Geometric Bounds: a Non-Iterative Analysis Technique for Closed Queueing Networks

Giuliano Casale, *Member, IEEE,* Richard R. Muntz, *Fellow, IEEE,* Giuseppe Serazzi, *Member, IEEE*

***Abstract*— We propose the *Geometric Bounds* (GB), a new family of fast and accurate non-iterative bounds on closed queueing network performance metrics that can be used in the on-line optimization of distributed applications. Compared to state-of-the-art techniques such as the Balanced Job Bounds (BJB), the GB achieve higher accuracy at similar computational costs, limiting the worst-case bounding error typically within $5\%-13\%$ when for the BJB it is usually in the range $15\%-35\%$. Optimization problems that are solved with the GB bounds return solutions that are much closer to the global optimum than with existing bounds.**

We also show that the GB technique generalizes as an accurate approximation to closed *fork-join* networks commonly used in disk, parallel and database models, thus extending the applicability of the method beyond the optimization of basic product-form networks.

***Index Terms*— Non-iterative bounds, performance optimization, closed queueing networks, fork-join systems.**

I. Introduction

THE ever increasing complexity of today's large distributed systems requires the availability of self-optimizing techniques for their management. Complexity stems from features such as the large number and the heterogeneity of hardware and software components, the intensity of the load, the high number of interconnected networks, the multi-layered architecture of applications, and the unpredictable fluctuations of traffic requests. In this scenario, self-managing systems are rapidly emerging [1], [2]. These systems dynamically adjust their configuration to continuously meet the system service requirements in presence of workloads with highly variable and unpredictable request patterns. Service requirements apply to different architectural layers and are expressed in terms of Quality of Service metrics, e.g., system throughput and response time, service availability, access failure rate, packet delay and drop rates. In general, they should be met continuously, both individually and in combination. Under these conditions, a system must be able to self-configure and take self-optimization decisions based on rules that search over the space of all the feasible parameter values. Given a performance model of the system, possibly hundreds of thousands of configurations can be evaluated by nonlinear programming methods [3] to find the best-possible configuration decision. The accuracy of the solution is also a critical factor, especially when violations of Service Level Agreements (SLA) result in economic penalties. Efficient yet accurate solution techniques are thus necessary.

G.Casale (casale@cs.wm.edu) is with the College of William and Mary, Computer Science Department, 140 McGlothlin-Street Hall, 23187-8795 Williamsburg, VA, US.

R.R.Muntz (muntz@cs.ucla.edu) is with UCLA, Computer Science Department, 3277A Boelter Hall, 90095-1596 Los Angeles, CA, US.

G.Serazzi (giuseppe.serazzi@polimi.it) is with the Politecnico di Milano, Dip. di Elettronica ed Informazione, via Ponzio 34/5, I-20133 Milan, Italy.

Analytic queueing network models are often used in performance optimization because of the robustness and efficiency of the available solution algorithms [4]. In particular, closed *product-form* queueing networks [5] are the most popular stochastic models for performance evaluation, thanks to the simple expression of their steady-state probability which assumes the form of an algebraic product and avoids the numerical solution of the underlying Markov chain. However, the computational complexity of exact solution techniques, even basic single class models, makes them infeasible when a very large number of problem instances must be solved in a limited amount of time. If estimates of the performance indexes, rather than their exact values, are sufficient to satisfy the requirements of the performance evaluation study, then approximate techniques may be adopted. Iterative local approximations [6] are more efficient than exact methods, but are usually much slower than single-step bounding techniques [7], [8], [9], [10], [11], [12]. On the other hand, the accuracy of single-step bounding techniques may be significantly worse.

In this paper, we introduce a new bounding technique that shifts the trade-off between computational costs and result accuracy towards the latter at a small increase of the former. The *Geometric Bounds* (GB) define a single-step bounding technique for closed product-form networks that provides inexpensive yet accurate results regardless of the dimensions of the system and of the workload. The method bounds performance indexes such as queue-lengths, throughput, utilizations and response times, which are among the most commonly used metrics in performance analysis and optimization. The GB bounds are derived by describing the queue-lengths with a geometric sequence of terms related to resource utilizations and are more accurate than any other non-iterative method, in particular they improve over the widely-used Balanced Job Bounds (BJB) [12]. We also address the known accuracy loss problem when there are delay servers in the network by introducing the *Geometric Square-root Bounds* (GSB). GB and GSB are validated extensively using a large number of random problem instances as well as stress cases proposed in the literature for the evaluation of other bounding techniques. We also show that the GB and the GSB bounds provide very good accuracy in the critical case of strongly unbalanced networks, where other known bounds are loose. This case is extremely important in real applications, where a dominant performance bottleneck can cause major performance slowdowns and therefore model accuracy for this case is desirable.

It is well-known that real systems can have features that significantly violate the assumptions of product-form models. In this case, more complex *non-product-form* networks are often considered. To improve the applicability of our method outside product-form modeling assumptions, we propose an extension of GB to networks with fork-join subsystems [13]. Recent work has noted the importance of fork-join networks for the performance optimization of parallel systems and in particular for tuning disk

arrays and distributed storage systems [14], [15], and for database (DB) modeling [16]. Due to the difficulty of obtaining a general closed-form expression of the equilibrium state probabilities, the focus of fork-join model research is on approximation techniques [17]. In particular, the increasing demand for inexpensive techniques which may be applied to the on-line performance tuning of storage systems [18] has lead to the extension of classical mean value analysis (MVA) techniques developed for product-form networks [19]. The proposed extension of the GB technique shows an accuracy that is comparable with that of existing MVA-based approximations of fork-join models, but at much lower computational costs. It also improves over established BJB-based approximation methods. We show in a case study that the impact of increased approximation accuracy can be very relevant for performance optimization of fork-join models.

The paper is organized as follows. In Section II we give background. Section III describes the GB and GSB bounds. Section IV presents the results of extensive accuracy validations. The extension of the GB technique to non-product-form fork-join networks is discussed in Section V. Experiments showing the impact of our bounds in system optimization are discussed in Section VI. Conclusions are drawn in Section VII. The final Appendix summarizes model parameters used in experiments.

II. Background

We consider a system which can be modeled as a network where N jobs cyclically visit a fixed number of resources. We model the first M resources as *queues*, indexed by $i = 1, \ldots, M$, where jobs may wait in line due to limited server capacity; the remaining resources are *delay servers* where jobs immediately receive service without queueing. For computational efficiency, we represent the system as a product-form[1] queueing network [5], in which servers have fixed mean service rate and the network is populated by N statistically indistinguishable jobs (single class model). A comprehensive introduction to product-form models and related background can be found in [20]; we also point to [22] for a glossary of standard queueing network terminology.

The parameter S_i is the average service time at queue i, $1 \leq i \leq M$; similarly, Z_j is the service time at a delay server j. We define V_i to be the average number of visits of a job to a resource i for each cycle in the system. We remark that it is known from long time [8] that the steady-state performance of a product-form network depends only on N, on the set of queue *service demands* $L_i = S_i V_i$, $1 \leq i \leq M$, and on the total delay $Z = \sum_j Z_j V_j$.

Throughout the paper, we focus on the following synthetic *mean* performance indexes which summarize the behavior of the queueing network at steady-state:

- the server *utilization* $U_i(N)$, i.e., the fraction of time in which the server of queue i is busy;
- the cumulative mean *waiting time* $W_i(N)$ experienced by a job while waiting and subsequently receiving service at queue i during the V_i visits;
- the *queue-length* $Q_i(N)$, i.e., the mean number of jobs waiting or receiving service at queue i;
- the system *throughput* $X(N)$, i.e., the mean completion rate of jobs, and the system *response time* $R(N) = \sum_{i=1}^{M} W_i(N)$, both measured at an arbitrary reference point in the network, which are synthetic indexes of network performance.

[1]Other models are possible, but the lack of exact solution formulas often makes the evaluation of these "non-product-form" networks prohibitively expensive (e.g., state space explosion) or inaccurate. An exception is the class of fork-join networks that can be approximated accurately.

TABLE I
Notation: Model Parameters and Performance Indices

M	number of queues in the network
N	number of jobs in the network
V_i	mean number of visits of jobs at queue i, $1 \leq i \leq M$
S_i	mean service time [s] of jobs at queue i, $1 \leq i \leq M$
$L_i = S_i V_i$	mean service demand [s] of jobs at queue i, $1 \leq i \leq M$
Z	total delay [s] imposed by the delay servers
$X(N)$	mean throughput of the network [job/s]
$U_i(N)$	utilization [%] of the server of queue i, $1 \leq i \leq M$
$Q_i(N)$	mean queue-length [jobs] at queue i, $1 \leq i \leq M$

TABLE II
Bounding Methods Notation

$L = \sum_{i=1}^{M} L_i$	cumulative service demand [s]
$L_{\max} = \max_{1 \leq i \leq M} L_i$	maximum service demand [s]
$X_{\max} = L_{\max}^{-1}$	maximum achievable throughput [job/s]

These quantities are usually computed with the Mean Value Analysis (MVA) algorithm [21] as

$$U_i(N) = L_i \, X(N), \quad 1 \leq i \leq M, \tag{1}$$

$$W_i(N) = L_i \left(1 + Q_i(N-1)\right), \quad 1 \leq i \leq M, \tag{2}$$

$$X(N) = N / \left(Z + R(N)\right) = N / \Big(Z + L + \sum_{i=1}^{M} L_i Q_i(N-1)\Big), \tag{3}$$

$$Q_i(N) = X(N) R_i(N) = U_i(N) \left(1 + Q_i(N-1)\right), \quad 1 \leq i \leq M, \tag{4}$$

where $L = \sum_{i=1}^{M} L_i$. The above MVA formulas define a recursive algorithm in terms of the queue-lengths $Q_i(N-1)$ which grows in complexity as $O(MN)$ time and $O(M)$ space. Table I summarizes the main notation given in this section.

A. Non-Iterative Bounding Techniques

The aim of the bounding techniques reviewed in this subsection is to provide *non-iterative* lower and upper bounds on (1)-(4), having computational cost *independent* of the particular value of the population size N, which is the most important source of computational complexity in the MVA algorithm. Since (1)-(4) allow to derive bounds on $R(N)$, $W_i(N)$ and $U_i(N)$ from bounds on $X(N)$ and $Q_i(N)$, the focus is on developing bounds for the last two quantities. Henceforth, we use the notation M^+ and M^- to indicate respectively upper and lower bounds on a performance metric M, e.g., X^+ and X^- denote upper and lower throughput bounds, respectively; similarly, Q_i^+ and Q_i^- denote queue-length upper and lower bounds.

Table II reports bound notation used in the rest of the paper; the best available non-iterative bounds, i.e., the ABA [11], BJB [12], [10] and PB [9] bounds, are summarized in Table III. The following remarks clarify the contents of the tables:

- the ABA bounds provide accurate results only when the network is lightly loaded or heavily congested. The saturation condition $X(N) \leq X_{\max} = L_{\max}^{-1}$ applies to all upper bounds and in experiments we always enforced it also for the BJB, PB, GB and GSB bounds.

