Ann Oper Res
DOI 10.1007/s10479-008-0443-x

Near optimal control of queueing networks over a finite time horizon

Yoni Nazarathy · Gideon Weiss

Abstract We propose a method for the control of multi-class queueing networks over a finite time horizon. We approximate the multi-class queueing network by a fluid network and formulate a fluid optimization problem which we solve as a separated continuous linear program. The optimal fluid solution partitions the time horizon to intervals in which constant fluid flow rates are maintained. We then use a policy by which the queueing network tracks the fluid solution. To that end we model the deviations between the queuing and the fluid network in each of the intervals by a multi-class queueing network with some infinite virtual queues. We then keep these deviations stable by an adaptation of a maximum pressure policy. We show that this method is asymptotically optimal when the number of items that is processed and the processing speed increase. We illustrate these results through a simple example of a three stage re-entrant line.

Keywords Queueing control · Multi-class queueing networks · Infinite virtual queues · Maximum pressure policies · Fluid approximations · Continuous linear programming

Introduction

Queueing networks are commonly used to model service, manufacturing and communication systems. In many situations one is interested in control of the network over a finite time horizon that attempts to minimize costs and maximize utility. In this respect, a sensible objective is to minimize the holding costs of the queues accumulated over the time horizon.

One theoretical approach to such finite horizon problems is to consider them as deterministic, discrete scheduling problems, cf. Lawler et al. (1993), Goemans and Williamson (1996) and Fleischer and Sethuraman (2003). In practice however these scheduling problems are too large to be tractable, and furthermore, an optimal schedule may not withstand the

Research supported in part by Israel Science Foundation Grant 249/02 and 454/05 and by European Network of Excellence Euro-NGI.

Y. Nazarathy · G. Weiss (✉)
Department of Statistics, The University of Haifa, Haifa, Israel
e-mail: gweiss@stat.haifa.ac.il

trial of application: As it is implemented over time, inaccuracies in the data and unexpected events (many small ones and a few large ones) accumulate and interfere with the solution, and there is no theory to say how close or far from optimum the result may be. Another theoretical approach is to model these problems by discrete stochastic systems and solve them as Markov decision problems, or approximate them on a diffusion scale by a continuous stochastic Brownian control problem, cf. Kushner (2001), Harrison (1988), Wein (1992), Kelly and Laws (1993), Harrison (1996), Harrison and Van Mieghem (1997) and Maglaras (1999). Markov decision problems or Brownian control problems usually focus on the optimization of the steady state of the system. They may therefore not be suitable for finite horizon problems, where typically the initial queue lengths and the total number of items processed are of the same order of magnitude, and one does not expect the system to reach steady state.

The problem which we address here has features of both approaches: In the finite time horizon we only schedule a finite batch of jobs, but we model these jobs in a multi-class queueing network. As suggested in Weiss (1992), the method we use is to solve a deterministic fluid optimization problem which approximates the system, and then use decentralized local on-line controls to track the fluid solution. To carry this out we integrate three recent ideas which have been developed independently: (1) Solution of separated continuous linear programs (SCLP) by means of an exact simplex type algorithm, see Weiss (2008). (2) Modeling of queueing systems with unlimited supply of work by means of infinite virtual queues, as used in Adan and Weiss (2005, 2006), Kopzon and Weiss (2002, 2007) and Weiss (2005). (3) Maximum Pressure policies for stochastic processing networks as described in Dai and Lin (2005), see also Dai and Lin (2006).

As a first step, we discard the detailed information on jobs, and aggregate them into classes which are characterized by average processing times, and by their routes through the system. This yields a multi-class queueing network, which we wish to control optimally (Sect. 1). Next, our method approximates the multi-class queueing network by a deterministic continuous multi-class fluid network for which we formulate a finite horizon optimal control problem which is solved as a SCLP. The optimal fluid solution partitions the finite time horizon into time intervals distinguished by sets of empty and non-empty fluid buffers, and by constant fluid flow rates (Sect. 2). We now associate with each time interval a multi-class queueing network where each empty fluid buffer corresponds to a standard queue and each non-empty fluid buffer corresponds to an infinite virtual queue. The state of this associated system measures the deviation of the original system from the fluid solution (Sect. 3).

We then implement an on-line control of the queueing network by the use of a maximum pressure policy, where the pressure is calculated from the state of the associated network. This keeps the deviations from the fluid solution stable, and so the queueing network tracks the optimal fluid solution (Sect. 4). We call this procedure the Maximum Pressure Fluid Tracking Policy (MaxFTP).

The solution of the fluid control problem is centralized, and performed at the outset. The maximum pressure tracking is decentralized, and performed on-line using at each queue its own queue length and the queue lengths of queues directly downstream from it. This scheme is asymptotically optimal in the following sense: If we scale up the number of jobs in the system, and speed up the processing by the same amount, then the discrete stochastic system will converge to the optimal fluid solution, and no other policy can achieve asymptotically better results (Sect. 5).

Our purpose in this paper is to introduce this method, and to sketch the proofs. In fact there is not much to prove, as most of the results we need were derived in Weiss (1992) and Dai and Lin (2005), and we need merely to adapt them to our framework, in Sect. 2, in Theorem 4.1 and Corollary 4.2. The main new result is the asymptotic optimality of MaxFTP which is proven in Theorem 5.1.

To illustrate MaxFTP, we describe its implementation for a simple re-entrant line with 2 servers and 3 classes (this network has been recently studied in Adan and Weiss (2006)). We demonstrate the effectiveness of our results by means of simulations in which the asymptotic attributes of MaxFTP are empirically tested (Sect. 6).

Related fluid approaches for controlling multi-class queueing networks have been used in Avram et al. (1995), Maglaras (1999, 2000), Chen and Meyn (1999), Meyn (2001), Meyn (2003), Chen et al. (2003) and others. MaxFTP is distinguished in that it is geared to control the transient finite horizon system, and for that purpose we use an optimal fluid solution.

In the context of wireless communication systems the proposed framework may be applied to mobile ad-hoc wireless networks (MANETs) that are both highly dynamic and heavily congested: When the network is highly dynamic, link conditions vary quickly and as a result link capacities, network topology and routes constantly change. When the network is heavily congested, the sojourn time of messages is high. Combining both highly variable dynamics and heavy congestion has the consequence that messages may experience changes in link capacities, network topology and routes while in transit. If a predictive, location based, routing scheme is employed (such as that presented by Shah and Nahrstedt in Shah and Nahrstedt (2002)), predictions regarding the relative stability of link conditions may be made and the duration of the finite time horizons determined. In this case, our finite horizon approach may be used repeatedly for the short durations in which link conditions are relatively stable.

1 Finite horizon multi-class queueing networks

Multi-class queueing networks (MCQN) have been studied extensively in the past 15 years. They were introduced in Harrison (1988) and since then were analyzed mostly in the context of infinite horizon models with respect to fluid or diffusion approximations, cf. Dai (1995), Bramson (1998) and Williams (1998).

A MCQN consists of $k \in \mathcal{K} = \{1, \dots, K\}$ job-classes and $i \in \mathcal{I} = \{1, \dots, I\}$ servers. Jobs of class k queue up in buffer k, and we let the queue length $Q_k(t)$ be the number of jobs of class k in the system at time t. We let $Q_k(0)$, $k \in \mathcal{K}$ be the initial queue lengths. Buffer k is served by server $\sigma(k)$, and the constituency of server i is $C_i = \{k \mid \sigma(k) = i\}$. In general a server may serve several classes, i.e. $|C_i| > 1$, hence the term multi-class. The topology of the network is described by the $I \times K$ constituency matrix $\mathbf{A}$ with elements $A_{ik} = 1$ if $k \in C_i$, $A_{ik} = 0$ otherwise.

