Multi-Server Markov Queueing Models:

Computational Algorithms

and ICT Applications

Tien Van Do PhD

Dissertation for the Doctor degree of
the Hungarian Academy of Sciences

Budapest University of Technology and Economics

2009.

Dedicated to Do Hoang Anh and Do Hoang Anh Thu

Több szerveres Markov kiszolgálási modellek:

számítási algoritmusok

és infokommunikációs alkalmazások

Do Van Tien PhD

MTA doktori disszertáció

Budapesti Műszaki és Gazdaságtudományi Egyetem

2009.

Összefoglalás

A matematikailag megalapozott teljesítőképességi elemzések jelentős szerepet gyakorolnak és fognak gyakorolni az infokommunikációs rendszerek specifikálásában, tervezésében és üzemeltetésében. Kutatásaimat a sorbanálláselmélet és alkalmazásai témakörében folytatom, jelen értekezés az eddig elért, legfontosabb tudományos eredményeimet ismerteti:

1. Több szerveres Markovi sorbanállási rendszer általánosítása és több problémára való alkalmazása

Kidolgoztunk egy módszert az ún. HetSigma sor (MM$\sum_{k=1}^{K} CPP_k/GE/c/L$ G) állandósult állapot-valószínűségeinek meghatározására. Bebizonyítottam a HetSigma sor stacionárius állapotvalószínűségeinek meghatározásában fontos szerepet játszó tulajdonságokat. Megadtam a szerzőtársam javasolt transzformációk következményeként a karakterisztikus mátrix polinom együtthatóira vonatkozó zárt képletet. Felismertem, hogy a HetSigma sor *alkalmas* néhány kommunikációs hálózatokban és informatikai rendszerekben levő probléma (optikai börszt kapcsolt multiplexer, MPLS forgalomirányítás, HSDPA, Apache Web szerver) modellezésére és elemzésére.

2. Az újra-próbálkozási sorokkal modellezhető rendszerek jellemzőinek hatékony meghatározása

Az újra-próbálkozási sorok számos infokommunikációs rendszer (DHCP szerver, vezetéknélküli hálózat) elemzésére használhatóak, ugyanakkor legtöbb esetben matematikailag kezelhetetlenek. A témakörben közelítő modelleket javasoltam. A közelítő modellek fontos tulajdonságainak felhasználásával hatékony algoritmusokat származtattam a rendszer állandósult állapotvalószínűségeinek meghatározására. A valóságot követő szimulációs vizsgálatokkal megmutattam, hogy a javasolt módszerek képesek pontosan meghatározni a teljesítőképességi paramétereket. Az általam kidolgozott algoritmusok számítás-igénye $O(c)$ (ahol c a szerverek száma), így azok nagyszámú erőforrással gazdálkodó rendszerek (DHCP szerver, mobil hálózat) méretezésére különösen alkalmasak.

Az M/M/c/N+c visszacsatolásos és újrapróbálkozási sorbanállási sort vizsgáltam. Felismertem és bebizonyítottam, hogy a karakterisztikus mátrix polinomnak $\lfloor (N + c + 2)/2 \rfloor$ db nulla értékű sajátértéke van. A Grassmann által kidolgozott algoritmust módosítottam a nulla értékű sajátérték kezelésére. Megmutattam, hogy az általam módosított eljárás a nagy kapacitású rendszer esetén a hagyományos algoritmusoknál gyorsabban tudja meghatározni a jellemzőket.

3. A CPP/M/c vakációs sorbanállási modell

A virtuális gép-szolgáltatók vakációs alapú karbantartási politikájának elemzésére egy új CPP/M/c vakációs sorbanállási modellt vezettem be. Bebizonyítottam a rendszer stabilitási feltételére vonatkozó tételt, ennek következményeként a vakációs alapú karbantartási politika a túlterhelés veszélye nélkül alkalmazható a virtuális-szerver szolgáltatóknál. Levezettem a pontos képleteket a rendszer állandósult állapot-valószínűségeinek meghatározására. Megadtam a rendszer sztochastikus dekompoziciójára vonatkozó tétel bizonyítását.

Contents

List of Figures

List of Tables

Acknowledgement

The first person I would like to thank is Prof. László Jereb, the supervisor of my PhD dissertation. He is one of my great teachers indeed. I thank him for steps and efforts which initiated my academic and research carrier.

I thank the late Prof. Sándor Csibi for lectures, his view on research and insights.

I am grateful to the support and various opportunities given by Prof. László Pap.

I would like to express my gratitude to Prof. Ram Chakka, my professional collaborative colleague, for exchanging thousands of emails on various topics. I thank Prof. Udo Krieger for his invitation to do joint research in his research group in Bamberg (Germany) on the summer of 2005 and 2006.

I have worked with the former students (Dr. Hung T. Tran, Dr. Zsolt Pándi, Dr. Thang Nhat Le, Csaba Király, Dénes Papp, Nam H. Do, Barnabás Kálmán, Gergő Buchholcz, Van Tuan Cuong, Mai T. Truong, Nguyen Thac Thong). The professional support of these former students and various colleagues in Department of Telecommunications, Budapest University of Technology and Economics is also acknowledged.

Part I

Introduction

1 Motivations and Contributions

1.1 Motivations

The performance modeling and evaluation activities play an important role in the design and operation of ICT (Information and Communication Technologies) systems. According to Kleinrock there are four types of solutions to predict the performance of networks as follows [74, 75]:

1. Conduct a mathematical analysis which yields explicit performance expressions.
2. Conduct a mathematical analysis which yields an algorithmic or numerical procedure.
3. Write and run a simulation program.
4. Build the system and then measure its performance.

Therefore, the search of new mathematical analysis methods for the performance evaluation of ICT systems has been an intensive research area.

The concept of Quasi Birth-Death (QBD) processes, as a generalization of the classical birth and death processes (e. g. the $M/M/1$ queue) was first introduced in the late 1960s by Wallace [123] and Evans [60]. The state space of a QBD process, in the Markovian framework, is defined on a two-dimensional lattice, finite in one dimension (finite or infinite in the other). The random variable in the finite dimension is the phase and the other variable is called the level [82, 97, 102]. Transitions in a QBD process are possible within the same level or between adjacent levels. It is observed that the QBD framework is a mature technique for the performance evaluation of many problems in telecommunications and computer networks [18, 33, 38, 54, 79, 80, 101, 103, 106, 124].

There are four main methods to compute the steady state probabilities of a QBD process.

- The first method by Seelen [108] is based on the truncation of the original Markov process to a finite state Markov chain. Utilizing the special structure an efficient iterative solution algorithm can be applied.

- The second method is to reduce the problem of infinite-states to a linear equation involving vector generating function and some unknown probabilities. The latter are then determined with the aid of the singularities of the coefficient matrix. A comprehensive treatment of that approach, in the context of a discretetime process with a general M/G/1 type structure, is presented in [64].

- The third way of solving these models is the well known matrix geometric method, first proposed by Evans [60, 102]. In this method a nonlinear matrix equation is first formed from the system parameters and the minimal nonnegative solution R of this equation is computed by an iterative method. The invariant vector is then expressed in terms of the powers of R. However, the number of iterations which are needed to compute R to a given accuracy is unknown. It can also be shown that for certain parameter values the computational efforts are uncertain and formidably large.

- The fourth method is known as the spectral expansion method [21, 31, 97]. It is based on expressing the invariant vector of the process in terms of the eigenvalues and left eigenvectors of a certain matrix polynomial. The generating function approach and the spectral expansion method are closely related. However, the latter computes steady state probabilities directly using an algebraic expansion while the former provides them through a transform. The interested reader can find more detail on the comparison of methods for QBD processes in [119].

It is confirmed by a number of works that the spectral expansion method is better than the matrix geometric one in a number of aspects [68, 70, 97]. Furthermore, the spectral expansion method is proved to be a mature technique for the performance analysis of various problems [21, 24–27, 29, 31–37, 47, 51, 52, 54, 55, 58, 61, 62, 66, 68, 96, 97, 117, 118, 120, 127] and can be used for network dimensioning purposes [44, 128].

1.2 Contributions

After the short summary of the background on the spectral expansion method in Chapter 2 in Part I, this dissertation summarizes the author's contributions (see the list of the author's publications in Appendix B) to the performance evaluation of ICT systems using the spectral expansion method in Chapters 3, 4, 5, 6, 7 and 8. These chapters are organized into three topics:

(1) a generalized Markov multi-server queue and ICT applications are presented in Part II,

(2) retrial queues and ICT applications are discussed in Part III, and

(3) vacation queues and an application are presented Part IV.

Chapter 3 in Part II: The Generalized Markov Multi-Server Queue

We propose a new generalized multi-server queue, referred here as the HetSigma queue [33, 49], in the Markovian framework, in order to model (cf. [33, 34, 49, 51, 54]) nodes in modern telecommunication networks. The queue has many of the necessary properties such as, joint (or, individual) Markov modulation of the arrival and service processes, superposition of K CPP (compound Poisson process) streams of (positive) customer arrivals, and a CPP of negative customer arrival stream in each of the modulating phases, a multi-server with c non-identical (can also be identical) servers, GE (generalized exponential) service times in each of the modulating phases and a buffer of finite or infinite capacity. Thus, the model can accommodate correlations of the inter-arrival times of batches, geometric as well as non-geometric batch size distributions of customers in both arrivals and services. The use of negative customers can facilitate modelling server failures, packet losses, load balancing, channel impairment in wireless networks, and in many other applications. An exact and computationally efficient solution of this new queue for the steady state probabilities and performance measures is developed. A closed form for the coefficient matrices of the characteristic matrix polinomial is derived. The fundamental properties which serve as the background for the efficient computational algorithm to obtain the steady state probabilities of the HetSigma queue are presented and proved.

The proposed model does provide a useful tool for the performance analysis of many problems of the emerging telecommunication systems and networks. The queuing model and its variants were successfully used to model Optical Burst/Packet (OBS) Switching networks [23, 34, 50], MPLS networks [32, 55, 104] and another variant to successfully compute the performance of the Apache web server [54]. The *HetSigma* model has been applied to model wireless networks [33, 34, 51].

Chapter 4 in Part II: Modeling Apache Web Server Software

We consider the non-threaded multi-processing module (MPM) *Prefork* of Apache's UNIX architecture [54]. For the first time, we propose a mathematically tractable analytic queueing model based on the *HetSigma* one to predict the performance of an Apache server with its load-dependent dynamic pool size. It is derived from certain Markov assumptions concerning the number of active service processes and applies a decomposition approach to the client population, the Web server and the TCP transport system. In particular, we suppose that TCP connection requests demanding the support of a HTTP service process follow a Markov-Modulated Compound Poisson Process (MMCPP). We use the MMCPP because it can cope with the fluctuations of the arrival rate, the autocorrelation of inter-arrival times of requests and also with simultaneous arrivals of multiple requests. As a consequence, the well-known WAGON traffic model of [90] is a special case of our more general workload model.
In addition to the development of a tractable analytic model we have also measured the performance of a real Apache Web server where the number of service processes does not follow a Markov property. Comparing the numerical results of the analytic model with those obtained by the actual measurements, we observe a rather good coincidence of mean values and an acceptable accuracy of our predictions regarding control purposes. Moreover, the results are used to identify the impact of major control parameters of an Apache configuration on relevant performance measures.

Chapter 5 in Part III: A Model for the Performance Evaluation of DHCP

We propose a retrial queueing model to approximate the performability of the DHCP dynamic allocation mechanism [48]. We show that interarrival times of DHCP requests from clients follow the exponential distribution. We make a relaxed assumption concerning the lease time sent by a DHCP server and the retrial process of clients. We develop an efficient computational algorithm to calculate the steady state probabilities and the performance measures of a continuous time discrete state Markov (CTMC) process associated with the proposed retrial queue. The computational algorithm has the computational complexity of the order $O(c)$, where c is the number of IP addresses in a DHCP server. Besides possessing the capability of tackling the problem with large c in *a very short time*, it also gives numerically stable solution. Here, it is worth emphasizing that many existing solutions [14, 84, 97, 99] of QBD processes have the computational time complexity of $O(c^3)$. It is shown via simulation of more detailed model than an analytical abstract one that the proposed approach is accurate to calculate the performance of the interaction

between the behavior of clients and the DHCP mechanism. A numerical study is also performed, which provides an insight for the impact of trade-off parameters and factors on the operation of the DHCP mechanism.

Chapter 6 in Part III: A New Algorithm to a Retrial Queue for Wireless Networks

Retrial queueing models have been applied to evaluate the impact of arriving new and handover calls on the call admission control mechanisms of cellular mobile networks. The fact that the retrial rate depends on the number of retried calls waiting in the system leads to an analytically intractable model. Therefore, approximation procedures should be used to compute the performability of the system. However, there exists a numerical problem concerning a recursive computational algorithm for the probabilities of the truncated state space, which is first identified by the author of this dissertation. Namely, the consideration of guard channels results on calculations with negative terms in the recursive algorithm. Negative terms and extremely small values involved in the computation are the main causes for a numerical instability in the recursive algorithm.

We construct a new numerically stable algorithm in order to solve the problem [46]. Moreover, we prove that the distribution of the number of calls in the orbit is the "mixture" of geometric distributions in the presence of the guard channel concept. It is worth emphasizing that the numerical stability, the small memory requirement for the implementation of the algorithm and the capability to obtain results with a large number of channels within a short time are the strengths of the new approach.

Chapter 7 in Part III: A New Algorithm for a Multi-Server Feedback Retrial Queue

We deal with an algorithm to find the eigenvalues of the matrix quadratic equation of a tridiagonal form based on the sign variations of the Sturm sequences. We consider a case where multiple zero-eigenvalues are involved.

The example for investigation is the M/M/c/N+c feedback queue with constant retrial rate, which is solved by the matrix-geometric method in [80]. However, the existing approaches face the state explosion problem when the queueing capacity $(N + c)$ is large. We prove [45] that the number of zero-eigenvalues of the characteristic matrix polynomial associated with the balance equation is $\lfloor (N + c + 2)/2 \rfloor$. As a consequence, the remaining eigenvalues inside the unit circle can be computed in a quick manner based on the Sturm sequences. It is worth emphasizing that the algorithm of [67] should be slightly modified in order to determine the remaining eigenvalues of the characteristic matrix polynomial inside the unit circle. Numerical results are presented to compare computation times which are needed by the

matrix geometric method, the pure spectral expansion approach, the method proposed by Naoumov et al. and the new algorithm to calculate the steady state probabilities. The comparison [45] clearly demonstrates the advantage of the new algorithm on the computation of the steady probabilities of the queue with a large queueing capacity.

Chapter 8 in Part IV: Vacation Queues and an Application to a Virtualized Server Environment

In order to extend the application opportunities of queues with working vacation to the performance evaluation of practical systems with bursty traffic, we introduce the CPP/M/c queue with working vacations [53]. In these queues customers arrive in a batch having a geometric size distribution. We present a method and give closed form expressions for the steady state probabilities. We also show that a stochastic decomposition property exists for the queue with working vacations.

To model the queueing and congestion phenomena arising from maintenance tasks of a virtualized server environment, we use the CPP/M/c multi-server system with Poisson batch arrivals and working vacations [53].

To model the maintenance activities, we have assumed that a certain number of servers goes simultaneously to a maintenance state for a random period when they have completed the service of jobs and find no further requests in the waiting line. By a relatively simple model we theoretically prove a property which has a significant impact on the organization of maintenance activities of virtualized servers [53]. It means that instead of migrating virtual servers to expensive physical backup servers during software maintenance, a wise and simple strategy based on the vacation approach can be used. That is, the system is not overloaded if we organize the maintenance according to the vacation model. We believe that our model can be useful for administrators to choose an appropriate parameter set for the maintenance activities.

2

The Spectral Expansion Method for QBD Processes

2.1 Definitions

Consider a two-dimensional continuous time, irreducible Markov chain X=$\{(I(t),\, J(t)),\, t \geq 0\}$ on lattice strips.

- $I(t)$ is called the phase (e.g.: the state of the environment) of the system at time t. Random variable $I(t)$ takes values from the set $\{0, 1, 2, \ldots, N\}$, where N is the maximum value of the phase variable.
- Random variable $J(t)$ is often called the level of the system at time t and takes a set of values $\{0, 1, \ldots, L\}$, where L can be finite or infinite.

The state space of Markov chain X is $\{(i, j) : 0 \leq i \leq N,\ 0 \leq j \leq L\}$. Let $p_{i,j}$ denote the steady state probability of the state (i, j) as

$$p_{i,j} = \lim_{t\to\infty} \mathbb{P}(I(t) = i, J(t) = j); \quad (i = 0, \ldots, N; \quad j = 0, 1, \ldots, L).$$

Vector $\mathbf{v}_j$ is defined as

$$\mathbf{v}_j = (p_{0,j}, \ldots, p_{N,j}) \quad (j = 0, 1, \ldots, L).$$

Since the sum of all the probabilities $p_{i,j}$ is 1.0, we have the normalization equation as

$$\sum_{j=0}^{L} \mathbf{v}_j \mathbf{e}_{N+1} = 1\ , \tag{2.1}$$

where $\mathbf{e}_{N+1}$ is a column vector of size $N + 1$ with all ones.

2.1.1 Continuous Time QBD Processes

Definition 2.1. *A continuous time Quasi-Birth-and-Death (QBD) process is formed when one-step transitions of the Markov chain X are allowed to states in the same level or in the two adjacent levels. That is, the dynamics of the process are driven by*

(a) purely phase transitions. $A_j(i,k)$ denotes the transition rate from state (i,j) to state (k,j) $(0 \le i,k \le N; j = 0,1,\ldots,L)$;

(b) one−step upward transitions. $B_j(i,k)$ is the transition rate from state (i,j) to state $(k,j+1)$ $(0 \le i,k \le N; j = 0,1,\ldots,L)$;

(c) one−step downward transitions. $C_j(i,k)$ is the transition rate from state (i,j) to state $(k,j-1)$ $(0 \le i,k \le N; j = 0,1,\ldots,L)$.

Let A_j, B_j and C_j denote $(N+1)\times(N+1)$ matrices with elements $A_j(i,k)$, $B_j(i,k)$ and $C_j(i,k)$, respectively. Note that their diagonal elements are zero. Let D^{A_j}, D^{B_j} and D^{C_j} be the diagonal matrices of size $(N+1)\times(N+1)$, defined by the i^{th} $(i = 0,\ldots,N)$ diagonal element as follows

$$D^{A_j}(i,i) = \sum_{k=0}^{N} A_j(i,k);\ D^{B_j}(i,i) = \sum_{k=0}^{N} B_j(i,k);\ D^{C_j}(i,i) = \sum_{k=0}^{N} C_j(i,k).$$

For the convenience of the presentation we define matrices $B_{-1} = 0$, $B_L = 0$ and $C_0 = 0$.

The steady state balance equations satisfied by the vectors $\mathbf{v}_j$ are

$$\mathbf{v}_j[D^{A_j} + D^{B_j} + D^{C_j}] = \mathbf{v}_{j-1}B_{j-1} + \mathbf{v}_j A_j + \mathbf{v}_{j+1}C_{j+1}\ \ \forall j\ . \qquad (2.2)$$

Assume that there exist thresholds T_1^* and T_2^* such that

$$\begin{aligned} A_j &= A \quad (T_2^* \ge j \ge T_1^*),\\ B_j &= B \quad (T_2^* \ge j \ge T_1^* - 1),\\ C_j &= C \quad (T_2^* + 1 \ge j \ge T_1^*). \end{aligned}$$

D^A, D^B and D^C are the corresponding diagonal matrices with the diagonal elements as

$$D^A(i,i) = \sum_{k=1}^{N} A(i,k),\ D^B(i,i) = \sum_{k=1}^{N} B(i,k),\ D^C(i,i) = \sum_{k=1}^{N} C(i,k).$$

The generator matrix of the QBD process is written as

$$\begin{bmatrix}
A_0^{(1)} & B_0 & 0 & 0 & \dots & \dots & \dots & \dots & \dots \\
C_1 & A_1^{(1)} & B_1 & 0 & \dots & \dots & \dots & \dots & \dots \\
0 & C_2 & A_2^{(1)} & B_2 & \dots & \dots & \dots & \dots & \dots \\
\vdots & \vdots & \vdots & \ddots & \dots & \dots & \dots & \dots & \\
0 & 0 & \dots & C_{T_1^*-1} & A_{T_1^*-1}^{(1)} & B_{T_1^*-1} & 0 & 0 & \dots \\
0 & 0 & \dots & 0 & C_{T_1^*} & A_{T_1^*}^{(1)} & B_{T_1^*} & 0 & \dots \\
0 & 0 & \dots & 0 & 0 & C_{T_1^*+1} & A_{T_1^*+1}^{(1)} & B_{T_1^*+1} & \dots \\
\vdots & \vdots & \vdots & \vdots & \dots & \dots & \ddots & \dots & \dots
\end{bmatrix}$$

$$= \begin{bmatrix}
A_0^{(1)} & B_0 & 0 & 0 & \dots & \dots & \dots & \dots & \dots & \dots \\
C_1 & A_1^{(1)} & B_1 & 0 & \dots & \dots & \dots & \dots & \dots & \dots \\
0 & C_2 & A_2^{(1)} & B_2 & \dots & \dots & \dots & \dots & \dots & \dots \\
\vdots & \vdots & \vdots & \ddots & \dots & \dots & \dots & \dots & \dots & \dots \\
0 & 0 & \dots & C_{T_1^*-1} & A_{T_1^*-1}^{(1)} & Q_0 & 0 & 0 & \dots & \dots \\
0 & 0 & \dots & 0 & C_{T_1} & Q_1 & Q_0 & 0 & \dots & \dots \\
0 & 0 & \dots & 0 & 0 & Q_2 & Q_1 & Q_0 & \dots & \dots \\
0 & 0 & \dots & 0 & 0 & 0 & Q_2 & Q_1 & Q_0 & \dots \\
\vdots & \vdots & \vdots & \vdots & \dots & \dots & \ddots & \dots & \dots & \dots
\end{bmatrix},$$

where $A_j^{(1)} = A_j - D^{A_j} - D^{B_j} - D^{C_j}$.

The j-independent balance equations can be rewritten as follows

$$\mathbf{v}_{j-1}Q_0 + \mathbf{v}_j Q_1 + \mathbf{v}_{j+1}Q_2 = 0 \quad (T_1^* \le j \le T_2^*), \tag{2.3}$$

where $Q_0 = B$, $Q_1 = A - D^A - D^B - D^C$, $Q_2 = C$.

2.1.2 Continuous Time QBD-M Processes

Definition 2.2. *The Markov chain X is called a continuous time quasi simultaneous-bounded-multiple births and simultaneous-bounded-multiple deaths (QBD-M) process if the balance equation for level j can be written as*

$$\sum_{i=0}^{y} \mathbf{v}_{j-y_1+i}Q_i = 0 \quad (T_1 \le j \le T_2), \tag{2.4}$$

where y, y_1, T_1 and T_2 are integer constants for a specific system, while Q_i are j-independent matrices of size $(N+1)\times(N+1)$.

The class of continuous time QBD-M processes is the generalization of QBD. As we will see in Chapter 3 and 4, QBD-M processes can be used to evaluate the performance of systems with multiple arrivals and/or multiple departures.

2.1.3 Generalized Exponential Distribution

Definition 2.3. *The versatile Generalized Exponential (GE) distribution is given in the following form:*

$$F(t) = P(W \leq t) = 1 - (1 - \phi)e^{-\mu t} \ (t \geq 0), \tag{2.5}$$

where W is the GE random variable with parameters μ, ϕ. Thus, the GE parameter estimation can be by obtained by $1/\nu$, the mean, and C^2_{coeff}, the squared coefficient of variation of the inter-event time of the sample as

$$1 - \phi = 2/(C^2_{coeff} + 1) \ ; \ \mu = \nu(1 - \phi) \ . \tag{2.6}$$

Remarks. For $C^2_{coeff} > 1$, the GE model is a mixed-type probability distribution having the same mean and coefficient of variation, and with one of the two phases having zero service time, or a bulk type distribution with an underlying counting process equivalent to a Batch (or Bulk) Poisson Process (BPP) with batch-arrival rate μ and geometrically distributed batch size with mean $1/(1 - \phi)$ and SCV $(C^2_{coeff} - 1)/(1 + C^2_{coeff})$ (cf. [111]). It can be observed that there is an infinite family of BPP's with the same GE-type inter-event time distribution. It is shown that, among them, the BPP with geometrically distributed bulk sizes (referred as the CPP) is the only one that constitutes a renewal process (the zero inter-event times within a bulk/batch are *independent* if the bulk size distribution is geometric [78]). The GE distribution is versatile, possessing pseudo-memoryless properties which make the solution of many GE-type queuing systems analytically tractable [78]. The choice of the GE distribution is often motivated by the fact that measurements of actual inter-arrival or service times may be generally limited and so only a few parameters (for example the mean and variance) can be computed reliably. Typically, when only the mean and variance can be relied upon, a choice of a distribution which implies least bias is that of GE-type distribution [78, 111].

Definition 2.4 (CPP)**.** *The inter-arrival time distribution of customers of the Compound Poisson Process (CPP) is GE with parameters (σ, θ). That is, the inter-arrival time probability distribution function is $1-(1-\theta)e^{-\sigma t}$. Thus, the arrival* point*-process has batches arriving at each point having independent and geometric batch-size distribution. Specifically the probability that a batch is of size s is $(1 - \theta)\theta^{s-1}$.* □

2.2 The Spectral Expansion Method for QBD-M Processes

Let $Q(\lambda)$ denote the characteristic matrix polynomial associated with the balance equation (2.4) as

$$Q(\lambda) = \sum_{i=0}^{y} Q_i \lambda^i. \tag{2.7}$$

If $(\lambda, \boldsymbol{\psi})$ is the left-eigenvalue and eigenvector pair of the characteristic matrix-polynomial, the following equation holds

$$\boldsymbol{\psi} Q(\lambda) = 0; \;\; det[Q(\lambda)] = 0. \tag{2.8}$$

Assume that $Q(\lambda)$ has d pairs of eigenvalue-eigenvectors. For the k^{th} $(k = 1, \ldots, d)$ non-zero eigenvalue-eigenvector pair, $(\lambda_k, \boldsymbol{\psi}_k)$, by substituting $\mathbf{v}_j = \boldsymbol{\psi}_k \lambda_k^j$ $(T_1 - y_1 \leq j \leq T_2 - y_1 + y)$ in the equations (2.4), it can be seen that this set of equations is satisfied. Hence, that is a particular solution. The equations can even be satisfied with $\boldsymbol{\psi}_k \lambda_k^{j+l_k}$ for any real l_k. It is easy to prove that the general solution for $\mathbf{v}_j$ is the linear sum of all the factors $(\boldsymbol{\psi}_k \lambda_k^{j-T_1+y_1})$ as

$$\mathbf{v}_j = \sum_{l=1}^{d} a_l \boldsymbol{\psi}_l \lambda_l^{j-T_1+y_1} \;\; (j = T_1 - y_1, T_1 - y_1 + 1, \ldots, T_2 - y_1 + y), \tag{2.9}$$

where a_l $(l = 1, \ldots, d)$ are constants.

Therefore, the steady state probability can be written as follows

$$p_{i,j} = \sum_{l=1}^{d} a_l \boldsymbol{\psi}_l(i) \lambda_l^{j-T_1+y_1} \;\; (j = T_1 - y_1, T_1 - y_1 + 1, \ldots, T_2 - y_1 + y). \tag{2.10}$$

An interesting property can be observed concerning the eigenvalues of $Q(\lambda)$ for QBD-M process X as follows. If $(\lambda_k, \boldsymbol{\psi}_k)$ is the left-eigenvalue and eigenvector pair of $Q(\lambda)$, then $(1/\lambda_k, \boldsymbol{\psi}_k)$ is the left-eigenvalue and eigenvector pair of $\overline{Q}(\lambda) = \sum_{i=0}^{y} Q_{y-i} \lambda^i$, the characteristic matrix polynomial of the dual process of X (cf. [31]).

2.2.1 Infinite QBD-M Processes

When L and T_2 are infinite (unbounded), consider the probability sum

$$\sum_{j=T_1-y_1}^{\infty} p_{i,j} = \sum_{j=T_1-y_1}^{\infty} \sum_{l=1}^{d} a_l \boldsymbol{\psi}_l(i) \lambda_l^{j-T_1+y_1}. \tag{2.11}$$

In order to ensure that this sum is less or equal to 1.0, the necessary condition is

$$a_k = 0,\ if\ |\lambda_k| \geq 1.$$

Thus, by renumbering the eigenvalues inside the unit circle, the general solution is obtained as

$$\mathbf{v}_j = \sum_{l=1}^{\chi} a_l \boldsymbol{\psi}_l \lambda_l^{j-T_1+y_1} \quad (j = T_1 - y_1, T_1 - y_1 + 1, \ldots), \tag{2.12}$$

$$p_{i,j} = \sum_{l=1}^{\chi} a_l \boldsymbol{\psi}_l(i) \lambda_l^{j-T_1+y_1} \quad (j = T_1 - y_1, T_1 - y_1 + 1, \ldots). \tag{2.13}$$

where χ is the number of eigenvalues that are present strictly within the unit circle. These eigenvalues appear some as real and others as complex-conjugate pairs, and as do the corresponding eigenvectors.

In order to determine the steady state probabilities, the unknown constants a_l are to be determined. Their number is χ. We still have other unknowns $\mathbf{v}_0, \mathbf{v}_1, \ldots, \mathbf{v}_{T_1-y_1-1}$. These unknowns are determined with the aid of the state dependent balance equations (their number is $T_1(N+1)$) and the normalization equation (2.1), out of which $T_1(N+1)$ are linearly independent. These equations can have a unique solution if and only if $(T_1 - y_1)(N+1) + \chi = T_1(N+1)$, or equivalently

$$\chi = y_1(N+1) \tag{2.14}$$

holds.

2.2.2 Finite QBD-M Processes

In order to compute the steady state probabilities, the unknown constants a_l are to be determined. Their number is d. We still have other unknowns $\mathbf{v}_0, \mathbf{v}_1, \ldots, \mathbf{v}_{T_1-y_1-1}, \mathbf{v}_{T_2-y_1+y+1}, \mathbf{v}_{T_2-y_1+y+2}, \ldots, \mathbf{v}_L$. Therefore, the number of unknowns is

$$d + (T_1 - y_1)(N+1) + (L - T_2 + y_1 - y)(N+1).$$

These unknowns are determined with the aid of the state dependent balance equations (their number is $T_1(N+1) + (L - T_2)(N+1)$) and the normalization equation, out of which $T_1(N+1) + (L-T_2)(N+1)$ are linearly independent. These equations can have a unique solution if and only if

$$d + (T_1 - y_1)(N+1) + (L - T_2 + y_1 - y)(N+1) = T_1(N+1) + (L - T_2)(N+1),$$

equivalently

$$d = y(N+1) \tag{2.15}$$

holds.

2.3 Comments and References

Besides the spectral expansion method, there are a number of methods to solve QBD and QBD-M processes such as in [3, 16, 40, 84, 101]. Some approaches are based on the fact that QBD-M processes can also be considered as a special case of Markov chains having non-skip-free structure, such as M/G/1 chains with multiple boundary or combined M/G/1-G/M/1 type structured chains. As a consequence, some recent numerical methods for non-skip-free Markov chains apply more or less straightforwardly to QBD-M processes. A set of methods presented in [17, 65] takes advantage of the special structure of the Markov chain along with that of re-blocking. Works [4, 71, 72] apply the concept of the linear system theory based on the generalized invariant subspace background.

Note that the focus of this dissertation is the development of performability analysis solutions based on the spectral expansion method. Therefore, the detailed discussion of other methods is outside the scope of this dissertation.

Part II

Generalized Markov Multi-Server Queue and ICT Applications

3

The Generalized Markov Multi-Server Queue[‖]

3.1 System Description

The *HetSigma* queue is defined as the MM $\sum_{k=1}^{K} CPP_k$/GE/c/L G-queue with heterogenous servers. The arrival process is the superposition of the MM $\sum_{k=1}^{K} CPP_k$ and an independent CPP of negative customers (denoted by G). The MM$\sum_{k=1}^{K} CPP_k$ is obtained by Markov modulation of the parameters of the superposition of K independent CPP streams. That is, the K independent CPP's are jointly Markov modulated by a single modulating Markov process. L is the capacity of the system.

We consider a case where the arrival (of positive and negative customers) and service processes are modulated by the same continuous time, irreducible Markov phase process. The system is a multi-server queue with c heterogeneous servers. It is described below, by illustrating the various components and their interactions.

3.1.1 Modulation

The arrival process is modulated by a continuous time, irreducible Markov process with $N+1$ states (or phases of modulation). Let Q be the generator matrix of this process, given by

$$Q = \begin{bmatrix} -q_0 & q_{0,2} & \cdots & q_{0,N} \\ q_{1,0} & -q_1 & \cdots & q_{1,N} \\ \vdots & \vdots & \ddots & \vdots \\ q_{N,0} & q_{N,1} & \cdots & -q_N \end{bmatrix},$$

[‖]R. Chakka and Tien Van Do. The MM $\sum_{k=1}^{K} CPP_k/GE/c/L$ G-Queue with Heterogeneous Servers: Steady state solution and an application to performance evaluation. *Performance Evaluation*, 64:191–209, March 2007.

where $q_{i,k}(i \neq k)$ is the instantaneous transition rate from phase i to phase k, and the diagonal elements, $-q_i = -\sum_{j=0}^{N} q_{i,j}$ $(i = 0, \ldots, N)$, where $q_{i,i} = 0 \ \ \forall i$.

Let $\mathbf{r} = (r_0, r_1, \ldots, r_N)$ be the vector of equilibrium probabilities of the modulating phases. Then, $\mathbf{r}$ is uniquely determined by the equations:

$$\mathbf{r}Q = 0; \qquad \mathbf{r}\mathbf{e}_{N+1} = 1,$$

where $\mathbf{e}_{N+1}$ stands for the column vector with $N+1$ elements, each of which is unity.

3.1.2 Customer Arrival Process

The arrival process is the superposition of K independent CPP [41] arrival streams of (positive) customers. The positive customers of the different arrival streams are not distinguishable. In the modulating phase i, the parameters of the GE inter-arrival time distribution of the k^{th} $(k = 1, 2, \ldots, K)$ positive customer arrival stream are $(\sigma_{i,k}, \theta_{i,k})$. That is, the inter- arrival time probability distribution function is $1 - (1 - \theta_{i,k})e^{-\sigma_{i,k}t}$, in phase i, for the k^{th} stream of positive customers. Thus, all the K arrival *point*-processes can be seen as batch-Poisson, with batches arriving at each point having geometric size distribution. Specifically, the probability that a batch is of size s is $(1 - \theta_{i,k})\theta_{i,k}^{s-1}$, in phase i, for the k^{th} stream of positive customers.

Let $\sigma_{i,.}, \overline{\sigma_{i,.}}$ be the average arrival rate of customer batches and customers in phase i respectively. Let $\sigma, \overline{\sigma}$ be the overall average arrival rate of batches and customers respectively. Then, we get

$$\sigma_{i,.} = \sum_{k=1}^{K} \sigma_{i,k} \ ; \ \ \overline{\sigma_{i,.}} = \sum_{k=1}^{K} \frac{\sigma_{i,k}}{(1 - \theta_{i,k})} \ ; \ \ \sigma = \sum_{i=0}^{N} \sigma_{i,.} r_i \ ; \ \ \overline{\sigma} = \sum_{i=0}^{N} \overline{\sigma_{i,.}} r_i. \qquad (3.1)$$

Because of the superposition of many CPP's, the overall arrivals in phase i can be considered as bulk-Poisson ($M^{[x]}$) with arrival rate $\sigma_{i,.}$ and with a batch size distribution in phase i, $\{\pi_{l/i}\}$, that is more general than mere geometric. The probability that this batch size is l (in phase i) is given by

$$\pi_{l/i} = \sum_{k=1}^{K} \frac{\sigma_{i,k}}{\sigma_{i,.}} (1 - \theta_{i,k}) \theta_{i,k}^{l-1}. \qquad (3.2)$$

The overall batch size distribution is then given by $\pi_{l/.} = \sum_{i=0}^{N} r_i \pi_{l/i}$. The joint probability $\{\pi_{i,l}\}$ (*i.e.,* the probability that a given batch is from phase i and of length l) is then given by $\pi_{i,l} = r_i \pi_{l/i}$.

3.1.3 Negative Customer Arrival Process

The arrival process is the superposition of K independent CPP [41] arrival streams of (positive) customers and an independent CPP of negative customers. In the modulating phase i, the parameters of the GE inter-arrival time distribution of the negative customer arrival process are (ρ_i, δ_i). That is, the inter- arrival time probability distribution function is $1-(1-\delta_i)e^{-\rho_i t}$ for the negative customers in phase i.

The uses of negative customers with appropriate killing discipline are of many fold, *viz.* for facilitating flow control studies, load balancing studies, to model breakdowns and to model packet losses caused by the arrival of corrupted packets, as explained in [27].

A negative customer removes a positive customer in the queue, according to a specified *killing discipline*. We consider here a variant of the RCE killing discipline (removal of customers from the end of the queue), where the most recent positive arrival is removed, but which does *not* allow a customer actually in service to be removed: a negative customer that arrives when there are no positive customers waiting to start service has no effect. We may say that customers in service are immune to negative customers or that the service itself is *immune servicing*. Such a killing discipline is suitable for modelling of load balancing where work is tranferred from overloaded queues but never work that is actually in progress.

