Multivariate Rational Approximants for Multiclass Closed Queuing Networks

Annie Cuyt and R.B. Lenin

Abstract—Closed Markovian networks of queues with multiclass customers and having a product form equilibrium state probability distribution are useful in the performance evaluation and design of computer and telecommunication systems. Therefore, the efficient computation of the normalizing function, the key element of the solution in product form, has attracted considerable effort. We consider a network that consists of one infinite-server (IS) station and one processor-sharing (PS) or FCFS single-server station. We use multivariate Newton-Padé approximants computed from data for small numbers of customers in each class, to estimate the normalizing function for a larger population in the network. The effectiveness and tremendous gain in computing time of this procedure are illustrated through various numerical experiments.

Index Terms—Normalizing function, stationary probability distribution, multivariate, Newton-Padé approximation, partial Padé approximation, convolution algorithm.

1 Introduction

In the past two decades, closed Markovian queuing networks have emerged as one of the most important tools for modeling computer systems, computer communication systems, online computer networks, and other real-time computer-based systems, and for analyzing their performance measures. This was because of the discovery of an important class of such networks known as "product-form networks" [16]. In these networks, the equilibrium state probability distribution of the network is the product of the distributions of each queue analyzed in isolation from the network subject to a "normalizing function" or "partition function" (function of the population size). Interestingly, most of the performance measures of a closed queuing network can be obtained from the knowledge of the normalizing function associated with the network [19].

The algorithms to compute the normalizing function of a closed queuing network with a single class of customers are rather simple. Moreover, the case of single-class queuing networks is of little interest because of its simplicity and limited usefulness. Multiclass closed queuing networks, which have a product-form equilibrium state probability distribution [3], are of much more interest and are widely used [22], [26], [21]. The evaluation of the normalizing function for such large networks is a well-known and difficult problem. The complexity of the algorithms for the calculation of the normalizing function in queuing networks with different classes of customers increases rapidly. The principal computational difficulty associated with these networks is that a simple closed-form expression for the normalizing function of the distribution is not known. In general, a direct determination of the normalizing function by a straightforward summation is computationally intractable.

In the past, several algorithms for the normalizing function have been constructed, usually based on the classical "convolution algorithm." But, all these algorithms share the common problem of long computing times. Even when the network contains only a moderate number of nodes, computing time already gets out of hand. In fact, the time complexity grows exponentially with the number of classes and polynomially with the number of customers in the network. See Bruel and Balbo [8], Conway and Georganas [11], and Lavenberg [22] for an overview. A standard way to address the computing time problem is to carry out an approximate computation based on simulation.

Recently, another approach has become available, based on the use of rational approximation techniques to compute the normalizing function [18]. The motivation behind this approach is that it is often possible to study interesting properties such as monotonicity, convexity, boundedness, and asymptotic behavior of the normalizing function (see, for example, [17], [20], [23]). For rational approximation techniques, such properties are a vital piece of information and lead to qualitative results.

It turns out that rational approximation methods are very promising for closed queuing networks with both single-class and multiclass customers. In [18], Gong and Yang have used rational approximants of some one-dimensional projections of the normalizing function in a multiclass network. This, of course, imposes restrictions on the parameters under investigation. Our objective is to generalize their result to the evaluation of the true multivariate (nonprojected) normalizing function. To this end, we will use the multivariate rational approximation technique developed in [14] and [1].

In this paper, we further analyze the closed queuing network with multiclass customers discussed in [18]. The normalizing function of this network is a multidimensional nonnegative integer valued function. When the number of

• *The authors are with the Department of Mathematics and Computer Science, University of Antwerp, Universiteitsplein 1, B-2610 Antwerp, Belgium. E-mail: {cuyt, lenin}@uia.ua.ac.be.*

Manuscript received 31 Jan. 2001; accepted 28 June 2001.
For information on obtaining reprints of this article, please send e-mail to: tc@computer.org, and reference IEEECS Log Number 113558.

customers in each class is small, its values can be computed within a reasonable time, using a convolution algorithm. These values will then be used to extrapolate the function for a larger population by means of multivariate Newton-Padé approximation techniques. It is well-known from the literature that many successful extrapolation techniques are based on the use of rational functions. Algorithms to reliably compute these rational approximants (Padé, interpolation, Newton-Padé) can be found in [10] and [4].

2 Multiclass Closed Queueing Network

We consider a multiclass product-form closed queuing network having two nodes: one infinite-server (IS) node and one processor-sharing (PS) or FCFS single-server node and a total of K_c customers in class c, for $c = 1, 2, \ldots, \tau$. We assume that there are no class transitions. This type of network essentially depicts a central server system with two nodes: One is an infinite-server node, called a "think node," and the other is a process-sharing node (single-server), called a "CPU node" (see [23] for details). The service times at node i for class c customers are exponentially distributed with rates

$$\mu_{ni}(c) = \begin{cases} n\mu_1(c), & i = 1, \\ \mu_2(c), & i = 2, \end{cases}$$

for $c = 1, 2, \ldots, \tau$, when there are n customers at node i. A class c customer upon receiving service at node i will proceed to node j with probability $p_{ij}(c)$ or leave the network at node i with probability $p_{i0}(c) = 1 - \sum_{j=1}^{2} p_{ij}(c)$. Let

$$\bar{n} = (n_{11}, n_{21}, \ldots, n_{\tau 1}; n_{12}, n_{22}, \ldots, n_{\tau 2})$$

be the state vector of the entire network, where n_{ci} denotes the number of class c customers at node i. The state space of the network is given by

$$\mathcal{S} = \{\bar{n} = (n_{11}, n_{21}, \ldots, n_{\tau 1}; n_{12}, n_{22}, \ldots, n_{\tau 2}) \mid n_{c1} + n_{c2} = K_c,\ c = 1, 2, \ldots, \tau\}.$$

Let η_{ci} be the relative traffic intensities of class c customers at node i obtained by solving

$$\eta_{ci}\mu_i(c) = \eta_{c1}\mu_1(c)p_{1i}(c) + \eta_{c2}\mu_2(c)p_{2i}(c), \quad i = 1, 2;\ c = 1, 2, \ldots, \tau,$$

which is a system of homogeneous equations and, hence, the quantities η_{ci} can be scaled.

