{k=0}^{N-1} \alpha^k g(x_k, \mu_k(x_k), w_k) \right\}; $$

see Fig. 4.0.1. Here, $J_\pi(x_0)$ denotes the cost associated with an initial state $x_0$ and a policy $\pi = \{\mu_0, \mu_1, \ldots\}$, and $\alpha$ is a scalar in the interval $[0, 1]$. When $\alpha$ is strictly less than 1, it has the meaning of a discount factor, and its effect is that future costs matter to us less than the same costs incurred at the present time.

Thus the infinite horizon cost of a policy is the limit of its finite horizon cost as the horizon tends to infinity. (We assume that the limit exists and is finite for the moment, and address the issue later.) The two types of problems, considered in Sections 4.2 and 4.3, respectively, are:
(a) **Stochastic shortest path problems** (SSP for short). Here, $\alpha = 1$ but there is a special cost-free termination state that we aim to reach.

(b) **Discounted problems**. Here, $\alpha < 1$ and there need not be a termination state. However, we will see that a discounted problem can be readily converted to an SSP problem. This can be done by introducing an artificial termination state to which the system moves with probability $1 - \alpha$ at every stage, thus making termination inevitable. As a result, our algorithms and analysis for SSP problems can be easily adapted to discounted problems.

**A Preview of Infinite Horizon Theory**

There are several analytical and computational issues regarding our infinite horizon problems. Many of them revolve around the relation between the optimal cost function $J^*$ of the infinite horizon problem and the optimal cost functions of the corresponding $N$-stage problems.

In particular, consider the undiscounted case and let $J_N(x)$ denote the optimal cost of the problem involving $N$ stages, initial state $x$, cost per stage $g(x, u, w)$, and zero terminal cost. This cost is generated after $N$ iterations of the DP algorithm

$$J_{k+1}(x) = \min_{u \in U(x)} E \left\{ g(x, u, w) + J_k(f(x, u, w)) \right\}, \quad k = 0, 1, \ldots, (4.1)$$

starting from the initial condition $J_0(x) = 0$ for all $x$.† The algorithm (4.1) is known as the value iteration algorithm (VI for short). Since the infinite horizon DP algorithm of Chapter 1. However, we have reversed the time indexing to suit our purposes. Thus the index of the cost functions produced by the algorithm is incremented with each iteration, and not decremented as in the case of finite horizon.

† This is just the finite horizon DP algorithm of Chapter 1. However, we have reversed the time indexing to suit our purposes. Thus the index of the cost functions produced by the algorithm is incremented with each iteration, and not decremented as in the case of finite horizon.
horizon cost of a given policy is, by definition, the limit of the corresponding $N$-stage costs as $N \to \infty$, it is natural to speculate that:

1. The optimal infinite horizon cost is the limit of the corresponding $N$-stage optimal costs as $N \to \infty$; i.e.,

$$J^*(x) = \lim_{N \to \infty} J_N(x)$$

(4.2)

for all states $x$.

2. The following equation should hold for all states $x$,

$$J^*(x) = \min_{u \in U(x)} E \left\{ g(x, u, w) + J^*(f(x, u, w)) \right\}.$$  (4.3)

This is obtained by taking the limit as $N \to \infty$ in the VI algorithm (4.1) using Eq. (4.2). Equation (4.3) is really a system of equations (one equation per state $x$), which has as solution the optimal costs-to-go of all the states. It can also be viewed as a *functional equation* for the optimal cost function $J^*$, and it is called *Bellman’s equation*.

3. If $\mu(x)$ attains the minimum in the right-hand side of the Bellman equation (4.3) for each $x$, then the policy $\{\mu, \mu, \ldots\}$ should be optimal. This type of policy is called *stationary*. Intuitively, optimal policies can be found within this class of policies, since the future optimization problem when starting at a given state looks the same regardless of the time when we start.

All three of the preceding results hold for SSP problems under our assumptions, as we will state later in Section 4.2 and prove in the appendix to this chapter. They also hold for discounted problems in suitably modified form that incorporates the discount factor. In fact the algorithms and analysis of this chapter are quite similar for discounted problems and SSP problems (under our assumptions), to the point where we may discuss a particular method for one of the two problems with the understanding that its application to the other problem can be straightforwardly adapted.

**Transition Probability Notation for Finite-State Infinite Horizon Problems**

Throughout this chapter we assume a finite-state discrete-time dynamic system, and we will use a special transition probability notation that is suitable for such a system. We generally denote states by the symbol $i$ and successor states by the symbol $j$. We will assume that there are $n$ states (in addition to the termination state for SSP problems). These states are denoted $1, \ldots, n$, and the termination state is denoted $t$. The control $u$ is constrained to take values in a given finite constraint set $U(i)$, which may
Sec. 4.1 Stochastic Shortest Path and Discounted Problems

depend on the current state $i$. The use of a control $u$ at state $i$ specifies the transition probability $p_{ij}(u)$ to the next state $j$, at a cost $g(i,u,j).$†

Given an admissible policy $\pi = \{\mu_0, \mu_1, \ldots\}$ [one with $\mu_k(i) \in U(i)$ for all $i$ and $k$] and an initial state $i_0$, the system becomes a Markov chain whose generated trajectory under $\pi$, denoted $\{i_0, i_1, \ldots\}$, has a well-defined probability distribution. The total expected cost associated with an initial state $i$ is

$$J_\pi(i) = \lim_{N \to \infty} E \left\{ \sum_{k=0}^{N-1} \alpha^k g(i_k, \mu_k(i_k), i_{k+1}) \mid i_0 = i, \pi \right\},$$

where $\alpha$ is either 1 (for SSP problems) or less than 1 for discounted problems. The expected value is taken with respect to the joint distribution of the states $i_1, i_2, \ldots$, conditioned on $i_0 = i$ and the use of $\pi$. The optimal cost from state $i$, i.e., the minimum of $J_\pi(i)$ over all policies $\pi$, is denoted by $J^*(i).$‡

The cost function of a stationary policy $\pi = \{\mu, \mu, \ldots\}$ is denoted by $J_\mu(i)$. For brevity, we refer to $\pi$ as the stationary policy $\mu$. We say that $\mu$ is optimal if

$$J_\mu(i) = J^*(i) = \min_\pi J_\pi(i), \quad \text{for all states } i.$$

As noted earlier, under our assumptions, we will show that there will always exist an optimal policy, which is stationary.

4.1 STOCHASTIC SHORTEST PATH AND DISCOUNTED PROBLEMS

We first consider the stochastic shortest path (SSP for short) problem, where we assume that there is no discounting ($\alpha = 1$), and that there is a special cost-free termination state $t$. Once the system reaches that state, it remains there at no further cost, i.e.,

$$p_{tt}(u) = 1, \quad g(t, u, t) = 0, \quad \text{for all } u \in U(t).$$

† To convert from the transition probability format to the system equation format used in the preceding chapters, we can simply use the system equation $x_{k+1} = u_k$, where $u_k$ is the disturbance that takes values according to the transition probabilities $p_{x_k u_k}(u_k)$.

‡ Because of the underlying Markov chain, stochastic DP problems are called Markovian decision problems by many authors, including Bellman. We will use instead the term “DP problem” to refer to both deterministic and stochastic problems.
We denote by $1, \ldots, n$ the states other than the termination state $t$; see Fig. 4.1.1.

With this notation, the Bellman equation (4.3) and the VI algorithm (4.1) take the following form.

**Bellman Equation and Value Iteration for SSP Problems:**

For all $i = 1, \ldots, n$, we have

$$J^*(i) = \min_{u \in U(i)} \left[ p_{it}(u)g(i, u, t) + \sum_{j=1}^{n} p_{ij}(u) (g(i, u, j) + J^*(j)) \right].$$

(4.4)

For all $i = 1, \ldots, n$, and any initial conditions $J_0(1), \ldots, J_0(n)$, the VI algorithm generates the sequence $\{J_k\}$ according to

$$J_{k+1}(i) = \min_{u \in U(i)} \left[ p_{it}(u)g(i, u, t) + \sum_{j=1}^{n} p_{ij}(u) (g(i, u, j) + J_k(j)) \right].$$

The right-hand bracketed expression in the Bellman equation (4.4) represents an expected value, which is similar to the one we have seen in earlier DP expressions. It is the sum of three terms:

(a) The contribution

$$p_{it}(u)g(i, u, t)$$
to the expected cost of the current stage of the terminating $i$-to-$t$
transition.

(b) The contribution
\[ \sum_{j=1}^{n} p_{ij}(u)g(i, u, j) \]
to the expected cost of the current stage of the nonterminating $i$-to-$j$
transitions.

(c) The optimal expected cost-to-go
\[ \sum_{j=1}^{n} p_{ij}(u)J^*(j) \]
starting from the next state $j$ [the optimal cost of $t$ is $J^*(t) = 0$, so
it does not appear in the preceding sum].

Note that the deterministic shortest path problem of Section 1.3.1 is
obtained as the special case of the SSP problem where for each state-control
pair $(i, u)$, the transition probability $p_{ij}(u)$ is equal to 1 for a unique state
$j$ that depends on $(i, u)$. Moreover, any deterministic or stochastic finite-
state, finite horizon problem with a termination state (cf. Section 1.3.3)
can be converted to an SSP problem. In particular, the reader may verify
that the finite-state $N$-step horizon stochastic DP problem of Section 1.2
can be obtained as a special case of an SSP problem by viewing as state
the pair $(x_k, k)$ and introducing an artificial zero-cost transition from each
pair $(x_N, N)$ to an artificial termination state $t$.

We are interested in problems where reaching the termination state $t$
is inevitable. Thus, the essence of the problem is to reach $t$ with minimum
expected cost. Throughout this chapter, when discussing SSP problems,
we will make the following assumption, which will be shown to guarantee
eventual termination under all policies.†

† The main analytical and algorithmic results for SSP problems are valid
under more general conditions, which involve the notion of a proper policy (see
the end-of-chapter references). In particular, a stationary policy is called proper
if starting from every state, it is guaranteed to eventually reach the destination.
The policy is called improper if it is not proper.

