ELSEVIER

Performance Evaluation 40 (2000) 27–46

PERFORMANCE EVALUATION
An International Journal

www.elsevier.com/locate/peva

A novel approach to queue stability analysis of polling models ☆

Rocky K.C. Chang *, Sum Lam

Department of Computing, The Hong Kong Polytechnic University, Kowloon, Hong Kong

Abstract

Previous work in the stability analysis of polling models concentrated mainly on stability of the whole system. This system stability analysis, however, fails to model many real-world systems for which some queues may continue to operate under an unstable system. In this paper we address this problem by considering *queue stability problem* that concerns stability of an individual queue in a polling model. We present a novel approach to the problem which is based on a new concept of queue stability orderings, dominant systems, and Loynes' theorem. The polling model under consideration employs an m-limited service policy, with or without prior service reservation; moreover, it admits state-dependent set-up time and walk time. Our stability results generalize many previous results of system stability. Furthermore, we show that stabilities of any two queues in the system can be compared solely based on their (λ/m)'s, where λ is the customer arrival rate to a queue.

Keywords: Queue stability analysis; Queue stability ordering; Loynes' theorem; Dominant systems; Polling models; Reservation schemes; State-dependent walk time and set-up time

1. Introduction

Polling models have been studied extensively, owing to their applications in the performance analysis of many computer and communications systems [1,2]. Recently, stability analysis of polling models has received a lot of attention [3–10]; this growing interest is perhaps due to both the depth and importance of the problem. Stability defines a system's achievable operating region, thus directly affecting the system performance, such as customer delay and maximum throughput. Previous work in the stability analysis of polling models concentrated mainly on *system stability* that addresses stability of the whole system. A polling system is considered stable if *all* queues in the system are stable; the system, by definition, is unstable if any queue in the system becomes unstable. By queue stability we mean that the queue length

☆ An earlier version of this paper was presented at the Performance and Control of Network Systems II Conference, in: Wai Sum Lai, Robert B. Cooper (Eds.), Proceedings of SPIE (The International Society for Optical Engineering), Vol. 3530, Boston, Mass., 2–4 November 1998.

* Corresponding author. Tel.: +852-2766-7258; fax: +852-2774-0842.
E-mail address: csrchang@comp.polyu.edu.hk (R.K.C. Chang)

0166-5316/00/$ – see front matter
PII: S0166-5316(99)00068-1

process for a queue with unlimited buffer space possesses a limiting distribution. Applications of stability results to the modeling of computer networking systems can be found in Refs. [11–13].

The system stability analysis, even though a difficult problem already, is inadequate in modeling many real-world systems. For example, token-passing LAN, polling scheme, and processor sharing schedule are all engineered such that an unstable queue will not cause other queues unstable. Therefore, the system stability analysis cannot address those stable queues which continue to operate under an unstable system. In this paper we address the scenario just described by considering a more general problem: *queue stability problem*. Queue stability concerns stability of an *individual* queue in a polling model; therefore, queue stability results generalize system stability results. Furthermore, queue stability analysis provides insight into how individual (or classes of) queues, equipped with certain service policies, interact with one another in sharing a single resource. Queue stability results are also needed for approximating queueing delay through interpolation techniques [14]. Our main contribution in this paper is a novel approach to the queue stability problem. We apply the approach to a polling system equipped with m-limited service policy, with or without prior service reservation; moreover, both set-up time and walk time are state-dependent. This polling model is considered *nonlinear* because the stability conditions are nonlinear functions of the customer arrival rates. We also extend the stability results to unlimited service policies; that is, $m = \infty$.

To the best of our knowledge, a formal queue stability analysis for polling models has not been undertaken, although there is a growing number of publications in the area of system stability analysis. Ibe and Cheng [8] considered polling systems with limited service policies, and employed a heuristic argument to obtain sufficient queue stability conditions. The necessity of the conditions was left unproved (see Corollary 4 in Section 5 for the proof); moreover, their approach cannot be extended to nonlinear polling systems, such as the one considered in this paper. Georgiadis and Szpankowski [7] employed Loynes' theorem for the stability of G/G/1 queue, dominant systems, and an induction procedure to prove the well-known system stability conditions for a gated m-limited policy. They and Tassiulas [15] applied the same approach to find system stability conditions for a ring network with spatial reuse. Their approach alone, however, is not sufficient for determining queue stability conditions, because queue stability analysis requires a comparison of stabilities of the queues in the system. Nevertheless, we shall adopt their approach of applying Loynes' theorem to a single queue in the system, if such comparison is known. Fricker and Jaïbi, on the other hand, considered a very general class of service policies, and each queue could employ different policies during each stage. They obtained system stability conditions through stochastic monotonicity property of Markov chains, and an inductive procedure [6]. Like the previous work just cited, their approach does not apply to the nonlinear polling model considered in this paper.

Another important result obtained in this paper is *queue stability ordering* that determines the sequence of queues becoming unstable if the rates of customers arriving to the queues are increasing proportionally. Stability ordering is a crucial instrument for us to obtain queue stability conditions in this paper. The concept of stability ordering had been alluded to in an earlier work in Ref. [17], and the queue, the first one responsible for the system instability, was termed a least stable queue [4]. Moreover, Fricker and Jaïbi [6] reported a similar stability ordering result. They showed that, if the queues are ordered according to a nonincreasing order of their λ/m, this ordering is the same as the order of the queues becoming unstable, where λ is the customer arrival rate to a queue. This result can be obtained based on only the system stability conditions. For example, given two queues with $(\lambda_1/m_1) > (\lambda_2/m_2)$, the stability ordering states that queue 1 will become unstable before queue 2. Their result, however, is only partial because nothing has been said about whether $(\lambda_1/m_1) > (\lambda_2/m_2)$ still holds, given that queue 1 becomes unstable before

queue 2. The proof of this new result, unlike the one given by Fricker and Jaïbi, requires queue stability analysis. By combining these two results, we can then show that the stabilities of any two queues in the system can be compared based on only their λ/m.

We organize the rest of this paper as follows. In Section 2 we describe the polling model considered in this paper, and explore the Markovian properties of the underlying queue length processes. In Section 3 we introduce our main approach to the queue stability problem, which is based on queue stability ordering, dominant systems, and Loynes' theorem; in particular, we obtain queue stability condition of a target queue for a given queue stability ordering. In Section 4 we present several queue stability ordering results. In Section 5, by combining the results in Sections 3 and 4, we present complete queue stability conditions for our polling model, and for several other special cases. Moreover, we apply our stability analysis to a pipeline polling system, which slightly deviates from the polling model described in Section 2. We finally conclude this paper in Section 6, with a discussion of our findings and future work needed in this area.

2. Model description and Markovian properties

The polling model considered in this paper consists of a single server, and a finite set of distributed queues. Let Q be the set of queues, and q_i, $i = 1, \ldots, |Q|$, be the members of Q. Each queue has infinite buffers to store incoming customers. The arrival process of customers to q_i is assumed to be Poisson with rate λ_i; the arrival process at a queue is independent of the arrival processes at other queues. B_i^k is the service time of the kth customer at q_i, and the service time process $\{B_i^k\}_{k=1}^{\infty}$ is i.i.d. with a finite mean $b_i > 0$. The service time process at a queue is assumed to be independent of the arrival processes at all queues, and of the service time processes at other queues. We also let $\rho_i = \lambda_i b_i$ and $\rho = \sum_{i=1}^{|Q|} \rho_i$. The server visits the queues in a deterministic and cyclic order: $q_1, q_2, \ldots, q_{|Q|}, q_1, q_2, \ldots$, and he serves the queues according to an m-limited service policy with two variants: *gated at server arrival instants* (GSA) and *gated at server departure instants* (GSD). Each queue is either a GSA queue or a GSD queue. The server will serve $\min(x, m_i)$ customers when he, upon his arrival, finds x customers in a GSA q_i, where m_i, a positive and finite integer, represents an upper limit on the number of services performed during each visit. The server, on the other hand, reserves $\min(y, m_i)$ number of services when he is about to leave behind y customers in a GSD q_i; therefore, he will serve only $\min(y, m_i)$ customers in his next visit. As a result, the GSD and GSA queues model reservation and nonreservation schemes, respectively.

Given the queue length (and the number of reserved services for the GSD queues) at the arrival instant of the server, the service policies are independent of the history of the system prior to the arrival of the server. We also assume that, after starting the service, the server will not be idle until the service is complete (work-conserving), and that the queueing discipline does not depend on the service time. Furthermore, $f_i(x)$, the number of customers served in a GSA q_i given x customers waiting in the queue upon the arrival of the server, is *monotonic and contractive* in x [18]; that is, if $x_1 \geq x_2$, then the two inequalities in Eq. (2.1) hold:

$$f_i(x_1) \geq f_i(x_2) \quad \text{and} \quad f_i(x_1) - f_i(x_2) \leq x_1 - x_2. \tag{2.1}$$

Similarly, $g_i(y)$, the number of services reserved by the server in a GSD q_i given y customers waiting in the queue just before making the service reservation, is monotonic and contractive in y.

2.1. State-dependent set-up time and walk time

The last two aspects of our polling model concern set-up time and walk time. Unlike previous work in the analysis of polling models, we allow both state-dependent set-up time and walk time in our model. The nonnegative set-up time is incurred between the arrival instant of the server and the actual start of service; the nonnegative walk time, between the departure instant of the server and the arrival instant of the server at the next queue. In general, both set-up time and walk time may depend on any system states prior to the start of the set-up and walk, respectively; however, they must be independent of the processes that occur after the completion of set-up time and walk time, respectively (independent of future). In this paper, we assume that the set-up time distribution for a queue depends on only the states of the queue at the arrival instants of the server; the walk time distribution, the states of the queue at the departure instants of the server. Given that the state of q_i at the arrival instant of the server is x, we denote the set-up time by $U_i(x)$ with a finite mean $u_i(x) > 0$, and let $\Theta_i(x)$ be the length of a *service period* at q_i, the total amount of time serving q_i. We also take $\overline{\Theta}_i(x) \stackrel{\text{def}}{=} U_i(x) + \Theta_i(x)$ as q_i's *extended service period*. Similarly, given that the state of q_i at the departure instant of the server is y, we denote the walk time by $V_i(y)$ with a finite mean $v_i(y) > 0$. [1]

Given the service policies, our model can accommodate any state-dependent set-up time and walk time, provided that the following two requirements are fulfilled by all queues:

1. $\overline{\Theta}_i(x)$ and $V_i(y)$ are stochastically monotonic in x and y, respectively; that is, if $x_1 \geq x_2$, then $\overline{\Theta}_i(x_1) \geq_{\text{st}} \overline{\Theta}_i(x_2)$, and similarly for $V_i(y)$. By stochastic monotonicity we mean $\overline{\Theta}_i(x_1) \geq_{\text{st}} \overline{\Theta}_i(x_2)$ if and only if $E[h(\overline{\Theta}_i(x_1))] \geq E[h(\overline{\Theta}_i(x_2))]$ for all monotonic increasing functions h.
2. There exists a finite $x_i^* > 0$ such that $\overline{\Theta}_i(x) \stackrel{\text{d}}{=} \overline{\Theta}_i^*$ for $x \geq x_i^*$; similarly, there exists a finite $y_i^* > 0$ such that $V_i(y) \stackrel{\text{d}}{=} V_i^*$ for $y \geq y_i^*$. Both $\overline{\Theta}_i^*$ and V_i^* are independent of each other, and of any other processes in the system, and have finite means of $\overline{\theta}_i^* > 0$ and $v_i^* > 0$, respectively.

