On the value function of a priority queue with an application to a controlled polling model

Ger Koole
Vrije Universiteit
De Boelelaan 1081a
1081 HV Amsterdam
The Netherlands
koole@cs.vu.nl

Philippe Nain
INRIA
2004 Route des Lucioles
06902 Sophia Antipolis
France
nain@sophia.inria.fr

Published in: **QUESTA, Vol. 34, pp. 199-214, 2000**

Abstract

We give a closed-form expression for the discounted weighted queue length and switching costs of a two-class single-server queueing model under a preemptive priority rule. These expressions are used to do a single step of policy iteration in a polling model with a dynamically controlled switching rule, starting from the preemptive priority rule. Numerical experiments show that this leads to a nearly optimal policy.

Keywords: priority queues, policy iteration, μc rule, polling systems.

1 Introduction

The application of Dynamic Programming (DP) to real-world problems is hindered to a large extent by the enormous number of possible states and controls. This phenomenon is known as the *curse of dimensionality* and it renders direct computation of optimal policies in most practical models impossible. The recent publication of [2] has renewed the interest in *reinforcement learning*, which is a possible way to circumvent the curse of dimensionality. The main idea in this approach is to construct approximate representations of the optimal value function [2, Sec. 2]. This involves the selection of an *approximation architecture*, that is a certain functional form with free parameters. These parameters have then to be tuned so as to provide the best fit of the function to be approximated (the optimal value function in the DP setting). The process of tuning the parameters is referred to as *learning* (or training) in the literature. Once this is achieved various methods centered around the *Bellman's equation* may be used to compute the suboptimal control [2, Sec. 4]. However, in order to break the curse of dimensionality the approximate representation of the value function — known as the *scoring function* — must be described with few parameters. The choice of a "good" scoring function is therefore the key to success. It requires in general lots of experiments as well as a solid knowledge of the system at hand.

This paper is a first step toward the application of these ideas to queueing networks for which optimal controls are notoriously difficult to compute due to the size of the state spaces involved. Our main result is the derivation of the form of the scoring function for a given two-dimensional queueing model.

More specifically, the subject of study is a single-server system with two classes of customers. Customers arrive according to Poisson processes and require exponential service times. We consider a preemptive resume priority discipline, that gives the highest priority to class 1. This means that class 1 is served to exhaustion, and if the server is attending class 2, then as soon as a class-1 customer arrives the server switches at once to class 1. There are holding costs for each unit of time that a customer spends in the system; furthermore there are lump costs for every time the server changes class. We calculate in Section 2 the expected discounted holding and switching costs over an infinite horizon associated with this policy. The final closed form expression is a sum of terms that are linear, quadratic and exponential in both queue lengths.

This closed form expression is useful by itself; below we discuss an application. But it is also interesting to note that it is possible to explain all terms of the formula, mainly in terms of busy periods of single server queues. This knowledge gives us a means to guess the form or even the values of value functions for other queueing models that could later on be used as scoring functions. As such the present results have a high potential for further application. Next to the formal proof of the closed-form expression we give in Section 3 an extensive heuristic explanation of each term. In fact, we derived the formula for the holding costs using straightforward but tedious calculations. After having understood all the terms we were able to guess the terms concerning switching costs. Construction of a formal proof afterwards (see proof of Theorem 2.2) confirmed that our intuition was correct.

As explained earlier we believe that with the present calculations we made an important step towards the application of reinforcement learning type of methods to queueing systems in the sense that what we have learned might be used to select good scoring functions.

The results in Section 2 may also be used directly to control a polling model. Indeed, consider the model as described above but without the preemptive priority switching rule. Instead we look for the policy that minimizes the expected discounted sum of holding and switching costs. If the switching costs are zero then the policy that gives preemptive priority to the class with the highest product of holding cost rate and service rate is optimal (the so-called μc rule, see e.g. [1, 3, 7]); if the switching costs are positive then the μc rule is no longer optimal in general. The optimal policy cannot easily be characterized and can only be determined using an iterative procedure [5]. This has motivated us to look at easy ways to calculate approximations of the optimal policy. Using the μc rule as a starting point and its associated value function obtained in Section 2 as scoring function, and using a method of [8] that involves a *single step* of the *policy iteration algorithm*, we have found policies that are surprisingly close to the optimal one. We refer the reader to the numerical results at the end of the paper for further details.

2 Results

We consider a two-class M/M/1 queue under the preemptive resume priority discipline [4]. We assume that class-1 customers have priority over class-2 customers. We denote by $\lambda_i > 0$ (resp. $\mu_i > 0$) the arrival (resp. service) rate of class-i customers. We assume that the stability condition $\sum_{i=1}^{2} \lambda_i/\mu_i < 1$ holds.

There are holding costs (c_i for customers of class-i) and switching costs when the server switches from one class to the other. Let s_1 (resp. s_2) be the instantaneous cost incurred when the server switches from class-1 (resp. class-2) to class-2 (resp. class-1) customers.

Let $X(t)$ (resp. $Y(t)$) be the number of class-1 (resp. class-2) customers in the system at time t and let $S(t) \in \{1,2\}$ be the position of the server at time t ($S(t) = z$ if the server is attending class-z customers at time t). For any r.v. Z, we denote $\mathbf{E}_x[Z]$ (resp. $\mathbf{E}_{x,y}[Z]$, $\mathbf{E}_{x,y,z}[Z]$) the conditional expectation of Z given that $X(0) = x$ (resp. given that $X(0) = x$ and $Y(0) = y$, given that $X(0) = x$, $Y(0) = y$ and $Z(0) = z$).

