View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by Research Papers in Economics

Optimization of multiclass queueing networks with changeover times via the achievable region approach: Part I, the single-station case

Dimitris Bertsimas * José Niño-Mora **

July 1996, revised July 1998

Abstract

We address the performance optimization problem in a single-station multiclass queueing network with changeover times by means of the achievable region approach. This approach seeks to obtain performance bounds and scheduling policies from the solution of a mathematical program over a relaxation of the system's performance region. Relaxed formulations (including linear, convex, nonconvex and positive semidefinite constraints) of this region are developed by formulating equilibrium relations satisfied by the system, with the help of Palm calculus. Our contributions include: (1) new constraints formulating equilibrium relations on server dynamics; (2) a flow conservation interpretation of the constraints previously derived by the potential function method; (3) new positive semidefinite constraints; (4) new work decomposition laws for single-station multiclass queueing networks, which yield new convex constraints; (5) a unified buffer occupancy method of performance analysis obtained from the constraints; (6) heuristic scheduling policies from the solution of the relaxations.

*Dimitris Bertsimas, Sloan School of Management and Operations Research Center, Rm E53-359, MIT, Cambridge, MA 02139, dbertsim@aris.mit.edu. This research was partially supported by grants from the Leaders for Manufacturing program at MIT and a Presidential Young Investigator Award DDM-9158118 with matching funds from Draper Laboratory. This research was completed in part, while the author was visiting the Graduate School of Business and the Operations Research Department of Stanford University during his sabbatical leave. The author would like to thank Professors Michael Harrison and Arthur Veinott for their hospitality, encouragement and many interesting discussions.

**José Niño-Mora, Department of Economics and Business, Universitat Pompeu Fabra, E-08005 Barcelona, Spain, jose.nino-mora@econ.upf.es, www.econ.upf.es/~ninomora. This research was completed during the author's stay at the Operations Research Center of MIT as a PhD student and a Postdoctoral Associate.

1 Introduction

We address the problem of scheduling a multiclass queueing network (MQNET) on a single server, who incurs changeover times when moving from one class to another, to minimize time-average holding costs. This type of system arises in a broad variety of application areas, including manufacturing systems and computer-communication networks (see, e.g., Levy and Sidi (1990), and Sidi, Levy and Fuhrmann (1992)). In a companion paper (see Bertsimas and Niño-Mora (1998)) we address the corresponding problem in a multi-station MQNET.

Previous studies of this system have addressed exclusively the analysis of specific scheduling policies. Gupta and Buzacott (1990) consider the analysis of a two-class system, whereas Sidi, Levy and Fuhrmann (1992) solve the mean delay analysis for the general model considered here under a cyclic server routing policy with exhaustive service.

Recent studies addressing the problems of obtaining efficient scheduling policies and performance bounds for multiclass queues with changeover times (see, e.g., Boxma, Levy and Weststrate (1991) and Bertsimas and Xu (1993)) have focused their attention on systems without job feedback. Even for such simpler systems, performance bounds that account for the effect of changeover times were previously available only for static policies, in which the server bases his scheduling decisions only on the state of the class he is currently visiting. Such bounds, however, do not allow us to assess the potential for improvement over a proposed policy that could be achieved by using effectively dynamic information: performance bounds that hold under dynamic policies are needed for this purpose.

We present in this paper new bounds on the performance of dynamic and static nonidling policies for a MQNET with changeover times attended by a single server. The bounds emerge as the values of mathematical programs (linear, convex and nonconvex). These programs, which yield sharper bounds at the expense of increased computations, arise from constraints that formulate equilibrium laws. We reveal the underlying law (flow conservation) that explains the linear programming bounds previously derived via potential functions. We further establish new work decomposition laws, and apply them to formulate new convex constraints. Further constraints arise from server dynamics relations, and from semidefinite relations. When specialized to well-solved cases, our formulations recover the buffer occupancy analysis method. We further propose heuristic policies extracted from the optimal solution to the formulations. Our methodology corresponds to the achievable region method to the optimal control of queueing systems (see, e.g., the survey by Bertsimas (1995)).

The rest of the paper is structured as follows: Section 2 introduces the MQNET model. Section 3 develops a linear set of constraints based on the flow conservation law $L^- = L^+$. Section 4 presents a set of nonlinear constraints that formulate equilibrium relations on the server dynamics. Section 5 develops a family of new work decomposition laws, from which a corresponding family of convex constraints is obtained. Section 6 shows how to strengthen the formulation with positive semidefinite

constraints. Section 7 summarizes the formulations resulting from the constraints developed in previous sections and report computational results. Section 8 develops a unified buffer occupancy method of performance analysis by specializing the flow conservation and server dynamics constraints to certain policies. Section 9 discusses the problem of designing server scheduling policies from the solution of the relaxations. Finally, in Section 10 we present some concluding remarks.

2 The model: a single-station MQNET with changeover times

We shall focus our performance optimization study on a versatile model of a MQNET with changeover times attended to by a single server. In contrast with previous studies on performance optimization of multiclass queues with changeover times, the model we consider here incorporates the feature of Bernoulli *job feedback*. For this model only the performance analysis problem had previously been investigated (see Gupta and Buzacott (1990), and Sidi, Levy and Fuhrmann (1992)). Note that for the special case of zero changeover times the performance optimization problem can be solved exactly (see Klimov (1974)).

We consider a queueing system consisting of a *single server* that provides service to a set $\mathcal{N} = \{1, \ldots, n\}$ of job classes. Exogenous job arrivals occur according to independent Poisson processes, with rate α_i for class i jobs (which we refer to henceforth as *i-jobs*) , and join corresponding queues (i.e., i-jobs join the *i-queue*) until their service starts. The service times of i-jobs are i.i.d., drawn from a general distribution, with mean β_i and second moment $\beta_i^{(2)}$. We denote the corresponding mean residual life (or mean age) by $r_i = \beta_i^{(2)}/2\beta_i$. Upon completion of its service, an i-job may *leave* the system, with probability p_{i0}, or it may be *fed back* for further service as a j-job, with probability p_{ij}. Let $\boldsymbol{P}$ be the matrix of p_{ij}. We assume that matrix $\boldsymbol{I} - \boldsymbol{P}$ is invertible, which ensures that a single job moving through the network eventually exits it. We further assume that all service and interarrival times are mutually independent.

In order to serve jobs of a given class, the server must *visit* the corresponding queue, incurring a *changeover time* for moving there from the last queue visited: if after visiting the i-queue the server moves to the j-queue he incurs a random changeover time having a general distribution with mean s_{ij} and second moment $s_{ij}^{(2)}$.

Jobs are selected for service according to a *scheduling policy*. We consider *admissible* policies to be *nonanticipative*, *nonpreemptive* and *stable*. *Nonanticipative* means that scheduling decisions make no use of future information, such as remaining service times of jobs in the system or future job arrival times. *Nonpreemptive* means that once the service of a job, or a server changeover, is initiated, it must continue to completion. By *stable* we mean that the network admits a steady-state equilibrium distribution with finite mean number of jobs. We shall further refer to the classes of *nonidling*, *dynamic* and *static* policies. By *nonidling* we mean that the server never stays idle

at a queue: it must be either serving jobs or engaged in a changeover. By *dynamic* we mean that scheduling decisions only depend on the current states of all queues. By *static* we mean that scheduling decisions may only depend on the state of the queue currently being visited.

Other model parameters of interest are the *total arrival rate* and the *traffic intensity*. The total arrival rate of j-jobs, denoted by λ_j, is the total rate at which both external and feedback jobs arrive to the j-queue. The λ_j's are computed by solving the linear system

$$\lambda_j = \alpha_j + \sum_{i \in \mathcal{N}} p_{ij} \lambda_i, \qquad \text{for } j \in \mathcal{N}.$$

The *traffic intensity* of j-jobs, denoted by $\rho_j = \lambda_j \, \beta_j$, is the equilibrium probability that the server is busy with a j-job at an arbitrary time. The total traffic intensity of the system is $\rho = \sum_{i \in \mathcal{N}} \rho_i$, and represents the equilibrium probability that the server is busy. The condition $\rho < 1$ is necessary, but not sufficient, to ensure that the system is *stable* (i.e., that all incoming jobs eventually leave the system). Notice that the condition $\rho < 1$ does guarantee stability in the model with zero changeover times, and also in the model with positive changeover times under certain policies (e.g., exhaustive and gated service).

We assume that the system operates in stochastic equilibrium and introduce the following stochastic processes that describe its evolution.

- $L_i(t)$ = number of i-jobs in the system at time t.
- $B_i(t) = 1$ if an i-job is in service at time t; 0 otherwise.
- $B(t) = 1$ if the server is busy at time t; 0 otherwise; notice that $B(t) = \sum_{i \in \mathcal{N}} B_i(t)$.
- $B_{ij}(t) = 1$ if the server is engaged in an $i \to j$ changeover at time t; 0 otherwise.

In what follows we write, for convenience of notation, $L_i = L_i(0)$, $B_i = B_i(0)$, $B = B(0)$, $B_{ij} = B_{ij}(0)$.

The performance optimization problem. The main system's *performance measure* we are concerned with is the vector $\boldsymbol{x} = (x_j)_{j \in \mathcal{N}}$ whose components are the mean numbers of each class in the system in steady-state, i.e.,

- $x_j = E[L_j]$.

We consider a *cost function*, $c(\boldsymbol{x})$ (possibly nonlinear). The *performance optimization problem* we consider is as follows: compute a lower bound $\underline{Z} \leq c(\boldsymbol{x})$ valid under a suitable class of scheduling policies, and design a scheduling policy whose performance nearly minimizes the cost $c(\boldsymbol{x})$.

We approach this problem via the achievable region approach, as described in the Introduction. Let $\mathcal{X}$ be the performance region spanned by performance vector $\boldsymbol{x}$ under all admissible policies. Our

first goal is to derive constraints on performance vector $\boldsymbol{x}$ that define a relaxation of performance region $\mathcal{X}$. Since it is not obvious how to derive constraints on $\boldsymbol{x}$ directly, we shall pursue an approach to accomplish this goal based on the following plan:

1. Identify *equilibrium relations* satisfied by the system and formulate them as constraints involving *auxiliary performance measures*, using Palm calculus.

2. Formulate additional *positive semidefinite* constraints on the auxiliary performance measures.

3. Formulate constraints that express the original performance measure, $\boldsymbol{x}$, in terms of the auxiliary ones.

Notice that this approach has a clear geometric interpretation: It corresponds to constructing a relaxation of the performance region of the $\boldsymbol{x}$'s by 1) *lifting* this region into a higher dimensional space, by means of auxiliary variables, 2) *bounding* the lifted region through constraints on the auxiliary variables, and 3) *projecting* back into the original space. *Lift and project* techniques have proven powerful tools for constructing tight relaxations of hard discrete optimization problems (see, e.g., Lovász and Schrijver (1991)).

We consider three types of auxiliary performance measures: The first type represents *Palm moments* of queue lengths with respect to the *point processes* defined by certain embedded epochs. The second type represents server's visit and changeover frequencies. We present these point processes and auxiliary performance measures in Table 1.

