Scheduling in polling systems

Adam Wierman[a,b] Erik M.M. Winands[c,d] Onno J. Boxma[b,d]

[a] *Computer Science Department, California Institute of Technology, 1200 E. California Boulevard, MC 256-80, Pasadena, CA 91125, USA.*

[b] *EURANDOM, P.O. Box 513, 5600 MB Eindhoven, The Netherlands*

[c] *Department of Technology Management, Technische Universiteit Eindhoven, P.O. Box 513, 5600 MB Eindhoven, The Netherlands*

[d] *Department of Mathematics and Computer Science, Technische Universiteit Eindhoven, P.O. Box 513, 5600 MB Eindhoven, The Netherlands*

Abstract

We present a simple *mean value analysis* (MVA) framework for analyzing the effect of scheduling within queues in classical asymmetric polling systems with gated or exhaustive service. Scheduling in polling systems finds many applications in computer and communication systems. Our framework leads not only to unification but also to extension of the literature studying scheduling in polling systems. It illustrates that a large class of scheduling policies behaves similarly in the exhaustive polling model and the standard M/GI/1 model, whereas scheduling policies in the gated polling model behave very differently than in an M/GI/1.

Key words: polling systems, scheduling policies, exhaustive service, gated service.

1 Introduction

Scheduling is a common mechanism for improving system performance without purchasing additional resources. Traditionally, simple scheduling policies such as First-Come-First-Served (FCFS) and Processor-Sharing (PS), which shares the service capacity equally among all jobs in the system, have been applied most frequently, and thus dominate the queueing literature. However, recently, system designs in a variety of application domains have begun to use policies that give priority to jobs with small service demands in order to reduce the mean response time (sojourn time) and mean queue length. For instance, variants of Shortest-Remaining-Processing-Time (SRPT)

Email addresses: adamw@caltech.edu (Adam Wierman), e.m.m.winands@tue.nl (Erik M.M. Winands), boxma@win.tue.nl (Onno J. Boxma).

and Foreground-Background (FB) have been applied in many computer applications, e.g. web servers [8,17] and routers [14,15]. This growing focus on priority-based policies has led to an increasing number of theoretical studies of such policies, e.g. [1,27,28] and the references therein. However, almost all theoretical studies of priority-based policies are performed in simple settings such as the M/GI/1 and G/GI/1. Our goal is to explore the impact of priority-based scheduling in a more complex queueing model: *polling systems.*

A polling system consists of a single server that polls a number of queues in a fixed order. Polling systems were first introduced in the late 1950s by Mack et al. [12,13] to model a patrolling repairman. Since the 1950s, polling systems have been used to model a wide range of applications in computer, communication, production, transportation, and maintenance systems. The ubiquitous nature of polling systems has meant that they have received an enormous amount of attention in the queueing community. Extensive surveys on polling systems and their applications may be found in [11,20,21,22,25].

Within a polling system there are a number of design decisions that the system operator needs to make. The system operator must decide (i) the order in which to serve the queues, (ii) how many customers to serve during each visit to a queue, and (iii) the order in which customers *within* each queue are served. The first decision can either be static, i.e. the polling order remains unchanged in the course of operation (see, e.g., [3,4]), or dynamic, i.e. the polling order changes over time (see, e.g., [9,31]). For the second decision a plethora of strategies have been proposed and analyzed, with most work focusing on the exhaustive and gated disciplines. Exhaustive service means that a queue must be empty before the server moves on, whereas in the case of gated service only those customers in the queue at the polling start are served. *The focus of the current paper is on the third scheduling decision: how to schedule jobs within each queue.* As a result, we will limit discussion to the most common configurations for the first two decisions: cyclic service order and exhaustive/gated service.

Although both the number of papers analyzing polling systems and the number of papers analyzing scheduling policies are impressive, the combination of the two has received very little attention. There are only a few exceptions where the effect of priority-based policies is studied in polling systems, for example, [6,19,23,24]. However, the results attained for such priority-based policies are mostly limited to pseudo-conservation laws and approximations that are exact only in special cases, e.g. symmetric polling systems.

Despite the lack of work analyzing scheduling in polling systems, there are many real-world examples where applications may benefit from scheduling in a non-FCFS manner. For example, in the computer science community polling models have recently been used to study the Bluetooth and 802.11 protocols as well as scheduling policies at routers and I/O subsystems in web servers. In many of these settings, the workloads are known to have high variability, and thus using a policy other than FCFS within the queues is appealing. Further, in these applications it is often desirable to give different requests different priority levels in order to provide differentiated service. Outside of computer systems, non-FCFS scheduling is used in polling systems in the area of production-inventory control. Specifically, it is used in the *stochastic economic lot scheduling problem* (SELSP), where multiple standardized products have to be produced on a single machine with significant setup times (see [29] for a survey on SELSP). In the SELSP, scheduling within the queues is necessary due to orders for the same product being placed by customers with different priority levels. Thus, the lack of research on scheduling in polling systems is not due to a paucity of applications. Rather, we believe it is primarily due to the following factors:

(1) It is natural to think that the impact of within-queue scheduling is small because it only influences the system performance locally, leaving the amount of time spent *outside* the targeted queue unaffected.
(2) The analysis of polling systems is difficult even in the simple case of FCFS scheduling: for FCFS explicit closed-form expressions for the mean delays are in general not known. Instead, the mean delay is expressed as the solution of a set of linear equations.

In this paper, we illustrate that the impact on mean response time from scheduling within a queue of a polling system can be dramatic. Further, in order to illustrate the impact of scheduling, we provide a unified analytic framework for studying scheduling in polling systems. This *mean value analysis* (MVA) framework allows the analysis of a variety of scheduling policies (many for the first time) in classical asymmetric polling systems with either gated or exhaustive service. Our framework illustrates a striking dichotomy between the impact of scheduling policies in exhaustive polling systems and gated polling systems. This reveals itself in the fact that a large class of scheduling policies behaves the same in exhaustive polling models as they do in the standard M/GI/1, whereas scheduling in gated polling models has a very different effect than in the M/GI/1.

The rest of the paper is structured as follows. In Section 2 we present a detailed model description. Sections 3 and 4 deal with the analysis of scheduling policies in polling systems with gated and exhaustive service, respectively. Finally, Section 5 contains some remarks and suggestions for future research.

2 Model description and notation

Throughout, we consider a polling system with one single server for $N \geq 1$ queues, in which there is infinite buffer capacity for each queue. The server

FB	*Foreground-Background* preemptively shares the server evenly among those jobs that have received the least amount of service so far.
FCFS	*First Come First Served* serves jobs in the order they arrive.
LCFS	*Last Come First Served* non-preemptively serves the job that arrived the most recently.
PLCFS	*Preemptive Last Come First Served* preemptively serves the most recent arrival.
PS	*Processor Sharing* serves all customers simultaneously, at the same rate.
SJF	*Shortest Job First* non-preemptively serves the job in the system with the smallest original size.
SRPT	*Shortest Remaining Processing Time* preemptively serves the job with the shortest remaining size.

Table 1
A brief description of the scheduling policies discussed in this paper.

visits and serves the queues in a fixed cyclic order. We index the queues by i, $i = 1, 2, \dots, N$, in the order of the server movement. For ease of presentation, all references to queue indices greater than N or less than 1 are implicitly assumed to be modulo N, e.g., queue $N + 1$ actually refers to queue 1.

The number of jobs served during a visit to a queue is determined by either exhaustive or gated service, as we described in Section 1. Then, during the visit to each queue, we allow jobs to be scheduled for service according to a variety of scheduling policies, summarized in Table 1. However, we limit the discussion to work-conserving disciplines, i.e. the server serves with the full service rate while at queue i if there is work in queue i. Customers arrive at all queues according to independent Poisson processes with rates λ_i, $i = 1, 2, \dots, N$. The service requirements at queue i are independent, identically distributed random variables with distribution function $F_i(x)$, density $f_i(x)$, mean $\mathbb{E}[X_i]$, and second moment $\mathbb{E}[X_i^2] < \infty$, $i = 1, 2, \dots, N$. We denote $\overline{F}_i(x) = 1 - F_i(x)$. When the server starts service at queue i, a setup time is incurred of which the first and second moment are denoted by $\mathbb{E}[S_i]$ and $\mathbb{E}[S_i^2] < \infty$ respectively. These setup times are identically distributed random variables, independent of any other event involved. The total setup time in a cycle is denoted by S with mean $\mathbb{E}[S] = \sum_{i=1}^{N} \mathbb{E}[S_i]$.[1] Two other important quantities are the mean residual service requirement and the mean residual setup time for queue i, which can be expressed as follows, respectively,

[1] Throughout, we limit our focus to the case $\mathbb{E}[S] > 0$. If this is not true, some subtleties appear due to the fact that the number of cycles with zero length tends to infinity. However, our results can be extended as discussed in [30].

$$\mathbb{E}[R_{X_i}] = \frac{\mathbb{E}[X_i^2]}{2\mathbb{E}[X_i]}, \quad \mathbb{E}[R_{S_i}] = \frac{\mathbb{E}[S_i^2]}{2\mathbb{E}[S_i]}, \qquad i = 1, 2, \ldots, N.$$

The occupation rate (utilization) ρ_i at queue i (excluding setups) is defined by $\rho_i = \lambda_i \mathbb{E}[X_i]$ and the total occupation rate ρ is given by $\rho = \sum_{i=1}^{N} \rho_i$. A necessary and sufficient condition for the stability of this polling system is $\rho < 1$ (see, e.g., [20]). We will always assume the queues are stable.

The cycle length of queue i, $i = 1, 2, \ldots, N$, is defined as the time between two successive arrivals (in case of the gated discipline) or departures (in case of the exhaustive discipline) of the server at this queue. It is well-known that the mean cycle length is independent of the queue involved (and the service discipline) and is given by (see, e.g., [20]) $\mathbb{E}[C] = \frac{\mathbb{E}[S]}{1-\rho}$. We note that although the first moments of the cycle lengths are identical for all queues, higher moments generally differ.