We are only interested in the MCQN over a finite time horizon $[0, T]$. We may assume that all the jobs which will be processed during that time are already in the system at time 0. This assumption is without loss of generality: The general multi-class model has an arrival stream $A(t)$ which is often thought of as being supplied by buffer 0 that represents the outside world and contains an infinite supply of jobs. Since we are only interested in a finite time horizon, the supply of jobs can be taken as finite, so that the outside world can be replaced by an additional buffer in the network with a finite initial supply of all the jobs that will be served, and with a dedicated server that will release them into the rest of the system as the arrival process.

For $\ell = 1, 2, \dots$, the ℓ's job out of buffer k requires processing amount $X_k(\ell)$, after which the job may either leave the system or move to another buffer. $S_k(t) = \max\{n \mid \sum_{\ell=1}^{n} X_k(\ell) \le t\}$ counts the number of jobs completed at buffer k by processing for a total time t. $\phi_{kk'}(\ell)$ is the indicator of the event that the ℓ's job out of buffer k moved into buffer $k' \in \mathcal{K} \setminus k$. Let $\Phi_{kk'}(n) = \sum_{\ell=1}^{n} \phi_{kk'}(\ell)$, this is a count of the number of jobs routed from buffer k to k' out of the first n jobs served at buffer k.

The MCQN is controlled by allocating processing times to the buffers. Let $T_k(t)$ be the total time allocated to buffer k by server $\sigma(k)$, during $[0, t]$. Then the dynamics of the queues are described by:

$$Q_k(t) = Q_k(0) - S_k(T_k(t)) + \sum_{k' \in \mathcal{K} \setminus k} \Phi_{k'k}(S_{k'}(T_{k'}(t))). \tag{1}$$

The allocated times have to satisfy:

$$T_k(0) = 0, \quad T_k(t) \text{ non decreasing,}$$
$$\sum_{k \in C_i} T_k(t) - T_k(s) \le t - s \quad \text{for all } s < t \text{ and each } i \in \mathcal{I}.$$

In particular each $T_k(t)$ is Lipschitz continuous with constant 1, and its derivative $\dot{T}_k(t)$ exists for almost all t, and satisfies:

$$\mathbf{A}\dot{T}(t) \le \mathbf{1}, \quad \dot{T}(t) \ge 0, \quad 0 < t < T, \tag{2}$$

where $\mathbf{1}$ denotes a vector of 1's.

Additional constraints have to be satisfied by $T_k(t)$, $Q_k(t)$, $k \in \mathcal{K}$. First and foremost, $Q_k(t) \ge 0$ and no processing can occur when $Q_k(t) = 0$. We will also assume throughout this paper that servers cannot be split so each server can work on only one job at a time. In addition we assume, that jobs are not preempted. Hence for almost all t, $\dot{T}_k = 0$ or $\dot{T}_k(t) = 1$, and $\dot{T}_k(t)$ can only change from 1 to 0 when $S_k(T_k(t))$ has a jump of 1, that is when the processing of a job is completed.

The cost associated with the MCQN over the finite time horizon is

$$V = \int_0^T \sum_{k=1}^{K} w_k Q_k(t) dt, \tag{3}$$

which is the total inventory cost over the time horizon with holding costs rates w_k. If $w_k = 1$ for all k then V is the total work in process during the time horizon, also equal to the sum of sojourn times over $[0, T)$ of all jobs, where we assume that the sojourn time of a job that does not leave the system by time T is T.

Minimization of V is also equivalent to maximization of the sum of the times from the completion of each job until T. Minimization of V when $X_k(\ell)$, $\phi_{kk'}(\ell)$ are known is an NP hard scheduling problem (job shop scheduling). The probabilistic version for infinite time horizon, with long term average cost minimization, is a Markov decision problem, which can sometimes be approximated by a Brownian control problem. Exact solution of the finite horizon problem under probabilistic assumptions is intractable. We will therefore attempt to solve the problem approximately.

To do so, we first of all discard all the detailed information of $X_k(\ell)$, $\phi_{kk'}(\ell)$, and retain only averages. Remarkably, our asymptotic results show that for large systems this loss of information does not degrade the performance: The value achieved by our method converges to the value of the optimal solution with full information (Theorem 5.1).

We assume the following about the sequences of processing times and routing of the jobs:

$$\lim_{t\to\infty} \frac{S_k(t)}{t} = \mu_k, \tag{4}$$

$$\lim_{n\to\infty} \frac{\Phi_{kk'}(n)}{n} = P_{kk'}, \tag{5}$$

$$\lim_{n\to\infty} \frac{1}{n}\sum_{\ell=1}^{n} X_k(\ell)^{1+\epsilon} \le C, \tag{6}$$

where the last requirement has to hold for some $\epsilon > 0$ and some $C < \infty$. μ_k is the long term average processing rate, and $P_{kk'}$ the long term routing proportion ($\mathbf{P}$ is the $K \times K$ matrix of these values).

It is customary in papers on MCQN to cast (4)–(6) in a probabilistic framework. One assumes a stochastic process on probability space Ω and requires (4)–(6) to hold for almost every $\omega \in \Omega$. Examination of the proofs in Dai and Lin (2005) shows that for our purposes this is not necessary: our results hold for every sequence of $X_k(\ell)$, $\phi_{kk'}(\ell)$ which satisfies (4)–(6).

We define the input output matrix of the network by:

$$\mathbf{R} = (\mathbf{I} - \mathbf{P}')\mathrm{diag}(\mu), \tag{7}$$

where $\mathbf{I}$ is the identity matrix and $\mathrm{diag}(\mu)$ is a matrix with the rates μ_k in the diagonal. Here $R_{k'k}$ is the long term average rate at which buffer k' is depleted as a result of processing buffer k:

$$R_{k'k} = \begin{cases} \mu_k & k' = k \\ -\mu_k P_{kk'} & k' \neq k. \end{cases}$$

Our approximate solution is in two stages: We first use $Q(0), w, \mathbf{R}, \mathbf{A}$ to formulate and solve a fluid optimization problem. This is done off line, centrally. We then use the fluid solution and the current queue lengths for decentralized tracking of the fluid solution.

The fluid approximation is suitable only when we process a large number of jobs over the time horizon. Our results are therefore asymptotic when the initial workload and processing speed are scaled up. We let the queueing network with $Q(0)$ and μ be our basic unit system and define a sequence of queueing networks. All the networks share the same sequence of processing times and routing indicators. For $N = 1, 2, \dots$, the N scaled network is represented by $Q_k^N(t), T_k^N(t)$, in which we have $Q_k^N(0) = NQ_k(0)$, and the processing times are $X_k(\ell)/N$. Thus the initial work load is N times larger than the basic network, and the processing is speeded up N fold. In particular, $\mu_k^N = N\mu_k$, and $\mathbf{R}^N = N\mathbf{R}$.

Example model

The example model that we use throughout this paper is the $K = 3$, $I = 2$ re-entrant line with $C_1 = \{1, 3\}$, $C_2 = \{2\}$. An illustration of the network is in Fig. 1. Routing is deterministic: jobs move from class 1 to class 2 and then to class 3, so that $\phi_{12}(\ell) = \phi_{23}(\ell) = 1$, $\ell = 1, 2, \dots$, and all other routing indicators are 0. The processing time sequences are drawn from independent exponential random variables. The processing rates are $\mu_1 = 1$, $\mu_2 = \frac{1}{4}$

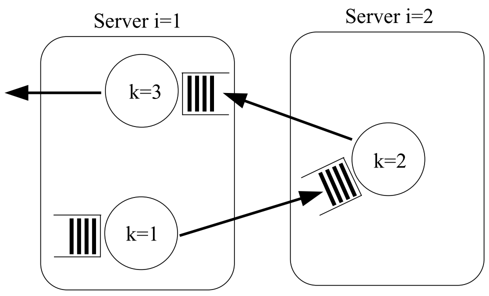

Fig. 1 Example network

and $\mu_3 = 1$. The resulting input-output matrix is:

$$\mathbf{R} = \begin{pmatrix} 1 & 0 & 0 \\ -1 & \frac{1}{4} & 0 \\ 0 & -\frac{1}{4} & 1 \end{pmatrix}.$$

We assume initial queue amounts: $Q_1(0) = 8$, $Q_2(0) = 1$, $Q_3(0) = 15$, a time horizon of $T = 40$ and holding costs $w_1 = w_2 = w_3 = 1$.