When a batch of negative customers of size $l\,(1 \leq l < j-c)$ arrives, l positive customers are removed from the end of the queue leaving the remaining $j-l$ positive customers in the system. If $l \geq j-c \geq 1$, then $j-c$ positive customers are removed, leaving none waiting to commence service (queue length equal to c). If $j \leq c$, the negative arrivals have no effect.

$\overline{\rho_i}$, the average arrival rate of negative customers in phase i and $\overline{\rho}$, the overall average arrival rate of negative customers are given by

$$\overline{\rho_i} = \frac{\rho_i}{1-\delta_i} \quad ; \quad \overline{\rho} = \sum_{i=0}^{N} r_i \overline{\rho_i}. \tag{3.3}$$

3.1.4 The GE Multi-Server

The service facility has c *heterogeneous* servers in parallel. A number of scheduling policies can be thought of. Though, in principle, a number of scheduling policies can indeed be modelled by following our methodology, the one that we have adopted in this Chapter, for illustration and detailed study, is as follows. A set of service priorities is chosen by giving each server a unique service priority: 1 is the highest and c is the lowest. This set can be chosen arbitrarily from the $c!$ different possible ways. However, the impact of choosing service priorities can be very high on the performance measures, whose study is not in the scope of this Chapter.

Each server is then numbered, without loss of generality, by its own service priority. The GE-distributed service time parameters of the nth server ($n = 1, 2 \ldots, c$), in phase i, are denoted by $(\mu_{i,n}, \phi_{i,n})$.

The service discipline is FCFS (First Come First Scheduled for service) and each server serves at most one positive customer at any given time. Customers, on their completion of service, leave the system. When the number of customers in the system, j, (including those in service if any) is $\geq c$, then only c customers are served with the rest $(j-c)$ waiting for service. When $j < c$, only the first j servers, (*i.e.*, servers numbered $1, 2, \ldots, j$), are occupied and the rest are idle. This is made possible by what is known as customer switching. Thus, when server n becomes idle, an awaiting customer would be taken up for service. If there is no awaiting customer, then a customer that is being served by the lowest possible priority server (*i.e.*, among servers $(c, c-1, \ldots, n+1)$) switches to server n. In such a switching, the (batch) service time is governed by either *resume or repeat with resampling*, thus preserving the Markov property. The switching is instantaneous and the switching time is treated negligible. Negative customers neither wait in the queue, nor are served.

The operation of the GE server is similar to that described for the CPP arrival processes above. However, the batch size associated with a service completion is bounded by one more than the number of customers waiting to commence service at the departure instant. When $c \leq j < L+1$, the maximum batch size at a departure instant is $j-c+1$, only one server being able to complete a service period at any one instant under the assumption of exponentially distributed batch-service times. Thus, here the probability that a departing batch is of size s is, $\sum_{n=1}^{c} \frac{\mu_{i,n}(1-\phi_{i,n})\phi_{i,n}^{s-1}}{\mu_{i.}}$ for $1 \leq s \leq j-c$ and $\sum_{n=1}^{c} \frac{\mu_{i,n}\phi_{i,n}^{j-c}}{\mu_{i.}}$ for $s = j-c+1$, where $\mu_{i.} = \sum_{n=1}^{c} \mu_{i,n}$. However, when $1 \leq j \leq c$, the departing batch has size 1 with probability 1.0 since each customer is already engaged by a server and there are then no customers waiting to commence service.

It is assumed that the first positive customer in a batch arriving at an instant when the queue length is less than c (so that at least one server is free) *never* skips service, *i.e.* always has an exponentially distributed service time [27]. However, even without this assumption the methodology described in this Chapter is still applicable.

The assumption, that the arrival and service processes are modulated by the same continuous time, irreducible Markov process with $N+1$ states (phases or modulating phases), does not limit the usage of the model. That is because, the case of different and independent modulating processes for the arrivals and the services can be traced back to our model: *i.e.*, if the number of the phases in the modulation of the arrival process is N_a and that of the service process is N_s independently, then we can convert this case into our model by considering an appropriate joint modulating Markov process with

$N + 1 = N_s N_a$ states.

3.1.5 Condition for Stability

L is the queueing capacity, in all phases, including the customers in service, if any. L can be finite or infinite. We assume, when the number of customers is j and the arriving batch size of positive customers is greater than $L - j$ (assuming finite L), that only $L - j$ customers are taken in and the rest are rejected.

When L is finite, the system is ergodic since the representing Markov process is irreducible. Otherwise, *i.e.* when the queueing capacity is unbounded, the overall average departure rate increases with the queue length, and its maximum (the overall average departure rate when the queue length tends to ∞) can be determined as

$$\overline{\mu} = \sum_{n=1}^{c} \sum_{i=0}^{N} \frac{r_i \mu_{i,n}}{1 - \phi_{i,n}}. \tag{3.4}$$

Hence, we conjecture the necessary and sufficient condition for the existence of steady state probabilities is

$$\overline{\sigma} < \overline{\rho} + \overline{\mu}. \tag{3.5}$$

3.2 The Steady State Balance Equations

The state of the system at any time t can be specified completely by two integer-valued random variables, $I(t)$ and $J(t)$. $I(t)$ varies from 0 to N (known as operative states), representing the phase of the modulating Markov chain, and $0 \leq J(t) < L + 1$ represents the number of positive customers in the system at time t, including those in service. The system is now modelled by a continuous time discrete state Markov process, $\overline{Y}$ (Y if L is infinite), on a rectangular lattice strip. Let $I(t)$, the operative state, vary in the horizontal direction and $J(t)$, the queue length or the *level*, in the vertical direction.

We denote the steady state probabilities by $\{p_{i,j}\}$, where $p_{i,j} = \lim_{t\to\infty} \mathbb{P}(I(t) = i, J(t) = j)$, and let $\mathbf{v}_j = (p_{0,j}, \ldots, p_{N,j})$.

The process $\overline{Y}$ evolves due to the following instantaneous transition rates:

(a) $q_{i,k}$ – purely lateral transition rate – from state (i, j) to state (k, j), for all $j \geq 0$ and $0 \leq i, k \leq N$ $(i \neq k)$, caused by a phase transition in the Markov chain governing the arrival phase process;

(b) $B_{i,j,j+s}$ – s-step upward transition rate – from state (i, j) to state $(i, j + s)$, for all phases i, caused by a new batch arrival of size s of positive customers. For a given j, s can be seen as bounded when L is finite and unbounded when L is infinite;

(c) $C_{i,j,j-s}$ – s-step downward transition rate – from state (i,j) to state $(i,j-s)$, $(j-s \geq c+1)$ for all phases i, caused by either a batch service completion of size s or a batch arrival of negative customers of size s;

(d) $C_{i,c+s,c}$ – s-step downward transition rate – from state $(i,c+s)$ to state (i,c), for all phases i, caused by a batch arrival of negative customers of size $\geq s$ or a batch service completion of size $s\,(1 \leq s \leq L-c)$;

(e) $C_{i,c-1+s,c-1}$ – s-step downward transition rate, from state $(i,c-1+s)$ to state $(i,c-1)$, for all phases i, caused by a batch departure of size $s\,(1 \leq s \leq L-c+1)$;

(f) $C_{i,j+1,j}$ – 1-step downward transition rate, from state $(i,j+1)$ to state (i,j), $(c \geq 2\,;\, 0 \leq j \leq c-2)$, for all phases i, caused by a single departure.

The transition matrices can be obtained as follows

$$
\begin{aligned}
B_{i,j-s,j} &= \sum_{k=1}^{K} (1-\theta_{i,k})\theta_{i,k}^{s-1}\sigma_{i,k} \quad (\forall i\ ;\ 0 \leq j-s \leq L-2\ ;\ j-s < j < L)\ ; \\
B_{i,j,L} &= \sum_{k=1}^{K}\sum_{s=L-j}^{\infty} (1-\theta_{i,k})\theta_{i,k}^{s-1}\sigma_{i,k} = \sum_{k=1}^{K} \theta_{i,k}^{L-j-1}\sigma_{i,k} \quad (\forall i\ ;\ j \leq L-1)\ ; \\
C_{i,j+s,j} &= \sum_{n=1}^{c} \mu_{i,n}(1-\phi_{i,n})\phi_{i,n}^{s-1} + (1-\delta_i)\delta_i^{s-1}\rho_i \\
&\qquad\qquad (\forall i\ ;\ c+1 \leq j \leq L-1\ ;\ 1 \leq s \leq L-j) \\
&= \sum_{n=1}^{c} \mu_{i,n}(1-\phi_{i,n})\phi_{i,n}^{s-1} + \delta_i^{s-1}\rho_i \ \ (\forall i\ ;\ j=c\ ;\ 1 \leq s \leq L-c) \\
&= \sum_{n=1}^{c} \phi_{i,n}^{s-1}\mu_{i,n} \qquad\qquad (\forall i\ ;\ j=c-1\ ;\ 1 \leq s \leq L-c+1) \\
&= 0 \qquad\qquad (\forall i\ ;\ c \geq 2\ ;\ 0 \leq j \leq c-2\ ;\ s \geq 2) \\
&= \sum_{n=1}^{j+1} \mu_{i,n} \qquad (\forall i\ ;\ c \geq 2\ ;\ 0 \leq j \leq c-2\ ;\ s=1)\ .
\end{aligned}
$$

Define

$$
\begin{aligned}
B_{j-s,j} &= \mathrm{Diag}\left[B_{0,j-s,j}, B_{1,j-s,j}, \ldots, B_{N,j-s,j}\right] \qquad (j-s < j \leq L)\ ; \\
B_s &= B_{j-s,j} \qquad (j < L) \\
&= \mathrm{Diag}\left[\sum_{k=1}^{K} \sigma_{0,k}(1-\theta_{0,k})\theta_{0,k}^{s-1}, \ldots, \sum_{k=1}^{K} \sigma_{N,k}(1-\theta_{N,k})\theta_{N,k}^{s-1}\right]\ ;
\end{aligned}
$$

$$
\begin{aligned}
\Sigma_k &= \mathrm{Diag}\,[\sigma_{0,k}, \sigma_{1,k}, \ldots, \sigma_{N,k}] \qquad (k = 1, 2, \ldots, K)\ ; \\
\Theta_k &= \mathrm{Diag}\,[\theta_{0,k}, \theta_{1,k}, \ldots, \theta_{N,k}] \qquad (k = 1, 2, \ldots, K)\ ; \\
\Sigma &= \sum_{k=1}^{K} \Sigma_k\ ; \\
R &= \mathrm{Diag}\,[\rho_0, \rho_1, \ldots, \rho_N]\ ;\ \Delta = \mathrm{Diag}\,[\delta_0, \delta_1, \ldots, \delta_N]\ ;
\end{aligned}
$$

$$
M_n = \mathrm{Diag}\,[\mu_{0,n}, \mu_{1,n}, \ldots, \mu_{N,n}] \qquad (n = 1, 2, \ldots, c)\ ;
$$

$$
\Phi_n = \mathrm{Diag}\,[\phi_{0,n}, \phi_{1,n}, \ldots, \phi_{N,n}] \qquad (n = 1, 2, \ldots, c)\ ;
$$

$$
\begin{aligned}
C_j &= \sum_{n=1}^{j} M_n \qquad (1 \le j \le c)\ ; \\
&= \sum_{n=1}^{c} M_n = C \qquad (j \ge c)\ ; \\
C_{j+s,j} &= \mathrm{Diag}\,[C_{0,j+s,j}, C_{1,j+s,j}, \ldots, C_{N,j+s,j}]\ ; \\
E &= \mathrm{Diag}(\mathbf{e}_N^{'})\ .
\end{aligned}
$$

Then, we get

$$
\begin{aligned}
B_s &= \sum_{k=1}^{K} \Theta_k^{s-1}(E - \Theta_k)\Sigma_k\ ;\ B_1 = B = \sum_{k=1}^{K} (E - \Theta_k)\Sigma_k\ ; \\
B_{L-s,L} &= \sum_{k=1}^{K} \Theta_k^{s-1}\Sigma_k\ ; \\
C_{j+s,j} &= \sum_{n=1}^{c} M_n(E - \Phi_n)\Phi_n^{s-1} + R(E - \Delta)\Delta^{s-1} \\
&\qquad (c+1 \le j \le L-1\ ;\ s = 1, 2, \ldots, L-j)\ ; \\
&= \sum_{n=1}^{c} M_n(E - \Phi_n)\Phi_n^{s-1} + R\Delta^{s-1} \qquad (j = c\ ;\ s = 1, 2, \ldots, L-c)\ ;
\end{aligned}
$$

$$
\begin{aligned}
&= \sum_{n=1}^{c} M_n\Phi_n^{s-1} \qquad (j = c-1\ ;\ s = 1, 2, \ldots, L-c+1)\ ; \\
&= 0 \qquad (c \ge 2\ ;\ 0 \le j \le c-2\ ;\ s \ge 2)\ ; \\
&= C_{j+1} \qquad (c \ge 2\ ;\ 0 \le j \le c-2\ ;\ s = 1)\ .
\end{aligned}
$$

The steady state balance equations are:

(1) for the L^{th} row or level:

$$\sum_{s=1}^{L} \mathbf{v}_{L-s} B_{L-s,L} + \mathbf{v}_L \left[Q - C - R \right] = 0 \; ; \tag{3.6}$$

(2) for the j^{th} row or level:

$$\sum_{s=1}^{j} \mathbf{v}_{j-s} B_s + \mathbf{v}_j \left[Q - \Sigma - C_j - R I_{j>c} \right] + \sum_{s=1}^{L-j} \mathbf{v}_{j+s} C_{j+s,j} = 0$$
$$(0 \leq j \leq L-1) \; ; \tag{3.7}$$

(3) normalization

$$\sum_{j=0}^{L} \mathbf{v}_j \mathbf{e}_{N+1} = 1 \; . \tag{3.8}$$

Note that $I_{j>c} = 1$ if $j > c$ else 0, and $\mathbf{e}_{N+1}$ is a column vector of size $N+1$ with all ones.

One can observe that there are infinite number of equations in infinite number of unknowns, *viz.* $v_0, v_1, \ldots$, when $L = \infty$. Also, each of the balance equation is infinitely long containing all the infinite number of unknowns, *viz.* $v_0, v_1, \ldots$. The coefficient matrices of the unknown vectors are j-dependent. This is a very complex system of equations for which there is neither an existing solution (exact or approximate) nor a solution methodology. Hence, in the next section we *transform* this system of equations to a *computable* form.

3.3 A Solution Method

Let us consider equations (3.6), (3.7) and (3.8). Each equation has *all* the unknown vectors $\mathbf{v_j}$'s. If L is unbounded, then each of these are infinite number of equations in infinite number of unknowns, $\mathbf{v_j}$'s, and each equation is infinitely long containing all the infinite number of unknowns. Also, the coefficient matrices of $\mathbf{v_j}$ are j-dependent. It may be noted that there has been neither a solution nor a solution methodology to solve these equations. In this Chapter a novel methodology is developed to solve these equations *exactly and efficiently*. First these complicated equations are *transformed* to a computable form. The resulting transformed equations are of the QBD-M type and hence can be solved by one of the several available methods, *viz.* the spectral expansion method, Bini-Meini's method or the matrix-geometric method with folding or block size enlargement [69].

3.3.1 Transforming the Balance Equations

Let the balance equation for level j be denoted by $< \mathbf{j} >$. Hence, $< \mathbf{0} >$, $< \mathbf{1} >$, ..., $< \mathbf{j} >$, ..., $< \mathbf{L} >$ are the balance equations for the levels $0, 1, \ldots, j, \ldots, L$ respectively. Substituting $B_{L-s,L} = \sum_{k=1}^{K} \Theta_k^{s-1}\Sigma_k$ and $B_s = \sum_{k=1}^{K} \Theta_{.k}^{s-1}(E - \Theta_k)\Sigma_k$ in (3.6, 3.7), we get the balance equations for level L and for all the other levels as:

$$< \mathbf{L} > : \qquad \sum_{s=1}^{L}\sum_{k=1}^{K} \mathbf{v}_{L-s}\Theta_k^{s-1}\Sigma_k + \mathbf{v}_L\,[Q - C - R] = 0;$$

$$< \mathbf{L-1} > : \qquad \sum_{s=1}^{L-1}\sum_{k=1}^{K} \mathbf{v}_{L-1-s}\Theta_k^{s-1}(E - \Theta_k)\Sigma_k + \mathbf{v}_{L-1}\,[Q - \Sigma - C_{L-1} - R] + \mathbf{v}_L C_{L,L-1} = 0;$$

$$\vdots$$

$$< \mathbf{j} > : \qquad \sum_{s=1}^{j}\sum_{k=1}^{K} \mathbf{v}_{j-s}\Theta_k^{s-1}(E - \Theta_k)\Sigma_k + \mathbf{v}_j\,[Q - \Sigma - C_j - R] + \sum_{s=1}^{L-j} \mathbf{v}_{j+s}C_{j+s,j} = 0 \;\; (j = L-2, L-3, \ldots, c+K+2);$$

$$< \mathbf{c+K+1} > : \qquad \sum_{s=1}^{c+K+1}\sum_{k=1}^{K} \mathbf{v}_{c+K+1-s}\Theta_k^{s-1}(E - \Theta_k)\Sigma_k + \mathbf{v}_{c+K+1}\,[Q - \Sigma - C_{c+K+1} - R] + \sum_{s=1}^{L-c-K-1} \mathbf{v}_{c+K+1+s}C_{c+K+1+s,c+K+1} = 0;$$

$$\vdots$$

$$< \mathbf{j} > : \qquad \sum_{s=1}^{j}\sum_{k=1}^{K} \mathbf{v}_{j-s}\Theta_k^{s-1}(E - \Theta_k)\Sigma_k + \mathbf{v}_j\,[Q - \Sigma - C_j - RI_{j>c}] + \sum_{s=1}^{L-j} \mathbf{v}_{j+s}C_{j+s,j} = 0 \;\; (j = c+K, c+K-1, \ldots, 0).$$

Define the functions, $F_{K,l}\,(l = 1, 2, \ldots, K)$ and $H_{c,n}\,(n = 1, 2, \ldots, c)$ as

$$\begin{aligned} F_{K,l} &= \sum_{1 \le k_1 < k_2 < \ldots < k_l \le K} \Theta_{k_1}\Theta_{k_2}\ldots\Theta_{k_l} \qquad (l = 1, 2, \ldots, K) \\ &= E \quad \text{if } l = 0 \\ &= 0 \quad \text{if } l \le -1 \text{ or } l > K, \end{aligned} \tag{3.9}$$

$$\begin{aligned} H_{c,n} &= \sum_{1 \le k_1 < k_2 < \ldots < k_n \le c} \Phi_{k_1}\Phi_{k_2}\ldots\Phi_{k_n} \qquad (n = 1, 2, \ldots, c) \\ &= E \quad \text{if } n = 0 \\ &= 0 \quad \text{if } n \le -1 \text{ or } n > c\,. \end{aligned} \tag{3.10}$$

These functions have the following alternate definitions, properties and recursion by which they can be conceived and computed quite easily.

$$\begin{aligned} F_{k,0} &= E \;\;,\;\; F_{k,k} = \prod_{i=1}^{k} \Theta_i \;\; (k = 1, 2, \ldots, K); \\ F_{k,l} &= 0 \;(k = 1, 2, \ldots, K;\, l < 0) \;\; ; \\ F_{k,l} &= 0 \;(k = 1, 2, \ldots, K;\, l > k) \end{aligned} \tag{3.11}$$

$$\begin{aligned} H_{m,0} &= E \;\;,\;\; H_{m,m} = \prod_{i=1}^{m} \Phi_i \;\; (m = 1, 2, \ldots, c); \\ H_{m,n} &= 0 \;(m = 1, 2, \ldots, c;\, n < 0) \;\; ; \\ H_{m,n} &= 0 \;(m = 1, 2, \ldots, c;\, n > m). \end{aligned} \tag{3.12}$$

The recursion, then, is

$$\begin{aligned} F_{1,0} &= E \;\; ; \;\; F_{1,1} = \Theta_1 \; ; \\ F_{k,l} &= F_{k-1,l} + \Theta_k F_{k-1,l-1} \;\; (2 \le k \le K \;,\; 1 \le l \le k-1) \; ; \end{aligned} \tag{3.13}$$

$$\begin{aligned} H_{1,0} &= E \;\; ; \;\; H_{1,1} = \Phi_1 \; ; \\ H_{m,n} &= H_{m-1,n} + \Phi_m H_{m-1,n-1} \;\; (2 \le m \le c \;,\; 1 \le n \le m-1) \; . \end{aligned} \tag{3.14}$$

Transformation 1. *Modify simultaneously the balance equations for levels $j\,(L-2-c \ge j \ge c+K+1)$, by the transformation:*

$$< \mathbf{j} >^{(1)} \quad \longleftarrow < \mathbf{j} > + \sum_{l=1}^{K} (-1)^l < \mathbf{j} - \mathbf{l} > F_{K,l} \quad (c+K+1 \le j \le L-2-c);$$

$$< \mathbf{j} >^{(1)} \quad \longleftarrow < \mathbf{j} > \qquad\qquad (j > L-2-c \text{ or } j < c+K+1).$$

The balance equation for level j after Transformation 1 is $< \mathbf{j} >^{(\mathbf{1})}$.

Transformation 2. *Modify simultaneously the balance equations for levels $j\,(L-2-c \geq j \geq c+K+1)$, by the transformation:*

$$\begin{aligned} < \mathbf{j} >^{(\mathbf{2})} \quad &\longleftarrow < \mathbf{j} >^{(\mathbf{1})} + \sum_{n=1}^{c} (-1)^n < \mathbf{j}+\mathbf{n} >^{(\mathbf{1})} H_{c,n}; \\ & \qquad (c+K+1 \leq j \leq L-2-c) \\ < \mathbf{j} >^{(\mathbf{2})} \quad &\longleftarrow < \mathbf{j} >^{(\mathbf{1})} \qquad (j > L-2-c \text{ or } j < c+K+1). \end{aligned}$$

The balance equation for level j after Transformation 2 is denoted by $< \mathbf{j} >^{(\mathbf{2})}$.

Transformation 3. *Modify simultaneously the balance equations for levels $j\,(L-2-c \geq j \geq c+K+1)$, by the transformation:*

$$\begin{aligned} < \mathbf{j} >^{(\mathbf{3})} \quad &\longleftarrow < \mathbf{j} >^{(\mathbf{2})} - < \mathbf{j}+\mathbf{1} >^{(\mathbf{2})} \Delta \qquad (c+K+1 \leq j \leq L-2-c), \\ < \mathbf{j} >^{(\mathbf{3})} \quad &\longleftarrow < \mathbf{j} >^{(\mathbf{2})} \qquad\qquad (j > L-2-c \text{ or } j < c+K+1)\ . \end{aligned}$$

The balance equation for level j after Transformation 3 is denoted by $< \mathbf{j} >^{(\mathbf{3})}$.

3.3.2 The j-independent Balance Equations

Theorem 3.1. *With these above three transformations, the transformed balance equation, $< \mathbf{j} >^{(\mathbf{3})}$'s, for the rows $(c+K+1 \leq j \leq L-2-c)$, will be of the form:*

$$\begin{aligned} \mathbf{v}_{j-K} Q_0 + \mathbf{v}_{j-K+1} Q_1 + \ldots + \mathbf{v}_{j+c+1} Q_{K+c+1} = 0 \\ (j = L-2-c, L-1-c, \ldots, c+K+1), \end{aligned} \qquad (3.15)$$

where $Q_0, Q_1, \ldots, Q_{K+c+1}$ are $K+c+2$ number of j-independent matrices which can be derived algebraically from the system parameters.

Proof. Consider any row j where $c+K+1 \leq j \leq L-2-c$. With Transformation 1, we get

$$< \mathbf{j} >^{(\mathbf{1})} \quad \longleftarrow \quad < \mathbf{j} > + \sum_{l=1}^{K} (-1)^l < \mathbf{j}-\mathbf{l} > F_{K,l}. \qquad (3.16)$$

Applying Transformation 2 to the j^{th} row, from the above (3.16), we obtain

$$< \mathbf{j} >^{(\mathbf{2})} \quad \longleftarrow < \mathbf{j} >^{(\mathbf{1})} + \sum_{n=1}^{c} (-1)^n < \mathbf{j}+\mathbf{n} >^{(\mathbf{1})} H_{c,n}. \qquad (3.17)$$

Expanding the terms, equation (3.17) can be written as

$$\begin{aligned} <\mathbf{j}>^{(\mathbf{2})} \longleftarrow \quad & <\mathbf{j}> + \sum_{l=1}^{K} (-1)^l <\mathbf{j}-\mathbf{l}> F_{K,l} \\ & + \sum_{n=1}^{c} (-1)^n \left[<\mathbf{j}+\mathbf{n}> + \sum_{l=1}^{K} (-1)^l <\mathbf{j}-\mathbf{l}+\mathbf{n}> F_{K,l} \right] H_{c,n}. \end{aligned} \tag{3.18}$$

Applying Transformation 3 to the j^{th} row, and substituting from the above (3.18), for $<\mathbf{j}+\mathbf{1}>^{(\mathbf{2})}$

$$\begin{aligned} <\mathbf{j}>^{(\mathbf{3})} \longleftarrow \quad & <\mathbf{j}> + \sum_{l=1}^{K} (-1)^l <\mathbf{j}-\mathbf{l}> F_{K,l} \\ & + \sum_{n=1}^{c} (-1)^n \left[<\mathbf{j}+\mathbf{n}> + \sum_{l=1}^{K} (-1)^l <\mathbf{j}-\mathbf{l}+\mathbf{n}> F_{K,l} \right] H_{c,n} \\ & - \left[<\mathbf{j}+\mathbf{1}> + \sum_{l=1}^{K} (-1)^l <\mathbf{j}+\mathbf{1}-\mathbf{l}> F_{K,l} \right] \Delta \\ & - \sum_{n=1}^{c} (-1)^n \left[<\mathbf{j}+\mathbf{1}+\mathbf{n}> + \sum_{l=1}^{K} (-1)^l <\mathbf{j}+\mathbf{1}-\mathbf{l}+\mathbf{n}> F_{K,l} \right] H_{c,n}\Delta. \end{aligned} \tag{3.19}$$

Expanding and grouping the terms together, equation (3.19) can be written as

$$<\mathbf{j}>^{(\mathbf{3})} \longleftarrow \sum_{m=-c-1}^{K} <\mathbf{j}-\mathbf{m}> G_{K,c,m} \ , \tag{3.20}$$

where

$$\begin{aligned} G_{K,c,m} &= \sum_{\substack{l-n=m \\ l=-1,\ldots,K \\ n=0,\ldots,c}} (-1)^{l+n} [F_{K,l} H_{c,n} + F_{K,l+1} H_{c,n}\Delta] \\ &= \sum_{n=0}^{c} (-1)^{m+2n} [F_{K,m+n} + F_{K,m+n+1}\Delta] H_{c,n} \\ &= (-1)^m \sum_{n=0}^{c} [F_{K,m+n} + F_{K,m+n+1}\Delta] H_{c,n} \quad (m = -1-c, \ldots, K). \end{aligned} \tag{3.21}$$

The balance equations $< \mathbf{j}+\mathbf{c}+\mathbf{1} >,\ldots,< \mathbf{j} >,\ldots,\quad < \mathbf{j}-\mathbf{l} >,\ldots,$ $< \mathbf{j}-\mathbf{K} >$, respectively are given by

$$\sum_{s=1}^{j+c+1}\sum_{k=1}^{K}\mathbf{v}_{j+c+1-s}\Theta_k^{s-1}(E-\Theta_k)\Sigma_k+\mathbf{v}_{j+c+1}\left[Q-\Sigma-C_{j+c}-R\right]$$
$$+\sum_{s=1}^{L-j-c-1}\mathbf{v}_{j+c+1+s}C_{j+c+1+s,j+c+1}=0;$$
$$\vdots$$
$$\sum_{s=1}^{j}\sum_{k=1}^{K}\mathbf{v}_{j-s}\Theta_k^{s-1}(E-\Theta_k)\Sigma_k+\mathbf{v}_j\left[Q-\Sigma-C_j-R\right]$$
$$+\sum_{s=1}^{L-j}\mathbf{v}_{j+s}C_{j+s,j}=0\ ;$$
$$\vdots$$

$$\sum_{s=1}^{j-l}\sum_{k=1}^{K}\mathbf{v}_{j-l-s}\Theta_k^{s-1}(E-\Theta_k)\Sigma_k+\mathbf{v}_{j-l}\left[Q-\Sigma-C_{j-l}-R\right]$$
$$+\sum_{s=1}^{L-j+l}\mathbf{v}_{j-l+s}C_{j-l+s,j-l}=0\ ;$$
$$\vdots$$

$$\sum_{s=1}^{j-K}\sum_{k=1}^{K}\mathbf{v}_{j-K-s}\Theta_k^{s-1}(E-\Theta_k)\Sigma_k+\mathbf{v}_{j-K}\left[Q-\Sigma-C_{j-K}-R\right]$$
$$+\sum_{s=1}^{L-j+K}\mathbf{v}_{j-K+s}C_{j-K+s,j-K}=0\ .$$

Substituting or applying the above to (3.20), for the coefficients (Q_{K-m}) of $\mathbf{v}_{j-m}$ in $< \mathbf{j} >^{(\mathbf{3})}$, we get

$$Q_{K-m} = \sum_{l=-1-c}^{m-1}\left[\sum_{n=1}^{K}\Theta_n^{m-l-1}(E-\Theta_n)\Sigma_n\right]G_{K,c,l}+$$
$$\left[Q-\Sigma-C_{j-m}-R\right]G_{K,c,m}+\sum_{l=m+1}^{K}\left[C_{j-m,j-l}\right]G_{K,c,l}$$
$$(m=j-L,\ldots,-2,-1,0,\ldots,K,\ldots,j)\ .\quad \boxed{3.22}$$

Also, for $m = -1-c, 0, \ldots, K$, substituting $C_{j-m} = C$ and $C_{j-m,j-l} = C_{j-l+l-m,j-l} = \sum_{n=1}^{c} M_n(E-\Phi_n)\Phi_n^{l-m-1} + R(E-\Delta)\Delta^{l-m-1}$ in (3.22), we get

$$\begin{aligned} Q_{K-m} &= \sum_{l=-1-c}^{m-1} \left[\sum_{n=1}^{K} \Theta_n^{m-l-1}(E-\Theta_n)\Sigma_n\right] G_{K,c,l} + [Q-\Sigma-C-R]\, G_{K,c,m} \\ &\quad + \sum_{l=m+1}^{K} \left[\sum_{n=1}^{c} M_n(E-\Phi_n)\Phi_n^{l-m-1} + R(E-\Delta)\Delta^{l-m-1}\right] G_{K,c,l} \\ &\qquad\qquad (m = -1-c, \ldots, 0, \ldots, K). \end{aligned} \quad \boxed{3.23}$$

With these above three transformations, the transformed balance equation, $<\mathbf{j}>^{(3)}$'s, for the rows $(c+K+1 \le j \le L-2-c)$, will be of the form:

$$\begin{aligned} \mathbf{v}_{j-K}Q_0 + \mathbf{v}_{j-K+1}Q_1 + \ldots + \mathbf{v}_{j+c+1}Q_{K+c+1} = 0 \\ (j = L-2-c, L-1-c, \ldots, c+K+1), \end{aligned}$$

where $Q_0, Q_1, \ldots, Q_{K+c+1}$ are $K+c+2$ number of j-independent matrices. The other coefficients, *i.e.* those of $\mathbf{v}_{j-K-1}, \mathbf{v}_{j-K-2}, \ldots, \mathbf{v}_0$ and of $\mathbf{v}_{j+c+2}, \mathbf{v}_{j+c+3}, \ldots$, can be shown to be zero (see Section 3.3.3). □

Remark 1. *Due to (3.15), the HetSigma queue can be analyzed in the QBD-M framework. The values of the threshold parameters and the constants for a QBD-M process are illustrated in Table 3.1.*

y_1	K
y	$K+c+1$
T_1	$K+c+1$
T_2	$L-2-c$

Table 3.1: Value of the constants for equation (2.4)

3.3.3 Some Important Properties

After obtaining $F_{K,l}$'s and $H_{c,n}$'s thus, $G_{K,c,k}, (k = -1-c, \ldots, K)$ can be computed from (3.21). Then, using them directly in (3.23), the required $Q_l\,(l = 0, 1, \ldots, K+c+1)$ can be computed.

An alternate way of computing the $G_{K,c,l}$'s is by the following properties and recursion which are obtained from (3.13), (3.14) and (3.21):

$$\begin{aligned} G_{k,n,l} &= G_{k,n-1,l} - \Phi_n G_{k,n-1,l-1} \\ &\qquad (2 \le k \le K\ ,\ -1 \le l+c \le k+n \le k+c), \\ G_{k,c,l} &= G_{k-1,c,l} - \Theta_k G_{k-1,c,l-1}\ (2 \le k \le K\ ,\ -1 \le l \le k+c). \end{aligned} \quad \boxed{3.24}$$

From (3.21) and (3.24), we have

$$\begin{aligned} G_{k+1,c,l} &= G_{k,c,l} - \Theta_{k+1} G_{k,c,l-1} \ , \\ G_{k+1,c,l} &= 0 \, , \text{ if } l \leq -2-c \text{ or } l \geq k+2 \ . \end{aligned} \tag{3.25}$$

Since, K itself is arbitrary in this section, let the Q_l's be designated differently to take that into account. Let $Q_{k-m}^{(k,h)} (m = j-L, j-L+1, \ldots, j)$ be the Q_l's of $< \mathbf{j} >^{(\mathbf{3})}$ when only the first k customer arrival streams and the first h servers are present and others are absent.

Theorem 3.2. *Referring to equation (3.22) for the row $j \, (c+K+1 \leq j \leq L-2-c)$, for all K, $Q_{K-m}^{(K,c)} = 0 \, (j-L \leq m \leq -2-c)$.*

Proof. Assume the proposition is *true* for any K=k. Hence, from (3.22), for the range $j-L \leq m \leq -2-c$, we have

$$\begin{aligned} Q_{k-m}^{(k,c)} &= \sum_{l=m+1}^{k} C_{j-m,j-l} G_{k,c,l} = \sum_{l=-1-c}^{k} C_{j-m,j-l} G_{k,c,l} \\ &\quad \text{(since } G_{k,c,l} = 0 \text{ if } l \leq -2-c) \\ &= 0 \quad (j-L \leq m \leq -2-c) \ . \end{aligned} \tag{3.26}$$

For $K = k+1$, from (3.22), we get

$$\begin{aligned} Q_{k+1-m}^{(k+1,c)} &= \sum_{l=m+1}^{k+1} C_{j-m,j-l} G_{k+c+1,l} \\ &= \sum_{l=m+1}^{k+1} C_{j-m,j-l} G_{k,c,l} - \Theta_{k+1} \sum_{l=m+1}^{k+1} C_{j-m,j-l} G_{k,c,l-1} \quad \text{(substituting (3.25))} \\ &= 0 - \Theta_{k+1} \sum_{l=m+1}^{k+1} C_{j-m,j-l} G_{k,c,l-1} \quad \text{(since } G_{k,c,k+1} = 0 \text{ \& using (3.26))} \\ &= -\Theta_{k+1} \sum_{l-1=m}^{k} C_{j-(m-1)-1,j-(l-1)-1} G_{k,c,(l-1)} \quad \text{(rearranging)} \\ &= -\Theta_{k+1} \sum_{l-1=(m-1)+1}^{k} C_{j-(m-1),j-(l-1)} G_{k,c,(l-1)} \\ &\qquad \text{(since } G_{k,c,m} = 0 \text{ \& } C_{j-(m-1)-1,j-(l-1)-1} = C_{j-(m-1),j-(l-1)}) \\ &= 0 \quad \text{(comparing with (3.26))} \ . \end{aligned}$$

Hence the proposition is true for $K = k+1$. Also, the proposition has been proved for $K = 2$ in Appendix A.1. Hence, the theorem is true for all values of $K \geq 2$. □

Theorem 3.3. *Referring to equation (3.22) for the row $j\,(K+1 \le j \le L-2-c)$, for all K, $Q^{(K,c)}_{K-m} = 0\,(K+1 \le m \le j)$.*

Proof. Assume the proposition is *true* for any $K = k$. Hence, from (3.22), we have

$$\begin{aligned} Q^{(k,c)}_{k-m} &= \sum_{l=-1-c}^{m-1} \left[\sum_{n=1}^{k} \Theta_n^{m-l-1}(E-\Theta_n)\Sigma_n\right] G_{k,c,l} \;\; (k+1 \le m \le j) \\ &= \sum_{l=-1-c}^{k} \left[\sum_{n=1}^{k} \Theta_n^{m-l-1}(E-\Theta_n)\Sigma_n\right] G_{k,c,l} = 0 \\ &\qquad\qquad (\text{since } G_{k,c,l} = 0 \text{ for } l > k)\ . \end{aligned} \tag{3.27}$$

Then, for $K = k+1$, writing down the expression for $Q^{(k+1)}_{k+1-m}$ from (3.22), substituting (3.25) as before and expanding the terms, we get

$$\begin{aligned} Q^{(k+1,c)}_{k+1-m} = &\sum_{l=-1-c}^{k+1} \left[\sum_{n=1}^{k} \Theta_n^{m-l-1}(E-\Theta_n)\Sigma_n + \Theta_{k+1}^{m-l-1}(E-\Theta_{k+1})\Sigma_{k+1}\right] G_{k,c,l} \\ -\Theta_{k+1} &\sum_{l=-1-c}^{k+1} \left[\sum_{n=1}^{k} \Theta_n^{m-l-1}(E-\Theta_n)\Sigma_n + \Theta_{k+1}^{m-l-1}(E-\Theta_{k+1})\Sigma_{k+1}\right] G_{k,c,l-1}. \end{aligned}$$

Using (3.27) and $G_{k,c,k+1} = 0$ in the above, it simplifies to

$$\begin{aligned} Q^{(k+1,c)}_{k+1-m} &= \sum_{l=-1-c}^{k+1} \Theta_{k+1}^{m-l-1}(E-\Theta_{k+1})\Sigma_{k+1}G_{k,c,l} \\ &\quad -\Theta_{k+1} \sum_{l-1=-2-c}^{k} \left[\sum_{n=1}^{k} \Theta_n^{(m-1)-(l-1)-1}(E-\Theta_n)\Sigma_n\right] G_{k,c,l-1} \\ &\quad -\Theta_{k+1} \sum_{l-1=-2-c}^{k} \Theta_{k+1}^{m-l-1}(E-\Theta_{k+1})\Sigma_{k+1}G_{k,c,l-1}\ . \end{aligned}$$

However the middle term of the R.H.S. above would be 0 for $k+1 \le m-1 \le j$ by comparing with (3.27) and by using $G_{k,c,-2-c} = 0$. Hence, we obtain, for $k+1 \le m-1 \le j$ and hence for $k+2 \le m \le j$,

$$\begin{aligned} Q^{(k+1,c)}_{k+1-m} &= \sum_{l=-1-c}^{k+1} \Theta_{k+1}^{m-l-1}(E-\Theta_{k+1})\Sigma_{k+1}G_{k,c,l} \\ &\quad - \sum_{l-1=-2-c}^{k} \Theta_{k+1}^{m-(l-1)-1}(E-\Theta_{k+1})\Sigma_{k+1}G_{k,c,l-1} \\ &= 0 \quad (m = k+2, k+3, \ldots, j) \;\; (\text{since } G_{k,c,k+1} = 0)\ . \end{aligned} \tag{3.28}$$

Hence, the proposition is true for $K = k + 1$. The proposition has already been proved for $K = 1$ in [27] and for $K = 2$ in Appendix A.1. Hence, the theorem is true. □

Theorem 3.4. *Referring to equation (3.22) for the row* $j\,(K+1 \leq j \leq L-2-c)$, *for all* K, $Q^{(K,c)}_{K-m}\,(m = -1-c, 0, \ldots, K)$ *are* j-independent.