The visit time θ_i of queue i, $i = 1, 2, \ldots, N$, is composed of the service period of queue i, i.e., the time the server spends serving customers at queue i, plus the *preceding* setup time in case of exhaustive service or plus the *succeeding* setup time in case of gated service. By virtue of these two different definitions, a queue is empty exactly at the end of its visit time in case of exhaustive service, while the queue before the gate is empty at the beginning of a visit time in case of gated service (all customers waiting for service are then placed behind the gate). Since the server is working a fraction ρ_i of the time on queue i, the mean of a visit period of queue i reads, for exhaustive service,

$$\mathbb{E}[\theta_i] = \rho_i \mathbb{E}[C] + \mathbb{E}[S_i], \qquad i = 1, 2, \ldots, N,$$

and, for gated service,

$$\mathbb{E}[\theta_i] = \rho_i \mathbb{E}[C] + \mathbb{E}[S_{i+1}], \qquad i = 1, 2, \ldots, N.$$

We define an (i, j)-period $\theta_{i,j}$ as the sum of j consecutive visit times starting in queue i, $j = 1, 2, \ldots, N$. The corresponding mean is given by

$$\mathbb{E}[\theta_{i,j}] = \sum_{n=i}^{i+j-1} \mathbb{E}[\theta_n], \qquad i = 1, 2, \ldots, N, \qquad j = 1, 2, \ldots, N.$$

Notice that in case $j = 1$ and $j = N$, $\mathbb{E}[\theta_{i,j}]$ is equal to the mean visit period $\mathbb{E}[\theta_i]$ of queue i and the mean cycle length $\mathbb{E}[C]$, respectively. The fraction of the time $q_{i,j}$ the system is in an (i, j)-period equals $q_{i,j} = \frac{\mathbb{E}[\theta_{i,j}]}{\mathbb{E}[C]}$, $i = 1, 2, \ldots, N$, $j = 1, 2, \ldots, N$, where $q_{i,N}$ equals 1 by definition. Moreover, the mean of a residual (i, j)-period is given by

$$\mathbb{E}[R_{\theta_{i,j}}] = \frac{\mathbb{E}[\theta_{i,j}^2]}{2\mathbb{E}[\theta_{i,j}]}, \qquad i = 1, 2, \ldots, N, \qquad j = 1, 2, \ldots, N, \tag{1}$$

with the remark that the second moments $\mathbb{E}[\theta_{i,j}^2]$ are still unknown at this stage. Notice that since for each fixed (i, j) the successive (i, j)-periods are not independent, they do not form a renewal process. This means, among others, that Equation (1) does not directly follow from the theory of regenerative processes. For a proof why this result is nevertheless still valid see, e.g., [7].

Our main interest is in the mean response time (sojourn time) $\mathbb{E}[T_i]$ of a type-i customer, $i = 1, 2, \ldots, N$, which is defined as the time in steady state from a customer's arrival at queue i until the completion of his service. Often, it will be more convenient to study the mean delay, $\mathbb{E}[D_i]$, which is defined as $\mathbb{E}[T_i] - \mathbb{E}[X_i]$.

By Little's Law, under non-preemptive policies, these mean delays can be related to the mean queue lengths (excluding the customer possibly in service) $\mathbb{E}[L_i]$, $i = 1, 2, \ldots, N$. The analysis of the present paper is oriented towards the determination of $\mathbb{E}[L_{i,n}]$, the mean queue length at queue i at an arbitrary epoch within a visit time of queue n, $i, n = 1, 2, \ldots, N$. The corresponding unconditional mean queue length $\mathbb{E}[L_i]$ can be expressed in terms of $\mathbb{E}[L_{i,n}]$ as follows: $\mathbb{E}[L_i] = \sum_{n=1}^{N} q_{n,1}\mathbb{E}[L_{i,n}]$, for $i = 1, 2, \ldots, N$.

3 Gated polling systems

We will now develop a simple analytic framework for analyzing scheduling policies in gated polling systems. This framework allows simple arguments to be used to obtain results for the mean delay of a wide variety of scheduling policies and illustrates that the comparison between scheduling policies in gated polling systems is remarkably simple – it is far less complex than in the M/GI/1 model. This is perhaps surprising due to the complexity of the underlying polling system. We first derive expressions for the mean delay of a variety of scheduling policies in terms of the mean residual cycle length in Section 3.1. The policies we consider are FCFS, LCFS, SJF, FB, and PS. In addition, we discuss m-class priority queues, which are of particular practical importance. After deriving the performance of these scheduling policies in terms of the mean residual cycle length, we analyze this quantity in detail in Section 3.2. Finally, we present numerical experiments in Section 3.3.

3.1 The effect of scheduling on mean delay

We will consider the mean delay of a tagged arrival of size x, j_x, to queue i. First note that because we are considering a gated polling system, job j_x will not receive service during the cycle into which it arrives (where we have to recall that a cycle is defined as the time between two successive arrivals of the server at queue i). Further, the length of time remaining in the cycle is simply the residual cycle length, R_{C_i}. Note that the age of the cycle at the arrival of j_x is equal in distribution to the residual of the cycle. The delay of

j_x is made up of three components: the residual cycle length, the amount of service given to arrivals after j_x (and during the same cycle), and the amount of service given to arrivals before j_x (and during the same cycle). To simplify the computation of the latter two components, we notice that all common scheduling policies obey the following two properties:

Property 1 *The contributions to the delay of j_x from each job that arrives before j_x and during the same cycle as j_x, denoted $c_1(X)$, are i.i.d.*

Property 2 *The contributions to the delay of j_x from each job that arrives after j_x and during the same cycle as j_x, denoted $c_2(X)$, are i.i.d.*

Now, we have the following simple representation for the mean delay for a job of size x, $\mathbb{E}[D_i(x)]$, under any policy that obeys Properties 1 and 2:

$$\begin{aligned}\mathbb{E}[D_i(x)] &= \mathbb{E}[R_{C_i}] + \mathbb{E}\left[\sum_{j=1}^{N_A(R_{C_i})} c_1(X_i^{(j)})\right] + \mathbb{E}\left[\sum_{j=1}^{N_A(R_{C_i})} c_2(X_i^{(j)})\right] \\ &= \mathbb{E}[R_{C_i}]\left(1 + \lambda_i \mathbb{E}[c_1(X_i)] + \lambda_i \mathbb{E}[c_2(X_i)]\right), \qquad (2)\end{aligned}$$

where $N_A(Y)$ is the number of arrivals during time Y, and $X_i^{(j)}$ is the job size of the jth arrival.

Using (2), we can now easily obtain formulas for the mean delay of a handful of common scheduling policies under gated polling models. Further, (2) immediately gives bounds on the attainable mean delay under any work conserving policy in gated polling systems. For instance, we see that

$$\mathbb{E}[R_{C_i}] \le \mathbb{E}[D_i(x)] \le \mathbb{E}[R_{C_i}](1 + 2\rho_i).$$

Both of these bounds are in fact tight. With a little work, it is possible to prove that the lower bound can be attained by a policy that preemptively gives jobs of size x highest priority, and the upper bound is attained for all job sizes under a policy that gives priority to the job with the longest remaining size. In addition, this gives us a tight upper bound on the overall mean delay, but a tight lower bound on $\mathbb{E}[D_i]$ does not follow immediately. Later, we will derive a tight lower bound by analyzing SJF, which optimizes $\mathbb{E}[D_i]$, in Section 3.1.3. The resulting bounds on the attainable mean delay are

$$\mathbb{E}[R_{C_i}](1 + \lambda \mathbb{E}[M_i]) \le \mathbb{E}[D_i] \le \mathbb{E}[R_{C_i}](1 + 2\rho_i),$$

where M_i is the minimum of two independent job sizes.

It is already evident that the formulas we derive in this section will be quite different than the results for scheduling policies in the M/GI/1 setting. Further, they are very different than the results we will derive for exhaustive polling systems. The results in the gated polling setting are much simpler and more elegant.

Finally, before moving to the analysis, it is important to note that there is no distinction between preemptive and non-preemptive policies in this setting since new arrivals must wait until after the next cycle for service. Thus, for instance, PLCFS and LCFS are equivalent, as are SJF and SRPT.

3.1.1 FCFS

We start with the simplest, most common scheduling policy: FCFS. The mean delay of FCFS in gated polling systems is well known, but it is useful to note how easily it follows from (2). In the case of FCFS, only arrivals before the tagged job will contribute to the delay of the tagged job. Thus, $\mathbb{E}[c_1(X_i)] = \mathbb{E}[X_i]$ and $\mathbb{E}[c_2(X_i)] = 0$, which gives the well-known result that

$$\mathbb{E}[D_i(x)]^{\mathsf{FCFS}} = \mathbb{E}[R_{C_i}]\,(1+\rho_i)\,, \tag{3}$$

which we refer to as the so-called *arrival relation.*

Note that this is not an explicit formula since $\mathbb{E}[R_{C_i}]$ is unknown. However, it is important to note that (since we are considering only work-conserving policies) $\mathbb{E}[R_{C_i}]$ is independent of the scheduling policy used at each queue. In the remainder of this section, we derive for each individual scheduling discipline an arrival relation in terms of $\mathbb{E}[R_{C_i}]$. We will describe how to calculate $\mathbb{E}[R_{C_i}]$ later in Section 3.2.

3.1.2 LCFS

Another simple, common policy is LCFS. Again, obtaining the mean delay of LCFS from (2) is simple. Since only arrivals after the tagged job contribute to the delay of the tagged job, we have $\mathbb{E}[c_1(X_i)] = 0$, and $\mathbb{E}[c_2(X_i)] = \mathbb{E}[X_i]$, which gives

$$\mathbb{E}[D_i(x)]^{\mathsf{LCFS}} = \mathbb{E}[R_{C_i}]\,(1+\rho_i)\,. \tag{4}$$

Notice that this is the same mean delay we obtained for FCFS. In fact, the same result can easily be shown to hold for all blind, list based policies, i.e. all policies that serve individual jobs to completion using an ordering (list) that is determined without using job sizes.

3.1.3 SJF

We now move beyond simple policies to size-based policies. It is easy to see that SJF optimizes the mean delay (and queue length) under gated polling systems since it is equivalent to SRPT in this setting, which has long been known to optimize queue length and mean delay [18].

Though SJF is a more complex policy than FCFS and LCFS , the mean delay of SJF can still be derived very easily from (2). In particular, if we consider the contribution of an arrival to the delay of a tagged job, the moment when the arrival occurred during the cycle is irrelevant – the arrival will contribute to the delay of the tagged job only if its size is smaller. Thus, we have $\mathbb{E}[c_1(X_i)] = \mathbb{E}[c_2(X_i)] = \mathbb{E}[X_i 1_{[X_i<x]}]$. Defining $\rho_i(x) = \lambda_i \mathbb{E}[X_i 1_{[X_i<x]}]$ then gives

$$\mathbb{E}[D_i(x)]^{\mathsf{SJF}} = \mathbb{E}[R_{C_i}] \left(1 + 2\rho_i(x)\right). \tag{5}$$

Comparing (5) with (3), we can immediately see that small job sizes perform better under SJF than FCFS, but that larger job sizes perform worse. In fact, the largest job sizes have $\mathbb{E}[D_i(x)]^{\mathsf{SJF}} \approx \mathbb{E}[R_{C_i}](1 + 2\rho_i)$, which (2) shows is the worst possible mean delay.

