Annales Mathematicae et Informaticae
***36** (2009) pp. 61–69*
http://ami.ektf.hu

A computational algorithm for the CPP/M/c retrial queue

Tien Van Do

Department of Telecommunications
University of Technology and Economics, Budapest

Submitted 24 November 2008; Accepted 20 April 2009

Abstract

This paper introduces the retrial CPP/M/c queue, which is the generalization of the M/M/c retrial queue. The arrival process of jobs into the queue follows the Compound Poisson Process (CPP). We present an efficient and numerically stable computational algorithm for the steady state probabilities.

Keywords: retrial queue, computational algorithm

MSC: 60-08, 60J22

1. Introduction

Retrial queues have formed one of intensive research topics in the queueing theory [1, 2, 3, 4, 8, 10, 12, 14, 16, 17]. The popularity of retrial queues is explained by the fact that retrial queues can be used to model various problems in real systems such as telecommunication networks, wireless networks and computer systems.

It is well-known that the main M/M/c retrial queue (where the retrial rate depends on the number of customers in the orbit) with $c > 2$ is mathematically untractable. The stationary distributions of the main M/M/c retrial queue with $c > 2$ can be computed using approximation techniques [8]. Falin and Templeton proposed a truncation model and a numerical tractable with a threshold in their book [8].

This paper generalizes the numerical tractable M/M/c retrial queue (where the retrial rate is independent of the number of customers in the orbit). We introduce the retrial CPP/M/c queue with batch arrivals following the Compound Poisson Process (CPP), where the interarrival times have the Generalized Exponential (GE) distribution. Note that the GE is the only distribution of least bias [9], if only the mean and variance are reliably computed from the measurement data. It has been

shown in the recent work [7] that the CPP is accurate enough to model Internet traffic (i.e.: CPP parameters were estimated from the captured Internet traffic) and to be used for the performance evaluation in telecommunication systems. We provide a stable computational algorithm for the proposed queue.

In Section 2 we give a description for the CPP/M/c retrial queue. In Section 3 we provide a computational algorithm. In Section 4 we show that our proposed algorithm finds the eigenvalue when the existing approach fails.

2. The CPP/M/c Retrial Queue

Request arrivals follow the CPP with parameter (λ, ω) $(0 \leqslant \omega < 1)$. That is, the inter- arrival time probability distribution function is $1 - (1-\omega)e^{-\lambda t}$. Thus, the arrival *point*-processes can be seen as batch-Poisson, with batches arriving at each point having geometric size distribution. The probability that a batch is of size s is $(1-\omega)\omega^{s-1}$.

The following notations are introduced.

- c is the number of servers.
- $I(t)$ denotes the number of busy servers at time t. Note that $I(t)$ varies within interval $[0, c]$.
- $J(t)$, which takes a value from 0 to ∞, represents the number of requests in the orbit at time t.

Service times are exponentially distributed with parameter μ. Clients which wait in the orbit retrial with rate ν (i.e.: the inter-repetition times are exponentially distributed with parameter ν). As a consequence, the system is modeled by Continuous Time Markov Chain (CTMC) $Y = \{I(t), J(t)\}$ with state space $\{0, 1, \ldots, c\} \times \{0, 1, \ldots\}$. We denote the steady state probabilities by $\pi_{i,j} = \lim_{t\to\infty} Prob(I(t) = i, J(t) = j)$, and introduce $\mathbf{v}_j = (\pi_{0,j}, \ldots, \pi_{c,j})$.

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) $(0 \leqslant i,k \leqslant c; j = 0, 1, \ldots)$, which is caused by either the departure or the arrival of customers. Matrix A_j is defined as the matrix with elements $A_j(i,k)$.

$$A_j = A = \begin{bmatrix} 0 & \lambda(1-\omega) & \lambda(1-\omega)\omega & \ldots & & & \lambda(1-\omega)\omega^{c-1} \\ \mu & 0 & \lambda(1-\omega) & \ldots & & & \lambda(1-\omega)\omega^{c-2} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & & \ldots & (c-1)\mu & 0 & \lambda(1-\omega) \\ 0 & 0 & & \ldots & 0 & c\mu & 0 \end{bmatrix}$$

