



**DUBLIN CITY UNIVERSITY
SCHOOL OF ELECTRONIC ENGINEERING**

**Extending the Range of Wireless LANs
using AODV**

Yvonne Rooney
April 2004

**BACHELOR OF ENGINEERING
IN
TELECOMMUNICATIONS ENGINEERING**

Supervised by Dr. John Murphy

Acknowledgements

I would like to thank my supervisors Mr. Seán Murphy and Dr. John Murphy for their guidance and support.

Declaration

I hereby declare that, except where otherwise indicated, this document is entirely my own work and has not been submitted in whole or in part to any other university.

Signed:

Date:

Abstract

It may be desirable to extend the range of Wireless Local Area Networks beyond 100 meters. One approach to do this would be to use ideas which arise in ad hoc networks.

A mobile ad-hoc network (MANET) is a collection of wireless mobile hosts forming a temporary network without the use of any pre-existing structure. Because of the improvised nature of such networks, a routing protocol is used to discover routes between nodes.

This report presents the performance attainable in the ad hoc network when various traffic models are implanted.

Table Of Contents

ACKNOWLEDGEMENTS	2
DECLARATION.....	2
ABSTRACT.....	3
TABLE OF CONTENTS	4
TABLE OF FIGURES.....	6
CHAPTER 1 – INTRODUCTION.....	7
1.1 AD HOC WIRELESS NETWORKS	7
1.2 STRUCTURE OF THE REPORT.....	7
1.3 PROBLEM STATEMENT	7
CHAPTER 2 – TECHNICAL BACKGROUND	7
2.1 ROUTING PROTOCOL	7
2.3 AD HOC ON-DEMAND DISTANCE VECTOR (AODV)	7
2.3.1 <i>How does AODV work?</i>	7
2.3.2 <i>Route Discovery</i>	7
2.3.3 <i>Error Messages</i>	7
2.3.4 <i>HELLO Messages</i>	7
2.4 TRAFFIC SOURCES.....	7
2.4.1 <i>Constant Bit Rate (CBR)</i>	7
2.4.2 <i>Voice over IP (VoIP)</i>	7
2.4.3 <i>Video Source (VBR)</i>	7
2.4.4 <i>FTP</i>	7
2.4.5 <i>TELNET</i>	7
2.4.6 <i>HTTP</i>	7
CHAPTER 3 – GLOMOSIM.....	7
3.1 <i>What is GloMosim?</i>	7
3.2 <i>GloMosim Architecture</i>	7

3.3	<i>GloMosim Simulations</i>	7
3.3.1	<i>Input file</i>	7
3.3.2	<i>Running Config.in</i>	7
3.3.3	<i>Visualization Tool</i>	7
CHAPTER 4 – ADDING NEW DATA SOURCES		7
4.1	STRUCTURE OF GLOMOSIM CODE	7
4.2	VOIP	7
4.2.1	<i>Scheduled Events</i>	7
4.3	MPEG VIDEO SOURCE (VBR).....	7
CHAPTER 5 – RESULTS AND ANALYSIS		7
5.1	PERFORMANCE EVALUATION	7
5.2	SIMULATIONS.....	7
5.3	RESULTS	7
5.3.1	<i>CBR Efficiency</i>	7
5.3.2	<i>CBR Latency</i>	7
5.3.4	<i>VoIP Efficiency</i>	7
5.3.5	<i>VoIP Latency</i>	7
5.3.6	<i>VBR Efficiency</i>	7
5.3.7	<i>VBR Latency</i>	7
CHAPTER 6 – CONCLUSION AND FURTHER RESEARCH		7
	<i>Further Research</i>	7
REFERENCES		7
APPENDIX I		7
	<i>Glomo.stat</i>	7
	<i>VBR Source Model</i>	7

Table of Figures

Figure 1-1 Transmission Range	7
Figure 2-1 AODV	7
Figure 3-1 GloMosim Architecture	7

Chapter 1 – Introduction

Within the last few years the use of Wireless LANs for providing Internet connectivity has increased. They can be used as an extension of or as an alternative to wired LANs within a building or area. Wireless LANs are infrastructure based wireless networks where the node communicates directly with an access point which is connected to the wired network. The access point acts as a gateway which transmits data between the WLAN and the wired network. A single access point can support a small group of users and can function up to a range of 100 meters.

However it may be desirable to extend this range. Ad hoc networking has been put forward as a possible solution to this limitation.

1.1 Ad hoc Wireless Networks

A way of overcoming the problem of dead zones i.e. areas without network coverage, within a wireless network is the use of ad hoc networks. Alternatively to infrastructure based wireless networks, ad hoc networks are infrastructureless wireless networks. They do not rely on any fixed infrastructure; instead they require the use of neighbour nodes to forward packets from one node to another until it reaches its destination. A packet could travel through a number of nodes before arriving at its destination. Therefore nodes are not limited to the transmission range of the access point.

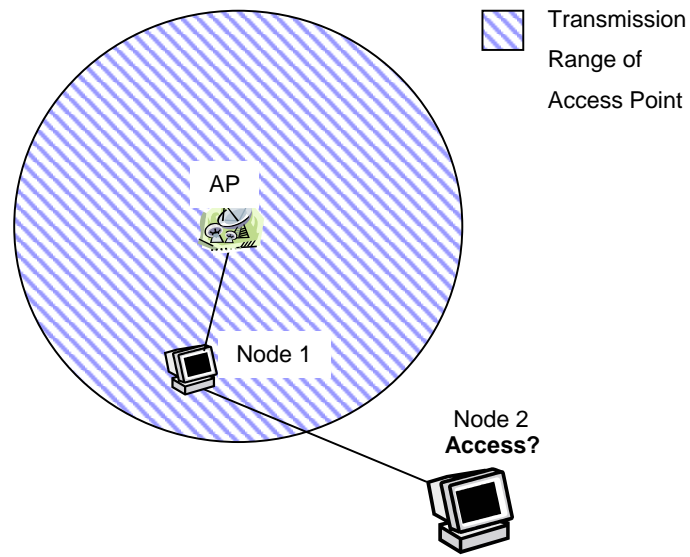


Figure 1-1 Transmission Range

As *Fig. 1-1* illustrates, Node 2 is outside the transmission range of the access point. However by using ad hoc routing protocols Node 2 can obtain Internet connectivity by using Node 1, which is within the range of the access point, as a relay node to forward packets to the access point. This route is maintained for as long as Node 2 requires it.

Although, this new approach of networking offers great flexibility to the world of wireless communications and raises some new challenges among the research community as far as routing, QoS, and security issues are concerned.

In this report we will be concentrating on the routing area and the effect of traffic loads on the ad hoc network. The objective of this study is the implementation, documentation and evaluation of simulation models for ad hoc networks. The commercially available simulation software GloMosim [3] was used for this purpose.

1.2 Structure of the Report

Chapter 2 provides the technical background, overview of the routing protocol used. Chapter 3 describes the simulation package used to perform the simulations. Description of the implementation of the proposed system design, along with a description of the new data sources and their integration is provided in Chapter 4. Chapter 5 features the discussion of the simulation results, while Chapter 6 offers an overall evaluation of the project. Chapter 6 also provides some conclusions and suggestions as to the directions that the future work may take in order to improve the system.

1.3 Problem Statement

The objective of this project is to investigate the limits of extending the range of Wireless LANs using ad hoc networking concepts.

The range of a WLAN access point is limited to 100m. It may be desirable to extend the range of the WLAN to greater than 100m. By using the ad hoc protocols the idea would be to permit users closer to the access point to act as relays for users outside of range.

The scope of the project involves simulating a number of scenarios which test the boundaries of the ad hoc networking protocol. Certain metrics such as Throughput, average end to end delay and packet delivery ratio will be measured.

Chapter 2 – Technical Background

2.1 Routing Protocol

There are certain characteristics of an ad hoc network that put a lot of stress on the routing layer. The characteristics are as follows:

- Nodes in a Mobile Ad-hoc Network (MANET) are typically constrained to the bandwidth of the wireless links. The routing protocol should therefore be efficient in its use of bandwidth. The message complexity should be kept to a minimum.
- Nodes within a MANET tend to be restricted on resources such as battery power and storage capacity. Therefore the routing protocol should not excessively flood the network or send multiple update messages.

The routing protocol which seemed the most appropriate for the simulations was the Ad hoc On-Demand Distance Vector (AODV). It is arguably most mature in standards development process. As well as this an implementation was readily available in the GloMosim [3] simulator. The next section will explain how AODV works.

2.3 Ad hoc On-Demand Distance Vector (AODV)

The Ad hoc On-Demand Distance Vector (AODV) algorithm allows multi hop routing between nodes wishing to establish and maintain an ad hoc network. AODV allows mobile nodes to obtain routes quickly and does not require nodes to maintain routes with nodes that are no longer active within the system. Instead of building a route for every destination within the network, a node only creates and maintains a route as it is needed. However when a route is no longer used, the route is deleted from the route table. AODV only allows nodes affected by a link breakage to be notified, their routing tables are then updated. This approach is known as source-initiated on-demand routing or reactive routing.

A feature, which originated from AODV's predecessor, Destination Sequence Distance Vector (DSDV), is the destination sequence number. The sequence number is incremented and given to each routing table entry. When a node is given a choice between two routes to the destination it will select the route with the greatest sequence number. A route with the greatest sequence number was issued later in time and therefore has more chances of being an accurate look at the current topology. This ensures loop freedom.

