Optimization of multiclass queueing networks with changeover times via the achievable region method: Part II, the multi-station case

Dimitris Bertsimas * José Niño-Mora **

September 1996, revised August 1998

*Dimitris Bertsimas, Sloan School of Management and Operations Research Center, Rm E53-363, MIT, Cambridge, MA 02139, dbertsim@mit.edu. This research was partially supported by grants from the Leaders for Manufacturing program at MIT, a Presidential Young Investigator Award DDM-9158118 with matching funds from Draper Laboratory, and NSF grant DMI-9610486. 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, jnimora@alum.mit.edu. Part of this research was performed during the author's stay at the Operations Research Center of MIT as a PhD student and a Postdoctoral Associate.

Abstract

We address the problem of scheduling a multi-station multiclass queueing network (MQNET) with server changeover times to minimize steady-state mean job holding costs. We present new lower bounds on the best achievable cost that emerge as the values of mathematical programming problems (linear, semidefinite, and convex) over relaxed formulations of the system's achievable performance region. The constraints on achievable performance defining these formulations are obtained by formulating system's equilibrium relations. Our contributions include: (1) a flow conservation interpretation and closed formulae for the constraints previously derived by the potential function method; (2) new work decomposition laws for MQNETs; (3) new constraints (linear, convex, and semidefinite) on the performance region of first and second moments of queue lengths for MQNETs; (4) a fast bound for a MQNET with N customer classes computed in N steps; (5) two heuristic scheduling policies: a priority-index policy, and a policy extracted from the solution of a linear programming relaxation.
Keywords: Multiclass queueing network, changeover times, optimal scheduling, performance region, linear programming relaxation, semidefinite programming, convex programming.
JEL classification: C61, C63.

1 Introduction

Multiclass queueing networks (MQNETs) provide a rich range of models for complex service systems in application areas that include manufacturing (see Buzacott and Shanthikumar (1993)) and computer-communication systems (see Gelenbe and Mitrani (1980)). The practical needs to evaluate and improve the performance of such systems have motivated extensive research efforts on the analysis, optimization and stability of MQNETs.

Most relevant MQNET models have not yielded an exact *performance analysis* (evaluating the system performance under a scheduling policy). This has only been achieved in a restricted range of models, such as product-form MQNETs (see Kelly (1979)), and certain single-server priority and polling systems (see Levy and Sidi (1990)). A more feasible research objective for those seemingly intractable MQNETs is to obtain *performance bounds* which can be efficiently computed. These bounds may be used to approximate the performance of a given scheduling policy, and to assess its suboptimality gap with respect to a performance objective.

The *performance optimization* problem (computing the optimal system performance under a range of scheduling policies, and finding a policy that achieves it) also appears computationally intractable in most MQNET models, as shown by Papadimitriou and Tsitsiklis (1994). Exact results have only been achieved in a range of systems that satisfy certain *work conservation* laws: for them simple priority-index policies have been shown to optimize linear performance objectives (see Bertsimas and Niño-Mora (1996a)). In more complex MQNETs researchers have focused their efforts on designing *heuristic* scheduling policies that exhibit a good empirical performance (see, e.g., Wein (1990)).

An important modeling feature that is absent in most studies on MQNETs with multiple service stations is the inclusion of *changeover times* (which a server incurs when changing service from one class to another). This is in contrast with the rather vast literature on single-station models with changeover times (usually called *polling systems*; see the survey by Levy and Sidi (1990)).

In this paper we address the performance optimization problem in multi-station MQNETs with changeover times by means of the *achievable region approach*, with the objective of developing a systematic method for computing performance bounds and designing scheduling policies that nearly optimize performance objectives. We have investigated the corresponding problem for single-station MQNETs in a companion paper (see Bertsimas and Niño-Mora (1998)).

The achievable region approach to performance optimization of queueing systems. The achievable region approach to performance optimization, surveyed in Bertsimas (1995), was introduced by Coffman and Mitrani (1980). It draws on the mathematical programming approach to optimization, as it seeks to characterize the *performance region* achievable by a system performance measure under a class of *admissible* scheduling policies. The goal is to formulate explicitly this region

by means of equality and inequality constraints. Since it may not be possible to formulate the exact performance region, we may have to settle for constructing a *relaxation* that contains it.

Coffman and Mitrani (1980) first addressed with this approach the problem of minimizing the class-weighted mean delay in a multiclass $M/M/1$ queue. They formulated exactly the system performance region as a polyhedron, and showed that the known optimality of priority-index policies (the $c\mu$-rule) follows from structural properties of this underlying polyhedron. The scope of the approach has since been extended to tackle a range of increasingly more complex systems. Drawing on earlier work by Federgruen and Groenevelt (1988) and Shanthikumar and Yao (1992), Bertsimas and Niño-Mora (1996) developed a unified approach for formulating the exact performance region in a wide variety of MQNETs that satisfy work conservation laws. They established that the strong structural properties of these performance optimization problems (optimality of priority-index policies) are a consequence of corresponding properties of their underlying polyhedral performance regions.

Researchers have sought recently to extend further the scope of the achievable region approach, with the aim of solving computationally hard performance optimization problems: restless bandits (see Bertsimas and Niño-Mora (1994)) and MQNETs (see Bertsimas, Paschalidis and Tsitsiklis (1994)-(1995) and Kumar and Kumar (1994)).

The two critical problems the achievable region approach needs to overcome when tackling a performance optimization problem are (a) generating constraints on the performance region, and (b) designing effective policies from the solution of the corresponding relaxations.

Regarding the first problem, an idea that has proven fruitful is to generate constraints by formulating stochastic *equilibrium relations* satisfied by the system. The kinds of equilibrium relations that have been so far used in the literature include the following:

1. *Work conservation laws*, which hold in single-server MQNETs under nonidling policies (the server never stops working when there are jobs in the system). These laws lead to an *exact* polyhedral characterization of the performance region (see Bertsimas and Niño-Mora (1996a)).
2. *Work decomposition laws*, which hold in single-server MQNETs that allow server idleness (such as that caused by changeover times). Bertsimas and Xu (1993), and Bertsimas and Niño-Mora (1998) have shown that these laws yield a *convex relaxation* of the system performance region, from which they obtain bounds and policies.
3. *Potential function recursions*, as developed by Bertsimas, Paschalidis and Tsitsiklis (1994)-(1995), and by Kumar and Kumar (1994). The use of potential functions has proven to be a powerful tool for generating a sequence of increasingly tighter polyhedral relaxations for Markovian MQNETs.

Although they have proven their value as powerful tools for generating constraints, the above approaches exhibit certain limitations:

1. The approach based on formulating work conservation laws is restricted to work-conserving systems, thus excluding systems with server changeover times, and multi-station MQNETs.
2. The approach based on formulating work decomposition laws has only been developed in single-server systems (see Bertsimas and Niño-Mora (1998)).
3. The potential function method is algebraic in nature: it does not provide a physical insight into the reason of its success.

The problem of designing in a systematic way effective scheduling policies for intractable MQNETs from the solution of the relaxations remains an open challenge. Previous work in this direction includes the dual-index policy proposed in Bertsimas and Niño-Mora (1994) for the restless bandit problem, and the policies for polling systems proposed in Bertsimas and Xu (1993) and in Bertsimas and Niño-Mora (1998).

Objective and contributions. Our objective in this paper is to support the thesis that the achievable region approach is an effective tool for solving hard performance optimization problems. We shall test this thesis by tackling via the approach the performance optimization problem in an open multi-station MQNET model with changeover times. In Bertsimas and Niño-Mora (1998) we address the corresponding problem in a single-station MQNET model with changeover times.

Our contributions include:

1. We develop *new constraints* on performance measures by formulating different kinds of equilibrium relations than those considered previously in the literature.
2. We reveal the physical origin of the constraints given by the potential function method, as formulating the classical *flow conservation law* of queueing theory $L^- = L^+$. This understanding leads to explicit and simple formulas for all higher order relaxations.
3. We provide the first known explicit relaxation for the performance region of second moments of queue lengths in a multi-station MQNET. The relaxation is a *semidefinite programming* problem, for which efficient (polynomial time) algorithms have been developed in recent years.
4. As a byproduct of the flow conservation constraints, we obtain directly *new work decomposition laws* for multi-station MQNETs. From these laws we derive a family of convex constraints that account explicitly for the effect of changeover times.
5. We adapt Klimov's one-pass algorithm for computing fast index-based performance bounds for MQNETS.
6. We propose *heuristic scheduling policies* based on the solution of the relaxations. First, we apply the flow conservation law appropriately in order to obtain relaxations for MQNETs with finite buffers, from which one can naturally extract policies. Second, we derive a bound on the optimal performance for a MQNET based on a relaxation that defines indices in the

network. These indices, which for the single-station MQNET case correspond to the optimal indices derived in Klimov (1974), naturally define priority-index policies for the multi-station MQNET case.

Structure of the paper. The rest of the paper is structured as follows: Section 2 introduces the MQNET model and formulates the corresponding performance optimization problem in terms of the achievable region approach. Sections 3-7 develop different families of performance constraints by formulating system equilibrium relations. The constraints presented in section 7 account explicitly for the impact of changeover time parameters. Section 8 presents several positive semidefinite constraints. Section 9 summarizes the bounds and the formulations developed previously and reports computational results. Section 10 proposes two heuristic policies extracted from the formulations.

We have summarized in Appendix A some basic results from the Palm calculus of point processes that are used throughout the paper.

2 The MQNET model

2.1 Model description

We consider a network of queues composed of M single-server stations and populated by N customer classes. The set of customer classes $\mathcal{N} = \{1, \ldots, N\}$ is partitioned into subsets $\mathcal{C}_1, \ldots, \mathcal{C}_M$, so that station $m \in \mathcal{M} = \{1, \ldots, M\}$ only serves classes in its *constituency* $\mathcal{C}_m$. We note that the single class index $i \in \mathcal{N}$ of a customer used here carries the same information as the usual pair of indices (j, m) used in much of the queueing network literature (see, e.g., Kelly (1979)) for identifying jobs present in the network, where an index denotes the job's current type and the other its current location. We further denote by $s(i)$ the station that services class i customers (which we shall refer to as *i-customers*). The network is *open*, so that customers arrive at the network from outside, follow a Markovian route through one or several queues (i-customers wait for service at the *i-queue*) and then leave the system. External i-customers' arrivals follow a Poisson process with rate α_i (if class i does not have external arrivals we let $\alpha_i = 0$). The service times of i-customers are i.i.d., having an exponential distribution with mean $\beta_i = 1/\mu_i$. Upon completion of its service at station $s(i)$, an i-customer may be routed for further service to the j-queue, with probability p_{ij}, or it may leave the system, with probability $p_{i0} = 1 - \sum_{j \in \mathcal{N}} p_{ij}$. We assume that routing matrix $\boldsymbol{P} = (p_{ij})_{i,j \in \mathcal{N}}$ is such that a single customer moving through the network eventually exits it, i.e., matrix $\boldsymbol{I} - \boldsymbol{P}$ is invertible. We further assume that all service times and arrival processes are mutually independent.