Proof. $Q^{(K,c)}_{K-m}$ for $m = -1-c, 0, \ldots, K$ are separately derived in (3.23). From the R.H.S. of (3.23), it is clear that $Q^{(K)}_{K-m}\,(m = -1-c, \ldots, K)$ are j- independent. □

Theorem 3.5. *Referring to equation (3.22) for the row* $j\,(K+1 \leq j \leq L-2-c)$, *for all* K, $\sum_{m=-1-c}^{K} Q^{(K,c)}_{K-m}$ *is* singular.

Proof. Assume the theorem is true for some $K = k$. The expressions for $Q^{(k,c)}_{k-m}$ and $Q^{(k+1,c)}_{k+1-m}$ are

$$\begin{aligned}
Q^{(k,c)}_{k-m} &= \sum_{l=-1-c}^{m-1} \left[\sum_{n=1}^{k} \Theta_n^{m-l-1}(E-\Theta_n)\Sigma_n\right] G_{k,c,l} \\
&\quad + \left[Q - \sum_{n=1}^{k}\Sigma_n - C - R\right] G_{k,c,m} \\
&\quad + \sum_{l=m+1}^{k} \left[C(E-\Phi)\Phi^{l-m-1} + R(E-\Delta)\Delta^{l-m-1}\right] G_{k,c,l} \\
&\qquad (m = -1-c, 0, \ldots, k), \\
Q^{(k+1)}_{k+1-m} &= \sum_{l=-1-c}^{m-1} \left[\sum_{n=1}^{k+1} \Theta_n^{m-l-1}(E-\Theta_n)\Sigma_n\right] G_{k+1,c,l} \\
&\quad + \left[Q - \sum_{n=1}^{k+1}\Sigma_n - C - R\right] G_{k+1,c,m} \\
&\quad + \sum_{l=m+1}^{k+1} \left[C(E-\Phi)\Phi^{l-m-1} + R(E-\Delta)\Delta^{l-m-1}\right] G_{k+1,c,l} \\
&\qquad (m = -1-c, 0, \ldots, k+1).
\end{aligned}$$

Substituting $G_{k+1,c,l} = G_{k,c,l} - \Theta_{k+1} G_{k,c,l-1}$ in the latter, we get

$$\begin{aligned} Q_{k+1-m}^{(k+1,c)} &= \sum_{l=-1-c}^{m-1} \left[\sum_{n=1}^{k+1} \Theta_n^{m-l-1} (E - \Theta_n) \Sigma_n \right] (G_{k,c,l} - \Theta_{k+1} G_{k,c,l-1}) \\ &\quad + \left[Q - \sum_{n=1}^{k+1} \Sigma_n - C - R \right] (G_{k,c,m} - \Theta_{k+1} G_{k,c,m-1}) \\ &\quad + \sum_{l=m+1}^{k+1} \left[C(E - \Phi)\Phi^{l-m-1} + R(E - \Delta)\Delta^{l-m-1} \right] (G_{k,c,l} - \Theta_{k+1} G_{k,c,l-1}). \end{aligned}$$

After expanding, rearranging and regrouping the terms, we get

$$\begin{aligned} Q_{k+1-m}^{(k+1,c)} &= Q_{k-m}^{(k)} - Q_{k-(m-1)}^{(k)} \Theta_{k+1} \\ &\quad + \sum_{l=-1-c}^{m-1} (E - \Theta_{k+1}) \Sigma_{k+1} \left[\Theta_{k+1}^{m-l-1} \right] G_{k+1,c,l} - \Sigma_{k+1} G_{k+1,c,m} \\ &\quad (m = -1-c, \ldots, k+1) \\ &= Q_{k-m}^{(k)} - Q_{k-(m-1)}^{(k)} \Theta_{k+1} \\ &\quad + (E - \Theta_{k+1}) \Sigma_{k+1} \sum_{l=-1-c}^{m-1} \left[\Theta_{k+1}^{m-l-1} G_{k+1,c,l} \right] - \Sigma_{k+1} G_{k+1,c,m} \\ &\quad (m = -1-c, \ldots, k+1). \end{aligned}$$

Then, summing up the terms from $m = -1-c, \ldots, k+1$, we get

$$\begin{aligned} \sum_{m=-1-c}^{k+1} Q_{k+1-m}^{(k+1,c)} &= \left[\sum_{m=-1-c}^{k} Q_{k-m}^{(k)} \right] [E - \Theta_{k+1}] \\ &\quad + \Sigma_{k+1} \left[\sum_{l=-1-c}^{m-1} \Theta_{k+1}^{m-l-1} G_{k+1,c,l} - \sum_{l=-2-c}^{m} \Theta_{k+1}^{m-l} G_{k+1,c,l} \right]. \end{aligned}$$

By substituting $G_{k+1,c,l} = G_{k,c,l} - \Theta_{k+1} G_{k,c,l-1}$ in the r.h.s. of the above equation and expanding, the terms other than the first term cancel off, leaving,

$$\sum_{m=-1-c}^{k+1} Q_{k+1-m}^{(k+1,c)} = \left[\sum_{m=-1-c}^{k} Q_{k-m}^{(k)} \right] [E - \Theta_{k+1}] . \qquad (3.29)$$

The above r.h.s. expression is clearly a singular matrix if the theorem is true for $K = k$, that is, $\left[\sum_{m=-1-c}^{k} Q_{k-m}^{(k)} \right]$ is singular, since $[E - \Theta_{k+1}]$ is a diagonal matrix. The theorem is easily proved for $K = 1$ and for $K = 2$ (Appendix A.1). Hence the theorem is true for any K. □

3.4 Summary

The HetSigma queue in the Markovian framework is proposed in order to model nodes in modern telecommunication networks [33]. An exact and computationally efficient solution of this new queue for steady state probabilities and performance measures is developed and presented. The fundamental properties which serve as the background for the efficient computational algorithm to calculate the steady state probabilities of the HetSigma queue are presented.

The HetSigma queue and its variants have been successfully applied to carry out the performance analysis of various problems in communication networks. The proposed model does provide a useful tool for the performance analysis of many problems of the emerging telecommunication systems and networks. The queuing model and its variants were successfully used to model Optical Burst/Packet (OBS) Switching networks [34], MPLS networks [55] and another variant to successfully compute web server performance [54]. The *HetSigma* model has been successful to model the wireless networks [33, 51]. Due to the limited space, we only present one application of the HetSigma queue in Chapter 4.

4

Modeling Apache Web Server Software[‖]

4.1 Introduction

Considering the service infrastructure of the current Internet, Web servers play a dominant role. It is their main task to receive and process requests of clients demanding specific objects from the servers and to return them by related responses. The associated request and response messages are transported by the Hypertext Transfer Protocol (HTTP) or Hypertext Transfer Protocol Secure (HTTPS) using TCP connections between the clients and the server. The resulting performance of the service delivery crucially depends on the proper and efficient processing of these tasks.
Regarding the software design of HTTP servers the concurrency of connection requests poses a critical issue. To tackle it, two orthogonal components have been implemented in the Web server software architecture on the host machine: the processing model and the dynamic size of a resource pool of instantiated parallel processes serving the requests (cf. [7]). Considering the processing model of the software architecture of an HTTP server, there are several choices, viz. a process based, a thread based, or a hybrid model which is a combination of the former two (cf. [93]). The behavior of the pool size is characterized by a fixed or varying number of concurrent HTTP processes (or threads) to serve incoming connection requests and, thus, determines the performance of the server. When a dynamic behavior of the pool size is incorporated into the server software, the number of processes (or threads) can vary depending on the load of the offered connection requests.
Nowadays, the Unix version of the Apache Web server constitutes one of the most frequently used software solutions of the Internet. In Apache 2.0 the introduction of the multi-processing module (MPM) combined with the dynamic pool size efficiently supports a stable and scalable operation. Moreover, it is currently considered as an important topic regarding the

[‖]Tien Van Do, U. R. Krieger, and R. Chakka. Performance modeling of an Apache Web Server with a Dynamic Pool of Service Processes. *Telecommunication Systems*, 39(2):117–129, 2008.

optimal resource allocation (cf. [7], [89]).
Therefore, we need to understand both the performance of different software solutions of a Web server, and the dynamic behavior of the pool size subject to various workload conditions. Slothouber [112] has proposed an open queueing network to model a Web server and predicted the response time versus the load. Dilley et al. [42, 43] have used layered queueing models regarding the performance evaluation of a Web server in the Internet and Intranet. Squillante et al. [113, 114] developed a Web traffic model using the access logs of the Web site of the Winter Olympic Games in Nagano, Japan, 1998. To analyze the tail behavior of the request latency, the authors fed these traffic processes into a Web server modelled by a $G/G/1$ queue. Reeser et al. [105] have presented an analytic queueing model of Web servers in a distributed environment. They have argued that a Poisson assumption regarding the process of HTTP (object) requests is reasonable. They have also shown that the performance predictions of their analytic model match well with results obtained by simulations. In [20] a queueing model $M/G/1/K^*$PS has been proposed to predict the performance metrics of a Web server in terms of the average response time, throughput and blocking probability. Recent work of the same authors ([6]) was carried out assuming that the arrival stream of HTTP requests forms a Markov-Modulated Poisson Process (MMPP).
However, all these previous investigations of a Web server have not studied in more detail the internal software architecture and the dynamic behavior of the number of available service processes and their impact on major performance indices applying analytic means.
Menasce [93] has been one of the first researchers who considered the software architecture of a Web server. In his performance evaluation approach a queueing network (QN) models the physical resources of the Web server and the software architecture of Apache is represented by a Markov chain. Liu et al. [89] have recognized the importance of the Apache directive *MaxClients* as a tuning parameter and proposed an online optimization to determine an optimal operating point. However, so far all presented models have not been able to capture explicitly the dynamic behavior of the pool size that handles the concurrency of requests. Moreover, they cannot cope with simultaneous arrivals of multiple requests arising from the client population.
In this Chapter we consider the non-threaded multi-processing module (MPM) *Prefork* of Apache's UNIX architecture. For the first time, we propose a mathematically tractable analytic queueing model to predict the performance of an Apache server with its load-dependent dynamic pool size. It is derived from certain Markovian assumptions concerning the number of active service processes and applies a decomposition approach to the client population, the Web server and the TCP transport system. In particular, we suppose that TCP connection requests demanding the support of an HTTP service process follow a Markov-Modulated Compound Poisson Process (MMCPP). We use the MMCPP because it can cope with

the fluctuations of the arrival rate, the autocorrelation of inter-arrival times of requests and also with simultaneous arrivals of multiple requests. As a consequence the well-known WAGON traffic model of [90] is a special case of our more general workload model.
In addition to the development of a tractable analytic model we have also measured the performance of a real Apache Web server where the number of service processes does not follow a Markovian property. To generate the workload as a generalization of the WAGON traffic model (cf. [89]), either the benchmarking program `ab` (cf. [2]) or `httperf` (cf. [98]) can be used. Comparing the numerical results of the analytic model with those obtained by the actual measurements, we observe a rather good coincidence of mean values and an acceptable accuracy of our predictions regarding control purposes. Moreover, the results are used to identify the impact of major control parameters of an Apache configuration on relevant performance measures.
Since we are only dealing with non-threaded multi-processing, modeling the thread-based Apache multi-processing module `winnt`[1] used by Microsoft Windows operating systems is out of the scope of our study. However, by appropriate modifications the presented methodology can be used to analyze this Apache MPM `winnt` as well.
The rest of the Chapter is organized as follows. In Section 4.2 we provide an overview of the operation of the Apache MPM *Prefork* with dynamic pool size. In Section 4.3 the proposed queueing model and its analysis are introduced. Section 4.4 presents some performance results and the validation of the proposed model by measurements. Finally, some conclusions are drawn in Section 4.5.

4.2 Multi-processing Operation of an Apache Web Server with Dynamic Pool Size

In this section we sketch the operation of an Apache 2.0 Web server that implements the non-threaded multi-processing module (MPM) *Prefork*. At the beginning of the operation, the Apache 2.0 running on top of UNIX starts a single master control process. The latter is responsible for launching child processes to handle the incoming or waiting HTTP service requests (see Fig. 4.1). The Web server listens for requests at the well-known TCP port number 80. Web browsers, i.e. the clients, are normally used to communicate with the Web server. When a client opens or clicks on the first URL during a specific session of the Web browser, the set-up of a TCP connection is initiated and one of the following three alternatives may happen (see Fig. 4.1):

[1] `winnt` has a single control process which gives birth to a single child process. The latter creates in turn threads to handle HTTP requests.

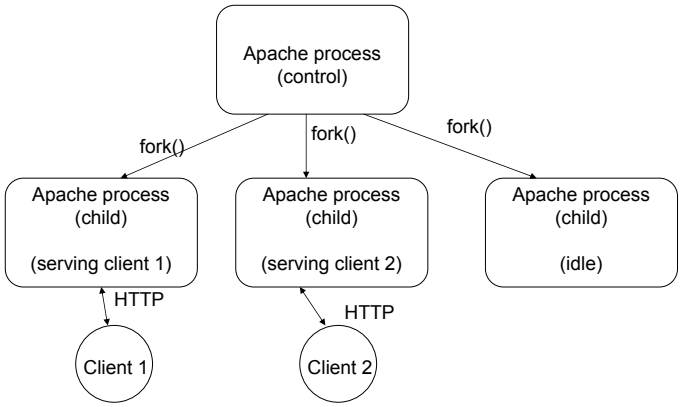


Figure 4.1: MPM prefork operation of an Apache Web server with a Unix operating system

1. One idle child process of the Apache HTTP server is allocated to handle the incoming TCP connection and the associated HTTP requests conveyed by that connection.

2. The TCP request waits in the software queue for an idle process and will get service either after the MPM has successfully launched an idle child process or when a child process has completed the service of another request.

3. The TCP connection and associated HTTP request is blocked if the Listen Queue[2] of the server system is full. The client receives a message that the Web server is not available. In our model the request is considered to be lost.

If a free child process is allocated, the related service process locates the Web page and its referenced resources as requested objects. It normally consists of a single HTML file, an HTML file with several inline images or other items. An object may be a static HTML file stored at the Web server or may be dynamically generated from databases by a special script engine. An obtained object is encapsulated as HTTP response message and handled by the TCP/IP stack. The latter encapsulates and segments the message into TCP/IP packets, and sends the resulting packet stream through the already opened TCP connection from the HTTP server to the browser. The service time of an HTTP request, i.e. the interval between the arrival time of the request at the HTTP server and the completion instant of the initiated processing and its resource allocations, i.e. the tear-down of the TCP connection and the release of the related HTTP service process, constitutes

[2]Note that the terms "software queue" and "Listen Queue" are interchangeable throughout the Chapter.

a basic performance index of the system. Obviously, the service time of a specific request and, hence, the overall response time of the HTTP server depend in a very complex manner on a number of different items, such as the physical resources of the operating machine with regard to CPU power and disks' operation, the number of other concurrently served requests competing for these physical resources of the Web server as well as the network conditions influencing the TCP flow-control algorithm.

In the Unix version of Apache 2.0, the number of available busy and idle service processes varies dynamically in terms of the load of TCP session requests. It creates a dynamic resource pool of HTTP service processes whose size is controlled by the following three main control parameters of the server software, called *directives* (cf. [7]):

- The *MaxClients* directive limits the maximal number of operating busy and idle HTTP service processes, therefore it sets a threshold N for the maximal number of simultaneously served requests. Any further connection attempt exceeding the MaxClient limit will normally be queued in the Listen Queue based on the *ListenBacklog* directive. Once a child process is released from serving a previous connection request, a new connection from the queue will be served in FCFS order.

- The *MaxSpareServers* directive determines the desired maximum number h_{max} of idle child service processes which are not handling a connection request. If the number of idle child processes exceeds *MaxSpareServers* $h_{\max}$, then the control process will kill idle processes at a predetermined rate ϵ.

- The *MinSpareServers* directive determines the desired minimum number $h_{\min}$ of idle child processes. If the number of idle child processes drops below h_{min} and the number of busy and idle processes in the system is less than N, then the parent process creates new child processes at a predetermined rate η.
  However, when the number of processes reaches the limit MaxClients (N), e.g. due to heavy traffic, and the number of idle process is smaller than h_{min}, no creation of a new child process is allowed.

In summary, it is the rationale of this design to serve the incoming HTTP requests in an efficient way saving the spare resources of the operating system. Considering the number of idle child processes in the host machine of the Web server, the MPM Prefork module tries as effectively as possible to keep this number at any given time less than or equal to $h_{\max}$ and greater than or equal to $h_{\min}$.

4.3 Modeling and Analysis of an Apache Web Server with the MPM Strategy Prefork

4.3.1 Hierarchical Workload Modeling of a Web Server

Following the approach of [39] the workload of a Web server can be described by a hierarchically layered model. It is generated by traffic streams of a client population and depicted in Figure 4.2. The associated processes that characterize the traffic patterns arising from the activity of a Web user and their corresponding time scales are related to the following levels:

- *Level of Web requests by users*: The behavior of a Web user can be characterized as an ON-OFF process. During the HTTP ON state, called activity period, the user is downloading a specific Web page and waiting for its presentation. The OFF state, called think time, simply describes the following silence period. Here the user is not sending any requests, only viewing and perhaps reading the downloaded document. During the OFF state no user traffic is generated at the other traffic layers beneath.

- *Level of Web pages with their corresponding objects*: A requested Web page with its objects denotes an HTML file and the referenced inline objects, e.g. images, included in such an HTML document.

- *Level of TCP sessions*: When the user clicks on a URL reference or enters a URL into the command line of the Web browser, the client first opens a TCP connection to the specified Web server. After the TCP connection is established, the client sends a HTTP request in the HTTP format typically by one TCP segment, that is encapsulated in one IP packet, over the already opened TCP connection to the HTTP server. The request normally specifies which HTTP version the client is using. If it is HTTP/0.9 or HTTP/1.0, the server will automatically work in the transitory connection mode, will only send one reply and then close the connection. If it is HTTP/1.1, it is assumed that a persistent connection is desired. This means that multiple HTTP requests can be sent through the opened TCP connection.

- *Packet level*: The HTTP object request and response messages for HTML files or object files and their referenced inline images etc. are encapsulated into TCP packets and segmented into a stream of IP packets. The resulting TCP/IP packet stream is transmitted along a path between the server and the requesting client inside the Internet and the transport is governed by the TCP congestion-control mechanism.

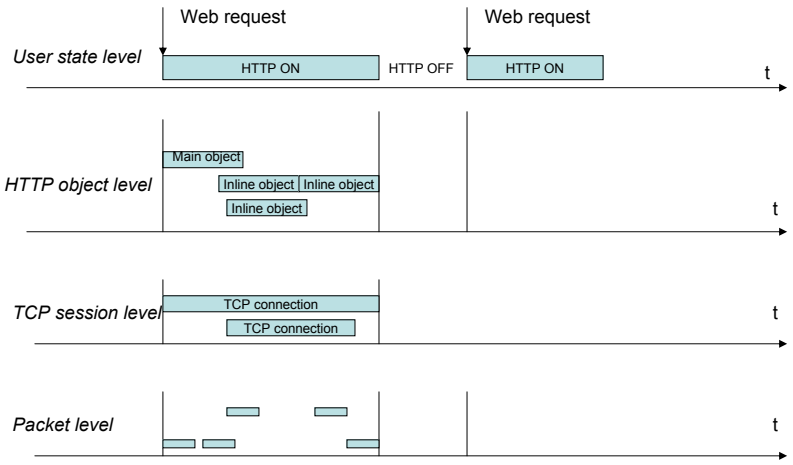


Figure 4.2: Hierarchical workload modeling of Web traffic

In this Chapter we primarily focus on the workload modeling at the TCP session level. Liu et al. [89] have proposed a related stochastic model (called WAGON) of the generated HTTP traffic between a specific user and a Web server at this time scale. The WAGON model can be described by a marked point process where the arrival times correspond to the start times of sessions followed by a cluster of further object requests and think times (see Figure 4.3). We follow this line of reasoning and subsequently extend the corresponding approach.

4.3.2 Resource Contention During Web Server Operation

Considering a scenario of a single Web server and multiple clients, contention about its resources may happen at the following two traffic layers (see Figure 4.4):

- *TCP/IP flow layer*: Flows of IP packets encapsulating TCP segments that are carrying HTTP requests of clients and responses of the Web server compete with other traffic streams in the switching nodes and on the links of the underlying transport network.

- *TCP session layer*:
  - *Child process level of the Web server*: At the child process level of the Web server (called software level), HTTP requests compete for such service processes controlled by the related software components of the server. An arriving request may wait in a Listen

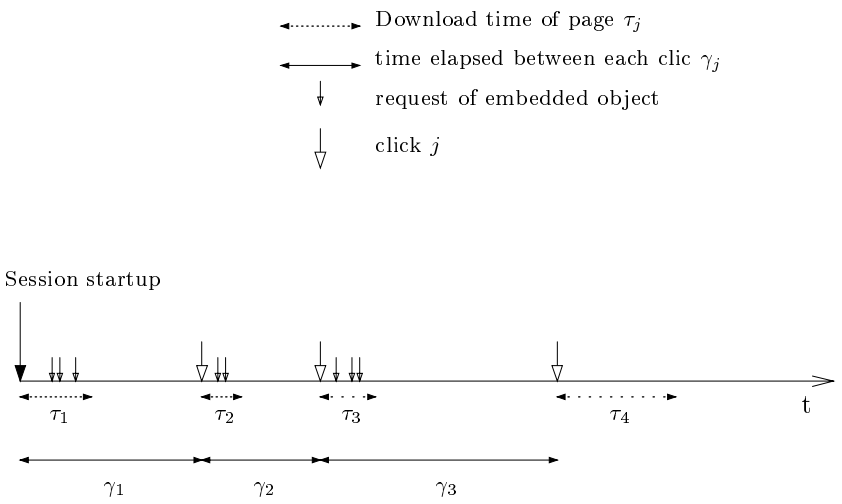


Figure 4.3: The WAGON traffic model

Queue for an available service process that can handle the request regarding a demanded page and its objects.

– *Level of physical resources*: At the kernel level of the host machine running the Web server (called hardware level), processes may wait to use any of the server's physical resources, i.e. CPU power and disk processing of the underlying machine etc. (see Fig. 4.4).

All these contentions have an impact on the performance experienced by the Web clients of the users. As a consequence of these mutual interactions of the corresponding queues at two different time scales, a queueing model which takes into consideration all these factors, such as the arrivals of requests, the transmission of every TCP/IP packet governed by TCP congestion control, contentions both in the transport network and about the physical resources of the server etc., becomes mathematically intractable. Therefore, we will study the Web server in isolation and try to model its offered workload in an appropriate manner (see Fig. 4.4).

4.3.3 The Markovian Performance Model

Considering the operation of a Web server at the page level in more detail, we can make the following observations:

- A TCP connection (i.e. a session) may be initiated when a client starts to download a new page and its objects from the Web server if a previously opened TCP connection has been closed.

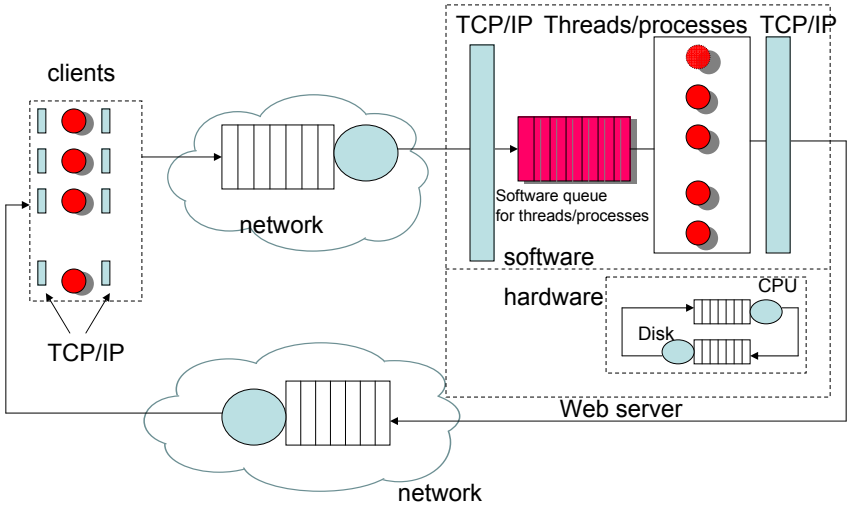


Figure 4.4: Resource contention and queueing in the operation of a Web server

- Multiple HTTP object requests can be transmitted over a single TCP connection between a client and the Web server.
- One new TCP connection requires a HTTP service process either from the group of idle processes or it is a newly created child process. It serves the new HTTP request to download a page and subsequent HTTP requests to further transfer its embedded objects in the case of a persistent connection.

To predict the performance of the dynamic pool size of HTTP service processes in the UNIX operating system of an Apache Web server, we have extended a convenient queueing model fed by these processing requests (cf. [93], [105]). As illustrated by Figure 4.4, it takes into consideration the impact and interaction of other components such as the client population, the transport protocol stack of TCP, as well as the software and resource contention level of a Web server. In the queueing system of the Web server at the child process level we primarily consider the arrivals of initial HTTP requests by TCP sessions which require the service of an idle or a new service process. Here all HTTP requests compete for service processes controlled by the software of the Web server. A request may wait in the software queue, i.e. Listen Queue, until a service process becomes available to handle it. At the physical level these processes may wait until they can use any of the server's physical resources, i.e. CPUs and disks. Thus, the associated service time of a specific HTTP request is strongly influenced by the interaction of other HTTP requests.
To generate a mathematically tractable model, we propose a system

decomposition of the queueing network describing the Web server (see Fig. 4.4). Our approach is based on the following Markovian workload model.

4.3.3.1 Arrival Process of HTTP Requests

Considering multiple simultaneous arrivals of HTTP requests to the Web server arising from the client population, we use the generalization of the WAGON traffic model at the TCP session level. Therefore, we assume that the arrival process of TCP connection requests demanding the service of child processes follows a Markov-Modulated Compound Poisson Process (MMCPP). Its inter-arrival times are governed by a Markov-Modulated Generalized Exponential (MMGE) distribution.

The arrival process is modulated by an irreducible continuous-time Markov chain X with M states called phases of modulation. Let Q_X denote its generator matrix. Then an off-diagonal element $Q_X(i,k) = qx_{i,k}, i \neq k$ describes the instantaneous transition rate from phase i to phase k, and the ith diagonal element is given by $Q_X(i,i) = -qx_i = -\sum_{l=1}^{i-1} qx_{i,l} - \sum_{l=i+1}^{M} qx_{i,l}$.

Let Ω_X denote the off-diagonal matrix of the transition rates of process X defined by $\Omega_X(i,k) = Q_X(i,k)$, $i \neq k$, $\Omega_X(i,i) = 0$, $1 \leq i \leq M$. Let $I_1(t)$, $1 \leq I_1(t) \leq M$, represent the phase of the modulating process X at any time t.

The arrivals in each modulating phase follow a Compound Poisson Process (CPP) process (cf. [77]). Strictly speaking, during the modulating phase i, $1 \leq i \leq M$, the probability distribution function of the inter-arrival times τ_i is governed by a Generalized Exponential (GE) distribution $\mathbb{P}(\tau_i = 0) = \theta_i$ and $\mathbb{P}(0 < \tau_i \leq t) = (1 - \theta_i)(1 - e^{-\sigma_i t})$ with the associated parameters (σ_i, θ_i). Note that the GE distribution is the least biased distribution with two parameters which can approximate other distributions by matching the first two moments (cf. [77]).

Thus, the point process of session arrivals during a given modulating phase can be considered as a batch Poisson process with a geometric batch size distribution, i.e. during phase i size $s \geq 1$ happens with probability $(1 - \theta_i)\theta_i^{s-1}$.

4.3.3.2 Service Time of HTTP Requests

The service time of HTTP requests is a result of the complex competition for physical resources of the Web server, the interactions with all HTTP requests concurrently served by other service processes and the TCP flow-control algorithm. Thus, it strongly depends on the contention for physical resources of CPUs and disks in the host machine. To approximate the latter a closed

queueing network (CQN) is used. It has been validated by [94] in their experimental test-bed that such a CQN model is accurate enough to estimate the response time of a HTTP server.
Let $T_p(k)$ denote the throughput of the Web server when there are k requests and p processes in the system (note that $T_p(k) = T_p(p)$, if $k \geq p$, i.e. there is no idle process in the system). Following the approach of [93] and describing the physical resources of the Web server by a CQN (see Fig. 4.4), we approximate the service time by the mean sojourn time of a request in this CQN. In this manner the associated conditional service rates of an equivalent server are determined by $T_p(k)$.

4.3.3.3 Modeling the Varying Number of the HTTP Service Processes

We introduce an integer-valued random variable $I_2(t)$ to describe the number of busy and idle HTTP service processes at time t (- it does not include the master process in the case of Apache 2.0). $I_2(t)$ $(1 \leq I_2(t) \leq N)$ varies due to the creation of a new service process by forking of the Apache master process or due to a process cancellation by killing an existing idle child process. When the number j of concurrent connection requests is larger than $I_2(t)$, the excess of requests beyond $I_2(t)$ resides in a software queue. We assume that at any time the number j of connection requests in the Web server is bounded by the fixed value L. It means that the software queue is resized accordingly to keep the maximum number of connection requests in the system equal to L. Note that this assumption is quite reasonable for the software of the Web server because this behavior guarantees the fairness for those requests arriving after different periods of low and high traffic load.
We model the changing number $I_2(t) = i_2$ of child processes that are available in the Web server to process the HTTP requests by $N \times N$ rate matrices U_j, $j = 0, 1, \ldots, L$. They represent the applied server strategy to regulate this number. To capture correctly the dynamics of this mechanism, that handles the idle service processes in the Apache MPM Prefork, the U_j's are constructed as follows:

- if $i_2 - j < h_{\min}, \quad \forall \quad j = 0, \ldots, L, i_2 = 1, \ldots, N-1$ then $U_j(i_2, i_2 + 1) = \eta$,
- if $i_2 - j > h_{\max}, \quad \forall \quad j = 0, \ldots, L, i_2 = 2, \ldots, N$ then $U_j(i_2, i_2 - 1) = \epsilon$,
- otherwise $U_j(i_2, k) = 0$.

Thus, the number of idle processes is kept smaller than or equal to $h_{\max} \geq 1$ and larger than or equal to $h_{\min}$. In state $i_2 = N$ a creation and in state $i_2 = 1$ a cancellation of a service process is not possible.

Here, we assume that the times between successive creations or terminations of HTTP service processes are generalized exponentially distributed and, hence, $I_2(t)$ is a Markov chain. As a consequence we propose a mathematically tractable performance model of the software architecture of the Web server. We will show in the subsequent Section 4.4 that despite this approximation our computationally efficient model can successfully and rather accurately predict the performance of the real operation of an Apache server. There the times between successive creations/terminations of service processes are neither exponentially distributed nor independent, but correlated, generally distributed entities.

4.3.4 Analysis and Solution of the Multi-server Model

The Apache Web server with its dynamic pool size is modelled by a continuous-time Markov chain $Z = \{[I_1(t), I_2(t), J(t)]; t \geq 0\}$ with state space $\{1, \ldots, M\} \times \{1, \ldots, N\} \times \{0, \ldots, L\}$. Here $J(t)$, $0 \leq J(t) \leq L$, represents the number of connection requests that are actually served in the HTTP server or wait to get service at time t.

Then the two state variables $(I_1(t), I_2(t))$ are lexicographically sorted to form a single variable $I(t)$ as illustrated in Table 4.1.

Table 4.1: Ordering of the phase variable $I(t)$

I	(I_1, I_2)
1	$(1,1)$
2	$(2,1)$
$\vdots$	$\vdots$
M	$(M,1)$
$M+1$	$(1,2)$
$\vdots$	$\vdots$
$2M$	$(M,2)$
$\vdots$	$\vdots$
$MN-M+1$	$(1,N)$
$\vdots$	$\vdots$
MN	(M,N)

The index transformation is determined by the following equations:

$$\begin{aligned} I(t) &= I_1(t) + (I_2(t) - 1)M, \\ I_2(t) &= f_2(I(t)) = \left\lfloor \frac{I(t)-1}{M} \right\rfloor + 1, \\ I_1(t) &= f_1(I(t)) = (I(t) - 1) \ mod \ M + 1. \end{aligned}$$

Then we get the equivalent Markov chain $Y = \{[I(t), J(t)]; t \geq 0\}$ with a generator matrix Q_Y to describe the operation of the non-threaded Web server with the MPM module Prefork. It evolves on a finite rectangular strip and belongs to the class of level-dependent Quasi Simultaneous-Multiple Births and Simultaneous-Multiple Deaths (QBD-M) processes (cf. [22]). Its possible transitions are determined by the following events with corresponding rates:

- $A_j(i,k)$ describes the purely lateral transition rate from state (i,j) to (k,j), for all $0 \leq j \leq L$ and $1 \leq i,k \leq MN, \quad i \neq k$. It is caused by either a phase transition of the modulating Markov process of the arrival stream or a change of the number $I_2(t)$ of available HTTP service processes. We can write $A_j = U_j \bigoplus \Omega_X$, where $\bigoplus$ denotes the Kronecker sum of two matrices.

- $B_{i,j,j+s}$ denotes the s-step upward transition rate from state (i,j) to $(i,j+s)$, $1 \leq s \leq L-j$, $0 \leq j \leq L$ and $1 \leq i \leq MN$. It is caused by a new batch arrival of size $s \geq 1$ of connection requests. For a given j, s can be considered as bounded variable if L is finite and unbounded if L is infinite. When $s \geq L - j \geq 0$ the queue gets full and $j + s - L$ requests are lost. Then it holds for $1 \leq i \leq MN$:

$$\begin{aligned} B_{i,j-s,j} &= (1-\theta_{f_1(i)})\theta_{f_1(i)}^{s-1}\sigma_{f_1(i)} \qquad 0 \leq j-s \leq L-2; 1 \leq j \leq L-1, \\ B_{i,j,L} &= \sum_{s=L-j}^{\infty} (1-\theta_{f_1(i)})\theta_{f_1(i)}^{s-1}\sigma_{f_1(i)} = \theta_{f_1(i)}^{L-1-j}\sigma_{f_1(i)} \quad 0 \leq j \leq L-1. \end{aligned}$$

- $C_{i,j,j-1}$ is the departure rate of HTTP connections, i.e., the Web server finishes the service of a request and the downward transition from state (i,j) to $(i,j-1)$, $1 \leq j \leq L$, $1 \leq i \leq MN$, represents the departure of the corresponding connection request from the HTTP server.
  We assume that a maximum of one request departs at any time. Therefore, the one-step downward transitions take place by the departures of single connection requests after their service completion. The service of requests depends on the contention for physical resources, i.e. CPU processing and disks' operations, in the host machine.
  Applying [93]'s approach with a CQN model of the Web server's physical resources (see Fig. 4.4) and Mean Value Analysis to solve it, we obtain the corresponding values of the throughput $T_p(j)$ for j requests and p active processes in the system. Hence, $C_{i,j,j-1} = T_{f_2(i)}(j)$ for $1 \leq i \leq MN$.