TABLE III
NON-ITERATIVE BOUNDING METHODS FOR CLOSED QUEUEING NETWORKS

ABA	$N/\left(Z+L\,N\right) \leq X(N) \leq \min\left(N/\left(Z+L\right), X_{\max}\right)$
BJB	$N/\left(Z+L+L_{\max}(N-1-ZX^{-})\right) \leq X(N) \leq N/\left(Z+L+M^{-1}L(N-1-ZX^{+})\right)$
PB	$N/\left(Z+L+\frac{\sum_{i=1}^{M} L_i^N (N-1-ZX^{-})}{\sum_{j=1}^{M} L_j^{N-1}}\right) \leq X(N) \leq N/\left(Z+L+\frac{\sum_{i=1}^{M} L_i^2 (N-1-ZX^{+})}{\sum_{j=1}^{M} L_j}\right)$

- The BJB bounds always offer greater accuracy than the ABA bounds. Further, the bounds hold true for any X^+ such that $X(N-1) \leq X^+ \leq X_{\max}$ and X^- such that $0 \leq X^- \leq X(N-1)$. In particular, the simplest BJB are obtained by setting $X^+ = X_{\max}$ and $X^- = 0$.
- the PB bounds differ from the other bounds in that they consider each individual value of the L_i's and thus are always more accurate than ABA and BJB. The increase of accuracy requires a small increase of computational cost due to some additional exponentiations. Similarly to the BJB, setting $X^+ = X_{\max}$ and $X^- = 0$ provides the simplest PB bounds.

We remark that accurate *iterative* bounds exist [23], [9], [24], but they are not considered in the paper because they require considerably higher computational costs than non-iterative bounds. As we remark in Section IV, our Geometric Bounds yet provide accuracy levels similar to those of iterative methods, but at a fraction of the cost.

III. GEOMETRIC BOUNDS

Our approach for developing new accurate non-iterative bounds consists of the following basic ideas. We derive bounds on queue-lengths by recursively expanding (4) as

$$Q_i(N) = U_i(N) + U_i(N)U_i(N-1) + U_i(N)U_i(N-1)U_i(N-2) + \ldots + U_i(N)U_i(N-1)U_i(N-2)\cdots U_i(2)U_i(1). \quad (5)$$

Noting that (5) is qualitatively similar to a geometric sum

$$y + y^2 + y^3 + \ldots + y^N, \quad (6)$$

we prove that it is always possible to compute appropriate y's such that (6) provides either a upper or a lower bound on $Q_i(N)$. This is fundamental to remove the iterative structure of the MVA recursion, since (6) can be computed non-iteratively as

$$y + y^2 + \ldots + y^N = \frac{y}{1-y} - \frac{y^{N+1}}{1-y}. \quad (7)$$

The computation of bounds using the last formula provides a strong computational gain compared to iterative schemes. That is, the computational cost of computing queue-length bounds is *independent* of the population size N, since this appears in the right hand side of (7) only as an exponent.

At the end of the section, we show how the bounds on $Q_i(N)$ also define bounds on the throughput $X(N)$. It can be easily verified that all the techniques presented in the next section have requirements that are population-independent and their computational costs grow as $O(M)$ both in time and space.

A. Queue-Length Bounds

This section derives the non-iterative expressions (7) of the GB queue-length bounds.

Theorem 1: The queue-length $Q_i(N)$ is bounded from below by

$$Q^-_{i,\mathrm{GB}}(N) = \begin{cases} \dfrac{y_i(N)}{1-y_i(N)} - \dfrac{y_i(N)^{N+1}}{1-y_i(N)}, & \text{if } L_i < L_{\max}; \\ \dfrac{1}{m_{\max}}\Big(N - ZX^+ - \displaystyle\sum_{k:L_k<L_{\max}} Q^+_{k,\mathrm{GB}}(N)\Big), & \\ & \text{if } L_i = L_{\max}; \end{cases} \quad (8)$$

for any X^+ such that $X(N) \leq X^+ \leq X_{\max}$ and where

$$y_i(N) = L_i\, N/(Z + L + L_{\max}N), \quad (9)$$

is the ratio y of the underlying geometric sum (7), the constant $m_{\max}$ is the number of queues with service demand $L_{\max}$, and $Q^+_{k,\mathrm{GB}}(N)$ is the GB upper bound in Theorem 2 for $L_i < L_{\max}$.

Proof: Case $L_i = L_{\max}$. This case follows by Little's Law [25] observing that

$$\sum_{i:L_i=L_{\max}} Q_i(N) = m_{\max}Q_i(N) = N - ZX(N) - \sum_{i:L_i<L_{\max}} Q_i(N),$$

and then replacing $X(N)$ and $Q_i(N)$ in the right hand side respectively by $X^+(N)$ and $Q^+_{i,\mathrm{GB}}(N)$.

Case $L_i < L_{\max}$. We consider the recurrence relation $Q^-_i(N) = L_i\, X^-_{BJB}(N)[1 + Q^-_i(N-1)]$, with $Q^-_i(0) = 0$ and where $X^-_{BJB}(N) = N/(Z + L + L_{\max}(N-1))$ is the lower BJB with $X^- = 0$. A comparison with (4) shows that $Q^-_i(N)$ is a lower bound on $Q_i(N)$. Moreover, it can be seen as a sequence in the form $f(N) = C(N)[1 + f(N-1)], f(0) = 0$, where $C(N) = aN/(b+N)$, with $a = L_i/L_{\max}$, $0 < a \leq 1$, and $b = ((Z+L)/L_{\max} - 1)$, $b \geq 0$. We show by induction on N that $f(N) \geq geom\left(\frac{N}{N+1}\,C(N+1), N\right)$, where $geom(x,n) = \sum_{k=1}^{n} x^k$. Then the geometric form follow from the fact that

$$\left(\frac{N}{N+1}\right) C(N+1) = \frac{L_i\, N}{Z + L + L_{\max}N} = y_i(N) < 1.$$

The base case of the induction is proved if $f(1) \geq geom(C(2)/2, 1)$, but this simplifies to $C(1) \geq C(2)/2$ that is always true by definition of $C(N)$ and by the ranges of a and b. The induction hypothesis can instead be written as $f(N-1) \geq geom(H(N), N-1)$, where $H(N) = (N-1)C(N)/N = a(N-1)/(b+N)$. Note that $0 \leq H(N) < 1$ because $0 < a \leq 1$, $b \geq 0$, and $N \geq 1$; then we wish to prove that

$$f(N) \geq geom(H(N+1), N) = H(N+1)\left(\frac{1 - H(N+1)^N}{1 - H(N+1)}\right).$$

Using the induction hypothesis, we have

$$f(N) = C(N)\,(1 + f(N-1)) \quad (10)$$
$$\geq C(N)\,(1 + geom(H(N), N-1)) \quad (11)$$
$$= C(N)\left(\frac{1-H(N)^N}{1-H(N)}\right). \quad (12)$$

Then, the result follows immediately if we can show that

$$C(N)\frac{1-H(N)^N}{1-H(N)} \geq H(N+1)\frac{1-H(N+1)^N}{1-H(N+1)},$$

but this can be easily verified by substituting the definitions of $H(N)$ and $C(N)$ into the inequality and considering the conditions $0 < a \leq 1$ and $b \geq 0$. ∎

Theorem 2: The queue-length $Q_i(N)$ is bounded from above by

$$Q^+_{i,\mathrm{GB}}(N) = \begin{cases} \dfrac{Y_i(N)}{1-Y_i(N)} - \dfrac{Y_i(N)^{N+1}}{1-Y_i(N)}, & \text{if } L_i < L_{\max}, \\ \dfrac{1}{m_{\max}}\Big(N - ZX^- - \displaystyle\sum_{k:L_k<L_{\max}} Q^-_{k,\mathrm{GB}}(N)\Big), & \\ & \text{if } L_i = L_{\max}; \end{cases} \quad (13)$$

for any X^+ and X^- such that $X(N) \leq X^+ \leq X_{\max}$ and $0 \leq X^- \leq X(N)$, and where

$$Y_i(N) = L_i\,X^+ \quad (14)$$

is the ratio y of the underlying geometric sum (7).

Proof: The case $L_i = L_{\max}$ can be proved as in the previous theorem. The case $L_i < L_{\max}$ follows by the fact that the utilization in product-form networks is monotonically increasing with N [26], thus expanding $Q_i(N)$ as a summation of utilizations this is immediately bounded as $Q_i(N) = U_i(N) + U_i(N)U_i(N-1) + \ldots \leq U_i(N) + U_i(N)^2 + \ldots + U_i(N)^N$. The final formula follows by considering an upper bound $Y_i(N)$ on $U_i(N)$ and using the closed-form formula of a geometric sum. ∎

B. Throughput Bounds

We extend the bounding result to throughput bounds by deriving a new *exact* expression of the throughput $X(N)$ which includes more information than the standard MVA formula (3). We begin by observing that a tight approximation of (3) should account also for the population constraint

$$\sum_{i=1}^{M} Q_i(N-1) = N - 1 - ZX(N-1), \quad (15)$$

which expresses the conservation of first moments of queue-lengths in closed systems [10]. This constraint significantly limits the feasible values of $R(N)$ in (3). We therefore integrate (15) into (3) with the aim of minimizing the "information loss" of replacing the exact queue-lengths $Q_i(N)$ by their corresponding GB bounds. Isolating in (3) and (15) the queues with service demand $L_i = L_{\max}$, after simple manipulations it is found the following new exact formula

$$X(N) = N/\left(Z + L + L_{\max}(N - 1 - ZX(N-1)) - D(N)\right) \quad (16)$$

where $D(N) = \sum_{i=1}^{M}(L_{\max} - L_i)Q_i(N-1)$. If we compare equation (16) with the BJB formula in Table III, we note that the lower BJB can be derived and proved by setting both $X(N-1)$ and $D(N)$ to their minimum values $X(N-1) = 0$ and $D(N) = 0$. This suggests a new general approach for deriving throughput bounds, where we replace $X(N-1)$ and $D(N)$ with suitable bounds. This yields the following result.

Theorem 3: The throughput $X(N)$ is bounded by $X^-_{GB}(N) \leq X(N) \leq X^+_{GB}(N)$, where

$$X^-_{GB}(N) = N/\Big(Z + L + L_{\max}(N - 1 - ZX^-) - \sum_{i=1}^{M}(L_{\max} - L_i)\,Q^-_{i,\mathrm{GB}}(N-1)\Big), \quad (17)$$

$$X^+_{GB}(N) = N/\Big(Z + L + L_{\max}(N - 1 - ZX^+) - \sum_{i=1}^{M}(L_{\max} - L_i)\,Q^+_{i,\mathrm{GB}}(N-1)\Big), \quad (18)$$

for any X^+ and X^- such that $X(N-1) \leq X^+ \leq X_{\max}$ and $0 \leq X^- \leq X(N-1)$.

Proof: The proof for both bounds follows immediately from (16). In addition, we verify that the denominator of (18) is always positive. Consider the worst-case $X^+(N-1) = X_{\max} = 1/L_{\max}$, the denominator of $X^+_{GB}(N)$ becomes

$$L + L_{\max}(N-1) - \sum_{i:L_i<L_{\max}} (L_{\max} - L_i)Q^-_{i,\mathrm{GB}}(N-1).$$

The quantity is always greater than zero, because the sum $\sum_{i:L_i<L_{\max}}(L_{\max} - L_i)\,Q^-_{i,\mathrm{GB}}(N-1)$ cannot be greater than $L_{\max}(N-1)$ under the constraint $\sum_{i:L_i<L_{\max}} Q^-_{i,\mathrm{GB}}(N-1) \leq N-1$. This proves that $X^+_{GB}(N)$ is always positive. ∎

C. Geometric Square-Root Bounds

Tight bounds for models with a large delay Z usually require an improved approximation of the term $ZX(N-1)$ in (16). This term quantifies the average number of jobs waiting at the delay servers in the network with one job less [10] and thus directly affects the queue-length values $Q_i(N-1)$. Previous work has obtained improved approximations either by iterative approaches [9], [10], or by non-iterative approximations, e.g., the Square Root Bounds (SQB) of [10]. The two methodologies provide similar accuracy. We generalize non-iterative approaches using the new exact relation (16). Consider the bounding relations [26], [10]

$$K(N)\,X(N) \leq X(N-1) \leq X(N), \quad (19)$$

where $K(N) = (N-1)/N$. If we replace in (16) the term $X(N-1)$ with $X^- = K(N)X(N)$ or $X^+ = X(N)$, then we have removed the dependence on $X(N-1)$. Each inequality can then be solved analytically for $X(N)$, as we show in the following generalization of the SQB bounds.