To obtain the overall mean delay under SJF, we need to integrate (5). The result gives us a simple characterization of the *optimal* mean delay:

$$\begin{aligned}\mathbb{E}[D_i]^{\mathsf{SJF}} &= \mathbb{E}[R_{C_i}] \left(1 + 2\lambda_i \int_0^\infty f_i(x) \int_0^x t f_i(t) dt dx\right) \\ &= \mathbb{E}[R_{C_i}] \left(1 + \lambda_i \int_0^\infty t (2 f_i(t) \overline{F}_i(t)) dt\right) \\ &= \mathbb{E}[R_{C_i}] \left(1 + \lambda_i \mathbb{E}[M_i]\right),\end{aligned} \tag{6}$$

where M_i is the minimum of two i.i.d. job sizes and the second line follows from the first by interchanging the integrals. Comparing (3) for FCFS and (6) for SJF shows:

$$\mathbb{E}[D_i]^{\mathsf{SJF}} \leq \mathbb{E}[D_i]^{\mathsf{FCFS}} \Leftrightarrow \mathbb{E}[M_i] \leq \mathbb{E}[X_i].$$

It is easy to see that under deterministic job sizes $\mathbb{E}[D_i]^{\mathsf{SJF}} = \mathbb{E}[D_i]^{\mathsf{FCFS}}$, but as the service distribution variability increases SJF provides more and more improvement over FCFS. For example, under an exponential distribution $\mathbb{E}[D_i]^{\mathsf{SJF}} = \mathbb{E}[R_{C_i}] (1 + \rho_i/2)$, and under distributions that have a decreasing failure rate (DFR) the improvement is even greater, since it is easily shown that under such distributions $\mathbb{E}[M_i] \leq \mathbb{E}[X_i]/2$.

3.1.4 FB

It is often the case that applications do not know job sizes, and therefore cannot use SJF to attain the optimal mean delay. In such cases, the age (attained service) of a job can often serve as an indication of the remaining size of the job. For instance, if job sizes have a decreasing (increasing) failure rate, DFR (IFR), service requirement distribution, then jobs with larger ages are likely to have larger (smaller) remaining sizes. Thus, FB (FCFS) is a “poor man’s SRPT” in the case of DFR (IFR) job sizes. In fact, FB and FCFS have been shown to optimize (among policies ‘blind’ to job sizes) the queue length distribution and the mean delay in G/GI/1 queues when job sizes are DFR and

IFR respectively [16]. Further, since these optimality results hold even when busy periods begin with an arbitrary batch arrival, they also hold in polling systems.

We focus on FB, with the motivation that DFR service distributions are common in computer and telecommunication applications. The mean delay under FB again follows easily from (2). The key observation is that any job that arrives during the same cycle as a tagged job of size x will contribute $\min(X_i, x)$ work to the mean delay of the tagged job, because the server will give an equal service rate to all jobs in the system throughout the visit period (since all new arrivals stay behind the gate). Thus, $\mathbb{E}[c_1(X_i)] = \mathbb{E}[c_2(X_i)] = \mathbb{E}[\min(X_i, x)]$. Defining $\hat{\rho}_i(x) = \lambda_i \mathbb{E}[\min(X_i, x)]$ then gives

$$\mathbb{E}[D_i(x)]^{\mathsf{FB}} = \mathbb{E}[R_{C_i}]\left(1 + 2\hat{\rho}_i(x)\right). \tag{7}$$

Like under SJF, FB clearly benefits small job sizes (compared to FCFS) while hurting large job sizes. In fact, it is again true that large job sizes are treated as badly as possible under any work conserving policy. Comparing (7) and (5) illustrates that FB behaves very similarly to SJF, though FB clearly pays a price for not using job sizes since $\rho_i(x) \le \hat{\rho}_i(x)$. This difference is accentuated when we look at $\mathbb{E}[D_i]$:

$$\begin{aligned}
\mathbb{E}[D_i]^{\mathsf{FB}} &= \mathbb{E}[R_{C_i}]\left(1 + 2\lambda_i \int_0^\infty f_i(x) \int_0^x \overline{F}_i(t)\,dt\,dx\right) \\
&= \mathbb{E}[R_{C_i}]\left(1 + 2\lambda_i \int_0^\infty \overline{F}_i(t)^2 dt\right) \\
&= \mathbb{E}[R_{C_i}]\left(1 + 2\lambda_i \mathbb{E}[M_i]\right), \qquad (8)
\end{aligned}$$

where M_i is again the minimum of two i.i.d. job sizes and the second line follows from the first by interchanging the integrals.

The comparison between (8) and (6) gives a clear picture of the price FB pays for not using job sizes. In addition, (8) gives a very simple comparison between FCFS and FB:

$$\mathbb{E}[D_i]^{\mathsf{FB}} \le \mathbb{E}[D_i]^{\mathsf{FCFS}} \Leftrightarrow \mathbb{E}[M_i] \le \mathbb{E}[X_i]/2.$$

Notice that equality holds under the exponential distribution, and FB is better under DFR distributions and worse under IFR distributions, as expected.

3.1.5 PS

PS is a policy that is widely used in computer systems due to its fairness properties and its simplicity, so it is important that we spend a moment on it here. However, in gated polling systems, PS is actually equivalent to FB, which we just discussed. In particular, because all jobs that arrive during a visit period will not receive service until the next visit, FB will always end up sharing the server evenly among all jobs in the queue, which is exactly what PS does. Thus, all the results we described for FB also hold for PS, including the fact that FB is optimal among blind policies for queue length and mean delay when job sizes are DFR.

3.1.6 m-class priority queues

The last policy we consider is probably the most important from a practical perspective, so we will spend the most time exploring its behavior. In an m-class priority queue, arrivals are tagged by their class, $1, \dots, m$, and then jobs from class i are given preemptive priority over jobs from classes $> i$. Within each class, jobs are served in FCFS order. These policies are often used in practical settings instead of idealized policies like SJF, e.g. [8,17].

The mean delay of a class j job, $\mathbb{E}[D_i^{(j)}]$, is easily derived from (2). Throughout, we use a superscript (j) to specify class j. For a tagged job of class j, all arrivals during the cycle from classes $< j$ and all earlier arrivals from class j will be served before the tagged job. Thus, we have that $\mathbb{E}[c_1(X_i^{(k)})] = \mathbb{E}[X_i^{(k)} 1_{[k \leq j]}]$ and $\mathbb{E}[c_2(X_i^{(k)})] = \mathbb{E}[X_i^{(j)} 1_{[k<j]}]$. Defining $\rho_i^{(j)} = \lambda_i^{(j)} \mathbb{E}[X_i^{(j)}]$ then gives:

$$\mathbb{E}[D_i^{(j)}] = \mathbb{E}[R_{C_i}](1 + 2\sum_{k<j} \rho_i^{(k)} + \rho_i^{(j)}). \tag{9}$$

From (9) the overall mean delay can be calculated using $\mathbb{E}[D_i] = \sum_j \frac{\lambda_i^{(j)}}{\lambda} \mathbb{E}[D_i^{(j)}]$. Though this formula is easy to write, it hides the answer to important questions such as how the distributions of the job sizes in each priority class affect the overall mean delay. In the remainder of the section we will develop a better understanding of this behavior. We will start by studying the case when there are only 2 priority classes, and then we will use the ideas illustrated by this simple case to study the general case of m priority classes.

$m = 2$: Two priority classes

Let us now look at this in the case of two priority classes to see how prioritization affects the overall response time. With two priority classes, we have:

$$\begin{aligned}
\frac{\mathbb{E}[D_i]}{\mathbb{E}[R_{C_i}]} &= \frac{\lambda_i^{(1)}}{\lambda_i}(1 + \rho_i^{(1)}) + \frac{\lambda_i^{(2)}}{\lambda_i}(1 + 2\rho_i^{(1)} + \rho_i^{(2)}) \\
&= 1 + \rho_i - \frac{\lambda_i^{(1)}}{\lambda_i}\rho_i^{(2)} + \frac{\lambda_i^{(2)}}{\lambda_i}\rho_i^{(1)} \\
&= 1 + \rho_i - \frac{\lambda_i^{(1)}\lambda_i^{(2)}}{\lambda_i}\left(\mathbb{E}[X_i^{(2)}] - \mathbb{E}[X_i^{(1)}]\right),
\end{aligned} \tag{10}$$

from which it follows that $\mathbb{E}[D_i] \le \mathbb{E}[D_i]^{\mathsf{FCFS}} \Leftrightarrow \mathbb{E}[X_i^{(1)}] \le \mathbb{E}[X_i^{(2)}]$.

So, prioritizing small job sizes is an effective heuristic. In fact, we can see from (10) that the optimal mean response time for a 2-class priority system will occur when there is a threshold t such that jobs with size $\le t$ have high priority and jobs with size $> t$ have low priority. The natural question, then, is "what is the optimal such t?" We can determine it as follows:

$$\begin{aligned}\frac{\mathbb{E}[D_i]}{\mathbb{E}[R_{C_i}]} &= 1 + \rho_i - \lambda_i F_i(t)\overline{F}_i(t)\left(\int_t^\infty s\frac{f_i(s)}{\overline{F}_i(t)}ds - \int_0^t s\frac{f_i(s)}{F_i(t)}ds\right)\\ &= 1 + \rho_i - \lambda_i\left(F_i(t)\int_t^\infty sf_i(s)ds - \overline{F}_i(t)\int_0^t sf_i(s)ds\right)\\ &= 1 + \rho_i + \lambda_i\left(\overline{F}_i(t)\mathbb{E}[X_i] - \int_t^\infty sf_i(s)ds\right),\end{aligned}$$

which is minimized when the final term is minimized. Taking derivatives, we see that $-f_i(t)\mathbb{E}[X_i] + tf_i(t) = 0$ only when $\mathbb{E}[X_i] = t$, assuming that $f_i(x) > 0$ for all x. It is easy to see that this is a minimum, so the optimal threshold is $t = \mathbb{E}[X_i]$.

The fact that the optimal threshold is simply $\mathbb{E}[X_i]$ in this setting regardless of the shape of the distribution is quite special. In the M/GI/1 setting the optimal threshold is far from insensitive to the shape of the service requirement distribution. Similarly, in the case of exhaustive service the optimal threshold turns out to be not insensitive.

m priority classes

We will now move beyond the 2-class priority setting and consider the behavior of an m-class system. In the m-class case we have, cf. (9),

$$\begin{aligned}\frac{\mathbb{E}[D_i]}{\mathbb{E}[R_{C_i}]} &= \sum_{j=1}^m \frac{\lambda_i^{(j)}}{\lambda_i}\left(1 + \rho_i^{(j)} + 2\sum_{k=1}^{j-1}\rho_i^{(k)}\right)\\ &= 1 + \rho_i + \sum_{j=1}^m \frac{\lambda_i^{(j)}}{\lambda_i}\left(\sum_{k=1}^{j-1}\rho_i^{(k)} - \sum_{k=j+1}^m \rho_i^{(k)}\right)\\ &= 1 + \rho_i - \lambda_i \sum_{j=1}^m \sum_{k=j+1}^m \frac{\lambda_i^{(j)}\lambda_i^{(k)}}{\lambda_i^2}\left(\mathbb{E}[X_i^{(k)}] - \mathbb{E}[X_i^{(j)}]\right), \qquad (11)\end{aligned}$$

where the last line comes from grouping terms having the same $\lambda_i^{(j)}\lambda_i^{(k)}$ multiplier. Equation (11) is the extension of (10). It shows that the improvement of the mean delay of the priority queue over FCFS is directly related to the differences between the means of the priority classes. In fact, this form implies that the m-class policy which optimizes $\mathbb{E}[D_i]$ will be a threshold based policy since the mean delay of any non-threshold based policy can be improved by interchanging mass between two priority classes that overlap so as to separate their means but not change their arrival rates.

