Project	**IEEE 802.16 Broadband Wireless Access Working Group <http://ieee802.org/16>**
Title	**Multihop System Evaluation Methodology: Traffic Models**
Date Submitted	**2006-05-05**
Source:	Gamini Senarath, Wen Tong, Peiying Zhu, Hang Zhang, David Steer, Derek Yu, Mark Naden, Dean Kitchener Nortel 3500 Carling Avenue Ottawa, On, K2H 8E9 Canada Gamini@nortel.com wentong@nortel.com
Re:	Response to a call for contributions for the Relay TG, see C80216j-06/001.pdf
Abstract	A set of traffic models and associated parameters to be used to evaluate the performance of multihop systems are proposed in this document.
Purpose	To propose a set of traffic models to be used for the multihop performance evaluation.
Notice	
Release	
Patent Policy and Procedures	

Multi-hop System Evaluation Methodology: Traffic Models

Gamini Senarath, Wen Tong, Peiying Zhu, Hang Zhang,
David Steer, Derek Yu, Mark Naden, Dean Kitchener
Nortel

1 Introduction

This document provides a set of definitions, assumptions, and a general framework for traffic modeling for multihop relay systems (e.g. 802.16j, LTE relay extensions) to arrive at system wide voice, data, video or mixed data, voice, video performance on the forward and reverse links.

TCP and Higher Layer Modeling: For simulation, higher layer modeling such as TCP protocol and associated parameters and network modeling are required. It is proposes that those provide in [1] be used for this purpose. Those are included in Appendix A and Appendix B respectively for easy reference.

It is proposed that the multi-hop simulations be done using two types of traffic assumptions for applications:

- A quick view of system performance is to be obtained using full queue analysis (e.g. assuming availability of sufficient amount of traffic at each node point which does not need realistic traffic models).
- A detailed and more representative simulations are to be done with realistic traffic (non- full queue).

Full Queue and Non-Full Queue Traffic: The full queue analysis are indicative of the actual performance and useful to carry out quick performance evaluations in the initial design validations. However, they can, to a reasonable accuracy, compare the performance of two systems or systems with different configurations. The extension to realistic traffic model increases complexity of the simulations and it increases the simulation run time. However, they produces results that are more indicative of the actual performance in practice.

2 Traffic models

This section describes the traffic models in detail. Section 2.1 addresses forward link and Section 2.2 the revers e link.

A major objective of multi-hop simulations is to provide the operator a view of how many users can be supported for a given service under a specified multihop configuration at a given coverage level. The traffic generated by a service should be accurately modeled in order to find out the performance. Traffic modeling can be simplified, as explained below, by not modeling the user arrival process and assuming full queue traffic. These are explained below.

Modeling of user arrival process: All the users are not active and they might not register for the same service. In order to avoid different user registration and demand models, the objective of the proposed simulation is made limited to evaluate the performance with the users who are maintaining a session with transmission activity. These can be used to determine the number of such registered users that can be supported. This document does not address the arrival process of such registered users, i.e. it does not address the statistics of subscribers that register and become active.

Full Queue model: In the full queue user traffic model, all the users in the system always have data to send or receive. In other words, there is always a constant amount of data that needs to be transferred, in contrast to bursts of data that follow an arrival process. This model allows the assessment of the spectral efficiency of the system independent of actual user traffic distribution type.

At the relay station, however, the traffic availability depends on the forwarded traffic from either base station, u ser or by another relay even in the full queue model.

The traffic models provided in the next sections describe only the non-full queue case.

2.1 Traffic Modeling for Forward Link Services

The services to be modeled for the forward link are listed in Table 1. FTP, Web browsing and VOIP was includ ed in [1] and [2]. We propose live video services such as video conferencing to be included for simulation. Traff ic models for live video will be proposed as a future contribution.

Table 1: Services to be considered for the forward link

#	Application	Traffic Category	Mandator y/Optional
1	FTP	Best-effort	M
2	Web Browsing	Interactive	M
3	VoIP	Real-time	M
4	Video Streaming	Streaming	M
5	Live Video	Interactive Real-time	O

.

2.1.1 FTP [1]

It is proposed that traffic model in [1] be used for FTP. A description is extracted from [1].

In FTP applications, a session consists of a sequence of file transfers, separated by *reading times*. The two main parameters of an FTP session are:

S : the size of a file to be transferred

D_{pc}: reading time, i.e., the time interval between end of download of the previous file and the user request for the next file.

The underlying transport protocol for FTP is TCP. The packet trace of an FTP session is shown in Figure 1.

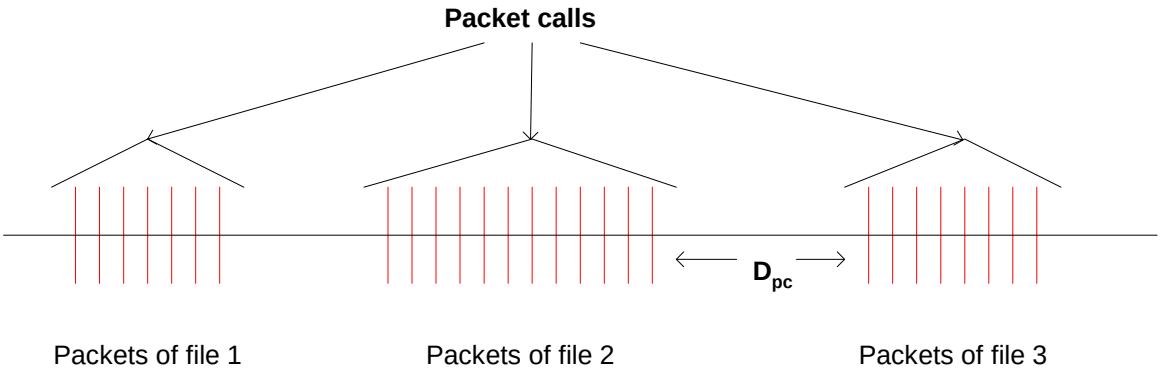

Figure 1 Packet Trace in a Typical FTP Session [1]

The parameters for the FTP application session are described in

Table 2.

