Approximate Solution of Multiclass Queueing Networks with Region Constraints

Jonatha Anselmi*, Giuliano Casale and Paolo Cremonesi
Politecnico di Milano, DEI
via Ponzio, 34/5, I-20133 Milan, Italy
{jonatha.anselmi, giuliano.casale, paolo.cremonesi}@polimi.it

***Abstract*—Among existing modeling techniques, queueing networks with "finite capacity regions" have largely proven to be effective in characterizing push-back effects and simultaneous resource possession in which a request holds more resources simultaneously. Queueing network models with finite capacity regions impose upper bounds on the number of jobs that can simultaneously reside in a set of service centers. For this reason they can be used to model application constraints. However, since they do not satisfy product-form assumptions, they are difficult to treat. In this paper we propose a novel approximate method for closed multiclass queueing networks containing finite capacity regions and shared constraints. Our approach is based on the Norton's theorem for queueing networks where a region is replaced by a single Flow Equivalent Service Center (FESC). We propose a population-mix driven definition of FESCs service rates which provides increased accuracy with respect to existing methods. We solve the resulting non-product-form network with a new approximate variant of the convolution algorithm proposed in the paper. A comparison with simulation shows that the algorithm typically has a 4% approximation error.**

***Index Terms*—Queueing networks, finite capacity regions, shared constraints, simultaneous resource possession, approximate solutions, flow equivalent service center.**

I. Introduction

Application constraints have different motivations and different behaviors. For instance, in typical client-server systems, a client task must wait in a blocked state if the pool of server tasks is busy responding to other clients: a saturated pool of server tasks will then slow down all of its clients. As another example, consider a web–based multi–level application having a pool of threads to process HTTP requests. The pool size is chosen in order to have a compromise between response time and number of rejected requests. When a new request arrives and finds no free thread, it waits in the web server queue. The web–based application can be described with a set of resources modeling the architecture (e.g., web, application and database servers) with a finite capacity region bounded by the size of the HTTP threads pool.

Very often application constraints are created explicitly, as it happens with admission control mechanism used to control the quality of the performance in client/server architectures. The goal of admission control algorithms is to admit as many requests as possible while satisfying some performance requirement. However, these two components of the goal are conflicting in nature. Several admission control schemes have been proposed in the literature (e.g., [6], [19]). Most of them operate by providing upper bounds to the number of requests concurrently serviced by one or more servers. It is difficult to model such effects using product-form queueing networks (see, e.g, [3]), which are more fit to predict the system performance by considering the hardware resources only [12], [11], [18], since they do not take into account effect of software contentions, and hence overestimate throughputs. Therefore, specialized models are required to accurately estimate performance metrics in presence of application constraints.

A first approach proposed in previous work is based on Layered Queueing Network (LQN) that consider software servers as queueing nodes [10]; the Method of Layers (MOL) is frequently used to solve LQN models [14]. Software tasks are organized into layers. A task in the first layer may only send synchronous calls to tasks belonging to adjacent layer. This queueing model is analyzed by decomposition, considering two adjacent layers of tasks at a time. A separate queueing model is developed to represent the contention in the CPU and the I/O devices. The two models are then combined using an iterative algorithm that it is iterated until the final performance metrics converge. A challenge with LQN models is their parametrization requires detailed information about the system, e.g., getting the correct values of interconnection among software servers/clients, and visit counts in the software layers is difficult in practice (e.g., the internal structure of commercial applications is often unknown). Hence, in presence of such difficulties, there is little advantage in adopting a sophisticate modeling technique as the LQN, and queueing networks with finite capacity regions [13] often suffice.

In this paper we propose a novel approximate method for closed multiclass queueing networks with finite capacity regions and shared constraints. We employ for the first time the *multiclass* Norton's theorem for queueing networks, where each region is approximated by a suitable Flow Equivalent Service Center (FESC). We define FESCs service rates drawing concepts from the asymptotic theory of [1] for multiclass queueing networks. Finally, we solve the resulting simplified network using an approximation derived from the convolution algorithm. A comparison with simulation shows that our algorithm significantly improves accuracy with respect to existing methods.

The structure of the paper is as follows. Section II introduces queueing networks with finite capacity regions and reviews existing solution algorithms. In Section III we explain the *population-mix* concept and the new ap-

proximation. In Section IV we discuss characteristics of product-form service rates and propose the variant of the convolution algorithm which is able to handle networks with non-product-form service rates. Experimental results and accuracy evaluation with respect to existing models are given in Section V. Finally, Section VI draws conclusions and outlines future work.