Because $\Theta_i(x)$ is stochastically equivalent to $\Theta_i(m_i)$ for $x \geq m_i$, the two requirements imply that $U_i(x)$ also is stochastically monotonic in x *only* for $x \geq x_i^*$; that is, $U_i(x) \stackrel{\text{d}}{=} U_i^*$ with a finite mean $u_i^* > 0$ for $x \geq x_i^* \geq m_i$. Finally, we observe that $\overline{\theta}_i^* = u_i^* + m_i b_i$. Note that the two requirements are very general, and they could accommodate a wide range of models, such as one that the set-up time is nonzero if the server finds an empty queue, but it is zero for a nonempty queue (see Section 5.1 for such a system). Of course, they include the following special cases also:

- Both set-up time and walk time are independent of each other, and of other processes in the system; that is, $U_i(x) \stackrel{\text{d}}{=} U_i^*, \forall x$, with a finite mean $u_i^* > 0$ and $V_i(y) \stackrel{\text{d}}{=} V_i^*, \forall y$, with a finite mean $v_i^* > 0$. In this case, it is clear that $\overline{\Theta}_i(x)$ is stochastically monotonic in x; furthermore, $\overline{\Theta}_i(x) \stackrel{\text{d}}{=} U_i^* + \Theta_i(m_i)$ for $x \geq m_i$.
- $U_i(x)$ is stochastically monotonic in x, and $U_i(x) \stackrel{\text{d}}{=} U_i^*$ for $x \geq x_i' > 0$ with a finite mean $u_i^* > 0$. This case is similar to the first case with $x_i^* = \max(x_i', m_i)$.
- $U_i(x) \stackrel{\text{d}}{=} U_i^*$ for $x \geq x_i' > 0$; $U_i(x) = 0$, otherwise. The result is the same as the second case.

If walk time depends also on the states of the queue at the arrival instants of the server, we could absorb the walk time into the extended service period; the two requirements for $V_i(y)$ are no longer needed for the GSA queues, but they are still required for the GSD queues. In the remainder of this paper, we assume that the set-up time and walk time for the GSA (GSD) queues depend on the queue length

[1] We may allow either set-up time or walk time to be zero, but not both.

(the queue length and the number of reserved services) at the respective time instants. Because of the monotonicity property of $g(y)$, the walk times for the GSD queues depend on only the queue lengths at the server departure instants. Moreover, when comparing two multi-dimensional variables, we understand that $(x_1, x_2) \geq (x_1', x_2')$ if and only if $x_1 \geq x_1'$ and $x_2 \geq x_2'$.

2.2. Markovian properties

After describing our polling model in the preceding discussion, we now explore the Markovian properties of the model in the rest of this section. The imbedded points are server arrival instants at the GSA queues, and server departure instants from the GSD queues. At the nth epoch, N_i^n is q_i's queue length and G_i^n, only for the GSD queues, is the number of reserved services for q_i. Let $\boldsymbol{\Phi}_Q^n \stackrel{\text{def}}{=} (\Phi_1^n, \ldots, \Phi_{|Q|}^n)$, $n \geq 1$, where $\Phi_i^n = N_i^n$, if q_i is a GSA queue; and $\Phi_i^n = (N_i^n, G_i^n)$ if q_i is a GSD queue. We also find it convenient to consider another process $\{\boldsymbol{\Phi}_Q^k(j)\}_{k=1}^{\infty}$, for which the imbedded points are only those related to q_j; $\boldsymbol{\Phi}_Q^k(j)$, $N_i^k(j)$, and $G_i^k(j)$ have similar meanings as before. Furthermore, we define the following quantities related to the kth ($k \geq 1$) visit at q_i:

- U_i^k is the set-up time at the beginning of the visit.
- F_i^k is the number of customers served during the visit.
- Θ_i^k is the length of the service period.
- V_i^k is the walk time incurred at the end of the visit.

In addition, we define $C^k(j)$ as the kth cycle time referenced at q_j, the amount of time elapsed between the kth and $(k+1)$th arrivals at (departures from) q_i if it is a GSA (GSD) queue. Moreover, we assume, throughout this paper, that in the beginning ($n = k = 1$) the server arrives at GSA (or departs from GSD) q_1.

Having defined the quantities for the kth visit at q_i, we now return to $\boldsymbol{\Phi}_Q^n$ and define the following quantities for all imbedded points:

- k_n is the cycle number, corresponding to the nth epoch; that is, $k_n = \lfloor (n-1)/|Q| \rfloor + 1$.
- J_n is the index of the queue associated with the nth epoch; that is, $J_n = n - |Q|(k_n - 1)$.
- T_n is the time instant of the nth epoch.
- $\mathcal{F}_i^n$ is the total number of customers served at q_i from the beginning up to T_n.
- Ψ_n is the time period between T_n and T_{n+1}.
- $A_i(t)$ is the total number of customers arrived at q_i during $(0, t]$.
- X_i^n is the total number of customers arrived at q_i between T_n and T_{n+1}; that is, $X_i^n = A_i(T_n + \Psi_n) - A_i(T_n)$.

To completely describe the queue length process, we need to consider four different scenarios, depending on the types of queues associated with two consecutive imbedded points. Eq. (2.2) corresponds to the case that the nth epoch is an arrival epoch, and the $(n+1)$th is a departure epoch; other cases can be obtained similarly. Note that if q_{J_n} is a GSA queue, $\Theta_{J_n}^{k_n} = \sum_{j=1}^{\min(N_{J_n}^n, m_{J_n})} B_{J_n}^{\mathcal{F}_{J_n}^n + j}$; moreover, if $q_{J_{n+1}}$ is a GSD queue, $\Theta_{J_{n+1}}^{k_{n+1}} = \sum_{j=1}^{G_{J_{n+1}}^n} B_{J_{n+1}}^{\mathcal{F}_{J_{n+1}}^{n+1} + j}$. We also adopt the notation $[x]^+ = \max(x, 0)$ in this paper. Finally, Lemma 1, the main result of this subsection, states the Markovian properties of the polling model.

$$
\begin{aligned}
&N_i^{n+1} = N_i^n + X_i^n \quad \text{if } i \neq J_n, J_{n+1},\\
&N_{J_n}^{n+1} = [N_{J_n}^n - m_{J_n}]^+ + X_i^n,\\
&N_{J_{n+1}}^{n+1} = N_{J_{n+1}}^n - G_{J_{n+1}}^n + X_i^n \quad \text{and} \quad G_{J_{n+1}}^{n+1} = \min(N_{J_{n+1}}^{n+1}, m_{J_{n+1}}), \quad \text{where}\\
&\Psi_n = U_{J_n}^{k_n} + \Theta_{J_n}^{k_n} + V_{J_n}^{k_n} + U_{J_{n+1}}^{k_{n+1}} + \Theta_{J_{n+1}}^{k_{n+1}}.
\end{aligned}
\tag{2.2}
$$

Lemma 1. *The process* $\{\boldsymbol{\Phi}_Q^n\}_{n=1}^{\infty}$ *is a (generally nonhomogeneous) Markov chain; furthermore,* $\{\boldsymbol{\Phi}_Q^k(j)\}_{k=1}^{\infty}$ *is a homogeneous, irreducible, and aperiodic Markov chain.*

Proof. The proof follows a standard argument, and is therefore omitted. □

3. The main approach

We adopt the definition for queue stability provided by Loynes [16]. A queue is *stable* if the distribution of the queue length process $\{N^n\}_{n=1}^{\infty}$ tends to a honest distribution function at all its points of continuity; that is, for $x \in \mathfrak{R}$, where $\mathfrak{R}$ is a set of real numbers,

$$\lim_{n\to\infty} \Pr\{N^n \le x\} = F(x) \quad \text{and} \quad \lim_{x\to\infty} F(x) = 1. \tag{3.1}$$

Moreover, the queue is *substable* if the distribution is bounded in probability sense; that is,

$$\lim_{x\to\infty} \liminf_{n\to\infty} \Pr\{N^n \le x\} = 1. \tag{3.2}$$

If the queue is not substable, then it is *unstable*. Similar definitions of stability apply also to multi-dimensional processes, with the understanding that two processes are compared based on their respective components.

We employ three instruments for solving the queue stability problem: stability ordering, appropriately constructed dominant systems, and Loynes' theorem for the stability of an isolated G/G/1 queue. Loynes' stability result essentially states that the arrival rate of customers must be less than the service rate. When applying to a particular queue in a polling system, Loynes' theorem still holds, provided that the cycle time process is stationary and ergodic [7]. As a preliminary result, Theorem 1 states the stability condition for a GSA or GSD queue for which the server takes a vacation after servicing the queue. Similar to Loynes' result, the stability condition for this system has an intuitive explanation: the average number of customers arrived during a cycle must be less than the maximum number of customers departed during a cycle.

Theorem 1. *Consider a GSA or GSD queue, and a server who always performs m services during his visit at the queue. If the queue is short of customers, the server generates just enough dummy customers to reach the limit m at his arrival (departure) for a GSA (GSD) queue. Moreover, the server takes a vacation after servicing the queue, and the vacation period is independent of the arrival process. If the cycle time process is a stationary and ergodic sequence with mean EC, then*

1. *if* $\lambda EC < m$, *then the queue is stable in the sense of Eq.* (3.1), *and*
2. *if* $\lambda EC > m$, *then the queue is unstable.*

Proof. Because there are always m customers to serve, the queue length process is given by

$$N^{n+1} = [N^n - m]^+ + X^n, \tag{3.3}$$

where X^n is the number of arrivals at the queue during the nth cycle. Eq. (3.3) is valid for both GSA and GSD queues, provided that the epochs are referred to the respective time instants. By letting $Y^{n+1} = N^{n+1} - X^n$, Eq. (3.3) becomes

$$Y^{n+1} = [Y^n + X^{n-1} - m]^+. \tag{3.4}$$

Because the cycle time is a stationary and ergodic sequence, and the arrival process is Poisson and independent of the cycle time, X^n is a stationary and ergodic sequence, thus fulfilling the stationarity requirement for Loynes' theorem. The queue is, therefore, stable if $E(X^{n-1} - m) < 0$ and unstable if $E(X^{n-1} - m) > 0$. Finally, we have $E(X^{n-1}) = \lambda EC$ to complete the proof. □

In addition to Theorem 1, we also need an appropriate dominant system, serving as an auxiliary system, for deriving stability conditions for, say, a target queue $q_t \in Q$, which is either a GSA queue or a GSD queue. The dominant system *dominates* the original system in the sense that the states (including the queue length) in the dominant system are stochastically greater than those in the original system, if both systems are started with identical initial states. In this dominant system, the queues are classified into either *persistent queues* or *nonpersistent queues*. A persistent queue in the dominant system always generates enough dummy customers, so that the server serves the queue to the maximum limit. As we shall see in Section 3.2, q_t and the queues that are less stable than q_t are persistent, and the queues that are more stable than q_t are nonpersistent. To construct the dominant system, we need the result of queue stability ordering. Therefore, we first define stability ordering, and note several stability ordering results in Section 3.1. By applying Theorem 1 to the target queue, we then obtain q_t's stability condition in the dominant system (Lemma 3). Finally, we prove that the stability condition obtained for the dominant system holds also for the original system (Lemma 4). Lemma 4, therefore, completes the derivation of q_t's stability condition for a given queue stability ordering.