Then, the queue length distribution [3] equals

$$P(\bar{n}) = \frac{1}{G(K_1, \ldots, K_\tau)} \prod_{i=1}^{2} f_i(n_{1i}, n_{2i}, \ldots, n_{\tau i}), \quad \bar{n} \in \mathcal{S}, \quad (1)$$

where

$$f_i(n_{1i}, n_{2i}, \ldots, n_{\tau i}) = \begin{cases} \begin{pmatrix} n_1 \\ n_{11}\ n_{21} \ldots n_{\tau 1} \end{pmatrix} \dfrac{\prod_{c=1}^{\tau} \eta_{c1}^{n_{c1}}}{n_1!}, & i = 1, \\ \begin{pmatrix} n_2 \\ n_{12}\ n_{22} \ldots n_{\tau 2} \end{pmatrix} \prod_{c=1}^{\tau} \eta_{c2}^{n_{c2}}, & i = 2, \end{cases} \quad (2)$$

and the normalizing function

$$G(K_1, \ldots, K_\tau) = \sum_{m_\tau=0}^{K_\tau} \cdots \sum_{m_1=0}^{K_1} \left\{ \left(\prod_{c=1}^{\tau} \frac{\eta_{c1}^{(K_c - m_c)}}{(K_c - m_c)!} \right) \left(\sum_{c=1}^{\tau} m_c \right)! \left(\prod_{c=1}^{\tau} \frac{\eta_{c2}^{m_c}}{m_c!} \right) \right\}. \quad (3)$$

Using Euler's integral

$$n! = \int_0^\infty \exp(-t) t^n dt,$$

(3) can be written as

$$G(K_1, K_2, \ldots, K_\tau) = \int_0^\infty \exp(-t) \prod_{c=1}^{\tau} \frac{(\eta_{c1} + \eta_{c2} t)^{K_c}}{K_c!} dt. \quad (4)$$

Let us now take a closer look at the function G when having only two classes of customers ($\tau = 2$). We denote K_1 and K_2, respectively, by K and L. From (4), we have,

$$G(K, L) = \int_0^\infty \exp(-t) \frac{(\eta_{11} + \eta_{12} t)^K}{K!} \frac{(\eta_{21} + \eta_{22} t)^L}{L!} dt. \quad (5)$$

We recall that the quantities η_{11}, η_{12} and η_{21}, η_{22} are the relative traffic intensities of class 1 and class 2 customers, respectively, in node 1 and node 2 and are inversely proportional to the respective service rates of the servers for class 1 and class 2 customers at node 1 and node 2.

Using the fact that for any constant $a \in \mathbb{R}$ and integer K,

$$\frac{a^K}{K!} < e^a,$$

it can be shown, under the condition that $\eta_{12} + \eta_{22} < 1$, that $G(K, L)$ is bounded:

$$\forall K, L \geq 0 : G(K, L) < \frac{e^{(\eta_{11} + \eta_{21})}}{1 - \eta_{12} - \eta_{22}}. \quad (6)$$

Hence, we classify our analysis and discussion into the cases:

1. G is bounded; $\eta_{12} + \eta_{22} < 1$ and
   a. $\eta_{11} < 1, \eta_{21} < 1$.
   b. $\eta_{11} < 1, \eta_{21} \geq 1$.
   c. $\eta_{11} \geq 1, \eta_{21} < 1$.
   d. $\eta_{11} \geq 1, \eta_{21} \geq 1$.
2. G is unbounded.

The condition that $\eta_{12} + \eta_{22} < 1$ implies that we are allowing less traffic ("Light traffic") in the central processing unit (CPU) of the network. This can be achieved by increasing the capacity of the CPU.

When $\eta_{12} + \eta_{22} < 1$, the function $G(K, L)$ attains its maximum in the neighborhood of the point (η_{11}, η_{21}),

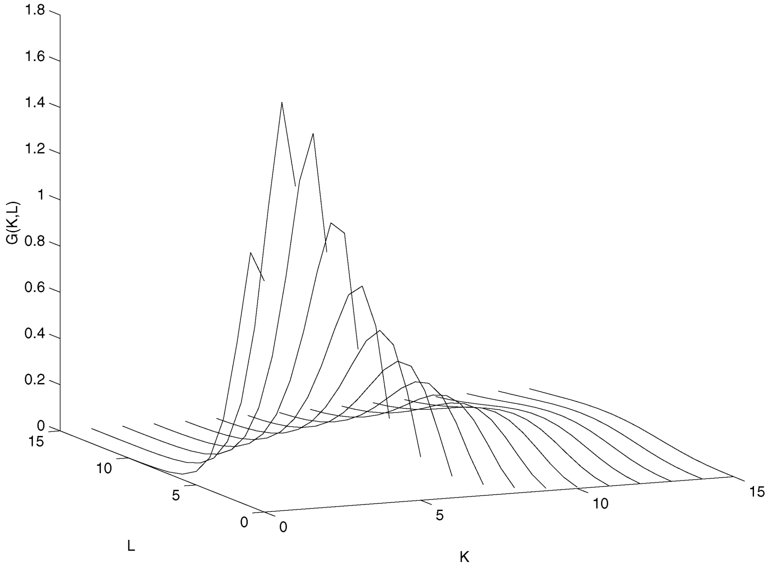


Fig. 1. Normalizing function for a two-class closed queuing network for $\eta_{11} = 0.9$, $\eta_{12} = 0.5$, $\eta_{21} = 0.7$, and $\eta_{22} = 0.4$ (using convolution algorithm).

depending on the values of η_{12} and η_{22}. Also, $G(K, L)$ decreases and tends to zero as K and L increase. In the following sections, we study the behavior of G for different ranges of η_{11} and η_{21} using the convolution algorithm [11].

2.1 Case 1

Depending on the values for η_{11} and η_{21}, the function $G(K, L)$ relocates its maximum. Cases 1a, 1b, 1c, and 1d, respectively, correspond to Figs. 1, 2, 3 and 4.

Case 1a arises when the service rates in the think node for class 1 and class 2 customers are large enough to make the respective traffic intensities, η_{11} and η_{21}, less than one ("Light traffic" at the think node).

Case 1b occurs when the service rates for class 2 customers are large enough to make the corresponding traffic intensity $0 < \eta_{21} < 1$, whereas the service rates for class 1 customers are small enough to make the corresponding traffic intensity $\eta_{11} \geq 1$ ("Heavy traffic" with respect to class 1 customers and "Light traffic" with respect to class 2 customers at the think node).

