Intelligent Routing Algorithm for Mobile Internet Protocol Television

By

Babangida Albaba Abubakar

A thesis submitted in partial fulfilment of the requirements of the University of Brighton for the Degree of Doctor of Philosophy

April, 2016

Table of Contents

Table of Figures

Abstract

Network bandwidth and server capacity are gradually becoming overloaded due to the high demand and rapid evolution of high quality multimedia services over the Internet. Internet Protocol Television (IPTV) is among the multimedia services that demand more of network and server resources, especially with the emergence of Mobile IPTV. It is imperative for the service providers to maintain good quality management services in order to satisfy their clients. To guarantee the required quality of service (QoS) and quality of experience (QoE) in IPTV, the server must have the required capacity and resources to serve all the clients' requests. The flexibility of IPTV services which provide users with the ability to stream multimedia content at anytime and anywhere they want, makes the demand for video-on-demand (VoD) services higher. However, the server bandwidth capacity is limited, and as such the numerous requests from the clients may exhaust all the available bandwidth depending on the number of requests at a given time, which may lead to the poor QoS and QoE.

In this research, a new algorithm called Intelligent Routing Algorithm for Mobile IPTV (IRA-MIPTV) is proposed. The algorithm combined features and advantages of Internet Protocol (IP), Mobile Ad hoc Network (MANET) characteristics and Content Delivery Network (CDN) based network architecture to improve on the QoS and QoE in mobile IPTV. The proposed algorithm is aimed at reducing total dependency on the server by the mobile nodes. The algorithm intelligently learns the best server or client to serve an incoming service request depending on the available server capacity and the number of requests received at a point in time. The routing decision is made by the Designated Server (DS) that selects and reroutes a request to the most appropriate server or client.

The novelty of this research work can simply be identified as the designing, developing and evaluating an Intelligent Routing Algorithm for mobile ITPV (IRA-MIPTV) that intelligently learns and select a reflective server or client to serve a particular service request on behalf of the server during high service demand. The selection depends on the server available bandwidth, load and proximity. The proposed algorithm also dynamically adjusts to server failure by assigning the role of designated server to the backup server and re-elect another backup server to guarantee service delivery at all times.

To validate the effectiveness of the proposed algorithm, different simulation tests were conducted using OPNET/Riverbed Modeler 18.0. A typical IPTV network, where packets are delivered over IP, and the proposed algorithm were modelled and incorporated into the Modeler. For the study to reflect more on real situations, live video programme was streamed using VLC media player. The packet's size and packet inter-arrival time data were collected and used in the simulation's environment. After conducting a series of simulation tests, the results showed that the proposed algorithm outperforms the normal IPTV system in server load reduction, high throughput and low amount of end-to-end delay, as well as adaptability and robustness. The results also showed that the efficiency of the proposed algorithm increases as the number of clients increase. It also confirmed that the algorithm reduces the server overload during high service request periods by using clients to serve some of the incoming service requests on behalf of the server. The algorithm produced low server and network load, low end-to-end delay, high throughput, adaptability and robustness.

Abbreviations and acronyms

The following are the abbreviations and acronyms used in this work:

ABAT	Available Bandwidth Estimation Tool
AS-FE	Application Support Functional Entity
ASN	Autonomous System Number
BDS	Backup Designated Server
BGCF	Breakout Gateway Control Function
CD&LCF	Content Distribution and Location Control Function
CD&SF	Content Delivery and Storage Function
CDF	Content Delivery Function
CDN	Content Delivery Network
CMTS	Cable Modem Termination System
CP	Content Protection
CPF	Content Provider Function
CSCF	Call Session Control Function
DNG	Delivery Network Gateway
DOCSIS	Data Over Cable Service Interface Specifications
DRM	Digital Rights Management
DS	Designated Server
DSG	DOCSIS Set-Top box Gateway
DVBSTP	Digital Video Broadcast Service discovery and selection Transport Protocol
DVR	Digital Video Recorder
EPG	Electronic Programme Guide
FB	Functional Block
FE	Functional Entity
FEC	Forward Error Correction
FFS	For Further Study

FLUTE	File Delivery over Unidirectional Transport
FREQ	Forward Request
HDTV	High Definition Television
HFC	Hybrid Fibre Coax
IANA	Internet Assigned Numbers Authority
IGMP	Internet Group Management Protocol
IMS	Internet Protocol Multimedia Subsystem
IPTV	Internet Protocol Television
ITF	Internet Protocol Television Terminal Functions
IW	Interworking
McCPF	Multicast Control Point Functional block
McRf	Multicast Replication Functional block
MGCF	Media Gateway Control Function
MLD	Multicast Listener Discovery protocol
MRFC	Multimedia Resource Function Controller
NACF	Network Attachment Control Function
NGN	Next Generation Network
NTP	Network Time Protocol
OAM&P	Operations, Administration, Maintenance and Provisioning
OSPF	Open Shortest Path First
PIM	Protocol Independent Multicasting
PVR	Personal Video Recorder
QAM	Quadrature Amplitude Modulation
QoE	Quality of Experience
QoS	Quality of Service
RACF	Resource and Admission Control Function
RERR	Reroute Error
RF	Radio Frequency
RFC	Request For Comments

RREQ	Reroute Request
RTP	Real-time Transport Protocol
RTSP	Real-Time Streaming Protocol
SADS	Service and Application Discovery and Selection
SC&DF	Service Control and Delivery Function
SCF	Service Control Function
SCP	Service and Content Protection
SHE	Super Head End
SIP	Session Initiation Protocol
SP	Service Protection
SSM	Source Specific Multicast
TCP	Transmission Control Protocol
TTL	Time To Live
UDP	User Datagram Protocol
URL	Universal Resource Locator
VCR	Video Cassette Recorder
VHO	Video Hub Office
VoD	Video on Demand
VSO	Video Serving Office

Acknowledgement

Praise is to Almighty Allah, the beneficent the merciful for making my study possible.

I wish to offer my sincere gratitude to my supervisors, Prof Miltos Petridis, Dr Deshinder Singh Gill and Mr Saeed Malekshahi Gheytassi for their support, guidance and inspiration. Thank you all for being excellent teachers and wonderful colleagues.

My special thanks to Petroleum Development Fund (PTDF) Nigeria, for sponsoring my studies, long life and prosperity to PTDF and Nigeria at large. My earnest thanks to Dr Eng. Muttaka Rabe Darma (former Executive Secretary, PTDF) for his moral support, encouragement and inspiration. Thank you for believing in me.

I would like to express my earnest gratitude to my parents for their prayers, love, encouragement and support throughout my study. I will forever remain grateful.

My heartfelt appreciation to my Wife Saihat, my kids (Fatima, Maryam, Aisha and Sadeeq) for their love, patient, understanding and support throughout my studies, your tolerance and motivation during my outburst moments gave me the strength to reach my goal. Also my special thanks to my brother Aliyu Albaba for his moral and financial support.

I would like to thank my examiners, Dr Diane Gan and Prof Haris Mouratidis, for their advice, suggestions and comments for improving my thesis.

My special appreciation to all my family, friend and well wishes for their moral support throughout my studies. Thank you.

Declaration

I declare that the research contained in this thesis, unless otherwise formally indicated within the text, is the original work of the author. The thesis has not been previously submitted to this or any other university for a degree, and does not incorporate any material already submitted for a degree.

Signed

Dated

Chapter 1: Introduction

1.1 Overview

Television and video on demand (VoD) services have been transformed from using the conventional radio signals and satellite technology to the use of the Internet to deliver video content to the end users. Internet Protocol Television (IPTV) is defined as the distribution of television or video content over a controlled Internet Protocol (IP) network [1]. The IPTV is not just about transmitting digital television services over internet technology; it is about reinventing television to better achieve the required goals and creating a video-centric next-generation Internet accessible on any device, be it mobile phone, computer, or smart TV, at any time and place the consumer chooses [2]. The major differences between IPTV and traditional television transmission are the digital video recorder (DVR), time-shifting capabilities (ability to stop, pause, and rewind real-time programmes) and a rich VoD environment, as well as providing services anywhere and at any time the client wishes. This makes demand for IPTV services in wireless networks higher and is expected to continue increasing over time [3].

To provide IPTV services with the essential guarantee of quality of service and quality of experience to the end users, the required minimum bandwidth by server to serve all the clients' requests has to be obtained. As the number of requests increases, the amount of bandwidth consumption also increases. However, the server bandwidth is limited and can be totally exhausted by the numerous requests from the clients. The numerous service requests can overload the server resources and capacity and congest the network, thus leading to poor quality of service.

To improve the required QoS and QoE, a Content Distribution Network (CDN) approach is being adopted and used by service providers. In the CDN approach, service providers replicate content over multiple distributed edge/replicative servers. The replicative servers that have the copy of the original content from the main server and are closer to the end user are used to serve the requests[4]. This approach is an effective way of improving the QoS and QoE by reducing the delay time, packet loss rate and server overload. Individual client's service requests are being rerouted to a server that is more appropriate in servicing the request. The selection of the suitable server depends on the number of network parameters such as: proximity to the end user, available bandwidth, throughput, requests volume and pattern and background traffic.

The Internet uses Internet Protocol (IP) for routing packets across different networks. IPTV, like any other Internet service, uses IP for packet routing. For the VoD, a separate connection is established for each request, thus leading to high bandwidth consumption, network conjunction and server overload. Bandwidth and throughput are among the major QoS parameters in delivering video content over the Internet. Therefore, there is need for an intelligent algorithm that allows the server to take advantage of clients' resources in serving some of the service requests during high service request demand. Such algorithm will reduce the total dependency on the server's bandwidth and resources, which will in turn improve the IPTV quality of service. The advantages of MANET, which allow mobile devices to communicate with each other without central control system, can be adopted to reroute packets from one client to the other in order to reducing the total dependency on the source node.

MANET is an autonomous system of mobile devices that are connected via wireless links without prior planning or need of any existing network infrastructure. The mobile nodes are communicating with each other without a centralized control system[3]. Mobile devices communicate with each other not only as a source or destination, but also as routers for packet forwarding in the wireless network.

Several studies have been conducted to improve the general IPTV quality of service. Some of these studies have been discussed in details, in next Chapters. However, the studies considered clients and network devices communications. The server nodes were not considered in those studies. Due to the significance of servers in providing IPTV services, this study emphases on server load reductions and work load.

The entire piece of this research is divided into four consecutive studies. The first study is for VoD services that use the unicast scheme. At this phase of the study, all the video streaming requests were served using unicast. The second study is the extension of the first study, where live and time shifted programmes using multicast scheme were added. The third study, with the inclusion of intelligent CDN-based architecture, added to the algorithm. At this stage multiple servers were added to the algorithm. The last study is the combination of all the studies into one Intelligent Routing Algorithm for Mobile IPTV. The election of a designated server and backup designated server are included at this stage. The OSPF election process is adopted to provide a backup mechanism in case of a server failure. The proposed algorithm intelligently selects a server or client to serve a request. In all the studies, the results show that the proposed algorithm improved the IPTV quality of service by providing high throughput, and reduced significant amounts of bandwidth consumption, server overload and network conjunction.

The algorithm was carefully designed, developed, implemented and tested in OPNET/Riverbed Modeler simulator. The features of the Modeler for supporting real-world data into its simulation environment make the simulation closer to the real application. A live video programme was streamed using BBC iPlayer and the video streaming data was captured using VLC media player. The captured data was stored in OPNET and used in all the simulation scenarios.

1.2 Motivation

The advantages, flexibility and mobility of wireless networks have extended wireless communication services to many areas, including IPTV. The great features of IPTV to provide access to video content to the end users wherever and whenever they choose, is making the demand for IPTV services higher and increasingly growing at an exponential rate.

The main motivation factor for this research is the poor quality of service and quality of experience encountered by IPTV services, especially during high service request demand. Some of the high demand situations include during international events, such as sports or festivals, and when a new popular movie is released, due to the high number of service demands on the server. Some of the major problems facing IPTV service delivery that resulted in the poor quality of service include packets lost, network connection, network devices and server overloads as well as high bandwidth consumption, especially on wireless mobile networks due to their limited resources compared with wired networks. Delivery of video content over the Internet requires a significantly higher amount of bandwidth to support the quality of service, reliability, scalability and security than the Internet's best-effort legacy might be able to provide.

The unique features of MANETs, such as absence of central administration and fixed network infrastructure, provide opportunities to many researchers in exploring new, related areas of study. Ad hoc wireless networks can be used to effectively optimise bandwidth consumption and enhance the general quality of service in IPTV. With the dynamic topology of MANETs to find and maintain destination addresses, routing information has to constantly be shared among the nodes in the network. However, sending too much redundant data to the wireless network is a waste of limited network resources. As a result, an effective technique needs to be in place to effectively utilize the advantages of MANETs and wireless networks with less use of network resources.

These requirements drove this research to present a better algorithm that will address some of the issues facing IPTV service delivery.

1.3 Aims and Objectives

This research aims to address the problems associated with delivery of video content over the Internet, such as high server workload and bandwidth consumption, as well as ensuring the general quality of service and quality of experience in IPTV. The main aim of this research is to develop, design and evaluate an intelligent routing algorithm for mobile IPTV, using IP, MANET and OSPF protocol techniques as well as considering the CDN approach, with the aim of effective delivery of video content over the Internet with the minimum required guarantee of quality of service. The objectives of this study include:

- Investigating existing IPTV and MANET routing protocols and algorithms
- Exploring different CDN server selection algorithms to find the best practice
- Finding the major problems facing the delivery of IPTV services in mobile devices
- Proposing a new intelligent routing algorithm for mobile IPTV
- Implementing the proposed algorithm
- Testing, comparing and evaluating the performance of the new algorithm

1.4 Research questions

- Can techniques be found and put in place to reduce bandwidth consumption and workload at the server node in IPTV?
- How can quality of service be enhanced in IPTV?
- How can other protocols be adopted to enhance IPTV quality of service?

1.5 Contribution to knowledge

The original contribution to knowledge of this research work can be described in the following points:

1. The leading originality of this research work is the systematic designing, developing and evaluating of the new intelligent routing algorithm for mobile IPTV called Intelligent Routing Algorithm for Mobile IPTV (IRA-MIPTV). Due to complexity of the algorithm,

it was developed in four different stages. The first stage deals with the unicast scheme only to provide video-on-demand service. The second stage combines the use of unicast and multicast schemes to deliver video-on-demand and live video programme services. The third stage is the extension of the first and second stages, where Content Delivery Network (CDN) architecture was added for load balancing across multiple servers. The fourth/final stage is the combination of all the three stages. The intelligent part of the algorithm was also added at this stage, where designated server and backup designated server are elected to address the issue of server failure and provide service delivery at all times.

2. Integrating the new algorithm into the standard server manager process in OPNET/Riverbed Modeler and the simulation of the IPTV network also provide the originality of the research work.
3. Based on the results obtained, the proposed algorithm adapts to different server capacities, such as bandwidth, number of request at a point in time and the location of the requesting client, to serve and an incoming request with the required quality of service. The algorithm also improved the IPTV general quality of service by providing high throughput and reduced bandwidth consumption, workload, packet end-to-end delay and jitter.

1.6 The structure of the thesis

The rest of the work is organised in 7 chapters as follows:

Chapter 2 reviews the IPTV features, schemes and services. It takes a closer look at the different functional architecture and the responsibility of each IPTV functional group. It also reviews the features of Mobile ad hoc Networks (MANETs) and current ad hoc routing protocols and algorithms. The chapter provides the essential knowledge concerning the development of ad hoc routing protocols and reviews on the different routing protocols and algorithms. Reviews on related studies, wireless ad hoc networks, current wireless technologies and CDN based server selection algorithms have also been discussed.

Chapter 3 discusses the methodology. The simulation tools used are explained in detail. The processes of integrating the proposed algorithm into OPNET are also discussed.

Chapter 4 introduces the implementation of the first phase of the proposed algorithm, where only unicast schemes were used in serving all the VoD requests received. This chapter details the designing, implementation, simulation testing and analyses of the results from the initial stage of the proposed algorithm.

Chapter 5 explains the implementation of the second phase of the proposed algorithm. It details how the first phase of the algorithm is extended to include both unicast and multicast schemes to serve an incoming request. The simulation results are also compared, analysed and discussed.

Chapter 6 explains different CDN-based network architecture, server selection algorithms and how new adaptive server selection algorithm for mobile IPTV is proposed. The design, implementation, simulation and results analyses of the server selection algorithm are also discussed.

Chapter 7 details the completed intelligent algorithm where all the studies are combined. The designated server and backup designated server election are also discussed. The implementation of the complete algorithm, simulation tests, results analyses and evaluation are all explained in detail.

Chapter 8 is for the discussion, conclusions and future work.

Chapter 2: Literature Review

2.1 IPTV

The definition of Internet Protocol Television (IPTV) as approved by ITU-T FG IPTV “is a multimedia services such as television/ video/ audio/ text/ graphics/ data delivered over Internet protocol (IP) based networks, managed to provide the required level of quality of service and experience, security, interactivity and reliability” [3]. Also [1]defined IPTV as the distribution of television or video contents over a controlled Internet protocol network, where the end users receives the contents through a set-top box which is connected to its normal broadband Internet connection. Therefore, IPTV can be described as a system through which television services and video on demand are delivered through controlled Internet protocol network using streaming technologies that managed and support quality of service (QoS), security, interactivity, and reliability.

IPTV isn’t only about transmitting digital television contents over the Internet technology but it is about creating new television services to better achieve the required goals and also to creates a video-centric next-generation Internet accessible on any device, such as mobile phones, computers, or Smart TVs, where and whenever consumer chooses [2]. The major differences between IPTV and traditional television transmission are the sophisticated digital video recorder (DVR) and time-shifting capabilities (ability to stop, pause, and rewind real-time programs) and a rich VOD environment. This makes demand for IPTV service in wireless networks higher and is expected to continue increasing over time [3].

Presently, IP is the initial point for all integrated services known as Triple play [5] and Quad play [6]. The model of Triple play is the integration of three services which include voice, high-speed data and television. Similarly, Quad play is the Triple play services plus the user mobility as illustrated in Figure 2.1.

The main characteristics of IPTV as stated in [7] are:

- Interactive TV support: IPTV systems have two channels. These channels allow the service provider to distribute interactive TV applications such as live television, high definition TV (HDTV) interactive games, quick searches on the Internet, etc.
- Time shifting. This service is used to record TV session allowing the costumer to watch the contents later.
- Personalized content: with the two-way communications feature of IPTV, it allows the end user to indicate, what does he wants to watch and when does he wants to watch it.
- Accessibility with several devices: IPTV contents can be viewed with several devices such as computers, mobile devices and televisions

Figure 2.1 IPTV Services

The main services offered by IPTV can simply be categorized into three main services:

a. Live television
b. Time shifted program
c. Video on Demand (VoD)

Currently, these services offered by IPTV uses two main schemes:

- **Multicast;** for delivering live video and time shifted programs
- **Unicast;** for video on demand and other applications.

2.1.1 IPTV Architecture

IPTV network architecture can simply be described as the connection of several broadband access systems that have the capability to support the required bandwidth for video delivery. The network topology can be divided into five parts as shown in figure 2.2. The parts were explained by [7] as follows:

- Head network
- Core/Backbone network
- Distribution network
- Access network
- Customer network

The head network: This is the video content network of the service provider which constitutes the fundamental core components of the infrastructure layer. The formation of this network is done through the devices that are capable of receiving, transforming and distributing the video contents to the clients. The content network serves as the main point of the infrastructure that receives request from all the subscribers and delivers contents to the set-top boxes.

This network has the most vital part of the service provider network as such all necessary action must be put in place to ensure security in information exchange by controlling accessibility to only authorized clients.

The backbone network: The distribution of video content from head network to distribution network is carried out by backbone network. The backbone network interconnects service providers and applications with the service providers. The technologies commonly uses by backbone network are Gigabit Ethernet, SONET/SDH, and xWDM technologies. The network may include different architecture such as point to point, ring, double ring, etc. Routing and switching between the aggregation routers and end routers are the most vital part of the IPTV backbone network infrastructure. The IPTV routers should be scalable and high-performance devices.

Distribution network: The distribution network connects the end of the backbone network with the aggregation router or access network. Data transmission and switching tasks are being performed by distribution network. Its main function is the information multiplexing from different service providers and it adapts the transport system to the specific characteristics of the subscriber loop.

Access network: The access network comprises of all the required facilities that transmit the contents to the clients and manages the clients' demands by the return channel. The first and most significant requirement of an access network is to have enough bandwidth to support multiple

IPTV channels for each subscriber, while allows other services such as telephony and data. The channel transmission is sent through multicast to the distribution network and access network. At present, xDSL and FTTx are the technologies often used by access network.

Customer network: The customer network is the network that provides communication and information exchange between devices connected to it and access network. The communication medium in this network may be wired or wireless. Each device connected to this network enjoys the services through the residential router. The router connects the customer network with the service provider network. The common technologies used in this network are the Fast Ethernet and Wi-Fi (IEEE 802.11n/g).

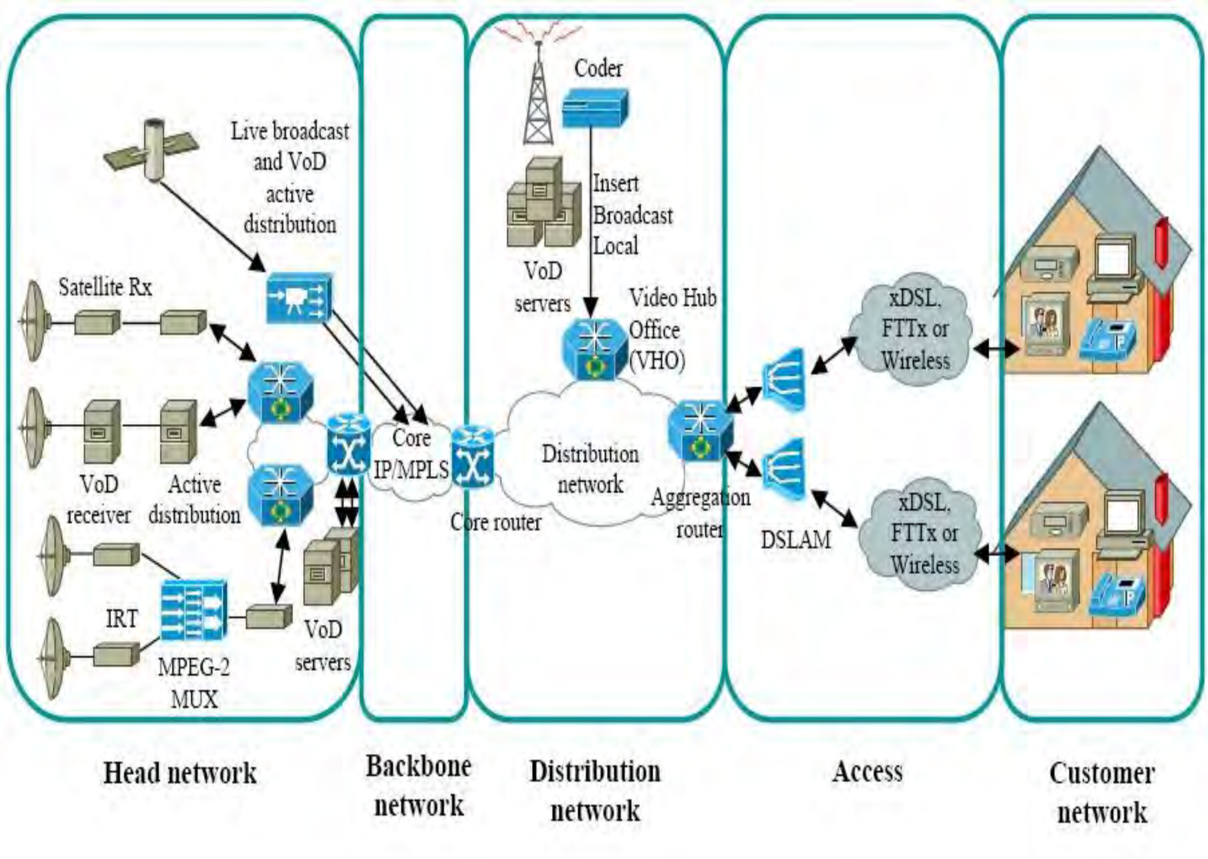


Figure 2.2 Typical IPTV Architecture [7]

2.1.2 IPTV Functional Architecture

Based on the recommendation of ITU-T [8], the IPTV functional architecture is classified into seven functional groups, which includes:

- End-user functions
- Application functions
- Service control functions
- Content delivery functions
- Network functions
- Management functions

- Content provider functions

The function and functional blocks described in figure 2.3 below, are common to all the architectural approaches, i.e. the Non-NGN IPTV, NGN-based non-IMS IPTV and NGN IMS-based IPTV functional architectures. These three functional approaches were described in [8] as follows:

1) **Non-NGN IPTV functional architecture:** The Non-NGN IPTV architecture is based on existing network components and protocols/interfaces. The technological components, protocols and interfaces used in this architecture are already in use and therefore this approach can be best described as a representation of typical existing IPTV networks providing IPTV services. This architectural approach can optionally be used as the foundation for evolution towards the other two IPTV architectures.

2) **NGN-based non-IMS IPTV functional architecture:** The NGN non-IMS IPTV architecture make use of the components of the NGN framework reference architecture as identified in [ITU-T Y.2012] to support the delivery of IPTV services, in conjunction with other NGN services if required.

3) **NGN IMS-based IPTV functional architecture:** The NGN IMS-based IPTV architecture utilizes components of the NGN architecture including the IMS component to support the provision of IPTV services in conjunction with other IMS services if required.

The IPTV functional groups described by [8] are as follows:

1. **End-user functions**

The end-user functions comprises of IPTV terminal functions and home network functions.

IPTV terminal functions: The responsibilities of IPTV terminal functions (ITF) are to collects control commands from the end-user and interacting with the application functions to obtain service information (e.g. Electronic Programme Guide - EPG), content licenses and decryption keys. The IPTV terminal functions interacts with the service control and content delivery functions to receive the IPTV services and also provide the capability for content reception, decryption and decoding.

- ***Application client functions:*** The application client functions exchange information with the IPTV application functions to support IPTV services and other interactive applications.
- ***Service and content protection client functions:*** The service and content protection (SCP) client functions interact with IPTV SCP functions to provide service and content protection. The SCP client functions verify the usage rights and decrypt and optionally watermark the content.
- ***Content delivery client functions:*** The content delivery client functions receive and control the delivery of the content from the content delivery and storage functions (CD&SF). After receiving the content, the content delivery client functions can optionally use the SCP client functions to decrypt and decode the content, and can also optionally support playback control.
- ***Control client functional block:*** The control client functional block allows the ITF to initiate service requests to

the IPTV service control functional block in order to prepare for the connection to the content delivery functions.

Home network functions: The home network functions provide the connectivity between the external network and each IPTV terminal device. These functions include IP connectivity, IP address allocation and configuration from the network functions to the IPTV terminal devices. All data, content and control traffic must pass through the home network functions in order to enter or exit the end-user's IPTV terminal device. The home network functions serve as the gateway between the IPTV terminal functions and the network functions. The home network functions are comprised of the following functional block.

- Delivery network gateway functional block: The delivery network gateway functional block provides IP connectivity between the external network and the IPTV terminal devices. IP Connectivity, acquiring IP address and configurations for the home network and IPTV terminal devises are parts of the functions of delivery network gateway.

2. **Application functions**

The IPTV application functions allow the IPTV terminal functions to select and acquire content. When receiving requests from IPTV terminal functions, the IPTV application functions executes application authorization and execution of IPTV service logic based on user profile, content metadata and other information retrieved from relevant entities. The IPTV application functions also interacts with content delivery functions to arrange the delivery of media content to IPTV terminal functions through content delivery functions.

The IPTV application profile can optionally include:

a. **End-user settings:** End-user settings include information related to the capabilities of the end user's IPTV terminal devices. An IPTV end user may be associated with more than one IPTV terminals with different capabilities.

- Global settings (e.g., language preference).
- Linear TV settings.
- List of subscribed linear TV service packages.
- VoD settings (e.g., parental control level).
- Personal video recorder (PVR) settings (PVR network/local preferences, PVR user limitations, PVR storage limit).
- IPTV service action data which encompass information related to the actions the user can optionally have taken while accessing services, e.g.:– list of linear TV services (or programmes) that the user has paused and is hence likely to resume later, including the bookmark value associated with the pause; – list of VoDs that the user has ordered, and associated status; – list of PVR contents that the user has asked to be recorded.

b. Content preparation functions: The content preparation functions control the preparation and aggregation of the contents such as VoD programme, TV channel streams, metadata and EPG data, as received from the content provider functions. The content preparation functions can optionally pre-process (e.g., transcode or edit) the content in advance of passing it to the content delivery, IPTV application and SCP functions.

Content preparation may optionally include the insertion of a watermark for the purpose of content tracing. Additionally, it may create content tracing metadata to facilitate subsequent embedding, into the content, of a content-tracing watermark. The content-tracing metadata is appropriate when multiple copies of the protected content will be created and distributed to end users.

c. **Service and content protection (SCP) functions:** The SCP functions manage the protection control of the services and content. Content protection includes control of access to contents and the protection of contents using methods such as encryption.

   Service protection includes authentication and authorization of access to services and sometimes protection of the services using methods such as encryption.

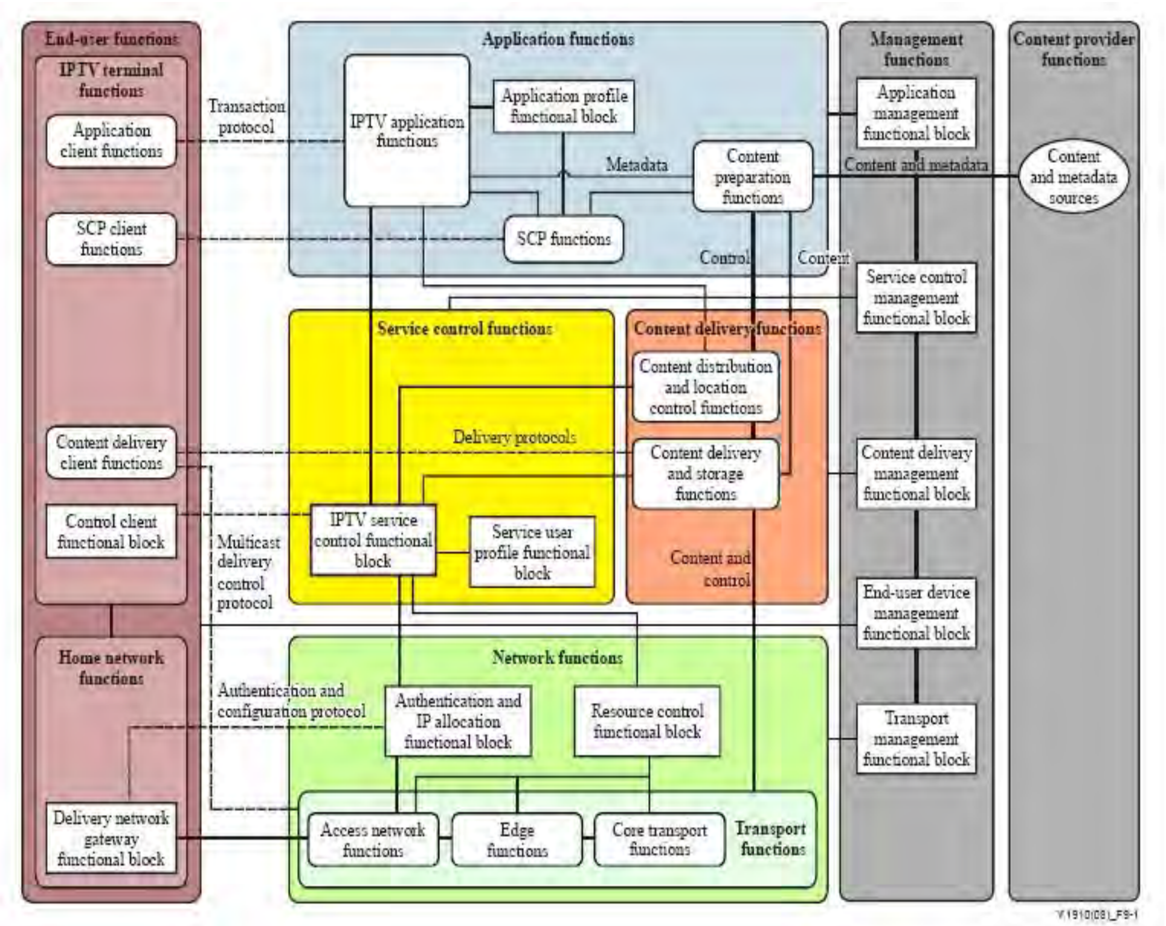


Figure 2.3 IPTV Functional Group Architecture [8]

3. **Service Control Functions**

   The IPTV service control functional block provides the functions to manage service initiation, modification and termination requests, service access control, establish and maintain the network and system resources required to support the IPTV services requested by the IPTV terminal functions.

   The Optional services provided by IPTV service control functional block includes:-

   - Provision of registration, authentication and authorization functions for the end-user functions
   - Process requests from IPTV application functions and forward them to the content delivery functions in order that the content delivery functions select the most appropriate content delivery and storage functions, for delivering content to the end-user functions
   - Request the content delivery functions or application functions to collect charging information.