Fix $\beta \in (0, \infty)$. Given that $X(0) = x$, $Y(0) = y$ and $S(0) = z$, the total discounted cost $V(x,y,z)$ incurred in $[0,\infty)$ is given by

$$V(x,y,z) = \mathbf{E}_{x,y,z}\left[\int_0^\infty e^{-\beta t}\,(c_1 X(t) + c_2 Y(t))\,dt + \sum_{n=1}^{\infty}\sum_{i=1}^{2} s_i e^{-\beta T_n^i}\right] \tag{1}$$

where T_n^1 (resp. T_n^2) denotes the nth switching time from class-1 (resp. class-2) to class-2 (resp. class-1) customers.

Before stating the main result of this paper, we recall below some known facts about the queueing system at hand. Since under the preemptive priority discipline the behavior of class-1 customers is not affected by that of class-2 customers, $\{X(t), t \geq 0\}$ is the queue-length process in an M/M/1 queue with arrival (resp. service) rate λ_1 (resp. μ_1). For this queue, the Laplace-Stieltjes Transform (LST) $z(\beta)$ of the busy period is obtained as the unique root in $(0,1)$ of the equation

$$\lambda_1 u^2 - (\lambda_1 + \mu_1 + \beta)u + \mu_1 = 0. \tag{2}$$

The *service completion time* of a class-2 customer is the time that elapses between his first entry in the server and his departure from the queue. Its LST $C(\beta) = \mathbf{E}[\exp(-\beta C)]$ is given by [4, formula (2.13), p. 86]

$$C(\beta) = \frac{\mu_2}{\mu_2 + \lambda_1(1 - z(\beta)) + \beta}. \tag{3}$$

Last, we introduce the LST $B(\beta)$ of the busy period in an M/G/1 queue with arrival rate λ_2 and LST of the service times $C(\beta)$. By using (3) and a standard result on the busy period in an M/G/1 queue (e.g. see [4, Eq. (2.15), p. 10]), we get that $B(\beta)$ is the unique root in $(0,1)$ of the equation

$$u - \frac{\mu_2}{\mu_2 + \lambda_1(1 - z(\lambda_2(1-u) + \beta)) + \lambda_2(1-u) + \beta} = 0. \tag{4}$$

If we define the busy period of class-2 customers as the time that elapses between the moment that a class-2 customer enters an empty system and the first time that a class-2 customer leaves an empty system behind, then the LST of the busy period of class-2 is given by $B(\beta)$.

An important role is played by the *optimality equation*. The term optimality equation comes from dynamic programming, which usually involves minimization as well. Here we apply it to a single fixed policy, the priority rule, but we stick to the terminology.

Lemma 2.1 (Optimality equation) *The value function $V(x,y,z)$ of the preemptive priority discipline is the unique solution of*

$$\begin{aligned}(\lambda_1 + \lambda_2 + \mu_1 + \beta)V(x,y,1) &= c_1 x + c_2 y + \lambda_1 V(x+1,y,1) + \lambda_2 V(x,y+1,1) \\ &\quad + \mu_1 V(x-1,y,1), \quad x > 0, y \geq 0 \qquad (5)\\ V(0,y,1) &= s_1 + V(0,y,2), \quad y > 0 \qquad (6)\end{aligned}$$

$$
\begin{aligned}
V(x,y,2) &= s_2 + V(x,y,1), \quad x>0, y\geq 0 && (7)\\
(\lambda_1+\lambda_2+\mu_2+\beta)V(0,y,2) &= c_2 y + \lambda_1 V(1,y,2) + \lambda_2 V(0,y+1,2) \\
&\quad +\mu_2 V(0,y-1,2), \quad y>0 && (8)\\
(\lambda_1+\lambda_2+\beta)V(0,0,z) &= \lambda_1 V(1,0,z) + \lambda_2 V(0,1,z), \quad z=1,2. && (9)
\end{aligned}
$$

Proof. The value function (5)-(9) has the form of Eq. (11.3.7) in [9], with the direct cost r and the sum of the switching costs (if applicable) and holding costs as in Eq. (11.3.3).

The existence and uniqueness follow from [9], Theorem 11.3.3. The conditions are satisfied if we take as function w for example $w((x,y,z)) = (c_1+c_2)((x+y)/\beta+(\lambda_1+\lambda_2+\mu_1+\mu_2)/\beta^2)+(s_1+s_2)(\lambda_1+\lambda_2+\mu_1+\mu_2+\beta)/\beta$ (which is in fact an upper bound to the value function $V(x,y,z)$). ∎

In a heuristical manner Lemma 2.1 can be derived by conditioning on the first event in the system. Note that delaying all costs by T units of time corresponds to discounting with a factor $\mathbf{E}[\exp(-\beta T)]$, which is exactly the LST of T in β. For example, in states $(x,y,1)$ with $x>0$, the time to the first event T is exponentially distributed with parameter $\lambda_1+\lambda_2+\mu_1$. Its LST is $(\lambda_1+\lambda_2+\mu_1)/(\lambda_1+\lambda_2+\mu_1+\beta)$. As this event occurs, it is with probability $\lambda_i/(\lambda_1+\lambda_2+\mu_1)$ an arrival of a class-i customer $(i=1,2)$ and with probability $\mu_1/(\lambda_1+\lambda_2+\mu_1)$ a departure of a class-1 customer. It can also be seen that

$$
\mathbf{E}_x\left[\int_0^T e^{-\beta t} c_1 x dt\right] = \frac{c_1 x}{\lambda_1+\lambda_2+\mu_1+\beta}.
$$

Putting this together gives

$$
\begin{aligned}
V(x,y,1) = \frac{c_1 x + c_2 y}{\lambda_1+\lambda_2+\mu_1+\beta} + \frac{\lambda_1+\lambda_2+\mu_1}{\lambda_1+\lambda_2+\mu_1+\beta}\Bigg(&\frac{\lambda_1}{\lambda_1+\lambda_2+\mu_1}V(x+1,y,1)\\
&+\frac{\lambda_2}{\lambda_1+\lambda_2+\mu_1}V(x,y+1,1) + \frac{\mu_1}{\lambda_1+\lambda_2+\mu_1}V(x-1,y,1)\Bigg),
\end{aligned}
$$

for $x>0$, $y\geq 0$, which is equivalent to (5). Relations (8) and (9) can be derived in a similar manner.