Table 2 FTP Traffic Model Parameters

Component	Distribution	Parameters	PDF
File size (S)	Truncated Lognormal	Mean = 2Mbytes Std. Dev. = 0.722 Mbytes Maximum = 5 Mbytes	$f_X = \frac{1}{\sqrt{2\pi}\sigma x}\exp\left[\frac{-(\ln x-\mu)^2}{2\sigma^2}\right], x \geq 0$ $\sigma = 0.35, \mu = 14.45$
Reading time (D_{pc})	Exponential	Mean = 180 sec.	$f_X = \lambda e^{-\lambda x}, x \geq 0$ $\lambda = 0.006$

2.1.2 Web Browsing [1]

Web browsing is the dominant application for broadband data systems, and has been studied extensively. It is proposed that the traffic model in [1] be used. The descriptions provided in [1] are included below.

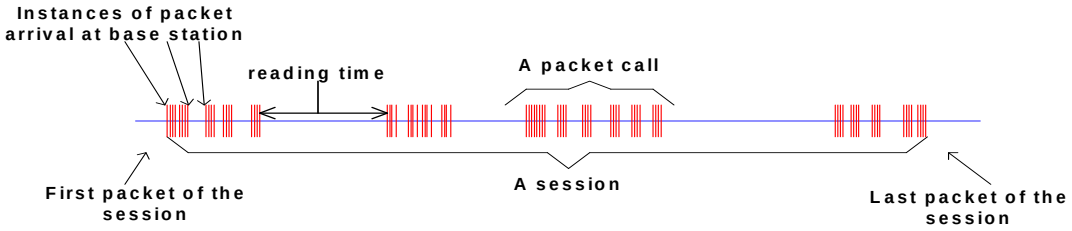

Figure 2 Packet Trace of a Typical Web Browsing Scheme [1]

Figure 2 shows the packet trace of a typical web browsing session. The session is divided into ON/OFF periods representing web-page downloads and the intermediate reading times. In **Figure 2**, the web-page downloads are referred to as packet calls. These ON and OFF periods are a result of human interaction where the packet call represents a user's request for information and the reading time identifies the time required to digest the web-page.

As is well known, web-browsing traffic is self-similar. In other words, the traffic exhibits similar statistics on different timescales. Therefore, a packet call, like a packet session, is divided into ON/OFF periods as in Figure-3. Unlike a packet session, the ON/OFF periods within a packet call are attributed to machine interaction rather than human interaction. In general, a web-page is constructed from many individually referenced objects. A web-browser will begin serving a user's request by fetching the initial HTML page using an HTTP GET request. After receiving the page, the web-browser will parse the HTML page for additional references to embedded image files such as the graphics on the tops and sides of the page as well as the stylized buttons. The retrieval of the initial page and each of the constituent *objects* is represented by ON period within the packet call while the parsing time and protocol overhead are represented by the OFF periods within a packet call. For simplicity, the term "page" will be used in this paper to refer to each packet call ON period. As a rule-of-thumb, a page represents an individual HTTP request explicitly initiated by the user. The initial HTML page is referred to as the "main object" and the each of the constituent objects referenced from the main object are referred to as an "embedded object".

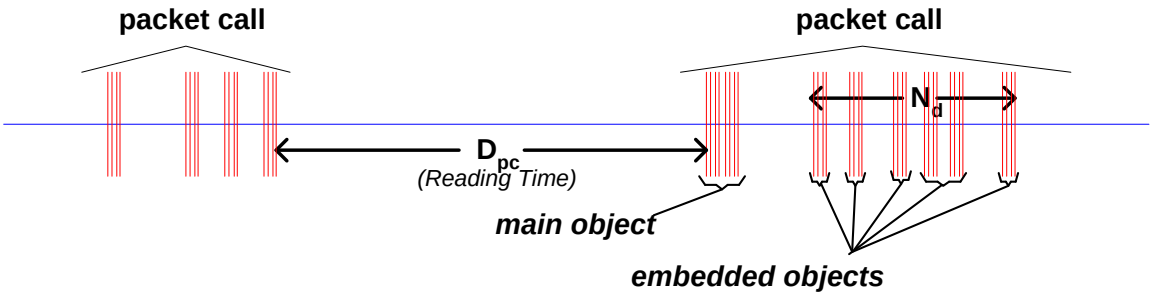

Figure-3 Contents in a Packet Call

The parameters for the web browsing traffic are as follows:

S_M: Size of the main object in a page

S_E: Size of an embedded object in a page

N_d: Number of embedded objects in a page

D_{pc}: Reading time

T_p: Parsing time for the main page

Table 3 HTTP Traffic Model Parameters [1]

Component	Distribution	Parameters	PDF
Main object size (S_M)	Truncated Lognormal	Mean = 10710 bytes Std. dev. = 25032 bytes Minimum = 100 bytes Maximum = 2 Mbytes	$f_x = \frac{1}{\sqrt{2\pi}\sigma x}\exp\left[\frac{-(\ln x-\mu)^2}{2\sigma^2}\right], x \geq 0$ $\sigma = 1.37, \mu = 8.35$
Embedded object size (S_E)	Truncated Lognormal	Mean = 7758 bytes Std. dev. = 126168 bytes Minimum = 50 bytes Maximum = 2 Mbytes	$f_x = \frac{1}{\sqrt{2\pi}\sigma x}\exp\left[\frac{-(\ln x-\mu)^2}{2\sigma^2}\right], x \geq 0$ $\sigma = 2.36, \mu = 6.17$
Number of embedded objects per page (N_d)	Truncated Pareto	Mean = 5.64 Max. = 53	$f_x = \frac{\alpha k^{\alpha}}{x^{\alpha+1}}, k \leq x < m$ $f_x = \left(\frac{k}{m}\right)^{\alpha}, x = m$ $\alpha = 1.1, k = 2, m = 55$ Note: Subtract k from the generated random value to obtain N_d
Reading time (D_{pc})	Exponential	Mean = 30 sec	$f_x = \lambda e^{-\lambda x}, x \geq 0$ $\lambda = 0.033$
Parsing time (T_p)	Exponential	Mean = 0.13 sec	$f_x = \lambda e^{-\lambda x}, x \geq 0$ $\lambda = 7.69$

Note: When generating a random sample from a truncated distribution, discard the random sample when it is outside the valid interval and regenerate another random sample.