2 The multi-class fluid network optimization problem

Fluid approximations have been a major tool in the research on multi-class queueing networks, they have been used to verify the stability of networks, to evaluate performance in steady state, and to control multi-class queueing networks so as to improve their steady state performance. For some relevant recent work see Chen and Yao (1993), Connors et al. (1994), Avram et al. (1995), Dai (1995), Chen and Meyn (1999), Meyn (2001, 2003) and Chen et al. (2003).

Corresponding to the MCQN, we formulate a multi-class fluid network (MCFN) optimization problem: find bounded measurable functions $u_k(t)$ and absolutely continuous functions $q_k(t)$ such that

$$\min V_f = \int_0^T w' q(t) dt \tag{8}$$

s.t.

$$q(t) = q(0) - \int_0^t \mathbf{R} u(s) ds \tag{9}$$

$$\mathbf{A} u(t) \le \mathbf{1} \tag{10}$$

$$q(t), u(t) \ge 0 \tag{11}$$

$$t \in [0, T].$$

Here the dynamics of $q(t)$ are given by

$$q_k(t) = q_k(0) - \mu_k T_k(t) + \sum_{k' \in \mathcal{K} \setminus k} P_{k'k} \mu_{k'} T_{k'}(t) \ge 0,$$

which is the fluid analog of the MCQN dynamics (1). The processing of fluid out of buffer k is at a deterministic continuous rate μ_k, and the fluid out of k is routed in exact proportions $P_{kk'}$ to the other buffers $k' \neq k$. Thus instead of the discrete stochastic nature of the MCQN the MCFN is a continuous deterministic system. A fraction $u_k(t)$ of the server $\sigma(k)$ is allocated to the fluid buffer k at time t, and $T_k(t) = \int_0^t u_k(s)ds$. $u_k(t)$ satisfy the same constraints as $\dot{T}$ in (2), but servers are infinitely divisible, so that $u_k(t)$ can take any value in $[0, 1]$.

The problem (8)–(11) is a special case of a separated continuous linear program (SCLP), cf. Anderson (1981), Anderson and Nash (1987), Bellman (1953) and Pullan (1993). A simplex based algorithm that solves a wide class of SCLP problems optimally in a finite number of steps has been recently found, see Weiss (2008). This has been an open problem for half a century. Current naive implementations of the simplex based algorithm are able to quickly solve MCFN problems in which the dimensions of K and I are of the order of 100.

The analysis in Weiss (2008) reveals important features of the solution, essential to our control method. The MCFN problem is always feasible and bounded, and the SCLP simplex algorithm will always produce a fluid solution that consists of a partition of the time horizon $0 = t_0 < t_1 < \cdots < t_M = T$, where M is bounded, and in each of the time intervals the server allocations $u_k(t)$ are constant, and as a result the fluid buffer levels $q_k(t)$ are continuous piecewise linear.

We let $\tau_m = (t_{m-1}, t_m)$ denote the m'th time interval of the solution, and we denote by $u_k^m = u_k(t),\ t \in \tau_m$ the values of the server allocations. During τ_m, server i will be busy for a fraction $\rho_i^m = \sum_{k \in C_i} u_k^m$. This is the utilization of server i during τ_m, and is always ≤ 1.

In each τ_m some of the buffers will be empty throughout the whole time interval, and the remaining buffers will be non-empty throughout the whole time interval. In general empty buffers are not inactive: they may have fluid flowing into them as well as out of them, with the inflow rate and outflow rates equal. We partition $\mathcal{K}$ during the m'th time interval as follows:

$$\mathcal{K}_0^m = \{k \mid q_k(t) = 0, t \in \tau_m\},$$
$$\mathcal{K}_\infty^m = \{k \mid q_k(t) > 0, t \in \tau_m\}.$$

In our control approach, the fluid solution, summarized by $\tau_m, u^m, \mathcal{K}_0^m, \mathcal{K}_\infty^m$, is calculated off-line at the outset (time 0), from the data T, w, $\mathbf{R}$, $\mathbf{A}$, and the initial fluid levels $q_k(0) = Q_k(0)$. This solution is made available to all the servers.

Solution of the example fluid network

The multi-class fluid network problem for our example is:

$$\min V_f = \int_0^T q_1(t) + q_2(t) + q_3(t)dt$$

s.t.

$$\begin{aligned} q_1(t) &= q_1(0) - \int_0^t \mu_1 u_1(s)ds \\ q_2(t) &= q_2(0) - \int_0^t \mu_2 u_2(s)ds + \int_0^t \mu_1 u_1(s)ds \\ q_3(t) &= q_3(0) - \int_0^t \mu_3 u_3(s)ds + \int_0^t \mu_2 u_2(s)ds \end{aligned} \tag{12}$$

$$
\begin{aligned}
& u_1(t) + u_3(t) \le 1 \\
& u_2(t) \le 1 \\
& u(t), q(t) \ge 0, \\
& t \in [0, T].
\end{aligned}
$$

To understand the dynamic of the fluid solution, we study three different feasible solutions of (12). These are shown in Figs. 2, 3, and 4 where we plot the values $\{q_1(t), q_1(t) + q_2(t), q_1(t) + q_2(t) + q_3(t)\}$ for $0 < t < T$. The three solutions are the last buffer first served (LBFS) solution, and minimum makespan solution and the optimal solution of (12) respectively.

The LBFS policy for a general re-entrant line with flow $1 \to 2 \to \cdots \to K$ gives priority to higher indexed buffers. In our example, server 1 gives priority to buffer 3 over buffer 1,

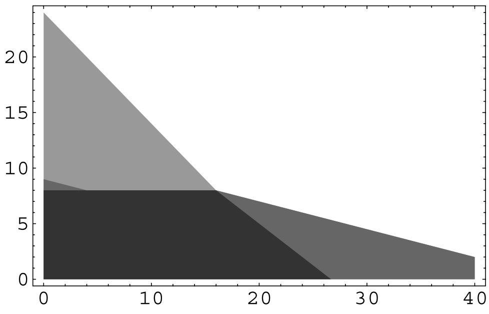

Fig. 2 LBFS. $V_f = 376$

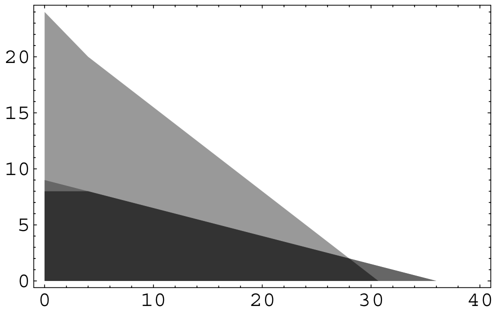

Fig. 3 Minimal makespan. $V_f = 360$

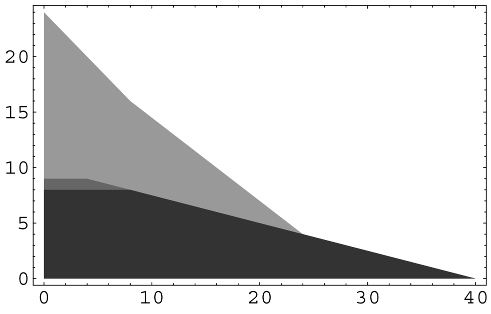

Fig. 4 Optimal SCLP. $V_f = 352$

Table 1 Fluid solution of the example network and parameters of the associated MCQNs with IVQs