II. Finite Capacity Regions

We consider queueing network models composed by centers that may belong to *finite capacity regions* (see [9] for an introduction). Each region applies one or more policies, called *finite capacity constraints*, which define upper bounds on the number of jobs that can simultaneously reside in its service centers. Customers are either in a *thinking* state, i.e., waiting at the terminals, or *ready*, i.e., visiting queues. Moreover, ready customers can be either in a buffer queue *waiting* to enter into a finite capacity region according to a FCFS access rule, or *active*, i.e., circulating inside the region and competing for the resources. In general, the distinction between active and waiting customers makes the analysis of finite capacity models much harder than in the product-form case. This is true in particular when the population belongs to multiple classes where two kinds of finite capacity constraints are possible [8]: *i*) a *dedicated constraint* bounds the number of customers in a region for a specific class, *ii*) a *shared constraint* limits the number of customers without class distinctions. Note that both types of constraints can be applied simultaneously to a region. Almost all solution techniques known in literature are based on simulation or approximate analytical methods. We now introduce notation and then briefly review previous work on approximations.

A. Notation

Consider a multiclass closed queueing network model with R customer classes, K service centers and M finite capacity regions. If not otherwise stated, index r will range from 1 to R, k from 1 to K, and m from 1 to M. We denote class r population by N_r, the population vector by $\vec{N} = (N_1, N_2, \ldots, N_R)$, and the total number of customer by $N = \sum_r N_r$. We denote by $\vec{n}$ any population vector such that $\vec{0} < \vec{n} \leq \vec{N}$, and $n = \sum_r n_r$. Each region m contains two or more service centers, and has a shared capacity constraint B_m. In other words, the total number of ready jobs in m never exceeds B_m. We also denote by

- $V_{k,r}$, the mean number of class-r visits at center k
- $S_{k,r}$, the mean class-r service time at center k
- $\mu_{k,r} \equiv \mu_{k,r}(\vec{n})$, the mean class-$r$ service rate when the current population at center k is $\vec{n}$
- Z_r, the mean class-r think time
- $\vec{e}_r$, the unit vector in direction r.

The performance measures of interest are per-class throughputs and the aggregate throughput, respectively denoted by $X_r(\vec{n})$ and $X(\vec{n}) = \sum_r X_r(\vec{n})$.

B. Review of Approximation Techniques

Approximate analytical methods for finite capacity models are mostly based on the decomposition principle and on hierarchical modeling techniques (e.g., [9], [15]) in which a subnetwork is replaced by a load-dependent station called *Flow Equivalent Service Center* (FESC). For product-form models, the FESC is equivalent to the original subnetwork, and can be used without introducing any degree of approximation in the model solution [7]. This is obtained by setting the service rates of the FESC equal to the throughputs of the subnetwork. Despite this approach is not exact outside the product-form case, for finite capacity models it can be an accurate approximation. The critical difficulty, which has the greatest impact on the accuracy of the method, is the definition of FESC service rates.

In [4], Brandwajn and McCormack propose a solution algorithm that handles only shared constraints and solves network with a single finite capacity region. It is based on the following *homogeneity assumptions*:

1. for each feasible network state, the class-r throughput of a region $\mathcal{R}$ does not depend on the current population of the classes $s \neq r$ in $\mathcal{R}$, but instead depends on their *average* population in $\mathcal{R}$ over all states;
2. if a center does not contain class r jobs, then an arriving class r customer immediately receives service regardless of the jobs of class $s \neq r$ in that station.

The key aspect of the previous assumptions is that the computational cost of solving multiclass models can be dramatically reduced. On the other hand, their impact on accuracy is significant.

A different approximate technique, proposed by Sauer in [15], defines FESC service rates that directly account for the finite capacity constraints. However, the way in which service rates are defined does not satisfy product-form assumptions, and the solution is computed analyzing the underlying Markov chain global balance equations. Hence, the related computational complexity makes unfeasible the analysis of networks with multiple finite capacity regions, since the number of global balance equations is beyond the capabilities of standard linear system solvers even for moderate population sizes.

In [8], Krzesinski and Teunissen propose a solution algorithm based on the homogeneity assumptions which handles both shared and dedicated constraints and multiple finite capacity regions. Hence, it extends the technique shown in [9]. The algorithm groups together stations belonging to the same region with an intensive use of the multiclass Linearizer technique (see, e.g., [3]), and iteratively estimates the average number of jobs inside the regions. The algorithm stops when performance metrics converge.

III. The Proposed Algorithm

In this section we propose a new approximate method for multiclass closed queueing network models with finite capacity regions. We assume that if finite capacity constraints are removed, the model satisfies product-form assumptions. We also assume that queues have fixed service rates and that the model includes only shared constraints.

In Section A we motivate our approach showing limitations of homogeneity assumptions based solution algorithms. In Section B we present the approximate algorithm, and in Section C we discuss FESC definition.