2.1.3 Voice over IP (VoIP) [2]

A VoIP call shall be assumed to be between one user and one wired user. In order to get an evaluation of the air interface the wireline and core network impairments are neglected.

VoIP Traffic Source

The G.729A decoder shall be simulated with an assumed 4 byte IP header. Each packet produced by the G.729A vocoder shall be appended with a 4 byte header that accounts for UDP/IP overhead, after header compression.

2.1.4 Video Streaming [1]

It is proposed, that the following model for streaming video traffic on the forward link which was extracted from [1], be used. Figure 4 describes Video streaming

the steady state of video streaming traffic from the network as seen by the base station. Latency of starting up t he call is not considered in this steady state model.

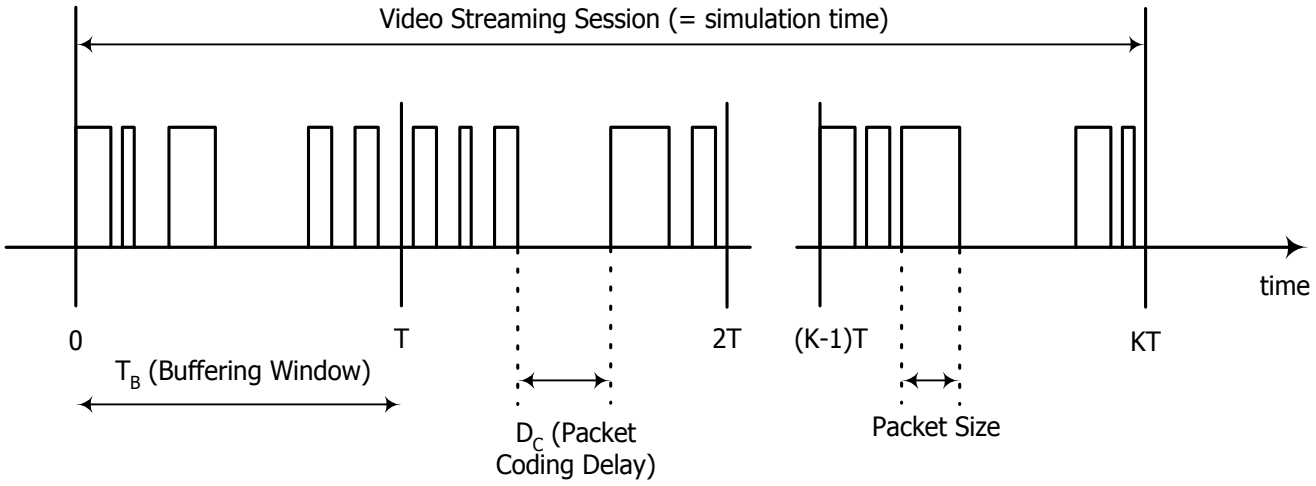

Figure 4 Near Real-Time Video Traffic Model [1]

A video streaming session is defined as the entire video and associated audio streaming call time, which is equa l to the simulation time for this model.

Each frame of video data arrives at a regular interval T determined by the number of frames per second (fps). E ach frame is decomposed into a fixed number of slices, each transmitted as a single packet. The size of these pa ckets/slices is distributed as a truncated Pareto. Encoding delay, Dc, at the video encoder introduces delay inter vals between the packets of a frame. These intervals are modeled by a truncated Pareto distribution. The param eter T_B is the length (in seconds) of the de-
jitter buffer window in the mobile station used to guarantee a continuous display of video streaming data. This parameter is not relevant for generating the traffic distribution but is useful for identifying periods when the real -
time constraint of this service is not met. At the beginning of the simulation, it is assumed that the mobile statio n de-
jitter buffer is full with (T_B x source video data rate) bits of data. Over the simulation time, data is "leaked" out of this buffer at the source video data rate and "filled" as forward link traffic reaches the mobile station. As a p erformance criterion, the simulation shall record the length of time, if any, during which the de-
jitter buffer runs dry.

The de-jitter buffer window for the video streaming service is a maximum of 5 seconds.

Using a source rate of 64 kbps, the video traffic model parameters are defined Table 4.

Table 4 Near Real-Time Video Traffic Model Parameters [1]

Information types	Inter-arrival time between the beginning of each frame	Number of packets (slices) in a frame	Packet (slice) size	Inter-arrival time between packets (slices) in a frame
Distribution	Deterministic (Based on 10fps)	Deterministic	Truncated Pareto (Mean= 50bytes, Max= 125bytes)	Truncated Pareto (Mean= 6ms, Max= 12.5ms)
Distribution Parameters	100ms	8	K = 20bytes α = 1.2	K = 2.5ms α = 1.2

2.1.5 Live Video Services

As mentioned before a traffic model will be provided for live video as a future contribution.

2.1.6 Modeling Reverse Link Traffic for the Forward Link only Simulations

HTTP requests and TCP ACKs come under this category. It is not known, what percentage of traffic would be acks and HTTP requests in a broadband systems. It is clear that the size of the access page increases with time, while ACK messages and HTTP requests remains the same. Therefore, we can expect that in the future systems, the impact of AC K and HTTP requests will be negligible compared to size of the data contents. However, since some traffic has to go through relay stations, the additional delays introduced by this path may have an impact which needs further investigation. It may be possible that these delays can be modeled by analyzing simulation results of the reverse link.

2.2 Traffic Modeling for Reverse Link Services

This section discusses the traffic modeling related to reverse link data traffic. Reverse link traffic to support forward link activities may be modeled using simulation results obtained for the reverse link as explained in Section 2.1.6.

Similarly the modeling of the forward link traffic to support the reverse link activity when carrying out reverse link only simulations, may be modeled using statistics obtained from the forward link simulations as described in Section 2.2.3.

2.2.1 FTP Upload / Email [1]

It is proposed that the FTP Upload and Email models used in [1] be used. They are provided below for ease of reference.

Since FTP uses TCP as its transport protocol, the TCP traffic model described in Appendix A [1], is used to represent the distribution of TCP packets for the FTP upload traffic on the RL.

The file upload and email attachment upload are modeled as in Table 5.