		Example data			
Time interval m	=	1	2	3	4
τ_m	=	(0, 4)	(4, 8)	(8, 24)	(24, 40)
(u_1^m, u_2^m, u_3^m)	=	(0, 0, 1)	(0, 1, 1)	$(\frac{1}{4}, 1, \frac{3}{4})$	$(\frac{1}{4}, 1, \frac{1}{4})$
(ρ_1^m, ρ_2^m)	=	(1, 0)	(1, 1)	(1, 1)	$(\frac{1}{2}, 1)$
$\mathcal{K}_0^m$	=	∅	∅	{2}	{2, 3}
$\mathcal{K}_\infty^m$	=	{1, 2, 3}	{1, 2, 3}	{1, 3}	{1}
$(\alpha_1^m, \alpha_2^m, \alpha_3^m)$	=	(0, 0, 1)	$(0, \frac{1}{4}, 1)$	$(\frac{1}{4}, 0, \frac{3}{4})$	$(\frac{1}{4}, 0, 0)$
$\mathbf{R}^m$	=	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{4} & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{4} & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ -1 & \frac{1}{4} & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ -1 & \frac{1}{4} & 0 \\ 0 & -\frac{1}{4} & 1 \end{pmatrix}$

and hence it allocates full capacity to buffer 3 until it is empty, and thereafter enough capacity is allocated to buffer 3 to keep it empty, and the remaining capacity is allocated to buffer 1. Server 2 is allocating full capacity to buffer 2. This is illustrated in Fig. 2. The cost over time $[0, 40]$ is 376. Note that under LBFS server 2 which is the bottleneck server is kept idle from time 4 to time 16. If we continue using LBFS after time 40 the system will empty at 48, for a total cost of 384, (over the time horizon $[0, 48]$).

We can avoid idleness of the bottleneck server, by allocating $u_1(t) = 1/4$ once buffer 2 is empty. Doing so will keep server 2 busy and the system will empty in minimal time: $(q_1(0) + q_2(0))\mu_2^{-1} = 36$. This is called a minimum makespan solution. It is illustrated in Fig. 3: Server 1 is allocated to buffer 3 until $t = 4$, when buffer 2 is empty, at which point server 1 allocates $u_1 = 1/4$ to buffer 1, and the remaining capacity $u_3 = 3/4$ to buffer 3, which is emptying at rate $3/4$. The total cost is 360.

The optimal solution, which minimizes the cost in (12) over the time horizon of $T = 40$ is presented in Fig. 4. It is a compromise between LBFS and minimal makespan. LBFS is employed until time $t = 8$, and after that, the bottleneck server is kept fully utilized. The optimal cost is 352. All details of this fluid solution are in Table 1.

Let t^* be the time at which we start to divert processing capacity from buffer 3 to buffer 1 in order that server 2 will be fully utilized. For minimum makespan $t^* = 4$, for the optimal solution it is $t^* = 8$ and for LBFS it is $t^* = 16$. The cost to empty the system is actually given by the quadratic function $W(t^*) = \frac{1}{2}(t^*)^2 - 8t^* + 384$ and it is minimized at $t^* = 8$. It thus happens that for our example this simple "local optimization" actually solves the optimization problem over all possible controls, as we find by solving the SCLP.

3 MCQNs with infinite virtual queues

The fluid solution indicates that in time interval τ_m, buffers $k \in \mathcal{K}_\infty^m$ are not empty throughout the entire time interval. Assume that we are able to track the fluid solution with the actual MCQN, so that $Q_k(t) > 0$, $k \in \mathcal{K}_\infty^m$ for all $t \in \tau_m$. In that case, the class k buffer always has work available, and so for the dynamics of the network, its level during τ_m is not relevant. This can be modeled by MCQN with infinite virtual queues, which we describe now. For some further examples of infinite virtual queues see Adan and Weiss (2005, 2006), Kopzon and Weiss (2002, 2007) and Weiss (2005).

A multi-class queueing network with infinite virtual queues (MCQN with IVQs) is defined as follows: It consists of classes $\mathcal{K} = \mathcal{K}_0 \cup \mathcal{K}_\infty$, servers $\mathcal{I}$ and constituency matrix $\mathbf{A}$. Queues of classes $k \in \mathcal{K}_0$ are standard queues. Queues of classes $k \in \mathcal{K}_\infty$ are infinite virtual queues: They always have an unlimited supply of jobs available for processing. Let $S_k(t)$ be the counting process of job completions after processing duration t. Let $\Phi_{kk'}(n)$ count the number of jobs routed from queue k to queue k' out of the first n job completions of class k, where $k \in \mathcal{K}$, and $k' \in \mathcal{K}_0 \backslash k$. Since buffers with infinite virtual queues are never empty, we do not keep record of jobs which are routed into them, hence we do not consider $\Phi_{kk'}(n)$ when $k' \in \mathcal{K}_\infty$.

For $k \in \mathcal{K}_0$ we let $Q_k(t) \geq 0$ denote the number of jobs of class k in the system at time t. For $k \in \mathcal{K}_\infty$ we indicate the relative state of the queue by counting the departures, and by relating them to some nominal input rate α_k. We denote this relative queue length by $R_k(t)$. For time allocations $T_k(t)$, $k \in \mathcal{K}$ we then have the dynamics of a MCQN with IVQs:

$$Z_k(t) = \begin{cases} Q_k(t) = Q_k(0) - S_k(T_k(t)) + \sum_{k' \in \mathcal{K} \backslash k} \Phi_{k'k}(S_{k'}(T_{k'}(t))) & k \in \mathcal{K}_0 \\ R_k(t) = R_k(0) - S_k(T_k(t)) + \alpha_k t & k \in \mathcal{K}_\infty. \end{cases} \tag{13}$$

Here $Q_k(0)$ are initial queue lengths, and $R_k(0)$ are some arbitrarily chosen initial values.

To summarize: The MCQN with IVQs is controlled by time allocations $T_k(t)$. The standard queues $Q_k(t)$, $k \in \mathcal{K}_0$ have input of jobs routed from other queues, and output $S_k(T_k(t))$. They are non-negative integer valued. The infinite virtual queues supply a stream of jobs into the system with their output $S_k(T_k(t))$ and sustain continuous input themselves at nominal rates α_k. Thus their relative level $R_k(t)$ is not restricted in sign and is not restricted to be integer.

The infinite virtual queues replace the input stream of standard MCQN. If $R_k(t)$ is stable in the sense that it is kept close to zero, then the departure process of the infinite virtual queue provides input to the rest of the system at the rate α_k. There are two new elements here which provide wider modeling capacity: (i) the input streams are controlled from within the network, by the allocation of processing time $T_k(t)$, and (ii) the infinite virtual queues share servers with other classes. In particular, if a server i serves some standard as well as some infinite virtual queues, then it will always have work to do, and so it can work at full utilization of $\rho_i = 1$, and yet, it is possible that the standard queues $Q_k(t)$ will be stable, and behave like queues in light traffic. An illustrative example of this is analyzed in Kopzon and Weiss (2007).

Let $\alpha = (\alpha_1, \ldots, \alpha_K)'$ be the vector of nominal input rates for $k \in \mathcal{K}_\infty$ and $\alpha_k = 0$ for $k \in \mathcal{K}_0$. The overall traffic intensity ρ for this exogenous input vector α is determined by the following linear program, which is a modified version of the *static planning problem LP* introduced in Harrison (2000):

$$\begin{aligned} \min \quad & \rho \\ \text{s.t.} \quad & \mathbf{R}u = \alpha, \\ & \mathbf{A}u \leq \mathbf{1}\rho, \\ & u \geq 0. \end{aligned} \tag{14}$$

Here $\mathbf{R}$ is the input-output matrix for the MCQN with IVQs and thus the first constraint in (14) reads:

$$\begin{aligned} &\mu_k u_k - \textstyle\sum_{k' \in \mathcal{K} \setminus k} P_{k'k} \mu_{k'} u_{k'} = 0, \quad && k \in \mathcal{K}_0 \\ &\mu_k u_k = \alpha_k, && k \in \mathcal{K}_\infty. \end{aligned} \tag{15}$$

A MCQN with IVQs is called *rate stable* if $\lim_{t\to\infty} \frac{1}{t} Z(t) = 0$. It can be shown that a necessary condition for rate stability of MCQN with IVQs, under any policy, is that the overall traffic intensity is $\rho \le 1$ (see Dai and Lin 2005).