In addition to those presented in Table 1, we consider a third type of auxiliary performance measures representing moments of queue lengths at an arbitrary time:

- $x_{ij} = E\left[L_j \mid B_i = 1\right]$ (mean number of j-jobs in the system at an arbitrary time during the service of i-jobs); $\boldsymbol{X} = (x_{ij})_{i,j\in\mathcal{N}}$;

- $x_j^0 = E\left[L_j \mid B = 0\right]$ (mean number of j-jobs in the system at an arbitrary time while the server is idle); $\boldsymbol{x}^0 = (x_j^0)_{j\in\mathcal{N}}$.

The first constraint we present is the elementary linear relation

$$\boldsymbol{x} = \boldsymbol{X}'\boldsymbol{\rho} + (1-\rho)\boldsymbol{x}^0, \tag{1}$$

which formulates the fact that at each time the server is either serving some job class or idling.

Other elementary constraints are the ones relating the mean number in the system at an arbitrary time during the service of jobs to the mean number at service completion epochs, can be derived as follows. The mean number of j-jobs at an arbitrary time during the service of an i-job is

$$x_{ij} = x_j^{S_i} + \alpha_j\, r_i,$$

Point process	Embedded epochs	Intensity	Performance measures
A_i^0	external class i job arrivals	α_i	
A_i	class i job arrivals	λ_i	
S_i	class i service initiation	λ_i	$x_k^{S_i} = E^{S_i}[L_k]$ $x_{kl}^{S_i} = E^{S_i}[L_k L_l]$
D_i	class i service completion	λ_i	$x_k^{D_i} = E^{D_i}[L_k]$ $x_{kl}^{D_i} = E^{D_i}[L_k L_l]$
F_{ij}	$i \to j$ job feedback	$\lambda_i p_{ij}$	$x_k^{F_{ij}} = E^{F_{ij}}[L_k]$
G_{ij}	$i \to j$ changeover completion	$y_{ij} = \frac{1}{s_{ij}} P\{B_{ij} = 1\}$	$x_k^{G_{ij}} = E^{G_{ij}}[L_k]$ $x_{kl}^{G_{ij}} = E^{G_{ij}}[L_k L_l]$
$G_j = \sum_{i \neq j} G_{ij}$	class j server visit initiation	y_j	$x_k^{G_j} = E^{G_j}[L_k]$ $x_{kl}^{G_j} = E^{G_j}[L_k L_l]$
H_{ij}	$i \to j$ changeover initiation	y_{ij}	$x_k^{H_{ij}} = E^{H_{ij}}[L_k]$ $x_{kl}^{H_{ij}} = E^{H_{ij}}[L_k L_l]$
$H_i = \sum_{j \neq i} H_{ij}$	class i server visit completion	y_i	$x_k^{H_i} = E^{H_i}[L_k]$ $x_{kl}^{H_i} = E^{H_i}[L_k L_l]$

Table 1: Auxiliary performance measures. All performance measures Z in the table are defined immediately *after* their corresponding event epoch. Since the sample paths are right-continuous for all performance measures Z, it follows that $Z^+ = Z$.

i.e., the number of j-jobs at the beginning of an i-service plus the mean number of external class j arrivals until an arbitrary time during an i-service. Furthermore, we have

$$x_j^{D_i} = x_j^{S_i} + \beta_i \alpha_j + p_{ij} - \delta_{ij}, \tag{2}$$

where δ_{ij} is Kronecker's delta function. Subtracting the previous equations, we obtain

$$x_{ij} = x_j^{D_i} + (r_i - \beta_i)\alpha_j + \delta_{ij} - p_{ij}. \tag{3}$$

In matrix notation,

$$\boldsymbol{X} = \boldsymbol{X}^D + (\boldsymbol{r} - \boldsymbol{\beta})\boldsymbol{\alpha}' + \boldsymbol{I} - \boldsymbol{P}, \tag{4}$$

where $\boldsymbol{X}^D = \left(x_j^{D_i}\right)_{i,j \in \mathcal{N}}$.

3 System flow conservation constraints

In this section we present a set of linear equality constraints on performance measures by formulating the classical *flow conservation law* of queueing theory $L^- = L^+$, which states that in a queueing system in which arrivals and departures are of unit size, the stationary probabilities of the number in system seen at arrival epochs and that seen at departure epochs are equal. These equations were first derived by Klimov (1974), by transform methods, in his pioneering performance optimization study of the model with zero changeover times.

Our contribution in this section is twofold: We present a new direct derivation of Klimov's result by means of Palm calculus, and we extend its scope by observing that the constraints are also valid for models with general changeover times under general policies.

Let $\boldsymbol{\Lambda} = \mathrm{Diag}(\boldsymbol{\lambda})$. The result is as follows:

Theorem 1 (Flow conservation equations) *Under any admissible scheduling policy, performance measures $\boldsymbol{x}$ and $\boldsymbol{X}^D$ satisfy the system of linear equations*

$$-\boldsymbol{\alpha}\boldsymbol{x}' - \boldsymbol{x}\boldsymbol{\alpha}' + (\boldsymbol{I} - \boldsymbol{P})'\boldsymbol{\Lambda}\boldsymbol{X}^D + \boldsymbol{X}^{D\prime}\boldsymbol{\Lambda}(\boldsymbol{I} - \boldsymbol{P}) = (\boldsymbol{I} - \boldsymbol{P})'\boldsymbol{\Lambda}\boldsymbol{P} + \boldsymbol{P}'\boldsymbol{\Lambda}(\boldsymbol{I} - \boldsymbol{P}). \tag{5}$$

Equivalently, by (4), performance measures $\boldsymbol{x}$ and $\boldsymbol{X}$ satisfy

$$-\boldsymbol{\alpha}\boldsymbol{x}' - \boldsymbol{x}\boldsymbol{\alpha}' + (\boldsymbol{I} - \boldsymbol{P})'\boldsymbol{\Lambda}\boldsymbol{X} + \boldsymbol{X}'\boldsymbol{\Lambda}(\boldsymbol{I} - \boldsymbol{P}) = (\boldsymbol{I} - \boldsymbol{P})'\boldsymbol{\Lambda}(\boldsymbol{I} - \boldsymbol{\theta}\boldsymbol{\alpha}') + (\boldsymbol{I} - \boldsymbol{\alpha}\boldsymbol{\theta}')\boldsymbol{\Lambda}(\boldsymbol{I} - \boldsymbol{P}), \tag{6}$$

with $\boldsymbol{\theta} = \boldsymbol{\beta} - \boldsymbol{r}$.

Remarks:

1. Constraints analogous to (5) have been derived for the *branching bandit* model with zero changeover times in Bertsimas, Paschalidis and Tsitsiklis (1995), using potential function techniques, and in Niño-Mora (1995), using the flow conservation interpretation we present here.

They further show that the region in $\boldsymbol{x}$-space defined by constraints analogous to (1), (4), (5), together with $\boldsymbol{x}^0 = \mathbf{0}$ and $\boldsymbol{x} \geq \mathbf{0}$, is the exact performance region of the $\boldsymbol{x}$'s. It is noteworthy that the flow conservation law $L^- = L^+$ leads to a *compact reformulation* (having polynomial size on the number of job classes) of the $\boldsymbol{x}$'s performance region, that involves the matrix of auxiliary variables $\boldsymbol{X}$, whereas the exact formulation on the original variables $\boldsymbol{x}$ was found to have exponential size in Bertsimas and Niño-Mora (1996).

2. Notice that constraints (5) do not involve changeover time parameters. This is because they are valid under *any* admissible scheduling policy, regardless of whether it is work-conserving.

3. An interesting consequence of constraints (5) is the following: They imply, together with relations (1) and (4), that the vector of expected queue lengths at an arbitrary time, $\boldsymbol{x}$, as well as the vector of expected queue lengths at an arbitrary server idling time, $\boldsymbol{x}^0$, are uniquely determined by the matrix of expected queue lengths at service completion epochs, $\boldsymbol{X}^D$. Therefore, in order to formulate the performance region of the $\boldsymbol{x}$'s we need only to focus on formulating constraints on matrix $\boldsymbol{X}^D$.

In the remainder of this section we present a new proof of Theorem 1 by showing, via Palm calculus, that Equations (5) simply formulate the flow conservation law $L^- = L^+$. We shall denote by $E^N[\cdot]$ the expectation operator with respect to the equilibrium distribution embedded at an *arbitrary epoch* in a point process N. In particular, $E^N[L^-] = E^N[L(0-)]$, $E^N[L^+] = E^N[L(0+)]$ where 0 is an epoch of point process N. The following basic result of Palm calculus will be needed throughout the paper.

Theorem 2 (Superposition of point processes) *Suppose point process N, having intensity λ, is the superposition of point processes $N_1, \ldots, N_K$, i.e., $N = N_1 + \cdots + N_K$, where process N_k has intensity λ_k. Then,*

$$P^N\{\cdot\} = \sum_{k=1}^{K} \frac{\lambda_k}{\lambda} P^{N_k}\{\cdot\}.$$

The following result lays the groundwork for our proof of Theorem 1 by showing how to express moments of queue lengths at certain job arrival and departure epochs in terms of performance measures $\boldsymbol{x}$ and $\boldsymbol{X}^D$. Let δ_{ij} denote Kronecker's delta function.

Proposition 1 *Under any admissible scheduling policy,*

(a)

$$E^{A_i}\left[L_j^-\right] = \frac{\alpha_i}{\lambda_i} x_j + \sum_{k \in \mathcal{N}} \frac{\lambda_k p_{ki}}{\lambda_i} x_j^{D_k} + \sum_{k \in \mathcal{N}} \frac{\lambda_k p_{ki}}{\lambda_i} (\delta_{kj} - p_{kj}); \tag{7}$$

(b)

$$\begin{aligned} E^{A_i + A_j}\left[L_i^- + L_j^-\right] &= \frac{\lambda_i}{\lambda_i + \lambda_j} E^{A_i}\left[L_i^-\right] + \frac{\lambda_j}{\lambda_i + \lambda_j} E^{A_j}\left[L_j^-\right] \\ &\quad + \frac{\lambda_i}{\lambda_i + \lambda_j} E^{A_i}\left[L_j^-\right] + \frac{\lambda_j}{\lambda_i + \lambda_j} E^{A_j}\left[L_i^-\right]; \end{aligned} \tag{8}$$

(c)

$$E^{D_i+D_j}\left[L_i^+ + L_j^+\right] = \frac{\lambda_i}{\lambda_i+\lambda_j}x_i^{D_i} + \frac{\lambda_j}{\lambda_i+\lambda_j}x_j^{D_j} + \frac{\lambda_i}{\lambda_i+\lambda_j}x_j^{D_i} + \frac{\lambda_j}{\lambda_i+\lambda_j}x_i^{D_j}. \quad (9)$$

Proof

(a) Since point process A_i can be represented as the superposition $A_i = A_i^0 + \sum_{k\in\mathcal{N}} F_{ki}$, Palm calculus yields

$$E^{A_i}\left[L_j^-\right] = \frac{\alpha_i}{\lambda_i}E^{A_i^0}\left[L_j^-\right] + \sum_{k\in\mathcal{N}}\frac{\lambda_k p_{ki}}{\lambda_i}E^{F_{ki}}\left[L_j^-\right]. \quad (10)$$

From the well-known PASTA (Poisson Arrivals See Time Averages) we have

$$E^{A_i^0}\left[L_j^-\right] = E\left[L_j\right] = x_j. \quad (11)$$

Furthermore, we have

$$\begin{aligned} E^{F_{ki}}\left[L_j^-\right] &= E^{D_k}\left[L_j^-\right], \\ &= E^{D_k}\left[L_j^+\right] - p_{kj} + \delta_{kj}, \end{aligned} \quad (12)$$

since the Bernoulli job feedback mechanism implies that the mean number of j-jobs in the system just before a $k \to i$ job feedback epoch equals the mean number of j-jobs present just before a k-job service completion epoch. Combining (11) with (12) yields identity (7).

