Chapter 2
Value Function Approximation in Complex Queueing Systems

Sandjai Bhulai

Abstract The application of Markov decision theory to the control of queueing systems often leads to models with enormous state spaces. Hence, direct computation of optimal policies with standard techniques and algorithms is almost impossible for most practical models. A convenient technique to overcome this issue is to use one-step policy improvement. For this technique to work, one needs to have a good understanding of the queueing system under study, and its (approximate) value function under policies that decompose the system into less complicated systems. This warrants the research on the relative value functions of simple queueing models, that can be used in the control of more complex queueing systems. In this chapter we provide a survey of value functions of basic queueing models and show how they can be applied to the control of more complex queueing systems.

Key words: One-step policy improvement, Relative value function, Complex queueing systems, Approximate dynamic programming

2.1 Introduction

In its simplest form, a queueing system can be described by customers arriving for service, waiting for service if it cannot be provided immediately, and leaving the system after being served. The term customer is used in a general sense here, and does not necessarily refer to human customers. The performance of the system is usually measured on the basis of throughput rates or the average time customers

S. Bhulai (✉)
Faculty of Sciences, Vrije Universiteit Amsterdam, De Boelelaan 1081a,
1081 HV Amsterdam, The Netherlands
e-mail: s.bhulai@vu.nl

R.J. Boucherie, N.M. van Dijk (eds.), *Markov Decision Processes in Practice*,
International Series in Operations Research & Management Science 248,
DOI 10.1007/978-3-319-47766-4_2

remain in the system. Hence, the average cost criterion is usually the preferred criterion in queueing systems (see, e.g., [18, 23]).

It is hardly necessary to emphasize the applicability of theory on queueing systems in practice, since many actual queueing situations occur in daily life. During the design and operation of such systems, many decisions have to be made; think of the availability of resources at any moment in time, or routing decisions. This effect is amplified by the advances in the field of data and information processing, and the growth of communication networks (see, e.g., [1]). Hence, there is a need for optimal control of queueing systems.

In general, the control of queueing systems does not fit into the framework of discrete time Markov decision problems, in which the time between state transitions is fixed. When the time spent in a particular state follows an arbitrary probability distribution, it is natural to allow the decision maker to take actions at any point in time. This gives rise to decision models in which the system evolution is described continuously in time, and in which the cost accumulates continuously in time as well. If the cost function is independent of the time spent in a state, such that it only depends on the state and action chosen at the last decision epoch, then we can restrict our attention to models in which the decision epochs coincide with the transition times. Moreover, if the time spent between decision epochs follows an exponential distribution, then the queueing system under study can be reformulated as a discrete-time Markov decision problem. This discretization technique, known as uniformization, is due to [12], and was later formalized by Serfozo [21].

After uniformization, Markov decision theory can, in principle, be applied to the control of queueing systems, and usually gives rise to infinite state spaces with unbounded cost functions. In this setting, value iteration and policy iteration algorithms can be used to derive optimal policies. However, these algorithms require memory storage of a real-valued vector of at least the size of the state space. For denumerably infinite state spaces this is not feasible, and one has to rely on appropriate techniques to bound the state space (see, e.g., [8]). But even then, memory may be exhausted by the size of the resulting state space; this holds even more for multi-dimensional models. This phenomenon, known as the curse of dimensionality (see [4]), gives rise to high-dimensional systems and calls for approximation methods to the optimal policies.

A first approximation method can be distilled from the policy iteration algorithm. Suppose that one fixes a policy which, while perhaps not optimal, is not totally unreasonable either. This policy is evaluated through analytically solving the Poisson equations induced by the policy in order to obtain the long-run expected average cost and the average cost value function under this policy. Next, using these expressions the policy can be improved by doing one policy improvement step. We know that this policy is better, i.e., has a smaller or equal (if already optimal) average cost. It must be expected that the improved policy is sufficiently complicated to render another policy evaluation step impossible.

The idea of applying one-step policy improvement goes back to [15]. It has been successfully applied by Ott and Krishnan [17] for deriving state-dependent routing schemes for high-dimensional circuit-switched telephone networks. The initial

policy in their system was chosen such that the communication lines were independent. In that case, the value function of the system is the sum of the value functions for the independent lines, which are easier to derive. Since then, this idea has been used in many papers, see, e.g., [10, 19, 20, 25]. The initial policy in these papers was chosen such that the queues were independent, reducing the analysis to single queues. For this purpose, [5, 6] presented a unified approach to obtain the value function of the various queueing systems for various cost structures.

In this chapter, we review the theory and applications of the relative value functions of the most fundamental queueing systems: the M/Cox(r)/1 single-server queue, the M/M/s multi-server queue, the M/M/s/s blocking system, and a priority queueing system. Each of these systems has its own distinguishing characteristics. The M/Cox(r)/1 queue is a very general single-server queue that allows for approximations to single-server systems with non-exponential service distributions. Both the M/M/s and the M/M/s/s multi-server queues typically occur in many service systems such as telecommunication systems, call centers, cash registers, etc. The priority queueing system is an intriguing example in which the queues are not independent, and this is reflected as well in the relative value function.

The different characteristics of the queueing systems that we study will be illustrated by three examples of controlled queueing systems. In the first example, we consider the problem of routing customers to parallel single-server queues, where each server has its own general service time distribution. We demonstrate how the M/Cox(r)/1 queue can approximate the single-server queues after which one-step of policy improvement leads to nearly optimal policies. In the second example, we study a skill-based routing problem in a call center. This is a highly complex problem in which agent groups in the system are highly dependent on each others states. We illustrate how the value function of the multi-server queue can be adjusted to take these dependencies into account. In the last example, we focus on a controlled polling system. Here the value function of the priority queueing system can be used to take actions that directly consider the dependencies between the states of the different queues.

The remainder of the chapter is structured as follows. In Sect. 2.2 we develop difference calculus that can be used to obtain the relative value function of most one-dimensional queueing systems. We proceed with the survey of the relative value functions of the fundamental queueing systems in Sect. 2.3 and list all relevant special cases as well. We then illustrate the use of these value functions through three examples in Sect. 2.4 on routing to parallel queues, Sect. 2.5 on skill-based routing in call centers, and Sect. 2.6 on a controlled polling model, respectively.

2.2 Difference Calculus for Markovian Birth-Death Systems

In this section we will study the relative value function for a Markovian birth-death queueing system. We shall derive a closed-form expression for the long-run expected average costs and the relative value function by solving the Poisson equa-

tions. The Poisson equations give rise to linear difference equations. Due to the nature of the Poisson equations, the difference equations have a lot of structure when the state description is one-dimensional. Therefore, it is worthwhile to study difference equations prior to the analysis of the birth-death queueing system. We shall restrict attention to second-order difference equations, since this is general enough to model birth-death queueing processes.

Let $V(x)$ be an arbitrary function defined on $\mathbb{N}_0$. Define the backward difference operator Δ as

$$\Delta V(x) = V(x) - V(x-1),$$

for $x \in \mathbb{N}$. Note that the value of $V(x)$ can be expressed as

$$V(x) = V(k-1) + \sum_{i=k}^{x} \Delta V(i), \tag{2.1}$$

for every $k \in \mathbb{N}$ such that $k \leq x$. This observation is key to solving first-order difference equations, and second-order difference equations when one solution to the homogeneous equation is known. We first state the result for first-order difference equations. The result can be found in Chap. 2 of Mickens [14], but is stated less precise there.

Lemma 2.1. *Let $f(x)$ be a function defined on $\mathbb{N}$ satisfying the relation*

$$f(x+1) - \gamma(x) f(x) = r(x), \tag{2.2}$$

with γ and r arbitrary functions defined on $\mathbb{N}$ such that $\gamma(x) \neq 0$ for all $x \in \mathbb{N}$. With the convention that an empty product equals one, $f(x)$ is given by

$$f(x) = f(1)\,Q(x) + Q(x) \sum_{i=1}^{x-1} \frac{r(i)}{Q(i+1)}, \quad \textit{with} \quad Q(x) = \prod_{i=1}^{x-1} \gamma(i).$$

Proof. Let the function $Q(x)$ be as stated in the theorem. By assumption we have that $Q(x) \neq 0$ for all $x \in \mathbb{N}$. Dividing Eq. (2.2) by $Q(x+1)$ yields

$$\Delta \left[\frac{f(x+1)}{Q(x+1)} \right] = \frac{f(x+1)}{Q(x+1)} - \frac{f(x)}{Q(x)} = \frac{r(x)}{Q(x+1)}.$$

From Expression (2.1) with $k = 2$ it follows that

$$f(x) = Q(x) \frac{f(1)}{Q(1)} + Q(x) \sum_{i=2}^{x} \frac{r(i-1)}{Q(i)} = f(1)\,Q(x) + Q(x) \sum_{i=1}^{x-1} \frac{r(i)}{Q(i+1)}.$$

Note that the condition $\gamma(x) \neq 0$ for all $x \in \mathbb{N}$ is not very restrictive in practice. If there is a state $y \in \mathbb{N}$ for which $\gamma(y) = 0$, then the analysis can be reduced to two other first-order difference equations, namely the part for states $x < y$, and the part for states $x > y$ for which $\gamma(y) = 0$ is the boundary condition.

The solution to first-order difference equations plays an important part in solving second-order difference equations when one solution to the homogeneous equation is known. In that case, the second-order difference equation can be reduced to a first-order difference equation expressed in the homogeneous solution. Application of Lemma 2.1 then gives the solution to the second-order difference equation. The following theorem summarizes this result.

Theorem 2.1. *Let $V(x)$ be a function defined on $\mathbb{N}_0$ satisfying the relation*

$$V(x+1)+\alpha(x)V(x)+\beta(x)V(x-1)=q(x), \tag{2.3}$$

with α, β, and q arbitrary functions defined on $\mathbb{N}$ such that $\beta(x)\neq 0$ for all $x\in\mathbb{N}$. Suppose that one homogeneous solution is known, say $V_1^h(x)$, such that $V_1^h(x)\neq 0$ for all $x\in\mathbb{N}_0$. Then, with the convention that an empty product equals one, $V(x)$ is given by

$$\frac{V(x)}{V_1^h(x)}=\frac{V(0)}{V_1^h(0)}+\left[\Delta\left[\frac{V(1)}{V_1^h(1)}\right]\right]\sum_{i=1}^{x}Q(i)+\sum_{i=1}^{x}Q(i)\sum_{j=1}^{i-1}\frac{q(j)}{V_1^h(j+1)\,Q(j+1)},$$

where $Q(x)=\prod_{i=1}^{x-1}\beta(i)\,V_1^h(i-1)/V_1^h(i+1)$.