2.3.1 How does AODV work?

When a node wishes to send a packet to destination node, it initiates a discovery process to locate it. If no route is found within a certain time period, the destination is classified as unreachable and the packet is dropped. However if the route to the destination is found, the source node updates its route table to the destination node and vice versa. The next hop is also noted in the routing table

Once a route is established, a maintenance process begins to monitor the status of the route. If the route becomes inactive, the route is deleted from the routing table. If a failure occurs on the route, the neighbour nodes to the destination and the source are informed using a certain control packet.

AODV defines the routing information in a table-driven manner. Each node holds a routing table for the path to the destination node. Therefore the source node does not need to know the complete path to the destination.

2.3.2 Route Discovery

As mentioned above, when a node wants to send a packet to a destination node, it initiates a discovery process. AODV does this by broadcasting a *Route Request* packet (RREQ) to its neighbouring nodes. The route request is forwarded by every other node until a route is found.

In Fig. 2 we can see Node 1, the *source node*, wishes to send a packet to the *destination node*, Node 3. In order to do this Node 1 must first obtain the routing information to the destination. A *RREQ* packet is broadcasted to Node 1's neighbours, Node 2 & Node 4. The *RREQ* packet contains the *destination address*, *source address*, *lifespan* and *RREQ Sequence number*.

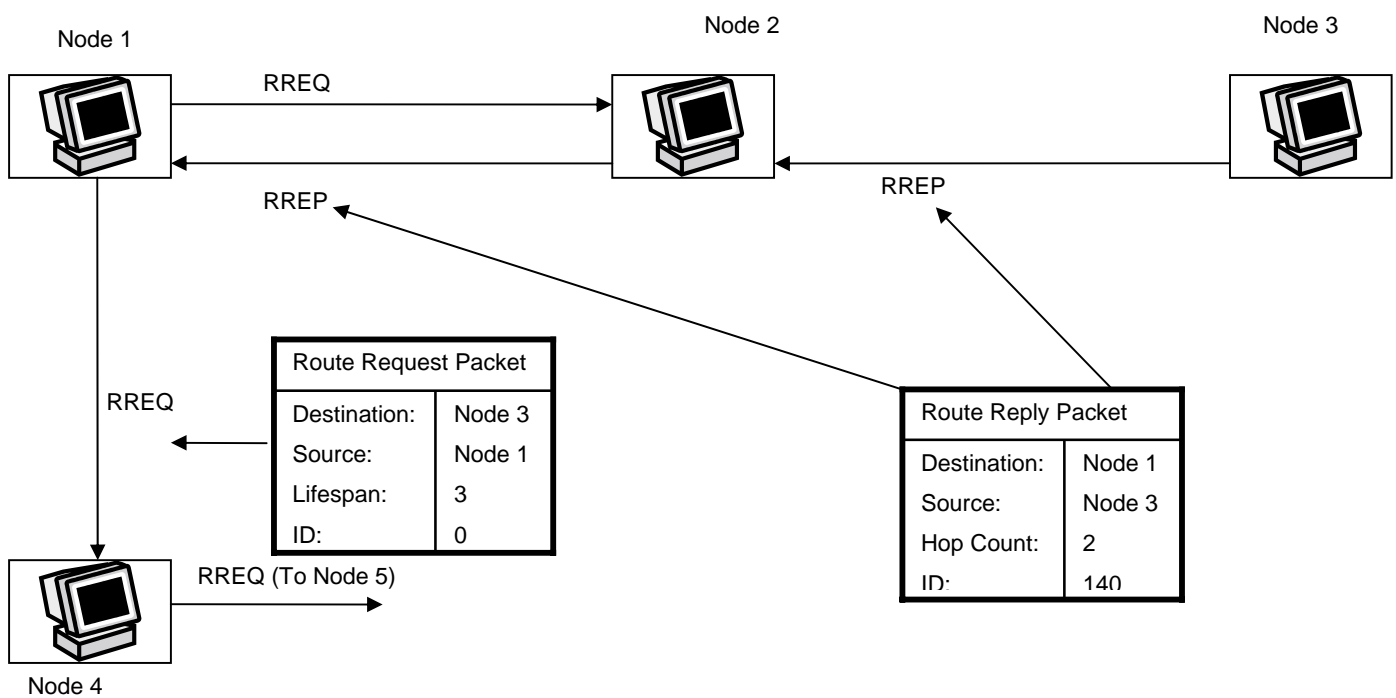


Figure 2-1 AODV

Upon receiving the packet Node 2 & Node 4 have two choices; if they know a route to the destination or if they are the destination they can send a *Route Reply (RREP)* message back to Node 1. Otherwise they will rebroadcast the *RREQ* to their neighbours. This process will continue until the expiration of the *RREQ* lifespan. If Node 1 does not receive a *RREP* within a certain time it will resend a *RREQ* only this time with a greater

lifespan and a new sequence number in order to expand the node search. The sequence number is used to ensure *RREQs* are not rebroadcast.

In *Fig. 2* Node 2 has a route to Node 3 and replies to the *RREQ* by sending a *RREP*. On the other hand, Node 4 does not have a route to Node 3; it rebroadcasts the *RREQ* to Node 5.

2.3.3 Error Messages

AODV makes use of a *Route Error (RERR)* Message to adjust routes when the topology of the network changes.

Whenever a node receives a *RERR* message it deletes all routes pertaining to the particular bad links. There are three scenarios where a node would broadcast a *RERR* message to its neighbours.

1. A node receives a data packet which needs to be forwarded, however this particular node does not have a route to the destination. The problem is that some other node believes that this node is on the route to the destination.
2. A node receives a *RERR* which causes one of its routes to become invalid. The node updates its routing table and forwards the *RERR* message to its neighbours.
3. A node detects that one of its neighbours is no longer available. The node looks at the routing table for routes that contain its invalid neighbour. It then sends a *RERR* message to the relevant nodes

2.3.4 HELLO Messages

As part of the AODV protocol, each node continually polls its neighbours, to keep track of them. It does this by sending a message at regular intervals, the message is known as an HELLO message.

2.4 Traffic Sources

In order to test the boundaries of an ad hoc network, realistic simulations needed to be performed. Therefore moderately realistic model of traffic sources were used, this section looks at the different traffic sources.

2.4.1 Constant Bit Rate (CBR)

This is simply a constant rate of data being transmitted by the node. In real life this could represent a mail merge, each email of equal size, transmitted at different time intervals.

2.4.2 Voice over IP (VoIP)

As the popularity of VoIP is increasing so too is the VoIP traffic. Therefore a VoIP traffic source needs to be created to fully test the ad hoc network. The Glomosim simulator will be used for these simulations, which unfortunately does not have a VoIP application. An exponential traffic source will need to be created.

An audio flow is divided into “talk spurt” (periods of audio activity) and “silence periods” (periods of audio inactivity, during which no packets are generated). The alternating periods of activity and silence are exponentially distributed.

2.4.3 Video Source (VBR)

The video source used was taken from [2] which presents a model for a variable-bit-rate MPEG video stream. The model takes the varying bit-rate over certain times, which can be seen in the varying scene changes. Three types of coded frames make up an MPEG sequence: *I* (intra frame), *P* (predicted frames) and *B* (backward predicted frames). I-frames are coded as still images i.e. JPEG, this contains all the information for that particular frame. The P-frames are coded as the difference between the particular frame and the previous I or P frame. B-frames are coded as the difference between the previous frame and the next frame of the sequence.

MPEG is an international standard which has been used in the broadcasting industry for years - MPEG is used in DVDs, CDs and other digital storage media.

Therefore MPEG is a common traffic source at the moment as MPEG videos are much smaller yet the quality is the same as other compression formats. Therefore this is an important traffic generator to incorporate into the simulations.

2.4.4 FTP

FTP - Protocol in the "TCP/IP" family for copying files from one computer to another. It stands for "File Transfer Protocol". Since FTP has become a very common method of moving files between two Internet sites, the FTP traffic generator will act as background traffic.

2.4.5 TELNET

TELNET - Protocol that provides terminal emulation using the TCP/IP protocols. Telnet allows users to log onto and access remote computers. Like FTP, TELNET is a common form of traffic. Therefore TELNET will be used as background traffic also.

2.4.6 HTTP

The protocol for moving hypertext files across the Internet. It requires a HTTP client program on one end, and an HTTP server program on the other end. HTTP is the most important protocol used in the World Wide Web.

In the software package GloMosim the HTTP model is adapted from the work published by Bruce Mah "An Empirical Model of HTTP Network Traffic", Proceedings of INFOCOM'97, Kobe, Japan, April 1997 - <http://www.employees.org/~bmah/Papers/Http-Infocom.pdf>. The World Wide Web and as a result http traffic has grown significantly since 1997 and as a result the Infocom http traffic model are no longer realistic and as such are considered out-dated. In order to provide a realistic simulation using http the application will need to be modified. This is outside the scope of the project.

Chapter 3 – GloMosim

3.1 What is GloMosim?

GloMosim is a library-based sequential and parallel simulator for wireless networks [3]. This has been developed using PARSEC, a C-based parallel simulation language. GloMosim can be modified to add new protocols and applications to the library. Therefore GloMosim is a good choice for implementing the different traffic sources mentioned in section 2.4.

GloMoSim is aimed at simulating models that may contain as many as 100,000 mobile nodes with a reasonable execution time; this is done by using node aggregation. As each entity needs to examine packets received only from the nodes located in the region it is simulating, many partitions are used to reduce the total search space for packet delivery. If a packet sent by a node located in Partition (0, 0) cannot reach the border of the partition, no message needs to be sent to the other partitions. Therefore, the other partitions do not have to examine the reception of the packet.

3.2 GloMosim Architecture

GloMosim is built using a layered approach, similar to the OSI seven layer network architecture. The GloMosim architecture can be seen below *Fig. 3*.

