Performance Modeling of Bandwidth Aggregation for TCP Connections

E. Elahi[1,*], N. Hassan[2,+], S. Asghar[3,+], A. Qayyum[4,+], A. Rafiq[5,*]
[*]*University of Sargodha, Sargodha, Pakistan* [+]*M. A. Jinnah University, Islamabad, Pakistan*
[1]*ehsan@corenet.org.pk,* [2]*sardarnajam@yahoo.com,* [3]*sohail.asghar@jinnah.edu.pk,* [4]*aqayyum@ieee.org,*
[5]*arafiq@uos.edu.pk*

Abstract

The proliferation of handheld devices with the support of multiple interfaces makes it a common requirement of users to have access to multiple access networks simultaneously in order to obtain increased performance. Aggregating bandwidth of two or more Internet connections makes Internet applications to use the total available bandwidth in order to increase the goodput and reliability with link redundancy. In this paper, a Bandwidth Aggregation System (BAS) along with its implementation is discussed. Moreover a queuing theory based analytical model to derive the performance for aggregating bandwidth using multiple available interfaces for TCP connections is presented.

Next, that model has been used to derive an expression for average data transmission rate when BAS is being used. In the end the performance of the model is validated by simulating the BAS and comparing the model predicted results with the experimental results. The comparison shows excellent agreement.

Keywords - Bandwidth Aggregation, Queuing Theory, Performance Modeling,

1. Introduction

The growth of Internet in recent years has proved to be a major driving force towards the tremendous increase in various wireless technologies to access the Internet. These technologies include GPRS, EDGE, CDMA2000, IEEE 802.11, UMTS etc. With the increasing demand of the users and the availability of service providers using different wireless technologies gives a push to the mobile devices manufacturers to make such devices which are equipped with more than one interface. It is now common that modern mobile terminals possess multiple network interfaces such as WLAN, Ethernet, GPRS, and optionally WiMax interfaces. In literature several architectures and protocols [1][2] are available which exploit the use of multiple interfaces in such 4G heterogeneous network environment. Most of the protocols are confined to propose the solution for vertical handovers and very few proposes the mechanism for BAS for example [1][3][4]. Bandwidth Aggregation is a technique through which users can accumulate the available bandwidth of the multiple available networks in order to increase the throughput and improved quality of experience (QoE). So for various solutions have been provided to handle the data stripping for bandwidth aggregation. These solutions can be categorized according to the layer of TCP/IP protocol stack on which these solutions operate. Providing the BAS service at transport layer neither need any changes in the existing protocol stack nor require any extra entity in the network thus providing application transparency.

The services offered by different wireless technologies such as data rate, QoS, pricing, coverage etc. are widely different to each other. The user has its own priorities to choose one or more network interfaces based upon the offered services. A BAS takes data from application, store it in its own buffers, strip it for multiple available networks according to the user preferences and send to the destination using multiple network interfaces. For sending data on multiple interfaces, BAS needs to create multiple TCP connections over each link with the destination counterpart of BAS. Each send buffer of TCP connection and one send buffer of BAS cumulatively represents the system as a network of queues in which data arrival rate and departure rate over multiple connections are all time varying.

In this paper, we present an implementation of a Bandwidth Aggregation system and its performance modeling using an analytical queuing model. The rest of the paper is organized as follows. Section 2 presents an overview of BAS and its implementation. In section 3 a survey of existing queuing models for TCP is presented. In section 4 a queuing model is derived from generic models for BAS and certain performance measures are discussed. In section 5 an experimental setup is described. In section 6 the accuracy of the derived queuing model is corroborated comparing the experiential observations with the model's predicted values. The paper is concluded in section 7.