We now state our main result.

Theorem 2.2 (Value function) *The total discounted cost $V(x,y,z)$ is given by*

$$
\begin{aligned}
V(x,y,1) &= f(x,y) + (r_1+r_1')z(\beta)^x + (r_2+r_2')\,z(g(\beta))^x\,B(\beta)^y, \quad x\geq 0,\, y>0 && (10)\\
V(x,0,1) &= f(x,0) + (r_1+r_1')z(\beta)^x + (r_2+r_2')\,z(g(\beta))^x + r_3'\,z(\beta+\lambda_2)^x, x\geq 0 && (11)\\
V(x,y,2) &= s_2 + V(x,y,1), \quad x>0,\, y\geq 0 && (12)\\
V(0,y,2) &= f(0,y) + r_1 + r_1' - s_1 + (r_2+r_2')\,B(\beta)^y, \quad y\geq 0, && (13)
\end{aligned}
$$

with

$$
\begin{aligned}
f(x,y) &:= c_1\,\frac{\lambda_1-\mu_1}{\beta^2} + c_2\,\frac{\lambda_2}{\beta^2} + c_1\,\frac{x}{\beta} + c_2\,\frac{y}{\beta} && (14)\\
g(\beta) &:= \lambda_2(1-B(\beta)) + \beta && (15)
\end{aligned}
$$

$$r_1 := c_1 \frac{z(\beta)}{\beta(1-z(\beta))} - c_2 \frac{C(\beta)}{\beta(1-C(\beta))} \tag{16}$$

$$r_1' := \frac{(s_1+s_2)\lambda_1 + \beta s_1}{\lambda_1(1-z(\beta))+\beta} \tag{17}$$

$$r_2 := c_2 \frac{C(\beta)}{\beta(1-C(\beta))} \cdot \frac{\lambda_1(1-z(\beta))+\beta}{\lambda_1(1-z(g(\beta)))+g(\beta)} \tag{18}$$

$$r_2' := \frac{\lambda_1 z(\beta+\lambda_2)}{\lambda_1(1-z(g(\beta))+g(\beta)} r_3' \tag{19}$$

$$r_3' := \frac{\lambda_2 s_1 - \lambda_1 s_2}{\lambda_1+\lambda_2+\beta} - s_1. \tag{20}$$

Proof. The proof consists in checking that $V(x,y,z)$ satisfies the Dynamic Programming (DP) equation given by (5)-(9).

By using the definition of $V(x,y,z)$ in (10)-(13) it is easily seen that this mapping satisfies (6) and (7). Let us now focus on the remaining equations (5), (8) and (9).

(i) *Checking validity of (5)*

Assume first that $x>0$ and $y>0$. Introducing (10) into (5) yields, after dropping terms that appear on both sides of (5),

$$\begin{aligned}
&(\lambda_1+\lambda_2+\mu_1+\beta)\left[(r_1+r_1')z(\beta)^x + (r_2+r_2')z(g(\beta))^x B(\beta)^y\right] \\
&= (r_1+r_1')z(\beta)^x\left[\lambda_1 z(\beta)+\lambda_2+\mu_1/z(\beta)\right] + (r_2+r_2')z(g(\beta))^x B(\beta)^y \\
&\quad \times\left[\lambda_1 z(g(\beta))+\lambda_2 B(\beta)+\mu_1/z(g(\beta))\right]
\end{aligned}$$

or, equivalently,

$$\begin{aligned}
0 &= (r_1+r_1')z(\beta)^{x-1}\left[-\lambda_1 z(\beta)^2+(\lambda_1+\mu_1+\beta)z(\beta)-\mu_1\right] \\
&\quad +(r_2+r_2')z(g(\beta))^{x-1}B(\beta)^y\left[-\lambda_1 z(g(\beta))^2+(\lambda_1+\mu_1+g(\beta))z(g(\beta))-\mu_1\right].
\end{aligned}$$

This identity is indeed true as both terms between squared brackets are equal to 0 from the definition of $z(\cdot)$ (see (2)).

Assume now that $x>0$ and $y=0$. Using now (10) and (11) it is easily seen that (5) reduces to

$$\begin{aligned}
0 &= (r_1+r_1')z(\beta)^{x-1}\left[-\lambda_1 z(\beta)^2+(\lambda_1+\mu_1+\beta)z(\beta)-\mu_1\right] \\
&\quad +(r_2+r_2')z(g(\beta))^{x-1}\left[-\lambda_1 z(g(\beta))^2+(\lambda_1+\mu_1+g(\beta))z(g(\beta))-\mu_1\right] \\
&\quad +r_3' z(\beta+\lambda_2)^{x-1}\left[-\lambda_1 z(\beta+\lambda_2)^2+(\lambda_1+\mu_1+\lambda_2+\beta)z(\beta+\lambda_2)-\mu_1\right].
\end{aligned}$$

We again observe that this identity holds true as each term between squared brackets vanishes from the definition of $z(\cdot)$.

(ii) *Checking validity of (8)*

Replacing $V(x,y,2)$ in (8) by its value given in (12)-(13) and dropping terms that appear on both sides of the resulting equation gives

$$\begin{aligned}
0 &= -(r_1+r_1')(\lambda_1(1-z(\beta))+\beta)+\lambda_1(s_1+s_2)+s_1\beta+c_1\frac{\mu_1}{\beta}-c_2\frac{\mu_2}{\beta} \\
&\quad +(r_2+r_2')B(\beta)^{y-1}\left[\lambda_2 B(\beta)^2-(\mu_2+\lambda_1(1-z(g(\beta)))+\lambda_2+\beta)B(\beta)+\mu_2\right]. \quad (21)
\end{aligned}$$

The term between squared brackets is equal to 0 from the definition (4) of $B(\beta)$. This observation, together with the use of the definition of r_1 and r_1', allows us to rewrite (21) as

$$\begin{aligned} 0 &= \frac{c_1}{\beta(1-z(\beta))}\left[\lambda_1 z(\beta)^2 - (\lambda_1+\mu_1+\beta)z(\beta)+\mu_1\right] \\ &\quad -\frac{c_2}{\beta(1-C(\beta))}\left[\mu_2 - (\mu_2+\lambda_1(1-z(\beta))+\beta)C(\beta)\right]. \end{aligned} \tag{22}$$

The first term in the left-hand side of (22) is equal to 0 from the definition (2) of $z(\cdot)$; the second term too from the definition (3) of $C(\beta)$.