The subsequent four propositions can also be shown under the weaker as-
sumption that there is at least one proper policy, and furthermore, every improper
policy is “bad” in the sense that it results in infinite expected cost from at least
one initial state (see [BeT89], [BeT91], or [Ber12], Chapter 3). These assump-
tions, when specialized to deterministic shortest path problems, are similar to
the assumptions of Section 1.3.1. They imply that there is at least one path to
the destination from every starting state and that all cycles have positive cost.
Assumption 4.1.1: (Termination is Inevitable Under All Policies) There exists an integer $m$ such that regardless of the policy used and the initial state, there is positive probability that the termination state will be reached after no more than $m$ stages; i.e., for all admissible policies $\pi$ we have

$$\rho_\pi = \max_{i=1,\ldots,n} P\{x_m \neq t \mid x_0 = i, \pi\} < 1.$$ 

Let $\rho$ be the maximum probability of not reaching $t$ within $m$ stages, over all starting states and policies:

$$\rho = \max_\pi \rho_\pi.$$ 

Note that $\rho_\pi$ depends only on the first $m$ components of the policy $\pi$. Furthermore, since the number of controls available at each state is finite, the number of distinct $m$-stage policies is also finite. Thus there can be only a finite number of distinct values of $\rho_\pi$, so that $\rho < 1$. This implies that the probability of not reaching $t$ over a finite horizon diminishes to 0 as the horizon becomes longer, regardless of the starting state and policy.

To see this, note that for any $\pi$ and any initial state $i$

$$P\{x_{2m} \neq t \mid x_0 = i, \pi\} = P\{x_{2m} \neq t \mid x_m \neq t, x_0 = i, \pi\} \cdot P\{x_m \neq t \mid x_0 = i, \pi\} \leq \rho^2.$$ 

More generally, for each $\pi$, the probability of not reaching the termination state after $km$ stages diminishes like $\rho^k$ regardless of the initial state, i.e.,

$$P\{x_{km} \neq t \mid x_0 = i, \pi\} \leq \rho^k, \quad i = 1, \ldots, n. \quad (4.5)$$

This implies that the limit defining the associated total cost vector $J_\pi$ exists and is finite, and is central in the proof of the following results.

In the absence of these assumptions, serious complications may occur. In particular, Bellman’s equation may have no solution or an infinite number of solutions. Moreover, it may have a unique solution, which is different than $J^*$; see [BeY16], and [Ber18a], Section 3.1.2. The book [Ber18a], Section 3.1.1, discusses a simple example, which in addition to $t$, involves a single state 1 at which we can either stay at cost $a$ or move to $t$ at cost $b$; anomalies occur when $a = 0$ and when $a < 0$. For analysis and extensions for different or more general SSP problems, see [DeR79], [Pat01], [Pat07], [Ber16b], [Ber18a], [Ber18e], [GuS18], [Ber19b].
We now describe the main theoretical results for SSP problems; the proofs are given in the appendix to this chapter. Our first result is that the infinite horizon version of the DP algorithm, which is VI [cf. Eq. (4.1)], converges to the optimal cost function $J^*$. The optimal cost $J^*(t)$ starting from $t$ is of course 0, so it is just neglected where appropriate in the subsequent discussion. Generally, $J^*$ is obtained in the limit, after an infinite number of iterations. However, there are important cases where convergence is obtained in finitely many iterations (see [Ber12], Chapter 3; a simple example is finite horizon problems that are formulated as SSP problems by adding an artificial termination state at the end of the horizon).

**Proposition 4.1.1: (Convergence of VI)** The optimal cost $J^*(i)$ is finite for each $i$. Moreover, given any initial conditions $J_0(1), \ldots, J_0(n)$, the sequence $\{J_k(i)\}$ generated by the VI algorithm

$$J_{k+1}(i) = \min_{u \in U(i)} \left[ p_{iu}(u)g(i, u, t) + \sum_{j=1}^{n} p_{ij}(u)(g(i, u, j) + J_k(j)) \right],$$

converges to $J^*(i)$ for each $i = 1, \ldots, n$.

Our next result is that the limiting form of the DP equation, Bellman’s equation, has $J^*$ as its unique solution.

**Proposition 4.1.2: (Bellman’s Equation)** The optimal cost function

$$J^* = (J^*(1), \ldots, J^*(n))$$

satisfies for all $i = 1, \ldots, n$, the equation

$$J^*(i) = \min_{u \in U(i)} \left[ p_{iu}(u)g(i, u, t) + \sum_{j=1}^{n} p_{ij}(u)(g(i, u, j) + J^*(j)) \right],$$

and is the unique solution of this equation.

The next proposition is a special case of the preceding two. It expresses that by restricting attention to a single policy $\mu$, we obtain a VI algorithm and a Bellman equation that are specific to $\mu$. 

Proposition 4.1.3: (VI and Bellman’s Equation for Policies)
For any stationary policy $\mu$, the corresponding cost function $J_\mu = (J_\mu(1), \ldots, J_\mu(n))$ satisfies for all $i = 1, \ldots, n$, the equation

$$J_\mu(i) = p_{it}(\mu(i)) g(i, \mu(i), t) + \sum_{j=1}^{n} p_{ij}(\mu(i)) \left( g(i, \mu(i), j) + J_\mu(j) \right),$$

and is the unique solution of this equation. Furthermore, given any initial conditions $J_0(1), \ldots, J_0(n)$, the sequence $\{J_k(i)\}$ generated by the VI algorithm that is specific to $\mu$,

$$J_{k+1}(i) = p_{it}(\mu(i)) g(i, \mu(i), t) + \sum_{j=1}^{n} p_{ij}(\mu(i)) \left( g(i, \mu(i), j) + J_k(j) \right),$$

converges to the cost $J_\mu(i)$ for each $i$.

Our final result provides a necessary and sufficient condition for optimality of a stationary policy.

Proposition 4.1.4: (Optimality Condition) A stationary policy $\mu$ is optimal if and only if for every state $i$, $\mu(i)$ attains the minimum in the Bellman equation (4.6).

Bellman Operators and Contraction Properties

We will now provide an insightful mathematical result about SSP problems, which is proved in the appendix with the aid of the preceding example. To this end let us introduce for any vector $J = (J(1), \ldots, J(n))$, the notation

$$(TJ)(i) = \min_{u \in U(i)} \left[ p_{it}(u) g(i, u, t) + \sum_{j=1}^{n} p_{ij}(u) \left( g(i, u, j) + J(j) \right) \right],$$

(4.7)

for all $i$, and

$$(T_\mu J)(i) = p_{it}(\mu(i)) g(i, \mu(i), t) + \sum_{j=1}^{n} p_{ij}(\mu(i)) \left( g(i, \mu(i), j) + J(j) \right),$$

(4.8)

for all $\mu$ and $i$. Here $T$ and $T_\mu$ are the DP operators (or Bellman operators as we will call them), which map the vector $J$ into the vectors

$$TJ = ((TJ)(1), \ldots,(TJ)(n)), \quad T_\mu J = ((T_\mu J)(1), \ldots,(T_\mu J)(n)).$$
Section 4.1 Stochastic Shortest Path and Discounted Problems

respectively. This is a very helpful shorthand notation that greatly simplifies the statement and proofs of analytical and algorithmic results. For example, Bellman’s equations can be written in terms of these operators as the fixed point equations $J^* = TJ^*$ and $J_\mu = T_\mu J_\mu$.

The next proposition states that $T$ and $T_\mu$ are contraction mappings, so the unique fixed point property of this mapping follows from general mathematical results about contraction mappings (see e.g., [Ber12], [Ber18a]). Moreover the contraction property provides a convergence rate estimate for VI, and is the basis for further analysis of exact and approximate methods for SSP problems (see the author’s monograph [Ber18a] for a theoretical development of DP, which is based on fixed point theory and an abstract Bellman operator viewpoint).

Proposition 4.1.5: (Contraction Property of the Bellman Operator) The Bellman operator $T$ of Eq. (4.7) is a contraction mapping with respect to some weighted norm

$$\|J\| = \max_{i=1,\ldots,n} \frac{|J(i)|}{v(i)},$$

defined by some vector $v = (v(1), \ldots, v(n))$ with positive components. In other words, there exists a positive scalar $\rho < 1$ such that for any two $n$-dimensional vectors $J$ and $J'$, we have

$$\|TJ - TJ'\| \leq \rho \|J - J'\|.$$

The same is true for $T_\mu$, the Bellman operator (4.8) of any policy $\mu$.

Among others, the preceding contraction property provides a convergence rate estimate for VI, namely that the generated sequence $\{J_k\}$ satisfies

$$\|J_k - J^*\| \leq \rho^k \|J_0 - J^*\|.$$

This follows from the fact that $J_k$ and $J^*$ can be viewed as the results of the $k$-fold application of $T$ to the vectors $J_0$ and $J^*$, respectively.

Bellman Equation and Value Iteration for Q-Factors

The results just given have counterparts involving Q-factors. The optimal Q-factors are defined for all $i = 1, \ldots, n$, and $u \in U(i)$ by

$$Q^*(i, u) = p_d(u)g(i, u, t) + \sum_{j=1}^{n} p_{ij}(u)(g(i, u, j) + J^*(j)).$$
As in the finite horizon case, $Q^*(i, u)$ can be interpreted as the cost of starting at $i$, using $u$ for the first stage, and using an optimal policy afterwards. Once $Q^*$ is computed by some method, an optimal policy $\mu^*$ can be obtained from the minimization

$$\mu^*(i) \in \arg \min_{u \in U(i)} Q^*(i, u), \quad i = 1, \ldots, n.$$ 

Similarly, if approximately optimal Q-factors $\tilde{Q}(i, u)$ are obtained by some method (model-based or model-free), a suboptimal policy $\tilde{\mu}$ can be obtained from the minimization

$$\tilde{\mu}(i) \in \arg \min_{u \in U(i)} \tilde{Q}(i, u), \quad i = 1, \ldots, n.$$ 

Our basic results relating Bellman’s equation and the VI algorithm are stated as follows.