Parts (b) and (c) are trivial.

■

We have now the building blocks for proving Theorem 1 by applying the flow conservation law $L^- = L^+$.

Proof of Theorem 1.

The identity corresponding to the jth diagonal element in (5) formulates the relation

$$E^{A_j}\left[L_j^-\right] = E^{D_j}\left[L_j^+\right] \quad (13)$$

using Proposition 1(a). The equality corresponding to row i and column j in (5) formulates the relation

$$E^{A_i+A_j}\left[L_i^- + L_j^-\right] = E^{D_i+D_j}\left[L_i^+ + L_j^+\right] \quad (14)$$

using Proposition 1(b, c).

Identity (6) follows straightforwardly from (5) by applying (4). ■

Remarks.

1. Notice that identities (13) and (14) formulate the flow conservation law $L^- = L^+$ as applied to the queues of j-jobs and $\{i,j\}$-jobs considered in isolation, for all pairs $\{i,j\}$ of job classes.

2. It is interesting to observe that we do not obtain additional constraints by formulating the flow conservation law

$$E^{\sum_{i\in S} A_i}\left[\sum_{i\in S} L_i^-\right] = E^{\sum_{i\in S} D_i}\left[\sum_{i\in S} L_i^+\right]$$

for subsets of job classes S of size larger than 3. The equations for $|S| \geq 3$ turn out to be implied by those for $|S| \leq 2$.

4 Server dynamics constraints

In this section we derive constraints that formulate equilibrium relations related to the server dynamics, and that account explicitly for the changeover times.

4.1 Server flow conservation constraints

We first present some elementary constraints (although nonlinear) that relate the performance measures x_j^0 and the performance measures $x_j^{H_{kl}}$ defined at changeover initiation epochs. These constraints involve the changeover frequencies y_{kl} (see Table 1). We further establish linear constraints on the visit and changeover frequencies (y_j, y_{ij}), that formulate server flow conservation relations.

Proposition 2 *Under any dynamic nonidling scheduling policy,*

(a)

$$x_j^0 = \sum_{k,l:\ k\neq l} \frac{s_{kl}y_{kl}}{1-\rho}\left(x_j^{H_{kl}} + \alpha_j \frac{s_{kl}^{(2)}}{2s_{kl}}\right) \tag{15}$$

$$\geq \sum_{k,l:\ k\neq l} \frac{\alpha_j y_{kl} s_{kl}^{(2)}}{2(1-\rho)} \tag{16}$$

(b)

$$\sum_{i,j:\ i\neq j} s_{ij}y_{ij} = 1-\rho. \tag{17}$$

(c)

$$y_i = \sum_{j\in\mathcal{N}\setminus\{i\}} y_{ij} = \sum_{j\in\mathcal{N}\setminus\{i\}} y_{ji}, \qquad \textit{for } i\in\mathcal{N}. \tag{18}$$

Proof

(a) Eq. (15) follows from the elementary relations (valid under nonidling policies)

$$\begin{aligned} x_j^0 &= \sum_{k,l:\ k\neq l} \frac{s_{kl}y_{kl}}{1-\rho} E\left[L_j \mid B_{kl}=1\right] \\ &= \sum_{k,l:\ k\neq l} \frac{s_{kl}y_{kl}}{1-\rho} \left(x_j^{H_{kl}} + \alpha_j \frac{s_{kl}^{(2)}}{2s_{kl}} \right). \end{aligned}$$

Inequality (16) follows directly from Eq. (15):

$$\begin{aligned} x_j^0 &= \sum_{k,l:\ k\neq l} \frac{s_{kl}y_{kl}}{1-\rho} E\left[L_j \mid B_{kl}=1\right] \\ &\geq \sum_{k,l:\ k\neq l} \frac{s_{kl}y_{kl}}{1-\rho} \alpha_j \frac{s_{kl}^{(2)}}{2s_{kl}}. \end{aligned}$$

(b) Eq. (17) formulates the requirements that policies must be nonidling, using the fact that $P\left\{B_{ij}=1\right\} = s_{ij}y_{ij}$.

(c) Eq. (18) formulates a simple flow conservation relation: the rates at which the server visits and leaves the i-queue are equal. ■

4.2 Server visit constraints

We derive in this section a family of nonlinear constraints by formulating a key relation between the point processes defined in Table 1. In the notation of Palm calculus, this relation is written as

$$G_i + D_i = H_i + S_i, \qquad \text{for } i \in \mathcal{N}, \tag{19}$$

and it expresses the elementary fact that, under any nonidling policy, each time a visit initiation or a service completion occurs, coincidentally there also occurs either a service beginning or a visit completion. Identity (19) was first observed by Eisenberg (1972), who applied it as a central tool in his pioneering analysis (via transform methods) of polling systems with changeover times.

We show next that identity (19) allows us to represent Palm moments at service completion epochs ($x_j^{D_i}$) in terms of Palm moments at visit initiation and completion epochs ($x_k^{G_i}$, $x_{kl}^{G_i}$, $x_k^{H_i}$, $x_{kl}^{H_i}$; see Table 1), and to formulate additional constraints between the two latter kinds of moments. We state and prove next our main result.

Theorem 3 *Under any dynamic nonidling policy,*

(a)

$$y_i\left(x_j^{H_i} - x_j^{G_i} \right) = \lambda_i(\alpha_j\beta_i + p_{ij} - \delta_{ij}) \qquad \textit{for } i,j \in \mathcal{N}. \tag{20}$$

(b)

$$y_i\left(x_{jk}^{H_i} - x_{jk}^{G_i} \right) = \lambda_i \Big[\left(\alpha_k\beta_i + p_{ik} - \delta_{ik} \right) x_j^{D_i} + \left(\alpha_j\beta_i + p_{ij} - \delta_{ij} \right) x_k^{D_i}$$

$$-\left(\alpha_j(p_{ik}-\delta_{ik})+\alpha_k(p_{ij}-\delta_{ij})+\alpha_j\delta_{kj}\right)\beta_i+\alpha_j\alpha_k\left(\beta_i^{(2)}-2\beta_i^2\right)$$
$$p_{ij}\delta_{ik}+p_{ik}\delta_{ij}+p_{ij}\delta_{kj}-\delta_{ij}\delta_{ik}-2p_{ik}p_{ij}\Big] \qquad \textit{for } i,j,k\in\mathcal{N}. \tag{21}$$

Proof

(a) By combining identity (19) with Superposition Theorem 2 we obtain the relation

$$y_i\left(x_j^{H_i}-x_j^{G_i}\right)=\lambda_i\left(x_j^{D_i}-x_j^{S_i}\right). \tag{22}$$

Eq. (20) now follows by substituting $x_j^{D_i}-x_j^{S_i}$ in (22) using Eq. (2).

(b) By combining identity (19) with Superposition Theorem 2, we obtain the relation

$$y_i\left(x_{jk}^{H_i}-x_{jk}^{G_i}\right)=\lambda_i\left(x_{jk}^{D_i}-x_{jk}^{S_i}\right), \qquad \text{for } i,j,k\in\mathcal{N}. \tag{23}$$

We next find an expression for the difference $x_{jk}^{D_i}-x_{jk}^{S_i}$.

We denote in what follows by $L_j^{S_i}$ and $L_j^{D_i}$ the number of j-jobs in the system at a typical i-job service initiation and completion epoch, respectively. We also let $N_j(v_i)$ be the number of external j-job arrivals during the service of a typical i-job, and let $\xi_{ij}=1$ if that job feeds back to the j-queue after completion of its service, and 0 otherwise. We have the relations

$$L_j^{D_i}=L_j^{S_i}+N_j(v_i)+\xi_{ij}-\delta_{ij},$$

and

$$L_k^{D_i}=L_k^{S_i}+N_k(v_i)+\xi_{ik}-\delta_{ik},$$

whence

$$\begin{aligned}L_j^{D_i}L_k^{D_i} &= L_j^{S_i}L_k^{S_i}+N_j(v_i)L_k^{S_i}+(\xi_{ij}-\delta_{ij})L_k^{S_i}+N_k(v_i)L_j^{S_i}+N_j(v_i)N_k(v_i)+(\xi_{ij}-\delta_{ij})N_k(v_i)\\ &\quad +(\xi_{ik}-\delta_{ik})L_j^{S_i}+(\xi_{ik}-\delta_{ik})N_j(v_i)+(\xi_{ij}-\delta_{ij})(\xi_{ik}-\delta_{ik}).\end{aligned}$$

The result now follows from the relations

$$E\left[N_j(v_i)L_k^{S_i}\right]=\alpha_j\beta_i x_k^{S_i},$$

$$E\left[(\xi_{ij}-\delta_{ij})L_k^{S_i}\right]=(p_{ij}-\delta_{ij})x_k^{S_i},$$

$$E\left[(\xi_{ik}-\delta_{ik})N_j(v_i)\right]=\alpha_j\beta_i(p_{ik}-\delta_{ik}),$$

$$E\left[N_j(v_i)N_k(v_i)\right]=\alpha_j\alpha_k\beta_i^{(2)}+\delta_{kj}\alpha_j\beta_i,$$

$$E\left[(\xi_{ij}-\delta_{ij})(\xi_{ik}-\delta_{ik})\right]=p_{ij}\delta_{kj}-p_{ij}\delta_{ik}-p_{ik}\delta_{ij}+\delta_{ij}\delta_{ik},$$

and Eqns. (2), (23). ■

4.3 Server changeover dynamics constraints

We derive in this section a family of constraints that formulate relations on the server changeover dynamics. These constraints allow us to express Palm moments of queue lengths at server visit initiation and completion epochs in terms of Palm moments at server changeover initiation epochs, and formulate additional relations among the latter.

Theorem 4 *Under any dynamic nonidling policy,*

(a)

$$y_i x_j^{G_i} = \sum_{k \in \mathcal{N} \setminus \{i\}} y_{ki} x_j^{G_{ki}}, \qquad \textit{for } i, j \in \mathcal{N}. \tag{24}$$

(b)

$$y_i x_j^{H_i} = \sum_{k \in \mathcal{N} \setminus \{i\}} y_{ik} x_j^{H_{ik}}, \qquad \textit{for } i, j \in \mathcal{N}. \tag{25}$$

(c)

$$x_j^{G_{ik}} = x_j^{H_{ik}} + \alpha_j s_{ik}, \qquad \textit{for } i, j, k \in \mathcal{N}, i \neq k; \tag{26}$$

(d)

$$y_i x_{jk}^{G_i} = \sum_{r \in \mathcal{N} \setminus \{i\}} y_{ri} x_{jk}^{G_{ri}} \qquad \textit{for } i, j, k \in \mathcal{N},\ j \neq k. \tag{27}$$

(e)

$$y_i x_{jk}^{H_i} = \sum_{r \in \mathcal{N} \setminus \{i\}} y_{ir} x_{jk}^{H_{ir}}, \qquad \textit{for } i, j, k \in \mathcal{N},\ j \neq k. \tag{28}$$

(f)

$$x_{jk}^{G_{ir}} = x_{jk}^{H_{ir}} + \alpha_j s_{ir} x_k^{H_{ir}} + \alpha_k s_{ir} x_j^{H_{ir}} + \alpha_j \alpha_k s_{ir}^{(2)}, \qquad \textit{for } i, j, k, r \in \mathcal{N},\ j \neq k. \tag{29}$$

Proof

(a) Eq. (24) follows by noticing that $G_i = \sum_{k \in \mathcal{N} \setminus \{i\}} G_{ki}$ and applying Superposition Theorem 2.

