

ESTIMATING BOTTLENECKS OF VERY LARGE MODELS

Giuliano Casale, Giuseppe Serazzi *

Abstract

Queueing theory has been extensively used since 70's to carry out performance evaluations of complex computer systems. However, handling the complexity of modern computer networks with thousands of servers and millions of customers is still a challenge. Indeed, although the identification of product-form queueing networks has promoted the development of computationally tractable exact solution techniques, the dimensionality of "very large" models makes it difficult to meet the requirements of either exact and approximate algorithms. In this paper we focus on the bottleneck analysis as a technique that may provide answers to many capacity planning questions (e.g., "which is the maximum number of transactions per second the computer system is able to process?" or "which is the minimum response time that can be achieved per transaction?") for "very large" computer installations with a limited complexity. The problem to be solved with this technique is the identification of the set of resources that may saturate among the thousands of components. We show some results concerning the application of the theory of convex polyhedra to bottleneck analysis that can sensibly reduce the complexity of the analysis. The connections between this technique and the behavior of closed and open models are also investigated.

1. Introduction

The success of the Internet in the last decade and the evolution of new technologies, like intranets and middlewares, have increasingly shifted the attention of performance engineers to the evaluation of complex distributed architectures. While in the 70's and in the 80's most enterprises had mainframe-based or client-server systems, often with a fixed and limited number of terminals, actual architectures frequently involve a large number of devices, that should process heterogeneous workloads, multiclass in nature. The intranet of a large company can involve thousands of geographically distributed servers, including file servers, email servers, application servers, firewalls and Web servers, hosting a variety of applications accessed by thousands of clients. It is clear that the execution patterns of these applications are usually diversified, and this naturally results in the presence of different classes of loadings. An example of the complexity of actual enterprise networks is given by the Intel computing infrastructure: in 2001 it comprised over 3.000 servers and 100.000 client PC's [27] and the prevision was to double the number of servers within 18 months.

Beyond the difficulty of modeling different applications, we have to deal with additional factors of complexity, like load balancers, VPNs or Web caching techniques, that worsen the problem of creating a simple, but effective, performance model. Nevertheless, the role of capacity planning and performance optimization is emphasized by the dimension of the architecture, which makes scaling a primary requirement, and this is relevant also for emerging distributed technologies, like grid computing and peer-to-peer file sharing, since a larger number of system can be involved.

Since the number of requests of each customer to corporate servers is potentially unbounded, open queueing networks are a natural choice for modeling large Internet-based networks. An accurate modeling, however, should take in account also the load imposed by batch processes or transaction-based software, which can be studied using mixed queueing network models. As a consequence, understanding the behavior of large scale open and closed queueing networks is relevant to the analysis of large systems. Furthermore, the development of specialized techniques to handle the computational complexity of solving such models is desirable.

In the following, we will refer to models with thousands of servers, millions of customers that belong to several classes as "very large models". In this paper we investigate on the applicability of bottleneck analysis technique to understand scalability problems in "very large" computer infrastructures. It is known that the resources that limit the overall performance of a system (independently of its dimensions) are the saturated ones, referred to as bottlenecks. While the identification of the bottleneck in a simple network with a single class of customers is a straightforward task, for "very large" networks this task is very hard to accomplish in a reasonable amount of time. In this paper we present a technique based on the theory of

*Dipartimento di Elettronica ed Informazione, Politecnico di Milano, P.za L. da Vinci 32, 20133 Milano, Italy.

E-mail: {casale,serazzi}@elet.polimi.it. This work has been partially supported by Cofin2001-Web and Fibr-Perf projects.

convex polyhedra for the identification of the potential bottlenecks of a very large network with tens of thousands servers, unlimited customers and up to seven customers classes. This strategy is shown to be a valuable optimization scheme for the asymptotic analysis of closed models.

The outline of the paper is as follows. Section 2 summarizes existing exact and approximate techniques for product-form queueing networks. Section 3 describes a technique based on the theory of convex polyhedra for the identification of the potential bottlenecks of very large open and closed models. Section 4 describes the bottleneck analysis of multiclass closed models and presents a new optimization scheme. Section 5 introduces the bottleneck analysis of “very large” open models.

2. Review of analysis techniques

This section briefly reviews product-form queueing networks and gives a survey of exact and approximate techniques available for solving open, closed and mixed networks of queues. Most of the described methods were developed for product-form queueing networks; nevertheless some of them have been extended to non product-form networks (e.g. [9, 21]).

Product-form (or separable) queueing networks were introduced for open models by J. Jackson in 1963 [28], who established a simple closed-form expression of the steady state probability distribution. The validity of the result is ensured under several conditions, like single customer class, Poisson arrivals, exponential service times and FCFS service disciplines. Jackson’s theorem is referred to as the first major breakthrough in separable queueing system theory. In 1967 Gordon and Newell extended Jackson’s work to closed models [24]. Their theorem states that the probability distribution of the system at the equilibrium can be computed if the value of the *normalization constant* is known. A further generalization was introduced in 1975, when F. Baskett, K.Mani Chandy, R.R. Muntz and F.G. Palacios proved the well-known BCMP theorem [8]. This extends the product-form solution to a wider class of networks including servers with non-exponential service times, different queueing disciplines, load-dependent service rates, population-dependent arrivals and multiple job classes. Further researches have found new cases in which the product-form solution is preserved [5], e.g. some networks with finite capacity queues [1, 39].

All these developments were of great interest for the definition of new solution techniques, since product-form queueing networks are computationally tractable. From an historical perspective, the development of efficient analytical techniques for computing the normalization constant was limited at the beginning by the dimensionality of the state space. Enumeration schemes are applicable only to trivial cases. A limited availability of computational capacity raised the problem of finding efficient solution techniques for queueing systems. These issues were first addressed in 1973 by J.P. Buzen [10] with the definition of the *convolution algorithm*, which uses a recursion on the number of servers, rather than on the number of states, to iteratively compute the normalization constant. The procedure was then extended to multi-class networks by M. Reiser and H. Kobayashi [42]. The convolution algorithm reduces the complexity of computing the normalization constant, but its complexity depends on the product of class population sizes. Some variants of the convolution algorithm that handle models with specific characteristics are available, like *Tree convolution* [32], for large models with many sparse routing chains, *DAC* [20] and *RECAL* [15] whose computation times grow as a polynomial function of the number of classes. Other techniques for computing the normalization constant are *LBANC* [11], *CCNC* [11], *Polya enumeration of polynomials* [30], *matrix geometric* [38, 48] and the *partial fraction method* [37].

The normalization constant gives a full description of the steady-state system behavior. However, mean values of performance measures are often enough for estimating performance of complex models. *Mean Value Analysis* (MVA) [43], based on the *arrival theorem* [33, 36], avoids a direct computation of the normalization constant using recursive expressions to calculate the mean performance indices. While MVA has higher storage requirements than the convolution algorithm and the same time complexity, its implementation is simpler in most cases.

