A Unifying Framework for the Approximate Solution of Closed Multiclass Queuing Networks

Paolo Cremonesi, Paul J. Schweitzer, and Giuseppe Serazzi, *Member*, *IEEE Computer Society*

Abstract—Queuing network models of modern computing systems must consider a large number of components (e.g., Web servers, DB servers, application servers, firewall, routers, networks) and hundreds of customers with very different resource requirements. The complexity of such models makes the application of exact solution techniques prohibitively expensive, motivating research on approximate methods. This paper proposes an interpolation-matching framework that allows a unified view of approximate solution techniques for closed product-form queuing networks. Depending upon the interpolating functional form and the matching populations selected, a large versatile family of new approximations can be generated. It is shown that all the known approximation strategies, including Linearizer, are instances of the interpolation-matching framework. Furthermore, a new approximation technique, based on a third-order polynomial, is obtained using the interpolation-matching framework. The new technique is shown to be more accurate than other known methods.

Index Terms—Queuing network models, approximate solution techniques, multiclass workloads.

1 Introduction

In recent years, computer-communication systems have become increasingly complex. Client-server architectures and Internet-based distributed systems involve interconnection of a large number of components via LANs and WANs. In addition, there has been a vast proliferation of workloads having widely varying resource requirements (consider, e.g., multimedia loads). Therefore, representative queuing network models of actual systems must deal with a large number of both components and customer classes. The computational requirements for solving such networks with exact solution techniques (when applicable) are known to be prohibitive, motivating the search for approximate techniques.

We consider product-form closed queuing networks with constant rate servers [3]. A convenient class of approximate methods are the so-called *local-based* techniques that approximate the queue lengths in the neighbor of a given population vector in order to transform the MVA recursion into a closed fixed-point system of equations. Several local approximation techniques have appeared in the literature [2], [5], [6], [7], [8], [11], [15] that were developed almost independently over a few years and very little systematic comparison of their accuracy and complexity has been carried out, partial exceptions being [13], [15], [16]. As a result, comparing their complexity and accuracy is hard if at all clear how it should be carried out. Furthermore, the description of the different techniques that have appeared thus far focuses on algorithmic rather than theoretical issues.

In this paper, we present a new analytical framework for the local approximation solution of product-form closed queuing network models. The new approach is called the *interpolation-matching technique* (IMT) since it is based on the local approximation of mean queue lengths by means of *interpolating* functions calibrated by *matching* at specific populations. The approach followed is not simply the description of a new approximation method, rather, it is the definition of a general framework able to reproduce *all* of the local approximation methods. The most popular local approximation methods (Schweitzer-Bard [2], [11], Chow [5], AQL [15], Linearizer [6], Queue Line [16], Fraction Line [16]) are, in fact, instances of the interpolation-matching technique, obtained by using different interpolating functions. The specific interpolating function for each method will be exhibited below.

The analysis of the interpolating functions used in IMT allows identification of the features associated with such functions that impact on the accuracy of the methods. For example, a second-order polynomial yields greater accuracy of the Linearizer over the Schweitzer-Bard method, which uses a first-order polynomial. Similarly, a third-order polynomial improves upon Linearizer accuracy, as we will show in the paper. The *interpolation matching technique* has several appealing properties, among which are:

- the possibility of analyzing, under a common analytical framework, all the local approximation methods presented in the literature and also the possibility of capturing the relationships among different methods;

- *P. Cremonesi and G. Serazzi are with the Dip. Elettronica e Informazione, Politecnico di Milano, P.za L. da Vinci 32, 20122 Milano, Italy. E-mail: {cremonesi, serazzi}@elet.polimi.it.*
- *P.J. Schweitzer is with the W.E. Simon Graduate School of Business Adminstration, University of Rochester, Rochester, NY 14627. E-mail: schweitzer@simon.rochester.edu.*

Manuscript received 15 Sept. 1999; revised 11 Feb. 2002; accepted 13 Feb. 2002.
For information on obtaining reprints of this article, please send e-mail to: tc@computer.org, and reference IEEECS Log Number 110588.

- the description of a systematic procedure for generating various classes of approximation techniques (linear, quadratic, cubic, splines, etc.) with different accuracy objectives as well as different computational complexity.

Two types of interpolation are identified and described in the paper: interpolation on "servers" and interpolation on "servers and classes." The former technique provides an approximation for the aggregate queue lengths of customers at the servers, while the latter approximates the queue lengths individually for each class of customers.

The paper is organized as follows: Section 2 reviews the most popular solution techniques, both exact methods and local approximations. Sections 3 and 4 are devoted to the description of the interpolation-matching techniques (IMT) on "servers" and on "servers and classes," respectively. The major reason for including Section 3 is to make Section 4 easier to read and understand since interpolation on "servers and classes" dominates interpolation on "servers" alone with respect to accuracy for a given computational order. In Section 5, a new approximate method (Linearizer++) is proposed. Section 6 suggests some guidelines for the choice of interpolating functions and matching populations. Section 7 is devoted to conclusions and future work.

2 Review of Solution Techniques

2.1 Exact Solution Techniques

Consider a product-form closed queuing network (BCMP) [3] with M constant rate servers and R customer classes. Let class r population be denoted by N_r, the population vector by $\underline{N} \equiv (N_1, N_2, \ldots, N_R)$, and the total population by $N \equiv \sum_{r=1}^{R} N_r$. Customers are not allowed to change class. The other parameters of the model are:[1]

- S_{ir} = mean service time of a class r customer for each visit to server i (if i is FCFS, this must be independent of r);
- V_{ir} = mean number of visits of a class r customer to server i;
- $L_{ir} \equiv V_{ir}S_{ir}$ = mean load placed on server i by a customer of class r.

Let $\mathbf{M} \equiv \{1, 2, \ldots, M\}$ be the set of server indexes and $\mathbf{R} \equiv \{1, 2, \ldots, R\}$ be the set of customer class indexes. Let $\mathbf{D_C}$ denote the set of Delay Centers and $\mathbf{Q_C}$ denote the set of Queuing Centers (first-come-first-served or processor sharing or last-come-first-served preemptive resume or service in random order [14]). Note that $\mathbf{M} \equiv \mathbf{D_C} \cup \mathbf{Q_C}$. Typically, the performance measures of interest in solving queuing network models are throughputs and response times by class. These values will be denoted by:

- $Q_{ir}(\underline{N})$ = mean number of class r customers at server i (either in service or on queue);
- $Q_i(\underline{N}) \equiv \sum_{r=1}^{R} Q_{ir}(\underline{N})$ = mean total number of customers at server i.

Such measures can be computed exactly, in a recursive fashion, using mean value analysis (MVA) [9], as:

$$Q_{ir}(\underline{N}) = \frac{N_r L_{ir}[1 + \chi(i) Q_i(\underline{N} - \underline{1}_r)]}{\sum_{j=1}^{M} L_{jr}\left[1 + \chi(j) Q_j(\underline{N} - \underline{1}_r)\right]} \tag{1}$$

with $Q_i(\underline{0}) = 0$, where $\underline{1}_r$ denotes the unit vector along the r axis and the χ-function is defined as follows:

$$(\underline{1}_r) \equiv \delta_{rs} \equiv \begin{cases} 1 & r = s \\ 0 & r \neq s \end{cases} \quad \chi(i) \equiv \begin{cases} 1 & i \in \mathbf{Q_C} \\ 0 & i \in \mathbf{D_C}. \end{cases}$$

The class r queue lengths computation can be combined into a single recursion to yield the total queue lengths at server i:

$$Q_i(\underline{N}) = \sum_{r=1}^{R} \frac{N_r L_{ir}[1 + \chi(i) Q_i(\underline{N} - \underline{1}_r)]}{\sum_{j=1}^{M} L_{jr}[1 + \chi(j) Q_j(\underline{N} - \underline{1}_r)]} \tag{2}$$

with initial conditions $Q_i(\underline{0}) = 0$. Note that, by construction, the conservation laws $\sum_{i=1}^{M} Q_{ir}(\underline{N}) = N_r$ are automatically satisfied by (1). The exact solution for queue lengths (1) has time computational complexity $O(MR\prod_{r=1}^{R}(1 + N_r))$ and space complexity $O(M\prod_{r \neq r_{\max}}(1 + N_r))$ (where r_{max} is a class with the maximum number of customers), which usually makes it impractical if $R > 4$. The same computational complexity and conclusion hold when the convolution algorithm [4], [10] is used to solve constant-rate product-form networks.

2.2 Approximation Techniques

All the approximation techniques try to compute $Q_i(\underline{N})$ at a *given population* $\underline{N}^0$ by guessing the behavior of $Q_i(\underline{N})$ for $\underline{N}$ in the neighborhood of $\underline{N}^0$. We will briefly review the best-known local approximations, namely Chow [5], Schweitzer-Bard [2], [11], Linearizer [6], and AQL [15]. In [12], a more detailed review is reported. In Sections 3 and 4, we will show that all the considered approximation techniques can be seen as instantiations of the interpolation-matching technique.