(iii) *Checking validity of (9)*

Assume first that $z = 1$. Using (10) and (11) we get that (9) is equivalent to

$$\begin{aligned} 0 &= \frac{c_1\mu_1}{\beta} - (r_1+r_1')(\lambda_1(1-z(\beta))+\beta) - (r_2+r_2')\left(\lambda_1(1-z(g(\beta)))+\lambda_2(1-B(\beta))+\beta\right) \\ &\quad -r_3'\left(\lambda_1+\lambda_2+\beta-\lambda_1 z(\beta+\lambda_2)\right). \end{aligned}$$

Using now the definition of r_1, r_1', r_2, r_2' and r_3' it is straightforward to check that indeed the above identity holds true.

The check in the case $z = 2$ is similar and is therefore omitted. ∎

3 Interpretation

In this section we interpret the result of Theorem 2.2. To do so, we study first a simple M/M/1 queue with rates λ_1 and μ_1. The objective is to find

$$W(x) = \mathbf{E}_x\left[\int_0^\infty e^{-\beta t}X(t)dt\right],$$

with $X(t)$ the queue length at t.

We can think of $X(t)$ as being constructed from 2 independent Poisson processes $\{N(t),\, t \geq 0\}$ and $\{M(t),\, t \geq 0\}$ with rates λ_1 and μ_1, respectively. If a departure occurs while $X(t) = 0$ then nothing happens; therefore M can be seen as the *potential* departure process. Let $L(t) = M(t) - (X(0) + N(t) - X(t))$, the number of virtual departures up to t. Thus

$$W(x) = \mathbf{E}_x\left[\int_0^\infty e^{-\beta t}\left(x+N(t)-M(t)\right)dt\right] + \mathbf{E}_x\left[\int_0^\infty e^{-\beta t}\,L(t)\,dt\right].$$

Define $T_x := \inf\{t > 0 : X(t) = 0\}$, the time until the system empties for the first time. It is easily understood that T_x is the sum of x independent busy periods of the M/M/1 queue. Therefore $\mathbf{E}[\exp(-\beta T_x)] = z(\beta)^x$. Furthermore, $X(t)$ is independent of the behavior of the queue from T_x on. Thus we can assume that as soon as 0 is reached a new queue with initial state 0 is started, with processes $\hat{N}$, $\hat{M}$, $\hat{X}$, and $\hat{L}$, with the same law, but independent of N, M, X, and L. Then

$$\begin{aligned} W(x) &= \mathbf{E}_x\left[\int_0^\infty e^{-\beta t}\left(x+N(t)-M(t)\right)dt\right] + \mathbf{E}_x\left[\int_{T_x}^\infty e^{-\beta t}L(t)\,dt\right] \\ &= \frac{x}{\beta} + \frac{\lambda_1-\mu_1}{\beta^2} + z(\beta)^x\,\mathbf{E}_0\left[\int_0^\infty e^{-\beta t}\left(\hat{X}(t)-\hat{N}(t)+\hat{M}(t)\right)dt\right] \\ &= \frac{x}{\beta} + \frac{\lambda_1-\mu_1}{\beta^2} + z(\beta)^x\left[W(0) - \frac{\lambda_1-\mu_1}{\beta^2}\right], \quad x \geq 0. \end{aligned} \tag{23}$$

To compute $W(0)$ we condition on the first event:

$$W(0) = \frac{\lambda_1}{\lambda_1 + \beta} W(1).$$

Using (23) for $x = 1$ it easily follows that

$$W(0) = \frac{\lambda_1 - \mu_1}{\beta^2} + \frac{\mu_1}{\beta(\lambda_1(1 - z(\beta)) + \beta)} = \frac{\lambda_1 - \mu_1}{\beta^2} + \frac{z(\beta)}{\beta(1 - z(\beta))},$$

the last equality following from the definition of $z(\cdot)$, as given in equation (2). In conclusion,

$$W(x) = \frac{x}{\beta} + \frac{\lambda_1 - \mu_1}{\beta^2} + z(\beta)^x \frac{z(\beta)}{\beta(1 - z(\beta))}. \tag{24}$$

The previous analysis taught us a methodology for constructing the value function for the preemptive priority model. The following key steps in the analysis of the M/M/1 queue will appear repeatedly in the sequel:

(i) The first terms are obtained by ignoring the fact that at some point in time some boundary ($x = 0$ here) is reached. These terms consist of terms for the initially available customers, new arrivals and departures (i.e., x/β, λ_1/β^2 and $-\mu_1/\beta^2$, respectively);

(ii) There is a correction term for reaching the boundary, which is the product of the LST of the time to reach the boundary (i.e. $z(\beta)^x$) and of a coefficient (see (23)) that consists of the value function at the boundary (i.e., $W(0)$) minus the first terms starting from the boundary (i.e., $(\lambda_1 - \mu_1)/\beta^2$). [Note that the latter term may be chosen so that both sides of (23) are equal on the boundary. This observation will be used later on.]