Thus, our goal in the remainder of the section is to determine the optimal threshold values for an m-class threshold based policy. Consider an m-class policy with thresholds $t^{(s)}$ such that $0 = t^{(0)} < t^{(1)} < \ldots < t^{(m)} = \infty$ that assigns jobs with size $x \in [t^{(s-1)}, t^{(s)})$ priority s. We will prove that the optimal thresholds are defined by

$$t^{(j)} = \frac{1}{\overline{F}_i(t^{(j-1)}) - \overline{F}_i(t^{(j+1)})} \int_{t^{(j-1)}}^{t^{(j+1)}} u f_i(u) du. \tag{12}$$

Notice the intuition behind the form of this relation: the threshold dividing classes j and $j+1$ is defined as the mean of the total distribution for jobs of classes j plus $j+1$, i.e. the mean of the distribution of jobs that the threshold is dividing.

To prove that (12) defines the optimal thresholds, we start by combining the terms in (12) that have the same $\mathbb{E}[X_i^{(s)}]$ in order to obtain

$$\frac{\mathbb{E}[D_i]}{\mathbb{E}[R_{C_i}]} = 1 + \rho_i - \lambda_i \sum_{s=1}^{m} \frac{\lambda_i^{(s)}}{\lambda_i} \mathbb{E}[X_i^{(s)}] \left(\sum_{j=1}^{s-1} \frac{\lambda_i^{(j)}}{\lambda_i} - \sum_{j=s+1}^{m} \frac{\lambda_i^{(j)}}{\lambda_i} \right). \tag{13}$$

Looking at this term by term, we notice that $\sum_{j=1}^{s-1} \frac{\lambda_i^{(j)}}{\lambda_i} = F_i(t^{(s-1)})$, $\sum_{j=s+1}^{m} \frac{\lambda_i^{(j)}}{\lambda_i} = \overline{F}_i(t^{(s)})$, and $\frac{\lambda_i^{(s)}}{\lambda_i} E X_i^{(s)} = \int_{t^{(s-1)}}^{t^{(s)}} u f_i(u) du$. Returning to (13) we have that

$$\frac{\mathbb{E}[D_i]}{\mathbb{E}[R_{C_i}]} = 1 + \rho_i - \lambda_i \sum_{s=1}^{m} \left(F_i(t^{(s-1)}) - \overline{F}_i(t^{(s)}) \right) \int_{t^{(s-1)}}^{t^{(s)}} u f_i(u) du. \tag{14}$$

Next, we observe that $F_i(t^{(s-1)}) - \overline{F}_i(t^{(s)}) = -\overline{F}_i(t^{(s-1)}) + F_i(t^{(s)})$. Using this, we can write the odd s terms in (14) as the LHS and the even s terms as the RHS and see that adjacent terms telescope leaving

$$\frac{\mathbb{E}[D_i]}{\mathbb{E}[R_{C_i}]} = 1 + \rho_i + \lambda_i \sum_{s=1}^{m-1} \left(1_{[s \text{ odd}]} - F_i(t^{(s)}) \right) \int_{t^{(s-1)}}^{t^{(s+1)}} u f_i(u) du. \tag{15}$$

To find the optimal cutoffs we look at the partial derivatives of (15) with respect to $t^{(s)}$ for $0 < s < m$:

$$
\begin{aligned}
\frac{d}{dt^{(s)}}\left(\frac{\mathbb{E}[D_i]}{\mathbb{E}[R_{C_i}]}\right) &= -\lambda_i \frac{d}{dt^{(s)}}\left\{\left(-1_{[s\,+\,1\text{ odd}]} + F_i(t^{(s+1)})\right)\int_{t^{(s)}}^{t^{(s+2)}} uf_i(u)du\right. \\
&\quad + \left(-1_{[s\text{ odd}]} + F_i(t^{(s)})\right)\int_{t^{(s-1)}}^{t^{(s+1)}} uf_i(u)dt \\
&\quad \left. + \left(-1_{[s\,-\,1\text{ odd}]} + F_i(t^{(s-1)})\right)\int_{t^{(s-2)}}^{t^{(s)}} uf_i(u)dt\right\} \\
&= -\lambda_i \left\{\left(1_{[s\,+\,1\text{ odd}]} - F_i(t^{(s+1)})\right)t^{(s)}f_i(t^{(s)}) + f_i(t^{(s)})\int_{t^{(s-1)}}^{t^{(s+1)}} uf_i(u)du\right. \\
&\quad \left. + \left(-1_{[s\,-\,1\text{ odd}]} + F_i(t^{(s-1)})\right)t^{(s)}f_i(t^{(s)})\right\} \\
&= -\lambda_i f_i(t^{(s)})\left\{\int_{t^{(s-1)}}^{t^{(s+1)}} uf_i(u)du - \left(F_i(t^{(s+1)}) - F_i(t^{(s-1)})\right)t^{(s)}\right\}, \quad (16)
\end{aligned}
$$

from which we can see that (when $f_i(x) > 0$ for all x) the only critical point occurs when

$$
t^{(s)} = \frac{1}{F_i(t^{(s+1)}) - F_i(t^{(s-1)})}\int_{t^{(s-1)}}^{t^{(s+1)}} uf_i(u)du. \qquad (17)
$$

It is easy to see that this is the global minimum.

Choosing the optimal threshold: Discussion and examples
We have seen that in the case of $m = 2$, the optimal cutoff is $t^{(1)} = \mathbb{E}[X_i]$ regardless of the service requirement distribution. This insensitivity to the shape of the distribution is quite surprising, and very different from what occurs in the standard M/GI/1 model and from what we will see in the case of exhaustive polling systems. However, when $m > 2$ the shape of the service requirement distribution plays a role in the choice of the optimal thresholds. We will illustrate this with two examples: the exponential and Pareto distributions.

Example 3.1 *In the case of the exponential distribution with rate μ and $m = 3$, we can solve for $t^{(1)}$ and $t^{(2)}$ very easily. Let $\overline{F}_i(u) = e^{-\mu u}$. Then, using (17) we get that the optimal thresholds satisfy $t^{(2)} = e^{\mu t^{(1)}} \int_{t^{(1)}}^{\infty} u\mu e^{-\mu u}du = t^{(1)} + 1/\mu$ and $t^{(1)} = \frac{1}{1-e^{-\mu t^{(2)}}} \int_0^{t^{(2)}} u\mu e^{-\mu u}du$. This then gives $\mu t^{(1)} = 1 - \mu t^{(2)}\mathrm{e}^{-\mu t^{(2)}}/(1 - \mathrm{e}^{-\mu t^{(2)}})$, yielding $(1 - \mu t^{(1)}) = 2e^{-(1+\mu t^{(1)})}$. Clearly there is only one solution where both $t^{(1)}$ and $t^{(2)}$ are positive. Further, notice that the solution will always have $t^{(1)} < \mathbb{E}[X_i]$ and $t^{(2)} > \mathbb{E}[X_i]$. As an example of what the thresholds will be, in the case of an exponential with mean 1, we obtain $t^{(1)} = 0.59$ and $t^{(2)} = 1.59$.*

Example 3.2 *In the case of the Pareto distribution, we have $\overline{F}_i(u) = (k/u)^{\alpha}$ with $\alpha > 1$. Then, using (17) we get that the optimal thresholds satisfy $t^{(2)} =$*

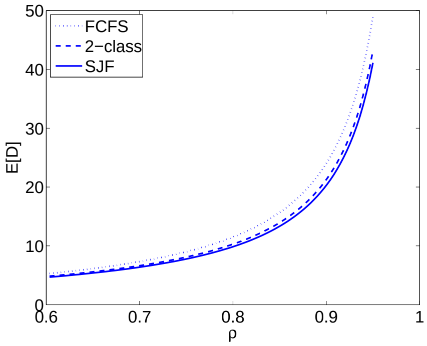

Figure 1. Impact of scheduling in symmetric gated polling systems for exponential service times with $E[X] = 1$.

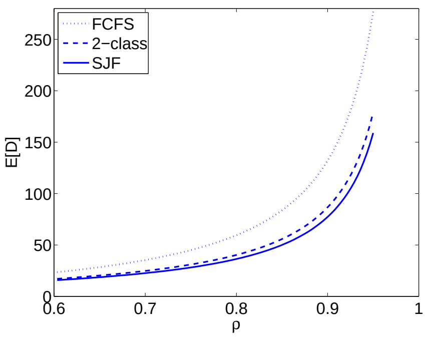

Figure 2. Impact of scheduling in symmetric gated polling systems for Weibull service times with $E[X] = 1$ and $E[X^2] = 20$.

$\left(\frac{t^{(1)}}{k}\right)^{\alpha} \int_{t^{(1)}}^{\infty} u\frac{\alpha k^{\alpha}}{u^{\alpha+1}}du = \frac{\alpha}{\alpha-1}t^{(1)}$ and $t^{(1)} = \frac{1}{1-\left(k/t^{(2)}\right)^{\alpha}} \int_{k}^{t^{(2)}} u\frac{\alpha k^{\alpha}}{u^{\alpha+1}}du$. This then gives $\left(1-\left(\frac{k}{t^{(2)}}\right)^{\alpha}\right) = \left(\frac{\alpha}{\alpha-1}\right)^{2}\left(\frac{k}{t^{(2)}}\right)\left(1-\left(\frac{k}{t^{(2)}}\right)^{\alpha-1}\right)$. Clearly $t^{(2)} = k$ is always a solution, but we are only interested in $t^{(2)} > k$. As an example of solving this, we can look at the case of $\alpha = 2$. Letting $z = k/t^{(2)}$ gives $1 - z^2 = 4z(1-z)$, which has roots $z = 1/3, 1$. This gives $t^{(2)} = 3k$ and $t^{(1)} = 1.5k$. Specializing further to the case when $\mathbb{E}[X_i] = 1$ gives $k = 0.5$ from which we obtain $t^{(2)} = 0.75$ and $t^{(2)} = 1.5$. Notice that these thresholds are more concentrated around the mean than in the case when job sizes are exponential.