<i>Layer</i>	<i>Protocols</i>
Application	CBR, FTP, HTTP and Telnet
Transport	TCP and UDP
Network (Routing)	IP with AODV, Bellman-Ford, DSR, Fisheye, LAR scheme 1, ODMRP, WRP
Data Link (MAC)	CSMA, IEEE 802.11 and MACA
Packet Reception Models	SNR bounded, BER based with BPSK/QPSK modulation
Radio Model	Noise Accumulating
Radio Propagation	Two ray and Free space
Mobility	Random waypoint, Random drunken, Trace based

Figure 3-1 GloMosim Architecture

The application layer will be the focus of this project by adding two more traffic sources. In the network layer the AODV routing protocol will be used in the simulations.

For this project GloMosim was installed on Redhat Linux 9.0. Along with GloMosim PARSEC was also installed, for compiling the GloMosim files.

The basic structure of GloMosim is as follows:

/doc – contains the documentation

/scenarios – contains directories of various sample configuration topologies

/main – contains the basic framework for GloMosim

/bin – contains the executables and the input/output files

/include – contains common include files

/application – contains code for the application layer i.e. files for traffic generation

/transport – contains the code for the transport layer

/network – contains the code for the network layer

/mac – contains the code for the Mac layer, including 802.11b

/radio – contains the code for the physical layer

All of these files need to be compiled from the /main directory. To do this in Linux, run “make” to create the executable “glomosim.exe”. For every application added, run “make” to compile the new “.pc” & “.h” files.

3.3 GloMosim Simulations

To perform simulations in GloMosim, the input file needs to be configured. The following sections describe how this is done.

3.3.1 Input file

Within the /bin directory the basic input file “config.in” can be found. This file contains the general simulation parameters for GloMosim. It is structured according to each layer and allows a protocol to be chosen for each layer. In the file, anything following “#,” is treated as a comment. Therefore in order to select a protocol, the “#” is deleted. An example of the “config.in” file can be found in Appendix I. Below is a snapshot of the input file

<pre>[config.in] NUMBER-OF-NODES 3 NODE-PLACEMENT FILE NODE-PLACEMENT-FILE ./nodes.input #NODE-PLACEMENT UNIFORM MAC-PROTOCOL 802.11 NETWORK-PROTOCOL IP ROUTING-PROTOCOL BELLMANFORD APP-CONFIG-FILE ./app.conf RADIO-TYPE RADIO-ACCNOISE</pre>	<pre>[nodes.input] 0 0 (20.2, 0.9, 0.11) 1 0 (80.4, 90.8, 0.17) 2 0 (60.7, 30.4, 0.10)</pre>
	<pre>[app.conf] #CBR <src_node> <dest_node> #<items><item_size> <interval_time> # <start_time> <end_time> CBR 0 1 10 512 1S 0S 0S</pre>

3.3.2 Running Config.in

To run the input file, from the /bin directory, type “GloMosim config.in”. The simulation will run and produce an output file “GLOMO.stat”. This file contains the resulting for each layer including the throughput, average end-to-end delay, hop distance, data packets transmitted and received etc. An example of “GLOMO.stat” can be found in Appendix I.

From the above file the simulation will have 3 nodes, positioned according to the file “nodes.input” above. The structure of “nodes.input” is <node address>, <0>, <(x, y, z)>. The MAC protocol chosen is 802.11, the Network protocol is IP and the application

chosen for this particular simulation is Constant Bit-Rate (CBR) from the “app.conf” file above. The structure of “app.conf” is *<source node address>*, *<destination node address>*, *<no. of items to send>*, *<item size>*, *<interval time>*, *<start time>*, *<end time>*. A simulation time is also added at the beginning of the file along with terrain dimensions.

3.3.3 Visualization Tool

The visualization tool is not dependent on the platform as it is coded in Java. Therefore in order to run the tool, Java the Java Development Kit (JDK) version 1.2 must be installed.

Once JDK is installed the Java VT files must be compiled. From the /java_gui directory, type “javac *.java”. To start the VT type “java GlomoMain&”.

The visualization tool was not used in the simulations as it was quite unstable and unreliable.

Chapter 4 – Adding new Data Sources

As mentioned before new traffic sources will need to be designed for the GloMosim package. Two traffic sources will be added, the VoIP (Exp) traffic generator and the Video source (VBR).

4.1 Structure of GloMosim Code

Within each directory of the GloMosim library there are three important files which must be altered or created to change or add new models. They are as follows:

- Header files (.h) – Basic definitions of functions and data objects
- .pc files – Actual functions
- Layer.pc – Layer Interface

Different steps have to be considered when adding a model or protocol to a layer in order to maintain consistency and proper functionality in GloMoSim.

Three main functions have to be elaborated to add a new model:

1. Initialization Function: It must allocate and initialize the model specific data.
2. Finalization Function: It generates the output statistics from the simulation run for this model.
3. Simulation Event Handling Function: It performs simulation actions when scheduled with an event.

The scheduling of events takes place as follows:

- Allocate the GloMoSim Message:
`Message* msg = GLOMO_MsgAlloc(node, MyID, MyLayer,
MyProtocol, MSG_LAYER_PROTO_MY_EVENT);`
- Set the Event Specific Information:
`MyEventInfoType* MyEventInfo;
GLOMO_MsgInfoAlloc(node, msg, sizeof(MyEventInfoType));`

```
MyEventInfo->MyFirstParameter =1;
```

- Schedule the Event:

```
GLOMO_MsgSend(node, message, MyChosenDelay);
```

4.2 VoIP

This was implemented by creating an exponential arrival rate. The exponential model was used as this describes the typical audio flow with VoIP.

As mentioned above, to add a new model to a layer three new functions are created

- Initialization Function
- Simulation Event Handling Function
- Finalization Function

The application layer will be the layer altered here. The file where most of the modifications will take place is “application.pc”. Four new files were also added to the layer, “exp_client.h”, “exp_server.h”, “exp_client.pc” & “exp_server.pc”.

In “application.pc” both header files were included. The EXP function was treated exactly like the CBR function in “application.pc”, the only difference is the number of Items to Send parameter is not used in EXP.

The EXP function was then entered in the Event Handling function and the Finalization function:

```
case APP_EXP_CLIENT:
```

```
AppLayerExpClient(node, msg);
```

```
break;
```

```
case APP_EXP_SERVER:
```

```
AppLayerExpServer(node, msg);
```

```
break;
```

The “exp_client.pc” and “exp_server.pc” files, which describe the functionality of EXP, were based on the CBR files. However the distribution is exponential for VoIP unlike CBR which is constant. The exponential distribution was therefore created in the “AppExpClientScheduleNextPkt” function. The function schedules the next packet the client will send, as the packet arrival rate is exponential the following formula was used:

$$\text{“nextScheduledPacketTime} = \text{clientPtr->meanInterval} * \log (1 - \text{temp});\text{”}$$

4.2.1 Scheduled Events

This function takes the value read from the input-configuration file (“app.conf”), and calls the appropriate initialization routine. The Finalization routine is also defined in the *application.pc* file. Within the Finalization function the command to print the stats is given which is then printed to *glomo.stat*.

4.3 MPEG Video Source (VBR)

The steps for creating a VBR traffic model are the same as the CBR traffic model.

- Alter *application.pc*
- Add new header file “vbr.h”
- Create two new .pc files “vbr_client.pc” and “vbr_server.pc”

Within “vbr_client.pc” the Initialization function is based on the CBR client. The parameters for the VBR source were then entered into the data pointer:

```

dataPtr->clientAddr = clientAddr;
dataPtr->serverAddr = serverAddr;
dataPtr->itemSize = itemSize;
dataPtr->meanInterval = meanInterval;
dataPtr->startTime = startTime;
dataPtr->endTime = endTime;
dataPtr->connectionId = dataPtr->sourcePort;

```

The timer is then set to initialise packet transfer at start time:

```

msg = MESSAGE_Alloc(node,
APP_LAYER,
APP_VBR_CLIENT,
MSG_APP_TimerExpired);
MESSAGE_InfoAlloc(node, msg, sizeof(AppTimer));
timer = (AppTimer *)MESSAGE_ReturnInfo(msg);
timer->sourcePort = dataPtr->sourcePort;
timer->type = APP_TIMER_SEND_PKT;
dataPtr->sessionStart = getSimTime(node) + startTime;
MESSAGE_Send(node, msg, startTime);

```

To implement the MPEG video source model the statistics for the Star Wars scene are taken from [2]. The three frames in an MPEG source are then taken into account:

```

if (next_frame_type == 'I'){
i->Get_Frame(Ui, size);
Up = Ui;
Ub = Ui;
if (prev_frame_type == 'B')
next_frame_type = 'B';
else next_frame_type = 'P';
prev_frame_type = 'I';
} // end if == I
else if (next_frame_type == 'P'){
p->Get_Frame (Up, size);
next_frame_type = 'B';
prev_frame_type = 'P';
} // end if == P
else if (next_frame_type == 'B'){
b->Get_Frame (Ub, size);
if (prev_frame_type != 'B') next_frame_type = 'B';
else if (GOP_count == 12)
{next_frame_type = 'I'; GOP_count = 0;}
else next_frame_type = 'P';
}

```

```

    prev_frame_type = 'B';
    } // end if == B
    GOP_count++;
    return(InterFrameInterval);

```

Where U_i , U_p , and U_b are the random seeds for the three frame types. The initial seed is set as u_0 .