(b) Eq. (25) follows analogously from Theorem 2 since $H_i = \sum_{k \in \mathcal{N} \setminus \{i\}} H_{ik}$.

(c) Eq. (26) is elementary, as it formulates the fact that the number of j-jobs in the system at the end of an $i \to k$ changeover equals that at the beginning plus the number of external j-job arrivals that occur during the changeover.

(d)-(e): they follow similarly as parts (a)-(b).

(f) Let $L_j^{H_{ir}}$ (resp. $L_j^{G_{ir}}$) be the length of the j-queue at the beginning (resp. end) of an $i \to r$ changeover, and let $N_j(v_{ir})$ be the number of external j-job arrivals during that changeover. We have

$$L_j^{G_{ir}} = L_j^{H_{ir}} + N_j(v_{ir}),$$

and

$$L_k^{G_{ir}} = L_k^{H_{ir}} + N_k(v_{ir}),$$

whence

$$L_j^{G_{ir}} L_k^{G_{ir}} = L_j^{H_{ir}} L_k^{H_{ir}} + N_j(v_{ir}) L_k^{H_{ir}} + N_k(v_{ir}) L_j^{H_{ir}} + N_j(v_{ir}) N_k(v_{ir}).$$

Now, since

$$E\left[N_j(v_{ir}) L_k^{H_{ir}}\right] = \alpha_j s_{ir} x_k^{H_{ir}}$$

and

$$E\left[N_k(v_{ir}) N_j(v_{ir})\right] = \alpha_k \alpha_j s_{ir}^{(2)},$$

we obtain Eq. (29). ■

5 Work decomposition constraints

In this section we derive a family of convex constraints by identifying and formulating new *work decomposition laws* satisfied by the system. A work decomposition law is a linear relation between the mean number in the system from each class at an arbitrary time and at an arbitrary time during a period when the server is idle. Our contributions include: (1) we present a family of new work decomposition laws for a multiclass $M/G/1$ queue with Bernoulli feedback, that extends the most general result previously known: Boxma's (1989) work decomposition law for multiclass $M/G/1$ queues; (2) we develop a new technique for deriving work decomposition laws: in contrast to previous derivations, based on probabilistic arguments (mainly *stochastic work decomposition* properties), we obtain the laws directly by reformulating linear flow conservation equations; (3) we present new families of convex constraints, by exploiting the physical interpretation of the new work decomposition laws.

The idea of deriving constraints from work decomposition laws was introduced in Bertsimas and Xu (1993). They derived a set of convex constraints from a work decomposition law due to Fuhrmann and Cooper (1985), which apply to multiclass $M/G/1$ queues with changeover times, but with no job feedback.

To present our work decomposition laws we introduce, for every subset $S \subseteq \mathcal{N}$ of job classes, the *S-workload* process, $\left\{V^S(t)\right\}_{t\in\Re}$, where $V^S(t)$ is the *total remaining service time* the server needs to perform on the S-jobs in the system at time t for first driving all of them out of classes in S. Notice that $\left\{V^S(t)\right\}_{t\in\Re}$ is the *total workload process* for the S-queue (obtained by aggregating S-jobs in a single queue). We further write $B^S(t) = \sum_{i\in S} B_i(t)$, i.e., $B^S(t)$ is the indicator of the server being busy with an S-job. We next define parameters V_i^S, for $i \in \mathcal{N}$, as the solution of the system of linear equations

$$V_i^S = \beta_i + \sum_{j\in\mathcal{S}} p_{ij} V_j^S, \qquad \text{for } i \in \mathcal{N}.$$

We shall refer to V_i^S, for $i \in S$, as the *S-workload of an i-job*, as it represents the mean remaining service time a current i-job receives until its class first leaves S following completion of its current service. As usual, we write $V^S = V^S(0), B^S = B^S(0)$.

We further denote by $\rho^0(S)$ the rate at which *external* S-work (corresponding to S-jobs) enters the system,

$$\rho^0(S) = \sum_{i \in S} \alpha_i V_i^S,$$

and write

$$\rho(S) = \sum_{i \in S} \rho_i.$$

For notational convenience we introduce set function $f(S)$ defined by

$$f(S) = \left\{ \sum_{i \in \mathcal{N}} \rho_i \left(V_i^S - \beta_i + r_i \right) \right\} \frac{\rho^0(S)}{1 - \rho^0(S)} + \sum_{i \in S} \rho_i V_i^S.$$

Notice that

$$f(\mathcal{N}) = \left\{ \sum_{i \in \mathcal{N}} \rho_i \left(V_i^{\mathcal{N}} - \beta_i + r_i \right) \right\} \frac{\rho}{1 - \rho} + \sum_{i \in \mathcal{N}} \rho_i V_i^{\mathcal{N}}.$$

We state and prove next the new work decomposition laws.

Theorem 5 (Work decomposition laws) *Under any dynamic scheduling policy, and for any subset $S \subseteq \mathcal{N}$ of job classes,*

(a)

$$\sum_{j \in S} V_j^S x_j \quad = \quad f(S) + \frac{1}{1 - \rho^0(S)} \sum_{i \in S^c} \sum_{j \in S} \lambda_i V_i^S V_j^S x_j^{S_i} + \frac{1 - \rho}{1 - \rho^0(S)} \sum_{j \in S} V_j^S x_j^0. \tag{30}$$

(b) *Identity (30) can be reformulated as*

$$\begin{aligned} E\left[V^S\right] \quad &= \quad f(S) - \sum_{i \in S} \rho_i \left(\beta_i - r_i \right) - \frac{\rho^0(S)}{1 - \rho^0(S)} \sum_{i \in S^c} \rho_i r_i \\ &\quad + \frac{1}{1 - \rho^0(S)} \sum_{i \in S^c} (\lambda_i V_i^S - \rho_i)) E^{S_i}\left[V^S\right] + \frac{1 - \rho(S)}{1 - \rho^0(S)} E\left[V^S \mid B^S = 0\right]. \end{aligned} \tag{31}$$

Proof

(a) In what follows we use the following notation: if $S, T \subseteq \mathcal{N}$, $\boldsymbol{z} = (z_i)_{i \in \mathcal{N}}$ is an n-vector, and $\boldsymbol{A} = (a_{ij})_{i,j \in \mathcal{N}}$ is an $n \times n$ matrix, we shall write

$$\boldsymbol{z}_S = (z_j)_{j \in S}, \quad \text{and} \quad \boldsymbol{A}_{ST} = (a_{ij})_{i \in S, j \in T}.$$

Let $\boldsymbol{v}$ denote the n-vector

$$\boldsymbol{v} = \begin{pmatrix} \boldsymbol{V}_S^S \\ \boldsymbol{0} \end{pmatrix},$$

and let us define the set function $b(S)$, for $S \subseteq \mathcal{N}$, by

$$b(S) = \frac{1}{2} \sum_{i \in S} \sum_{j \in S} V_i^S b_{ij} V_j^S,$$

where $\boldsymbol{B} = (b_{ij})_{i,j\in\mathcal{N}}$ is the $n \times n$-matrix

$$\boldsymbol{B} = -\boldsymbol{\alpha}\boldsymbol{x}' - \boldsymbol{x}\boldsymbol{\alpha}' + (\boldsymbol{I}-\boldsymbol{P})'\boldsymbol{\Lambda}\boldsymbol{X} + \boldsymbol{X}'\boldsymbol{\Lambda}(\boldsymbol{I}-\boldsymbol{P}) = (\boldsymbol{I}-\boldsymbol{P})'\boldsymbol{\Lambda}(\boldsymbol{I}-\boldsymbol{\theta}\boldsymbol{\alpha}') + (\boldsymbol{I}-\boldsymbol{\alpha}\boldsymbol{\theta}')\boldsymbol{\Lambda}(\boldsymbol{I}-\boldsymbol{P}),$$

with $\boldsymbol{\theta} = \boldsymbol{\beta} - \boldsymbol{r}$.

We first observe that, from the flow conservation equations (6) in Theorem 1, we obtain

$$\begin{aligned}
b(S) &= \frac{1}{2}\boldsymbol{v}'\{-\boldsymbol{\alpha}\boldsymbol{x}' - \boldsymbol{x}\boldsymbol{\alpha}' + (\boldsymbol{I}-\boldsymbol{P})'\boldsymbol{\Lambda}\boldsymbol{X} + \boldsymbol{X}'\boldsymbol{\Lambda}(\boldsymbol{I}-\boldsymbol{P})\}\boldsymbol{v} \\
&= -(\boldsymbol{v}'\boldsymbol{\alpha})(\boldsymbol{v}'\boldsymbol{x}) + \left\{ \begin{pmatrix} \boldsymbol{I}_S - \boldsymbol{P}_{SS} & -\boldsymbol{P}_{SS^c} \\ -\boldsymbol{P}_{S^cS} & \boldsymbol{I}_{S^c} - \boldsymbol{P}_{S^cS^c} \end{pmatrix} \begin{pmatrix} \boldsymbol{V}_S^S \\ \boldsymbol{0} \end{pmatrix} \right\}' \boldsymbol{\Lambda}\boldsymbol{X} \begin{pmatrix} \boldsymbol{V}_S^S \\ \boldsymbol{0} \end{pmatrix} \\
&= -(\boldsymbol{v}'\boldsymbol{\alpha})(\boldsymbol{v}'\boldsymbol{x}) + \begin{pmatrix} \boldsymbol{\beta}_S' & \boldsymbol{\beta}_{S^c}' - \boldsymbol{V}_{S^c}^{S\,\prime} \end{pmatrix} \boldsymbol{\Lambda} \begin{pmatrix} \boldsymbol{X}_{SS} & \boldsymbol{X}_{SS^c} \\ \boldsymbol{X}_{S^cS} & \boldsymbol{X}_{S^cS^c} \end{pmatrix} \begin{pmatrix} \boldsymbol{V}_S^S \\ \boldsymbol{0} \end{pmatrix} \\
&= -\rho^0(S)\sum_{j\in S} V_j^S x_j + \sum_{i\in S}\sum_{j\in S} \rho_i V_j^S x_{ij} - \sum_{i\in S^c}\sum_{j\in S} \lambda_i (V_i^S - \beta_i) V_j^S x_{ij}
\end{aligned} \tag{32}$$

Next, we notice that equations (1) can be written as

$$x_j = \sum_{i\in S} \rho_i x_{ij} + \sum_{i\in S^c} \rho_i x_{ij} + (1-\rho)x_j^0, \qquad \text{for } j \in \mathcal{N}. \tag{33}$$