   Service user profile functional block: The service user profile functional block is part of the service control functions that:

- stores end-user service profile (i.e., IPTV services subscribed to)
- stores subscriber-related data (e.g., who pays the incurred charges)
- stores end-user location data
- stores end-user presence status (e.g., online/offline)

- performs basic data management and maintenance functions:– updating and storage of "user subscription data" or "network data" (e.g., the current network access point and network location) and
- responses to queries for user profiles for authentication, authorization, service subscription information, subscriber mobility, location, presence.

4. Content Delivery Functions

The content delivery functions (CDF) perform cache and storage functionalities and deliver the content according to the request from the end-user functions. More than one instance of storage and delivery functionalities can optionally exist and the content delivery functions select the appropriate one(s). The content delivery functions control the distribution of content to multiple instances of storage and delivery functionalities in order to maintain the same content at the multiple instances. Content is distributed to the content delivery functions before or during the service offering process. Content delivery functions interact with end-user functions and it support unicast, multicast or both mechanisms.

The content delivery functions are comprised of content distribution and location control functions (CD&LCF) and Content delivery and storage functions (CD&SF) [8].

- **Content distribution and location control functions:** The CD&LCF include, but are not limited to:
  - Handling interactions with the IPTV service control functional block.
  - Controlling distribution of content from the content preparation functions to the content delivery and storage functions.

    - Information gathering concerning content delivery and storage functions, such as resource utilization, resource status (e.g., in-service and out-of-service), content distribution information and load status.
    - Choice of appropriate content delivery and storage functions to serve end-user functions according to certain criteria, e.g., information gathered and the terminal capability.

- **Content delivery and storage functions:** The CD&SF store & cache, process and distribute the content. The process is perform under the control of content preparation functions and the distribution is being perform among instances of content delivery and storage functions based on the policy of content distribution and location control functions.

  The content delivery and storage functions are responsible for delivering content to the content delivery client functions using the network functions e.g. unicast, multicast or both mechanisms.

The content delivery and storage functions include, but are not limited to:

- Handling interaction with the IPTV service control functional block.
- Handling content delivery to end-user functions.
- Caching and storing content and associated information.
- Insertion, watermarking, transcoding and encryption of the content.
- Distributing content within the content delivery and storage functions.
- Managing interaction with the content delivery client functions e.g., trick mode commands.

- Reporting status such as load status and availability to content distribution and location control functions.
- Generating charging information.

5. **Network Functions**

The network functions provide the IP layer connectivity to support IPTV services and the network functions are shared across all the services offered by IP to end end-user functions. Some of the network functions include:

- **Authentication and IP allocation functional block:** The authentication and IP allocation functional block offers the functions that authenticate the delivery network gateway functional block which connects to the network functions. It also provides allocation of IP addresses to the delivery network gateway functional block and optionally to the IPTV terminal functions.
- **Resource control functional block:** The resource control functional block provides functionality of controlling resources which have been allocated for the delivery of the IPTV services through the access network, edge and core transport functions.
- **Transport functions:** The transport functions provide IP layer connectivity between the content delivery functions and the end-user functions. The transport functions include access network functions, edge functions, core transport functions and gateway functions.
    - Access network functions: Access network functions are responsible for aggregating and forwarding the IPTV traffic sent by the end-user functions into the

edge of the core network and forwarding the IPTV traffic from the edge of the core network towards the end-user functions.

- Edge functions: Edge functions are responsible for forwarding the IPTV traffic aggregated by the access network functions towards the core network, and also to forward the IPTV traffic from the core network to the access network functions.
- Core transport functions: Core transport functions are responsible for forwarding IPTV traffic all over the core network.

6. The Management function

The management functions provide of the overall system monitoring and configuration functions. The functions can be centrally deployed or in a distributed method. Management functions include the following functional blocks:

- Application management functional block
- Content delivery management functional block
- Service control management functional block
- End-user device management functional block
- Transport management functional block

7. Content Provider Functions

The Content provider functions provide the content and associated metadata to content preparation functions, which include content and metadata sources. The content and metadata sources include content protection rights sources, content sources and metadata sources for the IPTV services.

2.2 IPTV Services

The two main schemes used in offering IPTV services are Multicast and Unicast.

2.2.1 IPTV Multicast

Multicasting is the networking technique of distributing the same packet concurrently to a group of consumers. Internet Protocol (IP) Multicasting is a proficient bandwidth-conserving mechanism where a source node simultaneously transmits the same content to a group of destination nodes called multicasting group [9]. In IPTV content, this means that the video transmission server transmit the same video contents to all the clients that subscribe to it at the same time. The multicast technics is suitable if a group of consumers require the same set of data at the same time, or when the common data can be receive and cache by a clients until it is needed.

IP multicast is a bandwidth conserving technology that reduces traffic by simultaneously transmitting a single stream of information to many corporate and homes recipients. Some of the applications that take advantage of multicast include video conferencing, corporate communications, distance learning, and distribution of software, stock quotes, and news. Figure 2.4 below shows how IP multicast transmit the same video content to multiple receivers simultaneously. IP multicast delivers application source traffic to multiple receivers without burdening the source or the receivers while using a minimum of network bandwidth [10]. At the diverge paths in the network, multicast packets are replicated by the router enable with Protocol Independent Multicast (PIM) and other supporting multicast protocols, which results to the efficient delivery of data to multiple receivers.

The primary advantage of using multicasting scheme is the conservation of network bandwidth. Where there is a common need for the same information needed by a group of consumers, multicast transmission may provide significant bandwidth savings of up to 1/N of the bandwidth compared to N separate unicast clients. Applications such as MPEG video requires high amount of available network bandwidth for single stream, in this case IP multicast is the best way to send to more than one receiver simultaneously.

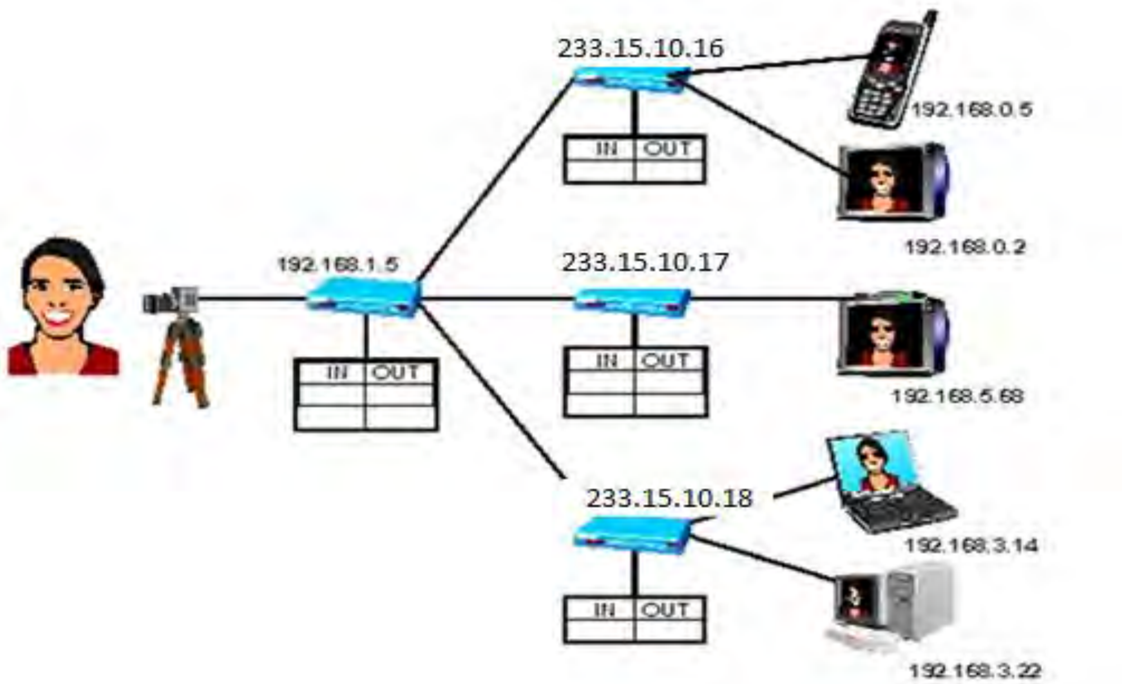


Figure 2.4 Multicast Tree Structure [11]

In the above example shown in Figure 2.4, the receivers whom are the selected multicast group that are interested in receiving the video data stream from the source. The receiving clients indicate their interest by sending a join group massage to the layer three devices such as routers that support Internet Group Management Protocol (IGMP). The routers are then responsible for delivering the data from the source to the intended recipients. When a recipient wanted to leave the group, it sends the leave group message to the routers. The routers dynamically create a multicast distribution tree by the use of Protocol Independent Multicast (PIM). The video data stream will after that be forwarded only to the network devices that are in the pathway between the source device and the receivers.

A multicast group is a subjective group of recipients that expresses an interest in receiving a particular data stream. The group has no physical or geographical boundaries, the hosts can be located anywhere on the Internet or any private Internetwork [10]. A host must be a member of multicast group to receive the data stream. IP multicast addresses specify a set of hosts that have joined an IP multicast group and declared their interest in receiving multicast stream selected for that group. The organisation that is responsible for control of IP multicast address assignment called Internet Assigned Numbers Authority (IANA) assigned the IPv4 Class D address space for the use of IP multicast. Hence, all IP multicast group addresses fall between 224.0.0.0 through 239.255.255.255 [10]. It should be noted that the Class D address range is used only for the group address or destination address of IP multicast transmission where as the source address for multicast datagrams is always the unicast source address.

Assignment of IP Multicast Address

Some of the IP multicast address ranges include; reserve link local address, globally scope address, GLOP, source specific multicast and limited scope address. For detailed information on this see [12]. Table 2.1 below show the IP multicast address ranges. IANA reserved addresses ranging from 224.0.0.0 to 244.0.0.255 (244.0.0.0/24) to be used by protocols on a local network section. Router should never forward packets with these addresses.

Table 2.1: Multicast addresses range [10]

Description	Range
Reserved link local address	224.0.0.0/24
Global scope addresses	224.0.1.0 to 238.255.255.255
Source specific multicast	232.0.0.0/8
GLOP addresses	233.0.0.0/8
Limited scope addresses	239.0.0.0/8

Normally packets with link local destination addresses are sent with a time to live (TTL) value of 1, which are not forwarded by a router. These addresses are being used by network protocol to communicate important routing information and automatic routing discovery. For instance Open Shortest Path First (OSPF) uses 224.0.0.5 and 224.0.0.6 to exchange link-state information [10].

Globally scoped addresses are addresses ranges from 224.0.1.0 to 238.255.255.255. These addresses are used to multicast data between organisations and over the Internet. IANA reserved some of these addresses to be used for multicast applications such as 224.0.1.1 for Network Time Protocol (NTP) [10]. Source Specific Multicast (SSM) is an extension of PIM protocol that allows effective data delivery mechanism. SSM addresses are reserved in the range 232.0.0.0/8.

GLOP is a process that reserved the addresses in the range of 233.0.0.0/8 for statically defined addresses by organisations that already have their

Autonomous System Number (ASN) reserve. The addresses range was originally assigned by RFC 2770 and is determined by the organisations 16-bit ASN allocation. For example, AS 62010 is written in hexadecimal format as F23A, if the two octets are separated you will get F2 and 3A and these will results to 242 and 58 when converted in decimal format. These values result in a subnet of 233.242.58.0/24 that would be globally reserved for AS 62010 to use [10].

The limited scope addresses sometimes refers to as administratively scope IP multicast are in the range of 239.0.0.0/8. It was described in RFC 2365 that limited scope address are to be constrained to a local group or organisation such as companies and university to use it for local multicast applications that are not forwarded within outside their domain. Normally routers are configured with filters to prevent multicast traffic in this addresses range from flowing outside of as autonomous system or any other define domain.

2.2.1.1 Multicast Distribution Trees:

Multicast distribution trees are created by the routers that have multicast capability for path control that IP multicast traffic will pass through in the network to deliver traffics to all the receivers. There are two basic types of multicast distribution tree; source tree and shared tree.

Source Trees

Source tree also known as the Shortest-Path Tree (SPT), is the simplest form of a multicast tree. As the name implies, it is small spanning tree with its shortest path root from the source and the branches forming a spanning tree through the network to the destination. Figure 2.4 shows an example of shortest path tree.

The notation (S,G) estimates the shortest path tree, where S is the source IP address and G is the group multicasting IP address. For instance as it shows in the figure 2.5 (S, G) will be (192.168.1.1, 224.1.1.1). The notation (S, G) implies that a separate SPT exist for each individual source sending to each group. For example if Host C is sending traffic to Host A and Host B, a separate shortest path tree would exist with a notation of 192.168.3.3, 224.1.1.1).

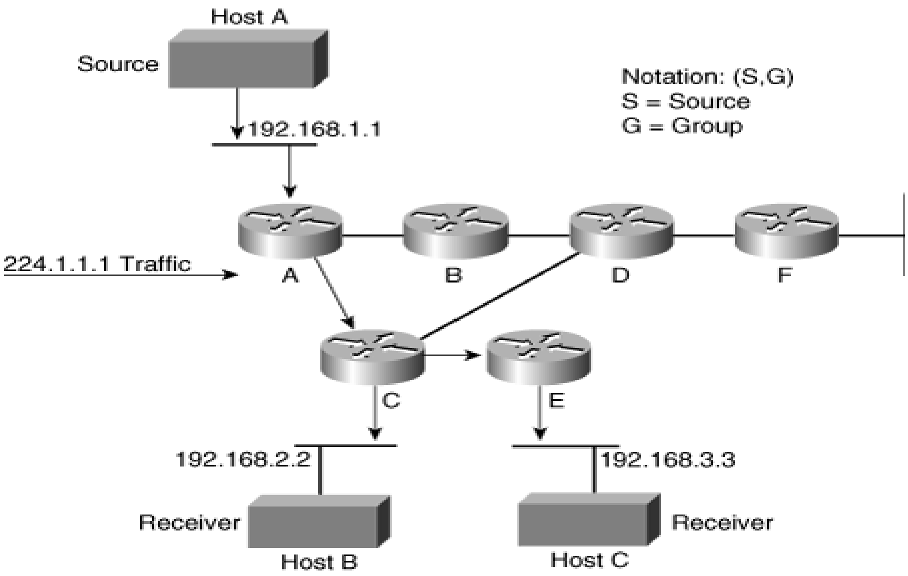


Figure 2.5 Source Tree [10]

Shared Trees

Shared tree uses a single common root placed at some chosen point in the network. This shared point is called rendezvous point (RP). Unlike the source tree where the root starts at the source, shared tree can have the RP point at any point in the network. Figure 2.6 shows the example of shared tree.

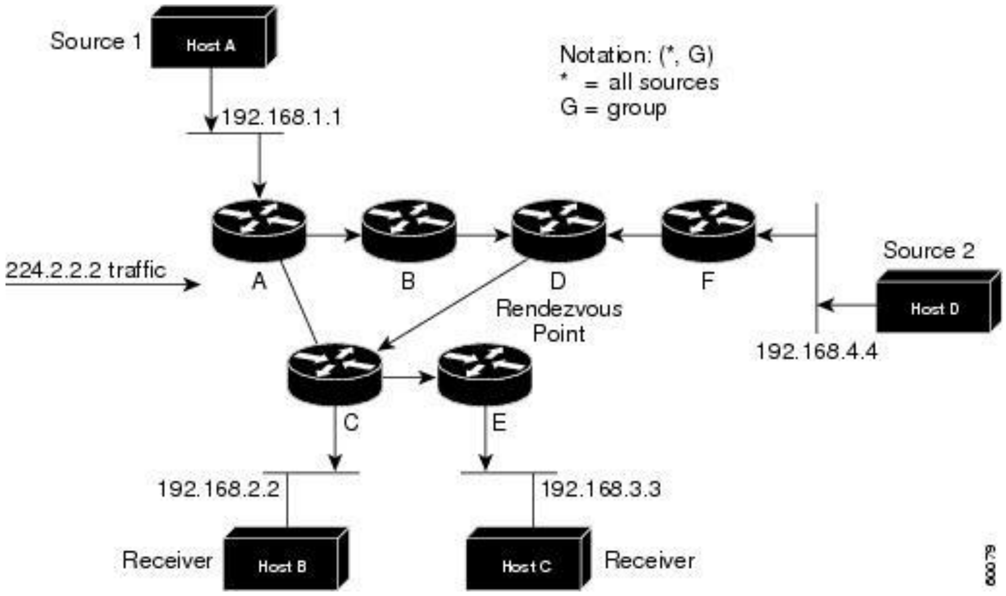


Figure 2.6 Shared Trees [10]

In the example above (figure 2.6), the rendezvous point is located at Router D. the source traffic is sent towards the rendezvous point on a source tree, then the traffic is forwarded on the shared tree from the RP to all the receivers except those receiver that is located between the source and the RP.

In IPTV services, the video contents are stores and serves from the source node to the multicasting group. Therefore, the shortest-path tree is the best option for this type of service. In this research I have considered the shortest-path tree in the designing of the proposed algorithm, as it provides not only the shortest path but also the control on the video content since the source node starts the process.

2.2.1.2 Advantages and disadvantages of Multicast

As we can see in the figure 2.3 above, the source node uses one connection stream to transit the content to the group of users, this shows that multicast scheme;

a. Saves network bandwidth,
b. Lower network congestion and eliminate traffic redundancy
c. Reduce source load.

IP Multicast is User Datagram Protocol (UDP) base, which is best effort in nature, however, despite the great advantages it has it is own shortcomings which include the following taking form [13].

a. Packets drop are expected in the best efforts nature of UDP
b. No congestion avoidance due to lack of TCP *back off* and *slow-start* window mechanism which automatically adjust the speed of data transfer and therefore provide a degree of congestion avoidance within the network.
c. Duplicates: Some multicast protocol mechanisms (e.g. Asserts, Registers and SPT Transitions) result in the occasional generation of duplicate packets and also routers are sending multicast packet to multiple interfaces, this will increase the probability that multiple copies of multicast packet may arrive at the receiver, until the multicast routing protocol converge and eliminate the redundant path.
d. Out of Order Delivery: Some multicast protocol mechanisms may also result in out of order delivery of packets.
e. From the source to the destinations each and every router and device used must support Internet Group Management Protocol (IGMP). This fact has significant influence on the network cost.

Transmitting digital video using IP multicast schemes saves bandwidth, but it does not guarantee sufficient video quality to all users that are receiving the same content. When a customer wants to change a channel, the set-top box must join the appropriate multicast group. Therefore it is difficult to guarantee the quality of the video properly in each subscriber TV.

2.2.2 Unicast

Internet Protocol (IP) Unicast is a mechanism where separate contents from the source server are sent to each destination host [9]. Unicast streaming establishes one to one connection between a server and client, in transmitting video contents, that is, each clients will receive a separate stream requested as it can be seen if figure 2.7.

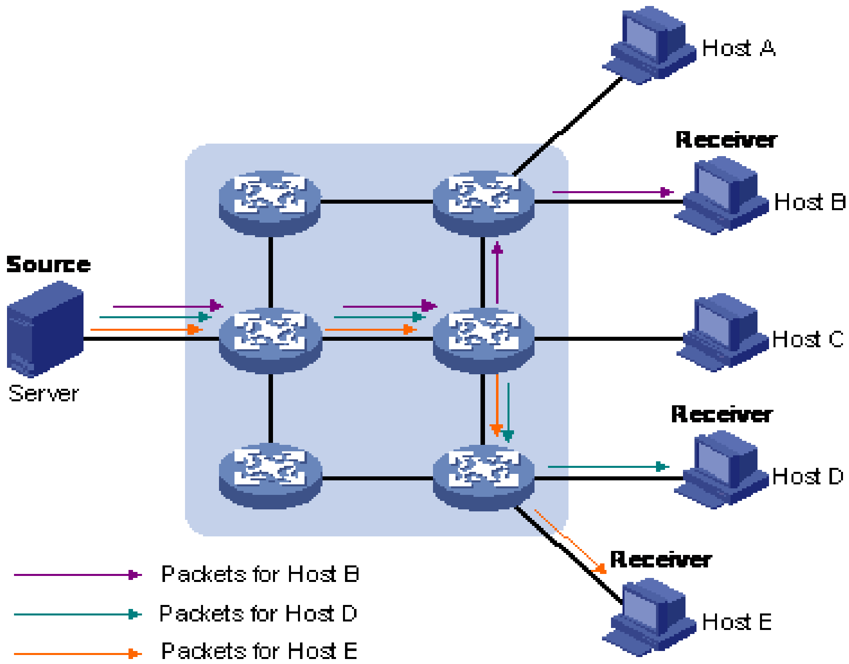


Figure 2.7: Unicast System

2.2.2.1 Advantages and disadvantages of unicast

Unicast scheme has several advantages, this include;

a) It is a predominant form of transmission on Local Area Network and within the Internet. LANs (e.g., Ethernet) and IP networks support the unicast transfer mode.
b) Most applications protocols such as http, SIP, SMTP, FTP and telnet, which make use of the Transmission Control Protocol (TCP) are familiar with the unicast standard.
c) Streaming is completely controlled from the source to the destination and security of access and content is guaranteed without any additional special software, which further increases the cost of the system implementation.

The major disadvantages of unicast scheme are:

a. High consumption of network bandwidth
b. Create high network congestion
c. Overload the source node.

As the network resources and bandwidth of the IPTV distribution network are finite, there more the subscribers are requesting IPTV services using the unicast scheme, the more easily the users jeopardise the QoE level [14]. From the above facts, we can understand that, unicast major problems are on the network bandwidth consumption, high network congestion and work overload to the server, which are major problems that cause poor quality of service (QoS) and user experience.

2.3 Related Study

Packets forwarding to mobile nodes that are connected via wireless links are always challenging task. The network topology may change frequently due to the nodes' movements. A good routing protocol is the one that always forward packets along or close to the shortest path from source to destination, and has the ability to adapt to topology changes quickly. Establishing efficient route between multi-points involves difficult tasks than that of the point-to-point routing. As such multicast routing is generally a challenging task. Multicast support in a wireless multi-hop and ad hoc network has received a lot of attention from researchers in the past and recent past, examples [15][16] and [17]. Several multicast solutions based on geographic solutions were proposed. According to [18] and [19], Multicast for Multiple Geographic regions (MgCast) was proposed for the nodes to construct and maintain a multicast tree by periodically broadcasting it is location information. This means that all the nodes have knowledge of the location of each other. Therefore the node that receives the multicast packet forwards it to the nearby nodes that are closer to the destinations.

Another approach was proposed in [20], in which the network setup is organized in squared zones that support two-tier membership management. Each node is aware of it is location and as such can easily find which zone it is located. In each zone, one leader is selected and responsible for the packet forwarding and management of the zone members that are involves in the multicast. The multicast packets are used to discover the path between the zones and the relevant source. According to [19], the scheme is efficient in mobility since the route discovery is done by the destinations, but still managing the membership, identifying and selecting the zone leader as well as the nodes most inform the leader on the multicast session they are involves results to network overhead.

Location guided Steiner tree was proposed in [21] for the small group multicast using heuristic techniques. In this work the source node estimates and starts to build the multicast tree, adding at each step the node that is closed to the already built tree. The packets are sent from the source node to the each destination in each sub-tree through the identified tree. The procedure is iteratively repeated to all the destinations. This approach is only to address small group of multicast nodes, hence to construct a heuristic several times to deliver each multicast packet results in the increase of its complexity. Figure 2.8 below illustrate Steiner tree structure.

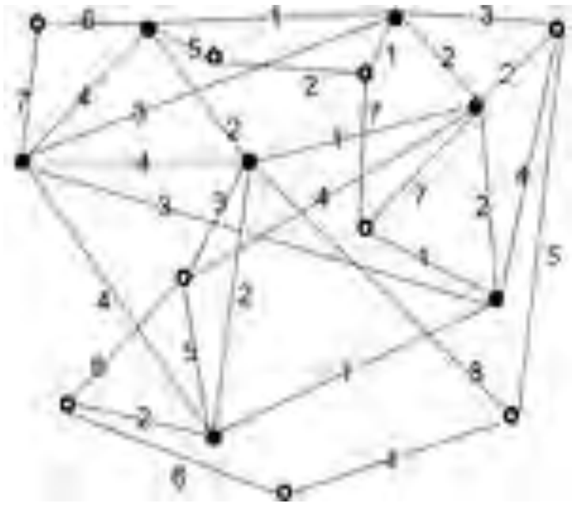


Figure 2.8: Steiner Tree structure [21]

Similar solution to the multicast problem was also proposed by [22] called Position-Based Multicast (PBM). PBM is designed to divides destinations to the appropriate set of neighbours and the packets are forwarded by the available nodes closes to the destination within the neighbours. Packets progress and the number of neighbours are considered as performance matrix.

Recently GEograhic Multicast protocol scheme (GEM) was proposed by [19] that do not require any exchange location information for routing purposes. This is achieved by setting competition rule by the relay node for packet forwarding, to be competed by the available nodes within the set relay radios. The nodes with highest probability of efficiency are selected using the Euclidean Steiner Tree (EST) theory. GEM breaks the operation requires by each hop into two phases. Phase 1 focus and decides where the packet should be forwarded which is to be identified by the current relay node with not having more than two directions as it is known in the EST literature that more than two directions doesn't provide any performance advantage it would only result in the increase of complexity. Phase 2, the relay node initiates the competition among the nodes within the radios area coverage by broadcast a message informing its current position and direction identified in phase 1. The receiving nodes estimate the progress matrix that qualifies how better it is in forwarding the packet along the identified direction and candidates itself as the next relay. The current relay node collects all the candidature and selects the best or bests in case two directions was selected among the nodes. Constant interaction is required in this phase unlike in phase 1.

The proposed GEM protocol scheme has shown efficient improvement. However, the assumption made in the scheme that the network has a large number of nodes, topology changes occur frequently and the traffic load is low is not always feasible. Whenever there is large number of nodes with frequent changes in location, traffic load should also be expected to be high with the constant interaction of the relay node and other nodes in phase 2. Therefore, the problem of multipoint routing is not yet been resolved almost all the solutions provided have their set back as well.

Another geographical unicast routing algorithm using no location service called Simile Wide-deploy Algorithm for ad hoc Networks (SWAN) was proposed in [23]. As described in their paper, the main contributions of the new algorithm are; low overhead, high delivery ratio, absence of a route repair process, self-adaptive, sending messages using an optimal route and its capability of sending messages in a network with voids or fragmented using angle correction and the proxy state. The paper also acknowledge contribution of some related work in the area, but observed that in those related works, the nodes retransmit control messages to all the other nodes (broadcast) makes it more expensive in terms of resources and scalability. Hence, the use of unicast that does not retransmit control messages to all nodes in the network will maximise the routing performance.

The proposed SWAN routing algorithm was designed to overcome some of the open issues faced by the previous works, particularly the use of broadcast, strict path definition, maintaining destination coordinates at the sender, ensuring forwarding when closeness is not the best and the problem of no connectivity for some nodes close to the destination.

Adaptive Hybrid Transmission (AHT) scheme for on demand mobile Internet protocol television (IPTV) based on IEEE 802.11 standard over a wireless mesh network is proposed by [24]. The algorithm is to enhance service blocking and reduces the overall bandwidth consumption in wireless system due to its limited resources compared to wired network. The hybrid mechanism uses the combination of multichannel multicasting and unicast scheme to serve requesting clients. From the network perspective, a network works effectively if it has enough capacity and/or the request rate from the users is relatively low. But if there are famous sports or popular movies, the requests are highly skewed and the large number of unicast for the same content would be established and transmitted over the network, which results in the huge inefficiency of both media server and bandwidth consumption of wireless system. To overcome this problem they suggested AHT mechanism. In this mechanism the most popular video is transmitted by multicast transmission and normal video is transmitted by unicast transmission.

A Vertical WLAN Handover Algorithm and Protocol was proposed by[25] to improve the IPTV QoE at the end user. The algorithm considered the used of Handover vertical techniques that allow mobile nodes to change between access points of different wireless technologies. The algorithm allows the mobile nodes to roam from one network to another. In their results, the proposed algorithm shows high packets lost and jitter during the roaming process. Also a quality of service negotiation mechanism for IPTV in heterogeneous networks was proposed by[26] . The proposed QoS mechanism regulates the quality level of the IPTV stream based on the network bandwidth and the user's device type. However, the mechanism doesn't include mobile networks.

All the proposed algorithms discussed above did contributed in one way or the other in increasing effectiveness in data transmission in mobile ad hoc networks. However, issues on reduction of server over load and bandwidth have not been tackled. Therefore, there is need for an algorithm that will not only increase the effectiveness of data communication within the client nodes but also enhance the efficiency of server node by reducing the server bandwidth consumption and workload. To deliver a successful IPTV services with the essential QoS and QoE, the server should have the required capacity and resources. During high demand, the numerous requests from clients may overload the source node, which leads to poor quality of service. To alleviate this problem, there should be a system whereby the server can take advantage of Mobile Ad hoc Network (MANET) routing protocol procedure to utilize the mobile client's resources.

2.4 Mobile ad hoc networks

A Mobile Ad hoc Network (MANET) is an autonomous system of mobile devices that are connected via wireless links without prior planning or need of any existing network infrastructure. The mobile nodes are communicating with each other without centralized control system[3]. Mobile devices are communicating to each other not only as source or destination but also as routers in wireless network. Routing and packet forwarding in MANETs are among the most active research topics in the area of wireless communication as can be seen in [23][27][28][29].

The mobile nodes are randomly moving freely and re-organize themselves arbitrary. Hence the network topology is rapidly changes and unpredictable. Ordinarily, the mobile nodes operates in similar ways like routers in wired networks where packets are being forwarded to the nearest neighbouring nodes towards the destination node [29][30]. The nodes that are within the radio range of each other communicate directly, while the remote nodes depend on neighbouring nodes for packets forwarding.

Routing is one of the major issues regarding mobile ad hoc networks, due to rapid topology changes which affect the general performance of routing protocols in wireless network compared with wired networks. Other challenges facing MANETs include dynamic topology, frequent network update, speed and security. There are several proposed MANETs Protocols, but the commonly used include Dynamic Source Routing (DSR), Ad hoc On-Demand Vector (AODV), Optimized Link State Routing (OLSR) and Dynamic Destination Sequenced Distance Vector

Routing (DSDV) as well as Zone routing protocol (ZRP) and Temporally-Ordered Routing Algorithm (TORA) [2][31].

Ad hoc routing scheme can be classified into two categories, topology based and position-based routing. However, topology-based further subdivided into proactive, reactive and hybrid approaches[28]. Another study in [31] described for mobile ad hoc networks protocols as proactive, reactive and hybrid.

2.4.1 Proactive/Table Driven Routing Protocol

Proactive routing protocols such as DSDV [32] and OLSR [33] maintain individual routing table/information about the available paths in the network even if the paths are not currently in use [31][28]. Each node maintains consistent and up-to-date routing information by sending a control message (HELLO messages) between nodes periodically for the nodes to update their routing tables. Maintaining information about the available routing path in the network can be greatly regarded as useful in the designing my new algorithm even though certain amount of bandwidth will be used in keeping track of this information.

2.4.2 Reactive/On-Demand Routing Protocol

Reactive routing protocols includes DSR [34] and AODV [35], maintains only the routes that are currently in use [28]. In reactive or on demand routing protocol the route discovery mechanism is stated by the source in finding the route to the destination and it remain valid till destination is unreachable or until the route is no longer needed [31]. Unlike the proactive/table driven protocols, all nodes needs not to maintain and up-date their routing information. Figure 2.9 shows the AODV route identification process.

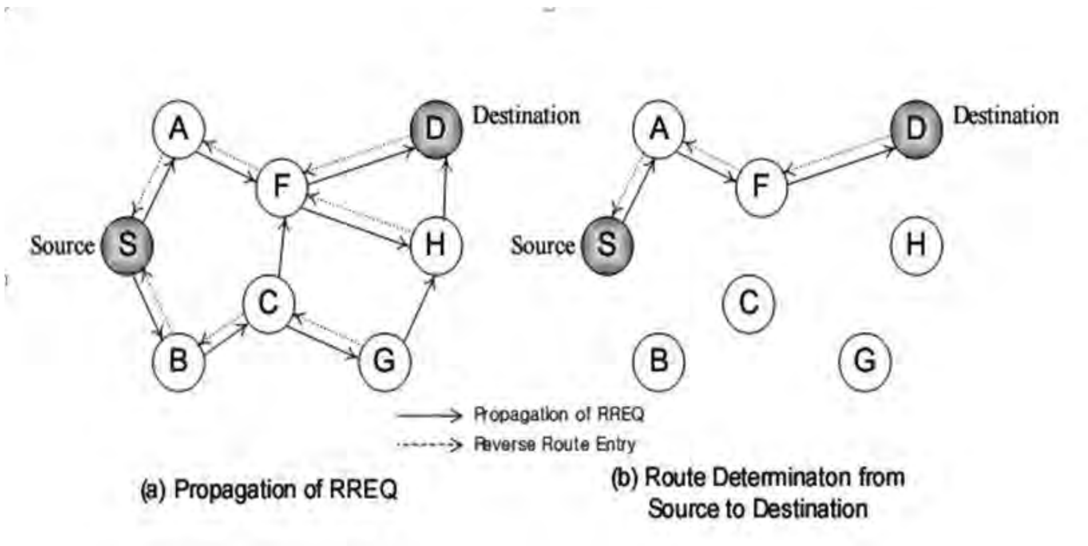


Figure 2.9 Example of AODV Route Identification Process

2.4.3 Hybrid routing Protocol

Hybrid routing protocols which includes ZRP [36] and TORA [37] combines the advantages of both proactive and reactive routing protocols. The routing is initially established with some proactively prospected routes and then serves the demand from additionally activated nodes through reactive flooding.