The network is controlled by a *scheduling policy*, which specifies dynamically how each server is allocated to waiting customers. Servers incur *changeover times* when moving from one queue to another: if after *visiting* the i-queue the corresponding server moves to the j-queue he incurs a

random changeover time having a general distribution with mean s_{ij} and second moment $s_{ij}^{(2)}$. Usual stochastic independence assumptions hold.

We shall refer to the following classes of scheduling policies: *dynamic* policies, under which scheduling decisions may depend on the current or past states of all queues; *static* policies, under which the scheduling decisions of each server depend only on the state of the queue he is currently visiting; *stable* policies, under which the queue length vector process has an equilibrium distribution with finite mean. We shall allow policies to be *preemptive* (a customer's service may be interrupted and resumed later). However, we require that once a changeover is initiated, it must continue to completion. We shall further refer to the class of *nonidling* policies, under which each server must be at any time either serving a customer or engaged in a changeover.

We define next other model parameters of interest. The *total arrival rate* of j-customers, denoted by λ_j, is the total rate at which both external and internal customers arrive to the j-queue. The λ_j's are computed by solving the 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-customers, denoted by $\rho_j = \lambda_j \beta_j$, is the time-stationary probability that a j-customer is in service. The *total traffic intensity* at station m is $\rho(\mathcal{C}_m) = \sum_{j \in \mathcal{C}_m} \rho_j$, and is the time-stationary probability that server m is busy. The condition

$$\rho(\mathcal{C}_m) < 1, \qquad \text{for } m \in \mathcal{M}$$

is necessary but not sufficient for guaranteeing the stability of any nonidling policy.

We assume that the system operates in a steady-state regime, under a stable policy, and introduce the following variables:

- $L_i(t)$ = number of i-customers in system at time t.
- $B_i(t) = 1$ if an i-customer is in service at time t; 0 otherwise.
- $B^m(t) = 1$ if server m is busy at time t; 0 otherwise; notice that $B^m(t) = \sum_{i \in \mathcal{C}_m} B_i(t)$. $B_{ij}(t) = 1$ if a server is engaged in a $i \to j$ changeover at time t; 0 otherwise.

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

2.2 The performance optimization problem

The main system *performance measure* we are concerned with is the vector whose components are the time-stationary mean number from each class in the system, denoted by $\boldsymbol{x} = (x_j)_{j \in \mathcal{N}}$, where

$$x_j = E\left[L_j\right], \qquad \text{for } j \in \mathcal{N}.$$

Performance variables	Interpretation
x_j; $\boldsymbol{x} = (x_j)_{j\in\mathcal{N}}$	$E\left[L_j\right]$
x_j^i; $\boldsymbol{X} = (x_j^i)_{i,j\in\mathcal{N}}$; $\boldsymbol{x}^i = (x_j^i)_{j\in\mathcal{N}}$	$E\left[L_j \mid B_i = 1\right]$
x_j^{0m}; $\boldsymbol{X}^0 = (x_j^{0m})_{m\in\mathcal{M},j\in\mathcal{N}}$; $\boldsymbol{x}^{0m} = (x_j^{0m})_{j\in\mathcal{N}}$	$E\left[L_j \mid B^m = 0\right]$
r_{ij}; $\boldsymbol{R} = (r_{ij})_{i,j\in\mathcal{N}}$	$E\left[B_i B_j\right]$
r_{ij}^k; $\boldsymbol{R}^k = (r_{ij}^k)_{i,j\in\mathcal{N}}$	$E\left[B_i B_j \mid B_k = 1\right]$
r_{ij}^{0m}; $\boldsymbol{R}^{0m} = (r_{ij}^{0m})_{i,j\in\mathcal{N}}$	$E\left[B_i B_j \mid B^m = 0\right]$
y_{ij}; $\boldsymbol{Y} = (y_{ij})_{i,j\in\mathcal{N}}$	$E\left[L_i L_j\right]$
y_{ij}^k; $\boldsymbol{Y}^k = (y_{ij}^k)_{i,j\in\mathcal{N}}$	$E\left[L_i L_j \mid B_k = 1\right]$
y_{ij}^{0m}; $\boldsymbol{Y}^{0m} = (y_{ij}^{0m})_{i,j\in\mathcal{N}}$	$E\left[L_i L_j \mid B^m = 0\right]$
f_{ij}; $\boldsymbol{F} = (f_{ij})_{i,j\in\mathcal{N}}$	rate of $i \to j$ changeovers
f_i; $\boldsymbol{f} = (f_j)_{j\in\mathcal{N}}$	rate of server visits to the i-queue

Table 1: Network performance measures.

Given a *performance cost function* $c(\boldsymbol{x})$ (possibly nonlinear), we shall investigate the following *performance optimization problem*: compute a lower bound $\underline{Z} \leq c(\boldsymbol{x})$ that is valid under a given class of admissible policies, and design a policy which nearly minimizes the cost $c(\boldsymbol{x})$.

We shall approach this problem via the achievable region approach, as described in the Introduction. Let $\mathcal{X}$ be the performance region achievable 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 the following plan: (1) identify system *equilibrium relations* and formulate them as constraints involving *auxiliary performance variables*; (2) formulate additional *positive semidefinite constraints* on the auxiliary performance variables; (3) formulate constraints that express the original performance vector, $\boldsymbol{x}$, in terms of the auxiliary variables.

Notice that this approach has a clear geometric interpretation: It corresponds to constructing a relaxation of the performance region of the natural variables, x_j, 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 for hard discrete optimization problems (see, e.g., Lovász and Schrijver (1991)).

We have summarized in Table 1 the performance measures considered in this paper.

3 Projection constraints

We present in this section several sets of linear equality constraints that express natural performance measures in terms of auxiliary ones. These constraints correspond geometrically to a *projection*: they allow us to recover the values of natural performance measures from the corresponding values of auxiliary ones.

Theorem 1 (Projection constraints) *Under any dynamic stable policy, the following equations hold:*

(a)

$$x_j = \sum_{i \in \mathcal{C}_m} \rho_i x_j^i + (1 - \rho(\mathcal{C}_m))\, x_j^{0m}, \qquad \textit{for } j \in \mathcal{N},\ m \in \mathcal{M}. \tag{1}$$

(b)

$$r_{ij} = \sum_{k \in \mathcal{C}_m} \rho_k r_{ij}^k + (1 - \rho(\mathcal{C}_m))\, r_{ij}^{0m}, \qquad \textit{for } i, j \in \mathcal{N},\ m \in \mathcal{M}. \tag{2}$$

(c) *If* $E\left[(L_1 + \ldots + L_N)^2\right] < \infty$ *then*

$$y_{ij} = \sum_{k \in \mathcal{C}_m} \rho_k y_{ij}^k + (1 - \rho(\mathcal{C}_m))\, y_{ij}^{0m}, \qquad \textit{for } i, j \in \mathcal{N},\ m \in \mathcal{M}. \tag{3}$$

Proof

The constraints in (a), (b) and (c) are elementary, as they follow by a conditioning argument, by noticing that at each time every server is either serving some customer class in its constituency or idling. ■

4 Lower bound constraints

We present in this section a new set of lower bound constraints on auxiliary performance variables. Our main result follows next.

Theorem 2 (Lower bound constraints) *Under any dynamic stable policy, the following linear constraints hold:*

(a)

$$r_{ij} \geq \max(0, \rho_i + \rho_j - 1), \qquad \textit{for } i, j \in \mathcal{N}. \tag{4}$$

(b)

$$x_j^i \geq \frac{r_{ij}}{\rho_i}, \qquad \textit{for } i, j \in \mathcal{N}, \tag{5}$$

$$x_j^i \geq \frac{\max(0, \rho_i + \rho_j - 1)}{\rho_i}, \qquad \textit{for } i, j \in \mathcal{N}. \tag{6}$$

(c)

$$x_j^{0m} \geq \max\left(0, \frac{\rho_j - \rho(\mathcal{C}_m)}{1 - \rho(\mathcal{C}_m)}\right), \qquad \textit{for } m \in \mathcal{M}, j \in \mathcal{N}. \tag{7}$$

(d)

$$r_{ij}^k \geq \max\left(0, \frac{r_{ki}+r_{kj}}{\rho_k} - 1\right), \qquad \textit{for } i,j,k \in \mathcal{N}. \tag{8}$$

(e)

$$r_{ij}^{0m} \geq \max\left(0, \frac{\max(0,\rho_i - \rho(\mathcal{C}_m)) + \max(0,\rho_j - \rho(\mathcal{C}_m))}{1-\rho(\mathcal{C}_m))} - 1\right), \qquad \textit{for } i,j \in \mathcal{N},\ m \in \mathcal{M}. \tag{9}$$

(f) *If* $E\left[(L_1 + \cdots + L_N)^2\right] < \infty$ *then*

$$y_{ij} \geq r_{ij}, \qquad \textit{for } i,j \in \mathcal{N}, \tag{10}$$

$$y_{ij}^k \geq r_{ij}^k, \qquad \textit{for } i,j,k \in \mathcal{N}, \tag{11}$$

$$y_{ij}^{0m} \geq r_{ij}^{0m}, \qquad \textit{for } i,j \in \mathcal{N},\ m \in \mathcal{M}. \tag{12}$$

Proof

(a) The result follows directly by subtracting equation

$$P\{B_i = 1, B_j = 0\} + P\{B_i = 0, B_j = 0\} = 1 - \rho_j$$

from

$$P\{B_i = 1, B_j = 0\} + P\{B_i = 1, B_j = 1\} = \rho_i.$$

(b) The result follows from

$$\begin{aligned} x_j^i &\geq P\{B_j = 1 \mid B_i = 1\} \\ &= \frac{r_{ij}}{\rho_i}. \end{aligned} \tag{13}$$

(c) We have

$$\begin{aligned} x_j^{0m} &\geq P\{B_j = 1 \mid B^m = 0\} \\ &= \frac{P\{B_j = 1, B^m = 0\}}{1-\rho(\mathcal{C}_m)}. \end{aligned} \tag{14}$$

Now, by subtracting

$$P\{B_j = 1, B^m = 1\} + P\{B_j = 0, B^m = 1\} = \rho(\mathcal{C}_m)$$

from

$$P\{B_j = 1, B^m = 1\} + P\{B_j = 1, B^m = 0\} = \rho_j$$

we obtain

$$P\{B_j = 1, B^m = 0\} \geq \rho_j - \rho(\mathcal{C}_m), \tag{15}$$

which, combined with (14) yields the result.