Theorem 4: For models with $Z > 0$, the throughput $X(N)$ is bounded by

$$2N/\left(b(N) + \sqrt{b^2(N) - 4ZL_{\max}(N-1)}\right) \leq X(N) \leq 2N/\left(b(N) + \sqrt{b^2(N) - 4ZL_{\max}N}\right), \quad (20)$$

where $b(N) = Z + L + L_{\max}(N-1) - \sum_{i:L_i<L_{\max}}(L_{\max} - L_i)Q_i(N-1) \geq 0$.

Proof: Solving (16) for $X(N)$ in the lower bound case we get the quadratic inequality

$$J(N)X^2(N) - \Big(Z + L + L_{\max}(N-1) - D(N)\Big)X(N) + N \geq 0,$$

where $J(N) = K(N)L_{\max}Z$, $D(N) = \sum_{i:L_i<L_{\max}}(L_{\max} - L_i)Q_i(N-1)$ and with associated discriminant

$$\Delta_1 = \Big(Z + L + L_{\max}(N-1) - D(N)\Big)^2 - 4(N-1)L_{\max}Z,$$

where we used the relation $K(N)N = N-1$. Similarly, in the upper bound case we get the second-order inequality

$$\Big(L_{\max}Z\Big)X^2(N) - \Big(Z+L+L_{\max}(N-1)-D(N)\Big)X(N)+N \leq 0,$$

with discriminant $\Delta_2 = \Big(Z + L + L_{\max}(N-1) - D(N)\Big)^2 - 4NL_{\max}Z$. Noting that $\Delta_1 \geq \Delta_2$, we first show that both inequalities are associated to real radices by proving that $\Delta_2 \geq 0$. From the asymptotic properties of queue-lengths [27], we have $D(N) < D(\infty) = L - m_{\max}L_{\max} \leq L - L_{\max}$. Thus,

$$\begin{aligned}\Delta_1 \geq \Delta_2 &\geq \Big(Z+L+L_{\max}(N-1)-(L-L_{\max})\Big)^2 - 4NL_{\max}Z \\ &= \Big(Z + L_{\max}N\Big)^2 - 4NL_{\max}Z = \Big(Z - L_{\max}N\Big)^2 \geq 0,\end{aligned}$$

which finally implies that $\Delta_1 \geq 0$ and $\Delta_2 \geq 0$.

Since both discriminants are non-negative, both quadratic equations are always associated with real radices and we can easily solve the inequalities, e.g., by computing the *reciprocal* solution formula[2] of a quadratic equality $a_2x^2 + a_1x + a_0 = 0$ given by $x = 2a_0/(-a_1 \pm \sqrt{a_1^2 - 4a_2a_0}), \quad x \neq 0$, which allows us to leave the usual dependence on N in the numerator. Bound formulas follow after noting that in the reciprocal solution formula only the radices with positive sign can provide positive throughput values and therefore negative radices have to be discarded. Finally, note that $b(N) \geq Z + L + L_{\max}(N - 1 - ZX(N-1)) - D(N) = Z + R(N)$, which immediately implies the observation $b(N) \geq 0$. This completes the proof of Theorem 4. ∎

The Geometric Square-root Bounds (GSB) are immediately obtained by replacing the queue-lengths in $b(N)$ by $Q^-_{i,\mathrm{GB}}(N-1)$ (GSB lower bound) or $Q^+_{i,\mathrm{GB}}(N-1)$ (GSB upper bound). We remark that *also* the GB upper and lower bounds provide good results for models with large delays, but the GSB bounds are typically more accurate.

D. Asymptotic Correctness

We conclude the section by showing the correctness of the proposed bounds under asymptotic growth of the population size N. This is particularly relevant for heavily-loaded models where the analysis investigates the limiting condition of the system under critical congestions. We show that, in this limit case, our bounds are asymptotically exact, i.e., they return the same asymptotic values of the exact MVA formulas (3)-(4) as given in [27]

$$X(\infty) = \lim_{N\to+\infty} X(N) = X_{\max} = L_{\max}^{-1},$$

$$Q_i(\infty) = \lim_{N\to+\infty} Q_i(N) = \begin{cases} \dfrac{L_i}{L_{\max} - L_i} & \text{if } L_i < L_{\max}, \\ +\infty, & \text{if } L_i = L_{\max}. \end{cases}$$

[2]This formula can be easily proved by first dividing the quadratic equality by $x^2 > 0$ and then solving for x^{-1} in the resulting equation with the usual quadratic solution formula.

TABLE IV
STRESS CASE 1 (ALMOST BALANCED DEMANDS, NO DELAY)

	Lower Bounds			$X(N)$	Upper Bounds		
N	BJB	PB	GB	Exact	GB	PB	BJB
2	4.2553	4.3174	4.3040	4.3174	4.3174	4.3174	4.3243
5	6.4935	6.6600	6.6859	6.7148	6.7524	6.7297	6.7568
10	7.8740	8.0219	8.1521	8.2062	8.2690	8.2700	8.3160
15	8.4746	8.5690	8.7663	8.8346	8.9051	8.9531	9.0090
20	8.8106	8.8672	9.0947	9.1682	9.2431	9.3387	9.4007
30	9.1743	9.1943	9.4290	9.4993	9.5818	9.7591	9.8280
40	9.3677	9.3748	9.5933	9.6540	9.7429	9.9838	10.000
60	9.5694	9.5703	9.7501	9.7924	9.8358	10.000	10.000
80	9.6735	9.6736	9.8233	9.8530	9.8765	10.000	10.000

Theorem 5: The GB queue-length bounds are asymptotically exact, i.e., $\lim_{N\to+\infty} Q^-_{i,\mathrm{GB}}(N) = Q_i(\infty)$ and $\lim_{N\to+\infty} Q^+_{i,\mathrm{GB}}(N) = Q_i(\infty)$, $1 \leq i \leq M$.

Proof: Case $L_i < L_{\max}$. Noting that $y_i(N) < 1$ and $\lim_{N\to+\infty} y_i(N) = L_i/L_{\max}$, we get

$$\begin{aligned}\lim_{N\to+\infty} Q^-_{i,\mathrm{GB}}(N) &= \lim_{N\to+\infty} \frac{y_i(N)}{1-y_i(N)} - \lim_{N\to+\infty} \frac{y_i(N)^{N+1}}{1-y_i(N)} \\ &= \frac{L_i/L_{\max}}{1 - L_i/L_{\max}} = \frac{L_i}{L_{\max} - L_i}. \quad (21)\end{aligned}$$

Case $L_i = L_{\max}$. The result follows from (8) by noting that the sum of the optimistic non-bottleneck queue-lengths converges to a finite value and that also $X^+(N)$ converges to the finite value $X_{\max}$, being $X(N)$ monotonically increasing with N and $X^+(N) \leq X_{\max}$. ∎

Theorem 6: The GB throughput bounds are asymptotically exact, i.e., $\lim_{N\to+\infty} X^-_{\mathrm{GB}}(N) = \lim_{N\to+\infty} X^+_{\mathrm{GB}}(N) = X(\infty) = X_{\max}$.

Proof: The theorem is an immediate consequence of the convergence of queue-length bounds and of the fact that $ZX^-(N-1) \leq ZX^+(N-1) \leq Z/L_{\max}$ are bounded quantities. ∎

Experiments in Section IV-A show that the GB rate convergence to the exact value is much faster than for BJB and PB.

IV. ACCURACY VALIDATION

We assess the accuracy of the proposed bounds using three approaches. In Section IV-A we evaluate the GB with the stress cases used in [9], [10] to study the accuracy of BJB and PB. In Section IV-B, we perform an accuracy analysis on 1000 randomly generated models with different service demands and number of queues. Finally, in Section IV-C we show that the GB can be used to compute tight throughput approximations that are more accurate than bounds, although no longer upper and lower limits to the exact value. Experimental results indicate that GB bounds outperform BJBs and PBs in all three validations and are always superior to these methods except for very small population values where the PBs are slightly more accurate.

We do not report a comparison with non-iterative bounds recently derived in [28] from the analysis of the normalizing constant of equilibrium state probabilities, since their application is limited to models without delay servers ($Z = 0$). This is a significant constraint for models of real systems where overheads not associated with queueing often exist. Nevertheless, comparative evaluations with these bounds indicate that the GBs again provide the best results.

TABLE V
STRESS CASE 2 (ALMOST BALANCED DEMANDS, LARGE DELAY)

	Lower Bounds					$X(N)$	Upper Bounds				
N	BJB(1)	BJB(2)	PB(1)	PB(2)	GSB	Exact	GSB	PB(2)	PB(1)	BJB(2)	BJB(1)
2	1.3605	1.4316	1.3668	1.4335	1.4319	1.4335	1.5195	1.4335	1.4335	1.4337	1.4337
5	2.8249	3.2670	2.8559	3.2884	3.3432	3.3636	3.5582	3.3671	3.3997	3.3694	3.4015
10	4.4053	5.3901	4.4512	5.4401	5.7569	5.8719	6.1410	5.9695	6.2631	5.9802	6.2702
15	5.4152	6.6796	5.4536	6.7281	7.2373	7.4677	7.8008	7.8685	8.6060	7.8911	8.6207
20	6.1162	7.4889	6.1434	7.5250	8.0856	8.3750	8.6467	8.6792	9.0531	8.7163	9.0806
30	7.0258	8.3926	7.0375	8.4084	8.9023	9.1858	9.3341	9.4132	9.5492	9.4678	9.5923
40	7.5901	8.8581	7.5948	8.8642	9.2640	9.5023	9.6095	9.7728	9.8182	9.8364	9.8705
60	8.2531	9.3065	8.2538	9.3073	9.5785	9.7397	9.8043	10.000	10.000	10.000	10.000
80	8.6300	9.5142	8.6301	9.5143	9.7147	9.8278	9.8594	10.000	10.000	10.000	10.000

A. Comparison on Stress Cases

We compare the GB throughput bounds with the BJB and PB bounds using the same four stress cases used in the literature for their validation [9], [10].

Stress Case 1: The network is *almost balanced*, with $M = 4$ queues and service demands $L_1 = 0.10$, $L_2 = 0.10$, $L_3 = 0.09$, $L_4 = 0.08$. The network has $m_{\max} = 2$ bottlenecks with $L_{\max} = 0.10$. The maximum throughput is $X_{\max} = 10.00$.

Stress Case 2: The network has the same queues of Stress Case 1 and an additional delay server with $Z = 1.00$. The maximum throughput is $X_{\max} = 10.00$.

Stress Case 3: The network is *unbalanced*, with $M = 4$ queues with $L_1 = 0.10$, $L_2 = 0.1$, $L_3 = 0.05$, and $L_4 = 0.04$. The network has $m_{\max} = 2$ bottlenecks with $L_{\max} = 0.10$. The maximum throughput is $X_{\max} = 10.00$.

Stress Case 4: The network has the same queues of Stress Case 3 and an additional delay server with $Z = 1.00$. The maximum throughput is $X_{\max} = 10.00$.