Each frame has a corresponding statistic file containing the relevant data for the Star Wars trace. The statistics are located in Appendix I. The information is then taken out of these files using the *fgets* command. The information is then used in the following code

```

     $U_i = u_0$ ;
    size_ = 0;
    next_frame_type = 'I';
    prev_frame_type = '';
    GOP_count = 3;
    inter_frame_interval_ = 1.0/30.0; // 30 frame per second
    i = new Frame (I_File_1);
    p = new Frame (P_File_1);
    b = new Frame (B_File_1);
    running_ = 1;
    timeout();

```

The transmission order should be as follows:

I P B B P B B P B B I B B

Display order:

I B B P B B P B B P B B I

Change of Code:

Upon implementing this MPEG video source I found the simulations were providing results which were not what I expected. I tried altering the code but the results were still incorrect.

Therefore I chose a separate VBR code which was a function of the *mean Interval*. The following code was used:

```

// VBR send data packet at variable rate which is a function of mean Interval
nextTime = (clocktype)
    ((-log((double)(1.0-((double)pc_erand(dataPtr->seed)))))) *
    ((double)dataPtr->meanInterval));

```

The simulation results are based on the new VBR model.

Chapter 5 – Results and Analysis

5.1 Performance Evaluation

The following metrics were used in analysing protocol performance. These metrics were suggested by the Mobile Ad-hoc Networks (MANET) working group for routing protocol evaluation. The metrics are chosen to evaluate the efficiency in addition to the effectiveness of the protocol.

- **Packet delivery ratio:** The ratio of data packets delivered to the destinations and data packets originated by the sources. This number presents the routing efficiency or the throughput of the protocol.
- **Hop distance:** Average number of hops travelled by data packets that reached their destinations. Hop distance is closely related to the packet delivery ratio. The higher the delivery rate, the higher the hop count. Since only the packets that reach their destination are recorded, a low hop count means that most of the data packets delivered are destined for nearby nodes, packets sent to nodes further away are likely dropped. Therefore the hop count measure provides us with information about the survivability of the protocol.
- **End-to-end packet delivery latency:** The average delivery delay of the data packets from the source to the destination. This includes all delays due to buffering during route discovery time, queuing at the interface queue, and retransmission latency at the MAC layer, as well as propagation and transmission time.

5.2 Simulations

There are many simulations which can be performed to test the ad hoc protocol, the following were chosen.

First Circle (within 100m)

Experiment	Number of Nodes	Voice	Video	CBR
A1.1	3	1	1	1
A1.2	6	2	2	2
A1.3	9	3	3	3
A1.4	15	5	5	5

Table 2: Nodes in first circle

Second Circle (outside 100m)

Nodes in first circle do not generate traffic

Experiment	Number of Nodes	Voice	Video	CBR
A2.1	3	1	1	1
A2.2	6	2	2	2
A2.3	9	3	3	3
A2.4	15	5	5	5

Table 3: Nodes in second circle

Third Circle (outside 200m)

Nodes in first & second circle do not generate traffic

Experiment	Number of Nodes	Voice	Video	CBR
A3.1	3	1	1	1
A3.2	6	2	2	2
A3.3	9	3	3	3
A3.4	15	5	5	5

Table 3: Nodes in third circle

Each simulation has a node acting as a gateway, providing Internet access to the other nodes. This gateway node is connected through a wired network.

For each scenario the following parameters were used:

- Simulation Time – 100s
- Terrain Dimension – 1500 x 1500
- Routing Protocol – AODV
- Node Placement - **/nodes.input** file
- Node Mobility - none
- Propagation Pathloss – Two Ray
- Radio Frequency – 2.4 GHz
- Bandwidth – 2Mbps
- Power Range – 100m
- Application - **/app.conf**

/nodes.input:

This file varies for each scenario, however the first scenario, containing three nodes, had the following data: $\langle \text{node address} \rangle$, $\langle 0 \rangle$, $\langle (x, y, z) \rangle$

3 0 (274.0 , 810.0 , 0.0)

4 0 (537.0 , 826.0 , 0.0)

6 0 (407.0 , 926.0 , 0.0)

/app.conf:

CBR – For each constant bit rate (CBR) traffic source the data packets are of length 512 bytes and the number of items to send is 100,000 packets. The destination of each of the data sessions is the node attached to the wired network.

VoIP – Each VoIP traffic source has a “talk spurt/time” of 0.02 seconds. The session begins at time 0.001s and ends at 100s.

VBR – Each VBR traffic source has a packet length of 512 bytes with a mean interval of 0.001 seconds, a start time of 0.001 seconds and an end time of 100 seconds.

5.3 Results

The results were analysed by looking at each of the traffic sources and how each application affected the efficiency of the system. The results were obtained using the GloMosim simulator. The simulation produced three seeds (sets of values) which were then averaged to provide the following results.

Note: Any negative values, i.e. less than 0, indicated in the following graphs are due to graphing anomalies encountered in Microsoft's Excel program and as such should be ignored.

5.3.1 CBR Efficiency

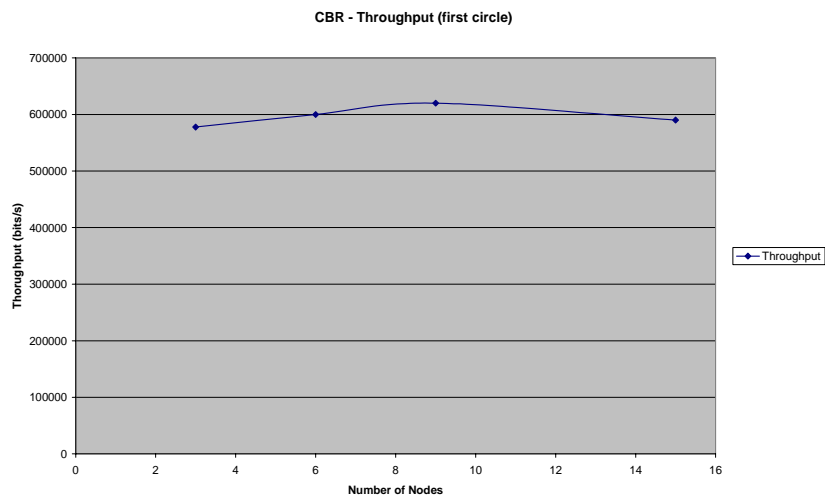


Figure 5-1 CBR Throughput in first circle

The simulation above was of a typical wireless network with all nodes within one hundred meters of the access point. The number of nodes increased from three to fifteen nodes. As can be seen in the above graph the throughput of the constant bit-rate application remains almost constant at 580,000 bits per second.

However in Figure 5-2 the constant bit-rate application is degraded once outside the range of the access point (i.e. second circle). The throughput performance decreases

dramatically (from 580,000 bits/s to 52,000 bits/s) when the number of nodes is increased to six. Again in Figure 5-3, with the nodes in the third circle creating the traffic, the throughput performance decreases with six or more nodes. The most significant degradation of performance can be seen between the second & third circle with three nodes. The efficiency has been degraded by 66 %.

The efficiency can also be seen in the packet delivery ratio. From Figure 5-4 the performance is ideal, there is no packet loss. As the nodes increase to six the ratio decreases to 0.7, which is still quite efficient. However there is significant packet loss when the number of nodes is increased to nine, at this point 90% of the packets are lost.

As the nodes mover further away from the access point the PDR decreases to 0.14, a loss of 86% of the packets. This is due to the fact that many route disconnects occur and a number of nodes that are part of the mesh transmit data packets. These transmissions cause collision and contention problems which make the scheme lose its effectiveness.