(d) The result follows directly by subtracting

$$P\{B_i = 0, B_j = 1 \mid B_k = 1\} + P\{B_i = 0, B_j = 0 \mid B_k = 1\} = P\{B_i = 0 \mid B_k = 1\} = 1 - \frac{r_{ki}}{\rho_k}$$

Point process	Epochs	Intensity	Stochastic intensity
A_i^0	external i-customer arrivals	α_i	$\lambda^{A_i^0}(t) = \alpha_i$
D_i^0	external i-customer departures	$\lambda_i p_{i0}$	$\lambda^{D_i^0}(t) = \mu_i p_{i0} B_i(t)$
T_{ij}	$i \to j$ customer routing	$\lambda_i p_{ij}$	$\lambda^{T_{ij}}(t) = \mu_i p_{ij} B_i(t)$

Table 2: Elementary network point processes and their intensities.

from

$$P\{B_i = 0, B_j = 1 \mid B_k = 1\} + P\{B_i = 1, B_j = 1 \mid B_k = 1\} = P\{B_j = 1 \mid B_k = 1\} = \frac{r_{kj}}{\rho_k}.$$

(e) The result follows by subtracting

$$P\{B_i = 0, B_j = 1 \mid B^m = 0\} + P\{B_i = 0, B_j = 0 \mid B^m = 0\} = P\{B_i = 0 \mid B^m = 0\}$$

from

$$P\{B_i = 0, B_j = 1 \mid B^m = 0\} + P\{B_i = 1, B_j = 1 \mid B^m = 0\} = P\{B_j = 1 \mid B^m = 0\},$$

and then applying inequality (15).

(f) The inequalities in (f) are elementary, as they follow from the relation $L_i \geq B_i$. ■

5 Flow conservation constraints

We present in this section a set of linear constraints on performance measures by formulating the classical *flow conservation law* of queueing theory $L^- = L^+$. This law states that, in a queueing system in which the queue size can increase or decrease only by unit steps, the stationary state probabilities of the number in system at arrival epochs and that at departure epochs are equal. These constraints were first derived for multi-station MQNETs by Bertsimas, Paschalidis and Tsitsiklis (1994), and by Kumar and Kumar (1994), through a potential function approach. The corresponding constraints for single-station MQNETs were obtained by Klimov (1974) via transform methods.

Our contribution in this section is twofold: (1) we reveal that the physical origin of the constraints produced by the potential function approach is the flow conservation law $L^- = L^+$; (2) we derive new closed formulae for all higher-order constraints (with the potential function approach these are generated recursively).

In particular, we shall apply the law $L^- = L^+$ to a family of queues obtained by aggregating customer classes, as explained next. Let $S \subseteq \mathcal{N}$.

Definition 1 (S-queue) *The S-queue is the queueing system obtained by aggregating customer classes in S. The number in system at time t in the S-queue is denoted by $L_S(t) = \sum_{j \in S} L_j(t)$.*

As usual we write $L_S = L_S(0)$, $L_S^- = L_S(0-)$, $L_S^+ = L_S(0+) = L_S(0)$.

We denote by A_S the point process of *net arrival epochs* to the S-queue, which consists of S-customer external arrival epochs and customer routing epochs from a class in S^c to a class in S. We can thus express point process A_S as the *superposition* (see Appendix A) of the elementary network point processes shown in Table 2, as follows:

$$A_S = \sum_{j \in S} A_j^0 + \sum_{i \in S^c} \sum_{j \in S} R_{ij},$$

Similarly we denote by D_S the point process of *net departure epochs* from the S-queue, consisting of S-customer external departure epochs and customer routing epochs from a class in S to a class in S^c,

$$D_S = \sum_{j \in S} D_j^0 + \sum_{j \in S} \sum_{i \in S^c} R_{ji}.$$

Notice that we ignore customer routing epochs within classes in S, since they do not change the number of customers in the S-queue.

For convenience of notation we shall also write

$$p(i, S) = \sum_{j \in S} p_{ij}$$

and

$$\alpha(S) = \sum_{j \in S} \alpha_j.$$

We denote the Palm probabilities and expectations with respect to point processes A_S and D_S by $P^{A_S}(\cdot)$, $E^{A_S}[\cdot]$ and $P^{D_S}(\cdot)$, $E^{D_S}[\cdot]$, respectively. The time-stationary distributions and expectations are denoted by $P(\cdot)$ and $E[\cdot]$, respectively.

We state and prove next our main result, which formulates the law $L^- = L^+$ as it applies to the S-queue: The stationary state probabilities of the number of customers in the S-queue just before a net customer arrival epoch and just after a net customer departure epoch to/from the S-queue are equal. The theorem formulates this identity between Palm distributions as a linear relation between time-stationary distributions, thus bridging the gap between them.

Theorem 3 (The law $L^- = L^+$ in MQNETs) *Under any dynamic stable policy, and for any subset of customer classes $S \subseteq \mathcal{N}$ and nonnegative integer l:*

(a)

$$P^{A_S}\left\{L_S^- = l\right\} = P^{D_S}\left\{L_S^+ = l\right\}. \tag{16}$$

(b) *Identity (16) is equivalently formulated as*

$$\alpha(S) P\left\{L_S = l\right\} + \sum_{i \in S^c} \lambda_i p(i, S) P\left\{L_S = l \mid B_i = 1\right\} = \sum_{i \in S} \lambda_i \left(1 - p(i, S)\right) P\left\{L_S = l + 1 \mid B_i = 1\right\}. \tag{17}$$

Proof

Part (a) follows directly by applying the flow conservation law $L^- = L^+$ to the number in system process $\{L_S(t)\}$ corresponding to the S-queue.

(b) The key tool we shall apply for expressing the Palm distributions in part (a) in terms of time-stationary distributions is Papangelou's theorem (Theorem 11 in Appendix A). First, we notice that arrival point process A_S admits a stochastic intensity (see Appendix A)

$$\lambda^S(t) = \alpha(S) + \sum_{i \in S^c} \sum_{j \in S} \mu_i p_{ij} B_i(t), \tag{18}$$

whereas the stochastic intensity of departure point process D_S is

$$\mu^S(t) = \sum_{i \in S} \mu_i \left(1 - p(i,S)\right) B_i(t). \tag{19}$$

Let $\lambda^S = E\left[\lambda^S(0)\right]$ and $\mu^S = E\left[\mu^S(0)\right]$. Notice that, by flow conservation, $\lambda^S = \mu^S$.

Now, by Papangelou's theorem, Eq. (18) and the relation $P\{B_i = 1\} = \rho_i$ we have

$$\begin{aligned} \lambda^S P^{A_S}\left\{L_S^- = l\right\} &= \lambda^S E^{A_S}\left[1\left\{L_S(0-) = l\right\}\right] \\ &= E\left[\lambda^S(0) 1\left\{L_S(0) = l\right\}\right] \\ &= \alpha(S) P\left\{L_S = l\right\} + \sum_{i \in S^c} \sum_{j \in S} \lambda_i p_{ij} P\left\{L_S = L \mid B_i = 1\right\}, \end{aligned} \tag{20}$$

and, similarly,

$$\begin{aligned} \mu^S P^{D_S}\left\{L_S^+ = l\right\} &= \mu^S P^{D_S}\left\{L_S^- = l+1\right\} \\ &= E\left[\mu^S(0) 1\left\{L_S(0) = l+1\right\}\right] \\ &= \sum_{i \in S} \lambda_i \left(1 - p(i,S)\right) P\left\{L_S = l+1 \mid B_i = 1\right\}. \end{aligned} \tag{21}$$

Now, equating (20) and (21) (by part (a)), and using the fact that $\lambda^S = \mu^S$ the result follows. ■

Taking expectations in identity (17) we obtain our next result, which formulates a linear relation between time-stationary moments of queue lengths.

Corollary 1 *Under any dynamic stable policy, and for any subset of customer classes $S \subseteq \mathcal{N}$ and positive integer K for which $E\left[(L_1 + \cdots + L_N)^K\right] < \infty$,*

$$\alpha(S) E\left[L_S^K\right] + \sum_{i \in S^c} \lambda_i p(i,S) E\left[L_S^K \mid B_i = 1\right] = \sum_{i \in S} \lambda_i \left(1 - p(i,S)\right) E\left[(L_S - 1)^K \mid B_i = 1\right]. \tag{22}$$

The equilibrium equations in Corollary 1 corresponding to $K = 1, 2$ and $S = \{i\}, \{i,j\}$, for $i, j \in \mathcal{N}$, yield directly the system of linear constraints on performance variables shown next. Let $\mathbf{\Lambda} = \mathrm{Diag}(\boldsymbol{\lambda})$.

Corollary 2 (Flow conservation constraints) *Under any dynamic stable policy, the following linear constraints hold:*

(a)

$$-\boldsymbol{\alpha x}' - \boldsymbol{x\alpha}' + (\boldsymbol{I}-\boldsymbol{P})'\boldsymbol{\Lambda X} + \boldsymbol{X}'\boldsymbol{\Lambda}(\boldsymbol{I}-\boldsymbol{P}) = (\boldsymbol{I}-\boldsymbol{P})'\boldsymbol{\Lambda} + \boldsymbol{\Lambda}(\boldsymbol{I}-\boldsymbol{P}). \tag{23}$$

(b) *If* $E\left[(L_1+\cdots L_N)^2\right] < \infty$, *then*

$$\alpha_j y_{jj} + \sum_{r\in\mathcal{N}} \lambda_r p_{rj} y^r_{jj} - \lambda_j y^j_{jj} + 2\lambda_j(1-p_{jj})x^j_j = \lambda_j(1-p_{jj}), \quad j\in\mathcal{N}, \tag{24}$$

$$\begin{aligned}\alpha_i y_{jj} + \alpha_j y_{ii} + 2(\alpha_i+\alpha_j)y_{ij} + \sum_{r\in\mathcal{N}} \lambda_r p_{ri} y^r_{jj} + \sum_{r\in\mathcal{N}} \lambda_r p_{rj} y^r_{ii} \\ + \sum_{r\in\mathcal{N}} 2\lambda_r(p_{ri}+p_{rj})y^r_{ij} - \lambda_i y^i_{jj} - \lambda_j y^j_{ii} - 2\lambda_i y^i_{ij} - 2\lambda_j y^j_{ij} \\ +2\lambda_i(1-p_{ii}-p_{ij})x^i_j + 2\lambda_j(1-p_{ji}-p_{jj})x^j_i = -\lambda_i p_{ij} - \lambda_j p_{ji}, \quad i,j\in\mathcal{N}.\end{aligned} \tag{25}$$

Remarks.

1. Eqns. (23) in Corollary 2 were first derived by Bertsimas, Paschalidis and Tsitsiklis (1994), and by Kumar and Kumar (1994) through a potential function method. In both papers the authors assumed the stronger condition that the second moment of the total number of customers in the network is finite, i.e., $E\left[(L_1+\cdots+L_N)^2\right] < \infty$. We only require, as in Kumar and Meyn (1996), finiteness of the corresponding first moment.
2. Bertsimas, Paschalidis and Tsitsiklis (1994) proposed a recursive algebraic procedure for generating higher-order constraints corresponding to Eqns. (22) in Corollary 1 (with $K \geq 2$). In contrast to their approach, we present in Corollary 1 closed formulae that reveal the simple structure of this family of equations.
3. Interestingly, for $K=1$, it can been seen that all the equations in (22) for $|S| \geq 3$ are implied by those with $|S| \leq 2$.