Case 1c corresponds to the situation where the service rates of the servers for class 1 customers are large enough to make the corresponding traffic intensity $0 < \eta_{11} < 1$, whereas, for class 2 customers, the service rates are small enough to make the corresponding traffic intensity $\eta_{21} \geq 1$ ("Light traffic" with respect to class 1 customers and "Heavy traffic" with respect to class 2 customers at the think node).

Finally, Case 1d corresponds to the situation where the service rates of the servers for class 1 and class 2 customers are small enough to make the respective traffic intensities η_{11} and η_{21} greater than or equal to one ("Heavy traffic" at the think node).

From the four given graphs of G, we observe that the bell-shaped structure in the surface moves further away from the origin as η_{11} and η_{21} increase.

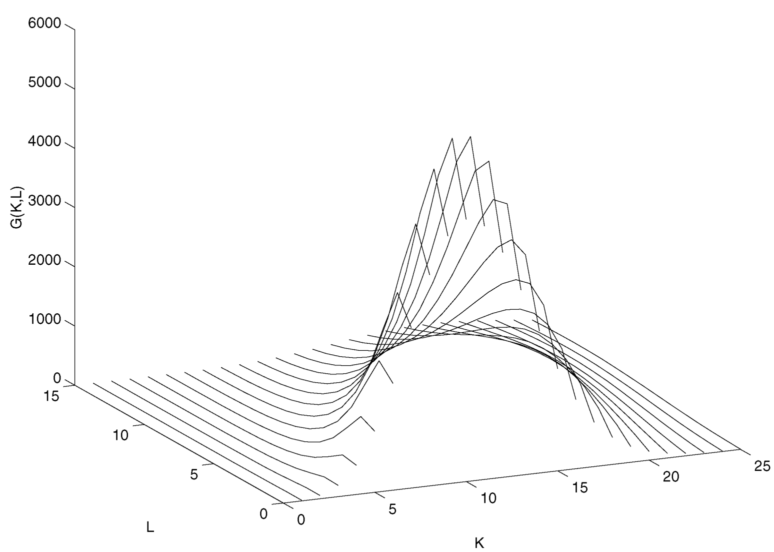


Fig. 2. Normalizing function for a two-class closed queuing network for $\eta_{11} = 10$, $\eta_{12} = 0.4$, $\eta_{21} = 0.5$, and $\eta_{22} = 0.5$ (using convolution algorithm).

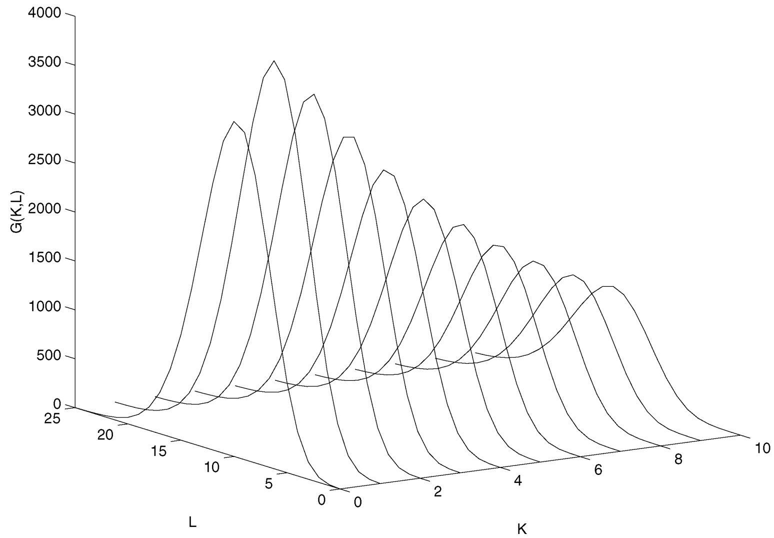


Fig. 3. Normalizing function for a two-class closed queuing network for $\eta_{11} = 0.3$, $\eta_{12} = 0.7$, $\eta_{21} = 10$, and $\eta_{22} = 0.2$ (using convolution algorithm).

2.2 Case 2

In this case, we relax the condition $\eta_{12} + \eta_{22} < 1$ and then it can be observed that the function G increases with K and L. The function $\log(G)$ behaves approximately linearly when either K or L is fixed (see Fig. 5).

Based on all the above observations, different multivariate Newton-Padé approximants must be chosen to model the behavior of the function $G(K, L)$, depending on the value of the parameters in play. This is explained in the next sections.

3 General Order Partial Multivariate Newton-Padé Approximation

Consider two sequences of real values $\{x_i\}_{i\in\mathbb{N}}$ and $\{y_j\}_{j\in\mathbb{N}}$ and let coalescent values (if any) get consecutive indices. For a bivariate function $G(x, y)$ known in the tuples (x_i, y_j), divided differences (with possible coalescence of coordinates) can be defined [14] by

$$
\begin{aligned}
&G[x_0][y_0] \\
&= G(x_0, y_0) \\
&G[x_0][y_0, \ldots, y_k] \\
&= \frac{G[x_0][y_1, \ldots, y_k] - G[x_0][y_0, \ldots, y_{k-1}]}{y_k - y_0} \\
&G[x_0, \ldots, x_k][y_0] \\
&= \frac{G[x_1, \ldots, x_k][y_0] - G[x_0, \ldots, x_{k-1}][y_0]}{x_k - x_0} \\
&G[x_0, \ldots, x_k][y_0, \ldots, y_l] \\
&= \frac{G[x_0, \ldots, x_k][y_1, \ldots, y_l] - G[x_0, \ldots, x_k][y_0, \ldots, y_{l-1}]}{y_l - y_0} \\
&= \frac{G[x_1, \ldots, x_k][y_0, \ldots, y_l] - G[x_0, \ldots, x_{k-1}][y_0, \ldots, y_l]}{x_k - x_0}.
\end{aligned}
$$