MCQN with IVQs for each time interval

We return to our original MCQN over the finite time horizon, with its fluid solution. We associate a MCQN with IVQs to each of the M intervals of the fluid solution. During τ_m the associated MCQN with IVQs is defined by the partition $\mathcal{K}_0^m$ and $\mathcal{K}_\infty^m$. The nominal input rates of $k \in \mathcal{K}_\infty^m$ are set to $\alpha_k^m = \mu_k u_k^m$, which is the optimal outflow rate from the non-empty fluid buffer in the solution of (12). The corresponding matrix $\mathbf{R}^m$ is that of (7) changed to have $R_{k'k}^m = 0$ for $k' \in \mathcal{K}_\infty^m$, $k' \ne k$.

Table 1 describes the 4 MCQN with IVQs associated with the 4 intervals of the fluid solution of the example network. The flow rates of the fluid solution, u_k^m, $k \in \mathcal{K}$, provide a feasible solution of (14, 15), with $\rho^m = \max_{i\in\mathcal{I}} \rho_i^m \le 1$. Hence the necessary condition for rate stability is satisfied by each of the MCQN with IVQs. Clearly, in the fluid solution, $q_k(t) = 0$ for $k \in \mathcal{K}_0^m$, and so $Q_k(t)$ is the deviation from the fluid solution for buffers $k \in \mathcal{K}_0^m$. For the classes $k \in \mathcal{K}_\infty^m$ the fluid solution has outflow at rate $\mu_k u_k^m$, and so $R_k(t)$ measures the deviation from the fluid outflow rate for $k \in \mathcal{K}_\infty^m$. If we keep $Z^m(t)$ rate stable we will keep these deviations small. Furthermore, for $k \in \mathcal{K}_\infty^m$, rate stability of $Z^m(t)$ will yield that $\Phi_{k'k}(S_{k'}(T_{k'}(t))) - P_{k'k}\mu_{k'} u_{k'}^m$ will also be stable, but this is the deviation between the actual fluid inflow into buffer k, and the fluid inflow in the fluid solution. It follows that keeping $Z^m(t)$ rate stable implies good tracking of the fluid solution during τ_m (this is formally stated in Theorem 5.1).

4 Maximum pressure policies

Maximum pressure policies were introduced by in Tassiulas (1995). They were studied by Stolyar (2004) in the context of one pass systems. They were further studied in the more general framework of stochastic processing networks (introduced by Harrison (2000, 2001, 2002)) in Dai and Lin (2005, 2006). Maximum pressure policies are distinguished by two properties: Under sufficient conditions they are rate stable for systems with overall traffic intensity $\rho \le 1$, and in heavy traffic ($\rho \approx 1$), networks with complete resource pooling have optimal diffusion scale approximations. We now briefly describe maximum pressure policies and their adaptation to MCQN with IVQs. The results presented in this section are an adaptation from Dai and Lin (2005).

Denote by $a_k = \dot{T}_k(t)$ the level at which class k jobs are served. When $a_k = 1$ server $\sigma(k)$ serves class k fully. When $a_k = 0$, there is no processing of jobs from class k. Momentarily assume that we allow processor sharing, thus $0 \le a_k \le 1$. The column vector $a = (a_1, \dots, a_K)'$ is an allocation. Let the set of feasible allocations $\mathcal{A}$ be defined as the non-negative vectors a such that $\sum_{k\in C_i} a_k \le 1$ for all $i \in \mathcal{I}$ (these are the conditions 2). $\mathcal{A}$ is a non-empty (contains 0), bounded convex polytope, and has a finite number of extreme points. Denote the set of extreme points of $\mathcal{A}$ by $\mathcal{E}$. While $\mathcal{A}$ summarizes the set of allocations that satisfy the resource consumption constraints, it may be that due to emptiness of buffers $k \in \mathcal{K}_0$ some allocations in A are not available at certain times. Denote by $\mathcal{A}(t) \subseteq \mathcal{A}$ the set of available allocations at time t. Let $\mathcal{E}(t) = \mathcal{E} \cap \mathcal{A}(t)$ denote the set of the extreme allocations which are available at time t.

Denote the column vector that is the state of a MCQN at time t by $z = (Z_1(t), \dots, Z_K(t))'$. Define the total network pressure for an allocation a to be $z'\mathbf{R}a$ (where z' is a row vector, $\mathbf{R}$ is the input-output matrix and a is a column allocation vector). A service policy is said to be a maximum pressure policy if at each time t, the network chooses an allocation $a^* \in \operatorname{argmax}_{a\in\mathcal{E}(t)} z'\mathbf{R}a$.

By following all the steps in the proofs of the results of Dai and Lin (2005) one can see that they hold without any change also for MCQN with IVQs, as defined in Sect. 3. We therefore adopt Dai and Lin's main throughput optimality result to our context:

Theorem 4.1 *Let $Z(t)$ be a MCQN with IVQs. Assume that the processing and routing counts satisfy conditions (4)–(6). Let ρ be the overall traffic intensity of the network for nominal input α, and assume that $\rho \le 1$. Then under maximum pressure policy with no splitting of servers and with no preemptions* $\lim_{t\to\infty} \frac{1}{t} Z(t) = 0$.

We make just one comment about the proof: One needs to show that MCQN with IVQs satisfy EAA assumption. The steps of the proof are as in Dai and Lin (2005), but we need to make use of the fact that $\phi_{kk'}(\ell) = 0$ for $k' \in \mathcal{K}_\infty$.

In fact we need the asymptotic result for slightly different scaling. Let $Z(t)$ be a MCQN with IVQs. Let $S_k(t)$, $\Phi_{kk'}(n)$ be the processing and routing counts of $Z(t)$, and let α be its nominal inputs vector. $Z^N(t)$ is called an N scaling of $Z(t)$ with initial conditions $Z^N(0)$, if it has processing counts $S_k^N(t) = S_k(Nt)$, routing counts $\Phi_{kk'}(n)$, and nominal input $N\alpha$. We then have:

Corollary 4.2 *Let $Z(t)$ be a MCQN with IVQs. Assume that the processing and routing counts satisfy conditions (4)–(6). Let ρ be the overall traffic intensity of the network for nominal input α, and assume that $\rho \le 1$. Let $Z^N(t)$ be a sequence of N scalings of $Z(t)$, with initial states $Z^N(0)$ that satisfy $Z^N(0)/N \to 0$ as $N \to \infty$. Then under a maximum pressure policy with no splitting of servers and with no preemptions, $Z^N(t)/N \to 0$ uniformly for $0 < t < T$ for any $T > 0$.*

As observed in Dai and Lin (2005) and Tassiulas (1995), the maximization of the pressure $z'\mathbf{R}a$ over $Aa \le \mathbf{1}$, $a \ge 0$ separates into maximization of the pressure for each of the servers. This in turn is achieved by

$$k^* \in \arg\max_{k\in C_i\cup 0} \left\{ \mu_k(Z_k(t) - \sum_{k'\in\mathcal{K}_0\setminus k} P_{kk'} Z_{k'}(t)), 0 \right\},$$

where $k^* = 0$ corresponds to idling because all available queues have non-positive pressure.

5 Maximum pressure tracking of the optimal fluid solution

We come now to integration of the SCLP fluid solution, the associated MCQN with IVQs, and the maximum pressure policy, as described in Sects. 3–5 into a policy for the solution of the finite horizon MCQN control problem of Sect. 1.