We now interpret the form of the value function $V(x, y, z)$ given in Theorem 2.2. To this end, we split the right-hand side of (1) in three terms:

$$V(x, y, z) = V_1(x) + V_2(x, y) + V_3(x, y, z) \tag{25}$$

with

$$\begin{aligned} V_1(x) &:= c_1 \, \mathbf{E}_x \left[\int_0^\infty e^{-\beta t} \, X(t) \, dt \right] \\ V_2(x, y) &:= c_2 \, \mathbf{E}_{x,y} \left[\int_0^\infty e^{-\beta t} \, Y(t) \, dt \right] \\ V_3(x, y, z) &:= \mathbf{E}_{x,y,z} \left[\sum_{n=1}^{\infty} \sum_{i=1}^{2} s_i e^{-\beta T_n^i} \right]. \end{aligned}$$

Because the first customer class is served with preemptive priority, $\{X(t),\, t \geq 0\}$ behaves as the queue length process in an M/M/1 queue and thus $V_1(x) = W(x)$ as given in (24). It can be checked (set $c_2 = s_1 = s_2 = 0$ in (10)-(13)) that Theorem 2.2 agrees with (24).

With the analysis for the M/M/1 queue in mind and, in particular, the interpretation in (i)-(ii), we now formulate an "educated guess" for $V_2(x, y)$.

If there were always class-1 customers in the system, then the total holding cost incurred by the y class-2 customers in the system at time 0 and by all class-2 customers arrived in $(0, \infty)$ would be $c_2 y/\beta + c_2 \lambda_2/\beta^2$.

Recall that there are no departures of class-2 customers in $[0, T_x)$ from both the definition of the preemptive priority rule and of T_x. Now assume that there are always class-2 customers in the system, i.e., we ignore the fact that we reach the x-axis at some point in time. In Section 2 we defined C as the LST of the service completion time of a class-2 customer, including interruptions by class-1 customers. Thus the nth departure of a class-2 customer occurs after a time that has LST $z(\beta)^x C(\beta)^n$, which results in a reduction in cost of $c_2 z(\beta)^x C(\beta)^n/\beta$. Summing this for all $n \geq 1$ gives

$$c_2 \frac{z(\beta)^x}{\beta} \left(C(\beta) + C(\beta)^2 + \cdots \right) = c_2 \, z(\beta)^x \frac{C(\beta)}{\beta \, (1 - C(\beta))}.$$

Given $X(0) = x$ and $Y(0) = y$, introduce $T_{x,y} := \inf\{t \geq 0 : X(t) = Y(t) = 0\}$ and set $\gamma_{x,y}(\beta) := \mathbf{E}[\exp(-\beta T_{x,y})]$. In words, $T_{x,y}$ is the first time that the system is empty. Because of the definition of B in Section 2 we find that $\gamma_{0,y}(\beta) = B(\beta)^y$. Compared to C, B is a class-2 busy period, thus it takes also into account the arrivals of class-2 during a service time. Let us now determine $\gamma_{1,0}(\beta)$. By conditioning on the first event to occur after time 0 we find the equation

$$\gamma_{1,0}(\beta) = \frac{1}{\lambda_1 + \lambda_2 + \mu_1 + \beta} \left(\lambda_1 \gamma_{1,0}(\beta)^2 + \lambda_2 B(\beta) \gamma_{1,0}(\beta) + \mu_1 \right),$$

whose solution is given by $\gamma_{1,0}(\beta) = z(\beta + \lambda_2(1 - B(\beta)))$, by (2). In conclusion

$$\gamma_{x,y}(\beta) = z(\beta + \lambda_2(1 - B(\beta)))^x \, B(\beta)^y. \tag{26}$$

Hence, the guess for $V_2(x, y)$ is therefore the following: for $x, y \geq 0$,

$$V_2(x,y) = c_2 \frac{y}{\beta} + c_2 \frac{\lambda_2}{\beta^2} - c_2 \, z(\beta)^x \frac{C(\beta)}{\beta \, (1 - C(\beta))} + \gamma_{x,y}(\beta) \left(V_2(0,0) - \left(\frac{c_2 \, \lambda_2}{\beta^2} - \frac{c_2 \, C(\beta)}{\beta(1 - C(\beta))} \right) \right), \tag{27}$$

the last term as to make sure that both sides of (27) coincide for $x = y = 0$.

It remains to evaluate $V_2(0, 0)$. For this, observe that

$$V_2(0,0) = \frac{\lambda_1}{\lambda_1 + \lambda_2 + \beta} \, V_2(1,0) + \frac{\lambda_2}{\lambda_1 + \lambda_2 + \beta} \, V_2(0,1)$$

which yields, with the help of (27),

$$V_2(0,0) = \frac{c_2 \, C(\beta)(\lambda_1(1 - z(\beta)) + \beta)}{\beta(1 - C(\beta)) \, (\lambda_1(1 - z(g(\beta))) + \lambda_2(1 - B(\beta)) + \beta)} + \frac{c_2 \lambda_2}{\beta^2} - \frac{c_2 C(\beta)}{\beta(1 - C(\beta))}.$$

Introducing this value of $V_2(0, 0)$ in (28) finally gives

$$\begin{aligned} V_2(x,y) \;\; = \;\; & c_2 \frac{\lambda_2}{\beta^2} + c_2 \frac{y}{\beta} - c_2 z(\beta)^x \, \frac{C(\beta)}{\beta(1 - C(\beta))} + c_2 z(\beta + \lambda_2(1 - B(\beta)))^x \, B(\beta)^y \\ & \times \left(\frac{C(\beta)(\lambda_1(1 - z(\beta)) + \beta)}{\beta(1 - C(\beta)) \, (\lambda_1(1 - z(g(\beta))) + \lambda_2(1 - B(\beta)) + \beta)} \right). \end{aligned} \tag{28}$$

Setting $c_1 = s_1 = s_2 = 0$ in Theorem (2.2) shows that this guess for $V_2(x, y)$ is indeed correct.