6 Workload decomposition constraints

In this section we derive a new family of linear 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 system from each class at an arbitrary time and at an arbitrary time during a period when some servers are idle. Our contributions include: (1) a family of new *work decomposition laws* for multi-station MQNETs, which extends the most general results known previously: Boxma's (1989) work decomposition law for multiclass $M/G/1$ queues, and Bertsimas and Niño-Mora's (1998) work decomposition laws for single-server MQNETs; (2) tighter network workload bounds, which improve upon the bounds derived by Bertsimas, Paschalidis and Tsitsiklis (1994); (3) new families of convex constraints for MQNETs with changeover times, obtained from the new work decomposition laws.

The idea of deriving performance constraints from work decomposition laws was introduced by Bertsimas and Xu (1993) in the setting of a multiclass $M/G/1$ queue with changeover times. They derived a set of convex constraints by applying a work decomposition law due to Fuhrmann and Cooper (1985). Bertsimas and Niño-Mora (1998) have extended the idea to single-server MQNETs with changeover times, presenting a family of new work decomposition laws, and applying them to formulate new convex performance constraints.

6.1 Work decomposition laws

In order to develop the new work decomposition laws we first present the following definition. Let $S \subseteq \mathcal{N}$ be a subset of customer classes.

Definition 2 (S-workload) *The workload process corresponding to the S-queue (see Definition 1) is called the S-*workload process*, denoted by $\{V^S(t)\}_{t\in\Re}$. $V^S(t)$ is thus the total remaining service time needed for first clearing the S-queue of all S-customers present at time t.*

We shall denote by $B_S^m(t)$ the indicator of the event that server m is busy with an S-customer at time t, i.e., $B_S^m(t) = \sum_{i\in S\cap\mathcal{C}_m} B_i(t)$. As before, we write $V^S = V^S(0), B_S^m = B_S^m(0)$.

We next define parameters V_i^S, for $i \in \mathcal{N}$, as the solution of the system of linear equations

$$V_i^S = \beta_i + \sum_{j\in S} p_{ij} V_j^S, \qquad \text{for } i \in \mathcal{N}. \tag{26}$$

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.

In what follows we shall use the following matrix 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

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

For example, we write Eqns. (26) in matrix form as

$$\boldsymbol{V}_S^S = \boldsymbol{\beta}_S + \boldsymbol{P}_{SS} V_S^S,$$

$$\boldsymbol{V}_{S^c}^S = \boldsymbol{\beta}_{S^c} + \boldsymbol{P}_{S^cS} \boldsymbol{V}_S^S,$$

where $\boldsymbol{\beta} = (\beta_i)_{i\in\mathcal{N}}$.

Furthermore, we shall denote by $\rho^0(S)$ the rate at which *external* S-work enters the system, i.e.,

$$\rho^0(S) = \sum_{j\in S} \alpha_j V_j^S,$$

and write

$$\rho(S) = \sum_{j \in S} \rho_j.$$

We state and prove next the new work decomposition laws, which formulate a decomposition of the mean workload in the S-queue, for every $S \subseteq \mathcal{N}$. Let $\mathcal{M}(S)$ denote the set of stations that service S-customers, and let $M(S) = |\mathcal{M}(S)|$ be its corresponding cardinality.

Theorem 4 (Work decomposition laws) *Under any dynamic stable policy, and for any subset $S \subseteq \mathcal{N}$ of customer classes:*

(a)

$$\begin{aligned}
\big(M(S) - \rho^0(S)\big) \sum_{j \in S} V_j^S x_j &= \sum_{j \in S} \rho_j V_j^S + \sum_{i \in S^c \cap \left(\cup_{m \in \mathcal{M}(S)} \mathcal{C}_m\right)} \sum_{j \in S} \rho_i V_j^S x_j^i \\
&\quad + \sum_{i \in S^c} \sum_{j \in S} \left(\lambda_i V_i^S - \rho_i\right) V_j^S x_j^i \\
&\quad + \sum_{m \in \mathcal{M}(S)} \sum_{j \in S} \left(1 - \rho(\mathcal{C}_m)\right) V_j^S x_j^{0m}. \qquad (27)
\end{aligned}$$

(b) *Identity (27) is equivalently formulated as*

$$\begin{aligned}
\big(M(S) - \rho^0(S)\big) E\left[V^S\right] &= \sum_{j \in S} \rho_j V_j^S + \sum_{i \in S^c} \left(\lambda_i V_i^S - \rho_i\right) E\left[V^S \mid B_i = 1\right] \\
&\quad + \sum_{m \in \mathcal{M}(S)} \left(1 - \rho(S \cap \mathcal{C}_m)\right) E\left[V^S \mid B_S^m = 0\right]. \qquad (28)
\end{aligned}$$

Proof

Let us define N-vector $\boldsymbol{v}$ by

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

and set function $b(S)$ by

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

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

$$\boldsymbol{B} = (\boldsymbol{I} - \boldsymbol{P})' \boldsymbol{\Lambda} + \boldsymbol{\Lambda}(\boldsymbol{I} - \boldsymbol{P}).$$

We then have, by the flow conservation equations (23) in Corollary 2, that

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

$$
\begin{aligned}
&= -\rho^0(S)\sum_{j\in S} V_j^S x_j + \sum_{i\in S}\sum_{j\in S}\rho_i V_j^S x_j^i - \sum_{i\in S^c}\sum_{j\in S}\left(\lambda_i V_i^S - \rho_i\right) V_j^S x_j^i \\
&= -\rho^0(S)\sum_{j\in S} V_j^S x_j - \sum_{i\in S^c}\sum_{j\in S}\lambda_i V_i^S V_j^S x_j^i + \sum_{i\in \mathcal{N}}\sum_{j\in S}\rho_i V_j^S x_j^i. \qquad (29)
\end{aligned}
$$

Now, from Eqns. (1) in Theorem 1 it follows that

$$
x_j = \sum_{i\in S\cap\mathcal{C}_m}\rho_i x_j^i + \sum_{i\in S^c\cap\mathcal{C}_m}\rho_i x_j^i + (1-\rho(\mathcal{C}_m))x_{mj}^0, \qquad \text{for } m\in\mathcal{M}. \qquad (30)
$$

Adding over $m\in\mathcal{M}(S)$ in (30) we obtain

$$
M(S)x_j = \sum_{i\in S}\rho_i x_j^i + \sum_{i\in S^c\cap\left(\cup_{m\in\mathcal{M}(S)}\mathcal{C}_m\right)}\rho_i x_j^i + \sum_{m\in\mathcal{M}(S)}\left(1-\rho(\mathcal{C}_m)\right)x_{mj}^0. \qquad (31)
$$

Now, simplifying (29) using (31) yields

$$
\begin{aligned}
b(S) &= \left(M(S)-\rho^0(S)\right)\sum_{j\in S} V_j^S x_j - \sum_{i\in S^c\cap\left(\cup_{m\in\mathcal{M}(S)}\mathcal{C}_m\right)}\sum_{j\in S} V_j^S \rho_i x_j^i \\
&\quad - \sum_{i\in S^c}\sum_{j\in S}\left(\lambda_i V_i^S - \rho_i\right) V_j^S x_j^i - \sum_{m\in\mathcal{M}(S)}\sum_{j\in S}\left(1-\rho(\mathcal{C}_m)\right) V_j^S x_j^{0m} \qquad (32)
\end{aligned}
$$

On the other hand, we have

$$
\begin{aligned}
b(S) &= \frac{1}{2}\boldsymbol{V}_S^{S\prime}\boldsymbol{B}_{SS}\boldsymbol{V}_S^S \\
&= \frac{1}{2}\begin{pmatrix} \boldsymbol{V}_S^{S\prime} & \boldsymbol{0}\end{pmatrix}\left\{(\boldsymbol{I}-\boldsymbol{P})'\boldsymbol{\Lambda}+\boldsymbol{\Lambda}(\boldsymbol{I}-\boldsymbol{P})\right\}\begin{pmatrix}\boldsymbol{V}_S^S \\ \boldsymbol{0}\end{pmatrix} \\
&= \begin{pmatrix} \boldsymbol{V}_S^{S\prime} & \boldsymbol{0}\end{pmatrix}(\boldsymbol{I}-\boldsymbol{P})'\boldsymbol{\Lambda}\begin{pmatrix}\boldsymbol{V}_S^S \\ \boldsymbol{0}\end{pmatrix} \\
&= \left\{\begin{pmatrix} \boldsymbol{I}_S-\boldsymbol{P}_{SS} & -\boldsymbol{P}_{SS^c} \\ -\boldsymbol{P}_{S^cS} & \boldsymbol{I}_{S^c}-\boldsymbol{P}_{S^cS^c}\end{pmatrix}\begin{pmatrix}\boldsymbol{V}_S^S \\ \boldsymbol{0}\end{pmatrix}\right\}'\boldsymbol{\Lambda}\begin{pmatrix}\boldsymbol{V}_S^S \\ \boldsymbol{0}\end{pmatrix} \\
&= \begin{pmatrix} \boldsymbol{\beta}_S' & \boldsymbol{\beta}_{S^c}' - \boldsymbol{V}_{S^c}^{S\,\prime}\end{pmatrix}\boldsymbol{\Lambda}\begin{pmatrix}\boldsymbol{V}_S^S \\ \boldsymbol{0}\end{pmatrix} \\
&= \sum_{j\in S}\rho_j V_j^S. \qquad (33)
\end{aligned}
$$

Finally, substituting (33) into (32) yields directly identity (27).