3.1. Stability ordering

A *stability ordering* specifies the order of queues becoming unstable if the system traffic increases according to a certain pattern. Without loss of generality, we consider a linear increase in the system traffic; that is, we represent the traffic vectors by parametric equations: $\lambda_i = r_i\lambda$, $i = 1, \ldots, |Q|$, where $r_i \geq 0 \ \forall i$ and $\lambda \geq 0$. Let $\boldsymbol{R}$ be the set of $(r_1, r_2, \ldots, r_{|Q|})$. If λ increases according to a given $\boldsymbol{r}_o \in \boldsymbol{R}$, one of the following three possible outcomes will occur: (1) q_i is *more stable than* q_j; (2) q_i is *as stable as* q_j, and (3) q_i is *less stable than* q_j. Items (1)–(3) are denoted by $q_i \succ q_j$, $q_i = q_j$, and $q_i \prec q_j$, respectively. Note that the operator $\succeq$ on Q, *at least as stable as*, is a partial ordering; that is, the following three properties hold: (1) $q_i \succeq q_i$; (2) if $q_i \succeq q_j$ and $q_j \succeq q_k$, then $q_i \succeq q_k$, and; (3) if $q_i \succeq q_j$ and $q_j \succeq q_i$, then $q_i = q_j$.

We shall show in Theorem 2 and Corollaries 1 and 2 of Section 4 that stabilities of any two queues in the system, whether they be GSA or GSD queues, can be compared solely based on their λ/m. This result thus enables us to compute stability ordering, and to partition the entire parameter space into regions, each of which corresponds to a unique stability ordering. Nevertheless, for the purpose of computing q_t's stability conditions, we need only regions that give the same set of queues that are more stable than q_t. To be precise, we define such a region by $\boldsymbol{\Gamma}_o = \boldsymbol{\Gamma}(q_t, \mathcal{M}_o, \mathcal{L}_o) = \{\boldsymbol{r} \in \boldsymbol{R} | \mathcal{M}(q_t, \boldsymbol{r}) = \mathcal{M}_o, \mathcal{L}(q_t, \boldsymbol{r}) = \mathcal{L}_o\}$, for which $\mathcal{M}(q_t, \boldsymbol{r}_o) = \{q_i \in Q | q_i \succ q_t \text{ for a given } \boldsymbol{r}_o \in \boldsymbol{R}\}$, $\mathcal{L}(q_t, \boldsymbol{r}_o) = \{q_i \in Q - \{q_t\} | q_t \succeq q_i \text{ for a given } \boldsymbol{r}_o \in \boldsymbol{R}\}$, and $\mathcal{M}_o$ and $\mathcal{L}_o$ are assumed given. In other words, the stability orderings within $\mathcal{M}_o$ and $\mathcal{L}_o$ are immaterial to computing q_t's stability conditions in $\boldsymbol{\Gamma}_o$. To determine q_t's stability region, we, therefore, first obtain stability region for a given $\boldsymbol{\Gamma}_o$; we then take a union of stability regions obtained for all possible $\boldsymbol{\Gamma}_o$'s.

Before leaving this section, we find it helpful to note a number of important points concerning stability ordering. We first let $Q_\infty \subseteq Q$ be a set of queues employing unlimited service policy ($m_i = \infty$), and other queues with finite limits. The first point is that $q_i \succ q_j$ for $q_i \in Q_\infty$ and $q_j \in Q - Q_\infty$ because, for

any given λ_j/m_j, we can always find a m_i that is large enough to ensure $(\lambda_i/m_i) < (\lambda_j/m_j)$. Second, $q_i = q_j$ for $q_i, q_j \in Q_\infty$; that is, queues equipped with unlimited policy are as stable as one another, simply because $\lim_{m_i\to\infty}(\lambda_i/m_i) = \lim_{m_j\to\infty}(\lambda_j/m_j) = 0$. Another explanation for the stability result is that neither q_i nor q_j will become unstable before the other; this statement can be illustrated by two examples. In the first example, a $q_i \in Q_\infty$ becomes unstable ($\rho_i > 1$). This queue, therefore, monopolizes the entire service; and other queues, with unlimited or limited policies, will also become unstable; an unstable $q_j \in Q - Q_\infty$, on the other hand, will not cause the queues in Q_∞ unstable. As a result, the queues in Q_∞ always become stable (or unstable) *at the same time*. The second example, given in Ref. [15], is more subtle than the first one. The system considered there consists of two queues with unlimited policies; and $\rho_i < 1$, $i = 1, 2$, but $\rho_1 + \rho_2 > 1$. Under this situation, *both* queues cause the system unstable. Each queue length returns to zero at the end of the visit (due to the unlimited policy), thus causing an oscillation of the queue lengths at the departure instants of the server; that is, q_1 has an empty queue while q_2 builds up a long queue, and the situation is reversed when switching to another queue. Both queue lengths at the arrival instants of the server, however, continue to build up; as a result, both queues become unstable at the same time. In general, queues employing unlimited policies are always as stable as one another, independent of other factors.

3.2. Dominant systems and queue stability

Having defined quantities related to queue stability ordering, in this section we consider the polling model for a given $\boldsymbol{\Gamma}_o = (q_t, \mathcal{M}_o, \mathcal{L}_o)$, denoted by $\Xi(\boldsymbol{\Gamma}_o)$, and a dominant system for $\Xi(\boldsymbol{\Gamma}_o)$, denoted by $\Xi^d(\boldsymbol{\Gamma}_o)$. We introduce additional superscript d to the quantities in $\Xi^d(\boldsymbol{\Gamma}_o)$, to distinguish them from those in the original system. Recall from the discussion at the beginning of this section that the queues in $\mathcal{M}_o$ are nonpersistent, and they behave identically in both $\Xi(\boldsymbol{\Gamma}_o)$ and $\Xi^d(\boldsymbol{\Gamma}_o)$. On the other hand, queues belonging to $\mathcal{L}_o \cup \{q_t\}$ are persistent in $\Xi^d(\boldsymbol{\Gamma}_o)$, and they behave differently in the dominant system in two aspects. First, the server always serves m_i customers at a GSA q_i by generating *just enough* dummy customers if the queue is short of customers to reach m_i. The server, similarly, always reserves m_i number of services upon his departure from a GSD q_i by generating *just enough* dummy customers. To be precise, upon departure from a GSD $q_i \in \mathcal{L}_o \cup \{q_t\}$ with $N_i^n < m_i$, both queue length and number of reserved services are inflated to m_i, $N_i^n = G_i^n = m_i$; on the other hand, N_i^n and G_i^n remain unchanged for $N_i^n \geq m_i$. Furthermore, we assume that both the initial queue length and the initial number of reserved services for a GSD q_i are at least m_i in $\Xi^d(\boldsymbol{\Gamma}_o)$. Second, the set-up time and walk time for a persistent queue q_i are given by U_i^* and V_i^*, respectively.

In Lemma 2 we first consider the mean cycle time for $\Xi(\boldsymbol{\Gamma}_o)$ and $\Xi^d(\boldsymbol{\Gamma}_o)$. We use EC to denote the former if all queues are stable, and $EC^d(\boldsymbol{\Gamma}_o)$ to denote the latter if the queues in $\mathcal{M}_o$ are stable. Then in Lemma 3 we use the approach in Ref. [7] to derive q_t's stability condition under $\Xi^d(\boldsymbol{\Gamma}_o)$. After that, in Lemma 4 we apply a standard argument for stochastic comparison, and a proposition in Ref. [15] to prove that q_t's stability conditions are the same under both $\Xi^d(\boldsymbol{\Gamma}_o)$ and $\Xi(\boldsymbol{\Gamma}_o)$. We adopt the following new notations for the lemmas: $\hat{\Phi}_i^k$ and $\tilde{\Phi}_i^k$ are the states of q_i at the kth arrival instant of the server and kth departure instant of the server, respectively. Relating back to our previous notations, we have $\hat{\Phi}_i^k \equiv \Phi_i^{(k-1)|Q|+i}$ for the GSA queues and $\tilde{\Phi}_i^k \equiv \Phi_i^{(k-1)|Q|+i}$ for the GSD queues. Moreover, we denote the mean set-up time and mean walk time for a stable q_i by $\overline{u}_i$ and $\overline{v}_i$, respectively.

Lemma 2. *Given that all queues in the polling model are stable, the mean cycle time for $\Xi(\boldsymbol{\Gamma}_o)$, with any given $\boldsymbol{\Gamma}_o$, is given by*

$$EC = \frac{\sum_{i=1}^{|Q|}(\overline{u}_i + \overline{v}_i)}{1-\rho}. \tag{3.5}$$

Given that all queues in a given $\mathcal{M}_o$ are stable, the mean cycle time for $\Xi^d(\boldsymbol{\Gamma}_o)$ is given by

$$EC^d(\boldsymbol{\Gamma}_o) = \frac{\sum_{q_i\in\mathcal{M}_o}(\overline{u}_i + \overline{v}_i) + \sum_{q_i\in\mathcal{L}_o\cup\{q_t\}}(u_i^* + v_i^* + m_i b_i)}{1-\sum_{q_i\in\mathcal{M}_o}\rho_i}. \tag{3.6}$$

Proof. We first consider the mean cycle time for $\Xi(\boldsymbol{\Gamma}_o)$, in which all queues are assumed to be stable. Eq. (3.7) gives the length of the kth cycle time referenced at q_1, and the change in the queue length between the kth and $(k+1)$th arrival instants of the server at q_1:

$$\begin{aligned} &C^k(1) = T_{n'+|Q|} - T_{n'} = \textstyle\sum_{\ell=n'}^{n'+|Q|-1}\Psi_\ell = \sum_{i=1}^{|Q|}(U_i^k + \Theta_i^k + V_i^k);\\ &N_i^{n'+|Q|} - N_i^{n'} = A_i(C^k(1)) - F_i^k, \quad i = 1,\ldots,|Q|, \end{aligned} \tag{3.7}$$

where $n' = (k-1)|Q|+1$. Taking expectation of Eq. (3.7) gives Eq. (3.8). We also substitute $E[U_i^k]$ by $E[U_i(\hat{\Phi}_i^k)]$, and similarly for other quantities, to emphasize their dependencies on the states of the queues:

$$\begin{aligned} &E[C^k(1)] = \textstyle\sum_{i=1}^{|Q|}(E[U_i(\hat{\Phi}_i^k)] + E[\Theta_i(\hat{\Phi}_i^k)] + E[V_i(\tilde{\Phi}_i^k)]);\\ &E[N_i^{n'+|Q|}] - E[N_i^{n'}] = E[A_i(C^k(1))] - E[F_i(\hat{\Phi}_i^k)], \quad i = 1,\ldots,|Q|. \end{aligned} \tag{3.8}$$