Table 2 Calculation of pressure in a re-entrant line, for interval τ_m. Pressure at buffer k depends on type of queue and queue length of classes $k,\ k+1$. By convention $K+1 \in \mathcal{K}_\infty^m$

	Re-entrant implementation	
	$k \in \mathcal{K}_0^m$	$k \in \mathcal{K}_\infty^m$
$k+1 \in \mathcal{K}_0^m$	$\mu_k(Q_k(t) - Q_{k+1}(t))$	$Z_k^m(t_{m-1}) + \mu_k(\alpha_k(t-t_{m-1}) - (S_k(T_k(t)) - S_k(T_k(t_{m-1}))) - Q_{k+1}(t))$
$k+1 \in \mathcal{K}_\infty^m$	$\mu_k Q_k(t)$	$Z_k^m(t_{m-1}) + \mu_k(\alpha_k(t-t_{m-1}) - (S_k(T_k(t)) - S_k(T_k(t_{m-1})))$

Maximum Pressure Fluid Tracking Policy (MaxFTP)

Phase 1 (at time 0, centralized): Use $Q(0),\ w, \mathbf{R}, \mathbf{A}$ to solve the fluid network problem (8)–(11), and obtain the time intervals τ_m, the sets of empty and non-empty fluid buffers $\mathcal{K}_0^m$, $\mathcal{K}_\infty^m$ and the flow rates $\alpha_k^m = u_k^m \mu_k$ for $k \in \mathcal{K}_\infty^m$.

Phase 2 (on-line, decentralized): Track the fluid solution, for $t \in [0, T)$ by applying a maximum pressure policy in each of the intervals τ_m, $m = 1, \ldots, M$ as follows:

- Let $Q(t)$ be the queue lengths process for $t \in \tau_m$.
- Let $Z^m(t)$ for $t \in \tau_m$ be the state process of an associated MCQN with IVQs $\mathcal{K}_\infty^m$, and nominal inputs α^m, such that the processing times and routings of $Z^m(t)$ are identical to those of Q for $t \in \tau_m$. (Note that Z^m is as defined in (13) but with the time shifted by t_{m-1}).
- Set initial values for Z^m:

$$Z_k^m(t_{m-1}) = \begin{cases} Q_k(t_{m-1}) & k \in \mathcal{K}_0^m \\ -h_k^m \alpha_k^m \sqrt{|Q(0)|} & k \in \mathcal{K}_\infty^m, \end{cases} \tag{16}$$

  where $|Q(0)| = \sum_{k \in \mathcal{K}} Q_k(0)$, and $h_k^m = 0$ if $k \in \mathcal{K}_0^m$ or if $k \in \mathcal{K}_\infty^m$ and $q_k(t_{m-1}) > 0$, and $h_k^m = \min_{\{k' : P_{k'k} > 0\}} h_{k'}^m + 1$ for the remaining k.
- At every time t let $\mathcal{E}(t)$ be the set of extreme allocations available for $Q(t)$.
  Let $Z^m(t)'\mathbf{R^m} a$ be the pressure of allocation a, calculated for $Z^m(t)$.
  Use the maximum pressure allocation, $\max_{a \in \mathcal{E}(t)} Z^m(t)'\mathbf{R^m} a$, without processor splitting or preemptions.

For the example network (and in fact for any re-entrant line), implementation of the maximum pressure fluid tracking policy is described in Table 2. When server i is available at time $t \in \tau_m$, it calculates the pressure of all buffers $k \in C_i$ which have $Q_k(t) > 0$ according to Table 2 and starts to process a job from the queue with the highest pressure if the resulting pressure is positive. Otherwise, it idles.

Our main result is:

Theorem 5.1 *Let $Q(t)$ be the queue length process of a finite horizon MCQN. Assume that the processing and routing counts satisfy conditions (4)–(6). Let $Q^N(t)$ be N scalings of $Q(t)$, with $Q^N(0) = NQ(0)$. Let $q(t)$ be the optimal fluid solution, and let V_f be its objective value.*

(i) *Let V^N denote the objective value of $Q^N(t)$ for any general policy. Then*

$$\liminf_{N \to \infty} \frac{1}{N} V^N \geq V_f$$

(ii) *Under MaxFTP policy*:

$$\lim_{N\to\infty} \frac{1}{N} Q^N(t) = q(t) \textit{ uniformly on } 0 \le t \le T$$

and $\lim_{N\to\infty} \frac{1}{N} V^N = V_f$.

Proof (i) Consider some general policy and let $\bar{V} = \liminf_{N\to\infty} \frac{1}{N} V^N$. Let r be a subsequence for which $\bar{V} = \lim_{r\to\infty} \frac{1}{r} V^r$. By the argument of Dai and Lin (2005), Section A.2 we can find a subsequence r' of the r such that $\lim_{r'\to\infty}(\frac{1}{r'} Q^{r'}(t), T^{r'}(t), \frac{1}{r'} V^{r'}) = (\bar{Q}(t), \bar{T}(t), \bar{V})$ uniformly on $[0, T]$, where $\bar{Q}(t), \bar{T}(t)$ are Lipschitz continuous fluid limits, and in particular $\bar{T}$ has derivative $\dot{\bar{T}}$ almost everywhere. One can see that $\bar{Q}(t), \dot{\bar{T}}(t), \bar{V}$ must be a feasible solution to (8–11). Hence, $\bar{V} \ge V_f$. (ii) We now consider the sequence $(Q^N(t), Z^N(t), T^N(t), V^N)$ under the MaxFTP policy, where Z^N are the processes of deviations from the fluid solution. As above we have that for a subsequence r $\lim_{r\to\infty}(\frac{1}{r} Q^r(t), \frac{1}{r} Z^r(t), T^r(t), \frac{1}{r} V^r) = (\bar{Q}(t), \bar{Z}(t), \bar{T}(t), \bar{V})$, uniformly on $[0, T]$. Our goal is to show that $\bar{Q}(t), \dot{\bar{T}}(t)$ equal the optimal solution of (8)–(11). We do this by induction on the intervals τ_m, $m = 1, \dots, M$. There is nothing to show for $t = 0$. Assume then that $\bar{Q}(t_{m-1}) = q(t_{m-1})$. Assume first that $q_k(t_{m-1}) > 0$ for all $k \in \mathcal{K}_\infty^m$. In that case we have that the initial values $Z_k^{m,N}(t_{m-1})$ as defined by (16) are equal to 0. We define:

$$\bar{\bar{t}} = \min\big[t_m, \inf\{t : t_{m-1} < t < t_m,\ \bar{Q}_k(t) = 0 \text{ for some } k \in \mathcal{K}_\infty^m\}\big].$$

By continuity of $\bar{Q}$, $\bar{\bar{t}} > t_{m-1}$. Let $t_{m-1} < \bar{t} < \bar{\bar{t}}$. Then for $r > r_0$ we will have $Q_k^r(t) > 0$ for $t_{m-1} < t < \bar{t}$, $k \in \mathcal{K}_\infty^m$. Hence for $r > r_0$ the MaxFTP policy will act on $Z^{m,r}(t)$ identical to max pressure policy. Consider then the sequence of scalings $Z^{m,r}$ under maximum pressure policy. The optimal solution of SCLP in the mth interval satisfies:

$$R^m u^m = \begin{bmatrix} R_{\mathcal{K}_0^m,\mathcal{K}_0^m} & R_{\mathcal{K}_0^m,\mathcal{K}_\infty^m} \\ 0 & \operatorname{diag}(\mu_{\mathcal{K}_\infty^m}) \end{bmatrix} u^m = \alpha^m,$$
$$A u^m \le 1, \quad u^m \ge 0,$$