2.2.1 The Chow Algorithm

The Chow algorithm [5] uses the following approximation:

$$Q_i(\underline{N} - \underline{1}_r) \approx Q_i(\underline{N}) \quad \text{for } \underline{N} \text{ near } \underline{N}^0. \tag{3}$$

By substituting (3) into the MVA recursion (2), we obtain the following set of M equations in closed (nonrecursive) fixed-point form:

$$Q_i(\underline{N}^0) = \sum_{r=1}^{R} \frac{N_r^0 L_{ir}\left[1 + \chi(i) Q_i(\underline{N}^0)\right]}{\sum_{j=1}^{M} L_{jr}\left[1 + \chi(j) Q_j(\underline{N}^0)\right]} \tag{4}$$

for the M unknowns $Q_i(\underline{N}^0)$. Unlike the MVA, the time complexity to solve (4), and related fixed point approximations, by successive substitutions, does not depend strongly on $\underline{N}^0$. However, the resulting Chow approximations for $Q_i(\underline{N}^0)$ are sufficiently accurate only for large populations. The major source of inaccuracy is that the approximation (3) does not preserve the total number of customers. The sum of the total queue lengths with one fewer customer of class r equals N,

1. Indexes i and j will range from 1 to M and indexes r, s, and t will range from 1 to R unless otherwise explicitly stated.

$$\sum_{i=1}^{M} Q_i(\underline{N}-\underline{1}_r) \approx \sum_{i=1}^{M} Q_i(\underline{N}) = N$$

instead of the correct $N-1$.We will henceforth refer to such a characteristic by saying that the approximation *does not scale* with the population. Similarly, we will check if

$$\sum_{i=1}^{M} Q_{ir}(\underline{N}-\underline{1}_s) = N_r - \delta_{rs}.$$

2.2.2 The Schweitzer-Bard Algorithm

The Schweitzer-Bard algorithm [2], [11] (henceforth called SB) improves upon the Chow algorithm by approximating $Q_{ir}(\underline{N}-\underline{1}_s)$ instead of $Q_i(\underline{N}-\underline{1}_s)$, with a function of $Q_{ir}(\underline{N})$. The approximation employed is

$$Q_{ir}(\underline{N}-\underline{1}_s) \approx Q_{ir}(\underline{N}) - \delta_{rs}\frac{Q_{ir}(\underline{N})}{N_r} \quad \text{for } \underline{N} \text{ near } \underline{N}^0. \tag{5}$$

By substituting approximation (5) into recursion (1), we obtain a nonrecursive set of MR nonlinear equations in closed fixed-point form

$$Q_{ir}(\underline{N}^0) = \frac{N_r^0 L_{ir}\left\{1+\chi(i)\left[Q_i(\underline{N}^0) - \frac{Q_{ir}(\underline{N}^0)}{N_r^0}\right]\right\}}{\sum_{j=1}^{M} L_{jr}\left\{1+\chi(j)\left[Q_j(\underline{N}^0) - \frac{Q_{jr}(\underline{N}^0)}{N_r^0}\right]\right\}} \tag{6}$$

for the MR unknowns $Q_{ir}(\underline{N}^0)$. As with the Chow approximation, the time to solve the fixed point set (6) by successive approximations does not depend strongly on $\underline{N}^0$. Note that the second term in (5) ensures that the approximation scales properly with the population. Note also that the set of equations (6) can be reduced to $M+R$ equations in $M+R$ unknowns, so very large systems can be analyzed [11].

2.2.3 The Linearizer Algorithm

With the definition

$$D_{rit}(\underline{N}) \equiv \frac{Q_{ir}(\underline{N}-\underline{1}_t)}{N_r-\delta_{rt}} - \frac{Q_{ir}(\underline{N})}{N_r},$$

the approximation adopted in the Linearizer method [6] is

$$D_{rit}(\underline{N}-\underline{1}_s) \approx D_{rit}(\underline{N}) \quad \text{for } \underline{N} \text{ near } \underline{N}^0. \tag{7}$$

By combining approximation (7) with the equations obtained by applying the MVA recursion (1) at the population vectors $\underline{N}^0$ and $\underline{N}^0-\underline{1}_s$, we obtain a set of $MR(2R+1)$ nonlinear equations in a fixed-point form in the $MR(2R+1)$ unknowns $Q_{ir}(\underline{N}^0)$, $Q_{ir}(\underline{N}^0-\underline{1}_s)$, and $D_{rit}(\underline{N}^0)$. Note that the following estimation is used

$$Q_{it}(\underline{N}^0-\underline{1}_r-\underline{1}_s) \approx (N_t^0-\delta_{tr}-\delta_{ts}) \left[\frac{Q_{it}(\underline{N}^0-\underline{1}_s)}{N_t^0-\delta_{ts}} + \frac{Q_{it}(\underline{N}^0-\underline{1}_r)}{N_t^0-\delta_{tr}} - \frac{Q_{it}(\underline{N}^0)}{N_t^0}\right], \tag{8}$$

which is symmetric in r and s and which scales properly with population.

2.2.4 The AQL Algorithm

The AQL (Aggregated Queue Length) method is based on an approximation similar to that of Linearizer, but, instead of the per-class queue lengths $Q_{ir}(\underline{N})$, aggregate per-server queue lengths $Q_i(\underline{N})$ are now considered. This significantly reduces the computational complexity with some decrease in the precision of the method. Defining

$$D_{ir}(\underline{N}) \equiv \frac{Q_i(\underline{N}-\underline{1}_r)}{N-1} - \frac{Q_i(\underline{N})}{N}$$

(where $N=\sum_{r=1}^{R} N_r$), the approximation adopted is

$$D_{ir}(\underline{N}-\underline{1}_s) \approx D_{ir}(\underline{N}) \quad \text{for } \underline{N} \text{ near } \underline{N}^0.$$

The resulting fixed-point problem consists of $M(2R+1)$ equations for the $M(2R+1)$ unknowns $Q_i(\underline{N}^0)$, $Q_i(\underline{N}^0-\underline{1}_r)$, and $D_{ir}(\underline{N}^0)$. Note that the following estimation is used

$$Q_i(\underline{N}^0-\underline{1}_r-\underline{1}_s) \approx (N^0-2) \left[\frac{Q_i(\underline{N}^0-\underline{1}_s)}{N^0-1} + \frac{Q_i(\underline{N}^0-\underline{1}_s)}{N^0-1} - \frac{Q_i(\underline{N}^0)}{N^0}\right], \tag{9}$$

which is symmetric in r and s and which scales properly with population.

3 Interpolation on "Servers"

This method, referred to as *interpolation on servers*, approximates the aggregate queue length $Q_i(\underline{N})$ of customers at each server. The key idea is the approximation

$$Q_i(\underline{N}) \approx T_i(\underline{N};\underline{P}_i) \quad \text{for } \underline{N} \text{ near } \underline{N}^0, \tag{10}$$

where the *trial functions* $T_i(\underline{N};\underline{P}_i)$ (also called *approximating functions*) are completely known except for MJ parameters, namely M vectors $\underline{P}_i$ each with J elements. Table 1 shows some examples of possible approximating functions. Note that the more parameters in a trial function, the worse the computational complexity of the resulting algorithm. A good trial function should yield a good approximation, defined in (10), with a small number of parameters. When the trial functions are linear in the parameters (see, e.g., the first three rows in Table 1), there is a considerable simplification in the formulation of the method, as described in Section 4.4.

In order to specify the MJ parameters $\underline{P}_i$, we need to impose MJ conditions upon approximation (10). Under the hypothesis that the trial functions $T_i(\underline{N};\underline{P}_i)$ are chosen properly (i.e., approximation (10) holds at least for $\underline{N}$ near $\underline{N}^0$), we can specify J population vectors $\mathbf{N} \equiv \{\underline{N}^{(1)};\underline{N}^{(2)};\ldots;\underline{N}^{(J)}\}$ (from here on referred to as the *matching populations*) and we can insist that MVA recursion (2) be satisfied at these J populations:[2]

2. In what follows, indices u, v, and w range from 1 to J unless otherwise explicitly specified.

TABLE 1
Examples of Trial Functions (f_u and g_v Are Arbitrary Functions of $\underline{N}$)

Approximation type	Approximating function	Parameters
Constant	$T_i(\underline{N},\underline{P}_i)=A_i$	$\underline{P}_i=\{A_i\}$ $J=1$
Linear	$T_i(\underline{N},\underline{P}_i)=A_i+\sum_{r=1}^{R}B_{ir}N_r$	$\underline{P}_i=\{A_i, B_{ir}\}$ $J=R+1$
Linear basis expansion	$T_i(\underline{N},\underline{P}_i)=\sum_{u=1}^{J}P_{iu}\,\mathrm{f}_u(\underline{N})$	$\underline{P}_i=\{P_{iu}\}$ J arbitrary
Generalized Padé expansion	$T_i(\underline{N},\underline{P}_i)=\frac{\sum_{w=1}^{K}A_{iu}\,\mathrm{f}_u(\underline{N})}{\sum_{v=1}^{L}B_{iv}\,\mathrm{g}_v(\underline{N})}$	$\underline{P}_i=\{A_{iu}, B_{iv}\}$ K and L arbitrary $J=K+L$

$$T_i\left(\underline{N}^{(u)},\underline{P}_i\right)=\sum_{r=1}^{R}\frac{N_r^{(u)}L_{ir}\left[1+\chi(i)T_i\left(\underline{N}^{(u)}-\underline{1}_r,\underline{P}_i\right)\right]}{\sum_{j=1}^{M}L_{jr}\left[1+\chi(j)T_j\left(\underline{N}^{(u)}-\underline{1}_r,\underline{P}_i\right)\right]}\quad(11)$$

$\underline{N}^{(u)}\in\mathbf{N}$.