$\forall j \geqslant 0.$

(b) $B_{j,s}(i,k)$ represents s-steps upward transition from state (i,j) to state $(k, j+s)$ $(0 \leqslant i,k \leqslant c; s \geqslant 1; j = 0,1,\ldots)$, which is due to the arrival of customers. In the similar way, matrix $B_{j,s}$ (B_s) with elements $B_{j,s}(i,k)$ is defined as

$$B_{j,s} = B_s = \begin{bmatrix} 0\,0\,0\ldots 0\,0 & \lambda(1-\omega)\omega^{s+c-1} \\ 0\,0\,0\ldots 0\,0 & \lambda(1-\omega)\omega^{s+c-2} \\ \vdots\,\vdots\,\vdots\,\vdots\,\vdots & \vdots \\ 0\,0 \quad \ldots 0\,0 & \lambda(1-\omega)\omega^{s} \\ 0\,0 \quad \ldots 0\,0 & \lambda(1-\omega)\omega^{s-1} \end{bmatrix} \quad \forall j \geqslant 0; s \geqslant 1.$$

(c) $C_j(i,k)$ is the transition rate from state (i,j) to state $(k,j-1)$ $(0 \leqslant i,k \leqslant c; j = 0,1,\ldots)$, which is due to the successful retry from the orbit. Matrix C_j $(\forall j \geqslant 1)$ with elements $C_j(i,k)$ is written as

$$C_j = C = \begin{bmatrix} 0\,\nu\,0\ldots 0\,0\,0 \\ 0\,0\,\nu\ldots 0\,0\,0 \\ \vdots\,\vdots\,\vdots\,\vdots\,\vdots\,\vdots\,\vdots \\ 0\,0 \quad \ldots 0\,0\,\nu \\ 0\,0 \quad \ldots 0\,0\,0 \end{bmatrix} \quad \forall j \geqslant 1.$$

D^A and D^C denotes diagonal matrices whose diagonal elements are the sum of the elements in the row of A and C. The following matrices are also introduced,

$$A^* = A - D^A,$$

$$\Lambda = Diag[\lambda\omega^c, \ldots, \lambda\omega, \lambda].$$

3. A Computational Procedure

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

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

For $j \geqslant 2$, we have

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

therefore,

$$\mathbf{v}_{j-1} B_1 + \mathbf{v}_j \left[A^* - \Lambda - D^C\right] + \mathbf{v}_{j+1} C - \mathbf{v}_{j-1} \left[A^* - \Lambda - D^C\right] \omega - \mathbf{v}_j C\omega = 0,$$

$$\mathbf{v}_{j-1}(B_1 - \left[A^* - \Lambda - D^C\right] \omega) + \mathbf{v}_j(\left[A^* - \Lambda - D^C\right] - C\omega) + \mathbf{v}_{j+1} C = 0.$$

So, we arrive at the Quasi-Birth-and-Death (QBD) form as follows

$$\mathbf{v}_{j-1}Q_0 + \mathbf{v}_j Q_1 + \mathbf{v}_{j+1}Q_2 = 0 \ \ (j \geqslant 2), \tag{3.1}$$

where $Q_0 = (B_1 - \left[A^* - \Lambda - D^C\right]\omega)$, $Q_1 = (\left[A^* - \Lambda - D^C\right] - C\omega)$, $Q_2 = C$. Note that $Q(x) = Q_0 + Q_1 x + Q_2 x^2$ is defined as the characteristic matrix polynomial associated with equations (3.1). Due to the QBD form, the steady state probabilities can be obtained with the existing methods like the matrix-geometric and its variants [6, 11, 15], and the spectral expansion [13]. However, the existing methods have the numerical problem (no results due to a very long-running time of computer programs implementing these methods) when c is large (the problem starts when c reaches a value of several hundreds). Therefore, in what follows we present a fast computational procedure to find the steady state probabilities.

We have

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

where

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

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 $det[Q(x)] = 0$ (c.f. [13]). Thus, the straightforward way to obtain the steady state probabilities is to find the eigenvalues of $Q(x)$ (see [5] for the methodology to find the eigensystem of the characteristic matrix polynomial). However, there exists an efficient method.