Table 5: **FTP Characteristics [1]**

Arrival of new users	Poisson with parameter λ
Upload file size	Truncated lognormal; lognormal pdf: $f_X = \frac{1}{\sqrt{2\pi}\sigma x}\exp\left[\frac{-(\ln x-\mu)^2}{2\sigma^2}\right], x \geq 0$ $\sigma = 2.0899, \quad \mu = 0.9385$ Min = 0.5 kbytes Max = 500 kbytes If the value generated according to the lognormal pdf is larger than Max or smaller than Min, then discard it and regenerate a new value. The resulting truncated lognormal distribution has a mean = 19.5 kbytes and standard deviation = 46.7 kbytes

The FTP traffic is simulated as follows:

At the beginning of the simulation there are 5 FTP users[1] waiting to transmit.

Before transmitting, call setup is performed for each user

Afterwards, FTP upload users arrive according to the Poisson arrival process, as defined in Table 5.

For each new FTP upload user coming into the system, call setup is performed

Each FTP upload user stays in the system until it finishes the transmission of its file

After an FTP upload user finishes the transmission of its file, it immediately leaves the system.

Since the arriving FTP users are dropped uniformly over 19 cells, it is possible the number of users can exceed the sector capacity. In that case, the new arrival should be blocked. The sector capacity is 43 in total. The blocking rate should be recorded.

2.2.2 HTTP Model [1]

The following figure is an example of events occurring during a HTTP session. The diagrams and the descriptions have been extracted from [1].

[1] In order to skip the transient period, the number of 5 initial FTP users is taken to represent the number of users at steady state.

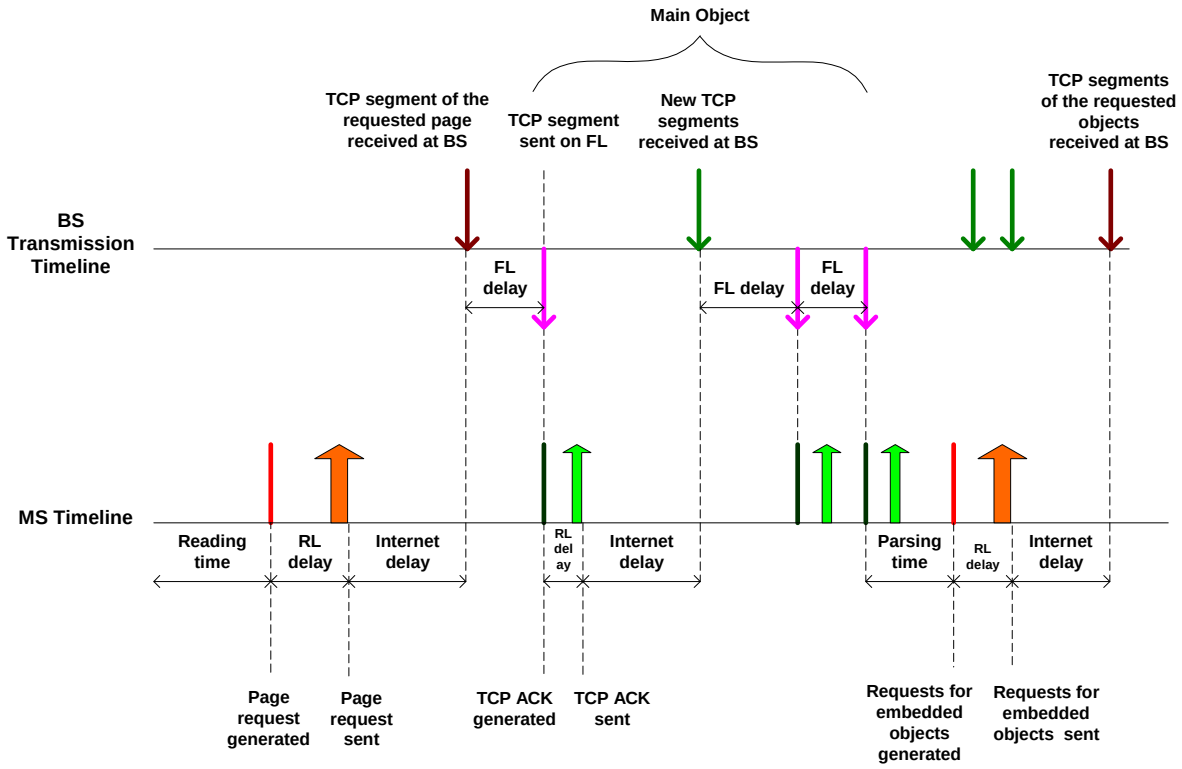

Figure 5: Example of events occurring during web browsing.

HTTP Traffic Model Parameters

Reading time (D_{pc}): modeled as in Table 6.

Internet delay (D_I): modeled as an exponentially distributed random variable with a mean of 50ms

Parsing time (T_p): modeled as in Table 6.

RL delay: specific for the implemented system. Includes RL packet transmission delay and scheduling delay (if scheduled)

FL delay (D_{FL}): defined as the time a TCP segment is first in the queue for transmission until it finishes transmission on forward link. The delay includes transmission delay and forward link scheduling delay. If there are multiple packets, each packet has its own additional contribution to the overall $D_{FL.}$

Number of TCP segments in the main object (N_M). $N_M = \lceil S_M /(MTU-40) \rceil$. The main object size, S_M, is generated according to Table 6.

Number of TCP segments in embedded object (N_E). $N_E = \lceil S_E /(MTU-40) \rceil$. The embedded object size, S_E, is generated according to Table 6.

Number of embedded objects (N_d). Modeled according to Table 6.

HTTP1.1 mode

The opening and the closing of the TCP connections is not modeled[2]

HTTP request size = 350 bytes

Requests for embedded objects are pipelined – all requests are buffered together

[2] This does not have much influence since in HTTP1.1 persistent TCP connections are used to download the objects (located at the same server) and the objects are transferred serially over a single TCP connection.

MTU size = 1500 bytes

ACK size = 12 bytes[3]

Every received TCP segment is acknowledged.