**Bellman Equation and Value Iteration for Q-Factors and SSP Problems:**

For all $i = 1, \ldots, n$, and $u \in U(i)$ we have

$$Q^*(i, u) = p_i(u)g(i, u, t) + \sum_{j=1}^{n} p_{ij}(u) \left( g(i, u, j) + \min_{v \in U(j)} Q^*(j, v) \right).$$

For all $i = 1, \ldots, n$, and $u \in U(i)$, and any initial conditions $Q_0(i, u)$, the VI algorithm generates the sequence $\{Q_k\}$ according to

$$Q_{k+1}(i, u) = p_i(u)g(i, u, t) + \sum_{j=1}^{n} p_{ij}(u) \left( g(i, u, j) + \min_{v \in U(j)} Q_k(j, v) \right).$$

Actually, the optimal Q-factors $Q^*(i, u)$ can be viewed as optimal state costs associated with a modified SSP problem, which involves a new state for each pair $(i, u)$ with transition probabilities $p_{ij}(u)$ to the states $j = 1, \ldots, n, t$; see Fig. 4.1.2. Then the preceding Bellman equation for the optimal Q-factors, together with the Bellman equation (4.6) for the optimal costs $J^*(j)$, can be viewed as the Bellman equation for the modified SSP problem.

**Temporal Differences and Cost Shaping**

Bellman’s equation can be written in an alternative form, which involves the differential

$$\dot{J} = J^* - V,$$
Sec. 4.1 Stochastic Shortest Path and Discounted Problems

185

State-Control Pairs

\( (i, u) \)

\( g(i, u, j) \)

\( p_{ij}(u) \)

\( j \)

Control \( v \)

Cost = 0

\( (j, v) \)

States

Figure 4.1.2 States, transition probabilities, and stage costs corresponding to a modified SSP problem, which yields the optimal Q-factors as well as the optimal costs. The states of this problem are the pairs \((i, u)\), \(u \in U(i)\), the original problem states \(i = 1, \ldots , n\), and the termination state \(t\). A control \(v \in U(j)\) is available only at the original system states \(j\), leading to the pair \((j, v)\) at cost 0. The transition from a pair \((i, u)\) leads to \(j\) with probability \(p_{ij}(u)\) and cost 0. The Bellman equation for this modified problem is

\[
Q^*(i, u) = p_{it}(u)g(i, u, t) + \sum_{j=1}^{n} p_{ij}(u) \left( g(i, u, j) + \min_{v \in U(j)} Q^*(j, v) \right),
\]

for the states \((i, u), u \in U(i)\), and

\[
J^*(j) = \min_{v \in U(j)} Q^*(j, v),
\]

for the states \(j = 1, \ldots , n\). Note that a policy \(\mu\) for this problem leads from a state \(j\) to the state \((j, \mu(j))\), so in any system trajectory, only pairs of the form \((j, \mu(j))\) are visited after the first transition.

where \(V = (V(1), \ldots , V(n))\) is any \(n\)-dimensional vector and \(V(t) = 0\). In particular, by subtracting \(V(i)\) from both sides of the Bellman equation (4.6), and adding and subtracting \(V(j)\) within the right-hand side summation, we obtain

\[
\hat{J}(i) = \min_{u \in U(i)} \left[ p_{it}(u)\hat{g}(i, u, t) + \sum_{j=1}^{n} p_{ij}(u)\left( \hat{g}(i, u, j) + \hat{J}(j) \right) \right],
\]

for all \(i = 1, \ldots , n\), where

\[
\hat{g}(i, u, j) = \begin{cases} 
  g(i, u, j) + V(j) - V(i) & \text{if } i, j = 1, \ldots , n, \\
  g(i, u, t) - V(i) & \text{if } i = 1, \ldots , n, \ j = t.
\end{cases}
\]

We refer to Eq. (4.9) as the variational form of Bellman’s equation, and to the modified cost per stage \(\hat{g}\) as the temporal difference corresponding to
V. Temporal differences play a significant role in several algorithmic RL contexts; see Section 5.5, and the approximate DP/RL books noted earlier.

Note that Eq. (4.9) is the Bellman equation for a cost-modified problem, where the cost per stage $g$ has been replaced by the temporal difference $\hat{g}$. Thus by applying Prop. 4.1.2 we have that $\hat{J} = J^* - V$ is the unique solution of this equation, so that $J^*$ can be obtained by solving either the original or the cost-modified version of the problem. Moreover, a policy $\mu$, has cost function $\hat{J}_\mu = J_\mu - V$ in the cost-modified problem. It follows that the original and the cost-modified SSP problems are essentially equivalent, and the choice of $V$ does not matter when exact DP methods are used to solve them. However, when approximations are used, different results may be obtained, which can be more favorable with an appropriate choice of $V$.

In particular, we have the option to choose $V$ and an approximation architecture methodology that matches the differential $\hat{J} = J^* - V$ better than it matches $J^*$. For example, we may obtain $V$ with some problem approximation scheme as a rough estimate of $J^*$, and then use a different approximation in value space scheme, based on different principles, for the corresponding cost-modified problem. We refer to this as cost shaping (the name “reward shaping” in used in the RL literature, for problems involving reward maximization). While cost shaping does not change the optimal policies of the original DP problem, it may change significantly the suboptimal policies produced by approximate DP methods, such as the ones we have discussed and will discuss further in the next two chapters; see the material on baselines and advantage updating in Section 3.4, policy gradient methods in Section 5.7.1, and biased aggregation in Section 6.5. A key fact is that with a choice of $V$ close to $J^*$, the range of values of $\hat{J}$ is reduced, and this tends to enhance the ability of approximation methods to capture small scale variations in $J^*$.

Discounted Problems

In this section, we consider the discounted problem, where there is a discount factor $\alpha < 1$. Using our transition probability notation, the Bellman equation and the VI algorithm take the following form.
Sec. 4.1 Stochastic Shortest Path and Discounted Problems

Figure 4.1.3 Transition probabilities for an $\alpha$-discounted problem and its associated SSP problem. In the latter problem, the probability that the state is not $t$ after $k$ stages is $\alpha^k$. The transition costs at the $k$th stage are $g(i, u, j)$ for both problems, but they must be multiplied by $\alpha^k$ because of discounting (in the discounted case) or because they are incurred with probability $\alpha^k$ when termination has not yet been reached (in the SSP case).

Bellman Equation and Value Iteration for Discounted Problems:

For all $i = 1, \ldots, n$, we have

$$J^*(i) = \min_{u \in U(i)} \sum_{j=1}^{n} p_{ij}(u) \left( g(i, u, j) + \alpha J^*(j) \right).$$

For all $i = 1, \ldots, n$, and any initial conditions $J_0(i), \ldots, J_0(n)$, the VI algorithm generates the sequence $\{J_k\}$ according to

$$J_{k+1}(i) = \min_{u \in U(i)} \sum_{j=1}^{n} p_{ij}(u) \left( g(i, u, j) + \alpha J_k(j) \right).$$

We will show that the discounted problem can be converted to an SSP problem for which the analysis of the preceding section applies. To see this, let $i = 1, \ldots, n$ be the states, and consider an associated SSP problem involving the states $1, \ldots, n$ plus an artificial termination state $t$, with state transitions and costs obtained as follows: From a state $i \neq t$, when control $u$ is applied, the next state is $j$ with probability $\alpha p_{ij}(u)$ at a cost $g(i, u, j)$, and $t$ with probability $1 - \alpha$ at zero cost; see Fig. 4.1.3.

Note that Assumption 4.1.1 of the preceding section is satisfied for this SSP problem, since $t$ is reached with probability $1 - \alpha > 0$ from every state in a single step.

Suppose now that we use the same policy in the discounted prob-
lem and in the associated SSP problem. Then, as long as termination has not occurred, the state evolution in the two problems is governed by the same transition probabilities. Furthermore, the expected cost of the $k$th stage of the associated shortest path problem is the expected value of $g(i_k, \mu^k(i_k), i_{k+1})$ multiplied by the probability that state $t$ has not yet been reached, which is $\alpha^k$. This is also the expected cost of the $k$th stage for the discounted problem. Thus starting from a given state, the cost of any given policy (as well as the optimal cost) is the same for the original discounted problem and for the associated SSP problem.

It follows that we can apply Props. 4.1.1-4.1.5 of the preceding section to the associated SSP problem and obtain corresponding results for the discounted problem, which properly incorporate the discount factor in accordance with the discounted-to-SSP equivalence just established.

**Proposition 4.1.6: (Convergence of VI)** The optimal cost $J^*(i)$ is finite for each $i$. Moreover, given any initial conditions $J_0(1), \ldots, J_0(n)$, the sequence $\{J_k(i)\}$ generated by the VI algorithm

$$J_{k+1}(i) = \min_{u \in U(i)} \sum_{j=1}^{n} p_{ij}(u) \left(g(i, u, j) + \alpha J_k(j)\right), \quad i = 1, \ldots, n,$$

converges to $J^*(i)$ for each $i$.

**Proposition 4.1.7: (Bellman’s Equation)** The optimal cost function

$$J^* = (J^*(1), \ldots, J^*(n))$$

satisfies for all $i = 1, \ldots, n$, the equation

$$J^*(i) = \min_{u \in U(i)} \sum_{j=1}^{n} p_{ij}(u) \left(g(i, u, j) + \alpha J^*(j)\right),$$

and is the unique solution of this equation.
Proposition 4.1.8: (VI and Bellman’s Equation for Policies)  
For any stationary policy \( \mu \), the corresponding cost function \( J_\mu = (J_\mu(1), \ldots, J_\mu(n)) \) is the unique solution of the equation
\[
J_\mu(i) = \sum_{j=1}^{n} p_{ij}(\mu(i)) \left( g(i, \mu(i), j) + \alpha J_\mu(j) \right), \quad i = 1, \ldots, n.
\]
Furthermore, given any initial conditions \( J_0(1), \ldots, J_0(n) \), the sequence \( \{J_k(i)\} \) generated by the VI algorithm
\[
J_{k+1}(i) = \sum_{j=1}^{n} p_{ij}(\mu(i)) \left( g(i, \mu(i), j) + \alpha J_k(j) \right), \quad i = 1, \ldots, n,
\]
converges to the cost \( J_\mu(i) \) for each \( i \).