A. Preliminary Results

We propose a novel approach for solving models with finite capacity regions. Iterative solutions based on the homogeneity assumptions step through the solution of single-class load-dependent models in order to determine the total number of jobs inside each region. The information loss due to the inability of considering simultaneously all classes is compensated by an iterative refinement of the current solution until all performance measures converge.

A.1 Bottleneck Switch

As observed in [1], in closed multiclass networks jobs tend to accumulate in different portions of the network according to the proportions of jobs classes and the number of potential bottleneck stations. It has been shown that *the bottleneck queue can migrate across different stations according to the population of the different classes.* For instance, consider a product-form model, and assume to evaluate queue-lengths as N_1 increases while keeping a fixed $N = N_1 + N_2 = 100$. We see in Figure 1 the result of such analysis on a model with all visits equal to 1 and service times $S_{1,1} = 10, S_{1,2} = 3, S_{2,1} = 7, S_{2,2} = 10$ where $Q1$ and $Q2$ denote the average number of jobs respectively in queue 1 and queue 2. What we see is that jobs can quickly migrate as a result of a change of the ratio between N_1 and N_2, and hence the ratio $\beta_1(\vec{N}) = N_1/N$, called *population-mix*, strongly affects performance of a queueing network. This observation generalizes to models with an increased number of stations and classes, where we need in general to consider the *population-mix* vector $\vec{\beta}(\vec{N}) = (\beta_1(\vec{N}), \ldots, \beta_r(\vec{N}), \ldots, \beta_R(\vec{N}))$, in which $\beta_r(\vec{N}) = N_r/N$. In general, depending on the population-mix vector $\vec{\beta}(\vec{N})$, also a finite capacity region can assume a significantly different behavior. For instance, consider the previous two-station and two-class model inside a finite capacity region of size $B_1 = 50$. In Figure 2, we evaluate queue lengths as N_1 increases while keeping a fixed $N = N_1 + N_2 = 100$. Note that at any time 50 jobs are in the waiting buffer. The evaluation has been performed by means of simulation using the JMT simulation engine [2] and selecting 99% level confidence intervals. What we see in Figure 2 is that jobs can quickly migrate from one queue to the other even in presence of a finite capacity region.

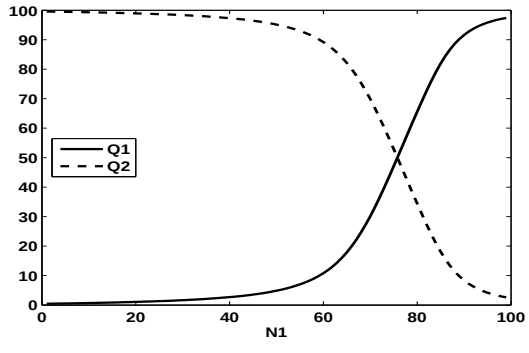

Figure 1. Job migration in a two-stations product-form model as a function of the class-1 population N_1 while keeping a constant $N = N_1 + N_2 = 100$.

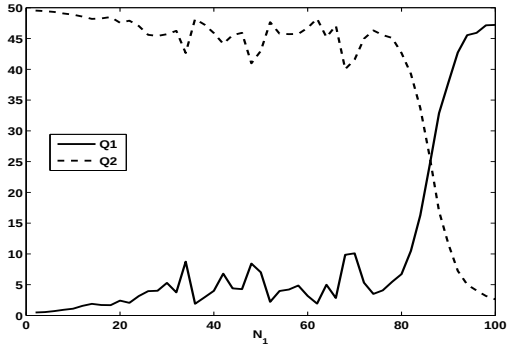

Figure 2. Job migration obtained by simulation in a two-stations model with a finite capacity region of size $B_1 = 50$.

A.2 Non-Uniqueness of Active Jobs Mixes

A second useful observation for the analysis of such models derives from the fact that the exact mixes of per-class *active* jobs are in general not known. In other words, the class-r mix of population $\vec{N}$ active jobs visiting region m, denoted by $\beta_{m,r}(\vec{N})$, is not known. This further complicates the analysis of such models since also for a fixed $\beta_r(\vec{N})$ the region behavior, as observed in Section A.1, depends on the actual mixes $\beta_{m,r}(\vec{N})$.