Table 6: HTTP Traffic Model Parameters

Component	Distribution	Parameters	PDF
Main object size (S_M)	Truncated Lognormal	Mean = 9055 bytes Std. dev. = 13265 bytes Minimum = 100 bytes Maximum = 100 Kbytes	If x > max or x < min, then discard and re-generate a new value for x. $f_X = \frac{1}{\sqrt{2\pi}\sigma x}\exp\left[\frac{-(\ln x-\mu)^2}{2\sigma^2}\right], x \geq 0$ $\sigma = 1.37, \quad \mu = 8.35$
Embedded object size (S_E)	Truncated Lognormal	Mean = 5958 bytes Std. dev. = 11376 bytes Minimum = 50 bytes Maximum = 100 Kbytes	$f_X = \frac{1}{\sqrt{2\pi}\sigma x}\exp\left[\frac{-(\ln x-\mu)^2}{2\sigma^2}\right], x \geq 0$ $\sigma = 1.69, \quad \mu = 7.53$ If x > max or x < min, then discard and re-generate a new value for x.
Number of embedded objects per page (N_d)	Truncated Pareto	Mean = 4.229 Max. = 53	$f_x = \frac{a_k^\alpha}{x^{\alpha+1}}, k \leq x < m$ $\alpha = 1.1, k = 2, m = 55$ Note: Subtract k from the generated random value to obtain N_d If x > max, then discard and re-generate a new value for x
Reading time (D_{pc})	Exponential	Mean = 30 sec	$f_x = \lambda_e^{-\lambda x}, x \geq 0$ $\lambda = 0.033$
Initial reading time (D_{ipc})	Uniform	Range [0, 10] s	$f_x = \frac{1}{b-a}, \quad a \leq x \leq b$ $a = 0, b = 10$
Parsing time (T_p)	Exponential	Mean = 0.13 sec	$f_x = \lambda_e^{-\lambda x}, x \geq 0$ $\lambda = 7.69$

[3] Compressed TCP/IP header (5 bytes from 40 bytes) and HDLC framing and PPP overhead (7 bytes).

Packet Arrival Model for HTTP

At the beginning of the simulation, call setup is performed for all HTTP users. After that, the simulation flow is described as follows:

Generate an initial reading time $D_{ipc.4}$ Wait D_{ipc} seconds.

Initiate the TCP window size W=1

Generate a request for the main page

Wait for the requests to go through the RL and reach the bases station (RL delay):

In case these are requests for embedded objects, wait until all requests reach the base station.

Generate an Internet delay D_I. Wait D_I seconds.

Generate random delays, which define the time instances when each of the TCP segment transmission is completed the FL. The number of these instances is:

For the main page:

At the very beginning of the packet call: 1.

Afterwards: min(2n, #of outstanding TCP segments on FL), where n is the number of ACKs received in the last physical layer packet (from the step 0)

For embedded objects:

At the very beginning of the transmission of embedded objects: min(W, $\sum_{i=1}^{N_d} N_E^i$).

Afterwards: min(2n, #of outstanding TCP segments on FL), where n is the number of ACKs received in the last physical layer packet (from the step 9 a.i.

Every time instance of the completed TCP segment transmission on FL generates an ACK on RL

Continue RL simulation – when ACK is generated, reduce the number of outstanding TCP packets by 1

Examine if the transmission of the very last TCP segment of the HTTP object is completed:

If no:

Proceed with simulation until next ACK or a group of n ACKs within a single physical layer packet is transmitted

Increase W:=W+n

Go to step 0

If yes, for main page:

Generate T_p (parsing time)

Generate requests for embedded objects

Continue RL simulation - transmit outstanding ACK(s) for the main page and accordingly increment W:=W+n for each group of n ACKs transmitted, until requests for embedded objects are generated

Go to step 0

If yes, for embedded objects:

Generate D_{pc} (reading time)

Continue RL simulation - transmit outstanding ACK(s) for the embedded objects

Go to step 0 when reading time expires or until all ACKs are transmitted, whichever is longer.

[4] The initial reading time is defined differently from subsequent reading times in order to ensure that all HTTP users finish the reading time within a limited period.

2.2.3 Modeling of Forward Link Traffic when carrying out a Reverse Link only simulation

HTTP requests and TCP ACKs come under this category. It is not known, what percentage of traffic would be acks and HTTP requests in a broadband systems. It is clear that the size of the access page increases with time, while ACK messages and HTTP requests remains the same. Therefore, we can expect that in the future systems, the impact of AC K and HTTP requests will be negligible compared to size of the data contents. However, since some traffic has to go through relay stations, the additional delays introduced by this path may have an impact which needs further investigation. It may be possible that these delays can be modeled by analyzing simulation results of the reverse link.

References

1. 3GPP2/TSG-C.R1002, “1xEV-DV Evaluation Methodology (V14)”, June 2003.

2. IEEE P 802.20™ PD-09 Version 1.0, September 23, 2005.

3. A Corlett, D.I. Pullin and S. Sargood, “Statistics of One-Way Internet Packet Delays,” *53rd IETF*, Minneapolis, March 2002.

Appendices

Appendix A TCP Model {1}

TCP is used as the higher layer transport protocol by various applications such as FTP and web browsing. Therefore, a TCP model is required to more accurately represent the distribution of TCP packets from these applications. It is proposed that the TCP models used in [1] be used. A description in [1] is provided below.

A.1 TCP Connection Set-up and Release Procedure

The TCP connection set-up and release protocols use a three-way handshake mechanism as described in Figure 6 and Figure 7. The connection set-up process is described below:

1. The transmitter sends a 40-byte SYNC control segment and wait for ACK from remote server.
2. The receiver, after receiving the SYNC packet, sends a 40-byte SYNC/ACK control segment.
3. The transmitter, after receiving the SYNC/ACK control segment starts TCP in slow-start mode (the ACK flag is set in the first TCP segment).

The procedure for releasing a TCP connection is as follows:

1. The transmitter sets the FIN flag in the last TCP segment sent.
2. The receiver, after receiving the last TCP segment with FIN flag set, sends a 40-byte FIN/ACK control segment.
3. The transmitter, after receiving the FIN/ACK segment, terminates the TCP session.