Proof. Note that V_1^h always exists, since a second-order difference equation has exactly two linearly independent homogeneous solutions (see Theorem 3.11 of Mickens [14]). The known homogeneous solution $V_1^h(x)$ satisfies

$$V_1^h(x+1)+\alpha(x)V_1^h(x)+\beta(x)V_1^h(x-1)=0. \tag{2.4}$$

Set $V(x)=V_1^h(x)\,u(x)$ for an arbitrary function u defined on $\mathbb{N}_0$. Substitution into Eq. (2.3) yields

$$V_1^h(x+1)\,u(x+1)+\alpha(x)V_1^h(x)\,u(x)+\beta(x)V_1^h(x-1)\,u(x-1)=q(x).$$

By subtracting Eq. (2.4) $u(x)$ times from this expression, and rearranging the terms, we derive

$$\Delta u(x+1)-\gamma(x)\Delta u(x)=r(x),$$

with

$$\gamma(x)=\frac{V_1^h(x-1)}{V_1^h(x+1)}\beta(x),\quad\text{and}\quad r(x)=\frac{q(x)}{V_1^h(x+1)}.$$

From Lemma 2.1 it follows that

$$\Delta u(x)=[\Delta u(1)]\,Q(x)+Q(x)\sum_{j=1}^{x-1}\frac{r(j)}{Q(j+1)},\quad\text{with}\quad Q(x)=\prod_{i=1}^{x-1}\gamma(i).$$

From Expression (2.1) it finally follows that

$$u(x) = u(0) + [\Delta u(1)] \sum_{i=1}^{x} Q(i) + \sum_{i=1}^{x} Q(i) \sum_{j=1}^{i-1} \frac{r(j)}{Q(j+1)}.$$

Since $V(x) = V_1^h(x)\, u(x)$ it follows that $u(x) = V(x)/V_1^h(x)$ for $x = 1, 2$.

From Theorem 3.11 of Mickens [14] it follows that a second-order difference equation has exactly two homogeneous solutions that are also linearly independent. In Theorem 2.1 it is assumed that one homogeneous solution V_1^h is known. From the same theorem it follows that the second homogeneous solution is determined by $V_2^h(x) = V_1^h(x) \sum_{i=1}^{x} Q(i)$. This can be easily checked as follows.

$$V_2^h(x+1) + \alpha(x) V_2^h(x) + \beta(x) V_2^h(x-1) =$$
$$V_1^h(x+1) \sum_{i=1}^{x+1} Q(i) + \alpha(x) V_1^h(x) \sum_{i=1}^{x} Q(i) + \beta(x) V_1^h(x-1) \sum_{i=1}^{x-1} Q(i) =$$
$$\sum_{i=1}^{x} Q(i) \Big[V_1^h(x+1) + \alpha(x) V_1^h(x) + \beta(x) V_1^h(x-1) \Big] +$$
$$V_1^h(x+1)\, Q(x+1) - \beta(x) V_1^h(x-1)\, Q(x) = 0.$$

The last equality follows from the fact that

$$Q(x+1) = \frac{V_1^h(x-1)}{V_1^h(x+1)} \beta(x)\, Q(x) = \frac{V_1^h(0)\, V_1^h(1)}{V_1^h(x)\, V_1^h(x+1)} \prod_{i=1}^{x} \beta(i), \quad x \in \mathbb{N}.$$

Note that the homogeneous solutions V_1^h and V_2^h are also linearly independent, since their Casorati determinant $C(x) = V_1^h(x)\, V_1^h(x+1)\, Q(x+1)$ is non-zero for all $x \in \mathbb{N}_0$ (see Sects. 3.2 and 3.3 of Mickens [14]).

With the use of Theorem 2.1 as a tool, we are ready to study birth-death queueing systems. The name birth-death stems from the fact that arrivals (birth) and departures (death) of customers occur only in sizes of one, i.e., batch arrivals and batch services are not allowed. Let state $x \in \mathcal{X} = \mathbb{N}_0$ denote the number of customers in the system. When the system is in state x, customers arrive according to a Poisson process with rate $\lambda(x)$. At the same time, customers receive service with an exponentially distributed duration at rate $\mu(x)$, such that $\mu(0) = 0$. Moreover, the system is subject to costs $c(x)$ in state x, where $c(x)$ is a polynomial function of x.

For stability of the system, we assume that

$$0 < \liminf_{x \to \infty} \frac{\lambda(x)}{\mu(x)} \leq \limsup_{x \to \infty} \frac{\lambda(x)}{\mu(x)} < 1.$$

Moreover, we assume that the transition rates satisfy

$$0 < \inf_{x \in \mathbb{N}_0} \big(\lambda(x) + \mu(x) \big) \leq \sup_{x \in \mathbb{N}_0} \big(\lambda(x) + \mu(x) \big) < \infty.$$

Without loss of generality we can assume that $\sup_{x\in\mathbb{N}_0}\big(\lambda(x)+\mu(x)\big)<1$. This can always be obtained after a suitable renormalization without changing the long-run system behavior. After uniformization, the resulting birth-death process has the following transition rate matrix.

$$\begin{array}{lll} & P_{0,0}=1-\lambda(0), & P_{0,1}=\lambda(0), \\ P_{x,x-1}=\mu(x), & P_{x,x}=1-\lambda(x)-\mu(x), & P_{x,x+1}=\lambda(x), \qquad x=1,2,\ldots. \end{array}$$

Inherent to the model is that the Markov chain has a unichain structure. Hence, we know that the long-run expected average cost g is constant. Furthermore, the value function V is unique up to a constant. As a result we can use this degree of freedom to set $V(0)=0$. Then, the Poisson equations for this system are given by

$$g+\big(\lambda(x)+\mu(x)\big)V(x)=\lambda(x)V(x+1)+\mu(x)V\big([x-1]^+\big)+c(x), \tag{2.5}$$

for $x\in\mathbb{N}_0$, where $[x]^+=\max\{0,x\}$. When the state space $\mathcal{X}$ is not finite, as is the case in our model, it is known that there are many pairs of the average cost g and the corresponding value function V that satisfy the Poisson equations (see, e.g., [7]). There is only one pair that is the correct solution, however, which we refer to as the unique solution. Finding this pair involves constructing a weighted norm such that the Markov chain is geometrically recurrent with respect to that norm. Next, this weighted norm poses extra conditions on the solution to the Poisson equations such that the unique solution to the Poisson equations can be obtained. Solving the Poisson equations therefore requires two steps; first one has to find an expression that satisfies the Poisson equations (existence), then one has to show that it is the unique solution (uniqueness).

In order to show the existence of solutions to the Poisson equations, we use the relation with the difference calculus by rewriting the Poisson equations in the form of Eq. (2.3), with

$$\alpha(x)=-\frac{\lambda(x)+\mu(x)}{\lambda(x)}, \quad \beta(x)=\frac{\mu(x)}{\lambda(x)}, \quad \text{and} \quad q(x)=\frac{g-c(x)}{\lambda(x)}.$$

Observe that $\beta(x)\neq 0$ for $x\in\mathbb{N}$ due to the stability assumption. Furthermore, note that for any set of Poisson equations, the constant function is always a solution to the homogeneous equations. This follows directly from the fact that P is a transition probability matrix. Hence, by applying Theorem 2.1 with $V_1^h(x)=1$ for $x\in\mathbb{N}_0$, it follows that the solution to Eq. (2.5) is given by

$$V(x)=\frac{g}{\lambda(0)}\sum_{i=1}^{x}Q(i)+\sum_{i=1}^{x}Q(i)\sum_{j=1}^{i-1}\frac{q(j)}{Q(j+1)}, \quad \text{with} \quad Q(x)=\prod_{i=1}^{x-1}\frac{\mu(i)}{\lambda(i)}.$$

We know that this expression is not the unique solution to the Poisson equations (see, e.g., [22]). Hence, we need to construct a weighted norm w such that the Markov chain is w-geometrically recurrent with respect to a finite set $M\subset\mathbb{N}_0$ to

show uniqueness (see Chap. 2 of Spieksma [22]), i.e., there should be an $\varepsilon > 0$ such that $\|{}_M P\|_w \leq 1 - \varepsilon$, where

$$ {}_M P_{ij} = \begin{cases} P_{ij}, & j \notin M, \\ 0, & j \in M, \end{cases} $$

and

$$ \|A\|_w = \sup_{i \in \mathcal{X}} \frac{1}{w(i)} \sum_{j \in \mathcal{X}} |A_{ij}| \, w(j). $$

Due to stability of the Markov chain there exists a constant $K \in \mathbb{N}_0$ such that $\lambda(x)/\mu(x) < 1$ for all $x > K$. Let $M = \{0, \ldots, K\}$, and assume that $w(x) = z^x$ for some $z > 1$. Now consider

$$ \sum_{y \notin M} \frac{P_{xy} w(y)}{w(x)} = \begin{cases} \lambda(K) z, & x = K, \\ \lambda(K+1) z + \big(1 - \lambda(K+1) - \mu(K+1)\big), & x = K+1, \\ \lambda(x) z + \big(1 - \lambda(x) - \mu(x)\big) + \frac{\mu(x)}{z}, & x > K+1. \end{cases} $$

We need to choose z such that all expressions are strictly less than 1. The first expression immediately gives $z < 1/\lambda(K)$. The second expression gives that $z < 1 + \mu(K+1)/\lambda(K+1)$. Solving the third expression shows that $1 < z < \mu(x)/\lambda(x)$ for $x > K+1$. Define $z^* = \min\{1/\lambda(K), \inf_{x \in \mathbb{N} \setminus M} \mu(x)/\lambda(x)\}$, and note that the choice of M ensures us that $z^* > 1$. Then, the three expressions are strictly less than 1 for all $z \in (1, z^*)$. Thus, we have shown that for $w(x) = z^x$ with $1 < z < z^*$, there exists an $\varepsilon > 0$ such that $\|{}_M P\|_w \leq 1 - \varepsilon$. Hence, the Markov chain is w-geometrically recurrent with respect to M.

Note that c is bounded with respect to the supremum norm weighted by w, since $c(x)$ is a polynomial in x by assumption. We know that the unique value function V is bounded with respect to the norm w (Chap. 2 of Spieksma [22]). Therefore, the value function cannot contain terms θ^x with $\theta > 1$, since the weight function can be chosen such that $z \in (1, \min\{\theta, z^*\})$. Consequently, $\|V\|_w = \infty$, hence the value function cannot grow exponentially with a growth factor greater than 1. Thus, we have the following corollary.

Corollary 2.1. *The value function of a stable birth-death queueing process cannot grow exponentially with a growth factor greater than 1.*

2.3 Value Functions for Queueing Systems

In this section, we provide the relative value functions of the most fundamental queueing systems: the M/Cox(r)/1 single-server queue with its special cases, the M/M/s multi-server queue, the M/M/s/s blocking system, and a priority queueing system. The value functions will be used in the examples later to illustrate their effectiveness in the control of complex queueing systems.