Since the Markov chain $Y(t)$ with its finite state space is ergodic, its corresponding steady-state probabilities $\{p_{i,j}\}$, $p_{i,j} = \lim_{t\to\infty} \mathbb{P}(I(t) = i, J(t) = j)$, exist. Let $\mathbf{v}_j = (p_{1,j}, \ldots, p_{NM,j})$ denote the row vector of these probabilities corresponding to level $J(t) = j$ for $0 \leq j \leq L$

and $\mathbf{v} = (\mathbf{v}_0, \mathbf{v}_1, \ldots, \mathbf{v}_L)$.
We define the diagonal matrices

$$\begin{aligned}
B_{j-s,j} &= diag\Big[B_{1,j-s,j}, B_{2,j-s,j}, ..., B_{NM,j-s,j}\Big], \qquad (0 \le j-s < j \le L),\\
B_s &= B_{j-s,j} \qquad (1 \le s \le j \le L-1)\\
&= E_N \bigotimes diag\Big[\sigma_1(1-\theta_1)\theta_1^{s-1}, ..., \sigma_M(1-\theta_M)\theta_M^{s-1}\Big],\\
\Sigma &= E_N \bigotimes diag[\sigma_1, \sigma_2, ..., \sigma_M],\\
\Theta &= E_N \bigotimes diag[\theta_1, \theta_2, ..., \theta_M],\\
C_j &= diag[C_{i,j,j-1}] = diag[T_1(\min(j,1)), ..., T_N(\min(j,N))] \bigotimes E_M\\
&\quad (0 < j \le L),
\end{aligned}$$

where $diag(\cdot)$ represents a diagonal matrix of the corresponding elements and E_N, E_M denote an NxN and MxM identity matrix, respectively.
Then we get (since $\Theta^0 = E_{NM}$ is the identity matrix):

$$\begin{aligned}
B_s &= \Theta^{s-1}(E_{MN} - \Theta)\Sigma \qquad (1 \le s \le L-1),\\
B_{L-s,L} &= \Theta^{s-1}\Sigma \qquad (1 \le s \le L).
\end{aligned}$$

Let $\mathbf{e}_{NM}$ and $\mathbf{e}_M$ be a column vector with all ones of size NM and M, respectively. D^{A_j}, D^{C_j} are those diagonal matrices whose diagonal elements are the sums of the elements in the corresponding rows of A_j and C_j, respectively.
Consequently, the steady-state balance equations $\mathbf{v} \cdot Q_Y = 0$ and normalization condition read as follows.

(1) For the L^{th} column of Q_Y, i.e. level $j = L$:

$$\sum_{s=1}^{L} \mathbf{v}_{L-s} B_{L-s,L} + \mathbf{v}_L[A_L - D^{A_L} - D^{C_L}] = 0,$$

(2) For the j^{th} column of Q_Y, i.e. level $0 < j \le L-1$:

$$\sum_{s=1}^{j} \mathbf{v}_{j-s} B_s + \mathbf{v}_j[A_j - D^{A_j} - \Sigma - D^{C_j}] + \mathbf{v}_{j+1} C_{j+1} = 0,$$

(3) For the 0^{th} column of Q_Y, i.e. level $j = 0$:

$$\mathbf{v}_0[A_0 - D^{A_0} - \Sigma] + \mathbf{v}_1 C_1 = 0,$$

(4) Normalization condition:

$$\sum_{j=0}^{L} \mathbf{v}_j \cdot \mathbf{e}_{NM} = 1.$$

The steady-state probabilities $\{p_{i,j}\}$ can be obtained either by either directly solving the steady-state balance equations or by advanced matrix-geometric methods or the spectral expansion method (cf. [95], [15], [81] and [100]). Before we can use these advanced matrix-geometric techniques or the spectral expansion method, the computational methodology proposed by [33] has to be applied.

4.3.5 Performance Measures

Then important performance measures can be determined in terms of the steady-state probabilities:

- The mean number of idle processes (and in a similar way the variance) is determined by

$$E(Idle) = \sum_{i=1}^{NM} \sum_{j=0}^{L} p_{i,j} \cdot \max(f_2(i) - j, 0). \tag{4.1}$$

- The probability p_W that a new TCP request has to wait until the Web server provides a new idle child process is determined as follows. Upon the arrival of a batch of size s, the number of idle processes is given by $\max(f_2(i)-j, 0)$ and $\max(j-f_2(i), 0)$ is the number of waiting processes. Hence, $\max(s - \max(f_2(i) - j, 0), 0)$ is the number of requests which cannot "immediately" get service by idle processes, i.e. they have to wait in the software queue for the birth of new `httpd` processes or get lost due to the shortage of buffering in state (i, j). Taking into account that a maximum of $L - j$ requests can be admitted in state (i, j), then $\min(\max(s-\max(f_2(i)-j, 0), 0), L-j-\max(f_2(i)-j, 0))$ is the number of requests which are forced to wait. The arrival rate of batches is given by $\sigma_{f_1(i)}$ and $\sigma_{f_1(i)}.(1-\theta_{f_1(i)}).\theta_{f_1(i)}^{s-1}$ determines the arrival rate of batches of size s in state (i, j). Therefore, p_W can be written as

$$\begin{aligned} p_W &= \sum_{i=1}^{NM} \sum_{j=0}^{L} \sum_{s=\max(f_2(i)-j,0)+1}^{\infty} p_{i,j} \cdot \frac{\sigma_{f_1(i)}(1-\theta_{f_1(i)})\theta_{f_1(i)}^{s-1} s}{\overline{\sigma}} \\ &\quad \cdot \frac{\min(\max(s - \max(f_2(i) - j, 0), 0), L - j - \max(f_2(i) - j, 0))}{s} \\ &= \sum_{i=1}^{NM} \sum_{j=0}^{L} \sum_{s=\max(f_2(i)-j,0)+1}^{\infty} p_{i,j} \cdot \frac{\sigma_{f_1(i)}(1-\theta_{f_1(i)})\theta_{f_1(i)}^{s-1}}{\overline{\sigma}} \\ &\quad \cdot \min(s - \max(f_2(i) - j, 0), L - j - \max(f_2(i) - j, 0)), \end{aligned} \tag{4.2}$$

where $\overline{\sigma}$ denotes the overall mean arrival rate of the individual requests given by

$$\overline{\sigma} \;=\; \sum_{l=1}^{M} \frac{\sigma_l}{(1-\theta_l)} r_l. \tag{4.3}$$

Here $\mathbf{r} = (r_1, r_2, \ldots, r_M)$ is the vector of the equilibrium probabilities of the modulating Markov chain X that is uniquely determined by the equations:

$$\mathbf{r} \cdot Q_X = 0 \quad ; \quad \mathbf{r} \cdot \mathbf{e}_M = 1.$$

4.4 Performance Results

4.4.1 Validation of the Analytic Model

To validate the proposed modeling approach, a measurement configuration of an Apache Web server operating on a host with a 3.00GHz Intel(R) Pentium(R) D CPU, 2 Mbyte cache, and 2 Gbyte memory has been set up under Linux. The corresponding workload described in Section 4.3.3.1 has been generated by an appropriate shell script running on another Linux machine in combination with the traffic generator `ab`[3] (see [2]). Moreover, the major statistics have been captured, e.g. the creation and killing times, the idle and busy times of each HTTP process.
In the following, we present some numerical results related to a HTTP/1.0 workload model. The Web server system has the parameters $h_{\min} = 5, h_{\max} = 10, N = 30, L = 40$. In our corresponding traffic model the modulating process X has only one state. This means that the arrivals of TCP connection requests constitute a recurrent process and its inter-arrival times are approximated by a GE distribution with the parameters (σ, θ). Hence, we capture only the burstiness of the arrivals without considering a sophisticated autocorrelation structure of the request process. Despite of this restriction, the derived results already indicate a sufficient level of accuracy of our new performance model. To generate the numerical results of the latter analytic model, we need the service time parameters as well as the killing and the creation rate of the `httpd` process. These parameters were determined based on the statistics, i.e. the creation and killing times, the idle and busy times, of each HTTP service process arising from the measurements.

In Table 4.2 we show a comparison between the measurements and the results arising from the solution of our analytic model by a procedure for QBD-M processes (cf. [22], [33]). The performance metric includes the

[3] We have used the tool `ab` since it is capable to initiate simultaneously a batch of HTTP requests. To generate complex arrival distributions of sessions, an additional shell script is needed.

Table 4.2: Comparison of the measurement results and the analytic solution

Parameters				Measurement Study			Analytic Model		
σ	θ	η	ϵ	E(Idle)	Var(Idle)	Waiting Prob.	E(Idle)	Var(Idle)	Waiting Prob.
5	0.4	0.003960	0.001500	9.92740	0.4323	0.000087	9.933774	0.333595	0.0003301
5	0.5	0.007042	0.003130	9.89174	0.6153	0.003466	9.906068	0.437426	0.0023988
5	0.6	0.011198	0.007337	9.83198	0.8833	0.015859	9.858125	0.602280	0.0122027
5	0.7	0.020465	0.016675	9.70886	1.3312	0.056009	9.762665	0.895090	0.0485737
5	0.8	0.062304	0.058661	9.47287	2.0765	0.175968	9.545315	1.460498	0.1602376
5	0.9	0.230746	0.227524	9.23774	3.8538	0.385522	8.876794	2.732862	0.4270081
10	0.4	0.006120	0.001749	9.87653	0.5695	0.000028	9.865037	0.477537	0.0001713
10	0.5	0.007774	0.003455	9.82228	0.7829	0.002989	9.812958	0.618592	0.0029087
10	0.6	0.016354	0.012161	9.72373	1.1271	0.015115	9.716133	0.852531	0.0146051
10	0.7	0.040675	0.036607	9.54612	1.6814	0.056359	9.525983	1.265490	0.0576934
10	0.8	0.095757	0.091942	9.15435	2.5931	0.178238	9.098354	2.055569	0.1883662
10	0.9	0.330487	0.327367	9.00565	4.5981	0.360757	7.918966	3.700704	0.4729701
20	0.4	0.007869	0.002951	9.82219	0.6871	0.000782	9.733936	0.671095	0.0006167
20	0.5	0.012012	0.007207	9.74126	0.9354	0.003211	9.628082	0.872976	0.0041383
20	0.6	0.020740	0.016131	9.60848	1.3171	0.015597	9.447366	1.195716	0.0200142
20	0.7	0.057788	0.053410	9.37257	1.9583	0.054940	9.083828	1.772014	0.0775575
20	0.8	0.130866	0.126864	8.88637	2.9865	0.166680	8.328340	2.831075	0.2433198
20	0.9	0.408192	0.405091	8.23474	5.3645	0.536450	7.083550	4.933103	0.5040300

mean $E(Idle)$ and the variance $Var(Idle)$ of the number of idle processes as function of the workload parameters σ and θ, as well as the probability p_W that a new TCP request needs to wait until the Web server offers a new `httpd` process, i.e., the probability that a new request does not find an idle `httpd` process upon its arrival.

In the selected scenario the measurements generate an average of 9.5 idle processes and a minimum and maximum of 8.2 and 9.9, respectively, whereas the analytic model has a range of $[7.1, 9.9]$ and a mean of 9.3 idle processes. Regarding the mean number of idle processes the analytic model generates an average relative error of less than 3% subject to the measured values, the relative error of the variance is about 20%. Considering the waiting probability the absolute error is less than 11.3%, however, the relative error is larger since the correlation of the arrivals is not fully considered in the selected scenario. For a properly tuned system, the waiting probability can be kept small. Regarding control purposes the model is accurate enough and can be successfully applied to predict the average behavior of idle processes.

In conclusion, the numerical results justify the claim that our mathematically tractable model can be used to predict rather accurately the mean-value performance of the MPM Prefork process, despite of its explicit assumption about the Markovian property of $I_2(t)$.

4.4.2 Performance Impact of Apache Directives

In real operation it is of major importance to analyze the impact of basic Apache directives such as *MaxClients*, *MaxSpareServers* and *MinSpareServers* on the performance indices perceived by the users such as the waiting probability and the efficiency observed by operators, e.g. in terms of the average number of idle processes. Therefore, we have depicted in Figure 4.5

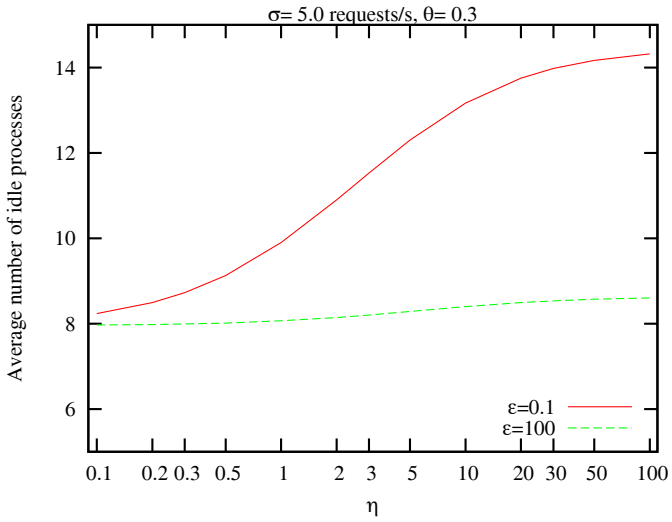


Figure 4.5: Average number of idle processes vs. η and ϵ

the average number of idle processes as function of the creation rate η and the killing rate ϵ for some specific parameter values of the request rate. The results quantitatively show that ϵ has a crucial influence on the number of idle processes. The average number is almost independent of the creation rate η if the killing rate ϵ is large enough. The rationale of this behavior is as follows: when the killing rate ϵ is large enough, the control policy of a Web server can achieve its goal in a very fast manner, i.e. to enforce a number of idle processes less than h_{max}. This means that the interval determined by the reaction to the critical load period until the compliance with the control policy is very short. From the point of view of an autonomous Web server operation, this behavior of the MPM Prefork module is useful. It is proved by the operational success of Apache Web servers every day.
The overall response time of the Web server observed by a user sending HTTP requests depends upon the physical configuration of the corresponding machine running the Web server, the parameter setting of the Web server, the size of the objects associated with the specific request and the network status. Since the response time at the server side plays a very significant role in the overall response time perceived by the users, we focus on those factors which have a major impact on the latter performance index.

Following the approach in [93] we have assumed in our analysis that after the successful TCP connection set-up the residual service time upon the acceptance of a HTTP request is approximated by a CQN. The latter provides the input of our service time model. If the machine running the Web server is powerful enough, then after the TCP connection set-up the service time should be small enough to satisfy the QoS requirement of a specific user.

From the users' perspective, the most disturbing event happens when there is no idle service process available upon the arrival of a request. Then the user does not receive any response and has to wait until a new service process is spawned by the Web server. Therefore, this waiting time to generate a new service instance can significantly contribute to the overall response time. In Table 4.2 we can observe, for example, that the resulting waiting probability is around 0.5 when $\sigma = 20$ and $\theta = 0.9$.

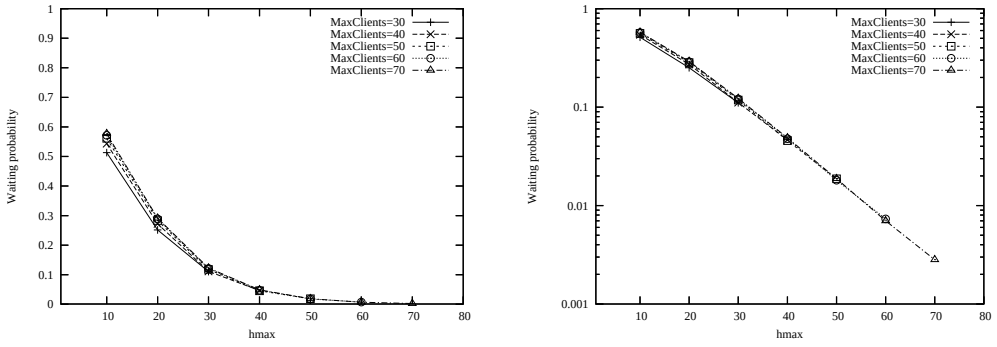


Figure 4.6: Waiting probability p_W vs. MaxClients N and MaxSpareServers $h_{\max}$ on a linear (left) and a logarithmic scale (right) for the parameters $\sigma = 20$ and $\theta = 0.9$

To further explore the impact of the Apache directives on the waiting probability p_W, we illustrate in Figure 4.6 the latter performance index as function of the parameters *MaxClients* N and *MaxSpareServers* $h_{\max}$ on a linear and a logarithmic scale of the vertical axis, respectively. Note that *MaxSpareServers* is upper bounded by *MaxClients*. However, the *MaxSpareServers* directive has a much stronger influence on the waiting probability governed by a nearly linear shape on the logarithmic scale, i.e. it follows an exponential decay characteristic in $h_{\max}$.
In summary, we have clearly shown by our analytic means that for the given non-threaded multi-processing module *Prefork* both the *MaxClients* and the *MaxSpareServers* directive have a strong impact on the waiting probability of HTTP requests and, therefore, on the response time of the Apache Web server.
Considering the autonomous operation and optimized control of an Apache Web server with MPM Prefork, we see that the appropriate dynamic tuning of these major directives and control parameters provides the highest potential to satisfy both the operator's efficiency and the users' QoS requirements.

4.5 Summary

In this Chapter we have studied the dynamic behavior of the resource pool of service processes that is implemented in the non-threaded multi-processing module (MPM) *Prefork* of an Apache Web server. To describe the latter and to analyze its performance we have proposed a new Markovian queueing model with a variable number of servers.
Applying a batch Markovian arrival process and the advanced spectral expansion method, numerical results on the performance of Apache's dynamic pool of service processes have been derived from this tractable analytic multi-server model. We have also compared it with actual measurements of the real dynamic pool-size behavior. The latter justify and validate the proposed decomposition modeling and analysis approach. It is derived from our major assumption that the modulating joint arrival and service process $I(t)$ is governed by a Markovian property. Furthermore, the model can be easily extended to capture multiple arrivals and multiple departures of connection requests.
In conclusion, we believe that our approach is a useful tool to predict and to tune the performance of an Apache Web sever with the non-threaded multi-processing module *Prefork*. It can also be extended to evaluate other Web server software architectures under UNIX such as the Apache 2.2 hybrid multi-threaded multi-processing module *Worker* which is the subject of future research.

Part III

Retrial Multi-Server Queues and ICT Applications

5

A Model for the Performance Evaluation of DHCP[‖]

5.1 Introduction

Dynamic Host Configuration Protocol (DHCP) is designed by the dynamic host configuration working group within the framework of the Internet Engineering Task Force (IETF). At present, DHCP is specified for Internet Protocol version 4 in IETF "draft standard" RFC 2131 [59] and for Internet Protocol version 6 in IETF RFC 4361 [86]. The main aim of DHCP is to provide an automatic mechanism for the allocation, configuration and management of IP addresses and IP networking parameters (netmask, router IP address, etc) for computers and devices in IP networks.

The important feature of DHCP is a "dynamic allocation" mechanism, which assigns an IP address to a client for a limited period of time (called a lease time). Therefore, a previously allocated IP address which is not used by one host can automatically be assigned to another host by a DHCP server implementing the dynamic allocation mechanism. It is recognized that the appropriate setting of a lease time in a DHCP server plays an important role in the efficient allocation of IP addresses. In [73], the authors investigated the impact of setting lease times using the data from the Georgia Tech campus network. However, due to the lack of a quantitative performability model and the lack of data at clients (whether they are forced to wait for an IP address), they only examined the utilization of the allocatable address space in a DHCP server.

We propose a method to quantitatively evaluate the performance of a DHCP dynamic allocation mechanism and the impact of a lease time. To construct a retrial queue and a tractable solution, the following steps are performed. We show that interarrival times of DHCP requests from clients follow the exponential distribution. We make a relaxed assumption concerning the lease time sent by a DHCP server and the retrials of clients.

[‖]Tien Van Do. "An Efficient Solution to a Retrial Queue for the Performability Evaluation of DHCP". *Computers and Operations Research*, 37(7):1191–1198, 2010, http://dx.doi.org/10.1016/j.cor.2009.05.014

We develop an efficient computational algorithm to calculate the steady state probabilities and the performance measures of a continuous time discrete state Markov (CTMC) process associated with the proposed retrial queue. It is shown via simulation of a more detailed model than an analytical abstract model of DHCP that the proposed model is accurate to calculate the performance of the interaction between the behavior of clients and the DHCP mechanism. A numerical study is also performed, which provides an insight for the impact of trade-off parameters and factors on the operation of DHCP.

The rest of this chapter is organized as follows. In Section 5.2, the overview of DHCP operation is presented. In Section 5.3, the proposed model and a computational algorithm is described. In Section 5.4 a numerical study is provided to reveal some interesting behaviors of the IP address allocation mechanism. Finally, the chapter is concluded in Section 5.5.

5.2 Overview of the DHCP Operation

The operation of DHCP assumes two roles. A centralized DHCP server manages a range of IP addresses allocated by a network administrator for a specific IP subnet. The communications between a DHCP server and a client are delivered by the DHCP protocol. A DHCP client software running on computers or devices normally sends a broadcast query (DHCPDISCOVERY message) requesting information from a DHCP server. The DHCP server checks whether the message is sent from the client with a permissible Media Access Control (MAC) address. If the client is authorized, the server assigns the client an IP address, a lease time, the subnet mask and the default gateway address encapsulated in the DHCPOFFER message.

Note that the whole process is performed in the similar way, if a client knows the IP address of a DHCP server in advance of the request of an IP address. The only exception is that a client sends DHCPREQUEST message instead of DHCPDISCOVERY message.

Three main modes for IP address allocation are supported: manual, automatic and dynamic allocation. The purpose of the “manual allocation” mode is to allow the network administrator to centrally store information concerning client hosts. In this mode the IP address is assigned by the network operator to a client host. After the identification of a specific client (e.g. based on hardware MAC address) DHCP sends a fixed IP address and configuration parameters (e.g.: the subnet mask, the default gateway address) for the client. This kind of operation is typically applied in a campus or LAN environment. In the mode ”automatic allocation”, a DHCP server assigns a permanent IP address to a client host.

The most important feature of DHCP is the “dynamic allocation” mechanism, which assigns an IP address to a client for a limited period of

time. A lease time is defined as a period of time for which the server gives a permission for a client to use the address. Note that a lease time is also sent to a client. Upon the expiration of the lease time, the allocated address becomes free and can be assigned to another client unless a client extends the right to use a specific IP address before the expiration of the lease time. This feature is often applied in the environment of Internet Service Providers because the reuse of scarce IP addresses is possible.

The decision that a DHCP client "leaves" the system or renews the use of the allocated IP address depends on the relation between the lease time and the holding time (e.g.: the working time) of clients. In order to extend the use of the allocated IP address the client sends a DHCPREQUEST message which includes the client's allocated IP address in the "requested IP address" option of a DHCPREQUEST message.

5.3 An Efficient Algorithm to a Retrial Queue Model for DHCP

5.3.1 A Retrial Queue

The size of the pool (i.e.: the number of allocatable IP addresses) is c. The fix lease time value sent by the DHCP server is denoted by T_l.

We assume the inter-arrival times of DHCP DISCOVERY messages are exponentially[1] distributed with a mean inter-arrival time $1/\lambda$.

Assume that the holding times (i.e.: how long does a client need an IP address) of clients are represented by random variable H with a cumulative distribution function $\mathbb{P}(H < x) = F(x)$. Upon the expiration of the lease time, the previously allocated address at the DHCP server becomes free and can be allocated to another client unless the client extends the use of a specific IP address before the expiration of the lease time. Let a denote the probability that DHCP clients leave (i.e.: switch off the computer) the system or do not renew the allocated IP address after the expiration of its lease time. We can write

$$a = \mathbb{P}(H < T_l) = F(T_l).$$

It is worth emphasizing that there is no any specific assumption concerning about the relation of the average holding time and the lease time in our model.

$I(t)$ denotes the number of allocated IP addresses at time t. Note that $0 \leq I(t) \leq c$ holds. A client who does not receive the allocation of an IP

[1]We process the log file of the DHCP server of our department between the period of January 2 and May 28, 2008. In Figure 5.1, the straight line of the Q-Q plot, where the interarrival times of DHCP requests between 8h and 18h during the investigation period to the DHCP server are plotted against the theoretical exponential distribution, confirms our assumption.

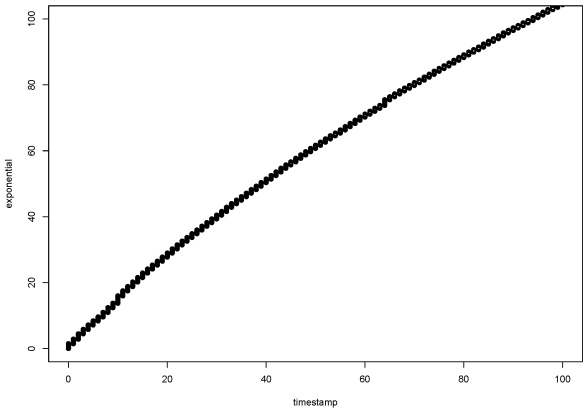


Figure 5.1: Q-Q plot for the interarrival times (measured in seconds) of DHCPDISCOVERY messages

address because the shortage (when $I(t) = c$) of IP addresses sets a timer to wait for a limited time and will retry the request for an IP address upon the expiration of backoff time. We model this phenomenon as the client joins the "virtual orbit". $J(t)$ represents the number of DHCP clients in the "orbit" at time t and takes values from 0 to ∞.

In order to have a mathematically tractable model, we make the following assumptions.

- Lease times are exponentially distributed with a mean lease time $1/\mu = T_l$.
- Clients waiting in the orbit repeat the request for the DHCP server with rate ν (i.e.: the inter-repetition times are exponentially distributed with parameter ν), which is independent from the number of waiting clients in the orbit.

Therefore, the presented approach below is the application of an approximate model for the DHCP mechanism presented in Section 5.2. It will be shown in Section 5.4 (through the comparison with the simulation of the DHCP mechanism) that the approximate model provides a quite good prediction for the performance measures of the DHCP dynamic allocation mechanism.

As a consequence, the system is modeled by a CTMC, $Y = \{I(t), J(t)\}$, with the state space $\{0, 1, \ldots, c\} \times \{0, 1, \ldots\}$.

Remarks: The stationary distributions of the main M/M/c retrial queue with $c > 2$ can be computed using approximation techniques [9, 12, 63]. Falin and Templeton proposed a truncation model and a numerically tractable solution with a threshold in their book [63], which is followed by the work [10]. The retrial queue presented in this chapter is indeed a numerically tractable

model [63] with 0 threshold value. However, only matrix-geometric solution is suggested in [63]. We show in the later section that we develop an efficient computational algorithm for the considered retrial queue and the evaluation of the DHCP dynamic allocation mechanism based on the considered retrial queue is accurate.

5.3.2 A Quasi-Birth-and-Death Representation

We denote the steady state probabilities by $p_{i,j} = \lim_{t\to\infty} \mathbb{P}(I(t) = i, J(t) = j)$, and introduce $\mathbf{v}_j = (p_{0,j}, \ldots, p_{c,j})$.

The evolution of Y is driven by the following transitions.

(a) $A_j(i,k)$ denotes a transition rate from state (i,j) to state (k,j) $(0 \le i,k \le c; j = 0,1,\ldots)$, which is caused by either the arrival of DHCPDISCOVERY requests or by the expiration of the lease time without the renewal of an allocated IP address. Matrix A_j is defined as the matrix with elements $A_j(i,k)$. Since A_j is j-independent, it can be written as

$$A_j = A = \begin{bmatrix} 0 & \lambda & 0 & \ldots & 0 & 0 & 0 \\ a\mu & 0 & \lambda & \ldots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & & \ldots & a(c-1)\mu & 0 & \lambda \\ 0 & 0 & & \ldots & 0 & ac\mu & 0 \end{bmatrix} \quad \forall j \ge 0.$$

(b) $B_j(i,k)$ represents one step upward transition from state (i,j) to state $(k,j+1)$ $(0 \le i,k \le c; j = 0,1,\ldots)$, which is due to the arrival of DHCPDISCOVERY requests when no free IP address is available in the IP address pool. In the similar way, matrix B_j (B) with elements $B_j(i,k)$ is defined as

$$B_j = B = \begin{bmatrix} 0 & 0 & 0 & \ldots & 0 & 0 & 0 \\ 0 & 0 & 0 & \ldots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & & \ldots & 0 & 0 & 0 \\ 0 & 0 & & \ldots & 0 & 0 & \lambda \end{bmatrix} \quad \forall j \ge 0.$$

(c) $C_j(i,k)$ is the transition rate from state (i,j) to state $(k,j-1)$ $(0 \le i,k \le c; j = 1,\ldots)$, which is due to the successful retrial of a request from the orbit. Matrix C_j $(\forall j \ge 1)$ with elements $C_j(i,k)$ is written as

$$C_j = C = \begin{bmatrix} 0 & \nu & 0 & \dots & 0 & 0 & 0 \\ 0 & 0 & \nu & \dots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & & \dots & 0 & 0 & \nu \\ 0 & 0 & & \dots & 0 & 0 & 0 \end{bmatrix} \quad \forall j \geq 1.$$

D^A and D^C are diagonal matrices whose diagonal elements are the sum of the elements in the corresponding row of A and C, respectively. The infinitesimal generator matrix of Y can be written as follows

$$\begin{bmatrix} A_{00} & \mathrm{B} & 0 & \dots & \dots & \dots & \dots \\ \mathrm{C} & Q_1 & \mathrm{B} & 0 & \dots & \dots & \dots \\ 0 & \mathrm{C} & Q_1 & \mathrm{B} & 0 & \dots & \dots \\ 0 & 0 & \mathrm{C} & Q_1 & \mathrm{B} & 0 & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \end{bmatrix}, \tag{5.1}$$

where $A_{00} = A - D^A - B$ and $Q_1 = A - D^A - B - D^C$.

Because of the special structure of the QBD, the steady state probabilities can be obtained with the existing methods like the matrix-geometric and its variants [14, 84, 99], and the spectral expansion [97]. However, the existing methods have the "state-space explosion" problem when c is large. The problem starts when c reaches a value of several hundreds (no results or a very long-running time of computer programs implementing these methods). Therefore, in what follows we present an efficient computational procedure to find the steady state probabilities.

5.3.3 An Efficient Computational Procedure

For $j \geq 1$, the balance equations are written as follows

$$\mathbf{v}_{j-1}B + \mathbf{v}_j Q_1 + \mathbf{v}_{j+1}C = 0 \ \ (j \geq 1). \tag{5.2}$$

$Q(x) = B + Q_1 x + Cx^2$ is defined as the characteristic matrix polynomial associated with equations (5.2). In the present chapter, $Q(x)$ is a tridiagonal matrix

$$Q(x) = \begin{bmatrix} q_{11}(x) & \lambda x + \nu x^2 & 0 & \dots & 0 & 0 & 0 \\ a\mu x & q_{2,2}(x) & \lambda x + \nu x^2 & \dots & 0 & 0 & 0 \\ 0 & a2\mu x & q_{3,3}(x) & \lambda x + \nu x^2 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & & \dots & a(c-1)\mu x & q_{c,c}(x) & \lambda x + \nu x^2 \\ 0 & 0 & & \dots & 0 & ac\mu x & q_{c+1,c+1}(x) \end{bmatrix}$$

where

$$\begin{aligned} q_{1,1}(x) &= -(\lambda+\nu)x, \\ q_{i,i}(x) &= -(\lambda+\nu+(i-1)a\mu)x \quad (i=2,\ldots,c), \\ q_{c+1,c+1}(x) &= \lambda-(\lambda+c\mu a)x. \end{aligned}$$

The steady state probabilities are closely related to the eigenvalue-eigenvector pairs $(x,\boldsymbol{\psi})$ of $Q(x)$, which satisfy $\boldsymbol{\psi}Q(x)=0$ and $det[Q(x)]=0$ (cf. [97]). It is easy to see that $Q(x)$ has c zero-eigenvalues. The corresponding independent eigenvectors for c zero-eigenvalues are $\boldsymbol{\psi}_1=\{1,0,\ldots,0\}$, $\boldsymbol{\psi}_3=\{0,1,0,\ldots,0\}$,...,$\boldsymbol{\psi}_c=\{0,0,\ldots,1,0\}$.

Note that if the system is ergodic, then the number of eigenvalues of $Q(x)$ of an infinite QBD process, which are inside the unit disk, is $c+1$ (see Section 2.2.1 and [97] for the proof). Therefore, $Q(x)$ should have a single eigenvalue x_0 inside the unit disk because $Q(x)$ has c zero-eigenvalues. Let $\boldsymbol{\psi}_0$ the corresponding left-hand-side eigenvector of $Q(\lambda)$ for the eigenvalue x_0.

As a consequence, the steady state probabilities can be expressed as follows

$$\begin{aligned} \mathbf{v}_j &= b_0\boldsymbol{\psi}_0 x_0^j \ (j\geq 1), \\ \mathbf{v}_0 &= \sum_{k=0}^{c} b_k\boldsymbol{\psi}_k, \end{aligned} \tag{5.3}$$

where b_i are the coefficients to be determined. Since the probabilities are greater or equal 0, $0<x_0<1$ holds.

The straightforward way to obtain the steady state probabilities is to find the eigenvalues of $Q(x)$ (see [13] for the methodology to find the eigensystem of the matrix polynomial). Then, one could use the balance equation for level 0

$$\mathbf{v}_0 A_{00}+\mathbf{v}_1 C=0 \tag{5.4}$$

and the normalization equation

$$\sum_{i=0}^{c}\sum_{j=0}^{\infty} p_{i,j} = \sum_{k=1}^{c} b_k\boldsymbol{\psi}_k\mathbf{e}+b_0\boldsymbol{\psi}_0\mathbf{e}/(1-x_0)=1 \tag{5.5}$$

to determine the coefficients b_i. Note that $\mathbf{e}$ is a $1\times(c+1)$ vector with all elements equal 1.

The key step towards the steady state probabilities is to determine x_0 and the corresponding eigenvector $\boldsymbol{\psi}_0$.

Theorem 5.1. *$0 < x_0 < 1$ is the root of $l_{c+1}(x)$, the last diagonal element of $L(x)$ when we make the LU decomposition of $Q(x) = L(x)U(x)$.*

Proof. Since $Q(x_0)$ is a tridiagonal matrix and $q_{i,i}(x_0) \neq 0$, the component matrices of the LU decomposition of $Q(x_0)$ are written as

$$L(x_0) = \begin{bmatrix} l_1(x_0) & 0 & 0 & \dots & 0 & 0 & 0 \\ a\mu x_0 & l_2(x_0) & 0 & \dots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & & \dots & a(c-1)\mu x_0 & l_c(x_0) & 0 \\ 0 & 0 & & \dots & 0 & ac\mu x_0 & l_{c+1}(x_0) \end{bmatrix},$$

$$U(x_0) = \begin{bmatrix} 1 & u_1(x_0) & \dots & 0 & 0 & 0 & 0 \\ 0 & 1 & u_2(x_0) & \dots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & & \dots & 0 & 1 & u_c(x_0) \\ 0 & 0 & & \dots & 0 & 0 & 1 \end{bmatrix},$$

where $l_i(x_0)$ $(i = 1, \dots, c+1)$ and $u_i(x_0)$ $(i = 1, \dots, c)$ are the elements of $L(x_0)$ and $U(x_0)$, respectively. After a simple algebra, it can be written as

$$\begin{aligned} l_1(x_0) &= q_{1,1}(x_0) = -(\lambda + \nu)x_0, \\ l_i(x_0) + a(i-1)\mu x_0 u_{i-1}(x_0) &= q_{i,i}(x_0), \quad (i = 2, \dots, c+1), \\ l_i(x_0)u_i(x_0) &= \lambda x_0 + \nu x_0^2, \quad (i = 1, \dots, c). \end{aligned} \tag{5.6}$$

Therefore, the determinant of $Q(x_0)$ is expressed as

$$Det[Q(x_0)] = Det[L(x_0)]Det[U(x_0)] = \prod_{i=1}^{c+1} l_i(x_0). \tag{5.7}$$

As the consequence of equation (5.6), $l_i(x_0) \neq 0$ $(1 < i \leq c)$ holds. Hence, $Det[Q(x_0)] = 0$ follows $l_{c+1}(x_0) = 0$. □

It is easy to prove that $l_{c+1}(0)$ is positive and $l_{c+1}(1)$ is negative. Therefore, a bisection algorithm can be proposed to determine x_0 as illustrated in Algorithm 5.1. Note that the recursive relations (see Algorithm 5.1) between $\psi_{0,i}$ and $\psi_{0,i+1}$ $(i = c, \dots, 1)$ are easily derived from equation

$$\boldsymbol{\psi}_0 Q(x_0) = \boldsymbol{\psi}_0 L(x_0)U(x_0) = 0.$$

Based on the property that multiplying an eigenvector with a scalar number results in an eigenvector, we can determine $\boldsymbol{\psi}_0 = \{\psi_{0,1}, \psi_{0,2}, \dots, \psi_{0,c+1}\}$ by setting $\psi_{0,c+1} = 1$ and using the recursive relations.