Note that $\underline{N}^0$ itself can be one point of the matching population set. The set (11) consists of MJ equations for the MJ parameters $\underline{P}_i$. Once the parameters $\underline{P}_i$ have been computed by solving the set (11), queue lengths $Q_i(\underline{N})$ can be estimated for any $\underline{N}$ from (10). The matching condition expressed by (11) assumes that the approximating functions (10) *interpolate* the queue lengths in between the matching populations and fit perfectly at the matching populations. Therefore, the approximating functions $T_i(\underline{N};\underline{P}_i)$ are called *interpolating functions*. The general steps of the interpolation strategy are:

Step 1. Specify the approximating functions $T_i(\underline{N};\underline{P}_i)$ for the queue lengths (10), with MJ parameters $\underline{P}_i$ to be determined;

Step 2. Specify the matching populations

$$\mathbf{N}\equiv\left\{\underline{N}^{(1)},\underline{N}^{(2)},\ldots,\underline{N}^{(J)}\right\};$$

Step 3. Solve the set of MJ equations (11) for the MJ parameters $\underline{P}_i$;

Step 4. Trivially evaluate the queue lengths from (10) once the $\underline{P}_i$s are known.

Variants of (11) consist of choosing $\underline{P}_i$ to match both functions and derivatives of (2) or to do a least squares fit over a prescribed range of populations. In all cases, the resulting equations (11) are nonlinear and must be solved by an iterative procedure. By taking J large enough and employing a sufficiently rich set of trial functions $T_i(\underline{N};\underline{P}_i)$, high accuracy can be achieved.

The following subsections give examples of the general interpolation strategy using different trial functions from Table 1. These particular trial functions were chosen because they correspond to some of the well-known approximation techniques described in Section 2 and they exhibit increasing generality and accuracy. Note that the case of linear basis expansion encompasses all the previous ones and yet is just a particular case of the general Padè approximation described in [1], [17].

3.1 Constant Approximation Function (Derivation of Chow Approximation)

The constant approximation is

$$Q_i(\underline{N})\approx T_i(\underline{N},\underline{P}_i)\equiv A_i\quad(\underline{P}_i=\{A_i\})\ \text{ for } \underline{N}\text{ near }\underline{N}^0,\quad(12)$$

which corresponds to using M parameters $\{A_i\}$ (i.e., M vectors $\underline{P}_i$ each with $J=1$ element). Therefore, only one matching population vector $\underline{N}^{(1)}$ is needed in order to find the M parameters A_i. If we take

$$\mathbf{N}\equiv\left\{\underline{N}^{(1)}\right\}=\{\underline{N}^0\},\quad(13)$$

the set of equations (11) reduces to

$$A_i=\sum_{r=1}^{R}\frac{N_r^0L_{ir}[1+\chi(i)A_i]}{\sum_{j=1}^{M}L_{jr}\left[1+\chi(j)A_j\right]}.\quad(14)$$

Equation (14) is identical with Chow's (4). It can be solved by successive substitutions, after which the mean queue lengths are estimated from (12) as $Q_i(\underline{N}^0)\approx A_i$. In the general procedure, we must work harder to solve the equations (11) for the parameters. The parameters are then substituted into (10) to estimate the mean queue lengths. The Chow approximation shortens both steps because (11) is conveniently in fixed-point form and because the parameters themselves are the mean queue lengths. We exploit this idea below.

3.2 Linear Approximation Function

The linear approximation is given by

$$Q_i(\underline{N}) \approx T_i(\underline{N}, \underline{P}_i) = A_i + \sum_{r=1}^{R} B_{ir} N_r \quad (\underline{P}_i = \{A_i, B_{ir}\}),$$
$$\text{for } \underline{N} \text{ near } \underline{N}^0, \tag{15}$$

and uses $M(R+1)$ parameters (i.e., M vectors $\underline{P}_i$ with $J = R+1$ elements each). Therefore, we need to fix $R+1$ population vectors. A convenient choice is

$$\mathbf{N} \equiv \{\underline{N}^0, \underline{N}^0 - \underline{1}_r\}. \tag{16}$$

Imposing the matching conditions (11) leads to a non-fixed-point problem for the parameters $\underline{P}_i$, which can then be transformed into a fixed-point problem (as shown in the Appendix). After solving for the parameters $\underline{P}_i$, the queue lengths are evaluated using (15). From a practical point of view, it may be difficult to invert the transformation from $T_i(\underline{N}^{(u)}; \underline{P}_i)$ to $\underline{P}_i$ (as was done in the Appendix). A better approach is to carefully choose the trial functions $T_i(\underline{N}; \underline{P}_i)$ such that $T_i(\underline{N}^{(u)}; \underline{P}_i)$ *coincide* with the *MJ* parameters $\underline{P}_i$. In this case, (11) is automatically a fixed-;point problem for the $\underline{P}_i$s. This objective can be achieved by requiring (15) to hold exactly at the matching populations (16), which leads to the following values for the coefficients $[A_i; B_{ir}]$:

$$\begin{cases} A_i = Q_i(\underline{N}^0) - \sum_{r=1}^{R} N_r^0 \left[Q_i(\underline{N}^0) - Q_i(\underline{N}^0 - \underline{1}_r) \right] \\ B_{ir} = Q_i(\underline{N}^0) - Q_i(\underline{N}^0 - \underline{1}_r). \end{cases} \tag{17}$$

By inserting (17) into (15), we obtain the new form T_i^* of the trial functions

$$Q_i(\underline{N}) \approx T_i^*(\underline{N}, \underline{P}_i^*) = Q_i(\underline{N}^0) - \sum_{r=1}^{R} (N_r - N_r^0) \left[Q_i(\underline{N}^0 - \underline{1}_r) - Q_i(\underline{N}^0) \right]$$
$$\text{for } \underline{N} \text{ near } \underline{N}^0, \tag{18}$$

which has the following desired properties:

- The original parameters $\underline{P}_i = \{A_i; B_{ir}\}$ are replaced by new parameters

$$\underline{P}_i^* = \{ Q_i(\underline{N}^0),\ Q_i(\underline{N}^0 - \underline{1}_r) \} = \left\{ Q_i\left(\underline{N}^{(u)}\right) \right\} \quad \underline{N}^{(u)} \in \mathbf{N}$$

  that are the mean queue lengths themselves and which, unlike the original parameters, have an intuitive meaning;
- The left and righthand sides of (18) match perfectly at populations **N**. Indeed, the righthand side of (18) is the unique first degree interpolating Lagrangian polynomial passing through the points $\{\underline{N}^{(u)}\}$.

The matching condition (11) applied to (18) yields the following fixed-point problem for the $Q_i(\underline{N}^{(u)})$ (compare with (46) to see complete equivalence):

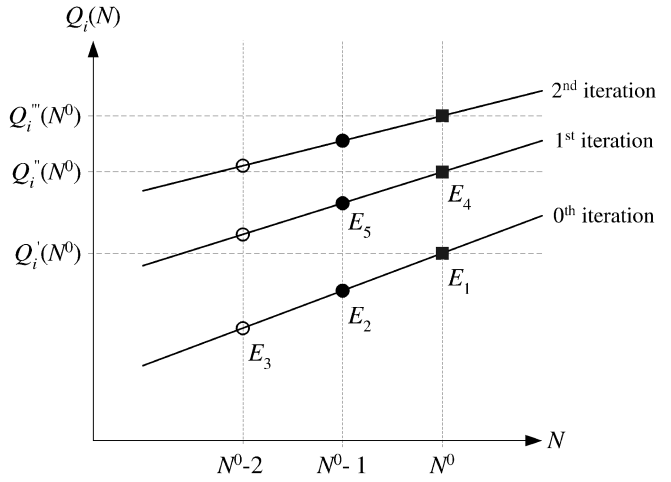

Fig. 1. Representation of IMT applied to a single-class case with linear approximating function.