In the following we assume that each region m is visited by at least B_m jobs. Let $\Omega_m(\vec{N})$ be the set of mixes related to all the possible ways in which we can arrange B_m active jobs from population $\vec{N}$ inside region m, i.e. excluding the waiting buffer. Formally, $\Omega_m(\vec{N})$ can be expressed as

$$\Omega_m(\vec{N}) = \left\{ \frac{\vec{q}}{B_m} \mid \sum_r q_r = B_m \ \wedge \ 0 \leq q_r \leq N_r \right\}. \quad (1)$$

Clearly, $\beta_{m,r}(\vec{N}) \in \Omega_m(\vec{N})$ and in general we have $|\Omega_m(\vec{N})| >> 1$. This means that for a given population $\vec{N}$ the mix of per-class active jobs inside a region is not unique as well as the behavior of the region. This observation makes more difficult the analysis because in order to solve the network through the definition of a suitable FESC we need a unique mix modeling the region behavior. For a given $\vec{N}$, a reasonable unique mix for region m can be obtained assuming that the average behavior of m is given by the average mix $\beta^*_{m,r}(\vec{N})$ of $\Omega_m(\vec{N})$, i.e.

$$\beta^*_{m,r}(\vec{N}) = \sum_{\vec{\beta}_m \in \Omega(\vec{N})} \frac{\beta_{m,r}}{|\Omega_m(\vec{N})|}. \quad (2)$$

Keeping fixed B_m, as $|\Omega_m(\vec{N})|$ increases, the average behavior of m becomes more difficult to approximate because

the region can assume more different behaviors. Note that keeping fixed B_m, $|\Omega_m(\vec{N})|$ can be increased only increasing populations N_r such that $N_r < B_m$, i.e. $|\Omega_m(\vec{N})|$ is maximum when each population N_r is greater than or equal to B_m since no further mixes can be added to $\Omega_m(\vec{N})$. Increasing such populations we add mixes to $\Omega_m(\vec{N})$ and expand the space of possible mixes. Hence, the larger the cardinality of $\Omega_m(\vec{N})$, the higher the probability to observe bottleneck switches, i.e. radical changes in the region. On the other hand, for small values of $|\Omega_m(\vec{N})|$ the behavior of m is similar for each mix in $\Omega_m(\vec{N})$ and the average behavior of m can be better approximated. For example, considering the case of a region with capacity $B_1 = 5$ and two classes of jobs, we have $\Omega_1(3,3) = \{(3,2),(2,3)\}$ and $\Omega_1(6,6) = \{(5,0),(4,1),(3,2),(2,3),(1,4),(0,5)\}$. While in the former case $2/5 \leq \beta_{1,r} \leq 3/5$, in the latter we have $0 \leq \beta_{1,r} \leq 1$ and the average behavior of the region becomes more difficult to approximate.

Let us consider again the two-station and two-class model of Section A.1 inside a finite capacity region of size $B_1 = 50$ with $N = N_1 + N_2 = 100$ jobs. For each population $\vec{N} = (N_1, N_2)$, in this specific case the cardinality of $\Omega(\vec{N})$ is $\min\left\{\beta_1(\vec{N}), \beta_2(\vec{N})\right\} N + 1$. In Figure 2 we notice that it is difficult to understand queue lengths behavior especially when the population-mix vector is balanced, i.e. $\beta_1(\vec{N}) = 1 - \beta_2(\vec{N}) \simeq 1/2$. Assuming $N_1 \leq N_2$, we have $\vec{\beta}_1(\vec{N}) \leq (N_1/B_1, 1)$ where the inequality holds component-wise. In fact, for $\beta_{1,1}(\vec{N}) = 1/2$, we expand the space of possible mixes since $\vec{\beta}_1(\vec{N}) \leq (1,1)$ and the probability to observe bottleneck switches in the region is higher. This explains the increased variance on queue lengths with respect to unbalanced $\beta_r(\vec{N})$ which represent the cases in which $|\Omega(\vec{N})|$ is small and the average behavior of active jobs can be better determined.

In existing solution algorithms (see [4], [8], [15]), the lack of a numerical validation performed with respect to models with balanced population mixes $\vec{\beta}$ or significantly high values of $|\Omega(\vec{N})|$ further motivates our approach.

B. The Algorithm

The use of single-class algorithms under homogeneity assumptions simply ignores previous observations. In order to overcome these limitations, we propose to explicitly take into account the role of $\vec{\beta}$ vector and Ω_m set when defining FESCs service rates.

The general structure of our solution technique is given in Algorithm 1. In steps 1-4, we replace each region with a suitable FESC: since the way in which we define FESCs service rates is the main innovation of our approach, a complete description is given below. In step 5, we build a non-product-form model which is solved (step 6) by an approximate variant of the convolution algorithm proposed later in Section IV.

Algorithm 1 General scheme of the approximate algorithm.

1: **for** each region m **do**
2: Create a FESC analyzing m in isolation:
3: a) Define service rates $\widehat{\mu}_{m,r}(\vec{n})$ for populations $\vec{n}$ such that $n \leq B_m$
b) Approximate service rates $\widehat{\mu}_{m,r}(\vec{n})$ for populations $\vec{n}$ such that $n > B_m$
4: **end for**
5: Replace all finite capacity regions with the FESCs
6: Solve the resulting unconstrained network with an approximation of the load-dependent convolution algorithm.