Now, from (32) and (33) we obtain

$$\sum_{j\in S} V_j^S x_j = \frac{b(S)}{1-\rho^0(S)} + \frac{1}{1-\rho^0(S)} \sum_{i\in S^c}\sum_{j\in S} \lambda_i V_i^S V_j^S x_{ij} + \frac{1-\rho}{1-\rho^0(S)} \sum_{i\in S} V_j^S x_j^0,$$

On the other hand, constant term $b(S)$ may be expressed in terms of $f(S)$ as follows:

$$\begin{aligned}
b(S) &= \begin{pmatrix} \boldsymbol{V}_S^{S\,\prime} & \boldsymbol{0} \end{pmatrix} (\boldsymbol{I}-\boldsymbol{P})'\boldsymbol{\Lambda}(\boldsymbol{I}-\boldsymbol{\theta}\boldsymbol{\alpha}') \begin{pmatrix} \boldsymbol{V}_S^S \\ \boldsymbol{0} \end{pmatrix} \\
&= \left\{ (\boldsymbol{I}-\boldsymbol{P}) \begin{pmatrix} \boldsymbol{V}_S^S \\ \boldsymbol{0} \end{pmatrix} \right\}' \boldsymbol{\Lambda} \begin{pmatrix} \boldsymbol{I}_S - \boldsymbol{\theta}_S\boldsymbol{\alpha}_S{}' & -\boldsymbol{\theta}_S\boldsymbol{\alpha}_{S^c} \\ -\boldsymbol{\theta}_{S^c}\boldsymbol{\alpha}_S{}' & \boldsymbol{I}_{S^c} - \boldsymbol{\theta}_{S^c}\boldsymbol{\alpha}_{S^c}' \end{pmatrix} \begin{pmatrix} \boldsymbol{V}_S^S \\ \boldsymbol{0} \end{pmatrix} \\
&= \begin{pmatrix} \boldsymbol{\beta}_S{}' & \boldsymbol{\beta}_{S^c}' - \boldsymbol{V}_{S^c}^{S\,\prime} \end{pmatrix} \boldsymbol{\Lambda} \begin{pmatrix} (\boldsymbol{I}_S - \boldsymbol{\theta}_S\boldsymbol{\alpha}_S{}')\boldsymbol{V}_S^S \\ -\boldsymbol{\theta}_{S^c}\boldsymbol{\alpha}_S{}'\boldsymbol{V}_S^S \end{pmatrix} \\
&= \begin{pmatrix} \boldsymbol{\rho}_S{}' & \boldsymbol{\rho}_{S^c}' - \boldsymbol{V}_{S^c}^{S\,\prime}\boldsymbol{\Lambda}_{S^c} \end{pmatrix} \begin{pmatrix} \boldsymbol{V}_S^S - \boldsymbol{\theta}_S\boldsymbol{\alpha}_S{}'\boldsymbol{V}_S^S \\ -\boldsymbol{\theta}_{S^c}\boldsymbol{\alpha}_S{}'\boldsymbol{V}_S^S \end{pmatrix} \\
&= \boldsymbol{\rho}_S{}'\boldsymbol{V}_S^S - (\boldsymbol{\rho}_S{}'\boldsymbol{\theta}_S)\left(\boldsymbol{\alpha}_S{}'\boldsymbol{V}_S^S\right) - (\boldsymbol{\rho}_{S^c}'\boldsymbol{\theta}_{S^c})\left(\boldsymbol{\alpha}_S'\boldsymbol{V}_S^S\right) + \left(\boldsymbol{V}_{S^c}^{S\,\prime}\boldsymbol{\Lambda}_{S^c}\boldsymbol{\theta}_{S^c}\right)\left(\boldsymbol{\alpha}_S{}'\boldsymbol{V}_S^S\right) \\
&= \left(1-\rho^0(S)\right) f(S) - \rho^0(S)\sum_{i\in S^c} \lambda_i r_i V_i^S,
\end{aligned} \tag{34}$$

which, combined with the relation $x_{ij} = x_j^{S_i} + \alpha_j r_i$, yields the result.

(b) It follows from the definition of the S-workload process that

$$E\left[V^S\right] = \sum_{j\in S} V_j^S x_j - \sum_{j\in S} \rho_j(\beta_j - r_j),$$

$$E^{S_i}\left[V^S\right] = \sum_{j\in S} V_j^S x_j^{S_i},$$

and

$$E\left[V^S \mid B=0\right] = \sum_{j\in S} V_j^S x_j^0,$$

which, combined with (30), yields

$$E\left[V^S\right] = f(S) - \sum_{i\in S}\rho_i\left(\beta_i - r_i\right) + \frac{1}{1-\rho^0(S)}\sum_{i\in S^c}\lambda_i V_i^S E^{S_i}\left[V^S\right] + \frac{1-\rho}{1-\rho^0(S)}E\left[V^S \mid B=0\right]. \quad (35)$$

Eq. (31) follows from Eq. (35) by substituting $E\left[V^S \mid B=0\right]$ using the elementary relations

$$E\left[V^S \mid B^S=0\right] = \frac{\rho(S^c)}{1-\rho(S)}E\left[V^S \mid B^{S^c}=1\right] + \frac{1-\rho}{1-\rho(S)}E\left[V^S \mid B=0\right], \quad (36)$$

and

$$\rho(S^c)E\left[V^S \mid B^{S^c}=1\right] = \sum_{i\in S^c}\rho_i\left(E^{S_i}\left[V^S\right] + \rho^0(S)r_i\right). \quad (37)$$

■

Remarks:

1. Notice that for $S=\mathcal{N}$ Theorem 5 yields the work decomposition law

$$\sum_{i\in\mathcal{N}} V_i^{\mathcal{N}} x_i = f(\mathcal{N}) + \sum_{i\in\mathcal{N}} V_i^{\mathcal{N}} x_i^0, \quad (38)$$

which was first derived by Boxma (1989) by a stochastic work decomposition argument. It means that the total mean workload (i.e, the total amount of service needed to clear the system of all jobs currently present) decomposes into two terms: a) the total mean workload in the corresponding work-conserving system, $f(\mathcal{N})$, and b) the total mean workload at an arbitrary time when the server is idle.

2. For $S\subset\mathcal{N}$ the work decomposition laws in Theorem 5 are new, as they do not follow from Boxma's (1989) stochastic work decomposition theorem. The assumption in Boxma's theorem that is violated here is that arrivals during idle periods for the S-queue are not Poisson, as they include jobs fed back from the S^c-queue.

3. From Theorem 5 we obtain the linear workload bound on S-jobs

$$\sum_{i\in S} V_i^S x_i \geq f(S), \quad (39)$$

valid under all dynamic policies. In addition, for systems with zero changeover times inequality (39) is satisfied at equality under any dynamic nonidling policy that gives nonpreemptive priority to S-jobs over S^c-jobs. The reason is that under such policies $x_j^{S_i}=0$ for $i\in S^c$ and

$j \in S$, and $\boldsymbol{x}^0 = \mathbf{0}$. In particular, inequality (39) holds with equality when $S = \mathcal{N}$. Therefore, for the special case of zero changeover times performance vector $\boldsymbol{x}$ satisfies the *generalized work conservation laws* introduced by Bertsimas and Niño-Mora (1996), and thus the performance region spanned by the $\boldsymbol{x}$'s under dynamic nonidling policies is the polyhedron defined by the family of inequalities (39), for $S \subset \mathcal{N}$, together with the equation $\sum_{i \in \mathcal{N}} V_i^{\mathcal{N}} x_i = f(\mathcal{N})$. Tsoucas (1991) first identified the structure of workload bounds (39), but did not evaluate the $f(S)$ function. These workload bounds generalize those discovered by Gelenbe and Mitrani (1980) for multiclass $M/GI/1$ queues without job feedback.

5.1 Strengthened workload bounds from work decomposition laws

In this section we apply the work decomposition laws in Theorem 5 to develop two new families of workload bounds, which strengthen bound (39) by incorporating the effect of positive changeover times.

The first family of workload bounds incorporates the effect of changeovers in a linear manner, and is valid under dynamic nonidling policies.

Theorem 6 (Linear workload bounds) *Under any dynamic nonidling policy,*

$$\sum_{j \in S} V_j^S x_j \geq f(S) + \frac{1}{2} \frac{\rho^0(S)}{1 - \rho^0(S)} \sum_{k,l:\ k \neq l} s_{kl}^{(2)} y_{kl}, \qquad \text{for } S \subseteq \mathcal{N}. \tag{40}$$

Proof

The result follows directly from Theorem 5(a), together with the inequalities $x_j^{S_i} \geq 0$ and inequality (16) in Proposition 2. ■

The second family of workload bounds incorporates the effect of changeovers in a nonlinear yet convex way, and is valid only under the restricted class of *static* nonidling policies.

Theorem 7 (Convex workload bounds) *Under any* static *nonidling policy the following constraints hold, for $S \subseteq \mathcal{N}$:*

(a)

$$\frac{1}{1 - \rho(S)} \left[\sum_{i \in S^c, j \in S} \rho_i V_j^S x_j^{S_i} + \rho^0(S) \sum_{i \in S^c} \rho_i r_i \right] + \frac{1 - \rho}{1 - \rho(S)} \sum_{j \in S} V_j^S x_j^0 \geq \rho^0(S) \frac{1 - \rho(S)}{2 \sum_{i \in S} y_i} \tag{41}$$

(b)

$$\sum_{j \in S} V_j^S x_j \geq f(S) - \frac{\rho^0(S)}{1 - \rho^0(S)} \sum_{i \in S^c} \rho_i r_i + \frac{(1 - \rho(S))^2 \rho^0(S)}{2 (1 - \rho^0(S))} \frac{1}{\sum_{i \in S} y_i}. \tag{42}$$

(c) *For $S = \mathcal{N}$, inequalities (41) and (42) are also valid under any* dynamic *policy:*

$$\sum_{j \in \mathcal{N}} V_j^{\mathcal{N}} x_j^0 \geq \frac{1}{2} \frac{\rho(1 - \rho)}{\sum_{j \in \mathcal{N}} y_j},$$

and

$$\sum_{j\in\mathcal{N}} V_j^{\mathcal{N}}\, x_j \geq f(\mathcal{N}) + \frac{1}{2}\,\frac{\rho(1-\rho)}{\sum_{j\in\mathcal{N}} y_j}.$$

Proof

(a) To prove inequality (41) we introduce the random variable I^S to be the length of a typical server *vacation* away from serving S-jobs. We observe that, under a *static* policy, the mean S-workload at a typical time when the server is not serving S-jobs, $E\left[V^S \mid B^S = 0\right]$, is bounded below by the external S-workload that arrives to the system from the instant the server starts a vacation away from serving S-jobs until a typical time during that vacation, i.e.,

$$E\left[V^S \mid B^S = 0\right] \geq \rho^0(S)\frac{E\left[\left(I^S\right)^2\right]}{2E\left[I^S\right]}. \tag{43}$$

Substituting (36) and (37) to the inequality (43) and using the relation

$$E\left[I^S\right] = \frac{1-\rho(S)}{\sum_{i\in S} y_i},$$

valid under nonidling policies, together with the inequality $E\left[\left(I^S\right)^2\right] \geq E\left[I^S\right]^2$, we obtain (41).

(b) This inequality reformulates that in part (a) by applying Theorem 5(a, b) together with Eqns. (35)-(37).