$$\begin{cases} Q_i(\underline{N}^0) = \sum_{r=1}^{R} \frac{N_r^0 L_{ir} \left[1 + \chi(i) Q_i(\underline{N}^0 - \underline{1}_r)\right]}{\sum_{j=1}^{M} L_{jr} \left[1 + \chi(j) Q_j(\underline{N}^0 - \underline{1}_r)\right]} \\ Q_i(\underline{N}^0 - \underline{1}_s) = \\ \sum_{r=1}^{R} \frac{(N_r^0 - \delta_{rs}) L_{ir} \left\{1 + \chi(i) \left[Q_i(\underline{N}^0 - \underline{1}_r) + Q_i(\underline{N}^0 - \underline{1}_s) - Q_i(\underline{N}^0)\right]\right\}}{\sum_{j=1}^{M} L_{jr} \left\{1 + \chi(j) \left[Q_j(\underline{N}^0 - \underline{1}_r) + Q_j(\underline{N}^0 - \underline{1}_s) - Q_j(\underline{N}^0)\right]\right\}} \end{cases} \tag{19}$$

for the $M(R+1)$ unknowns $Q_i(\underline{N}^0)$ and $Q_i(\underline{N}^0 - \underline{1}_r)$. As desired, the fixed-point form has been obtained and the new parameters $Q_i(\underline{N}^{(u)})$ have a direct intuitive interpretation. Note that the approximation used in (19) is:

$$Q_i(\underline{N}^0 - \underline{1}_r - \underline{1}_s) \approx Q_i(\underline{N}^0 - \underline{1}_r) + Q_i(\underline{N}^0 - \underline{1}_s) - Q_i(\underline{N}^0), \tag{20}$$

which is a direct consequence of (18). In addition, (20) is symmetric in r and s and scales properly with the population. A graphical representation of the fixed-point problem generated by a linear approximation function in the single-class case is shown in Fig. 1. The horizontal axis represents the population N; the vertical axis shows the queue length for station i. Approximation (18) in this simple case is the equation of a straight line. The IMT is used according to the following strategy (see Fig. 1, N^0 is the population of interest):

Step 1. We start with estimate E_1 and E_2 of $Q_i(N^0)$ and $Q_i(N^0 - 1)$;

Step 2. Linear interpolation based on (18) gives E_3;

Step 3. The fixed-point problem (19) gives E_4 and E_5.

Steps 2 and 3 are repeated until values of $Q_i(N^0)$ converge (black boxes in Fig. 1).

The approach of using trial functions with the queue lengths themselves as parameters is advantageous because it always yields fixed-point equations for the Q_is. In the following, it will be used whenever possible. Approximation (15) can be extended to polynomials of higher degree, so (18) will contain higher-degree Lagrangian interpolation polynomials.

3.3 Another Linear Approximation (Derivation of AQL Approximation)

The AQL method is similar to the linear approximation method of Section 3.2, except that $Q_i(\underline{N})/N$ rather than $Q_i(\underline{N})$ is considered as linear. Equation (15) is replaced by

$$\frac{Q_i(\underline{N})}{N} \approx \frac{T_i(\underline{N}, \underline{P}_i)}{N} = A_i + \sum_{r=1}^{R} B_{ir} N_r \quad (\underline{P}_i = \{A_i, B_{ir}\}) \quad \text{for } \underline{N} \text{ near } \underline{N}^0. \tag{21}$$

If (21) is required to hold exactly at the matching populations (16), approximation (20) is replaced by

$$\frac{Q_i(\underline{N}^0 - \underline{1}_r - \underline{1}_s)}{N^0 - 2} = \frac{Q_i(\underline{N}^0 - \underline{1}_r)}{N^0 - 1} + \frac{Q_i(\underline{N}^0 - \underline{1}_s)}{N^0 - 1} - \frac{Q_i(\underline{N}^0)}{N^0},$$

which is exactly the AQL approximation defined by (9).

3.4 The Linear Basis Expansion

More general trial functions than the ones described in the previous sections can be used. For example, one could use a Padé approximation (see Table 1) or, more generally, any smooth (analytic) function. In this section, we use a linear combination of independent basis functions. Let us consider J known, linearly independent, basis functions $\mathrm{f}_u(\underline{N})$ and attempt a trial function

$$Q_i(\underline{N}) \approx T_i(\underline{N}, \underline{P}_i) = \sum_{u=1}^{J} P_{iu} f_u(\underline{N}) \quad \text{for } \underline{N} \text{ near } \underline{N}^0, \tag{22}$$

with the MJ parameters P_{iu} to be determined. This choice of $T_i(\underline{N}; \underline{P}_i)$ includes both the approximation functions described above as special cases (e.g., the constant approximation is obtained when $J = 1$ and $\{\mathrm{f}_u(\underline{N})\} = \{1\}$ and the linear approximation is obtained when $J = R + 1$ and $\{\mathrm{f}_u(\underline{N})\} = \{1; N_r\}$). Higher-degree polynomials may be accommodated as well. As in the previous case, it is possible to change the parameters so that the new parameters are the queue lengths themselves at some given populations, namely by specifying J population vectors $\mathbf{N} \equiv \{\underline{N}^{(1)}; \underline{N}^{(2)}; \ldots; \underline{N}^{(J)}\}$ and requiring (22) to hold exactly for these populations. The $\{P_{it}\}$ are then specified in terms of the $Q_i(\underline{N}^{(v)})$ by the linear equations

$$Q_i\left(\underline{N}^{(v)}\right) = \sum_{u=1}^{J} P_{iu} \ \mathrm{f}_u\left(\underline{N}^{(v)}\right) \quad \underline{N}^{(v)} \in \mathbf{N}. \tag{23}$$

Because they are linear, the set of equations (23) can be inverted to express the P_{iu}s in terms of the queue lengths at the matching populations. We introduce the following matrix notation:

$$\begin{aligned} &\mathbf{P} \equiv [P_{iu}] && M \times J \text{ matrix, unknown} \\ &\mathbf{X} \equiv [X_{iv}] \equiv \left[Q_i\left(\underline{N}^{(v)}\right)\right] && M \times J \text{ matrix, unknown} \\ &\mathbf{Y} \equiv [Y_{uv}] \equiv \left[\mathrm{f}_u\left(\underline{N}^{(v)}\right)\right] && J \times J \text{ matrix, known.} \end{aligned} \tag{24}$$

Equation (23) can now be rewritten as $\mathbf{X} = \mathbf{P} \times \mathbf{Y}$. Matrix $\mathbf{Y}$ is always nonsingular because the basis functions are assumed to be linearly independent. Let $\mathbf{W} \equiv [W_{vu}] \equiv \mathbf{Y}^{-1}$. Therefore, approximation (22) becomes

$$Q_i(\underline{N}) \approx \sum_{u=1}^{J} \sum_{v=1}^{J} X_{iv} W_{vu} \mathrm{f}_u(\underline{N}) \quad \text{for } \underline{N} \text{ near } \underline{N}^0. \tag{25}$$

By substituting (25) into the MVA recursion (11) and matching at $N^{(v)} \in \mathbf{N}$, we obtain the fixed–point problem

$$X_{iv} = \sum_{r=1}^{R} \frac{N_r^{(v)} L_{ir} \left[1 + \chi(i) \sum_{u=1}^{J} \sum_{z=1}^{J} X_{iz} W_{zu} \mathrm{f}_u\left(\underline{N}^{(v)} - \underline{1}_r\right)\right]}{\sum_{j=1}^{M} L_{jr} \left[1 + \chi(j) \sum_{u=1}^{J} \sum_{z=1}^{J} X_{jz} W_{zu} \mathrm{f}_u\left(\underline{N}^{(v)} - \underline{1}_r\right)\right]} \quad 1 \le v \le J, \tag{26}$$

consisting of MJ equations for the MJ unknowns $\mathbf{X}$. These correspond to (11) with $\mathbf{P} \equiv \mathbf{X}$ and are the generalization of (19). Once the matrix $[X_{iv}]$ is available, it is possible to estimate $Q_i(\underline{N})$ for any $\underline{N}$ using (25). Note again that the $[X_{iv}]$ have an immediate interpretation as queue lengths (see (24)).

The possibility exists of replacing (26) by an alternative scheme where (25) is inserted into both sides of (2) and matched at a set of J populations different from $\mathbf{N}$. We recommend against this because the equations, unlike (26), are not in fixed-point form; we have not found any advantages in accuracy, provided $\mathbf{N}$ was chosen reasonably.

4 Interpolation on "Servers" and "Classes"

The interpolation-matching technique described in the previous section has an intrinsic limitation: It can approximate only the aggregate queue lengths Q_i, but not the queue lengths per customer class Q_{ir}. Therefore, it can encompass algorithms like AQL, but not SB or Linearizer. In this section, we extend the formulation of the interpolation matching technique in order to approximate the Q_{ir}s individually. The extension parallels the formulation in Section 3. The basic approximation used in this case is

$$Q_{ir}(\underline{N}) \approx T_{ir}(\underline{N}, \underline{P}_{ir}) \quad \text{for } \underline{N} \text{ near } \underline{N}^0,$$

where the trial functions T_{ir} depend upon MR vectors $\underline{P}_{ir}$, each with J real elements. We specify J matching populations $\mathbf{N} \equiv \{\underline{N}^{(1)}; \underline{N}^{(2)}; \ldots; \underline{N}^{(J)}\}$ and impose the matching conditions

$$T_{ir}\left(\underline{N}^{(u)}, \underline{P}_{ir}\right) = \frac{N_r^{(u)} L_{ir} \left(1 + \sum_{s=1}^{R} T_{is}\left(\underline{N}^{(u)} - 1_r, \underline{P}_{is}\right)\right)}{\sum_{j=1}^{M} L_{jr} \left(1 + \sum_{s=1}^{R} T_{js}\left(\underline{N}^{(u)} - 1_r, \underline{P}_{js}\right)\right)} \quad \underline{N}^{(u)} \in \mathbf{N}. \tag{27}$$

Equation (27) defines a set of MRJ equations for the MRJ parameters. The following subsections illustrate the trial functions chosen to reproduce approximation methods corresponding to the SB and Linearizer. The more general case of linear basis functions includes all the previous ones and will be used in Section 5 to derive a new cubic approximation.