Performance comparison study of DSR and AODV conducted by[38]. The results of the performance metrics show that AODV outperformed DSR on throughput, delay, network overhead and packet delivery ratio. It also shows that it consumes less energy than DSR. In a similar study conducted by[39], shows that as number of nodes increases the performance of DSR decreases and the performance of AODV increases. Therefore, they concluded that AODV perform better than DSR on throughput, end-to-end delay and number of packet dropped on the high density Network. In another study presented by [40], focuses on the quality of service such as average time delay and the routing load overhead. The study shows that DSR outperforms AODV in less density network and AODV performed much better in a high density networks.

The reactive routing protocol has advantage on saving the network bandwidth and reduces unnecessary network congestion as the source node does the route discovery and maintenance. Other advantage of this protocol is the way of maintaining only those routes that are available. Hence, based on these advantages I have considered the reactive/on-demand routing algorithm in the design of the proposed algorithms.

In this research, Internet Protocol (IP) and AODV routing processes are considered and adopted. The new proposed algorithm adopted the AODV routing mechanism where the source node finds and maintains route to the destination node. Maintaining information about the available routing path by source node can be greatly regarded as useful in the designing of the proposed algorithm as less amount of bandwidth is used in keeping track of the routing information by the nodes only when need. Other parameters used in the designing of the proposed algorithm are explained in various stages of the design in each chapter.

Chapter 3: Methodology

3.1 Introduction

To evaluate the effectiveness of the proposed algorithm, IPTV network has to be implemented. Deploying IPTV network is not only time consuming, expensive and require different technical skills but also involve government policies and approval. Erecting radio mass transmitters, cabling and data centres may cause environmental problem. To deploy such infrastructure even if all the resources are available, it requires government approval and licence, which may take longer time due to government policies and bureaucracy.

Therefore, the method that has been used in this research work to analyse and evaluate the effectiveness of the proposed algorithm is the combination of statistical analysis and simulation model analysis. Network simulation provides solution to model network behaviours by collecting and calculating communications statistical data between modelling devices. The common method used in large-scale simulation studies is called Discrete Event Simulation (DES). DES enables modelling with more reality and accuracy as well as broad applicability [41]. DES creates a tremendous and comprehensive, packet-by-packet model for the performance of network to be predicted. However, it often requires a significant computational power, especially for very large-scale simulation, the procedure can be time consuming. It may take hours or even days to complete. However, simulation provides accurate solutions for either a node-to-node queuing system or a network of queues, from simple algorithm to complex protocol.

Mathematical analysis and modelling can provide fast understanding and solutions to some of the problems under study. It is generally quicker than simulation, but in many cases is erroneous or inapplicable. For many situations analytical models are not available, and even if it is available many of these models are modelled using approximations, which yields lack of accuracy. For example for a network of queues to be to be analytically model, it can either be decomposed via the Kleinrock independence assumption or be solved using a hop-by-hop single system analysis [42][43], both of which lose the expected accuracy. As the networking protocol slightly become more complex, the difficulties and accuracy of the mathematical modelling will greatly exacerbated.

The combination of mathematical/statistical analysis and simulation method provides accessibility to the advantages of these methods and overcomes their disadvantages. The mathematical analysis provides faster solution and less computational power while the DES provides accuracy, the combination of this method is called hybrid simulation [44].

3.2 Statistical Analysis

Statistical analysis can simply be defined as data analytics component. It is a science of examining raw data with the purpose of drawing conclusion. In other words, "Statistics is the branch of scientific method which deals with the data obtained by counting or measuring the properties of populations of natural phenomena. In this definition 'natural phenomena' includes all the happenings of the external world, whether human or not.” [45]

These definitions indicate that statistical analysis has changed from being grounded firmly in the world of measurement and scientific analysis into the world of exploration, comprehension and decision-making [46]. The use of statistical analysis has grown immensely from a comparatively small set of specific application areas such as experimental design to almost every part of life. Understanding information and making well-informed decisions on the basis of such understanding, is the primary function of modern statistical methods.

3.3 Network Simulation

Network simulations can be regarded as time-clocked simulation or discrete event simulation.

- The time-clocked simulation is whereby the simulation progresses through the iterative progressing of time slots. Events within the iterated time slot are executed while simulation is progressing. Figure 3.1 below shows the flowchart of time-clocked simulation.
- The discrete event simulation is whereby the simulation progresses through the execution of the scheduled next event. Simulation time is updated after the next scheduled event is executed. Figure 3.2 below shows the flowchart of DES simulation.

The difference between time slot in time-based simulation and the simulation time is that, the time slot in time-based simulation refers to the real world clock time. While the simulation time refers to the time used in running the model. Hence these time slots are two different concepts.

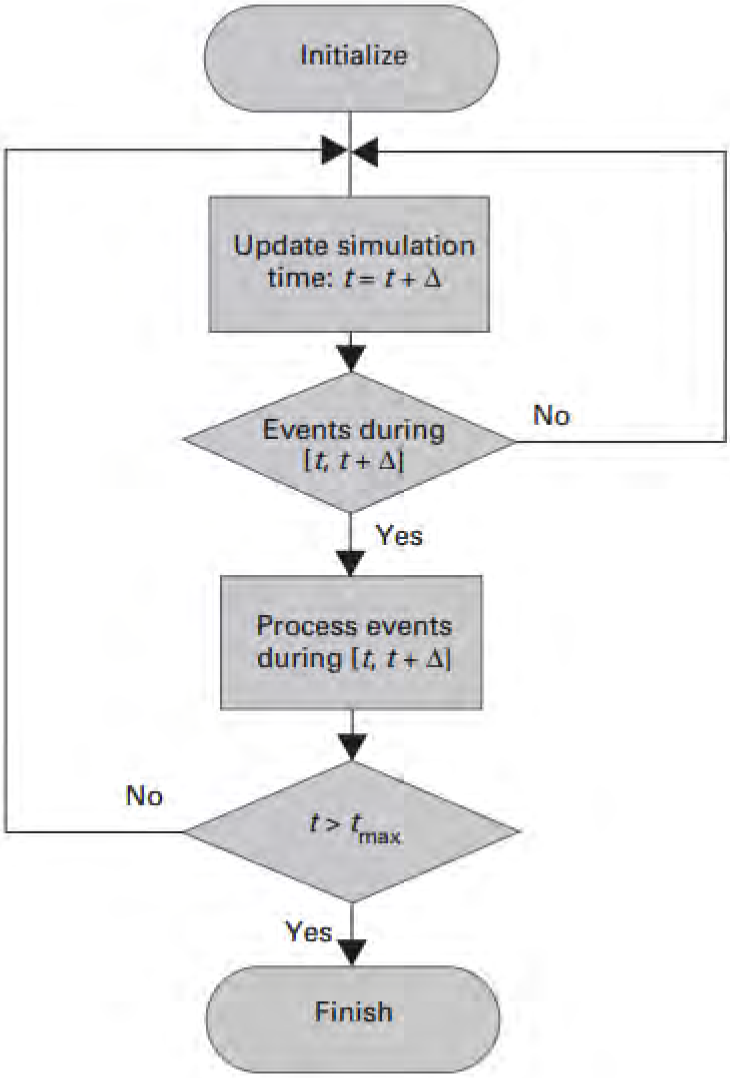


Figure 3.1 Time-based simulation [42]

The time-clocked simulation usually iterates through all time slots regardless of whether there are events or not within a particular time slot. Therefore, it is in efficient to use time-clocked simulation in a burst-like system with long silent periods, when there are no events in many continuous slots, since it will still iterate all those time slots without events being processed.

The discrete event simulation iterates only through the scheduled events that must be processed in a regimented style, this overcome the shortcomings of time-clocked simulation. For the above reason, most modern simulators support the approach of discrete event simulation.

Basic discrete event simulator framework should have the following elements to be able to execute DES [42], this include:

- Random number generators - to represent different random variables such as packet size, packet inter-arrival times, noise, system processing time etc., as system inputs
- Simulation time - which can be updated to allow simulation to progress
- Prioritized event lists - to list events to be executed one by one
- Simulation finishing conditions – which include simulation duration and some other customized termination conditions

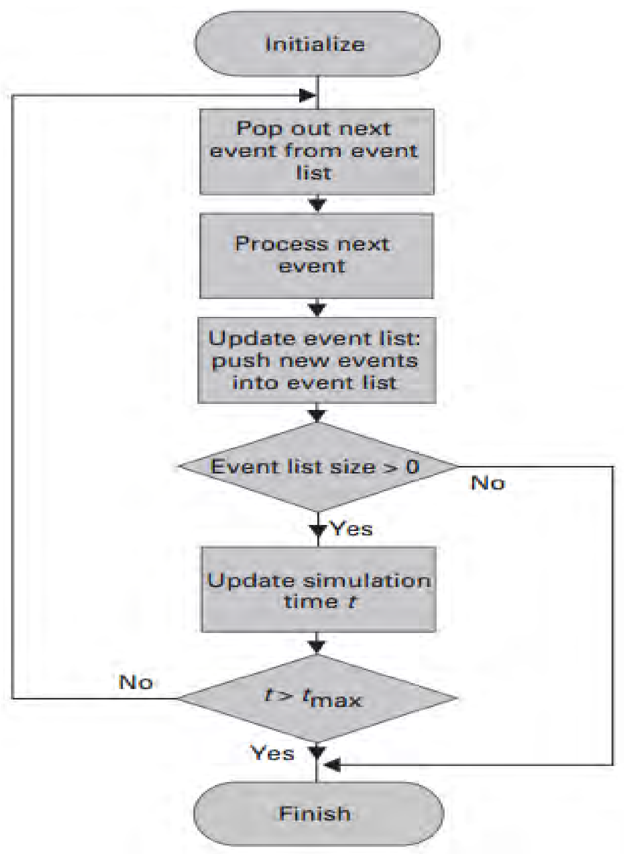


Figure 3.2 Event based simulation [42]

The basic DES simulator structure has three stages which include initialization, simulation and clean-up stages [42]. The DES pseudo-code structure is shown in Algorithm 4.1 below. The initialization stage is the stage where all state variables are populated with initial values, such as simulation time, event list, statistics, and memories.

```
void main()
{
    //~initialization
    initialize_variables ();
    alocate_memories ();
    ...
    //~simulation kernel operations
    while(simulation_time < finish_time)
    {
        current_event = pop_next_event_from_list ();
        process_event(current_event);
        update_simulation_time ();
        ...
    }
    //~finishing up
    write_records_to_file ();
    free_memories ();
    ...
}
```

In the simulation kernel stage, main loop is used to run the simulation until a simulation termination condition is satisfied such as simulation finish time. During this stage scheduled events are processed one after the other as long as the termination condition is not satisfied. The event processing may include formula calculation, collecting statistics, moving events to event list, cleaning up invalid variables and free memories, creating new variables and memories, etc. The simulation time is updated whenever an event is processed and the updated value is calculated based on the particular event being executed. This process continues until termination conditions are satisfied.

As soon as the simulation stage is over, the clean-up activity will begin before simulation ends; these activities include writing records into files, freeing memories, and so on.

There are many network simulators such as OPNET, NS and OMNeT++ which are accepted and widely used. Among these simulators, OPNET is chosen due to its competency in simulating both explicit DES and hybrid simulation modes. Also, many researches such as [47][48][49][50][51], were conducted using OPNET modeller. Similarly, other simulation features like co-simulation, parallel simulation, high-level architecture, and system-in-the-loop interactive simulations are all supported by OPNET.

3.4 The OPNET

Optimized Network Engineering Tools (OPNET) was created by OPNET Technologies, Inc., which was founded in 1986. OPNET is network simulation software that provides solutions in the aspects of communications networks [52] and it covers the following areas:-

- Application performance management
- Network Planning
- Network Engineering
- Network Mapping
- Network Optimization
- Operations
- Research and development

The OPNET products for application performance management include ACE Analyst, ACE Live, OPNET Panorama, and IT Guru Systems Planner. These products are for analytics networked applications, end-user experience monitoring & real-time network analytics, real-time application monitoring & analytics, and systems capacity management for enterprises respectively.

OPNET products for network planning, engineering, and operations include IT/SP Guru Network Planner for network planning and engineering for enterprises and service providers, SP Guru Transport Planner for transport network planning and engineering, NetMapper for automated up-to-date network diagramming, IT/SP Sentinel for network audit, security and policy-compliance for enterprises and service providers, SP Sentinel for network audit, security and policy-compliance for service providers, and OPNET nCompass for providing a unified, graphical visualization of large, heterogeneous production networks for enterprises and service providers.

OPNET products for network research and development include OPNET Modeler, OPNET Modeler Wireless Suite, and OPNET Modeler Wireless Suite for Defence.

For the purpose of this research, the research program product (OPNET modeller) will be used.

3.5 OPNET Modeler

OPNET Modeler is a discrete event simulator that is widely used by researchers, engineers, university students, and the United State department of defence. OPNET Modeler has a user-friendly graphic user interface (GUI), supported by object-oriented and hierarchical modelling, debugging, and analysis. It supports hybrid simulation, analytical simulation, 32-bits and 64-bits fully parallel simulation and providing many other features [42].

OPNET Modeler has grid-computing support for distributed simulation and Its System-in-the-Loop interface allows simulation with external computer systems, which provides real-world data into the simulation environment. The interface for integrating with external objects files, libraries, and its in-build wider collection of numerous type of protocols and hundreds of different vendor computer and communication devices model with source code as well as inclusion of development environment to enable users to model different types of network devices and protocols of their choice accelerated the research and development process for designing and analysing of different communication networks, devices, protocols, and applications [52].

OPNET Modeler provides a comprehensive development environment to be used as a platform to develop models of a wide range of systems applicable to local area network (LAN), wide area network (WAN), metropolitan area network (MAN), performance modelling, hierarchical Internetwork planning, research and development of protocols and communication network architecture and devices, mobile network, sensor network and satellite network etc.

However, developing a new model and simulation study using the standard model incorporated in the Modeler is easy and straight-forward, but developing a new or modifying existing model, devices or protocol could become a nightmare.

OPNET modeler is designed and structured in hierarchical method which consists of three hierarchical layers; Network, Node and Process levels as show in Figure 3.3 below.

The Network Domain is the top-level view of the simulation study where the network topology and attributes are specified. The network model consists of nodes, links and subnets as well as propagation of network attributes values such as protocols, device configuration parameters, simulation statistics etc.

A node represents a network device or a group of devices which include router, workstation, servers switches, hubs etc., while a link represent point-to-point or bus links. Subnets are used for the organization of network components and represent the actual network constructs. The network devices (nodes and links) are designed in a form of icons to help user to easily locate and use the correct device. The devices are in form of vendor specific device and generic devices as shown in figure 3.4.

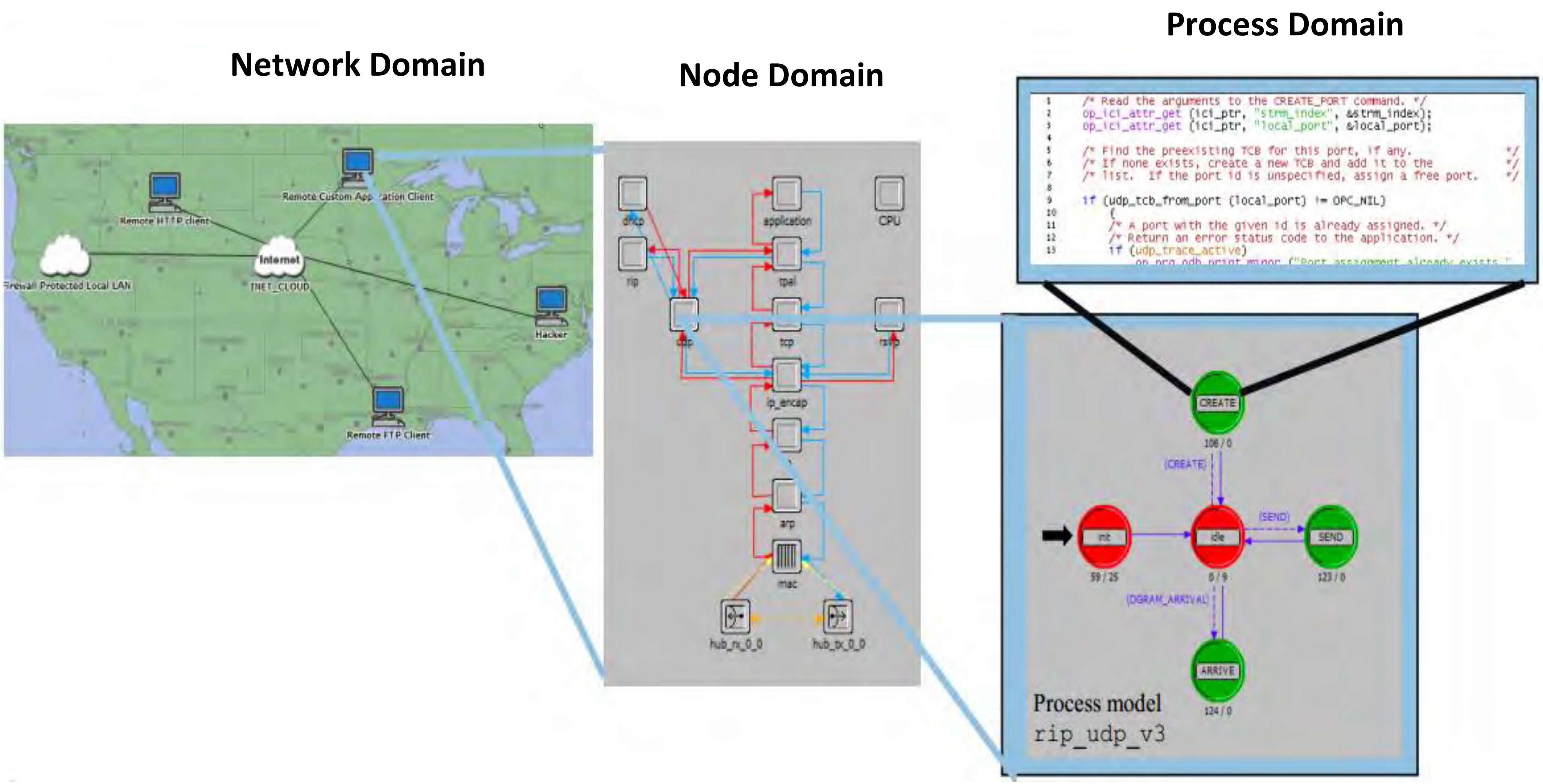


Figure 3.3 The Three Tiered OPNET Hierarchy

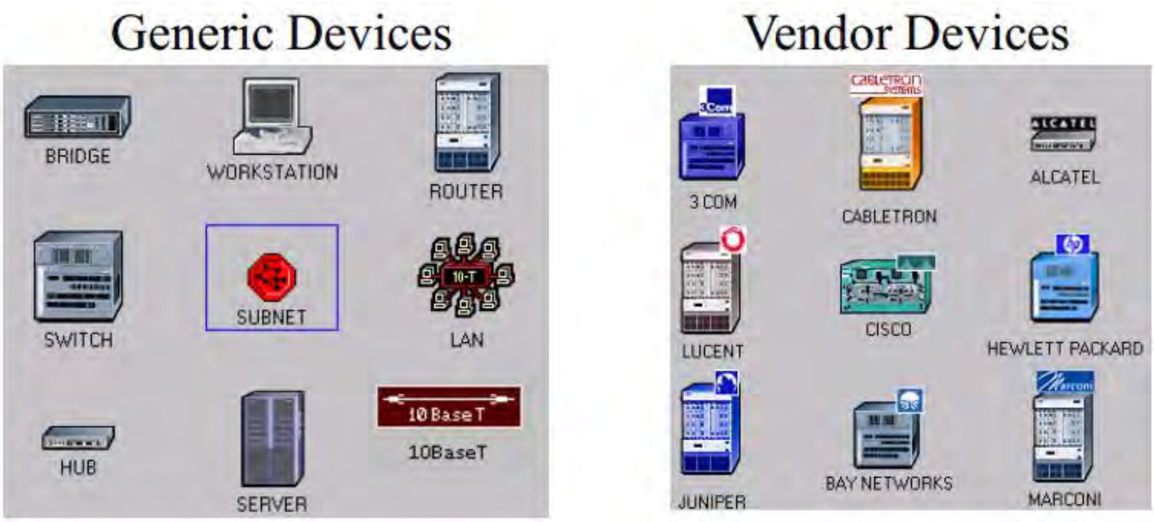
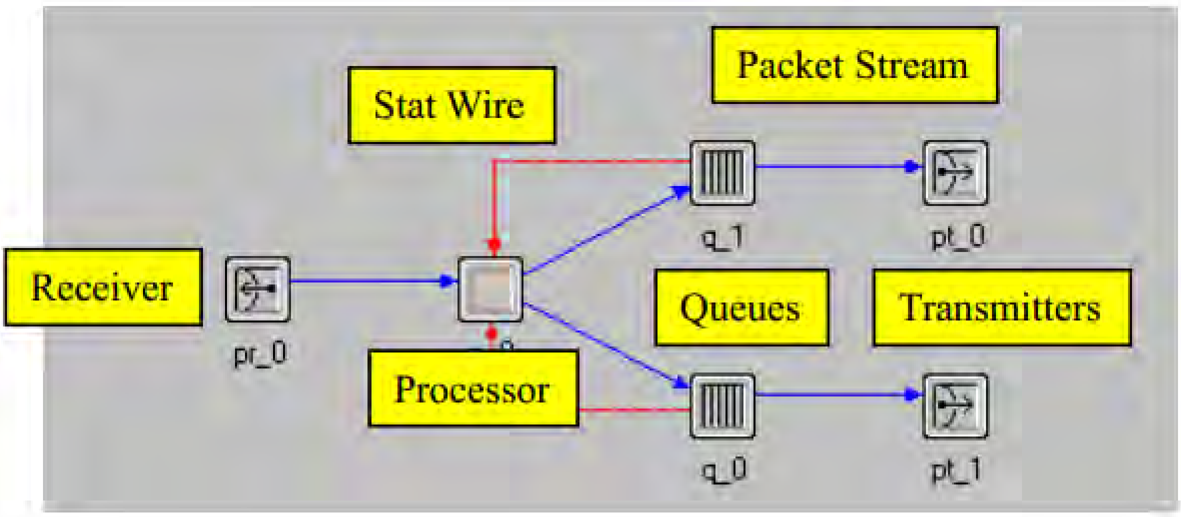


Figure 3.4 OPNET Icons

Node Domain is the basic building blocks (modules) which include processors, queues, and transceivers. The processors are fully programmable through their process model while the queues safeguard and manage data packets. The transceivers (transmitters and receivers) are node interfaces and the packet streams and statistic wires are the interfaces between modules. Figure 3.5 shows the node domain.

Figure 3.5 Node Domain



Process domain is the model operation of a particular networking process or technology. The process domain consists of state transition diagrams, blocks of C/C++ code, OPNET Kernel Procedures (KPs), state variables and temporary variables. Figure 3.6 shows the node domain of a single process in a router.

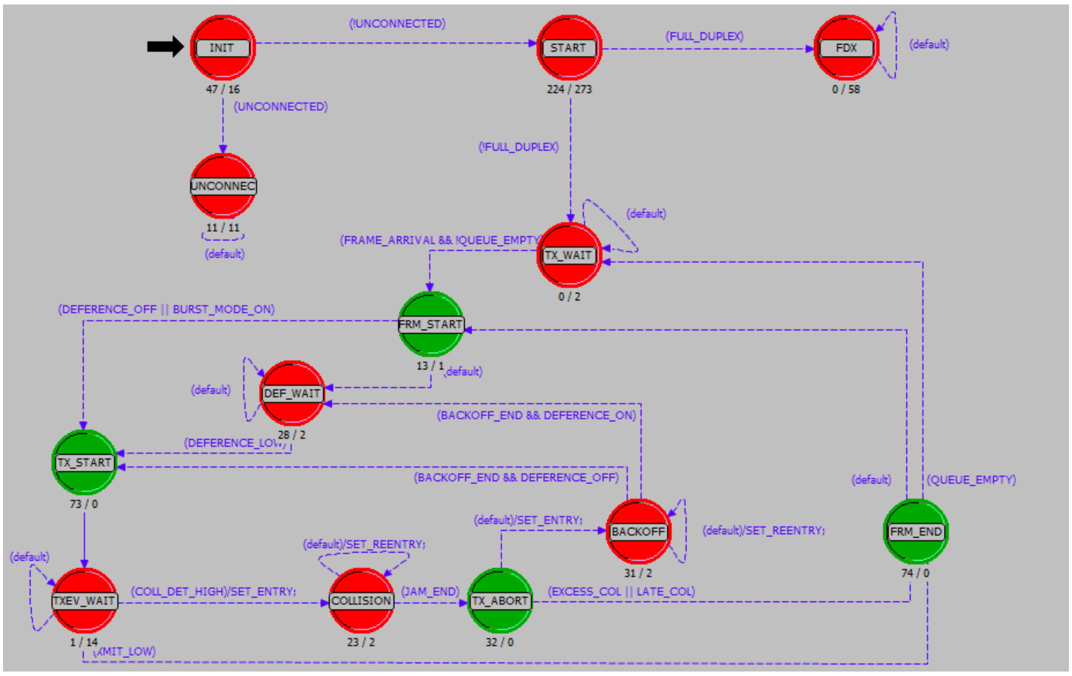


Figure 3.6 Process model

3.6 Integrating the Proposed Algorithms in OPNET Modeler

OPNET Modeler provides flexible and highly cohesive architecture which allows reusability and extensibility of existing models. The proposed algorithm is incorporated in the existing OPNET Ethernet server. The node model can supports the layering of protocol function, dynamic inter-module monitoring and arbitrary node architecture. Figure 3.7 and 3.8

shows the ethernet_server_adv node Ethernet_server_application_precess model respectively.

The ethernet_server_adv model represents a server node with server applications running over TCP/IP and UDP/IP that supports one Ethernet connection at 10 Mbps, 100 Mbps, or 1 Gbps. The operational speed is determined by the connected link's data rate. Packets are routed on a first come-first serve (FCFS) basis and may encounter queuing at the lower protocol layers, depending on the transmission rates of the corresponding output interface. The supported protocols by this node include; RIP, UDP, IP, TCP, Ethernet, Fast Ethernet and Gigabit Ethernet.

The proposed algorithms are incorporated in the video_streaming_server_mgr process in OPNET environment. This server process is responsible of sending video streaming packets to the video streaming clients. It is a child process of client server process manager (clsvr_mgr). The server process manager is responsible of serving all requests of all the application services supported by the server such as e-mail, web browsing, printer, data base, file transfer etc. Hence, clsvr_mgr process calls the services of video_streaming_server_mgr process whenever a video streaming request is received. The Algorithm's code was written, compiled and saved in a different OPNET video streaming server.

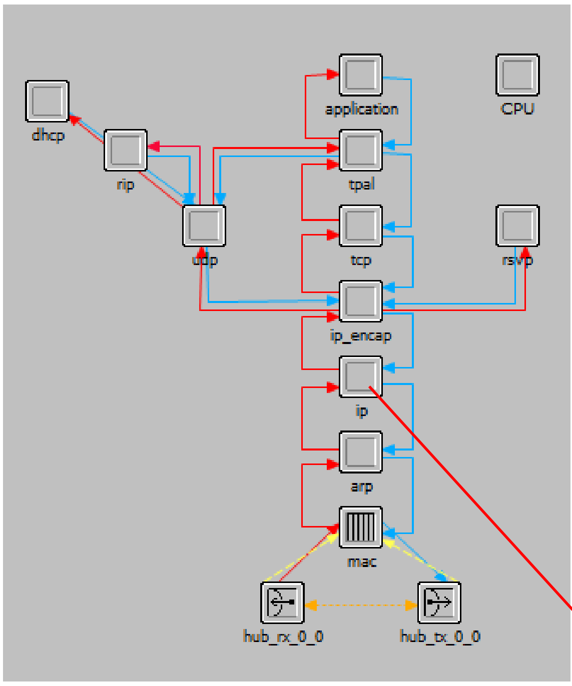


Figure 3.7 Ethernet_server_node model

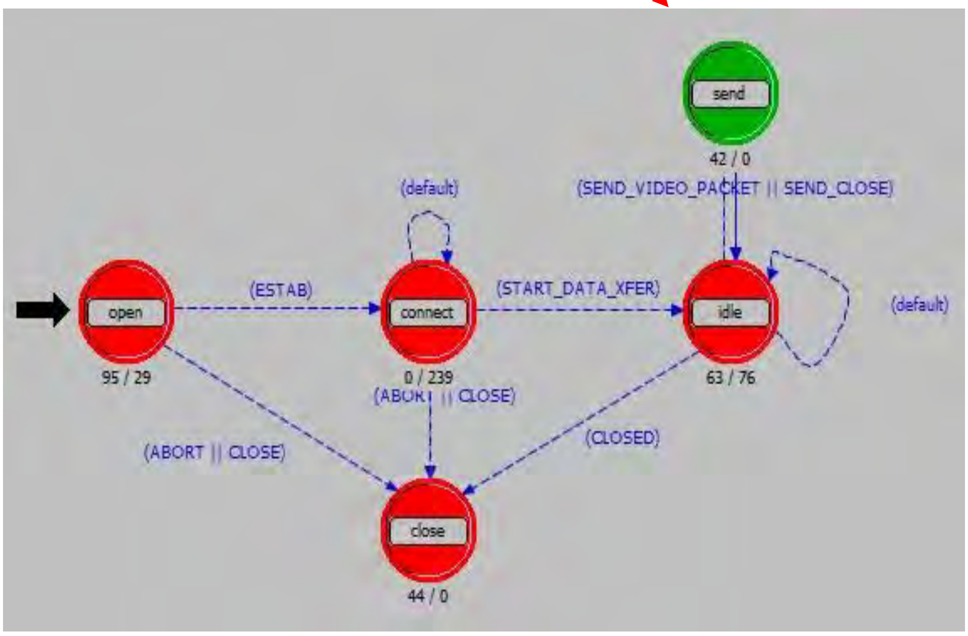


Figure 3.8 Ethernet_server_application_precess model

OPNET modeler supports real-world data in to its simulation environment so as the simulation can be as closer to the real application as possible. Video streaming traffic (data) can be captured using video streaming software such as VLC and the data are saved in a file, which is use in OPNET Modeler to simulate the same traffic [44]. In this research work, a live video programme (winter Olympics 2014) was streamed using BBC iPlayer [53] and the date were captured using VLC software. The captured data were saved in a file and the file was stored in the OPNET primary directory (that is, the first directory listed in OPNET) in the model directories and used to simulate all the scenarios.

OPNET uses C programming language to create, send and destroy packets from the video streaming server to the client nodes. The proposed algorithm was also coded and incorporated in to the video_streaming_server_mgr process. Figure 3.9 shows some portion of the code.

```
/* This state is responsible for sending video frames for   */
/* (LIVE only) video streaming objects that are within a    */
/* page in an http application.                              */

/* Get the key that will be used in the dispatch table to   */
/* find the record that corresponds to the video streaming  */
/* attributes of the video object in the http page.          */
http_video_key = (OmsT_Dt_Key) (intrpt_code - GNAC_SEND_VIDEO_FRAME_OFFSET);

/* Get the record based on the interrupt code from the      */
/* dispatch table.                                           */
http_video_record_ptr = (GnaT_Http_Video_Record *) oms_dt_item_info_get (http_video_record_handle, http_video_key);

/* Calculate the outgoing packet size based on the frame    */
/* size distribution handle that can be found on the record.*/
pk_size = 8 * ceil (tpal_dist_positive_outcome (http_video_record_ptr->frame_size_dist_handle, http_video_record_ptr->sess_ptr->app_name,
                                                "Http.Page Properties.Streamed Video Properties.Frame Size (bytes)"));

/* Determime the creating time of the next frame.           */
next_video_frame_time = current_sim_time + tpal_dist_positive_outcome (http_video_record_ptr->frame_inter_arrival_time_dist_handle,
        http_video_record_ptr->sess_ptr->app_name, "Http.Page Properties.Streamed Video Properties.Frame Inter-arrival Time (seconds)");

/* Just a sanity check                                       */
if (next_video_frame_time < current_sim_time)
    next_video_frame_time = current_sim_time;

/* Install the session ICI                                       */
op_ici_install (http_video_record_ptr->sess_ptr->ici_ptr);

/* Start from the original packet that already contains useful  */
/* fields there like http obj name etc (or just use that one    */
/* if this is the last transmission).                            */
if ((next_video_frame_time - http_video_record_ptr->start_time) > http_video_record_ptr->video_length)
    {
    pk_ptr = http_video_record_ptr->orig_pkt;

    /* Update the packet creation time to the current time,     */
    /* which is used for delay statistics at the receiving end. */
    op_pk_creation_time_set (pk_ptr, op_sim_time ());
    }
else
    pk_ptr = op_pk_copy (http_video_record_ptr->orig_pkt);

/* Set the packet size in bits                                   */
op_pk_total_size_set (pk_ptr, (OpT_Packet_Size) pk_size);

/* Set the service type of the outgoing packet to HTTP           */
op_pk_nfd_set (pk_ptr, "service type", GnaC_App_Type_Http);

/* Create a field in GNA packet containing video streaming      */
```

Figure 3.9 C code for the video function block

3.7 Details of the Simulation

The equipment used in simulating the proposed algorithm and the normal IPTV are explained in the following points:

a. Hardware: High speed windows 7 desktop computer was dedicated and used for the simulations throughout the stages of this research work
b. Software: OPENT/Riverbed was used to simulate the IPTV network in this study. Also VideoLAN (VLC) media player was used to capture all incoming video streaming traffic during 2014 winter Olympics games over the Internet. The data were incorporated in all the simulation scenarios for real validity
c. Connection used: Wi-Fi connection was used in capturing the video streaming traffic over the Internet
d. No of simulation run: Each scenario simulation was run for 5 to 7 times and the best result was chosen.
e. Simulation run time: The scenarios simulations were run for 1 hour however, on average it took 6 hours to complete. The last part of the algorithm was simulated for 30 minutes as the computer was hanging when it was simulated for 60 and 45 minutes. This is due to complicated tasks that were running in the final part of the algorithm

CHAPTER 4: Unicast Bandwidth Efficiency Routing Algorithm (UBERA)

4.1 Introduction

The main characteristic of MANETs as discussed in Chapter 2 is the absence of fixed structure for mobile devices communications. This characteristic provides some certain advantages in the deployment of MANET, as routing tasks are distributed over the network devices. Some of these advantages include; there is no investment cost on the infrastructure and the total dependency on the source node has been overcome. However, the frequent movements of mobile devices is a serious issue without a proper and well-designed routing algorithm that takes into account the devices mobility to enhance the end-to-end packet delivery with minimum jitter. The proposed algorithm uses the concepts of both Internet Protocol (IP) and MANET routing protocols to enhance the end-to-end delivery of IPTV services and reducing the total dependency on the server node, minimum end-to-end packet delay and ensuring end user quality of experience.