2.3.1 The M/Cox(r)/1 Queue

Consider a single server queueing system to which customers arrive according to a Poisson process with parameter λ. The service times are independent identically distributed and follow a Coxian distribution of order r. Thus, the service of a customer can last up to r exponential phases. The mean duration of phase i is μ_i for $i = 1,\dots,r$. The service starts at phase 1. After phase i the service ends with probability $1-p_i$, or it enters phase $i+1$ with probability p_i for $i = 1,\dots,r-1$. The service is completed with certainty after phase r, if not completed at an earlier phase. We assume that $p_i > 0$ for $i = 1,\dots,r-1$ to avoid trivial situations. Let $\mathcal{X} = \{(0,0)\} \cup \mathbb{N} \times \{0,\dots,r-1\}$ denote the state space, where for $(x,y) \in \mathcal{X}$ the component x represents the number of customers in the system, and y the number of completed phases of the service process.

Assume that the system is subject to costs for holding a customer in the system. Without loss of generality we assume that unit costs are incurred for holding a customer per unit of time in the system. Let $u_t(x)$ denote the total expected cost up to time t when the system starts in state x. Let $\gamma(i) = \prod_{k=1}^{i} p_k$ for $i = 0,\dots,r-1$ with the convention that $\gamma(0) = 1$. Note that the Markov chain satisfies the unichain condition. Assume that the stability condition

$$\sum_{k=1}^{r} \gamma(k-1)\frac{\lambda}{\mu_k} = \sum_{k=1}^{r}\prod_{l=1}^{k-1} p_l \frac{\lambda}{\mu_k} < 1,$$

holds, such that, consequently, the average cost $g = \lim_{t\to\infty} u_t(x)/t$ is independent of the initial state x due to Proposition 8.2.1 of Puterman [18]. Therefore, the dynamic programming optimality equations for the M/Cox(r)/1 queue are given by

$$\begin{aligned}
g + \lambda V(0,0) &= \lambda V(1,0), \\
g + (\lambda+\mu_i)V(x,i-1) &= \lambda V(x+1,i-1) + p_i\mu_i V(x,i) + \\
&\quad (1-p_i)\mu_i V(x-1,0) + x, \qquad i = 1,\dots,r-1, \\
g + (\lambda+\mu_r)V(x,r-1) &= \lambda V(x+1,r-1) + \mu_r V(x-1,0) + x.
\end{aligned}$$

The solution to this set of equations is given in the following theorem.

Theorem 2.2 (Theorem 1 in [5]). *Let $\gamma(i) = \prod_{k=1}^{i} p_k$ for $i = 0,\dots,r-1$ with the convention that $\gamma(0) = 1$. Define*

$$\alpha = \frac{\sum_{k=1}^{r} \frac{\gamma(k-1)}{\mu_k}}{1 - \sum_{k=1}^{r} \frac{\gamma(k-1)}{\mu_k}\lambda}, \text{ and } a_0 = \sum_{k=1}^{r} \frac{1-\gamma(k-1)}{\mu_k}\lambda\alpha - \sum_{k=1}^{r}\sum_{l=1}^{k-1} \frac{\gamma(l-1)}{\mu_k\mu_l}\lambda(1+\lambda\alpha).$$

The solution to the Poisson equations is then given by the average cost $g = \lambda(\alpha + a_0)$ and the corresponding value function

$$V(x,y) = \alpha \frac{x(x+1)}{2} + \left[a_0 + \left[\frac{1}{\gamma(y)} - 1 \right] \alpha - \sum_{k=1}^{y} \frac{\gamma(k-1)}{\gamma(y)} \frac{1+\lambda\alpha}{\mu_k} \right] x$$

$$- \left[a_0 + \sum_{k=y+1}^{r} \frac{\gamma(k-1)}{\gamma(y)} \frac{\lambda}{\mu_k} \left(\sum_{l=1}^{k-1} \frac{\gamma(l-1)}{\gamma(k-1)} \frac{1+\lambda\alpha}{\mu_l} - \left[\frac{1}{\gamma(k-1)} - 1 \right] \alpha \right) \right],$$

for $(x,y) \in \mathcal{X}$.

2.3.2 Special Cases of the M/Cox(r)/1 Queue

In this section we consider important special cases of the M/Cox(r)/1 queue. This includes queues with service-time distributions that are modeled by the hyper-exponential (H_r), the hypo-exponential (Hypo$_r$), the Erlang (E_r), and the exponential (M) distribution.

The $M/H_r/1$ Queue

The single server queue with hyper-exponentially distributed service times of order r is obtained by letting the service times consist only of one exponential phase with parameter μ_i with probability q_i for $i = 1, \ldots, r$. Note that the hyper-exponential distribution has the property that the coefficient of variation is greater than or equal to one. Unfortunately, the hyper-exponential distribution is not directly obtained from the Coxian distribution through interpretation, but rather from showing that the Laplace-transforms of the distribution functions are equal for specific parameter choices. In the case of $r = 2$ this result follows from, e.g., Appendix B of Tijms [24]. The general case is obtained by the following theorem.

Theorem 2.3 (Theorem 4 in [5]). *Under the assumption that* $\mu_1 > \cdots > \mu_r$, *a Coxian distribution with parameters* $(p_1, \ldots, p_{r-1}, \mu_1 \ldots, \mu_r)$ *is equivalent to a hyper-exponential distribution with parameters* $(q_1, \ldots, q_r, \mu_1, \ldots, \mu_r)$ *when the probabilities* p_i *are defined by*

$$p_i = \frac{\sum_{j=i+1}^{r} q_j \prod_{k=1}^{i} (\mu_k - \mu_j)}{\mu_i \sum_{j=i}^{r} q_j \prod_{k=1}^{i-1} (\mu_k - \mu_j)}, \tag{2.6}$$

for $i = 1, \ldots, r-1$.

The $M/$Hypo$_r/1$ Queue

The single server queue with hypo-exponentially distributed service times of order r is obtained by letting the service be the sum of r independent random variables that are exponentially distributed with parameter μ_i at phase i for $i = 1, \ldots, r$. Thus,

it can be obtained from the M/Cox(r)/1 queue by letting $p_1 = \cdots = p_{r-1} = 1$. The optimality equations are given by

$$\begin{aligned} g + \lambda V(0,0) &= \lambda V(1,0), \\ g + (\lambda + \mu_i)V(x,i-1) &= \lambda V(x+1,i-1) + \mu_i V(x,i) + x, \qquad i = 1,\ldots,r-1, \\ g + (\lambda + \mu_r)V(x,r-1) &= \lambda V(x+1,r-1) + \mu_r V(x-1,0) + x. \end{aligned}$$

Define $\beta(i) = \sum_{k=1}^{i}(1/\mu_k)$, then the average cost is given by

$$g = \frac{\lambda\beta(r)}{1-\lambda\beta(r)} - \frac{\lambda^2}{1-\lambda\beta(r)} \sum_{k=1}^{r} \frac{\beta(k-1)}{\mu_k},$$

and under the assumption $\lambda\beta(r) < 1$ the value function becomes

$$\begin{aligned} V(x,y) = \frac{\beta(r)x(x+1)}{2\big(1-\lambda\beta(r)\big)} - \frac{x}{1-\lambda\beta(r)} \left[\lambda \sum_{k=1}^{r} \frac{\beta(k-1)}{\mu_k} + \beta(y)\right] \\ + \frac{\lambda}{1-\lambda\beta(r)} \sum_{k=1}^{y} \frac{\beta(k-1)}{\mu_k}. \end{aligned}$$

The $M/E_r/1$ Queue

The single server queue with Erlang distributed service times of order r is obtained by letting the service be the sum of r independent random variables having a common exponential distribution. Thus, it can be obtained from the M/Cox(r)/1 queue by letting $p_1 = \cdots = p_{r-1} = 1$ and $\mu = \mu_1 = \cdots = \mu_r$. Note that the Erlang distribution can also be seen as a special case of the hypo-exponential distribution, and has a coefficient of variation equal to $1/r \leq 1$. The optimality equations are given by

$$\begin{aligned} g + \lambda V(0,0) &= \lambda V(1,0), \\ g + (\lambda + \mu)V(x,i-1) &= \lambda V(x+1,i-1) + \mu V(x,i) + x, \qquad i = 1,\ldots,r-1, \\ g + (\lambda + \mu)V(x,r-1) &= \lambda V(x+1,r-1) + \mu V(x-1,0) + x. \end{aligned}$$

The average cost is given by

$$g = \frac{\lambda r}{\mu - \lambda r} - \frac{\lambda^2 r(r-1)}{2\mu(\mu - \lambda r)},$$

and under the assumption $\lambda r/\mu < 1$ the value function becomes

$$V(x,y) = \frac{rx(x+1)}{2(\mu - \lambda r)} - \frac{x}{(\mu - \lambda r)} \left[\frac{\lambda r(r-1)}{2\mu} + y\right] + \frac{\lambda y(y-1)}{2\mu(\mu - \lambda r)}.$$

The M/M/1 Queue

The standard single server queue with exponentially distributed service times is obtained by having one phase in the M/Cox(r)/1 queue only, i.e., $r = 1$. Let $\mu = \mu_1$, then the optimality equations are equivalent to

$$g + (\lambda + \mu)V(x) = \lambda V(x+1) + \mu V([x-1]^+) + x,$$

where $[x]^+ = \max\{x, 0\}$. The average cost is given by $g = \lambda/(\mu - \lambda)$, and under the assumption $\lambda/\mu < 1$, the value function becomes

$$V(x) = \frac{x(x+1)}{2(\mu - \lambda)}.$$

2.3.3 The M/M/s Queue

Consider a queueing system with s identical independent servers. The arrivals are determined by a Poisson process with parameter λ. The service times are exponentially distributed with parameter μ. An arriving customer that finds no idle server waits in a buffer of infinite size. We focus on the average number of customers in the system represented by generating a cost of x monetary units when x customers are present in the system. The Poisson equations for this M/M/s queue are then given by

$$\begin{aligned}
g + \lambda V(0) &= \lambda V(1), \\
g + (\lambda + x\mu)V(x) &= x + \lambda V(x+1) + x\mu V(x-1), && x = 1, \ldots, s-1, \\
g + (\lambda + s\mu)V(x) &= x + \lambda V(x+1) + s\mu V(x-1), && x = s, s+1, \ldots,
\end{aligned}$$

From the first equation we can deduce that $V(1) = g/\lambda$. The second equation can be written as Eq. (2.3) with $\alpha(x) = -(\lambda + x\mu)/\lambda$, $\beta(x) = x\mu/\lambda$, and $q(x) = g/\lambda$. From Theorem 2.1 we have that

$$V(x) = \frac{g}{\lambda} \sum_{i=1}^{x} \sum_{k=0}^{i-1} \frac{(i-1)!}{(i-k-1)!} \left(\frac{\lambda}{\mu}\right)^{-k} - \frac{1}{\lambda} \sum_{i=1}^{x} (i-1) \sum_{k=0}^{i-2} \frac{(i-2)!}{(i-k-2)!} \left(\frac{\lambda}{\mu}\right)^{-k}.$$