It is easy to see that $Q(x)$ has c eigenvalues of value ω. The corresponding independent eigenvectors for c eigenvalues are $\boldsymbol{\psi}_1 = \{1, 0, \dots, 0\}$, $\boldsymbol{\psi}_3 = \{0, 1, 0, \dots, 0\}$, …, $\boldsymbol{\psi}_c = \{0, 0, \dots, 1, 0\}$. Note that if the system is ergodic, then the number of eigenvalues of $Q(x)$, which are inside the unit disk, is $c+1$. Therefore, $Q(x)$ should have another eigenvalue called x_0 inside the unit disk. Let ψ_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

$$\begin{aligned} \mathbf{v}_j &= b_0\boldsymbol{\psi}_0 x_0^j + \omega^j \sum_{i=1}^{c} b_i \boldsymbol{\psi}_i \ (j \geqslant 1) \\ &= b_0\boldsymbol{\psi}_0 x_0^j + \omega^j \mathbf{b}, \end{aligned} \tag{3.2}$$

where b_i are the coefficients to be determined and $\mathbf{b} = \sum_{i=1}^{c} b_i\psi_i = \{b_1, b_2, \dots, b_c, 0\}$.

Since the probabilities are greater than or equal to zero, $0 < x_0 < 1$ holds. Furthermore, $x_0 \neq \omega$ should hold to ensure that (c, j) states are reachable. It is observed that the key step towards the steady state probabilities is to determine x_0 and the corresponding eigenvector $\boldsymbol{\psi}_0$.

Theorem 3.1. *$0 < x_0 < 1$ 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

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

$$U(x_0) = \begin{bmatrix} 1 & u_{1,2} & \dots & u_{1,c-2} & u_{1,c} & u_{1,c+1} \\ 0 & 1 & u_{2,3} \dots & u_{2,c} & u_{2,c+1} \\ \vdots & \vdots & \vdots\ \vdots\ \vdots & \vdots & \vdots & \\ 0 & 0 & \dots 0 & 1 & u_{c,c+1} \\ 0 & 0 & \dots 0 & 0 & 1 \end{bmatrix},$$

where

$$\begin{aligned} l_1(x_0) &= q_{1,1}(x_0) = (\lambda + \nu)(\omega - x), \\ u_{1,i} &= q_{1,i}(x_0)/l_1(x_0) \ \ (i = 2, \dots, c+1), \\ u_{j,i} &= (q_{j,i}(x_0) - q_{j,j-1}(x_0)u_{j-1,i})/l_j(x_0); (i = 2, \dots, c+1;\ j = 2, \dots, i-1), \\ l_i(x_0) &= q_{i,i}(x_0) - q_{i,i-1}u_{i-1,i} \ \ (i = 2, \dots, c+1). \end{aligned}$$

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

$$Det[Q(x_0)] = Det[L(x_0)]Det[U(x_0)] = \prod_{i=1}^{c+1} l_i(x_0) \tag{3.3}$$

As the consequence of equation (3.3), we have $l_i(x_0) \neq 0$ $(1 < i \leqslant c)$. Hence, $Det[Q(x_0)] = 0$ follows $l_{c+1}(x_0) = 0$. □

It is also easy to prove that $l_{c+1}(0)$ is positive and $l_{c+1}(1)$ is negative. Therefore, a bisection algorithm in Figure 1 can be proposed to determine x_0 and $\boldsymbol{\psi}_0 = \{\psi_{0,1}, \psi_{0,2}, \dots, \psi_{0,c+1}\}$.

In what follows, we present a method to determine $\mathbf{b}$ and b_0. First, we prove that $\mathbf{b} = 0$ holds. We have $\boldsymbol{\psi}_0 Q(x_0) = 0$ because $(x_0, \boldsymbol{\psi}_0)$ is a eigenvalue/vector pair of $Q(x)$. This means,

$$\boldsymbol{\psi}_0(B_1 - [A^* - \Lambda - D^C]\omega + x_0([A^* - \Lambda - D^C] - C\omega) + x_0^2 C) = 0.$$

Algorithm 1 Bisection algorithm to determine x_0 and the calculation of ψ_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 (3.3)
  if $l_{c+1}(x_0) > 0$ **then**
    $x_{0,d} = x_0$
  else
    $x_{0,u} = x_0$
  end if
until $|l_{c+1}(x_0)| < \epsilon$
$\psi_{0,1} = 1$
for $i = 1$ to c **do**
  $\psi_{0,i+1} = \frac{\sum_{j=1}^{i} \psi_{0,i} q_{j,i}(x_0)}{i\mu(\omega - x_0)}$
end for
return x_0, ψ_0