A number of specialized methods have been derived from exact MVA. *Tree-structured MVA* [44, 26] uses the concepts introduced by the Tree convolution algorithm and the aggregation techniques of the *flow equivalent server method* (FES) [13] to define an MVA recursion that has to be preferred for large models with sparse routing chains. *MVAC* [16] is specialized for networks with few stations and many customer classes. Further researches has also extended MVA to handle mixed models with single server nodes [49].

The complexity of exact methods grows exponentially in most cases, and this prevents their use in the evaluation of very large models. A valuable alternative are *local approximation techniques*, which are less expensive in time and space requirements, but yield just an “estimate” of performance indices. Well-known local approximation methods are Chow [14], Schweitzer-Bard [7, 46], Linearizer [12], and AQL [51]. The rationale behind these methods is finding a suitable approximation of the class r mean queue length $Q_{ir}(\underline{N})$ (or of the aggregated queue length $Q_i(\underline{N})$) at the i -th node with population vector \underline{N} . This value is then used to express MVA recursive equations for mean queue lengths in fixed-point form. The *Chow algorithm* assumes that in the neighborhood of a given population \underline{N}^0 the queue size $Q_i(\underline{N} - \underline{1}_r)$ can

be substituted with $Q_i(N)$. Its time complexity does not depend strongly on the total population N^0 . However, the approximation employed is accurate only for a large number of customers, since it does not scale with the population, i.e. $\sum_i Q_i(N - \underline{1}_r) \neq N - 1$. The *Schweitzer-Bard* (SB) algorithm was the first local approximation scheme. SB yields better results than Chow, at the cost of a small increase in the computational complexity. The adopted approximation for class r queue length is given by:

$$Q_{ir}(N^0 - \underline{1}_s) \approx \frac{N_r - 1_{r=s}}{N_r} Q_{ir}(N^0)$$

where N_r is class r population size, and $1_{r=s}$ is one only when $r = s$, zero otherwise. The SB approximation does not depend strongly on N^0 , but scales with the population achieving better precision. For each iteration of MVA equations in fixed-point form the computational complexity of SB algorithm is $O(MR^2)$, where M is the number of nodes and R the number of job classes. A different trade-off between time requirements and accuracy is provided by the *Linearizer* [12], that approximates the queue lengths considering the network after the removal of two customers. This allows to achieve better precision than SB, while the computational complexity becomes $O(MR^5)$. The *Aggregated Queue Length* (AQL) method, instead, uses an approximation that resembles that of Linearizer, but applied to the aggregated queue lengths $Q_i(N)$. The method suffers a decrease in the accuracy compared to the other techniques, but the overall time complexity is improved.

A new analytical framework, the *interpolation-matching technique* (IMT), has been proposed in [17] as a generalization of existing local approximation strategies. Any IMT algorithm can be specified by fixing the interpolating functions that approximate the queue lengths and the population vectors on which the MVA recursion is evaluated. The choice of these parameters affects the trade-off between computational complexity and accuracy. IMT yielded the definition of a new strategy, namely *Linearizer++*, which shows better accuracy than Linearizer, with an increased computational cost of $O(MR^8)$.

A different kind of approximated strategy is provided by *PANACEA* [35, 41]. This is a package for large open, closed and mixed queueing models that implements an efficient approximation scheme to calculate the normalization constant. The approximation is obtained by expanding an integral representation of the normalizing constant in power series. The asymptotic expansion is possible for networks under not heavily loaded conditions (under "normal usage", i.e. with utilizations below $80 \div 85\%$). PANACEA is numerically stable and provides analytical bounds for the errors incurred from the truncation of the expansion. The main attractive of this technique is its computation time that grows as a polynomial in the number of classes and is independent from the population vector.

Another popular class of approximation techniques is *bottleneck analysis* [45], which is based on the observation that the stations with the highest utilization are usually the main sources of delay. Thus identifying bottleneck stations is a simple way to optimize real systems, since improving their performances usually yields a sensible performance gain. This is in particular valuable when it is possible to forecast the a posteriori behavior of the upgraded system, since the identification of the new bottleneck after a modification can be dangerous. A distinguishing advantage of bottleneck analysis over other solution techniques is its limited computational complexity. For instance, bottleneck analysis of single class networks is straightforward, since the primary bottleneck is simply the one with the maximum loading L_{max} [18]. However, this approach is unsatisfactory in saturated networks, where we must take in account that part of the incoming load is masked by bottleneck stations, and this could be done by means of a non-linear functional equation [23]. However, the most interesting application of bottleneck analysis relies in the analysis of large models with multiple customer classes. When considering finite populations, heuristics models can be used. A general approach to decompose any fixed point model by guessing the bottleneck set is described in [47]. A way to perform the bottleneck analysis of models subject to variable workload is discussed by J.Luthi and G.Haring in [34]. Due to the difficulty in studying these models, existing results have mainly focused on the asymptotic condition of the network when the population size goes to infinity. In this case the performance indices of the models can be obtained using the techniques presented in [3, 4]. Recently [25] have shown a new technique for computing the normalization constant of closed models, which can help in understanding the asymptotic behavior of the network.

Bound analysis is another popular technique that shares with bottleneck analysis the small computational requirements. In capacity planning and in other fields, computing an upper or a lower bound for the performance indices of a large model can be enough to understand if the system satisfies some design requirements. Previous work on bound analysis have addressed the problem for both single and multiple class models: [18] gives a survey on basic results for single class models, [50, 31] presents balanced job bounds for single and multiclass closed models, [22, 29] introduce techniques specific to multiclass models, and [19] studies the connections between the performance indices of multiclass models and those of single class models.

3. Potential bottlenecks identification

In this section we present a method for the identification of the resources that can saturate, under an appropriate workload, in a multiclass model. We show how the concept of convex hull [40], which is of primary importance in combinatorial optimization and computational geometry, can help in determining whether or not a station can ever saturate with a given workload. The method is shown to be sufficient for identifying the sets of resources that can saturate together. The technique is based on the analysis of the loading matrix only, while all the information concerning arrival rates or populations of the different classes are not required. Sections 4 and 5 show how this additional knowledge can help in analyzing the actual behavior of the network. The results presented here hold for open, closed and mixed product-form queueing networks.

Let us consider a queueing network with customers partitioned in R classes according to their service demand patterns. We consider networks composed only by constant-rate queueing centers. Let $\mathbf{R} = \{1, 2, \dots, R\}$ denote the set of class indices and $\mathbf{M} = \{1, 2, \dots, M\}$ the set of station indices. The terms L_{ir} are the service demands of class- r jobs at station i ¹, grouped in a $M \times R$ loading matrix \mathbf{L} . The station i with $\max_i(L_{ir})$ for a given r is the *natural bottleneck* of class r . We consider the general case of systems with *multiple bottlenecks*, i.e. natural bottlenecks placed on different stations. It is clear that networks with the same natural bottleneck for all classes, the *common bottleneck* models, are a specialization of the former case.