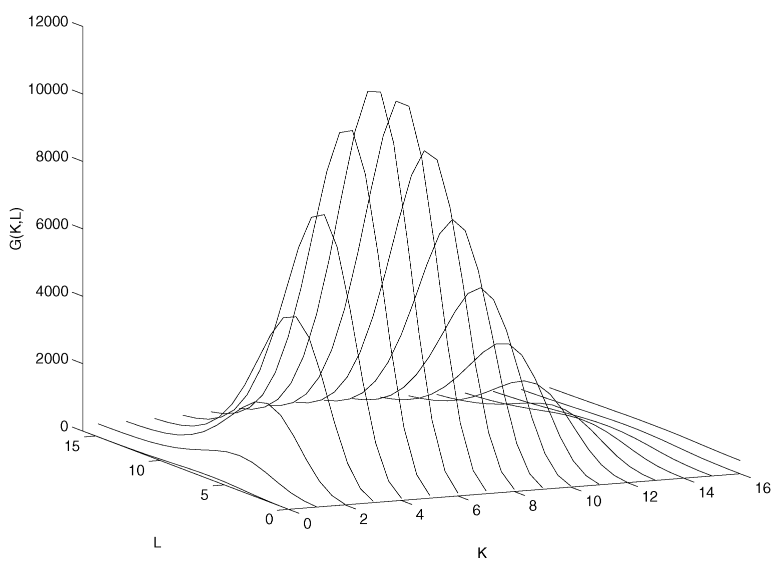


Fig. 4. Normalizing function for a two-class closed queuing network for $\eta_{11} = 6$, $\eta_{12} = 0.3$, $\eta_{21} = 6$, and $\eta_{22} = 0.4$ (using convolution algorithm).

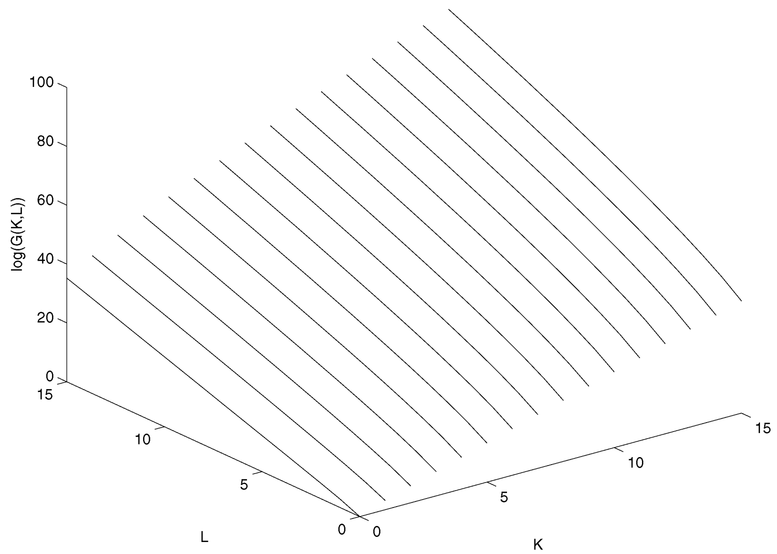


Fig. 5. Normalizing function for a two-class closed queuing network for $\eta_{11} = 10$, $\eta_{12} = 12$, $\eta_{21} = 7$, and $\eta_{22} = 10$ (using convolution algorithm).

Let us also use the following basis functions for the real-valued polynomials in two variables:

$$B_{ij}(x,y) = \prod_{k=0}^{i-1}(x - x_k)\prod_{l=0}^{j-1}(y - y_l), \tag{7}$$

which are bivariate polynomials of degree $i + j$.

The notion of partial Padé approximant was introduced by Brezinski [7] for univariate functions $G(x)$: Some of the Padé approximation conditions were dropped in exchange for some information on poles or zeros of $G(x)$. Let the polynomials $v_k(x)$ and $w_l(x)$, respectively, represent k zeros and l poles of G. The partial Padé approximation problem for G then consists of finding polynomials $p(x)$ and $q(x)$, respectively, of degree n and m and satisfying

$$(Gqw_l - pv_k)(x) = O(x^{n+m+1}). \tag{8}$$

The rational function $(pv_k)/(qw_l)$ is called the partial Padé approximant to G of order $(n+k, m+l)$. It is easy to see that, if $v_k(0) \neq 0$, the rational function p/q is the Padé approximant of order (n,m) to Gw_l/v_k [7]. The concept has been generalized to the multivariate case as follows [1]:

Let the polynomials $V_k(x,y)$ and $W_l(x,y)$, respectively, represent some knowledge about the zeros and poles of $G(x,y)$,

$$\begin{aligned} V_k(x,y) &= \sum_{(i,j)\in S} v_{ij}B_{ij}(x,y)\\ \#S &= k+1\\ W_l(x,y) &= \sum_{(i,j)\in T} w_{ij}B_{ij}(x,y)\\ \#T &= l+1, \end{aligned}$$

and consider the following approximation problem. Let the finite subset $I \subset \mathbb{N}^2$ index those data points (x_i, y_j) that will be used as interpolation points. The knowledge of $G(x,y)$ in these interpolation points (x_i, y_j) enables us to write a formal Newton series development for G,

$$\begin{aligned} G(x,y) &= \sum_{(i,j)\in\mathbb{N}^2} c_{ij}B_{ij}(x,y),\\ c_{ij} &= G[x_0,\ldots,x_i][y_0,\ldots,y_j]. \end{aligned}$$

To generalize (8) we look for polynomials P and Q, respectively, indexed by finite index sets N (from "Numerator") and D (from "Denominator"),

$$\begin{aligned} P(x,y) &= \sum_{(i,j)\in N} a_{ij}B_{ij}(x,y)\\ Q(x,y) &= \sum_{(i,j)\in D} b_{ij}B_{ij}(x,y), \end{aligned}$$

and satisfying

$$(GQW_l - PV_k)(x,y) = \sum_{(i,j)\in\mathbb{N}^2\setminus I} d_{ij}B_{ij}(x,y). \tag{9}$$

The index sets N and D also satisfy

$$\begin{aligned} &N \subset I\\ &\#(I \setminus N) = \#D - 1\\ &(i,j) \in I \Longrightarrow 0 \le k \le i, 0 \le l \le j : (k,l) \in I. \end{aligned}$$

Assuming that $V_k(x_i, y_j) \neq 0$, condition (9) implies that

$$\left(\frac{GW_l}{V_k}Q - P\right)(x,y) = \sum_{(i,j)\in\mathbb{N}^2\setminus I} e_{ij}B_{ij}(x,y).$$