2. Bandwidth Aggregation System (BAS)

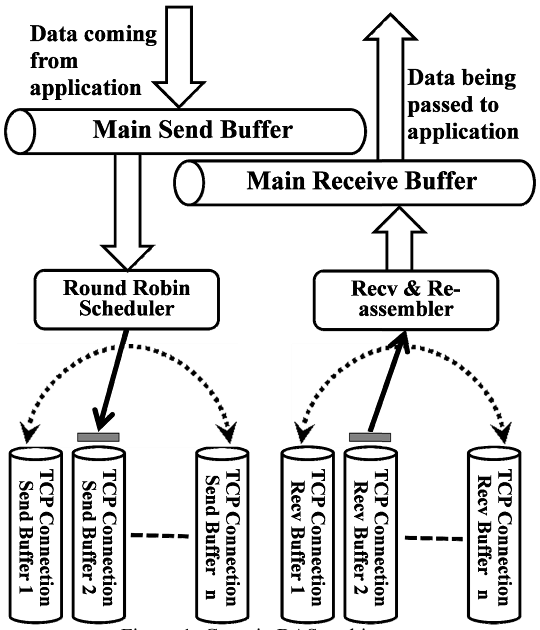
Data coming from application
Data being passed to application
Main Send Buffer
Main Receive Buffer
Round Robin Scheduler
Recv & Re-assembler
TCP Connection Send Buffer 1
TCP Connection Send Buffer 2
TCP Connection Send Buffer n
TCP Connection Recv Buffer 1
TCP Connection Recv Buffer 2
TCP Connection Recv Buffer n

Figure 1: Generic BAS architecture

Consider a scenario in which a user is downloading a 100MB file from a remote server. It takes approximately 14 minutes to complete the download over a link with 1Mbps bandwidth. Now if the user has two distinct links with 1Mbps bandwidth each and download half part of the same 100MB file on each link, the file will be completed in approximately half time. This happens with the help of a BAS. The BAS takes data from the application, append its own header and send to its counterpart at destination node through one of the TCP connections. The BAS header contains an association identifier to represent the association between two hosts and a sequence number. As the data packets of the same connection may reorder while sent over multiple paths due to the variable link characteristics therefore a sequence number at this level is necessary. The association identifier has the role of a de-multiplexer as there may be more than one associations being run on single BAS for multiple applications.

There are two main modules in a BAS as shown in Figure 1. The module that is responsible for data stripping, appending header and sending over different interfaces is called "BAS scheduler". The BAS scheduler maintains the observed throughput for each connection when sending data so that it can send data to each interface as per its observed throughput in order to avoid head of line blocking problem. There is another module in the system known as "BAS re-assembler". It is responsible for reception of data, detaching BAS header and passing to the receiving application if data received is in order otherwise it buffers data in it's receive buffer until it becomes in order. Architecture of a BAS is shown in Figure 1.

The protocol and computational overhead may be negligible because if the BAS is using 64 bit header with 32 bits for association identifier and 32 bits for sequence number field and if the average data size is 2KB then the protocol overhead will just be 0.4%. The BAS architecture presented in this paper is quite a generic architecture and can fit for any kind of protocol with similar objectives.

The advantages of a BAS being deployed above the transport layer and below application layer are that there is no need to re-implement TCP or any other protocol above or below this system. Moreover unlike [5], any extra entity in the network is not required as compared to such systems being deployed at the network layer. The generic architecture presented in this paper also does not require any changes in current socket API for application developers and also it can have support for legacy applications. The BAS described in [6] is also the session layer solution but it requires a new API and consequently could not serve the legacy applications. Although the system presented in [6] has the advantage of selecting the transport layer protocol based upon the application requirements.

The disadvantages of using such systems may be fairness issue at the shared bottleneck. As there is no such mechanism to ensure that the traffic of each sub-flow will be passed through distinct end-to-end paths so there may exists some single bottleneck on the way through which the traffic of more than one sub-flow may pass. The only assumption that there does not exist any shared bottleneck end-to-end is that the system will use multiple addresses at both endpoints to get different paths to the destination [7]. Although it is not guaranteed but path diversity and connection diversity can help in improving TCP throughput significantly.

The implementation of a BAS is fairly simple. The BAS is implemented by overloading the socket API's *socket, connect, listen, accept, send and recv* calls. Each *send* call just write the data into the *Main_Send_Buffer* if space is available while each *recv* call just read the data from the head of *Main_Recv_Buffer*. There are two main threads running in a BAS named as scheduler and re-assembler. Algorithm 1 describes the necessary steps involved in scheduler and Algorithm 2 describes the necessary steps involved in re-assembler. As mentioned earlier that in this paper we are presenting a very generic design of a BAS. The transport layer or session layer protocols implementing such a system may have their own connection/association establishment and termination procedures. Such types of mechanisms are out of scope of this paper.