C. FESC Construction

The main issue we address in this section is the definition of FESCs which replace regions and take into account finite capacity effects. To model such effects in single class networks, good results have been obtained using service rate saturation (see, e.g., [9]) which states that throughput cannot increase when the capacity constraint is active. In multiclass environments the approximation is more difficult to obtain. Since jobs of different classes have different service demands, the departure rate from a region depends on the mix of requests in m. As shown in Section A, the population-mix vector $\vec{\beta}(\vec{n}) = (\beta_1(\vec{n}), \ldots, \beta_R(\vec{n}))$ strongly affects the performance of a queueing network model. Indeed, whenever the number of jobs in a multiclass region overflows the capacity constraint, it is not sufficient to saturate the FESC service rates to a constant value as in single class models since the departure rate strongly depends on the mix $\vec{\beta}_m(\vec{n}) = (\beta_{m,1}(\vec{n}), \ldots, \beta_{m,R}(\vec{n}))$ of active jobs inside region m. Let $\vec{n}$ be the vector of both waiting and active jobs which visit region m. The key issue for the definition of class-r FESC service rate, $\mu_{m,r}(\vec{n})$, is the knowledge of the exact proportion of *active* jobs in region m. Since this value is not known (see Section A.2), we approximate the class-r mix of active jobs in region m as $\beta_{m,r}(\vec{n}) = n_r/n$. In other words, we approximate the population-mix vector of active jobs in region m with vector

$$\vec{\beta}_m(\vec{n}) = \left(\frac{n_1}{n}, \ldots, \frac{n_R}{n}\right) \tag{3}$$

which represents a measure of the average population mix belonging to $\Omega_m(\vec{n})$. We do not consider the average population mix (2) since its computation requires the inefficient complete exploration of $\Omega_m(\vec{n})$. Approximation (3) states that the population-mix vector of *active* jobs in m is equal to the population-mix vector of *ready* jobs in m. This is in general an approximation, since it considers also waiting jobs. FESCs service rates are then defined as

$$\widehat{\mu}_{m,r}(\vec{n}) = \begin{cases} \widehat{X}_{m,r}(\vec{n}) & \{\vec{n} \mid n \leq B_m\} \\ \widehat{X}_{m,r}(\vec{s}_m) & \text{otherwise} \end{cases} \tag{4}$$

where $\widehat{X}_{m,r}$ denotes the class-r throughput of region m in isolation and $\vec{s}_m = (s_{m,1}, \ldots, s_{m,R})$ is a population vector related to region m such that $s_{m,r} = s_{m,r}(\vec{n}) = \beta_r(\vec{n})B_m$. Hence, vector $\vec{s}_m$ is such that $\sum_r s_{m,r} = B_m$ and $\beta_r(\vec{s}_m) = \beta_r(\vec{n})$. The exact value of $\widehat{X}_{m,r}(\vec{n})$, for all population vectors $\{\vec{n} \mid n \leq B_m\}$, is computed by a single

run of the MVA [3]. Hence, for such populations the way in which we define $\widehat{\mu}_{m,r}(\vec{n})$ is exact since the actual mix of active jobs in m is clearly known and equal to 1. Term $\widehat{X}_{m,r}(\vec{s}_m)$ is the approximation which models finite capacity effects and encodes the fact that no more than B_m jobs can reside in m.

In general vector $\vec{s}_m$ has non-integral elements and the computation of $\widehat{X}_{m,r}(\vec{s}_m)$ can be performed by running the MVA for non-integral populations [5]. The non-integral parts of $\vec{s}_m$, i.e. $\vec{s}_m - \lfloor \vec{s}_m \rfloor$, vary for each $\vec{n}$ and a different run of non-integral MVA is needed for each population vector overflowing capacity constraint. Since this approach is inefficient, for population vectors $\{\vec{n} \mid n > B_m\}$ we round $\vec{s}_m$ to the nearest integral vector redefining $\widehat{\mu}_{m,r}(\vec{n})$ as

$$\widehat{\mu}_{m,r}(\vec{n}) = \begin{cases} \dfrac{\beta_r(\vec{n})B_m}{[\vec{s}_m]_r}\widehat{X}_{m,r}([\vec{s}_m]) & [\vec{s}_m]_r \neq 0 \\ \beta_r(\vec{n})B_m\widehat{X}_{m,r}([\vec{s}_m] + \vec{e}_r) & [\vec{s}_m]_r = 0 \end{cases} \quad (5)$$

where $[\cdot]$ returns the nearest integral vector to $\cdot$, and $\beta_r(\vec{n})B_m/[\vec{s}_m]_r$ and $\beta_r(\vec{n})B_m$ are scaling factors which interpolate $\widehat{X}_{m,r}(\vec{s}_m)$. Thus, (5) defines service rates of m by simply reusing throughputs computed with the MVA.