Algorithm 5.1 Bisection algorithm to determine x_0 and the calculation of $\boldsymbol{\psi}_0$

Initialize the required accuracy ϵ
$x_{0,u} = 1.0$, $x_{0,d} = 0$
repeat
  $x_0 = \frac{x_{0,u}+x_{0,d}}{2}$
  calculate $l_{c+1}(x_0)$ based on equation (5.6)
  if $l_{c+1}(x_0) > 0$ **then**
    $x_{0,d} = x_0$
  else
    $x_{0,u} = x_0$
  end if
until $|l_{c+1}(x_0)| < \epsilon$
$\psi_{0,c+1} = 1$
for $i = c$ to 1 **do**
  $\psi_{0,i} = \frac{-ai\mu x_0}{l_i(x_0)}\psi_{0,i+1}$
end for
return x_0, $\boldsymbol{\psi}_0$

Let us introduce $\mathbf{b}_{cf} = \sum_{i=1}^{c} b_i\boldsymbol{\psi}_i = \{b_1, b_2, \ldots, b_c, 0\}$. From equations (5.3) and (5.4), we can write

$$\sum_{i=0}^{c} b_i\boldsymbol{\psi}_i \left[D^A + B - A\right] = b_0\boldsymbol{\psi}_0 x_0 C,$$

$$\mathbf{b}_{cf}\left[D^A + B - A\right] = b_0\boldsymbol{\psi}_0 x_0 C - b_0\boldsymbol{\psi}_0\left[D^A + B - A\right],$$

$$\begin{bmatrix} \lambda b_1 - \mu a b_2 \\ -\lambda b_1 + (\lambda + \mu a)b_2 - 2\mu a b_3 \\ \vdots \\ -\lambda b_{i-1} + (\lambda + (i-1)\mu a)b_i - i\mu a b_{i+1} \\ \vdots \\ -\lambda b_{c-1} + (\lambda + (c-1)\mu a)b_c \\ -\lambda b_c \end{bmatrix} =$$

$$\begin{bmatrix} \text{- } b_0\lambda\psi_{0,1} + b_0\mu a\psi_{0,2} \\ b_0(x_0\nu\psi_{0,1} + \lambda\psi_{0,1} - (\lambda + \mu a)\psi_{0,2} + 2\mu a\psi_{0,3}) \\ \vdots \\ b_0(x_0\nu\psi_{0,i-1} + \lambda\psi_{0,i-1} - (\lambda + (i-1)\mu a)\psi_{0,i} + i\mu a\psi_{0,i+1}) \\ \vdots \\ b_0(x_0\nu\psi_{0,c} + \lambda\psi_{0,NN} - (\lambda + c\mu a)\psi_{0,c+1}) \end{bmatrix}. \quad (5.8)$$

The computation of the coefficients is based on the following observations

- both the left and right hand side of equation (5.8) are vectors.
- the last element (i.e.: element c+1) of the left hand side of equation (5.8) contains only b_c.
- only b_c and b_{c-1} are in element c of the left hand side of equation (5.8). only b_{c-1}, b_{c-2} and b_{c-3} are in element $c-1$ of the left hand side of equation (5.8), etc.

As a consequence, b_c can be expressed in b_0. Then, b_i ($i = c-1, \ldots, 1$) can be calculated recursively. That is, all b_i ($i = c, \ldots, 1$) can be given in terms of b_0. Thus, b_0 can be determined from the normalization equation. Then other coefficients $b_1, \ldots, b_c$ are calculated.

In summary, the steps of the proposed method are

- the application of Algorithm 5.1 to find root x_0,
- the computation of $\boldsymbol{\psi}_0$,
- the calculation of coefficient b_0 and $\mathbf{b}$.

5.3.4 Performance Measures

Note that performance parameters related to the DHCP dynamic allocation mechanism are obtained as follows:

- average number of occupied IP addresses

$$N_{occ} = \sum_{i=1}^{c} i \sum_{j=0}^{\infty} p_{i,j} = \sum_{i=1}^{c} i(p_{i,0} + \sum_{j=1}^{\infty} b_0 \boldsymbol{\psi}_{0,i+1} x_0^j) = \sum_{i=1}^{c} i(p_{i,0} + \frac{b_0 \boldsymbol{\psi}_{0,i+1} x_0}{1 - x_0}), \tag{5.9}$$

- average number of clients waiting in the orbit

$$N_{orbit} = \sum_{j=1}^{\infty} j \sum_{i=0}^{c} p_{i,j} = \sum_{j=1}^{\infty} j b_0 \sum_{i=0}^{c} \boldsymbol{\psi}_{0,i+1} x_0^j = \frac{b_0 x_0}{(1 - x_0)^2} \sum_{i=0}^{c} \boldsymbol{\psi}_{0,i+1}. \tag{5.10}$$

5.4 Case Study

Three scenarios are investigated in this section. The first scenario represents a case which may happen in a private company or in a small campus. In this case, a small number ($c = 250$) of IP addresses can be allocated to clients. The second and third scenarios correspond to a case where a large number ($c = 1000$ and $c = 3000$) of IP addresses are available to clients. In three

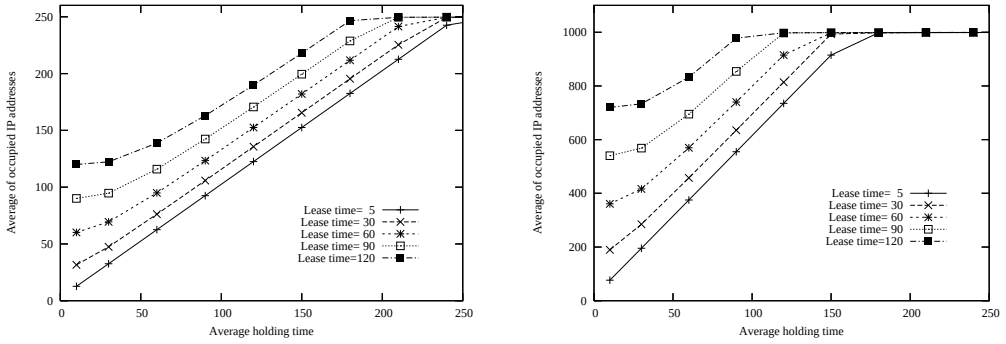
Lease time: $T_l = 5$ minutes				
Average holding time	Analytical Model		Simulation (conf. level=99%)	
t_h (minutes)	N_{occ}	N_{orbit}	N_{occ}	N_{orbit}
10	12.7075	0	12.715630	0
30	32.5694	0	32.590180	0
60	62.5347	0	62.574540	0
90	92.5231	0	92.582617	0
120	122.5170	0	122.596091	0
150	152.5140	0	152.612031	0
180	182.5120	0.000004	182.628786	0.000002
Lease time: $T_l = 30$ minutes				
10	31.5719	0	31.591576	0
30	47.4593	0	47.490397	0
60	76.2448	0	76.293760	0
90	105.832	0	105.899821	0
120	135.624	0	135.709828	0
150	165.500	0	165.605376	0
180	195.416	0.000437	195.540905	0.000423
Lease time: $T_l = 60$ minutes				
10	60.1491	0	60.186734	0
30	69.3911	0	69.436389	0
60	94.9186	0	94.980781	0
90	123.309	0	123.385416	0
120	152.49	0	152.587474	0
150	181.995	0	182.111794	
180	211.664	0.03816	211.799544	0.033517
Lease time: $T_l = 90$ minutes				
10	90.0111	0	90.068141	0
30	94.7156	0	94.774729	0
60	115.850	0	115.921970	0
90	142.378	0	142.471148	0
120	170.573	0	170.679458	0
150	199.473	0.001519	199.600627	0.001356
180	228.734	1.299020	228.881151	1.016362
Lease time: $T_l = 120$ minutes				
10	120.001	0	120.076530	0
30	122.239	0	122.315397	0
60	138.782	0	138.872755	0
90	162.954	0	163.052606	0
120	189.837	0.000067	189.961514	0.000060
150	217.916	0.154400	218.052797	0.114559
180	246.618	67.667500	247.106000	66.585000

Table 5.1: Analytical and simulation results ($c = 250$, $\lambda = 1$ requests/minute)

cases, we choose $1/\nu = 30$ s and the exponential distribution of holding times (i.e.: $F(x) = 1 - e^{-x/t_h}$), where t_h is the mean holding time. The request rate, λ, is of 1 request/minute for the first scenario, and 6 requests/minute for the second and third scenario.

For the first case, we present the comparison of our model with simulation. We have developed an own simulation program in language C based on the SimPack toolkit[2] and the statistical module[3] from Politecnico di Torino, which have been used for many simulation studies. Note that the simulation model follows the real interaction of clients and the DHCP mechanism as much as possible. Therefore, it is different from the analytical model presented in Section 5.3 in three aspects:

- the retrial rate from the orbit: in the simulation the retrial rate depends on $J(t)$ (i.e.: each waiting client retrial after $1/\nu$), while the retrial rate in the queueing model is of fixed value when $J(t) > 0$.
- the holding time: in the simulation we simulate the phenomenon of the holding time of a specific request[4], while in the queueing model we use parameter a to take into account the phenomenon of the holding time.
- the lease time: the allocated lease times are of fixed value in the real DHCP operation and our simulation model, while the lease times are exponentially distributed in the queueing model.



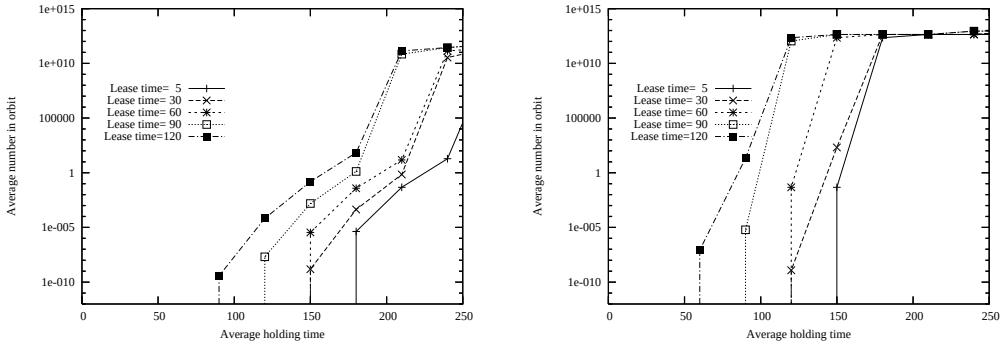
(a) $c = 250$, $\lambda = 1$ requests/minute (b) $c = 1000$, $\lambda = 6$ requests/minute

Figure 5.2: Average number of occupied IP addresses

[2] `http://www.cise.ufl.edu/~fishwick/simpack.html`

[3] We use the statistical module (`http://www.telematica.polito.it/class/statistics.ps.gz`) to collect simulation data and to perform the analysis of simulation runs.

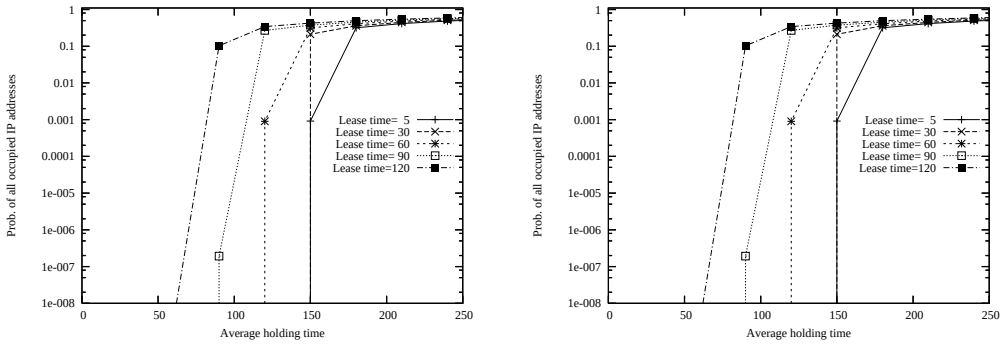
[4] The lease time sent to each a client is of a fixed value in a specific simulation and each client independently retries an IP requests after 30 s (it is the normal value observed in a DHCP client software implemented in the present operating systems).



(a) $c = 250$, $\lambda = 1$ requests/minute (b) $c = 1000$, $\lambda = 6$ requests/minute

Figure 5.3: Average number of requests waiting in the orbit

That means, the simulation model does not follow the assumption of the analytical one. Note that the simulation results are generated with the confident level of 99%. Simulation runs are stopped when the relative precision (i.e.: the ratio of the half-width of the confidence interval and the mean of collected observations) of N_{occ} reaches 0.099%. The collected measures for N_{orbit} show high variability and the relative precision of N_{orbit} is $\pm 49\%$. As observed from Table 5.1 the agreement between the simulation and analytical results is excellent concerning N_{occ}. The analytical values of N_{orbit} are within the confidence interval.



(a) $c = 250$, $\lambda = 1$ requests/minute (b) $c = 1000$, $\lambda = 6$ requests/minute

Figure 5.4: Probability that all IP addresses are being allocated

We plot the average number of occupied IP addresses versus the average holding time and the lease time in Figures 5.2, the average number of requests waiting in the orbit versus the average holding time and the lease time in Figure 5.3, and the probability that all IP addresses are being allocated in

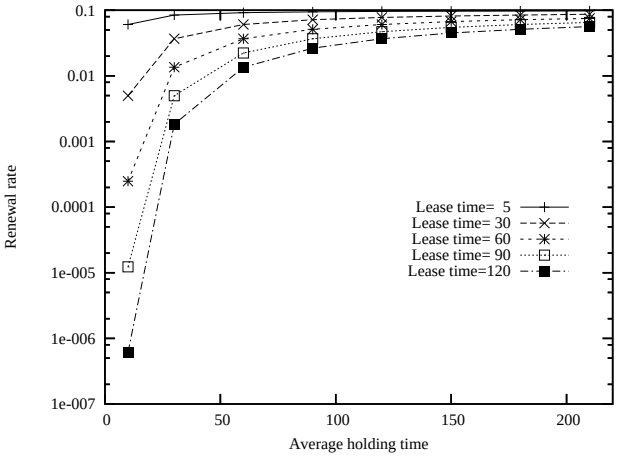
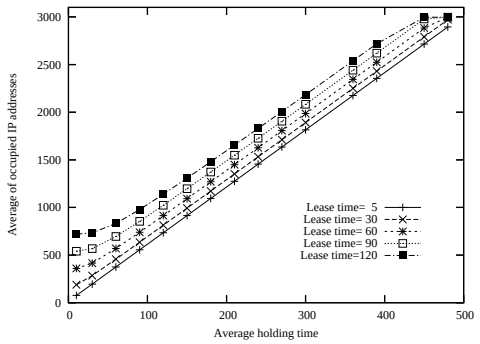
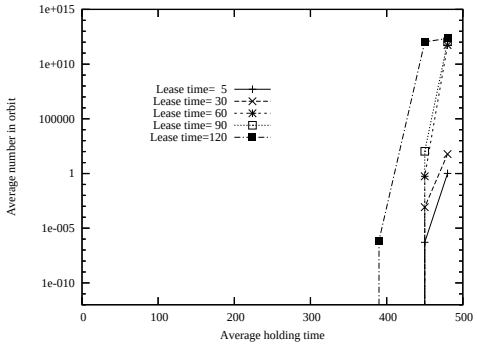


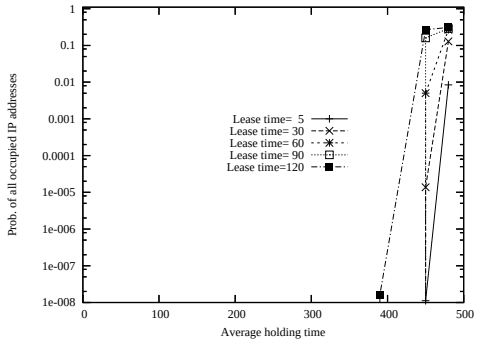
Figure 5.5: Renewal rate ($\lambda = 6$ requests/minute)



(a) Occupied IP address



(b) Waiting in the orbit



(c) Probability

Figure 5.6: The third scenario ($c = 3000$, $\lambda = 6$ requests/minute)

Figure 5.4. It can be observed that the system is overloaded when the average holding time is higher than 200 min.

The most important resource of a DHCP server is the pool of IP addresses, so the efficient allocation of IP addresses poses a crucial issue for the network administrator. As one observes that the allocation of IP addresses can be controlled with the appropriate setting of the lease length. If a DHCP server is not overloaded, then the smaller the lease time is, the more efficient the allocation of IP address (Figure 5.2) and the smaller the number of requests waiting in the orbit is (Figure 5.3). For example in the second scenario when the average holding time is 90 min and a lease time has a value of 30 min, the average number of occupied IP addresses is 635 (365 free IP addresses are available in average). If we change the setting of a lease time to 120 min, only 186 free IP addresses are available in a DHCP server. It is worth emphasizing that the small value setting of the lease time has the impact of increased number (load) of renewal messages (DHCPREQUEST). Similar observations can be obtained in the third scenario (Figure 5.6) as well (the only difference between the second and third scenario that we increase the size of the IP address pool to 3000).

In Figure 5.5, we show the rate of renewal messages versus the lease time and the average holding time. We observe that the smaller the lease time is, the larger the rate of renewal messages is, which contrasts with the behavior of the average number of occupied IP addresses versus the lease time and the average holding time (Figure 5.2). Therefore, the trade-off parameter of the DHCP dynamic allocation mechanism is the rate of renewal messages. That is, the choice of an appropriate lease time depends on the processing capacity (how many messages can be handled during one minute or one second) of a DHCP server.

We compare the running time needed by different methods to get the steady state probabilities of the QBD process. Four methods are considered in the numerical comparison: the original spectral expansion method [31], the classical matrix-geometric method based on the successive substitution (SS) procedure [102], the variant of the matrix-geometric method [101] and the proposed procedure. Note that the stopping criteria for the methods is 10^{-10}.

In Figure 5.7, we plot the computational time of the methods (implemented in Mathematica) versus c on a machine with Intel® Xeon® E5410 2.33GHz processor. It can be observed from Figure 5.7 that the proposed method has the smallest computation time amongst the methods. Moreover, the original spectral expansion method, the classical matrix-geometric method and the Naoumov et al. method fail[5] at the large value of c (e.g. $c > 600$) due to the state explosion problem. For example the

[5]Either caused by the shortage of memory or no results are obtained in a reasonable time.

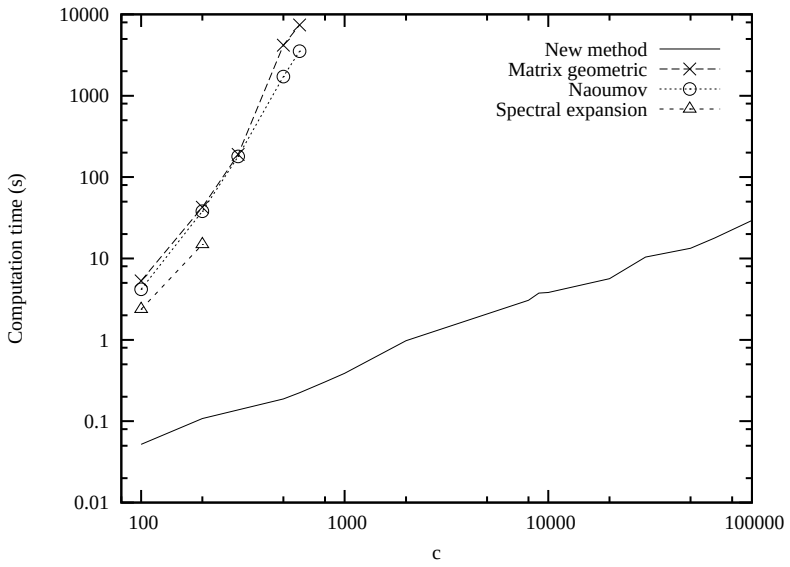


Figure 5.7: Computational time versus c of the analytical method

classical matrix-geometric method needs more than 2 hours and the Naoumov et al. method takes approximately 2 hours) to get results for $c = 600$. The classical spectral expansion method could not give reliable numerical results for $c > 300$ because a large value of c gives rise to an ill-conditioned linear system of algebraic equations, while the proposed method takes a remarkable short time (approximately *30* s) to compute results for a much higher c value ($c = 100000$).

5.5 Summary

We have provided a methodology to evaluate the performability of the DHCP dynamic allocation mechanism. We are able to derive a solution approach with the QBD model for the performability analysis of the DHCP mechanism, which computational complexity is of the order $O(c)$. Here, it is worth noting that many existing solutions [31, 82, 97] of QBD processes have the computational time complexity of $O(c^3)$. Besides possessing the capability of tackling the problem with large c (the number of available IP addresses is of the order of tens of thousands in the DHCP server) in *a very short time*, it gives numerically stable solution too. We are not aware of the works of any other authors who have succeeded in solving QBD models with such large c (that is, number of phases in the QBD). Therefore, our analytical model for the performability analysis of DHCP with a large number of IP addresses can efficiently be used in the design and dimensioning process for ISP.

6

A New Computational Algorithm to a Retrial Queue for Wireless Cellular Networks with Guard Channels[‖]

6.1 Introduction

Retrial queues have been used to model the queueing problem of new and handover calls in cellular mobile networks [11, 12, 56, 91, 92, 110, 121] and telecommunication systems [5, 8–10, 107, 115, 125]. The fact that the retrial rate depends on the number of retried calls waiting in the system leads to an analytically intractable model [12, 63]. Therefore, approximation procedures should be used to compute the performability of the system. In the literature, there are only two notable contributions which explicitly dealt with the retrial problem in the presence of *guard channels*: a computational method [121] based on the truncation of the state space and an approach [92] based on approximate Continuous Time Markov Chains (CTMC) with Boolean flags to take into account the presence of retrial calls.

Tran-Gia and Mandjes [121] proposed a retrial queueing model to evaluate the performance of cellular mobile networks which takes into account the customer retrial phenomenon, the fresh and handover calls, and the guard channel concept. They provided a recursive algorithm to calculate the probabilities of the truncated state space. It is worth mentioning that the consideration of guard channels in the queueing model leads to calculations with negative terms in a recursive algorithm. Negative terms and extremely small values involved in the computation contribute to a numerical instability in the recursive algorithm. This numerical problem is first discovered in this Chapter.

Ajmone Marsan et al. [92] presented several approximate Continuous Time Markov Chain (CTMC) models with one and two Boolean flags for the computation of blocking probabilities of new and handover calls in cellular mobile networks. They assumed that the number of customers in the orbit

[‖]Tien Van Do. *An Efficient Computational Method for Retrial Queues to Cellular Mobile Systems with Guard Channels.* Submitted

follows a geometric distribution. It will be shown later that this assumption is only valid when there are no guard channels in the system. That is, there is an accuracy issue concerning the estimation of the number of waiting customers.

Moreover, there is a need to have a fast and numerically stable computation method when a cellular area can serve a large number of calls because of the increase of the system capacity due to advanced techniques (some GSM cells with sectorial antenna and employing half rate coding for voice can accommodate a significant number of voice channels, adaptive coding and modulation techniques such as one introduced in recent standards of wireless systems [1, 76]).

The aim of this Chapter is to propose a new numerically stable and computationally efficient algorithm in order to deal with the problems mentioned above. We also show that the distribution of the number of calls in the orbit of the approximate queueing model is the mixture of geometric distributions in the presence of guard channels. It is worth emphasizing that the numerical stability, small memory requirement for the implementation of the algorithm and the capability to obtain accurate results within a short time when there is a large number of channels, are the strengths of the new approach.

The rest of this Chapter is organized as follows. The considered system is described in Section 6.2. Existing approximation approaches and a new computational approach are presented in Section 6.3. Some numerical results are demonstrated in Section 6.4. Finally, the Chapter is concluded in Section 6.5.

6.2 The Basic Problem

We consider a particular cell in a cellular mobile system with infinite user population, where there are c channels to serve incoming calls. The interarrival times of new and handover calls are exponentially distributed with rate λ_F and λ_H, respectively. Let $\lambda = \lambda_F + \lambda_H$. Call durations (of new and handover calls) in the cell follow an exponential distribution with mean $1/\mu$. A blocked call due to the lack of capacity or the resource allocation policy (e.g.: the guard channel concept) will retry with probability θ and rate α_0. Note that θ is used to represent the degree of impatience of users. Throughout this Chapter, the orbit is defined as a collection of blocked calls which will repeat a request for service (see Figure 6.1, cf. [92, 121]). Note that retried calls are allowed to capture the guard channels similarly to the assumption in [92, 121].

The number of channels which are exclusively reserved for handover calls is given by n_H. If the number of occupied channels is larger than $c - n_H$, only an arriving handover call is accepted (Figure 6.1).

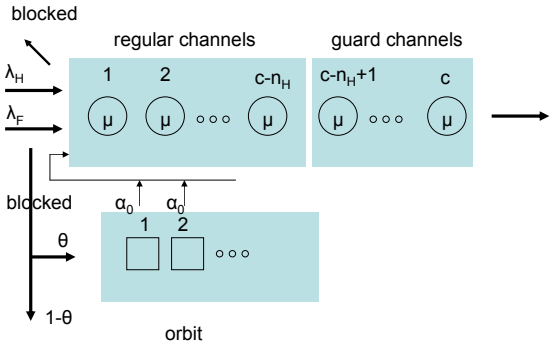


Figure 6.1: A retrial queueing model

The system is described by a two-dimensional continuous time Markov process $(I(t), J(t))$, where $I(t)$ $(0 \leq I(t) \leq c)$ represents the number of occupied channels and $J(t)$ $(0 \leq J(t))$ is the number of customers waiting for reattempt at time t. The steady state probabilities are denoted by $p_{i,j} = \lim_{t\to\infty} \mathbb{P}(I(t) = i, J(t) = j)$. We introduce $\mathbf{v}_j = (p_{0,j}, \ldots, p_{c,j})$. It is worth emphasizing that the system is analytically intractable. Therefore, approximation approaches should be applied to evaluate the performance of the system.

6.3 Approximation Procedures

6.3.1 The TGM Algorithm

Tran-Gia and Mandjes [121] applied the truncation of the state space (i.e.: from $\{0, \ldots, c\} \times \{0, 1, \ldots\}$ to $\{0, \ldots, c\} \times \{0, 1, \ldots, q\}$ for an appropriate large q value). It is observed that all probabilities can be written as functions of $p_{0,q} = K_0$ with the use of the appropriate order of the balance equations. Therefore, a nearly recursive algorithm (we refer to it as the TGM algorithm) is applied to express the steady state probabilities in K_0. Next, the normalization equation is used to calculate K_0 and the steady state probabilities. However, there exists a problem of numerical instability concerning the TGM algorithm because negative terms are involved in the computation and K_0 is extremely small for a number of cases (see Section 5.4).

6.3.2 The AJM Method

Ajmone Marsan et al. [92] proposed several approximated Continuous Time Markov Chain (CTMC) models with one and two Boolean flags for the problem (throughout this Chapter, the term AJM approach is used to refer to the models defined by Ajmone Marsan et al.). The key ideas of the AJM approach are as follows. (i) The states of the approximate CTMCs are

described by the number of active calls in the cell and one (or two) Boolean variable(s) indicating the presence of blocked calls. (ii) The estimation of the average number ($E(J)$) of blocked calls in the system is based on an assumption that the number of customers in the orbit follows a geometric distribution. The retrials are approximated with an interrupted Poisson process with rate proportional to $E(J)$. (iii) The rate of retrials which leave the orbit empty and the rate of retrials which leave the orbit nonempty is estimated from P (the steady-state probability that a departing customer leaves the orbit nonempty) and $E(J)$.

A CTMC with a reduced state space can be constructed, for which the knowledge of P and $E(J)$ is needed. Therefore, an iterative procedure was proposed to compute P and $E(J)$ (cf. [92]).

6.3.3 A New Computational Algorithm

In this section we present a new computational algorithm. In order to ease the comprehension, we show the computational procedure for $n_H = 1$. The same algorithm can be applied with the modification of appropriate elements in transition matrices defined below for $n_H > 1$. We approximate the original system (described in Section 6.2 where the individual retrial rate of each call is α_0) with a system where rate α of all repeated calls is independent of the number of calls in the orbit. The computation of α is based on the iteration, which will be presented at the end of this Section.

6.3.3.1 Notations

A Quasi-Birth-and-Death representation of the approximated system is introduced with the following notations.

- $A_j(i,k)$ denotes a transition rate from state (i,j) to state (k,j) $(0 \leq i,k \leq c; j = 0,1,\ldots)$, which is caused by either the departure of a call after service or the arrival of a call. For $j \geq 0$, matrix A_j is defined as the matrix with elements $A_j(i,k)$. Since A_j is j-independent, it can be written as

$$A_j = A = \begin{bmatrix} 0 & \lambda & 0 & \ldots & 0 & 0 & 0 \\ \mu & 0 & \lambda & \ldots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & & \ldots & (c-1)\mu & 0 & \lambda_H \\ 0 & 0 & & \ldots & 0 & c\mu & 0 \end{bmatrix}.$$

- $B_j(i,k)$ represents one step upward transition from state (i,j) to state $(k,j+1)$ $(0 \leq i,k \leq c; j = 0,1,\ldots)$, which is due to a call joining the orbit. In the similar way, matrix B_j (B) with elements $B_j(i,k)$ is

defined as

$$B_j = B = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 & 0 & 0 \\ 0 & 0 & 0 & \dots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & & \dots & 0 & \lambda_F\theta & 0 \\ 0 & 0 & & \dots & 0 & 0 & \lambda_F\theta \end{bmatrix} \quad \forall j \geq 0.$$

- $C_j(i,k)$ is the transition rate from state (i,j) to state $(k,j-1)$ $(0 \leq i,k \leq c; j = 1,\dots)$, which is due to a call which leaves the orbit. Matrix C_j $(\forall j \geq 1)$ with elements $C_j(i,k)$ is written as

$$C_j = C = \begin{bmatrix} 0 & \alpha & 0 & \dots & 0 & 0 \\ 0 & 0 & \alpha & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & & \dots & (1-\theta)\alpha & \alpha \\ 0 & 0 & & \dots & 0 & (1-\theta)\alpha \end{bmatrix}.$$

D^A and D^C are diagonal matrices whose diagonal elements are the sum of the elements in the row of A and C. The infinitesimal generator matrix of Y is given by

$$\begin{bmatrix} A_{00} & B & 0 & \dots & \dots & \dots & \dots \\ C & Q_1 & B & 0 & \dots & \dots & \dots \\ 0 & C & Q_1 & B & 0 & \dots & \dots \\ 0 & 0 & C & Q_1 & B & 0 & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \end{bmatrix},$$

where $A_{00} = A - D^A - B$ and $Q_1 = A - D^A - B - D^C$.

The balance equations are written as

$$\mathbf{v}_0 A_{00} + \mathbf{v}_1 C = 0, \tag{6.1}$$

$$\mathbf{v}_{j-1} B + \mathbf{v}_j Q_1 + \mathbf{v}_{j+1} C = 0 \ \ (j \geq 1). \tag{6.2}$$

In addition, the normalization equation is

$$\sum_{i=0}^{c} \sum_{j=0}^{\infty} p_{i,j} = 1. \tag{6.3}$$

Since the system is described by a homogeneous Quasi-Birth and Death process, the steady state probabilities can be obtained with the existing methods like the matrix-geometric and its variants [14, 84, 99], or the spectral

expansion [47, 97]. However, the existing methods have the state space explosion problem (no results due to a very long-running time of computer programs implementing these methods) and numerically instable due to the ill-conditioned linear equations (which are needed to solve for the steady-state probabilities) when c is large (the problem starts when c reaches a value of several hundreds).

6.3.3.2 A Spectral Expansion Solution

The characteristic matrix polynomial associated with equations (6.2) is $Q(x) = B + Q_1 x + Cx^2$. Assume that $Q(x)$ has d pairs of eigenvalue-eigenvectors $(x_i, \boldsymbol{\psi}_i)$, thus satisfying the equation:

$$\boldsymbol{\psi}_i Q(x_i) = 0; \;\; det[Q(x_i)] = 0 \;\text{ for } i = 0, \ldots, d-1. \tag{6.4}$$

For the k^{th} $(k = 0, \ldots, d-1)$ eigenvalue-eigenvector pair, $(x_k, \boldsymbol{\psi}_k)$, by substituting $\mathbf{v}_j = \boldsymbol{\psi}_k x_k^j$ in the equations (6.2), this set of equations is satisfied. This means $\mathbf{v}_j = \boldsymbol{\psi}_k x_k^j$ is a particular solution. As a consequence, the general solution for $\mathbf{v}_j$ is a linear sum of all the factors $(\boldsymbol{\psi}_k x_k^j)$. That is, we can write

$$\mathbf{v}_j \;\; = \;\; \sum_{k=0}^{d} b_k x_k^j \boldsymbol{\psi}_k \; (j \geq 0), \tag{6.5}$$

where $(x_0, \boldsymbol{\psi}_0)$, ..., $(x_d, \boldsymbol{\psi}_d)$ are left-hand-side eigenvalue-eigenvector pairs of $Q(x)$ and $b_0, \ldots, b_d$ are coefficients to be determined. In order to satisfy the normalization equation (6.3), only $|x_i| < 1$ should be considered. Without the loss of generality, we let d is the number of these eigenvalues x_i. The number of coefficients $b_0, \ldots, b_d$ to be determined is $d + 1$. We can utilize equation (6.1) and the normalization equation to compute the coefficients. Thus, the number of linear equations is $c+2$. However, only $c+1$ are linearly independent. Therefore, the linear equations have a unique solution if and only if $c = d$. This means, the exact number of eigenvalues $(|x_i| < 1)$ is $c+1$.

In summary, the first step of the spectral expansion (SE) method [97] finds the eigenvalues and eigenvectors (after the linearization of the problem to the generalized eigenproblem, the implicitly restarted Arnoldi method [85] is applied to the generalized eigenproblem). Then, coefficients $b_0, \ldots, b_c$ are to be computed as the solution of linear equations based on equation (6.1) and the normalization equation. However, the SE method is numerically instable[1] if c is large because of the state explosion problem. It is worth emphasizing that the methods [14, 84, 99] have the same problem as well. In section 6.3.3.3 we propose a new computational algorithm which does not suffer the problem mentioned above.

[1] then incorrect results are produced.

6.3.3.3 Proposed Computational Algorithm

In the present Chapter, $Q(x)$ is a tridiagonal matrix

$$Q(x) = \begin{bmatrix} q_{0,0}(x) & q_{0,1}(x) & 0 & \cdots & 0 & 0 \\ q_{1,0}(x) & q_{1,1}(x) & q_{1,2}(x) & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & q_{c-1,c-2}(x) & q_{c-1,c-1}(x) & q_{c-1,c}(x) \\ 0 & 0 & & \cdots & q_{c,c-1}(x) & q_{c,c}(x) \end{bmatrix},$$

where the elements of $Q(x)$ are expressed as

$$\begin{aligned} q_{0,0}(x) &= -(\lambda+\alpha)x, \\ q_{i,i}(x) &= -(\lambda+\alpha+i\mu)x \quad (i=1,\ldots,c-2), \\ q_{i,i+1}(x) &= \lambda x+\alpha x^2 \quad (i=1,\ldots,c-2), \\ q_{i,i-1}(x) &= i\mu x \quad (i=1,\ldots,c), \\ q_{c-1,c}(x) &= \lambda_H x+\alpha x^2, \\ q_{c-1,c-1}(x) &= \lambda_F\theta-(\lambda_F\theta+\lambda_H+(c-1)\mu)x \\ &\quad -(\alpha+(1-\theta)\alpha)x+(1-\theta)\alpha x^2, \\ q_{c,c}(x) &= \lambda_F\theta-(\lambda_F\theta+c\mu+(1-\theta)\alpha)x \\ &\quad +(1-\theta)\alpha x^2. \end{aligned}$$

It is observed that only $q_{c-1,c-1}(x)$ and $q_{c,c}(x)$ is not divideable by x amongst the elements of the characteristic polynomial. Thus $Q(x)$ has $c-1$ zero-eigenvalues ($x_0 = x_1 = \ldots = x_{c-2} = 0$). The corresponding independent eigenvectors for $c-1$ zero-eigenvalues are $\boldsymbol{\psi}_0 = \{1,0,\ldots,0\}$, $\boldsymbol{\psi}_1 = \{0,1,0,\ldots,0\}$,...,$\boldsymbol{\psi}_{c-2} = \{0,0,\ldots,1,0,0\}$. As a consequence, the steady state probabilities can be expressed as follows

$$\mathbf{v}_j = b_{c-1}\boldsymbol{\psi}_{c-1}x_{c-1}^j + b_c\boldsymbol{\psi}_c x_c^j \ (j \geq 1), \tag{6.6}$$

$$\mathbf{v}_0 = \sum_{k=0}^{c} b_k\boldsymbol{\psi}_k = \mathbf{b} + b_{c-1}\boldsymbol{\psi}_{c-1} + b_c\boldsymbol{\psi}_c, \tag{6.7}$$

where $\mathbf{b} = \sum_{k=0}^{c-2} b_k\boldsymbol{\psi}_k = \{b_0, b_1, \ldots, b_{c-2}, 0, 0\}$.

Remarks. Since $x_{c-1} \neq x_c$ holds in order to ensure that eigenvectors $\boldsymbol{\psi}_k$ ($k = 1,\ldots,c$) are orthogonal, it is readily observed from (6.6) that the probability distribution of the number of calls in the orbit is the "mixture" of geometric distributions when $n_H = 1$ holds, which does not confirm the assumption [92] that the number of customers in the orbit follows a geometric distribution (it is only when $n_H = 0$). It can also be proved that the number of zero-eigenvalues of $Q(x)$ is $c - n_H$ for $n_H \geq 1$.