Proposition 4.1.9: (Optimality Condition) A stationary policy \( \mu \) is optimal if and only if for every state \( i \), \( \mu(i) \) attains the minimum in the Bellman equation (4.11).

Bellman’s equation (4.11) has a familiar DP interpretation. At state \( i \), the optimal cost \( J^*(i) \) is the minimum over all controls of the sum of the expected current stage cost and the expected optimal cost of all future stages. The former cost is \( g(i, u, j) \). The latter cost is \( J^*(j) \), but since this cost starts accumulating after one stage, it is discounted by multiplication with \( \alpha \).

Similar to Prop. 4.1.5, there is a contraction mapping result and convergence rate estimate for VI. To this end we introduce the Bellman operators
\[
(TJ)(i) = \min_{u \in U(i)} \sum_{j=1}^{n} p_{ij}(u) \left( g(i, u, j) + \alpha J(j) \right), \quad i = 1, \ldots, n, \quad (4.12)
\]
and
\[
(T\mu J)(i) = \sum_{j=1}^{n} p_{ij}(\mu(i)) \left( g(i, \mu(i), j) + \alpha J(j) \right), \quad i = 1, \ldots, n, \quad (4.13)
\]
in analogy with their SSP counterparts of Eqs. (4.7) and (4.8). Similar to the SSP case, Bellman’s equations can be written in terms of these
operators as the fixed point equations $J' = T^* J_{\mu} = T_{\mu}^* J_{\mu}$. The following
contraction result, shown in the appendix, is useful for the analysis of exact
and approximate methods for discounted problems.

**Proposition 4.1.10: (Contraction Property of the Bellman Operator)** The Bellman operators $T$ and $T_{\mu}$ of Eqs. (4.12) and (4.13)
are contraction mappings of modulus $\alpha$ with respect to the maximum norm
\[ \|J\| = \max_{i=1,\ldots,n} |J(i)|. \]

In particular, for any two $n$-dimensional vectors $J$ and $J'$, we have
\[ \|TJ - TJ'\| \leq \alpha\|J - J'\|, \quad \|T_{\mu}J - T_{\mu}J'\| \leq \alpha\|J - J'\|. \]

As in the SSP case, the results just given have counterparts involving the optimal Q-factors, defined by

$$ Q^*(i, u) = \sum_{j=1}^{n} p_{ij}(u)(g(i, u, j) + \alpha J^*(j)), \quad i = 1, \ldots, n, \quad u \in U(i). $$

They can be obtained from the corresponding SSP results, by viewing the discounted problem as a special case of the SSP problem. Once $Q^*$ or an approximation $\tilde{Q}$ is computed by some method (model-based or model-free), an optimal policy $\mu^*$ or approximately optimal policy $\tilde{\mu}$ can be obtained from the minimization

$$ \mu^*(i) \in \arg \min_{u \in U(i)} Q^*(i, u), \quad i = 1, \ldots, n, $$

or the approximate version

$$ \tilde{\mu}(i) \in \arg \min_{u \in U(i)} \tilde{Q}(i, u), \quad i = 1, \ldots, n. $$

Our basic results relating Bellman’s equation and the VI algorithm
are stated as follows.
Bellman Equation and Value Iteration for Q-Factors and Discounted Problems:

For all $i = 1, \ldots, n$, and $u \in U(i)$ we have

$$Q^*(i, u) = \sum_{j=1}^{n} p_{ij}(u) \left( g(i, u, j) + \alpha \min_{v \in U(j)} Q^*(j, v) \right). \quad (4.14)$$

For all $i = 1, \ldots, n$, and $u \in U(i)$, and for any initial conditions $Q_0(i, u)$, the VI algorithm generates the sequence $\{Q_k\}$ according to

$$Q_{k+1}(i, u) = \sum_{j=1}^{n} p_{ij}(u) \left( g(i, u, j) + \alpha \min_{v \in U(j)} Q_k(j, v) \right). \quad (4.15)$$

Finally, let us mention that the cost shaping idea discussed for SSP problems, extends readily to discounted problems. In particular, the variational form of Bellman’s equation takes the form

$$\hat{J}(i) = \min_{u \in U(i)} \sum_{j=1}^{n} p_{ij}(u) (\hat{g}(i, u, j) + \alpha \hat{J}(j)), \quad i = 1, \ldots, n,$$

for any given vector $V$, where

$$\hat{g}(i, u, j) = g(i, u, j) + \alpha V(j) - V(i), \quad i = 1, \ldots, n,$$

is the temporal difference corresponding to $V$; cf. Eqs. (4.9) and (4.10).

### 4.1.1 Policy Iteration

The major alternative to VI is policy iteration (PI for short). This algorithm starts with a stationary policy $\mu^0$, and generates iteratively a sequence of new policies $\mu^1, \mu^2, \ldots$. The algorithm has solid convergence guarantees when implemented in its exact form, as we will show shortly.

The closest analog to PI that we have encountered so far is the rollout algorithm of Section 2.4. There we have started with some policy and produced an improved policy through a process of cost function evaluation and one-step or multistep lookahead minimization. This idea is extended in the context of PI, which consists of multiple successive policy evaluations and policy improvements, i.e., a form of perpetual rollout, which we discussed in Chapter 3.

Consider first the SSP problem. Here, each PI consists of two phases: policy evaluation and policy improvement; see Fig. 4.1.4. The algorithm takes the following form.
Exact Policy Iteration: SSP Problems

Given the typical policy $\mu^k$:

**Policy evaluation** computes $J_{\mu^k}(i)$, $i = 1, \ldots, n$, as the solution of the (linear) system of Bellman equations

$$J_{\mu^k}(i) = p_{it}(\mu^k(i))g(i, \mu^k(i), t) + \sum_{j=1}^n p_{ij}(\mu^k(i)) \left( g(i, \mu^k(i), j) + J_{\mu^k}(j) \right),$$

(cf. Prop. 4.1.3).

**Policy improvement** then computes a new policy $\mu^{k+1}$ as

$$\mu^{k+1}(i) \in \arg \min_{u \in U(i)} \left[ p_{it}(u)g(i, u, t) + \sum_{j=1}^n p_{ij}(u) \left( g(i, u, j) + J_{\mu^k}(j) \right) \right],$$

for $i = 1, \ldots, n$.

The process is repeated with $\mu^{k+1}$ used in place of $\mu^k$, unless we have

$$J_{\mu^{k+1}}(i) = J_{\mu^k}(i)$$

for all $i$, in which case the algorithm terminates with the policy $\mu^k$.

The counterpart for discounted problems is as follows.
Sec. 4.1 Stochastic Shortest Path and Discounted Problems

193

Exact Policy Iteration: Discounted Problems

Given the typical policy \( \mu^k \):

**Policy evaluation** computes \( J_{\mu^k} (i) , i = 1, \ldots , n \), as the solution of the (linear) system of Bellman equations

\[
J_{\mu^k} (i) = \sum_{j=1}^{n} p_{ij} \left( \mu^k (i) \right) \left( g(i, \mu^k(i), j) + \alpha J_{\mu^k} (j) \right), \quad (4.16)
\]

(cf. Prop. 4.1.8).

**Policy improvement** then computes a new policy \( \mu^{k+1} \) as

\[
\mu^{k+1} (i) \in \arg \min_{u \in U(i)} \sum_{j=1}^{n} p_{ij} (u) \left( g(i, u, j) + \alpha J_{\mu^k} (j) \right), \quad i = 1, \ldots , n,
\]

for \( i = 1, \ldots , n \).

The process is repeated with \( \mu^{k+1} \) used in place of \( \mu^k \), unless we have \( J_{\mu^{k+1}} (i) = J_{\mu^k} (i) \) for all \( i \), in which case the algorithm terminates with the policy \( \mu^k \).

The following proposition, shown in the appendix, establishes the validity of PI, including finite termination with an optimal policy.

**Proposition 4.1.11: (Convergence of Exact PI)** For both the SSP and the discounted problems, the exact PI algorithm generates an improving sequence of policies, i.e.,

\[
J_{\mu^{k+1}} (i) \leq J_{\mu^k} (i), \quad \text{for all } i \text{ and } k,
\]

and terminates with an optimal policy.

The proof of the policy improvement property (4.18) is quite intuitive and is worth summarizing for the discounted problem. Let \( \mu \) be a policy and \( \overline{\mu} \) be the policy obtained from \( \mu \) via a PI. We want to show that \( J_{\overline{\mu}} \leq J_{\mu} \). To this end, let us denote by \( J_N \) the cost function of a policy that applies \( \overline{\mu} \) for the first \( N \) stages and applies \( \mu \) at every subsequent stage. We have the Bellman equation

\[
J_{\mu} (i) = \sum_{j=1}^{n} p_{ij} \left( \mu (i) \right) \left( g(i, \mu(i), j) + \alpha J_{\mu} (j) \right),
\]
which together with the policy improvement equation (4.17) implies that

\[ J_1(i) = \sum_{j=1}^{n} p_{ij}(\pi(i))\left(g(i, \pi(i), j) + \alpha J_\mu(j)\right) \leq J_\mu(i). \]  

(4.19)

From the definition of \( J_2 \) and \( J_1 \) we have

\[ J_2(i) = \sum_{j=1}^{n} p_{ij}(\pi(i))\left(g(i, \pi(i), j) + \alpha J_1(j)\right), \]  

(4.20)

so the preceding two relations imply that

\[ J_2(i) \leq J_1(i) \leq J_\mu(i), \quad \text{for all } i. \]  

(4.21)

Continuing similarly, we obtain

\[ J_{N+1}(i) \leq J_N(i) \leq J_\mu(i), \quad \text{for all } i \text{ and } N. \]  

(4.22)

Since \( J_N \to J_\pi \) (cf. Prop. 4.1.7), it follows that \( J_\pi \leq J_\mu \).

In practice, a lot of cost improvement is often obtained with the first few policies that are generated by the PI algorithm. This may happen even if the number of iterations needed for termination is quite large.