Here,

$$\left(\frac{GW_l}{V_k}\right)(x,y) = \sum_{(i,j)\in\mathbb{N}^2} \tilde{c}_{ij}B_{ij}(x,y), \tag{10}$$

with the coefficients, $\tilde{c}_{ij}$, being given by

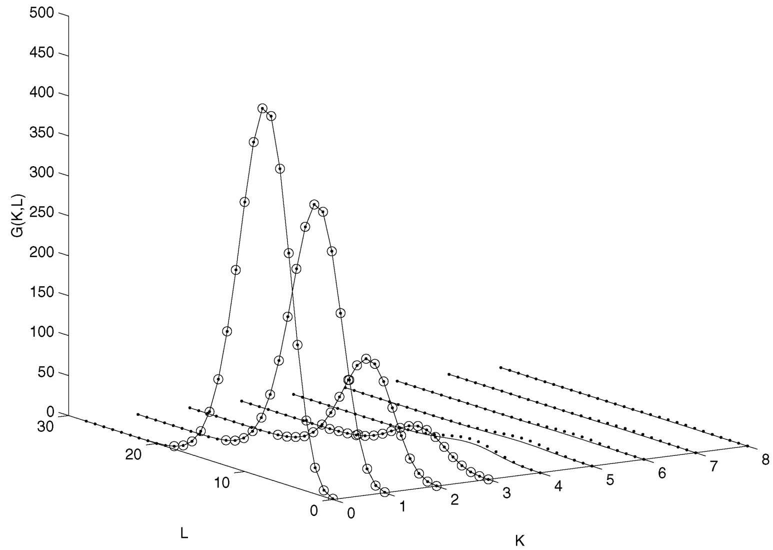


Fig. 6. Normalizing function for a two-class closed queuing network for $\eta_{11} = 0.5$, $\eta_{12} = 0.2$, $\eta_{21} = 8$, $\eta_{22} = 0.1$ (using Newton-Padé approximation).

$$\gamma_{ij} = \left[\left(\frac{GW_l}{V_k}\right)V_k\right][x_0,\ldots,x_i][y_0,\ldots,y_j] = \sum_{\mu=0}^{i}\sum_{\nu=0}^{j} \tilde{c}_{\mu\nu} V_k[x_\mu,\ldots,x_i][y_\nu,\ldots,y_j]$$

and

$$\begin{aligned}\gamma_{ij} &= (GW_l)[x_0,\ldots,x_i][y_0,\ldots,y_j]\\ &= \sum_{(t,u)\in T} W_l[x_0,\ldots,x_t][y_0,\ldots,y_u]G[x_t,\ldots,x_i][y_u,\ldots,y_j]\\ &= \sum_{(t,u)\in T} w_{tu} G[x_t,\ldots,x_i][y_u,\ldots,y_j].\end{aligned}$$

From (10), P/Q appears to be the Newton-Padé approximant to $(GW_l/V_k)(x,y)$. If the Newton-Padé approximant P/Q is denoted by $[N/D]_I^{GW_l/V_k}$, then we introduce the notation $\{N, S/D, T\}_I^G$ for the partial Newton-Padé approximant $(PV_k)/(QW_l)$ to $G(x,y)$.

Using known results for general order multivariate Newton-Padé approximants, the partial Newton-Padé approximant $\{N, S/D, T\}_I^G$ can be expressed as a ratio of determinants involving the coefficients $\tilde{c}_{ij}$ from the formal Newton series expansion (10) for $(GW_l/V_k)(x,y)$. Let us number the indices in D by $(d_0,e_0),(d_1,e_1),\ldots,(d_m,e_m)$ and the indices in $I \setminus N$ by $(h_1,k_1),\ldots,(h_m,k_m)$. If the rank of the coefficient matrix of the linear conditions arising from (9) is maximal, then $Q(x,y)$ and $P(x,y)$ are given by

$$Q(x,y) = \begin{vmatrix} B_{d_0e_0}(x,y) & \cdots & B_{d_me_m}(x,y)\\ \tilde{c}_{d_0h_1,e_0k_1} & \cdots & \tilde{c}_{d_mh_1,e_mk_1}\\ \vdots & & \vdots\\ \tilde{c}_{d_0h_m,e_0k_m} & \cdots & \tilde{c}_{d_mh_m,e_mk_m}\end{vmatrix}, \qquad (11)$$

$$P(x,y) = \begin{vmatrix} \sum_{(i,j)\in N}\tilde{c}_{d_0i,e_0j}B_{ij}(x,y) & \cdots & \sum_{(i,j)\in N}\tilde{c}_{d_mi,e_mj}B_{ij}(x,y)\\ \tilde{c}_{d_0h_1,e_0k_1} & \cdots & \tilde{c}_{d_mh_1,e_mk_1}\\ \vdots & & \vdots\\ \tilde{c}_{d_0h_m,e_0k_m} & \cdots & \tilde{c}_{d_mh_m,e_mk_m}\end{vmatrix}, \qquad (12)$$

and can be computed using the E-algorithm [12].

4 Results from the Rational Extrapolation Technique

The polynomials $V_k(x,y)$ and $W_l(x,y)$ introduced in the previous section will be chosen appropriately, depending on the cases we distinguished in the behavior of $G(K,L)$.

4.1 Light Traffic in the Think Node

Let us first consider the situation where there is light traffic at the think node for at least one class of customers. This corresponds to Cases 1a, 1b, and 1c. In these cases, we can choose $V_k = 1 = W_l$ and the simplest Newton-Padé approximant to model G correctly is simply $[N/D]_I^G$ with

$$\begin{aligned}N &= \{(0,0)\}\\ D &= I.\end{aligned}$$

Since Newton-Padé approximants satisfy the reciprocal covariance property [1], the approximant $[N/D]_I^G$ equals the inverse of the polynomial interpolant $[D/N]_I^{1/G}$.

In Fig. 6, one can find the result for the parameter values $\eta_{11} = 0.5$, $\eta_{12} = 0.2$, $\eta_{21} = 8$, and $\eta_{22} = 0.1$: The dotted line represents the Newton-Padé approximant, the full line still represents the function G computed via the convolution algorithm, and the circled entries have been used as interpolation points and, hence, depict I. Here, $I = [0,\ldots,4]\times[0,\ldots,19]$. Because of the choice made for $[N/D]_I^G$, the circled entries also depict D.