Algorithm 1: *Data_Schedule(AID, CID[ ], PM[ ], c_ctr)*

Input: Association identifier, Array of connection descriptors under current association, Performance measure that is the observed throughput in previous iteration of each connection, Total number of connection under current association

Output: Strip data taken from Send_Buffer and send on each connection based upon each PM, Update PM

```
1: for i =1 to c_ctr do
2:     if PM[i] > Minimum_Data_Packet_Size then
3:         {Get data of size equal PM[i], append header and send on
             CID[i]}
4:         {Based upon return value of send() call, update PM[i]}
5:     else
6:         {Wait for certain number of iterations and then try to send
             data packet of Minimum_Data_Packet_Size}
7:         {Based upon return value of send() call, update PM[i]}
8:     end if
9: end for
```

Algorithm 2: *Data_Re-assemble(AID, CID[ ], c_ctr)*

Input: Association identifier, Array of connection descriptors under current association, Total number of connections under current association

Output: Data packets received from each connection, detach header and according to sequence number place in Main_Recv_Buffer

```
1: for i =1 to c_ctr do
2:     if socket CID[i] is readable then
3:         {receive packet, parse header, place in Main_Recv_Buffer at
             appropriate place according to the sequence number}
4:     end if
5: end for
```

3. Existing Queuing Models for TCP

There are numerous models available in the literature to predict the performance of TCP connections. Some of those are based upon queuing theory and some uses Markov Chains. Some of these models use few parameters from the IP networks for example RTT, packet loss ratio etc. as in [8] [9]. Some uses primitive network parameters for example data rates, buffer size, propagation delays etc. as in [10]. There are various approaches to model TCP performance. A detailed discussion on such approaches can be found in [11].

Usually for TCP connections, M/G/∞ queuing model is used because it is assumed that the arrival rate and the service rate remains constant in time when TCP connection is in steady state and theoretically there can be infinite number of servers in the system to serve TCP connections, since there is no limitation on the number of TCP connections in any given state within a system. If we look at the BAS in general as a network of queues then it could be argued that the arrival and departure rates are time varying and also there is finite number of TCP connections under one association. The number of these servers is also a time varying measure. So there are two most appropriate queuing models that may partially fit in this scenario are M(t)/M/c(t) [12] and M(t)/M(t)/c(t) [13]. In order to allow the arrival and service rates to vary with time, [12] replaced λ (arrival rate) with time varying function $\lambda(t)$ and [13] replaced μ (service rate) with time varying function $\mu(t)$.

4. Proposed Queuing Model

In this paper a simplified and generic queuing model of type *(M/M/c):(FCFS/N/∞)* is derived. It is assumed that for small intervals of time and in steady TCP conditions, the arrival and departure rates will be time in-varying and both follow the Poison distribution but the departure rate μ will be different for each individual TCP connection. For example in the BAS if two network interfaces are being used for bandwidth aggregation out of which one is of type GPRS, and the other one is of WLAN then the departure rate of GPRS and WLAN will be different. A very generic queuing model of this type is presented in [14]. The model is extended in such a way that the departure rate or service rate μ will be different for each connection instead of same as presented in [14]. Following are the definitions of basic variables being used to model a BAS at an instance of time t. It is assumed that at time instance t, the system will be in steady state:

n = number of data packets in the system (in main buffer plus all packets in TCP buffers)

λ_n= Arrival rate given n packets in the system

γ_i = Departure Rate of *ith* connection

μ_n= Cumulative departure rate given n packets in the system. It is the sum of all individual departure rates of c number of connections. It could be defined as:

$$\mu_n = \begin{cases} \sum_{i=1}^{n} \gamma_i & 0 \le n < c \\ \sum_{i=1}^{c} \gamma_i & c \le n \le N \end{cases} \qquad (1)$$

As N is the total number of packets that the system can accommodate therefore when the system approached this limit, it stops receiving more bytes from the application. In this case λ becomes zero. According to [14] λ_n can be defined as:

$$\lambda_n = \begin{cases} \lambda, & 0 \le n \le N \\ 0, & n > N \end{cases} \qquad (2)$$

For the system to work in steady state, following condition must hold:

$$\rho = \frac{\lambda}{\mu_n} < 1$$

$$\rho = \frac{\lambda}{\sum_{i=1}^{c} \gamma_i} \quad (3)$$

Let p_n be the probability that n packets are in the system then [14] defines it as:

$$p_n = \left(\frac{\lambda_{n-1}\lambda_{n-2}\lambda_{n-3}\ldots\ldots\lambda_0}{\mu_n \mu_{n-1} \mu_{n-2} \ldots\ldots \mu_1} \right) p_0 \quad (4)$$