Let $\rho = \lambda/(s\mu)$. For $x = s, s+1, \ldots$ we have

$$V(x) = V(s) - \frac{(x-s)\rho}{1-\rho} \frac{g}{\lambda} + \left[\frac{(x-s)(x-s+1)\rho}{2(1-\rho)} + \frac{(x-s)\big(\rho + s(1-\rho)\big)\rho}{(1-\rho)^2} \right] \frac{1}{\lambda} +$$

$$\frac{(1/\rho)^{x-s} - 1}{1-\rho} \left[\frac{\rho}{1-\rho} \frac{g}{\lambda} + h(s) - h(s-1) - \frac{\big(\rho + s(1-\rho)\big)\rho}{\lambda(1-\rho)^2} \right].$$

The long-term expected average costs, which represents the average number of customers in the system in this case, is given by

$$g = \frac{(s\rho)^s \rho}{s!(1-\rho)^2}\left[\sum_{n=0}^{s-1}\frac{(s\rho)^n}{n!} + \frac{(s\rho)^s}{s!(1-\rho)}\right]^{-1} + s\rho.$$

2.3.4 The Blocking Costs in an M/M/s/s Queue

Consider the previous queueing system with s identical independent servers but with no buffers for customers to wait. An arriving customer that finds no idle server is blocked and generates a cost of one monetary unit. Let state x denote the number of customers in the system. The Poisson equations for this M/M/s/s queue are then given by

$$\begin{aligned} g + \lambda V(0) &= \lambda V(1), \\ g + (\lambda + x\mu)V(x) &= \lambda V(x+1) + x\mu V(x-1), \qquad x = 1,\ldots,s-1, \\ g + s\mu V(s) &= \lambda + s\mu V(s-1). \end{aligned}$$

In order to obtain the relative value function V, we repeat the argument as in the case of the M/M/s queue. This yields the following value function.

$$\begin{aligned} V(x) &= \frac{g}{\lambda}\sum_{i=1}^{x}\sum_{k=0}^{i-1}\frac{(i-1)!}{(i-k-1)!}\left(\frac{\lambda}{\mu}\right)^{-k} \\ &= \frac{(\lambda/\mu)^s/s!}{\sum_{i=0}^{s}(\lambda/\mu)^i/i!}\sum_{i=1}^{x}\sum_{k=0}^{i-1}\frac{(i-1)!}{(i-k-1)!}\left(\frac{\lambda}{\mu}\right)^{-k}. \end{aligned} \tag{2.7}$$

The average costs is given by

$$g = \frac{(\lambda/\mu)^s/s!}{\sum_{i=0}^{s}(\lambda/\mu)^i/i!}\,\lambda.$$

2.3.5 Priority Queues

Consider a queueing system with two classes of customers arriving according to a Poisson process. There is only one server available serving either a class-1 or a class-2 customer with exponentially distributed service times. A class-i customer has arrival rate λ_i, and is served with rate μ_i for $i = 1,2$. The system is subject to holding costs and switching costs. The cost of holding a class-i customer in the system for one unit of time is c_i for $i = 1,2$. The cost of switching from serving a class-1 to a class-2 customer (from a class-2 to a class-1 customer) is s_1 (s_2, respectively).

The system follows a priority discipline indicating that class-1 customers have priority over class-2 customers. The priority is also preemptive, i.e., when serving a class-2 customer, the server switches immediately to serve a class-1 customer upon arrival of a class-1 customer. Upon emptying the queue of class-1 customers, the service of class-2 customers, if any present, is resumed from the point where it was interrupted. Due to the exponential service times, this is equivalent to restarting the service for this customer.

Let state (x,y,z) for $x,y \in \mathbb{N}_0$, $z \in \{1,2\}$ denote that there are x class-1 and y class-2 customers present in the system, with the server serving a class-z customer, if present. Let $\rho_i = \lambda_i/\mu_i$ for $i = 1,2$ and assume that the stability condition $\rho_1 + \rho_2 < 1$ holds. Then the Markov chain is stable and $g < \infty$ holds. Furthermore, the Markov chain satisfies the unichain condition, hence the Poisson equations are given by

$$\begin{aligned}
g + (\lambda_1+\lambda_2+\mu_1)V(x,y,1) &= c_1 x + c_2 y + \lambda_1 V(x+1,y,1) + \lambda_2 V(x,y+1,1) + \\
&\quad \mu_1 V(x-1,y,1), && x > 0, y \geq 0, \\
V(0,y,1) &= s_1 + V(0,y,2), && y > 0, \\
V(x,y,2) &= s_2 + V(x,y,1), && x > 0, y \geq 0, \\
g + (\lambda_1+\lambda_2+\mu_2)V(0,y,2) &= c_2 y + \lambda_1 V(1,y,2) + \lambda_2 V(0,y+1,2) + \\
&\quad \mu_2 V(0,y-1,2), && y > 0, \\
g + (\lambda_1+\lambda_2)V(0,0,z) &= \lambda_1 V(1,0,z) + \lambda_2 V(0,1,z), && z = 1,2.
\end{aligned}$$

First observe that, given the values of $V(x,y,1)$, the values of $V(x,y,2)$ are easily obtained by considering the difference equations: $V(x,y,2) = V(x,y,1) + (\lambda_1 s_2 - \lambda_2 s_1)/(\lambda_1+\lambda_2)\mathbb{1}(x=0,y=0) - s_1\mathbb{1}(x=0,y>0) + s_2\mathbb{1}(x>0,y\geq 0)$. Therefore, we will only show the expression for $V(x,y,1)$, as $V(x,y,2)$ is easily derived from it. Let $V = V_g + V_{c_1} + V_{c_2} + V_{s_1} + V_{s_2}$ be the decomposition of the value function. Then the previous observation directly prescribes that V_g, V_{c_1}, and V_{c_2} are independent of z. Furthermore, the value function V_{c_1} equals the value function of the single server queue due to the preemptive priority behavior of the system.

The other value functions can be derived by setting $V(x,y) = c_1 x^2 + c_2 x + c_3 y^2 + c_4 y + c_5 xy + c_6$ with constants c_i for $i = 1,\ldots,6$ to be determined. Substitution of this equality into the optimality equations yields a new set of equations that is easier to solve. Let θ be the unique root $\theta \in (0,1)$ of the equation

$$\lambda_1 x^2 - (\lambda_1+\lambda_2+\mu_1)x + \mu_1 = 0.$$

Then, solving for the constants yields the solution to the optimality equations, which is given by

$$V_g(x,y,z) = -\frac{x}{\mu_1(1-\rho_1)}\,g,$$

$$V_{c_1}(x,y,z) = \frac{x(x+1)}{2\mu_1(1-\rho_1)}\,c_1 + \frac{\rho_1 x}{\mu_1(1-\rho_1)^2}\,c_1,$$

$$V_{c_2}(x,y,z) = \frac{\lambda_2 x(x+1)}{2\mu_1^2(1-\rho_1)(1-\rho_1-\rho_2)}c_2 + \frac{\rho_2(\mu_1-\lambda_1+\mu_2\rho_1)x}{\mu_1^2(1-\rho_1)^2(1-\rho_1-\rho_2)}c_2 +$$

$$\frac{y(y+1)}{2\mu_2(1-\rho_1-\rho_2)}c_2 + \frac{xy}{\mu_1(1-\rho_1-\rho_2)}c_2,$$

$$V_{s_i}(x,y,1) = \frac{\lambda_1\theta}{\lambda_1+\lambda_2}\frac{\rho_1\rho_2 x}{1-\rho_1}s_i + \frac{\lambda_1\theta}{\lambda_1+\lambda_2}\frac{\lambda_1 y}{\mu_2}s_i + \frac{\lambda_1}{\lambda_1+\lambda_2}s_i\mathbb{1}(y>0) +$$

$$\frac{\lambda_1}{\lambda_1+\lambda_2}(1-\theta^x)s_i\mathbb{1}(x>0,y=0), \qquad i=1,2,$$

with

$$g = \frac{\rho_1}{1-\rho_1}c_1 + \frac{\rho_2(\mu_1-\mu_1\rho_1+\mu_2\rho_1)}{\mu_1(1-\rho_1)(1-\rho_1-\rho_2)}c_2 + (s_1+s_2)\times$$

$$\left[\lambda_1\left\{\rho_1\left(\frac{\lambda_1\theta}{\lambda+\lambda_2}-1\right)+\frac{\lambda_1}{\lambda_1+\lambda_2}(1-\theta)\right\}+\lambda_2\left\{\frac{\lambda_1\theta}{\lambda_1+\lambda_2}\frac{\lambda_1}{\mu_2}+\frac{\lambda_1}{\lambda_1+\lambda_2}\right\}\right].$$

2.4 Application: Routing to Parallel Queues

In this section we illustrate how the value function can be used to obtain nearly optimal policies for dynamic routing problems to parallel single server queues. The general idea is to start with a policy such that each queue behaves as an independent single server queue. By doing so, the average cost and the value function can be readily determined from the results of the previous sections. Next, one step of policy iteration is performed to obtain an improved policy without having to compute the policy iteratively.

The control problem that we study can be formalized as follows. Consider two parallel infinite buffer single server queues. The service times of server i are Coxian distributed with order r_i with parameters $(p_1^i,\ldots,p_{r_i-1}^i,\mu_1^i,\ldots,\mu_{r_i}^i)$ for $i=1,2$. Furthermore, queue i has holding costs h_i for $i=1,2$. An arriving customer can be sent to either queue 1 or queue 2. The objective is to minimize the average costs. Note that the distribution of the service times does not necessarily need to be Coxian. Since the class of Coxian distributions is dense in the set of non-negative distribution functions, we can approximate these distributions with a Coxian distribution by using, e.g., the Expectation-Maximization (EM) algorithm (see [2]). The EM algorithm is an iterative scheme that minimizes the information divergence given by the Kullback-Leibler information for a fixed order r. For the cases we have consid-

ered, a Coxian distribution of order $r = 5$ was sufficiently adequate to describe the original distribution.

Let x_i be the number of customers in queue i, and y_i the number of completed phases of the customer in service, if there is one, at queue i for $i = 1,2$. Given that the service time distribution is adequately described by a Coxian distribution, the optimality equations for this system are given by

$$\begin{aligned} g + \big(\lambda + \mathbb{1}_{\{x_1>0\}}\,\mu^1_{y_1+1} + \mathbb{1}_{\{x_2>0\}}\,\mu^2_{y_2+1}\big)V(x_1,y_1,x_2,y_2) &= h_1x_1 + h_2x_2 + \\ \lambda \min\{V(x_1+1,x_2,y_1,y_2), V(x_1,x_2+1,y_1,y_2)\} &+ \\ \mathbb{1}_{\{x_1>0\}}\,\mu^1_{y_1+1}[p^1_{y_1+1}V(x_1,x_2,y_1+1,y_2) + \overline{p}^1_{y_1+1}V(x_1-1,x_2,0,y_2)] &+ \\ \mathbb{1}_{\{x_2>0\}}\,\mu^2_{y_2+1}[p^2_{y_2+1}V(x_1,x_2,y_1,y_2+1) + \overline{p}^2_{y_2+1}V(x_1,x_2-1,y_1,0)], & \end{aligned}$$

for $(x_i,y_i) \in \{(0,0)\} \cup \mathbb{N} \times \{0,\ldots,r_i-1\}$ with $p^i_{r_i} = 0$ and $\overline{p}^i_y = 1 - p^i_y$ when $i = 1,2$.