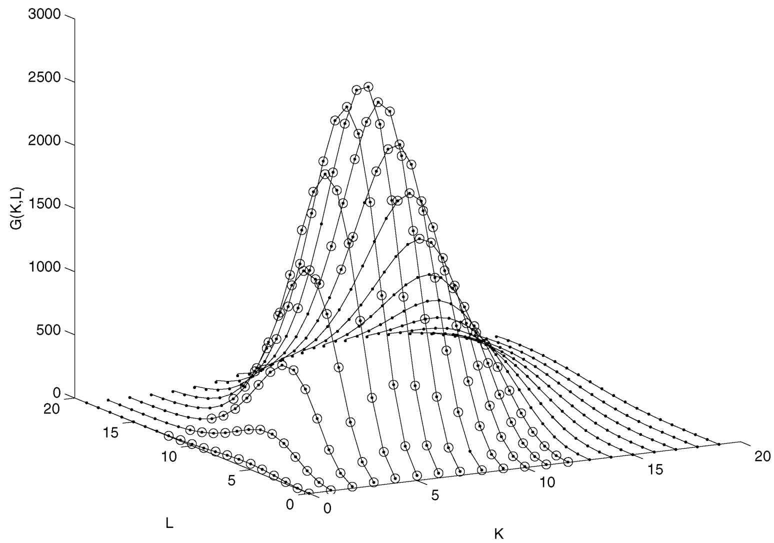


Fig. 7. Normalizing function for a two-class closed queuing network for $\eta_{11} = 5$, $\eta_{12} = 0.5$, $\eta_{21} = 5$, and $\eta_{22} = 0.4$ (using Netwon-Padé approximation).

4.2 Heavy Traffic in the Think Node

This corresponds to Case 1d, which needs different partial Newton-Padé models depending on the values of η_{11} and η_{21}. In case η_{11} and η_{21} are small, the bell-shaped structure is still sufficiently close to the origin to make use of full Newton-Padé approximation. Computing the function values at the interpolation points is the most expensive part of the algorithm. If a sufficiently small number of interpolation points can pick up most of the surface structure, then a general order Newton-Padé approximant with $V_k = 1 = W_l$ does the job.

For instance, for $\eta_{11} = 5$, $\eta_{12} = 0.5$, $\eta_{21} = 5$, and $\eta_{22} = 0.4$, Fig. 7 shows both the function G computed using the convolution algorithm (full line) and the Newton-Padé approximant $[N/D]_I^G$ (dotted line) with

$$
\begin{aligned}
N &= \{(i,j) \mid 0 \le i+j \le 10\}\\
D &= \{(i,j) \mid 0 \le i+j \le 12\}\\
I &= \{(i,j) \mid 0 \le i \le 12, 0 \le j \le 12\}\setminus\\
&\quad \{(9,11),(9,12),(10,10),(10,11),(10,12),(11,9),\ldots,\\
&\qquad (11,12),(12,9),\ldots,(12,12)\}.
\end{aligned}
$$

It is not at all difficult to get a good estimate of how N, D, and I should be chosen, based on the information that a univariate bell-shaped function (take, for instance, the normal distribution) is nicely approximated by a Newton-Padé approximant of degree n in the numerator and $n+2$ in the denominator. The set I is constructed such that it is as symmetric as possible and such that it satisfies the conditions (10). In Table 1, some values of G and computing times can be compared. How one can obtain equally good numerical results with even fewer data points is explained further on.

When η_{11} and η_{21} are large, then the previous approach also works, provided one is willing to compute enough interpolation points, in other words, function values of $G(K,L)$ to pick up the structure of the surface. Of course, one would prefer to only compute values $G(K,L)$ for small K and L and this would not be sufficient for the full Newton-Padé approximation technique since the bell-shape is further away from the origin. Consequently, the advantages offered by the partial Newton-Padé approximation technique come into play.

If we choose, for instance,

$$
\begin{aligned}
V_k(K,L) &= \alpha^4 \eta_{11}^2 \eta_{21}^2\\
W_l(K,L) &= \left[(K-\eta_{11})^2 + (\alpha\eta_{11})^2\right]\left[(L-\eta_{21})^2 + (\alpha\eta_{21})^2\right]\\
&\qquad (1+\gamma(K-\eta_{11}) + 2\gamma(L-\eta_{21}))
\end{aligned}
$$

with

$$
\begin{aligned}
\alpha &= 3/4\\
\gamma &= 1/1000,
\end{aligned}
$$

then a full Newton-Padé approximant can be computed for the function GW_l/V_k, which is equivalent to a partial Newton-Padé approximant for G, and it will suffice to sample GW_l/V_k only for small values of K and L, as can be seen in Fig. 8 and Table 2. For larger values of K and L, the approximant is accurate enough and the convolution algorithm does not have to be used.

Let us briefly discuss the choice of the factor $W_l(K,L)$. In view of the bell-shaped curve for G, we introduce a quadratic factor with complex zeros in the denominator

TABLE 1
Comparison of Values of $G(K,L)$ and $[N/D]_I^G$ for Larger K and L

K	L	Convolution		Newton-Padé	
		CPU time	$G(K,L)$	$[N/D]_I^G(K,L)$	CPU time
17	18	11.03	1.8524e+2	1.8574e+2	3.38
18	18	12.73	1.7886e+2	1.7804e+2	3.38
19	17	12.56	1.9763e+2	1.9617e+2	3.38
19	18	14.52	1.6848e+2	1.6876e+2	3.38
19	19	15.39	1.3949e+2	1.3881e+2	3.38

Total CPU time in seconds including computation of function values at interpolation points.

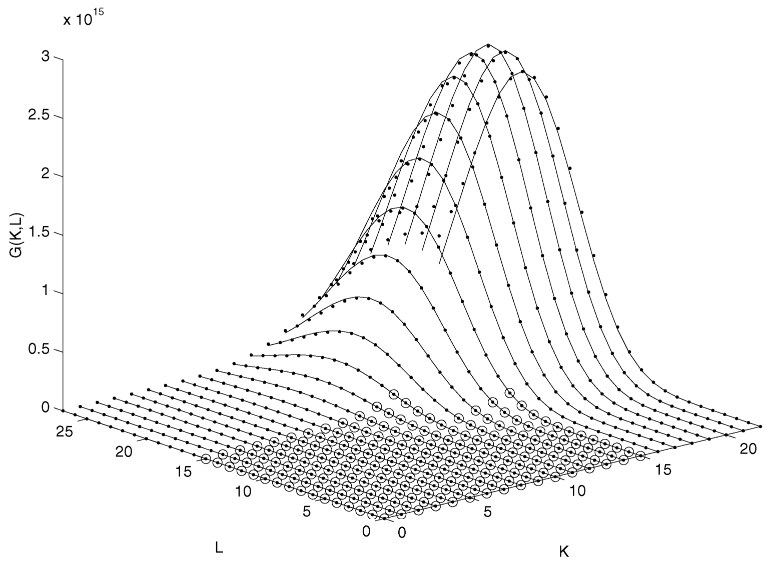


Fig. 8. Normalizing function for a two-class closed queuing network for $\eta_{11} = 20$, $\eta_{12} = 0.2$, $\eta_{21} = 20$, and $\eta_{22} = 0.1$ (using partial Newton-Padé approximation).

of the Newton-Padé approximant. The parameter α somehow determines the "width" of the bell-shape. When increasing α, the bell-shape becomes "fatter" and the values of G for larger K and L are overestimated. When decreasing α, the bell-shape becomes "thinner" and the values of G for larger K and L are underestimated. The parameter γ was introduced because, without the extra factor $(1 + \gamma(K - \eta_{11}) + 2\gamma(L - \eta_{21}))$, the values of G at the top of the bell-shape were overestimated. So, we reduced the values of the approximant by introducing a factor just slightly larger than 1 in the denominator.