Using equations (1), (2) and (3), p_n can be written as:

$$\left.\begin{aligned} p_n &= \left(\frac{\lambda^n}{\prod_{j=1}^{n} [\sum_{i=1}^{j} \gamma_i]} \right) p_0 \quad \text{for } 0 \le n < c \\ &\text{and} \\ p_n &= \left(\frac{\rho^{n-c}\lambda^c}{\prod_{j=1}^{c} [\sum_{i=1}^{j} \gamma_i]} \right) p_0 \quad \text{for } c \le n \le N \end{aligned}\right\} (5)$$

The expression for p_0 can be derived from $\sum_{n=0}^{N} p_n = 1$

So

$$p_0 = \frac{1}{\sum_{n=0}^{c-1} \left(\frac{\lambda^n}{\prod_{j=1}^{n} [\sum_{i=1}^{j} \gamma_i]} \right) + \left(\frac{\lambda^c}{\prod_{j=1}^{c} [\sum_{i=1}^{j} \gamma_i]} \right) \left(\frac{1-\rho^{N-c}}{1-\rho} \right)}$$

Next the most commonly used steady-state measures of performance in a queuing scenario are computed. Let N_q be the expected number of packets in queue. The packets start being queued in the *Main_data_buffer* when the number of packets in the system becomes more than the total number of packets which could be accommodated in all TCP send buffers. N_q can be computed as:

$$N_q = \sum_{n=S+1}^{N} (n - S) p_n$$
$$= \sum_{n=S+1}^{N} (n - S) \left(\frac{\rho^{n-c}\lambda^c}{\prod_{j=1}^{c} [\sum_{i=1}^{j} \gamma_i]} \right) p_0 \quad (6)$$

Here S is the total number of packets which could be accommodated in all TCP send buffers. Let N_s be the expected number of packets in the system that can be derived from N_q as follows:

$$N_s = N_q + c\rho \quad (7)$$

In order to derive an expression to calculate the average data transmission rate, we need to calculate the expected waiting time in queue, W_q, and expected waiting time in service, W_s. These performance measures could be derived using Little's law as explained in [14].

$$W_s = \frac{Ns}{\lambda_{\text{eff}}} \quad (8)$$

$$W_q = \frac{N_q}{\lambda_{\text{eff}}} \quad (9)$$

Where λ_{eff} [14] is the effective arrival rate in the system that may be equal to λ if all packets coming from application could be accommodated in the system and there may occurs a situation when some data packets could not be accommodated in the BAS then $\lambda > \lambda_{eff}$. λ_{eff} can be computed as follows:

$$\lambda_{\text{eff}} = \lambda - \lambda_{\text{lost}} \quad (10)$$
$$\lambda_{\text{lost}} = \lambda p_N$$

While λ_{lost} is the rate at which the application tries to send packets but the buffers in the BAS are already full. The packets per unit time average data transmission rate GP can be calculated as follows:

$$GP = \frac{N_q}{W_s} \quad (11)$$

The following sections describe how this model is validated with the actual results.

5. Experimental Setup

In order to validate our derived model, NCTUNS 5.0 [15] is used as a network simulator. Figure 2 shows the simulation scenario. In this simulation, a multi-interface mobile node acts as client and communicates with a fixed node acting as TCP server. The link between router (node 2) and fixed node (node 1) is of 10Mbps Ethernet link while the available links from mobile node (node 3) to the router through different access technologies is variable depending upon our scenario as shown in Table 1. A TCP based traffic generator and receiver applications along with BAS support were used on mobile node and fixed node respectively. The traffic generator connects with the receiver through each available interface and starts sending fixed sized packets of 2KB. The generator application uses fixed sized TCP connection send buffer for each connection and each buffer can accommodate maximum of 50 data packets. The main send buffer of BAS can accommodate maximum 150 packets. It means that if there are three number of interfaces being used ($c = 3$) then the system can accommodate maximum 300 packets so N will be 300.