Take as initial control policy the Bernoulli policy with parameter $\eta \in [0,1]$, i.e., the policy that splits the arrivals into two streams such that arrivals occur with rate $\eta\lambda$ at queue 1, and with rate $(1-\eta)\lambda$ at queue 2. Under the Bernoulli policy, the optimality equation is obtained by replacing the term $\lambda \min\{V(x_1+1,x_2,y_1,y_2), V(x_1,x_2+1,y_1,y_2)\}$ with $\eta\lambda V(x_1+1,x_2,y_1,y_2) + (1-\eta)\lambda V(x_1,x_2+1,y_1,y_2)$. Hence, it follows that the two queues behave as independent single server queues for which the average cost g_i and the value function V_i for $i = 1,2$ can be determined from Theorem 2.2. The average cost g' and the value function V' for the whole system is then given by the sum of the individual average costs $g = g_1 + g_2$ and the sum of the individual value functions $V' = V_1 + V_2$, respectively. Note that for a specified set of parameters, the optimal Bernoulli policy obtained by minimizing with respect to η is straightforward. We shall therefore use the optimal Bernoulli policy in the numerical examples. The policy improvement step now follows from the minimizing action in $\min\{V'(x_1+1,x_2,y_1,y_2), V'(x_1,x_2+1,y_1,y_2)\}$.

The Coxian distribution allows for many interesting numerical experiments. Therefore, we restrict ourselves to four representative examples that display the main ideas. We shall use the notation g_B, g', and g^* for the average cost obtained under the optimal Bernoulli policy, the one-step improved policy, and the optimal policy, respectively. Moreover, we set $h_1 = h_2 = 1$, and $\lambda = \frac{3}{2}$ for the first example, and $\lambda = 1$ for the other three examples.

We first start with two queues having a Coxian $C(2)$ distribution. Queue 1 has an Erlang E_2 distribution with parameter $\mu = 2$, such that the mean and the variance of the service time is 1 and 2, respectively. The parameters at queue 2 are chosen such that the mean remains 1, but the variance increases to 3, 4, and 5, respectively. Table 2.1 summarizes the results, and shows that the one-step improved policy has a close to optimal value. The proportional extra cost $(g' - g^*)/g^*$ is practically zero in all cases.

Parameters for queue 2	g_B	g'	g^*
$p^2 = \frac{2}{3}, \mu^2 = (2, \frac{4}{3})$	5.147786	3.208688	3.208588
$p^2 = \frac{1}{2}, \mu^2 = (2, 1)$	5.405949	3.332179	3.332038
$p^2 = \frac{2}{5}, \mu^2 = (2, \frac{4}{5})$	5.652162	3.445815	3.445787

Table 2.1: Numerical results for $r = 2$ with $p^1 = 1, \mu^1 = (2,2)$

Next, we show a similar experiment with $r = 5$. The service distribution at queue 1 is fixed at a hypo-exponential Hypo_5 distribution with parameter $\mu = (2,3,2,3,4)$. Again the one-step improved policy performs quite well. Table 2.2 shows that the greatest proportional extra cost is given by 0.03 (the third experiment).

Parameters for queue 2	g_B	g'	g^*
$p^2 = (\frac{9}{10}, \frac{4}{5}, \frac{7}{10}, \frac{3}{5}), \mu^2 = (2,3,2,3,4)$	6.175842	3.787954	3.783727
$p^2 = (\frac{3}{5}, \frac{7}{10}, \frac{4}{5}, \frac{9}{10}), \mu^2 = (2,3,2,3,4)$	3.729859	2.493349	2.480818
$p^2 = (\frac{2}{5}, \frac{1}{5}, \frac{4}{5}, \frac{1}{2}), \quad \mu^2 = (3,2,4,2,3)$	1.399628	1.169286	1.132408

Table 2.2: Numerical results for $r = 5$ with $p^1 = (1,1,1,1), \mu^1 = (2,3,2,3,4)$

In the third example we take an Erlang E_2 distribution with parameter $\mu = 2$ at queue 1, and a lognormal distribution with parameters $\mu = 0.5$ and $\sigma = 1$ at queue 2. Recall that the probability density function f of the lognormal distribution is given by

$$f(x) = \frac{1}{x\sqrt{2\pi}\sigma} \cdot e^{-\frac{1}{2}\left(\frac{\ln(x)-\mu}{\sigma}\right)^2},$$

for $x > 0$. We approximate the lognormal distribution with a $\text{Cox}(r)$ distribution of order $r = 2, 5, 10$, and 20, respectively, using the EM-algorithm. We also compare this with a two-moment fit of the Coxian distribution. Let X be a random variable having a coefficient of variation $c_X \geq \frac{1}{2}\sqrt{2}$, then the following is suggested by Marie [13]: $\mu_1 = 2/\mathbb{E}X$, $p_1 = 0.5/c_X^2$, and $\mu_2 = p_1\mu_1$.

The results of the EM-algorithm and the 2-moment fit are displayed in Fig. 2.1. The fit with the Cox(20) distribution is omitted, since it could not be distinguished from the lognormal probability density. Therefore, the optimal value when using the Cox(20) distribution can be considered representative for the optimal value when using the lognormal distribution. Note that the EM-approximation with the Cox(2) distribution captures more characteristics of the lognormal distribution than the 2-moment fit. This result is also reflected in Table 2.3, since the value of the optimal policy g^* for the Cox(2) distribution is closer to the value of g^* when the Cox(20) distribution is used. The greatest proportional extra cost for the EM-approximations is given by 0.02 (EM fit with $r = 2$).

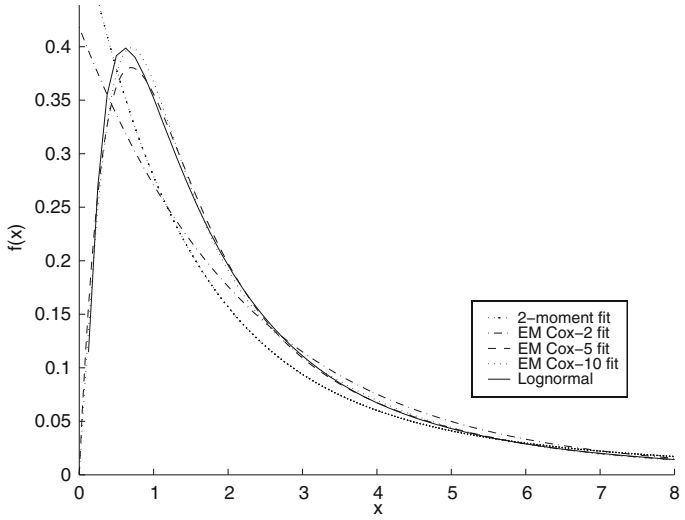

Fig. 2.1: Lognormal ($\mu = 0.5$, $\sigma = 1$) probability density

Approximation for queue 2	g_B	g'	g^*
2-moment fit	4.617707	3.021571	2.976950
EM algorithm with $r = 2$	4.554838	2.982955	2.933345
EM algorithm with $r = 5$	4.526013	2.965625	2.919847
EM algorithm with $r = 10$	4.527392	2.963318	2.917011
EM algorithm with $r = 20$	4.527040	2.963311	2.917169

Table 2.3: Lognormal distribution: numerical results with $p^1 = 1, \mu^1 = (2,2)$

In the final example we take an Erlang E_2 distribution with parameter $\mu = 2$ at queue 1, and a Weibull distribution with parameters $a = 0.3$ and $b = 0.107985$. Recall that the probability density function f of the Weibull distribution is given by

$$f(x) = ax^{a-1} \frac{e^{-(\frac{x}{b})^a}}{b^a},$$

for $x > 0$. Note that the parameters a and b are chosen such that the Weibull distribution has mean 1. We approximate the Weibull distribution by a Cox(r) distribution of order $r = 2, 5, 10$, and 20, respectively, using the EM-algorithm. The results of the EM-algorithm are depicted in Fig. 2.2. Again we omitted the fit of the Cox(20) distribution, since it could not be distinguished from the Weibull probability density. Moreover, since the coefficient of variation is less than $\frac{1}{2}\sqrt{2}$ we also omitted the two-moment fit. The results in Table 2.4 again indicate that the one-step improved policy has a close to optimal value, since the proportional extra cost is practically zero.

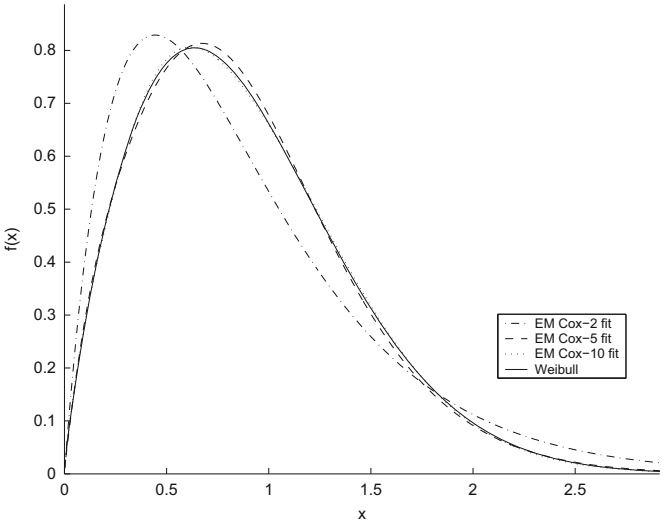

Fig. 2.2: Weibull ($a = 1.8$, $b = 1$) probability density

Approximation for queue 2	g_B	g'	g^*
EM algorithm with $r = 2$	1.563547	1.167233	1.167232
EM algorithm with $r = 5$	1.524032	1.148638	1.148511
EM algorithm with $r = 10$	1.522566	1.148570	1.148100
EM algorithm with $r = 20$	1.522387	1.148826	1.148552

Table 2.4: Weibull distribution: numerical results with $p^1 = 1, \mu^1 = (2,2)$

The previous examples show that the one-step policy improvement method yields nearly optimal policies, even when non-Markovian service distributions are approximated by a Coxian distribution. For the lognormal and the Weibull distribution that we studied, a Coxian distribution of order $r = 5$ was already sufficient for an adequate approximation. Note that the one-step improved policy can be easily obtained for more than two queues. In this section we restricted attention to two queues, since the numerical computation of the value of the optimal policy g^* becomes rapidly intractable for more than two queues. Observe that the computational complexity is exponential in the number of queues in contrast to a single step of policy iteration that has linear complexity in the number of queues.