IV. Product-form Approximation of Service Rates

The FESCs service rates defined by (5) do not satisfy product-form and the solution of queueing networks with such service centers requires the solution of the underlying Markov chain (see, e.g., [15]). Since this solution makes the analysis intractable even for small models, in the following we propose a solution method that let us solve such non-product-form models with an approximate variant of the convolution algorithm. For product-form queueing networks the following steady state probability distribution holds [3]: $P(\vec{n}_1, \ldots, \vec{n}_K) = G^{-1} \prod_k F_k(\vec{n}_k)$, where G is the normalizing constant. From the discussions in [16], [17], in our case it follows that for class-specific queue-dependent service rates $\mu_{k,r}(\vec{n})$, the following equation holds

$$F_k(\vec{n}) = \frac{1}{\mu_{k,r}(\vec{n})} F_k(\vec{n} - \vec{e}_r), \ \vec{n} \geq \vec{e}_r \quad (6)$$

where r may be any of the job classes and $F_k(\vec{0}) = 1$. Let us define a *path* as a sequence of population vectors which can be spanned in the computation of $F_k(\vec{n})$ with (6). Since r is an arbitrary class, we can perform a recursion along a generic path from $\vec{n}$ to $\vec{0}$ obtaining the same value of $F_k(\vec{n})$. Whenever (6) does not well define $F_k(\vec{n})$, the rate function $\mu_{k,r}(\vec{n})$ does not satisfy product form [16]. $F_k(\vec{n})$ is well defined if and only if the following condition holds,

$$\mu_{k,i}(\vec{n})\mu_{k,j}(\vec{n} - \vec{e}_i) = \mu_{k,j}(\vec{n})\mu_{k,i}(\vec{n} - \vec{e}_j) \quad (7)$$

where $i \neq j$, $\vec{n} - \vec{e}_i > \vec{e}_j$ and $\vec{n} - \vec{e}_j > \vec{e}_i$. For stations with FCFS scheduling, (7) is a necessary condition to guarantee product-form. In our work, $\widehat{F}_m(\vec{n})$ is related to the FESC belonging to the resulting network representing region m. For population vectors $\{\vec{n} \mid n \leq B_m\}$, functions $\widehat{F}_m(\vec{n})$ are well defined since the way in which we define service rates guarantees product-form [7]. However, for population vectors $\{\vec{n} \mid n > B_m\}$, in general $\widehat{F}_m(\vec{n})$ is not well defined. This means that during the recursion the value of $\widehat{F}_m(\vec{n})$ depends, for such $\vec{n}$, on the particular path chosen from $\vec{n}$ to $\vec{0}$. This can be easily verified considering a two-class, two-station model with population vector $\vec{n} = (2, 1)$, capacity $B = 2$ and loadings $D_{1,1} = 0.5$, $D_{2,1} = 2$, $D_{1,2} = D_{2,2} = 1$, where $D_{m,r} = V_{m,r}/\mu_{m,r}(\vec{e}_r)$. Hence, in order to gain a path-independent value for $\widehat{F}_m(\vec{n})$ we compute the value of $\widehat{F}_m(\vec{n})$ by taking the average of all *border* paths of the state diagram, i.e. $\{\vec{n} - \vec{e}_{r_1}, \ldots, \vec{n} - N_{r_1}\vec{e}_{r_1}, \vec{n} - N_{r_1}\vec{e}_{r_1} - \vec{e}_{r_2}, \ldots, \vec{n} - N_{r_1}\vec{e}_{r_1} - N_{r_2}\vec{e}_{r_2}, \ldots, e_{rR}, \vec{0}\}$ where $r_i \in \{1, \ldots, R\}$. Let us denote by $p \equiv p(\vec{n})$ the number of non-empty classes of vector $\vec{n}$. The set of all border paths has cardinality equal to $p!$. However, since in general queueing networks with finite capacity region with more than four classes are intractable and in practice are rarely used, cardinality of $\Phi(\vec{n})$, for $R \leq 4$, is small and does not affect the algorithm performance. Our approximate variant of the convolution algorithm computes $\widehat{F}_m(\vec{n})$ for population vectors $\{\vec{n} \mid n > B_m\}$ as

$$\widehat{F}_m(\vec{n}) = \frac{1}{p!} \sum\nolimits_{\varphi \in \Phi(\vec{n})} \widehat{F}_m^{\varphi}(\vec{n}) \quad (8)$$

where $\Phi(\vec{n}) = \{\varphi \mid \varphi \text{ is a border path from } \vec{n} \text{ to } \vec{0}\}$. Using (8), we approximate the solution of the resulting network without solving the Markov chain which is needed in order to solve of non-product-form models but makes the problem intractable.