If all queues are stable, according to an isolation lemma (Lemma 5 in Ref. [7]), the joint queue length process is substable. Furthermore, because G_i^n is bounded, the last statement implies that the Markov chain $\{\boldsymbol{\Phi}_Q^k(1)\}_{k=1}^\infty$ is ergodic, and its limiting distribution is denoted by ϕ. We now start the process $\{\boldsymbol{\Phi}_Q^k(1)\}_{k=1}^\infty$ by ϕ, and it is well-known that the process is stationary and ergodic. As a result, the expected values in Eq. (3.8) no longer depend on the cycle number; therefore, let $\hat{\Phi}_i^k \stackrel{d}{=} \hat{\Phi}_i$, $\tilde{\Phi}_i^k \stackrel{d}{=} \tilde{\Phi}_i$, and $C^k(1) \stackrel{d}{=} C(1)$ for $k \geq 1$. Moreover, $E[N_i^{n'+|Q|}] - E[N_i^{n'}] = 0$ and $E[A_i(C^k(1))] = \lambda_i E[C(1)]$. We thus have $E[F_i(\hat{\Phi}_i)] = \lambda_i E[C(1)]$; by applying Wald's identity, $E[\Theta_i(\hat{\Phi}_i)] = \rho_i E[C(1)]$. As a result, $E[C^k(1)]$ in Eq. (3.8) becomes

$$E[C(1)] = \sum_{i=1}^{|Q|}(E[U_i(\hat{\Phi}_i)] + \rho_i E[C(1)] + E[V_i(\tilde{\Phi}_i)]). \tag{3.9}$$

Using our notations, $E[U_i(\hat{\Phi}_i)] = \overline{u}_i$ and $E[V_i(\tilde{\Phi}_i)] = \overline{v}_i$; as a result, we arrive at Eq. (3.5). It is clear from the derivation that the mean cycle time is independent of the choice of the reference queue, and of the service policy.

We can conduct a similar analysis for $\Xi^d(\boldsymbol{\Gamma}_o)$, in which the queues in $\mathcal{M}_o$ are assumed to be stable. Clearly, the process $\{\boldsymbol{\Phi}_Q^{k,d}(1)\}_{k=1}^\infty$ is also a Markov chain; for q_t and queues in $\mathcal{L}_o$, $E[U_i(\hat{\Phi}_i^{k,d})] + E[\Theta_i(\hat{\Phi}_i^{k,d})] + E[V_i(\tilde{\Phi}_i^{k,d})] = u_i^* + v_i^* + m_i b_i$ for $k \geq 1$. As a result, we consider a reduced system, denoted by $\{\boldsymbol{\Phi}_{\mathcal{M}_o}^{k,d}(1)\}_{k=1}^\infty$, consisting of only the queues in $\mathcal{M}_o$, and we treat the time periods incurred by q_t and queues in $\mathcal{L}_o$ as independent walk times in the reduced system. Note that these new walk times do not affect the Markovian property of the reduced system. Furthermore, by invoking the isolated lemma again, the joint queue length process for $\mathcal{M}_o$ is substable. Consequently, the Markov chain $\{\boldsymbol{\Phi}_{\mathcal{M}_o}^{k,d}(1)\}_{k=1}^\infty$ is ergodic, and has a limiting distribution ϕ. The same arguments used in the last case apply. □

Lemma 3. *q_t is stable in $\Xi^d(\boldsymbol{\Gamma}_o)$ if $\lambda_t EC^d(\boldsymbol{\Gamma}_o) < m_t$, and it is unstable if $\lambda_t EC^d(\boldsymbol{\Gamma}_o) > m_t$.*

Proof. We first consider the case of $\mathcal{M}_o \neq \emptyset$. Assuming the queues in $\mathcal{M}_o$ to be stable guarantees a nonempty stability region for q_t. Without loss of generality, the cycle starts from $q_1 \in \mathcal{M}_o$. Moreover, the rest of the proof assumes that q_1 is a GSA queue, and the proof also applies to a GSD q_1. For a GSA q_t, its queue length process is given by

$$\begin{aligned} N_t^{k+1,d}(1) &= [N_t^{k,d}(1) + \hat{X}_t^{k,d}(1) - m_i]^+ + \tilde{X}_t^{k,d}(1) \\ &\leq \max(N_t^{k,d}(1) + X_t^{k,d}(1) - m_i, X_t^{k,d}(1)), \quad k \geq 1, \end{aligned} \tag{3.10}$$

where $\hat{X}_t^{k,d}(1)$ is the number of arrivals at q_t between the kth arrival instant of the server at q_1, and the kth arrival instant of the server at q_t. $\tilde{X}_t^{k,d}(1)$, on the other hand, is the number of arrivals at q_t between the kth arrival instant of the server at q_t, and the $(k+1)$th arrival instant of the server at q_1. Clearly, $X_t^{k,d}(1) = \hat{X}_t^{k,d}(1) + \tilde{X}_t^{k,d}(1)$. Furthermore, we define another process $\{\mathcal{N}_t^{k,d}\}_{k=1}^{\infty}$ such that $\mathcal{N}_t^{k+1,d} = \max(\mathcal{N}_t^{k,d} + X_t^{k,d}(1) - m_i, X_t^{k,d}(1))$ and $N_t^{1,d}(1) = \mathcal{N}_t^{1,d}$. From the preceding construction, it is easy to see that $\mathcal{N}_t^{k,d} \geq N_t^{k,d}(1)$ for $k \geq 1$. As a result, stability of $\{\mathcal{N}_t^{k,d}\}_{k=1}^{\infty}$ implies stability of our original process $\{N_t^{k,d}(1)\}_{k=1}^{\infty}$; the stability conditions for $\{\mathcal{N}_t^{k,d}\}_{k=1}^{\infty}$ can be obtained directly from Theorem 1, provided that the cycle time process is stationary and ergodic.

We next consider a GSD q_t and its queue length process is given by

$$\begin{aligned} N_t^{k+1,d}(1) &= \max(N_t^{k,d}(1) + \hat{X}_t^{k,d}(1) - m_i, m_i) + \tilde{X}_t^{k,d}(1) \\ &\leq \max(N_t^{k,d}(1) + X_t^{k,d}(1) - m_i, X_t^{k,d}(1) + m_i), \quad k \geq 1, \end{aligned} \tag{3.11}$$

where $\hat{X}_t^{k,d}(1)$, $\tilde{X}_t^{k,d}(1)$, and $X_t^{k,d}(1)$ are defined similarly as before, except that the first two quantities are referenced to the kth departure instant of the server from q_t (instead of arrival instant). Because $N_t^{k,d}(1) \geq m_i$ for $k \geq 1$ (q_t is a persistent queue), let $\overline{N}_t^{k,d}(1) = N_t^{k,d}(1) - m_i$ and Eq. (3.11) becomes

$$\overline{N}_t^{k+1,d}(1) \leq \max(\overline{N}_t^{k,d}(1) + X_t^{k,d}(1) - m_i, X_t^{k,d}(1)). \tag{3.12}$$

Following the approach for the GSA queues, we define a new process $\{\mathcal{N}_t^{k,d}\}_{k=1}^{\infty}$ with $N_t^{1,d}(1) = \mathcal{N}_t^{1,d}$; similar to the GSA queues, stability of $\{\mathcal{N}_t^{k,d}\}_{k=1}^{\infty}$ implies stability of $\{N_t^{k,d}(1)\}_{k=1}^{\infty}$. Stability conditions for $\{\mathcal{N}_t^{k,d}\}_{k=1}^{\infty}$ also can be obtained directly from Theorem 1, provided that the cycle time process is stationary and ergodic. To show the stationarity and ergodicity of the cycle time process, we note from the proof of Lemma 2 that the Markov chain $\{\boldsymbol{\Phi}_{\mathcal{M}_o}^{k,d}(1)\}_{k=1}^{\infty}$ is ergodic, and has a limiting distribution ϕ. By starting the system with ϕ, $\{\boldsymbol{\Phi}_{\mathcal{M}_o}^{k,d}(1)\}_{k=1}^{\infty}$ is a stationary and ergodic process; consequently, the cycle time process is stationary and ergodic (see Ref. [7] for the details). The preceding analysis applies also to the case of $\mathcal{M}_o = \emptyset$; the cycle time is given by $\sum_{i=1}^{|Q|}(U_i^* + \Theta_i(m_i) + V_i^*)$, and is stationary and ergodic. □

Lemma 4. *The stability conditions in Lemma* 3 *hold for $\Xi(\boldsymbol{\Gamma}_o)$ also.*

Proof. Sufficiency (stable q_t in $\Xi^d(\boldsymbol{\Gamma}_o) \Rightarrow$ stable q_t in $\Xi(\boldsymbol{\Gamma}_o)$): We prove the sufficiency part by showing that $\boldsymbol{\Phi}_Q^{k,d}(1) \geq_{st} \boldsymbol{\Phi}_Q^k(1), k = 1, 2, \ldots$, provided that both systems are under identical initial states; therefore, it is sufficient to prove by induction that $\boldsymbol{\Phi}_Q^{n,d} \geq_{st} \boldsymbol{\Phi}_Q^n$ for $n \geq 1$. Thus, we assume that $\boldsymbol{\Phi}_Q^{n,d} \geq_{st} \boldsymbol{\Phi}_Q^n$ for $\ell + 1 - |Q| \leq n \leq \ell$, where $\ell \geq |Q|$, and we set out to prove that $\boldsymbol{\Phi}_Q^{\ell+1,d} \geq_{st} \boldsymbol{\Phi}_Q^{\ell+1}$. As mentioned in Section 2, there are four cases to consider. We consider here only the case that q_{J_ℓ}

is a GSA queue and $q_{J_{\ell+1}}$ is a GSD queue, because this case contains all the elements required for proving similar results for the other three cases. Based on Eq. (2.2), the proof for $\boldsymbol{\Phi}_Q^{\ell+1,\mathrm{d}} \geq_{\mathrm{st}} \boldsymbol{\Phi}_Q^{\ell+1}$ is equivalent to proving three inequalities: (1) $\Psi_\ell^{\mathrm{d}} \geq_{\mathrm{st}} \Psi_\ell$, (2) $[N_{J_\ell}^{\ell,\mathrm{d}} - m_{J_\ell}]^+ \geq_{\mathrm{st}} [N_{J_\ell}^{\ell} - m_{J_\ell}]^+$, and (3) $N_{J_{\ell+1}}^{\ell,\mathrm{d}} - G_{J_{\ell+1}}^{\ell,\mathrm{d}} \geq_{\mathrm{st}} N_{J_{\ell+1}}^{\ell} - G_{J_{\ell+1}}^{\ell}$. The inequality (2) is obviously true because $N_{J_\ell}^{\ell,\mathrm{d}} \geq_{\mathrm{st}} N_{J_\ell}^{\ell}$. Furthermore, $N_{J_\ell}^{\ell,\mathrm{d}} \geq_{\mathrm{st}} N_{J_\ell}^{\ell}$ implies that $\overline{\Theta}_{J_\ell}^{k_\ell,\mathrm{d}} \geq_{\mathrm{st}} \overline{\Theta}_{J_\ell}^{k_\ell}$, because of the stochastic monotonicity property (s.m.p.) of the extended service period. Besides, we denote q_{J_ℓ}'s queue length at the k_ℓth departure instant of the server in $\Xi(\boldsymbol{\Gamma}_{\mathrm{o}})$ by $\tilde{N}_{J_\ell}^{k_\ell}$, which is given by

$$\tilde{N}_{J_\ell}^{k_\ell} = [N_{J_\ell}^{\ell} - m_{J_\ell}]^+ + A_{J_\ell}(\overline{\Theta}_{J_\ell}^{k_\ell}). \tag{3.13}$$