It is worth pointing out that $V_1(x)$ and $V_2(x, y)$ could have be computed by using standard analytical results on priority queues (see e.g. [4]). Besides the fact that such an approach yields lengthy calculation, it also does not give much insight on the form of these functions, as opposed to the method used here.

We are now going to use what we have learned so far to make a guess for $V_3(x,y,z)$, a quantity that cannot easily be derived by a direct method.

To begin with, observe that $V_3(x,y,z)$ need only to be computed for $x \geq 0$, $y > 0$, $z = 1$, for $x > 0$, $y = 0$, $z = 2$ and for $x = y = 0$, $z = 1, 2$. Indeed, by definition of the cost structure we already know that

$$V_3(0,y,2) = -s_1 + V_3(0,y,1), \;\; y > 0, \tag{29}$$

and

$$V_3(x,y,2) = s_2 + V_3(x,y,1), \;\; x > 0, \;\; y \geq 0. \tag{30}$$

Assume first that $x \geq 0$, $y > 0$ and $z = 1$. If there were always class-2 customers in the system, then the total switching cost incurred in $[T_{x,y}, \infty)$ would be

$$z(\beta)^x \Big(s_1 + \frac{\lambda_1}{\lambda_1 + \beta} \Big(s_2 + z(\beta) \Big(s_1 + \frac{\lambda_1}{\lambda_1 + \beta} \Big(s_2 + \cdots \Big) \Big) \Big) \Big) = z(\beta)^x \, r_1' \tag{31}$$

where r_1' is defined in (17). As $y > 0$, we always arrive at $x = y = 0$ at the service completion of a class-2 customer. Therefore the LST for reaching the origin is multiplied by $V_3(0,0,2)$ plus a correction term. This term does not follow directly from $V_3(x,y,z)$ for $x = y = 0$, as we assumed that $y > 0$, and thus the formula need not be valid at the origin. Therefore we calculate the correction term.

If there were always class-2 customers in $[T_{x,y}, \infty)$ (recall that $T_{x,y}$ is the first time such that $Y(t) = 0$, with LST $\gamma_{x,y}(\cdot)$), then the total switching cost would be

$$\gamma_{x,y}(\beta) \frac{\lambda_1}{\lambda_1 + \beta} \Big(s_2 + z(\beta) \Big(s_1 + \frac{\lambda_1}{\lambda_1 + \beta} \Big(s_2 + \cdots \Big) \Big) \Big) = \gamma_{x,y}\,(\beta)(r_1' - s_1). \tag{32}$$

The last term has to subtracted, thus the following guess is made for $V_3(x,y,1)$ when $x \geq 0$, $y > 0$:

$$V_3(x,y,1) = z(\beta)^x \, r_1' + \gamma_{x,y}(\beta) \, (V_3(0,0,2) + s_1 - r_1'). \tag{33}$$

It remains to handle the case when $x > 0$, $y = 0$ and $z = 1$. This is the most tricky case as state $(0,0)$ can either be reached from the y-axis (i.e. from $(0,1)$) or from the x-axis (i.e. from $(1,0)$) depending on whether or not at least one class-2 customer enters the system in $[0, T_x)$.

Thus another correction term is called for. The leading term, which is the discounted probability of reaching the origin without class-2 arrivals, is given by

$$\mathbf{E}_x \left[\int_0^\infty e^{-\beta t} \mathrm{I}\{\text{no class-2 arrivals up to } t\} T_x(dt) \right] = \int_0^\infty e^{-(\beta+\lambda_2)t} T_x(dt) = z(\beta + \lambda_2)^x.$$

The coefficient is $V_3(0,0,1)$ minus a correction term; this correction term is equal to $s_1 + V_3(0,0,2)$, because at arrival at the boundary no switch to class-2 is necessary. In conclusion,

$$\begin{aligned} V_3(x,0,1) \;\; = \;\; & r_1' \, z(\beta)^x + \big(V_3(0,0,2) + s_1 - r_1' \big) \, z(g(\beta))^x \\ & + z(\beta + \lambda_2)^x \, \left(V_3(0,0,1) - V_3(0,0,2) - s_1 \right). \end{aligned} \tag{34}$$

It remains to evaluate $V_3(0,0,z)$ for $z = 1, 2$. This is done with the identities

$$V_3(0,0,z) = \frac{\lambda_1}{\lambda_1 + \lambda_2 + \beta} \, V_3(1,0,z) + \frac{\lambda_2}{\lambda_1 + \lambda_2 + \beta} \, V_3(0,1,z), \quad z = 1, 2$$

together with relations (29), (30), (33), (34) and the definition of r_1'. We finally find

$$\begin{aligned} V_3(x,y,1) &= r_1' z(\beta)^x + r_2' \gamma_{x,y}(\beta), \quad x \geq 0, y > 0 \\ V_3(0,y,2) &= r_1' - s_1 + r_2' B(\beta)^y, \quad y \geq 0 \\ V_3(x,0,1) &= r_1' z(\beta)^x + r_2' \gamma_{x,y}(\beta) + r_3' z(\beta+\lambda_2)^x, \quad x \geq 0 \end{aligned} \tag{35}$$

where r_2' and r_3' are defined in (19) and (20), respectively.

Setting $c_1 = c_2 = 0$ in Theorem 2.2 we indeed find the value for $V_3(x,y,z)$ given above.

4 Application

In this section we will use the result in Theorem 2.2 to control a polling model. This model is the same as the model that we studied in Section 2, except for the fact that the server is now allowed to switch class each time an event in the system occurs. To be able to introduce the different policies in a concise way, we introduce the following terminology. Set $\mu = \max\{\mu_1, \mu_2\}$ and e_a the ath unity vector, v^+ means $\max\{v, 0\}$, componentwise if v and 0 are vectors. Define $\gamma = \lambda_1 + \lambda_2 + \mu + \beta$. With a slight abuse of terminology, customers of class-i will sometimes be referred to as customers in queue i.