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

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

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

and

$$
E\left[V^S \mid B_i = 1\right] = \sum_{j\in S} V_j^S x_j^i,
$$

which, combined with Eq. (27) yields

$$\begin{aligned}\left(M(S)-\rho^0(S)\right)E\left[V^S\right] &= \sum_{j\in S}\rho_j V_j^S + \sum_{i\in S^c\cap\left(\cup_{m\in\mathcal{M}(S)}\mathcal{C}_m\right)}\rho_i E\left[V^S \mid B_i=1\right] \\ &+ \sum_{i\in S^c}\left(\lambda_i V_i^S-\rho_i\right)E\left[V^S \mid B_i=1\right] \\ &+ \sum_{m\in\mathcal{M}(S)}\left(1-\rho(\mathcal{C}_m)\right)E\left[V^S \mid B^m=0\right]. \end{aligned} \tag{34}$$

Identity (28) now follows by simplifying Eq. (34) using the elementary relations

$$E\left[V^S \mid B_S^m=0\right] = \frac{\rho\left(S^c\cap\mathcal{C}_m\right)}{1-\rho\left(S\cap\mathcal{C}_m\right)}E\left[V^S \mid B_{S^c}^m=1\right] + \frac{1-\rho\left(\mathcal{C}_m\right)}{1-\rho\left(S\cap\mathcal{C}_m\right)}E\left[V^S \mid B^m=0\right] \tag{35}$$

and

$$\rho\left(S^c\cap\mathcal{C}_m\right)E\left[V^S \mid B_{S^c}^m=1\right] = \sum_{i\in S^c\cap\mathcal{C}_m}\rho_i E\left[V^S \mid B_i=1\right]. \tag{36}$$

■

Remark. Identity (28) in Theorem 4(b) may be interpreted physically in terms of work decomposition, as it says that the mean network S-workload decomposes into three components: 1) a constant term, independent of the policy, 2) a linear combination of the conditional mean S-workloads during the service of S^c-customers, and 3) a linear combination of the conditional mean S-workloads during idle periods of servers who service S-customers. In particular, for $S=\mathcal{N}$ Eq. (28) yields

$$E\left[V^{\mathcal{N}}\right] = \frac{\sum_{j\in\mathcal{N}}\rho_j V_j^{\mathcal{N}}}{M-\rho(\mathcal{N})} + \sum_{m\in\mathcal{M}}\frac{1-\rho(\mathcal{C}_m)}{M-\rho(\mathcal{N})}E\left[V^{\mathcal{N}} \mid B^m=0\right], \tag{37}$$

which means that the total mean network workload decomposes into a constant term plus a linear convex combination of the conditional mean network workloads during servers idle times. Therefore, identity (28) extends the work decomposition laws developed by Boxma (1989) and by Bertsimas and Niño-Mora (1998) for single-station systems to multi-station MQNETs.

As an application of the work decomposition laws in Theorem 4 we present next a family of workload bounds for MQNETs, which improve upon the workload bounds developed in Bertsimas, Paschalidis and Tsitsiklis (1994). Let us define a set function $g(S)$ on subsets S of customer classes by

$$\begin{aligned} g(S) &= \frac{\sum_{j\in S}\rho_j V_j^S}{M(S)-\rho^0(S)} + \frac{\sum_{i\in S^c\cap\left(\cup_{m\in\mathcal{M}(S)}\mathcal{C}_m\right)}\sum_{j\in S}V_j^S\max\left(0,\rho_i+\rho_j-1\right)}{M(S)-\rho^0(S)} \\ &+ \frac{\sum_{i\in S^c}\sum_{j\in S}\left(\lambda_i V_i^S-\rho_i\right)V_j^S\max\left(0,\frac{\rho_i+\rho_j-1}{\rho_i}\right)}{M(S)-\rho^0(S)} \\ &+ \frac{\sum_{m\in\mathcal{M}(S)}\sum_{j\in S}V_j^S\max\left(0,\rho_j-\rho(\mathcal{C}_m)\right)}{M(S)-\rho^0(S)}. \end{aligned} \tag{38}$$

Corollary 3 (Workload bounds) *Under any dynamic stable policy, the following workload bounds hold:*

$$\sum_{j\in S} V_j^S x_j \geq g(S), \qquad \textit{for } S \subseteq \mathcal{N}. \tag{39}$$

Proof

Inequality (39) follows directly by combining work decomposition Eq. (27) in Theorem 4(a) and the lower bounds in Theorem 2(b)-(c). ■

Remarks

1. The workload bounds in Corollary 3 improve upon the ones developed by Bertsimas, Paschalidis and Tsitsiklis (1994): they showed that under any dynamic and stable scheduling policy,

$$\sum_{j\in S} V_j^S x_j \geq \frac{\sum_{j\in S} \rho_j V_j^S}{M(S) - \rho^0(S)}, \qquad \text{for } S \subseteq \mathcal{N}. \tag{40}$$

2. In the special case of single-server MQNETs, it follows from identity (27) that the workload bound in (40) is achieved under any dynamic nonidling policy that gives preemptive service priority to S-customers over S^c-customers. This shows that performance measure $\boldsymbol{x}$ satisfies the work conservation laws in Bertsimas and Niño-Mora (1996a), and it follows from their work that the family of inequality constraints in (40), for $S \subset \mathcal{N}$, together with the equation $\sum_{j\in\mathcal{N}} V_j^{\mathcal{N}} x_j = \sum_{j\in\mathcal{N}} \rho_j V_j^{\mathcal{N}}/(1-\rho(\mathcal{N}))$, formulate exactly the performance region of the x_j's.

7 Convex constraints for MQNETs with changeover times

We present in this section constraints on achievable performance that account for the effect of servers changeover times. We first establish some elementary linear constraints on visit and changeover frequencies (f_j, f_{ij}; see Table 1).

Proposition 1 *Under any dynamic stable policy,*

(a)

$$f_i = \sum_{j\in\mathcal{C}_{s(i)}\setminus\{i\}} f_{ij} = \sum_{j\in\mathcal{C}_{s(i)}\setminus\{i\}} f_{ji}, \qquad \textit{for } i \in \mathcal{N}. \tag{41}$$

(b) *If the policy is nonidling, then*

$$\sum_{i,j\in\mathcal{C}_m : i\neq j} s_{ij} f_{ij} = 1 - \rho(\mathcal{C}_m), \qquad \textit{for } m \in \mathcal{M}. \tag{42}$$

Proof

(a) Eq. (41) formulates a simple flow conservation relation: the rates at which server $s(i)$ visits and leaves the i-queue are equal.

(b) Eq. (42) formulates the elementary identity

$$\sum_{i,j\in\mathcal{C}_m} P\{B_{ij}=1\} = 1-\rho(\mathcal{C}_m),$$

which holds under the nonidling assumption. Notice that we have used the identity $P\{B_{ij}=1\} = s_{ij}f_{ij}$. ■

In order to develop the new convex constraints we introduce the following concept from the vacation queues literature:

Definition 3 (Vacation) *We say that server $m \in \mathcal{M}$ is taking a* vacation *away from a set of customer classes $S \subseteq \mathcal{C}_m$ when he is not serving S-customers.*

Consider now the point process $N_{m,S}$ of epochs at which server m initiates a *vacation* away from $S \cap \mathcal{C}_m$-customers (which we refer to henceforth as a *server m S-vacation*), for $S \subseteq \mathcal{N}$. We also let $I_{m,S}$ be a random variable with the equilibrium distribution of a server m S-vacation interval, and define $B_{m,S}(t)$ as the indicator that server m is busy at time t with an S-customer, i.e., $B_{m,S}(t) = \sum_{j\in S\cap\mathcal{C}_m} B_j(t)$.

In the next result we establish lower bounds for the mean number of j-customers in system during changeover periods and during server vacations, respectively, and develop an expression for mean server vacation times, in terms of visit and changeover frequencies. We define set function $h(S)$ by

$$h(S) = \frac{1}{2}\left\{\alpha_j\left(1-\rho(S\cap\mathcal{C}_m)\right) + \sum_{r\in\mathcal{N}\setminus S}\mu_r p_{rj}\max\left(0,\rho_r-\rho(S\cap\mathcal{C}_m)\right)\right\}, \qquad \text{for } S\subseteq\mathcal{N}. \tag{43}$$

Proposition 2 *Under any policy that is static, nonidling and stable, we have:*

(a) *For $m \in \mathcal{M}$ and $j,k,l \in \mathcal{C}_m$, with $k \neq l$,*

$$E\left[L_j \mid B_{kl}=1\right] \geq \alpha_j\frac{s_{kl}^{(2)}}{2s_{kl}} + \sum_{r\in\mathcal{N}\setminus\mathcal{C}_m}\frac{\mu_r p_{rj}s_{kl}^{(2)}}{2s_{kl}^2}\frac{\max(0,\rho_r+s_{kl}f_{kl}-1)}{f_{kl}}. \tag{44}$$

(b) *For $S \subseteq \mathcal{N}, m \in \mathcal{M}(S)$,*

$$E\left[I_{m,S}\right] = \frac{1-\rho(S\cap\mathcal{C}_m)}{\sum_{j\in S\cap\mathcal{C}_m} f_j}. \tag{45}$$

(c) *For $S \subseteq \mathcal{N}, m \in \mathcal{M}(S), j \in S\cap\mathcal{C}_m$,*

$$E\left[L_j \mid B_{m,S}=0\right] \geq h(S)\frac{1}{\sum_{j\in S\cap\mathcal{C}_m} f_j}. \tag{46}$$

Proof

(a) Consider the point process H_{kl} of $k \to l$ server changeover initiation epochs. We introduce random variable v_{kl}^*, the elapsed time of a typical $k \to l$ changeover period that started at time 0, as seen by a *random observer*. Notice that, by random incidence, $E\left[v_{kl}^*\right] = s_{kl}^{(2)}/2s_{kl}$. Let

us denote by z_j^{kl} the mean number of j-customers arriving during time interval $[0, v_{kl}^*)$. Since $E[L_j \mid B_{kl} = 1] \geq z_j^{kl}$, our next goal is to find a lower bound on z_j^{kl}.

Notice first that, during a $k \to l$ changeover period, the point process of j-customer arrivals has a stochastic intensity at time t given by

$$\alpha_j + \sum_{r \in \mathcal{N} \setminus \mathcal{C}_m} \mu_r p_{rj} B_r(t).$$

By definition of stochastic intensity (see Appendix A), we have, under *static* policies,

$$\begin{aligned} z_j^{kl} &= E^{H_{kl}}\left[\int_0^{v_{kl}^*} \alpha_j \, dt\right] + \sum_{r \in \mathcal{N} \setminus \mathcal{C}_m} \mu_r p_{rj} E^{H_{kl}}\left[\int_0^{v_{kl}^*} B_r(t) \, dt\right] \\ &= \alpha_j \frac{s_{kl}^{(2)}}{2 s_{kl}} + \sum_{r \in \mathcal{N} \setminus \mathcal{C}_m} \mu_r p_{rj} P\{B_r = 1, B_{kl} = 1\} \frac{s_{kl}^{(2)}}{2 s_{kl}^2 f_{kl}}, \end{aligned} \tag{47}$$

since under such policies

$$\begin{aligned} E^{H_{kl}}\left[\int_0^{v_{kl}^*} B_r(t) \, dt\right] &= P\{B_r = 1 \mid B_{kl} = 1\} \frac{s_{kl}^{(2)}}{2 s_{kl}} \\ &= P\{B_r = 1, B_{kl} = 1\} \frac{s_{kl}^{(2)}}{2 s_{kl}^2 f_{kl}}. \end{aligned}$$

Now, from

$$P\{B_r = 1, B_{kl} = 0\} + P\{B_r = 1, B_{kl} = 1\} = \rho_r$$

and

$$P\{B_r = 1, B_{kl} = 0\} + P\{B_r = 0, B_{kl} = 0\} = 1 - s_{kl} f_{kl}$$

it follows that

$$P\{B_r = 1, B_{kl} = 1\} \geq \max(0, \rho_r + s_{kl} f_{kl} - 1).$$

Combining this inequality with Eq. (47), and with the relation $E[L_j \mid B_{kl} = 1] \geq z_j^{kl}$ yields the result.