3.2 Residual cycles

Throughout the last section, we derived formulas for the mean delay of policies in terms of the mean residual cycle length. Thus, we have isolated the effects of the setup times and the dependencies between visit times into one quantity, which is independent of the scheduling discipline (as long as the scheduling discipline is work-conserving). This allowed us to perform very simple comparisons of the mean delays across all the scheduling disciplines we have considered. Unfortunately though, no general explicit closed-form expression for the mean residual cycle lengths is known. However, we can calculate these mean residual cycle lengths numerically using the recently developed *mean value analysis* (MVA) for FCFS polling systems [30]. Although [30] studies only FCFS polling systems, it also provides – as a by-product – the mean residual visit time, and thus the mean residual cycle lengths, which are independent of the service discipline.

Before going further, we need some additional notation. In the case of gated service, all customers waiting in a queue at the start of a visit time of this queue are placed behind a gate meaning that they are served in the current cycle. However, customers arriving during a visit time of their queue are placed before this gate and are only served in the next cycle. With this difference understood, it is clear that, in case $i = j$, $L_{i,j}$ is the sum of two auxiliary variables, $L_{i,i} = \overline{L}_{i,i} + \widetilde{L}_{i,i}$, where $\overline{L}_{i,i}$ and $\widetilde{L}_{i,i}$ represent the queue length behind and before the gate, respectively. Recall that the customer in service is excluded. In case $i \neq j$, all customers in queue i are located before the gate, i.e., $L_{i,j} = \widetilde{L}_{i,j}$, $i \neq j = 1, 2, \ldots, N$. The corresponding unconditional queue length L_i has mean $\mathbb{E}[L_i] = \sum_{n=1}^{N} q_{n,1}\mathbb{E}[\widetilde{L}_{i,n}] + q_{i,1}\mathbb{E}[\overline{L}_{i,i}]$.

For $i = 1, 2, \ldots, N$ and $j = 1, 2, \ldots, N$, under FCFS scheduling, we have the following set of equations

$$\sum_{n=1}^{N} q_{n,1}\mathbb{E}[\widetilde{L}_{i,n}] + q_{i,1}\mathbb{E}[\overline{L}_{i,i}] = \lambda_i \mathbb{E}[R_{C_i}]\,(1+\rho_i)\,, \tag{18}$$

$$\sum_{n=i}^{i+j-1} \frac{q_{n,1}}{q_{i,j}}\mathbb{E}[\widetilde{L}_{i,n}] = \lambda_i \mathbb{E}[R_{\theta_{i,j}}], \tag{19}$$

$$\mathbb{E}[R_{\theta_{i,1}}] = \mathbb{E}[\overline{L}_{i,i}]\mathbb{E}[X_i] + \frac{\mathbb{E}[S_{i+1}]}{\mathbb{E}[\theta_{i,1}]}\mathbb{E}[R_{S_{i+1}}] + \frac{\rho_i \mathbb{E}[C]}{\mathbb{E}[\theta_{i,1}]}(\mathbb{E}[R_{X_i}] + \mathbb{E}[S_{i+1}]), \tag{20}$$

$$\begin{aligned}\mathbb{E}[R_{\theta_{i,j}}] = \frac{q_{i,1}}{q_{i,j}} \Bigg(&\mathbb{E}[R_{\theta_{i,1}}] \prod_{n=1}^{j-1}(1+\rho_{i+n}) + \sum_{n=1}^{j-1}(\mathbb{E}[S_{i+n+1}] \\ &+\mathbb{E}[\widetilde{L}_{i+n,i}]\mathbb{E}[X_{i+n}]) \prod_{m=n+1}^{j-1}(1+\rho_{i+m}) \Bigg) + (1 - \frac{q_{i,1}}{q_{i,j}})\mathbb{E}[R_{\theta_{i+1,j-1}}].\end{aligned} \tag{21}$$

Elimination of $\mathbb{E}[R_{\theta_{i,j}}]$ from (18) and (19) with the help of (20) and (21) yields a set of $N(N+1)$ linear equations for equally many unknowns $\mathbb{E}[\overline{L}_{i,i}]$ and $\mathbb{E}[\widetilde{L}_{i,n}]$. The solution to these equations yields the mean residual cycle length $\mathbb{E}[R_{C_i}] = \mathbb{E}[R_{\theta_{i,N}}]$, from with we can easily obtain the mean delays for all the scheduling disciplines under consideration.

3.3 Numerical evaluation

We now present some simple numerical experiments illustrating the performance of scheduling policies in gated polling systems. Of course, a wide variety of cases can be studied: different number of queues, choice of service requirement distributions and their parameters, choice of setup distributions and their parameters, etcetera. However, the aim of the present section is to provide some simple illustrative cases which show the potential of scheduling in polling systems, so we will only present a few small examples.

For the first case, we consider a symmetric two-queue polling system with gated service. Suppose that the service and setup times follow exponential distributions with means equal to 1. In this system, we compare three scheduling disciplines, i.e., the optimal policy SJF , the most important one from a practical perspective m-class priority and the standard one FCFS, where we take for the m-class priority systems the number of priority classes equal to 2. Recall that in Section 3.1 we have proven that in the case of two priority classes the optimal threshold is independent of the service requirement distribution and is given by $\mathbb{E}[X_i] = 1$. It goes without saying that the other policies analyzed in the present paper can be evaluated just as easily, but we have omitted them for reasons of presentation.

Figure 1 shows the mean delay of an arbitrary customer as a function of the total load ρ for these three scheduling disciplines under an exponential service distribution. The order of the policies is not surprising, but what is surprising is the fact that the performance of the two-class priority discipline is so close to optimal, i.e. SJF, even though we have distinguished only two priority classes. Furthermore, it is important to observe that, on average, the mean delay for SJF policy is 15% lower than that of FCFS; a gain system operators can achieve without the need of purchasing additional resources. For the second case, all input parameters are taken the same as in the first case, but now the service requirement distribution follows a highly variable Weibull distribution with $E[X] = 1$ and $E[X^2] = 20$. Under this more variable distribution $E[M] = 1/8$, thus the improvement of SJF over FCFS is even more pronounced.

4 Exhaustive service discipline

We now move to the case of exhaustive polling systems. We again develop a framework that allows simple arguments to be used to obtain results for the mean delay of a variety of scheduling policies. The framework will illustrate that the effectiveness of scheduling within queues in exhaustive polling systems is comparable to the effectiveness of scheduling in the M/GI/1 model. This is in contrast to the results we just explored for gated polling systems. Again, we first derive expressions for the mean delay of a variety of scheduling policies in terms of the mean residual cycle length in Section 4.1, and then we analyze the mean residual cycle length in Section 4.2. The policies we consider in this section include FCFS, LCFS, PLCFS, SJF, SRPT, and m-class priority queues.

4.1 The effect of scheduling on mean delay

We will consider the mean delay of a tagged arrival of size x, j_x, to queue i. First note that because we are considering an exhaustive polling system, job j_x will complete during the cycle into which it arrives (unlike in the gated case). We recall that, for the exhaustive discipline, a cycle is defined as the time between two successive departures of the server from queue i. When the tagged job arrives, it will need to wait at least until the server returns to queue i. With probability $\frac{\mathbb{E}[S_i]}{\mathbb{E}[C]}$ it arrives during the setup of queue i and must wait $\mathbb{E}[R_{S_i}]$ before the server returns to queue i. Further, with probability $(1 - q_{i,1})$, the tagged job arrives during an intervisit period and must wait $\mathbb{E}[R_{\theta_{i+1,N-1}}] + \mathbb{E}[S_i]$ time before the server returns to queue i. Let us define the expected time until the server returns to queue i as

$$\mathbb{E}[V_i] = \frac{\mathbb{E}[S_i]}{\mathbb{E}[C]}\mathbb{E}[R_{S_i}] + (1 - q_{i,1})(\mathbb{E}[R_{\theta_{i+1,N-1}}] + \mathbb{E}[S_i]).$$

In addition to waiting $\mathbb{E}[V_i]$ time before receiving service and the job size x itself, depending on the scheduling policy, the response time of j_x may include time devoted to serving (i) jobs that arrive after j_x begins service, (ii) jobs that arrived before j_x, (iii) jobs that arrived after j_x and before j_x receives service. We denote the contribution of the first piece as $c_1(x)$ and the second piece as $c_2(W_i)$, where W_i represents the stationary work at queue i. To simplify the computation of the third component, we notice that many common scheduling policies obey the following property:

Property 3 *The contribution to the delay of j_x from all jobs that arrive after j_x and before j_x receives service, denoted $c_3(X_i)$, is i.i.d. Further, once j_x receives service, no service is given to any jobs that arrived before j_x.*

Many common policies obey Property 3, e.g. FCFS, LCFS, PLCFS, SRPT, and SJF. However, Property 3 does not hold under PS or FB. We will discuss this further in Section 4.1.8. Any policy which obeys Property 3 will have the following representation for the mean response time of a job of size x:

$$\begin{aligned}
\mathbb{E}[D_i(x)] &= \mathbb{E}[c_1(x)] + \mathbb{E}[V_i] + \mathbb{E}\left[\sum_{j=1}^{N_A(V_i)} B_{c_3(X_i^{(j)})}(c_3(X_i^{(j)}))\right] + \mathbb{E}[B_{c_3(X_i)}(c_2(W_i))] \\
&= \mathbb{E}[c_1(x)] + \mathbb{E}[V_i]\left(1 + \frac{\lambda_i \mathbb{E}[c_3(X_i)]}{1 - \lambda_i \mathbb{E}[c_3(X_i)]}\right) + \frac{\mathbb{E}[c_2(W_i)]}{1 - \lambda_i \mathbb{E}[c_3(X_i)]} \\
&= \mathbb{E}[c_1(x)] + \frac{\mathbb{E}[V_i] + \mathbb{E}[c_2(W_i)]}{1 - \lambda_i \mathbb{E}[c_3(X_i)]}, \qquad (22)
\end{aligned}$$

where $N_A(Y)$ is the number of arrivals during time Y, $X_i^{(j)}$ is the job size of the jth arrival, and $B_{X_i}(Y)$ is the length of a busy period started by Y work where service requirements of arrivals have i.i.d. sizes X_i.

Using (22), we can now easily obtain formulas for the mean delay of a handful of common scheduling policies under exhaustive polling models.