4.1 Linear Approximation Function (Derivation of Schweitzer-Bard)

We consider the linear approximation,

$$Q_{ir}(\underline{N}) \approx T_{ir}(\underline{N}, \underline{P}_{ir}) = A_{ir}N_r \quad \text{for } \underline{N} \text{ near } \underline{N}^0, \tag{28}$$

that uses MR parameters $\{A_{ir}\}$ (i.e., $J = 1$) and R trial functions $\{\mathrm{f}_r(\underline{N})\} = \{N_r\}$. Changing the parameters from $\{A_{ir}\}$ to $\{Q_{ir}(\underline{N}^0)\}$ and requiring that (28) holds exactly at $\underline{N}^0$, we must have $A_{ir} = Q_{ir}(\underline{N}^0)/\underline{N}^0_r$. Therefore, (28) becomes

$$Q_{ir}(\underline{N}) \approx T_{ir}(\underline{N}, \underline{P}_{ir}) = \frac{Q_{ir}(\underline{N}^0)}{N^0_r} N_r \quad \text{for } \underline{N} \text{ near } \underline{N}^0. \tag{29}$$

Using (29) to compute the queue length at the population vector $\underline{N}^0 - \underline{1}_s$, we obtain

$$Q_{ir}(\underline{N}^0 - \underline{1}_s) \approx Q_{ir}(\underline{N}^0) - \delta_{rs}\frac{Q_{ir}(\underline{N}^0)}{N^0_r}, \tag{30}$$

which is exactly the SB approximation (5). The fixed-point problem obtained using (30) in the matching conditions (27) becomes identical to (6).

4.2 Quadratic Approximation Function (Derivation of Linearizer)

In this section, we will show that, with a proper choice of basis functions and matching populations, the IMT is equivalent to the Linearizer approximation described in Section 2.2.3. Let us consider a quadratic expansion for the queue lengths

$$\begin{aligned} &Q_{ir}(\underline{N}) \approx T_{ir}(\underline{N}, \underline{P}_{ir}) = A_{ir}N_r + \sum_{t=1}^{R} B_{irt}N_rN_t \\ &(\underline{P}_{ir} = \{A_{ir}, B_{irt}\}) \quad \text{for } \underline{N} \text{ near } \underline{N}^0 \end{aligned} \tag{31}$$

with $MR(R+1)$ coefficients P_{ir} to be determined (i.e., MR vectors each with $J = R+1$ real elements) and basis functions $\{N_r; N_rN_s\}$. This approximation yields the one used by Linearizer if we take

$$\mathbf{N} \equiv \{\underline{N}^0, \underline{N}^0 - \underline{1}_r\} \tag{32}$$

as matching populations. As before, it is possible to change the parameters by requiring (31) to hold exactly for the $(R+1)$ populations (32)

$$\begin{cases} Q_{ir}(\underline{N}^0) = T_{ir}(\underline{N}^0, \underline{P}_{ir}) \\ Q_{ir}(\underline{N}^0 - 1_s) = T_{ir}(\underline{N}^0 - 1_s, \underline{P}_{ir}). \end{cases} \tag{33}$$

These conditions lead to the new trial functions

$$\begin{aligned} &Q_{ir}(\underline{N}) \approx T^*_{ir}(\underline{N}, \underline{P}_{ir}) = \\ &N_r\left\{\frac{Q_{ir}(\underline{N}^0)}{N^0_r} + \sum_{t=1}^{R}\left(N_t - N^0_t\right)\left[\frac{Q_{ir}(\underline{N}^0)}{N^0_r} - \frac{Q_{ir}(\underline{N}^0 - 1_t)}{N^0_r - \delta_{rt}}\right]\right\}, \end{aligned}$$

with the new parameters $\underline{P}^*_{ir} = \{Q_{ir}(\underline{N}^0); Q_{ir}(\underline{N}^0 - \underline{1}_t)\}$. Setting $\underline{N} = \underline{N}^0 - \underline{1}_s - \underline{1}_r$ leads to the Linearizer approximation (8).

4.3 The Linear Basis Expansion

The generalization of the above approximation functions is obtained by approximating the queue length Q_{ir} for any class r at a given station i with the following linear combination of J basis functions $\mathrm{f}_{ru}(\underline{N})$:

$$Q_{ir}(\underline{N}) \approx \sum_{u=1}^{J} P_{riu}\mathrm{f}_{ru}(\underline{N}) \quad \text{for } \underline{N} \text{ near } \underline{N}^0, \tag{34}$$

where $\{P_{riu}\}$ is a set of MRJ coefficients to be determined. As already pointed out, it is convenient to change parameters from P_{riu} to $Q_{ir}(\underline{N} \in \mathbf{N})$, where $\mathbf{N} \equiv \{\underline{N}^{(1)}; \underline{N}^{(2)}; \ldots; \underline{N}^{(J)}\}$ is a specified set of J populations (which may include $\underline{N}^0$ itself). By insisting upon equality between the left and right sides of (34) at these populations, we obtain the MRJ equations

$$Q_{ir}\left(\underline{N}^{(v)}\right) \equiv \sum_{u=1}^{J} P_{riu}\mathrm{f}_{ru}\left(\underline{N}^{(v)}\right) \quad \underline{N}^{(v)} \in \mathbf{N}, \tag{35}$$

which are sufficient to determine the MRJ P_{riu}. Let's introduce the following notation:

$$\begin{aligned} &\mathbf{P}_r \equiv [P_{riu}] \\ &\mathbf{X}_r \equiv [X_{riv}] \equiv \left[Q_{ir}\left(\underline{N}^{(v)}\right)\right] \\ &\mathbf{Y}_r \equiv [Y_{ruv}] \equiv \left[f_{ru}\left(\underline{N}^{(v)}\right)\right]. \end{aligned}$$

For each r, $\mathbf{P}_r$ and $\mathbf{X}_r$ are unknown $M \times R$ matrices, while $\mathbf{Y}_r$ is a known $J \times J$ matrix. Equation (35) can then be rewritten as

$$\mathbf{X}_r = \mathbf{P}_r \times \mathbf{Y}_r.$$

The $\mathbf{Y}_r$ matrices are nonsingular if we assume the basis functions f_{ru} for each r to be linearly independent. Let $\mathbf{W}_r \equiv [W_r]_{uv} \equiv \mathbf{Y}_r^{-1}$, so $\mathbf{P}_r = \mathbf{X}_r \times \mathbf{W}_r$. Approximation (35) becomes

$$Q_{ir}(\underline{N}) \approx \sum_{u=1}^{J}\sum_{v=1}^{J} X_{riv}[W_r]_{vu}\mathrm{f}_{ru}(\underline{N}) \quad \text{for } \underline{N} \text{ near } \underline{N}^0. \tag{36}$$

By substituting the above expression into the MVA recursion, we obtain the following fixed-point problem (in the MRJ unknowns $\mathbf{X}_r$):

$$\begin{aligned} X_{riv} = \frac{N^{(v)}_r L_{ir}\left[1 + \chi(i)\sum_{s=1}^{R}\sum_{u=1}^{J}\sum_{z=1}^{J} X_{siz}[W_s]_{zu}\mathrm{f}_{su}\left(\underline{N}^{(v)} - \underline{1}_r\right)\right]}{\sum_{j=1}^{M} L_{jr}\left[1 + \chi(j)\sum_{s=1}^{R}\sum_{u=1}^{J}\sum_{z=1}^{J} X_{sjz}[W_s]_{zu}\mathrm{f}_{su}\left(\underline{N}^{(v)} - \underline{1}_r\right)\right]} \\ 1 \le v \le J. \end{aligned} \tag{37}$$

Once the $\mathbf{X}_r$ matrices are available, it is possible to estimate the $Q_{ir}(\underline{N})$ for any $\underline{N}$ using (36).

4.4 Generalization of the IMT Technique

The most general case of IMT uses the following approximation

$$Q_{ir}(\underline{N}) \approx T_{ir}(\underline{N}, \underline{P}_{ir}), \tag{38}$$

where the parameters $\underline{P}_{ir}$ enter the trial function in a *nonlinear* way. In all cases, the matching procedure gives a *nonlinear* set of equations for the parameters $\underline{P}_{ir}$. We have two sets of equations, the interpolation equations (38):

$$Q_{ir}\left(\underline{N}^{(v)}\right) \approx T_{ir}\left(\underline{N}^{(v)}, \underline{P}_{ir}\right) \tag{39}$$

and the matching equations (27)

$$T_{ir}\left(\underline{N}^{(v)}, \underline{P}_{ir}\right) = \frac{N_r^{(v)} L_{ir}\left[1 + \chi(i) \sum_{s=1}^{R} T_{is}\left(\underline{N}^{(v)} - \underline{1}_r, \underline{P}_{is}\right)\right]}{\sum_{j=1}^{M} L_{jr}\left[1 + \chi(j) \sum_{s=1}^{R} T_{js}\left(\underline{N}^{(v)} - \underline{1}_r, \underline{P}_{js}\right)\right]}. \tag{40}$$

These are $2MRJ$ equations for the $2MRJ$ unknowns Q_{ir} and $\underline{P}_{ir}$ and, in general, they are not in fixed-point form for the $\underline{P}_{ir}$s. A straightforward procedure is to first solve (40) for $\underline{P}_{ir}$ and subsequently use (39) to calculate Q_{ir}.