4.2 Unicast Bandwidth Efficiency Routing Algorithm (UBERA)

UBERA is a routing algorithm that effectively utilises network devices' bandwidth and resources in sending video streaming packets to the designated destination [54]. Rather than always depending on one source node to serve each service request received from client nodes, some of the requests are redirected to client nodes within the network and location of the requesting client that have already requested and received the packet(s) to forward it to the requesting node. Each client in the network belongs to a group and the grouping is done based on the common clients' watching behaviour and location. Stipulated time is set within which the packet is believed to be available in the client's storage, after which the source node deletes the information about that packet in its routing database. All the forwarding nodes in the network send routing information to the source node with the updates of their availability and location. In this research, 2 Km mobility radius within each group is used. Mobility radius is settled at 2 Km for MANET connection after several tests that were conducted using numbers between 1 and 5 Km. The results shows that 1 KM produced plane results while 3 to 5 KM produced high connection failure within the nodes. Also 5 minutes is set as the time expected for each packet to be available in the client device storage, without overloading it. The 5 minutes was selected after values from 1 minute to 10 minutes were tested, and the simulation results show no packets lost was encountered from 1 min to 5 mins due to client's storage overload. Whereas, packets lost were observed from 6 mins upward. Therefore, the time is settled at 5 mins. The time is set from the time the packet was sent from the server.

Thus

$$t_{exp} = 5 - t_{sent}$$

To predict the success of the proposed algorithm, it is imperative to measure the probability of successfully obtaining and forwarding available requested packet(s) from one or more IPTV client to the other. Based on consecutive steps described below for searching and forwarding packet(s) process, a binomial distribution theory is used to evaluate the probability of success,.

Where ***n*** is the total number of clients on the network and

k as the number of clients that have the packet(s).

p is the probability of success

q is the probability of failure

Assume that the probability that the packet is found and is successfully transmitted is; ***p*** = 0.3 and ***q*** = (1 – p).

Thus:

$$(p + q)^n = \sum_{k=0}^{n} \binom{n}{k} p^k q^{n-k}$$

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

In this research study, 20 mobile devices and 100 mobile devices were used in two separate projects to evaluate the effectiveness of the proposed algorithm.

Using the above formula:

In the first project, ***n*** = 20 and

k starts from 0 to ***n***,

the probability ***p*** that the packet(s) is found in one or more client, is calculated using the binomial distribution theory formula as:

$$(p + q)^{20} = \sum_{k=0..20}^{20} \binom{20}{0..20} p^k q^{20-k} \quad = 0.9997$$

In the second project, ***n*** = 100

k starts from 0 to ***n***

the probability ***p*** that that packet(s) is found on one or more clients is calculated as:

$$(p + q)^{20} = \sum_{k=0..20}^{20} \binom{20}{0..20} p^k q^{20-k} \quad = 1.000$$

Therefore, this shows that the probability that a packet(s) is found in one or more clients and transmitted successfully is very high in both projects, thus the server workload will be reduced by requesting other clients to forward the requested packets on its behalf. These results tally with the simulation results discussed in Section 4.4, where the server load is reduced in both projects considerably.

The algorithm pseudo code is presented in Algorithm 4.1 and operational step are explain below.

Input: Time; number of FREQ

Output: Serve Request

Procedure:

```
Request of X_{i...n} Packet(s)
If X_i is available in the array j[ ]
        if time <= 5 && FREQ = 0
                Send FREQ
        Else send Xi
        Update
Else check X_i in arrays j+1…n[ ]
End
```

Algorithm 4.1: UBERA Pseudo Code

Step 1: When a video streaming request is received by the server node from a client node, the source node first checks its routing information to identify any node(s) that already received the same requested packet(s) $X_{i..n}$ within the node group.

Step 2: If the client node that has the packet(s) $X_{i..n}$ is identified then the server checks the time the packet was sent

Step 3: If the time is within the set time, then it checks the number of Forward Request (***FREQ***) messages sent to that client.

Step 4: If there is no ***FREQ*** message sent to that client, then the server node sends a ***FREQ*** message to that client with the destination address in the message header for the packet to be forwarded to the requesting client and then update its routing table and end.

Step 5: If no client is found within the group of the requesting client or the set time elapsed or there is already a ***FREQ*** message sent to that client.

Step 6: The server checks the same packet in the other groups

Step 7: If the client node that has the packet(s) $X_{i..n}$ is identified in the other groups, then the server checks the time the packet was sent and number of ***FREQ*** messages.

Step 8: If the time is within the set time and there is no ***FREQ*** message sent to that client, then the server node sends a ***FREQ*** message to that client with the destination address in the message header for the packet to be forwarded to the requesting node, update its routing table and end.

Step 9: If no client node is found that has the packet or the stipulated time elapsed or there is a ***FREQ*** message sent to that client, then the source node has to serve the request directly by forwarding the requested packet(s) $X_{i..n}$ to the requesting node, updates its routing information and ends.

In all the steps described above, route reply messages (RREP) propagated back to the source node when the route path has been discovered. Also route error message (RERR) is propagated back to the source node to be notified on any broken link.

When the source node received many clients' requests at the same time, the requests should better be served using multicasting scheme. The proposed algorithm is limited to the use of unicast scheme at this stage. However, a multicast scheme is included in the next chapter.

The server node is also responsible for the nodes grouping and keeping up to date information in its routing table. The nodes are being grouped based on their similarities with other nodes in term of their viewing behaviour and are within the same geographical location. If a request was received from a particular client node for the first time, i.e. if the node recently joined the network, the source node will group that node in a group called a 'temporary group', until enough information have been gathered to be placed in a group that is appropriate.

The grouping is a measure to reduce the amount of time for the server to search all the client nodes in the network for a particular packet(s). It is easy and faster to search and find packet(s) on action movies within clients that are watching action movies than those that are watching comedies.

Configuration parameter	Value
Wireless technology	IEEE 802.11n
Mobility	Random waypoint
Packet Reception Power Threshold	-95dB
Data rate	6.5 Mbps (base) / 60 Mbps (max)
Start of data transmission	normal (100, 5) seconds
End of data transmission	End of simulation
Duration of Simulation	3600 seconds (1hr)
Packet inter-arrival time	Based on the captured data
Packet size exponential	Based on the captured data

TABLE 4.1: SUMMARY OF NODE CONFIGURATION

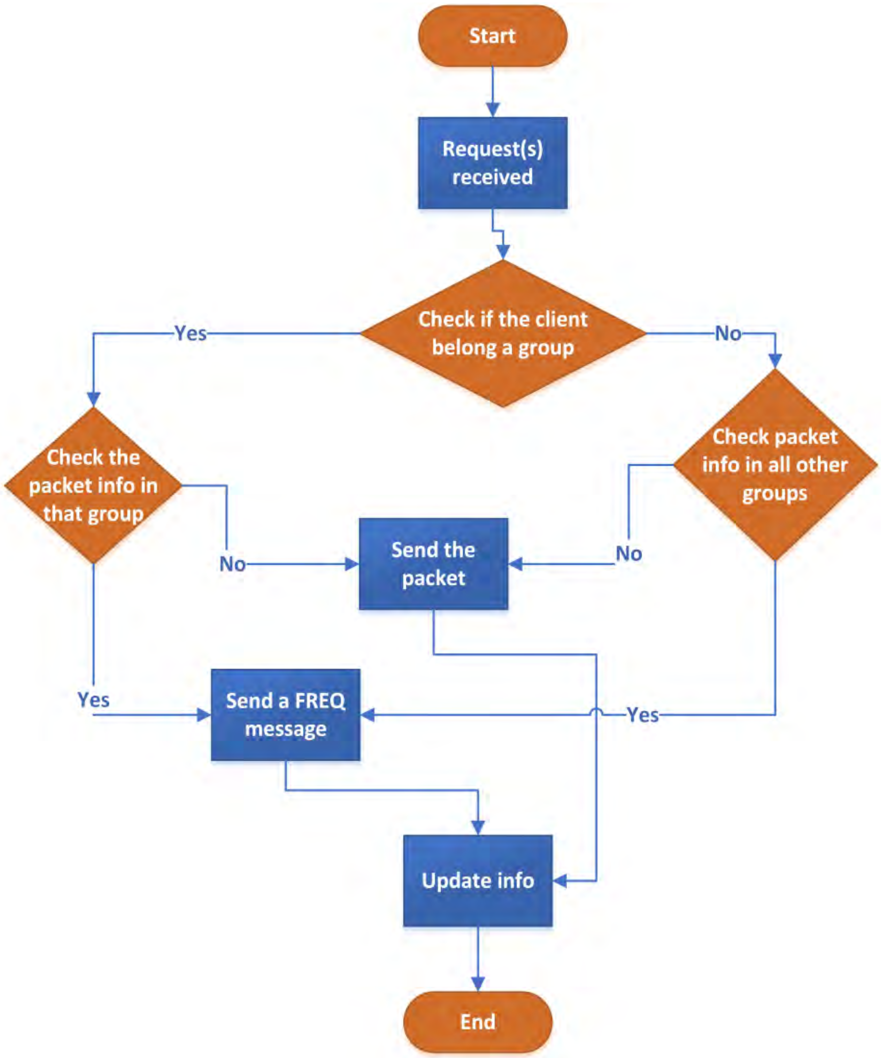


Figure 4.1 Unicast Bandwidth Efficiency Routing Algorithm Flowchart

4.3 Description of Projects and Scenarios

The simulation projects are designed and modelled to represents the typical IPTV network, whereby IPTV clients are streaming video contents from the video-streaming server over the Internet. The client nodes are inside the subnet and the server is remotely connected over the Internet. Figure 4.6 illustrated the network model. Two projects were created with two scenarios in each project. In the first project called UBERA Project 1, is modelled with 20 mobile phones in two groups, with 10 in each group. The clients are streaming video content from the video streaming server over the Internet as shown in Figure 4.7. The second project is called UBERA Project 2, is modelled with 100 mobile phones in ten groups of ten mobile phones each. The clients are streaming video contents from the video streaming server over the Internet as shown in Figure 4.8.

The numbers 20 and 100 are arrived after several simulations were conducted using the number of clients from 10 150. The best results that represent small and large number of clients are 20 and 100. The projects were created with a small number and large number of devices to test, analyse and evaluate the efficiency of the proposed algorithm as the number of client's increases. The projects and scenarios are explained as follows:

4.3.1. UBERA Project 1

In this project 20 mobile phones were used in modelling IPTV network. Two groups were created with 10 mobile phones each. The clients are streaming video content from a video-streaming server over the Internet. The project was created with few number of devices to evaluate the efficiency of the proposed algorithm in a low service demand situation. Scenarios created in this project are:

Scenario 1: 20 mobile phones streaming video packets from video streaming server over the Internet **with the normal IPTV**.

Scenario 2: 20 mobile phones streaming video packets from video streaming server over the Internet with the **proposed algorithm (UBERA)**.

4.3.2 UBERA Project 2

This project is modelled with 100 mobile phones in 10 groups of 10 devices each. The clients are streaming video content from a video-streaming server over the Internet. This project was created with large number of clients to evaluate the efficacy of the proposed algorithm as the number of client's increases. The scenarios created in this project are:

Scenario 3: Consist of 100 mobile phones in 10 groups, each group consist of 10 mobile devices. The devices are streaming video packets from video streaming server over the Internet using **the normal IPTV**

Scenario 4: 100 mobile phones in 10 groups of 10 mobile phones each, streaming video packets from normal video streaming server over the Internet **using Proposed algorithm (UBERA).**

All the four scenarios were simulated for one hour each. The designing of the project with different scenarios and different number of nodes is to evaluate the performance of the algorithm in high and low service demand. Comparison was made between the scenarios with the proposed algorithm and the scenarios with normal IPTV to evaluate the level of its effectiveness. The node configurations are presented in Table 4.1

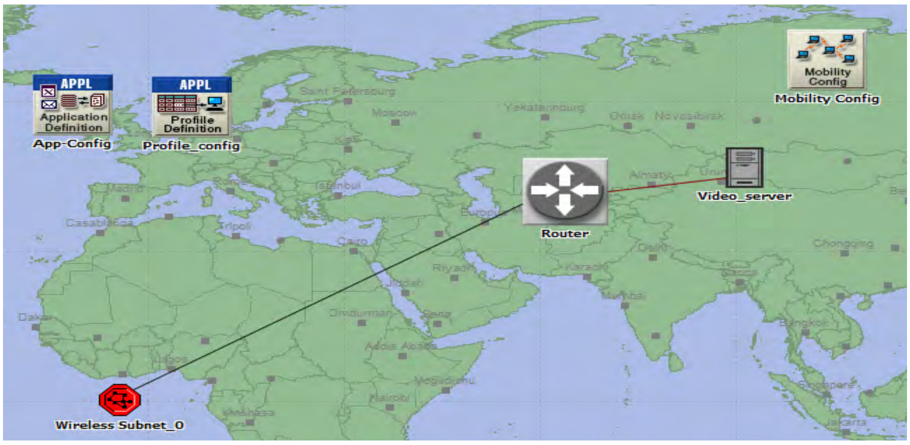


Figure 4.2 Network Model

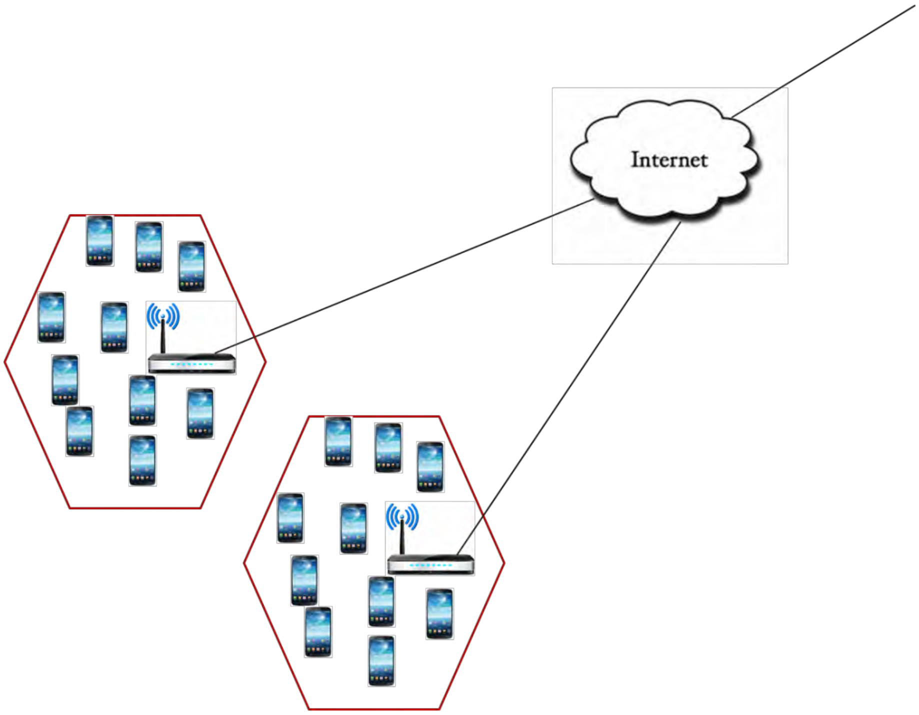


Figure 4.3: Two groups of ten mobile phones in the subnet

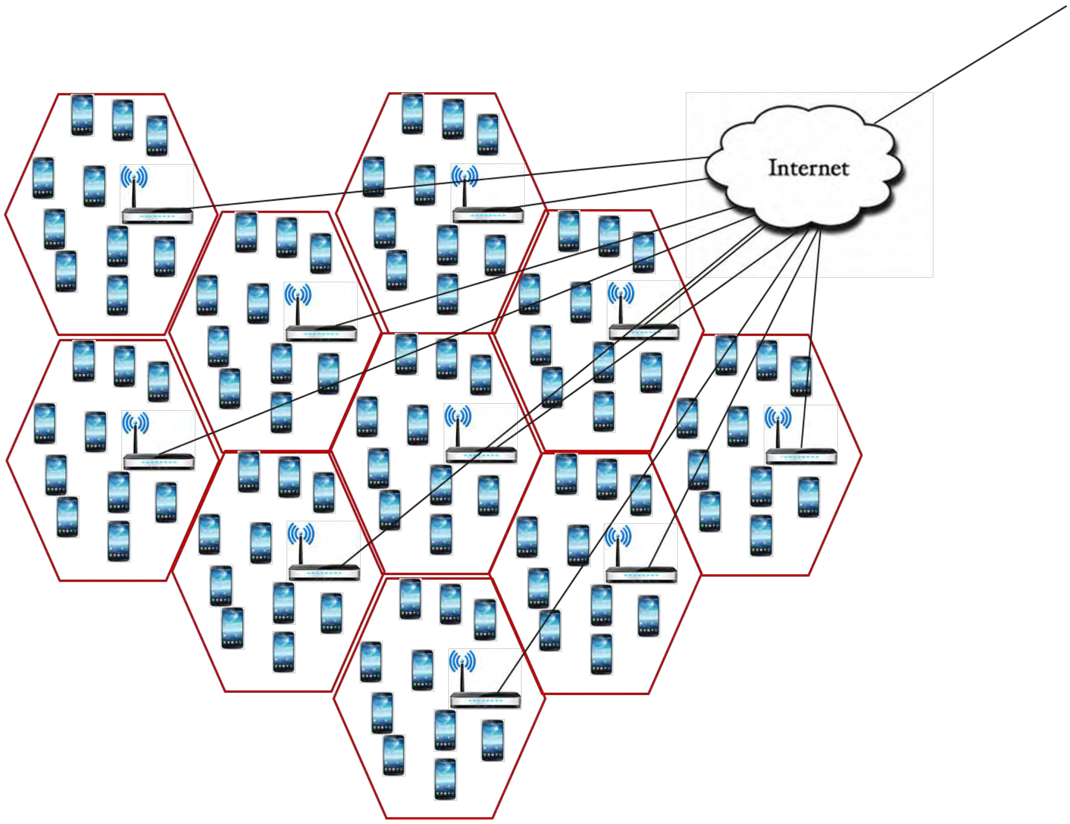


Figure 4.4 Ten groups of ten mobile phones in the subnet

4.4 Results Analyses and Evaluations

The quality of service parameters considered in this chapter include average wireless LAN delay, average delay veriations, average video streaming packets sent by the server and average server load. These parameters were compared and analysed in both projects. As defined by [44] wireless LAN delay is the end-to-end delay for all the packets received by the wireless LAN Medium Access Control (MAC) layers of all wireless LAN nodes in the network and forwarded it to higher layer. This delay includes the medium access delay at the source MAC. Packet delay veriation (jitter) is the variance among packets end-to-end delays for all the video streaming packets from the source node to the destination node. Server load represents the total load in bits/sec at the source node. Average video streaming packets is the average number of packets per/sec sent to the transport layer by the source node to all the video streaming applications in the network.

4.4.1 UBERA Project 1: Results Analyses and Evaluation

Figure 4.5 shows the average packet end-to-end wireless LAN delay results for Scenario 1 and Scenario 2 in Project 1 (20 mobile phones). As discussed earlier in the previous section, Scenario 1 is the the normal IPTV and Scenario 2 is the proposed algorithm (UBERA). The results show that the proposed algorithm (UBERA) has 33% less average wireless LAN delay compared to the normal IPTV (refer to the table of data in appendix A). Thus the proposed algorithm enhance the IPTV QoS compared with the normal IPTV by reducing the average end-to-end wireless LAN delay.

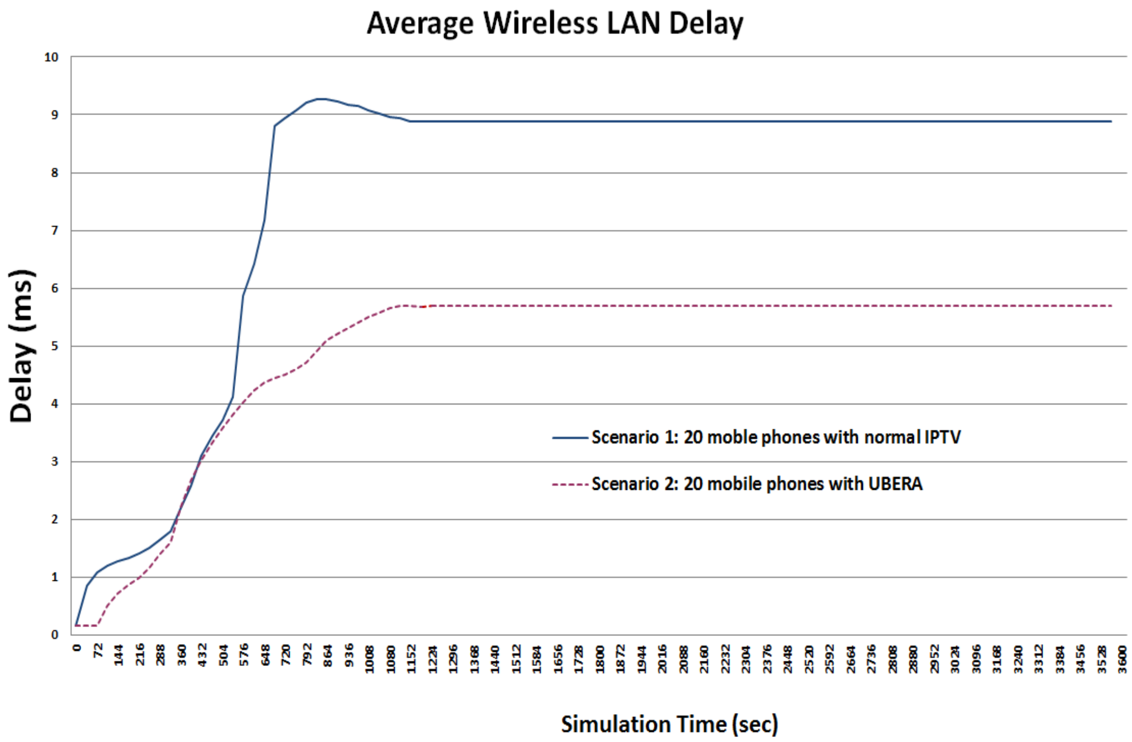


Figure 4.5 Average Wireless LAN Delay

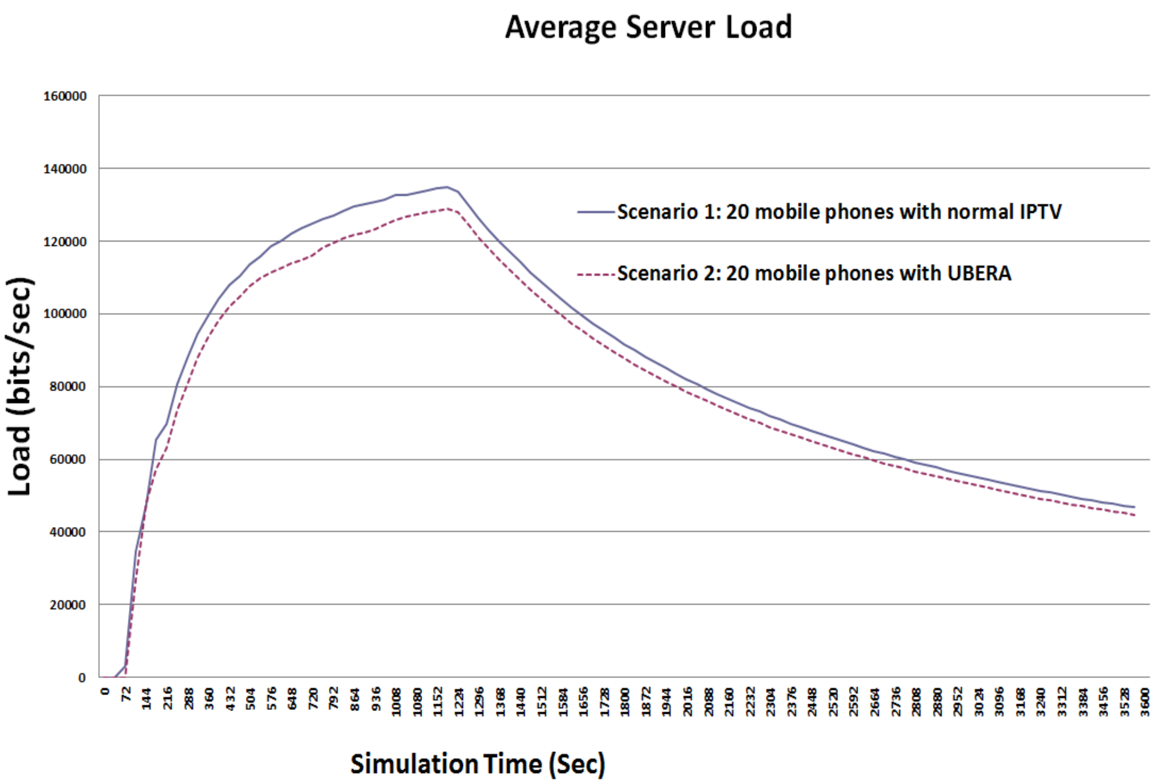


Figure 4.6 Average Server load

Figure 4.6 shows the average server load results for Scenario 1 and Scenario 2 in this project. Scenario 1 is the normal IPTV while Scenario 2 is the proposed algorithm (UBERA). As expected, Scenario 2 has the lowest server load compared to Scenario 1. The results show that, the proposed algorithm reduced approximately 3% of the server load compared with the normal IPTV. Thus, the proposed algorithm improved in the IPTV QoS. Figure 4.7 presents the average video streaming packet sent for Scenario 1 and Scenario 2. The results show that Scenario 2 has approximately 3% less packets sent by the server compared to Scenario 1 (Refer to table of data in Appendix A) . Therefore, the proposed algorithm outperformed the normal IPTV by reducing the amount of server workload.

Average delay variation (jitter) results for Scenario 1 and Scenario 2 is presented in figure 4.8. Real-time applications such as audio and video are sensetive to jitter. High amount of jitter leads to poor video quality of service. The results show that, the proposed algorithm produced 9% high jitter compared to the normal IPTV. This is due to the time taken by the server to search for available packets in the clients. Thus, at this stage buffering machanism should be applied to the proposed algorithm in order to reduce the high jittter effect and improved the general quality of service and quality of experience in IPTV services.

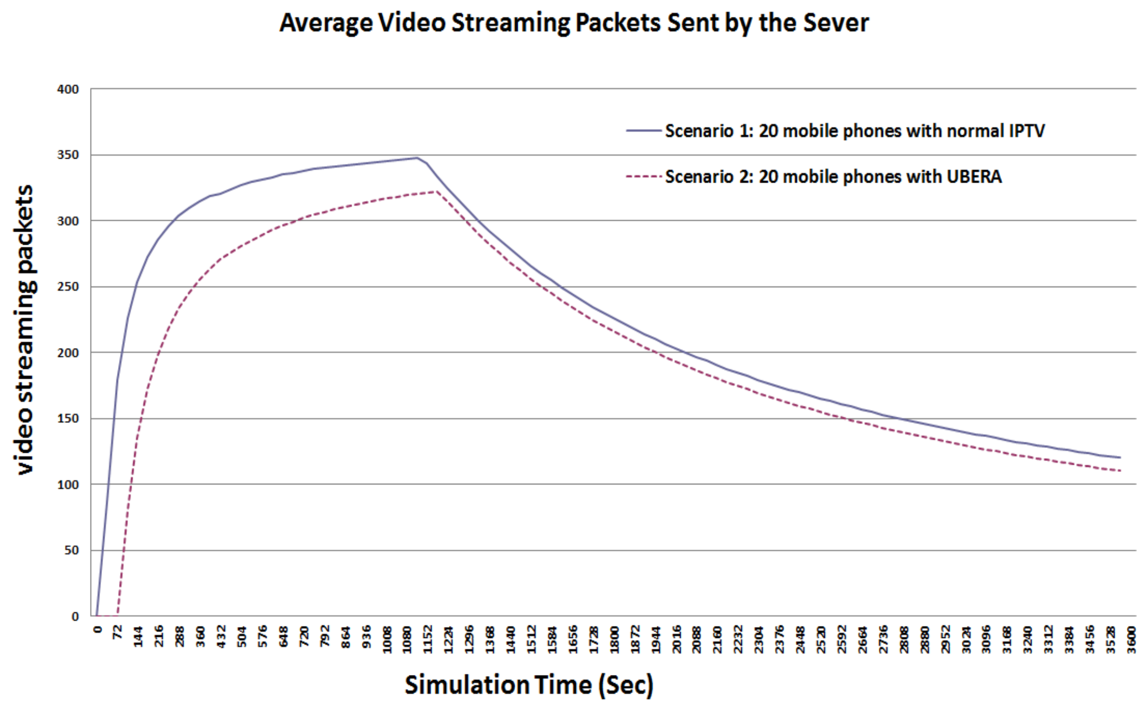


Figure 4.7 Average video streaming packets sent by the Server

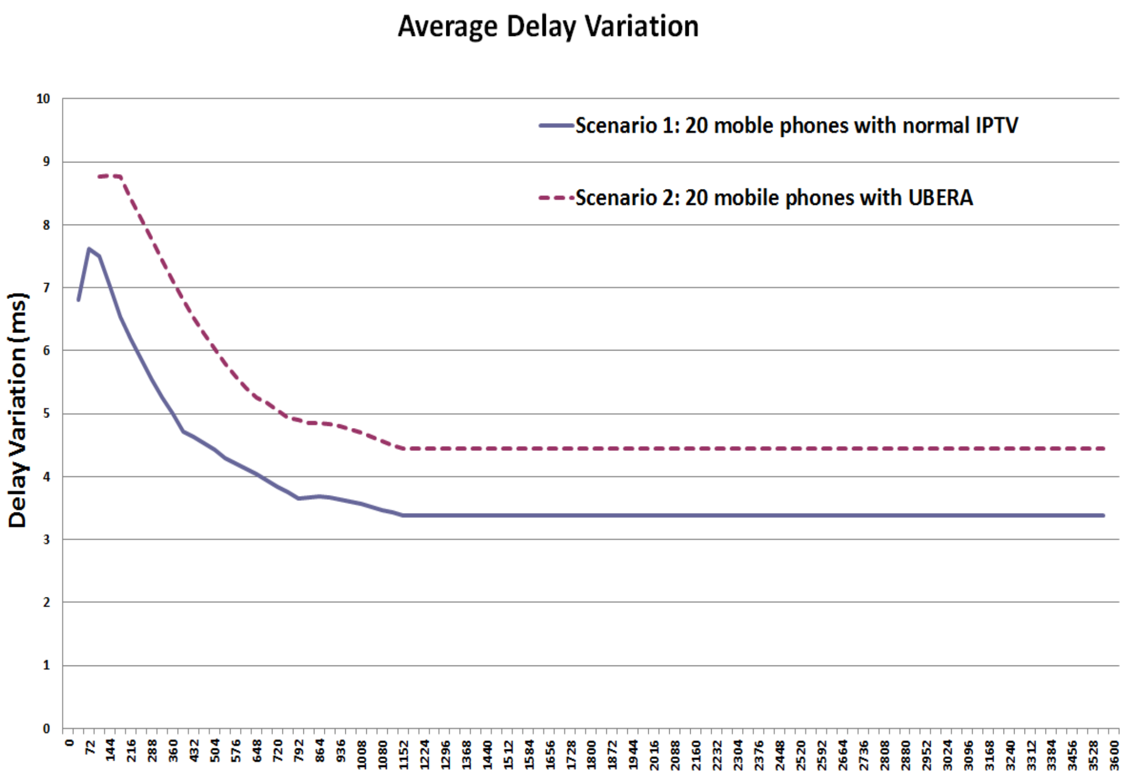


Figure 4.8 Average Delay Variation (Jitter)

4.4.2 UBERA Project 2: Results Analyses and Evaluation

Project 2 is modelled in 10 groups of 10 mobile phones each, totalling 100 mobile phones. the main idea of this project is to evalute the effeciancy of the algorithm as the number of clients (demand) increases. Two scenarios (Scenario 3 and 4) were created in this project as explained in section 4.3. Scenario 3 is the normal IPTV while Scenario 4 is the proposed algorithm (UBERA). Figure 4.9 shows the average end-to-end wireless LAN delay results for Scenario 3 and 4. The results shows that the proposed algorithm reduced avaragely 5% end-to-end wireless LAN delay compared to normal IPTV.