For sets $A(x,y,z) \subset \{1,2\}$, $A(x,y,z) \neq \emptyset$, define the DP operator T^A for some $f : \mathbb{N}^2 \times \{1,2\} \to \mathbb{R}$ as follows:

$$\begin{aligned} T^A f(x,y,z) &= \frac{c_1 x + c_2 y}{\gamma} + \min_{a \in A(x,y,z)} \Big\{ s_z \mathrm{I}\{z \neq a\} + \frac{\lambda_1}{\gamma} f(x+1,y,a) \\ &\quad + \frac{\lambda_2}{\gamma} f(x,y+1,a) + \frac{\mu_a}{\gamma} f(((x,y)-e_a)^+, a) + \frac{\mu - \mu_a}{\gamma} f(x,y,a) \Big\}. \end{aligned} \tag{36}$$

This leads to the following optimality equation that generalizes (5)-(9):

Lemma 4.1 (Optimality equation) *The value function $V^A(x,y,z)$ for some choice of action sets $A(x,y,z)$ is the unique solution of*

$$V^A(x,y,z) = T^A V^A(x,y,z). \tag{37}$$

The minimizing actions constitute the minimizing policy for the specified action sets. The value function V^A can also be obtained by value iteration, i.e.,

$$V^A(x,y,z) = \lim_{n \to \infty} (T^A)^n 0. \tag{38}$$

Proof. The proof is the same as that of Lemma 2.1; the conditions in [9], Theorem 11.3.3, also guarantee the convergence of value iteration. ∎

In fact, if we choose $A(x,y,z) = 1$ if $x > 0$, $A(0,y,z) = 2$ if $y > 0$, and $A(0,0,z) = z$ then (37) corresponds to giving preemptive priority to queue 1. For later comparison this policy is depicted in Table 1 for $x, y \leq 10$. A "1" (resp. "2") means switching to 1 (resp. 2) if the server is not already at the first (resp. second) queue; a "." means that the server should stay where it is. Using Theorem 2.2 we can compute the expected discounted cost in any state. For instance, $V(2,2,1) = 77.773347$ with the parameters given in Table 2; the holding costs account for about one third (26.147901) of the total cost, the rest being switching costs.

Note furthermore that the holding costs are at the lowest possible level: it is known (see e.g. [3]) that for the model without switching costs the policy that gives preemptive priority to the queue with highest value of $\mu_i c_i$ (the μc rule) is optimal. As $\mu_1 c_1 > \mu_2 c_2$ the μc rule corresponds in the current case to serving queue 1 with preemptive priority.

$y=10$	2	1	1	1	1	1	1	1	1	1	1
9	2	1	1	1	1	1	1	1	1	1	1
8	2	1	1	1	1	1	1	1	1	1	1
7	2	1	1	1	1	1	1	1	1	1	1
6	2	1	1	1	1	1	1	1	1	1	1
5	2	1	1	1	1	1	1	1	1	1	1
4	2	1	1	1	1	1	1	1	1	1	1
3	2	1	1	1	1	1	1	1	1	1	1
2	2	1	1	1	1	1	1	1	1	1	1
1	2	1	1	1	1	1	1	1	1	1	1
0	·	1	1	1	1	1	1	1	1	1	1
	$x=0$	1	2	3	4	5	6	7	8	9	10

Table 1. Preemptive priority rule, priority for queue 1.

We are interested in determining the optimal policy for the case that $A(x,y,z) = \{1,2\}$ for all x, y, and z. For this choice of the action sets, denote T^A by T^*. Deriving the optimal value function V^* in this case is not easy; it is not expected that there exists a derivable closed-form expression for it. The reason for this lies in the complexity of the optimal policy (see Table 2). The only way to determine the optimal policy is by iteration. We executed the value iteration procedure (38) for a state space truncated at a sufficiently high level. In Table 2 we plot the optimal policy for a certain choice of the parameters.

$y=10$	2	·	1	1	1	1	1	1	1	1	1
9	2	·	1	1	1	1	1	1	1	1	1
8	2	·	1	1	1	1	1	1	1	1	1
7	2	·	1	1	1	1	1	1	1	1	1
6	2	·	1	1	1	1	1	1	1	1	1
5	2	·	1	1	1	1	1	1	1	1	1
4	2	·	1	1	1	1	1	1	1	1	1
3	2	·	·	1	1	1	1	1	1	1	1
2	2	·	·	1	1	1	1	1	1	1	1
1	·	·	·	·	1	1	1	1	1	1	1
0	·	1	1	1	1	1	1	1	1	1	1
	$x=0$	1	2	3	4	5	6	7	8	9	10

Table 2. Optimal switching policy for $\lambda_1 = \lambda_2 = 1$, $\mu_1 = 6$, $\mu_2 = 3$, $c_1 = 2$, $c_2 = 1$, $s_1 = s_2 = 2$, $\beta = 0.05$. $V^*(2,2,1) = 65.416897$.

The total cost for the optimal policy, starting in state $(2,2,1)$, is 65.416897. If we compare the optimal policy with the μc rule (Table 1), we see several striking differences. Of course the optimal policy switches less frequently than under the μc rule since there must be at least two class-1 customers in the system before it is optimal to serve these customers. The threshold level is not the same for all y: if y is close to 0, then the threshold level is higher than for larger values of y. This has an intuitive explanation: if the server leaves queue 2 for queue 1 while leaving customers behind, then it has to return to queue 2 at some point to serve the remaining customers. To avoid this it is cost-efficient to empty queue 2 if there are few customers in it before switching to queue 1. Finally, the optimal policy is not work-conserving: in state $(0,1,1)$ it is optimal to stay at server 1.

In [5] an approximating policy was constructed that was simple and reasonably good, by taking a work-conserving policy with the same threshold level for all states with $y > 0$. The resulting policy for the current case is depicted in Table 3. Using value iteration with the right sets A we computed the value function V', e.g. $V'(2,2,1) = 68.137829$. Compared to the μc rule it is already close to the optimal policy. However, we can find an even better policy, starting from the μc rule.