2.5 Application: Dynamic Routing in Multiskill Call Centers

Consider a multi-skill call center in which agents handle calls that require different skills. We represent the skill set $\mathcal{S}$ by $\mathcal{S} = \{1, \ldots, N\}$. We assume that calls that require skill $s \in \mathcal{S}$ arrive to the call center according to a Poisson process with rate λ_s. Let $\mathcal{G} = \mathcal{P}(\mathcal{S})$ denote the groups with different skill sets defined by the power set of all the skills. Let $\mathcal{G}_s = \{G \in \mathcal{G} \,|\, s \in G\}$ denote all skill groups that include skill $s \in \mathcal{S}$. The agents are grouped according to their skills and can therefore be indexed by the elements of $\mathcal{G}$. Each group $G \in \mathcal{G}$ consists of S_G agents, and serves a call that requires a skill within group G with independent exponentially distributed times with parameter μ_G.

Scenario 1: A Call Center with No Waiting Room

Suppose that we are dealing with a loss system, i.e., there are no buffers at all in our model. Hence, there is no common queue for calls to wait, so every arriving call has either to be routed to one of the agent groups immediately, or has to be blocked and is lost. Also, when a call is routed to a group that has no idle servers left, the call is lost. The objective in this system is to minimize the average number of lost calls.

Let us formulate the problem as a Markov decision problem. Fix an order of the elements of $\mathcal{G}$, say $\mathcal{G} = (G_1, \ldots, G_{2^N-1})$, and define the state space of the Markov decision problem by $\mathcal{X} = \prod_{i=1}^{2^N-1} \{0, \ldots, S_{G_i}\}$. Then the $|\mathcal{G}|$-dimensional vector $\mathbf{x} \in \mathcal{X}$ denotes the state of the system according to the fixed order, i.e., x_G is the number of calls in service at group $G \in \mathcal{G}$. Also, represent by the $|\mathcal{G}|$-dimensional unit vector e_G the vector with zero at every entry, except for a one at the entry corresponding to G according to the fixed order. When a call that requires skill $s \in \mathcal{S}$ arrives, the decision maker can choose to block the call, or to route the call to any agent group $G \in \mathcal{G}_s$ for which $x_G < S_G$. Hence, the action set is defined by $\mathcal{A}_s = \{G \in \mathcal{G}_s \,|\, x_G < S_G\} \cup \{b\}$, where action b stands for blocking the call. Therefore, for an arriving call that requires skill $s \in \mathcal{S}$ the transition rates $p(\mathbf{x}, a, \mathbf{y})$ of going from $\mathbf{x} \in \mathcal{X}$ to $\mathbf{y} \in \mathcal{X}$ after taking decision $a \in \mathcal{A}_s$ are given by $p(\mathbf{x}, b, \mathbf{x}) = \lambda_s$, $p(\mathbf{x}, a, \mathbf{x} + e_a) = \lambda_s$ when $a \neq b$, and zero otherwise. The transition rates for the service completions are given by $p(\mathbf{x}, a, \mathbf{x} - e_G) = x_G \mu_G$ for $\mathbf{x} \in \mathcal{X}$, $G \in \mathcal{G}$ and any action a. The objective is modeled by the cost function $c(\mathbf{x}, a) = 1$ for any $\mathbf{x} \in \mathcal{X}$ and $a = b$.

The tuple $(\mathcal{X}, \{\mathcal{A}_s\}_{s \in \mathcal{S}}, p, c)$ defines the Markov decision problem. After uniformizing the system (see Sect. 11.5 of Puterman [18]) we obtain the dynamic programming optimality equation for the system given by

$$g + \Big[\sum_{s\in\mathcal{S}} \lambda_s + \sum_{G\in\mathcal{G}} S_G\mu_G\Big] V(\mathbf{x}) = \sum_{s\in\mathcal{S}} \lambda_s \min\{1 + V(\mathbf{x}), V(\mathbf{x}+e_G) \,|\, G \in \mathcal{G}_s, x_G < S_G\}$$
$$+ \sum_{G\in\mathcal{G}} x_G\mu_G V(\mathbf{x}-e_G) + \sum_{G\in\mathcal{G}} (S_G - x_G)\mu_G V(\mathbf{x}), \tag{2.8}$$

where g is the long-term expected average costs, and V is the relative value function.

Note the unusual place of the minimization in Expression (2.8): the actions in our case depend on the type of arriving call. It is easy to rewrite the optimality equation in standard formulation (see, e.g., Chap. 5 of Koole [11]), but this would complicate the notation considerably.

For the problem of skill-based routing we will base the approximate relative value function on a static randomized policy for the initial routing policy. Let $\mathcal{G}^{(n)} = \{G \in \mathcal{G} \,|\, |G| = n\}$ be all agent groups that have exactly n skills for $n = 1,\ldots,N$. Define accordingly, $\mathcal{G}_s^{(n)} = \{G \in \mathcal{G}^{(n)} \,|\, s \in G\}$ to be all agent groups that have n skills, including skill $s \in \mathcal{S}$. For a given skill s, this creates a hierarchical structure $\mathcal{G}_s^{(1)},\ldots,\mathcal{G}_s^{(N)}$ from specialized agents having only one skill to cross-trained generalists having all skills. For a call center with three skills, we would have three levels in the hierarchy: the specialists (groups $\{1\}$, $\{2\}$, and $\{3\}$), the agents with two skills (groups $\{1, 2\}$, $\{1, 3\}$, and $\{2, 3\}$), and the generalists (group $\{1, 2, 3\}$). In the following paragraphs we will describe the steps in the one-step policy improvement algorithm in more detail using this call center with three skills as illustration.

Initial Policy

The initial policy $\hat{\pi}$, on which we will base the approximate relative value function, tries to route a call requiring skill s through the hierarchy, i.e., it tries $\mathcal{G}_s^{(1)}$ first, and moves to $\mathcal{G}_s^{(2)}$ if there are no idle agents in $\mathcal{G}_s^{(1)}$. In $\mathcal{G}_s^{(2)}$ there may be more than one group that one can route to. The initial policy routes the call according to fixed probabilities to the groups in $\mathcal{G}_s^{(2)}$, i.e., it splits the overflow stream from $\mathcal{G}_s^{(1)}$ into fixed fractions over the groups in $\mathcal{G}_s^{(2)}$. The call progresses through the hierarchy whenever it is routed to a group with no idle agents until it is served at one of the groups, or it is blocked at $\mathcal{G}_s^{(N)}$ eventually. The rationale behind the policy that sends calls to the agents with the fewest number of skills first has been rigorously justified in [9, 16]. To illustrate this for three skills: the overflow of calls from group $\{1\}$ are split by fixed fractions α and $\overline{\alpha} = 1 - \alpha$ in order to be routed to groups $\{1,2\}$ and $\{1,3\}$, respectively. The overflow process from groups $\{2\}$ and $\{3\}$ are treated accordingly by the fractions β and γ, respectively.

We have yet to define how to choose the splitting probabilities in the initial routing policy. In order to define these, we ignore the fact that the overflow process is not a Poisson process, and we consider all overflows to be independent. Thus, we do

as if the overflow process at group $G \in \mathcal{G}_s^{(i)}$ is a Poisson process with rate λ_G times the blocking probability. Together with the splitting probabilities, one can compute the rate of the Poisson arrivals at each station in $\mathcal{G}^{(i+1)}$ composed of the assumed independent Poisson processes. The splitting probabilities are then chosen such that the load at every group in $\mathcal{G}^{(i+1)}$ is balanced. To illustrate this for the call center with three skills, recall the Erlang loss formula $B(\lambda,\mu,S)$ for an M/M/*S*/*S* queue with arrival rate λ and service rate μ,

$$B(\lambda,\mu,S) = \frac{(\lambda/\mu)^S/S!}{\sum_{i=0}^{S}(\lambda/\mu)^i/i!}. \tag{2.9}$$

The overflow rate at group $\{i\}$ for $i = 1,2,3$ is then given by $\lambda_i B(\lambda_i,\mu_{\{i\}},S_{\{i\}})$. Hence, the arrival rate at the groups in $\mathcal{G}^{(2)}$ are given by

$$\begin{aligned}
\lambda_{\{1,2\}} &= \lambda_1 B(\lambda_1,\mu_{\{1\}},S_{\{1\}})\,\alpha + \lambda_2 B(\lambda_2,\mu_{\{2\}},S_{\{2\}})\,\beta,\\
\lambda_{\{1,3\}} &= \lambda_1 B(\lambda_1,\mu_{\{1\}},S_{\{1\}})\,(1-\alpha) + \lambda_3 B(\lambda_3,\mu_{\{3\}},S_{\{3\}})\,\gamma,\\
\lambda_{\{2,3\}} &= \lambda_2 B(\lambda_2,\mu_{\{2\}},S_{\{2\}})\,(1-\beta) + \lambda_3 B(\lambda_3,\mu_{\{3\}},S_{\{3\}})\,(1-\gamma).
\end{aligned}$$

The splitting probabilities can be determined by minimizing the average cost in the system. Since the average cost under this policy is the sum of the blocking probabilities of each queue in the system, the optimal splitting probability could create asymmetry in the system, i.e., one queue has a very low blocking probability whereas another has a high blocking probability. Numerical experiments show that a situation with a more balanced blocking probability over all queues leads to better one-step improved policies. Therefore, we choose the splitting probabilities such that

$$\frac{\lambda_{\{1,2\}}}{S_{\{1,2\}}\mu_{\{1,2\}}} = \frac{\lambda_{\{1,3\}}}{S_{\{1,3\}}\mu_{\{1,3\}}} = \frac{\lambda_{\{2,3\}}}{S_{\{2,3\}}\mu_{\{2,3\}}}.$$

In case this equation does not have a feasible solution, we choose the splitting probabilities such that we minimize

$$\left|\frac{\lambda_{\{1,2\}}}{S_{\{1,2\}}\mu_{\{1,2\}}} - \frac{\lambda_{\{1,3\}}}{S_{\{1,3\}}\mu_{\{1,3\}}}\right| + \left|\frac{\lambda_{\{1,2\}}}{S_{\{1,2\}}\mu_{\{1,2\}}} - \frac{\lambda_{\{2,3\}}}{S_{\{2,3\}}\mu_{\{2,3\}}}\right| + \left|\frac{\lambda_{\{1,3\}}}{S_{\{1,3\}}\mu_{\{1,3\}}} - \frac{\lambda_{\{2,3\}}}{S_{\{2,3\}}\mu_{\{2,3\}}}\right|.$$

Finally, the arrival rate at the last group with all skills is given by

$$\lambda_{\{1,2,3\}} = \sum_{G\in\mathcal{G}^{(2)}} \lambda_G B(\lambda_G,\mu_G,S_G).$$

Policy Evaluation

In the policy evaluation step, one is required to solve the long-run expected average costs and the relative value function from the Poisson equations for the policy $\hat{\pi}$. Note that for our purpose of deriving an improved policy, it suffices to have the relative value function only. Under the initial policy this leads to solving the relative value function of a series of multi-server queues in tandem. This is a difficult problem that is yet unsolved even for tandem series of single server queues. Therefore, we choose to approximate the relative value function based on the assumptions made in defining the initial policy.