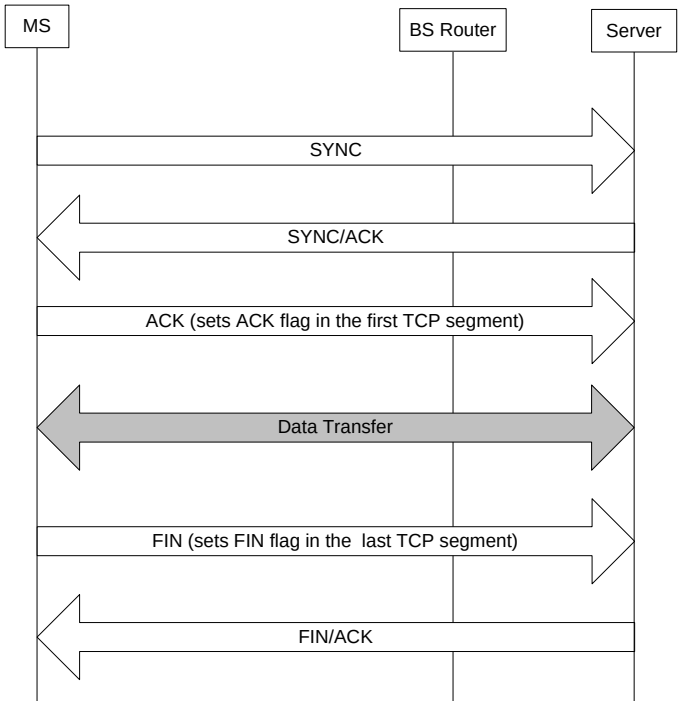

Figure 6: TCP connection establishment and release for Uplink data transfer

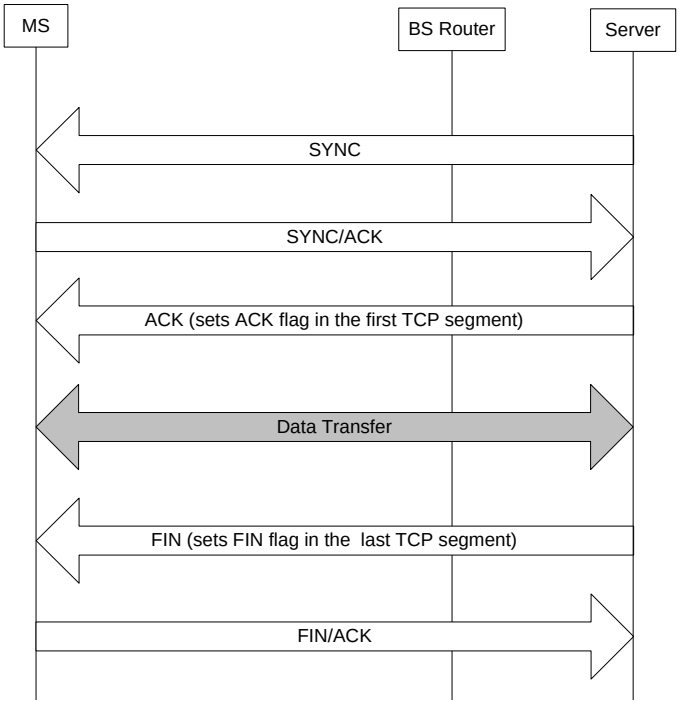

Figure 7: TCP connection establishment and release for Downlink data transfer

A.2 TCP slow start Model

The amount of outstanding data that can be sent without receiving an acknowledgement (ACK) is determined by the minimum of the congestion window size of the transmitter and the receiver window size. After the connection establishment is completed, the transfer of data starts in slow-start mode with an initial congestion window size of 1 segment. The congestion window increases by one segment for each ACK packet received by the sender regardless of whether the packet is correctly received or not, and regardless of whether the packet is out of order or not. This results in exponential growth of the congestion window i.e. after n RTTs (Round Trip Times), the congestion window size is 2^n. segments

A.3 UL (Uplink) slow start model

This UL slow start process is illustrated in Figure 8. The round-trip time in Figure 8, $\tau_{rt,}$ consists of two components, see Table 7:

$$\tau_{rt} = \tau_u + \tau_l$$

where τ_u = the sum of the time taken by a TCP data segment to travel from the base station router to the server plus the time taken by an ACK packet to travel from the server to the client; τ_l = the transmission time of a TCP data segment over the access link from the client to the base station router. τ_u is further divided into two components; τ_2 = the time taken by a TCP data segment to travel from the base station router to the server plus the time taken by an ACK packet to travel from the server back to the base station router and τ_3 = the time taken by the ACK packet to travel from the base station router to the client.