### 4.1.2 Optimistic and Multistep Policy Iteration

The PI algorithm that we have discussed so far uses exact policy evaluation of the current policy \( \mu^k \) and one-step lookahead policy improvement, i.e., it computes exactly \( J_{\mu^k} \), and it obtains the next policy \( \mu^{k+1} \) by a one-step lookahead minimization using \( J_{\mu^k} \) as an approximation to \( J^* \). We will now describe a more flexible algorithm, called optimistic PI, whereby \( J_{\mu^k} \) is approximated by any number of VIs corresponding to \( \mu^k \) (cf. Prop. 4.1.8) and the policy improvement is done using multistep lookahead (alternative names used in the literature are “modified PI” and “generalized PI”).

Optimistic PI aspires (somewhat optimistically) to obtain a good successor policy using an inaccurate evaluation of the current policy. It starts with a function \( J_0 \), an initial guess of \( J^* \). It generates a sequence \( \{J_k\} \) and an associated sequence of policies \( \{\mu^k\} \). The \( k \)th iteration starts with a function \( J_k \) and first generates \( \mu^k \). It then generates \( J_{k+1} \) using \( m_k \) iterations of the VI algorithm that corresponds to \( \mu^k \), starting with \( J_k \). We will prove that \( \{J_k\} \) and \( \{\mu^k\} \) asymptotically converge to \( J^* \) and an optimal policy, respectively.
Optimistic Policy Iteration: Discounted Problems

Given the typical function $J_k$:

**Policy improvement** computes a policy $\mu^k$ such that

$$\mu^k(i) \in \arg \min_{u \in U(i)} \sum_{j=1}^{n} p_{ij}(u) \left( g(i, u, j) + \alpha J_k(j) \right), \quad i = 1, \ldots, n. \quad (4.23)$$

**Optimistic policy evaluation** starts with $\hat{J}_{k,0} = J_k$, and uses $m_k$ VI iterations for policy $\mu^k$ to compute $\hat{J}_{k,1}, \ldots, \hat{J}_{k,m_k}$ according to

$$\hat{J}_{k,m+1}(i) = \sum_{j=1}^{n} p_{ij}(\mu^k(i)) \left( g(i, \mu^k(i), j) + \alpha \hat{J}_{k,m}(j) \right), \quad (4.24)$$

for all $i = 1, \ldots, n$, $m = 0, \ldots, m_k - 1$, and finally sets $J_{k+1} = \hat{J}_{k,m_k}$.

From Eq. (4.24), it can be seen that one way to interpret optimistic PI is that we approximate $J_{\mu^k}$ by using $\mu^k$ for $m_k$ stages, and adding a terminal cost function equal to the current cost estimate $J_k$ instead of using $\mu^k$ for an additional infinite number of stages. Accordingly, simulation-based approximations of optimistic PI, evaluate the cost function $J_{\mu^k}$ by using $m_k$-stage trajectories, with the cost of future stages accounted using $J_k$, which plays the role of a terminal cost function approximation.

The convergence properties of optimistic PI are solid, although it may require an infinite number of iterations to converge to $J^*$. To see why this is so, suppose that we evaluate each policy with a single VI (i.e., $m_k = 1$). Then Eqs. (4.23) and (4.24) yield $J_{k+1} = TJ_k$, and the method is essentially identical to the VI method, which requires an infinite number of iterations to converge.

The following proposition, shown in the appendix, establishes the validity of optimistic PI. There is a corresponding convergence property for SSP problems, but its currently available proof is fairly complicated. It is given in Section 3.5.1 of the book [Ber12]. Asynchronous versions of optimistic PI also involve theoretical convergence difficulties, which are discussed in Section 2.6.2 of [Ber12] and Section 2.6.3 of [Ber18a].

**Proposition 4.1.12: (Convergence of Optimistic PI)** For the discounted problem, the sequence $\{J_k\}$ generated by the optimistic PI algorithm satisfies $J_k \to J^*$, while the generated policies $\mu^k$ are optimal for all $k$ sufficiently large.

The proof of the proposition is based on the policy improvement line of proof we gave earlier, following Prop. 4.1.11. In particular, if $J_0$ satisfies
Approximate Policy Iteration for Infinite Horizon Problems  Chap. 4

\( T_{\mu_0} J_0 \leq J_0 \), the argument of Eqs. (4.19)-(4.22) can be used to show that \( J^* \leq J_{k+1} \leq J_k \) for all \( k \). Moreover, the proof of the appendix argues that we may assume that \( T_{\mu_0} J_0 \leq J_0 \) holds without loss of generality, since we may add a constant to \( J_0 \) without affecting the sequence \( \{\mu^k\} \) generated by the algorithm. The proof of the appendix also shows that the generated policies \( \mu^k \) are optimal after some \( k \), but this fact cannot ordinarily be exploited in practice because the verification that \( \mu^k \) is optimal requires additional computations that complicate the method.

**Multistep Policy Improvement**

The motivation for multistep policy improvement is that it may yield a better policy \( \mu^{k+1} \) than the one obtained with one-step policy improvement. In fact this makes even more sense when the evaluation of \( \mu^k \) is approximate, since then the longer lookahead may compensate for errors in the policy evaluation. We give below the method in its exact nonoptimistic form (in a different version it may be combined with optimistic PI, i.e., with policy evaluation performed approximately with a finite number of value iterations starting from some cost function approximation).

<table>
<thead>
<tr>
<th>Multistep Lookahead Exact Policy Iteration: Discounted Problems</th>
</tr>
</thead>
</table>

Given the typical policy \( \mu^k \):

**Policy evaluation** computes \( J_{\mu^k}(i) \), \( i = 1, \ldots, n \), as the solution of the (linear) system of Bellman equations

\[
J_{\mu^k}(i) = \sum_{j=1}^{n} p_{ij}(\mu^k(i)) \left( g(i, \mu^k(i), j) + \alpha J_{\mu^k}(j) \right), \quad i = 1, \ldots, n,
\]

(cf. Prop. 4.1.3).

**Policy improvement with \( \ell \)-step lookahead** then solves the \( \ell \)-stage problem with terminal cost function \( J_{\mu^k} \). If \( \{\hat{\mu}_0, \ldots, \hat{\mu}_{\ell-1}\} \) is the optimal policy of this problem, then the new policy \( \mu^{k+1} \) is \( \hat{\mu}_0 \).

The process is repeated with \( \mu^{k+1} \) used in place of \( \mu^k \), unless we have \( J_{\mu^{k+1}}(i) = J_{\mu^k}(i) \) for all \( i \), in which case the algorithm terminates with the policy \( \mu^k \).

Exact PI with multistep lookahead has the same solid convergence properties as its one-step lookahead counterpart: it terminates with an optimal policy, and the generated sequence of policies is monotonically improving. The proof is similar to the ones for other forms of PI and will not be given.
### 4.1.3 Policy Iteration for Q-factors

Similar to VI, we may also equivalently implement PI through the use of Q-factors. To see this, note that the policy improvement step may be implemented by minimizing over $u \in U(i)$ the expression

$$Q_\mu(i, u) = \sum_{j=1}^{n} p_{ij}(u)(g(i, u, j) + \alpha J_\mu(j)), \quad i = 1, \ldots, n, \ u \in U(i),$$

which we view as the Q-factor of the pair $(i, u)$ corresponding to $\mu$. Moreover we have

$$J_\mu(j) = Q_\mu(j, \mu(j)), \quad \text{(cf. Prop. 4.1.8)}.$$  

The following algorithm is thus obtained; see Fig. 4.1.5.

**Exact Policy Iteration for Q-Factors: Discounted Problems**

Given the typical policy $\mu^k$:

**Policy evaluation** computes $Q_{\mu^k}(i, u)$, for all $i = 1, \ldots, n$, and $u \in U(i)$, as the solution of the (linear) system of equations

$$Q_{\mu^k}(i, u) = \sum_{j=1}^{n} p_{ij}(u)\left(g(i, u, j) + \alpha Q_{\mu^k}(j, \mu^k(j))\right). \quad \text{(4.25)}$$
Policy improvement then computes a new policy $\mu^{k+1}$ as

$$\mu^{k+1}(i) \in \arg\min_{u \in U(i)} Q_{\mu^k}(i, u), \quad i = 1, \ldots, n. \tag{4.26}$$

The process is repeated with $\mu^{k+1}$ used in place of $\mu^k$, unless we have $Q_{\mu^k}(i, \mu^{k+1}(i)) = Q_{\mu^k}(i, \mu^k(i))$ for all $i$, in which case the algorithm terminates with the policy $\mu^k$.

Note that the system (4.25) has a unique solution, since from the uniqueness of solution of Bellman’s equation, any solution must satisfy

$$Q_{\mu^k}(j, \mu^k(j)) = J_{\mu^k}(j).$$

Hence the Q-factors $Q_{\mu^k}(j, \mu^k(j))$ are uniquely determined, and then the remaining Q-factors $Q_{\mu^k}(i, u)$ are also uniquely determined from Eq. (4.25).

The PI algorithm for Q-factors is mathematically equivalent to PI for costs, as given in the preceding subsection. The only difference is that we calculate all the Q-factors $Q_{\mu^k}(i, u)$, rather than just the costs $J_{\mu^k}(j) = Q_{\mu^k}(j, \mu^k(j))$, i.e., just the Q-factors corresponding to the controls chosen by the current policy. However, the remaining Q-factors $Q_{\mu^k}(i, u)$ are needed for the policy improvement step (4.26), so no extra computation is required. It can be verified also that the PI algorithm (4.25)-(4.26) can be viewed as the PI algorithm for the discounted version of the modified problem of Fig. 4.1.2.

### 4.2 APPROXIMATION IN VALUE SPACE - PERFORMANCE BOUNDS

In this section we will discuss the general framework for approximation in value space for infinite horizon DP, beginning with discounted problems. We will also provide error bound that help guide the approximation process.