Resources that never saturate are classified in two groups, namely *dominated stations* and *masked-off stations*. A station i is said to be dominated if there exists another station j such that

$$L_{jr} > L_{ir} \quad \forall r. \quad (1)$$

A descending ordering of \mathbf{L} , according to the values of any component, say class 1 loadings, can help in excluding such stations from the analysis, since many dominated stations are in a row below that of the lowest natural bottleneck². The masked-off class, instead, is composed of stations that, while not dominated, are prevented by other stations from becoming a bottleneck. In Theorem 1 we show a sufficient condition for identifying masked-off stations.

We now show how to determine the potential bottlenecks through the convex hull problem. Let us consider the loading matrix \mathbf{L} : this is basically a set of row-vectors of R components, which describe a collection of M points in a space. Thinking at the loading factors L_{ir} as coordinates, we can map each station i to a vector \underline{l}_i in \mathbb{R}^R (henceforth the *loadings space*). We denote $\mathcal{P}(\mathbf{L}) = \{\underline{l}_i : i \in \mathbf{M}\}$ the set of points in the loadings space induced by matrix \mathbf{L} .

Now, let us consider a generic vector $\underline{l}_i \in \mathcal{P}(\mathbf{L})$. We denote $\Pi(\underline{l}_i)$ the set of all possible projections of \underline{l}_i , i.e. the set of all vectors derived by \underline{l}_i by setting to zero k of its components, for all possible $k = 1, \dots, R$. Since a member of $\mathcal{P}(\mathbf{L})$ can have some coordinate set to zero, it is possible that the projection of a point is again a member of $\mathcal{P}(\mathbf{L})$. For example, the projection of $l_2 = (0, 50)$ on the class-2 axis is again l_2 . We define $\Pi(\mathbf{L})$ as the set of all projections of the points in $\mathcal{P}(\mathbf{L})$ that are not members of $\mathcal{P}(\mathbf{L})$ itself:

$$\Pi(\mathbf{L}) = \bigcup_{i=1}^M \Pi(\underline{l}_i) - \mathcal{P}(\mathbf{L}) \quad \underline{l}_i \in \mathcal{P}(\mathbf{L})$$

According to this definition, the stations of the network are all members of $\mathcal{P}(\mathbf{L})$, while the points in $\Pi(\mathbf{L})$ are not mapped from any real resource. In the rest of the section the terms station and point of $\mathcal{P}(\mathbf{L})$ are used without distinction. This definition of $\Pi(\mathbf{L})$ allows us to know whether or not a point is mapped from a real station.

We now give an introductory definition:

Definition 1. Characteristic polytope 1. Let \mathbf{L} be the loading matrix of a multiclass queueing network with M queueing centers and R classes. The characteristic polytope $C_{\mathbf{L}}$ of \mathbf{L} is the convex hull of the set $A = \mathcal{P}(\mathbf{L}) \cup \Pi(\mathbf{L})$.

The *convex hull* of a set of points A is the smallest convex set containing A . Once computed, $C_{\mathbf{L}}$ yields a list of faces and vertices that can be used to determine the set of potential bottlenecks. The complexity of existing algorithmic approaches for solving the convex hull problem is discussed at the end of the section. Henceforth we will use the notation $V(C_{\mathbf{L}})$ to denote the set of vertices of the characteristic polytope. The coordinates of a vertex are in capital letters (e.g. L_i or \underline{L}_i).

A simple example of convex hull for a three class queueing model with the loading matrix of Figure 1 is presented in Figure 2. Each boxed value L_{ij} of the loading matrix $\underline{\mathbf{L}}$ indicates that station i is the natural bottleneck for class j . The most interesting properties of the characteristic polytope are shown in the following theorems:

¹Unless otherwise specified, in the rest of the paper the resource indices i, j and k will be implicitly assumed to range from 1 to M . Similarly r and s , the class indices, will range from 1 to R .

²The domination condition $L_{jr} > L_{ir} \forall r$ can be relaxed to $L_{jr} \geq L_{ir}$ if $L_{jr} = L_{ir} = 0$. For instance, $(0, 10, 50)$ dominates $(0, 5, 45)$, since the number of class-1 jobs does not affect the utilization of the two stations.

$$\mathbf{L} = \begin{pmatrix} 905 & 499 & 707 \\ 846 & 629 & 217 \\ 200 & 930 & 410 \\ 186 & 353 & 952 \\ 378 & 476 & 482 \\ 92 & 215 & 573 \\ 26 & 100 & 16 \end{pmatrix}$$

Figure 1. Loadings on the different resources for a queueing model with seven stations and three customer classes

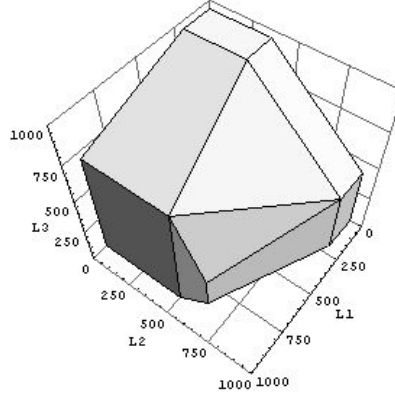


Figure 2. Characteristic polytope $C_{\mathbf{L}}$ of the queueing model of Figure 1

Theorem 1. Masked-off stations. The stations mapped to an internal point of $C_{\mathbf{L}}$ are masked-off.

Proof. Let l_k be a internal point of $C_{\mathbf{L}}$. This implies that for any point \underline{l} of the loadings space there exists a $\epsilon > 0$ such that the point $\underline{v} = \underline{l}_k + \lambda \underline{l}$ is in $C_{\mathbf{L}}$ for all $\lambda \in \mathbb{R} : 0 \leq \lambda < \epsilon$. In particular, this holds true when $\underline{l} = \underline{l}_k \in C_{\mathbf{L}}$. If $\lambda = 0$ we have trivially $\underline{l}_k = \underline{v}$, otherwise we can always find a point $\underline{v} \in C_{\mathbf{L}}$ such that $\underline{l}_k < \underline{v}$. Since we are working with a convex set, we can express \underline{v} as a convex combination of at most $d + 1$ (affinely independent) vertices [40], where d is the affine dimension of the set (i.e. 0 for a point, 1 for a line, ...), thus:

$$\underline{v} = \alpha_1 \underline{V}_1 + \alpha_2 \underline{V}_2 + \dots + \alpha_{d+1} \underline{V}_{d+1}, \quad \sum_{i=1}^{d+1} \alpha_i = 1, \quad \alpha_i \geq 0$$

where $\underline{V}_i \in V(C_{\mathbf{L}})$. Since $C_{\mathbf{L}}$ is the convex hull of $\mathcal{P}(\mathbf{L}) \cup \Pi(\mathbf{L})$, each vertex \underline{V}_i belongs only to one between $\mathcal{P}(\mathbf{L})$ and $\Pi(\mathbf{L})$. Considering that each vertex $\underline{V}_i \in \Pi(\mathbf{L})$ is the projection of a point in $\mathcal{P}(\mathbf{L})$, we can always find a vector $\underline{L}_i \in \mathcal{P}(\mathbf{L})$ such that

$$\underline{L}_i \in \mathcal{P}(\mathbf{L}) : \underline{V}_i \leq \underline{L}_i, \quad \forall i : 1 \leq i \leq d + 1, \quad \underline{V}_i \in C_{\mathbf{L}}$$

where the strict inequality holds if $\underline{V}_i \in \Pi(\mathbf{L})$. With this result we now get

$$\underline{l}_k < \underline{v} \leq \alpha_1 \underline{L}_1 + \alpha_2 \underline{L}_2 + \dots + \alpha_{d+1} \underline{L}_{d+1}, \quad \sum_{i=1}^{d+1} \alpha_i = 1, \quad \alpha_i \geq 0$$