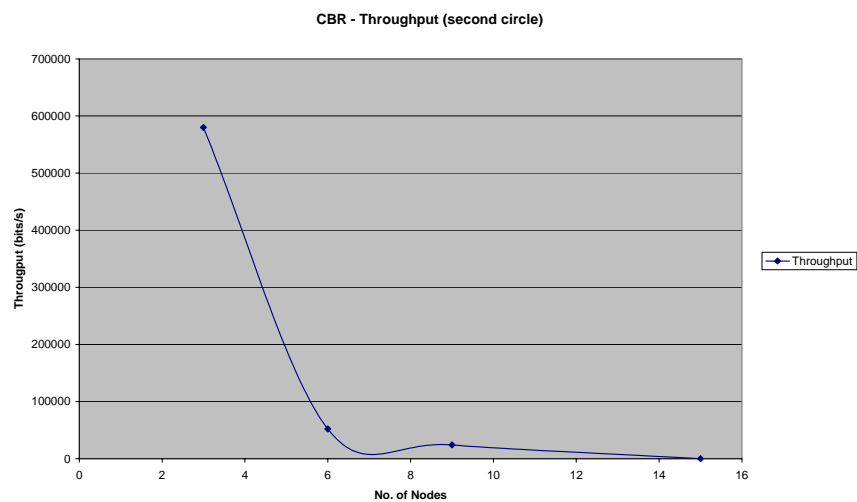


Figure 5-2 CBR Throughput of second circle

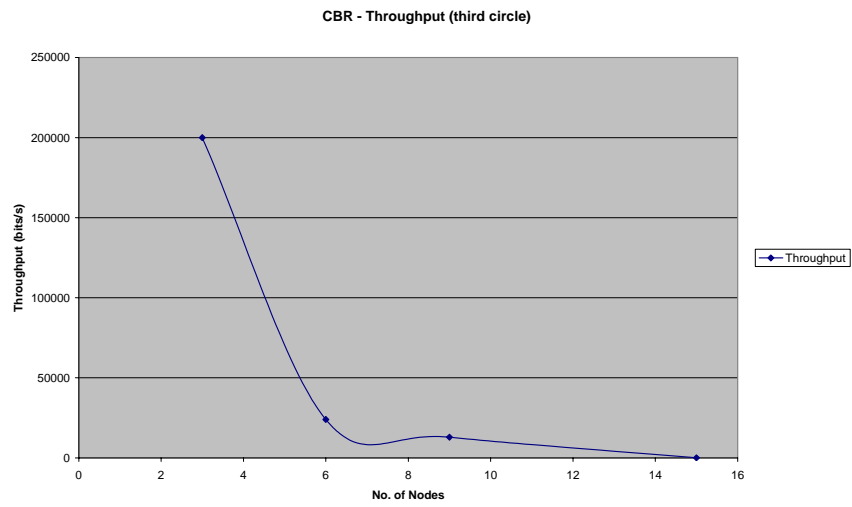


Figure 5-3 CBR Throughput of third circle

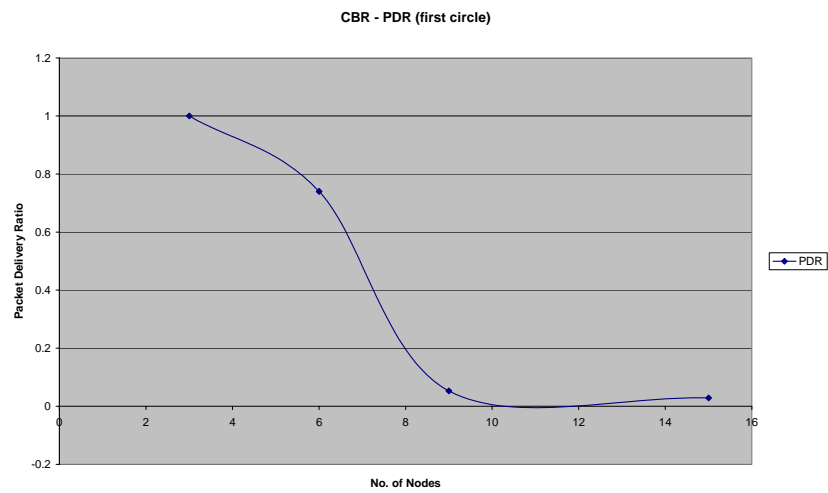


Figure 5-4 CBR PDR of first circle

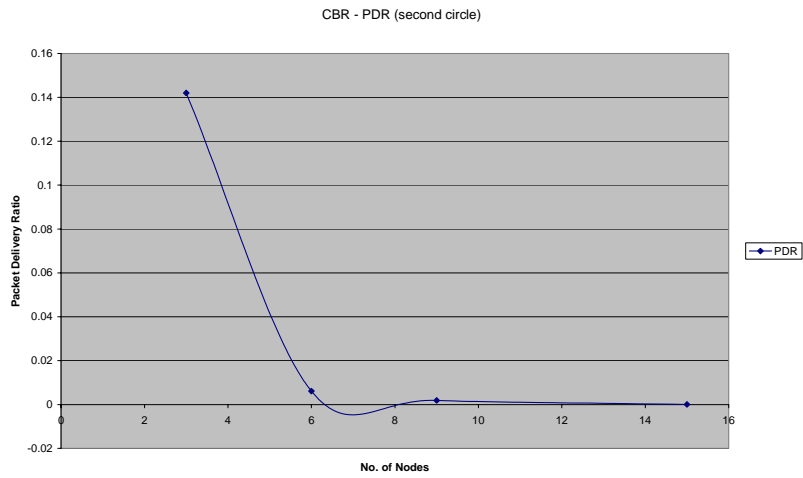


Figure 5-5 CBR PDR of second circle

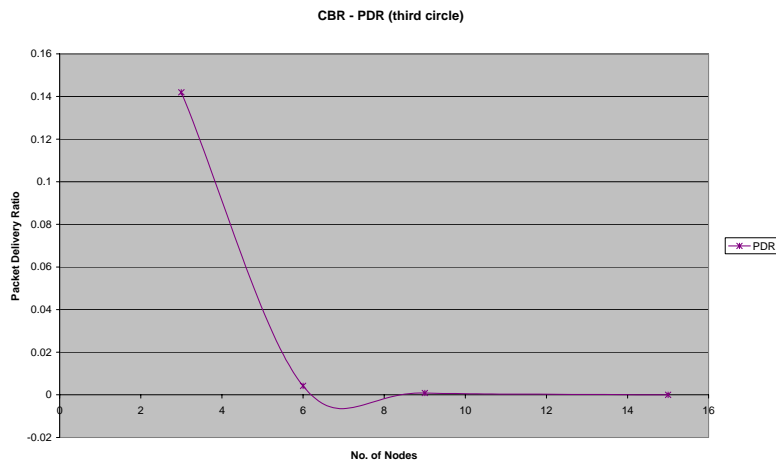


Figure 5-6 CBR PDR of third circle

5.3.2 CBR Latency

The End-to-end delay of a constant bit-rate is presented in figures 5-7, 5-8, 5-9. As expected, there are longer delays as the number of nodes increases. The delays for the first and second circle are similar with a maximum delay of 3.3s at 15 nodes. At the third circle we see greater delay of 4.6 seconds at 15 nodes. This is due to the AODV routing protocol. Only the data packets that survived to reach their destination can be measured. Those packets that are delivered in AODV take alternate and longer hop routes (Figures 5-10 to 5-12) and therefore have long delays, as the nodes increase. The latency is

increased but not to the extent where the application is completely ineffective in an ad hoc network.