We point the reader to [9], [10] for additional details on the above experiments. There are at least two simultaneous conditions that make all above models stress cases for bounds:

1) none of the models is perfectly balanced, thus in all models we do not obtain tight approximations using the BJB. This lets us understand to what extent the proposed bounds are able to account for variability in the service demands.
2) in all four cases, $X(N)$ converges very slowly to the asymptotic value $X_{\max}$ due to the presence of multiple bottleneck queues [29], i.e., $m_{\max} > 1$. This complicates the approximation and in particular for the optimistic bounds which tend to diverge quite quickly from the exact throughput curve.

Tables IV-VII show the lower and upper BJB, PB, GB and GSB throughput bounds obtained on the four stress cases and the underlined values denote the most accurate results. Figure 1 illustrates the GB and the GSB in comparison to the PB bounds which are always more accurate than the BJB. The upper GSB is computed using the utilization bound $Y_i(N) = \min\{L_i X^+_{PB}(N), L_i\, X_{\max}\}$ where the X^+ bound inside X^+_{PB} is chosen equal to the upper ABA bound of Table III. The PB and BJB bounds in the stress cases 2 and 4 are computed using the iterative extensions for models with delays defined in [9], [10]. When specified, the number within brackets represents the number of iterations required to compute the bound, e.g., PB(2) iteratively computes throughput bounds for the populations N, $N-1$ and $N-2$. We remark that we limited our comparison to BJB and PB bounds obtained with no more than two iterations since we observed that additional iterations provide a negligible increase of accuracy.

TABLE VI
STRESS CASE 3 (UNBALANCED DEMANDS, NO DELAY)

	Lower Bounds			$X(N)$	Upper Bounds		
N	BJB	PB	GB	Exact	GB	PB	BJB
2	5.1282	5.3604	5.2989	5.3604	5.3604	5.3604	5.5172
5	7.2464	7.3414	7.6725	7.8026	7.8620	8.0332	8.6207
10	8.4034	8.4070	8.8216	8.9302	9.0157	9.6346	10.000
15	8.8757	8.8759	9.2313	9.3024	9.3750	10.000	10.000
20	9.1324	9.1324	9.4348	9.4828	9.5238	10.000	10.000
30	9.4044	9.4044	9.6338	9.6591	9.6774	10.000	10.000
40	9.5465	9.5465	9.7303	9.7458	9.7561	10.000	10.000
60	9.6931	9.6931	9.8240	9.8315	9.8361	10.000	10.000
80	9.7680	9.7680	9.8696	9.8739	9.8765	10.000	10.000

As we can see from tables IV-VII, the proposed GB and GSB bounds are very accurate and much closer to the exact values for the great majority of models. The GB and GSB are slightly less precise than the PB only for very small population values, i.e., $N \leq 10$, where obviously all bounds are very tight to the exact throughput values. This is a consequence of the fact that the PB are designed to approximate very lightly loaded models, where the queue-lengths approximately grow in a linear fashion with N. The increase of accuracy of the GBs is particularly evident in the rapid convergence to the exact value when the network becomes congested. These results clearly indicate the effectiveness and the robustness of the proposed bounds on the most problematic cases of almost balanced and unbalanced models.

B. Comparison on Random Models

We now evaluate bound accuracy on a testbed of 1000 random models. Queue service demands are drawn from a uniform distribution in $[0.00, 1.00]$. To stress the models with a delay server, we consider a large Z, such that $Z = 10\, L_{\max}$. The number of queues is again drawn from an uniform distribution in $[5, 50]$. In order to provide conservative results, for each bound X_{bound} we consider the following error function:

$$\Delta_{err} = \max_{2 \leq N \leq 1000} \frac{|X_{bound}(N) - X_{exact}(N)|}{X_{exact}(N)},$$

where $X_{exact}(N)$ is the exact throughput computed with the MVA algorithm [21]. This index considers, for each model, the maximum relative error for $2 \leq N \leq 1000$. We do not consider the trivial case $N = 1$ because it is immediately $X(1) = 1/(Z + L)$. The results are shown in tables VIII and IX. It can be

TABLE VII
STRESS CASE 4 (UNBALANCED DEMANDS, LARGE DELAY)

	Lower Bounds					$X(N)$	Upper Bounds				
N	BJB(1)	BJB(2)	PB(1)	PB(2)	GSB	Exact	GSB	PB(2)	PB(1)	BJB(2)	BJB(1)
2	1.4388	1.5238	1.4566	1.5283	1.5265	1.5283	1.6358	1.5283	1.5283	1.5310	1.5310
5	2.9586	3.4760	2.9743	3.4893	3.5996	3.6280	3.8950	3.6352	3.6637	3.6673	3.6895
10	4.5662	5.6838	4.5673	5.6853	6.2334	6.4220	6.8289	6.5861	6.8581	6.7426	6.9605
15	5.5762	6.9785	5.5763	6.9786	7.7640	8.1331	8.5067	8.8359	9.2455	9.1929	9.4937
20	6.2696	7.7666	6.2696	7.7667	8.5579	8.9455	9.1230	9.7029	9.8139	10.000	10.000
30	7.1599	8.6183	7.1599	8.6183	9.2376	9.4828	9.5341	10.000	10.000	10.000	10.000
40	7.7071	9.0420	7.7071	9.0420	9.5077	9.6591	9.6807	10.000	10.000	10.000	10.000
60	8.3449	9.4372	8.3449	9.4372	9.7251	9.7973	9.8047	10.000	10.000	10.000	10.000
80	8.7051	9.6143	8.7051	9.6143	9.8138	9.8558	9.8594	10.000	10.000	10.000	10.000

noted that the mean, median (denoted as "med") and maximum of Δ_{err} are *always* favorable to GB and GSB.

We conclude the experiment by extrapolating the behavior of the error Δ_{err} as a function of the population N. We present in Figure 2 the cumulative distribution function of the population on which Δ_{err} is computed, i.e., the population where the GB and GSB error is maximal. The results indicate that for the GSB the maximal error lies always in the range $N \in [2, 120]$ and similarly for the GSB it is always in $N \in [2, 140]$. This indicates that our choice of the range $2 \leq N \leq 1000$ for the computation of Δ_{err} is sufficient to capture the maximum error. Further, it also indirectly confirms that the bounds always converge rapidly to the asymptotic value X_{max}, since after $N = 140$ the error starts to decrease in all cases.

C. Bound-Based Throughput Estimates

The third validation consists in comparing the effectiveness of all bounds in determining throughput approximations. These are particularly useful in all cases where the analysis requirements are not concerned with determining bounds on solution accuracy, but instead one seeks for maximum accuracy without resorting to more expensive iterative techniques.

We follow the approach of [23] and, given two bounds $X^+(N)$ and $X^-(N)$, we consider the following approximation of $X(N)$

$$A(N) = \frac{2X^+(N)X^-(N)}{X^+(N) + X^-(N)},$$

that is the harmonic mean of the two bounds. This implies a maximal relative error

$$\max_{X(N)\in[X^-(N),X^+(N)]} \frac{|A(N) - X(N)|}{X(N)} = \frac{X^+(N) - X^-(N)}{X^+(N) + X^-(N)},$$

which represents the degree of uncertainty in approximating $X(N)$ by $A(N)$. We evaluate this measure on 1000 random models with the same characteristics of those used in the previous section. For each model, we determine the largest maximal relative error for the populations from $N = 2$ to $N = 1000$. This provides again a worst-case estimate of bound performance. The numerical results are shown in Table X and Table XI. As we can see, the accuracy of the GB and GSB estimates is even doubled with respect to the PB and BJB estimates. This indicates that, on average, the worst-case gap between the upper and the lower GB and GSB bounds is much lower than the gap between the PB or the BJB bounds. Furthermore, we observe that this result is highly competitive with those achievable by iterative bound hierarchies. For instance, in order to achieve a similar worst-case throughput error, the bound hierarchies of [30] require approximately $3 - 4$ iterations, each having a computational cost similar to that of computing the GB or the GSB bounds. This indicates that our bounds are highly competitive with much more expensive iterative techniques.

TABLE VIII
RELATIVE ERROR ($0 \equiv 0\%, 1 \equiv 100\%$) FOR 1000 RANDOM MODELS WITHOUT DELAYS ($Z = 0$).

Lower Bounds					Upper Bounds			
mean	med	stdev	max	Δ_{err}	mean	med	stdev	max
.049	.049	.014	.107	GB	.087	.091	.019	.123
.140	.141	.029	.233	PB	.107	.108	.016	.154
.149	.149	.026	.233	BJB	.229	.231	.038	.353

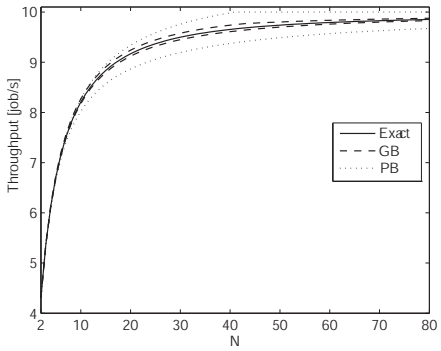
TABLE IX
RELATIVE ERROR ($0 \equiv 0\%, 1 \equiv 100\%$) FOR 1000 RANDOM MODELS WITH LARGE DELAYS ($Z/L_{\max} = 10$).

Lower Bounds					Upper Bounds			
mean	med	stdev	max	Δ_{err}	mean	med	stdev	max
.068	.067	.015	.116	GSB	.089	.091	.013	.131
.152	.151	.024	.225	PB(2)	.104	.105	.014	.152
.156	.156	.022	.225	BJB(2)	.207	.210	.035	.297
.243	.234	.040	.381	PB(1)	.113	.113	.012	.157
.247	.238	.039	.381	BJB(1)	.211	.213	.032	.297

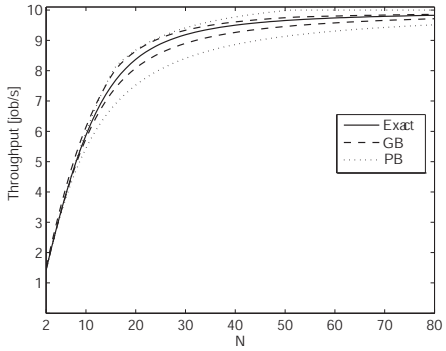
V. APPROXIMATION OF CLOSED FORK-JOIN NETWORKS

In this section the GB bounds are extended to approximate closed queueing network models with fork-join subsystems. This application shows how GB bounds can be effectively generalized beyond the product-form case, providing a new approximation technique for models that play an important role in real systems' optimization. Other extensions may be possible, but the lack of research on non-iterative methods for other classes of non-product form models makes it difficult to derive further generalizations.

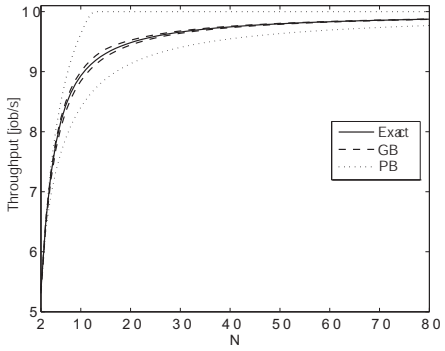
A fork-join network is characterized by the presence of subnetworks where jobs are served in parallel by multiple servers. This is effective in several models, e.g., when characterizing a disk read request that is served in parallel by mirrored drives. A fork-join subsystem labeled by k is composed by a fork node, a join node and P_k queues in parallel. Each time a new job arrives to the fork node, it is divided into P_k sibling tasks which are each sent to a distinct queue of the subsystem. After all tasks have been served, they are reassembled at the join node into the single original job, which then leaves the subsystem. A graphical representation of



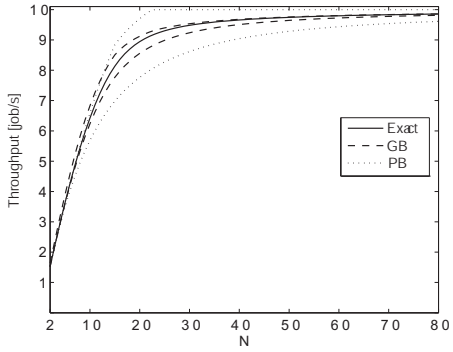
(a) Stress Case 1: almost balanced demands, no delay.