(c) This follows because the vacation argument used in part (a) is also valid under dynamic policies for $S = \mathcal{N}$. ■

6 Positive semidefinite constraints

We present in this section a set of *positive semidefinite constraints* that may be used to strengthen the formulations obtained through equilibrium relations. These constraints formulate the fact that the performance measures we are considering are moments of random variables. The basic idea is as follows: Given a vector $\boldsymbol{z}$ and a matrix $\boldsymbol{Z}$ of real numbers, consider the following question: What is a necessary and sufficient condition under which, for some random vector $\boldsymbol{\zeta}$, $\boldsymbol{z} = E\left[\boldsymbol{\zeta}\right]$ and $\boldsymbol{Z} = E\left[\boldsymbol{\zeta}\boldsymbol{\zeta}'\right]$? It is easily seen that the required condition is that matrix $\boldsymbol{Z} - \boldsymbol{z}\boldsymbol{z}'$ -which represents the covariance matrix of $\boldsymbol{\zeta}$- is *positive semidefinite*: $\boldsymbol{Z} - \boldsymbol{z}\boldsymbol{z}' \succeq \boldsymbol{0}$. We may reformulate this condition as

$$\begin{bmatrix} 1 & \boldsymbol{z}' \\ \boldsymbol{z} & \boldsymbol{Z} \end{bmatrix} \succeq \boldsymbol{0}.$$

Applying this idea to the performance measures introduced above yields directly the following result. Let us write $\boldsymbol{X}^N = \left(x_{kl}^N\right)_{k,l\in\mathcal{N}}$, and $\boldsymbol{x}^N = \left(x_k^N\right)_{k\in\mathcal{N}}$, for point processes $N = S_i, D_i, G_i, H_i, G_{ij}$ and H_{ij}.

Theorem 8 *Under any dynamic nonidling scheduling policy, the following positive semidefinite constraints hold:*

$$\begin{bmatrix} 1 & \boldsymbol{x}^{N\prime} \\ \boldsymbol{x}^N & \boldsymbol{X}^N \end{bmatrix} \succeq \mathbf{0}, \qquad \text{for } N = S_i, D_i, G_i, H_i, G_{ij}, H_{ij}. \tag{44}$$

Remark:

The problem of minimizing a linear objective subject to positive semidefinite constraints, called a *semidefinite programming problem*, has received considerable attention in the mathematical programming literature due to applications in discrete optimization and control theory. There are several efficient interior point algorithms (see, e.g., the survey by Vandenberghe and Boyd (1996)) to solve semidefinite programming problems. Theorem 8 adds a new and, we believe, interesting application of semidefinite programming in stochastic optimization.

7 Formulations and their power

In this section we illustrate how the constraints on performance measures derived in the previous sections can be used to provide performance bounds for single-station MQNETs with changeover times, by solving appropriate mathematical programming problems. We also compare these bounds computationally. We consider in what follows a linear cost function

$$c(\boldsymbol{x}) = \sum_{j \in \mathcal{N}} c_j x_j.$$

7.1 A nonconvex relaxation

By combining all equilibrium relations derived in previous sections we summarize the proposed bound in the following theorem.

Theorem 9 *The following optimization problem provides a lower bound for a single-station MQNET with changeover times under all dynamic nonidling policies:*

$$\begin{aligned} \underline{Z}_{nonconvex} \quad = \quad & \min c(\boldsymbol{x}) \\ & \textit{subject to} \end{aligned} \tag{45}$$

$$\boldsymbol{x} = \left(\boldsymbol{X}^D + (\boldsymbol{r} - \boldsymbol{\beta})\boldsymbol{\alpha}' + \boldsymbol{I} - \boldsymbol{P}\right)' \boldsymbol{\rho} + (1 - \rho)\boldsymbol{x}^0, \tag{46}$$

$$-\boldsymbol{\alpha}\boldsymbol{x}' - \boldsymbol{x}\boldsymbol{\alpha}' + (\boldsymbol{I} - \boldsymbol{P})'\boldsymbol{\Lambda}\boldsymbol{X}^D + \boldsymbol{X}^{D\prime}\boldsymbol{\Lambda}(\boldsymbol{I} - \boldsymbol{P}) = (\boldsymbol{I} - \boldsymbol{P})'\boldsymbol{\Lambda}\boldsymbol{P} + \boldsymbol{P}'\boldsymbol{\Lambda}(\boldsymbol{I} - \boldsymbol{P}). \tag{47}$$

$$\boldsymbol{X}^D \geq \boldsymbol{\beta}\boldsymbol{\alpha}' + \boldsymbol{P} - \boldsymbol{I}, \tag{48}$$

$$x_j^0 = \sum_{k,l:\ k\neq l} \frac{s_{kl}y_{kl}}{1-\rho}\left(x_j^{H_{kl}} + \alpha_j \frac{s_{kl}^{(2)}}{2s_{kl}}\right). \tag{49}$$

$$\sum_{i,j:\ i\neq j} s_{ij}y_{ij} = 1-\rho. \tag{50}$$

$$\sum_{j\in\mathcal{N}\setminus\{i\}} y_{ij} = \sum_{j\in\mathcal{N}\setminus\{i\}} y_{ji}, \qquad \textit{for } i\in\mathcal{N}. \tag{51}$$

$$\sum_{k\in\mathcal{N}\setminus\{i\}} \left(y_{ik}x_j^{H_{ik}} - y_{ki}(x_j^{H_{ki}} + \alpha_j s_{ki})\right) = (\alpha_j\beta_i + p_{ij} - \delta_{ij})\,\lambda_i, \qquad \textit{for } i,j\in\mathcal{N} \tag{52}$$

$$\begin{aligned}
&\sum_{r\in\mathcal{N}\setminus\{i\}} \left(y_{ir}x_{jk}^{H_{ir}} - y_{ri}(x_{jk}^{H_{ri}} + \alpha_j s_{ri}x_k^{H_{ri}} + \alpha_k s_{ri}x_j^{H_{ri}} + \alpha_j\alpha_k s_{ri}^{(2)})\right) = \\
&\qquad \lambda_i\Bigg[\left(\alpha_k\beta_i + p_{ik} - \delta_{ik}\right)x_j^{D_i} + \left(\alpha_j\beta_i + p_{ij} - \delta_{ij}\right)x_k^{D_i} \\
&\quad -\left(\alpha_j(p_{ik}-\delta_{ik}) + \alpha_k(p_{ij}-\delta_{ij}) + \alpha_j\delta_{kj}\right)\beta_i + \alpha_j\alpha_k\left(\beta_i^{(2)} - 2\beta_i^2\right) \\
&\qquad p_{ij}\delta_{ik} + p_{ik}\delta_{ij} + p_{ij}\delta_{kj} - \delta_{ij}\delta_{ik} - 2p_{ik}p_{ij}\Bigg] \qquad \textit{for } i,j,k\in\mathcal{N}.
\end{aligned} \tag{53}$$

$$\begin{bmatrix} \mathbf{1} & \boldsymbol{x}^{H_{ij}\prime} \\ \boldsymbol{x}^{H_{ij}} & \boldsymbol{X}^{H_{ij}} \end{bmatrix} \succeq \mathbf{0}, \qquad \textit{for } i,j\in\mathcal{N}. \tag{54}$$

$$\boldsymbol{x}, \boldsymbol{x}^0, \boldsymbol{Y}, \boldsymbol{X}^D, \boldsymbol{x}^{H_{ij}}, \boldsymbol{X}^{H_{ij}} \geq \mathbf{0}.$$

Proof

Eq. (46) follows by combining (1) and (4).

Eqns. (47), (49), (50), and (51) are exactly Eqns. (5), (15), (17), and (18) respectively.

Inequalities (48) follow from Eq. (2) using $\boldsymbol{X}^S \geq \mathbf{0}$.

Eq. (52) results by substituting Eqns. (24), (25) and (26) into Eq. (20).

Similarly, Eq. (53) results by substituting Eqns. (27), (28) and (29) into Eq. (21).

Finally, the semidefinite constraint (54) is obtained from (44) for $N = H_{ij}$. ■

Remark: Constraints (46) express performance measure $\boldsymbol{x}$ in terms of the auxiliary variables $\boldsymbol{X}^D$ and $\boldsymbol{x}^0$. Relations (47) provide additional constraints linking $\boldsymbol{x}$ and $\boldsymbol{X}^D$. Relations (48) provide lower bounds on $\boldsymbol{X}^D$. Relations (49) express the variables $\boldsymbol{x}^0$ in terms of the auxiliary variables $\boldsymbol{Y}$ and $\boldsymbol{x}^{H_{ij}}$, $i,j\in\mathcal{N}$. Relations (50) and (51) provide constraints on $\boldsymbol{Y}$. Relations (52) link the variables $\boldsymbol{Y}$ and $\boldsymbol{x}^{H_{ij}}$, $i,j\in\mathcal{N}$. Relations (53) link the new variables $\boldsymbol{X}^{H_{ij}}$ and $\boldsymbol{X}^D$. Finally, semidefinite constraints (54) link the variables $\boldsymbol{x}^{H_{ij}}$ and $\boldsymbol{X}^{H_{ij}}$.

On the solution of problem (45)

The optimization problem (45) is a nonlinear programming problem involving $O(n^4)$ variables. Unfortunately it is nonconvex due to the presence of the products $y_{ki}x_j^{H_{ki}}$. However, if the values of $\boldsymbol{Y}$ are known, the problem (in the remaining variables $\boldsymbol{w} = (\boldsymbol{x}, \boldsymbol{X}^D, \boldsymbol{x}^{H_{ij}}, \boldsymbol{X}^{H_{ij}})$) is a convex semidefinite problem. Conversly, if the variables $\boldsymbol{w}$ are known, problem (45) is a linear optimization problem. Given that there are very efficient algorithms to solve linear and semidefinite optimization problems, we can fix $\boldsymbol{Y}$, solve for $\boldsymbol{w}$, then given $\boldsymbol{w}$ solve for $\boldsymbol{Y}$ and so on. More formally we propose the following algorithm:

Iterative Algorithm for solving (45)

1. **Initialization.** Start with a given $\boldsymbol{Y}_0$; set $\boldsymbol{w}_0 = \boldsymbol{0}$; set $k = 0$; fix $\epsilon > 0$;

2. **Semidefinite optimization problem.** Let $\boldsymbol{Y} = \boldsymbol{Y}_0$; with $\boldsymbol{Y}$ fixed, solve the resulting semidefinite optimization problem (45) for the variables $\boldsymbol{w}$; let $\overline{\boldsymbol{w}}$ the resulting optimal solution; set $\boldsymbol{w}_{k+1} = \overline{\boldsymbol{w}}$;

3. **Linear optimization problem.** Let $\boldsymbol{w} = \overline{\boldsymbol{w}}$; with $\boldsymbol{w}$ fixed, solve the resulting linear optimization problem (45) for the variables $\boldsymbol{Y}$. Let $\overline{\boldsymbol{Y}}$ the resulting optimal solution; set $\boldsymbol{Y}_{k+1} = \overline{\boldsymbol{Y}}$;

4. **Convergence test.** If $||\boldsymbol{w}_{k+1} - \boldsymbol{w}_k|| \leq \epsilon$ and $||\boldsymbol{Y}_{k+1} - \boldsymbol{Y}_k|| \leq \epsilon$, stop; else set k:=k+1 and go to step 2;

In Section 8 we illustrate that for specific classes of policies, like polling table and randomized routing policies, we can calculate explicitly in terms of the original data the variables $\boldsymbol{Y}$ and $\boldsymbol{y}$. As we noticed earlier, problem (45) becomes a (convex) semidefinite optimization problem for which there are very efficient algorithms. As we discuss in Section 9, when solving problem (45), we calculate the optimal values $\hat{x}_j^{H_j}$ of first order moments of queue lengths at visit completion epochs. These values give rise to the following policy: If the server is currently serving j-jobs, terminate the current visit when the queue length after a service completion does not exceed $\hat{x}_j^{H_j}$, i.e., $L_j \leq \hat{x}_j^{H_j}$. Notice that if $\hat{x}_j^{H_j} = 0$, then the server follows an exhaustive policy at the j-queue. In this way, given a class of policies in which we can calculate the routing variables $\boldsymbol{Y}$, we can find a threshold policy for servicing jobs in the system by solving a convex semidefinite optimization problem.