(b) The intensity of point process $N_{m,S}$ is easily seen to be $\sum_{j \in S \cap \mathcal{C}_m} f_j$. Now, under a nonidling policy, the duration of an S-vacation for server m coincides with the total time that server is not serving S-customers between two consecutive points of point process $N_{m,S}$. Therefore, under nonidling static policies,

$$E[I_{m,S}] = \frac{1 - \rho(S \cap \mathcal{C}_m)}{\sum_{j \in S \cap \mathcal{C}_m} f_j},$$

which proves the result.

(c) Consider the point process $N_{m,S}$ of server m S-vacation initiation epochs. We introduce the random variable ${I_{m,S}}^*$, the elapsed time of a typical server m S-vacation period that started at time 0, as seen by a random observer. Notice that, by random incidence, $E[{I_{m,S}}^*] = E\left[{I_{m,S}}^2\right] / 2E[I_{m,S}]$. Let us denote by z_j the mean number of j-customers that arrive during time interval $[0, {I_{m,S}}^*)$. Since, clearly, $E[L_j \mid B_{m,S} = 0] \geq z_j$, our next goal is to find a lower bound on z_j.

We first observe that during a server m S-vacation the point process of j-customer arrivals has a stochastic intensity at time t given by

$$\alpha_j + \sum_{r \in \mathcal{N} \setminus S} \mu_r p_{rj} B_r(t).$$

By definition of stochastic intensity,

$$\begin{aligned} z_j &= E^{N_{m,S}} \left[\int_0^{I_{m,S}^*} \alpha_j \, dt \right] + \sum_{r \in \mathcal{N} \setminus S} \mu_r p_{rj} E^{N_{m,S}} \left[\int_0^{I_{m,S}^*} B_r(t) \, dt \right] \\ &= \alpha_j E\left[I_{m,S}^*\right] + \sum_{r \in \mathcal{N} \setminus S} \mu_r p_{rj} P\left\{B_r = 1, B_{m,S} = 0\right\} \frac{E\left[I_{m,S}^*\right]}{1 - \rho(S \cap \mathcal{C}_m)}, \end{aligned} \tag{48}$$

since

$$\begin{aligned} E^{N_{m,S}} \left[\int_0^{I_{m,S}^*} B_r(t) \, dt \right] &= P\left\{B_r = 1 \mid B_{m,S} = 0\right\} E\left[I_{m,S}^*\right] \\ &= P\left\{B_r = 1, B_{m,S} = 0\right\} \frac{E\left[I_{m,S}^*\right]}{1 - \rho(S \cap \mathcal{C}_m)}. \end{aligned}$$

Now, from

$$P\left\{B_r = 1, B_{m,S} = 1\right\} + P\left\{B_r = 1, B_{m,S} = 0\right\} = \rho_r$$

and

$$P\left\{B_r = 1, B_{m,S} = 1\right\} + P\left\{B_r = 0, B_{m,S} = 1\right\} = \rho(S \cap \mathcal{C}_m)$$

it follows that

$$P\left\{B_r = 1, B_{m,S} = 0\right\} \geq \max(0, \rho_r - \rho(S \cap \mathcal{C}_m)).$$

Combining this inequality with Eqns. (48) and (45), and using the fact that

$$E\left[I_{m,S}^*\right] = \frac{E\left[I_{m,S}^2\right]}{2E\left[I_{m,S}\right]} \geq \frac{1}{2} E\left[I_{m,S}\right]$$

yields the result. ■

The next result presents two families of convex constraints on performance variables.

Theorem 5 *Under any policy that is static, nonidling and stable, the following convex constraints hold:*

(a) *For $m \in \mathcal{M}$ and $j \in \mathcal{C}_m$,*

$$\begin{aligned} x_j^{0m} \geq & \sum_{k,l \in \mathcal{C}_m : k \neq l} \frac{\alpha_j s_{kl}^{(2)}}{2\left(1 - \rho(\mathcal{C}_m)\right)} f_{kl} + \\ & \sum_{k,l \in \mathcal{C}_m : k \neq l} \sum_{r \in \mathcal{N} \setminus \mathcal{C}_m} \frac{\mu_r p_{rj} s_{kl}^{(2)}}{2 s_{kl} \left(1 - \rho(\mathcal{C}_m)\right)} \max(0, \rho_r + s_{kl} f_{kl} - 1). \end{aligned} \tag{49}$$

(b) *For $S \subseteq \mathcal{N}$, $m \in \mathcal{M}(S)$ and $j \in S \cap \mathcal{C}_m$,*

$$\sum_{i \in S^c \cap \mathcal{C}_m} \rho_i x_j^i + \left(1 - \rho(\mathcal{C}_m)\right) x_j^{0m} \geq h(S) \frac{1 - \rho(S \cap \mathcal{C}_m)}{\sum_{j \in S \cap \mathcal{C}_m} f_j}. \tag{50}$$

Proof

(a) The result follows directly by substituting inequality (44) to the elementary identity

$$x_j^{0m} = \sum_{k,l \in \mathcal{C}_m} \frac{s_{kl} f_{kl}}{1 - \rho(\mathcal{C}_m)} E\left[L_j \mid B_{kl} = 1\right],$$

valid under nonidling policies.

(b) The result follows directly from Proposition 2(c), by noticing that

$$E\left[L_j \mid B_{m,S} = 0\right] = \frac{1}{1 - \rho(S \cap \mathcal{C}_m)} \left\{ \sum_{i \in S^c \cap \mathcal{C}_m} \rho_i x_j^i + (1 - \rho(\mathcal{C}_m))\, x_j^{0m} \right\}.$$

■

Remark: Notice that constraints (50) are nonlinear, yet convex, which makes them computationally tractable. Notice further that the nonlinear term in them involves the server visit frequencies f_i's, which are not known in general. However, the achievable values of the f_i's are constrained by linear equality constraints (41) and (42) in Proposition 1. Combining these constraints yields improved convex bounds.

8 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 may be outlined as follows: Given a vector $\boldsymbol{z}$ and a symmetric real matrix $\boldsymbol{Z}$, consider the following question: What is a necessary and sufficient condition that captures the fact that, 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 the matrix $\boldsymbol{Z} - \boldsymbol{z}\boldsymbol{z}'$, which represents the covariance matrix of $\boldsymbol{\zeta}$, be positive semidefinite, i.e., $\boldsymbol{Z} - \boldsymbol{z}\boldsymbol{z}' \succeq \boldsymbol{0}$. This condition is formulated in matrix notation as

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

Applying this idea to the performance variables introduced in Table 1 yields directly the following result.

Theorem 6 *Under any dynamic stable policy, the following semidefinite constraints hold:*

(a)

$$\begin{bmatrix} 1 & \boldsymbol{\rho}' \\ \boldsymbol{\rho} & \boldsymbol{R} \end{bmatrix} \succeq \boldsymbol{0}, \tag{51}$$

$$\begin{bmatrix} 1 & \frac{1}{\rho_k}\boldsymbol{R}_{k\cdot} \\ \frac{1}{\rho_k}\boldsymbol{R}_{\cdot k} & \boldsymbol{R}^k \end{bmatrix} \succeq \mathbf{0}, \qquad \textit{for } k \in \mathcal{N}. \tag{52}$$

(b) *If* $E\left[\left(\sum_{j\in\mathcal{N}} L_j\right)^2\right] < \infty$, *then*

$$\begin{bmatrix} 1 & \boldsymbol{x}' \\ \boldsymbol{x} & \boldsymbol{Y} \end{bmatrix} \succeq \mathbf{0}, \tag{53}$$

$$\begin{bmatrix} 1 & \boldsymbol{x}^{k\prime} \\ \boldsymbol{x}^k & \boldsymbol{Y}^k \end{bmatrix} \succeq \mathbf{0}, \qquad \textit{for } k \in \mathcal{N}, \tag{54}$$

$$\begin{bmatrix} 1 & \boldsymbol{x}^{0m\prime} \\ \boldsymbol{x}^{0m} & \boldsymbol{Y}^{0m} \end{bmatrix} \succeq \mathbf{0}, \qquad \textit{for } m \in \mathcal{M}. \tag{55}$$

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 Vandenberghe and Boyd (1996) for a comprehensive review) to solve semidefinite programming problems. Theorem 6 adds a new and, we believe, interesting application of semidefinite programming in stochastic optimization.

9 Summary of bounds and their power

In previous sections we used various equilibrium relations to derive constraints on performance variables which are valid under all suitable classes of scheduling policies. While we have focused there on the physical meaning of these relations, we show in this section how they can be used to provide performance bounds for MQNETs by solving appropriate mathematical programming problems.

We shall consider in what follows a linear cost function

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

and denote by Z the minimum cost achievable under the appropriate class of policies (dynamic stable or static, nonidling and stable) policies,

$$Z = \min\left\{\sum_{j\in\mathcal{N}} c_j x_j \mid \boldsymbol{x} \in \mathcal{X}\right\}.$$

We have summarized in Table 3 several lower bounds and their corresponding mathematical programming formulations, obtained by selecting appropriate subsets of the constraints developed in previous sections.

Bound	Formulation	# variables	# constraints	Constraints
Z_{AG} [a]	linear program	$O(N)$	$O(2^N)$	(39)
Z_{LP1}	linear program	$O(N^2)$	$O(N^2)$	(1), (6), (7), (23)
Z_{LP2}	linear program	$O(N^3)$	$O(N^3)$	(1)-(3), (4)-(12), (23), (25)
Z_{SD1}	semidefinite program	$O(N^2)$	$O(N^2)$	(1), (4), (5), (7), (23), (51)
Z_{SD2}	semidefinite program	$O(N^3)$	$O(N^3)$	(1)-(3), (4)-(12), (23), (25), (51)-(55)
Z_{CONVEX} [b]	convex program	$O(N^2)$	$O(2^N)$	(1), (6), (7), (23), (41), (42), (49), (50)

Table 3: Bounds and formulations.

[a] Computed by N-steps Klimov's algorithm

[b] Bound accounts for changeover times

For example, the lower bound Z_{LP1} is obtained by solving the linear program

$$\begin{aligned} Z_{LP1} = \quad & \max \textstyle\sum_{j\in\mathcal{N}} c_j x_j \\ & \text{subject to} \\ & (1), (6), (7), (23) \end{aligned}$$

An index-based lower bound computed in N steps. The bound Z_{AG}, shown in Table 3, requires further explanation. We shall show how Z_{AG} is computed in N steps by combining one-pass Klimov's adaptive greedy algorithm with the workload bounds in Corollary 3. Klimov (1974) developed his one-pass N-step *adaptive greedy* algorithm (shown in Figure 1) for computing the priority indices that define the optimal policy in the special case of a single-server MQNET. Bertsimas and Niño-Mora (1996a) analyzed Klimov's algorithm using linear programming. The bound we present next is a byproduct of their analysis.