After a simple algebra, we obtain

$$\boldsymbol{\psi}_0 B_1 + (x_0 - \omega)\boldsymbol{\psi}_0([A^* - \Lambda - D^C] + Cx_0) = 0. \tag{3.4}$$

$\boldsymbol{\psi}_0 B_1$ is a row vector with the first c zero-elements because B_1 is the matrix with the last nonzero-column. Therefore, due to (3.4), vector $\boldsymbol{\psi}_0([A^* - \Lambda - D^C] + Cx_0)$ should have the first c elements equal to zero.

We can write the balance equation for level 0 as

$$\mathbf{v}_0 [A^* - \Lambda] + \mathbf{v}_1 C = 0,$$

which follows

$$\begin{aligned} \mathbf{v}_0 &= \mathbf{v}_1 C[\Lambda - A^*]^{-1} \\ &= (b_0 \boldsymbol{\psi}_0 x_0 + \omega \mathbf{b}) C[\Lambda - A^*]^{-1}. \end{aligned} \tag{3.5}$$

Substituting (3.5) into the balance equation for level $J = 1$,

$$\mathbf{v}_0 B_1 + \mathbf{v}_1 \left[A^* - \Lambda - D^C\right] + \mathbf{v}_2 C = 0,$$

we obtain

$$\mathbf{v}_1 (C \left[\Lambda - A^*\right]^{-1} B_1 + \left[A^* - \Lambda - D^C\right]) + \mathbf{v}_2 C = 0.$$

Using (3.2), we get the following expression for $\mathbf{b}$ after some algebraic steps

$$\begin{aligned} (b_0 \boldsymbol{\psi}_0 x_0 + \omega \mathbf{b})(C[\Lambda - A^*]^{-1} B_1 + \left[A^* - \Lambda - D^C\right]) + (b_0 \boldsymbol{\psi}_0 x_0^2 + \omega^2 \mathbf{b}) C = 0, \\ b_0 \boldsymbol{\psi}_0 x_0 (C \left[\Lambda - A^*\right]^{-1} B_1 + \left[A^* - \Lambda - D^C\right] + x_0 C) + \end{aligned}$$

$$\begin{aligned}
\omega\mathbf{b}(C\left[\Lambda - A^*\right]^{-1}B_1 + \left[A^* - \Lambda - D^C\right] + \omega C) = 0,\\
-b_0\boldsymbol{\psi}_0 x_0(C[\Lambda - A^*]^{-1}B_1 + \left[A^* - \Lambda - D^C\right] + x_0 C) =\\
\omega\mathbf{b}(C[\Lambda - A^*]^{-1}B_1 + \left[A^* - \Lambda - D^C\right] + \omega C),\\
\mathbf{b} = -(b_0/\omega)\boldsymbol{\psi}_0 x_0(C[\Lambda - A^*]^{-1}B_1 + \left[A^* - \Lambda - D^C\right] + x_0 C)\\
(C(\Lambda - A^*)^{-1}B_1 + (A^* - \Lambda - D^C) + \omega C)^{-1}.
\end{aligned}$$

It is observed that $\boldsymbol{\psi}_0 x_0 C[\Lambda - A^*]^{-1}B_1$ is a row vector with the first c elements equal to zero because B_1 is the matrix with the last nonzero-column and recall that vector $\boldsymbol{\psi}_0([A^* - \Lambda - D^C] + Cx_0)$ has the first c elements equal to zero. As consequence $\mathbf{b}$ is the vector with the first c elements equal to zero, which means $\mathbf{b}$ is a zero-vector.