Summing the components of each vector and using the basic relation:

$$U_{ir} = \frac{L_{ir}}{L_{jr}} \cdot U_{jr} \quad (2)$$

we find

$$U_k < \alpha_1 U_1 + \alpha_2 U_2 + \dots + \alpha_{d+1} U_{d+1}, \quad \sum_{i=1}^{d+1} \alpha_i = 1, \quad \alpha_i \geq 0$$

that is meaningful since $\underline{L}_i \in \mathcal{P}(\mathbf{L})$ guarantees that all the points are associated to a real station. Inserting $U_i \leq 1$ in the previous inequality, it follows that

$$U_k < \alpha_1 U_1 + \alpha_2 U_2 + \dots + \alpha_{d+1} U_{d+1} \leq \sum_{i=1}^{d+1} \alpha_i = 1, \quad \alpha_i \geq 0$$

which proves the theorem. \square

Theorem 1 suggests that finding the characteristic polytope $C_{\mathbf{L}}$ can be an effective way for excluding a considerable number of stations from the candidate bottleneck set. The reason why we introduced $\Pi(\mathbf{L})$ is to ensure that all the dominated and masked-off stations are included in the characteristic polytope as internal points. We now try to understand whether a point on a face of $C_{\mathbf{L}}$ can saturate. It is clear that all the natural bottlenecks are vertices of $C_{\mathbf{L}}$, since on the dimension corresponding to the class for which they are natural bottlenecks they must be an extreme point of the convex hull. Moreover, it is possible that some points on the boundary faces are dominated (e.g. the origin) and thus cannot saturate. The second theorem we prove clarifies whether or not the points lying on a boundary face can saturate.

Theorem 2. Boundary faces. If there exists a set of $d + 1$ affinely independent stations that are in saturation and lie on the same d -dimensional face F of $C_{\mathbf{L}}$, then also the other stations on F saturate.

Proof. We recall that the dimension of a face F is the maximum number of affinely independent vectors of the smallest affine space S that contains it. Thus we can express any point $L_k \in F \subset S$ as an affine combination of $d + 1$ affinely independent vectors:

$$\underline{L}_i = \sum_{k=1}^p a_i l_k, \quad \sum_{k=1}^{d+1} a_i = 1 \quad (3)$$

This holds true, in particular, when the $l_k \in \mathcal{P}(\mathbf{L})$ are the saturating stations. Then, summing the components and using relation (2), we get:

$$U_k = \sum_{i=1}^p \alpha_i U_i = \sum_{i=1}^{k+1} \alpha_i = 1$$

that proves the theorem. \square

The affine independence condition of Theorem 2 rules out degenerate cases like collinear saturating stations on the same face of dimension 2 (e.g., on a polygon). A first consequence of Theorem 2 is that the saturation of $d + 1$ vertices on a d dimensional face can extend the saturation to a larger number of stations, since also the internal stations of F must saturate. To understand the consequences of this, let us consider the characteristic polytope of a three-class network. For now we assume that only the vertices can saturate. Theorem 2 allows the following types of saturations for the vertices of a face:

1. a single vertex saturates in isolation;
2. two adjacent vertices saturate with their common edge;
3. two non-adjacent vertices saturate with the internal points of the segment that connects them;
4. three or more vertices saturate, yielding the saturation of the whole face;

The first and the last case are straightforward. For the second case, instead, we note that if there exists an edge with its vertices in saturation, then also its internal stations saturate, but they do not yield the saturation of the whole face because three or more points on a line are not affinely independent. The third case, instead, is quite complex and deals with the case where the face is non-simplicial, i.e. its vertices are not independent (e.g., a square or a pentagon in three dimensions). This degenerate case can be ruled out removing the rows of the loading matrix that are not independent or doing a triangulation [40] of the non-simplicial faces. In the latter case, each of these faces is partitioned in a simplicial complex, i.e. a set of faces with the same dimension and with independent vertices only, and the analysis can be performed considering each face of the complex as a distinct face of the characteristic polytope. However, the dependency between vertices must be traced to ensure that the saturation of a single face of the simplicial complex yields the saturation of all the other faces of the complex. For instance, a square is decomposed in a set of two triangles A and B , but the saturation

of A yields also that of B . To avoid unnecessary complexity³, in the rest of the section we will assume that every d -dimensional face with all its vertices in $\mathcal{P}_{\mathbf{L}}$ has only $d + 1$ affinely independent vertices.

We now relax the assumption that only the vertices can saturate. The following theorem investigates whether an internal station of a boundary face can saturate in isolation.

Theorem 3. Internal points of the boundary faces. If an internal station of a d -dimensional face F saturates and F has all its $d + 1$ affinely independent vertices in $\mathcal{P}_{\mathbf{L}}$, then all stations on F saturate.

Proof. Let \underline{l}_k be the internal point of a d -dimensional face F . It can be expressed as a convex combination of $d + 1$ vertices:

$$\underline{l}_k = \alpha_1 \underline{L}_1 + \alpha_2 \underline{L}_2 + \dots + \alpha_{d+1} \underline{L}_{d+1}, \quad \sum_{i=1}^{d+1} \alpha_i = 1, \quad \alpha_i \geq 0$$

Since the point is internal and all faces have been assumed to have only independent vertices, the condition $\alpha_i > 0 \forall i$ holds. Then the equality

$$U_k = \alpha_1 U_1 + \alpha_2 U_2 + \dots + \alpha_{d+1} U_{d+1} = 1, \quad \sum_{i=1}^{d+1} \alpha_i = 1, \quad \alpha_i > 0$$

implies $U_1 = \dots = U_{d+1} = 1$ since all terms U_i range in $[0, 1]$. Then we have $d + 1$ affinely independent stations in saturation and from Theorem 2 all stations on the face saturate. \square

Therefore, working with simplicial faces only we are sure that the potential bottlenecks are the vertices of the characteristic polytope.

We have not yet considered the case of the internal stations of a face with some vertices in $\Pi(\mathbf{L})$. Since the stations in $\Pi(\mathbf{L})$ are all dominated, there must exist at least $d + 1$ stations to have a d -dimensional face in saturation. While Theorem 2 still holds in this case, the extension of Theorem 3 is not easy, mainly because there is no relationship between the sum of the components of the vertices in $\Pi(\mathbf{L})$ and the utilization of the stations in $\mathcal{P}(\mathbf{L})$. However, the points of the face that are members of $\mathcal{P}(\mathbf{L})$ can be grouped in a simplicial complex applying a triangulation on their convex hull. Using the simplicial complex, instead of the whole face, we can again apply Theorem 3.