However, it has been observed that, when comparing Project 1 to Project 2, the average wireless LAN delay produced in Project 2 (100 devices) is twice higher than what was produced in Project 1 (20 devices). This is obvious as the network congestion increases the delay also increases. However, considering the number of devices in project 2 it 5 times more than in Project 1, it can be concluded that as the number of clients increases the efficiency of the algorithm increases.

Figure 4.10 shows the average server load for Scenario 3 and Scenario 4. Scenario 3 is the normal IPTV, while Scenario 4 is the proposed algorithm. The results show that the proposed algorithm produced 8% less server load compared to the normal IPTV. Thus, the proposed algorithm performed better by reducing the amount of server load. When comparing the two projects, Project 2 produced 2 times more server load compared to Project 1. However, when considering the number of devices in project 2 is 5 times than the devices in Project 1, it can be concluded that, as the munber of clients increases the efficiency of proposed algorithm increases in reducing the server workload.

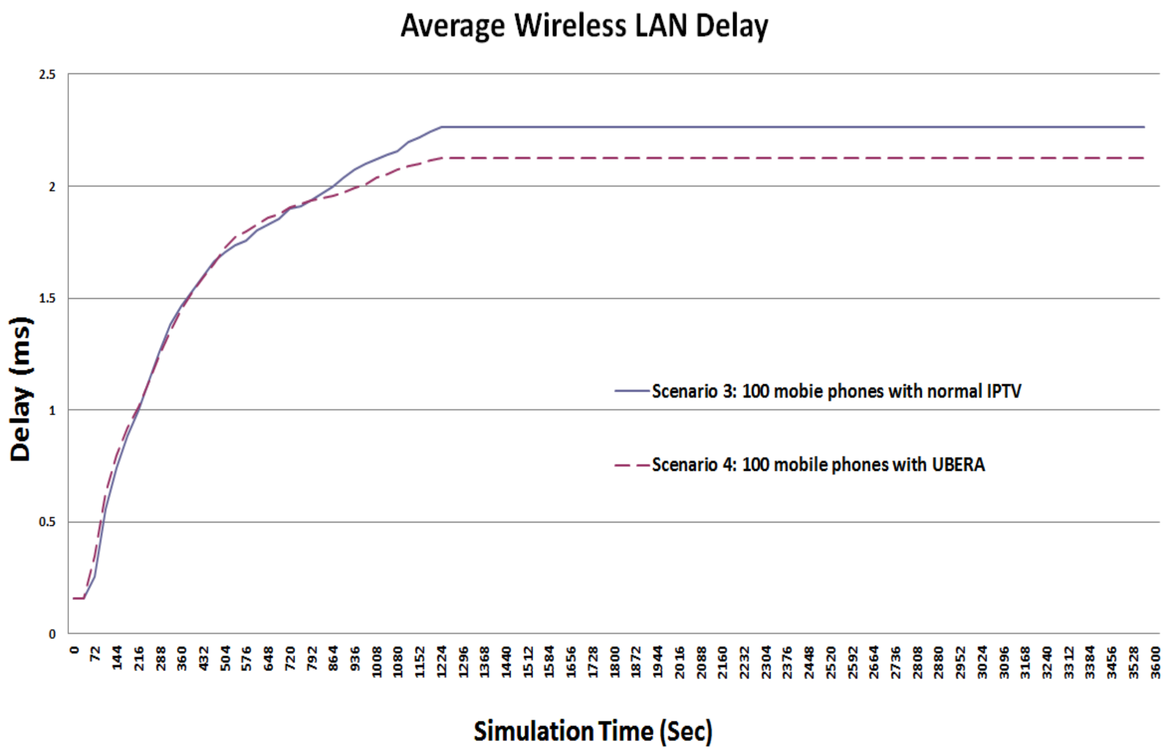


Figure 4.9 Average Wireless LAN Delay

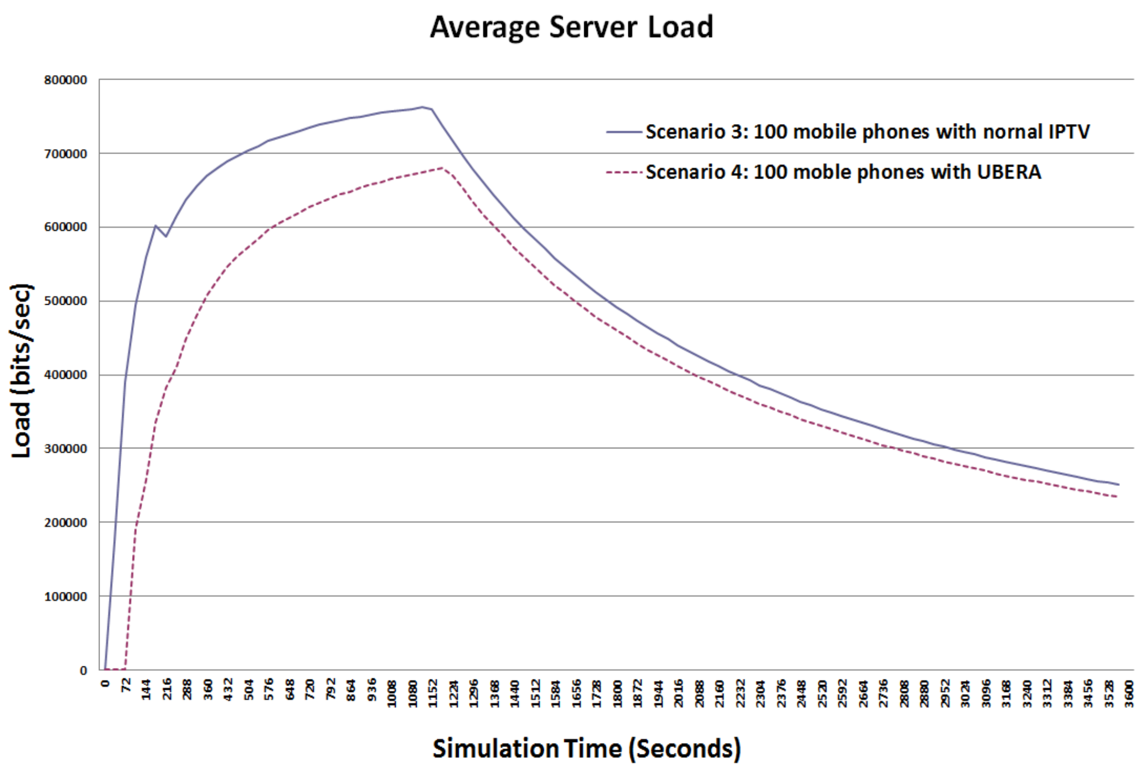


Figure 4.10 Average Server load

Figure 4.11 display the average packets sents by the server for Scenario 3 and Scenario 4. The results show that the proposed algorithm reduced averagely 7% of the packets sent by the server compared to normal IPTV. Thus, the proposed algorithm as anticipated reduces the server workload. While comparing Project 1 to Project 2, the amount of packet sent by the server in Project 1 is considerably more than what was sent in Project 2. This is due to the number of clients that have the packets are more in Project 2 than that of Project 1. Thus, the efficieny of the algorithm improved as the number of clients increases.

Average delay veriation (jitter) results for Scenario 3 and Scenario 4 is presented in Figure 4.12. The results show that the proposed algorithm (UBERA) produced averagely 8% higher jitter than the normal IPTV. Similarly, when comparing Project 1 with Project 2, the amount of jitter produced in Project 2 is twice than what is produced in Project 1. However, critical observation revealed that the efficiency of the proposed algorithm has improved in Project 2 comparing the number of devices in Project 2 are 5 times more than that of Project 1. Thus, the proposed algorithm improves as the number of clients increases.

4.4.3. Conclusion

For all the results analysed in both projects and scenarios, it can be concluded that the proposed algorithm performed well in reducing the average amount of server load and packet end-to-end delay. Although the amount of average delay variation (jitter) produced by the proposed algorithm is 8% higher than what the normal IPTV produced. However, this can be addressed by buffering packets at the client storage before playing them. This may not be the best solution to the jitter, improvement is expected at the next designing stage of the proposed algorithm. As earlier explained also, this is the first stage of the algorithm design, where unicast scheme (VoD) was the only IPTV services condidered. However, improvements are expected and the multicast scheme will be included in the next chapter. The results also show that as the number of clients increases, the efficiency of the proposed algorithm also increases.

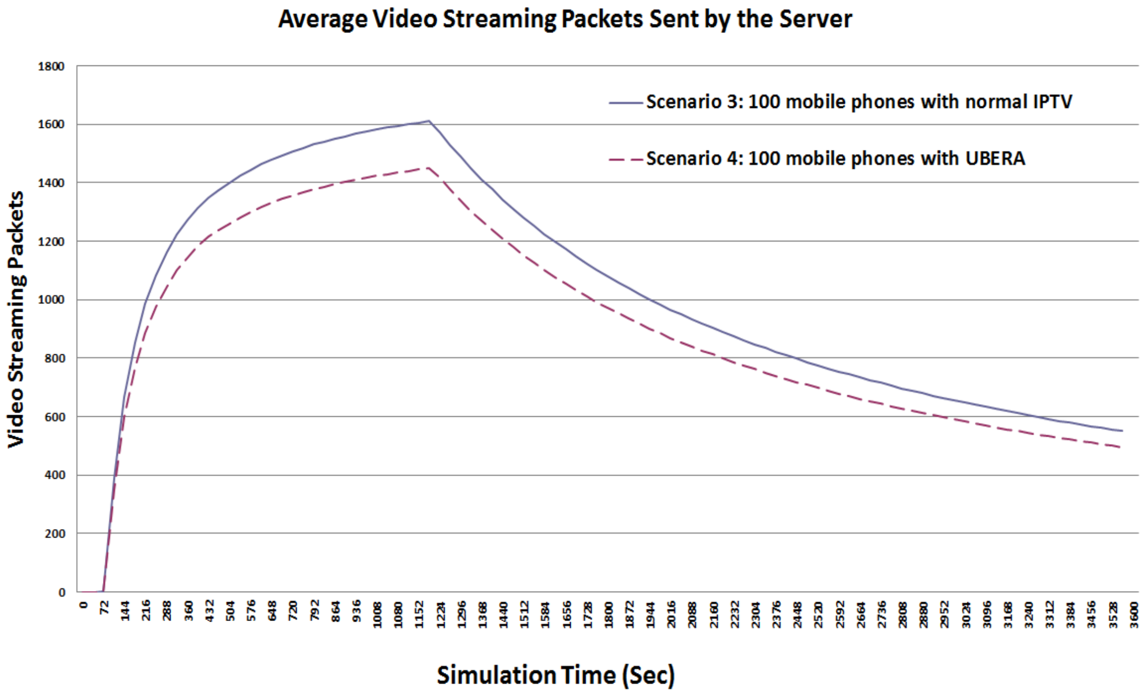


Figure 4.11 Average video streaming packets sent by the Server

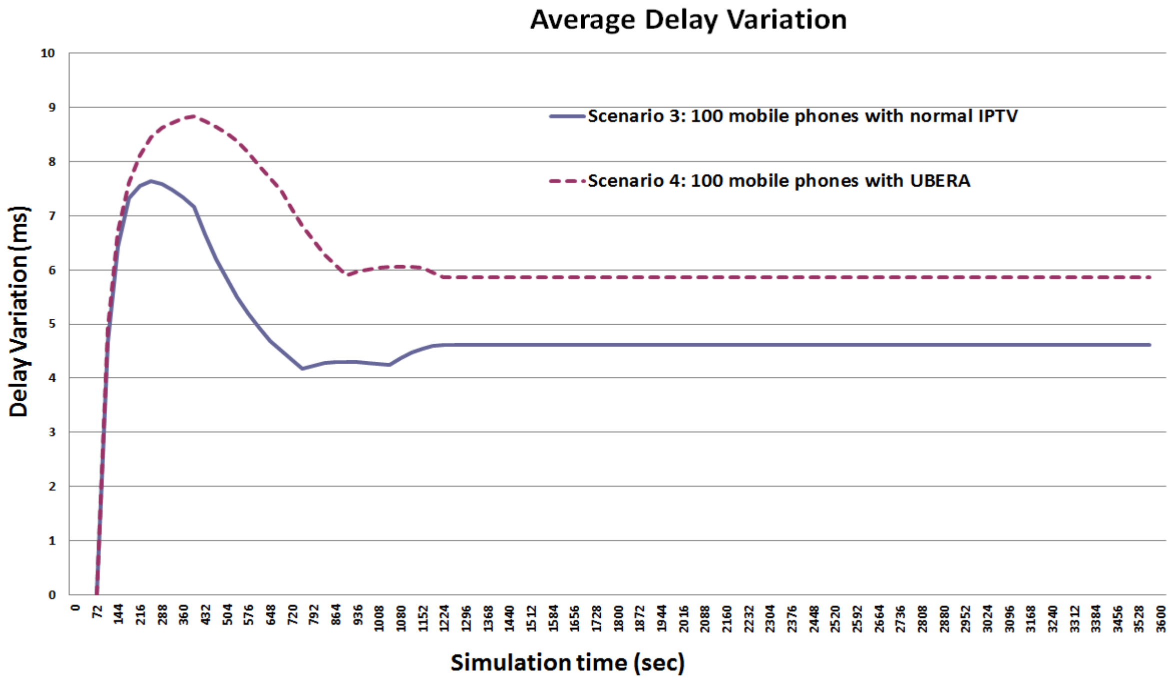


Figure 4.12 Average Delay Variation (Jitter)

Chapter 5. Effective Resource Utilization Routing Algorithm for IPTV

5.1 Introduction

IPTV services are delivered using both unicast and multicast schemes as explained in Chapter 2. Video-on-Demand (VoD) is delivered using unicast scheme, where each client is streaming requested contents from the server separately. Other live and time shifted services such as news, sports etc are delivered using multicast scheme, where clients are grouped together and served the same content at the same time. Previous proposed algorithm discussed in Chapter 4 uses only unicast to serve the video-on-demand requests. However, there should be an algorithm that deals with both unicast and multicast schemes in serving IPTV clients.

5.2 Effective Resource Utilization Routing Algorithm for IPTV (ERURA)

Effective Resource Utilization Routing Algorithm (ERURA) for IPTV [55] is an extension of Unicast Bandwidth Efficiency Routing Algorithm (UBERA) [54] discussed in Chapter 4 above. The server uses the algorithm (UBERA) to choose between multicast and unicast to serve an incoming service request received from clients. The extension of UBERA is obvious in order to include both unicast and multicast schemes to include all the IPTV services. Several algorithms have been researched and proposed by researchers. These algorithms were discussed in Chapter 2, to mention few here are; a heuristic algorithm that construct a shared-route in multicast routing networks that provide communications between multi-users [56]. Maximum delay, average delay and estimated delay between nodes were adopted and measured for performance analysis. The results show that the algorithm outperforms the other algorithms. Also a Heuristic Gradient Based Multicast Routing Policy for Dynamic Network was proposed by[57], which the routing policy is adapted to the network conditions based on the gradient that multicast group members join and leave a multicast session dynamically.

However, both proposed studies are only for the multicasting scheme, the unicast scheme is not considered. Therefore, there should be an algorithm that includes both unicast and multicast for the successful delivery of all the IPTV services. The Effective Resource Utilization Routing Algorithm is proposed to address this problem, taking into account both unicast and multicast schemes.

The proposed algorithm is designed, modelled and simulated in OPNET Modeler and the simulation results show that, the proposed algorithm provides unlimited virtual bandwidth and reduced significant server workload without affecting the general quality of service. Figure 5.1 illustrates the flowchart and Algorithm 5.1 presents the pseudo-code of the proposed algorithm. The algorithm operations are explained in the following steps:

Step 1: When the server has received a video streaming request(s) from a client node(s), it first checks the number of the same requests at that particular time.

Step 2: If the number of requests is two or more, then the server creates a multicasting group, and adds the requesting clients into it and sends the requested packets through the multicast scheme.

Step 3: If the number of request is one, the server first checks its database for any node(s) that already received the same requested packet(s) within the node group.

Step 4: If the packet(s) information is found on a client, then the server checks the time it was sent and the number of forward request (FREQ) messages sent to that client for packet(s) forwarding.

Step 5: If there is no FREQ message sent to that client and the set time has not expired, then the server sends a FREQ to that client with the destination address in the message header for the packet to be forwarded to the requesting client.

Step 6: If there is no client found that has the packet(s) within the group of the requesting client or the set time has expired, or a client that has the packet(s) is found but already engaged on another packet(s) forwarding activity, then the server checks other groups for any client that might have received the packet(s).

Step 7: If the client node that has the packet(s) is identified in the other groups and the set time is valid and there is no packet(s) forwarding activity on that client, then the server node sends a FREQ message to that client with the destination address in the message header for the packet(s) to be forwarded to the requesting node.

Step 8: If no client node is found that has the packet(s) or the set time has expired or there is forwarding process going on that client, then the server sent the requested packet(s) to the requesting node.

Step 9: The server updates the packet information to its database.

In all the steps described above, route reply messages (RREP) are propagated back to the source node when the route path is discovered. Also a route error message (RERR) is propagated back to the server for any broken link.

Input: Time; number of FREQ

Output: Serve Request

Procedure:

```
Request of X_i…n Packet(s)
If i ≥ 2
        Serve using multicast
elseIf X_i is available in the array j[ ]
        if time <= 5 && FREQ = 0
                Send FREQ
        Else send Xi
        Update
Else check X_i in arrays j+1…n[ ]
End
```

Algorithm 5.1: ERURA Pseudo-code

The server node is responsible for the nodes grouping and keeping up to date information about groups in its database. The nodes are being grouped based on their similarities with other nodes in term of their viewing behaviour and are within the same geographical location. If a request was received from a particular client node for the first time, i.e. if the node newly joined the network, the client will be added to a group called “temporary group”, until enough information is gathered to be placed in the appropriate group. The grouping is a measure aim at reducing the amount of time for the server to search all the client nodes in the network for a particular packet(s). Example, it may be faster to search and find packet(s) on action movies within clients that are watching action movies than those that are watching comedies.

To avoid overloading a particular client node, FREQ message is limited to only one per client

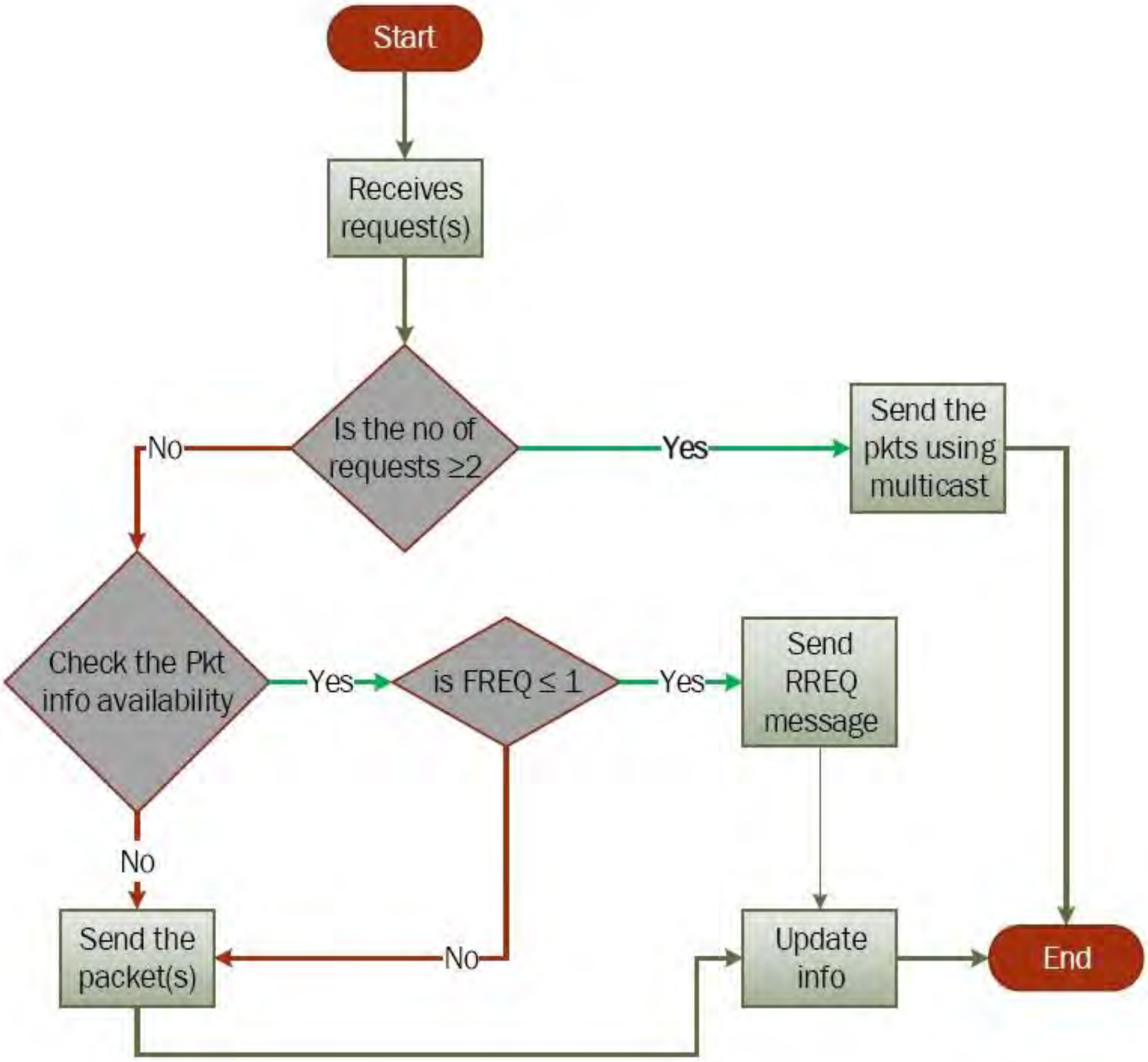


Figure 5.1 Effective Resource Utilization Routing Algorithm for IPTV

5.3 Description of Projects and Scenarios

The proposed algorithm is designed, modelled and simulated in OPNET Modeler 17.5. As discussed previously in Chapter 2, AODV is one of the MANET routing protocols that is widely used. It is a reactive or on-demand routing protocol that route discovery mechanism is started by the source in finding the route to the destination. The route remains valid till destination is unreachable or until the route is no longer needed. AODV is one of the algorithms that are built in OPNET. Therefore, in this study AODV routing algorithm is amended to include other aspects of the proposed algorithm.

OPNET Modeler supports real-world data into its simulation environment so as the simulation can be as close to the real application as possible. Video streaming traffic (data) can be captured using video streaming software such as VLC and the data are saved in a file, which is use in OPNET Modeler to simulate the same traffic. In this research work, a live video programme (2014 Winter Olympics) was streamed using the BBC iPlayer [53] and the streaming data was captured using VLC. The captured file was stored in the OPNET primary directory and used to simulate all the projects and scenarios. The summary of node configuarations are stated in the Table 5.1.

The simulation projects are designed and modelled to represents the typical IPTV network with the normal IPTV and the proposed algorithm. Unicast and multicast scheme are used in serving the IPTV service requests. Two projects were designed to evaluate the efficiency of the proposed algorithm. The projects and scenarios are explained as follows:

5.3.1 ERURA Project 1

This project was designed with 20 mobile phones in two groups of ten mobile phones each. The clients are streaming video content from video streaming server over the Internet as shown in Figure 5.2. The main reason of creating this project is to evaluate the effectiveness of the proposed algorithm against the normal IPTV with small number of devices. Two scenarios were created for the evaluation process and are are explained as follows:

Scenario 1: Normal IPTV

Scenario 2: Proposed algorithm (ERURA)

5.3.2 ERURA Project 2

Project was modelled with 100 mobile phones in ten groups of ten mobile phones each. The mobile clients are streaming video contents from video streaming server over the Internet, as illustrated in Figure 5.3. This project was created with 100 mobile phones to evaluate the efficacy of the proposed algorithm against the normal IPTV as the number of clients increases. Similarly, two scenarios were created and explaned as follwos:

Scenario 3: Normal IPTV

Scenario 4: Proposed algorith (ERURA)

Table 5.1: Summary of node configuration

Configuration parameter	Value
Wireless technology	IEEE 802.11n
Mobility	Random waypoint
Packet Reception Power Threshold	-95dB
Data rate	6.5 Mbps (base) / 60 Mbps (max)
Start of data transmission	normal (100, 5) seconds
End of data transmission	End of simulation
Duration of Simulation	3600 seconds (1hr)
Packet inter-arrival time	Based on the captured data
Packet size exponential	Based on the captured data

Figure 5.2: Network setup for Project 1

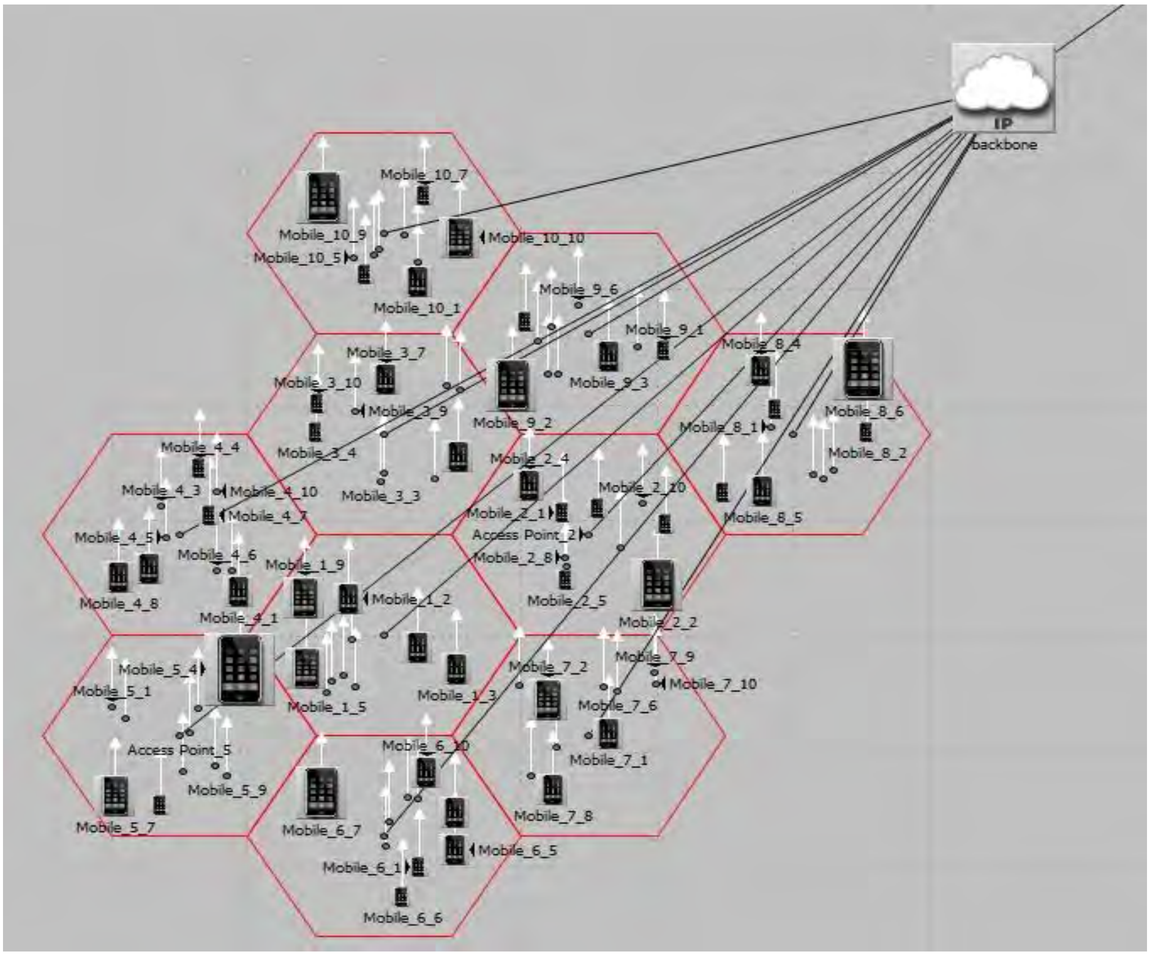


Figure 5.3: Network setup for Project 2

5.4 Results Analyses and Evaluations

The parameters considered in this research work include the average packet delay variation (jitter), average packet end-to-end delay and average server load. These parameters were compared and analysed in all the scenarios. As explained in the previous chapter, the parameters have been defined by [44] as: packet delay variations is the variance among packets end-to-end delay for all the video streaming packets from the source node to the destination node; packet end-to-end delay is the time taken to send the video streaming packets from the source node to the destination node application layer for all the nodes in the network. Server load represents the total load in bits/sec at the source node.

The scenarios created were compared analysed and evaluated. As mentioned in Section 5.3, the proposed algorithm (ERURA) uses MANET characteristics and Internet Protocol (IP) combined together to serve video streaming requests from video streaming server to the clients. It also uses both unicast and multicast schemes for service delivery. While normal IPTV uses only IP to serve the incoming service requests. The results for each project are explained as follows:

5.4.1 ERURA Project 1: Results Analyses and Evaluation

Figure 5.4 show the results for Scenarios 1 and 2. As mestioned earlier in previous section, Scenario 1 is the normal IPTV services, while Scenario 2 is the proposed algorithm. The results show that Scenario 2 (proposed algorithm) produced high packet delay variation to about 7% in average compared to Scenario 1 (normal IPTV). However, this problem can be address by buffering packets on the clients storage before they are played. However, ways for improvents will be explored at the next stage.