Consistently with the corresponding finite horizon schemes of Chapter 2, the general idea is to compute some approximation $\hat{J}$ of the optimal cost function $J^*$, and then use one-step or multistep lookahead to implement a suboptimal policy $\hat{\mu}$. Thus, a one-step lookahead policy applies at state $i$ the control $\hat{\mu}(i)$ that attains the minimum in the expression

$$\min_{u \in U(i)} \sum_{j=1}^{n} p_{ij}(u)(g(i, u, j) + \alpha \hat{J}(j)), \tag{4.27}$$

see Fig. 4.2.1.
Approximation in Value Space - Performance Bounds

Approximate minimization

\[
\min_{u \in U(i)} \sum_{j=1}^{n} p_{ij}(u) \left( g(i, u, j) + \alpha \hat{J}(j) \right)
\]

First Step “Future”

Approximations:
- Replace \(E\{\cdot\}\) with nominal values (certainty equivalence)
- Adaptive simulation
- Monte Carlo tree search

Computation of \(\tilde{J}\):
- Problem approximation
  - Rollout
  - Approximate PI
  - Parametric approximation
  - Aggregation

**Figure 4.2.1** Schematic illustration of various options for approximation in value space with one-step lookahead in infinite horizon problems. The lookahead function values \(\tilde{J}(j)\) approximate the optimal cost-to-go values \(J^*(j)\), and can be computed by a variety of methods. There may be additional approximations in the minimization over \(u_k\) and the computation of the expected value.

Similarly, at state \(i\), a two-step lookahead policy applies the control \(\tilde{\mu}(i)\) attaining the minimum in the preceding equation, where now \(\tilde{J}\) is obtained itself on the basis of a one-step lookahead approximation. In other words, for all states \(j\) that can be reached from \(i\), we have

\[
\tilde{J}(j) = \min_{u \in U(j)} \sum_{m=1}^{n} p_{jm}(u) \left( g(j, u, m) + \alpha \hat{J}(m) \right),
\]

where \(\tilde{J}\) is some approximation of \(J^*\). Thus \(\tilde{J}\) is the result of a single value iteration starting from \(\tilde{J}\). Policies with lookahead of more than two stages are similarly defined. The “effective one-step” cost approximation \(\tilde{J}\) in \(\ell\)-step lookahead is the result of \(\ell - 1\) successive value iterations starting from some initial approximation \(\hat{J}\). Otherwise expressed, \(\ell\)-step lookahead with \(\tilde{J}\) at the end is the same as one-step lookahead with \(T^{\ell-1}\tilde{J}\) at the end, where \(T\) is the Bellman operator [cf. Eq. (4.12)].

**Types of Approximation in Value Space Schemes**

In Chapter 2 we described several types of limited lookahead schemes, where \(\tilde{J}\) is obtained in different ways, such as problem approximation, rollout, and others. Some of these schemes can be fruitfully adapted to infinite horizon problems; see Fig. 4.2.1. For example, the problem approximation approaches of Section 2.3 admit straightforward extensions to infinite horizon settings, whereby the function \(\tilde{J}(j)\) in Eq. (4.27) is obtained by solving
exactly an infinite horizon (or even finite horizon) problem that is related to the original in some way. Aggregation is another possible approximation approach, which will be discussed in Chapter 6.

Imperfect state observation problems pose special challenges in the case of an infinite horizon. Such problems can be reformulated to ones involving perfect observation of the belief state, which, however, is infinite dimensional (cf. Section 1.3.6).† Problem approximation based on forms of certainty equivalence is particularly interesting within this context, because a natural approximating problem is often evident. For example, the lookahead function \( \tilde{J} \) may be derived by solving a perfect state information variant of the original problem, where an estimate of the system state is used as if it were exact. The perfect state information variant may be tractable, because it may be deterministic or it may involve a modest number of states.

In this chapter, the approximation in value space schemes that we will focus on are primarily based on approximate (optimistic) PI and operate as follows, starting with an initial policy \( \mu^0 \):

(a) We generate (usually off-line) several policies \( \mu^0, \mu^1, \ldots, \mu^m \).

(b) We evaluate each policy \( \mu^k \) approximately, with a cost function \( \tilde{J}_{\mu^k} \), often by using a parametric approximation/neural network approach, and possibly including the use of truncated rollout.

(c) We generate the next policy \( \mu^{k+1} \) using one-step or multistep policy improvement based on \( \tilde{J}_{\mu^k} \).

(d) We use (on-line) the approximate evaluation \( \tilde{J}_{\mu^m} \) of the last policy in the sequence as the lookahead approximation \( \tilde{J} \) in the one-step lookahead minimization \( 4.27 \), or its multistep counterpart.

We will view rollout as a simple form of approximate PI, which involves a single policy iteration, executed with the aid of simulation. The rollout may be truncated and supplemented with a (potentially sophisticated) terminal cost function approximation.

In what follows in this section, we will give some performance bounds for limited lookahead schemes, for rollout, and for approximate PI. For purposes of easy reference, we provide here that Bellman operator formulas that we have introduced in the preceding section.

For SSP problems:

\[
(TJ)(i) = \min_{u \in U(i)} \left[ p_{it}(u)g(i, u, t) + \sum_{j=1}^{n} p_{ij}(u)(g(i, u, j) + J(j)) \right], \tag{4.28}
\]

† Because of the infinite dimensionality of the belief space, the theory of Chapter 4 has to be extended before it can be applied, since it was developed for finite state space. Usually this is fairly straightforward for discounted problems, but less so for SSP problems. We will not provide further discussion in this book.
for all $i = 1, \ldots, n$, and

$$
(T_{\mu} J)(i) = p_u(\mu(i))g(i, \mu(i), t) + \sum_{j=1}^{n} p_{ij}(\mu(i)) \left( g(i, \mu(i), j) + \alpha J(j) \right), \quad (4.29)
$$

for all policies $\mu$ and states $i = 1, \ldots, n$.

For discounted problems:

$$
(T J)(i) = \min_{u \in U(i)} \sum_{j=1}^{n} p_{ij}(u) \left( g(i, u, j) + \alpha J(j) \right), \quad (4.30)
$$

and

$$
(T_{\mu} J)(i) = \sum_{j=1}^{n} p_{ij}(\mu(i)) \left( g(i, \mu(i), j) + \alpha J(j) \right), \quad (4.31)
$$

for all policies $\mu$ and states $i = 1, \ldots, n$.

4.2.1 Limited Lookahead

We will first consider performance bounds for $\ell$-step lookahead. In particular, for a given state $i_0$, let $\hat{\mu}_0, \ldots, \hat{\mu}_{\ell-1}$ attain the minimum in the $\ell$-step lookahead minimization

$$
\min_{\mu_0, \ldots, \mu_{\ell-1}} E \left\{ \sum_{k=0}^{\ell-1} \alpha^k g(i_k, \mu_k(i_k), j_k) + \alpha^\ell \tilde{J}(i_\ell) \right\}.
$$

We focus on the suboptimal policy that applies control $\hat{\mu}(i_0) = \hat{\mu}_0(i_0)$, and we refer to $\hat{\mu}$ as the $\ell$-step lookahead policy corresponding to $\tilde{J}$. Equivalently, in the shorthand notation of the Bellman operators $T$ and $T_{\hat{\mu}}$ of Eqs. (4.30) and (4.31), the $\ell$-step lookahead policy $\hat{\mu}$ is defined by

$$
T_{\hat{\mu}}(T_{\ell-1} \tilde{J}) = T^\ell \tilde{J}.
$$

We will derive a bound for the performance of $\hat{\mu}$ in part (a) of the following proposition, proved in the appendix.

We will also derive a bound for the case of a useful generalized one-step lookahead scheme [part (b) of the following proposition]. This scheme aims to reduce the computation to obtain $\hat{\mu}(i)$, by performing the lookahead minimization over a subset $U(i) \subset U(i)$. Thus, the control $\hat{\mu}(i)$ used in this scheme is one that attains the minimum in the expression

$$
\min_{u \in U(i)} \sum_{j=1}^{n} p_{ij}(u) \left( g(i, u, j) + \alpha \tilde{J}(j) \right).
$$
We identify a subset $\mathcal{U}(i)$ of promising controls, and to save computation, we restrict attention to this subset in the one-step lookahead minimization.

**Proposition 4.2.1: (Limited Lookahead Performance Bounds)**

(a) Let $\tilde{\mu}$ be the $\ell$-step lookahead policy corresponding to $\tilde{J}$. Then

$$
\|J_{\tilde{\mu}} - J^*\| \leq \frac{2\alpha\ell}{1 - \alpha}\|\tilde{J} - J^*\|, 
$$

(4.32)

where $\|\cdot\|$ denotes the maximum norm $\|J\| = \max_{i=1,...,n} |J(i)|$.

(b) Define

$$
\hat{J}(i) = \min_{u \in \mathcal{U}(i)} \sum_{j=1}^{n} p_{ij}(u)(g(i, u, j) + \alpha \hat{J}(j)), \quad i = 1, \ldots, n, 
$$

(4.33)

where $\mathcal{U}(i) \subset \mathcal{U}(i)$ for all $i = 1, \ldots, n$, and let $\hat{\mu}$ be the one-step lookahead policy obtained by minimization in the right side of this equation. Then we have

$$
J_{\hat{\mu}}(i) \leq \hat{J}(i) + c \frac{1}{1 - \alpha}, \quad i = 1, \ldots, n, 
$$

(4.34)

where

$$
c = \max_{i=1,...,n} (\hat{J}(i) - \tilde{J}(i)).
$$

An important point regarding the bound (4.32) is that $\tilde{\mu}$ is unaffected by a constant shift in $\tilde{J}$ [an addition of a constant $\beta$ to all values $\tilde{J}(i)$]. Thus $\|J - J^*\|$ in Eq. (4.32) can be replaced by the potentially smaller number

$$
\min_{\beta \in \mathbb{R}} \max_{i=1,...,n} |\tilde{J}(i) + \beta - J^*(i)|. 
$$

(4.35)

Another interesting point is that it is sufficient to take maximum in the preceding expression over just the states $i$ that can occur following the $\ell$ steps of lookahead, thus potentially improving the bound of Eq. (4.32) even further. This can be used to obtain sharper versions of Prop. 4.2.1(a), as well as the related subsequent Prop. 5.1.3(a), although we will not pursue this analysis further.