The consequences of Theorems 2 and 3 are interesting, since they prove that the behavior of the vertices and the topology of the faces are enough to identify the set of potential bottlenecks for the system. The following section discusses how additional informations on the workload let us determine the set of actual bottlenecks.

Let us finally give a brief look at the computational complexity of the convex hull problem. Solving the problem for n points in R dimensions has a variable complexity depending on R , n and on the size h of the output (i.e. the number of vertices). At present time the algorithm of choice for computing convex hulls is Quickhull [6], which performs well from 2 to 8 dimensions. The conjectured complexity is $O(n \log(h))$ when $R \leq 3$ and up to $O(n^{1+\lfloor R/2 \rfloor} / (h \lfloor R/2 \rfloor!))$ in higher dimensions. Actually, with a usual PC it is possible to compute in few minutes a convex hull of $n = 3 \cdot 10^6$ points in 6 dimensions, $n = 3 \cdot 10^4$ points in 7 dimensions or $n = 2 \cdot 10^3$ points in 8 dimensions. While this cardinality can appear excessive for real models, it must be reminded that n is the cardinality of $A = \mathcal{P}(\mathbf{L}) \cup \Pi(\mathbf{L})$ which can be very large since the cardinality of $\Pi(\mathbf{L})$ can be up to $M(2^R - 1)$. A simple way to reduce the size of $\Pi(\mathbf{L})$ is grouping the points by the strictly positive components, and then excluding the dominated stations from each group⁴. In general, the size of the reduction depends on the number of distinct natural bottlenecks and on their relative position to the origin of the loadings space. The best-case is when the model has a common bottleneck station, which trivially dominates all the others. The worst-case is when the network has R different natural bottlenecks that lie close to the origin. This is the case of a system with a strongly unbalanced distribution of the loadings. Considering this, in the worst case we can study models with tens of thousands of non-dominated stations with six customer classes, hundreds of stations with seven classes, while with eight classes the technique becomes unfeasible since only small models with tens of stations can be processed. For a comparative description of existing convex hull algorithms see [2]. While the convex hull approach to the identification of potential bottlenecks does not allow to study models with tens or even hundreds of customers classes, for models with up to seven classes it can be an interesting technique. Indeed the most interesting application of this approach, namely the definition of an optimization scheme for studying actual bottlenecks in multiclass closed networks, will be shown in the next section.

³Actually, in most cases the triangulation is not unique and this is usually sufficient to exclude the saturation of non-adjacent vertices. However, for the general case it is quite difficult to rule out the presence of convex sets with a unique triangulation between the boundary faces of the characteristic polytope.

⁴This can be formulated again as a convex hull problem in lower dimension and thus it is very efficient on the groups with up to three strictly positive components

4. Actual bottlenecks identification of multiclass closed networks

We now focus our attention on multiclass closed queueing networks. The aim of this section is to review existing techniques for the bottleneck analysis of this type of models [3, 4] and to introduce an optimization scheme based on the potential bottleneck identification scheme presented in Section 3.

In what follows we will use the same notations and we will briefly present the results described in [4]. Consider a closed queueing network with the number of customers for each class specified by the population vector $\underline{N} = \{N_1, N_2, \dots, N_R\}$. We denote $N = \sum_{r \in \mathbf{R}} N_r$ the total number of customers. We define the *population mix* as the vector $\underline{\beta} = \{\beta_1, \beta_2, \dots, \beta_R\}$ whose components are given by

$$\beta_r = \frac{N_r}{N}, \quad \sum_{r \in \mathbf{R}} \beta_r = 1$$

The loading matrix \mathbf{L} , the population vector $\underline{\beta}$ and the total number of customers N give a full specification of the model. To avoid unnecessary complexity we assume that the rows of \mathbf{L} are linearly independent. According to Theorem 2 of Section 3 this is sufficient to study the behavior of all the bottlenecks. Because of the complexity of describing multiclass queueing networks, we concentrate on the asymptotic behavior of the system; then we assume N to be always a large fixed number. We denote U_{ij} the class- j utilization of station i when we keep constant the population mix $\underline{\beta}$ and let the total number of customer N tend to infinity. We call this type of growth *proportional growth*, opposed to the *unbalanced growth* where only a single population class is increased.

Consider the case of a network with R classes in which the conditions

$$L_{rr} > L_{ir}, \quad \forall r \forall i \quad i \neq r \quad (4)$$

and

$$L_{rr} > (L_{1r}, \dots, L_{ir}, \dots, L_{Rr}) > L_{j1}, \quad \forall r \forall i \quad i \neq r, \quad j = R+1, \dots, M \quad (5)$$

hold. Condition (4) simply states that distinct classes have distinct natural bottlenecks which are the first R stations of the loading matrix. This avoids the case where two or more classes have the same station as natural bottleneck, which can be addressed as an intermediate case between the one we are considering and that of a common bottleneck model [3]. This also implies that only networks with $M \geq R$ will be considered. Condition (5), instead, requires that the set of natural bottlenecks dominate all other stations, which therefore have no chance of saturating. Thus, for now, we do not consider networks with masked-off stations.

The simplest inspection we can take to analyze the system is to study the utilization of the potential bottlenecks as the components of $\underline{\beta}$ vary, keeping fixed the population N . We refer to the set of all feasible population mixes as the β space. In general this inspection leads to the evidence that in several cases distinct resources saturate together. Let us call $\mathbf{B} = \mathbf{B}(\underline{\beta})$ the set of bottleneck stations for a given mix $\underline{\beta}$ ⁵. In addition, limited changes in the components of $\underline{\beta}$ may affect the set of bottleneck stations, while others do not, thus suggesting that the β space can be partitioned in adjacent regions, the *saturation sectors*, each characterized by a different set of saturating stations.