Figure 5-7 CBR End-to-end delay of first circle

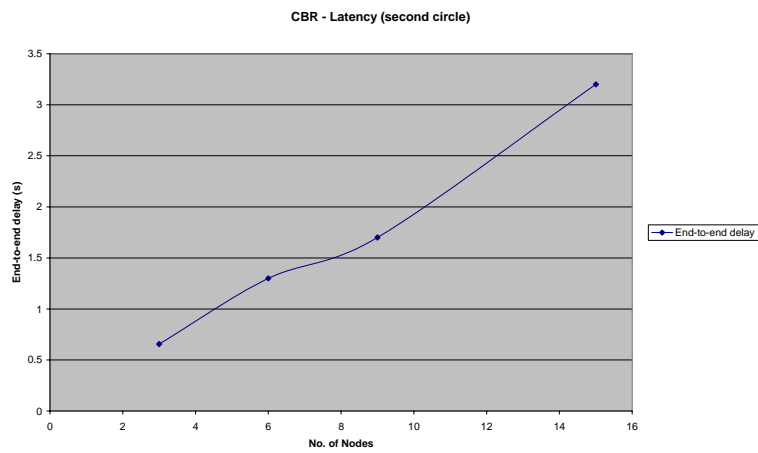


Figure 5-8 CBR End-to-end delay of second circle

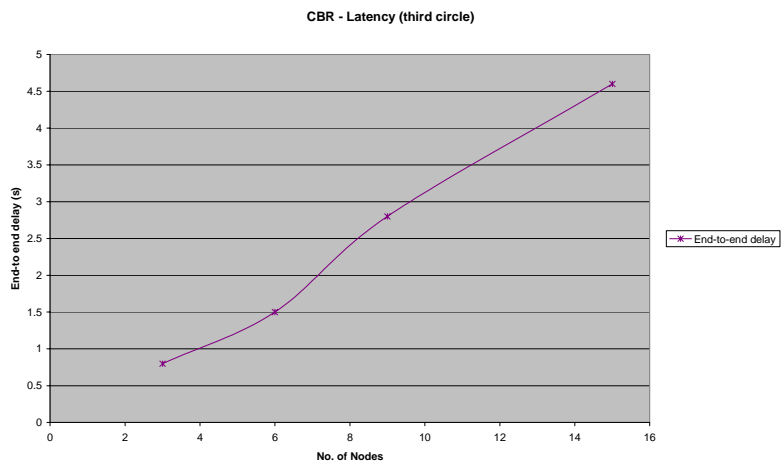


Figure 5-9 CBR End-to-end delay of third circle

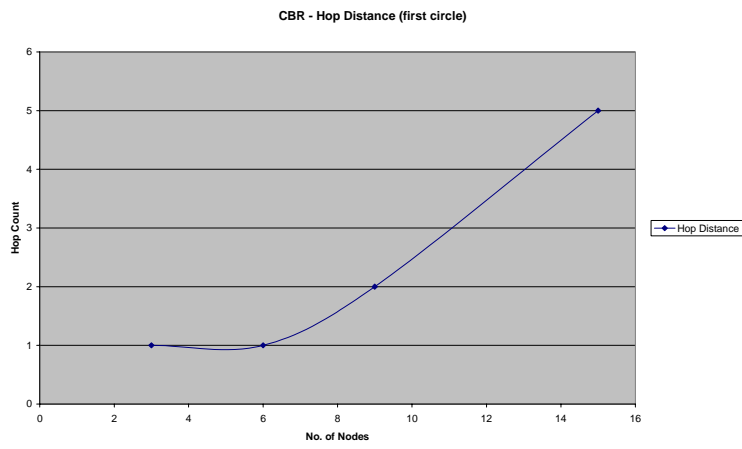


Figure 5-10 CBR Hop Distance of first circle

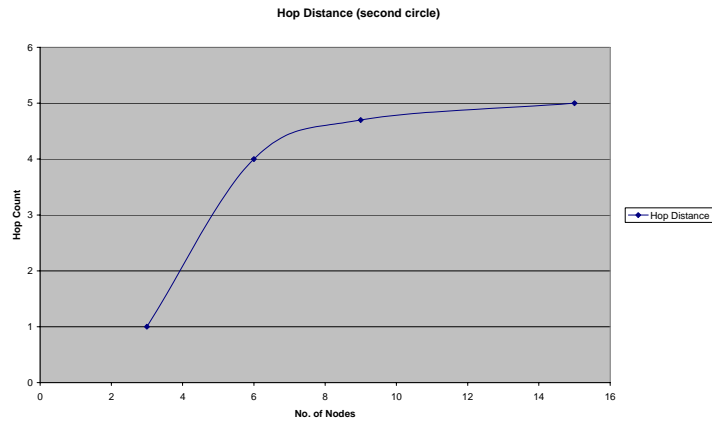


Figure 5-11 CBR Hop Distance of second circle

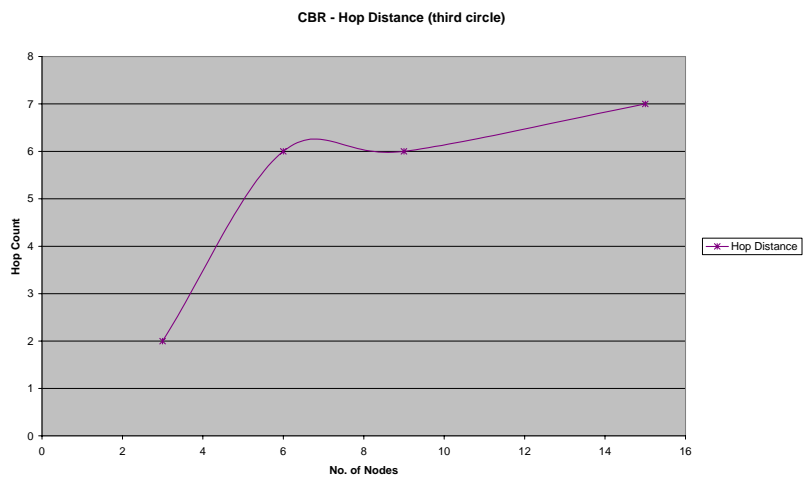


Figure 5-12 CBR Hop Distance of third circle

5.3.4 VoIP Efficiency

In Figure 5-13 we can see the throughput is not linearly dependant on the number of nodes. This has more to do with the exponential on-off distribution of data packets in VoIP. We can see from Figure 5-14 the overall throughput trend decreases as the node density increases, similar to the CBR second and third circles previously discussed.

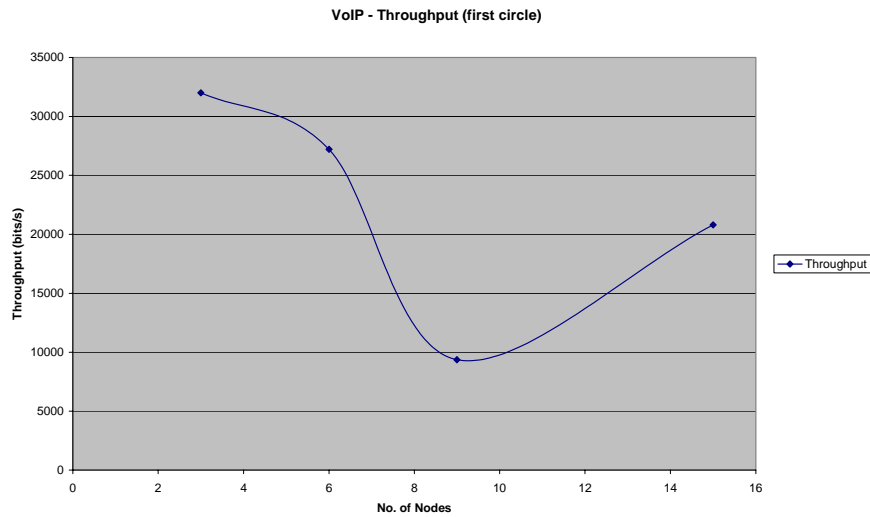


Figure 5-13 VoIP Throughput of first circle

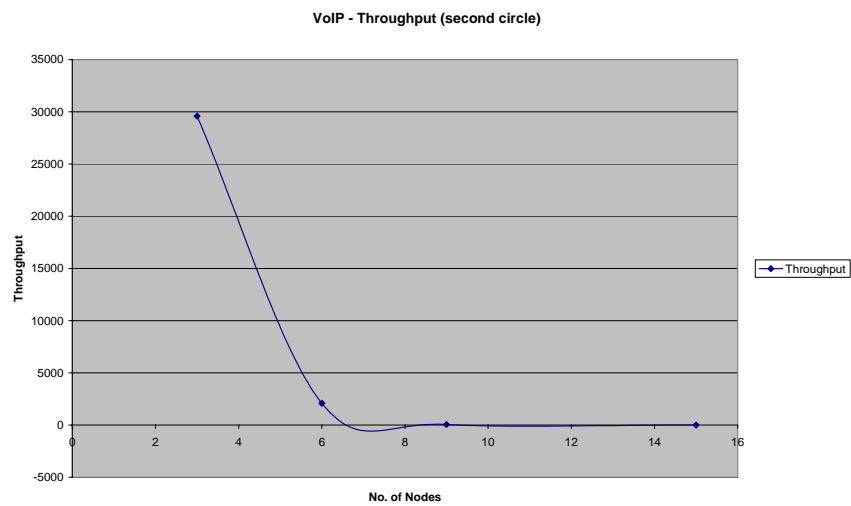


Figure 5-14 VoIP Throughput of second circle

5.3.5 VoIP Latency

The latency of the VoIP traffic, within range of the access point, increases very gradually up to around seven nodes. At this point the delay increases dramatically as more nodes are added to the system. As we move further away from the access point i.e. the second and third circle we see the latency increasing at earlier node densities and more dramatically.

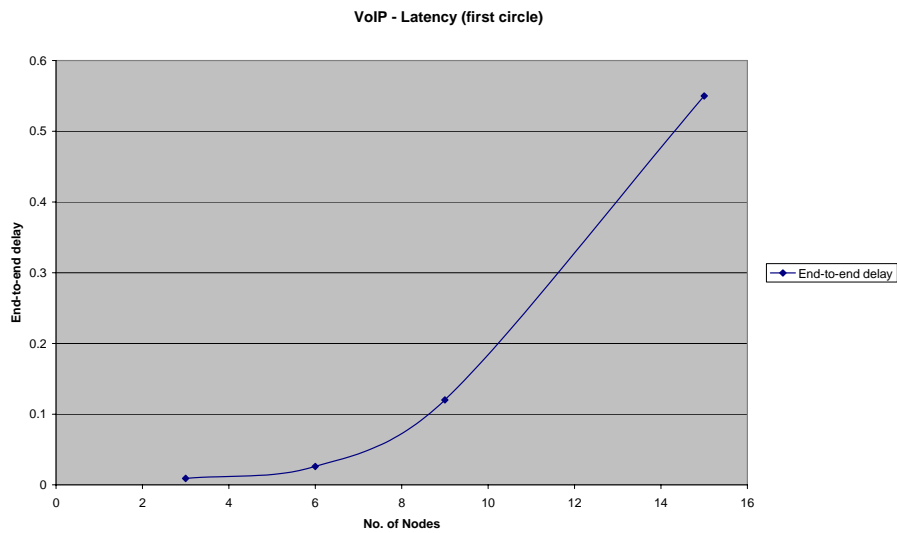


Figure 5-15 VoIP Latency of first circle

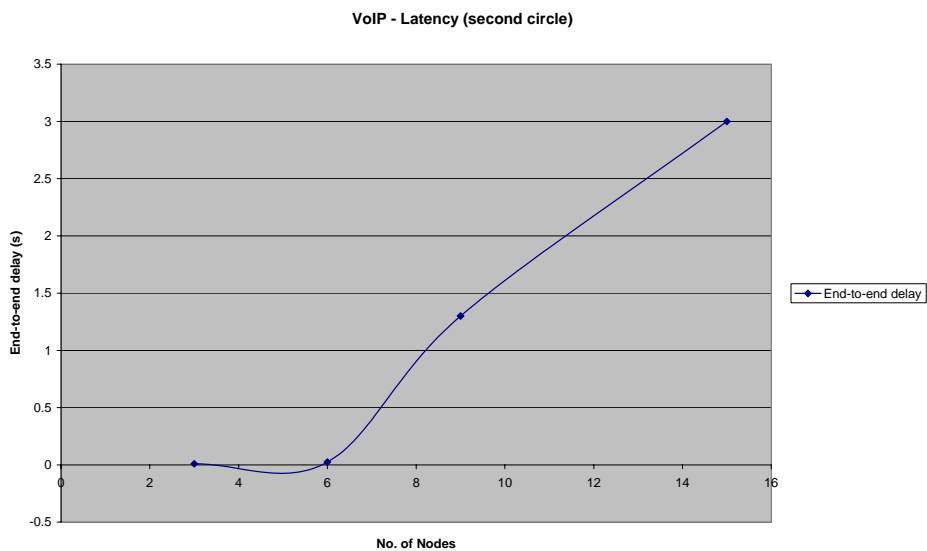


Figure 5-16 VoIP Latency of second circle

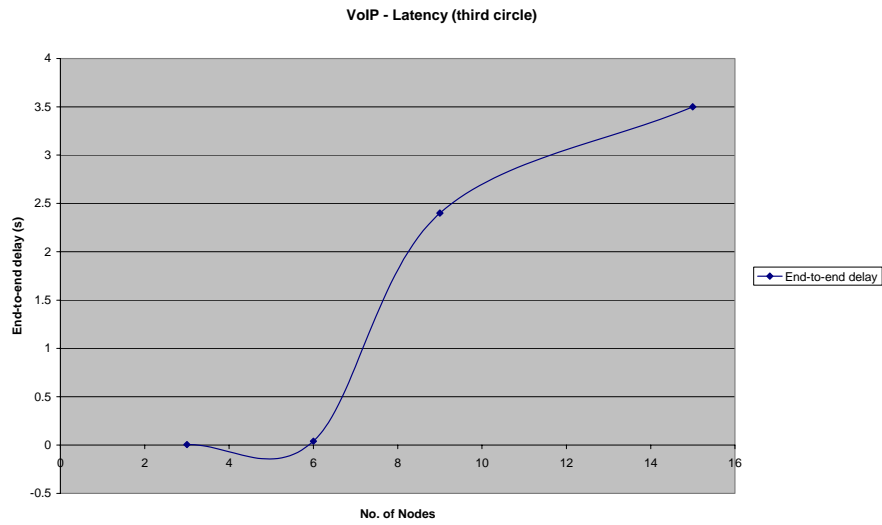


Figure 5-17 VoIP Latency of third circle

5.3.6 VBR Efficiency

The VBR throughput of the first circle remains constant, similar to the CBR traffic sources. However, as expected, the throughput is less for the VBR traffic source due to the non-constant nature of the source.