4.1.1 FCFS

We start with the simplest policy, FCFS. The mean delay of FCFS in exhaustive polling systems is well-known, but it serves as a useful example of using (22). In the case of FCFS, only arrivals before the tagged job will contribute to the delay of the tagged job. Thus, $\mathbb{E}[c_1(x)] = 0$, $\mathbb{E}[c_2(W_i)] = \mathbb{E}[W_i]$ and $\mathbb{E}[c_3(X_i)] = 0$, which gives

$$E[D_i(x)]^{\mathsf{FCFS}} = \mathbb{E}[V_i] + \mathbb{E}[W_i].$$

To calculate $\mathbb{E}[W_i]$, we use Little's Law to write $\mathbb{E}[D_i(x)]^{\mathsf{FCFS}}$ in terms of the mean number in queue, $\mathbb{E}[L_i]^{\mathsf{FCFS}}$:

$$\mathbb{E}[D_i(x)]^{\mathsf{FCFS}} = \mathbb{E}[V_i] + \rho_i \mathbb{E}[R_{X_i}] + E[L_i]^{\mathsf{FCFS}} \mathbb{E}[X_i].$$

Recalling that $\mathbb{E}[D_i(x)]^{\mathsf{FCFS}} = \mathbb{E}[V_i] + \mathbb{E}[W_i]$ gives

$$\mathbb{E}[W_i] = \frac{\rho_i(\mathbb{E}[V_i] + \mathbb{E}[R_{X_i}])}{1-\rho_i}, \quad \text{and} \quad \mathbb{E}[D_i(x)]^{\mathsf{FCFS}} = \frac{\mathbb{E}[V_i] + \rho_i \mathbb{E}[R_{X_i}]}{1-\rho_i}.$$

In order to view this in terms of the mean residual cycle length, we use the well-known result that:

$$\mathbb{E}[D_i(x)]^{\mathsf{FCFS}} = \mathbb{E}[R_{C_i}](1-\rho_i). \tag{23}$$

It follows that

$$\mathbb{E}[R_{C_i}] = \frac{\mathbb{E}[V_i] + \mathbb{E}[W_i]}{1-\rho_i} = \frac{\mathbb{E}[V_i] + \rho_i \mathbb{E}[R_{X_i}]}{(1-\rho_i)^2}. \tag{24}$$

This will be useful for other policies as well since all work conserving policies have the same mean residual cycle lengths. The calculation of $\mathbb{E}[R_{C_i}]$ is delayed until Section 4.2.

4.1.2 LCFS

Another simple, common policy is LCFS, for which $\mathbb{E}[c_1(x)] = 0$, $\mathbb{E}[c_2(W_i)] = \rho_i \mathbb{E}[R_{X_i}]$, and $\mathbb{E}[c_3(X_i)] = \mathbb{E}[X_i]$ and, thus:

$$\mathbb{E}[D_i(x)]^{\mathsf{LCFS}} = \frac{\mathbb{E}[V_i] + \rho_i \mathbb{E}[R_{X_i}]}{1-\rho_i} = \mathbb{E}[R_{C_i}](1-\rho_i) = E[D_i(x)]^{\mathsf{FCFS}}. \tag{25}$$

LCFS is not alone in having $\mathbb{E}[D_i]$ the same as FCFS. As in the M/GI/1 queue, it is easy to see that all non-preemptive policies that do not use size information have the same mean response time under exhaustive polling systems.

4.1.3 PLCFS

Moving beyond non-preemptive policies, let us now consider PLCFS. Obtaining the mean response time of PLCFS from (22) is simple. Since all arrivals after the tagged job contribute to the response time, we have $\mathbb{E}[c_1(x)] = \rho_i x/(1-\rho_i)$. Further, $\mathbb{E}[c_2(W_i)] = 0$ and $\mathbb{E}[c_3(X_i)] = \mathbb{E}[X_i]$, which gives:

$$\mathbb{E}[D_i(x)]^{\mathsf{PLCFS}} = \frac{\rho_i x + \mathbb{E}[V_i]}{1-\rho_i} = \mathbb{E}[R_{C_i}](1-\rho_i) + \frac{\rho_i}{1-\rho_i}(x - \mathbb{E}[R_{X_i}]). \tag{26}$$

Thus, we can see that $\mathbb{E}[D_i(x)]^{\mathsf{PLCFS}} \leq \mathbb{E}[D_i(x)]^{\mathsf{FCFS}} \Leftrightarrow x \leq \mathbb{E}[R_{X_i}]$, which is the same relation as in the M/GI/1 setting.

4.1.4 Extending the framework

Though we can handle simple policies using (22), in order to handle priority-based policies we need to extend the framework because determining $\mathbb{E}[c_2(W_i)]$ under such policies can be problematic.

To handle such policies we will view $\mathbb{E}[c_2(W_i)]$ as the work in a "transformed" FCFS queue, which will allow us to mimic the derivation in Section 4.1.1. In particular, we will see that the following property holds under SJF, SRPT, and many other priority-based policies.

Property 4 *The contribution $c_2(W_i)$ can be viewed as the work in a "transformed"* FCFS *system where jobs arrive according to a Poisson process with rate λ_i having i.i.d. sizes $c_2'(X_i)$ and a different (maybe dependent) stream of jobs may arrive while the server is idle following a general (maybe non-Poisson) process. The resulting stationary amount of remaining work of the job receiving service is denoted $c_2''(R_{X_i})$.* [2]

As a simple example of Property 4, note that under FCFS the transformed system is the same as the original system, which gives $\mathbb{E}[c_2'(X_i)] = \mathbb{E}[X_i]$ and $\mathbb{E}[c_2''(R_{X_i})] = \rho_i \mathbb{E}[R_{X_i}]$. We will see other examples of transformed systems in the next sections, but let us first examine the implications of Property 4.

Denote the number of jobs in the queue of the "transformed" system as L_i' and the delay in the transformed FCFS queue as $D_i^{\mathsf{FCFS}'}$. Recall that the mean delay in a FCFS queue is simply the work in the system plus $\mathbb{E}[V_i]$, thus $\mathbb{E}[V_i] + \mathbb{E}[c_2(W_i)] = \mathbb{E}[D_i^{\mathsf{FCFS}'}]$. Given a policy obeys Property 4, we can write

$$\mathbb{E}[D_i]^{\mathsf{FCFS}'} = \mathbb{E}[V_i] + \mathbb{E}[c_2''(R_{X_i})] + \mathbb{E}[L_i']\mathbb{E}[c_2'(X_i)],$$

which gives using Little‘s law

$$\mathbb{E}[D_i]^{\mathsf{FCFS}'} = \frac{\mathbb{E}[V_i] + \mathbb{E}[c_2''(R_{X_i})]}{1 - \lambda_i \mathbb{E}[c_2'(X_i)]}.$$

Combining the above with (22) gives

[2] Note that this quantity does not assume that there is a job at the server, and thus is a function of the load as well as the service distribution.

$$
\begin{aligned}
\mathbb{E}[D_i(x)] =& \, \mathbb{E}[c_1(x)] + \frac{\mathbb{E}[V_i] + \mathbb{E}[c_2''(R_{X_i})]}{(1-\lambda_i\mathbb{E}[c_2'(X_i)])(1-\lambda_i\mathbb{E}[c_3(X_i)])} \\
=& \, \mathbb{E}[R_{C_i}]\left(\frac{(1-\rho_i)^2}{(1-\lambda_i\mathbb{E}[c_2'(X_i)])(1-\lambda_i\mathbb{E}[c_3(X_i)])}\right) \\
& + \left(\mathbb{E}[c_1(x)] - \frac{\rho_i\mathbb{E}[R_{X_i}] - \mathbb{E}[c_2''(R_{X_i})]}{(1-\lambda_i\mathbb{E}[c_2'(X_i)])(1-\lambda_i\mathbb{E}[c_3(X_i)])}\right). \qquad (27)
\end{aligned}
$$

The form of (27) is quite illustrative. The first term captures the growth as a function of the mean residual cycle length and the second term captures the tradeoff between giving priority to jobs that arrived earlier versus jobs that arrived later. In addition, (27) illustrates an important comparison between the M/GI/1 model and exhaustive polling systems. Recalling that $\mathbb{E}[D_i]^{\mathsf{FCFS}} = \mathbb{E}[R_{C_i}](1-\rho_i)$, we have that

$$
\begin{aligned}
\mathbb{E}[D_i(x)] =& \, \mathbb{E}[D_i(x)]^{\mathsf{FCFS}}\left(\frac{(1-\rho_i)}{(1-\lambda_i\mathbb{E}[c_2'(X_i)])(1-\lambda_i\mathbb{E}[c_3(X_i)])}\right) \\
& + \left(\mathbb{E}[c_1(x)] - \frac{\rho_i\mathbb{E}[R_{X_i}] - \mathbb{E}[c_2''(R_{X_i})]}{(1-\lambda_i\mathbb{E}[c_2'(X_i)])(1-\lambda_i\mathbb{E}[c_3(X_i)])}\right). \qquad (28)
\end{aligned}
$$

The important point about the above is that the contribution functions $c_i[\cdot]$ are independent of the polling system. So, the only place the polling system impacts (28) is through $\mathbb{E}[D_i(x)]^{\mathsf{FCFS}}$. *Thus, the qualitative relationships between the mean delay of policies that satisfy Properties 3 and 4 are insensitive of the underlying structure of the polling system and only depend on the fact that queues are served exhaustively.* Note that the quantitative differences between policies will depend on the structure of the polling systems though, since the relative weights of the two terms in (28) depend on the magnitude of $E[D_i(x)]^{\mathsf{FCFS}}$.

4.1.5 SJF

Now, let us consider a size-based policy in order to illustrate how to apply (27). SJF is an important policy to consider because it optimizes the mean response time among all non-preemptive policies.

To analyze SJF, consider a transformed FCFS queue where jobs of size $\geq x$ are only allowed to arrive at the moment they begin to receive service in the standard SJF queue. Thus, jobs of size $< x$ still obey a Poisson process but jobs with size $\geq x$ do not. The mean response time for the tagged job is the same in both of these queues. Thus, for SJF, we have that $\mathbb{E}[c_1(x)] = 0$, $\mathbb{E}[c_2'(X_i)] = \mathbb{E}[X_i 1_{[X_i<x]}]$, $\mathbb{E}[c_2''(R_{X_i})] = \rho_i\mathbb{E}[R_{X_i}]$, and $\mathbb{E}[c_3(X_i)] = \mathbb{E}[X_i 1_{[X_i<x]}]$. Applying (27) gives:

$$
\mathbb{E}[D_i(x)]^{\mathsf{SJF}} = \frac{\mathbb{E}[V_i] + \rho_i\mathbb{E}[R_{X_i}]}{(1-\rho_i(x))^2} = \mathbb{E}[R_{C_i}]\left(\frac{1-\rho_i}{1-\rho_i(x)}\right)^2, \qquad (29)
$$

where $\rho_i(x) = \lambda_i \mathbb{E}[X_i 1_{[X_i<x]}]$. Thus, we can see that $\mathbb{E}[D_i(x)]^{\mathsf{SJF}} \leq \mathbb{E}[D_i(x)]^{\mathsf{FCFS}} \Leftrightarrow \rho_i(x) \leq 1 - \sqrt{1-\rho_i}$, which also holds in the M/GI/1 setting.