Specifically, let us run Klimov's algorithm on input $(\boldsymbol{c}, \boldsymbol{V})$, where $\boldsymbol{c} = (c_j)_{j\in\mathcal{N}}$ is the cost vector and $\boldsymbol{V} = (V_i^S)_{i\in\mathcal{N}, S\subseteq\mathcal{N}}$, with the V_i^S's given by (26). The algorithm produces as output a vector $\bar{\boldsymbol{y}} = (\bar{y}(S))_{S\subseteq\mathcal{N}}$ and a vector of indices $\boldsymbol{\gamma} = (\gamma_i)_{i\in\mathcal{N}}$. We assume for ease of notation that

$$\gamma_1 \leq \gamma_2 \leq \cdots \leq \gamma_N .$$

Let set function $g(S)$ be given by (38), and let us define

$$Z_{AG} = \gamma_1 g\left(\{1, \ldots, N\}\right) + (\gamma_2 - \gamma_1) g\left(\{2, \ldots, N\}\right) + \cdots + (\gamma_N - \gamma_{N-1}) g\left(\{N\}\right).$$

Theorem 7 *The value Z_{AG} is a lower bound on the optimal value Z.*

Proof

Input: $(\boldsymbol{c}, \boldsymbol{V})$.

Output: $(\pi, \bar{\boldsymbol{y}}, \boldsymbol{\gamma})$, where $\pi = (\pi_1, \ldots, \pi_N)$ is a permutation of $\mathcal{N}$, $\overline{\boldsymbol{y}} = (\overline{y}(S))_{S \subseteq \mathcal{N}}$ and $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_N)$.

Step 0. Set $S_1 = \mathcal{N}$; set $\bar{y}(S_1) = \min \left\{ \frac{c_i}{V_i^{S_1}} : i \in S_1 \right\}$;

pick $\pi_1 \in \operatorname{argmin} \left\{ \frac{c_i}{V_i^{S_1}} : i \in S_1 \right\}$;

set $\gamma_{\pi_1} = \bar{y}(S_1)$.

Step k. For $k = 2, \ldots, N$:

set $S_k = S_{k-1} \setminus \{\pi_{k-1}\}$; set $\bar{y}(S_k) = \min \left\{ \frac{c_i - \sum_{j=1}^{k-1} V_i^{S_j} \bar{y}(S_j)}{V_i^{S_k}} : i \in S_k \right\}$;

pick $\pi_k \in \operatorname{argmin} \left\{ \frac{c_i - \sum_{j=1}^{k-1} V_i^{S_j} \bar{y}(S_j)}{V_i^{S_k}} : i \in S_k \right\}$;

set $\gamma_{\pi_k} = \gamma_{\pi_{k-1}} + \bar{y}(S_k)$.

Step N+1. For $S \subseteq \mathcal{N}$: set

$$\overline{y}(S) = 0, \qquad \text{if } S \notin \{S_1, \ldots, S_N\}.$$

Figure 1: Klimov's adaptive greedy algorithm.

Bertsimas and Niño-Mora (1996a) showed that vector $\bar{\boldsymbol{y}}$ is a feasible solution of the linear program

$$(LD) \qquad \underline{Z} = \max \sum_{S \subseteq \mathcal{N}} g(S) y(S)$$
$$\text{subject to} \sum_{S : i \in S \subseteq \mathcal{N}} V_i^S y(S) \leq c_i, \qquad \text{for } i \in \mathcal{N}$$
$$y(S) \geq 0, \qquad \text{for } S \subseteq \mathcal{N},$$

which is the dual of

$$(LP) \qquad \underline{Z} = \min \sum_{i \in \mathcal{N}} c_i x_i$$
$$\text{subject to} \sum_{i \in S} V_i^S x_i \geq g(S), \qquad \text{for } S \subseteq \mathcal{N}$$
$$x_i \geq 0, \qquad \text{for } i \in \mathcal{N}.$$

Furthermore, they showed that

$$\gamma_i - \gamma_{i-1} = \bar{y}(\{i, \ldots, N\}), \text{ for } i \in \mathcal{N}.$$

It thus follows that $Z_{AG} \leq \underline{Z}$. Since, in addition, we have by Corollary 3 that $\underline{Z} \leq Z$, the result follows. ■

Performance bounds for second moments. In previous sections we have focused our attention on computing performance bounds for first moments of queue lengths. We now turn our attention to finding performance bounds for second moments. To the best of our knowledge, there has not

been any characterization of the performance region of second moments in the literature, even for single-server MQNETs.

We consider now a performance cost function that involves second-order moments. In particular, given costs c_j and h_j associated with class j customers, we consider the problem of finding a lower bound on the cost

$$\sum_{j \in \mathcal{N}} \left(c_j E[L_j] + h_j E[L_j^2] \right), \tag{56}$$

valid under all admissible policies.

We can compute a lower bound on the optimal expected cost by solving the semidefinite programming problem with a quadratic cost function of minimizing objective (56) subject to the constraints corresponding to the bound Z_{SD2} in Table 3.

9.1 Numerical Results

We performed some limited numerical experiments to assess the quality of some of the bounds we derived. The network we considered consists of two stations. Class 1 arrives at station 1, then visits station 2 forming class 2, it revisits station 2 forming class 3, visits station 1 forming class 4, and finally exits from the network. Both the interarrival times of class 1 and the service times of all classes are exponentially distributed. The arrival rate $\lambda = 1$. The mean service times satisfy: $\beta_1 = 0.25\beta_2$ and $\beta_3 = 0.25\beta_4$. Therefore, the traffic intensities at the two stations are $\rho_1 = \beta_1 + \beta_4$, and $\rho_2 = \beta_2 + \beta_3$.

Classes 1 and 4 compete for service at station 1 and have changeover times $s_{14} = s_{41}$. Similarly, Classes 2 and 3 compete at Station 2 and have changeover times $s_{23} = s_{32}$. We define the changeover ratio (CH): $CH = s_{14}/\beta_1 = s_{23}/\beta_3$, i.e., we select the changeover times so that the changover ratio at each station is the same.

Table 4 reports computational results for parameters such that $\rho_1 = \rho_2$. We simulated all four possible priority policies, and report the performance of the best one. While it is possible that priority policies are weak policies, the lower bound Z_{CONVEX} seems also weak, as the traffic intensity increases. The quality of the bound is insensitive to the changeover ratio.

10 From formulations to policies for MQNETs

We consider in this section the problem of designing a policy that nearly minimizes a performance objective $\sum_{j \in \mathcal{N}} c_j x_j$. Unlike in the single station case, the relaxations we have considered for MQNETs do not provide an optimal policy for this problem. In this section we propose two techniques to extract heuristic policies from the relaxations.

CH	ρ_1	ρ_2	Z_{CONVEX}	$Z_{PRIORITY}$
0.0	0.2	0.2	0.43	0.54
0.2	0.2	0.2	0.52	0.63
0.4	0.2	0.2	0.71	0.83
0.6	0.2	0.2	0.87	1.01
0.8	0.2	0.2	1.09	1.24
1.0	0.2	0.2	1.31	1.43
0.0	0.5	0.5	1.12	2.16
0.2	0.5	0.5	1.25	2.33
0.4	0.5	0.5	1.43	2.72
0.6	0.5	0.5	1.62	3.09
0.8	0.5	0.5	1.84	3.51
1.0	0.5	0.5	2.17	4.42
0.0	0.9	0.9	3.05	17.12
0.2	0.9	0.9	3.47	18.31
0.4	0.9	0.9	4.13	21.73
0.6	0.9	0.9	4.92	25.86
0.8	0.9	0.9	6.13	30.55
1.0	0.9	0.9	8.39	41.77

Table 4: The performance of the bound Z_{CONVEX}, and the best priority policy as a function of the changoever ratio CH, and the traffic intensities ρ_1, ρ_2.

10.1 A priority-index policy for MQNETs

The first policy we propose is defined as follows:

1. Compute indices $\gamma_1, \ldots, \gamma_N$ by running Klimov's algorithm (see Figure 1) on input $(\boldsymbol{c}, \boldsymbol{V})$.
2. Schedule customers at each station by giving higher preemptive priority to customer classes with higher index γ_i.

Notice that the policy is optimal for the single station case. In the multi-station case one needs to consider the issue of whether the proposed policy is stable.

From a physical point of view, we can interpret the policy as follows: We create a new fictitious station, which can be interpreted as if all servers of the network are pulled into a single resource. The arrival rates, processing times and routing information remain the same. The indices $\boldsymbol{\gamma}$ are exactly the optimal Klimov indices in this fictitious single-server network. Notice that the indices do not have any information on the structure of the network, namely which classes are served by which server. They only take into account the work that the network needs to process.

As in Klimov (1978), it can be shown that the index γ_i may be interpreted as the maximum rate of decrease in holding cost rate per unit of network processing time for a customer whose current class is i, i.e.,

$$\gamma_i = \max_{S \ni i} \frac{c_i - \sum_{j \in S^c} p_{ij}(S) c_j}{V_i^S}, \qquad \text{for } i \in \mathcal{N},$$

where $p_{ij}(S)$ is the probability that a customer currently in class $i \in S$ visits class $j \in S^c$ after first leaving classes in S. Notice that

$$p_{ij}(S) = p_{ij} + \sum_{k \in S} p_{ik} p_{kj}(S), \qquad \text{for } i \in S,\ j \in S^c.$$

10.2 Policies from relaxations for networks with finite buffers

We assume that the total number of customers in each station in the network is bounded by C.

Recall that $L_S = \sum_{i \in S} L_i$. We introduce the following variables for $i = 1, \ldots, N$, $m = 1, \ldots, M$ and $l = 0, \ldots, C$:

$$z_{i,m,l} = P\left\{L_{\mathcal{C}_m} = l \mid B_i = 1\right\},$$

$$z_{m,l} = P\left\{L_{\mathcal{C}_m} = l\right\}.$$

Theorem 3 specialized for $S = \mathcal{C}_m$ gives the following equations:

$$\alpha(\mathcal{C}_m) z_{m,l} + \sum_{i \in \mathcal{C}_m^c} \lambda_i p(i, \mathcal{C}_m) z_{i,m,l} = \sum_{i \in \mathcal{C}_m} \lambda_i \left(1 - p(i, \mathcal{C}_m)\right) z_{i,m,l+1},$$

where $z_{i,m,C+1} = 0$.

We next consider the relaxation that involves both the variables $\boldsymbol{z}, \boldsymbol{Z}$, as well as the variables $\boldsymbol{x}, \boldsymbol{X}$. The proof of the theorem is immediate and thus omitted.