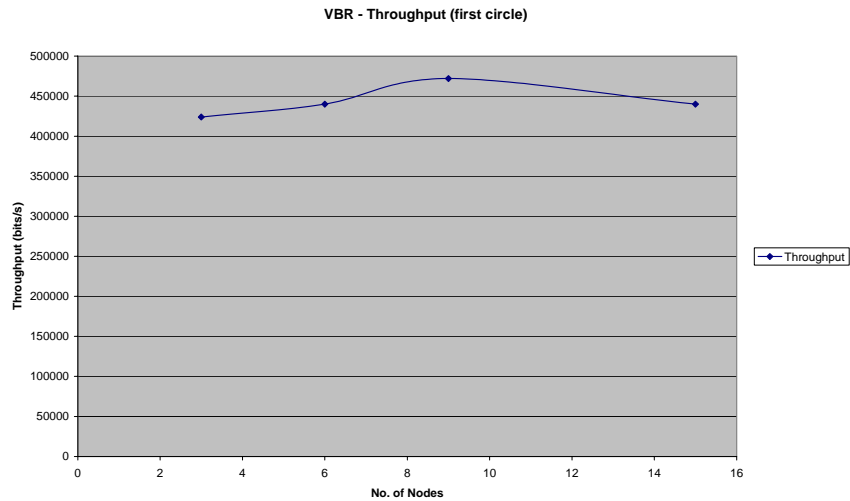


Figure 5-19 VBR Throughput of first circle

We can see from Figure 5-20 the throughput for the VBR source in the second circle decreases sharply as the node density is increased to nine.

In Figure 5-21 we see the efficiency of the application decreases similarly to the second circle. However the throughput in the third circle for three nodes is 22% less than the equivalent in the second circle.

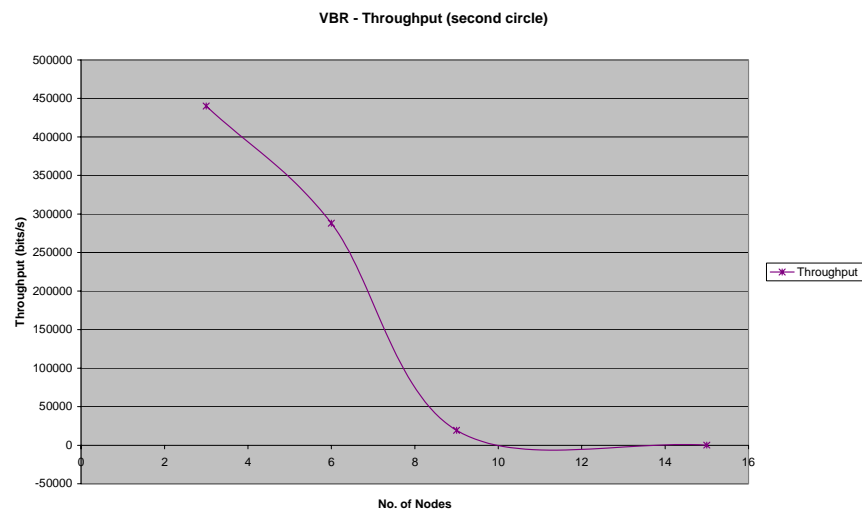


Figure 5-20 VBR Throughput of second circle

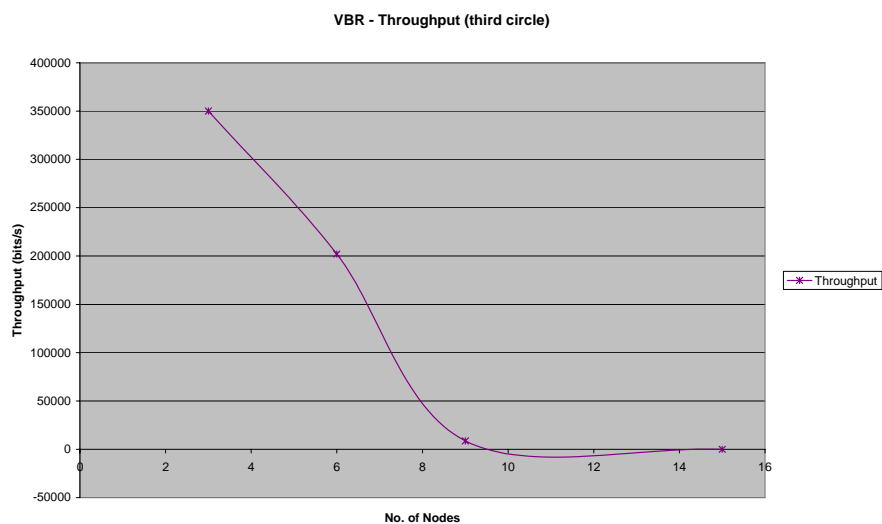


Figure 5-21 VBR Throughput of third circle

5.3.7 VBR Latency

The latency of the VBR traffic, within range of the access point, increases very gradually up to around eight nodes (delay = circa 0.06s). At this point the delay increases dramatically as more nodes are added to the system reaching 0.5 seconds at a density of 15 nodes. As we move further away from the access point i.e. the second and third circle we see the latency increasing at earlier node densities and more dramatically, the maximum reaching 3 seconds and 4 seconds at 15 nodes respectively.