While the general rules that describe the migration of the bottlenecks have been identified, they are quite difficult to prove. The behavior of the bottleneck set has been conjectured in [4] as follows:

1. when the population mix $\underline{\beta}$ yields the saturation of only one station, the class- r asymptotic utilization of the bottleneck is given by $U_{ir}(\underline{\beta}) = \beta_r$
2. when the population mix $\underline{\beta}^*$ yields the saturation of a set \mathbf{B} of stations, their asymptotic mean queue lengths grow all to infinity while satisfying the relation

$$\lim_{N \rightarrow \infty} \frac{n_t(N)}{\sum_{b \in \mathbf{B}} n_b(N)} = \gamma_t(\underline{\beta}^*) \quad \forall t \in \mathbf{B}, \quad \sum_t \gamma_t = 1 \quad (6)$$

3. when the population mix $\underline{\beta}^*$ corresponds to the edge between a saturation sector where station i saturates and a sector where it does not saturate, then $\gamma_i(\underline{\beta}^*) = 0$
4. in all cases the non-bottleneck stations have limited queue lengths

⁵The indices b and t will be assumed to range between 1 and the cardinality $B \leq R$ of \mathbf{B}

The *bottleneck relative strengths* γ represent the relative speed of the queue length growth for a bottleneck station compared to that of the other bottlenecks. Henceforth we will denote $\underline{\beta}^*$ a population mix within a common saturation sector. These assumptions are sufficient to derive from the MVA equations of the per-class utilization the following expression of the population mix as a function of the γ 's:

$$\beta_r^* = \frac{U_{ir}^*(\underline{\beta}^*)}{L_{ir}} \sum_{b \in \mathbf{B}} \gamma_b L_{br}, \quad i \in \mathbf{B} \quad (7)$$

Since the edges of the saturations sector are identified by values of the γ 's equal to zero, their coordinates in the β space can be found from equation (7) if the per-class utilization $U_{ir}(\underline{\beta}^*)$ are known. We therefore use the following system of linear equations based on the basic relation (2) to derive the information:

$$\begin{aligned} \sum_{r \in \mathbf{R}} U_{br}(\underline{\beta}^*) &= 1, \quad \forall b \in \mathbf{B} \\ U_{br} &= \frac{L_{br}}{L_{rr}} \cdot U_{rr}, \quad \forall b \in \mathbf{B}, \quad r \in \mathbf{R} \end{aligned} \quad (8)$$

This is at most a system of R^2 equations in R^2 variables, because no more than R independent stations can saturate together [4]. A solution using a Gaussian-Elimination algorithm can be found in a time complexity of $O(R^6)$ which is independent from the number of stations. However, in the general case of a system with masked-off stations the cardinality of \mathbf{M} can dramatically affect the number of possible saturation sectors; then, the number of linear systems we must solve to perform the bottleneck analysis can be of the order of 10^{10} systems for a system with 100 non-dominated stations and 7 classes. We will discuss in the next paragraphs a possible approach to reduce the complexity of this problem.

Using system (8) we can find the vertices of the saturation sector considering the R cases where there exists a station b such that $\gamma_b \neq 0, \gamma_t = 0 \quad \forall t \neq b$. The regions in which $R' < R$ stations saturate are called the *partial saturation sectors*, opposed to the *global saturation sector* where the maximum number of stations saturates. Several partial saturation sectors can be found by analyzing the system when there is a *segregated class*, i.e. a class for which the number of customers is equal to zero. Considering that the existence of a saturation sector for R' stations requires the existence of the partial saturation sector for all possible subsets of $R' - 1$ stations, it is possible to use the information gathered when analyzing the system with one segregated class to exclude some infeasible partial saturation sectors when looking at sectors with two bottlenecks. A similar argument applies for saturation sectors with more than two bottleneck. Thus, searching for the vertices of the saturation sectors in the case of a segregated class and then look for sectors with more saturating stations can reduce the number of systems to be solved.

While this optimization can be sufficient to perform an analysis of a complex system without masked-off stations, it is reasonable that real models can involve a large number of masked-off stations. In this case the analysis is quite similar; the main difference is that also masked-off stations can be involved in the saturation and so more than a single global saturation sector can exist. This fact can dramatically increase the number of systems (8) to be solved. Even a small system with 4 classes and 10 non-dominated stations can have over 600 distinct saturation sectors; a bigger system with 7 classes and 500 non-dominated stations can have up to $3 \cdot 10^{15}$ different saturation sectors.

Since the dimensionality of this problem makes intractable the bottleneck analysis even for medium-sized networks, an effective optimization scheme is clearly required. We sketch a simple optimization based on the results of Section 3. If one knows the vertices and the topology of the faces of the characteristic polytope, the set of all possible saturating stations is easily determined. Hence, this minimizes the number of linear systems to be solved. Moreover this allows to compute easily whether two saturation sectors are adjacent and this information could be used to minimize the number of coordinates β_r to be computed.

To show the quality of the optimization we show some preliminary results on 100 randomly generated multiclass closed models. Table 1 shows the effectiveness of the optimization scheme: even for small models we have a strong reduction in the number of systems to be solved. We also tried to estimate the entity of the reduction on a large problem with 100.000 stations and 6 classes: the new optimization scheme required $4 \cdot 10^6$ linear systems to be solved, which can be easily computed in few minutes on a typical PC, while determining the number of systems to be computed with the old strategy took to computational effort to be computed. Therefore, it is conceivable the entity of the reduction increases with the complexity of the model.

5. Actual bottleneck identification of multiclass open networks

In this section we consider the bottleneck analysis of multiclass open queueing networks. Usually the workload of these models is specified in terms of the arrival rate λ_r for each class r . We denote $\underline{\lambda} = \{\lambda_1, \lambda_2, \dots, \lambda_R\}$ the arrival rate vector. Using the job flow balance assumption [18] this means that in a time period T the number of completed jobs C_r equals

M	$R = 3$			$R = 4$		
	Old	New	Δ	Old	New	Δ
10	108.72	42.50	-60.56%	297.78	114.82	-61.44%
100	1914.88	173.06	-90.96%	33299.83	1123.02	-99.66%
1000	9712.82	390.56	-95.98%	384068.5	3954.01	-98.97%

Table 1. Mean number of linear systems to be solved for 100 randomly generated multiclass closed models with 3 and 4 classes for the cases $M = 10$, $M = 100$ and $M = 1000$ stations.

the number of arrived jobs N_r . Thus, using the basic relation $C_r = \lambda_r T$ we can define the population mixes β_r for open models as:

$$\beta_r = \frac{N_r}{N} = \frac{C_r}{C} = \frac{\lambda_r}{\lambda}, \quad \sum_{r \in \mathbf{R}} \beta_r = 1 \quad (9)$$

where $\lambda = \sum_{r \in \mathbf{R}} \lambda_r$ is the global arrival rate to the system, $C = \sum_{r \in \mathbf{R}} C_r$ is the global number of completed jobs, and $N = \sum_{r \in \mathbf{R}} N_r$ is the global number of jobs arrived to the system during the time interval T .

In this section we limit the bottleneck analysis to systems where the workload does not grow beyond the value where a station becomes saturated.

While a description of the behavior of open models in the β space is possible, we can take advantage of the direct relation between the input workload $\underline{\lambda}$ and the utilizations $U_i(\underline{\lambda})$. Thus, it is more natural to describe the behavior of the system in the space of the λ_i 's rather than in the β space. We will discuss later in the section the relation between these two spaces. Looking at the bottleneck analysis of closed systems a number of questions on the bottleneck analysis of open models naturally arise. First, we are interested in identifying bottlenecks as a function of the arrival rate vector $\underline{\lambda}$. Second, we want to understand whether partial and global saturation sectors exists also in open models. Finally, we should consider the computational complexity of the procedure.