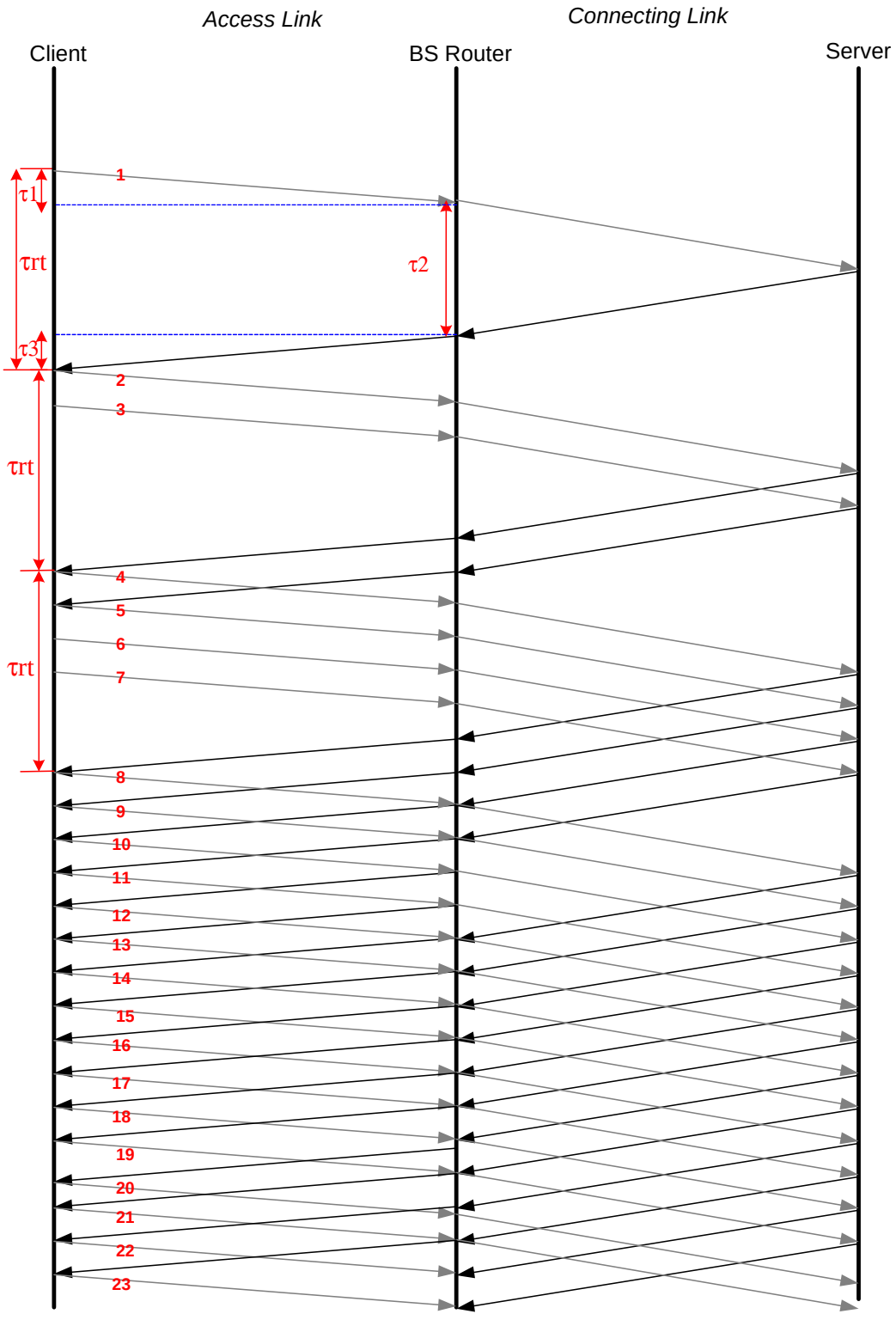

Figure 8: TCP Flow Control During Slow-Start; τ_l = Transmission Time over the Access Link (UL); τ_{rt} = Roundtrip Time

Table 7 **Delay components in the TCP model for the UL upload traffic**

Delay component	Symbol	Value
The transmission time of a TCP data segment over the access link from the client to the base station router.	τ_1	nined by the access link throughput
The sum of the time taken by a TCP data segment to travel from the base station router to the server and the time taken by an ACK packet to travel from the server to the base station router.	τ_2	See 0
The time taken by a TCP ACK packet to travel from the base station router to the client.	τ_3	See 0

A.4 DL (Downlink) slow start model

This DL slow start process is illustrated in Figure 9. The round-trip time in Figure 9, $\tau_{rt,}$ consists of two components, see Table 8:

$$\tau_{rt} = \tau_d + \tau_4$$

where τ_d = the sum of the time taken by an ACK packet to travel from the client to the server and the time taken by a TCP data segment to travel from the server to the base station router; τ_4 = the transmission time of a TCP data segment over the access link from the base station router to the client. τ_d is further divided into two components; τ_5 = the time taken by a TCP ACK to travel from the base station router to the server plus the time taken by a TCP packet to travel from the server back to the base station router and τ_3 = the time taken by the TCP packet to travel from the base station router to the client.