Fig. 8 displays both the partial Newton-Padé approximant $\{N, S/D, T\}_I^G = [N/D]_I^{GW_l/V_k}$ (dotted line) and the function $G(K, L)$ (full line) computed using the convolution algorithm for $\eta_{11} = 20$, $\eta_{12} = 0.2$, $\eta_{21} = 20$, and $\eta_{22} = 0.1$. The index sets are given by

$$N = \{(i,j) \mid 0 \le i + j \le 12\}$$
$$D = \{(i,j) \mid 0 \le i + j \le 10\}$$
$$I = \{(i,j) \mid 0 \le i, j \le 15\} \setminus \{(i,j) \mid 12 \le i, j \le 15\}.$$

4.3 Heavy Traffic in the Network

In Case 2, where the function $G(K, L)$ is unbounded, it is not a matter of getting an approximation to G that is correct within a few significant digits, but it is important to get the magnitude of G estimated correctly. Therefore, one is usually interested in $\log(G(K, L))$ rather than $G(K, L)$ itself. Since one can show that $\log(G(K, L))$ behaves linearly for

TABLE 2
Comparison of Values of $G(K, L)$ and $\{N, S/D, T\}_I^G(K, L)$ for Larger K and L

K	L	Convolution		Partial Newton-Padé	
		CPU time	$G(K,L)$	$\{N, S/D, T\}_I^G(K,L)$	CPU time
20	20	18.58	2.6407(15)	2.6644(15)	5.78
20	24	25.53	1.7046(15)	1.7007(15)	5.78
21	21	21.9	2.4465(10)	2.4433(10)	5.78
21	22	22.07	2.2401(15)	2.2465(15)	5.78
22	22	24.92	2.0661(15)	2.0526(15)	5.78

Total CPU time in seconds including computation of function values at interpolation points.

TABLE 3
Comparison of Values of $\log(G(K, L))$ and $[N/D]_I^{\log(G)}$ for Larger K and L

K	L	Convolution		Newton-Padé	
		CPU time	$\log(G(K,L))$	$[N/D]_I^{\log(G)}(K,L)$	CPU time
33	35	126.55	2.7014(2)	2.7015(2)	0.31
34	30	99.47	2.5351(2)	2.5352(2)	0.31
34	35	132.76	2.7406(2)	2.7407(2)	0.31
35	30	105.35	2.5735(2)	2.5736(2)	0.31
35	35	141.2	2.7798(2)	2.7798(2)	0.31

Total CPU time in seconds including computation of function values at interpolation points.

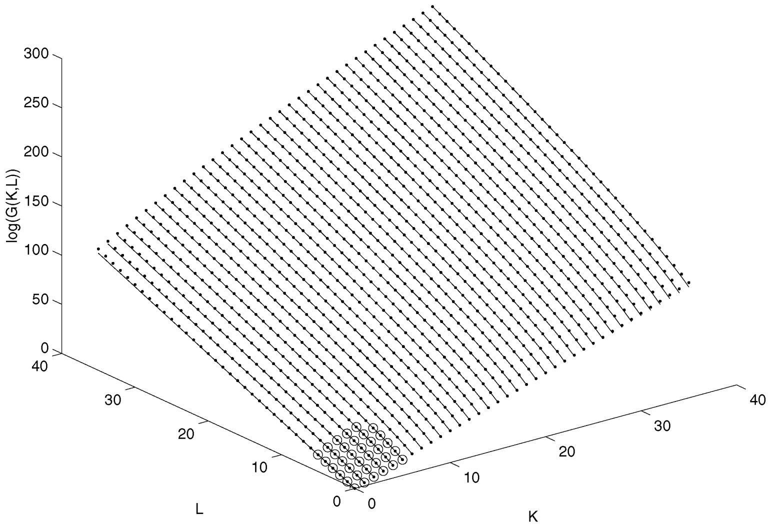


Fig. 9. Normalizing function for a two-class closed queuing network for $\eta_{11} = 2$, $\eta_{12} = 25$, $\eta_{21} = 4$, and $\eta_{22} = 30$ (using Newton-Padé approximation).

large values of K and L, it is easy to approximate it by a rational function which is of total degree $n + 1$ in the numerator and n in the denominator.

In Fig. 9, both the function $\log(G(K, L))$ and its Newton-Padé approximant $[N/D]^G_I$ are shown for $\eta_{11} = 2$, $\eta_{12} = 25$, $\eta_{21} = 4$, and $\eta_{22} = 30$. In Table 3, some values and computing times can be compared. The index sets are given by

$$N = \{(i, j) \mid 0 \le i + j \le 5\}$$
$$D = \{(i, j) \mid 0 \le i + j \le 4\}$$
$$I = \{(i, j) \mid 0 \le i, j \le 5\} \setminus \{(5, 5)\}.$$

5 Conclusion and Future Work

We have thoroughly examined the network under consideration in the event of light and heavy traffic in the think node and in the event of heavy traffic in the network. In all cases, the approximation method can provide a high quality result provided that a sensible choice is made for the (partial) Newton-Padé approximant.

When an improper choice is made for the (partial) Newton-Padé approximant, the occurrence of unwanted poles in the multivariate approximating rational function can disturb the quality of the output. Therefore, future work will include the study of pole-free regions.

At the same time, we want to study the optimal placement of the sample points (circled entries), as in [6], [5], and automate this procedure in order to achieve high quality results with low degree approximants. Results on this problem are in preparation.