The bound (4.32) suggests that performance is improved when the length $\ell$ of the lookahead is increased, and also when the lookahead cost approximation $\tilde{J}$ is closer to the optimal cost $J^*$ (when modified with an optimal constant shift $\beta$). Both of these conclusions are intuitive and also
Sec. 4.2  Approximation in Value Space - Performance  Bounds  203

Figure 4.2.2 A two-state problem for proving the tightness of the performance bound of Prop. 4.2.1(b) (cf. Example 4.2.1). All transitions are deterministic as shown. At state 1 there are two possible decisions: move to state 2 at cost 0 (policy $\mu^*$) or stay at state 1 at cost $2\alpha\epsilon$ (policy $\mu$).

consistent with practical experience. Note that we are not guaranteeing that multistep lookahead will lead to better performance than one-step lookahead; we know that this is not necessarily true (cf. Example 2.2.1). It is the performance bound that is improved with multistep lookahead.

Regarding the bound (4.34), we note that it guarantees that when $c \leq 0$, the cost $J_{\tilde{\mu}}$ of the one-step lookahead policy is no larger than $\tilde{J}$. The case $c = 0$ is equivalent to $\tilde{J} \leq \tilde{J}$, which resembles the consistent improvement condition for deterministic rollout methods (cf. Section 2.4.1). If $\tilde{J} = J_\mu$ for some policy $\mu$ with $\mu(i) \in \mathbb{U}(i)$ for all $i$ (as in the case of the pure form of rollout to be discussed in Section 5.1.2), then $c = 0$, and from Eq. (4.34) it follows that we have cost improvement, i.e., $J_{\tilde{\mu}} \leq J_\mu$.

Unfortunately, the bound (4.32) is not very reassuring when $\alpha$ is close to 1. Nonetheless, the following example shows that the bound can be tight even in very simple problems with just two states. What is happening here is that an $O(\epsilon)$ difference in single stage cost between two controls can generate an $O(\epsilon/(1 - \alpha))$ difference in policy costs (through discounted accumulation over an infinite number of steps), yet it can be “nullified” in Bellman’s equation by an $O(\epsilon)$ difference between $J^*$ and $\tilde{J}$.

**Example 4.2.1**

Consider the two-state discounted problem shown in Fig. 4.2.2, where $\epsilon$ is a positive scalar and $\alpha \in [0, 1)$ is the discount factor. Here there are two control choices at state 1: move to state 2 at cost 0 (policy $\mu^*$) or stay at state 1 at cost $2\alpha\epsilon$ (policy $\mu$). The optimal policy is $\mu^*$, and the optimal cost-to-go function is $J^*(1) = J^*(2) = 0$. Consider the cost function approximation $\tilde{J}$

$$\tilde{J}(1) = -\epsilon, \quad \tilde{J}(2) = \epsilon,$$

so that

$$\|\tilde{J} - J^*\| = \epsilon.$$
The policy \( \mu \) that decides to stay at state 1 is a one-step lookahead policy based on \( \bar{J} \), because

\[
2\alpha \epsilon + \alpha \bar{J}(1) = \alpha \epsilon = 0 + \alpha \bar{J}(2).
\]

Moreover, we have

\[
J_\mu(1) = \frac{2\alpha \epsilon}{1 - \alpha} = \frac{2\alpha}{1 - \alpha} \|\bar{J} - J^*\|,
\]

so the bound of Eq. (4.32) holds with equality when \( \ell = 1 \).

### 4.2.2 Rollout

Let us first consider rollout in its pure form, where \( \bar{J} \) in Eq. (4.27) is the cost function of some stationary policy \( \mu \) (also called the base policy or base heuristic), i.e., \( \bar{J} = J_\mu \). Then, the rollout policy is the result of a single policy iteration starting from \( \mu \). The policy evaluation that yields the costs \( J_\mu(j) \) needed for policy improvement may be done in any suitable way. Monte-Carlo simulation (averaging the costs of many trajectories starting from \( j \)) is one major possibility. Of course if the problem is deterministic, a single simulation trajectory starting from \( j \) is sufficient, in which case the rollout policy is much less computationally demanding. Note also that in discounted problems the simulated trajectories can be truncated after a number of transitions, which is sufficiently large to make the cost of the remaining transitions insignificant in view of the discount factor.

An important fact is that in the pure form of rollout, the rollout policy improves over the base policy, consistent with the finite horizon case; cf. Section 2.4. This is shown by the following proposition [a special case of Prop. 4.2.1(b) as noted earlier], and it is to be expected since rollout is one-step PI, so the general policy improvement property of PI applies. A related result is given as Lemma 5.9.1 in the appendix (Section 5.9.3).

**Proposition 4.2.2: (Cost Improvement by Rollout)** Let \( \tilde{\mu} \) be the rollout policy obtained by the one-step lookahead minimization

\[
\min_{u \in \mathcal{U}(i)} \sum_{j=1}^{n} p_{ij}(u) \left( g(i, u, j) + \alpha J_\mu(j) \right),
\]

where \( \mu \) is a base policy [cf. Eq. (4.33) with \( \bar{J} = J_\mu \)] and we assume that \( \mu(i) \in \mathcal{U}(i) \subset U(i) \) for all \( i = 1, \ldots, n \). Then \( J_{\tilde{\mu}} \leq J_{\mu} \).

Let us also mention the variation of rollout that uses multiple base heuristics, and simultaneously improves on all of them. This variant, also
called parallel rollout because of its evident parallelization potential, is similar to its finite horizon counterpart; cf. Section 2.4.1.

**Example 4.2.2 (Rollout with Multiple Heuristics)**

Let \( \mu_1, \ldots, \mu_M \) be stationary policies, let 
\[
\tilde{J}(i) = \min \{ J_{\mu_1}(i), \ldots, J_{\mu_M}(i) \}, \quad i = 1, \ldots, n,
\]
let \( \overline{U}(i) \subset U(i) \), and assume that 
\[
\mu_1(i), \ldots, \mu_M(i) \in \overline{U}(i), \quad i = 1, \ldots, n.
\]
Then, for all \( i \) and \( m = 1, \ldots, M \), we have
\[
\hat{J}(i) = \min_{u \in \overline{U}(i)} \sum_{j=1}^{n} p_{ij}(u) \left( g(i, u, j) + \alpha \tilde{J}(j) \right) \leq \min_{u \in \overline{U}(i)} \sum_{j=1}^{n} p_{ij}(u) \left( g(i, u, j) + \alpha J_{\mu_m}(j) \right) \leq \sum_{j=1}^{n} p_{ij}(\mu_m(i)) \left( g(i, \mu_m(i), j) + \alpha J_{\mu_m}(j) \right) = J_{\mu_m}(i),
\]
which, by taking minimum of the right-hand side over \( m \), yields
\[
\hat{J}(i) \leq \tilde{J}(i), \quad i = 1, \ldots, n.
\]
Using Prop. 4.2.1(b), we see that the rollout policy \( \tilde{\mu} \), obtained by using \( \tilde{J} \) as one-step lookahead approximation satisfies
\[
J_{\tilde{\mu}}(i) \leq \tilde{J}(i) = \min \{ J_{\mu_1}(i), \ldots, J_{\mu_M}(i) \}, \quad i = 1, \ldots, n,
\]
i.e., it improves over each of the policies \( \mu_1, \ldots, \mu_M \).

**Truncated Rollout with Multistep Lookahead and Terminal Cost Function Approximation**

Let us next discuss a truncated variant of the rollout approach, whereby we use \( \ell \)-step lookahead, we then apply rollout with policy \( \mu \) for a limited number of steps, and finally we approximate the cost of the remaining steps using some terminal cost approximation \( \tilde{J} \); see Fig. 4.2.3 for a case where \( \ell = 2 \). We can view this form of rollout as a single optimistic policy iteration combined with multistep lookahead (the iteration is optimistic because it evaluates \( \mu \) with \( m \) value iterations starting from \( \tilde{J} \)). This type of algorithm
Approximate Policy Iteration for Infinite Horizon Problems  Chap. 4

Selective Depth Lookahead Tree

Base Heuristic

States $i_k$

States $i_{k+1}$

States $i_{k+2}$

Variable Length Rollout Selective Depth Rollout Adaptive Simulation Policy

Variable Length Rollout Selective Depth Rollout Policy

Adaptive Simulation

Terminal Cost Function

Figure 4.2.3 Illustration of two-step lookahead, rollout with a policy $\mu$ for a limited and state-dependent number of steps, and a terminal cost function approximation $\tilde{J}$. A Monte Carlo tree search scheme may also be used for multistep lookahead; cf. Section 2.4.2.

was used in Tesauro’s rollout-based backgammon player [TeG96] (it was also used in AlphaGo in a modified form, with Monte Carlo tree search in place of ordinary limited lookahead). We will give more details later.

Note that the three components of this rollout scheme (multistep lookahead, rollout with $\mu$, and cost approximation $\tilde{J}$) can be designed independently of each other. Moreover, while the multistep lookahead is implemented on-line, $\mu$ and $\tilde{J}$ should ordinarily be available from an earlier off-line computation.

The following proposition generalizes the performance bounds given for limited lookahead (cf. Props. 4.2.1). In particular, part (a) of the proposition follows by applying Prop. 4.2.1(a), since the truncated rollout scheme of this section can be viewed as $\ell$-step approximation in value space with terminal cost function $T_\mu^n \tilde{J}$ at the end of the lookahead, where $T_\mu$ is the Bellman operator corresponding to $\mu$. 

Proposition 4.2.3: (Performance Bounds for Truncated Rollout with Terminal Cost Function Approximation) Let \( \ell \) and \( m \) be positive integers, let \( \mu \) be a policy, and let \( \tilde{J} \) be a function of the state. Consider a truncated rollout scheme consisting of \( \ell \)-step lookahead, followed by rollout with a policy \( \mu \) for \( m \) steps, and a terminal cost function approximation \( \tilde{J} \) at the end of the \( m \) steps. Let \( \tilde{\mu} \) be the policy generated by this scheme.