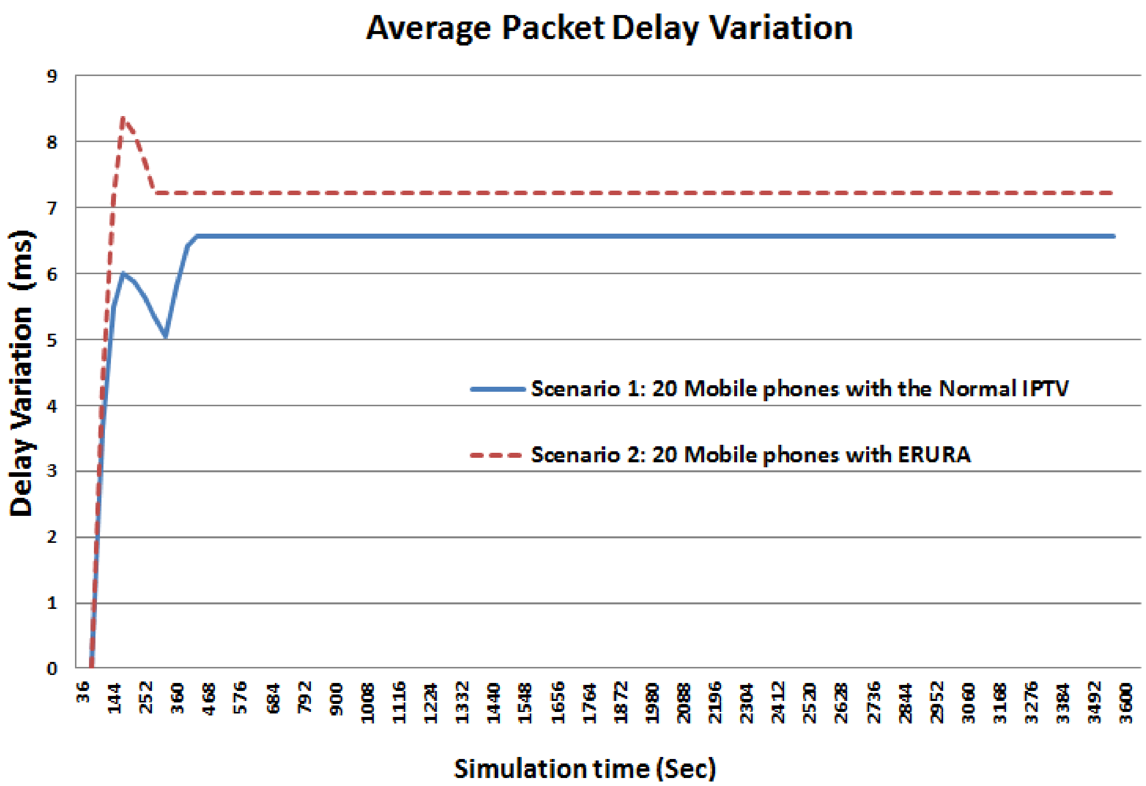


Figure 5.4: Average Packet Delay Variation (jitter)

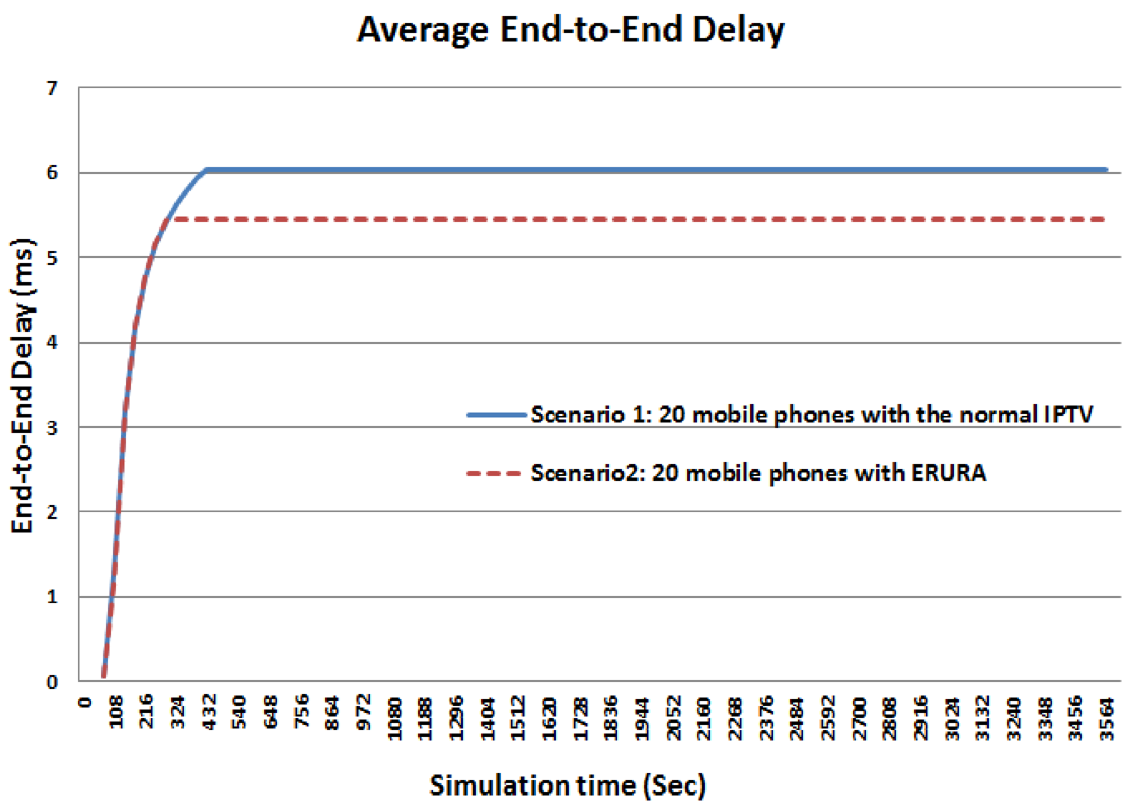


Figure 5.5: Average Packet End-to-End Delay

Figure 5.5 show the average packet end-to-end delay results for Scenario 1 and Scenario 2. The results show that Scenario 2 (proposed algorithm) produced average 12% less packet end-to-end delay compared to Scenario 1 (normal IPTV). This is because it is faster to send a packet from client to client than from the server to the clients because their proximity. Therefore, the proposed algorithm perform well in reducing packet end-to-end delay compared with the normal IPTV. Thus, it enhanced the IPTV QoS.

The average server load results for Scenario 1 and Scenario 2 is presented in Figure 5.6. The results show that Scenario 2 (proposed algorithm) reduced the amount of server load to 5% in the average compared to Scenario 1 (normal IPTV). Similarly, the proposed algorithm outperformed the normal IPTV.

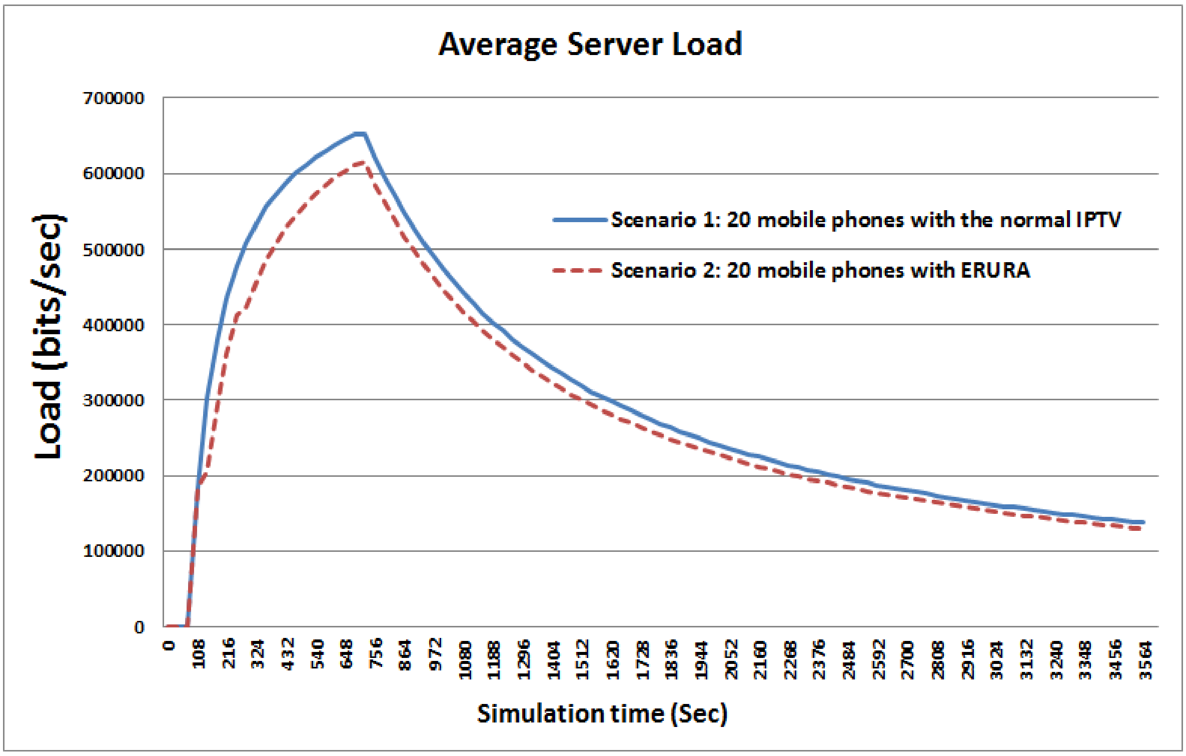


Figure 5.6: Average Server load

5.4.2 ERURA Project 2: Results Analyses and Evaluation

This project was modelled with 100 mobile phones as earlier in Section 5.3 The aim is so analyse and evaluate the efficiency of the proposed algorithm with larger number of clients to evalute it is efficiency as the number of clients increases. Similarly, two scenarios were created in this project , that is Scenario 3 and Scenario 4. As stated also in Section 5.3, Scenario 3 is the normal IPTV, while Scenario 4 is the proposed algorithm. Figure 5.7 show the average packet delay variation for Scenario 3 and 4. The results show the proposed algorithm produced about 40% more delay variation compared with the normal IPTV. Similarly, when the two projects were compared, Project 2 produced higher packet delay variation compared with Project 1. This is due to the number of clients in Project 2 are more than in Project 1. Therefore, for the algorithm to performed effecyively, it is recommended to use packet bufferng machanism as recommended ealier.

Figure 5.8 show the packet end-to-end delay results for Scenario 3 and 4. The results show that Scenario 4 (proposed algorithm) produced in average about 8% less amount of end-to-end delay compared with Scenario 3 (normal IPTV). This is due some of the packets that were forwarded by the clients, and the proximity between clients are closer than the server. When comparing Project 1 with Project 2, the amount of packet end-to-end delay produces by the Project 2 is slightly higher than that of Project 1. This is due to the number of clients in Project 2 are more than in Project 1. Thus, in both projects the proposed algorithm outperformed the normal IPTV. Also as the number of clients increases the efficeincy of proposed algorithm increases.

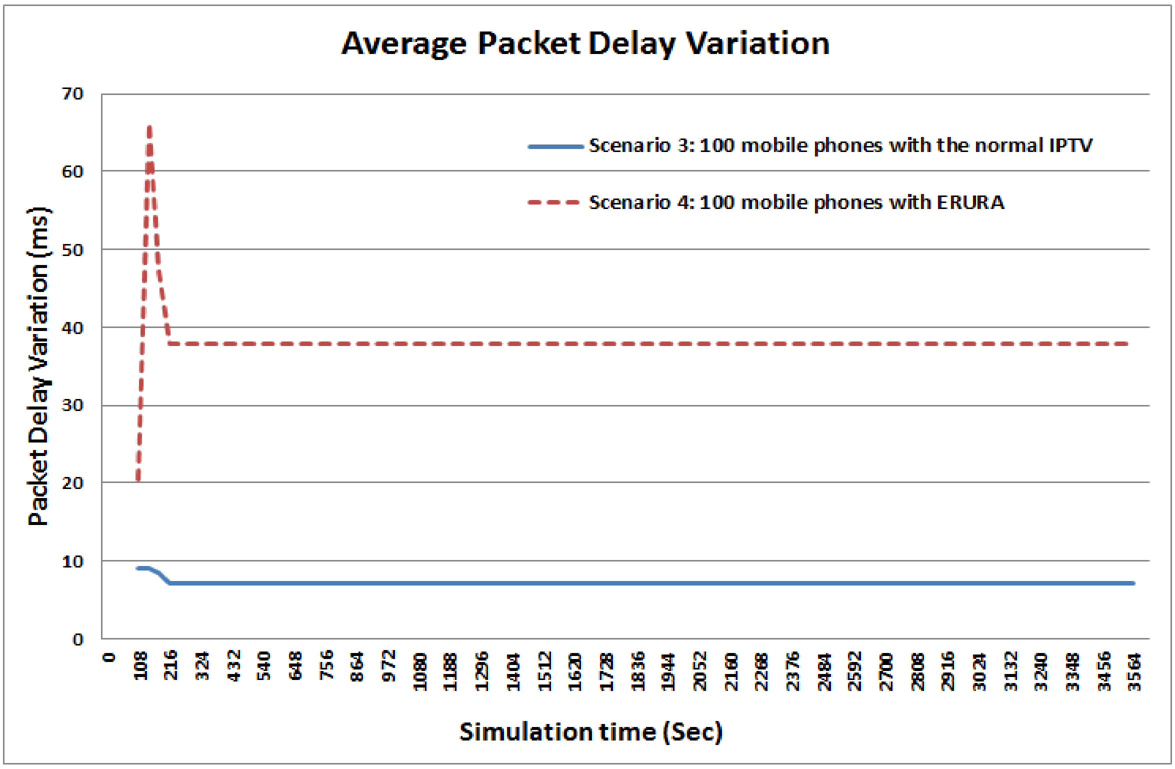


Figure 5.7: Average Packet Delay Variation

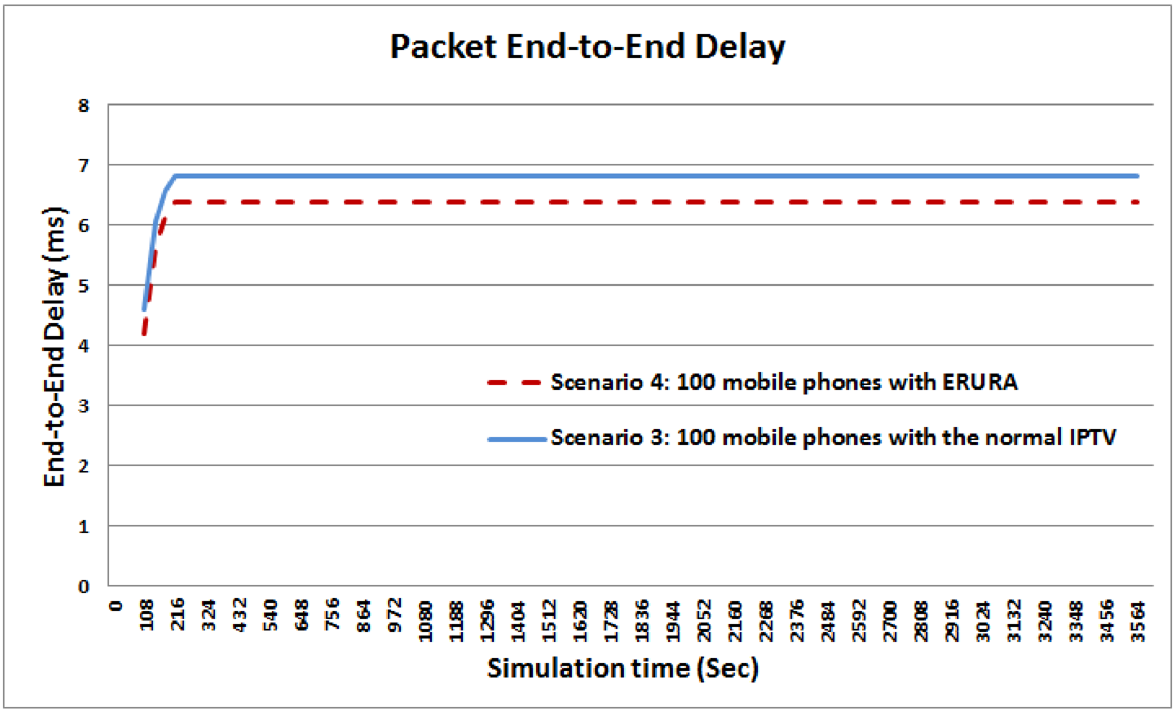


Figure 5.8: Average Packet End-to-End Delay

Average server load results for Scenario 3 and 4 are shown in Figure 5.9. The results show that Scenario 4 (proposed algorithm) reduced amount of server load to 13% in the average compared to Scenario 3 (normal IPTV). This is due to some of the requests were served by the clients; thus, it reduced the total server workload. When the two projects were compared, Project 2 reduced significant amount of server load compared to Scenario 1. Similarly, this is due to the number of clients that forwarded the packets are more in Project 2 than in Project 1. Therefore, as the number of clients increases the efficiency of the proposed algorithm increases. Thus, in both projects the proposed algorithm reduces significant amount of the server load.

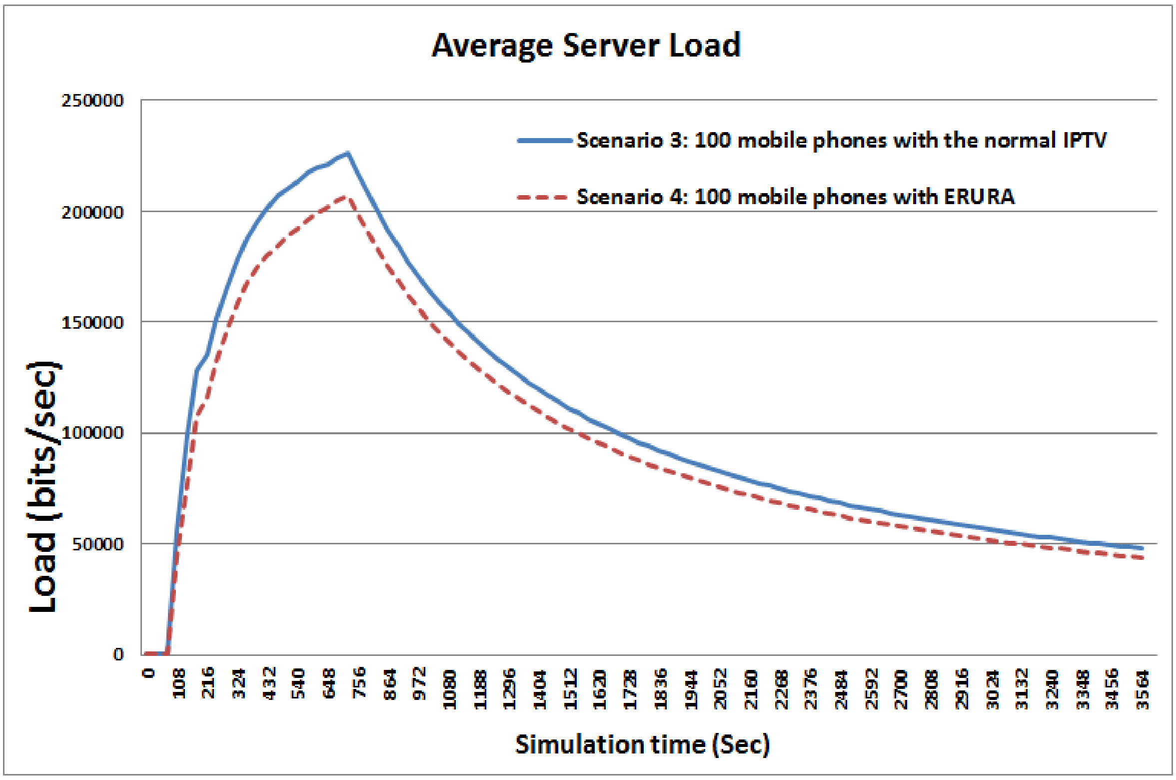


Figure 5.9: Average Server Load

5.5 Conclusion

Based on the results analysed in both projects and all scenarios, it can be concluded that, the proposed algorithm outperformed the normal IPTV on server load reduction and packet end-to-end delay. Thus, the proposed algorithm enhanced the QoS and QoE in IPTV service. However, the amount of jitter produced by the proposed algorithm is higher than what the normal IPTV produced. Although, as recommended, this problem can be addressed by buffering packets on the clients storage before playing them. Exploring new idea to solve this problem or at least to improve on it, is still on going at this stage. Though, the proposed algorithm in this chapter has covered the use of unicast and multicast in delivering IPTV services. However, recently service providers are using CDN based network architecture to enhance the IPTV service delivery. This architecture has been explained and incooperated in the next chapter.

Chapter 6: Adaptive CDN-Based Bandwidth Conserving Algorithm for Mobile IPTV (BCA-CDN)

6.1 Introduction

The present-day network bandwidth and server capacity is gradually becoming overloaded due to the high demand and rapid evolution of high quality multimedia services over the Internet. Internet Protocol Television (IPTV) is among the multimedia services that demand more of network and server resources, especially with the emergence of Mobile IPTV. IPTV is defined as a multimedia service delivered over Internet Protocol (IP) based networks. It is managed to provide the required level of QoS and QoE, security, interactivity and reliability[3]. IPTV provides an alternative to traditional television services in creating a video-centric next-generation Internet service accessible on any device, such as a mobile phone, tablets, computer, or HDTV, at anytime and anyplace the consumer chooses [2]. This makes demand for IPTV service in wireless networks higher and with the demand expected to grow over time [3].

Providing required quality of service and quality of experience to the end users has been a challenging issue for the service providers. It is imperative for the service providers to maintain good quality management services in order to satisfy their clients. Clients are now considered as one of the key players in the design of network control parameters by constantly monitoring and analysing the interaction between end users and various applications as well as obtaining feedback from the users.

To improve the IPTV required QoS and QoE, a Content Distribution Network (CDN) approach is being adopted and used by service providers. In the CDN approach, service providers replicate contents over multiple distributed servers, where edge/replicative servers that have the copy of the original contents from the main server and are closer to the end users are used to serve the requests. Individual client's service requests are being rerouted to a server that is more appropriate in servicing the request. The selection of the suitable server depends on the number of network parameters such as proximity to the end user, available bandwidth, throughput, requests volume and pattern, background traffic etc. This approach is an effective way of improving the QoS and QoE by reducing the delay time, and packet loss rate. However, the servers are sometimes overloaded due to the high demand for IPTV services, especially during international events, such as sports, which results to the poor QoS.

The Internet uses Internet Protocol (IP) for packets routing across different networks. IPTV like any other Internet services uses IP for packet routing. For the Video-on-Demand (VoD) separate connection is established for each request, thus, leads to high bandwidth consumption, network conjunction and server overload. To reduce the total dependency on the server by all the clients, the proposed algorithm in this paper, uses IP and Mobile Ad hoc Networks (MANET) protocol characteristics. Clients are requested to serve some of the service requests on behalf of the server. Some of the packets sent by the server to the clients can be rerouted to other clients.

MANET is an autonomous system of mobile devices that are connected via wireless links without prior planning or need for any existing network infrastructure. The mobile nodes communicate with each other without a centralized control system[3]. Mobile devices communicate with each other not only as a source or destination, but also as routers for packet forwarding in a wireless network. The proposed algorithm is an extension of our previous works [54][58] that combined and used the features of IP, MANET and CDN to improve the IPTV quality of service, by reducing the bandwidth consumption and server overload.

Bandwidth and throughput are among the major QoS parameters in delivering video content over the Internet. Several server selection algorithms were proposed using Available Bandwidth Estimation Tools (ABETs), to provide an estimate of the available bandwidth to be used in the server selection process. The available bandwidth of an end-to-end network path is defined by [59][60] as "a streaming capacity, that is, the amount of traffic that can be sent along the path without congesting it". The majority of proposed bandwidth availability estimation tools are classified into two categories: the Probe Rate Model (PRM) and the Probe Gap Model (PGM) [59]. The PRM uses packet trains and sends them along the network path; this model is regarded as self-induced congestion. The PGM uses packet pairs and bases its estimation on the differences between their input and output time gaps.

The PRM model has been used in many available bandwidth estimation tools and proved accurate [59]. However, it has its drawbacks. For example, to estimate the available bandwidth, the traffic rate sent must be equal to or greater than the available bandwidth. This may lead to congestion in the network path. Several tools such as Delphy [61], TOPP [62], PathLoad [63], IGI/PTR [64], pathChip [65], Bart [66] use this model. Moreover, [67] show that PGM can underestimate the available bandwidth of multi-hops with one-hop persistent traffic [59]. In this work PathLoad is used as a tool for estimating the end-to-end available bandwidth due to its accuracy. The remaining part of this chapter is organised in five sections. Section 6.2 provides the background and related work, section 6.3 provides a description of the proposed algorithm, section 6.4 describes the projects and scenarios, section 6.5 provides a detailed analysis of the simulation results and section 6.6 provides the conclusion.

6.2 Background And Related Work

This section reviews the research conducted on the server selection method for CDN and describes the motivation behind choosing clients to temporarily serve some of the requests received by the server.

A QoE-based server selection algorithm in the context of CDN architecture was proposed by [4]. Realistic characteristics of the server selection process were used to formalize the selection model as a sequential decision problem solved by the multi-armed bandit (MAB) paradigm. The results show that the approach yields significant improvements in terms of user perception, compared to traditional methods. However, the proposed algorithm does not take into account periods of high service demand; instead it only takes into account normal service demand. An algorithm called “An Efficient Algorithm for the Video Server Selection Problem” is proposed by [68]. The main objective of the algorithm is load balancing among multiple distributed network servers to minimize delay for a video request to be served. However, the proposed algorithm tackles only the issue of load balancing to reduce delay, which is only one aspect of network performance measure. Other parameters, such as proximity, available bandwidth etc., are not considered in their research work.

Recently, Hyunwoo et al [69] proposed a dynamic network condition-aware video server selection algorithm over wireless networks. The algorithm considers some network information from the client edge node (such as Internet service provider, Wi-Fi router, Radio network control etc.) to the content servers. However, this algorithm does not guarantee end-to-end quality of service, since the end user node is not considered. Huan et al [70] also proposed an algorithm for joint optimization for content replication and server selection for video-on-demand. This proposed algorithm demonstrates considerable reduction of load and delay. However, it also requires that the local server have a considerably large amount of storage capacity. This makes it very expensive, or even impossible, to execute. It also does not consider the use of mobile devices in its design.

A context-aware server selection algorithm for mobile thin client computing is proposed by [71]. The algorithm is appropriate for mobile devices and it considers their dynamic mobility changes. However, more than one migration from one server to the other due to the server response time lapse, as designed by the proposed algorithm, may increase the amount of delay time, leading to poor quality of service. Furthermore, Chang et al [72] proposed a server selection algorithm with minimum latency for heterogeneous cloud services. The Video-on-Demand service is considered in the design of the proposed algorithm, but it can be extended to other services. The proposed algorithm shows a considerably greater amount of latency reduction than the others (YouTube and random algorithms). However, as it has been explained in the paper that, the proposed algorithm only considers a fixed distance between the end user and the video service provider. As such, it does not consider the dynamic mobility of mobile devices.

An essential observation is that all the proposed algorithms presented above consider normal or low request service demand. However, in practice, during high demands, e.g., during international events like sport tournaments or new movie releases, the server may be overloaded, which results in high packet delay, network congestion, jitter and loss of packets. All these parameters are sensitive to video application. Also, the design of some of the algorithms presented above considers only fixed nodes and no mobile nodes are taken into consideration. With the current trend of mobile IPTV, mobile devices must be taken into consideration in all aspects of service delivery. To alleviate this problem we propose a new algorithm that considers mobile devices and uses clients to serve some of the incoming service requests during high demand periods, depending on the server's available bandwidth.

Figure 6.1: Adaptive CDN-Based Bandwidth Conserving Algorithm for Mobile IPTV

6.3 Description of The Proposed Algorithm

Adaptive CDN-Based Bandwidth Conserving Algorithm (BCA-CDN) for Mobile IPTV is a content delivery network network-based algorithm that adapts to different server bandwidth capacity and packet availability to improve the QoS. It uses mobile clients to serve some of the incoming requests during high service demand situations, depending on the server's available resources and the proximity of the servers to the requesting client as illustrated in Figure 1. The forwarding clients are selected based on the initial agreement reached between the clients and the service providers and are compensated for the services offered. The proposed algorithm has been designed for CDN-based Mobile IPTV, but can be extended to other CDN services.

The algorithm is modelled and incorporated in the servers, where the selection policy is taking place based on the network information stored in its database. It performs the selection process by choosing the appropriate server or client to serve the request. Riverbed Modeler 18.0 was used for the simulation of the network setup. The simulation results show that the algorithm provides not only unlimited virtual bandwidth but significantly reduces workload at the main server which in turn improves the general quality of service and experience. Figures 2 and 3 illustrate the flowchart and the pseudo-code of the proposed algorithm and its operations are explained in the following steps:

Step 1: When the edge/replicative server has received a video streaming request(s) from a client node(s), it checks the availability of the requested packet(s)

Step 2: It computes the bandwidth needed to serve the request(s)

Step 3: It compares the available bandwidth with the requested bandwidth

Step 4: If the requested bandwidth is less than or equal to the available bandwidth, it serves the request(s)

Step 5: If the requested bandwidth is greater than the available bandwidth, it checks the availability of the requested packet(s) on the clients

Step 6: If the packet(s) is available on any client, it checks the number of forward requests (FREQ) on that client.

Step 7: If the number of FREQ is less than or equal to 1, it then sends a Re-Route Request (RREQ) message to that client.

Step 8: The database is updated

Step 9: If the packet is not available on any of the clients or the client that has the packet has more than 1 FREQ message, the algorithm then places that request on the waiting list.

Step 10: A flag is set for available bandwidth.

Step 11: If the flag is on, the algorithm serves the next client on the waiting list by repeating step 3.

Step 12: If the requested packet(s) is not available and the requested bandwidth is less than the available bandwidth, it then requests the packets from the main server.

Step 13: Packets are forwarded to the requesting client

Step 14: The database is updated

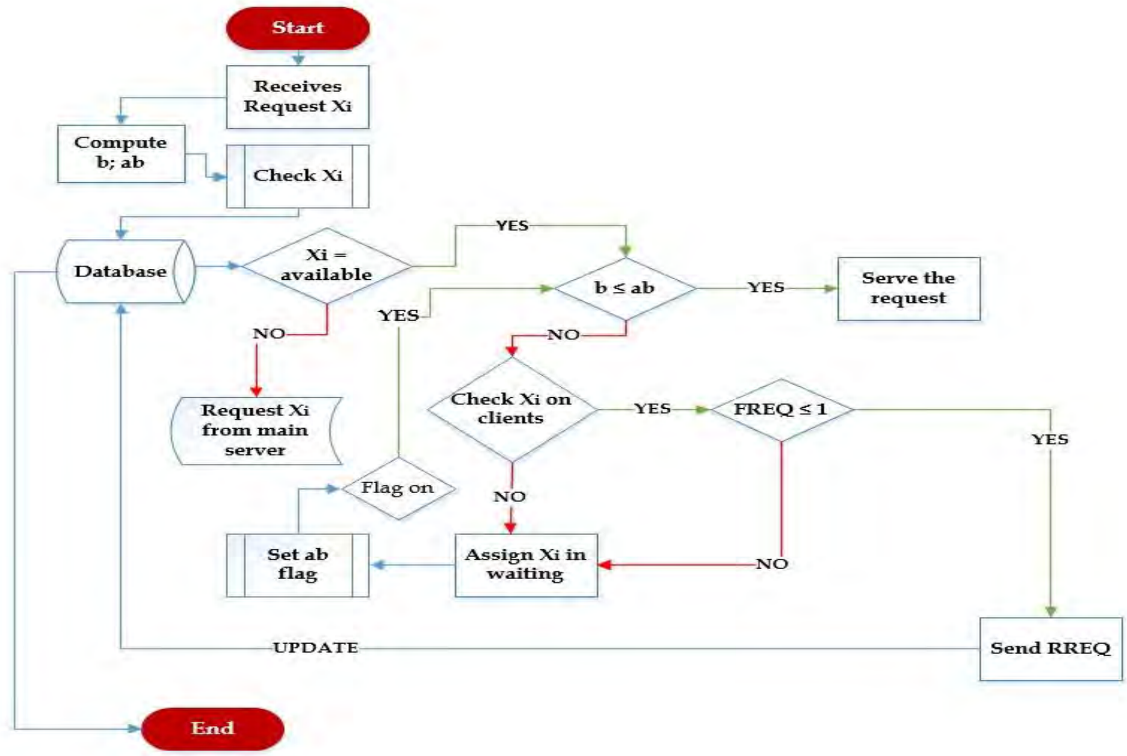


Figure 6.2: **Adaptive CDN-Based Bandwidth Converving Algorithm Flowchart**

Input: available bandwidth
Output: serve Request
Procedure:

```
REQ of Packet xi…n
    If xi is available
        Check bandwidth [ab]
        While  ab < b
    If xi ,  j[ ] = available && FREQ ≤ 1
            send RREQ
            Else
            assign REQ →Wlist
        EndWhile
        Serve REQ
        Update Dbase
    Else REQ xi from the main server
    send xi
    Update database
```

Algorithm 6.1: **Pseudo code**

6.4 BCA-CDN Description of Projects and Scenarios

The performance of the proposed algorithm was evaluated using two projects. Each project was presented with two scenarios as showed in Figure 4a and 4b. The first project (BCA-CDN Project 1) was modelled with twenty mobile phones in two groups of ten mobile phones each. Two scenarios (Scenario 1 and Scenario 2) were created: Scenario 1 is modelled with typical CDN-based IPTV network, while Scenario 2 is modelled with the proposed algorithm. The second project (BCA-CDN Project 2) was modelled with fifty mobile phones in five groups of ten mobile phones each. Two scenarios (Scenario 3 and Scenario 4) were created: Scenario 3 is the typical CDN-based IPTV network setup, while Scenario 4 is modelled with the proposed algorithm.

Live Winter Olympics video streaming data was captured using VLC software [73] during the 2014 Winter Olympics. The captured data was used for simulation in Riverbed Modeler to make the simulation as close to a real-life situation as possible. The summary of simulation parameters is given in Table 1.