hence, comparing with (14) we see that $\rho \le 1$ for the network $Z^{m,1}$. Hence, by Corollary 4.2, $\lim_{r\to\infty} Z^{m,r}(t) = 0$ on $0 < t < \bar{t}$, and because $\bar{t} < \bar{\bar{t}}$ was arbitrary the same holds for $0 < t < \bar{\bar{t}}$. We then have for all $0 < t < \bar{\bar{t}}$:

$$\bar{Q}_k(t) = \bar{Z}_k(t) = 0, \quad k \in \mathcal{K}_0^m,$$

$$\bar{Z}_k(t) = \alpha_k(t - t_{m-1}) - \mu_k(\bar{T}_k(t) - \bar{T}_k(t_{m-1})) = 0 \quad \text{implies } \dot{\bar{T}}_k(t) = \alpha_k/\mu_k, \quad k \in \mathcal{K}_\infty^m$$

so $\bar{Q}_k(t) = q_k(t)$, $k \in \mathcal{K}_0^m$, and $\dot{\bar{T}}_k(t) = u_k^m$, $k \in \mathcal{K}_\infty^m$. We next obtain for $\mathcal{K}_0^m$:

$$\bar{Q}_{\mathcal{K}_0^m}(t) = 0 - R_{\mathcal{K}_0^m,\mathcal{K}_0^m}\big[\bar{T}_{\mathcal{K}_0}(t) - \bar{T}_{\mathcal{K}_0}(t_{m-1})\big] - R_{\mathcal{K}_0^m,\mathcal{K}_\infty^m}[\bar{T}_{\mathcal{K}_\infty}(t) - \bar{T}_{\mathcal{K}_\infty}(t_{m-1})] = 0$$
$$\text{implies } \dot{\bar{T}}_{\mathcal{K}_0}(t) = -(R_{\mathcal{K}_0^m,\mathcal{K}_0^m})^{-1} R_{\mathcal{K}_0^m,\mathcal{K}_\infty^m} \operatorname{diag}(\alpha_k/\mu_k)_{k\in\mathcal{K}_\infty^m}$$

and so $\dot{\bar{T}}_k(t) = u_k^m$, $k \in \mathcal{K}_0^m$. Finally we substitute these values into

$$\bar{Q}_{\mathcal{K}_\infty^m}(t) = \bar{Q}_{\mathcal{K}_\infty^m}(t_{m-1}) - R_{\mathcal{K}_\infty^m,\mathcal{K}_0^m}[\bar{T}_{\mathcal{K}_0}(t) - \bar{T}_{\mathcal{K}_0}(t_{m-1})] - R_{\mathcal{K}_\infty^m,\mathcal{K}_\infty^m}[\bar{T}_{\mathcal{K}_\infty}(t) - \bar{T}_{\mathcal{K}_\infty}(t_{m-1})]$$

to get $\bar{Q}_k(t) = q_k(t)$, $k \in \mathcal{K}_\infty^m$. Since $\bar{Q}_k(\bar{\bar{t}}) = q_k(\bar{\bar{t}})$ we must have $\bar{\bar{t}} = t_m$.

Consider now the case that for some of the $k \in K_\infty^m$, $q_k(t_{m-1}) = 0$. We sketch the proof in this case. The MaxFTP policy will start off at time t_{m-1} with negative pressure in these buffers. As a result there will be no processing out of these buffers for a duration of $h_k^m \sqrt{N|Q(0)|}$. All the buffers with $h_k^m = 0$ will be processed according to maximum pressure. It can then be seen that the buffers with $h_k^m > 0$ will fill up with a quantity of jobs of the order of magnitude $\sqrt{N}$ by the time they reach positive pressure. As a result we get that $\frac{d}{dt}\bar{Q}(t_{m-1}) > 0$, and the proof proceeds as before.

We have shown that each fluid limit must equal the optimal fluid solution, and we also know that every sequence of scalings has a subsequence which has a fluid limit. This proves that $\lim_{N\to\infty} \frac{1}{N} Q^N(t) = q(t)$, uniformly on $[0, T]$. In particular this implies that $\lim_{N\to\infty} \frac{1}{N} V^N = V_f$. □

6 Simulation results

We simulated the example network for N scalings of up to $N = 10^6$. We generated the processing times for each of the classes as pseudo random i.i.d. exponential random variables. For each single replicate we used $3 \times 4 = 12$ generator seeds which gave us long sequences of processing times for each of the three buffers in each of the four time intervals of the fluid solution. We then created N scalings of each replicate as defined in Sect. 4.

Figure 5 illustrates the simulation results for 4 such replicates (the four columns) and N scalings of $N = \{1,\ 10,\ 100\}$. In the figure we show for each of the 12 simulated queueing processes the values of $Q^N(t)$ plotted against the optimal fluid solution for that N (this is the fluid solution $q(t)$ multiplied by N for each t). The illustrations show that even for $N = 10$, the approximation is quite good.

Figure 6 examines the asymptotics of our policy in more detail and on a mass scale. Here we have plotted unscaled deviations, namely $Q_k^N(t) - Nq_k(t)$, at the time points $t = 4,\ 8,\ 24,\ 40$, which are the breakpoint times of the optimal fluid solution. This is performed for each queue $k = 1, 2, 3$. For 100 replicates we have performed the following N scalings: $N = \{1,\ 5\times 10^3,\ 10^4,\ 5\times 10^4,\ 10^5,\ 2\times 10^5,\ 4\times 10^5,\ 6\times 10^5,\ 8\times 10^5,\ 10^6\}$. The N

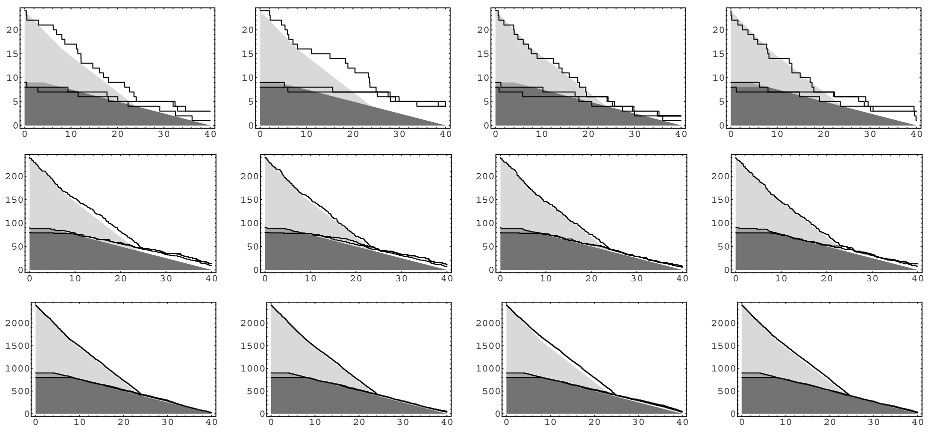

Fig. 5 Example realizations: 4 replicates (columns) with scalings $N = \{1, 10, 100\}$ for each replicate

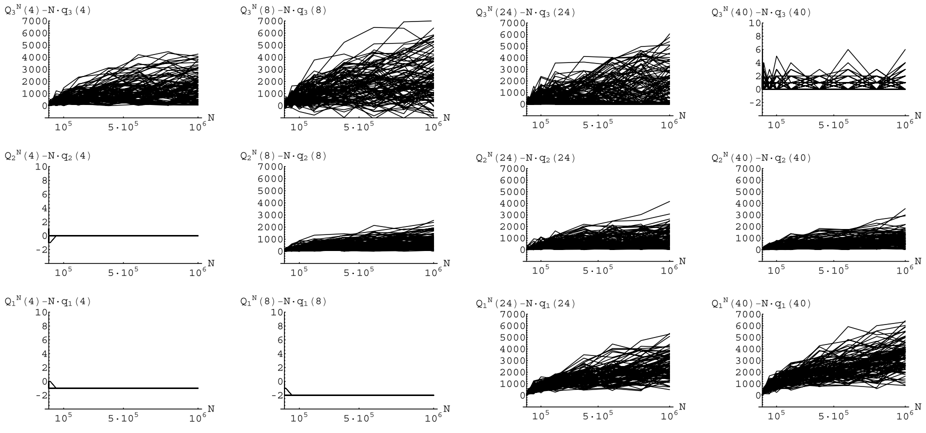

Fig. 6 Empirical asymptotics: 100 replicates with N scalings up to $N=10^6$

scalings of each replicate are plotted as a continuous line. This allows us to appreciate how the deviations evolve along a single sample path, as the scaling increases.