For the approximation we assume that the overflow process is indeed a Poison process and that they are independent from all other overflow processes. We approximate the relative value function by the sum of the relative value functions for each agent group. More formally, let $V_G(x_G, \lambda_G, \mu_G, S_G)$ be the relative value function for an M/M/S_G/S_G system with arrival rate λ_G and service rate μ_G, evaluated when there are x_G calls present. The value function for the whole system is then approximated by

$$\hat{V}(\mathbf{x}) = \sum_{G \in \mathcal{G}} V_G(x_G, \lambda_G, \mu_G, S_G). \tag{2.10}$$

Policy Improvement

By substitution of the relative value function determined in the policy evaluation step into the optimality equations, one can derive a better policy by evaluating

$$\min\{1 + \hat{V}(\mathbf{x}), \hat{V}(\mathbf{x} + e_G) \,|\, G \in \mathcal{G}_s, x_G < S_G\} \; =$$

$$\min\{1, V_G(x_G + 1, \lambda_G, \mu_G, S_G) - V_G(x_G, \lambda_G, \mu_G, S_G) \,|\, G \in \mathcal{G}_s, x_G < S_G\}.$$

The last term follows from subtracting $V(\mathbf{x})$ from all terms in the minimization and using the linear structure of $V(\mathbf{x})$.

In Table 2.5 we find the results of the one-step policy improvement algorithm for scenario 1. The parameters for λ_X, μ_X, and S_X in the table are organized for the groups $\{1\}$, $\{2\}$, and $\{3\}$. The parameters for μ_{XY} and S_{XY} are ordered for groups $\{1,2\}$, $\{1,3\}$, and $\{2,3\}$, respectively. The greatest proportional extra cost $\Delta = (g' - g^*)/g^*$ over all experiments in Table 2.5 is 12.9% (the last experiment). Other experiments show that the proportional extra costs lies within the range of 0–13%. Thus, we can see that the improved policy has a good performance already after one step of policy improvement.

Scenario 2: A Call Center with Specialists and Generalists Only

Suppose that we are dealing with a call center having agents with one skill (specialists) and fully cross-trained agents (generalists) only, i.e., $\mathcal{G} = (G_1, \ldots, G_{|\mathcal{S}|}, G_{|\mathcal{S}|+1}) = (\{1\}, \ldots, \{N\}, \{1, \ldots, N\})$. In this scenario we assume that calls that require skill $s \in \mathcal{S}$ are pooled in a common infinite buffer queue after which they are assigned to

λ_X	μ_X	μ_{XY}	S_X	S_{XY}	g	g'	g^*	Δ
6 6 6	1.0 1.0 1.0	1.0 1.0 1.0	2 2 2	2 2 2	0.361	0.349	0.344	1.505
6 5 4	2.0 1.5 1.0	1.5 1.0 1.0	2 2 2	2 2 2	0.170	0.147	0.143	2.781
7 6 5	2.0 1.5 1.0	1.5 1.5 1.0	3 3 3	2 2 2	0.119	0.099	0.096	3.264
7 6 5	1.5 1.0 1.0	1.5 1.0 1.0	3 3 3	3 3 2	0.130	0.107	0.103	3.930
6 5 4	2.0 1.5 1.0	1.5 1.0 1.0	3 3 3	2 2 2	0.072	0.057	0.054	5.672
6 5 4	2.0 1.5 1.0	1.5 1.5 1.0	3 3 3	2 2 2	0.061	0.046	0.042	8.011
10 4 4	1.5 1.0 1.0	1.5 1.5 1.0	2 2 2	2 2 2	0.281	0.274	0.252	8.816
10 6 3	1.5 1.0 1.0	1.5 1.0 1.0	3 3 3	3 3 2	0.160	0.148	0.131	12.851

Table 2.5: Numerical results for scenario 1 with $\mu_{\{1,2,3\}} = 1$ and $S_{\{1,2,3\}} = 2$

a specialist or a generalist in a first-come first-served order. The objective in this system is to minimize the average number of calls in the system, which in its turn is related to the average waiting time of a call.

In addition to the state of the agents groups, we also have to include the state of the queues in the state space. For this purpose, let the $|\mathcal{S}|$-dimensional vector $\mathbf{q}$ denote the state of the queues, i.e., q_s is number of calls in queue s that require skill $s \in \mathcal{S}$. Moreover, the system also has to address the agent-selection problem and the call-selection problem. The first problem occurs when one has to assign an agent to an arriving call. The second problem occurs when an agent becomes idle and a potential call in the queue can be assigned. Following the same approach as in scenario 1, we obtain the dynamic programming optimality equation given by

$$\begin{aligned} g + \Big[\sum_{s\in\mathcal{S}} \lambda_s + \sum_{G\in\mathcal{G}} S_G \mu_G\Big] V(\mathbf{q},\mathbf{x}) = \sum_{s\in\mathcal{S}} q_s + \sum_{G\in\mathcal{G}} x_G + \sum_{s\in\mathcal{S}} \lambda_s \tilde{V}(\mathbf{q}+e_s,\mathbf{x}) + \\ \sum_{G\in\mathcal{G}} x_G \mu_G \tilde{V}(\mathbf{q},\mathbf{x}-e_G) + \sum_{G\in\mathcal{G}} (S_G - x_G)\mu_G V(\mathbf{q},\mathbf{x}), \end{aligned} \tag{2.11}$$

where $\tilde{V}(\mathbf{q},\mathbf{x}) = \min\{V(\mathbf{q}-e_s,\mathbf{x}+e_G) \,|\, s \in \mathcal{S}, G \in \mathcal{G}_s, q_s > 0, x_G < S_G\} \cup \{V(\mathbf{q},\mathbf{x})\}$. The first two terms on the right-hand side represent the cost structure, i.e., the number of calls in the system. The third and fourth term represent the agent-selection and the call-selection decisions, respectively.

Initial Policy

In principle, we could take as initial policy $\hat{\pi}$ a similar policy as used in scenario 1: a fraction α_s of type s calls are assigned to the corresponding specialists, and a fraction $1-\alpha_s$ are assigned to the generalists. Instead of using the relative value function for the M/M/S/S queue, we could use the relative value function for the M/M/S queue. However, this approximation would then assume a dedicated queue in front of the group of generalists. Consequently, a customer that is sent to the group of generalists that are all busy would still have to wait when a specialist of that call type is idle. Therefore, this Bernoulli policy does not efficiently use the resources in the system, and leads to an inefficient one-step improved policy. To overcome the inefficiency of the Bernoulli policy, we instead use a policy that uses the specialists first, and assigns a call to a generalist only if a generalist is available while all specialists that can handle that call type are busy. The effectiveness of this policy has been rigorously justified in [9, 16].

Policy Evaluation

The relative value function for the policy $\hat{\pi}$, that uses specialists first and then generalists, is complicated. The policy $\hat{\pi}$ creates a dependence between the different agent groups that prohibit the derivation of a closed-form expression for the relative value function. Therefore, we approximate the relative value function by $\hat{V}$ as follows. Let $V_W(x,\lambda,\mu,S)$ be the relative value function of an M/M/S queueing system. The approximation $\hat{V}$ for the policy $\hat{\pi}$ is then given by

$$\hat{V}(\mathbf{q},\mathbf{x}) = \sum_{s\in\mathcal{S}} V_W(q_s+x_s,\tilde{\lambda}_s,\mu_{\{s\}},S_{\{s\}}) + V_W\big((q_1+x_1-S_1)^+ + \cdots + (q_N+x_N-S_N)^+ + x_{N+1},\lambda_{G_{N+1}},\mu_{G_{N+1}},S_{G_{N+1}}\big),$$

with $\tilde{\lambda}_s = \lambda_s\big(1-B(\lambda_s,\mu_{\{s\}},S_{\{s\}})\big)$ the approximate rate to the specialists of call type s, and $\lambda_{G_{N+1}} = \sum_{s\in\mathcal{S}} \lambda_s B(\lambda_s,\mu_{\{s\}},S_{\{s\}})$ the approximate rate to the generalists. Note that this approximation follows the idea of the motivating initial policy in that it ensures that all idle specialists of type s, given by $S_s - x_s$, are used for all calls in the queue of the same type. This results in $(q_s-[S_s-x_s])^+ = \max\{q_s+x_s-S_s,0\}$ calls that cannot be served. These calls are therefore waiting for a generalist to become idle. The approximation does take into account also the fact that a specialist can become idle before a generalist, and immediately assigns a call to the specialist.

The idea behind the approximation is that it roughly estimates the different flows to the different agent groups and then computes the value function as if the calls are waiting simultaneously at the two queues where they can be served. Note that strictly hierarchical architectures in which agents groups are structured so that no overflow of calls has to be split between two possible subsequent agent pools can be dealt with similarly. Observe that it might be possible that the overflow to the generalists is larger than their service rate. However, the system will still be stable

because the actual number of calls that will be served by the specialists will be higher, unless the system load is quite close to one.

Policy Improvement

In the policy improvement step, the initial policy $\hat{\pi}$ is improved by evaluating

$$\min\{\hat{V}(\mathbf{q}-e_s,\mathbf{x}+e_G)\,|\,s\in\mathcal{S},G\in\mathcal{G}_s,q_s>0,x_G<S_G\}\cup\{\hat{V}(\mathbf{q},\mathbf{x})\}.$$

Note that in this case the policy has to be determined for both the agent-assignment and the call-selection decisions.

The results for scenario 2 with also three skills are given in Table 2.6. The first line seems to suggest that no significant gains over the static policy can be obtained when the service rates of the specialists and the generalists are equal to each other. In Tables 2.7, 2.8, 2.9 we have varied the service rate of the generalists under different loads. The results show that as the system is offered higher loads, the gains by usingthe one-step improved policy are also higher as compared to the static routing

λ_X	μ_X	S_X	$\mu_{\{1,2,3\}}$	$S_{\{1,2,3\}}$	g	g'	g^*	Δ
6 6 6	2.0 2.0 2.0	3 3 3	2.0	3	11.043	10.899	10.746	1.429
6 3 3	1.5 1.0 1.0	3 3 3	1.0	3	20.122	19.259	18.361	4.892
6 5 4	2.5 2.0 1.5	2 2 2	2.0	3	13.186	11.562	11.314	2.187
6 5 4	1.8 1.8 1.2	3 3 3	1.2	3	16.348	14.931	14.602	2.253
7 6 5	2.5 2.0 1.5	3 3 3	2.0	2	15.269	13.310	13.127	1.397
7 6 5	1.8 1.8 1.2	3 3 3	1.8	3	23.260	20.091	19.125	5.054
10 6 3	3.0 2.0 1.0	3 3 3	1.0	2	31.834	26.844	25.184	6.594
6 5 4	3.0 2.0 1.0	3 3 3	1.0	2	20.083	14.403	13.795	4.404

Table 2.6: Numerical results for scenario 2

policy. Next, we scale the system such that the increase in the offered load was compensated by faster service rates or by an increase in the number of servers. Table 2.10 shows that when the service rates are scaled, the gains over the static policy remain roughly unaffected. However, when more servers are added, it slightly decreases. From the other lines in Table 2.6 we can conclude that the gain over the static policy is greater in asymmetric systems. In conclusion, we can observe that the improved policy has good performance and that its performance is close to the performance of the optimal policy.