Also, in [20], the asymptotic behavior of the normalizing function G is studied for closed BCMP networks with single-server nodes and no infinite-server nodes. Instead of projecting the multidimensional function G onto a one-dimensional subspace, as in [20], we propose approximating G using the same technique as introduced here.

In the meantime, the rational approximation procedure has also proven effective [13] in the estimation of the cell loss probablity in finite M/G/1-type queues [24]. In that case, the gain in computing time was tremendous while the numerical output remained highly reliable.

References

[1] J. Abouir and A. Cuyt, "Multivariate Partial Newton-Padé and Newton-Padé Type Approximants," *J. Approximation Theory,* vol. 72, pp. 301-316, 1993.

[2] J.G.A. Baker and P. Graves-Morris, *Padé Approximants,* second ed. Cambridge Univ. Press, 1996.

[3] F. Baskett, K.M. Chandy, R.R. Muntz, and F. Palacios, "Open, Closed and Mixed Networks of Queues with Different Classes of Customers," *J. ACM,* vol. 22, pp. 248-260, 1975.

[4] S. Becuwe and A. Cuyt, "Multivariate Padé Approximants: Homogeneous or Not, That's the Question," *Int'l Assoc. Math. and Computers in Simulation (IMACS) 2000 Proc.,* M. Deville and R. Owens, eds., 2000.

[5] J.-P. Berrut and H.D. Mittelmann, "Rational Interpolation through the Optimal Attachment of Poles to the Interpolating Polynomial," *Numerical Algorithms,* vol. 23, pp. 315-328, 2000.

[6] J.-P. Berrut and H.D. Mittelmann, "The Linear Rational Collocation Method with Iteratively Optimized Poles for Two-Point Boundary Value Problems," submitted for publication, 2000.

[7] C. Brezinski, "Partial Padé Approximation," *J. Computers and Applied Math.,* vol. 54, pp. 210-233, 1988.

[8] S.C. Bruel and G. Balbo, *Computational Algorithms for Closed Queueing Networks.* Elsevier, 1980.

[9] J.P. Buzen, "Computational Algorithms for Closed Queueing Networks with Exponential Servers," *Comm. ACM,* vol. 16, pp. 527-531, 1973.

[10] S. Cabay and R. Meleshko, "A Weakly Stable Algorithm for Padé Approximants and the Inversion of Hankel Matrices," *SIAM J. Matrix Analysis and Applications,* vol. 14, pp. 735-765, 1993.

[11] A.E. Conway and N.D. Georganas, *Queueing Networks—Exact Computational Algorithms.* MIT Press, 1989.

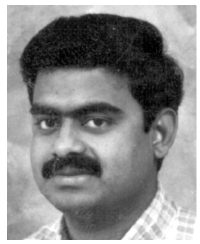
[12] A. Cuyt, "A Recursive Computation Scheme for Multivariate Rational Interpolants," *SIAM J. Numerical Analysis,* vol. 24, pp. 228-239, 1987.

[13] A. Cuyt, R.B. Lenin, G. Willems, C. Blondia, and P. Rousseeuw, "Rational Approximation Techniques to Compute Cell Loss Probabilities in Multiplexer Models," in preparation.

[14] A. Cuyt and B. Verdonk, "General Order Newton-Padé Approximants for Multivariate Functions," *Numerische Mathematika,* vol. 43, pp. 293-307, 1984.

[15] A. Cuyt and B. Verdonk, "Different Techniques for the Construction of Multivariate Rational Interpolants," *MIA, Nonlinear Numerical Methods and Rational Approximation,* A. Cuyt, ed., Holland: Reidel, 1988.
[16] N.M. van Dijk, *Queueing Networks and Product Forms: A Systems Approach.* Wiley, 1993.
[17] P. Glasserman and D. Yao, *Monotone Structure in Discrete-Event Systems.* Wiley, 1994.
[18] W.B. Gong and H. Yang, "Rational Approximants for Some Performance Analysis Problems," *IEEE Trans. Computers,* vol. 44, pp. 1394-1404, 1995.
[19] P.G. Harrison and N.M. Patel, *Performance Modelling of Communication Networks and Computer Architectures.* Addison-Wesley, 1993.
[20] C. Knessl and C. Tier, "Asymptotic Expansions for Large Closed Queuing Networks with Multiple Job Classes," *IEEE Trans. Computers,* vol. 41, pp. 480-488, 1992.
[21] S.S. Lam and J.W. Wong, "Queueing Network Models of Packet Switching Networks. Part-2: Networks with Population Size Constraints," *Performance Evaluation,* vol. 3, pp. 161-180, 1982.
[22] *Computer Performance Modeling Handbook,* S.S. Lavenberg, ed. Academic Press, 1983.
[23] J. McKenna, D. Mitra, and K.J. Ramakrishnan, "A Class of Closed Markovian Queuing Networks: Integral Representations, Asymptotic Expansions, and Generalizations," *Bell Systems Technical J.,* vol. 60, pp. 599-641, 1981.
[24] M.F. Neuts, *Structured Stochastic Matrices of M/G/1 Type and Their Applications.* New York: Marcel Dekker, 1989.
[25] 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.
[26] C.H. Sauer and K.M. Chandy, *Computer Systems Performance Modeling.* Englewood Cliffs, N.J.: Prentice Hall, 1981.

Annie Cuyt received the Doctor Scientiae degree in 1982 from the University of Antwerp (UIA), Belgium, where she now teaches several computing courses. She is also the research director of the FWO-Vlaanderen, the Flemish Science Foundation. She is the author of more than 90 publications in international journals and conference proceedings and the author or editor of several books. Her current interests are in reliable computing and rational approximation theory. She is an editorial board member of the journal *Reliable Computing*. Since 1997, she has also served as a member of the scientific committee of the Flemish Science Foundation.



R.B. Lenin received the BSc degree in mathematics from madras University, Chennai, India, in 1992. He received the MSc and PhD degrees in mathematics from the Indian Institute of Technology Madras, Chennai, India, in 1994 and 1998, respectively. He worked as a postdoctoral fellow at the University of Twente, The Netherlands, during 1998-1999 on the project "Stochastic Networks" sponsored by NWO, The Netherlands. Since 1999, he has worked as a postdoctoral fellow at the University of Antwerp, Belgium, on the project "Computational Methods for Performance Evaluation and Simulation of Complex Technical Systems," funded by FWO (Flemish Division), Belgium. His recent research interests include extrapolation methods and the computation of rational interpolants with prescribed poles with applications to computer systems and communication networks.