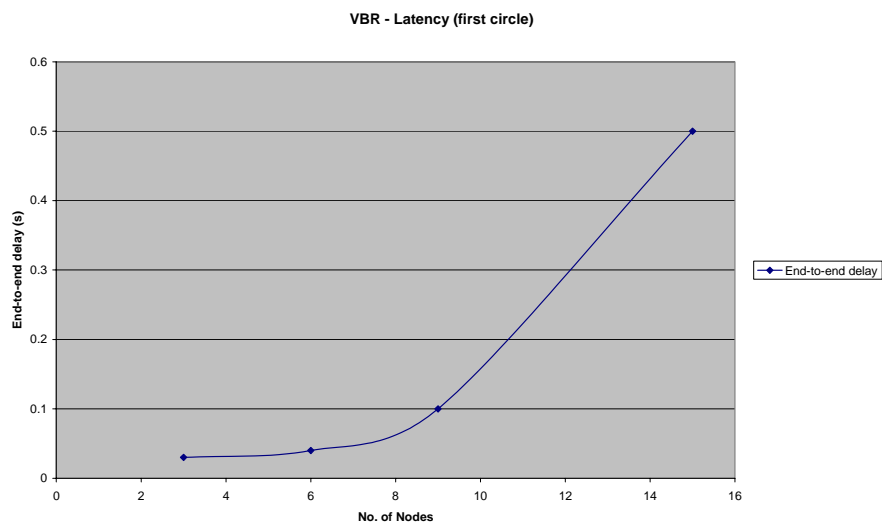


Figure 5-22 VBR End-to-end delay first circle

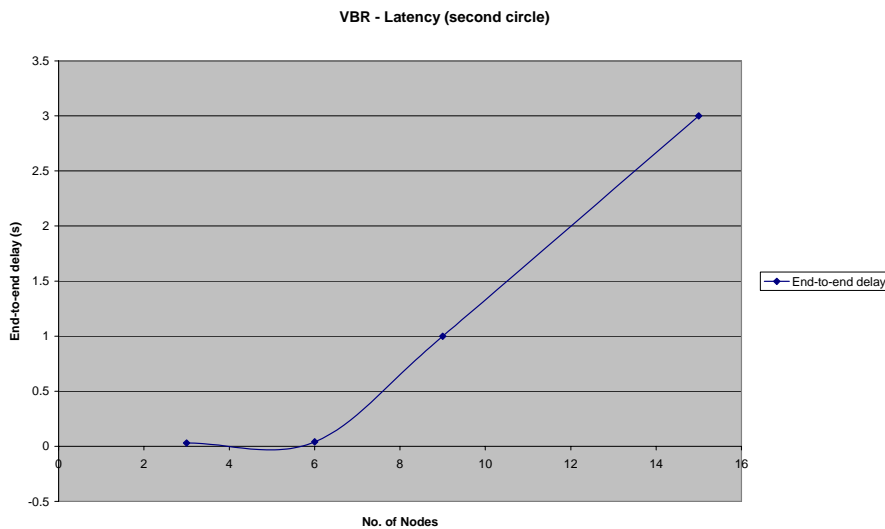


Figure 5-23 VBR End-to-end delay second circle

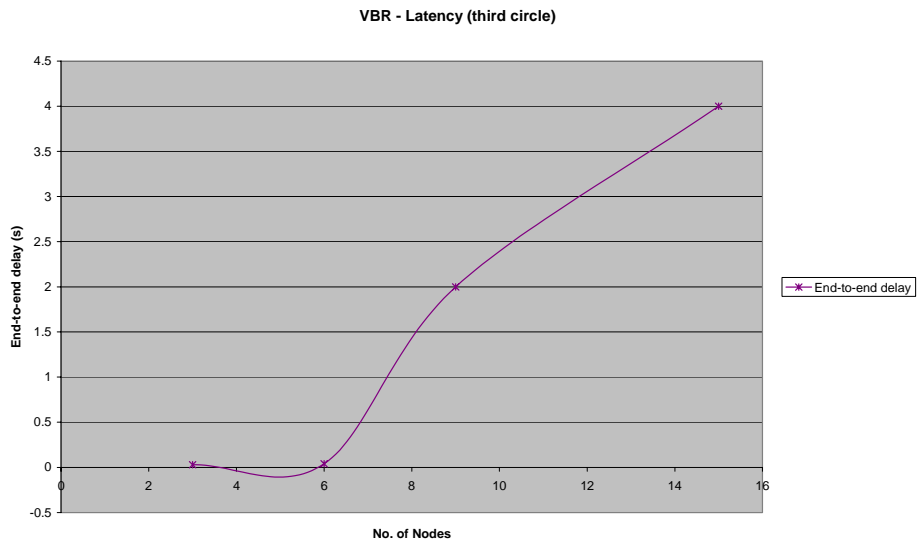


Figure 5-24 VBR End-to-end delay third circle

Chapter 6 – Conclusion and Further Research

From the results obtained in the course of the project we can deduce the boundaries of an ad hoc network. The ad hoc network is restricted to a two hop distance from the access point. Any distance greater than this will decrease the efficiency of the network as well as increase the latency.

We have also seen the ad hoc network should be restricted to six to eight nodes outside the access point range. As the traffic load and node density increased towards fifteen nodes, the efficiency decreased dramatically with a significant loss of packets.

Latency up to one second, while not ideal, can be workable. Therefore we can define the boundaries of the ad hoc network by stating that node densities, which incur latencies greater than one second, are inefficient for general use.

The ad hoc network seems to be quite capable of handling a constant bit-rate source but only as far as the second circle. Once the CBR traffic source enters the third circle, the efficiency of the MANET is degraded considerably with a node density of four.

From the simulations presented in this project we can ascertain that an ad hoc network is more suited for constant bit-rate and variable bit-rate traffic sources rather than VoIP.

Further Research

Update HTTP traffic model

The HTTP traffic model currently available in GloMosim is out of date. A new HTTP application could be created to reflect current web statistics. The HTTP traffic model could then be used as background traffic on the simulations in this project.

Use of more detailed models

To further evaluate the ad hoc network, more detailed and realistic channel models with fading and obstacles in the simulation.

Further protocol issues:

Fairness: Issues of fairness within an ad hoc network is an important aspect yet to be explored. Who gets priority?

Security: Some form of encryption will need to be explored to hinder foreign agent (nodes from other subnets) intercepting the data flow between other nodes within the ad hoc network.

References

- [1] Charles E. Perkins “Ad hoc On-Demand Distance Vector Routing” (draft-perkins-manet-aodvbis-00.txt)
- [2] Marwan Krunz, Satish K. Tripathi, “On the Characterization of VBR MPEG Streams
- [3] Lokesh Bajaj, Mineo Takai, Rajat Ahuja, Ken Tang, Rajive Bagrodia, Mario Gerla, “GloMoSim: A Scalable Network Simulation Environment”

Appendix I

Glomo.stat

Node: 0, Layer: AppCbrServer, (0) Client address: 21
Node: 0, Layer: AppCbrServer, (0) First packet received at [s]: 91.410686401
Node: 0, Layer: AppCbrServer, (0) Last packet received at [s]: 247.410946401
Node: 0, Layer: AppCbrServer, (0) Average end-to-end delay [s]: 0.021366975
Node: 0, Layer: AppCbrServer, (0) Session status: Closed
Node: 0, Layer: AppCbrServer, (0) Total number of bytes received: 100352
Node: 0, Layer: AppCbrServer, (0) Total number of packets received: 196
Node: 0, Layer: AppCbrServer, (0) Throughput (bits per second): 5146
Node: 0, Layer: AppFtpClient, from 0 to 1 (cid = 1), start = 150023013308, end = 150249638521 ns (closed), bytes sent = 7908 B, bytes recv = 279 B, throughput = 279156 bps
Node: 1, Layer: AppFtpServer, from 0 to 1 (cid = 2), start = 150024618917, end = 150252804942 ns (closed) bytes sent = 279 B, bytes recv = 7908 B, throughput = 277247 bps
Node: 2, Layer: AppTelnetClient, from 2 to 3 (cid = 1), start = 150013892282, end = 160013892282 ns (closed), bytes sent = 16 B, bytes recv = 592 B, throughput = 12 bps
Node: 3, Layer: AppTelnetServer, from 2 to 3 (cid = 2), start = 150020010923, end = 160024017837 ns (closed), bytes sent = 592 B, bytes recv = 16 B, throughput = 12 bps
Node: 10, Layer: AppCbrClient, (0) Server address: 28
Node: 10, Layer: AppCbrClient, (0) First packet sent at [s]: 82.490000000
Node: 10, Layer: AppCbrClient, (0) Last packet sent at [s]: 197.490000000
Node: 10, Layer: AppCbrClient, (0) Session status: Closed
Node: 10, Layer: AppCbrClient, (0) Total number of bytes sent: 24064
Node: 10, Layer: AppCbrClient, (0) Total number of packets sent: 47
Node: 10, Layer: AppCbrClient, (0) Throughput (bits per second): 1674
Node: 14, Layer: AppCbrClient, (0) Server address: 17
Node: 14, Layer: AppCbrClient, (0) First packet sent at [s]: 107.800000000
Node: 14, Layer: AppCbrClient, (0) Last packet sent at [s]: 273.900000000
Node: 14, Layer: AppCbrClient, (0) Session status: Closed
Node: 14, Layer: AppCbrClient, (0) Total number of bytes sent: 77824
Node: 14, Layer: AppCbrClient, (0) Total number of packets sent: 152
Node: 14, Layer: AppCbrClient, (0) Throughput (bits per second): 3748
Node: 16, Layer: AppCbrServer, (0) Client address: 18
Node: 16, Layer: AppCbrServer, (0) First packet received at [s]: 70.024710156
Node: 16, Layer: AppCbrServer, (0) Last packet received at [s]: 95.024230156

Node: 16, Layer: AppCbrServer, (0) Average end-to-end delay [s]: 0.024660156
Node: 16, Layer: AppCbrServer, (0) Session status: Closed
Node: 16, Layer: AppCbrServer, (0) Total number of bytes received: 3072
Node: 16, Layer: AppCbrServer, (0) Total number of packets received: 6
Node: 16, Layer: AppCbrServer, (0) Throughput (bits per second): 983
Node: 18, Layer: AppCbrClient, (0) Server address: 16
Node: 18, Layer: AppCbrClient, (0) First packet sent at [s]: 70.000000000
Node: 18, Layer: AppCbrClient, (0) Last packet sent at [s]: 95.000000000
Node: 18, Layer: AppCbrClient, (0) Session status: Closed
Node: 18, Layer: AppCbrClient, (0) Total number of bytes sent: 3072
Node: 18, Layer: AppCbrClient, (0) Total number of packets sent: 6
Node: 18, Layer: AppCbrClient, (0) Throughput (bits per second): 983
Node: 21, Layer: AppCbrClient, (0) Server address: 0
Node: 21, Layer: AppCbrClient, (0) First packet sent at [s]: 91.390000000
Node: 21, Layer: AppCbrClient, (0) Last packet sent at [s]: 247.390000000
Node: 21, Layer: AppCbrClient, (0) Session status: Closed
Node: 21, Layer: AppCbrClient, (0) Total number of bytes sent: 100352
Node: 21, Layer: AppCbrClient, (0) Total number of packets sent: 196
Node: 21, Layer: AppCbrClient, (0) Throughput (bits per second): 5146
Node: 28, Layer: AppCbrServer, (0) Client address: 10
Node: 28, Layer: AppCbrServer, (0) First packet received at [s]: 82.500210000
Node: 28, Layer: AppCbrServer, (0) Last packet received at [s]: 197.500450000
Node: 28, Layer: AppCbrServer, (0) Average end-to-end delay [s]: 0.012599387
Node: 28, Layer: AppCbrServer, (0) Session status: Closed
Node: 28, Layer: AppCbrServer, (0) Total number of bytes received: 24064
Node: 28, Layer: AppCbrServer, (0) Total number of packets received: 47
Node: 28, Layer: AppCbrServer, (0) Throughput (bits per second): 1674

VBR Source Model

I Model

-0.0125 0.6
-0.003 0.6
0.003 1

B Model

-0.5 0.000524242
-0.495 0.00143923
-0.485 0.00143923
-0.165 0.00144905
-0.155 0.00147469
-0.145 0.00150033
-0.135 0.00152597
-0.125 0.00155303
-0.115 0.00157606
-0.105 0.00159051
-0.095 0.00159051
-0.075 0.00161377
-0.065 0.0016218
-0.055 0.0016218
-0.025 0.50041
-0.015 0.50041
0.015 0.999211
0.025 0.999211
0.045 0.999219
0.055 0.999227
0.065 0.99925
0.075 0.99925
0.095 0.999265
0.105 0.999288
0.115 0.99932
0.125 0.999346
0.135 0.999371
0.145 0.999397
0.155 0.999416
0.165 0.999416
0.485 0.999476
0.495 1

P Model

-0.065 0.0028257
-0.055 0.00592267
-0.045 0.437586
-0.035 0.437586
0.005 0.453554
0.015 0.471467

0.025	0.945624
0.035	0.963401
0.045	0.982046
0.055	0.998982
0.065	1