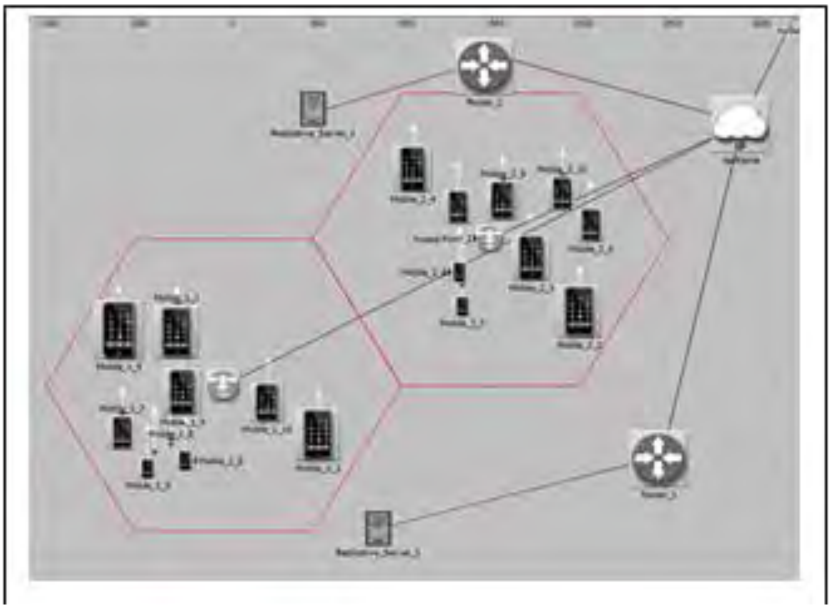


Figure 6.3: Network Topology Project 1

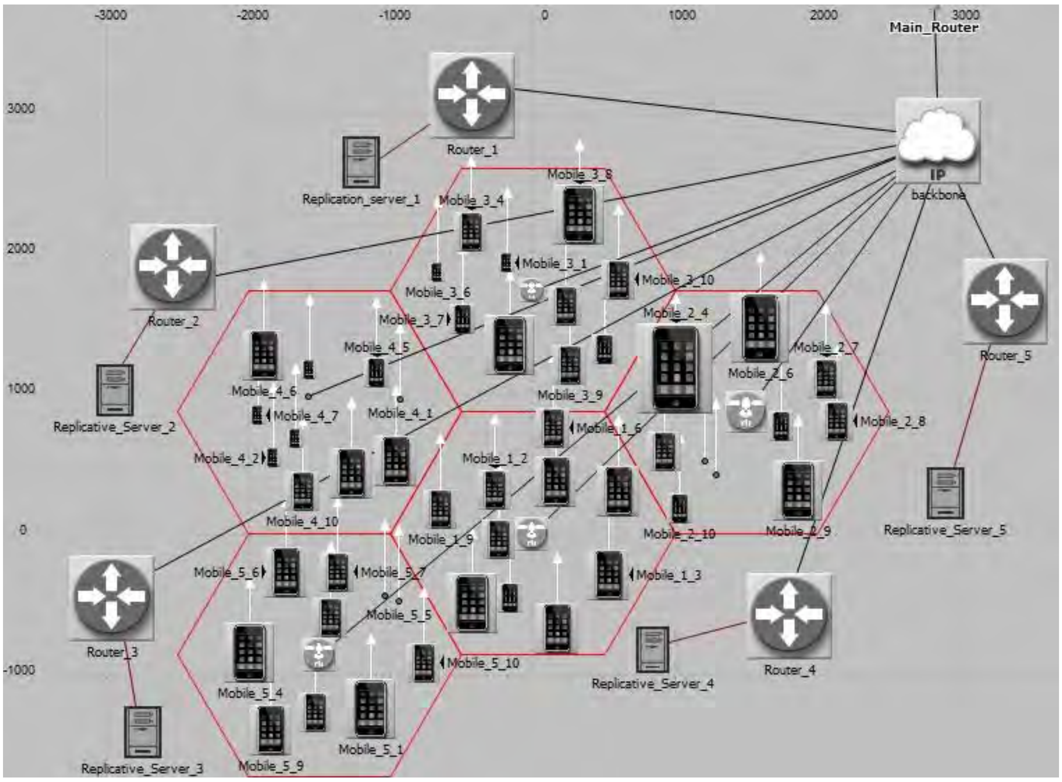


Figure 6.4: Network Topology for Project 2

Configuration parameter	Value
Wireless technology	802.11n
Mobility	Random waypoint
Packet Reception Power Threshold	-95dB
Data rate	6.5 Mbps (base) / 60 Mbps (max)
Start of data transmission	normal (100, 5) seconds
End of data transmission	End of simulation
Duration of Simulation	3600 seconds (1hr)
Packet inter-arrival time	Based on the captured data
Packet size exponential	Based on the captured data

TABLE 6.1: SUMMARY OF NODE CONFIGURATION

6.5 Results Analyses and Evaluations

The QoS and QoE parameters considered in this research work include average end-to-end delay, average packet delay variations, average server loads and throughput. These parameters are defined by [44]: end-to-end delay is the time taken to send the video streaming packets from the source node to the destination node application layer for all the nodes in the network; packet delay variations is the variance among end-to-end packet delays for all the video streaming packets from the source node to the destination node; the average server load represents the average load in bits/sec at the source node at a given point in time; and throughput is the amount of packets that are successfully processed at a given number of times. These parameters were analysed and compared in all the projects and scenarios. The project results analyses are presented below.

6.5.1 BCA-CDN Project 1: Results Analyses and Evaluations

As mentioned in Section 6.4, Project 1 consists of two scenarios, Scenarios 1 and 2. Scenario 1 is the scenario with the normal CDN-based IPTV setup, while Scenario 2 is the scenario with the proposed algorithm. Figure 6.5 shows the average server load for both Scenario 1 and 2. The results show a rapid increase in server load at the beginning of the simulation which stabilises within the first 20 minutes of the simulation time in both scenarios. This is due to the number of consecutive requests received from the clients at the beginning of the simulation. However, Scenario 2 reduced the average server load to 18% in average throughout the simulation. Therefore, the proposed algorithm performed as expected by reducing the server load.

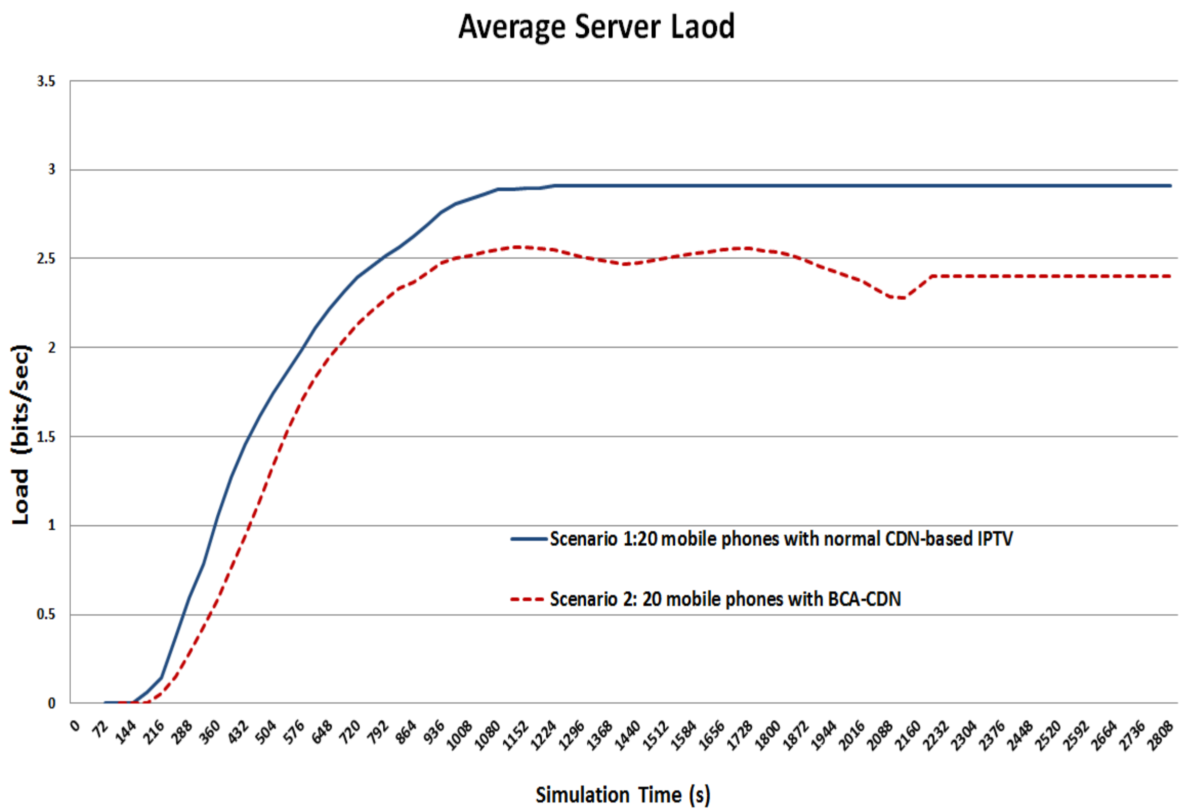


Figure 6.5: Average Server Load

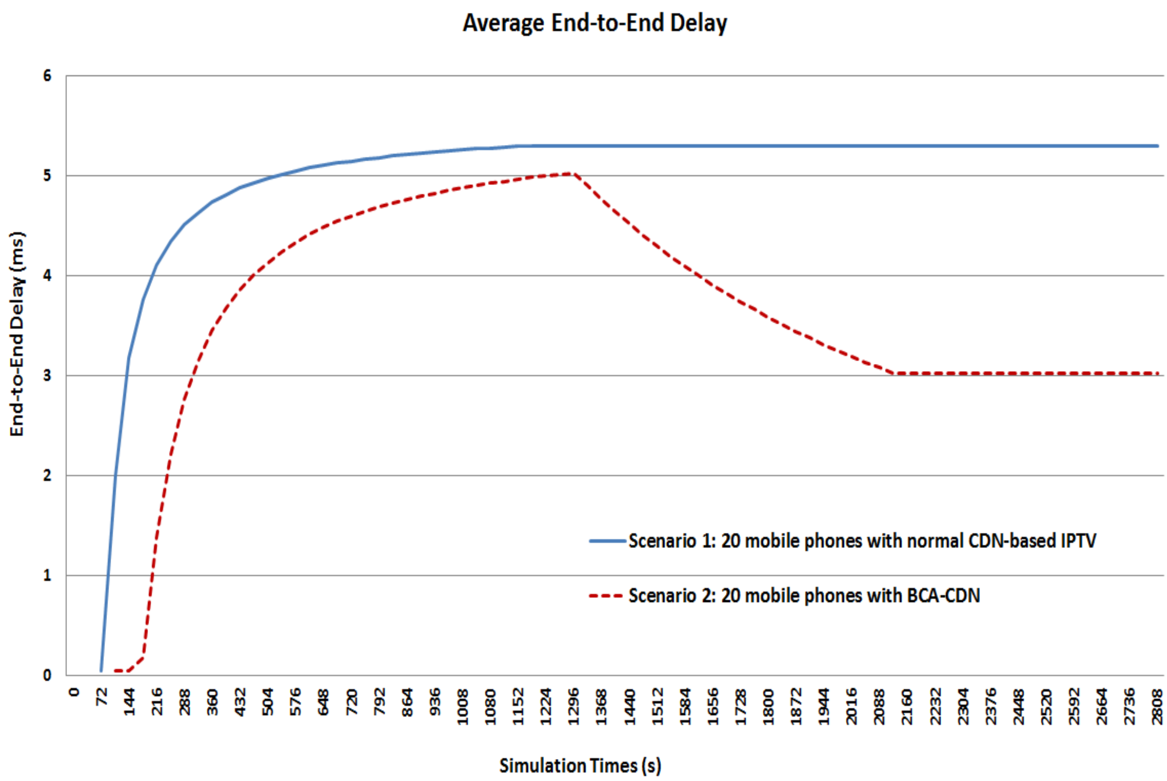


Figure 6.6: Average End-to-End Delay

Figure 6.6 shows the average end-to-end delay for Scenario 1 and 2. The results show that Scenario 2 (with the proposed algorithm) produced less end-to-end packet end-to-end delay compared with Scenario 1 (normal CDN-based IPTV network). Both scenarios produced high end-to-end delays – up to 5 milliseconds in case of Scenario 2 and about 5.4 milliseconds in case of Scenario 1 – within the first 25 minutes of simulation time. However, Scenario 2 showed a rapid decrease of end-to-end delays and maintained the average of 3 miliiseconds after the first 25 minutes up to the end of simulation. Also averagely the proposed algorithm has reduced 21% of the end-to-end delay compared to the normal IPTV. This is due to the fact that the number of clients that had the requested packets increased after the first 25 minutes of simulation, and it is faster to deliver the packets from client to client than from the server, due to their proximity.

Figure 6.7 shows the average delay variations (Jitter) for both scenarios. Scenario 2 produced morethan 50% jitter compared to Scenario 1. This is due to the number of processes executed by the server in-between serving different requests. However, this can be addressed by buffering packets before they are played.

Figure 6.8 shows the average throughputs for Scenario 1 and 2. Both scenarios show high throughput at the beginning of the simulation, which decreases towards the end. This is due to the increase in the number of requests received over time. However, Scenario 2 produced 23% high throughput in average compared to Scenario 1 throughout the simulation, demonstrating that the proposed algorithm performed well.

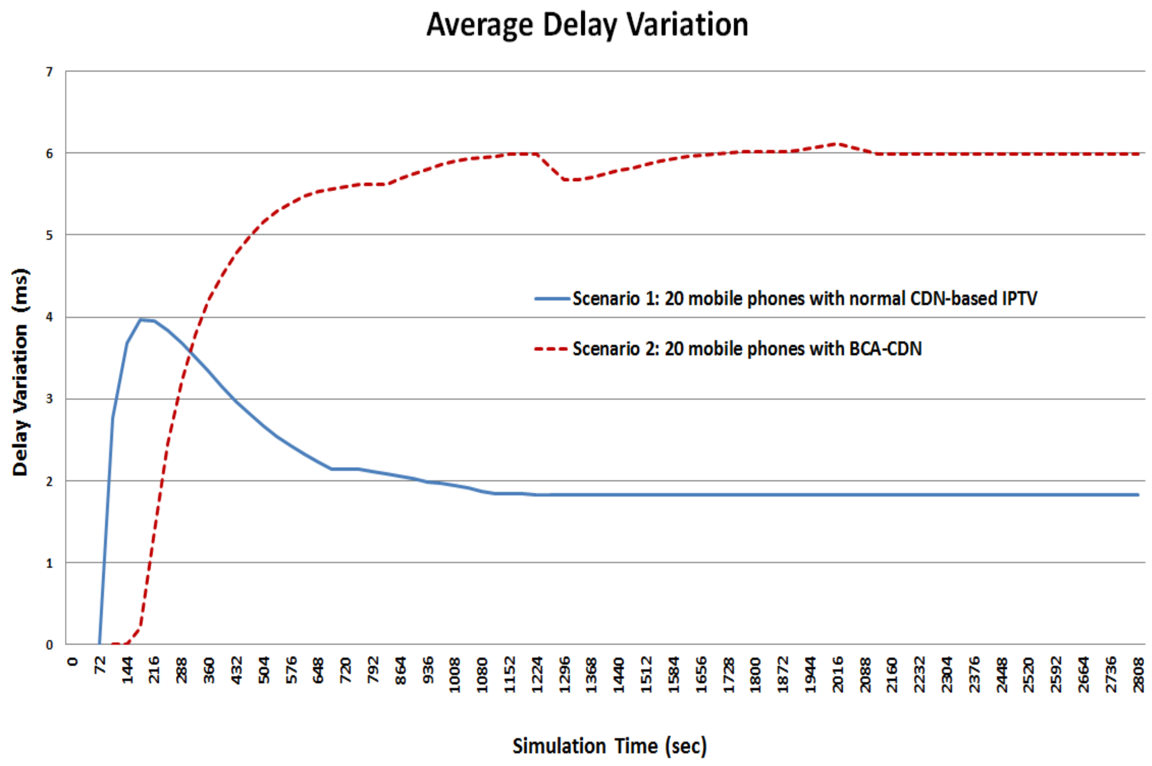


Figure: 6.7: Average Delay Variation

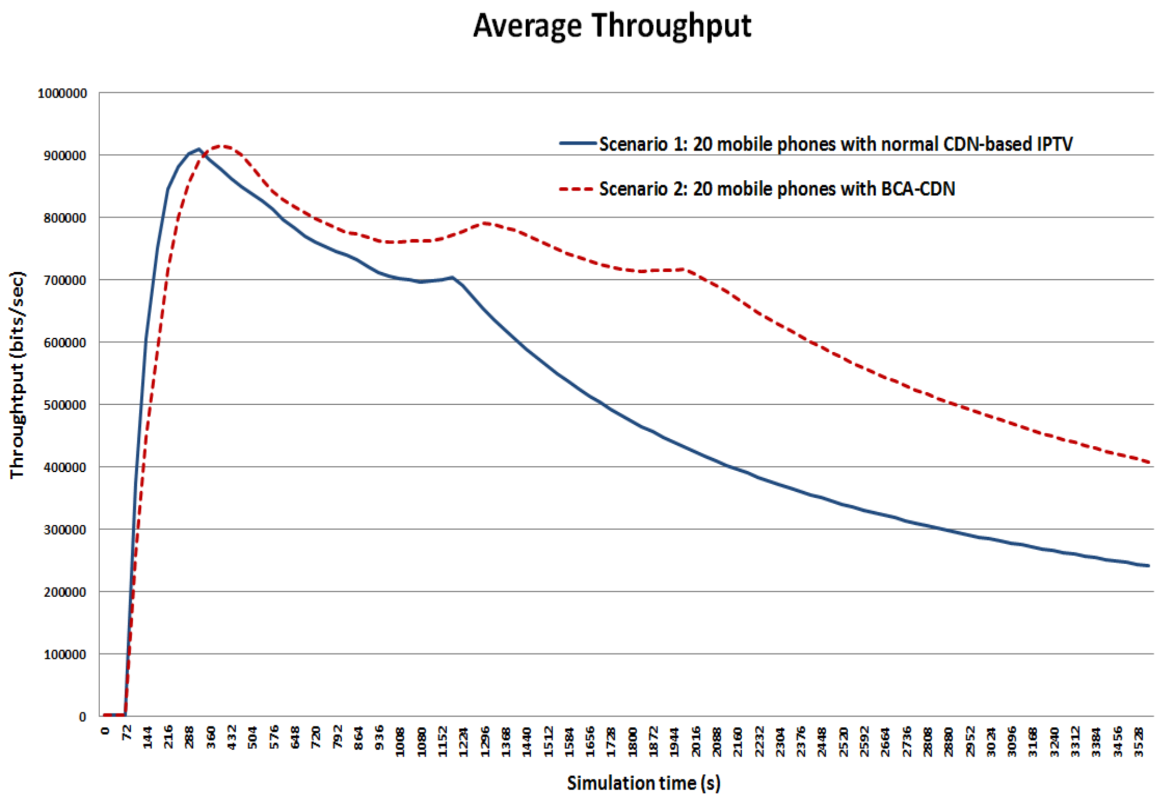


Figure 6.8: Throughput

6.5.2 BCA-CDN Project 2: Results Analyses and Evaluations

As stated in section 6.4, Project 2 was designed with five groups of ten mobile devices each. Two scenarios were created: Scenarios 3 and 4. Scenario 3 is the scenario with the normal CDN-based IPTV network setup, while Scenario 4 is the scenario with the proposed algorithm. Figure 6.9 shows the average server load for both scenarios. Scenario 4 shows significant reduction of server load compared to Scenario 3. The reduction of server load by about 50% to 80% by the proposed algorithm confirms that it works effectively during high service demand. It also shows the provision of unlimited virtual resources by the proposed algorithm. The results also indicate that, the more mobile devices are available on the network, the more effective the algorithm becomes. This is due to the higher number of clients receiving and forwarding the packets in Project 2 than in Project 1.

Figure 6.10 shows the average end-to-end delay for Scenarios 3 and 4. In both scenarios, the end-to-end delay slightly increases within the first 25 minutes of simulation. However, Scenario 4 produced 24% less end-to-end delay compared to Scenario 3 throughout the simulation. It also shows that the end-to-end delay for Project 2 is very low compared with Project 1. This also confirms that the algorithm works well in a time of high demand.

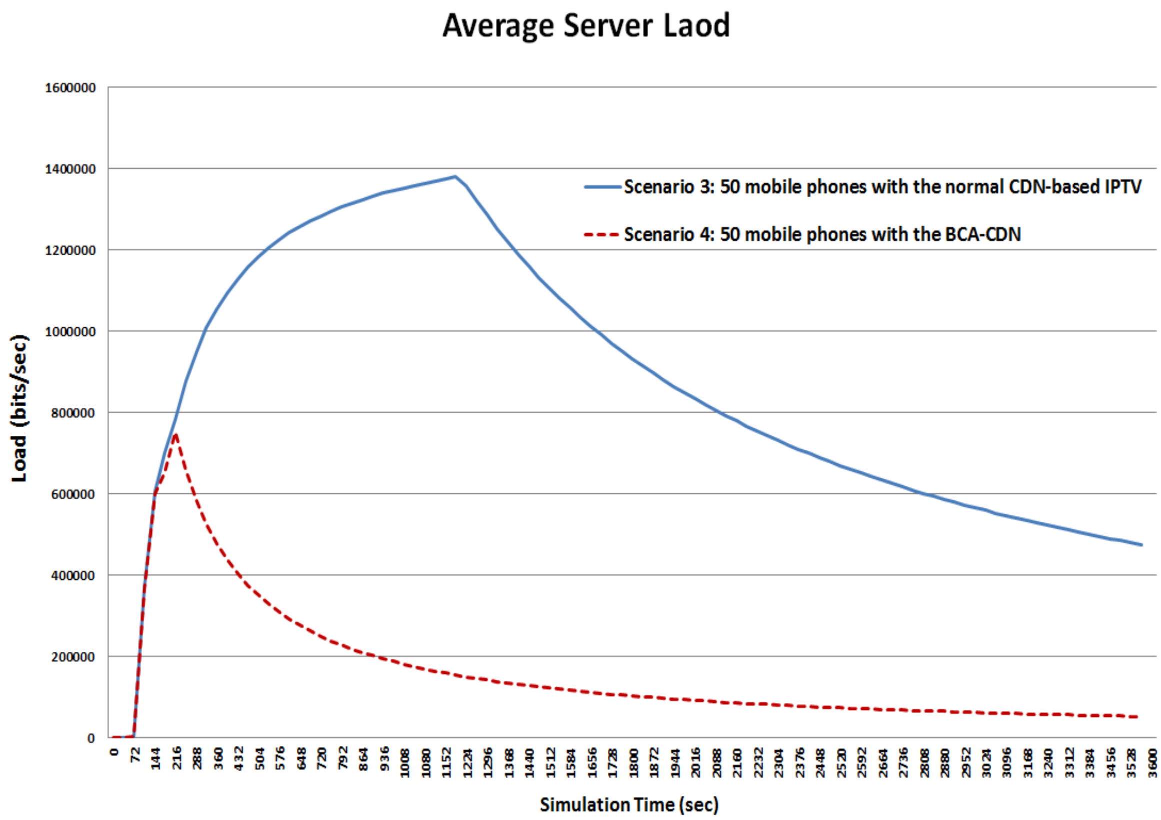


Figure 6.9: Average Server Laod

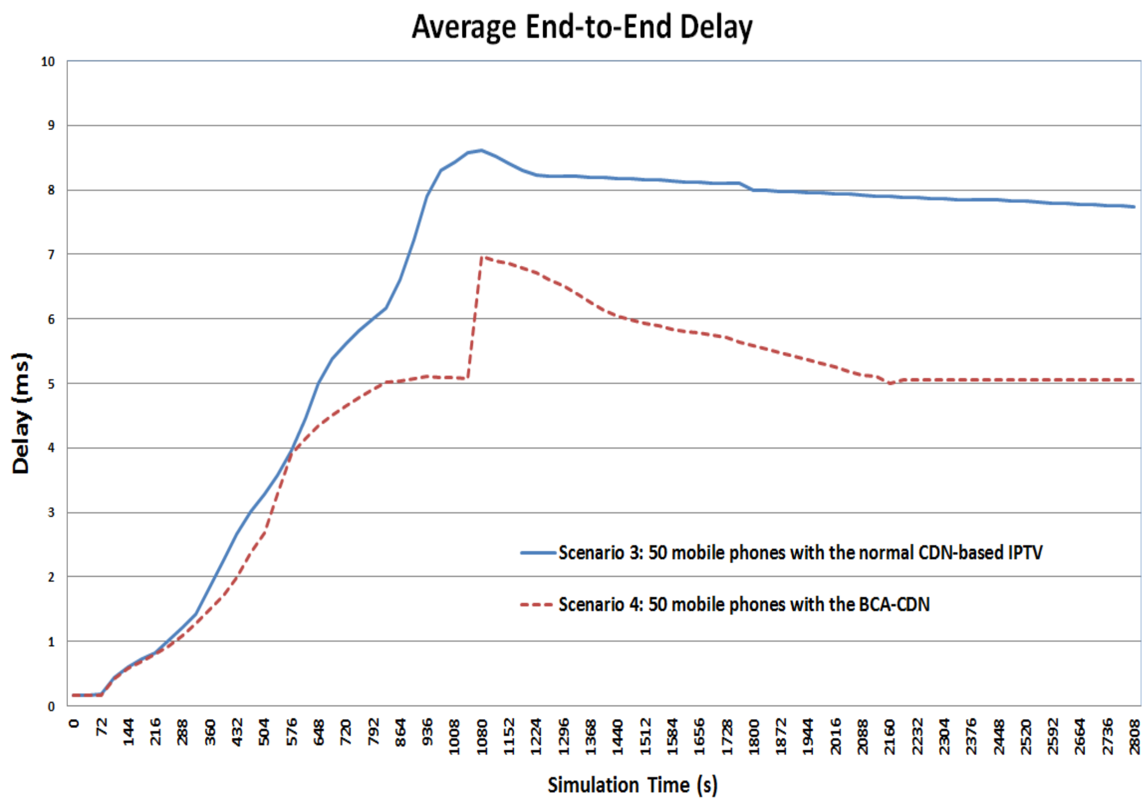


Figure 6.10: Average End-to-End Delay

Figure 6.11 shows the average packet delay variation (jitter) for Scenarios 3 and 4. The results show that a significantly higher amount of delay variation accrued in Scenario 4 than in Scenario 3 to about 28% in average. However, the algorithm should be expected to work effectively if the packets are buffered before being played. Nonetheless, if Project 1 is compared with Project 2, the latter shows improvement in delay variation by reducing up to 48% in average. That also confirms the effectiveness of the algorithm during high demand.

The average throughputs for Scenarios 3 and 4 are shown in figure 6.12. Both scenarios show very high throughput from the beginning of the simulation but a gradual decrease towards the end. This is due to the increase in packet requests over time. However, Scenario 4 shows very high throughput of about 30% in average compared with Scenario 3 throughout the simulation.

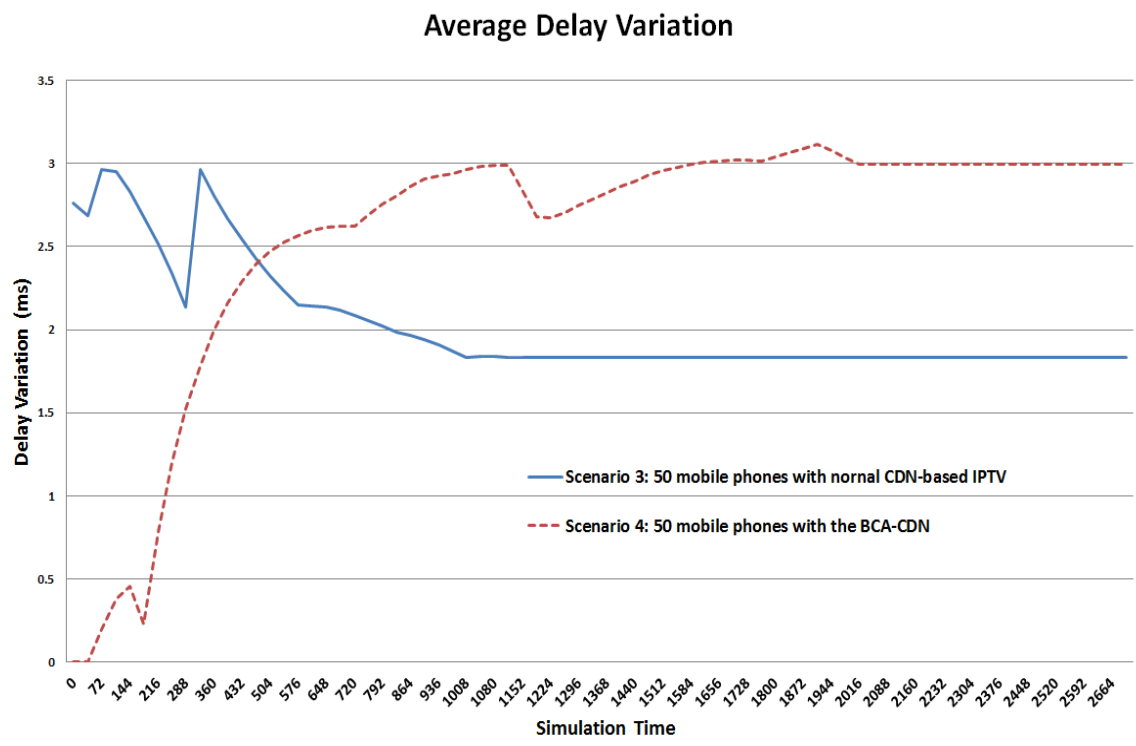


Figure 6.11: Average Delay Variation

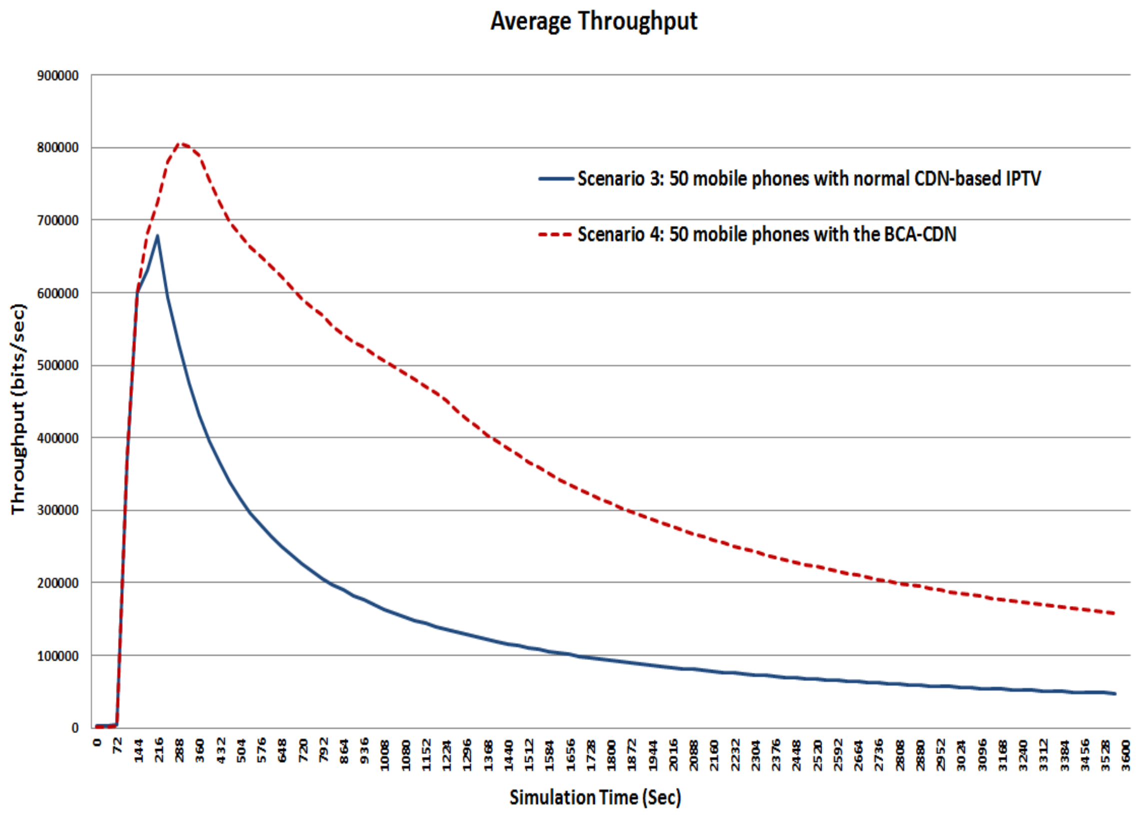


Figure 6.12: Average Throughput

6.6 Conclusion

In this chapter an Adaptive CDN-based Bandwidth Conserving Algorithm for Mobile IPTV is proposed. The proposed algorithm combines the features of IP, MANET and CDN to improve the IPTV QoS. Clients are requested to forward available packets to the requesting client during high service request periods. The proposed algorithm is compared with the normal CDN-based IPTV network, where incoming requests are only served by the servers through IP. The simulations were conducted using live data captured during the 2014 Winter Olympics. Analysis of the simulation results in both projects and sets of scenarios, leads to conclude that the proposed algorithm performed well in requesting some of the clients to forward some of the available packets on behalf of the server during high service demand. It also confirmed that the proposed algorithm outperforms the normal CDN-based IPTV system in server load reduction, high throughput and low amount of end-to-end delay. Despite the average delay variation is relatively high in the scenarios with the proposed algorithm, it is expected to work effectively if the packets are buffered on the client storage before playing. However, the amount of jitter was reduced to its minimum at this stage compared to the previous stages. Hash algorithm will be introduced at the next chapter to provide faster packet searching, which may solve the jitter problem. The results also show that the efficiency of the proposed algorithm increases as the number of clients increases. This confirms that the algorithm reduces server overload during high service request periods by using clients to serve some of the service requests on behalf of the server.

Importantly, the main server is a central point of requests by all the edge/replicative servers for video content that is not available on them. Therefore, when the main server fails, all of the IPTV services fail. In consequence, there is need for an intelligent algorithm that will reassign the responsilities of the main server to one of the edge/replicative servers in the event that the main server fail. This additional functionality of the proposed algorithm is added and discussd in the following chapter.

Chapter 7: Intelligent Routing Algorithm for Mobile IPTV (IRA-MIPTV)

7.1 Introduction

Routing algorithms play significant roles in network path selection. A good routing algorithm should be able to find an optimal path with low overhead, robust, stable and fast convergence. Several routing algorithms were developed for specific kinds of networks and for general routing purposes. These algorithms have been used for different applications depending on their specifications. However, nodes dynamics, bandwidth, server overload, QoS and multicasting issues remain challenging.

Applying intelligence into networking has been one of the fastest growing areas of research [74]. A network should be able to support the increasing level of applications that require co-existence with the existing infrastructure as well as future models. In order to meet the demand for improved networks, the nodes need to be intelligent and capable of making decisions on their own.

Automatic network intelligence was discussed in [75] where highly prevalent studies were applied to make the network intelligent, which is significant to the communication, QoS, security and protocol architecture. In a network perspective, intelligent algorithm is an algorithm that learns and adapts to any network changes that may occur in order to provide continues good quality services. IPTV QoS and QoE are among the major issues concerning service providers. As discussed in the previous chapters, bandwidth consumption, server overload and packet end-to-end delay are among the problems facing IPTV services.