7.2 Convex relaxations

In this section we propose several simpler but convex relaxations that can be used as an alternative to problem (45).

Theorem 10 *The following optimization problems provide lower bounds for a single-station MQNET with changeover times under the classes os policies specified:*

(a) The linear optimization problem (under dynamic nonidling policies)

$$\underline{Z}_{linear} = \min c(\boldsymbol{x})$$

$$\text{subject to } (46), (47), (48), (50), (51), (16)$$

$$\boldsymbol{x}, \boldsymbol{x}^0, \boldsymbol{X}^D, \boldsymbol{Y} \geq \boldsymbol{0},$$

involving $O(n^2)$ variables and constraints.

(b) The convex optimization problem (under static nonidling policies)

$$\underline{Z}_{convex1} = \min c(\boldsymbol{x})$$

$$\text{subject to } (46), (47), (48), (50), (51), (16)$$

$$\frac{\rho(S^c)}{1-\rho(S)} \left[\sum_{i \in S^c, j \in S} \rho_i V_j^S (x_j^{D_i} - \beta_i \alpha_j - p_{ij}) + \rho^0(S) \sum_{i \in S^c} \rho_i r_i \right]$$

$$+ \frac{1-\rho}{1-\rho(S)} \sum_{j \in S} V_j^S x_j^0 \geq \rho^0(S) \frac{1-\rho(S)}{2 \sum_{i \in S} \sum_{j \in \mathcal{N} \setminus \{i\}} y_{ij}} \quad S \subseteq \mathcal{N} \tag{55}$$

$$\boldsymbol{x}, \boldsymbol{x}^0, \boldsymbol{X}^D, \boldsymbol{Y} \geq \boldsymbol{0},$$

involving $O(n^2)$ variables but an exponential number of convex constraints.

(c) The convex optimization problem (under static nonidling policies)

$$\underline{Z}_{convex2} = \min c(\boldsymbol{x}) \tag{56}$$

$$\text{subject to } (40), (42), (17), (18)$$

$$\boldsymbol{x}, \boldsymbol{Y}, \boldsymbol{y} \geq \boldsymbol{0},$$

involving also $O(n^2)$ variables but an exponential number of convex constraints.

Proof

(a) By maintaining only the linear constraints in the formulation (45) and using (16) instead of (49) the bound $\underline{Z}_{linear}$ is obtained.

(b) The bound $\underline{Z}_{convex1}$ is obtained from $\underline{Z}_{linear}$ by adding the family of convex constraints (41). Notice that we used Eq. (2) to relate variables $x_j^{S_i}$ and $x_j^{D_i}$. Moreover, we used the relation $y_i = \sum_{j \in \mathcal{N} \setminus \{i\}} y_{ij}$ (Eq. (18)).

(c) The bound $\underline{Z}_{convex2}$ is immediate from Theorem 6. ■

Remark: In terms of their strength, the proposed relaxations can be ordered as follows:

$$\underline{Z}_{linear} \leq \underline{Z}_{convex2} \leq \underline{Z}_{convex1} \leq \underline{Z}_{nonconvex},$$

i.e., there is a tradeoff between computational requirements and the strength of the bound.

ρ	$\underline{Z}_{linear}$	$\underline{Z}_{convex2}$	$\underline{Z}_{convex1}$	$\frac{\text{Bestpriority}}{\underline{Z}_{convex1}}$
0.05	1.31	1.36	1.36	1.03
0.10	1.52	1.59	1.59	1.03
0.20	1.92	2.04	2.07	1.05
0.40	2.84	2.89	3.20	1.07
0.60	4.32	4.51	5.03	1.09
0.80	8.93	9.31	10.61	1.09
0.90	17.31	19.43	21.33	1.13
0.95	63.25	67.41	69.57	1.18

Table 2: The effect of the traffic intensity on the quality of the bounds.

7.3 Computational Results

In order to compare our bounds computationally we designed the following computational experiment. There are $N = 4$ classes of jobs. The changeover times are exponentially distributed, with mean changeover time from class i to class j equal to one third of the service time of class i. Service times are distributed according to a mixed generalized Erlang distribution with two stages, and coefficient of variation two. Costs are chosen so that $c_i = 2c_{i+1}$, and mean service times are such that $\beta_i = 0.5\beta_{i+1}$. Arrivals are Poisson, and are varied in the same proportion to change the traffic intensity ρ. There is no feedback.

Table 2 summarizes the effect of the traffic intensity on the quality of the bounds. The value $\underline{Z}_{linear}$ was computed using CPLEX 5.0 on a Sparc20 Sun workstation, and $\underline{Z}_{convex2}$, $\underline{Z}_{convex1}$ were computed using NPSOL on a Sparc20 Sun workstation. In order to compare the bounds to a reasonable feasible policy, we have simulated all $4! = 24$ possible static priority policies, and we report the performance of the best priority policy found. The key insight is that as the system approaches heavy traffic the degree of suboptimality worsens. While this might be due to the fact that the quality of the bounds weakens as the traffic intensity increases, we suspect that this is due to the fact that the performance of static priority policies worsens in heavy traffic.

Our next experiment aims at understanding the effect of the changeover times. We again have 4 classes, and we have selected parameters as in the previous experiment. We fixed $\rho = 0.8$. All the changeover times were selected equal to the same value D. In Table 3 we report the performance of the bounds in relation to the best static priority policy as D increases. The quality of the bounds is relatively insensitive to the value of D, although they slightly worsen with higher changeover times.

D	$\underline{Z}_{linear}$	$\underline{Z}_{convex2}$	$\underline{Z}_{convex1}$	$\frac{\text{Bestpriority}}{\underline{Z}_{convex1}}$
0.01	3.93	3.95	3.95	1.061
0.10	4.13	4.33	4.33	1.063
0.50	5.21	5.45	5.49	1.067
1.00	8.17	8.67	8.93	1.069
2.00	12.49	13.04	13.48	1.072
5.00	23.87	24.91	25.17	1.079
10.00	44.28	46.26	47.14	1.081

Table 3: The effect of changeover times to the quality of the bounds.

8 Performance analysis

In this section we show that the flow conservation and server dynamics constraints, developed in Sections 3 and 4 respectively, yield a unified method of performance analysis for the models and policies analyzed exactly in the literature. In particular, we shall show that the system of equations given by those constraints corresponds precisely to the classical *buffer occupancy approach* for the performance analysis of polling systems.

The buffer occupancy approach was introduced by Cooper and Murray (1969) to analyze cyclic service systems with zero changeover times. The approach has since been developed to analyze a variety of single-server systems with nonzero changeover times, including systems with a periodic service order (see Eisenberg (1972)), systems with a random service order (see Kleinrock and Levy (1988)), and systems with job feedback (see Sidi, Levy and Fuhrmann (1992)). See also the monograph by Takagi (1986) and the survey by Levy and Sidi (1990).

We shall illustrate the approach by analyzing the constraints in the following special cases: the server visits the queues according to a cyclic, polling table or random service order policy, and the service at each queue is exhaustive. The approach consists of two stages: The first stage determines the visit and changeover frequencies. The second stage computes the mean queue lengths at an arbitrary time.

8.1 Stage 1: Computation of visit and changeover frequencies

We show next that when the server visits the queues according to a cyclic order, polling table or random visit order policy, then we can compute efficiently the visit and changeover frequencies, y_i, y_{ij}.

Cyclic order policy. Under a *cyclic service* policy the server visits the queues corresponding to different classes in cyclic order, say, $1, 2, \ldots, n, 1, \ldots$. It is easily seen from a symmetry argument

and Proposition 2 that the changeover and visit frequencies are given by

$$y_i = y_{12} = \cdots = y_{n-1,n} = y_{n1} = \frac{1-\rho}{s_{12} + \cdots + s_{n-1,n} + s_{n1}}.$$

Polling table order policy. Under a *polling table* policy the server visits queues in a periodic order given by a sequence $T(1), T(2), \ldots, T(m)$, i.e., it first visits class $T(1)$, then class $T(2)$, etc. Let $M(i,j)$ be the number of $i \to j$ changeovers and let $M(i)$ be the number of server visits to the i-queue during a cycle in the polling table. Again, a symmetry argument and Proposition 2 yield

$$y_{ij} = (1-\rho)\frac{M(i,j)}{\sum_{k \neq l} s_{kl} M(k,l)},$$

and

$$y_i = (1-\rho)\frac{M(i)}{\sum_{k \neq l} s_{kl} M(k,l)}.$$

Random visit order policy. Under a *random visit order* policy (see Kleinrock and Levy (1988)) the server visits the queues in a random order according to Bernoulli routing probabilities: after visiting the i-queue, it decides to visit next the j-queue with probability r_{ij}. Clearly, we have $y_{ij}/y_i = r_{ij}$. Combining this relation with Proposition 2 we obtain that the visit and changeover frequencies are determined by solving the linear system

$$\sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N} \setminus \{i\}} s_{ij} r_{ij} y_i = 1 - \rho,$$

$$y_i = \sum_{j \in \mathcal{N} \setminus \{i\}} r_{ji} y_j, \qquad \text{for } i \in \mathcal{N}.$$

8.2 Stage 2: Computation of mean queue lengths

Once the visit and changeover frequencies have been determined, the server dynamics constraints developed in Section 4 become linear. We show next how those constraints yield an exact performance analysis, by focusing our attention in the special case that the server follows an *exhaustive service* policy at each queue (the server continues serving a class until the corresponding queue becomes empty), and the order in which the server visits the queues is either cyclic or in random order.