The identification of the bottlenecks in the λ space is quite simple. Let us consider the feasible region in the λ space defined by the M constraints

$$U_i(\underline{\lambda}) = \lambda_r \cdot L_{ir} \leq 1 \quad \forall i \in \mathbf{M} \quad (10)$$

It is clear that a station i becomes saturated if the equality $U_i(\underline{\lambda}) = 1$ holds, i.e. the constraint is active. Therefore for a given $\underline{\lambda}$ the set of bottleneck stations is simply the set of stations whose utilization constraint is active. As from basic optimization theory, any set of active constraints defines a face F of the convex hull of (10). Thus we can associate to the set of bottleneck stations a face F of the convex hull of (10). This operation can be carried out using any software for solving the convex hull problem (e.g., Qhull [6]). Moreover, there is a correspondence between the faces of the characteristic polytope defined in Section 3 and the faces of the convex hull P of system (10), since a set of stations that saturate together must lie on the same face both in P and in the characteristic polytope $C_{\mathbf{L}}$.

A first interesting consequence of working in the λ space is that, if the rows of the loading matrix are all linearly independent, then only a point in the β space can yield a global saturation of a set of stations. This because in R dimensions a set of R independent hyperplanes $U_i(\underline{\lambda})$ can intersect in at most a single point $\lambda^* = (\lambda_1^*, \dots, \lambda_R^*)$, and from (9) we have that

$$\frac{\beta_k}{\beta_j} = \frac{\lambda_k}{\lambda_j}, \quad \sum_{r \in \mathbf{R}} \beta_r = 1 \quad (11)$$

which implies that a single point in the λ space is mapped to a single point in the β space. This proves that a global saturation cannot occur in a region of the β space with dimensions greater than that of a point. A similar argument can be applied to partial saturation sectors with $R' < R$ saturating stations, which are the intersection of R' hyperplanes in the λ space: under the assumption of independence for the rows of the loading matrix \mathbf{L} , this region has dimension R' . For instance, a three class network can have a partial saturation sector with two saturating stations on a line in the λ space. In the β space this corresponds again to a line, since according to relation (11) a constant ratio between λ_j and λ_k yields a constant ratio between β_j and β_k . A similar argument applies to faces in higher dimensions, since a change of coordinates to the β space is simply a projection on the hyperplane $\sum_{r \in \mathbf{R}} \beta_r = 1$ which is equivalent to $\sum_{r \in \mathbf{R}} \lambda_r = \lambda$ in the λ space: hence a decrease on the dimension of the partial saturation sector can happen only if the face is orthogonal to $\sum_{r \in \mathbf{R}} \lambda_r = \lambda$, but this is impossible⁶. According to this, the information gathered in the λ space and the ones on the β space are equivalent and a characterization of the convex hull of (10) is enough to perform a full bottleneck analysis.

The computational complexity of determining the convex hull of given a set of hyperplanes is the same of computing it

⁶This is due to the fact that any active constraint $L_{i1}\lambda_1 + \dots + L_{iR}\lambda_R = 1$ is orthogonal to $\lambda_1 + \dots + \lambda_R = \lambda$ only if $L_{i1} + \dots + L_{iR} + \lambda = 0$ which has no feasible solutions.

from a set of points. Hence, in the case of open models we have no gains in considering the characteristic polytope instead of the convex hull in the β space. Therefore determining the convex hull in the λ space is preferable.

As a concluding remark, we note that results of the bottleneck analysis of open models are more general than those for closed models, since they give full information about the bottlenecks under all possible workloads. Suppose that the utilization of bottleneck stations is $U_b = 0.60$, then applying the arguments described in this section to the system

$$U_i(\underline{\lambda}) = \lambda_r \cdot L_{ir} \leq 0.6 \quad \forall i \in \mathbf{M} \quad (12)$$

allows us to prove that the saturating stations are the same of the original system (10) when considering the same population mix $\underline{\beta}$ of system (12). Indeed the value 0.6 is just a scale factor for the dimension of the convex hull and does not affect its projection on $\sum_{r \in \mathbf{R}} \lambda_r = \lambda$. Therefore the analysis of the 'beta' space derived from the convex hull of (10) is enough to fully characterize the behavior of the bottlenecks under all possible workloads.

6. Summary and conclusions

In this paper we have presented a technique for the identification of bottlenecks, either potential and actual, in very large models. We investigated the relationship between bottleneck analysis and the convex hull problem, showing that for open models these problems are equivalent. We also introduced an effective optimization scheme, based on usual algorithms for solving the convex hull problem, that allows bottleneck analysis of closed queueing networks in a very efficient way. Our preliminary analysis showed that very large closed models with up to several tens of thousands stations and six classes can be solved.

References

- [1] I. F. Akyildiz. Exact product form solution for queueing networks with blocking. *IEEE Trans. on Computers*, 1987.
- [2] D. Avis, D. Bremner. How good are convex hull algorithms? In *Symp. on Computational Geometry*, 20–28, 1995.
- [3] G. Balbo, G. Serazzi. Asymptotic analysis of multiclass queueing networks: common bottlenecks. *Performance Evaluation*, 26:51–72, 1996.
- [4] G. Balbo, G. Serazzi. Asymptotic analysis of multiclass queueing networks: multiple bottlenecks. *Performance Evaluation*, 30:115–152, 1997.
- [5] S. Balsamo. Product form queueing networks. In G. Haring C. Lindemann, and M. Reiser eds., *Performance Evaluation: Origins and Directions - LNCS 1769*, 377–401. Springer, 2000.
- [6] C. B. Barber, D. P. Dobkin, and H. Huhdanpaa. The quickhull algorithm for convex hulls. *ACM Trans. on Mathematical Software*, 22(4):469–483, 1996.
- [7] Y. Bard. Some extensions to multiclass queueing network analysis. In E. Gelemb A. Butrimenko, and M. Arato eds., In *Proc. Fourth Intl Symp. On Modeling and Perf. Eval. of Computer Systems*, 51–62, 1979.
- [8] F. Baskett, K. Mani Chandy, R.R. Muntz, and F. G. Palacios. Open, closed and mixed networks of queues with different classes of customers. *Journal of the ACM*, 22(2):248–260, April 1975.
- [9] R. M. Bryant, A.E. Krzesinski, M. S. Lakshmi, and K. M. Chandy. The MVA priority approximation. *ACM Trans. on Computer Systems*, 2(4):335–359, 1984.
- [10] J. P. Buzen. Computational algorithm for closed queueing networks with exponential servers. *Communications of the ACM*, 16(9):527–531, 1973.
- [11] K.M. Chandy, C. Sauer. Computational algorithms for product form queueing networks. *Communications of the ACM*, 23(10):573–583, 1980.
- [12] K.M. Chandy, D. Neuse. Linearizer: A heuristic algorithm for queueing network models of computing systems. *Communications of the ACM*, 25(2):126–134, 1982.
- [13] K.M. Chandy, U. Herzog, and L. Woo. Parametric analysis of queueing networks. *IBM Journal of Research and Development*, 1(1):36–42, 1975.
- [14] W. M. Chow. Approximations for large scale closed queueing networks. *Performance Evaluation*, 3(1):1–12, 1983.
- [15] A.E. Conway, N.D. Georganas. RECAL: A new efficient algorithm for the exact analysis of multiple-chain closed queueing networks. *Journal of the ACM*, 33(4):768–791, 1986.
- [16] A.E. Conway, E. de Sousa e Silva, and S.S. Lavenberg. Mean value analysis by chain of product form queueing networks. *IEEE Trans. on Computers*, 38(3):432–442, 1989.
- [17] P. Cremonesi, P. J. Schweitzer, and G. Serazzi. A unifying framework for the approximate solution of closed multiclass queueing networks. *IEEE Trans. on Computers*, 51(12):1423–1434, 2002.
- [18] P. J. Denning, J. P. Buzen. The operational analysis of queueing network models. *ACM Computing Surveys*, 10(3):225–261, 1978.
- [19] L.W. Dowdy, B. M. Carlson, A. T. Krantz, and S. K. Tripathi. Single-class bounds of multi-class queueing networks. *Journal of the ACM*, 39(1):188–213, 1992.
- [20] E. de Souza e Silva, S. S. Lavenberg. Calculating joint queue-length distributions in product-form queueing networks. *Journal of the ACM*, 36(1):194–207, 1989.
- [21] D. L. Eager, J. N. Lipscomb. The AMVA priority approximation. *Performance Evaluation*, 8:173–193, 1988.
- [22] D. L. Eager, K. C. Sevcik. Bound hierarchies for multiple-class queueing networks. *Journal of the ACM*, 33(1):179–206, 1986.