(b) Stress Case 2: almost balanced demands, large delay.

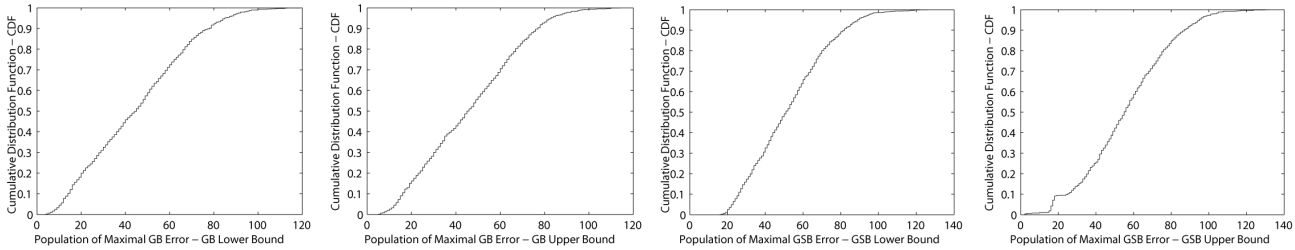


(c) Stress Case 3: unbalanced demands, no delay.



(d) Stress Case 4: unbalanced demands, large delay.

Fig. 1. Comparison of the GB bounds with the PB on the four stress cases of tables IV-VII. The results of the BJB are omitted because always less accurate than the corresponding results of the PB.



(a) (b) (c) (d)

Fig. 2. Distribution of the population N corresponding to the maximal error Δ_{err} of the GB and GSB. For the GB the maximal error is for $N \in [2, 120]$; for the GSB it is similarly always in $N \in [2, 140]$.

a model with a queue and a fork-join subsystem composed by P_k queues is given in Figure 3. Without loss of generality, we assume for this section that networks are composed by fork-join subsystems only, e.g., the network in Figure 3 has two fork-join subsystems composed respectively by 1 and P_k queues.

In [31] it is shown that, if the queues inside the fork-join subsystem k have all first-come-first-served service discipline, exponential distribution of service times and same average service demand L_k, then the response time of jobs at a fork-join subsystem is approximately given by

$$R_k^{fj}(N) \approx L_k \left(H_k + A_k^{fj}(N)\right) \approx L_k \left(H_k + Q_k^{fj}(N-1)\right), \tag{22}$$

where $A_k^{fj}(N)$ is the average number of jobs inside the fork-join subsystem k as seen by an arriving job, $Q_k^{fj}(N-1)$ is the average number of jobs inside k and $H_k = \sum_{i=1}^{P_k} 1/i \geq 1$ is the P_k-th harmonic number. Applying Little's Law to (22), we see that the average number of jobs in a fork-join subsystem may be approximated as

$$Q_k^{fj}(N) \approx L_k \, X^{fj}(N) \left(H_k + Q_k^{fj}(N-1)\right), \tag{23}$$

where

$$X^{fj}(N) = \frac{N}{\sum_{j=1}^{M} R_j^{fj}} \approx \frac{N}{\sum_{j=1}^{M} L_j [H_j + Q_j^{fj}(N-1)]}$$

approximates the throughput of the network measured at an arbitrarily-chosen reference subsystem. The following BJB-like

TABLE X
MAXIMAL RELATIVE ERROR ($0 \equiv 0\%, 1 \equiv 100\%$) OF APPROXIMATIONS BASED ON BOUNDS FOR MODELS WITHOUT DELAYS ($Z = 0$)

	mean	med	stdev	max
GB	.061	.064	.014	.084
PB	.123	.126	.019	.167
BJB	.180	.183	.021	.230

TABLE XI
MAXIMAL RELATIVE ERROR ($0 \equiv 0\%, 1 \equiv 100\%$) OF APPROXIMATIONS BASED ON BOUNDS FOR MODELS WITH LARGE DELAYS ($Z/L_{\max} = 10$)

	mean	med	stdev	max
GSB	.077	.079	.010	.102
PB(2)	.129	.130	.015	.170
BJB(2)	.177	.179	.015	.210
PB(1)	.185	.179	.028	.286
BJB(1)	.232	.230	.022	.299

approximation for $X^{fj}(N)$ has also been proposed [32]:

$$X^{fj}_{BJB}(N) \approx \frac{N}{\sum_{j=1}^{M} L_j H_j + L_{\max}(N-1)}, \quad (24)$$

where $L_{\max} = \max_{j:1\leq j\leq M} L_j$. It is currently conjectured that (24) is a lower bound on $X^{fj}(N)$. In that case, it would be also possible to show that the GB approximation we propose below also defines bounds for fork-join systems.

We introduce an approximation of fork-join models that resembles the GB bounds for product-form networks. We will refer to this technique as the GB *fork-join approximation*. We remark that our technique does *not* give bounds and the main contribution is that it provides much improved accuracy with respect to (24) at similar computational costs.

*Approximation 1 (*GB *Fork-Join Queue-Length):* In a closed network with fork-join subsystems, the number of jobs in the fork-join subsystem k is approximated by

$$Q^{fj}_k(N) \approx Q^{fj}_{k,\mathrm{GB}}(N) = H_k \left(\frac{y_k(N)}{1-y_k(N)} - \frac{y_k(N)^{N+1}}{1-y_k(N)} \right) \quad (25)$$

where $y_k(N) = L_k\, N/(\sum_{j=1}^{M} L_j H_j + L_{\max} N) < 1$.

Proof: The above approximation derives from the following passages. We rewrite (23) as $Q^{fj}_k(N)/H_k \approx L_k X^{fj}(N)[1 + Q^{fj}_k(N-1)/H_k]$ and we use (24) to define the recursion $Q^{fj}_{k,\mathrm{GB}}(N)/H_k \approx L_k\, X^{fj}_{BJB}(N)[1 + Q^{fj}_{k,\mathrm{GB}}(N-1)/H_k]$. This has form analogous to the class of recursion considered in the proof of the lower GB queue-length bound and where the coefficients have form $C(N) = L_k\, X^{fj}_{BJB}(N) = aN/(b+N)$, with $a = L_k/L_{\max} \leq 1$ and $b = \sum_j (L_j H_j/L_{\max}) - 1$. Noting that $H_j \geq 1$ for all subsystems j, it is $b \geq (\sum_j L_j)/L_{\max} - 1 \geq 0$. Therefore, the approximation follows with arguments analogous to the GB queue-length bounds. ■

Finally, using the same arguments that have lead to the derivation of the GB throughput bounds, we approximate the fork-join throughput as

$$X^{fj}_{GB}(N) \approx \frac{N}{\sum_{j=1}^{M} L_j H_j + L_{\max}(N-1) - D_{fj}(N)}, \quad (26)$$

where $D_{fj}(N) = \sum_{k:L_k \leq L_{max}} (L_{max} - L_k) Q^{fj}_{k,\mathrm{GB}}(N-1)$.

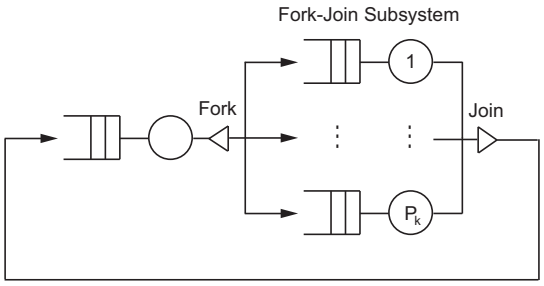


Fig. 3. Example of a closed network including a fork-join subsystem composed by P_k parallel queues.

TABLE XII
RESULTS FOR THE RESPONSE TIME [S] APPROXIMATIONS IN THE FORK-JOIN EXAMPLE 1

	$R^{fj}(N)$	Fork-Join Response Time Approximation					
N	SIMUL	MVAFJ	GBFJ	BJBFJ	Δ_{MVA}	Δ_{GB}	Δ_{BJB}
2	9.95	9.98	10.22	12.50	+0.03	+0.27	+2.55
5	17.01	17.11	17.92	21.50	+0.10	+0.91	+4.49
10	30.54	30.61	31.96	36.50	+0.07	+1.42	+5.96
15	45.12	45.15	46.48	51.50	+0.03	+1.36	+6.38
20	60.03	60.03	61.19	66.50	+0.00	+1.16	+6.47
25	75.01	75.01	76.00	81.50	+0.00	+0.99	+6.49
30	90.01	90.01	90.86	96.50	+0.00	+0.85	+6.49
35	105.02	105.02	105.75	111.50	+0.00	+0.73	+6.48
40	120.02	120.02	120.67	126.50	+0.00	+0.65	+6.48
45	135.02	135.02	135.60	141.50	+0.00	+0.58	+6.48
50	150.02	150.02	150.55	156.50	+0.00	+0.53	+6.48

A. Numerical Results

To assess the accuracy of the GB Fork-Join (GBFJ) approximation we compare it with the simulation and BJB Fork-Join (BJBFJ) approximation results presented in [19]. Since the BJBFJ approximation (24) is the only existing inexpensive approximation for closed fork-join models, the objective of our analysis is to show that the proposed method can be significantly more accurate than the BJBFJ approximation at similar costs.

We consider the two models used in [19] for validation, which have the following characteristics:

Fork-Join Example 1: The model considered in Table XII has $M = 3$ fork-join subsystems, with service demands $L_1 = 3$, $L_2 = 2$, $L_3 = 1$, and composed by $P_1 = 1$, $P_2 = 2$, and $P_3 = 3$ parallel queues, respectively.

Fork-Join Example 2: The model considered in Table XIII has $M = 3$ fork-join subsystems, with service demands $L_1 = 1$, $L_2 = 2$, $L_3 = 3$, and composed by $P_1 = 1$, $P_2 = 2$, and $P_3 = 3$ parallel queues, respectively.

A third model presented in [19] is not considered here since, being the model balanced, the BJBFJ approximation already provides optimal accuracy. We point the reader to [19] for additional details on the above experiments.