By contrast, in the previous examples, (39) was inverted explicitly to obtain $\underline{P}_{ir}$ as a function of Q_{ir} and then (40) was interpreted as a set of simultaneous equations for Q_{ir} similar to (4), (19), (26), and (37), thus obtaining a fixed-point problem for the queue lengths at the matching populations. The advantages of such an approach are:

- The first step of the strategy, inverting (39), must be executed only once when the new IMT algorithm is created, so its computational cost can be ignored;
- We obtain a fixed-point problem, which is much easier to solve than the more general equation (40);
- The unknowns of the fixed-point problem are the queue lengths, which have an immediate meaning, unlike the parameters $\underline{P}_{ir}$.

The most important feature, however, is that the number of equations equals the number of parameters, so the matching conditions (40) serve uniquely to determine the parameters $\underline{P}_{ir}$, even when they are not in fixed-point form. Two straightforward extensions are to let the basis functions $\mathrm{f}_{ru}(\underline{N})$ in (34) depend on i and let the matching points $\mathrm{N} \equiv \{\underline{N}^{(v)}\}$ in (35) depend on i and r. The notation becomes more complicated, but the underlying ideas remain the same. With the proposed extensions, it is also possible to treat the more general case of Padè approximations.

5 An Example of IMT Application: The Linearizer++ Algorithm

To illustrate the ease of constructing new approximation algorithms using the IMT, we propose a cubic expression as the interpolating function for the queue lengths. The equivalence between SB and linear interpolation function and between Linearizer and quadratic interpolation function has motivated the investigation of higher order interpolation polynomial functions in order to find more accurate approximating methods. The specific form of cubic interpolation—compare with (28) and (31)—is

$$Q_{ir}(\underline{N}) \approx A_{ir} N_r + \sum_{s=1}^{R} B_{irs} N_r N_s + \sum_{t=1}^{R} \sum_{s \le t}^{R} C_{irst} N_r N_s N_t$$
$$(\underline{P}_{ir} = \{A_{ir}, B_{irs}, C_{irst}\}) \quad \text{for } \underline{N} \text{ near } \underline{N}^0.$$

There are $MR[(R+1)(R+2)/2]$ unknown coefficients $\underline{P}_{ir}$ (i.e., MR parameter vectors each with $J = (R+1(R+2)/2$ real elements). Choosing the following matching population vectors:

$$\mathbf{N} \equiv \left\{\underline{N}^{(0)}, \underline{N}^{(0)} - \underline{1}_r, \underline{N}^{(0)} - \underline{1}_r - \underline{1}_s\right\},$$

leads to a new approximation method (referred to as *Linearizer++*) which shows much better accuracy than Linearizer. Table 2 shows the results obtained for over 4,800 randomly generated models (100 models for each specification of: M = 2, 4, 6, 8 servers, R = 2, 3 classes of customers, and N= 5, 10, 50, 100, 500, 1,000 total customers). Since queuing centers usually contribute more to the errors than delay centers do, only queuing centers have been considered in the experiments. The values of the loadings range from 0 to 100. All the parameters were selected independently and their values are uniformly distributed over their respective ranges. The table presents the average $\mu(e)$, the standard deviation $\sigma(e)$, and the maximum $\max(e)$ of the normalized error e of each model, defined as

$$e = \max_{i,r}\left|\frac{Q_{ir}(\underline{N})_{\mathrm{MVA}} - Q_{ir}(\underline{N})_{\mathrm{APP}}}{N_r}\right|,$$

where $Q_{ir}(\underline{N})_{\mathrm{MVA}}$ and $Q_{ir}(\underline{N})_{\mathrm{APP}}$ are the exact and approximate solutions, respectively, for the considered model. Let us remark that the value of e is the maximum error on the per-class customers at the servers of each model. As the table shows, the error of Linearizer++ is, on average, 1/3 of the error of Linearizer, with half the standard deviation and max error of Linearizer. In all the models, the errors e of Linearizer++ are lower then the ones of Linearizer. In general, the errors increase with the number of stations and with the number of classes.

The complexity of the three methods (SB, Linearizer, and Linearizer++) is proportional to the number of iterations n needed to solve the fixed-point problem, and to the complexity of one iteration. The complexity of each iteration is $O(M\ R^2\ J^3)$ with $J = 1$ for SB, $J = (R+1)$ for Linearizer, and $J = (R+1)(R+2)/2$ for Linearizer++. The number of iterations clearly depends on the starting point and on the termination criteria and, a priori, it may also be influenced by the number of classes, the number of stations, and the number of customers.

Fig. 2 shows the average number of iterations (and the standard deviation) as a function of the number of stations M and the number of classes R, for the three local approximation techniques. Each point in the figure is the average of 1,800 values, corresponding to 100 models, with six different numbers of customers (N = 5, 10, 50, 100, 500, 1,000) and three different approximation techniques (SB, Linearizer, Linearizer++).

An interesting result that can be drawn from the figure, is that the number of customers N and the approximate method adopted have a very limited influence on the number of iterations required for the method to converge.

TABLE 2
Normalized Errors of Three Local Approximation Techniques

	Schweitzer-Bard			Linearizer			Linearizer++		
2 classes	$\mu(e)$	$\sigma(e)$	max(e)	$\mu(e)$	$\sigma(e)$	max(e)	$\mu(e)$	$\sigma(e)$	max(e)
2 stations	0.0063	0.0112	0.0470	0.0005	0.0012	0.0112	0.0001	0.0003	0.0029
4 stations	0.0134	0.0148	0.0708	0.0011	0.0016	0.0092	0.0003	0.0008	0.0090
6 stations	0.0160	0.0159	0.0654	0.0014	0.0021	0.0146	0.0004	0.0009	0.0073
8 stations	0.0168	0.0153	0.0602	0.0015	0.0021	0.0204	0.0005	0.0010	0.0083
Global	0.0131	0.0150	0.0708	0.0011	0.0018	0.0204	0.0003	0.0008	0.0090
3 classes	$\mu(e)$	$\sigma(e)$	max(e)	$\mu(e)$	$\sigma(e)$	max(e)	$\mu(e)$	$\sigma(e)$	max(e)
2 stations	0.0102	0.0145	0.0746	0.0007	0.0014	0.0107	0.0001	0.0005	0.0048
4 stations	0.0176	0.0166	0.0922	0.0014	0.0020	0.0184	0.0004	0.0008	0.0057
6 stations	0.0196	0.0164	0.0723	0.0017	0.0023	0.0212	0.0005	0.0010	0.0090
8 stations	0.0207	0.0168	0.0838	0.0019	0.0023	0.0128	0.0006	0.0012	0.0089
Global	0.0170	0.0166	0.0922	0.0014	0.0021	0.0212	0.0004	0.0009	0.0090

On the other side, the number of iterations increases with the number of stations M and decreases with the number of classes R.

6 Design Guidelines for New Approximation Techniques

This section describes the guidelines for the design of new approximation techniques using IMT. Any IMT algorithm is specified by choosing:

- an integer number J (i.e., the number of unknown vector parameters), which is a dominant factor in the computational complexity of the method;
- the interpolating functions $T_{ir}(\underline{N}, \underline{P}_{ir})$, with *MJR* parameters $\underline{P}_{ir}$ to be determined;
- the matching points $\mathbf{N} \equiv \{\underline{N}^{(1)}, \underline{N}^{(2)}, \ldots, \underline{N}^{(J)}\}$.

The issues regarding each of these choices are described in the following subsections.

6.1 Choice of J

Experience suggests that good choices for J are $J = R + 1$ and $J = (R+1)(R+2)/2$ because they satisfy all three of the previously stated general rules. Note that these choices for J are implicitly made by AQL, Linearizer, and Linearizer++.

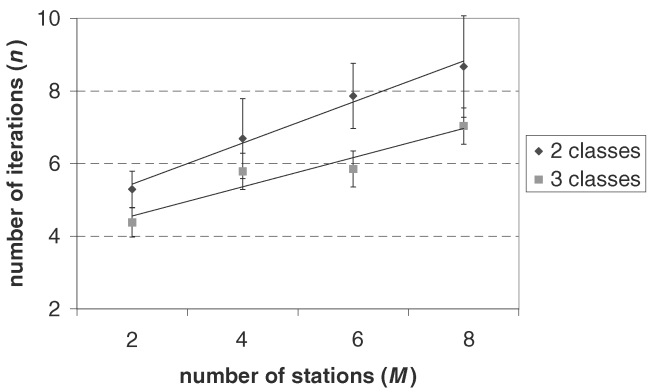

Fig. 2. Average number of iterations as a function of the number of stations and the number of classes for the three local approximation techniques.