A similar expression for $\tilde{N}_{J_\ell}^{k_\ell,\mathrm{d}}$ can also be obtained; moreover, $\tilde{N}_{J_\ell}^{k_\ell,\mathrm{d}} \geq_{\mathrm{st}} \tilde{N}_{J_\ell}^{k_\ell}$ because $N_{J_\ell}^{\ell,\mathrm{d}} \geq_{\mathrm{st}} N_{J_\ell}^{\ell}$ and $\overline{\Theta}_{J_\ell}^{k_\ell,\mathrm{d}} \geq_{\mathrm{st}} \overline{\Theta}_{J_\ell}^{k_\ell}$. Finally, $\tilde{N}_{J_\ell}^{k_\ell,\mathrm{d}} \geq_{\mathrm{st}} \tilde{N}_{J_\ell}^{k_\ell}$ implies that $V_{J_\ell}^{k_\ell,\mathrm{d}} \geq_{\mathrm{st}} V_{J_\ell}^{k_\ell}$, because of the s.m.p. of the walk time process.

We now turn to $q_{J_{\ell+1}}$, and denote $q_{J_{\ell+1}}$'s state at the $k_{\ell+1}$th arrival instant of the server in $\Xi(\boldsymbol{\Gamma}_{\mathrm{o}})$ by $(\hat{N}_{J_{\ell+1}}^{k_{\ell+1}}, \hat{G}_{J_{\ell+1}}^{k_{\ell+1}})$, where

$$\hat{N}_{J_{\ell+1}}^{k_{\ell+1}} = N_{J_{\ell+1}}^{\ell} + A_{J_\ell}(\overline{\Theta}_{J_\ell}^{k_\ell} + V_{J_\ell}^{k_\ell}). \tag{3.14}$$

A similar expression for $\hat{N}_{J_{\ell+1}}^{k_{\ell+1},\mathrm{d}}$ can also be obtained; moreover, the previous monotonicity results for q_{J_ℓ} imply that $\hat{N}_{J_{\ell+1}}^{k_{\ell+1},\mathrm{d}} \geq_{\mathrm{st}} \hat{N}_{J_{\ell+1}}^{k_{\ell+1}}$. Furthermore, the induction hypothesis $G_{J_{\ell+1}}^{\ell,\mathrm{d}} \geq_{\mathrm{st}} G_{J_{\ell+1}}^{\ell}$ yields $\hat{G}_{J_{\ell+1}}^{k_{\ell+1},\mathrm{d}} \geq_{\mathrm{st}} \hat{G}_{J_{\ell+1}}^{k_{\ell+1}}$, because the numbers of reserved services are the same at both epochs. As a result, $\overline{\Theta}_{\ell+1}^{k_{\ell+1},\mathrm{d}} \geq_{\mathrm{st}} \overline{\Theta}_{\ell+1}^{k_{\ell+1}}$ because of the s.m.p. of the extended service period; this inequality, with the earlier results for the GSA queue, proves inequality (1).

To prove inequality (3), we first note that $N_{J_{\ell+1}}^{\ell} = N_{J_{\ell+1}}^{\ell+1-|Q|} + \overline{X}_{J_{\ell+1}}^{k_\ell}$, where $\overline{X}_{J_{\ell+1}}^{k_\ell}$ is the number of arrivals at $q_{J_{\ell+1}}$ between $T_{\ell+1-|Q|}$ and T_ℓ, and $G_{J_{\ell+1}}^{\ell} = G_{J_{\ell+1}}^{\ell+1-|Q|}$. Therefore

$$N_{J_{\ell+1}}^{\ell} - G_{J_{\ell+1}}^{\ell} = (N_{J_{\ell+1}}^{\ell+1-|Q|} - G_{J_{\ell+1}}^{\ell+1-|Q|}) + \overline{X}_{J_{\ell+1}}^{k_\ell}. \tag{3.15}$$

A similar expression can also be obtained for the dominant system. We are now ready to show that $N_{J_{\ell+1}}^{\ell+1-|Q|,\mathrm{d}} - G_{J_{\ell+1}}^{\ell+1-|Q|,\mathrm{d}} \geq_{\mathrm{st}} N_{J_{\ell+1}}^{\ell+1-|Q|} - G_{J_{\ell+1}}^{\ell+1-|Q|}$ and $\overline{X}_{J_{\ell+1}}^{k_\ell,\mathrm{d}} \geq_{\mathrm{st}} \overline{X}_{J_{\ell+1}}^{k_\ell}$. For the first inequality, we note that $G_{J_{\ell+1}}^{\ell,\mathrm{d}} \geq_{\mathrm{st}} G_{J_{\ell+1}}^{\ell}$ implies that $N_{J_{\ell+1}}^{\ell+1-|Q|,\mathrm{d}} \geq_{\mathrm{st}} N_{J_{\ell+1}}^{\ell+1-|Q|}$, because of the monotonicity property of the service policy. The first inequality, therefore, holds because $N_{J_{\ell+1}}^{\ell+1-|Q|,\mathrm{d}} \geq_{\mathrm{st}} N_{J_{\ell+1}}^{\ell+1-|Q|}$ and the contractive property of the service policy; the second inequality holds because $\boldsymbol{\Phi}_Q^{n,\mathrm{d}} \geq_{\mathrm{st}} \boldsymbol{\Phi}_Q^{n}$, $\ell + 1 - |Q| \leq n \leq \ell$, implies that $T_{\ell+1-|Q|}^{d} - T_\ell^{d} \geq_{\mathrm{st}} T_{\ell+1-|Q|} - T_\ell$.

To complete the proof, we use induction to prove that $\boldsymbol{\Phi}_Q^{n,\mathrm{d}} \geq_{\mathrm{st}} \boldsymbol{\Phi}_Q^{n}$ for $1 \leq n \leq |Q|$. Assuming that $\boldsymbol{\Phi}_Q^{n,\mathrm{d}} \geq_{\mathrm{st}} \boldsymbol{\Phi}_Q^{n}$ for $1 \leq n \leq \ell$, where $\ell < |Q|$, we set out to prove that $\boldsymbol{\Phi}_Q^{\ell+1,\mathrm{d}} \geq_{\mathrm{st}} \boldsymbol{\Phi}_Q^{\ell+1}$. The proof is similar to the previous case, and is therefore skipped.

Necessity (unstable q_{t} in $\Xi^{\mathrm{d}}(\boldsymbol{\Gamma}_{\mathrm{o}}) \Rightarrow$ unstable q_{t} in $\Xi(\boldsymbol{\Gamma}_{\mathrm{o}})$): We consider $\Xi^{\mathrm{d}}(\boldsymbol{\Gamma}_{\mathrm{o}})$ in which the queues in $\mathcal{M}_{\mathrm{o}}$ are assumed to be stable. According to the previous analysis in the proof of Lemma 2, the Markov chain $\{\boldsymbol{\Phi}_{\mathcal{M}_{\mathrm{o}}}^{k,\mathrm{d}}(1)\}_{k=1}^{\infty}$ is ergodic; consequently, the cycle time process is stationary and ergodic. By setting $\lambda_{\mathrm{t}} > m_{\mathrm{t}}/EC^{\mathrm{d}}(\boldsymbol{\Gamma}_{\mathrm{o}})$, we know from Lemma 3 that q_{t} is unstable in $\Xi^{\mathrm{d}}(\boldsymbol{\Gamma}_{\mathrm{o}})$ and, due to the stability

ordering, the queues in $\mathcal{L}_o$ are unstable as well; that is, $\lim_{k\to\infty} N_i^{k,d}(1) = \infty$ for $q_i \in \mathcal{L}_o \cup \{q_t\}$. We now apply Proposition 4 in Ref. [15] to $\{\boldsymbol{\Phi}_Q^{k,d}(1)\}_{k=1}^{\infty}$ with modifications to suit our polling model, and we present the modified proposition in Proposition 1. □

Proposition 1. *We consider the Markov chain* $\{\boldsymbol{\Phi}_Q^{k,d}(1)\}_{k=1}^{\infty}$. *Assume that it is known that if the process starts from state* $\boldsymbol{u}$, *then for all* $q_i \in \mathcal{L}_o \cup \{q_t\}$, $\lim_{k\to\infty} N_i^{k,d}(1) = \infty$. *Then, given any bounded one-dimensional set A, there is a state* $\boldsymbol{c}$ *such that* $c_i \notin A$ *for all* $q_i \in \mathcal{L}_o \cup \{q_t\}$, *and* $\Pr\{N_i^{k,d}(1) \notin A, q_i \in \mathcal{L}_o \cup \{q_t\}, k \geq 1 | \boldsymbol{\Phi}_Q^{1,d}(1) = \boldsymbol{c}\} > 0$.

We set A to $[0, \max_{q_i \in \mathcal{L}_o \cup \{q_t\}} d_i]$, where $d_i = \max(x_i^*, y_i^* + m_i)$. From Section 2.1, x_i^* and y_i^* are defined for a GSA q_i such that $U_i(x) \stackrel{d}{=} U_i^*$ for $x \geq x_i^*$, and $V_i(y) \stackrel{d}{=} V_i^*$ for $y \geq y_i^*$. Similarly for a GSD q_i, they are defined such that $U_i(x_1, x_2) \stackrel{d}{=} U_i^*$ for $(x_1, x_2) \geq (x_i^*, m_i)$, and $V_i(y) \stackrel{d}{=} V_i^*$ for $y \geq y_i^*$, where x_1 and x_2 are the queue length and number of reserved services at the server arrival instants, respectively. By applying Proposition 1 to our model, we conclude that there is a set of sample paths of positive probability for which $N_i^{k,d}(1) > d_i$, $k = 1, 2, \ldots$. As a result, on this set of sample paths the queues in $\Xi^d(\boldsymbol{\Gamma}_o)$ and $\Xi(\boldsymbol{\Gamma}_o)$ are identical; that is, the queues in $\mathcal{L}_o \cup \{q_t\}$ are unstable under $\Xi(\boldsymbol{\Gamma}_o)$ also.