$y=10$	2	·	1	1	1	1	1	1	1	1	1
9	2	·	1	1	1	1	1	1	1	1	1
8	2	·	1	1	1	1	1	1	1	1	1
7	2	·	1	1	1	1	1	1	1	1	1
6	2	·	1	1	1	1	1	1	1	1	1
5	2	·	1	1	1	1	1	1	1	1	1
4	2	·	1	1	1	1	1	1	1	1	1
3	2	·	1	1	1	1	1	1	1	1	1
2	2	·	1	1	1	1	1	1	1	1	1
1	2	·	1	1	1	1	1	1	1	1	1
0	·	1	1	1	1	1	1	1	1	1	1
	$x=0$	1	2	3	4	5	6	7	8	9	10

Table 3. Threshold policy for $\lambda_1 = \lambda_2 = 1$, $\mu_1 = 6$, $\mu_2 = 3$, $c_1 = 2$, $c_2 = 1$, $s_1 = s_2 = 2$, $\beta = 0.05$. $V'(2,2,1) = 68.137829$.

Consider V, the value function of the preemptive priority rule. Now for each x, y, and z determine $A^*(x,y,z)$, the minimizing action in $T^*V(x,y,z)$. The resulting policy is shown in Table 4. Note how well this policy follows the behavior of the optimal policy close to the origin; only for high values of y the threshold value is different for the current choice of parameters. Using value iteration to compute the value V^a of this approximation policy we found $V^a(2,2,1) = 65.497223$, very close to the performance of the optimal policy.

$y=10$	2	·	·	1	1	1	1	1	1	1	1
9	2	·	·	1	1	1	1	1	1	1	1
8	2	·	·	1	1	1	1	1	1	1	1
7	2	·	·	1	1	1	1	1	1	1	1
6	2	·	·	1	1	1	1	1	1	1	1
5	2	·	·	1	1	1	1	1	1	1	1
4	2	·	·	1	1	1	1	1	1	1	1
3	2	·	·	1	1	1	1	1	1	1	1
2	2	·	·	1	1	1	1	1	1	1	1
1	·	·	·	·	1	1	1	1	1	1	1
0	·	1	1	1	1	1	1	1	1	1	1
	$x=0$	1	2	3	4	5	6	7	8	9	10

Table 4. One-step improvement for $\lambda_1 = \lambda_2 = 1$, $\mu_1 = 6$, $\mu_2 = 3$, $c_1 = 2$, $c_2 = 1$, $s_1 = s_2 = 2$, $\beta = 0.05$. $V^a(2,2,1) = 65.497223$.

To compute the value of the approximating policy we used the value iteration scheme (38), which is an iterative procedure, and therefore numerically demanding. Computing the value of the optimal policy (and by doing so we obtain the optimal policy) does hardly need more computer time than computing the value of a fixed policy. Thus computing the value of the approximating policy is about as difficult as computing the optimal policy. However, note that to determine the approximating policy itself (and not its value), no iterative procedure was needed: starting from the value of the preemptive priority rule the policy was determined by executing the minimization in (36) once. Thus we derived a procedure to obtain nearly

optimal policies that demands hardly any computational effort, as opposed to a numerically demanding iterative procedure for obtaining the optimal policy.

This approximation procedure can be seen as a single step of *policy iteration* (see e.g. [9], Section 6.4 and its application to continuous time models in Section 11.3.4) with a policy for which the value function is known. This idea is not new: it has already been applied to a simple telecommunication network in [8] and to routing to parallel queues in [10, 6]. In contrast with the current work [8, 10, 6] deal with average cost models and initial policies for which the different state components behave independently.

5 Concluding remarks

Using a single step of the policy improvement algorithm we were able to derive a nearly optimal policy for a simple polling model. To do so, we computed the value function associated with a given policy (the μc rule) and entered this value in the algorithm. In this process we gain some insights on what a good scoring function might look like for related queueing systems even though we might not be able to derive it explicitly. If this is the case, then the missing coefficients could be obtained by combining learning procedures (typically simulations) and estimation techniques (gradient methods). Ongoing research in this direction bear on the same model but with more than 2 classes as well as on some routing problems in queues. Work is also being conducted on average cost models.

References

[1] J. S. Baras, D.-M. Ma and A. M. Makowski. K competing queues with geometric requirements and linear costs: the μc-rule is always optimal. *Systems and Control Letters*, 6:173–180, 1985.

[2] D. P. Bertsekas and J. N. Tsitsiklis. *Neuro-Dynamic Programming.* Athena Scientific, 1996.

[3] C. Buyukkoc, P. Varaiya, and J. Walrand. The $c\mu$ rule revisited. *Advances in Applied Probability*, 17:237–238, 1985.

[4] N. K. Jaiswal. *Priority Queues.* Academic Press, New York, 1968.

[5] G.M. Koole. Assigning a single server to inhomogeneous queues with switching costs. *Theoretical Computer Science*, 182:203–216, 1997.

[6] G.M. Koole. The deviation matrix of the $M/M/1/\infty$ and $M/M/1/N$ queue, with applications to controlled queueing models. In *Proceedings of the 37th IEEE Conference on Decision and Control*, pages 56–59, 1998.

[7] P. Nain and D. Towsley. Optimal scheduling in a machine with stochastic varying processing rate. *IEEE Transactions on Automatic Control*, 39:1853–1855, 1994.

[8] T. J. Ott and K. R. Krishnan. Separable routing: A scheme for state-dependent routing of circuit switched telephone traffic. *Annals of Operations Research*, 35:43–68, 1992.

[9] M. L. Puterman. *Markov Decision Processes.* Wiley, 1994.

[10] S. A. E. Sassen, H. C. Tijms, and R. D. Nobel. A heuristic rule for routing customers to parallel servers. *Statistica Neerlandica*, 51:107–121, 1997.