6.2 Choice of the Interpolation Functions

The largest degree of freedom when defining a new IMT method arises with the choice of the interpolating functions. Experience suggests that:

- Choosing the trial functions to be finite-degree polynomials leads to convenient Lagrangian interpolating polynomials and gives good insight into the interpolation method. All the known approximation methods use polynomials. Increasing the degree of the polynomial from constant to linear then to quadratic and cubic is observed to increase the method accuracy, with Linearizer++ being the most precise. However, the increase of accuracy with the increase of the polynomial degree is not expected to continue indefinitely because of the numerical instability that characterizes high degree polynomial approximations. We have observed that instability arises with a degree greater than 3. In addition, high degree polynomials are computationally unattractive as the number of customer classes increases. The instability of the high degree polynomials arises because the matching population set now include points very far from $\underline{N}^0$ and such points do not contribute significantly to the accuracy of estimating $Q_i(\underline{N}^0)$.
- It is desirable to choose the trial functions $T_{ir}(\underline{N}; \underline{P}_{ir})$ such that $\sum_{i=1}^{M} T_{ir}(\underline{N}, \underline{P}_{ir}) = N_r$ is an identity (i.e., the scaling condition described in Section 2.2.1 must be verified) and similarly choose trial functions $T_i(\underline{N}; \underline{P}_i)$ such that $\sum_{i=1}^{M} T_i(\underline{N}, \underline{P}_i) = \sum_{r=1}^{R} N_r$ is an identity. Inspection of (18), (29), and (31) shows that SB, Linearizer, and Linearizer++ satisfy this condition and that it is straightforward to accomplish this.
- SB, Linearizer, and Linearizer++ use Lagrangian polynomials for interpolating the functions Q_{ir}/N_r rather than Q_{ir} and provide good results. A reason for this is that the function Q_{ir}/N_r is smoother than Q_{ir}, so a low degree polynomial satisfactorily approximates it. More generally, it appears desirable to attempt interpolations on functions of the queue lengths that are known to be smoother than the

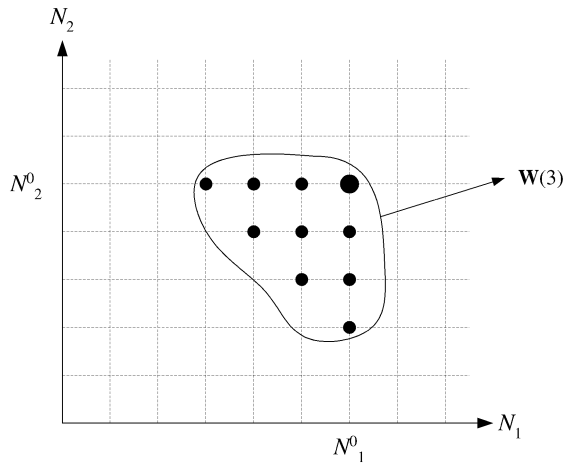

Fig. 3. The "wake set" in a two-class network.

queue lengths themselves. This is the motivation for the use of generalized Padé approximations.

- Other choices of trial functions besides linear basis functions (Lagrangian interpolation polynomials) have been explored. In particular, generalized Padé approximations appear to be both convenient and flexible. The choice of trial functions determines the ease of solving (27), which are usually not in fixed-point form.

6.3 Choice of the Matching Points

The choice of the matching points $\underline{N}^{(t)}$ have a great impact on the accuracy of an IMT method. Our intuition, based on the experiments performed, suggests that the matching points:

- Should be in some way "fair" to the different classes of customers;
- Should be chosen near $\underline{N}^0$: IMT methods are local, thus they provide good results locally (i.e., at points near $\underline{N}^0$);
- Should be below $\underline{N}^0$ (i.e., for networks with fewer customers than $\underline{N}^0$) because of the well-known property of the upward MVA recursion to smooth errors [8];
- Should constitute a "compact" set, without gaps in it (see the following section).

A good choice of matching points is

$$\mathbf{N} \equiv \left\{\underline{N}^0, \underline{N}^0 - \underline{1}_r\right\},$$

which is the choice made by Linearizer and AQL, or

$$\mathbf{N} \equiv \left\{\underline{N}^{(0)}, \underline{N}^{(0)} - \underline{1}_r, \underline{N}^{(0)} - \underline{1}_r - \underline{1}_s\right\},$$

which is the choice made by Linearizer++. Good choices of the matching points are

$$\mathbf{N} = \mathbf{W}(v) \equiv \left\{\, \underline{N} \mid \quad \left(N_r N^0_r \ \forall r\right) \ \wedge \ \left(N \geq N^0 - v\right) \right\}, \quad (41)$$

where v refers to the maximum distance from $\underline{N}^0$. Let us remark that all the points contained in the set $\mathbf{W}(v)$ influence $Q_i(\underline{N}^0)$ in the MVA recursion. We refer to such set of points as the *wake set* for the point $\underline{N}^0$. Fig. 3 shows the wake set for a two-class network with $v = 3$.

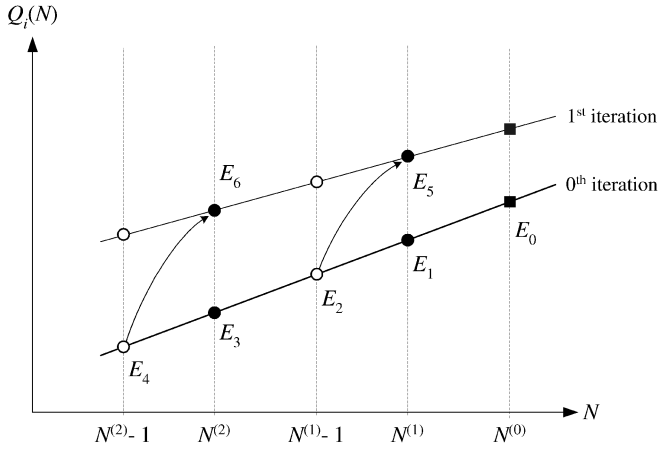

Fig. 4. Graphical representation of IMT applied at a simple single-class case with a linear approximating function and a generic matching population set.

6.3.1 Characteristics of the Matching Population Set

We found good results if our choice of $\mathbf{N} = \{\underline{N}^{(1)}; \underline{N}^{(2)}; \ldots; \underline{N}^{(J)}\}$ met three criteria:

1. $\underline{N}^0 \in \mathbf{N}$;
2. Each $\underline{N}^{(u)}$ lies in the wake set of $\underline{N}^0$;
3. Each $\underline{N}^{(u)}$ lies in the wake set of other numbers $\underline{N}^{(v)}$.

The explanation is that the interpolation formula $Q_i(\underline{N}) = T_i(\underline{N}; \underline{P}_i)$ is implicitly assumed to hold perfectly at *all* the points $\{\underline{N}^{(1)}; \underline{N}^{(1)} - \underline{1}_r; \underline{N}^{(2)}; \underline{N}^{(2)} - \underline{1}_r; \ldots\}$ $\forall r$. This is too much to expect. Compare, for example, Fig. 1 and Fig. 4, both showing a graphical representation of linear approximation (15) with a single ($R = 1$) and two parameters ($J = 2$). Fig. 4 shows the most general (and worst) choice of matching population set. Starting with estimate E_1 and E_2 of the queue lengths at populations $\underline{N}^{(1)}$ and $\underline{N}^{(2)}$, linear interpolation gives E_3 and E_4, which are used to iterate the method. Linear interpolation also gives E_0, which is used to test the method convergence. One is therefore attempting, in the worst case, to fit a straight line through five points $\{N^{(0)}; N^{(1)}; N^{(1)} - 1; N^{(2)}; N^{(2)} - 1\}$. One of these five points is eliminated by rule 1 and another by rules 2 and 3. The result is to have to fit a straight line through three points $\{N^{(0)}; N^{(0)} - 1; N^{(0)} - 2\}$, a task which is much more likely to succeed, as shown in Fig. 1.

7 Conclusions

In this paper, we have presented an interpolation-matching technique that provides an analytical framework under which all the known local approximation techniques for the solution of closed product-form queuing networks can be described and analyzed.

We have identified the characteristics and the parameters of approximating methods that impact on their complexity and accuracy. In particular, we have shown that differences in the accuracy are related to the specific forms of the corresponding interpolating functions. As an example of the power of the IMT technique, throughout its application, we have constructed a new algorithm that exhibits better accuracy results than previous ones.

Guidelines are given for the construction of new approximation techniques with different accuracy and computational complexity by properly selecting the matching points and the interpolating functions.

Our current investigation focuses on choosing noninteger populations in order to be closer to target population; on choosing the function parameters to locally match both functions and derivatives of the MVA recursion or to do a least squares fit over a prescribed range of populations; on choosing the trial functions and extending the results to load-dependent servers.