$Q(x_i)$ ($i = c-1, c$) is a tridiagonal matrix and $q_{k,k}(x_i) \neq 0$ (for $k = 0,\ldots,c-2$). Hence, the component matrices of the LU decomposition of

$Q(x_i)$ are written as

$$L(x_i) = \begin{bmatrix} l_1(x_i) & 0 & 0 & \dots & 0 & 0 & 0 \\ \mu x_i & l_2(x_i) & 0 & \dots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & & \dots & (c-1)\mu x_1 & l_c(x_i) & 0 \\ 0 & 0 & & \dots & 0 & c\mu x_i & l_{c+1}(x_i) \end{bmatrix}, \quad (6.8)$$

$$U(x_i) = \begin{bmatrix} 1 & u_1(x_i) & \dots & 0 & 0 & 0 & 0 \\ 0 & 1 & u_2(x_i) & \dots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & & \dots & 0 & 1 & u_c(x_i) \\ 0 & 0 & & \dots & 0 & 0 & 1 \end{bmatrix}, \quad (6.9)$$

where $l_j(x_i)$ $(j = 1, \dots, c+1)$ and $u_j(x_i)$ $(j = 1, \dots, c)$ are the elements of $L(x_i)$ and $U(x_i)$, respectively. After a simple algebra, it can be written as

$$\begin{aligned} l_1(x_i) &= q_{1,1}(x_i) = -(\lambda + \nu)x_i, \\ l_j(x_i) + \alpha(j-1)\mu x_0 u_{j-1}(x_0) &= q_{j,j}(x_i), \quad (j = 2, \dots, c+1), \\ l_j(x_i)u_j(x_i) &= \lambda x_i + \nu x_i^2, \quad (j = 1, \dots, c). \end{aligned} \quad (6.10)$$

Therefore, the determinant of $Q(x_i)$ is expressed as

$$Det[Q(x_i)] = Det[L(x_i)]Det[U(x_i)] = \prod_{j=1}^{c+1} l_j(x_i). \quad (6.11)$$

As the consequence of equation (6.10), $l_j(x_i) \neq 0$ holds for $(1 < j < c)$ $i = c-1, c$. Hence, $Det[Q(x_i)] = 0$ follows $l_c(x_i)l_{c+1}(x_i) = 0$, which means x_i is the root of $l_c(x)l_{c+1}(x)$. To compute the roots of $l_c(x)l_{c+1}(x)$ in interval $(0, 1)$, several algorithms can be applied. In this Chapter, we use the Brent method [19] to find x_i in interval $(0, 1)$.

For each eigenvalue x the corresponding eigenvector $\boldsymbol{\xi}(x) = \{\xi_0(x), \xi_1(x), \dots, \xi_c(x)\}$ can be determined by equation $\boldsymbol{\xi}(x)Q(x) = 0$ and $\xi_0(x) = 1$ (because multiplying an eigenvector with a scalar number results in an eigenvector).

$$\begin{aligned} 1 &= \xi_0(x), \\ 0 &= \xi_0(x)q_{0,0}(x) + \xi_1(x)q_{1,0}(x), \\ 0 &= \xi_{i-1}(x)q_{i-1,i}(x) + \xi_i(x)q_{i,i}(x) + \xi_{i+1}(x)q_{i+1,i}(x) \\ &\quad i = 1, \dots, c-1, \\ 0 &= \xi_{c-1}(x)q_{c-1,c}(x) + \xi_c(x)q_{c,c}(x). \end{aligned} \quad (6.12)$$

After a simple algebra on equation (6.1), we obtain

$$\mathbf{b} = -b_{c-1}\boldsymbol{\psi}_{c-1} - b_c\boldsymbol{\psi}_c - (b_{c-1}\boldsymbol{\psi}_{c-1}x_{c-1} + b_c\boldsymbol{\psi}_c x_c)A_{00}^{-1}. \quad (6.13)$$

That is, $b_0, \ldots, b_{c-2}$ are expressed in b_{c-1} and b_c (it is worth emphasizing that the inverse of A_{00} can be efficiently obtained by the recursive formula in [122] because A_{00} is a tridiagonal matrix). Moreover, the last two elements of vector $\mathbf{b}$ are zero. Hence, b_{c-1} can be expressed in b_c as well with setting one of these last two elements to zero (note that these two equations are not independent). Let $\boldsymbol{e}$ denote a column vector of size $c+1$ with all elements equal to 1. Utilizing the normalization equation

$$1 = \sum_{i=0}^{c} \sum_{j=0}^{\infty} p_{i,j} = (\mathbf{b} + \frac{b_{c-1}}{1 - x_{c-1}} \boldsymbol{\psi}_{c-1} + \frac{b_c}{1 - x_c} \boldsymbol{\psi}_c).\mathbf{e}, \tag{6.14}$$

b_c can be computed first. Then coefficients b_i $(i = c-1, \ldots, 0)$ can be determined.

All the steps of the proposed computational procedure are summarized in Algorithm 6.1.

Algorithm 6.1 The proposed computation algorithm

{α_0 is the retrial rate of each call in the orbit}
{α is the retrial rate of the approximated model}
{$E_n(J)$ is the average number of calls in the orbit in the approximated model}
$\alpha = \alpha_0$
repeat
  Compute x_{c-1} and x_c (i.e the roots of $l_c(x_i)l_{c+1}(x_i) = 0$ based the Brent algorithm)
  Compute $\boldsymbol{\psi}_{c-1}$ and $\boldsymbol{\psi}_c$
  Express $\mathbf{b}$ based on (6.13)
  Express b_{c-1} in b_c
  Compute b_c based on the normalization equation
  Compute all b_i $(i = c-1, \ldots, 1)$
  Compute $E_n(J)$
  $\alpha = \alpha_0 E_n(J)$
until $E_n(J)$ converges

6.4 Numerical Results

In the following, we present a comparison between the original TGM approach, the AJM method and the proposed algorithm. Note that the recursive algorithm proposed by Tran-Gia and Mandjes requires some algebraic steps. Therefore it is most suitably implemented with the use of a computer algebra system (CAS) which allows the manipulation

of mathematical expressions in symbolic form (it is possible to use the conventional programming languages like C, C++, etc to implement the algorithm, however it is tiresome work to make algebraic steps). In the present Chapter, we apply the CAS tool `Mathematica` (http://www.wolfram.com) for the implementation of the TGM algorithm. We programmed the AJM approach based on a CMTC model with one flag.

Similarly to the numerical example[2] in [121], we use the following parameter values unless stated otherwise: $c = 15$, $1/\mu = 120$ s, $\alpha_0/\mu = 20$, $\lambda_F/\lambda_H = 24$ and $\theta = 1.0$. The offered load is defined by $\rho = \lambda/(c\mu)$. All the computations are performed with machine precision 10^{-16}.

6.4.1 The Numerical Problem of the TGM Algorithm

We report the obtained results versus q concerning the TGM approach in Table 6.1. It is observed that the recursive algorithm for the main M/M/m retrial queue converges from an appropriate large value of q, while the TGM approach gives ill results at a large value of q.

6.4.2 The Lower Bound and Upper Bound of The Average Number of Busy Channels

Now we investigate the average number of busy channels (i.e. $E(I)$) in the system described in Section 6.2). We have the following intuitive observations concerning $E(I)$ when $\theta = 1.0$ holds.

- $E(I)$ is greater than or equal to $E_{lower}(I)$ which is defined as the average number of busy servers of the considered system with appropriate setting of input parameters such that there is no handover call in the system ($\lambda_H = 0$) and the guard channel is not used to accept either new or retried calls. That means, the system now is modeled by the classical main M/M/m retrial queue with $m = c - 1$ and the arrival rate equal to λ_F.

- $E(I)$ is upper-bounded by $E_{upper}(I)$ which is the average number of busy servers of the classical main M/M/m retrial queue with $m = c$ and the arrival rate equal to λ.

In Table 6.2, $E(I)$ values computed by our method ($E_n(I)$), the TGM algorithm and the AJM approach, are compared against $E_{lower}(I)$ and $E_{upper}(I)$, which shows that our method produces results within the lower and upper bound, and close to the simulation results. The results obtained

Table 6.1: Results by the TGM approach vs q

q	$\theta = 1.0$, $\rho = 0.9$			
	$Pr_{TGM}(I = c-1)$	$Pr_{TGM}(I = c)$	$E_{TGM}(I)$	$E_{TGM}(J)$
5	0.0979144	0.060649	9.71497	0.145959
6	0.0978954	0.0606622	9.71492	0.146905
7	0.0978893	0.0606671	9.71491	0.147343
8	0.0978872	0.060669	9.71491	0.147546
9	0.0978864	0.0606698	9.71491	0.14764
10	0.0978861	0.0606701	9.71491	0.147683
15	0.0978859	0.0606703	9.71491	0.14772
18	0.0980308	0.0607493	9.71779	0.147921
20	0.0656716	0.0441439	7.80233	0.106539
25	0.0072863	0.0000391481	7.33665	-0.00144331
30	-0.0000952519	0.0000255201	4.82715	-0.000411096
35	-0.0335317	-0.0000583539	2.06684	-0.0111385
40	-0.015773	0.00159368	5.30805	-0.00615581
45	-0.0159268	0.0146501	10.0892	-0.0463782
50	0.00207165	$-4.64961*10^{-8}$	5.96606	0.00339289
60	-0.0292324	-0.00370845	5.0368	0.0016369
70	0.00368597	-0.00691	9.72773	0.00775322
80	0.0013226	-0.00065035	5.98353	0.00231422
90	-0.0916223	-0.0000688937	5.31296	-0.0419629
100	-0.0663172	0.00352705	2.99921	-0.0293509

Table 6.2: $E(I)$ –the average number of busy servers– compared with the thresholds

ρ	$E_{lower}(I)$	new method	$E_{upper}(I)$	TGM	AJM	Simulation
0.1	1.344	1.5	1.5	1.5	1.5	1.499949
0.2	2.688	3.0	3.0	3.0	2.99999	3.0000
0.3	4.032	4.49999	4.5	4.49964	4.49923	4.499798
0.4	5.376	5.99979	6.0	5.98881	5.98708	5.999880
0.5	6.72	7.49805	7.5	7.39322	7.41707	7.497120
0.6	8.064	8.99012	9.0	8.52668	8.70443	8.989000
0.7	9.048	10.4666	10.5	9.24279	9.77815	10.453600
0.8	10.752	11.9143	12.0	9.59074	10.6223	11.884100
0.9	12.096	13.3194	13.5	9.71491	11.2653	13.250500

by the TGM and AJM method is only close to our method and within the bounds when the load is moderate or small.

6.4.3 Comparison with Simulation

We have built a simulation for the original and analytically intractable system described in Section 6.2, where the retrial rate depends[3] on the number of waiting calls in the orbit. In Table 6.3 we present some illustrative results to show the accuracy of the new method concerning to the blocking probabilities of calls and the average number of occupied channels. Note that the simulation results are generated with the confidential level of 99%, and the ratio of the half-width of the confidence interval and the mean of collected observations is 0.099%.

Table 6.3: Comparison with simulation

ρ	$\theta = 0.1$							
	Simulation				Analytical			
	$E_s(I)$	$E_s(J)$	$\mathbb{P}_s(I = c-1)$	$\mathbb{P}_s(I = c)$	$E_n(I)$	$E_n(J)$	$\mathbb{P}_n(I = c-1)$	$\mathbb{P}_n(I = c)$
0.1	1.500250	0.000000	0.000000	0.000000	1.5	$5.6\ 10^{-13}$	$7.5\ 10^{-10}$	$6.5\ 10^{-12}$
0.2	3.000170	0.000000	0.000002	0.000000	2.99999	$4.2\ 10^{-9}$	0.000002	$4.8\ 10^{-8}$
0.3	4.500090	0.000000	0.000161	0.000005	4.49925	$4.5\ 10^{-7}$	0.000178	0.000005
0.4	5.992940	0.000000	0.002161	0.000091	5.98737	0.00000	0.002236	0.000078
0.5	7.418670	0.000485	0.011387	0.000611	7.41839	0.000046	0.011449	0.000502
0.6	8.712900	0.001767	0.034116	0.002205	8.70786	0.000165	0.033841	0.001781
0.7	9.796540	0.004544	0.071726	0.005546	9.78445	0.000408	0.070401	0.004325
0.8	10.649000	0.008906	0.118981	0.010615	10.6317	0.000784	0.116733	0.008202
0.9	11.302100	0.014681	0.170403	0.017407	11.2777	0.001282	0.167009	0.013212
	$\theta = 0.3$							
0.1	1.500250	0.000000	0.000000	0.000000	1.5	$1.9\ 10^{-12}$	$7.5\ 10^{-10}$	$1.5\ 10^{-11}$
0.2	3.000600	0.000000	0.000002	0.000000	2.99999	$1.5\ 10^{-8}$	0.0000027	$1.1\ 10^{-7}$
0.3	4.500360	0.000000	0.000164	0.000011	4.49932	0.0000015	0.000178	0.000011
0.4	5.994750	0.000000	0.002234	0.000224	5.99969	0.001129	0.002387	0.0012942
0.5	7.431050	0.001831	0.011728	0.001494	7.4239	0.000174	0.011521	0.0011505
0.6	8.750470	0.007213	0.035735	0.005646	8.72407	0.000643	0.034040	0.0040884
0.7	9.862670	0.019015	0.074871	0.014397	9.78445	0.000408	0.070401	0.0043254
0.8	10.760900	0.038974	0.123618	0.028256	10.6317	0.000785	0.116733	0.0082025
0.9	11.455700	0.067699	0.175191	0.046835	11.2777	0.001282	0.167009	0.0132115
	$\theta = 1.0$							
0.1	1.500250	0.00	0.00	0.000	1.5	$1.99\ 10^{-11}$	$7.5\ 10^{-10}$	$7.4\ 10^{-11}$
0.2	3.000920	0.000000	0.000003	0.000000	3.0	$2.5\ 10^{-7}$	0.000003	$6.0\ 10^{-7}$
0.3	4.500000	0.000073	0.000181	0.000043	4.49999	0.000041	0.000186	0.0000669
0.4	5.999880	0.001795	0.002508	0.000912	5.99969	0.001129	0.002387	0.0012942
0.5	7.497120	0.017323	0.014018	0.007419	7.49959	0.011952	0.012508	0.0097432
0.6	8.989000	0.095756	0.043888	0.033434	8.99538	0.071775	0.041080	0.0373981
0.7	10.453600	0.379311	0.091433	0.102135	10.4503	0.305958	0.0761227	0.118219
0.8	11.884100	1.244660	0.140123	0.239130	11.8743	1.09335	0.167975	0.20876
0.9	13.250500	4.081900	0.155641	0.468458	13.2458	4.22319	0.153844	0.470658

[2]Note that the results of the TGM approximation was not compared with simulation [121].

[3]It is the only difference between the simulation and the proposed approach is the retrial rate of calls waiting in the orbit.

6.4.4 Computation with a Large Value of c

In this Section we compare the computational cost of the AJM and our proposed method. Note that the computation time by the TGM method is not compared due to its numerical instability. It is also worth mentioning that Domenech-Benlloch et al. [56] proposed the generalized truncation approach for the main retrial queue M/M/c (without guard channels) based on the homogenization of the problem and the matrix-geometric approach. The approach [56] can be appropriately modified to handle the current problem (i.e. retrial queue with guard channels). However, it certainly takes more time to get results[4] than the AJM method since the approach by Domenech-Benlloch et al. [56] has to compute the rate matrix and solve a significant number of linear equations. In addition, the approach [56] needs a significant amount of memory to store matrices during the computation.

In Table 6.4, we report the computation times needed by different methods (the results are obtained in a machine with processor Quad-Core Xeon E5410 2.33GHz running Linux) versus c with $\rho = 0.9$, $1/\mu = 120$ s, $\alpha_0/\mu = 20$, $\lambda_F/\lambda_H = 24$ and $\theta = 1.0$, which show that the new method is very fast and efficient in the term of memory usage.

Table 6.4: Computational time in seconds

c	AJM	New method
500	79	16
1000	334	70
1500	693	178

6.5 Conclusions

We have presented a new stable algorithm in order to deal with the numerical problems concerning the computation of the steady state probabilities of wireless systems with retrial calls and guard channels. We have also shown that the distribution of the number of calls in the orbit of the approximate queueing model is the sum of the corresponding components of geometric sequences. The numerical stability, the small memory requirement for the implementation of the algorithm and the capability to obtain results for a large number of channels are the strengths of the new approach.

[4]It is also stated in [56] that their approach results in an increase in the computational cost compared to the AJM approach

7

An Efficient Computation Algorithm for a Multi-Server Feedback Retrial Queue with a Large Queueing Capacity[‖]

7.1 Introduction

There are two main methods to find the steady state probabilities for QBD processes on semi-infinite strips. The matrix-geometric [102] method (which is widely used to analyze queues [18, 87]) and its variants [82, 101] are numerical approaches to recursively compute the rate matrix (the minimal nonnegative matrix solution) of the matrix quadratic equation. The spectral expansion method is based on the eigenvalues and eigenvectors of the characteristic matrix polynomial [47, 97]. It is worth emphasizing that there have been no works which consider or develop an algorithm for queues involving zero-eigenvalues (see [28, 30, 33, 67, 97] for more detail). In this Chapter, we deal with an algorithm to find the eigenvalues of the matrix quadratic equation of a tridiagonal form based on the sign variations of the Sturm sequences. We consider a case where multiple zero-eigenvalues are involved.

The example for investigation is the M/M/c/N+c feedback queue with constant retrial rate, which is solved by the matrix-geometric method in [80]. However, the existing approaches face the state explosion problem when the queueing capacity $(N + c)$ is large. We prove that the number of zero-eigenvalues of the characteristic matrix polynomial associated with the balance equation is $\lfloor (N + c + 2)/2 \rfloor$. As a consequence, the remaining eigenvalues inside the unit circle can be computed in a quick manner based on the Sturm sequences. It is worth emphasizing that the algorithm of [67] should be slightly modified in order to determine the remaining eigenvalues of the characteristic matrix polynomial inside the unit circle. Numerical

[‖]Tien Van Do. "An Efficient Computation Algorithm for a Multiserver Feedback Retrial Queue with a Large Queueing Capacity". *Applied Mathematical Modelling*. http://dx.doi.org/10.1016/j.apm.2009.10.025

results are presented to compare computation times which are needed by the matrix geometric method, the pure spectral expansion approach, the method proposed by Naoumov et al. and the new algorithm to calculate the steady state probabilities. The comparison clearly demonstrates the advantage of the new algorithm on the computation of the steady state probabilities of the queue with a large queueing capacity.

The rest of this Chapter is organized as follows. In Section 7.2, the M/M/c/N+c feedback queue with constant retrial rate is described. A main theoretical result concerning the number of zero-eigenvalues of the characteristic matrix polynomial associated with the balance equation and the modified algorithm to calculate the eigenvalues is presented in Section 7.3. Numerical results are demonstrated in Section 7.4. Finally, Section 7.5 concludes the Chapter.

7.2 A System Description

The M/M/c/N+c feedback queue with constant retrial rate has a limited waiting position of size N and c homogeneous servers. Service times are exponentially distributed with parameter μ. External customers arrive according to a Poisson process with rate λ. Upon the arrival, an external customer

- either is served if there is a free server,
- or occupies a waiting position if all c servers are busy and there is a free waiting position,
- or is blocked and is forced to leave the system forever if all c servers are busy and waiting positions are occupied.

When one of the servers becomes free, the customer in the first waiting position immediately starts getting served.

A customer who leaves a system after service either join the retrial group (orbit) for another service with probability q $(0 \leq q < 1)$ or leave the system forever with probability $p = 1 - q$. Customers in the retrial group request service with constant retrial rate σ, which is independent of the number of customers in the retrial group [80]. Note that a customer in the orbit can enter the service facility at the retrial instant only when there are idle servers.

The system is completely described by two random variables. $I(t)$ $(0 \leq I(t) \leq c + N)$ is a random variable to denote the number of customers in the system (being served by servers or waiting in the waiting positions). $J(t)$ $(J(t) \geq 0)$ is a random variable to represent the number of customers in the retrial group. $Y = \{I(t), J(t)\}$ is a Continuous Time Markov Chain (CTMC) with a state space $\{0, 1, \ldots, c + N\} \times \{0, 1, \ldots\}$. We denote the

steady state probabilities by $\pi_{i,j} = \lim_{t\to\infty} \mathbb{P}(I(t) = i, J(t) = j)$, and introduce $\mathbf{v}_j = (\pi_{0,j}, \ldots, \pi_{c+N,j})$.

The evolution of Y is driven by the following transitions.

(a) $A_j(i,k)$ denotes a transition rate from state (i,j) to state (k,j) $(0 \leq i,k \leq n = c+N; j = 0,1,\ldots)$. These transitions happen due to either the arrival of an external customer or the departure of a customer from the system. $A_j(i,k)$ does not depend on j, so we can write

$$A_j(i,k) = A(i,k) = \begin{cases} p\mu \min(i,c) & \text{if } k = i-1 \text{ and } i \geq 1 \\ \lambda & \text{if } k = i+1 \\ 0 & \text{otherwise} \end{cases} . \quad (7.1)$$

(b) $B_j(i,k)$ represents one step upward transition from state (i,j) to state $(k,j+1)$ $(0 \leq i,k \leq n = c+N; j = 0,1,\ldots)$. These transitions are due to customers who join the retrial group.

$B_j(i,k)$ is independent of j, thus it is valid

$$B_j(i,k) = B(i,k) = \begin{cases} q\mu \min(i,c) & \text{if } k = i-1 \text{ and } i \geq 1 \\ 0 & \text{otherwise} \end{cases} . \quad (7.2)$$

(c) $C_j(i,k)$ is the transition rate from state (i,j) to state $(k,j-1)$ $(0 \leq i,k \leq n = c+N; j = 1,\ldots)$. These transitions are initiated by requests from the retrial group. $C_j(i,k)$ does not depend on j, so we can write

$$C_j(i,k) = C(i,k) = \begin{cases} \sigma & \text{if } k = i+1 \text{ and } i < c \\ 0 & \text{otherwise} \end{cases} . \quad (7.3)$$

$A(i,k)$, $B(i,k)$ and $C(i,k)$ are the elements of A, B and C matrices, respectively. We introduce diagonal matrices D^A, D^B and D^C. The diagonal elements are the sum of the elements in the row of A, B and C. The infinitesimal generator matrix of Y can be written as follows

$$\begin{bmatrix} A_{00} & Q_0 & 0 & \ldots & \ldots & \ldots & \ldots \\ Q_2 & Q_1 & Q_0 & 0 & \ldots & \ldots & \ldots \\ 0 & Q_2 & Q_1 & Q_0 & 0 & \ldots & \ldots \\ 0 & 0 & Q_2 & Q_1 & Q_0 & 0 & \ldots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \end{bmatrix}, \quad (7.4)$$

where $A_{00} = A - D^A - D^B$, $Q_0 = B$, $Q_1 = A - D^A - D^B - D^C$ and $Q_2 = C$. The forms of the matrices are illustrated in Table 7.1.

$$A=\begin{bmatrix} 0 & \lambda & 0 & 0 & 0 & 0 \\ p\mu & 0 & \lambda & 0 & 0 & 0 \\ 0 & 2p\mu & 0 & \lambda & 0 & 0 \\ 0 & 0 & 2p\mu & 0 & \lambda & 0 \\ 0 & 0 & 0 & 2p\mu & 0 & \lambda \\ 0 & 0 & 0 & 0 & 2p\mu & 0 \end{bmatrix}; Q_0 = B = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ q\mu & 0 & 0 & 0 & 0 & 0 \\ 0 & 2q\mu & 0 & 0 & 0 & 0 \\ 0 & 0 & 2q\mu & 0 & 0 & 0 \\ 0 & 0 & 0 & 2q\mu & 0 & 0 \\ 0 & 0 & 0 & 0 & 2q\mu & 0 \end{bmatrix};$$

$$Q_2 = C = \begin{bmatrix} 0 & \sigma & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix};$$

$$Q_1 = \begin{bmatrix} -\lambda-\sigma & \lambda & 0 & 0 & 0 & 0 \\ p\mu & -\lambda-\mu-\sigma & \lambda & 0 & 0 & 0 \\ 0 & 2p\mu & -\lambda-2\mu & \lambda & 0 & 0 \\ 0 & 0 & 2p\mu & -\lambda-2\mu & \lambda & 0 \\ 0 & 0 & 0 & 2p\mu & -\lambda-2\mu & \lambda \\ 0 & 0 & 0 & 0 & 2p\mu & -2\mu \end{bmatrix};$$

$$Q(x) = \begin{bmatrix} (-\lambda-\sigma)x & \lambda x+\sigma x^2 & 0 & 0 & 0 & 0 \\ q\mu+p\mu x & (-\lambda-\mu-\sigma)x & \lambda x+\sigma x^2 & 0 & 0 & 0 \\ 0 & 2q\mu+2p\mu x & (-\lambda-2\mu)x & \lambda x & 0 & 0 \\ 0 & 0 & 2q\mu+2p\mu x & (-\lambda-2\mu)x & \lambda x & 0 \\ 0 & 0 & 0 & 2q\mu+2p\mu x & (-\lambda-2\mu)x & \lambda x \\ 0 & 0 & 0 & 0 & 2q\mu+2p\mu x & -2\mu x \end{bmatrix}.$$

Table 7.1: Matrices for $c = 2$ and $N = 3$

7.3 An Efficient Computational Algorithm

7.3.1 The Number of Zero Eigenvalues

For $j \geq 1$, the balance equations is written as follows

$$\mathbf{v}_{j-1}Q_0 + \mathbf{v}_j Q_1 + \mathbf{v}_{j+1}Q_2 = 0 \ \ (j \geq 1). \tag{7.5}$$

$Q(x) = Q_0 + Q_1 x + Q_2 x^2$ is defined as the characteristic matrix polynomial associated with equations (7.5). In the present Chapter, $Q(x)$ is a tridiagonal matrix

$$Q(x) = \begin{bmatrix} q_{0,0}(x) & q_{0,1}(x) & 0 & \dots & 0 & 0 & 0 \\ q_{1,0}(x) & q_{1,1}(x) & q_{1,2}(x) & \dots & 0 & 0 & 0 \\ 0 & q_{2,1}(x) & q_{2,2}(x) & q_{2,2}(x) & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & & \dots & q_{n-1,n-2}(x) & q_{n-1,n-1}(x) & q_{n-1,n}(x) \\ 0 & 0 & & \dots & 0 & q_{n,n-1}(x) & q_{n,n}(x) \end{bmatrix}$$

where

$$\begin{aligned}
q_{0,0}(x) &= -(\lambda+\sigma)x, \\
q_{i,i-1}(x) &= q\mu\min(i,c)+p\mu\min(i,c)x && (i=1,\ldots,n), \\
q_{i,i}(x) &= -(\lambda+\mu\min(i,c)+C(i,i+1))x && (i=1,\ldots,n-1), \\
q_{i,i+1}(x) &= \lambda x + C(i,i+1)x^2 && (i=0,\ldots,n-1), \\
q_{n,n}(x) &= -c\mu x.
\end{aligned}$$

The steady state probabilities are closely related to the eigenvalue-eigenvector pairs $(x_i, \boldsymbol{\psi}_i)$ of $Q(x)$, which satisfy $\boldsymbol{\psi}Q(x) = 0$ and $det[Q(x)] = 0$ (cf. [31, 97]). If the system is ergodic, then the number of eigenvalues of the characteristic polynomial $Q(x)$ with *a degree of* $n+1$, which are strictly inside the unit circle, has to be $n+1$ (see the proof in Section 2.2.1). So we can write

$$\mathbf{v}_j = \sum_{i=0}^{n} a_i x_i^j \boldsymbol{\psi}_i \quad (j \geq 0), \tag{7.6}$$

where x_i are the eigenvalues inside the unit circle. Coefficients a_i can be determined from the balance equation for level $j = 0$ and the normalization equation.

Theorem 7.1. *The number of zero-eigenvalues of* $Q(x)$ *is* $\lfloor (n+2)/2 \rfloor$.

Proof. We provide a proof with mathematical induction. Let $[Q(x)]_{\{0,\ldots,k\}}$ denote a submatrix formed by the first $k+1$ rows and columns of $Q(x)$. It is easy to verify that $det[Q(x)]_{\{0,1\}} = 0$ has one zero-root and $det[Q(x)]_{\{0,1,2\}} = 0$ has two zero-roots.

Assume that $det[Q(x)]_{\{0,\ldots,k-2\}} = 0$ $(k \geq 3)$ has $\lfloor k/2 \rfloor$ zero-roots and $det[Q(x)]_{\{0,\ldots,k-1\}} = 0$ has $\lfloor (k+1)/2 \rfloor$ zero-roots.

$[Q(x)]_{\{0,\ldots,k\}}$ is a tridiagonal matrix. Therefore we can write

$$det[Q(x)]_{\{0,\ldots,k\}} = q_{k,k}(x)det[Q(x)]_{\{0,\ldots,k-1\}} - q_{k,k-1}(x)q_{k-1,k}(x)det[Q(x)]_{\{0,\ldots,k-2\}}.$$

Note that $q_{k,k}(x)$ and $q_{k-1,k}(x)$ is divideable by x, while $q_{k,k-1}(x)$ is not. As a consequence, $det[Q(x)]_{\{0,\ldots,k\}} = 0$ has $\lfloor k/2 \rfloor + 1 = \lfloor (k+2)/2 \rfloor$ zero-roots. □

7.3.2 A Computational Algorithm

Following [67], $\boldsymbol{\psi}Q(x) = 0$ can be written as,

$$\begin{aligned}
0 &= \psi_0 q_{0,0}(x) + \psi_1 q_{1,0}(x), \\
0 &= \psi_{i-1} q_{i-1,i}(x) + \psi_i q_{i,i} x + \psi_{i+1} q_{i+1,i}(x) \quad (i=1,\ldots,n-1), \\
0 &= \psi_{n-1} q_{n-1,n} + \psi_n q_{n,n}(x),
\end{aligned}$$

where $\boldsymbol{\psi} = \{\psi_0, \ldots, \psi_n\}$.

We set $\psi_0 = 1$ and $q_{n+1,n} = 1 + x$, therefore

$$\begin{aligned} \psi_1(x) &= -q_{0,0}(x)/q_{1,0}(x), \\ \psi_{i+1}(x) &= -\frac{\psi_i(x)q_{i,i}(x) + \psi_{i-1}(x)q_{i-1,i}(x)}{q_{i+1,i}(x)} \qquad (i = 1, \ldots, n). \end{aligned} \tag{7.7}$$

It is proved by [67] that the sequence $\{\psi_i(x), i = 0, \ldots, n+1\}$ associated with the characteristic matrix polynomial of tridiagonal form is a Sturm sequence within a given interval if for any fixed x within this interval $\psi_0(x) = 1$ and $\psi_i(x) = 0$, $i = 1, \ldots, n$ implies $\psi_{i-1}(x)\psi_{i+1}(x) < 0$. Furthermore, the number of sign variations is defined by [67] as

$$nsv(x) = \#\{\psi_i(x)\psi_{i+1}(x) < 0, 0 \le i \le n\} + \#\{\psi_i(x) = 0, 0 \le i \le n\}. \tag{7.8}$$

Grassmann [67] has reported a divide-and-conquer procedure (called `getx` in Algorithm 7.1) to find eigenvalues inside the unit circle for QBD processes with the characteristic polynomial matrix of a tridiagonal form if they are all non-zero. Grassmann's algorithm discards any interval $(x_1, x_2]$ if $nsv(x_1) = nsv(x_2)$. To compute the eigenvalues in interval $(0, 1)$, it is proposed to start the algorithm with `getx(0,n+1,1,0)` (cf.: [67]). It is also mentioned [67] that handling zero-eigenvalues will be developed in future.

Algorithm 7.1 `getx` procedure

```
{Xeg is the vector of eigenvalues}
{ε is the required accuracy}
PROCEDURE getx(x1, nx1, x2, nx2)
if nx1 == nx2 then
  Return
end if
if x2 − x1 < ε then
  if nx1 == nx2 + 1 then
    Xeq_nx2 = x1
  end if
  Return
end if
x = (x1+x2)/2
nx = nsv(x)
Call getx(x1, nx1, x, nx)
Call getx(x, nx, x2, nx2)
END OF PROCEDURE getx
```

If the QBD process of the M/M/c/N+c feedback queue with constant retrial rate is ergodic, then the number of eigenvalues inside the unit circle is

$n+1$. We have proved that the number of zero-eigenvalues is $\lfloor (n+2)/2 \rfloor$. In order to deal with zero-eigenvalues, two modifications are needed.

- The modified function for the number of sign variations is defined as follows
$$mnsv(x) = \#\{\psi_i(x)\psi_{i+1}(x) < 0, 0 \leq i \leq n\}. \quad (7.9)$$
The new function for the number of sign variations should be applied inside the `getx` function as well.

- The modified initialization should be applied as illustrated in Algorithm 7.2 to find the remaining non-zero eigenvalues of $Q(x)$. Note that $mnsv(\epsilon) = n - \lfloor n/2 \rfloor$ because the number of other eigenvalues inside the unit circle is $n - \lfloor n/2 \rfloor$.

Note that for each eigenvalue, the corresponding eigenvector can be determined with equation (7.7), then the steady state probabilities can be computed.

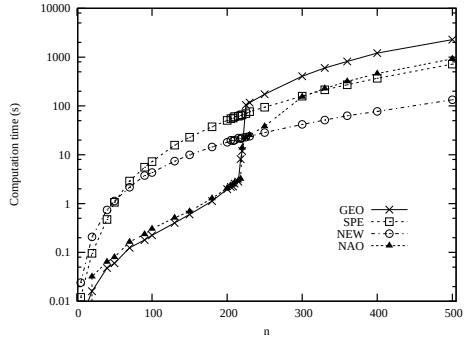
Algorithm 7.2 Initialization to find $n - \lfloor n/2 \rfloor$ non-zero eigenvalues inside the unit circle

{ ϵ is the required accuracy}
{Xeg is the vector of eigenvalues indexed from 0 to $n - \lfloor n/2 \rfloor$}
$x_1 := \epsilon$
$x_2 := 1 - \epsilon$
$nx_1 := mnsv(x_1)$
$nx_2 := mnsv(x_2)$
Call getx(x_1, nx_1, x_2, nx_2)

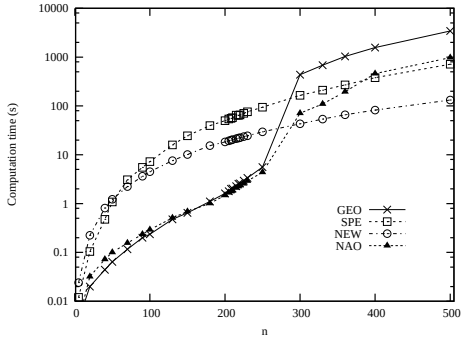
7.4 Numerical Results

In this Section, we illustrate the efficiency of the computation method for a large queueing capacity vs other methods (direct computation of the eigenvalues of the characteristic matrix polynomial and the matrix geometric method). As already analyzed in [47], both the spectral expansion and the matrix-geometric method have the same complexity of solving the unknowns after the computation of eigenvalue-eigenvectors and the rate matrix. Therefore, we compare the computational time needed to obtain the eigenvalues/eigenvectors (for the new method and the direct calculation of the eigenvalues) and the rate matrix in the case of the matrix-geometric method. Four methods are compared in this section: the original spectral expansion method [31], the successive substitution procedure

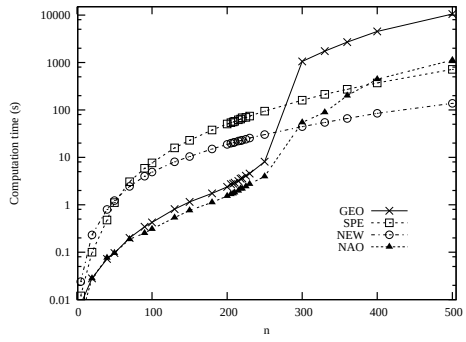
of the matrix-geometric method [102] proposed by Kumar et al. [80] for the M/M/c/N+c feedback queue with constant retrial rate, the variant of the matrix-geometric method [101] (which is considered as one of the best and latest improvements for the original matrix-geometric method) and the new algorithm.



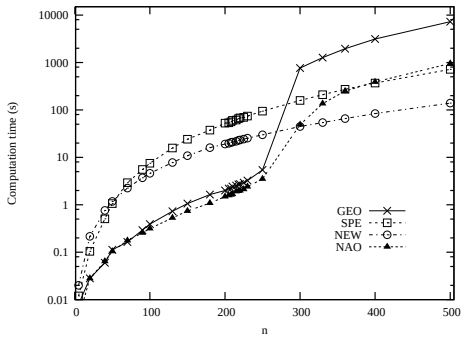
(a) $\lambda = 0.1$



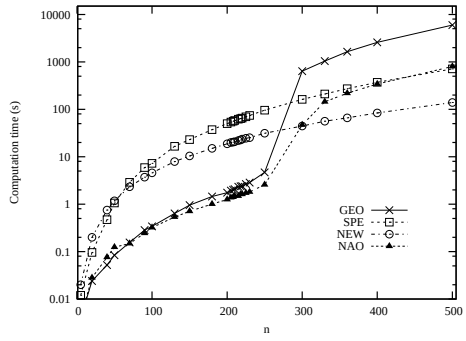
(b) $\lambda = 0.2$



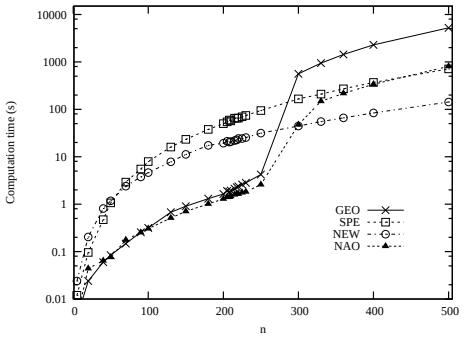
(c) $\lambda = 0.5$



(d) $\lambda = 0.9$



(e) $\lambda = 0.95$



(f) $\lambda = 0.99$

Figure 7.1: Computational time in seconds

For the numerical study, we choose the following parameters $N = 3$, $\sigma = 2.0$, $\mu = 1.0/c$, $p = 0.6$, $q = 0.4$. The accuracy is $\epsilon = 10^{-10}$. This value is also the stopping criteria for the matrix geometric approach. Results

were produced in a machine with Intel® Xeon® E5410 2.33GHz processor. In Figure 7.1, we plot the computation time versus the queueing capacity n and λ. In the plots the GEO, SPE, NAO and NEW denotes results obtained by successive substitution procedure, original spectral expansion, an improvement for the matrix-geometric method by Naoumov et al. and the new method, respectively. It is observed that the matrix geometric method (the successive substitution procedure and a procedure proposed by Naoumov et al.) takes the smallest time to compute the steady state probabilities when n is smaller than 300. Note that the difference between Naoumov's method and the successive substitution (SS) procedure is that the SS procedure involves a less computation effort (matrix multiplication, substraction and addition) in one iteration than Naoumov's method. However, the convergence of the SS procedure is slower than Naoumov's method. When n is small, the computation effort is a dominant factor in the case of Naoumov's method, so we can observe that the computation time is higher than the SS procedure in the plots.