To determine coefficient b_0, we use the normalisation equation

$$1 = \sum_{i=0}^{c}\sum_{j=0}^{\infty}\pi_{i,j} = \mathbf{v}_0\mathbf{e} + \frac{b_0 x_0}{1 - x_0}\boldsymbol{\psi}_0\mathbf{e} = b_0 x_0\boldsymbol{\psi}_0 C[\Lambda - A^*]^{-1}\mathbf{e} + \frac{b_0 x_0}{1 - x_0}\boldsymbol{\psi}_0\mathbf{e}.$$

4. Numerical Example

The proposed procedure is implemented in `Mathematica` (`http://www.wolfram.com`). We compare our algorithm and the solution of equation $det[Q(x)] = 0$ (i.e.: the direct way to determine the eigenvalues of the characteristic polynomial) with the following parameter values $\nu = 20$, $\omega = 0.26$, $\lambda = 2.3$ and $\mu = 1.0$. It is observed that our algorithm gives a correct result for root x_0 for all cases, while the direct solution of equation $det[Q(x)] = 0$ in Mathematica using a built-in function is not always correct.

- The built-in function of Mathematica finds that $det[Q(x)]$ has roots $x \to 0.26$, $x \to 0.26$, $x \to 0.26$, $x \to 0.26$, $x \to 0.859258$, $x \to 1$ and $x \to 10.3527$ when $c = 4$ holds. Our algorithm finds x_0 equal to 0.859258.
- With the built-in function of Mathematica $det[Q(x)]$ has roots $x \to 0.26 - 1.63875 \cdot 10^{-7}i$, $x \to 0.26 + 1.63875 \cdot 10^{-7}i$, $x \to 0.26$, $x \to 0.26$, $x \to 0.26$, $x \to 0.736433$, $x \to 1$, $x \to 5.93992$ and $x \to 55.9869$ when $c = 5$ holds. Note that $Q(x)$ does not have a complex eigenvalue in this case. Our algorithm results in $x_0 = 0.736433$.

The numerical results confirm a claim that we have developed a numerically stable algorithm for the solution the CPP/M/c retrial queue.

References

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

[2] Artalejo, J.R., Economou, A., Lopez-Herrero, M.J., Algorithmic approximations for the busy period distribution of the M/M/c retrial queue, *European Journal of Operational Research*, 176:1687–1702, 2007.

[3] Artalejo, J.R., Pozo, M., Numerical Calculation of the Stationary Distribution of the Main Multiserver Retrial Queue, *Annals of Operations Research*, 1-4:41–56, 2002.

[4] Artalejo, J.R., Gómez-Corral, A., Retrial Queueing Systems, *Springer*, 2008.

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

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

[7] Do, T.V., Chakka, R., Harrison, P.G., 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.

[8] Falin, G.I., Templeton, J.G.C., Retrial Queues, *Chapman & Hall*, London, 1997.

[9] Kouvatsos, D., Entropy maximisation and queueing network models, *Annals of Operations Research*, 48:63–126, 1994.

[10] Kumar, B., Raja, J., On multiserver feedback retrial queues with balking and control retrial rate, *Annals of Operations Research*, 141:211–232(22), January 2006.

[11] Latouch, G., Ramaswami, V., A logarithmic reduction algorithm for quasi-birth-death processes, *Applied Probability*, pages 650–674, 1993.

[12] Lopez-Herrero, M., A maximum entropy approach for the busy period of the M/G/1 retrial queue, *Annals of Operations Research*, 141:271–281(11), January 2006.

[13] Mitrani, I., Chakka, R., Spectral expansion solution for a class of Markov models: Application and comparison with the matrix-geometric method, *Performance Evaluation*, 23:241–260, 1995.

[14] Mushko, V., Jacob, M., Ramakrishnan, K., Krishnamoorthy, A., Dudin, A., Multiserver queue with addressed retrials, *Annals of Operations Research*, 141:283–301(19), January 2006.

[15] Naoumov, V., Krieger, U., Wagner, D., Analysis of a Multi-server Delay-loss System with a General Markovian Arrival Process. In S.R. Chakravarthy and A.S. Alfa, editors, *Matrixanalytical methods in Stochastic models*, pages 43–66. Marcel Dekker, 1997.

[16] Roszik, J., Sztrik, J., Performance analysis of finite-source retrial queues with non-reliable heterogeneous servers, *Journal of Mathematical Sciences*, (146):6033–6038, 2007.

[17] Sztrik, J., Tool supported performance modelling of finite-source retrial queues with breakdowns, *Publicationes Mathematicae*, (66):197–211, 2005.

Tien Van Do
Department of Telecommunications
University of Technology and Economics
Budapest
Magyar tudósok krt. 2.
H-1117, Hungary
e-mail: do@hit.bme.hu