In order to compute the vector of mean queue lengths at an arbitrary time, $\boldsymbol{x}$, we propose the following three-step procedure:

Step 1. Compute the first moments of queue lengths at server visit completion epochs ($x_j^{H_i}$) by solving the linear system of size $O(n^2)$

$$\sum_{k \in \mathcal{N} \setminus \{i\}} \left(y_{ik} x_j^{H_i} - y_{ki} (x_j^{H_k} + \alpha_j s_{ki}) \right) = (\alpha_j \beta_i + p_{ij} - \delta_{ij})\, \lambda_i, \qquad \text{for } i, j \in \mathcal{N},$$

with the additional constraints

$$x_j^{H_j} = 0, \qquad \text{for } j \in \mathcal{N}.$$

Step 2. Compute the second moments of queue lengths at server visit completion epochs ($x_{ij}^{H_k}$), and the first moments of queue lengths at service completion epochs ($x_j^{D_i}$) by solving the linear system of size $O(n^3)$

$$\sum_{r\in\mathcal{N}\setminus\{i\}} \left(y_{ir}x_{jk}^{H_i} - y_{ri}(x_{jk}^{H_r} + \alpha_j s_{ri}x_k^{H_r} + \alpha_k s_{ri}x_j^{H_r} + \alpha_j\alpha_k s_{ri}^{(2)})\right) =$$

$$\lambda_i\Bigg[\left(\alpha_k\beta_i + p_{ik} - \delta_{ik}\right)x_j^{D_i} + \left(\alpha_j\beta_i + p_{ij} - \delta_{ij}\right)x_k^{D_i}$$

$$-\left(\alpha_j(p_{ik}-\delta_{ik}) + \alpha_k(p_{ij}-\delta_{ij}) + \alpha_j\delta_{kj}\right)\beta_i + \alpha_j\alpha_k\left(\beta_i^{(2)} - 2\beta_i^2\right)$$

$$p_{ij}\delta_{ik} + p_{ik}\delta_{ij} + p_{ij}\delta_{kj} - \delta_{ij}\delta_{ik} - 2p_{ik}p_{ij}\Bigg] \qquad \text{for } i,j,k \in \mathcal{N}.$$

with the additional constraints

$$x_{jk}^{H_j} = 0, x_{jk}^{H_k} = 0, \qquad \text{for } j,k \in \mathcal{N}.$$

Step 3. Compute the mean queue lengths at an arbitrary time (x_j) from the n equations corresponding to the diagonal elements of the flow conservation constraints

$$-\boldsymbol{\alpha x}' - \boldsymbol{x\alpha}' + (\boldsymbol{I}-\boldsymbol{P})'\boldsymbol{\Lambda X}^D + \boldsymbol{X}^{D\,\prime}\boldsymbol{\Lambda}(\boldsymbol{I}-\boldsymbol{P}) = (\boldsymbol{I}-\boldsymbol{P})'\boldsymbol{\Lambda P} + \boldsymbol{P}'\boldsymbol{\Lambda}(\boldsymbol{I}-\boldsymbol{P}).$$

Remarks.

1. The equations in Steps 1 and 2 in the above procedure follow from Eqns. (52) and (53) by using the relations

$$x_k^{H_{ij}} = x_k^{H_i}, x_{ij}^{H_{kl}} = x_{ij}^{H_k},$$

valid under cyclic service and random visit order policies.

2. The constraints

$$x_j^{H_j} = 0, x_{jk}^{H_j} = 0, x_{jk}^{H_k} = 0,$$

are clearly valid when the server follows an exhaustive service policy at each queue.

3. The systems of linear equations in Steps 1 and 2 correspond precisely to those obtained via the buffer occupancy approach.

4. Notice that in Step 3 performance measure $\boldsymbol{x}$ is computed without first computing $\boldsymbol{x}^0$, as was done in Sidi, Levy and Fuhrmann (1992).

9 Design of efficient heuristic policies

We briefly discuss in this section the problem of designing heuristic scheduling policies that nearly optimize a given performance objective from the solution of the relaxations, as described in Section 2. First, notice that a scheduling policy is characterized by specifying the following two subpolicies:

1. A *server routing* policy, that governs *where* the server is routed when it leaves a queue.

2. A *server visit* policy, that specifies *when* the server finishes serving jobs in a queue.

Server routing policies. Several authors have recently addressed the problem of designing efficient server routing policies from estimates of optimal visit and changeover frequencies. Boxma, Levy and Weststrate (1991) have developed a methodology for determining efficient polling tables, in which the critical step is the determination of nearly optimal visit frequencies (our y_i's). Bertsimas and Xu (1993) have proposed two kinds of server routing policies from estimates of optimal changeover frequencies (our y_{ij}'s): 1) a random visit order policy, with server routing probabilities given by $r_{ij} = y_{ij}/y_i$, and 2) a polling table policy, where the parameters $M(i,j)$ of the polling table, as defined in the previous section, are obtained via *integer programming* techniques.

It is intuitive that the quality of these heuristic policies should improve with better estimates of the optimal visit and changeover frequencies, as provided by the improved formulations presented in Section 7.

Server visit policies. No previous studies known to the authors have addressed the problem of designing efficient server visit policies from estimates of optimal system performance measures. An approach for designing policies based on a heavy traffic approximation of the model has recently been developed by Reiman and Wein (1994), in the setting of a two-class queue with setup times. We propose two different policies, which use estimates $\hat{x}_j^{H_j}$ of optimal mean queue lengths at visit completion epochs (obtained by solving problem (45) and estimates $\hat{y_i}$ of optimal visit frequencies, respectively: The first policy terminates the server visit to the current queue, say queue j, as soon as the queue length after a service completion epoch does not exceed $\hat{x}_j^{H_j}$, i.e., $L_j \leq \hat{x}_j^{H_j}$. Notice that if $\hat{x}_j^{H_j} = 0$, then the server follows an exhaustive policy at the j-queue. The second policy terminates a visit to the current queue, say queue j, after completing at most $\lceil \lambda_i/\hat{y}_i \rceil$ services. Notice that λ_i/y_i is the mean number of j-jobs served during a server visit to the j-queue.

10 Conclusions

We have studied by means of the achievable region approach an important and hard performance optimization problem: optimal scheduling in a multiclass single-station queueing network with

changeover times. We have used several equilibrium relations to derive constraints on achievable performance. We believe that the results presented above support the claim that the achievable region approach is an effective tool for obtaining performance bounds in stochastic optimization problems.

Acknowledgement

We would like to thank Jay Sethuraman for performing the computational experiments reported in Tables 2 and 3.

References

[1] Baccelli, F. and Brémaud, P. (1994). *Elements of Queueing Theory: Palm-Martingale Calculus and Stochastic Recurrences*. Springer-Verlag, Berlin.

[2] Bertsimas, D. (1995). The achievable region method in the optimal control of queueing systems; formulations, bounds and policies. *Queueing Syst.* **21** 337-389.

[3] Bertsimas, D. and Niño-Mora, J. (1994). Restless bandits, linear programming relaxations and a primal-dual heuristic. Working paper 3727-94 MSA, Sloan School of Management, MIT.

[4] Bertsimas, D. and Niño-Mora, J. (1996). Conservation laws, extended polymatroids and multi-armed bandit problems; a polyhedral approach to indexable systems. *Math. Oper. Res.* **21** 257-306.

[5] Bertsimas, D. and Niño-Mora, J. (1996a). Optimization of multiclass queueing networks with changeover times via the achievable region approach: Part II, the multiple-station case. Working paper, Operations Research Center, MIT.

[6] Bertsimas, D., Paschalidis, I. and Tsitsiklis, J. (1994). Optimization of multiclass queueing networks: Polyhedral and nonlinear characterizations of achievable performance. *Ann. Appl. Probab.* **4** 43-75.

[7] Bertsimas, D., Paschalidis, I. C. and Tsitsiklis, J. N. (1995). Branching bandits and Klimov's problem: Achievable region and side constraints. *IEEE Trans. Automat. Control* **40** 2063-2075.

[8] Bertsimas, D. and Xu, H. (1993). Optimization of polling systems and dynamic vehicle routing problems on networks. Working paper, Operations Research Center, MIT.

[9] Boxma, O. J. (1989). Workloads and waiting times in single-server systems with multiple customer classes. *Queueing Syst.* **5** 185-214.

[10] Boxma, O. J. (1995). Static optimization of queueing systems. R. P. Agarwal, ed., *Recent Trends in Optimization Theory and Applications,* World Scientific Publishing.

[11] Boxma, O. J., Levy, H. and Weststrate, J. A. (1991). Efficient visit frequencies for polling tables: Minimization of waiting cost. *Queueing Syst.* **9** 133-162.

[12] Burke, P. J. (1956). The output of a queueing system. *Oper. Res.* **4** 699-704.

[13] Buzacott, J. A. and Shanthikumar, J. G. (1993). *Stochastic Models of Manufacturing Systems.* Prentice Hall, Englewood Cliffs, NJ.

[14] Coffman, E. G., Jr. and Mitrani, I. (1980). A characterization of waiting time performance realizable by single server queues. *Oper. Res.* **28** 810-821.

[15] Cooper, R. B. and Murray, G. (1969). Queues served in cyclic order. *Bell Syst. Tech. J.* **48** 675-689.

[16] Eisenberg, M. (1972). Queues with periodic service and changeover time. *Oper. Res.* **20** 440-451.

[17] Federgruen, A. and Groenevelt, H. (1988). Characterization and optimization of achievable performance in general queueing systems. *Oper. Res.* **36** 733-741.

[18] Fuhrmann, S. W. and Cooper, R. B. (1985). Stochastic decompositions in the $M/G/1$ queue with generalized vacations. *Oper. Res.* **33** 1117-1129.

[19] Gelenbe, E. and Mitrani, I. (1980). *Analysis and Synthesis of Computer Systems.* Academic Press, London.

[20] Gupta, D. and Buzacott, J. A. (1990). A production system with two job classes, changeover times and revisitation. *Queueing Syst.* **6** 353-368.

[21] Kelly, F. P. (1979). *Reversibility and Stochastic Networks*, Wiley, New York.

[22] Kleinrock, L. and Levy, H. (1988). The analysis of random polling systems. *Oper. Res.* **36** 716-732.

[23] Klimov, G. P. (1974). Time sharing service systems I. *Theory Probab. Appl.* **19** 532-551.

[24] Kumar, S. and Kumar, P. R. (1994). Performance bounds for queueing networks and scheduling policies. *IEEE Trans. Autom. Control* **39** 1600-1611.

[25] Levy, H. and Sidi, M. (1990). Polling systems: Applications, modeling, and optimization. *IEEE Trans. Comm.* **38** 1750-1760.

[26] Lovász, L. and Schrijver, A. (1991). Cones of matrices and set-functions and $0-1$ optimization. *SIAM J. Optim.* **1** 166-190.

[27] Niño-Mora, J. (1995). *Optimal Resource Allocation in a Dynamic and Stochastic Environment: A Mathematical Programming Approach.* PhD Dissertation, Sloan School of Management, MIT.

[28] Papadimitriou, C. H. and Tsitsiklis, J. N. (1994). The complexity of optimal queueing network control. Working Paper LIDS 2241, MIT.

[29] Papangelou, F. (1972). Integrability of expected increments and a related random change of time scale. *Trans. Amer. Math. Soc.* **165** 483-506.

[30] Reiman, M. I. and Wein, L. M. (1994). Dynamic scheduling of a two-class queue with setups. Working paper, Sloan School of Management, MIT.

[31] Shanthikumar, J. G. and Yao, D. D. (1992). Multiclass queueing systems: Polymatroidal structure and optimal scheduling control. *Oper. Res.* **40** S293-299.

[32] Sidi, M., Levy, H. and Fuhrmann, S. W. (1992). A queueing network with a single cyclically roving server. *Queueing Syst.* **10** 121-144.

[33] Takagi, H. (1986). *Analysis of polling systems.* MIT Press, Cambridge, MA.

[34] Tsoucas, P. (1991). The region of achievable performance in a model of Klimov. Research Report RC16543, IBM T.J. Watson Research Center, Yorktown Heights, NY.

[35] Vandenberghe, L. and Boyd, S. (1996). Semidefinite programming. *SIAM Review* **38** 49-95.