For $n < 50$, the original spectral expansion method needs less time than the new method. The new method outperforms other methods for $n > 300$, where the computation time of the successive substitution procedure jumps to very high. The difference is more than one order of magnitude. It is worth mentioning that the memory requirement of the new method is minimal (e.g. no need to store matrices).

7.5 Conclusions

We have proved that the number of zero-eigenvalues of the characteristic matrix polynomial associated with the balance equation is $\lfloor (N + c + 2)/2 \rfloor$. We present an approach to deal with the multiple zero-eigenvalues. As a consequence, the steady state probabilities are determined in an efficient way for the M/M/c/N+c feedback queue with a large queueing capacity.

Numerical results are presented to compare computation times which are needed by the matrix geometric method, the pure spectral expansion approach, Naoumov's method and the new algorithm to calculate the steady state probabilities. The comparison clearly demonstrates the advantage of the new algorithm on the computation of the steady probabilities of the queue with a large queueing capacity.

Part IV

Multi-Server Queues with Working Vacations and ICT Applications

8 Vacation Queues and Applications[‖]

8.1 Introduction

At present, virtualization constitutes a main trend in information systems and advanced business engineering. Recent studies have shown that a proportion of 39% among 808 of the largest companies worldwide apply server virtualization to achieve new business goals and to provide more efficient services to their customers. Disaster recovery, avoidance of service outage and dynamic load balancing represent some of the most important areas for the application of the rapidly evolving virtualization concepts. Compared to existing service technologies 25% of the cost or even more can be saved by these methods.

In this context, virtualization means either to let a federation of servers appear as multiple computing entities or to let many computing entities appear as a single computer. The latter is commonly called server aggregation or grid computing. It is identified by an IDC research report (http://www.idc.com) that virtualization of system resources in severs with an x86 compatible instruction set is one disruptive technology. In the near future it may initiate a paradigm shift in IT industry providing new powerful services like enhanced server hosting.

As indicated by various studies, it is the rationale behind this trend that virtualization can reduce the infrastructure and IT management cost. The reason is that it substantially improves the utilization of the physical infrastructure, i.e. servers, storage systems and network components, while it can provide the same safety and performance compared to a solution where each ASP obtains a separate physical machine/server from the owner of the infrastructure. It is another advantage that the infrastructure can provide

‖

- Tien Van Do and Udo R. Krieger. "A performance model for maintenance tasks in an environment of virtualized servers". In *IFIP/TC6 NETWORKING 2009*, pages 931–942, Aachen, Germany, 5 2009.
- Tien Van Do. M/M/1 retrial queue with working vacations. *Acta Informatica*, 47:(1) pp. 67-75. (2010)

in a flexible manner different service packages concerning specific operating systems running on top of the same hardware.

From a practical perspective, it is observed that virtualization is a well founded area. However, there are no theoretical investigations which consider contention problems arising in the virtualized environment of a server farm. To model the interaction between application service providers and an infrastructure provider this Chapter studies the CPP/M/c queue with a compound Poissonian arrival process (CPP) and working vacations.

Such vacation queues have been an intensively studied research topic of queueing theory (cf. [57, 88, 109, 116, 126]). However, most of those studies assume a Poissonian arrival model [57, 88, 109, 126] or a model of single arrivals [116]. Regarding the performance evaluation of practical systems, this assumption limits the application of vacation queues.

Recently, queues with working vacations have obtained a big attention, see, e.g., the work of Servi et al. [109] and Liu et al. [88]. It is motivated by the performance evaluation of Wavelength Division Multiplexing (WDM) in optical systems. In this respect, the multi-server queue introduced here is indeed a generalization of the M/M/c system with synchronous vacations [126] regarding two different aspects, namely, the Poissonian batch arrivals and working vacations.

The rest of the Chapter is organized as follows. In Section 8.2, we first provide a description of the CPP/M/c model with working vacations (WV), develop then a matrix-analytic solution approach and prove some interesting property of this CPP/M/c WV-queue. Then some illustrative numerical results are presented in Section 8.3. Finally, we summarize our findings in the conclusions.

8.2 Modeling the Maintenance of a Virtualized Server Environment

8.2.1 Description of the Maintenance Model

In a virtualization environment three different roles can be identified (see Figure 8.1):

- users/applications,
- application service providers and
- owners of the hardware infrastructure.

Applications and related services, e.g. Web servers with Web, information retrieval and business services, are provided by application service providers

that require virtual machines from an infrastructure owner to run their virtualized application servers.

In this environment two interrelated categories of Service Level Agreement (SLA) can be defined:

- an SLA between users and application service providers specifying the service requirements, e.g. the response time and availability of a service, etc,
- an SLA between application service providers and an infrastructure owner.

The SLAs between users and application service providers are complicated and they also depend on the nature of the hosted applications.

To operate the infrastructure efficiently, it is recognized that advanced management tools are needed. In this respect, system management activities should also include the tasks of managing both virtual servers and physical resources efficiently.

In this Chapter, we consider the interaction between application service providers and an infrastructure owner. We suppose that there are c virtual servers available in the server pool of the infrastructure owner. To realize a pay-as-you-go approach, application service providers can initiate requests for servers to the provider of the infrastructure and server releases after task completion.

We assume that server requests arrive in batches following the Compound Poisson Process (CPP) (cf. [41]). This means that the inter-arrival times follow a Generalized Exponential (GE) distribution. The arrival process is motivated by the fact that GE is the only distribution of least bias [41] if only

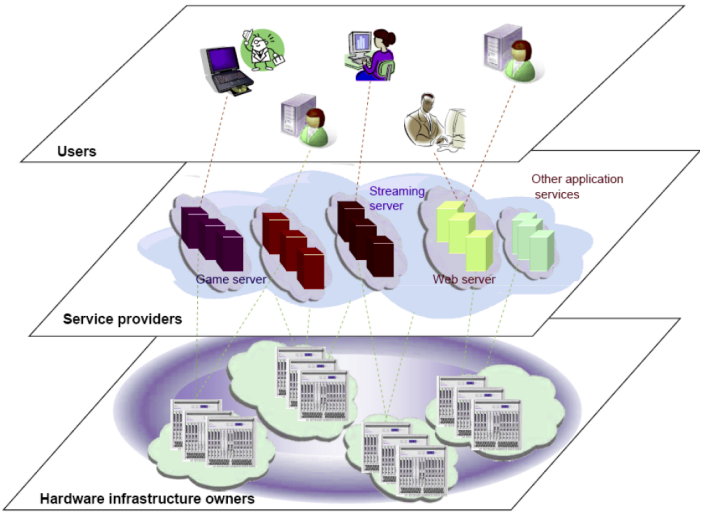


Figure 8.1: Utility computing environment based on virtual machines

the mean and variance of inter-arrival times can be reliably computed by the available measurement data. This situation typically arises in virtualized computing environments exploiting the capabilities of monitoring systems. It has been shown by recent studies [51, 54] that the CPP is sufficiently accurate to model Internet traffic in a Web server environment (i.e. the relevant CPP parameters have been estimated by the captured Internet traffic) and that it can be applied to the performance evaluation of wireless telecommunication systems.

To create a reliable computing system with these c servers, the provider of the infrastructure can initiate specific maintenance actions, e.g. software updates, a virtual server live migration etc., when any d servers become idle after a service completion instant. This kind of maintenance activities are modeled in such a way that d servers take a simultaneous vacation. During such a vacation period, the residual $c-d$ servers do not take a vacation even if they are idle. To ensure the mathematical tractability of the model, we assume that the durations of the vacation periods are independent, identical exponentially distributed random variables with parameter θ. The service rate of each server which is not in a vacation state is given by an independent exponential distribution with parameter μ. A server on vacation can serve customers following an independent exponential distribution with rate μ_v. Note that an application service provider who receives the allocation of a server which is on vacation may pay less as a form of compensation.

8.2.2 Analysis of an Advanced Multi-Server Model with Working Vacations

Here, we consider the CPP/M/c multi-server queue with working vacations, infinite waiting room and First In First Out (FIFO) service principle that we have derived as performance model to analyze maintenance tasks in a virtualized server environment.

The arrival process of customer requests is determined by a Compound Poisson Process (CPP) with parameters (λ, ω). It means that the probability distribution function of the inter-arrival times τ is defined by $\mathbb{P}\{\tau = 0\} = \omega \in (0, 1)$ and $\mathbb{P}\{0 < \tau < t\} = (1-\omega)(1-e^{-\lambda t})$. Therefore, the arrival process can be seen as a batch Poisson process whose batches of the random size S arriving at some epoch follow a geometric distribution $\mathbb{P}\{S = s\} = (1-\omega)\omega^{s-1}, s \geq 1$, with mean $\mathbb{E}(S) = 1/(1-\omega)$ and variance $\mathrm{Var}(S) = \omega/(1-\omega)^2$.

The requests are served by c servers following a specific working-vacations policy with independent, identical, exponentially distributed service and vacation times with rates μ, μ_v, θ, respectively. Let us suppose that there are no servers on vacation due to maintenance activities. Then a simultaneous vacation period of d servers starts if there are d idle servers after a service completion. At the end of a simultaneous vacation period of these d servers, three alternatives are possible:

- if there are no waiting customers, the d servers stay idle and are ready to serve any arriving new customers;

- if there are $c-d<j<c$, customers in the system, $j-c+d$ returning servers immediately start serving these customers and the other $c-j$ returning servers become idle;

- if there are $j \geq c$ customers in the system, the d returning servers all start serving these customers immediately.

At any time t the state of the system $Y(t)=(I(t),J(t))$ can be completely specified by two integer-valued random variables:

- $I(t)=\begin{cases} 0 & \text{if } d \text{ servers are on vacation at time } t \\ 1 & \text{if there are no servers on vacation at time } t \end{cases}$

- $J(t)$ represents the number of customers in the system at time t including any in service or the waiting room.

The system is now modeled by a continuous-time discrete state Markov process $Y=\{I(t),J(t)\}$ on a rectangular lattice strip $S=\{0,1\}\times\mathbb{N}_0$ due to our Markovian assumptions. We denote its corresponding steady-state probabilities by $\pi=\{\pi_{i,j}\}_{(i,j)\in S}$, where $\pi_{i,j}=\lim_{t\to\infty}\mathbb{P}\{I(t)=i,J(t)=j\}$, and let $\mathbf{v}_j=(\pi_{0,j},\pi_{1,j})$ be the partitioned vector of state probabilities.

The one-step transitions of the Markov chain Y have a specific tridiagonal block structure since the possible transitions are driven by following events:

(a) changing the status of $I(t)$, i.e. from the vacation to non-vacation of servers. Then $A_j(i,k)$ denotes the corresponding transition rate from state (i,j) to state (k,j), $i,k\in\{0,1\}, j\geq 0$. Let

$$A=A_j=\begin{bmatrix} 0 & \theta \\ 0 & 0 \end{bmatrix},\ \forall j\geq 0; \quad \text{and} \quad A^*=\begin{bmatrix} -\theta & \theta \\ 0 & 0 \end{bmatrix}.$$

(b) the arrivals of customers. Then $B_{i,j,s}$ is the rate of the $s-$step upward transition from state (i,j) to state $(i,j+s)$, $i\in\{0,1\}, j\geq 0$, caused by a batch arrival of size s and

$$B_{i,j,s} \;=\; (1-\omega)\omega^{s-1}\lambda, \quad j\geq 0, i\in\{0,1\}, s\geq 1.$$

(c) the departures of customers. $C_j(i,k)$ is the transition rate from state (i,j) to state $(k,j-1)$; $i,k\in\{0,1\}, j\geq 0$. Then we get:

$$C_j = \begin{cases} \begin{bmatrix} j\mu & 0 \\ 0 & j\mu \end{bmatrix}, & 1 \le j \le c-d \\ \begin{bmatrix} (c-d)\mu + \mu_v & 0 \\ (c-d+1)\mu & 0 \end{bmatrix}, & j = c-d+1 \\ \begin{bmatrix} (c-d)\mu + (j-c+d)\mu_v & 0 \\ 0 & j\mu \end{bmatrix}, & c-d+1 < j \le c \\ \begin{bmatrix} (c-d)\mu + d\mu_v & 0 \\ 0 & c\mu \end{bmatrix} = C, & j > c. \end{cases}$$

Note that by a transition from $(1, c-d+1)$ to $(0, c-d)$ after a service completion with rate $(c-d+1)\mu$ we get a simultaneous vacation of d servers.

Let $\mathrm{Diag}(x)$ denote the diagonal matrix defined by a row vector x and $E \in \mathbb{R}^{2\times 2}$ be the identity matrix. We introduce the following notations

$$\begin{aligned} \Lambda &= \mathrm{Diag}[\lambda, \lambda] = \lambda E; \quad \Omega = \mathrm{Diag}[\omega, \omega] = \omega E; \\ B_s &= B_{j,s} = \mathrm{Diag}[(1-\omega)\omega^{s-1}\lambda, (1-\omega)\omega^{s-1}\lambda], \quad j \ge 0, \end{aligned}$$

and obtain

$$\begin{aligned} B_s &= \Omega^{s-1}(E-\Omega)\Lambda = \omega^{s-1}(1-\omega)\lambda E, \quad j \ge 1, \\ \Lambda &= \sum_{s=1}^{\infty} B_s = \lambda E. \end{aligned}$$

Theorem 8.1. *The necessary and sufficient condition for the existence of the steady-state probabilities of the process $Y = (I, J)$ is determined by*

$$\frac{\lambda}{c\mu} + \omega < 1 \quad \Leftrightarrow \quad \rho = \frac{\lambda}{(1-\omega)\cdot c\mu} < 1. \tag{8.1}$$

Remark 2. *The standard condition (8.1) states that the traffic intensity ρ must be less than one to achieve the ergodicity of Y. Neither the rate θ of the vacations period nor the number d of simultaneous servers on vacations have an impact on the stability of the system. In other words, the system will not be overloaded due to a maintenance activity. Indeed, this is good news for a system administrator who shall organize the maintenance tasks of idle virtual machines.*

Proof. The steady-state balance equation of the M/G/1-like upper Hessenberg system can be written as follows:

$$\sum_{s=1}^{j} \mathbf{v}_{j-s} B_s + \mathbf{v}_j \left[A^* - \Lambda - D^{C_j}\right] + \mathbf{v}_{j+1} C_{j+1} = 0, \qquad \forall j \ge 1. \tag{8.2}$$

Here D^{C_j} are diagonal matrices whose diagonal elements are the sum of the elements in the rows of C_j. Note that by construction $D^{C_j} = C_j$ holds for all $j \neq c - d + 1$.

For $j \geq c + 1$ we can write

$$\sum_{s=1}^{j} \mathbf{v}_{j-s} B_s + \mathbf{v}_j \left[A^* - \Lambda - C\right] + \mathbf{v}_{j+1} C = 0. \tag{8.3}$$

Substituting $B_s = \Omega^{s-1}(E - \Omega)\Lambda$ into this equation (8.3), we get

$$\sum_{s=1}^{j} \mathbf{v}_{j-s} \Omega^{s-1}(E - \Omega)\Lambda + \mathbf{v}_j \left[A^* - \Lambda - C\right] + \mathbf{v}_{j+1} C = 0 \qquad \forall j \geq c + 1, \tag{8.4}$$

and

$$\sum_{s=1}^{j-1} \mathbf{v}_{j-1-s} \Omega^{s-1}(E - \Omega)\Lambda + \mathbf{v}_{j-1} \left[A^* - \Lambda - C\right] + \mathbf{v}_j C = 0 \qquad \forall j \geq c + 2. \tag{8.5}$$

If we multiply equation (8.5) by Ω and then subtract the result from equation (8.4), we obtain the three-term recurrence equations

$$\mathbf{v}_{j-1}[\Lambda - A^*\Omega + C\Omega] + \mathbf{v}_j \left[A^* - \Lambda - C - C\Omega\right] + \mathbf{v}_{j+1} C = 0 \ , \ j \geq c + 2,$$

$$\mathbf{v}_{j-1} Q_0 + \mathbf{v}_j Q_1 + \mathbf{v}_{j+1} Q_2 = 0 \ , \ j \geq c + 2, \tag{8.6}$$

where $Q_0 = \Lambda - A^*\Omega + C\Omega$, $Q_1 = A^* - \Lambda - C - C\Omega$, $Q_2 = C$.

$Q(x) = Q_0 + Q_1 x + Q_2 x^2$ is defined as the characteristic matrix polynomial associated with the equations (8.6). It is proved in [97] that the solution of these matrix equations (8.6) is closely related to the eigenvalues and left-eigenvectors of the polynomial $Q(x)$. If $(x, \boldsymbol{\psi})$ is an eigenvalue-eigenvector pair of $Q(x)$, then it holds

$$\boldsymbol{\psi} Q(x) = 0, \ \ \det[Q(x)] = 0.$$

Consequently, we obtain:

$$\begin{aligned}
\det[Q(x)] &= \det \begin{bmatrix} q_{00}(x) & \theta x - \theta\omega \\ 0 & q_{11}(x) \end{bmatrix} = q_{00}(x) q_{11}(x), \\
q_{00}(x) &= \lambda + ((c-d)\mu + d\mu_v)\omega + \omega\theta - \\
&\quad (\lambda + (c-d)\mu + d\mu_v + ((c-d)\mu + d\mu_v)\omega + \theta)x + ((c-d)\mu + d\mu_v)x^2, \\
q_{11}(x) &= \lambda + c\mu\omega - (\lambda + c\mu + c\mu\omega)x + c\mu x^2 = (1 - x)(\lambda + c\mu\omega - c\mu x).
\end{aligned}$$

Therefore, $Q(x)$ has four eigenvalues

$$\begin{aligned}
x_1 &= \frac{1}{2G}\{H + G - \sqrt{(H+G)^2 - 4G(\lambda + ((c-d)\mu + d\mu_v)\omega + \omega\theta)}\}, \\
x_2 &= \frac{1}{2G}\{(H + G + \sqrt{(H+G)^2 - 4G(\lambda + ((c-d)\mu + d\mu_v)\omega + \omega\theta)}\}, \\
x_3 &= \lambda/(c\mu) + \omega, \ \ x_4 = 1,
\end{aligned}$$

where

$$\begin{aligned} G &= (c-d)\mu + d\mu_v, \\ H &= \lambda + ((c-d)\mu + d\mu_v)\omega + \theta \end{aligned}$$

holds.

Note that $\psi_1 = (1, (\theta\omega - \theta x_1)/q_{11}(x_1))$ is the left-hand-side (LHS) eigenvector of $Q(x)$ for the eigenvalue x_1, and $\psi_3 = \psi_4 = (0, 1)$ are the LHS eigenvectors of $Q(x)$ for the eigenvalues x_3 and x_4, respectively.

Since $\omega < 1$ holds, we have

$$\begin{aligned} (\lambda + ((c-d)\mu + d\mu_v)\omega + \omega\theta) &< H, \\ 4G(\lambda + ((c-d)\mu + d\mu_v)\omega + \omega\theta) &< 4GH, \\ (H+G)^2 - 4G(\lambda + ((c-d)\mu + d\mu_v)\omega + \omega\theta) &> (H+G)^2 - 4GH, \\ 0 < x_1 < \frac{1}{2G}(H + G - |H - G|) &\leq 1, \\ x_2 > \frac{1}{2G}(H + G + |H - G|) &\geq 1. \end{aligned}$$

Applying results from [97] (summarized in Section 2.2.1), it is a necessary and sufficient condition for the ergodicity of the Markov chain Y that the number of eigenvalues of $Q(x)$ inside the unit disk is given by 2. Therefore, $x_3 < 1$ is required which yields condition (8.1). □

The steady-state balance equations of $J(t) \in \{0, \ldots, c+1\}$ can be written in the following form:

$$\begin{aligned} \mathbf{v}_0 \left[A^* - \Lambda\right] + \mathbf{v}_1 C_1 &= 0, \\ \sum_{s=1}^{j} \mathbf{v}_{j-s} B_s + \mathbf{v}_j \left[A^* - \Lambda - D^{C_j}\right] + \mathbf{v}_{j+1} C_{j+1} &= 0\,, \quad 1 \leq j \leq c+1. \end{aligned}$$

For $j \geq c+1$ the steady-state probabilities can be expressed as follows (cf. [97]):

$$\begin{aligned} \mathbf{v}_j &= \alpha\boldsymbol{\psi}_1 x_1^j + \beta\boldsymbol{\psi}_3 x_3^j, \\ \pi_{0,j} &= \alpha x_1^j, \\ \pi_{1,j} &= \alpha\frac{\theta\omega - \theta x_1}{q_{11}(x_1)} x_1^j + \beta x_3^j, \end{aligned} \tag{8.7}$$

where α and β are coefficients that have to be determined by the boundary conditions.

Furthermore, we have to satisfy the normalization equation:

$$\sum_{j=0}^{\infty}\sum_{i=0}^{1} \pi_{i,j} = 1. \tag{8.8}$$

Consequently, we have to determine the vectors $\mathbf{v}_j$, $0 \leq j \leq c$, α and β. The total number of these unknowns is given by $2(c+1)+2 = 2(c+2)$. To determine these unknowns, we have the steady-state balance equations of the levels $j = 0, \ldots, c+1$ and the normalization equation. Thus, $2(c+2)+1$ is the number of boundary equations, among those only $2(c+2)$ equations are independent.

It can be observed from the steady-state balance equations of $J(t) \in \{0, \ldots, c\}$ that $\mathbf{v}_j$, $1 \leq j \leq c$ and $j \neq c-d+1$, can be expressed as a function of $\mathbf{v}_0$, i.e $\pi_{0,0}$ and $\pi_{1,0}$, and $\mathbf{v}_{c-d+1}$. Therefore, we have only six unknowns $(\pi_{0,0}, \pi_{1,0}, \pi_{0,c-d+1}, \pi_{1,c-d+1}, \alpha, \beta)$, which can be solved efficiently using the steady-state balance equations of the states $J(t) = c$, $J(t) = c-d$, $J(t) = c+1$ and the normalization equation.

8.2.3 Conditional Stochastic Decomposition

In the following, we prove a conditional stochastic decomposition property for the CPP/M/c queue with working vacations.

Theorem 8.2. *If the ergodicity condition for the CPP/M/c queue with working vacations holds, then the conditional steady-state queue length* $J_b = lim_{t\to\infty}\{J(t)-c-1|J(t) > c, I(t) = 1\}$ *provided that the server system is not on a working vacation can be decomposed into the sum of two independent random variables*

$$J_b = J_0 + J_c.$$

Here J_0 *is the conditional steady-state queue length of the CPP/M/c queue without vacations and* J_c *is the additional steady-state queue length due to vacations.*

Proof. The probability that the server is busy and the number of jobs is larger than c is determined by:

$$\begin{aligned} P_b = P\{J(t) > c, I(t) = 1\} &= \sum_{j=c+1}^{\infty} \pi_{1,j} = \sum_{j=c+1}^{\infty} \left(\alpha \frac{\theta\omega - \theta x_1}{q_{11}(x_1)} x_1^j + \beta x_3^j \right) \\ &= \alpha \frac{\theta\omega - \theta x_1}{q_{11}(x_1)} \frac{x_1^{c+1}}{1-x_1} + \beta \frac{x_3^{c+1}}{1-x_3}. \end{aligned}$$

The probability generating function of J_b can be expressed as follows:

$$\begin{aligned} G_{J_b}(z) &= \sum_{j=0}^{\infty} P\{J_b = j\} z^j = \sum_{j=0}^{\infty} \frac{\pi_{1,j+c+1}}{P_b} z^j \\ &= \frac{1}{P_b} \sum_{j=0}^{\infty} \left(\alpha \frac{\theta\omega - \theta x_1}{q_{11}(x_1)} x_1^{j+c+1} + b x_3^{j+c+1} \right) z^j \\ &= \frac{1}{P_b} \left(\alpha \frac{\theta\omega - \theta x_1}{q_{11}(x_1)} x_1^{c+1}/(1-x_1 z) + \beta x_3^{c+1}/(1-x_3 z) \right). \end{aligned}$$

The steady-state probabilities of the CPP/M/c queue without vacations can be obtained by setting $\theta = 0$, $d = 0$ and $\mu_v = \mu$. The probability that the number of customers in the CPP/M/c queue without vacations is given by $\pi_j = \beta^* x_3^j$ for $j \geq c+1$, where β^* is an appropriate coefficient. Therefore, $G_{J_0}(z) = \beta^* \frac{x_3^{c+1}}{1-x_3 z}$ follows for the probability generating function of J_0. These relations yield the stated result. □

8.3 Illustrative Numerical Results

In this section we present some numerical results to illustrate the impact of the model parameters on the formulation of an effective maintenance policy, i.e. how many servers should be simultaneously on vacations. For demonstration purposes, we investigate the average number of customer requests waiting for free servers

$$\begin{aligned}
\mathbb{E}(L_Q) &= \sum_{j=c+1}^{\infty} (j-c) \cdot (\pi_{0,j} + \pi_{1,j}) \\
&= \sum_{j=c+1}^{\infty} (j-c) \cdot \left(\alpha[1 + \frac{\theta\omega - \theta x_1}{q_{11}(x_1)}] x_1^j + \beta x_3^j \right) \\
&= \frac{\alpha[1 + \frac{\theta\omega - \theta x_1}{q_{11}(x_1)}] x_1^{(c+1)}}{(1-x_1)^2} + \frac{\beta x_3^{(c+1)}}{(1-x_3)^2}
\end{aligned}$$

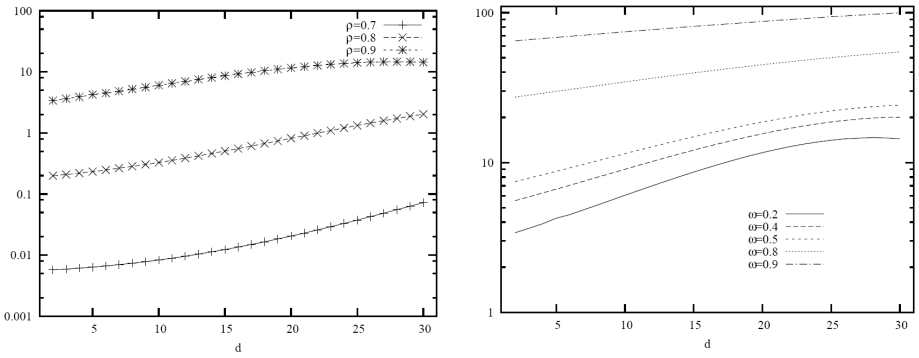
as major performance metrics and select some illustrative parameter set. Other characteristics like the mean number of requests in the system

$$\mathbb{E}(L) = \sum_{j=1}^{\infty} j \cdot (\pi_{0,j} + \pi_{1,j}) \,,$$

or the mean number of active servers $\mathbb{E}(N_V) = \sum_{j=1}^{\infty} \min(j,c) \cdot (\pi_{0,j} + \pi_{1,j})$ and the throughput $\eta = \sum_{j=1}^{\infty} \mathbf{v}_j \cdot C_j \cdot \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \sum_{j=1}^{\infty} \sum_{i=0}^{1} \pi_{i,j} \cdot (C_j(i,0) + C_j(i,1))$ can be computed in a similar manner.

In Figure 8.2(a) we plot the average number of waiting requests $\mathbb{E}(L_Q)$ versus d for the following parameter set of a high load regime: $c = 100$ servers, $\omega = 0.2$, $\theta = 1.0$, $\mu = 5.0$, $\mu_v = 2.5$, $\rho = \lambda/[(1-\omega)c\mu] \in [0.7, 0.9]$. It generates batch arrivals of mean size $\mathbb{E}(S) = 1.25$ and variance $\mathrm{Var}(S) = 0.3125$ for a high traffic intensity $0.7 \leq \rho \leq 0.9$ and assumes that the average service time $1/\mu$ of requests needs only 20 % of the mean maintenance time $1/\theta$ while during these maintenance periods the latter service time is extended by 100 % compared to the normal operation mode.

Considering the average number $\mathbb{E}(L_Q)$ of requests waiting in the system, it is observed that increasing the load ρ from 0.7 to 0.8 or from 0.8 to 0.9 generates an increment of one order of magnitude. To show the impact of the size S of arriving batches and the influence of $\omega = 1 - 1/\mathbb{E}(S)$ and $\mathbb{E}(S) \in \{1.25, 1.67, 2, 5, 10\}$, respectively, we use the set of the same parameters but fix the load at $\rho = 0.8$. Figure 8.2(b) illustrates the average number of waiting requests $\mathbb{E}(L_Q)$ versus d and ω. In Figure 8.3 $\mathbb{E}(L_Q)$ is plotted against d and μ_v for the load $\rho = 0.9$ and a mean batch size of $\mathbb{E}(S) = 1/(1-\omega) = 1.25$.



(a) (b)

Figure 8.2: Average number $\mathbb{E}(L_Q)$ of waiting requests versus d for different traffic load ρ (left) and different control parameter $\omega = 1 - 1/\mathbb{E}(S)$ of the mean batch size $\mathbb{E}(S)$ (right)

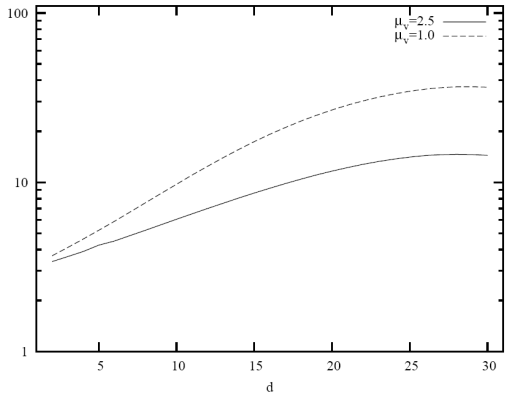


Figure 8.3: Average number $\mathbb{E}(L_Q)$ of waiting requests versus d and μ_v

It is observed that batch arrivals have the strongest impact on the average number of waiting customers. The impact of the offered load ρ and the service rate μ_v during maintenance can be handled by choosing an appropriate number d of servers under maintenance.

8.4 Summary

To model the queueing and congestion phenomena arising from maintenance tasks of a virtualized server environment, we have presented in this Chapter a CPP/M/c multi-server system with Poissonian batch arrivals and working vacations.

In the proposed queueing system the inter-arrival times of jobs requesting service by a virtualized server follow a Generalized Exponential distribution. To model the maintenance activities, we have assumed that a certain number of servers goes simultaneously to a maintenance state for a random period when they have completed the service of jobs and find no further requests in the waiting line.

Analyzing the arising Markovian model by the spectral expansion method, we have derived a new expression for the steady-state probabilities and proved a conditional stochastic decomposition property. The validation of the approach in a testbed and the estimation of the parameters by gathered data is a topic of future research.

In conclusion, we believe that the proposed Markovian multi-server system with working vacations can serve as a useful tool to define efficient maintenance policies in the virtualized environment of current server farms.

Part V

Appendices

A

Proofs related to the HetSigma queue

A.1 Illustration for $K = 2$

Now, let us look at the other Q_l's in (3.20) pertaining to $K = 2$. Then, the coefficients Q_{-n} of $\mathbf{v}_{j-K-n}$ $(n \geq 1)$ for possible values of n are from (3.22)

$$\begin{aligned} Q_{-1} &= \sum_{l=-1-c}^{2} \left[\Theta_1^{2+c-l}(E-\Theta_1)\Sigma_1 + \Theta_2^{2+c-l}(E-\Theta_2)\Sigma_2\right] G_{2,c,l} \ , \\ Q_{-n} &= \sum_{l=-1-c}^{2} \left[\Theta_1^{2+c+n-l-1}(E-\Theta_1)\Sigma_1 + \Theta_2^{2+c+n-l-1}(E-\Theta_2)\Sigma_2\right] G_{2,c,l} \\ &\qquad\qquad (n = 1, 2, \ldots, j-K) \ . \end{aligned} \tag{A.1}$$

Theorem A.1. *Referring to equation (A.1) for all c,* $Q_{-n} = 0 \qquad (n = 1, 2, \ldots, j-K)$.

Proof. For $c = 1$, we have

$$\begin{aligned} G_{2,1,-2} &= \Delta\Phi_1, \\ G_{2,1,-1} &= -(\Delta + \Phi_1 + \Theta_1\Delta\Phi_1 + \Theta_2\Delta\Phi_1), \\ G_{2,1,0} &= E + \Theta_1\Delta + \Theta_2\Delta + (\Theta_1 + \Theta_2 + \Theta_1\Theta_2\Delta)\Phi_1, \\ G_{2,1,1} &= -(\Theta_1 + \Theta_2 + \Theta_1\Theta_2\Delta + \Theta_1\Theta_2\Phi_1), \\ G_{2,1,2} &= \Theta_1\Theta_2, \end{aligned}$$

$$\begin{aligned} G_{2,1,-2} = \Delta\Phi_1,\ G_{2,1,-1} &= -(\Delta + \Phi_1 + \Theta_1\Delta\Phi_1 + \Theta_2\Delta\Phi_1), \\ G_{2,1,0} &= E + \Theta_1\Delta + \Theta_2\Delta + (\Theta_1 + \Theta_2 + \Theta_1\Theta_2\Delta)\Phi_1, \\ G_{2,1,1} = -(\Theta_1 + \Theta_2 + \Theta_1\Theta_2\Delta + \Theta_1\Theta_2\Phi_1),\ G_{2,1,2} &= \Theta_1\Theta_2. \end{aligned}$$

Expanding and simplifying (A.1), we get

$$Q_{-n} = 0 \qquad (n = 1, 2, \ldots, j-K). \tag{A.2}$$

Assume the proposition is *true* for any $c = h$. Hence,

$$\begin{aligned} Q_{-1} &= \sum_{l=-1-h}^{2} \left[\Theta_1^{2+h-l}(E-\Theta_1)\Sigma_1 + \Theta_2^{2+h-l}(E-\Theta_2)\Sigma_2\right] G_{2,h,l} \\ Q_{-n} &= \sum_{l=-1-h}^{2} \left[\Theta_1^{2+h+n-l-1}(E-\Theta_1)\Sigma_1 + \Theta_2^{2+h+n-l-1}(E-\Theta_2)\Sigma_2\right] G_{2,h,l} \\ & \qquad\qquad (n = 1, 2, \ldots, j-K)\ . \end{aligned} \tag{A.3}$$

For $c = h + 1$ we have from (3.24),

$$\begin{aligned} Q_{-n} &= \sum_{l=-1-h-1}^{2} \left[\Theta_1^{2+h+1+n-l-1}(E-\Theta_1)\Sigma_1 + \Theta_2^{2+h+1+n-l-1}(E-\Theta_2)\Sigma_2\right] G_{2,h+1,l} \\ &= \sum_{l=-1-h-1}^{2} \left[\Theta_1^{2+h+1+n-l-1}(E-\Theta_1)\Sigma_1 + \Theta_2^{2+h+1+n-l-1}(E-\Theta_2)\Sigma_2\right] \\ & \quad [G_{2,h,l} - \Phi_{h+1} G_{2,h,l-1}] \\ &= 0 \qquad\qquad (n = 1, 2, \ldots, j-K)\ . \end{aligned} \tag{A.4}$$

Therefore, the theorem is true. □

The coefficients Q_{K-m} of v_{j-m}

$$\begin{aligned} Q_{K-m} &= \sum_{l=m+1}^{K} \left[\sum_{n=1}^{c} M_n(E-\Phi_n)\Phi_n^{l-m-1} + R(E-\Delta)\Delta^{l-m-1}\right] G_{K,c,l} \\ &= \sum_{l=-1-c}^{K} \left[\sum_{n=1}^{c} M_n(E-\Phi_n)\Phi_n^{l-m-1} + R(E-\Delta)\Delta^{l-m-1}\right] G_{K,c,l} \\ & \qquad\qquad (m = j-L, \ldots, -2-c). \end{aligned} \tag{A.5}$$

We can prove in the similar way as the above, that $Q_{K-m} = 0$ $(K = 2; m = j - L, \ldots, -2 - c)$.

A.2 Automatic Validation of Equations

In this section we an empirical validation, using Mathematica for some theorems in Section 3.3.3 (note that the rigorous proofs are presented therein). The theorems can be symbolically validated for any values of N, K and c by Mathematica.