Theorem 8 *For $C = \infty$ the optimal solution value of the following infinitely dimensional linear*

program provides a lower bound on the minimum expected holding cost rate

$$
\begin{aligned}
\underline{Z} = \min \quad & \boldsymbol{c}'\boldsymbol{x} \\
\textit{subject to} \quad & -\boldsymbol{\alpha}\boldsymbol{x}' - \boldsymbol{x}\boldsymbol{\alpha}' + (\boldsymbol{I}-\boldsymbol{P})'\boldsymbol{\Lambda}\boldsymbol{X} + \boldsymbol{X}'\boldsymbol{\Lambda}(\boldsymbol{I}-\boldsymbol{P}) = (\boldsymbol{I}-\boldsymbol{P}')\boldsymbol{\Lambda} + \boldsymbol{\Lambda}'(\boldsymbol{I}-\boldsymbol{P}) \\
& \alpha(\mathcal{C}_m) z_{m,l} + \sum_{i \in \mathcal{C}_m^c} \lambda_i p(i, \mathcal{C}_m) z_{i,m,l} = \\
& \sum_{i \in \mathcal{C}_m} \lambda_i \left(1 - p(i, \mathcal{C}_m)\right) z_{i,m,l+1}, \quad \forall i, m, l \\
& \sum_{j \in \mathcal{C}_m} x_j^i = \sum_{l=0}^{C} l z_{iml} \quad \forall i, m \\
& \sum_{j \in \mathcal{C}_m} x_j = \sum_{l=0}^{C} l z_{ml} \quad \forall m \\
& x_j \geq \sum_{i \in \mathcal{C}_m} \rho_i x_j^i, \quad \forall j, m \\
& z_{jl} \geq \sum_{i \in \mathcal{C}_m} \rho_i z_{ijl}, \quad \forall j, l, m \\
& z_{ml} \leq 1, \quad \forall m, l \\
& \boldsymbol{x}, \boldsymbol{X}, \boldsymbol{z}, \boldsymbol{Z} \geq \boldsymbol{0}.
\end{aligned}
$$

For finite C, the above linear program does not give a formal bound, because equilibrium relations (23) do not necessarily hold with finite C. However, if we do not include these constraints and remove variables x_j from the formulation we do obtain a valid bound.

For $C = \infty$, the above linear program is not interesting as it would be very difficult to solve. However, if we truncate the state space, by imposing the condition that $z_{i,j,C+1} = 0$, we heuristically expect that the bound for finite C would be close to the bound for $C = \infty$. Moreover, as the number of variables of the linear program of Theorem 8 is $O(NMC)$, the problem is tractable. Its main advantage is that we can obtain heuristic policies from this linear program as follows.

A Heuristic Policy

1. We solve the formulation of Theorem 8.
2. When there is a service completion at station m, the server is set to work on class i with probability

$$P\{B_i = 1 \mid L_{\mathcal{C}_m} = l\} = \frac{P\{L_{\mathcal{C}_m} = l \mid B_i = 1\} P\{B_i = 1\}}{P\{L_{\mathcal{C}_m} = l\}} = \frac{z_{iml}\rho_i}{z_{ml}}.$$

   The server selects to idle with probability

$$1 - \sum_{i \in \mathcal{C}_m} \frac{z_{iml}\rho_i}{z_{ml}}.$$

In general, the optimal policy would be to decide the probabilities that

$$P\{B_i = 1 \mid \boldsymbol{L} = \boldsymbol{l}\},$$

where $\boldsymbol{L} = (L_1, \ldots, L_N)$ and $\boldsymbol{l} = (l_1, \ldots, l_N)$. Under the proposed heuristic policy, the server bases the decision of which customer to serve next, if any, on the total number of customers in its station. The policy has the attractive feature of being decentralized once the linear program is solved, as it only uses information that is local to the server.

A Some basic results from the Palm calculus of point processes

In this appendix we review for the reader's reference some basic notions and results from the Palm calculus of point processes that are used throughout the paper. For a thorough and rigorous treatment of the subject we refer the reader to Baccelli and Brémaud (1994).

Consider a discrete stochastic process $\{L(t)\}_{t\in\Re}$, with sample paths right-continuous with left limits, representing the state evolution of a stochastic system, and let $N = \{T_n\}_{n=-\infty}^{\infty}$ be a point process of related epochs, with $\cdots < T_{-1} < 0 \leq T_0 < T_1 < \cdots$. We may interpret $L(t)$ as the system state at time t, and T_n as the nth event epoch. We assume that processes $\{L(t)\}_{t\in\Re}$ and $N = \{T_n\}_{n=-\infty}^{\infty}$ are *adapted* to a common *history* $\{\mathcal{F}_t\}_{t\in\Re}$, and that they are *stationary*, which captures mathematically the intuitive notion that the system evolution and the stream of epochs are *time-homogeneous*.

For ease of notation we write $L = L(0)$, $L^- = L(0-)$ and $L^+ = L(0+)$, where $L(0-)$ and $L(0+)$ denote the left and right limits of $L(t)$ at $t = 0$, respectively. We denote $P\{L = l\}$ the *equilibrium probability* that the system state at an *arbitrary time* (such as $t = 0$) is l, and write the corresponding expectation as $E[L]$. We denote $P^N\{L = l\}$ the *equilibrium probability* that the system state *embedded* at an *arbitrary epoch* is l, and write the corresponding expectation as $E^N[L]$. $P^N\{\cdot\}$ is the *Palm probability* with respect to stationary point process N, and $E^N[\cdot]$ is the corresponding *Palm expectation*. By definition of Palm probability, $T_0 = 0$, i.e., time $t = 0$ corresponds to an arbitrary epoch of N.

Intensity and stochastic intensity

We denote $N[a, b)$ the number of points/event epochs that lie on time interval $[a, b)$, with $a < b$.

Definition 4 (Intensity) The expected number of points that lie in a unit length interval,

$$\lambda = E\left[N\left([0, 1)\right)\right],$$

is called the *intensity* of N.

The intensity of a point process may be interpreted as a *global* measure of the rate of points/epochs per unit time.

In some applications, such as queueing systems, the frequency at which events take place may depend on the current state of the system. For example, in an $M/M/2$ queue, departures happen at a higher rate when the two servers are busy than when only one is. This intuitive notion of local density of points/frequency of epochs in a point process is captured by the concept of *stochastic intensity.*

Let $\{\lambda(t)\}_{t\in\Re}$ be a nonnegative process, adapted to the history $\{\mathcal{F}_t\}_{t\in\Re}$.

Definition 5 (Stochastic Intensity) The process $\{\lambda(t)\}_{t\in\Re}$ is called an $\mathcal{F}_t$-*stochastic intensity* of N if

(i) it is locally integrable; that is, $\int_B \lambda(s)ds < \infty$ for all bounded Borel sets B; and

(ii) For all $a < b$,

$$E[N(a,b) \mid \mathcal{F}_a] = E\left[\int_a^b \lambda(s)\, ds \mid \mathcal{F}_a\right].$$

The value $\lambda(t)$ may be interpreted as the instantaneous rate at which points/epochs occur at time t.

Superposition of point processes

Let $N_1, \ldots, N_K$ be stationary point processes, defined in a common probability space. Let $\lambda_1, \ldots, \lambda_K$ be their respective finite intensities. Assume that point process N may be obtained through the *superposition* of processes $N_1, \ldots, N_K$, i.e., process N has a point at time t if any of the processes $N_1, \ldots, N_K$ has a point at that time. We shall write then $N = N_1 + \cdots + N_K$. The intensity of N can be shown to be $\lambda = \lambda_1 + \cdots + \lambda_K$. The following theorem represents the Palm expectation with respect to the composite process N in terms of the Palm probabilities with respect to the elementary processes N_k.

Theorem 9 (Superposition) *The following relation holds:*

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

Thinning of a point process and conditioning

Let $\mathcal{A}$ be a measurable event, and consider the point process obtained by counting only points from process N at which event $\mathcal{A}$ happens. We refer to the resulting point process $N_{\mathcal{A}}$ as a *thinned* process. The next result relates the Palm probabilities with respect to the original process N and the thinned process $N_{\mathcal{A}}$. Let $\lambda(N)$ and $\lambda(N_{\mathcal{A}})$ denote the intensities of point processes N and $N_{\mathcal{A}}$, respectively.

Theorem 10 *The following relations hold:*

(a)

$$P^{N_{\mathcal{A}}}\{\cdot\} = P^N\{\cdot \mid \mathcal{A}\}.$$

(b)

$$\lambda(N_{\mathcal{A}}) = \lambda(N)P^N(\mathcal{A}).$$

Relating time and event expectations: Papangelou's formula

Papangelou's formula is a fundamental and powerful result that provides the link between time-stationary probability, Palm probability and stochastic intensity.

Theorem 11 (Papangelou (1972)) *If N admits a stochastic intensity $\{\lambda(t)\}_{t\in\Re}$, then*

$$E[\lambda(0)L(0)] = \lambda E^N\left[L^-\right].$$

Several important results of queueing theory on the relation between the queueing state distributions at an arbitrary time and at an arbitrary epoch follow directly from Papangelou's formula.

Theorem 12 (PASTA: Poisson Arrivals See Time Averages) *If N is a Poisson process, then*

$$E^N\left[L^-\right] = E[L].$$

Theorem 13 (Conditional PASTA) *Assume that N admits a stochastic intensity $\{\lambda(t)\}_{t\in\Re}$, with $\lambda(t) = \mu B(t)$, and where $B(t) \in \{0,1\}$ for all $t \in \Re$. Then,*

$$E^N\left[L^-\right] = E[L \mid B = 1].$$

Acknowledgement

We would like to thank Jay Sethuraman for performing the computational experiments reported in Section 9.

References

[1] Baccelli, F. and P. Brémaud (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 J. Niño-Mora (1994). Restless bandits, linear programming relaxations and a primal-dual heuristic. *Oper. Res.*, to appear.

[4] Bertsimas, D. and J. Niño-Mora (1996a). 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 J. Niño-Mora (1998). Optimization of multiclass queueing networks with changeover times via the achievable region approach: Part I, the single-station case. *Math. Oper. Res.* to appear.

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

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

[8] Bertsimas, D. and H. Xu (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] Burke, P.J. (1956). The output of a queueing system. *Oper. Res.* **4** 699-704.

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

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

[13] Finch, P.D. (1959). On the distribution of queue size in queueing problems. *Acta Math. Hungar.* **10** 327-336.

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

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

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

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

[18] Klimov, G.P. (1978). Time sharing service systems II. *Theory Probab. Appl.* **23** 314-321.

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

[20] Kumar, P.R. and S.P. Meyn (1996). Duality and linear programs for stability and performance analysis of queueing networks and scheduling policies. *IEEE Trans. Autom. Control* **41** 4-16.

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

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

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

[24] Papadimitriou, C.H. and J.N. Tsitsiklis (1993). The complexity of optimal queueing network control. Working Paper LIDS 2241, MIT.

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

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

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

[28] Wein, L.M. (1990). Scheduling networks of queues: Heavy traffic analysis of a two-station network with controllable inputs. *Oper. Res.* **38** 1065-1078.