To obtain the overall mean delay of SJF, we can simply integrate (29):

$$\mathbb{E}[D_i]^{\mathsf{SJF}} = \mathbb{E}[R_{C_i}] \int_0^\infty \left(\frac{1-\rho_i}{1-\rho_i(x)}\right)^2 f_i(x)dx.$$

Unfortunately though, no closed-form solution is available for this integral. It is easy to see however that $\int_0^\infty \left(\frac{1-\rho_i}{1-\rho_i(x)}\right)^2 f_i(x)dx \leq 1-\rho_i$ and thus $\mathbb{E}[D_i]^{\mathsf{SJF}} \leq \mathbb{E}[D_i]^{\mathsf{FCFS}}$ as expected.

4.1.6 SRPT

As in the M/GI/1 setting, SRPT optimizes mean response time in exhaustive polling systems. However, the mean delay of SRPT has not been derived in this setting. But, the analysis of SRPT follows easily from what we have just described for SJF because SRPT also satisfies Property 4.

In the case of SRPT, the transformed system that we use has jobs with original size $< x$ arrive at the same instants as normal, but has jobs with original size $\geq x$ arrive to the server at the moment they obtain remaining size x. Thus, they always arrive when the transformed system is idle. Thus, we obtain $\mathbb{E}[c_2'(X_i)] = \mathbb{E}[X_i 1_{[X_i<x]}]$ and $\mathbb{E}[c_2''(R_{X_i})] = \hat{\rho}_i(x)\mathbb{E}[R_{\min(X_i,x)}]$, where $\hat{\rho}_i(x) = \lambda_i \mathbb{E}[\min(X_i, x)]$. Further, noting that new arrivals contribute to the response time of the tagged job only when they are smaller than the remaining size of the tagged job, we have $\mathbb{E}[c_3(x)] = \mathbb{E}[X_i 1_{[X_i<x]}]$ and $\mathbb{E}[c_1(x)] = \int_0^x (\frac{1}{1-\rho_i(t)} - 1)dt = \int_0^x \frac{\rho_i(t)}{1-\rho_i(t)}dt$, where $\frac{dt}{1-\rho_i(t)}$ should be interpreted as the length of a busy period started by dt work including all new arrivals of size $< t$. Applying (27) then gives:

$$\begin{aligned}\mathbb{E}[D_i(x)]^{\mathsf{SRPT}} &= \int_0^x \frac{\rho_i(t)}{1-\rho_i(t)}dt + \frac{\mathbb{E}[V_i] + \hat{\rho}_i(x)\mathbb{E}[R_{\min(X_i,x)}]}{(1-\rho_i(x))^2} \\ &= \mathbb{E}[R_{C_i}]\left(\frac{1-\rho_i}{1-\rho_i(x)}\right)^2 + \int_0^x \frac{\rho_i(t)}{1-\rho_i(t)}dt - \frac{\rho_i \mathbb{E}[R_{X_i}] - \hat{\rho}_i(x)\mathbb{E}[R_{\min(X_i,x)}]}{(1-\rho_i(x))^2}. \quad (30)\end{aligned}$$

As with SJF, we can obtain the overall mean delay of SRPT by integrating (30); however, such integration must be done numerically. But, without resorting to numerics, it is already evident that SRPT can provide significant reductions in mean delay when compared to FCFS and even SJF.

4.1.7 m-class priority queues

We now move to m-class priority queues. We will limit our discussion to non-preemptive priority queues so that the results can be contrasted with the results from the gated polling systems in Section 3.1.6.

The mean delay of a class j job, $\mathbb{E}[D_i^{(j)}]$, is again easily derived from (27). Forgoing the details since they parallel the analysis of SJF, we have that $\mathbb{E}[c_1(X_i)] = 0, \mathbb{E}[c_2'(X_i)] = \mathbb{E}[X_i^{(k)} 1_{[k \le j]}], \mathbb{E}[c_2''(X_i)] = \rho_i \mathbb{E}[R_{X_i}]$, and $\mathbb{E}[c_3(X_i)] = \mathbb{E}[X_i^{(k)} 1_{[k<j]}]$. Thus, (27) gives:

$$\mathbb{E}[D_i^{(j)}] = \mathbb{E}[R_{C_i}] \left(\frac{(1-\rho_i)^2}{(1 - \sum_{k<j} \rho_i^{(k)})(1 - \sum_{k \le j} \rho_i^{(k)})} \right), \tag{31}$$

where $\rho_i^{(j)} = \lambda_i^{(j)} \mathbb{E}[X_i^{(j)}]$. Notice that the mean delay of SJF can be obtained by taking the appropriate limits. From (31) we can calculate the overall mean delay using $\mathbb{E}[D_i] = \sum_j \frac{\lambda_i^{(j)}}{\lambda_i} \mathbb{E}[D_i^{(j)}]$. As with the gated case, this formula is easy to write but it hides the behavior of the mean delay as a function of the job sizes of each class. As in the gated case, it is straightforward to show that the mean delay will be minimized when priority is given to the classes that have small service requirements. Thus, it again makes sense to consider threshold based policies. However, unlike the gated case, we cannot derive a closed form expression for the optimal threshold. This is not surprising since such an expression does not exist for the M/GI/1 setting either. However, in the case of 2 priority classes, we can determine the optimal threshold and contrast it with our results for gated polling systems in Section 3.1.6.

$m = 2$: The optimal threshold
In the case of two priority classes, we can simplify the expression for the mean delay. In particular, letting t be the threshold used by the policy, we have

$$\frac{\mathbb{E}[D_i]}{\mathbb{E}[D_i]^{\mathsf{FCFS}}} = \frac{\lambda_i^{(1)}}{\lambda_i} \frac{1-\rho_i}{1-\rho_i^{(1)}} + \frac{\lambda_i^{(2)}}{\lambda_i} \frac{1}{1-\rho_i^{(1)}} = \frac{1-\rho_i F_i(t)}{1-\rho_i^{(1)}}.$$

Differentiating this expression, we find

$$\frac{d}{dt}\left(\frac{\mathbb{E}[D_i]}{\mathbb{E}[D_i]^{\mathsf{FCFS}}} \right) = \frac{-\rho_i f_i(t)(1-\rho_i^{(1)}) + \lambda_i t f_i(t)(1-\rho_i F_i(t))}{(1-\rho_i^{(1)})^2},$$

which gives that the mean delay is minimized when the threshold satisfies

$$\frac{t}{\mathbb{E}[X_i]} = \frac{1 - \lambda_i \int_0^t s f_i(s) ds}{1 - \rho_i F_i(t)}.$$

Though this is not explicit, it can be solved easily in the case of many common service distributions. For instance, if job sizes are chosen uniformly from the range $(0, a)$, then the optimal threshold is $t = \frac{2}{\lambda_i}(1 - \sqrt{1-\rho_i})$. Further, if job sizes are exponential with mean $1/\mu_i$, the optimal threshold satisfies $\frac{\mu_i}{\lambda_i} - \frac{e^{-\mu_i t}}{\mu_i t - 1} = 1$.

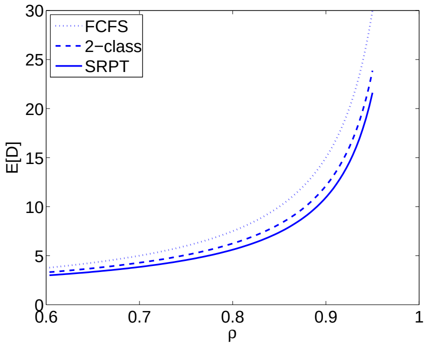

Figure 3. Impact of scheduling in symmetric exhaustive polling systems for exponential service times with $E[X] = 1$.

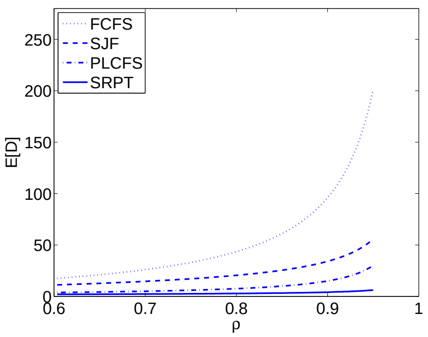

Figure 4. Impact of scheduling in exhaustive polling systems for Weibull service times with $E[X] = 1$ and $E[X^2] = 20$.

Notice the difference between these results and what we found for gated polling systems. In the gated case, the optimal threshold for 2 priority classes was $\mathbb{E}[X]$ regardless of the service distribution. In contrast, here the optimal threshold is $\geq \mathbb{E}[X_i]$ for all service distributions (note that the optimal threshold is an increasing function of λ_i and as $\lambda_i \to 0$, $t \to \mathbb{E}[X_i]$) and depends greatly on the shape of the distribution.

4.1.8 Policies that do not obey Properties 3 and 4

Though we have seen that many common policies obey Properties 3 and/or 4, there are also policies that do not satisfy them. Foremost, PS does not satisfy either 3 or 4. Similarly, all PS-type policies such as Discriminatory, Weighted, and Multi-level PS also violate these properties. Thus, our analytic framework does not apply to these policies.

In fact, it is easy to see that these policies are fundamentally more difficult to analyze in exhaustive polling systems than they are in the M/GI/1 model (and of course more difficult than in gated polling systems). To see this, notice that an analysis of the mean delay of PS in exhaustive polling systems depends on understanding the transient behavior of the queue length distribution under PS in the M/GI/1 model, which is known to be a very difficult problem [10]. Thus, we leave the analysis of PS-type policies as an open question and note that, unlike policies that satisfy Properties 3 and/or 4, the behavior of PS will be very different than it is in the stationary M/GI/1 setting. However, not every policy that violates Properties 3 and/or 4 is difficult to analyze in exhaustive polling systems, e.g. FB violates these properties but can be analyzed directly.