The Mathematica code for the validation of Theorem 3.2 and 3.3 is illustrated in Table A.1.

```
(*This is a Q_{K - m}*) QPolyCoeff[NN_,K_,c_,m_]:=Block[{},
 Q=Array[q,{NN+1,NN+1},{0,0}];
 For[i = 0, i <= NN,
    i++, {seged = Sum[q[i, j], {j, 1, NN}] - q[i, i]; q[i, i] = -seged }];
              (*this to compute the diagonal elements of the Q matrix*)
 TheSigmaMatrix=Array[sigmaM,{NN+1,K},{0,1}];
 TheThetaMatrix=Array[thetaM,{NN+1,K},{0,1}];
 For[i=1,i<=K,i++,Sigma[i]=DiagonalMatrix[Transpose[TheSigmaMatrix][[i]]]];
 SumSigma=Sum[Sigma[i],{i,1,K}];
 For[i=1,i<=K,i++,Theta[i]=DiagonalMatrix[Transpose[TheThetaMatrix][[i]]]];
 TheMuMatrix=Array[mu,{NN+1,c},{0,1}];
 ThePhiMatrix=Array[phiM,{NN+1,c},{0,1}];

 For[i=1,i<=c,i++,M[i]=DiagonalMatrix[Transpose[TheMuMatrix][[i]]]];
 Cmatrix=Sum[M[i],{i,1,c}];
 For[i=1,i<=c,i++,Phi[i]=DiagonalMatrix[Transpose[ThePhiMatrix][[i]]]];
 Delta=DiagonalMatrix[Array[deltaM,NN+1]];
 R=DiagonalMatrix[Array[rho,NN+1]];
 H[1,-1]=IdentityMatrix[NN+1]-IdentityMatrix[NN+1];
 F[k_,l_]:=If[l>=0&&l<=k,F[k-1,l]+Theta[k].F[k-1,l-1],H[1,-1]];
 F[1,0]=IdentityMatrix[NN+1];
 F[1,1]=Theta[1];F[1,-1]=IdentityMatrix[NN+1]-IdentityMatrix[NN+1];
 H[k_,l_]:=If[l>=0&&l<=k,H[k-1,l]+Phi[k].H[k-1,l-1],H[1,-1]];
 H[1,0]=IdentityMatrix[NN+1];
 H[1,1]=Phi[1];
 H[1,-1]=IdentityMatrix[NN+1]-IdentityMatrix[NN+1];
 G[k_]:=Power[-1,k] Sum[(F[K,k+n]+F[K,k+n+1] Delta) H[c,n],{n,0,c}];
 Return[Sum[MatrixPower[Theta[n],m-l-1].
          (IdentityMatrix[NN+1]-Theta[n]).Sigma[n]. G[l],{l,-1-c,m-1},{n,1,K}]+
          (Q-SumSigma-Cmatrix-R).G[m]+
        Sum[(Sum[M[n].(IdentityMatrix[NN+1]-Phi[n]).
         MatrixPower[Phi[n],l-m-1],{n,1,c}]+
        R.(IdentityMatrix[NN+1]-Delta).MatrixPower[Delta,l-m-1]).G[l],{l,m+1,K}]]
];

NN = 2; K = 3; c = 4;
Print[Simplify[QPolyCoeff[NN, K, c, K +1]]];
{{0, 0, 0},{0,0,0},{0, 0, 0}}
Print[Simplify[QPolyCoeff[NN, K,c,K+2]]];
{{0,0,0},{0,0,0},{0,0,0}}
Print[Simplify[QPolyCoeff[NN,K,c,-c-2]]];
{{0,0,0},{0,0,0},{0,0,0}}
Print[Simplify[QPolyCoeff[NN, K, c, -c -3]]];
{{0,0,0},{0,0,0},{0,0,0}}


NN = 1; K = 2; c = 2;
Print[Simplify[QPolyCoeff[NN, K, c, K +1]]];
{{0,0},{0,0}}
Print[Simplify[QPolyCoeff[NN, K, c, K + 2]]];
{{0,0},{0,0}}
Print[Simplify[QPolyCoeff[NN, K, c, -c - 2]]];
{{0,0},{0,0}}
Print[Simplify[QPolyCoeff[NN, K, c, -c - 3]]];

{{0,0},{0,0}}
```

Table A.1: Mathematica code to symbolically verify that the Q_l ($l < 0$ or $l > K + c + 1$) are zero. This program works for any values of N, K and c (NN is used because N is a reserved keyword in Mathematica)

```
SS[k_, h_] := Sum[QPolyCoeff[NN, k, h, m], {m, -h - 1, k}];

NN = 1; K = 2; c = 2;
Simplify[Det[SS[K, c]]];
Out[99]=0

NN = 3; K = 4; c = 3;
Simplify[Det[SS[K, c]]];
Out[100]=0
```

Table A.2: Mathematica code to automatically and symbolically verify that $\sum_{m=-1-c}^{k} Q_{k-m}^{(k)}$ is singular. Note that is valid for any values of N, K and c.

The Mathematica code for the validation of Theorem 3.5 is presented in Table A.2. It can be seen that the determinant of $\sum_{m=-1-c}^{K} Q_{K-m}^{(K,c)}$ is zero.

B

Representative publications

Journal papers

[J1] M Ajmone Marsan, A Bianco, T V Do, L Jereb, R Lo Cigno, M Munafo. ATM Simulation with CLASS. *Performance Evaluation* 24:(1-2) pp. 137-159. (1995) (IF: 0.513)

[J2] Tien Van Do, Thong T Nguyen, Hung T Tran, G Kalvach, B Varga. Topology Optimization of an Overlay ATM Network in an SDH Infrastructure. *Computer Networks*. 34:(1) pp. 199-210. (2000) (IF: 0.390)

[J3] R. Chakka and Tien Van Do. The MM $\sum_{k=1}^{K} CPP_k/GE/c/L$ G-Queue with Heterogeneous Servers: Steady state solution and an application to performance evaluation. *Performance Evaluation*, 64:191–209, March 2007. (IF=0.616)

[J4] Tien Van Do, U. R. Krieger, and R. Chakka. Performance modeling of an Apache Web Server with a Dynamic Pool of Service Processes. *Telecommunication Systems*, 39(2):117–129, 2008. (IF=0.40)

[J5] Tien Van Do, R. Chakka. A New Performability Model for Queueing and FDL-related Burst Loss in Optical Switching Nodes, *Computer Communications*, accepted (IF=0.884*)

[J6] Tien Van Do. Comments on Multi-Server System with Single Working Vacation. *Applied Mathematical Modelling* 33 (12) (2009) 4435–4437. (IF=0.931*)

[J7] Tien Van Do. An Efficient Solution to a Retrial Queue for the Performability Evaluation of DHCP. *Computers and Operations Research* 37(7):1191–1198, 2010. http://dx.doi.org/10.1016/j.cor.2009.05.014 (IF=1.366*)

*IF of 2008

[J8] Tien Van Do. An Efficient Computation Algorithm for A Multiserver Feedback Retrial Queue With A Large Queueing Capacity *Applied Mathematical Modelling*. http://dx.doi.org/10.1016/j.apm.2009.10.025 (IF=0.931*)

[J9] Tien Van Do. M/M/1 retrial queue with working vacations. *Acta Informatica*, 47:(1) pp. 67-75. (2010) http://dx.doi.org/10.1007/s00236-009-0110-y (IF=0.789*)

[J10] Tien Van Do. Modeling a Resource Contention in the Management of Virtual Organizations. *Information Sciences*

[J11] Tien Van Do. *An Efficient Computational Method for Retrial Queues to Cellular Mobile Systems with Guard Channels*. Submitted for publications

[J12] Tien Van Do and Ram Chakka. *Simulation and Analytical Approaches for Estimating the Performability of a Multicast Address Dynamic Allocation Mechanism*. Submitted for publications

[J13] Tien Van Do and Ram Chakka. *An Efficient Method to Compute the Rate Matrix for Retrial Queues with Large Number of Servers.* Submitted for publications

[J14] Tien Van Do. A Computational Algorithm for the CPP/M/c Retrial Queue. *Annales Mathematicae Et Informaticae* (1787-5021) (2009).

Book chapters

[B1] D Papp, Tien Van Do, Ram Chakka, X M T Truong. Repair strategies on the operation of MPLS routing In: Nejat Ince, Arnold Bragg (szerk.) Recent Advances in Modeling and Simulation Tools for Communication Networks and Services. Berlin ; Heidelberg: Springer-Verlag, 2007. pp. 319-330. (ISBN:0387739076)

[B2] Tien Van Do, Nam H. Do, and Ram Chakka. Performance evaluation of the high speed downlink packet access in communications networks based on high altitude platforms. In Khalid Al-Begain, Armin Heindl, and Miklós Telek, editors, *ASMTA*, volume 5055 of *Lecture Notes in Computer Science*, pages 310–322. Springer, 2008.

[B3] Ram Chakka, Tien Van Do, and Zsolt Pandi. A Generalized Markovian Queue and Its Applications to Performance Analysis in Telecommunications Networks. In D. Kouvatsos, editor, *Performance Modelling and Analysis of Heterogeneous Networks*, pages 371–387. River Publisher, 2009.

[B4] Tien Van Do, Denes Papp, Ram Chakka, and Mai X T Truong. A Performance Model of MPLS Multipath Routing with Failures and Repairs of the LSPs. In D. Kouvatsos, editor, *Performance Modelling and Analysis of Heterogeneous Networks*, pages 27–43. River Publisher, 2009.

[B5] Tien Van Do and Ram Chakka. *Generalised QBD Processes, Spectral Expansion and Performance Modelling Applications.* In Next Generation Internet: Performance Evaluation and Applications edited by D. D. Kouvatsos, Lecture Notes in Computer Science, Vol. LNCS 5233, Springer-Verlag, Heidelberg, Germany ISBN: 978-3-540-99500-5

[B6] Ram Chakka and Tien Van Do. *Some New Markovian Models for Traffic and Performance Evaluation Telecommunication Networks.* In Next Generation Internet: Performance Evaluation and Applications edited by D. D. Kouvatsos, Lecture Notes in Computer Science, Vol. LNCS 5233, Springer-Verlag, Heidelberg, Germany ISBN: 978-3-540-99500-5

Conference papers

[C1] H. T. Tran and Tien Van Do. A new iterative method for systems with batch arrivals and batch departures. In *Proceedings of CNDS'00*, pages 131–137, San Diego, USA, 2000.

[C2] Hung T. Tran and Tien Van Do. Comparison of some Numerical Methods for QBD-M Processes via Analysis of an ATM Concentrator. In *Proceedings of 20th IEEE International Performance, Computing and Communications Conference, IPCCC 2001*, Pheonix, USA, 2001.

[C3] Ram Chakka, Tien Van Do. The MM $\sum_{k=1}^{K} CPP_k/GE/c/L$ G-Queue and its applications in load balancing of MPLS networks. In: IEEE LCN 2002 Conference., USA 2002., pp. 0735-0736.

[C4] Ram Chakka, Tien Van Do, and Zsolt Pandi. Generalized Markovian queues and applications in performance analysis in telecommunication networks. In D. D. Kouvatsos, editor, *the First International Working Conference on Performance Modelling and Evaluation of Heterogeneous Networks (HET-NETs 03)*, pages 60/1–10, July 2003.

[C5] Tien Van Do, R. Chakka, and P. G. Harrison. An integrated analytical model for computation and comparison of the throughputs of the UMTS/HSDPA user equipment categories. In *MSWiM '07: Proceedings of the 10th ACM Symposium on Modeling, analysis, and simulation of wireless and mobile systems*, pages 45–51, New York, NY, USA, 2007. ACM.

[C6] Tien Van Do and Udo R. Krieger. A performance model for maintenance tasks in an environment of virtualized servers. In *IFIP/TC6 NETWORKING 2009*, pages 931–942, Aachen, Germany, 5 2009.

Bibliography

[1] 3GPP Technical Report 25.214, version 7.0.0. *Physical layer procedures (FDD)*. The 3GPP project, March 2006.

[2] ab – Apache HTTP server benchmarking tool. http://httpd.apache.org/docs/2.2/programs/ab.html.

[3] N. Akar and K. Sohraby. Finite and Infinite QBD Chains: A Simple and Unifying Algorithmic Approach. In *Proceedings of IEEE INFOCOM*, pages 1105–1113, 1997.

[4] N. Akar and K. Sohraby. Matrix-geometric Solutions in M/G/1 type Markov Chains with Multiple Boundaries: A Generalized State-space Approach. *International Teletraffic Congress*, March, 1997.

[5] B. Almási, J. Roszik, and J. Sztrik. Homogeneous finite-source retrial queues with server subject to breakdowns and repairs. *Mathematical and Computer Modelling*, (42):673–682, 2005.

[6] M. Andersson, J. Cao, M. Kihl, and C. Nyberg. Performance Modeling of an Apache Web Server with Bursty Arrival Traffic. In *Proceedings of the International Conference on Internet Computing, Las Vegas, USA*, June 2003.

[7] Apache Web Server. http://www.apache.org.

[8] Jesús R. Artalejo. Analysis of an M/G/1 queue with constant repeated attempts and server vacations. *Computers & OR*, 24(6):493–504, 1997.

[9] Jesús R. Artalejo, Antonis Economou, and Antonio Gómez-Corral. Applications of maximum queue lengths to call center management. *Computers & OR*, 34(4):983–996, 2007.

[10] Jesús R. Artalejo, Antonis Economou, and Antonio Gómez-Corral. Algorithmic analysis of the Geo/Geo/c. *European Journal of Operational Research*, 189(3):1042–1056, 2008.

[11] Jesus R. Artalejo and Antonio Gómez-Corral. Channel idle periods in computer and telecommunication systems with customer retrials. *Telecommunication Systems*, 24(1):29–46, 2003.

[12] Jesús R. Artalejo and Antonio Gómez-Corral. *Retrial Queueing Systems*. Springer, 2008.

[13] Z. Bai, J. Demmel, J. Dongarra, A. Ruhe, and H. van der Vorst, editors. *Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide*. SIAM, Philadelphia, 2000.

[14] D. Bini and B. Meini. On the solution of a nonlinear matrix equation arising in queueing problems. *SIAM Journal on Matrix Analysis and Applications*, 17(4):906–926, 1996.

[15] D. Bini and B. Meini. On the Solution of a Nonlinear Matrix Equation Arising in Queueing Problems. *SIAM J. Matrix Anal. Appl.*, 17:906–926, 1996.

[16] D. Bini and B. Meini. Improved Cyclic Reduction for Solving Queueing Problems. *Numerical Algorithms*, 15:57–74, 1997.

[17] D. Bini and B. Meini. Using Displacement Structure for Solving Non-Skip-Free M/G/1 Type Markov Chains. pages 17–37, 1998.

[18] C. Blondia and O. Casals. Statical Multiplexing of VBR Source: A Matrix-Analytic Approach. *Performance Evaluation*, 16:5–20, 1992.

[19] R.P. Brent. *Algorithms for Minimization without Derivatives*. Prentice-Hall, Englewood Cliffs, NJ, 1972.

[20] J. Cao, M. Andersson, C. Nyberg, and M. Kihl. Web Server Performance Modeling Using an M/G/1/K*PS Queue. In *Proceedings of 10th IEEE International Conference on Telecommunications*, volume 2, pages 1501–1506, 23 February–1 March 2003.

[21] R. Chakka. Spectral Expansion Solution for some Finite Capacity Queues. *Annals of Operations Research*, 79:27–44, 1998.

[22] R. Chakka and T. V. Do. Some new Markovian models for traffic and performance analysis in telecommunication networks, Tutorial Paper. In D. D. Kouvatsos, editor, *Proceedings of the Second International Working Conference on Performance Modelling and Evaluation of Heterogeneous Networks (HET-NETs 04)*, pages T6/1–31, Ilkley, UK, July 2004.

[23] R. Chakka, T. V. Do, and Z. Pandi. Exact Solution for the MM $\sum_{k=1}^{K} CPP_k/GE/c/L$ G-Queue and its Application to the Performance Analysis of an Optical Packet Switching Multiplexor. In *Proceedings of the 10th International Conference on Analytical and Stochastic Modelling Techniques and Applications, Nottingham, United Kingdom*, June 2003.

[24] R. Chakka, T.V. Do, and Z. Pandi. Generalized Markovian queues and applications in performance analysis in telecommunication networks. In D. D. Kouvatsos, editor, *the First International Working Conference on Performance Modelling and Evaluation of Heterogeneous Networks (HET-NETs 03)*, pages 60/1–10, July 2003.

[25] R. Chakka and P. G. Harrison. Analysis of MMPP/M/c/L queues. In *Proceedings of the Twelfth UK Computer and Telecommunications Performance Engineering Workshop*, pages 117–128, Edinburgh, 1996.

[26] R. Chakka and P. G. Harrison. The Markov modulated CPP/GE/c/L queue with positive and negative customers. In *Proceedings of the 7th IFIP ATM Workshop*, Antwerp, Belgium, 1999.

[27] R. Chakka and P. G. Harrison. A Markov modulated multi-server queue with negative customers - the MM CPP/GE/c/L G-queue. *Acta Informatica*, 37:881–919, 2001.

[28] R. Chakka and P. G. Harrison. A Markov Modulated Multi-server Queue with Negative Customers-The $MM\ CPP/GE/c/L$ G-Queue. *Acta Informatica*, 37:881–919, 2001.

[29] R. Chakka and P. G. Harrison. The MMCPP/GE/c queue. *Queueing Systems: Theory and Applications*, 38:307–326, 2001.

[30] R. Chakka and P. G. Harrison. The MMCPP/GE/c Queue. *Queueing Systems*, 38:307–326, 2001.

[31] Ram Chakka. *Performance and Reliability Modelling of Computing Systems Using Spectral Expansion*. PhD thesis, University of Newcastle upon Tyne (Newcastle upon Tyne), 1995.

[32] Ram Chakka and Tien Van Do. The $MM \sum_{k=1}^{K} CPP_k/GE/c/L$ G-Queue and Its Application to the Analysis of the Load Balancing in MPLS Networks. In *27th Annual IEEE Conference on Local Computer Networks (LCN 2002), 6-8 November 2002, Tampa, FL, USA, Proceedings*, pages 735–736, 2002.

[33] Ram Chakka and Tien Van Do. The MM $\sum_{k=1}^{K} CPP_k/GE/c/L$ G-Queue with Heterogeneous Servers: Steady state solution and an application to performance evaluation. *Performance Evaluation*, 64:191–209, March 2007.

[34] Ram Chakka, Tien Van Do, and Zsolt Pandi. A Generalized Markovian Queue and Its Applications to Performance Analysis in Telecommunications Networks. In D. Kouvatsos, editor, *Performance Modelling and Analysis of Heterogeneous Networks*, pages 371–387. River Publisher, 2009.

[35] Ram Chakka, Enver Ever, and Orhan Gemikonakli. Joint-state modeling for open queuing networks with breakdowns, repairs and finite buffers. In *15th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS)*, pages 260–266. IEEE Computer Society, 2007.

[36] Ram Chakka and Isi Mitrani. Multiprocessor systems with general breakdowns and repairs. In *SIGMETRICS*, pages 245–246, 1992.

[37] Ram Chakka and Isi Mitrani. Heterogeneous multiprocessor systems with breakdowns: Performance and optimal repair strategies. *Theor. Comput. Sci.*, 125(1):91–109, 1994.

[38] M. Chien and Y. Oruc. High performance concentrators and superconcentrators using multiplexing schemes. *IEEE Transactions on Communications*, 42:3045–3050, 1994.

[39] H-K. Choi and J. O. Limb. A Behavioral Model of Web Traffic. In *Proceedings of the Seventh IEEE International Conference on Network Protocols,(ICNP'99)*, pages 327–334, 1999.

[40] G. Ciardo and E. Smirni. ETAQA: An Efficient Technique for the Analysis of QBD-processes by Aggregation. *Performance Evaluation*, 36-37:71–93, 1999.

[41] D. Kouvatsos. Entropy Maximisation and Queueing Network Models. *Annals of Operations Research*, 48:63–126, 1994.

[42] J. Dilley, R. Friedrich, T. Jin, and J. Rolia. Measurement Tools and Modeling Techniques for Evaluating Web Server Performance. In *Proceedings of the 9th International Conference on Modelling Techniques and Tools for Computer Performance Evaluation* , June 3-6 1997.

[43] J. Dilley, R. Friedrich, T. Jin, and J. Rolia. Web Server Performance Measurement and Modeling Techniques. *Performance Evaluation*, 33:5–26, 1998.

[44] T. V. Do, B. Kálmán, C. Király, and Z. Pándi. A tool for the service planing and management of multi-layer networks. In *Networks 2000: 9th International Telecommunications Network Strategy and Planning Symposium, Canada*, 2000.

[45] Tien Van Do. An efficient computation algorithm for a multiserver feedback retrial queue with a large queueing capacity. *Applied Mathematical Modelling, http://dx.doi.org/10.1016/j.apm.2009.10.025.*

[46] Tien Van Do. An Efficient Computational Method for Retrial Queues to Cellular Mobile Systems with Guard Channels. *submitted for publications*, 2009.

[47] Tien Van Do. Comments on multi-server system with single working vacation. *Applied Mathematical Modelling*, 33(12):4435–4437, 2009.

[48] Tien Van Do. An Efficient Solution to a Retrial Queue for the Performability Evaluation of DHCP. *Computers & OR*, 37(7):1191–1198, 2010.

[49] Tien Van Do and Ram Chakka. A Direction Towards the Generalisation of QBD Processes and Applications for Performance Evaluation, Tutorial Paper. In D. D. Kouvatsos, editor, *Proceedings of the Second International Working Conference on Performance Modelling and Evaluation of Heterogeneous Networks (HET-NETs 06)*, pages T10/1–46, Ilkley, UK, June 2006.

[50] Tien Van Do and Ram Chakka. A New Performability Model for Queueing and FDL-related Burst Loss in Optical Switching Nodes. *Computer Communications*, 2009.

[51] Tien Van Do, Ram Chakka, and Peter G. Harrison. An integrated analytical model for computation and comparison of the throughputs of the UMTS/HSDPA user equipment categories. In *MSWiM '07: Proceedings of the 10th ACM Symposium on Modeling, analysis, and simulation of wireless and mobile systems*, pages 45–51, New York, NY, USA, 2007. ACM.

[52] Tien Van Do, Nam H. Do, and Ram Chakka. Performance evaluation of the high speed downlink packet access in communications networks based on high altitude platforms. In Khalid Al-Begain, Armin Heindl, and Miklós Telek, editors, *ASMTA*, volume 5055 of *Lecture Notes in Computer Science*, pages 310–322. Springer, 2008.

[53] Tien Van Do and Udo R. Krieger. A performance model for maintenance tasks in an environment of virtualized servers. In *IFIP/TC6 NETWORKING 2009*, pages 931–942, Aachen, Germany, 5 2009.

[54] Tien Van Do, Udo R. Krieger, and R. Chakka. Performance modeling of an apache web server with a dynamic pool of service processes. *Telecommunication Systems*, 39(2):117–129, 2008.

[55] Tien Van Do, Denes Papp, Ram Chakka, and Mai X T Truong. A Performance Model of MPLS Multipath Routing with Failures and Repairs of the LSPs. In D. Kouvatsos, editor, *Performance Modelling and Analysis of Heterogeneous Networks*, pages 27–43. River Publisher, 2009.

[56] Ma Jose Domenech-Benlloch, Jose Manuel Gimenez-Guzman, Vicent Pla, Jorge Martinez-Bauset, and Vicente Casares-Giner. Generalized truncated methods for an efficient solution of retrial systems. *Mathematical Problems in Engineering*, 2008, 2008.

[57] B. T. Doshi. Queueing systems with vacations – a survey. *Queueing Syst. Theory Appl.*, 1(1):29–66, 1986.

[58] Steve Drekic and Winfried K. Grassmann. An eigenvalue approach to analyzing a finite source priority queueing model. *Annals OR*, 112(1-4):139–152, 2002.

[59] R. Droms. Dynamic Host Configuration Protocol. RFC 2131 (Draft Standard), March 1997. Updated by RFCs 3396, 4361.

[60] R. V. Evans. Geometric Distribution in some Two-dimensional Queueing Systems. *Operations Research*, 15:830–846, 1967.

[61] Enver Ever, Orhan Gemikonakli, and Ram Chakka. A mathematical model for performability of beowulf clusters. In *Annual Simulation Symposium*, pages 118–126. IEEE Computer Society, 2006.

[62] Enver Ever, Orhan Gemikonakli, and Ram Chakka. Analytical modelling and simulation of small scale, typical and highly available beowulf clusters with breakdowns and repairs. *Simulation Modelling Practice and Theory*, 17(2):327–347, 2009.

[63] G. I Falin and J. G. C. Templeton. *Retrial Queues*. Chapman & Hall, London, 1997.

[64] H. R. Gail, S. L. Hantler, and B. A. Taylor. Spectral analysis of M/G/1 type Markov chains. Technical Report RC17765, IBM Research Division, 1992.

[65] H. R. Gail, S. L. Hantler, and B. A. Taylor. Non-Skip-Free M/G/1 and G/M/1 Type Markov Chains . *Advances in Applied Probability*, 29:733–758, 1997.

[66] Winfried K. Grassmann. The use of eigenvalues for finding equilibrium probabilities of certain markovian two-dimensional queueing problems. *INFORMS Journal on Computing*, 15(4):412–421, 2003.

[67] Winfried K. Grassmann. The use of eigenvalues for finding equilibrium probabilities of certain markovian two-dimensional queueing problems. *INFORMS Journal on Computing*, 15(4):412–421, 2003.

[68] Winfried K. Grassmann and Steve Drekic. An analytical solution for a tandem queue with blocking. *Queueing System*, (1-3):221–235, 2000.

[69] B. Haverkort and A.Ost. Steady state analyses of infinite stochastic Petri nets: A comparison between the spectral expansion and the matrix geometric methods. In *Proceedings of the 7th International Workshop on Petri Nets and Performance Models*, pages 335–346, Saint Malo, France, 1997.

[70] B. Haverkort and A. Ost. Steady State Analysis of Infinite Stochastic Petri Nets: A Comparing between the Spectral Expansion and the Matrix Geometric Method. In *Proceedings of the 7th International Workshop on Petri Nets and Performance Models*, pages 335–346, 1997.

[71] R. Jafari and K. Sohraby. General Discrete-Time Queueing Systems with Correlated Batch Arrivals and Departures. In *Proceedings of IEEE INFOCOM*, volume 1, pages 181–188, 2000.

[72] R. Jafari and K. Sohraby. Combined M/G/1-G/M/1 type structured chains: a simple algorthmic solution and applications. In *Proceedings of IEEE INFOCOM*, volume 2, pages 1065–1074, 2001.

[73] Manas Khadilkar, Nick Feamster, Matt Sanders, and Russ Clark. Usage-based dhcp lease time optimization. In *IMC '07: Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, pages 71–76, New York, NY, USA, 2007. ACM.

[74] L. Kleinrock. On the modeling and analysis of computer networks. *Proceedings of the IEEE*, 81(8):1179–1191, August 1993.

[75] Leonard Kleinrock. *Queueing Systems. Vol I: Theory* . John Wiley & Sons, Inc, 1975.

[76] Troels Emil Kolding, Klaus Ingemann Pedersen, Jeroen Wigard, Frank Frederiksen, and Preben Elgaard Mogensen. High Speed Downlink Packet Access: WCDMA Evolution . *IEEE Vehicular Technology Society (VTS) News*, 50:4–10, 2003.

[77] D. Kouvatsos. Entropy maximisation and queueing network models. *Annals of Operations Research*, 48:63–126, 1994.

[78] D. D. Kouvatsos. A maximum entropy analysis of the G/G/1 Queue at Equilibrium. *Journal of Operations Research Society*, 39:183–200, 1998.

[79] U. R. Krieger, V. Naoumov, and D. Wagner. Analysis of a Finite FIFO Buffer in an Advanced Packet-Switched Network. *IEICE Trans. Commun.*, E81-B:937–947, 1998.

[80] B. K. Kumar, R. Rukmani, and V. Thangaraj. On multiserver feedback retrial queue with finite buffer. *Applied Mathematical Modelling*, 33(4):2062–2083, 2009.

[81] G. Latouche and V. Ramaswami. *Introduction to Matrix Analysis Methods in Stochastic Modeling*. ASA–SIAM Series on Statistics and Applied Probability, 1999.

[82] G. Latouche and V. Ramaswami. *Introduction to Matrix Analytic Methods in Stochastic Modeling*. ASA-SIAM Series on Statistics and Applied Probability, 1999.

[83] G. Latouche and V. Ramaswami. *Introduction to Matrix Analytic Methods in Stochastic Modeling*. ASA-SIAM Series on Statistics and Applied Probability. 1999.

[84] Guy Latouche and V. Ramaswami. A logarithmic reduction algorithm for quasi-birth-death processes. *Applied Probability*, pages 650–674, 1993.

[85] R. B. Lehoucq, D. C. Sorensen, and C. Yang. *ARPACK Users' Guide: Solution of Large-Scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods*. SIAM, ISBN-13: 978-0-898714-07-4, 1998.

[86] T. Lemon and B. Sommerfeld. Node-specific Client Identifiers for Dynamic Host Configuration Protocol Version Four (DHCPv4). RFC 4361 (Proposed Standard), February 2006.

[87] Chuen-Horng Lin and Jau-Chuan Ke. Multi-server system with single working vacation. *Applied Mathematical Modelling*, 33(7):2967–2977, 2009.

[88] W. Liu, X. Xu, and N. Tian. Stochastic decompositions in the M/M/1 queue with working vacations. *Oper. Res. Lett.*, 35:595–600, 2007.

[89] Xue Liu, Lui Sha, Yixin Diao, Steven Froehlich, Joseph L. Hellerstein, and Sujay Parekh. Online Response Time Optimization of Apache Web Server. In *Eleventh International Workshop on Quality of Service (IWQoS 2003)*, pages 461–478, 2003.

[90] Z. Liu, N. Niclausse, C. Jalpa-Villanueva, and S. Barbier. Traffic model and performance evaluation of web servers. Technical Report 3840, INRIA, 1999.

[91] Fumiaki Machihara and Midori Saitoh. Mobile customer model with retrials. *European Journal of Operational Research*, 189(3):1073–1087, 2008.

[92] Marco Ajmone Marsan, Giovanni De Carolis, Emilio Leonardi, Renato Lo Cigno, and Michela Meo. Efficient Estimation of Call Blocking Probabilities in Cellular Mobile Telephony Networks with Customer Retrials. *IEEE Journal on Selected Areas in Communications*, 19(2):332–346, 2001.

[93] D. A. Menasce. Web Server Software Architectures. *IEEE Internet Computing*, 7(6):78–81, November/December 2003.

[94] D. A. Menasce, R. Dodge, and D. Barbara. Perserving QoS of E-commerce Sites Through Self-Tunning: A Performance Model Approach. In *Proceedings of the ACM Conference on E-Commerce, Tampa, Florida, USA*, pages 224–234, October 14-17 2001.

[95] I. Mitrani and R. Chakka. Spectral Expansion Solution for a Class of Markov Models: Application and Comparison with the Matrix–Geometric Method. *Performance Evaluation*, 23:241–260, 1995.

[96] Isi Mitrani. Approximate solutions for heavily loaded markov-modulated queues. *Perform. Eval.*, 62(1-4):117–131, 2005.

[97] Isi Mitrani and Ram Chakka. Spectral expansion solution for a class of Markov models: Application and comparison with the matrix-geometric method. *Performance Evaluation*, 23:241–260, 1995.

[98] D. Mosberger and T. Jin. httperf: A tool for measuring web server performance. In *First Workshop on Internet Server Performance (WISP 98)*, pages 59–67, ACM, June 1998.

[99] V. Naoumov, U. Krieger, and D. Wagner. Analysis of a Multi-server Delay-loss System with a General Markovian Arrival Process. In S.R. Chakravarthy and A.S. Alfa, editors, *Matrix-analytical methods in Stochastic models*, pages 43–66. Marcel Dekker, 1997.

[100] V. Naoumov, U. R. Krieger, and D. Warner. Analysis of a Multi–Server Delay–Loss System with a General Markovian Arrival Process. In *In S. R. Chakravarthy and A. S. Alfa, editors, Matrix–Analytic Methods in Stochastic Models, Marcel Dekker*, volume 183 of Lecture Notes in Pure and Applied Mathematics, September 1996.

[101] V. Naoumov, U. R. Krieger, and D. Warner. Analysis of a Multi-Server Delay-Loss System With a General Markovian Arrival Process. In S. R. Chakravarthy and A. S. Alfa, editors, *Matrix-analytic methods in stochastic models*, volume 183 of *Lecture Notes in Pure and Applied Mathematics*. Marcel Dekker, September 1996.

[102] M. F. Neuts. *Matrix Geometric Soluctions in Stochastic Model*. Johns Hopkins University Press, Baltimore, 1981.

[103] Z. Niu and Y. Takahashi. An Extended Queueing Model for SVC-based IP-over-ATM Networks and its Analysis. In *Proceedings of IEEE GLOBECOM*, pages 490–495, 1998.

[104] Papp Dénes and Truong Thi Xuan Mai and Do Van Tien. MPLS többutas elvezetés teljesítőképességi elemzése. *Magyar Távközlés*, May 2006.

[105] P. Reeser and R. Hariharan. Analytical Model of Web Servers in Distributed Environments. In *Proceedings of the 2nd ACM International Workshop on Software and Performance WOSP'2000, Ottawa, Canada*, pages 158–167, Sept. 17-20 2000.

[106] Emilia Rosti, Evgenia Smirni, and Kenneth C. Sevcik. On processor saving scheduling policies for multiprocessor systems. *IEEE Trans. Comp*, 47:47–2, 1998.

[107] J. Roszik and J. Sztrik. Performance analysis of finite-source retrial queues with non-reliable heterogeneous servers. *Journal of Mathematical Sciences*, (146):6033–6038, 2007.

[108] L. P. Seelen. An Algorithm for Ph/Ph/c queues. *European Journal of Operational Research*, 23:118–127, 1986.

[109] L. D. Servi and S. G. Finn. M/M/1 queues with working vacations (M/M/1/WV). *Perform. Eval.*, 50(1-4):41–52, 2002.

[110] Leslie D. Servi. Algorithmic solutions to two-dimensional birth-death processes with application to capacity planning. *Telecommunication Systems*, 21(2):205–212, 2002.

[111] Charalambos Skianis and Demetres Kouvatsos. An Information Theoretic Approach for the Performance Evaluation of Multihop Wireless Ad Hoc Networks . In D. D. Kouvatsos, editor, *Proceedings of the Second International Working Conference on Performance Modelling and Evaluation of Heterogeneous Networks (HET-NETs 04)*, pages P81/1–13, Ilkley, UK, July 2004.

[112] L. P. Slothouber. A Model of Web Server Performance. In *http://www.geocities.com/webserverperformance*, 1995.

[113] M. S. Squillante, D. D. Yao, and L. Zhang. Web Traffic Modeling and Web Server Performance Analysis. In *Proceedings of the 38th IEEE International Conference on Decision & Control, Phoenix, Arizona, USA*, pages 4432–4439, December 1999.

[114] M. S. Squillante, D. D. Yao, and L. Zhang. Web Traffic Modeling and Web Server Performance Analysis. *ACM SIGMETRICS Performance Evaluation Review*, 27(3):24–27, December 1999.

[115] J. Sztrik. Tool supported performance modelling of finite-source retrial queues with breakdowns. *Publicationes Mathematicae*, (66):197–211, 2005.

[116] N. Tian and Z. G. Zhang. Stationary Distributions of GI/M/c Queue with PH Type Vacations. *Queueing Syst. Theory Appl.*, 44(2):183–202, 2003.

[117] Hung T. Tran and Tien Van Do. Generalised invariant Subspace based Method for Steady State Analysis of QBD-M Proceses. *Periodica Polytechnica, Ser. El. Eng.*, 44:159–178, 2000.

[118] Hung T. Tran and Tien Van Do. Comparison of some Numerical Methods for QBD-M Processes via Analysis of an ATM Concentrator. In *Proceedings of 20th IEEE International Performance, Computing and Communications Conference, IPCCC 2001*, Pheonix, USA, 2001.

[119] Hung Tuan Tran. *Applied Methods for some Planning and Analysis Problems in Telecommunications Networking* . PhD thesis, Budapest University of Technology and Economics, 2003.

[120] Hung Tuan Tran and Tien Van Do. Computational Aspects for Steady State Analysis of QBD Processes . *Periodica Polytechnica, Ser. El. Eng*, pages 179–200, 2000.

[121] Phuoc Tran-Gia and Michel Mandjes. Modeling of customer retrial phenomenon in cellular mobile networks. *IEEE Journal on Selected Areas in Communications*, 15(8):1406–1414, 1997.

[122] R. Usmani. Inversion of a tridiagonal Jacobi matrix. *Linear Algebra Appl.*, 212/213:413–414, 1994.

[123] V. L. Wallace. *The Solution of Quasi Birth and Death Processes Arising from multiple Access Computer Systems.* PhD thesis, University of Michigan, 1969.

[124] Adam Wierman, Takayuki Osogami, Mor Harchol-Balter, and Alan Scheller-Wolf. How many servers are best in a dual-priority M/PH/k system? *Perform. Eval.*, 63(12):1253–1272, 2006.

[125] Patrick Wüchner, János Sztrik, and Hermann de Meer. Finite-source M/M/S retrial queue with search for balking and impatient customers from the orbit. *Computer Networks*, 53(8):1264–1273, 2009.

[126] Z. G. Zhang and N. Tian. Analysis on queueing systems with synchronous vacations of partial servers. *Perform. Eval.*, 52(4):269–282, 2003.

[127] Y. Zhao and Winfried K. Grassmann. A numerically stable algorithm for two server queue models. *Queueing Syst.*, 8(1):59–79, 1991.

[128] Z. Zsóka, L Jereb, T. Izsó, and F. Unghváry. FLEXPLANET, a Flexible Multi-Layer Network Design Tool. In *Networks 2008: 13th International Telecommunications Network Strategy and Planning Symposium*, 2008.