- [23] J.B. Goodman, W. A. Massey. The non-ergodic jackson network. *Journal of Applied Probability*, 21:860–869, 1984.
- [24] W. Gordon, G. Newell. Closed queueing systems with exponential servers. *Operations Research*, 15(2):254–265, 1967.
- [25] P. Harrison, S. Coury. On the asymptotic behaviour of closed multiclass queueing networks. *Performance Evaluation*, 47(2-3):131–138, 2002.
- [26] K. Hoyme, S. Bruell, P. Afshari, and R. Kain. A tree-structured mean value analysis algorithm. *ACM Trans. on Computer Systems*, 4(2):178–185, 1986.
- [27] Intel corporation. Managing e-business solutions in the distributed enterprise. White Paper, 2001. http://www.intel.com/business/bss/infrastructure/managing/intel_osc.pdf.
- [28] J. Jackson. Jobshop-like queueing systems. *Management Science*, 10(1):131–142, 1963.
- [29] T. Kerola. The composite bound method for computing throughput bounds in multiple class environments. *Performance Evaluation*, 6, 1:1–9, 1986.
- [30] H. Kobayashi. A computational algorithm for queue distributions via polya theory of enumeration. In M. Butrimenko, M. Arato eds., In *Proc. of the 4th Intl Symposium on Modelling and Perf. Eval. of Computer Systems - vol. 1*, 79–88. North-Holland, 1979.
- [31] J. Kriz. Throughput bounds for closed queueing networks. *Performance Evaluation*, 4:1–10, 1984.
- [32] S. Lam, Y. Lien. A tree convolution algorithm for the solution of queueing networks. *Communications of the ACM*, 26(3):203–215, 1983.
- [33] S.S. Lavenberg, M. Reiser. Stationary state probabilities at arrival instants for closed queueing networks with multiple types of customers. *Journal of Applied Probability*, 17:1048–1061, 1980.
- [34] J.Luthi, G. Haring. Bottleneck analysis for computer and communication systems with workload variabilities and uncertainties. In I. Troch and F. Breitenecker eds., *Proc. of 2nd Intl Symposium on Mathematical Modelling*, 525–534, 1997.
- [35] J. McKenna, D. Mitra and K.G. Ramakishnan,. A class of closed markovian queueing networks: Integral representations, asymptotic expansions, generalizations. *The Bell System Technical Journal*, 60(5):599–641, 1981.
- [36] I. Mitrani, K.S. Sevcik. The distribution of queueing network states at input and output instants. *Journal of the ACM*, 28(2):358–371, 1981.
- [37] F.R. Moore. Computational model of a closed queueing network with exponential servers. *IBM Journal of Research and Development*, 16(6):567–572, 1972.
- [38] M. F. Neuts. *Matrix-Geometric Solutions in Stochastic Models: An Algorithmic Approach*. Johns Hopkins University Press, Baltimore, 1981.
- [39] R. O. Onvural, H. G. Perros. Some equivalencies on closed exponential queueing networks with blocking. *Performance Evaluation*, 9:111–118, 1989.
- [40] F. P. Preparata, M. Shamos. *Computational Geometry*. Springer-Verlag, 1985.
- [41] K.G. Ramakishnan, and D. Mitra. PANACEA: An integrated set of tools for performance analysis in *Modelling Techniques and Tools for Computer Evaluations* R. Puigjaner and D. Potier eds., *Plenum Press*, NY, 1988, pp. 25– 40.
- [42] M. Reiser, H. Kobayashi. Queueing networks with multiple closed chains: Theory and computational algorithms. *IBM Journal of Research and Development*, 19(3):283–294, 1975.
- [43] M. Reiser, S.S. Lavenberg. Mean-value analysis of close multichain queueing networks. *Journal of the ACM*, 27(2):312–322, 1980.
- [44] C.H. Sauer, S. Tucci. The tree MVA algorithm. *Performance Evaluation*, 5(3):187–195, 1985.
- [45] P. J. Schweitzer. A survey of mean value analysis, its generalizations, and applications, for networks of queues. In *Proc. Second Intl Workshop Netherlands Natl Network for the Math. on Operations Research*, 1991.
- [46] P. J. Schweitzer. Approximate analysis of multiclass closed queueing networks of queues. In *Proc. Intl Conf. Stochastic Control and Optimization*, 1979.
- [47] P. J. Schweitzer, G. Serazzi, and M. Broglia. A survey of bottleneck analysis in closed network of queues. In R. D. Nelson, L. Donatiello eds., *Perf. Eval. of Computer and Communication Systems, Joint Tutorial Papers of Performance '93 and Sigmetrics '93*, 491–508. Springer-Verlag, 1993.
- [48] M. S. Squillante. Magic: A computer performance modeling tool based on matrix-geometric techniques. In *Fifth Intl Conf. on Modelling Techniques and Tools for Computer Perf. Eval.*, 411–425, 1991.
- [49] J. Zhorjan, E. Wong. The solution of separable queueing network models using mean value analysis. *Performance Evaluation*, 8(4):255–270, August 1988.
- [50] J. Zhorjan, K.C. Sevcik, D. L. Eager, and B. Galler. Balanced job bound analysis of queueing networks. *Communications of the ACM*, 25(2):134–141, 1982.
- [51] J. Zhorjan, D.L. Eager, and H. M. Sweillam. Accuracy, speed and convergence of approximate mean value analysis. *Performance Evaluation*, 8(4):255–270, 1988.