As may be expected, the deviations all appear to be of the order of magnitude of $\sqrt{N}$. An exception is for the N scalings for $k=3$ and $t=40$, where the deviations look like queue lengths of a queue in light traffic (indeed the utilization of server 1 in the last interval is $1/2$).

References

Adan, I. J. B. F., & Weiss, G. (2005). A two node Jackson network with infinite supply of work. *Probability in Engineering and Informational Sciences*, *19*, 191–212.

Adan, I. J. B. F., & Weiss, G. (2006). Analysis of a simple Markovian re-entrant line with infinite supply of work under the LBFS policy. *Queueing Systems Theory and Applications*, *54*, 169–183.

Anderson, E. J. (1981). A new continuous model for job-shop scheduling. *International Journal Systems Science*, *12*, 1469–1475.

Anderson, E. J., & Nash, P. (1987). *Linear programming in infinite dimensional spaces*. Chichester: Wiley-Interscience.

Avram, F., Bertsimas, D., & Ricard, M. (1995). Fluid models of sequencing problems in open queueing networks: an optimal control approach. In F. P. Kelly & R. Williams (Eds.), *IMA volumes in mathematics and its applications: Vol. 71. Stochastic networks* (pp. 199–234). New York: Springer.

Bellman, R. (1953). Bottleneck problems and dynamic programming. *Proceedings National Academy of Science*, *39*, 947–951.

Bramson, M. (1998). State space collapse with application to heavy traffic limits for multiclass queueing networks. *Queueing Systems Theory and Applications*, *30*, 89–148.

Chen, H., & Yao, D. (1993). Dynamic scheduling of a multi class fluid network. *Operations Research*, *41*, 1104–1115.

Chen, M., Pandit, C., & Meyn, S. P. (2003). In search of sensitivity in network optimization. *Queueing Systems Theory and Applications*, *44*, 313–363.

Chen, R. R., & Meyn, S. P. (1999). Value iteration and optimization of multiclass queueing networks. *Queueing Systems Theory and Applications*, *32*, 65–97.

Connors, D., Feigin, G., & Yao, D. (1994). Scheduling semiconductor lines using a fluid network model. *IEEE Transactions on Robotics and Automation*, *10*, 88–98.

Dai, J. G. (1995). On positive Harris recurrence of multi-class queueing networks: a unified approach via fluid limit models. *Annals of Applied Probability*, *5*, 49–77.

Dai, J. G., & Lin, W. (2005). Maximum pressure policies in stochastic processing networks. *Operations Research*, *53*, 197–218.

Dai, J. G., & Lin, W. (2006). Asymptotic optimality of maximum pressure policies in stochastic processing networks. *Annals of Applied Probability* (submitted).

Fleischer, L., & Sethuraman, J. (2003). Approximately optimal control of fluid networks. IBM Research Report

Goemans, M. X., & Williamson, D. P. (1996). The primal dual method for approximation algorithms and its application to network design problems. In D. Hochbaum (Ed.) *Approximation algorithms* (pp. 144–191).

Harrison, J. M. (1988). Brownian models of queueing networks with heterogeneous customer populations. In W. Fleming & P. L. Lions (Eds.), *Proceedings of the IMA workshop on stochastic differential systems*. Berlin: Springer.

Harrison, J. M. (1996). The BIGSTEP approach to flow management in stochastic processing networks. In F. P. Kelly, S. Zachary, & I. Ziedins (Eds.), *Royal statistical society lecture note series: Vol. 4. Stochastic networks: theory and applications* (pp. 147–186). Oxford: Oxford University Press.

Harrison, J. M. (2000). Brownian models of open processing networks: canonical representation of workload. *Annals of Applied Probability*, *10*, 75–103.

Harrison, J. M. (2001). A broader view of Brownian networks. *Annals of Applied Probability*, *13*, 1119–1150.

Harrison, J. M. (2002). Stochastic networks and activity analysis. In Y. Suhov (Ed.), *Analytic methods in applied probability, In memory of Fridrik Karpelevich*. Providence: American Mathematical Society.

Harrison, J. M., & Van Mieghem, J. (1997). Dynamic control of Brownian networks: State space collapse and equivalent workload formulations. *Annals of Applied Probability*, *7*, 747–771.

Kelly, F. P., & Laws, C. N. (1993). Dynamic routing in open queueing networks: Brownian models, cut constraints and resource pooling. *Queueing Systems Theory and Applications*, *13*, 47–86.

Kopzon, A., & Weiss, G. (2002). A push pull queueing system. *Operations Research Letters*, *30*, 351–359.

Kopzon, A., & Weiss, G. (2007). A push pull system with infinite supply of work. Preprint.

Kushner, H. J. (2001). *Heavy traffic analysis of controlled queueing and communication networks*. Berlin: Springer.

Lawler, E. L., Lenstra, J. K., Rinnooy Kan, A. H. G., & Shmoys, D. B. (1993). Sequencing and scheduling, algorithms and complexity. In S. C. Graves & A. H. G. Rinnooy (Eds.), *Handbooks in operations research and management science: Vol. 4. Logistics of production and inventory*. Amsterdam: North Holland.

Maglaras, C. (1999). Dynamic scheduling in multiclass queueing networks: Stability under discrete-review policies. *Queueing Systems Theory and Applications*, *31*, 171–206.

Maglaras, C. (2000). Discrete-review policies for scheduling stochastic networks: Trajectory tracking and fluid-scale asymptotic optimality. *Annals of Applied Probability*, *10*, 897–929.

Meyn, S. P. (2001). Sequencing and routing in multiclass queueing networks. Part I: Feedback regulation. *SIAM Journal Control and Optimization*, *40*, 741–776. IEEE international symposium on information theory. Sorrento, Italy, June 25 – June 30 2000.

Meyn, S. P. (2003). Sequencing and routing in multiclass queueing networks. Part II: Workload relaxations. *SIAM Journal Control and Optimization*, *42*, 178–217.

Shah, S., & Nahrstedt, K. (2002). Predictive location-based QoS routing in mobile ad hoc networks. In *Proceedings of IEEE international conference on communications*.

Pullan, M. C. (1993). An algorithm for a class of continuous linear programs. *SIAM Journal Control and Optimization*, *31*, 1558–1577.

Stolyar, A. L. (2004). MaxWeight scheduling in a generalized switch: State space collapse and equivalent workload minimization under complete resource pooling. *Annals of Applied Probability*, *14*(1), 1–53.

Tassiulas, L. (1995). Adaptive back-pressure congestion control based on local information. *IEEE Transactions on Automatic Control*, *40*, 236 Ð-250.

Wein, L. M. (1992). Scheduling networks of queues: Heavy traffic analysis of a multistation network with controllable inputs. *Operations Research*, *40*, S312–S334.

Weiss, G. (1999). Scheduling and control of manufacturing systems—a fluid approach. In *Proceedings of the 37 Allerton conference*, Monticello, Illinois 21–24, September 1999, (pp. 577–586).

Weiss, G. (2005). Jackson networks with unlimited supply of work. *Journal of Applied Probability*, *42*, 879–882.

Weiss, G. (2008). A simplex based algorithm to solve separated continuous linear programs. *Mathematical Programming*, *115*, 151–198.

Williams, R. J. (1998). Diffusion approximations for open multiclass queueing networks: sufficient conditions involving state space collapse. *Queueing Systems Theory and Applications*, *30*, 27–88.