4. Computing the queue stability ordering

In the last section we assume a given queue stability ordering; in this section we present in Corollary 2 a necessary and sufficient condition for comparing stabilities of any two queues in the system. To prove the queue stability result, we first present in Theorem 2 a condition under which two queues in the system are as stable as each other. Consequently, we obtain in Corollary 2 a condition, under which all queues are as stable as one another; we refer the system to be in a *load-balanced* state.

Theorem 2. *Consider any two queues* $q_i, q_j \in Q$; q_i *and* q_j *are as stable as each other,* $q_i = q_j$, *for any* $\boldsymbol{r} \in \boldsymbol{R}$ *for which* $(r_i/m_i) = (r_j/m_j)$ (*or* $(\lambda_i/m_i) = (\lambda_j/m_j)$).

Proof. Let us start with the notation $\boldsymbol{R}^{i,j} = \{\boldsymbol{r} \in \boldsymbol{R} | (r_i/m_i) = (r_j/m_j)\}$. In the following we first prove that there exists at least an $\boldsymbol{r}$ that gives $q_i = q_j$ (existence); we then show that those $\boldsymbol{r}$'s that give $q_i = q_j$ must belong to $\boldsymbol{R}^{i,j}$ (exclusiveness).

Existence. We prove this part using the argument of contradiction by assuming that such an $\boldsymbol{r}$ does not exist; that is, the stability boundaries of q_i and q_j do not intersect. This assumption, therefore, implies that q_i's stability region is a subset (excluding any overlaps of the two stability boundaries) of q_j's, or vice versa, because the stability regions of both queues should be bounded and closed. As a result, one queue is always more stable than the other for any $\boldsymbol{r} \in \boldsymbol{R}$; this conclusion is obviously invalid for any limited policies, thus contradicting the assumption that the $\boldsymbol{r}$ that gives $q_i = q_j$ does not exist.

Exclusiveness. We assume that there is an $\boldsymbol{r}_o$ that gives $q_i = q_j$ but it does not belong to $\boldsymbol{R}^{i,j}$, and we shall show that this $\boldsymbol{r}_o$ does not exist. Towards this end, we consider a subset of $\boldsymbol{R}$: $\boldsymbol{R}(i, j) = \{\boldsymbol{r} \in \boldsymbol{R} | r_k = c_k \text{ for } k \neq i, j\}$, where c_k is a constant; that is, we fix the arrival rates to q_k, $k \neq i, j$. We then proceed to find the instability regions for q_i and q_j in $\boldsymbol{R}(i, j)$ by computing the instability condition of the more stable queue in two regions: $q_i \succeq q_j$ and $q_j \succeq q_i$, denoted by $\boldsymbol{R}^i(i, j)$ and $\boldsymbol{R}^j(i, j)$, respectively. Clearly, $\boldsymbol{R}(i, j) = \boldsymbol{R}^i(i, j) \cup \boldsymbol{R}^j(i, j)$.

Let us first consider the case of $q_i \succeq q_j$. Because the arrival rates to q_k, $k \neq i, j$, are fixed, stabilities of both q_i and q_j in this subspace are affected only by r_i and r_j. Furthermore, by definition, the stabilities

of the queues that are at least as stable as q_i are not affected by r_j; as a result, $\mathcal{M}(q_i, \boldsymbol{r})$ and $\mathcal{L}(q_i, \boldsymbol{r})$ are invariant for any $\boldsymbol{r} \in \boldsymbol{R}^i(i, j)$, and we denote this set of stability orderings by $\boldsymbol{\Gamma}_o = (q_i, \mathcal{M}_o, \mathcal{L}_o)$. By applying Lemmas 2–4 to q_i, we thus obtain instability conditions for q_i and q_j for the first case in Eqs. (4.1) and (4.2).

$$\lambda_i > \frac{m_i}{EC^d(\boldsymbol{\Gamma}_o)} \quad \text{for } \boldsymbol{r} \in \boldsymbol{R}^i(i, j), \tag{4.1}$$

where

$$EC^d(\boldsymbol{\Gamma}_o) = \frac{\sum_{q_k \in \mathcal{M}_o}(\overline{u}_k + \overline{v}_k) + \sum_{q_k \in \mathcal{L}_o \cup \{q_i, q_j\}}(u_k^* + v_k^* + m_k b_k)}{1 - \sum_{q_k \in \mathcal{M}_o} \rho_k}. \tag{4.2}$$

Because the cases of $q_i \succeq q_j$ and $q_j \succeq q_i$ are symmetrical, we observe that the mean cycle time for the case of $q_j \succeq q_i$ is given by Eq. (4.2) also. As a result, we obtain the overall instability conditions for q_i and q_j in $\boldsymbol{R}(i, j)$:

$$\lambda_i > \frac{m_i}{EC^d(\boldsymbol{\Gamma}_o)} \quad \text{for } \boldsymbol{r} \in \boldsymbol{R}(i, j) \quad \text{and} \quad \lambda_j > \frac{m_j}{EC^d(\boldsymbol{\Gamma}_o)} \quad \text{for } \boldsymbol{r} \in \boldsymbol{R}(i, j). \tag{4.3}$$

The earlier assumption about $\boldsymbol{r}_o$ and Eq. (4.3) implies that q_i always becomes unstable at $\lambda_i = m_i / EC^d(\boldsymbol{\Gamma}_o)$ for any $\boldsymbol{r}_o \in \boldsymbol{R}^i(i, j)$; the conclusion is obviously invalid, thus contradicting the assumption that the $\boldsymbol{r}_o$ does not belong to $\boldsymbol{R}^{i,j}$. The same contradiction can be said also for any $\boldsymbol{r}_o \in \boldsymbol{R}^j(i, j)$. Lastly, note that the conclusion is independent of the subset $\boldsymbol{R}(i, j)$. □

Corollary 1. *All queues are as stable as one another, $q_1 = q_2 = \cdots = q_{|Q|}$, for any $\boldsymbol{r} \in \boldsymbol{R}$ for which $(\lambda_1/m_1) = (\lambda_2/m_2) = \cdots = (\lambda_{|Q|}/m_{|Q|})$.*

Proof. The proof is straightforward, and is therefore omitted. □

Corollary 2. *For any $\boldsymbol{r} \in \boldsymbol{R}$, $q_i \prec (\succ) q_j$ iff $(\lambda_i/m_i) > (<)(\lambda_j/m_j)$.*

Proof. We again consider $\boldsymbol{R}(i, j)$, in particular, the subset that satisfies $(r_i/m_i) > (r_j/m_j)$. First, we observe that there is at least an $\boldsymbol{r}$ in this subset that gives $q_i \prec q_j$; the obvious one is given by $r_i \neq 0$ and $r_j = 0$. Second, we argue that in this subset either $q_i \prec q_j$ or $q_i \succ q_j$ is true; otherwise, the two stability boundaries should have at least one intersection in the subset. The latter statement implies that there is at least an $\boldsymbol{r}$ in this subset that yields $q_i = q_j$, but we know from the proof for Theorem 2 that this result cannot be true. Combining the two items above completes the proof. □

5. Stability results

We are now ready to present the complete stability conditions for our polling model, and also for several special cases and a pipeline polling system. By combining the results in Sections 3 and 4, we obtain in Theorem 3 stability conditions for individual queues in the polling model. We then show that the stability results yield closed-form stability conditions for a number of special cases. For example, in Corollary 3 we first consider system stability conditions for our model, which generalize the previous results obtained by Kuehn [17] and later proved rigorously by Georgiadis and Szpankowski [7] for the GSA policy. In Corollary 4 we obtain closed-form queue stability conditions if the walk time and set-up time are independent, and the results are identical to the sufficient conditions obtained by Ibe and Cheng

for the GSA policy [8]. In Corollary 5 we obtain the well-known stability condition for a class of queues employing unlimited service policies. Because those unlimited queues are as stable as one another, the stability condition in Corollary 5 serves as the system stability condition for the set of queues. Finally, in Section 5.1 we apply the queue stability analysis to a pipeline polling system.

Although our proofs do not cover stability boundaries, previous analyses of similar systems indicated that the stability boundaries usually fall into the instability region. An evidence for supporting this proposition comes from the pseudo-conservation laws for polling models, which give exact expressions for a weighted sum of mean waiting times [19,20]. The pseudo-conservation laws, therefore, immediately yield mean waiting times for symmetric systems, which consist of identical queues. The mean waiting times obtained for both unlimited and limited policies are indeed unbounded at the stability boundaries. Furthermore, if one or more queues operate at their stability boundaries in our polling model, the Markov chain $\{\boldsymbol{\Phi}_Q^k(j)\}_{k=1}^{\infty}$ is believed to be *recurrent null*, because it is still possible for the system to reach the *null state* (when all queues are empty) but the mean recurrence time is unbounded.

Theorem 3. *For a given* $\boldsymbol{\Gamma}_o = \boldsymbol{\Gamma}(q_t, \mathcal{M}_o, \mathcal{L}_o)$, q_t *is stable if Eq.* (5.1) *holds; otherwise, it is unstable with possible exception of the boundaries*.

$$\lambda_t < \frac{m_t}{\sum_{q_i \in \mathcal{M}_o} (\overline{u}_i + \overline{v}_i) + \sum_{q_i \in \mathcal{L}_o \cup \{q_t\}} (u_i^* + v_i^* + m_i b_i)} \left(1 - \sum_{q_i \in \mathcal{M}_o} \rho_i \right). \tag{5.1}$$

The entire stability region for q_t *is given by the union of stability regions obtained for all possible* $\boldsymbol{\Gamma}_o$*'s, which can be obtained from Corollary* 2.

Proof. By combining Lemmas 2–4. □

In Fig. 1 we show the parameter space partitioned into six regions, each of which is associated with a unique stability ordering. For example, the stability ordering for region II is given by $q_3 \succ q_2 \succ q_1$; and

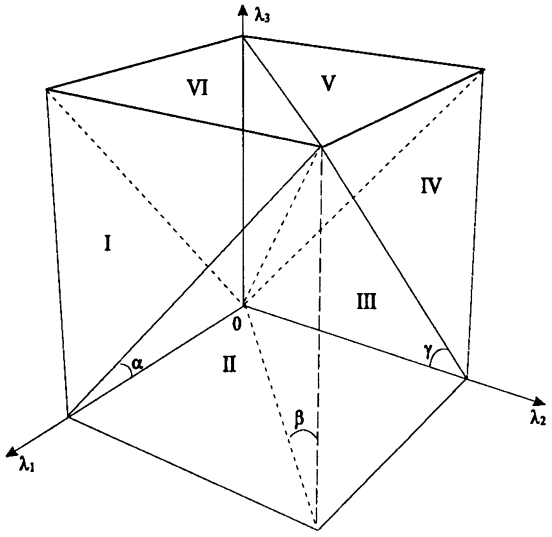


Fig. 1. Parameter regions corresponding to six queue stability orderings for a three-queue system ($\alpha : (r_2/m_2) = (r_3/m_3)$; $\beta : (r_1/m_1) = (r_2/m_2)$; $\gamma : (r_1/m_1) = (r_3/m_3)$).

that for region VI, $q_2 \succ q_1 \succ q_3$. Nevertheless, when computing stability conditions for q_1, for example, there are at most four $\boldsymbol{\Gamma}_o$'s to consider, which are given by region I ∪ region II, region III, region VI, and region IV ∪ region V.