V. Experimental results

In this section we show experimental results for the proposed approximation comparing our results with simulation results which have been obtained using the JMT simulation engine [2] choosing 99% level confidence intervals. We evaluate accuracy by measuring the percentage relative error $(|X_S - X_A|/X_S) \cdot 100\%$, where X_A and X_S refer to overall throughputs of our approximation and simulation, respectively. We validate the approximation on a class of models with exactly ten load-independent queues and a single *delay*. The *delay* does not belong to any finite capacity region. The number of regions M is randomly drawn from $\{1, 2, 3, 4\}$. Queues service rates $\mu_{k,r}(\vec{e}_r)$ range in $[0.01, 12]$ jobs per unit of time and visits $V_{k,r}$ in $[0.1, 5]$. Loadings $D_{k,r}$ are computed as $V_{k,r}/\mu_{k,r}(\vec{e}_r)$. Average *think* times Z_r range in [0.01, 10]. In order to limit the cost of simulations we decrease population sizes and proportionally scale capacity ranges when increasing the number of classes. For models with 2 (respectively 3 and 4) classes, we consider a maximum of 200 (100 and 60) jobs, a maximum of 100 (33 and 15) per-class jobs and capacities never exceed 40 (20 and 12) units. In order to better stress the finite capacity effects, we assume that all regions have a finite capacity and each queue belongs to one region. Moreover, we generated 400 random models ensuring that $N > max_{1 \leq m \leq M} B_m$. On an Intel Xeon 2.80GHz processor with hyperthreading technology, our method requires

an average of 10 seconds while simulation takes more than 6 minutes to converge. In Table 1 we show mean, median and standard deviation of the percentage relative errors by varying the number of classes. We conclude that the results of our algorithm are in good agreement with simulation.

R	Mean	Median	Std. Dev.
2	11.5%	2.8%	31.8%
3	13.1%	4.9%	22.6%
4	11.6%	4.4%	22.8%

TABLE I

Percentage relative errors.

We also compare our method results against models published in literature. The model under investigation (see, e.g., [8], [15]) consists of two classes of customers, a processor sharing center representing the CPU, four FCFS queues representing the disks with identical service times and a delay. In the first comparison there is a single finite capacity region composed by the four disks while in the second we have two regions with both two disks. In Tables II and III we show response times of our method (APP), simulation (SIM) and method [8] (KT). Consider for instance the cases in which the capacity constraints limit at the most population $\vec{N}$ inside regions, i.e., when the approximation is strongly stressed: we improve significantly the accuracy in all cases, except for the class-1 response time of case $\vec{N} = (20, 2)$, $B_1 = B_2 = 1$ where we are slightly less precise than KT. Consider also the case of a single region with $\vec{N} = (20, 2)$ and $B_1 = 2$: with respect to the average value of confidence interval, for class-1 response time we provide an error less than 3% instead of 28%. Also for the other cases, we conclude that in general our approximation returns more accurate results.

VI. Conclusions

In this paper we propose an approximate method for multiclass closed queueing networks with finite capacity regions based on the construction of FESCs which replace regions and model finite capacity effects. We propose to solve the resulting non-product-form network with a variant of the convolution algorithm. We performed a numerical study to assess the quality of our approximation and experimental results show the approximation has a good accuracy. In addition, a comparison with existing techniques on published models shows improved accuracy. We leave as future work the extension to overlapped and nested finite capacity regions and open models in order to integrate the approximation in admission control schemes.

			Class 1 Resp. time			Class 2 Resp. time		
N_1	N_2	B_1	KT	SIM	APP	KT	SIM	APP
		6	.73	(.71, .74)	0.73	4.71	(4.20, 5.02)	4.73
20	2	4	.83	(.72, .74)	0.75	4.65	(4.19, 4.97)	4.76
		2	1.40	(.99,1.03)	1.04	4.44	(4.60, 5.55)	5.43
		10	1.01	(.98,1.01)	1.01	7.34	(6.76, 8.03)	7.59
30	3	6	1.13	(1.01.1.04)	1.04	7.11	(6.17, 8.07)	7.63
		2	2.50	(2.06,2.16)	2.32	6.09	(7.63, 9.33)	9.70
		18	1.48	(1.40,1.44)	1.48	12.95	(10.54,12.46)	13.38
40	4	12	1.49	(1.46,1.52)	1.49	12.90	(12.80,15.26)	13.38
		6	1.80	(1.57,1.62)	1.60	11.75	(11.68,13.91)	13.26