Tables XII and XIII show the results in approximating the network response time $R^{fj}(N)$. The SIMUL column reports the simulation results of [19]; the MVAFJ column is obtained with (22) and the iterative approximation (23); the GBFJ column employs (13) for approximating $Q^{fj}_k(N-1)$ in (22) and, using

TABLE XIII
RESULTS FOR THE RESPONSE TIME [S] APPROXIMATIONS IN THE FORK-JOIN EXAMPLE 2

	$R^{fj}(N)$	Fork-Join Response Time Approximation					
N	SIMUL	MVAFJ	GBFJ	BJBFJ	Δ_{MVA}	Δ_{GB}	Δ_{BJB}
2	11.84	11.97	12.10	12.50	+0.13	+0.26	+0.66
5	19.30	19.81	20.17	21.50	+0.51	+0.87	+2.20
10	33.28	33.77	34.39	36.50	+0.49	+1.11	+3.22
15	47.96	48.29	48.97	51.50	+0.33	+1.01	+3.54
20	62.83	63.06	63.70	66.50	+0.23	+0.87	+3.67
25	77.74	77.93	78.52	81.50	+0.19	+0.78	+3.76
30	92.68	92.85	93.38	96.50	+0.17	+0.70	+3.82
35	107.66	107.79	108.28	111.50	+0.13	+0.62	+3.84
40	122.66	122.76	123.20	126.50	+0.10	+0.54	+3.84
45	137.64	137.73	138.13	141.50	+0.09	+0.49	+3.86
50	152.62	152.71	153.08	156.50	+0.09	+0.46	+3.88

the population constraint, computes the approximation

$$R^{fj}(N) \approx \sum_{j=1}^{M} L_j\,[H_j + Q_j^{fj}(N-1)] \approx \sum_{j=1}^{M} L_j\,H_j + L_{\max}(N-1) - \sum_{i:L_i<L_{\max}} (L_{\max}-L_i) Q_{GB,i}^{fj}(N-1). \quad (27)$$

Finally, the BJBFJ column computes (24) with Little's Law as $R_{BJB}^{fj}(N) \approx N/X_{BJB}^{fj}(N)$. The last three columns in each table indicate the difference between the value obtained with the approximation technique indicated and the value obtained with simulation, e.g., $\Delta_{MVA} = R_{MVAFJ} - R_{SIMUL}$.

The presented results indicate that GBFJ approximation offers an accuracy level very close to the more expensive results of the fork-join MVA iterative approximation. Further, the technique has computational complexity that is independent of N and thus similar to that of the inexpensive BJBFJ approximation. Concerning this point, we remark that the computational cost of GB for fork-join systems is in practice the same of the product-form case, since only few additional multiplications, related to the presence of the H_j terms, are required. Hence, also the computational requirements of the GBFJ approximation grow linearly with M and independently of N.

VI. IMPACT ON PERFORMANCE OPTIMIZATION

We describe the impact of the GB bounds in performance optimization and in particular we focus on throughput maximization, which can be shown to be equivalent in closed systems to response time minimization [25]. First, we show the accuracy improvement of the GB bounds on simple models of load-balancing between two or three queues, where we illustrate the direct connection between optimization results and bound accuracy, as well as the robustness of GB as model complexity grows. Later, we consider a case study involving a much more complex fork-join model of a distributed application in a multi-tier environment, showing again the gains of our approximation method with respect to existing schemes. The results obtained in this section prove that the GB can significantly improve the solution of optimization programs which are the basic building blocks of quality-of-service (QoS) control algorithms used in the off-line or on-line optimization of distributed applications.

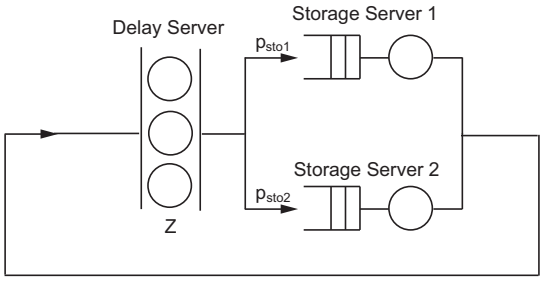


Fig. 4. Load balancing example with two storage servers. Requests inter-arrival times are modeled by a delay server.

A. Product-Form Networks and Nonlinear Optimization

We begin by showing a simple example that illustrates the practical importance of non-iterative bounds in system optimization. In particular, we give evidence that more expensive iterative solution methods such as the exact MVA algorithm [21] or the approximate MVA (AMVA) [33] can occasionally show numerical instabilities that degrade solution accuracy on models where instead bounds provide accurate solutions. This is an additional reason that motivates the use of bounds and promotes their application in optimization programs that are subject to this type of instabilities.

We consider a system where applications access data of two mirrored storage systems sto_1 and sto_2. We suppose that a self-optimization controller is integrated in the storage system and periodically optimizes the fraction of requests p_{sto_1} and $p_{sto_2} = 1 - p_{sto_1}$ directed to sto_1 and sto_2, respectively. The optimization criteria that drives controller decisions can meet different profiles, e.g., performance maximization or energy consumption minimization. For illustrating purposes, we here assume an average of $N = 10$ concurrently active applications, each accessing the storage for reading files with an average service time of $S_{sto_1} = 20ms$ and $S_{sto_2} = 30ms$. In real systems, these values can be easily identified by linear regression of measured utilization samples. It is also possible to show with standard operational analysis [8] that, for this specific model, the mean visits are $V_{sto_1} = p_{sto_1}$ and $V_{sto_2} = p_{sto_2}$, thus $L_{sto_1} = p_{sto_1} S_{sto_1}$ and $L_{sto_1} = p_{sto_2} S_{sto_2}$. We model the inter-arrival time of requests to the storage as a delay $Z = 13ms$. The example model is depicted in Figure 4.

According to these definitions, a MVA-based throughput maximization program may be formulated as

$$\begin{aligned} &\max \quad X(N) \\ &\text{s.t.} \quad X(n) = n \Big/ \Big(Z + \sum_{i=sto_1,sto_2} (p_i S_i + p_i S_i Q_i(n-1)) \Big); \\ &\quad Q_i(n) = X(n) p_i S_i (1 + Q_i(n-1)); \\ &\quad Q_i(0) = 0; \\ &\quad p_{sto_1} + p_{sto_2} = 1; \\ &\quad p_{sto_1} \geq 0, p_{sto_2} \geq 0, \end{aligned}$$

where $i = sto_1, sto_2$; the constraints are for all $n = 1, \ldots, N$. The program is not computationally challenging, as it seeks for the routing probabilities p_{sto_1} and p_{sto_2} that maximize $X(N)$ over the small feasible space $p_{sto_1} + p_{sto_2} = 1$. A similar, even less expensive, program can be obtained with the iterative AMVA technique, by integrating in the program the fixed-point

TABLE XIV
THROUGHPUT IN THE LOAD-BALANCING EXAMPLE IN FIGURE 4. ASTERISKS INDICATE INCORRECT EVALUATIONS OF THE SOLVER.

p_{sto_1}	METHOD	$X(N)$ [job/ms]	ITER
0.50	MVA	0.0661	60
0.50*	AMVA	(0.1022, expected 0.0669)	9
0.50	PB	0.0581	3
0.50	BJB	0.0578	2
0.85	MVA	0.0588	29
0.85*	AMVA	(0.2191, expected 0.0588)	6
0.85	PB	0.0533	3
0.85	BJB	0.0533	2
0.99*	MVA	(−64778.3, expected 0.0505)	481
0.99*	AMVA	(0.1021, expected 0.0505)	9
0.99	PB	0.0473	3
0.99	BJB	0.0473	2

iteration [33]

$$Q_i(N-1) = \left(\frac{N-1}{N}\right) Q_i(N), \quad i = 1, \ldots, M.$$

Similar simplification based on non-iterative bounds may be obtained, for example by expressing $X(N)$ using the lower BJB and lower PB of Table III.

We have evaluated the four programs using the nonlinear solver BONMIN 1.0 [36], which is based on an interior-point algorithm (IPOPT) and which we have coupled with the AMPL interpreter [37] for model generation. Throughout experiments we have used default program options for BONMIN and IPOPT and disabled AMPL's pre-solver. In particular, in order to understand how BONMIN evaluates each sub-model associated to a particular choice of p_{sto_1} and $p_{sto_2} = 1 - p_{sto_1}$, we have fixed p_{sto_1} and collected the final value $X(N)$ and the required number of iterations. Table XIV reports our observations for different values of p_{sto_1}. The marked entries indicate instances in which the throughput value provided by the solver is incorrect, intuitively as a result of numerical problems. For example, for $p_{sto_1} = 0.99$, the AMVA converges incorrectly to a value 0.1021 job/ms, whereas a stand-alone AMVA implementation would return an accurate value of 0.0505 job/ms as the exact MVA. On all performed experiments, both the BJB and the PB non-iterative bounds were numerically robust. Nevertheless, although well-behaved, the BJB and PB were significantly less accurate than the correct values of the MVA or the AMVA technique. The problem of deriving accurate non-iterative bounds has been addressed in this paper by introducing the GB and GSB bounds. For instance, for the case $p_{sto_1} = 0.99$ the GSB bound give an accurate value of 0.0501 job/ms at negligible costs compared to the MVA/AMVA.

Although the impact of the described problems may vary with the solver and with the algorithm used to evaluate the nonlinear program, they clearly indicate that iterative methods for queueing networks may show numerical problems when used within nonlinear optimization programs. This further motivates the development in this paper of the GB technique. In the next subsections, we show how the GB technique improves the accuracy of optimization results compared to BJB and PB.

B. Accuracy of Optimization Results

We focus on the solution of optimization models using bounds and discuss the relative merits of the GB with respect to the BJB and PB in this setting. For ease of interpretation, we begin by considering a simple load-balancing model similar to the one used in the previous subsection. The model is composed by two servers in parallel with associated routing probabilities $1 - p$ and p, service times $S_1 = 5s$ and $S_2 = 1s$, respectively. We seek for the value of p which maximizes the throughput $X(N)$ for a population of $N = 10$ requests. To simplify the analysis of the results, in this case we also set $Z = 0$. This simple example has a single optimization variable p, thus we solve it immediately with MATLAB by searching for the best throughput in the range $p \in [0, 1]$. We have compared the optimum routing determined by the exact MVA with the optimum value predicted by the optimization program where we have used the lower BJB, lower PB or lower GB to compute $X(N)$. The focus is on lower bounds since a maximization of a lower bound usually implies an improvement of the exact value, while an upper bound maximization does not necessarily imply any real improvement of the exact value. All exact and bounding throughput curves have a unique maximum in the range $p \in [0.833, 0.867]$ as depicted in Figure 5. As we can see, the increased accuracy of the GB immediately results in a better optimal routing decision $p = 0.860$ than the PB ($p = 0.847$) and the BJB ($p = 0.833$), being much closer to the theoretical best obtainable by the MVA algorithm ($p = 0.867$). This improved accuracy is particularly important in nonlinear optimization, where even a small variation of a variable, here the throughput $X(N)$ or p, can result in a significant change of the considered cost function. (For instance, a power consumption cost function typically grows cubically with the clock of the servers; thus, a routing imposing a larger load on one or more systems that are forced to re-scale their frequencies may consistently impact on energy costs [38].)

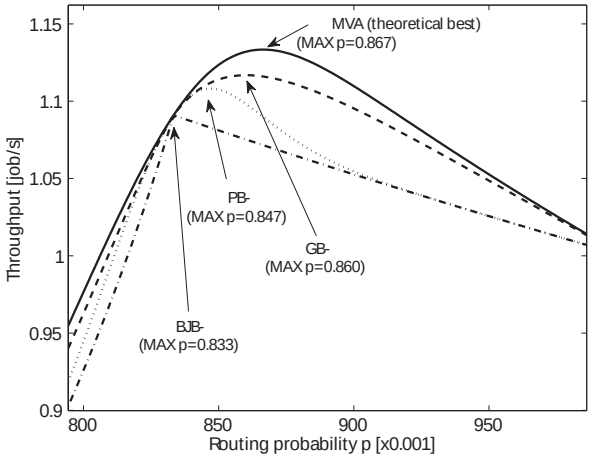


Fig. 5. Outcome of a simple routing optimization model with two queues. Increased bound accuracy leads to routing decisions closer to the optimum result of the exact MVA.