4.2 Residual cycles

In Section 4.1 we have been able to express the mean delay of a variety of scheduling disciplines in terms of the (unknown) mean residual cycle length, where this quantity is independent of the specific scheduling discipline. To compute these unknowns, we again make use of the MVA for FCFS polling systems [30], which yields, as a spin-off, the mean residual cycle lengths under all work conserving policies. In particular, the MVA equations of [30] give that for $i = 1, 2, \ldots, N$, and $j = 1, 2, \ldots, N-1$,

$$\sum_{n=1}^{N} q_{n,1}\mathbb{E}[L_{i,n}] = \frac{\lambda_i}{1-\rho_i}\Big(\rho_i\mathbb{E}[R_{B_i}] + \frac{\mathbb{E}[S_i]}{\mathbb{E}[C]}\mathbb{E}[R_{S_i}] + (1-q_{i,1})(\mathbb{E}[R_{\theta_{i+1,N-1}}] + \mathbb{E}[S_i])\Big), \tag{32}$$

$$\lambda_i\mathbb{E}[R_{\theta_{i+1,j}}] = \sum_{n=i+1}^{i+j} \frac{q_{n,1}}{q_{i+1,j}}\mathbb{E}[L_{i,n}], \tag{33}$$

$$\mathbb{E}[R_{\theta_{i,1}}] = \frac{1}{1-\rho_i}\left(\mathbb{E}[L_{i,i}]\mathbb{E}[X_i] + \frac{\rho_i\mathbb{E}[C]}{\mathbb{E}[\theta_{i,1}]}\mathbb{E}[R_{X_i}] + \frac{\mathbb{E}[S_i]}{\mathbb{E}[\theta_{i,1}]}\mathbb{E}[R_{S_i}]\right), \tag{34}$$

and for $j = 2, 3, \ldots, N$,

$$\mathbb{E}[R_{\theta_{i,j}}] = \frac{q_{i,1}}{q_{i,j}}\left(\frac{\mathbb{E}[R_{\theta_{i,1}}]}{\prod_{n=1}^{j-1}(1-\rho_{i+n})} + \sum_{n=1}^{j-1}\frac{\mathbb{E}[S_{i+n}] + \mathbb{E}[L_{i+n,i}]\mathbb{E}[X_{i+n}]}{\prod_{m=n}^{j-1}(1-\rho_{i+m})}\right) + (1-\frac{q_{i,1}}{q_{i,j}})\mathbb{E}[R_{\theta_{i+1,j-1}}], \tag{35}$$

with as unknowns the mean residual (i, j)-periods $\mathbb{E}[R_{\theta_{i,j}}]$ and the mean conditional queue lengths $\mathbb{E}[L_{i,n}]$. The set (32)-(35) can be solved numerically and the solution yields, among other things, the mean residual cycle lengths $\mathbb{E}[R_{C_i}] = \mathbb{E}[R_{\theta_{i+1,N}}]$. Subsequently, the unconditional mean queue lengths and the mean delays can be computed for all scheduling disciplines.

4.3 Numerical evaluation

We will now move to illustrating the results for scheduling policies in exhaustive polling systems. Our numeric examples use the same system as in the gated case, i.e. a symmetric two-queue system with exponentially distributed setup times with mean 1. Figures 3 and 4 show the output as function of the total load ρ under exponential and Weibull service distributions, respectively. In Figure 3, we compare FCFS, 2-class priority, and SRPT in the case of an exponential service distribution. As in the gated case, we can conclude that the simple 2-class priority discipline is close to optimal and that proper scheduling has a significant impact on the system performance. However, the latter effect is much more pronounced in the exhaustive case than in the gated case. The reason for this effect is that the exhaustive discipline takes advantage of preemption implying that small jobs that arrive during a visit have really small response times since they can preempt. However, in the gated case, small jobs cannot have their response times improved nearly as much since they will always include the residual of the cycle length. Next, in Figure 4, we compare FCFS, SJF, PLCFS, and SRPT in the case of a highly variable Weibull service distribution. This figure illustrates the need to use preemptive scheduling when the job size distribution is highly variable. Though 2-class priority policies nearly matched SRPT in Figure 3, in this figure even SJF (which uses an infinite number of priority classes) is far from SRPT, and is outperformed by PLCFS, which does not use job size information.

5 Discussion and extensions

In this paper, we have studied the impact of scheduling *within* queues in polling systems. The vast prior literature studying polling systems has largely ignored this topic, however we find that the impact of scheduling within queues can be dramatic. One could postulate the (perhaps intuitively appealing) claim that scheduling within a queue has only a minor effect on overall system performance. Namely, one could argue that such a local decision only influences a small part of the delay of a customer, since a major part consists of the time until the server returns to the queue under consideration, which is unaffected by the scheduling policy. The results in this paper refute this assertion. The explanation for this is that at polling instants there is often a large batch of jobs waiting for service and, thus, the order in which these jobs are served really matters.

In order to arrive at the above conclusion, we have developed a simple unified framework for analyzing scheduling policies in classical asymmetric polling systems with either gated or exhaustive service. This framework provides the first analysis of many scheduling policies in polling systems, e.g. SJF, SRPT, and m-class priority queues, which has provided a number of interesting discoveries. For instance, the optimal threshold for 2-class priority queues is insensitive to the service distribution (beyond the mean) under gated polling systems, but extremely sensitive to the service distribution under exhaustive polling systems. Further, this framework significantly extends the MVA for FCFS polling systems developed in [30]. One of the most striking observations provided by this framework is the fact that a large class of scheduling policies behaves the same in exhaustive polling models as in the standard M/GI/1 model, whereas scheduling policies in gated polling models have a very different effect than in the M/GI/1 model. This difference manifests itself not only in the complexity of the analysis, but also in the impact a scheduling discipline has on the overall mean delay.

We have limited ourselves to relatively simple polling systems in this document. However, MVA for FCFS polling systems has been shown to also apply to the following variants [30]: (i) systems with Poisson batch arrivals, (ii) systems with fixed polling tables and (iii) discrete-time polling systems. Moreover, the analysis can be extended without further complication to models with mixtures of gated and exhaustive service, i.e., where some of the queues are gated and some are exhaustive. In turn, this implies that our framework can be readily extended in the same directions.

As stated in the introduction, the decision studied in the present paper - the order in which customers are served – is only one of the three design decisions a system operator must make. In the present paper, we have solved this issue to optimality. For the other two decisions – the order in which queues are served and how many customers served during each visit to a queue - approximately optimal solutions are already available in literature [3,4,5,9,26]. Thus, we can now study a polling system where every decision uses the (approximately) optimal solution, which can illustrate how much system performance can be improved *without* purchasing additional resources.

References

[1] S. Borst, O. Boxma, R. Núñez Queija, and B. Zwart. The impact of the service discipline on delay asymptotics. *Performance Evaluation* 54: 175–206, 2003.

[2] O. Boxma. Workloads and waiting times in single-server systems with multiple customer classes. *Queueing Systems* 5: 185–214, 1989.

[3] O. Boxma, H. Levy, and J. Weststrate. Efficient visit frequencies for polling tables: minimization of waiting cost. *Queueing Systems* 9(1–2): 133–162, 1991.

[4] O. Boxma, H. Levy, and J. Weststrate. Efficient visit orders for polling systems. *Performance Evaluation* 18: 103–123, 1993.

[5] O. Boxma, and D. Down. Dynamic server assignment in a two-queue model. *European Journal of Operational Research* 103: 101–115, 1997.

[6] L. Fournier, and Z. Rosberg. Expected waiting times in polling systems under priority disciplines. *Queueing Systems* 9: 419–440, 1991.

[7] P. Franken, D. Koenig, U. Arndt, and V. Schmidt. *Queues and Point Processes.* John Wiley & Sons, 1982.

[8] M. Harchol-Balter, B. Schroeder, N. Bansal, and M. Agrawal. Implementation of SRPT scheduling in web servers. *ACM Transactions on Computer Systems* 21(2): 207-233, 2003.

[9] M. Hofri, and K. Ross. On the optimal control of two queues with server setup times and its analysis. *SIAM Journal on Computing* 16(2): 399–420, 1987.

[10] M.Yu. Kitaev. The $M/G/1$ processor-sharing model: transient behavior. *Queueing Systems* 14: 239–273, 1993.

[11] H. Levy, and M. Sidi. Polling systems: applications, modeling and optimization. *IEEE Transactions on Communications* 38(10): 1750–1760, 1990.

[12] C. Mack, T. Murphy, and N. Webb. The efficiency of N machines unidirectionally patrolled by one operative when walking time and repair times

are constants. *J. of the Royal Statistical Society Series B* 19(1): 166–172, 1957.

[13] C. Mack. The efficiency of N machines uni-directionally patrolled by one operative when walking time is constant and repair times are variable. *Journal of the Royal Statistical Society Series B* 19(1): 173–178, 1957.

[14] I. Rai, G. Urvoy-Keller, and E. Biersack. Analysis of LAS scheduling for job size distributions with high variance. In *Proceedings of ACM Sigmetrics*, 2003.

[15] I. Rai, G. Urvoy-Keller, M. Vernon, and E. Biersack. Performance modeling of LAS based scheduling in packet switched networks. In *Proceedings of ACM Sigmetrics-Performance*, 2004.

[16] R. Righter, J. Shanthikumar, and G. Yamazaki. On external service disciplines in single stage queueing systems. *J. of Applied Probability* 27: 409–416, 1990.

[17] M. Rawat, and A. Kshemkalyani. SWIFT: Scheduling in web servers for fast response time. In *Symposium on Network Computing and Applications*, 2003.

[18] L. Schrage. A proof of the optimality of the shortest remaining processing time discipline. *Operations Research* 16: 687–690, 1968.

[19] S. Shimogawa, and Y. Takahashi. A note on the pseudo-conservation law for a multi-queue with local priority. *Queueing Systems* 11(1-2): 145–151, 1992.

[20] H. Takagi. Queueing analysis of polling models: an update. In *Stochastic Analysis of Computer and Communication Systems*, H. Takagi (ed.), North-Holland, Amsterdam, 267–318, 1990.

[21] H. Takagi. Queueing analysis of polling models: progress in 1990-1994. In *Frontiers in Queueing: Models, Methods and Problems*, J.H. Dshalalow (ed.), CRC Press, Boca Raton, 119–146, 1997.

[22] H. Takagi. Analysis and application of polling models. In *Performance Evaluation: Origins and Directions*, G. Haring, C. Lindemann and M. Reiser (eds.), Lecture Notes in Comp. Sci., vol. 1769, Springer, Berlin, 423–442, 2000.

[23] Y. Takahashi, and B. K. Kumar. Pseudo-Conservation Law for a priority polling system with mixed service strategies. *Perf. Eval.* 23(2): 107-120, 1995.

[24] Z. Tsai, and I. Rubin. Mean delay analysis of a message priority-based polling scheme. *Queueing Systems* 11: 223–240, 1992.

[25] V. Vishnevskii, and O. Semenova. Mathematical methods to study the polling systems. *Automation and Remote Control* 67: 173–220, 2006.

[26] M. van Vuuren, and E. Winands. Iterative approximation of k-limited polling systems. *Queueing Systems* 55(3) 161-178, 2007.

[27] A. Wierman, and M. Harchol-Balter. Classifying scheduling policies with respect to unfairness in an M/GI/1. In *Proceedings of ACM Sigmetrics*, 2003.

[28] A. Wierman, M. Harchol-Balter, and T. Osogami. Nearly insensitive bounds on SMART scheduling. In *Proceedings of ACM Sigmetrics*, 2005.

[29] E. Winands, I. Adan, and G.-J. van Houtum. The stochastic economic lot scheduling problem: a survey. *BETA WP-133*, Beta Research School for Operations Management and Logistics, 2005.

[30] E. Winands, I. Adan, and G.-J. van Houtum. Mean value analysis for polling systems. *Queueing Systems* 54(1): 45–54, 2006.

[31] U. Yechiali. Optimal dynamic control of polling systems. In *Proceedings 13th International Teletraffic Congress, Workshop: Queueing, Performance and Control in ATM*, J.W. Cohen and C.D. Pack (eds.), North-Holland Publ. Cy., Amsterdam, 205–218, 1991.