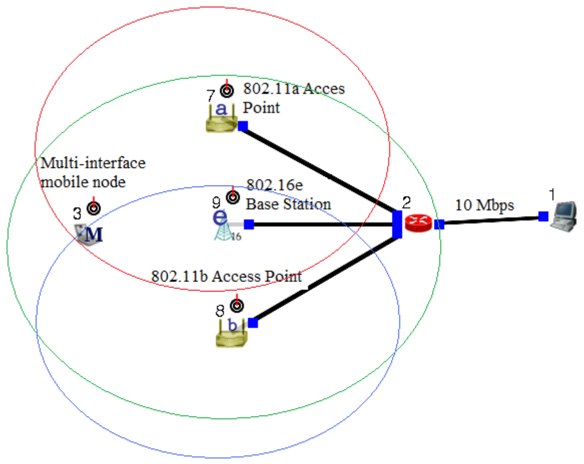

Figure 2: Simulation Scenario

6. Model Validation

The graph of our derived probability density function (pdf) for n number of packet in the system is shown in Figure 3 and its CDF is shown in Figure 4. There are total of six experiments conducted and results are compared with model predicted results. The summery of the experiments and comparison is presented in Table 1. The comparison is also presented in a graph as shown in Figure 5. The duration of each experiment is 100 seconds. The first experiment involved using three interfaces with 638pps (packets per seconds) of data arrival rate, λ, from application to BAS and cumulative data departure rate μ_n of all three connections as 640pps. In subsequent two experiments, the number of connections is gradually decreased. In the forth experiment, the transmission rate of each connection is decreased to almost half and then results are gathered as show in Table 1. In all six experiments, the observed and predicted values for data transmission rates show an excellent agreement. Here one point is to be noted about the gradual decrease in value of λ. As mentioned earlier that for a queuing system to work under steady-state condition, the arrival rate must be kept less than the service rate so that $\rho = \lambda/\mu_n < 1$. Therefore the value of λ is kept less than the total departure rate μ_n. If λ is greater than μ_n then value of λ_{lost} and N_q increases but the average data transmission rate will always be less than or ideally equal to μ_n. If $\lambda \ll \mu_n$ then the N_q and W_q becomes negligible because the probability that a packet arrives and found each connection buffer full becomes negligible.

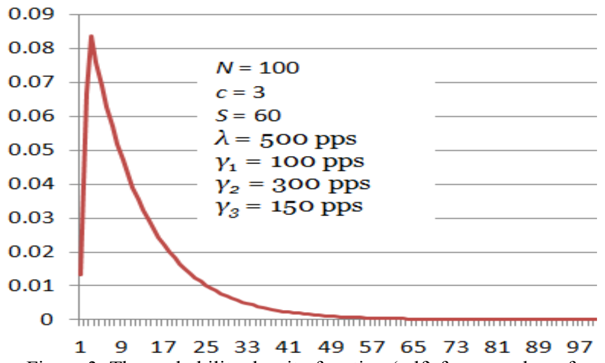

Figure 3: The probability density function (pdf) for n number of packets in the system

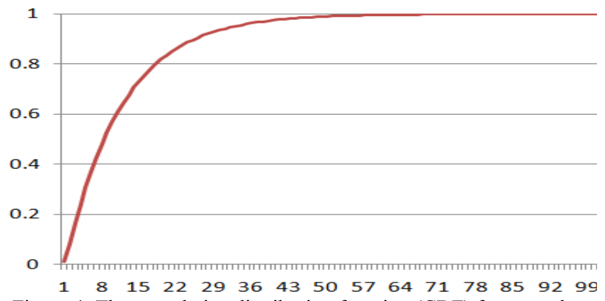
Figure 4: The cumulative distribution function (CDF) for n number of packets in the system

C	λ	γ_1	γ_2	γ_3	μ_n	GP (Analytical)	GP (Experimental)
3	638	320	192	128	640	572.74	567
2	506	320	192	0	512	456.62	453.91
1	312	320	0	0	320	286.3	283.75
3	223	128	64	32	224	196.73	197.76
2	190	128	64	0	192	174.02	169.73
1	125	128	0	0	128	115.94	113.34