Corollary 3. *The polling model is substable if and only if Eq.* (5.2) *holds.*

$$\lambda_i < \frac{m_i}{\sum_{k\neq i}(\overline{u}_k + \overline{v}_k) + u_i^* + v_i^* + m_i b_i}\left(1 - \sum_{k\neq i}\rho_k\right), \quad i = 1, \ldots, |Q|. \tag{5.2}$$

Furthermore, if the walk time and set-up time are independent of each other, and of other processes in the system, then Eq. (5.2) *is reduced to the well-known result stated as follows:*

$$\frac{\lambda_i w}{1-\rho} < m_i, \quad i = 1, \ldots, |Q|, \tag{5.3}$$

where $w = \sum_{i=1}^{|Q|}(u_i^* + v_i^*)$.

Proof. Recall that $\{\boldsymbol{\Phi}_Q^k(1)\}_{k=1}^{\infty}$ is substable if $\{N_i^k(1)\}_{k=1}^{\infty}$ is stable according to Eq. (3.1) for $i = 1, \ldots, |Q|$. One approach of deriving the system stability conditions is, therefore, to find the queue stability condition of a *least stable queue* (LSQ) for every possible $\boldsymbol{\Gamma}_o$. An LSQ is one with a maximum value of λ/m. Thus, the overall system stability region is an union of stability regions obtained for the LSQs in these $\boldsymbol{\Gamma}_o$'s; however, we adopt another approach that avoids an explicit partitioning of the parameter space. We first observe that each queue is an LSQ in a nonempty parameter space; note that this observation is valid for any polling systems that do not employ unlimited service policies. We then derive stability regions for $q_i, i = 1, \ldots, |Q|$, for each of which we assume that q_i is *always* the LSQ by setting $\mathcal{M}_o = Q - \{q_i\}$ and $\mathcal{L}_o = \emptyset$ for all $\boldsymbol{r} \in \boldsymbol{R}$, and we denote the resulted stability region by $\boldsymbol{S}_i$. As a result, the system stability region is given by $\bigcap_{i=1}^{|Q|}\boldsymbol{S}_i$ because, by the definition of LSQ, the operation of set intersection includes only stability regions of the *true* LSQs. Moreover, the inequality for each i in Eq. (5.2) defines the stability region $\boldsymbol{S}_i$. □

Corollary 4. *If the walk time and set-up time are independent of other processes in the system with* $w = \sum_{i=1}^{|Q|}(u_i^* + v_i^*)$, *the queue stability condition in Eq.* (5.1) *is equivalent to*

$$\frac{\lambda_t w}{m_t} + \sum_{i=1}^{|Q|}\min\left(\frac{\lambda_t m_i}{m_t}, \lambda_i\right) b_i < 1. \tag{5.4}$$

Proof. A simple algebraic manipulation of Eq. (5.1) yields Eq. (5.4). □

Corollary 5. *We consider our polling model, in which one or more queues employ unlimited service policy by setting their* m_i *to* ∞ *(the set of unlimited queues is denoted by* $Q_\infty \subseteq Q$*). The necessary and sufficient stability condition for* Q_∞ *is given by*

$$\sum_{q_i \in Q_\infty}\rho_i < 1. \tag{5.5}$$

Proof. We apply the stability conditions in Theorem 3 by setting the limit m to infinity for the unlimited queues. Because the unlimited queues are as stable as one another, we need to ensure identical (λ/m)'s for these queues. We, therefore, set $(\lambda_i/m_i) = (1/t), \forall q_i \in Q_\infty$, where $t > 0$ is a real number, but t does not take on all values because m_i is an integer. Now we apply Theorem 3 to a $q_t \in Q_\infty$ by setting

t to ∞. Note that $\mathcal{M}_o = \emptyset$ for any given $\boldsymbol{\Gamma}_o$ because no other queues are more stable than an unlimited queue; we thus have

$$\lambda_t < \frac{1}{\sum_{q_i \in Q} \lim_{t\to\infty}(u_i^* + v_i^* + m_i b_i)/m_t}. \tag{5.6}$$

All the limits in Eq. (5.6) go to zero except for $\sum_{q_i \in Q_\infty} \lim_{t\to\infty}(m_i b_i/m_t)$, in which the limit for a $q_i \in Q_\infty - \{q_t\}$ is equal to (ρ_i/λ_t); consequently, we arrive at Eq. (5.5). □

5.1. A pipeline polling system

Lastly, we consider a pipeline polling system in this section, motivated by a satellite system based on a polling scheme with reservation [21]. All the assumptions and notations about the arrival and service processes in the pipeline polling system are the same as before, except that the service times here are constants, denoted by b_i for q_i (because the satellite system is a time slotted system). When the server (satellite) is about to leave behind y customers (packets) in q_i (ith earth station), he reserves $g_i(y)$ number of services for his next visit and

$$g_i(y) = \max(1, \min(y, m_i)). \tag{5.7}$$

From Eq. (5.7), the pipeline polling policy is the same as the GSD policy for $y > 0$; however, the server in the pipeline polling system still reserves one service at his departure for $y = 0$. This reserved service will be wasted if the server finds an empty queue again at his next visit; this wasted service time may be treated as set-up time in our model. On the other hand, the server will serve one customer if he finds a nonempty queue next time, and the set-up time is zero in this case. In addition, the server has prior knowledge of the service schedule through the reservation scheme; therefore, it takes no time to switch from one queue to another by polling the next queue while serving the current one. As a result, the set-up time for q_i in the pipeline polling system is given by $U_i(x) = b_i$ if $x = 0$ and $U_i(x) = 0$, otherwise, and the walk time is zero for all queues. Although the pipeline polling policy differs from the GSD policy when the server leaves an empty queue, we could apply the same analysis in Sections 2–4 to this system. First, it is simple to show that this policy also is monotonic and contractive. Second, it is straightforward to show that the two requirements in Section 2 are satisfied by this system, because $\overline{\Theta}_i(x_1, x_2)$ is monotonic in (x_1, x_2) and $\overline{\Theta}_i(x_1, x_2) = m_i b_i$ for $(x_1, x_2) \geq (m_i, m_i)$, where x_1 is the queue length and x_2 the number of reserved services for q_i at the arrival instant of the server. Consequently, Theorem 4 gives the queue stability results for this system; system stability conditions can be obtained using the approach in the proof for Corollary 3, and the results are given in Ref. [4].

Theorem 4. *Consider the pipeline polling system with a given* $\boldsymbol{\Gamma}_o = \boldsymbol{\Gamma}(q_t, \mathcal{M}_o, \mathcal{L}_o)$; q_t *is stable if and only if Eq.* (5.8) *holds.*

$$\lambda_t < \frac{m_t}{\sum_{q_i \in \mathcal{M}_o} b_i + \sum_{q_i \in \mathcal{L}_o \cup \{q_t\}} m_i b_i} \left[1 - \sum_{q_i \in \mathcal{M}_o} \rho_i \left(1 - \frac{1}{\overline{f}_i^{\,d}(\boldsymbol{\Gamma}_o)} \right) \right], \tag{5.8}$$

where $\overline{f}_i^{\,d}(\boldsymbol{\Gamma}_o)$ *is the mean number of customers of* $q_i \in \mathcal{M}_o$ *served in* $\Xi^d(\boldsymbol{\Gamma}_o)$, *given that the server visits a nonempty* q_i. *The entire stability region for* q_t *is given by the union of stability regions obtained*

Table 1
Three queues: $m_i = 6$, $\lambda_2 = 0.33$ (if stable), $\lambda_3 = 0.28$ (if stable)

Stability for q_i, $i \neq 1$		Stability conditions for q_1	
Stable	Unstable	Approximation	Simulation
q_2, q_3		$\lambda_1 < 0.3900$	$\lambda_1 < 0.3899$
q_2	q_3	$\lambda_1 < 0.3350*$	$\lambda_1 < 0.3350$
q_3	q_2	$\lambda_1 < 0.3600*$	$\lambda_1 < 0.3600$
	q_2, q_3	$\lambda_1 < 0.3333*$	$\lambda_1 < 0.3333$

for all possible $\boldsymbol{\Gamma}_o$'s. Moreover, the stability ordering results stated in Theorem 2 *and Corollaries* 1 *and* 2 *apply to this system.*

Proof. The results in Lemmas 3 and 4 apply also to a q_t in the pipeline polling system, of course with a different $EC^d(\boldsymbol{\Gamma}_o)$. To obtain $EC^d(\boldsymbol{\Gamma}_o)$ for the pipeline polling system, let $\pi_{i,0}$ be the steady-state probability that $q_i \in \mathcal{M}_o$ is empty when the server visits the queue. For $q_i \in \mathcal{L}_o \cup \{q_t\}$, $E[U_i(\hat{\Phi}_i^{k,d})] + E[\Theta_i(\hat{\Phi}_i^{k,d})] + E[V_i(\tilde{\Phi}_i^{k,d})] = m_i b_i$ for $k \geq 1$; for $q_i \in \mathcal{M}_o$, $E[U_i(\hat{\Phi}_i)] + E[\Theta_i(\hat{\Phi}_i)] + E[V_i(\tilde{\Phi}_i)] = \pi_{i,0} b_i + \rho_i EC^d(\boldsymbol{\Gamma}_o)$. Therefore, we have

$$EC^d(\boldsymbol{\Gamma}_o) = \frac{\sum_{q_i \in \mathcal{M}_o} \pi_{i,0} b_i + \sum_{q_i \in \mathcal{L}_o \cup \{q_t\}} m_i b_i}{1 - \sum_{q_i \in \mathcal{M}_o} \rho_i}. \tag{5.9}$$

Moreover, $E[F_i(\hat{\Phi}_i)]$, the mean number of customers of $q_i \in \mathcal{M}_o$ served in a cycle, is given by

$$E[F_i(\hat{\Phi}_i)] = (1 - \pi_{i,0}) \overline{f}_i^d(\boldsymbol{\Gamma}_o) = \lambda_i EC^d(\boldsymbol{\Gamma}_o). \tag{5.10}$$

From Eq. (5.10), $\pi_{i,0} = 1 - (\lambda_i EC^d(\boldsymbol{\Gamma}_o))/\overline{f}_i^d(\boldsymbol{\Gamma}_o)$ for $q_i \in \mathcal{M}_o$. By substituting the expression for $\pi_{i,0}$ into Eq. (5.9) and rearranging the terms, we obtain the mean cycle time and, subsequently, the stability conditions from Lemmas 3 and 4. The proofs for the stability ordering results are the same as before. □

We present numerical results for the stability conditions of the pipeline polling system in Tables 1–3, in which q_1 is the target queue in all cases. To make it simple, we also let the service time be one and m_i be six for all queues. We compute q_1's stability conditions using Eq. (5.8) with two methods to estimate