TABLE II

First case: a single finite capacity region

				Class 1 Resp. time			Class 2 Resp. time	
N_1	N_2	B_1	B_2	KT	SIM	APP	KT	SIM	APP
		3	3	.79	(.73, .75)	.74	4.51	(4.28,5.05)	4.75
20	2	2	2	.92	(.78, .80)	.78	4.64	(4.94,5.81)	4.88
		1	1	.98	(.97,1.01)	1.06	4.10	(4.76,5.68)	5.94
		5	5	1.04	(.97,1.00)	1.02	7.26	(6.84,8.06)	7.60
30	3	3	3	1.20	(1.06,1.09)	1.07	7.05	(6.23,7.43)	7.69
		1	1	1.68	(1.79,1.85)	1.90	6.44	(7.87,9.67)	9.92
		9	9	1.48	(1.46,1.50)	1.49	12.60	(12.82,15.15)	13.38
40	4	6	6	1.55	(1.42,1.47)	1.50	12.91	(11.53,13.61)	13.37
		3	3	1.83	(1.59,1.64)	1.62	12.95	(11.19,13.48)	13.28

TABLE III

Second case: two finite capacity regions

References

[1] G. Balbo and G. Serazzi, *Asymptotic analysis of multiclass closed queueing networks: Common bottlenecks*, Per. Ev., 26 (1996), pp. 51–72.

[2] M. Bertoli, G. Casale, and G. Serazzi, *Java modelling tools: an open source suite for queueing network modelling and workload analysis*, in Proc. of QEST, Sep 2006, pp. 119–120.

[3] G. Bolch, S. Greiner, H. de Meer, and K. S. Trivedi, *Queueing Networks and Markov Chains*, Wiley-Inter., 2005.

[4] A. Brandwajn and W. M. McCormack, *Efficient approximation for models of multiprogramming with shared domains*, in Proc. ACM SIGMETRICS 1984, NY, USA, 1984, pp. 186–194.

[5] L. Dowdy and K. Gordon, *Algorithms for nonintegral degrees of multiprogramming in closed queueing networks*, (1982).

[6] N. Gautam and S. Seshadri, *Performance analysis for e-business: Impact of long range dependence.*, El. Co. Res., 2 (2002), pp. 233–253.

[7] P. Kritzinger, S. V. Wyk, and A. Krzesinski, *A generalization of norton's theorem for multiclass queueing networks*, Perf. Eval., 2 (1982), pp. 98–107.

[8] A. E. Krzesinski and P. Teunissen, *Multiclass queueing networks with population constrained subnetworks.*, in SIGMETRICS, 1985, pp. 128–139.

[9] E. D. Lazowska, J. Zahorjan, G. S. Graham, and K. C. Sevcik, *Quantitative System Performance*, Prentice-Hall, Englewood Cliffs, New Jersey, 1984.

[10] D. A. Menascé, *Simple analytic modeling of software contention*, SIGMETRICS Perf. Eval. Rev., 29 (2002), pp. 24–30.

[11] D. A. Menascé and H. Gomaa, *A method for design and performance modeling of client/server systems*, IEEE Trans. Softw. Eng., 26 (2000), pp. 1066–1085.

[12] J. E. Neilson, C. M. Woodside, D. C. Petriu, and S. Majumdar, *Software bottlenecking in client-server systems and rendezvous networks*, IEEE Tr. Soft. Eng., 21 (1995), pp. 776–782.

[13] R. N.-Q. O. J. Boxmaa, N. Hegdeb, *Exact and approximate analysis of sojourn times in finite discriminatory processor sharing queues*, Int. J. Electron. Commun., (2006), pp. 109–115.

[14] J. A. Rolia and K. C. Sevcik, *The method of layers*, IEEE Trans. Softw. Eng., 21 (1995), pp. 689–700.

[15] C. H. Sauer, *Approximate solution of queueing networks with simultaneous resource possession*, IBM J. R. and D., 25 (1981).

[16] C. H. Sauer, *Computational algorithms for state-dependent queueing networks*, ACM Tr. Comp. Syst., 1 (1983), pp. 67–92.

[17] C. H. Sauer and K. M. Chandy, *Computer Systems Performance Modelling*, Prentice-Hall, Englewood Cliffs, NJ, 1981.

[18] A. V. Srinivas and D. Janakiram, *A model for characterizing the scalability of distributed systems*, SIGOPS, 39 (2005), pp. 64–71.

[19] N. Ye, *Qos-centric stateful resource management in information systems*, Inf. Systems Frontiers, 4 (2002), pp. 149–160.