In order to better understand the result, we point out an important negative characteristic of the BJB. In this example the lower BJB has the form

$$X^{-}_{BJB}(N) = \frac{N}{pS_2 + (1-p)S_1 + \max\{pS_2, (1-p)S_1\}(N-1)},$$

which has a maximum for $p = S_1/(S_1 + S_2)$, that is, for the routing decision such that $L_1 = L_2$ and the two service demands

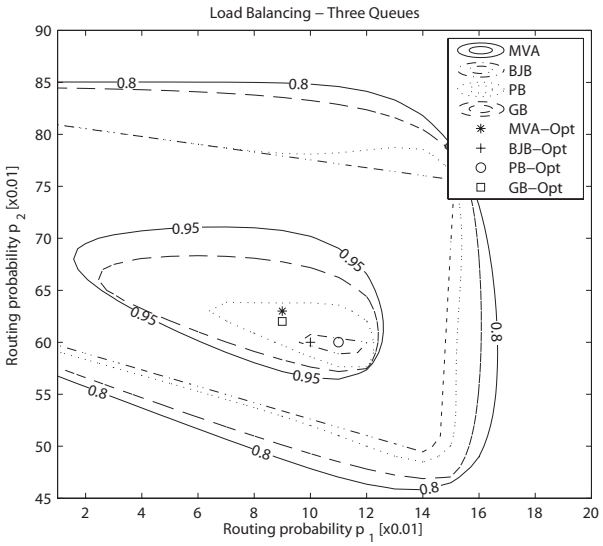


Fig. 6. Contour levels for a simple routing optimization model with three queues. The contour levels of the BJB, PB and GB refer to the same throughput value of the closest MVA contour line. The contour lines are computed for values equal to $0.8X_{MVA}^{Opt}$ and $0.95X_{MVA}^{Opt}$, where $X_{MVA}^{Opt} = 1.4701$ denotes the MVA optimal solution to the load balancing problem. Similarly, the notation GB-Opt denotes the optimum solution computed by the GB: the closer to the MVA-Opt maximum a result is, the better it is. Therefore, GB-Opt (marked by "□") again provides the best result among bounds. In addition, it could be noted that the contour levels of the GB are much closer to the contour levels of the MVA solution than the BJB and PB contour levels.

are balanced. Thus, the optimum routing decision determined by the BJB *ignores* the size of the population N, i.e., the optimal p is the same regardless of the intensity of the workload. Clearly, this is a negative property of the BJB, since the network can exhibit very different congestion levels at light and heavy loads which cannot be accounted by the BJB. Conversely, both PB and GB optimal p's change with N according to quite complex formulas, with the GB providing a better result thanks to its higher accuracy.

These considerations readily generalize to models with larger number of queues. For instance, if we add to the example a third queue such that $S_1 = 5$s, $S_2 = 1$s, $S_3 = 2$s, and denote the routing probabilities of the three queues as p_1, p_2 and $p_3 = 1 - p_1 - p_2$, respectively, the GB provides again the best routing decision. In fact, the throughput is now a three-dimensional curve that is function of p_1 and p_2, with contour levels shown in Figure 6; each group of contour levels indicates a set of points where the different curves assume the same throughput value. The contour levels clearly indicate that the GB provides the best approximation of the exact MVA throughput curve since the GB contour levels (and thus the gradients considered by the nonlinear solver) are much more similar to the exact MVA contour levels than the contour levels of the PB and of the BJB. Moreover, the optimum GB throughput (marked in the figure by "□") is almost identical to the exact MVA optimum (denoted by "$*$"). The PB and the BJB both provide coarse approximations of the MVA curve and have much farther optimums from the exact MVA optimum.

C. Case Study

We conclude our analysis by presenting a case study related to the optimization of the throughput of a distributed application

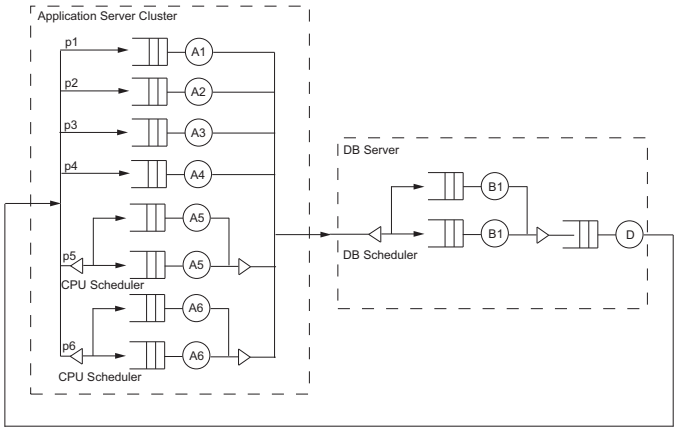


Fig. 7. Case study: fork-join model of a distributed application in a multi-tier architecture.

running in a multi-tier architecture. The model we consider is depicted in Figure 7 and represents a typical two tier application, relying on an application server cluster and on a shared database. An incoming request is first served by the application server A_k with probability p_k, with $\sum_{j=1}^{6} p_j = 1$. Then, the execution proceeds with queries at the shared DB server, after which it is sent back to the requesting application. Note that the identical names given to some queues in Figure 7 indicate that the speed of the two servers inside a fork-join subsystem is the same, i.e., both queues inside the bottommost application server A_6 have service demand L_{A_6}.

The application servers A_1 to A_4 run on single-core machines and therefore have a single server; the application servers A_5 and A_6 are modeled instead as dual-core machines and the results at the different CPUs require synchronization. The DB server implements a query scheduling algorithm similar to the one presented in [16], which imposes synchronizations that are modeled by the fork-join subsystem. Finally, the queue D summarizes communication overheads.

This architectural model is analogous to recent models for dynamic resource provisioning of multi-tier applications [39] and for the characterization of real J2EE applications [40]. In particular, we parameterize the model according to service demands similar to those measured in [40] and set $L_{A_1} = p_1 12.98ms$, $L_{A_2} = p_2 13.64ms$, $L_{A_3} = p_3 24.42ms$, $L_{A_4} = p_4 24.42ms$; the dual-core machines have similar demands scaled by the number of identical processors, i.e., $L_{A_5} = p_5 13.64/2 = p_5 6.82ms$ and $L_{A_6} = p_6 24.42ms = p_6 12.21ms$; the DB server is parameterized by $L_{B_1} = 10.64/2 = 5.32ms$, the final communication overhead by $L_D = 1.12ms$. All servers have a first-come first-served service discipline. The outgoing link from queue D is used as reference point for the computation of throughput.

We assume that the system periodically performs a self-optimization in order to determine the best load balancing decision for the routing probabilities p_k and according to the current workload level. We investigate different workload intensities, with the population N ranging in the interval $[2, 50]$. Since larger population values overload the system, we limit our analysis to the non-asymptotic conditions $N \leq 50$. We also stress that this optimization model greatly differs from the two considered earlier in this section, since the presence of synchronization at the fork-join subsystems makes the optimization task much more complex

TABLE XV
CASE STUDY: OPTIMAL SOLUTIONS TO THE ROUTING PROBLEM PROVIDED BY THE BJBFJ AND GBFJ APPROXIMATIONS

p_k	BJBFJ Optimum	GBFJ Optimum
p_1	0.1259	0.2021
p_2	0.0939	0.1851
p_3	0.0000	0.0031
p_4	0.0000	0.0008
p_5	0.7800	0.4858
p_6	0.0000	0.1229

and the outcome unpredictable with existing theory.

The optimization is performed using the BJB fork-join approximation (BJBFJ) and the GB fork-join approximation (GBFJ). The cost function is throughput maximization. We have used the `fmincon` function of the optimization toolbox of MATLAB 2007a with default parameters for all tests.

In all experiments, the GBFJ approximation always provided the best results and the BJBFJ was never more accurate than the GBFJ. We have found that the worst-case deviation of the two optimization programs corresponds to the value $N = 10$, for which the throughputs computed, for validation purposes, with the fork-join MVA are $X(N) = 0.1541$ job/ms for the BJBFJ and $X(N) = 0.1685$ job/ms for the GBFJ, respectively. Table XV reports the obtained routing probabilities for the BJBFJ and the GBFJ. We recall that in the case of fork-join models it is not possible to compute an exact theoretical optimum since exact formulas do not exist and the model under evaluation is too complex to be solved using numerical methods applied to the underlying Markov chain. Therefore, we perform a double check on the optimality of the results by defining a simulation model of the system using the JSIMWIZ simulator of the Java Modelling Tools suite [41]. Using this simulation model, we have tested the performance of the distributed architecture using the two routing profiles of Table XV. In the simulation, we have used a sample space of 10 millions and a 95% confidence interval. The simulation with the BJBFJ routing solution yields a mean throughput of $X^{sim}_{BJB}(N) = 0.1565$ job/ms, with a 95% confidence interval given by $I_{BJB} = [0.1557, 0.1573]$; the GBFJ routing provides a much better throughput of $X^{sim}_{GB}(N) = 0.1702$ job/ms, with 95% confidence interval equal to $I_{GB} = [0.1693, 0.1711]$. This further confirms the benefit of using the GBFJ technique instead of the BJBFJ technique.

From the results we see that the BJBFJ and the GBFJ consistently indicate that with $N = 10$ jobs the slowest servers A_3 and A_4 should not process jobs in order to avoid performance degradations, that is, $p_3 \approx p_4 \approx 0$. Nevertheless, the GBFJ can much better differentiate the allocation of jobs across the remaining queues, for instance, p_6 is not set to zero as in the BJBFJ solution, but a consistent fraction of the workload (12.29%) is routed to this server. This is a consequence of the fact that GBFJ computes an approximation for the queue-length of each station in the network, whereas the BJBFJ does not approximate each individual queue-length value. Therefore, the GBFJ estimates detailed network information that is not available to the BJBFJ, evaluating more accurately the feasible configurations and exploiting better the capacity of each queue.

VII. CONCLUSIONS

Problems related to the performance tuning of systems require the accurate and efficient solution of optimization models based on queueing networks. In this paper we have proposed the Geometric Bounds (GB), a fast and accurate bounding technique for the computation of performance measures which are frequently used by QoS control algorithms, such as queue-lengths, throughputs and response times. We have shown that the proposed GB bounds, in spite of their simple formulas which are related to partial sums of geometric sequences, are more accurate than known bounding techniques even on unbalanced models with delays and multiple bottlenecks, which are the hardest to approximate. The extension to closed fork-join networks of the GB technique illustrates the possibility of successfully applying the GB approach outside the product-form case and finds application to models of real systems such as disk drives or databases.

Finally, we have discussed the impact on performance optimization of our results with simple examples and a case study, showing that the increased accuracy of the GB provides a significant improvement of optimization results in comparison to existing bounds.

ACKNOWLEDGMENT

The authors wish to thank Emilia Rosti for her suggestions which significantly improved this paper. We also thank the anonymous reviewers for many detailed and useful comments.