Table 2
Six queues: $m_i = 6$, $\lambda_2 = 0.18$ (if stable), $\lambda_{3-6} = 0.11$ (if stable)

Stability for q_i, $i \neq 1$		Stability conditions for q_1	
Stable	Unstable	Approximation	Simulation
q_{2-6}		$\lambda_1 < 0.3799$	$\lambda_1 < 0.3662$
q_{2-5}	q_6	$\lambda_1 < 0.2450$	$\lambda_1 < 0.2444$
q_{2-4}	q_5, q_6	$\lambda_1 < 0.2000$	$\lambda_1 < 0.1997$
q_{3-6}	q_2	$\lambda_1 < 0.2780$	$\lambda_1 < 0.2785$
q_{3-5}	q_2, q_6	$\lambda_1 < 0.2333$	$\lambda_1 < 0.2231$
q_3, q_4	q_2, q_5, q_6	$\lambda_1 < 0.1950$	$\lambda_1 < 0.1950$
q_3	q_2, q_{4-6}	$\lambda_1 < 0.1780*$	$\lambda_1 < 0.1780$
	q_{2-6}	$\lambda_1 < 0.1666*$	$\lambda_1 < 0.1666$

Table 3
Twelve queues: $m_i = 6$, $\lambda_{2-3} = 5\lambda$ (if stable), $\lambda_{4-12} = 3\lambda$ (if stable), $\lambda = 0.02162$

Stability for $q_i, i \neq 1$		Stability conditions for q_1	
Stable	Unstable	Approximation	Simulation
q_{2-12}		$\lambda_1 < 0.2006$	$\lambda_1 < 0.1904$
q_{2-11}	q_{12}	$\lambda_1 < 0.1325$	$\lambda_1 < 0.1320$
q_{2-10}	$q_{11,12}$	$\lambda_1 < 0.1099$	$\lambda_1 < 0.1099$
$q_{2,4-12}$	q_3	$\lambda_1 < 0.1542$	$\lambda_1 < 0.1528$
$q_{2,4-11}$	$q_{3,12}$	$\lambda_1 < 0.1244$	$\lambda_1 < 0.1242$
$q_{2,4-10}$	$q_{3,11,12}$	$\lambda_1 < 0.1095$	$\lambda_1 < 0.1094$
q_{4-12}	$q_{2,3}$	$\lambda_1 < 0.1388$	$\lambda_1 < 0.1383$
q_{4-11}	$q_{2,3,12}$	$\lambda_1 < 0.1203$	$\lambda_1 < 0.1202$
q_{4-10}	$q_{2,3,11,12}$	$\lambda_1 < 0.1092$	$\lambda_1 < 0.1092$
q_{4-9}	$q_{2,3,10-12}$	$\lambda_1 < 0.1018$	$\lambda_1 < 0.1018$
q_{4-8}	$q_{2,3,9-12}$	$\lambda_1 < 0.0965$	$\lambda_1 < 0.0965$
q_{4-7}	$q_{2,3,8-12}$	$\lambda_1 < 0.0926$	$\lambda_1 < 0.0926$
q_{4-6}	$q_{2,3,7-12}$	$\lambda_1 < 0.0895$	$\lambda_1 < 0.0895$
$q_{4,5}$	$q_{2,3,6-12}$	$\lambda_1 < 0.0870$	$\lambda_1 < 0.0870$
q_4	$q_{2,3,5-12}$	$\lambda_1 < 0.0850*$	$\lambda_1 < 0.0850$
	q_{2-12}	$\lambda_1 < 0.0833*$	$\lambda_1 < 0.0833$

$\overline{f}_i^{\mathrm{d}}(\boldsymbol{\Gamma}_{\mathrm{o}})$ — vacation model (labeled as *Approximation*) and computer simulation (labeled as *Simulation*). We also mark those cases, in which we can obtain exact results, by '$*$' (when there are no more than two stable queues in the system). All the numerical results indicate that the stability conditions obtained from Eq. (5.1) and the vacation model are very accurate.

6. Conclusions and future work

We proposed in this paper a novel approach to the queue stability problem for a fairly general polling model, and we obtained a number of interesting results for the polling model. First, provided with identical arrival processes, service processes, and service limits, the GSA policy is as stable as the GSD policy in the sense that the stability condition for a GSA queue remains the same if the queue switches to the GSD policy. Moreover, we conjecture that the GSA policy gives a smaller average delay than the GSD policy because reservation incurs additional delay; however, when approaching the stability boundary, the difference in the delay performance for the two policies quickly diminishes. Second, as noted from Eq. (5.1) and (5.8), the state-dependent set-up time and walk time *nonlinearize* the stability conditions; for example, the unknown quantity $\overline{f}_i^{\mathrm{d}}(\boldsymbol{\Gamma}_{\mathrm{o}})$ *summarizes* these nonlinear interactions for the pipeline polling system. On the other hand, the stability conditions are analytically computable if the set-up time and walk time are independent. Third, the stability conditions for unlimited policies are independent of the state-dependent set-up time and walk time. The set of unlimited queues (Q_∞) behave essentially like a single queue from the perspective of stability. Moreover, in a system mixed with limited and unlimited policies, the maximum normalized utilization of the server available to the queues in $Q - Q_\infty$ is given by $1 - \sum_{q_i \in Q_\infty} \rho_i$, provided that the queues in Q_∞ are stable.

More importantly, the contribution of this paper goes beyond the polling models considered here. We believe that our approach to the queue stability problem can be applied to analyze other service policies,

variants of polling models, and even general single-resource contention systems, such as buffered Aloha; these results will be forthcoming. In addition, this work can be extended in various directions. We are currently devising a general method to compute queue stability ordering, thus eliminating constructed proofs for different polling models. We are also investigating how other nonPoisson traffic models, such as the self-similar process, affect the system and queue stability conditions. Another important area is to apply these newly obtained queue stability results to the design and analysis of computer and communications systems; potential areas are capacity estimation and a schedulability study of ATM and wireless networks, and formulation of internet pricing models.

Acknowledgements

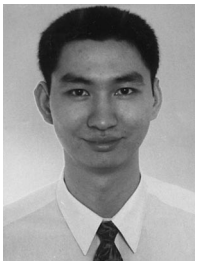
This work was partially supported by The Hong Kong Polytechnic University Central Research Grants 350/492 and 351/690. The authors are grateful to the anonymous reviewers for carefully reading this paper. The authors also thank Prof. Wojciech Szpankowski of Purdue University, Prof. Hideaki Takagi of the University of Tsukuba, and Prof. Hong Chen of The Hong Kong University of Science and Technology for their useful comments in an earlier version of this paper.

References

[1] O.J. Boxma, H. Takagi, Polling Models (special issue), Queueing Systems 11 (1992).
[2] H. Levy, M. Sidi, Polling systems: applications, modeling, and optimization, IEEE Trans. Commun. 38 (1990) 1750–1760.
[3] E. Altman, P. Konstantopoulos, Z. Liu, Stability, monotonicity and invariant quantities in general polling systems, Queueing Systems 11 (1992) 35–57.
[4] K.C. Chang, Stability conditions for a pipeline polling scheme in satellite communications, Queueing Systems 14 (1993) 339–348.
[5] D. Down, On the stability of polling models with multiple servers, J. Appl. Probab. 35 (1998) 925–935.
[6] C. Fricker, M.R. Jaïbi, Monotonicity and stability of periodic polling models, Queueing Systems 15 (1994) 211–238.
[7] L. Georgiadis, W. Szpankowski, Stability of token passing rings, Queueing Systems 11 (1992) 7–33.
[8] O.C. Ibe, X. Cheng, Stability conditions for multiqueue systems with cyclic service, IEEE Trans. Automat. Control 33 (1988) 102–103.
[9] L. Massoulié, Stability of non-Markovian polling systems, Queueing Systems 21 (1995) 67–95.
[10] V. Sharma, Stability and continuity of polling systems, Queueing Systems 16 (1994) 115–137.
[11] L. Georgiadis, W. Szpankowski, L. Tassiulas, A scheduling policy with maximal stability region for ring networks with spatial reuse, Queueing Systems 19 (1995) 131–148.
[12] S. Gorinsky, S. Baruah, T. Marlowe, A. Stoyenko, Exact and efficient analysis of schedulability in fixed-packet networks: a generic approach, Proceedings of the IEEE INFOCOM, 1997, pp. 585–592.
[13] K.C. Chang, A hybrid analytic-simulation approach to compute throughput of FDDI networks, Proceedings of the Communication Networks Modelling and Simulation Conference, La Jolla, CA, 1996, pp. 233–238.
[14] M.J. Fischer, C.M. Harris, J. Xie, An interpolation approximation for expected wait in a time-limited loop system, Technical report, Systems Engineering and Operations Research Department, George Mason University, 1997.
[15] L. Georgiadis, W. Szpankowski, L. Tassiulas, Stability analysis of quota allocation access protocols in ring networks with spatial reuse, IEEE Trans. Inform. Theory 43 (1997) 923–937.
[16] R. Loynes, The stability of a queue with non-independent inter-arrival and service times, Proc. Camb. Philos. 58 (1962) 497–520.
[17] P.J. Kuehn, Multiqueue systems with non-exhaustive cyclic-service, Bell System Tech. J. 58 (1979) 671–698.
[18] H. Levy, M. Sidi, O.J. Boxma, Dominance relations in polling systems, Queueing Systems 6 (1990) 155–172.
[19] O.J. Boxma, B. Meister, Pseudo-conservation laws in cyclic-service systems, J. Appl. Probab. 24 (1987) 949–964.
[20] K.C. Chang, D. Sandhu, Pseudo-conservation laws in cyclic-service systems with a class of limited service policies, Ann. Oper. Res. 35 (1992) 209–229.

[21] F. Akashi, K. Kobyashi, J. Namiki, K. Watanabe, Pipeline polling for satellite communications, Technical report CS85-66, IEICE, 1985.

Rocky K.C. Chang received his B.Sc. degree in electrical engineering from Virginia Polytechnic Institute and State University in Blacksburg, Virginia, in 1983. He received his M.E. degree in electrical engineering in 1985, M.S. degree in operations research and statistics in 1987, and the Ph.D. degree in computer system engineering in 1990, all from Rensselaer Polytechnic Institute, Troy, New York. From 1991 to 1993, he was in the Computer Science Department of IBM Thomas J. Watson Research Center, Yorktown Heights, New York. Since then he has been an Assistant Professor in the Department of Computing, The Hong Kong Polytechnic University, Kowloon, Hong Kong (SAR). His research interests include performance evaluation of computer and communications systems, TCP/IP protocol design and analysis, and queueing theory. Dr. Chang is a member of IEEE and ACM.



Sum Lam received his B.E. degree in computer science and engineering from Xian Jiaotong University, Xian, China, in 1994; and M.Phil. degree in computing from The Hong Kong Polytechnic University, Hong Kong (SAR), China, in 1997. He is currently a lecturer in the Department of Computing, The Hong Kong Polytechnic University, where he is also working toward his Ph.D. degree.