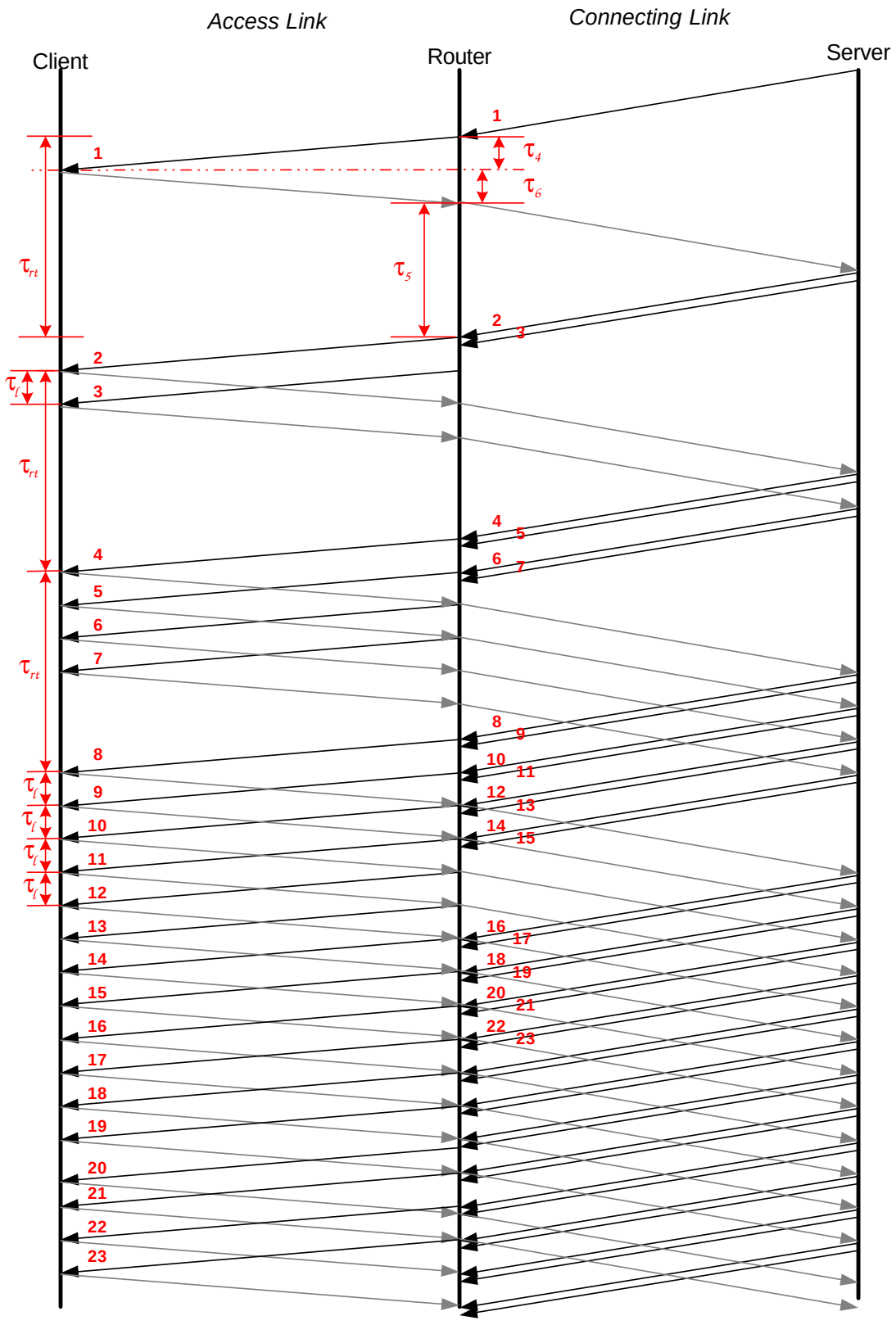

Figure 9 TCP Flow Control During Slow-Start; τ_l = Transmission Time over the DL; τ_{rt} = Roundtrip Time

Table 8 **Delay components in the TCP model for the DL traffic**

Delay component	Symbol	Value
The transmission time of a TCP data segment over the access link from the base station router to the client.	τ_4	nined by the access link throughput
The sum of the time taken by a TCP ACK to travel from the base station router to the server and the time taken by TCP data packet to travel from the server to the base station router.	τ_5	See 0
The time taken by a TCP ACK to travel from the client to the base station router.	τ_6	See 0

From Figure 8 and Figure 9, it can be observed that, during the slow-start process, for every ACK packet received by the sender two data segments are generated and sent back to back. Thus, at the mobile station (base station), after a packet is successfully transmitted, two segments arrive back-to-back after an interval $\tau_u = \tau_2 + \tau_3$ ($\tau_d = \tau_5 + \tau_6$). Based on this observation, the packet arrival process at the mobile station for the upload of a file is shown in Figure 10. It is described as follows:

1. Let S = size of the file in bytes. Compute the number of packets in the file, $N = \lceil S/(MTU-40) \rceil$. Let W = size of the initial congestion window of TCP. The MTU size is fixed at 1500 bytes

2. If N>W, then W packets are put into the queue for transmission; otherwise, all packets of the file are put into the queue for transmission in FIFO order. Let P=the number of packets remaining to be transmitted beside the W packets in the window. If P=0, go to step 6

3. Wait until a packet of the file in the queue is transmitted over the access link

4. Schedule arrival of next two packets (or the last packet if P=1) of the file after the packet is successfully ACKed. If P=1, then P=0, else P=P-2

5. If P>0 go to step 3

6. End.

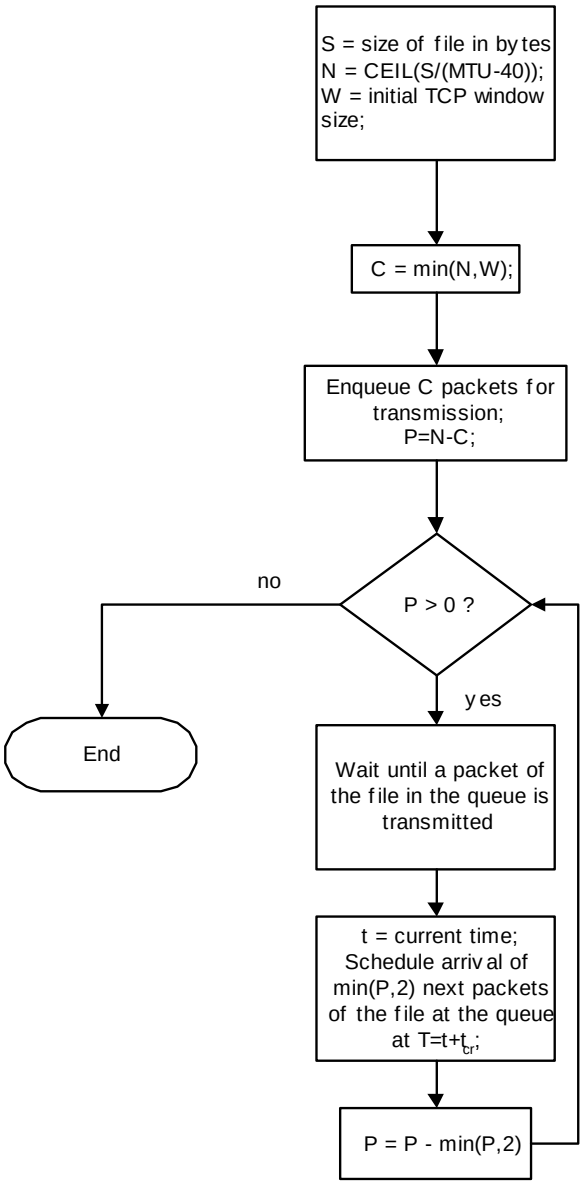

Figure 10 Packet Arrival Process at the mobile station (base station) for the upload (download) of a File Using TCP

Appendix B Backhaul Network Modeling [2]

B.1 Network Delay model [2]

The one-way Internet packet delay is modeled using a shifted Gamma distribution [**3**] with the parameters shown in Table 9. The packet delay is independent from packet to packet.

Table 9 Parameters for the shifted Gamma Distribution

Scale parameter (α)	1
Shape parameter (β)	2.5
Probability density function (PDF)	$f(x) = \frac{(x/\alpha)^{\beta-1} e^{-x/a}}{\alpha \cdot \Gamma(\beta)}$ $\Gamma(.)$ is the gamma function
Mean	$\alpha\beta$
Variance	$\alpha^2\beta$
Shift	See Table 10

Two values, 7.5ms and 107.5ms are used for the shift parameter in order to model the domestic routes and the International routes respectively. The users' routes are selected randomly at the time of drop with the distribution shown in Table 10.

Table 10 Shift parameter for the Domestic and International IP routes

IP Route Type	**Percentage of users**	**Shift parameter**	**Mean one-way IP packet delay**
Domestic	80%	7.5ms	10ms
International	20%	107.5ms	110ms

B.2. Network Loss model

The transmission of IP packets between the base station (server) and the server (base station) is assumed error free.

Table 11 Internet Loss Model

IP packet error rate	0% (lossless packet transmission)