APPENDIX

This appendix shows how to obtain a fixed–point problem for the parameters $\{A_{ir}; B_{ir}\}$ from the linear approximation defined in Section 3.2:

$$Q_i(\underline{N}) \approx T_i(\underline{N}, \underline{P}_i) = A_i + \sum_{r=1}^{R} B_{ir} N_r(\underline{P}_i = \{A_i, B_{ir}\}), \quad \text{for } \underline{N} \text{ near } \underline{N}^0, \tag{42}$$

when using the matching points

$$\mathbf{N} \equiv \{\underline{N}^0, \underline{N}^0 - \underline{1}_r\}.$$

If, for each $\underline{N} \in \mathbf{N}$, we write the matching condition (11) together with the approximation (42), we obtain the following equations:

$$A_i + \sum_{r=1}^{R} B_{ir} N_r^0 = \sum_{s=1}^{R} \frac{N_s^0 L_{is}\left\{1+\chi(i)\left[A_i + \sum_{r=1}^{R} B_{ir}\left(N_r^0 - \delta_{rs}\right)\right]\right\}}{\sum_{j=1}^{M} L_{js}\left\{1+\chi(j)\left[A_j + \sum_{r=1}^{R} B_{jr}\left(N_r^0 - \delta_{rs}\right)\right]\right\}} \tag{43}$$

$$A_i + \sum_{r=1}^{R} B_{ir}\left(N_r^0 - \delta_{rt}\right) = \sum_{s=1}^{R} \frac{\left(N_s^0 - \delta_{st}\right) L_{is}\left\{1+\chi(i)\left[A_i + \sum_{r=1}^{R} B_{ir}\left(N_r^0 - \delta_{rs} - \delta_{rt}\right)\right]\right\}}{\sum_{j=1}^{M} L_{js}\left\{1+\chi(j)\left[A_j + \sum_{r=1}^{R} B_{jr}\left(N_r^0 - \delta_{rs} - \delta_{rt}\right)\right]\right\}}. \tag{44}$$

These are $M(R+1)$ equations for the $M(R+1)$ unknowns $\{A_i; B_{ir}\}$. They are neither linear nor in fixed-point form. The fixed-point form can be obtained by adding and subtracting (43) and (44):

$$\begin{cases} A_i = \sum_{s=1}^{R} \frac{N_s^0 L_{is}\left\{1+\chi(i)\left[A_i+\sum_{r=1}^{R} B_{ir}\left(N_r^0-\delta_{rs}\right)\right]\right\}}{\sum_{j=1}^{M} L_{js}\left\{1+\chi(j)\left[A_j+\sum_{r=1}^{R} B_{jr}\left(N_r^0-\delta_{rs}\right)\right]\right\}} - \sum_{r=1}^{R} B_{ir} N_r^0 \\ B_{it} = \sum_{s=1}^{R} \frac{\left(N_s^0-\delta_{st}\right) L_{is}\left\{1+\chi(i)\left[A_i+\sum_{r=1}^{R} B_{ir}\left(N_r^0-\delta_{rs}-\delta_{rt}\right)\right]\right\}}{\sum_{j=1}^{M} L_{js}\left\{1+\chi(j)\left[A_j+\sum_{r=1}^{R} B_{jr}\left(N_r^0-\delta_{rs}-\delta_{rt}\right)\right]\right\}} + \\ \qquad - \sum_{s=1}^{R} \frac{N_s^0 L_{is}\left\{1+\chi(i)\left[A_i+\sum_{r=1}^{R} B_{ir}\left(N_r^0-\delta_{rs}\right)\right]\right\}}{\sum_{j=1}^{M} L_{js}\left\{1+\chi(j)\left[A_j+\sum_{r=1}^{R} B_{jr}\left(N_r^0-\delta_{rs}\right)\right]\right\}}. \end{cases} \tag{45}$$

Successive substitutions on this set of equations will presumably converge to the parameters $\{A_i; B_{ir}\}$ and then (15) will produce an estimate of $Q_i(\underline{N})$.

ACKNOWLEDGMENTS

This work was supported in part by the MIUR COFIN 2001 project. The authors would like to thank Francesco Schiavoni for his contribution to this work and the reviewers for their helpful comments.

REFERENCES

[1] G.A. Baker and P. Graves-Morris, "Padé Approximations," *Encyclopedia of Math. and Its Applications,* second ed., vol. 59, Cambridge, U.K.: Cambridge Univ. Press, 1996.

[2] Y. Bard, "Some Extensions to Multiclass Queueing Network Analysis," *Proc. Fourth Int'l Symp. Modeling and Performance Evaluation of Computer Systems,* A. Butrimenko M. Arato, and E. Gelenbe, eds., pp. 51-62, 1979.

[3] F. Baskett, K.M. Chandy, R.R. Muntz, and R. Palacios, "Open, Closed and Mixed Networks of Queues with Different Classes of Customers," *J. ACM,* vol. 22, no. 2, pp. 248-260, 1975.

[4] J.P. Buzen, "Computational Algorithms for Closed Queueing Networks with Exponential Servers," *Comm. ACM,* vol. 16, no. 9, pp. 527-531, 1973.

[5] W.M. Chow, "Approximations for Large Scale Closed Queueing Networks," *Performance Evaluation,* vol. 3, no. 1, pp. 1-12, 1983.

[6] K.M. Chandy and D. Neuse, "Linearizer: A Heuristic Algorithm for Queueing Network Models of Computing Systems," *Comm. ACM,* vol. 25, no. 2, pp. 126-134, 1982.

[7] D.L. Eager and K.C. Sevcik, "Performance Bound Hierarchies for Queueing Networks," *ACM Trans. Computer Systems,* vol. 1, no. 2, pp. 99-115, 1983.

[8] D.L. Eager and K.C. Sevcik, "Bound Hierarchies for Multiple-Class Queueing Networks," *J. ACM,* vol. 33, no. 4, pp. 179-206, 1986.

[9] M. Reiser and S. Lavenberg, "Mean-Value Analysis of Closed Multichain Queueing Networks," *J. ACM,* vol. 27, no. 2, pp. 313-322, 1980.

[10] M. Reiser and H. Kobayashi, "Queueing Networks with Multiple Closed Chains: Theory and Computational Algorithms," *IBM J. Research and Development,* vol. 19, pp. 283-294, 1975.

[11] P.J. Schweitzer, "Approximate Analysis of Multiclass Closed Queueing Networks of Queues," *Proc. Int'l Conf. Stochastic Control and Optimization,* Apr. 1979.

[12] P.J. Schweitzer, "A Survey of Mean Value Analysis, Its Generalizations, and Applications, for Networks of Queues," *Proc. Second Int'l Workshop Netherlands Nat'l Network for the Math. on Operations Research,* Feb. 1991.

[13] E. de Souza e Silva and R.R. Muntz, "A Note on the Computational Cost of the Linearizer Algorithm," *IEEE Trans. Computers,* vol. 39, no. 6, pp. 840-842, June 1990.

[14] J.R. Spirn, "Queueing Networks with Random Selection for Service," *IEEE Trans. Software Eng.,* vol. 3, no. 1, pp. 287-289, 1979.

[15] J. Zahorjan, D.L. Eager, and H.M. Sweillam, "Accuracy, Speed and Convergence of Approximate Mean Value Analysis," *Performance Evaluation,* vol. 8, no. 4, pp. 255-270, 1988.

[16] H. Wang and K.C. Sevcik, "Experiences with Improved Approximate Mean Value Analysis Algorithms," *Proc. 10th Int'l Conf. Modeling Techniques and Tools,* R. Puigjaner, N.N. Savino, and B. Serra, eds., pp. 280-291, 1998.
[17] P. Cremonesi, F. Schiavoni, P.J. Schweitzer, and G. Serazzi, "An Interpolation-Matching Framework for the Approximate Solution of Closed Multiclass Queuing Networks," Internal Report N 27/1997, Dipartimento di Elettronica e Informazione, Politecnico di Milano, 1997.

Paolo Cremonesi received the MSc degree in aerospace engineering in 1992 and the PhD degree in computer science in 1996, both from the Politecnico di Milano, Milan, Italy. He is currently an assistant professor of computer science with the Politecnico di Milano. His current research interests include high-performance computing modeling and other topics related to the performance evaluation of computer systems and networks.

Paul J. Schweitzer is a professor of business administration at the W.E. Simon Graduate School of Business Administration at the University of Rochester, New York, where he teaches in the general areas of information technology, stochastic processes, management science, and operations management. His current research areas include Markovian decision processes and iterative aggregation theory. He has worked in the field of performance evaluation for more than 30 years and is the inventor of approximate Mean Value Analysis, the topic of this paper.

Giuseppe Serazzi received the Laurea degree in mathematics from the University of Pavia, Italy, in 1969. He is a professor in the Computer Science Department at the Politecnico di Milano, Milano, Italy. He spent four years with the University of Milano before joining the Politecnico in 1991. From 1978 to 1987, he was an associate professor in the Department of Mathematics, University of Pavia. His current research interests include modeling and other topics related to performance evaluation of computer systems, Web servers, intranets, Internet, and high-performance computing. He is a member of the IEEE Computer Society.