Note that our proposed method is scalable, and can easily be used for bigger call centers. In this section, however, we restricted ourselves to a call center with three skills, since the computation of optimal policies becomes numerically difficult for bigger call centers. The optimal policy grows exponentially in the number of skills, while a single step of policy iteration has linear complexity.

$\mu_{\{1,2,3\}}$	g	g'	g^*	Δ
1.5	9.012	8.933	8.928	0.051
1.6	8.773	8.707	8.703	0.045
1.7	8.559	8.504	8.500	0.047
1.8	8.366	8.320	8.316	0.045
1.9	8.192	8.154	8.150	0.044
2.0	8.035	8.006	7.999	0.091
2.1	7.892	7.864	7.851	0.166
2.2	7.762	7.713	7.675	0.496
2.3	7.644	7.561	7.489	0.962
2.4	7.535	7.405	7.302	1.418
2.5	7.436	7.238	7.118	1.690

Table 2.7: Scenario 2—$\lambda_X = 5$, $\mu_X = 2$, $S_X = 3$, and $S_{\{1,2,3\}} = 3$

$\mu_{\{1,2,3\}}$	g	g'	g^*	Δ
1.5	13.674	13.142	12.877	2.055
1.6	12.995	12.595	12.344	2.035
1.7	12.405	12.106	11.872	1.969
1.8	11.891	11.665	11.453	1.848
1.9	11.440	11.265	11.080	1.671
2.0	11.043	10.899	10.746	1.429
2.1	10.692	10.449	10.445	0.046
2.2	10.379	10.179	10.160	0.189
2.3	10.101	9.934	9.881	0.532
2.4	9.851	9.712	9.610	1.062
2.5	9.626	9.494	9.347	1.568

Table 2.8: Scenario 2—$\lambda_X = 6$, $\mu_X = 2$, $S_X = 3$, and $S_{\{1,2,3\}} = 3$

$\mu_{\{1,2,3\}}$	g	g'	g^*	Δ
1.5	27.101	25.142	23.387	7.502
1.6	25.138	22.841	21.638	5.559
1.7	23.306	20.923	20.099	4.097
1.8	21.623	19.326	18.755	3.042
1.9	20.099	17.992	17.586	2.304
2.0	18.737	16.867	16.571	1.784
2.1	17.533	15.910	15.690	1.401
2.2	16.475	15.086	14.920	1.114
2.3	15.551	14.370	14.240	0.915
2.4	14.746	13.741	13.631	0.802
2.5	14.044	13.183	13.082	0.771

Table 2.9: Scenario 2—$\lambda_X = 7$, $\mu_X = 2$, $S_X = 3$, and $S_{\{1,2,3\}} = 3$

λ_X	μ_X	S_X	$\mu_{\{1,2,3\}}$	$S_{\{1,2,3\}}$	g	g'	g^*	Δ
6 5 4	2.5 2.0 1.5	2 2 2	2.0	3	13.186	11.562	11.314	2.187
12 10 8	5.0 4.0 3.0	2 2 2	4.0	3	13.187	11.564	11.313	2.215
24 20 16	10.0 8.0 6.0	2 2 2	8.0	3	13.190	11.567	11.310	2.273
12 10 8	2.5 2.0 1.5	4 4 4	2.0	6	18.625	17.789	17.562	1.290
24 20 16	5.0 4.0 3.0	4 4 4	4.0	6	18.628	17.792	17.559	1.326
24 20 16	2.5 2.0 1.5	8 8 8	2.0	12	31.925	31.380	30.991	1.253
30 20 10	2.5 2.0 1.5	15 15 15	2.0	15	28.924	27.996	27.256	2.714
30 20 20	2.5 2.0 1.5	20 20 20	2.0	20	35.290	34.829	32.015	8.788

Table 2.10: Scenario 2—scaling of the system

2.6 Application: A Controlled Polling System

The relative value function of the priority queue in Sect. 2.3.5 can be seen as a server assignment problem in which a fixed priority discipline is used, namely the preemptive priority resume policy. When the server can change at any instant, more freedom is introduced leading to lower average costs under the optimal policy. The question then becomes what the optimal policy is.

Let $c_1 > 0$ and $c_2 > 0$. In the case that the switching costs are identical to zero the optimal policy is equal to the well-known μc rule (see, e.g., [3]). This means that priority is given to a class-i customer based on the value of $\mu_i c_i$; priority is given to the queue with higher values of $\mu_i c_i$. When switching costs are greater than zero, the optimal policy is not known and a numerical method such as policy or value iteration has to be used. In this section we illustrate how to use one-step policy improvement for this problem.

In principle, one could apply a Bernoulli policy to the server, i.e., the server works a fraction α of the time on queue 1 and a fraction $1-\alpha$ on queue 2. However, in order to take the dependencies between the two queues into account in the initial policy, we can also use the priority policy, of which we denote the value function by V_p, of Sect. 2.3.5. Let $\lambda_1 = \lambda_2 = 1$, $\mu_1 = 6$, $\mu_2 = 3$, $c_1 = 2$, $c_2 = 1$, and $s_1 = s_2 = 2$. Note that with these parameters, the priority policy coincides with the μc rule. Define $\mu = \max\{\mu_1, \mu_2\}$. Then, define for some fixed x and y

$$z_{k,l} = s_k \mathbb{1}_{\{k \neq l\}} + c_1 x + c_2 y + \lambda_1 V_p(x+1,y,l) + \lambda_2 V_p(x,y+1,l) + \\ \mu_l V_p\big(((x,y)-e_l)^+, l\big) + (\mu - \mu_l) V_p(x,y,l)$$

as the expected bias if the server immediately switches from position k to position l and uses the preemptive priority rule afterwards. The one-step improved policy is simply the policy that minimizes for each (x,y,k) the expression $\min_a\{z_{k,a}\}$. Hence, the actions are defined by $a_{x,y,k} = \arg\min\{z_{k,a}\}$.

Table 2.11 shows the results for the one-step policy improvement. The average cost resulting from using a single step of policy iteration is 3.09895. This is a reduction of the costs by 14.6% as compared to using the μc rule where the average cost

is 3.62894. By using policy iteration to find the optimal policy the results hardly improve; the average cost is at lowest 3.09261. Surprisingly enough, the optimal policy is found in two steps of policy iteration. The fast convergence of the policy iteration algorithm is not coincidental; the average cost of the policies generated by policy iteration converge at least exponentially fast to the minimum cost, with the greatest improvement in the first few steps.

Iteration	Average cost	Comment
0	3.62894	μc rule
1	3.09895	One-step policy improvement
2	3.09261	Optimal policy

Table 2.11: Policy iteration results

References

1. E. Altman, Applications of Markov decision processes in communication networks: a survey, in *Handbook of Markov Decision Processes*, edited by E.A. Feinberg, A. Shwartz (Kluwer, Dordrecht, 2002)
2. S. Asmussen, O. Nerman, M. Olsson, Fitting phase type distributions via the EM algorithm. Scand. J. Stat. **23**, 419–441 (1996)
3. J.S. Baras, D.-J. Ma, A.M. Makowski, k competing queues with geometric service requirements and linear costs: the μc rule is always optimal. Syst. Control Lett. **6**, 173–180 (1985)
4. R. Bellman, Adaptive Control Processes: A Guided Tour (Princeton University Press, Princeton, NJ, 1961)
5. S. Bhulai, On the value function of the M/Cox(r)/1 queue. J. Appl. Probab. **43**(2), 363–376 (2006)
6. S. Bhulai, G.M. Koole, On the structure of value functions for threshold policies in queueing models. J. Appl. Probab. **40**, 613–622 (2003)
7. S. Bhulai, F.M. Spieksma, On the uniqueness of solutions to the Poisson equations for average cost Markov chains with unbounded cost functions. Math. Meth. Oper. Res. **58**(2), 221–236 (2003)
8. S. Bhulai, A.C. Brooms, F.M. Spieksma, On structural properties of the value function for an unbounded jump Markov process with an application to a processor-sharing retrial queue. Queueing Syst. **76**(4), 425–446 (2014)
9. P. Chevalier, N. Tabordon, R. Shumsky, Routing and staffing in large call centers with specialized and fully flexible servers. Working paper (2004)
10. E. Hyytiä, J. Virtamo, Dynamic routing and wavelength assignment using first policy iteration. Technical Report COST 257, Helsinki University of Technology (2000)
11. G.M. Koole, Stochastic Scheduling and Dynamic Programming. CWI Tract, vol. 113 (CWI, Amsterdam, 1995)

12. S.A. Lippman, Applying a new device in the optimization of exponential queueing systems. Oper. Res. **23**, 687–710 (1975)
13. R.A. Marie, Calculating equilibrium probabilities for $\lambda(m)/C_k/1/N$ queues, in *Proceedings of the International Symposium on Computer Performance Modeling* (1980)
14. R.E. Mickens, Difference Equations: Theory and Applications (Chapman & Hall, London, 1990)
15. J.M. Norman, Heuristic Procedures in Dynamic Programming (Manchester University Press, Manchester, 1972)
16. E.L. Örmeci, Dynamic admission control in a call center with one shared and two dedicated service facilities. IEEE Trans. Autom. Control **49**, 1157–1161 (2004)
17. T.J. Ott, K.R. Krishnan, Separable routing: a scheme for state-dependent routing of circuit switched telephone traffic. Ann. Oper. Res. **35**, 43–68 (1992)
18. M.L. Puterman, Markov Decision Processes: Discrete Stochastic Dynamic Programming (Wiley, New York, 1994)
19. H. Rummukainen, J. Virtamo, Polynomial cost approximations in markov decision theory based call admission control. IEEE/ACM Trans. Netw. **9**(6), 769–779 (2001)
20. S.A.E. Sassen, H.C. Tijms, R.D. Nobel, A heuristic rule for routing customers to parallel servers. Statistica Neerlandica **51**, 107–121 (1997)
21. R.F. Serfozo, An equivalence between continuous and discrete time Markov decision processes. Oper. Res. **27**, 616–620 (1979)
22. F.M. Spieksma, Geometrically ergodic Markov chains and the optimal control of queues. Ph.D. thesis, Leiden University (1990)
23. S. Stidham Jr., R.R. Weber, A survey of Markov decision models for control of networks of queues. Queueing Syst. **13**, 291–314 (1993)
24. H.C. Tijms, Stochastic Models: An Algorithmic Approach (Wiley, New York, 1994)
25. D. Towsley, R.H. Hwang, J.F. Kurose, MDP routing for multi-rate loss networks. Comput. Netw. ISDN **34**, 241–261 (2000)