REFERENCES

[1] *Proceedings of the 4th IEEE International Conf. on Autonomic Computing (ICAC-07)*. IEEE Press, 2007.
[2] K. Whisnant, Z. Kalbarczyk, and R. K. Iyer, "A system model for dynamically reconfigurable software." *IBM Systems Journal*, vol. 42, no. 1, pp. 45–59, 2003.
[3] C. A. Floudas, *Nonlinear and Mixed Integer Optimization*. Oxford University Press, 1995.
[4] M. Bennani and D. A. Menascè, "Resource allocation for autonomic data centers using analytic performance," in *2nd IEEE Int'l Conf. on Autonomic Comp.*, IEEE Press, 2005.
[5] F. Baskett, K. M. Chandy, R. R. Muntz, and F. G. Palacios, "Open, closed, and mixed networks of queues with different classes of customers." *JACM*, vol. 22, no. 2, pp. 248–260, 1975.
[6] K. M. Chandy and D. Neuse, "Linearizer: A heuristic algorithm for queuing network models of computing systems," *Comm. ACM*, vol. 25, no. 2, pp. 126–134, 1982.
[7] W. C. Cheng and R. R. Muntz, "Bounding errors introduced by clustering of customers in closed product-form queuing networks," *JACM*, vol. 43, no. 4, pp. 641–669, 1996.
[8] P. J. Denning and J. P. Buzen, "The operational analysis of queueing network models," *ACM Comp. Surv.*, vol. 10, no. 3, pp. 225–261, 1978.
[9] C. H. Hsieh and S. Lam, "Two classes of performance bounds for closed queueing networks," *PEVA*, vol. 7, no. 1, pp. 3–30, 1987.
[10] J. Kriz, "Throughput bounds for closed queueing networks," *Perf. Eval.*, vol. 4, no. 1, pp. 1–10, 1984.
[11] R. R. Muntz and J. W. Wong, "Asymptotic properties of closed queueing network models," in *Proc. Ann. Princeton Conf. on Inf. Sci. and Sys.*, 1974, pp. 348–352.
[12] J. Zahorjan, K. C. Sevcik, D. L. Eager, and B. Galler, "Balanced job bound analysis of queueing networks," *Comm. ACM*, vol. 25, no. 2, pp. 134–141, 1982.
[13] E. Varki and L. W. Dowdy, "Analysis of fork-join queueing networks," in *Proc. of ACM SIGMETRICS 1996* pp. 232–241.
[14] A. Thomasian and J. Menon, "Raid5 performance with distributed sparing," *IEEE T. Par. Distrib. Syst.*, vol. 8, no. 6, pp. 640–657, 1997.
[15] E. Varki, A. Merchant, J. Xu, and X. Qiu, "Issues and challenges in the performance analysis of real disk arrays," *IEEE T. Par. Distrib. Syst.*, vol. 15, no. 6, pp. 559–574, 2004.
[16] S. Ceri, C. Gennaro, S. Paraboschi, and G. Serazzi, "Effective scheduling of detached rules in active databases," *IEEE T. Knowl. Data Eng*, vol. 15, no. 1, pp. 2–13, 2003.
[17] J. C. S. Lui, R. R. Muntz, and D. Towsley, "Computing performance bounds of fork-join parallel programs under a multiprocessing environment," *IEEE T. Par. Distrib. Syst.*, vol. 9, no. 3, pp. 295–311, 1998.

[18] G. Alvarez, E. Borowsky, S. Go, T. Romer, R. Becker-Szendy, R. Golding, A. Merchant, M. Spasojevic, A. Veitch, and J. Wilkes, "Minerva: An automated resource provisioning tool for large-scale storage systems," *ACM T. Comp. Sys.*, vol. 19, no. 4, pp. 483–518, 2001.

[19] E. Varki, "Mean value technique for closed fork-join networks," in *Proc. of ACM SIGMETRICS* 1999, pp. 103–112.

[20] H. Kobayashi, *Modelling and analysis: an introduction to system performance evaluation methodology*. Addison Wesley, 1978.

[21] M. Reiser and S. S. Lavenberg, "Mean-value analysis of closed multichain queueing networks," *JACM*, vol. 27, no. 2, pp. 312–322, 1980.

[22] G. Bolch, S. Greiner, H. de Meer, and K. S. Trivedi, *Queueing Networks and Markov Chains*. John Wiley and Sons, 1998.

[23] D. L. Eager and K. C. Sevcik, "Bound hierarchies for multiple-class queueing networks," *JACM*, vol. 33, no. 1, pp. 179–206, 1986.

[24] M. M. Srinivasan, "Successively improving bounds on performance measures for single class product form queueing networks," *IEEE T. Comp.*, vol. 36, no. 9, pp. 1107–1112, 1987.

[25] J. D. C. Little, "A proof of the queueing formula $L = \lambda W$," *Operations Research*, pp. 9:383–387, 1961.

[26] L. W. Dowdy, D. L. Eager, K. D. Gordon, and L. V. Saxton, "Throughput concavity and response time convexity," *Inform. Proc. Letters*, vol. 19, no. 4, pp. 209–212, 1984.

[27] G. Balbo and G. Serazzi, "Asymptotic analysis of multiclass closed queueing networks: Common bottlenecks," *PEVA*, vol. 26, no. 1, pp. 51–72, 1996.

[28] A. Harel, S. Namn, and J. Sturm, "Simple bounds for closed queueing networks," *Queueing Systems*, vol. 47, no. 1, pp. 125–135, 1999.

[29] L. Lipsky, C. H. Lieu, A. Tehranipour, and A. van de Liefvoort, "On the asymptotic behavior of time-sharing systems," *Comm. ACM*, vol. 25, no. 10, pp. 707–714, 1982.

[30] D. L. Eager and K. C. Sevcik, "Performance bound hierarchies for queueing networks," *ACM T. Comp. Sys.*, vol. 1, no. 2, 99–115, 1983.

[31] E. Varki, "Response time analysis of parallel computer and storage systems," *IEEE T. Par. Distrib. Syst.*, vol. 12(11), pp. 1146–1161, 2001.

[32] E. Varki, L.W. Dowdy, "Quick performance bounding techniques for computer and storage systems with parallel resources", under review (available at: http://citeseer.ist.psu.edu/669758.html).

[33] P. J. Schweitzer, "Approximate analysis of multiclass closed networks of queues," in *Int'l Conf. on Stoch. Control and Optim.*, 1979, pp. 25–29.

[34] Y. Bard, "Some extensions to multiclass queueing network analysis," in *Proc. 3rd Int'l Symp. on Model. and Perf. Eval. of Comp. Sys.*, M. Arato, A. Butrimenko, and E. Gelenbe, Eds., 1979, pp. 51–62.

[35] Z. Liu, L. Wynter, C. H. Xia, and F. Zhang, "Parameter inference of queueing models for it systems using end-to-end measurements." *PEVA*, vol. 63, no. 1, pp. 36–60, 2006.

[36] P. Bonami et al., *An Algorithmic Framework for Convex Mixed Integer Nonlinear Programs*, IBM Research Report RC23771 (W0511-023), 2005.

[37] R. Fourer, D. M. Gay, and B. W. Kernighan, *AMPL – A Modeling Language for Mathematical Programming*. The Scientific Press, 1993.

[38] Y. Chen, A. Das, W. Qin, A. Sivasubramaniam, Q. Wang, and N. Gautam, "Managing server energy and operational costs in hosting centers," in *Proc. of SIGMETRICS 2005*, ACM Press, 2005, pp. 303–314.

[39] B. Urgaonkar, G. Pacifici, P. J. Shenoy, M. Spreitzer, and A. N. Tantawi, "An analytical model for multi-tier internet services and its applications," in *Proc. of SIGMETRICS 2005*, ACM Press, 2005, pp. 291–302.

[40] S. Kounev and A. P. Buchmann, "Performance modeling and evaluation of large-scale J2EE applications," in *Int. CMG Conference*. Computer Measurement Group, 2003, pp. 273–283.

[41] 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 2006 Conf.*, IEEE Press, pp. 119–120. Tool and paper available at http://jmt.sourceforge.net.

APPENDIX A: SUMMARY OF EXPERIMENT PARAMETERS

TABLE IV: $M = 4$, $L_1 = 0.10$, $L_2 = 0.10$, $L_3 = 0.09$, $L_4 = 0.08$, $Z = 0$, $N \in [2, 5, 10, 15, 20, 30, 40, 60, 80]$.
TABLE V: $M = 4$, $L_1 = 0.10$, $L_2 = 0.10$, $L_3 = 0.09$, $L_4 = 0.08$, $Z = 1$, $N \in [2, 5, 10, 15, 20, 30, 40, 60, 80]$.
TABLE VI: $M = 4$, $L_1 = 0.10$, $L_2 = 0.10$, $L_3 = 0.05$, $L_4 = 0.04$, $Z = 0$, $N \in [2, 5, 10, 15, 20, 30, 40, 60, 80]$.
TABLE VII: $M = 4$, $L_1 = 0.10$, $L_2 = 0.10$, $L_3 = 0.05$, $L_4 = 0.04$, $Z = 1$, $N \in [2, 5, 10, 15, 20, 30, 40, 60, 80]$.
TABLE VIII AND TABLE X: M random in $[5, 50]$, L_i random in $[0, 1]$ for all $1 \leq i \leq M$, $Z = 0$, $N \in [2, 1000]$.
TABLE IX AND TABLE XI: M random in $[5, 50]$, L_i random in $[0, 1]$ for all $1 \leq i \leq M$, $Z = 10 \max_i L_i$, $N \in [2, 1000]$.
TABLE XII: fork-join, $M = 3$, $L_1 = 3$, $L_2 = 2$, $L_3 = 1$, $P_1 = 3$, $P_2 = 2$, $P_3 = 1$, $N \in [2, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50]$.
TABLE XIII: fork-join, $M = 3$, $L_1 = 3$, $L_2 = 2$, $L_3 = 1$, $P_1 = 1$, $P_2 = 2$, $P_3 = 3$, $N \in [2, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50]$.
TABLE XIV: $M = 2$, $L_1 = p_{sto1}20$, $L_2 = p_{sto2}30$, $Z = 13$, $N = 10$, $p_{sto1} = 1 - p_{sto2} = \{0.50, 0.85, 0.99\}$.
TABLE XV: fork-join, $L_{A_1} = p_1 12.98$, $L_{A_2} = p_2 13.64$, $L_{A_3} = p_3 24.42$, $L_{A_4} = p_4 24.42$, $L_{A_5} = p_5 6.82$, $L_{A_6} = p_6 12.21$, $L_{B_1} = 5.32$, $L_D = 1.12$, $N \in [2, 50]$.

Giuliano Casale received the MS and the PhD degrees in computer engineering from the Politecnico di Milano, Milan, Italy, in 2002 and 2006, respectively. From January 2007 he is a postdoctoral research associate at the College of William and Mary, Williamsburg, Virginia, where he studies the performance impact of burstiness in systems. In Fall 2004 he was a visiting scholar at UCLA studying bounds for queueing networks. His research interests include queueing theory, Markov processes, performance optimization, and simulation. He is a member of the ACM, IEEE, and officer of the IEEE Hampton Roads Section.

Richard R. Muntz was born in Jersey City, New Jersey. He received the BEE from Pratt Institute in 1963, the MEE from New York University in 1966, and the Ph.D. in Electrical Engineering from Princeton University in 1969. He is a member of the Board of Directors for SIGMETRICS and is the current chair of IFIP Working Group 7.3. He is a past associate editor for the Journal of the ACM and was editor-in-chief of ACM Computing Surveys from 1992 to 1995. He is a member of Sigma Xi, Tau Beta Pi, a Fellow of the Association for Computing Machinery, and a Fellow of the IEEE. He is the recipient of the 2006 ACM Sigmetrics Achievement Award. His current research interests are scientific database systems, multimedia storage and database systems, data mining and computer system performance evaluation.

Giuseppe Serazzi is Professor at the Computer Science Department of Politecnico di Milano, Milan, Italy, since 1991. He received a Laurea in Mathematics from University of Pavia, Pavia, Italy, in 1969. He spent four years at the University of Milano before going to Politecnico. During 1978-87 he has been associate professor at the Department of Mathematics, University of Pavia. His current interests include workload characterization, modeling and other topics related to the performance evaluation of computer systems and networks.