(a) We have
\[
\| J_{\tilde{\mu}} - J^* \| \leq \frac{2\alpha \ell}{1 - \alpha} \| T^m_{\mu} \tilde{J} - J^* \|, \tag{4.36}
\]
where \( T_{\mu} \) is the Bellman operator of Eq. (4.31), and \( \| \cdot \| \) denotes the maximum norm \( \| J \| = \max_{i=1, \ldots, n} | J(i) | \).

(b) We have
\[
J_{\tilde{\mu}}(i) \leq \tilde{J}(i) + \frac{c}{1 - \alpha}, \quad i = 1, \ldots, n,
\]
where
\[
c = \max_{i=1, \ldots, n} ((T_{\mu} \tilde{J})(i) - \tilde{J}(i)).
\]

(c) We have
\[
J_{\tilde{\mu}}(i) \leq J_{\mu}(i) + \frac{2}{1 - \alpha} \| \tilde{J} - J_{\mu} \|, \quad i = 1, \ldots, n.
\]

Some of the insights provided by the preceding proposition are:

(a) Part (a) of the proposition implies that as the size of lookahead \( \ell \) increases, the bound on the performance of the rollout policy also improves. Moreover, assuming that \( \mu \) is close to optimal (so that \( T^m_{\mu} \tilde{J} \) is close to \( J^* \) as \( m \to \infty \)), the bound on the performance of the rollout policy \( \tilde{\mu} \) improves as \( m \) increases (the improvement is enhanced if \( \tilde{J} \) is also close to \( J_{\mu} \)).

(b) Part (c) suggests that if \( \tilde{J} \) is close to \( J_{\mu} \), the performance of the rollout policy \( \tilde{\mu} \) is nearly improved relative to the performance of the base policy \( \mu \). This is consistent with the cost improvement property of rollout, cf. Prop. 4.2.2. Part (b) admits a similar interpretation.

In summary, the guidelines for truncated rollout that these results

\[\uparrow\] Note that \( \tilde{\mu} \) is unaffected by a constant shift in \( \tilde{J} \), so that an optimized constant \( \beta \) may be added to \( \tilde{J} \) in the bound (4.36) [cf. Eq. (4.35)].
suggest are to choose as large lookahead as practical, and choose $\tilde{J}$ as close to $J_\mu$ or $J^*$ as possible. It is not clear to what extent increasing $m$, the length of rollout of $\mu$, improves performance, and some examples suggest that $m$ should not be chosen to be very large. A small value of $m$ is also beneficial in another way: it limits the amount of computation needed for rollout, and reduces the variance of the cost estimates. In practice, for an infinite horizon problem, $m$ is ordinarily chosen in some empirical fashion.

Regarding the terminal cost approximation $\tilde{J}$ in truncated rollout schemes, it may be heuristic, based on problem approximation, or based on a more systematic simulation methodology. For example, the values $J_\mu(i)$ may be computed by simulation for all $i$ in a subset of representative states, and $\tilde{J}$ may be selected from a parametric class of vectors by a least squares regression of the computed values. This approximation may be performed off-line, outside the time-sensitive restrictions of a real-time implementation, and the result $\tilde{J}$ may be used on-line in place of $J_\mu$ as a terminal cost function approximation. Note that a good choice of terminal cost approximation is crucial for some types of SSP problems where most or all of the cost is incurred upon reaching the termination state (for example winning or losing a game). Note also that once cost function approximation is introduced at the end of the rollout, the cost improvement property of the rollout policy over the base policy may be lost [cf. Prop. 4.2.3(c)].

The truncated rollout scheme of Fig. 4.2.3 has been adopted in the rollout backgammon algorithm of Tesauro and Galperin [TeG96]. The policy $\mu$ and the terminal cost function approximation $\tilde{J}$ were provided by the TD-Gammon algorithm of Tesauro [Tes94], which was based on a neural network, trained using a form of optimistic policy iteration and TD($\lambda$). A similar algorithm (with Monte Carlo tree search instead of $\ell$-step lookahead) was used in the AlphaGo program (Silver et al. [SHM16]), with $\mu$ and $\tilde{J}$ obtained using an approximate PI scheme and a deep neural network.

### 4.2.3 Approximate Policy Iteration

When the number of states is very large, the policy evaluation step and/or the policy improvement step of the PI method may be implementable only through approximations. In an approximate PI scheme, each policy $\mu^k$ is evaluated approximately, with a cost function $\tilde{J}_{\mu^k}$, often with the use of a feature-based architecture or a neural network, and the next policy $\mu^{k+1}$ is generated by (perhaps approximate) policy improvement based on $\tilde{J}_{\mu^k}$.

To formalize this type of procedure, we assume a policy evaluation error satisfying

$$\max_{i=1,\ldots,n} |\tilde{J}_{\mu^k}(i) - J_{\mu^k}(i)| \leq \delta,$$

(4.37)
and a policy improvement error satisfying

\[
\max_{i=1,\ldots,n} \left| \sum_{j=1}^{n} p_{ij} (\mu_{k+1}^{j}(i)) \left( g(i, \mu_{k+1}^{j}(i), j) + \alpha \bar{J}_{\mu_k}(j) \right) \right. \\
- \min_{u \in \mathcal{U}(i)} \left. \sum_{j=1}^{n} p_{ij}(u) \left( g(i, u, j) + \alpha \bar{J}_{\mu_k}(j) \right) \right| \leq \epsilon,
\]

(4.38)

where \( \delta \) and \( \epsilon \) are some nonnegative scalars. Here \( \epsilon \) includes the simulation errors, plus any additional errors due to function approximation. Also \( \delta \) is a measure of the accuracy of the lookahead minimization in the policy improvement operation (in many cases \( \delta = 0 \)).

The following proposition, proved in the appendix (and also in the original source \cite{BeT96}, Section 6.2.2), provides a performance bound for discounted problems (a similar result is available for SSP problems; see \cite{BeT96}, Section 6.2.2).

**Proposition 4.2.4: (Performance Bound for Approximate PI)**

Consider the discounted problem, and let \( \{\mu_k\} \) be the sequence generated by the approximate PI algorithm defined by the approximate policy evaluation (4.37) and the approximate policy improvement (4.38). Then the policy error

\[
\max_{i=1,\ldots,n} \left| J_{\mu_k}^i(i) - J^*(i) \right|
\]

becomes less or equal to

\[
\epsilon + \frac{2\alpha \delta}{(1 - \alpha)^2} \]  

asymptotically as \( k \to \infty \).

The preceding performance bound is important, because it is in qualitative agreement with the empirical behavior of approximate PI. Typically, in the beginning, the method tends to make rapid and fairly monotonic progress, but eventually it gets into an oscillatory pattern. This happens after \( J_{\mu_k} \) gets within an error zone of size

\[
\epsilon + \frac{2\alpha \delta}{(1 - \alpha)^2}
\]

or smaller, and then \( J_{\mu_k} \) oscillates fairly randomly within that zone; see Fig. 4.2.4. In practice, the error bound of Prop. 4.2.4 tends to be pessimistic, so the zone of oscillation is usually much narrower than what is suggested by the bound. However, the bound itself can be proved to be tight, in worst
Figure 4.2.4 Illustration of typical behavior of approximate PI. In the early iterations, the method tends to make rapid and fairly monotonic progress, until $J_{\mu_k}$ gets within an error zone of size less than

$$\frac{\epsilon + 2\alpha \delta}{(1 - \alpha)^2}.$$ 

After that $J_{\mu_k}$ oscillates randomly within that zone. The figure is oversimplified since it shows the difference $J_{\mu_k} - J^*$ at a single state. For different states the nature of the error oscillation may be different.

case. This is shown with an example in the book [BeT96], Section 6.2.3.

Note that the bound of Prop. 4.2.4 also holds in the case of infinite state and control spaces discounted problems, when there are infinitely many policies (see [Ber18a], Prop. 2.4.3).

Performance Bound for the Case Where Policies Converge

Generally, the policy sequence $\{\mu_k\}$ generated by approximate PI may oscillate between several policies, as noted earlier. However, under some circumstances the sequence will converge to some policy $\tilde{\mu}$, i.e.,

$$\mu^{k+1} = \mu^k = \tilde{\mu} \quad \text{for some } k.$$ 

(4.39)

An important case where this happens is aggregation methods, which will be discussed in Chapter 6. In this case the behavior of the method is more regular, and we can show the following bound, which is more favorable than the one of Prop. 4.2.4 by a factor $1/(1 - \alpha)$, as illustrated in Fig. 4.2.5. For the proof, see [Ber19a] (or the original source [BeT96], Section 6.2.2).
Figure 4.2.5 Illustration of typical behavior of approximate PI when policies converge. The method tends to make monotonic progress, and $J_{\mu_k}$ converges within an error zone of size less than $(\epsilon + 2\alpha \delta)/(1 - \alpha)$.

Proposition 4.2.5: (Performance Bound for Approximate PI when Policies Converge) Let $\hat{\mu}$ be a policy generated by the approximate PI algorithm under conditions (4.37), (4.38), and (4.39). Then we have

$$\max_{i=1,\ldots,n} |J_{\hat{\mu}}(i) - J^*(i)| \leq \frac{\epsilon + 2\alpha \delta}{1 - \alpha}.$$

We finally note that related performance bounds hold for optimistic PI methods, where the policy evaluation is performed with just a few approximate value iterations (cf. Section 4.6.2). These bounds are similar to the ones of the nonoptimistic method, and do not suggest superiority of one type of PI method over the other. Their derivation is quite complicated; see [Ber12], Chapter 2, or [Ber18a], Section 2.5.2, and the end-of-chapter references. Moreover, there are versions of the preceding bounds, which apply to more general abstract DP problems, such as the semi-Markov problems of Section 4.4 and others; see [Ber18a], Section 2.4.1.

4.3 ASYNCHRONOUS DISTRIBUTED ALGORITHMS IN DYNAMIC PROGRAMMING
4.4 ASYNCHRONOUS DISTRIBUTED POLICY ITERATION

4.5 APPROXIMATE POLICY ITERATION - PARTITIONED ARCHITECTURES

4.6 APPROXIMATE POLICY ITERATION - PARTIAL OBSERVATION PROBLEMS

4.7 NOTES AND SOURCES