Table 1: Experimental and analytical results

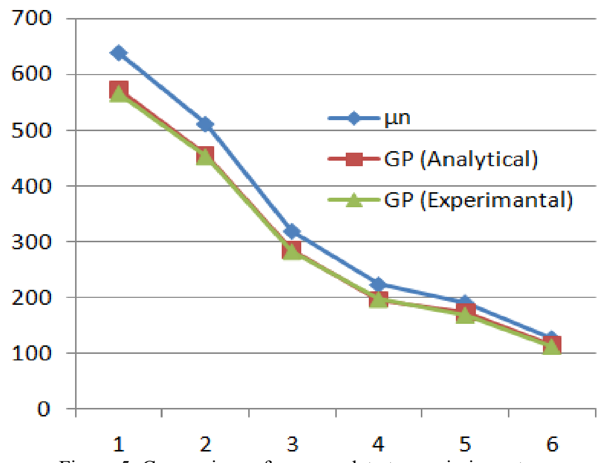

Figure 5: Comparison of average data transmission rates

Figure 5 shows three plots. The upper plot illustrates the selected value of cumulative transmission rate of all TCP connections. The two plots in the lower side shows the actual and predicted average data transmission rates. The predicted values were calculated using equation 11. The observed average data transmission rate is the average of transmission rate observed during 100 seconds time period of the experiment. The trend of the plots shows that as the number of connections participating for bandwidth aggregation decreases so does the average transmission rates. This shows the validity of using a BAS in order to improve the throughput. Moreover the predicted or analytical values also decrease as the number of connections decrease.

7. Conclusion

In this paper a generic architecture for a Bandwidth Aggregation System (BAS) along with its implementation is presented. Moreover it is argued that a BAS can be described as a network of queues. Based on this argument a queuing model is derived for the performance evaluation of a BAS. The model is used to derive an expression to predict the average data transmission rate for TCP connection in a BAS under steady-state conditions. In the end the predicted values are validated with the experimental results and found that model's predicted results and experimental results have an excellent agreement.

Although the average transmission rate is based on how fast the TCP drained the packets from its connection send buffer and TCP drain its connection send buffer only when it receives an acknowledgement from the other end but as a future work, it is planned to incorporate the TCP receive buffer size, receiving rate in a BAS and receiving rate of application in the model so that the effects of noisy links and slow receiving rate of applications could be predicted.

9. References

[1] Yousaf, M. Qayyum, A. "On End-to-End Mobility Management in 4G Heterogeneous Wireless Networks" In proceedings of IEEE Internationa Networking and Communications Conference, INCC 2008.

[2] I. F. Akyildiz, J. Xie & S. Mohanty, "A Survey of Mobility Management in Next-Generation All-IP-based Wireless Systems", IEEE Wireless Communications, August 2004.

[3] H. Hsieh, R. Sivakumar, "pTCP: An End-to-End Transport Layer Protocol for Striped Connections", In Proceedings of IEEE ICNP, 2002.

[4] Ming Zhang et al. "A transport layer approach for improving end-to-end performance and robustness using redundant paths", in proceedings of USENIX Annual Technical Conference, 2004.

[5] K. R. Evensen et al. "A Network-Layer Proxy for Bandwidth Aggregation and Reduction of IP Packet Reordering", 34th Annual IEEE Conference on Local Computer Networks, LCN 2009.

[6] A. Habib, N. Christin, and J. Chuang, "Taking advantage of multihoming with session layer striping" 9th IEEE Global Internet Symposium, 2006.

[7] A. Ford, et al. "Architectural Guidelines for Multipath TCP Development", IETF Internet draft; draft-ietf-mptcp-architecture-00, Feb-2010.

[8] J. Padhye,V. Firoiu, D. Towsley, and J.Kurose, "Modeling TCP Throughput: A Simple Model and its Empirical Validation," Proc. ACM SIGCOMM'98, Sept., 1998.

[9] N. Cardwell, S. Savage, T. Anderson, "Modeling TCP Latency," Infocom 2000, Tel Aviv, Israel, March 2000.

[10] M. Garetto, et al., "A Detailed and Accurate Closed Queuing Network Model of Many Interacting TCP Flows," IEEE INFOCOM 2001, Anchorage, Alaska, USA, April 2001

[11] M. Garetto, et al., "Modeling Short-Lived TCP Connections with Open Multiclass Queuing Networks," PFHSN 2002, Germany, April, 2002

[12] Ingolfsson, A., "Modeling the M(t)/M/s(t) Queue with an Exhaustive Discipline" working paper, 2005.
http://www.business.ualberta.ca/aingolfsson/publications.htm

[13] Raik Stolletz, "Approximation of the non-stationary M(t)/M(t)/c(t)-queue using stationary queuing models: The stationary backlog-carryover approach", European Journal of Operational Research
Volume 190, Issue 2, October 2008, Pages 478-493

[14] Hamdy A. Taha, "Operations Research: An Introduction (8th Edition)", Prentice-Hall, Inc., Upper Saddle River, NJ, 2007

[15] Wang, S. Y., Chou, C. L., Lin, C. C. "The Design and Implementation of the NCTUns Network Simulation Engine," Simulation Modelling Practice and Theory, 15,57-81 (2007).