This chapter details the design, implementation and validation of the proposed algorithm called Intelligent Routing Algorithm for Mobile IPTV

(IRA-MIPTV) as a solution to the some of the problems facing IPTV service delivery during high service request demand. IRA-MIPTV is an intelligent algorithm that adapts to different servers’ capacities in order to improve IPTV QoS. The algorithm intelligently learn severs’ capacities such as available bandwidth, load and status in order to elect the main server and the backup server as well as chosen the suitable server or a client to serve a particular service request. The routing decision depends on the number of service requests received at a point in time, available bandwidth and the proximity to the requesting client from the server. Clients are requested to forward some of the packets received from the server to other clients during high demand service requests. The forwarding clients are selected based on the initial agreement reached between clients and service providers. The clients are compensated for the services offered. The IRA-MIPTV is an extension of our work discussed in previous chapters. For its intelligent capabilities, this algorithm combines the use of unicast, multicast, CDN-based architecture, and OSPF election features to improve the mobile IPTV services’ delivery. Figure 7.2 demonstrates the operation of the algorithm.

The algorithm is modelled and incorporated in the servers, where the selection policy is taking place. The selection decision is based on the network information stored in the database. The main/designated server performs the selection process by choosing an appropriate server to serve an incoming service request. The servers may also request clients to forward packets to other clients during high demand period. To validate the efficiency of the algorithm, Riverbed Modeler 18.0 was used to simulate the network setups. Figure 7.1 shows the network topology model in Riverbed. The simulation results analysed show that the proposed algorithm provides not only unlimited virtual bandwidth but also reduced significant server load, end-to-end delay and jitter as well as increased throughput which, in turn, improves the general quality of service.

Figure 7.1: Network Topology

7.2 Dynamic Routing Protocol

For an algorithm to become adaptive, it must be able to handle the dynamic changes that may occur in the network setup. Several dynamic routing protocols were adopted to automatically adjust to the different network circumstances. Open Short Path First (OSPF) is one of the dynamic routing protocols that are widely used. It is a link state routing protocol that uses Dijkstra's Shortest Path First Algorithm [74], as illustrated in Algorithm 7.1. OSPF uses election process to elect a Designated Router (DR) and the Backup Designated Router (BDR) on each multi-access segment. The BDR is elected as a backup mechanism in case the DR goes down. The idea behind electing DR and BDR is that routers have a central point of contact for information exchange. Instead of each router exchanging updates with every other router on the segment, every router exchanges information with the DR and BDR. The DR and BDR relay the information to other routers. This reduces the amount of information exchange between all other routers in network. Mathematically it can be express as

$$\mathbf{X(n*n)\ to\ X(n)}$$

Where n is the number of routers on a multi-access segment.

Similarly, the proposed algorithm uses the concept of OSPF election process, where the Designated Server (DS) and Backup Designated Server (BDS) are selected in a multi-server access network. The idea is that, between all the servers in the network, DS and BDS are elected so that DS is responsible for deciding which server will be serving request depending on the QoS parameters under consideration. The DS also serve as a central contact between other servers. The BDS is elected as a backup in case the DS goes down.

Dijkstra Shortest Paths (*G,v*)

Input: Simple undirected weighted graph *G* with nonnegative edge weights and a distinguished vertex *v* of *G*

Output: A label *D[u]*, for each vertex *u* of *G* such that *D[u]* is the distance from *v* to *u* in *G*

D[v] ← 0

for each vertex $u \neq v$ of *G* **do**

D[u] ← + ∞

Let a priority queue *Q* contain all the vertices of *G* using the *D* labels as keys

while *Q* is not empty **do**

{Pull a new vertex *u* into the cloud}

u ← Q.removeMin()

for each vertex *z* adjacent to *u* such is in *Q* **do**

{perform the ***relaxation*** procedure on edge (*u,z*)}

if *D[u] + w((u,z)) < D[z]* **then**

D[z] ←D[u] + w((u,z))

Change to *D[z]* the key of vertex z in Q.

Return the label *D[u] of each vertex u*

Algorithm 7.1: Dijkstra's algorithm for the single source shortest path problem for a graph *G,* starting from a vertex *v* [75].

7.3 Related Study

Several researches have been conducted in this area by academics and service providers to improve the quality of service and quality of experience in IPTV. Many of these researches were discussed in Chapter 2, however, other related studies are as follows: A scalable and reliable

IPTV service through collaborative dispatching protocol was proposed by [76]. In this study, they used station-wise collaboration service to route service request to appropriate stations with regards to cost-effectiveness and load distribution. The results obtained in this study show that the collaborative dispatching protocol using iCloud improves the performance and scalability of on-demand IPTV. However, IPTV services comprise both on-demand and multicast scheme but the proposed protocol only considers on-demand which is usually delivered using unicast. The IPTV multicast has been left out.

Also in another study proposed by [77], the approach is to perform residual bandwidth optimization with QoS guarantee to support IPTV services on the Internet backbone. The proposed approach combines the use of genetic algorithm with variable neighbourhood search and bandwidth estimation. The experimental results of the approach were compared with OSPF in term of link utilization distribution. However, the network architecture considered in this approach does not consider mobile devices. Thus, the assurance of IPTV service to deliver to end users at anytime and anywhere is compromised. Also in another study presented in [78], they revisits their proposed approach above for alteration for better result. Yet, no mobile aspect of the IPTV is considered as no mobile devices are used in the approach.

A study carried out by [79], proposes multi-path routing algorithm with bandwidth and delay constraints. They propose a heuristic algorithm and presented a polynomial algorithm to obtain a time approximation for the reliable multipath routing. However, the study analyses and results are based on time estimation using heuristic and polynomial time approximation algorithm. Several network QoS parameters such as server capacity, available bandwidth, proximity and network traffic were not considered.

A SDN/OpenFlow application for IPTV multicast is proposed by [80]. The experiments compare the transmission time of the first joint/receive packet to a client when using Dijkstra's and Prim's algorithms. However, as they made mention in their conclusion, the study is a starting point and is limited to only IPTV multicast service.

In all the studies discussed above the algorithms lack the intellectual ability for the nodes to automatically adapt to network situation and did not cover all the services offered by IPTV.

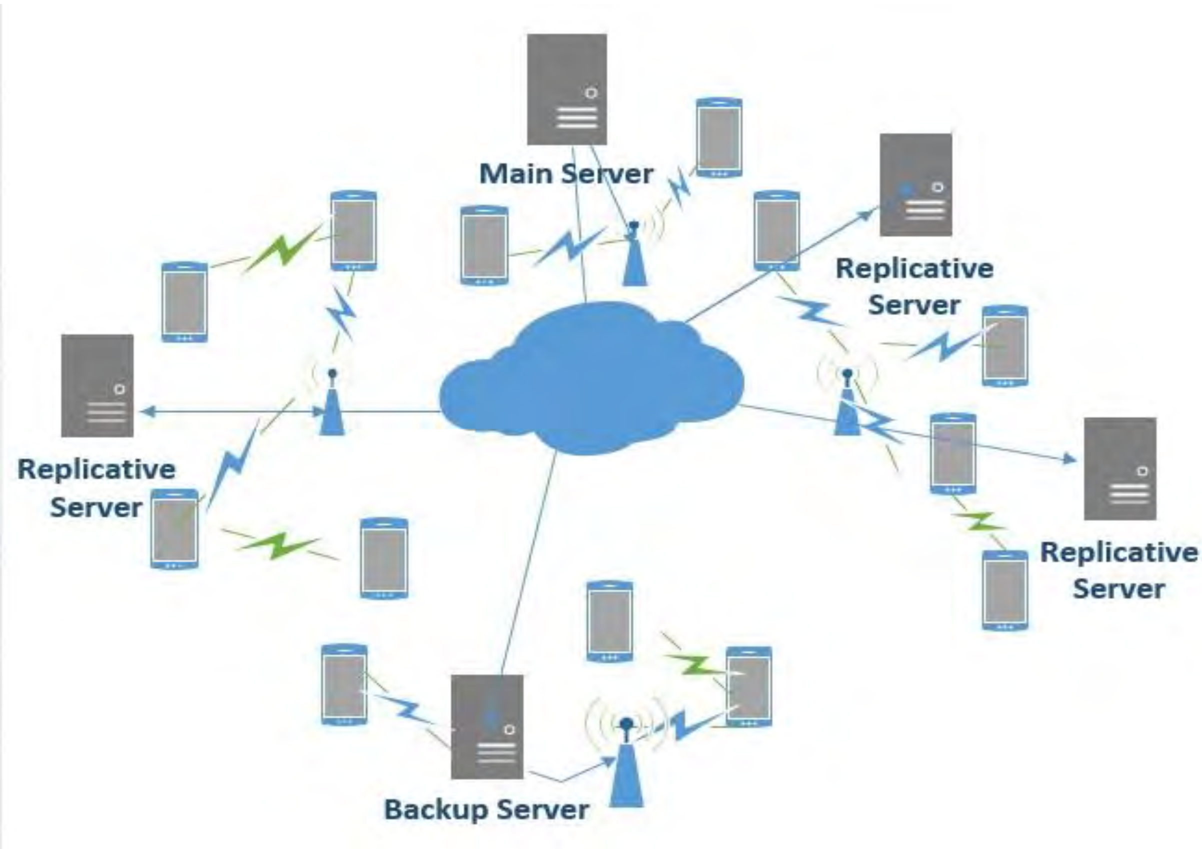


Figure 7.2: Intelligent Routing Algorithm for Mobile IPTV (IRA-MIPTV)

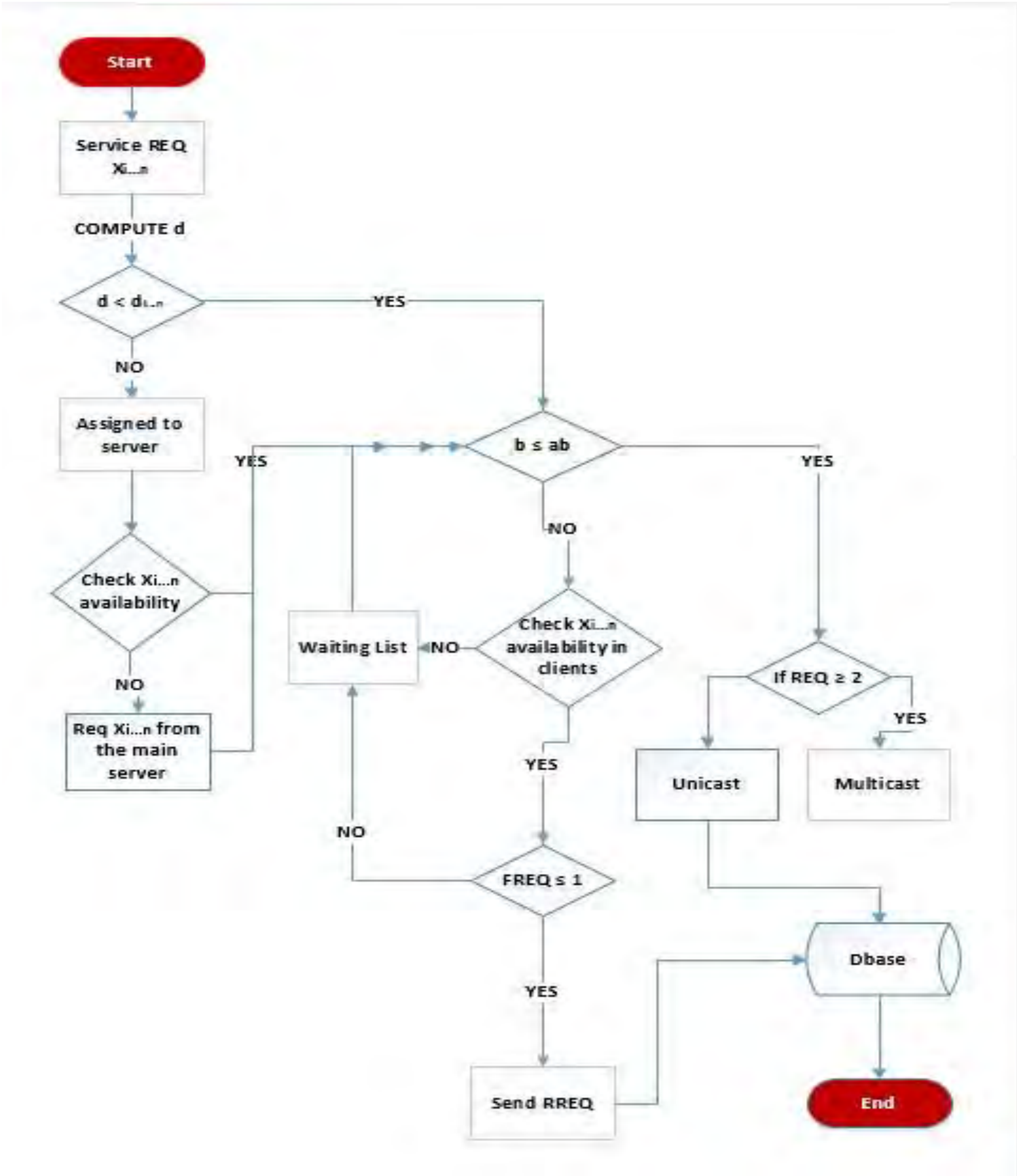


Figure 7.3: IRP-MIPTV Flowchart

7.4 Cryptographic Hashing

The cryptographic hashing is a function of exchanging an arbitrary block of data with a fixed-size "hash" value. The data can be of any size and type but the hash value size is always the same. The uses of cryptographic hashing include content verification and file identification. The use of file identification provides the ability to quickly find data in hash tables, a method commonly used by search engines. Message Digest algorithm 5 (MD5) is among the cryptographic hashing algorithm that is widely used for data integrity check[81].

It was invented by Professor Ronald Rivest in 1991 and is the third message digest algorithm created by him after MD2 and MD4. It is currently a standard, as stated by Internet Engineering Task Force (IETF) Request for Comments (RFC) 1321[82]. The MD5 algorithm generates a fixed 32 digits hexadecimal hash value for any file, regardless of it is type and size. Figure 7.4 shows an example of MD5 hash function.

MD5 hashing algorithm was used in the designing of the proposed algorithm to provide data integrity check and faster packet searching. The hash values of all the packets served are stored in a hash table in the server's storage, and it was in the packet searching.

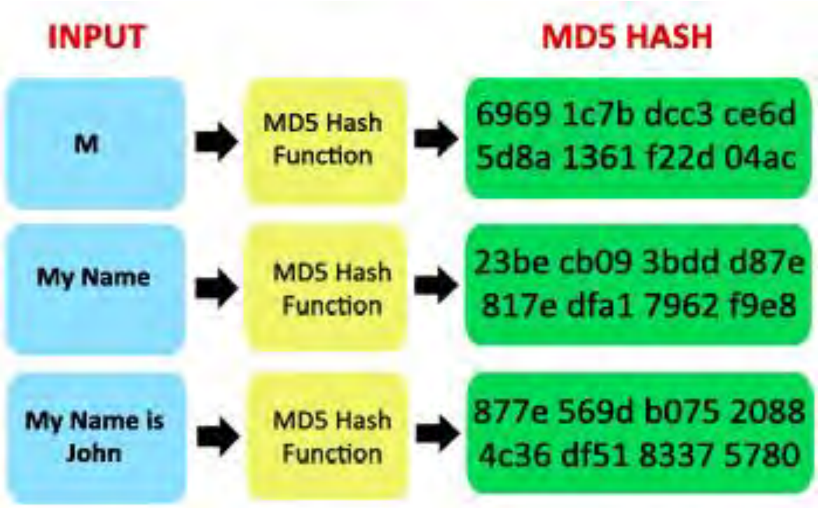


Figure 7.4: MD5 Hash Function

7.5 Description of the Proposed Algorithm

The IRA-MIPTV is an intelligent algorithm aimed at providing the expected level of QoS in IPTV services, especially during high demand requests. The servers use the algorithm to intelligently learn the server's available bandwidth, load and status to elect designated server (DS) and Backup Designated Server (BDS) as well as selecting the best server or client to serve an incoming service request depending on the number of requests received at a point in time. The decision made by the DS is to select and reroute a request to the most appropriate server. While the edge/replicative server select and requests a client to forward packet if need arise. Figures 7.3 and Algorithm 7.2 illustrate the flowchart and pseudo code of the proposed algorithm. Its operations are explained in the following steps:

Step 1: The main server receives service request(s) from client node(s)

Step 2: It calculates and compares the distances from the client(s) to all servers in the network and selects the lowest

Step 3: It assigns the request(s) to the server with the lowest distance

Step 4: The server checks the packet availability. If it is available, ***go to*** step 6

Step 5: If the packet(s) is not available, it requests from the main server

Step 6: If the main server's distance is the lowest then it compares the bandwidth needed to serve the request with its available bandwidth

Step 7: If the requested bandwidth is lower than or equal to the available bandwidth, it checks the number of requests at a point in time

Step 8: If there are two or more requests, it groups the clients into a multicast group and serves the requests as multicast

Step 9: If there is one request, it uses unicast to serve the request

Step 10: If the requested bandwidth is higher than the available bandwidth, it checks the availability of the packet(s) in any of the clients

Step 11: It first checks the group of the requesting client before checking other groups. If the packet is available in any of the clients, then it checks the number of Forward Requests (FREQ) on that client

Step 12: If the number of FREQ is less than or equal to 1, it sends a Re-Route Request (RREQ) message to that client

Step 13: Update database

Step 14: If the packet is not available in any of the clients, or the client that has the packet has more than 1 FREQ messages, it puts that request on the waiting list

Step 15: Set a flag for available bandwidth

Step 16: If the flag is on, serve the next client on the waiting list by repeating step 4.

7.6 IRA-MIPTV Election Process

The election of Designated Server (DS) and Backup Designated Server (BDS) is conducted via Hello messages. Hello packets are exchanged via IP multicast packets in the network area. The server with the highest available bandwidth is elected as the DS. The same process is repeated for the BDS. In a case of a tie, the server with the highest throughput is elected.

There are three steps for DS and BDS election process:

Step 1: A server starting the IRP-MIPTV process listen to IRP hello messages, if none is received within the dead timer, it declare itself the DS. The default dead time is 30 seconds.

Step 2: If hello messages from other servers are received, the server with the highest available bandwidth is elected as DS, and election process start again for BDS.

Step 3: If two or more servers have the same available bandwidth, the server with the highest throughput is elected as DS, and the election process start again for BDS. After a DS and BDS are elected, election does not take place again unless the DS or BDS is down.

Input: *distance $d_{i\dots n}$ from client to the servers*
$(\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$
Output: *Select server S_i; RREQ*

Procedure:

```
REQ of Packet xi…n
 While server status = DS
   Get d [d1…n ]
   bandwidth [ab]
         If  d < di
               update DBase
         Else
               select Si
         While  b ≤ ab
               if  xi in  j[ ] = available && FREQ ≤ 1
                     send RREQ
               Else
                     assign REQ →Wlist
         EndWhile
   Serve REQ
   Update Dbase
EndWhile
```

Algorithm 7.2: Pseudo code for IRP-MIPTV

7.7 IRA-MIPTV Description of Projects and Scenarios

To analyse the performance of the proposed algorithm, two different projects were created; Project 1 and Project 2. The projects were created to test the effectiveness of the proposed algorithm as the number of clients increase. Project 1 was created with small number of devices while Project 2 was created with number of devices twice than that of Project 1. The projects and scenarios are explained as follows:

7.7.1 IRA-MIPTV Project 1

This project is modelled with 30 mobile devices, consisting of 10 devices in each group. Figure 7.5a illustrates the project model, where a main server and 3 edge/replicative servers were used. Several simulations were conducted with number of client's ranges from 10 to 100, where best results were chosen. At this project, 30 mobile devices are considered, because it gives meaningful results that represent small number of clients. The project is aimed at testing the efficiency of the proposed algorithm with small number of clients.

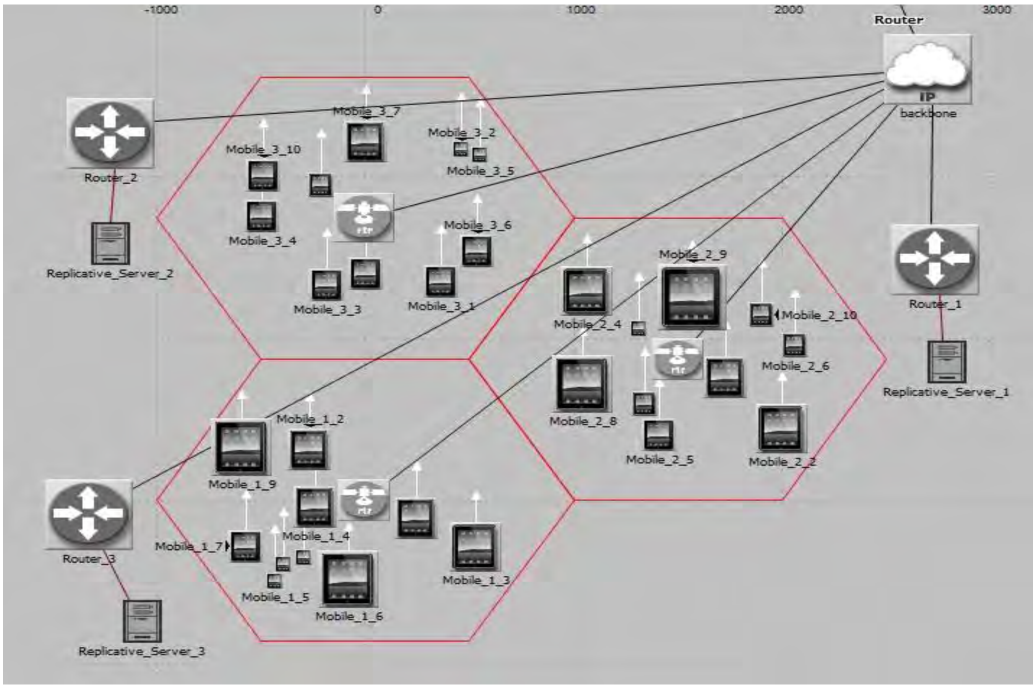


Figure 7.5a: Project 1 Network Topology

7.7.2 IRA-MIPTV Project 2

This project is modelled with 70 mobile devices, consisting of 7 groups with 10 devices in each. Figure 7.5b illustrates the project model. Similarly, a main server and 7 replicative servers were used in this project. Several simulations were also conducted with number of client's ranges from 10 to 100, where best results were chosen. At this project, the highest number of devices that the simulation was successfully completed is 70.

Thus, 70 mobile devices are considered. The difference between Project 1 and Project 2 is simply the number of devices used. This is to evaluate the efficiency of the algorithm as the number of clients increase in the network.

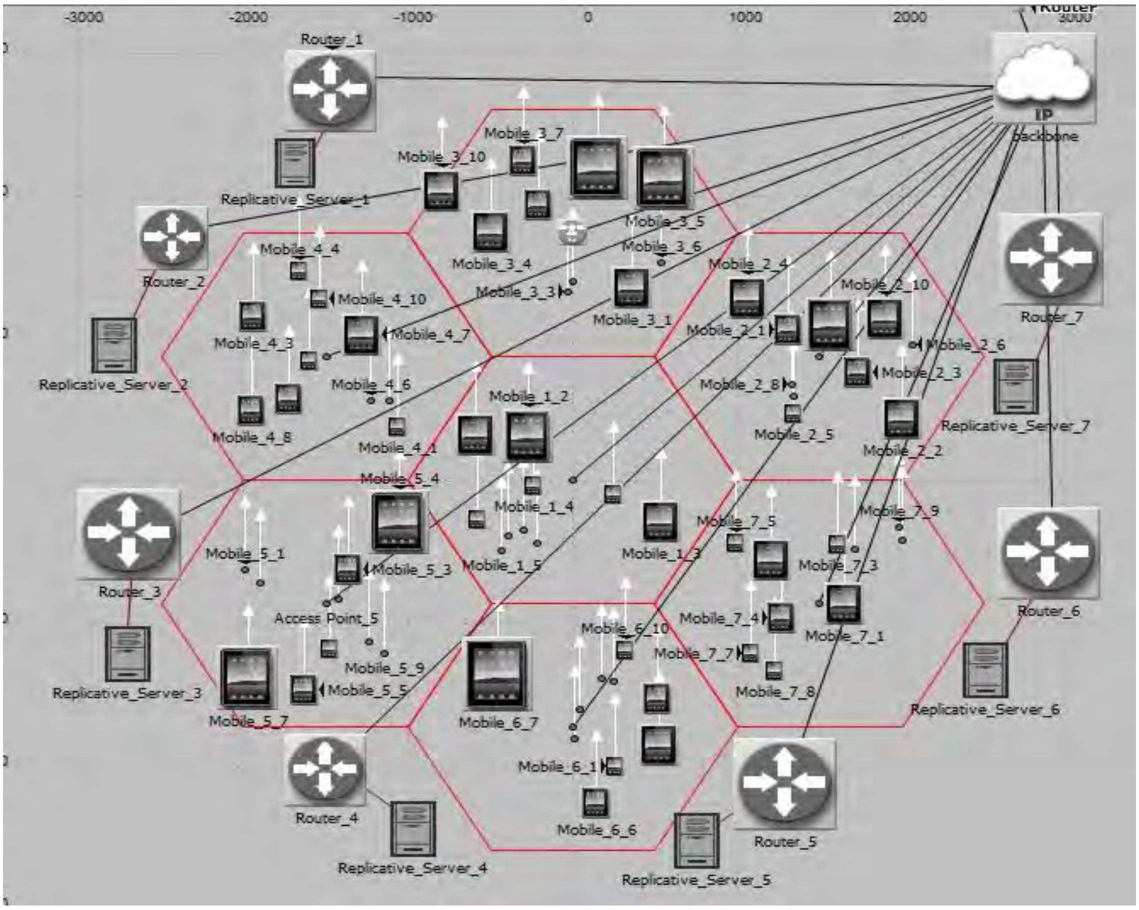


Figure 7.5b: Project 2 Network Topology

7.7.3 Scenarios

Four scenarios were created, compared and evaluated in both projects. The scenarios were created to test the efficiency of the algorithm in different circumstances. The descriptions of the scenarios are as follows:

Scenario 1: Normal CDN-based IPTV setup.

Scenario 2: The proposed algorithm (IRA-MIPTV), but the designated server deliberately crashed during simulation

Scenario 3: Proposed algorithm with no backup server elected prior to the designated server failure

Scenario 4: Proposed algorithm with the backup server elected before the designated server failed.

In both projects, live winter Olympics video streaming data were captured using VLC software [53]. The data were captured during the Winter Olympics games in 2014. The data were used in the Riverbed Modeler simulation environment to make the simulation as close to real-life situation as possible. The summary of simulation parameters is given in Table 7.1 below.

Configuration parameter	Value
Wireless technology	802.11n
Mobility	Random waypoint
Packet Reception Power Threshold	-95dB
Data rate	6.5 Mbps (base) / 60 Mbps (max)
Start of data transmission	normal (100, 5) seconds
End of data transmission	End of simulation
Duration of Simulation	30 minutes
Packet inter-arrival time	Based on the captured data
Packet size exponential	Based on the captured data

TABLE 7.1: SUMMARY OF NODE CONFIGURATION

7.8 Results Analyses and Evaluations

Real-time applications, such as video and audio, are sensitive to delay and jitter and require high amounts of bandwidth and throughtput for successful service delivery. In order to have a good evaluation of the proposed algorithm, the QoS parameters considered include packet end-to-end delay, packets delay variations (jitter), server loads and throughput. These parameters were defined by [44]; the end-to-end delay is the time taken to send the video streaming packets from the source node to the destination node application layer for all the nodes in the network. The packets delay variations is the variance among packets end-to-end delays for all the video streaming packets from the source node to the destination node. The server load represents the load in bits/sec at the source node at a point in time. Throughput is the amount of packets that are successfully processed in a given amount of time. These parameters were analysed and compared in all the projects and scenarios.

As explained in the previous section, in both projects, four scenarios were created, tested, compared and analysed to assess the level of effectiveness of the proposed algorithm. The scenario comparisons are detailed as follows:

Scenario 1: Normal CDN-based IPTV

Vs

Scenario 2: Proposed algorithm but the designated server deliverately crashed during simulation

At this stage, the sever was deliverately crashed to test and evaluate the efficacy of the proposed algorithm and to show the limitations of the algorithm when the server fails, compared with the normal CDN-based IPTV.

Scenario 1: Normal CDN-based IPTV

Vs

Scenario 4: Proposed algorithm with backup server elected prior to the designated server breakdown

This is to evaluate the adaptiveness of the algorithm by automatically assigning the role of failed designated server to the backup server, which then becomes the new designated server, compared with the normal IPTV.

Scenario 3: Proposed algorithm with no backup server elected prior to the designated server failure

Vs

Scenario 4: Proopsed algorithm with backup server elected prior to the designated server failure

This comparison is to check the overhead that may occur during the server election and re-assigning process.

The evaluation and results analyses for the projects are as follows:

7.8.1 IRA-MIPTV Project 1: Results Analyses and Evaluation

As earlier mentioned in section 7.7.1, several simulations were conducted with number of client's ranging from 10 to 100, where best results were chosen. At this project, 30 mobile devices are considered, because it gives meaningful results that represent small number of clients. The four scenarios created as explained in the previous section were used to test the efficacy of the proposed algorithm using the quality of service parameters under consideration.

Scenario 1 vs Scenario 2

The packet end-to-end delay results for Scenario 1 (i.e. normal CDN-based IPTV setup) and Scenario 2 (i.e. proposed algorithm with the designated server intentionally crashed during simulation) are presented in Figure 7.6. The results are compared, analysed and evaluated. It shows that, the proposed algorithm reduced averagely 9% of packet end-to-end delays compared to the normal IPTV. The highest end-to-end delay produced by the proposed algorithm is 7.4ms while the normal IPTV produced up to 8ms. Therefore, the proposed algorithm performed well in reducing the packet end-to-end delay compared to normal IPTV, which in turn improved the general QoS. However, the server crashed at exactly 414 secs of the simulation time. As soon as the designated server crashed, all the services failed. As such there is a need for the algorithm to intelligently handle such unforeseen circumstances. Thus, this problem has been addressed and explained in Scenario 4.

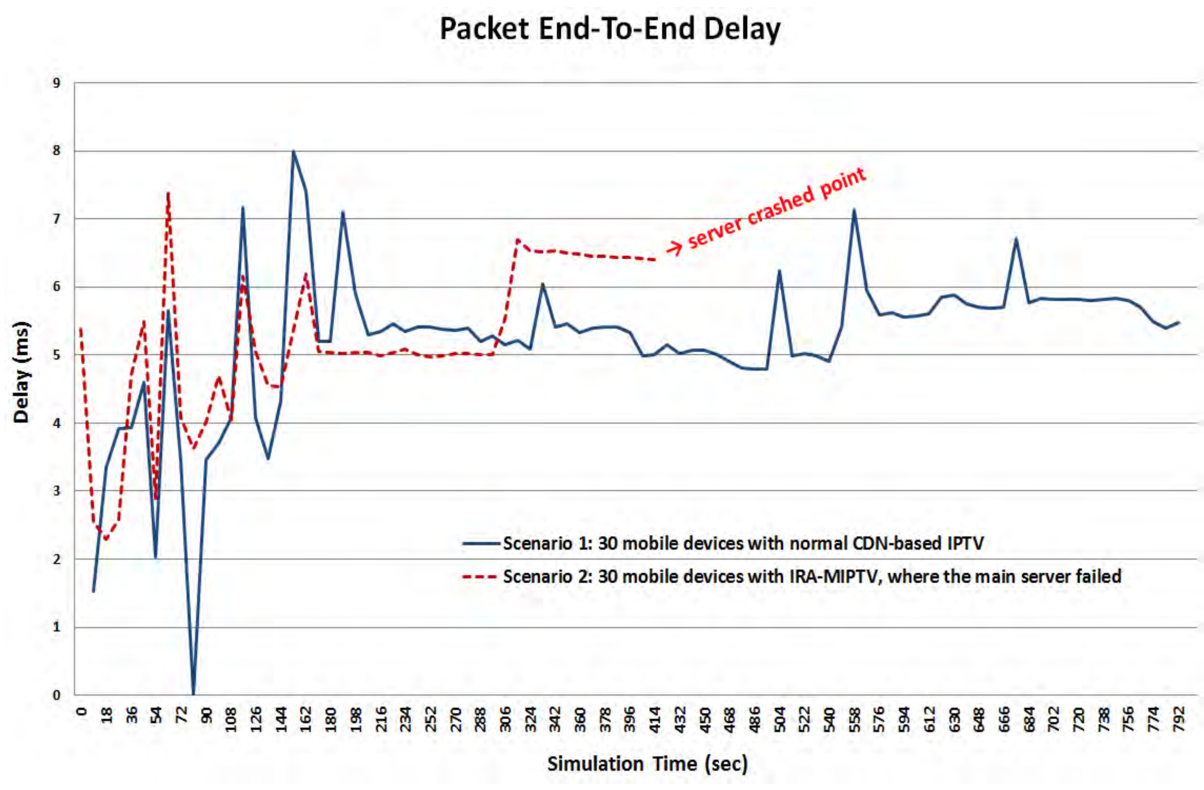


Figure 7.6: Packet End-to-End Delay

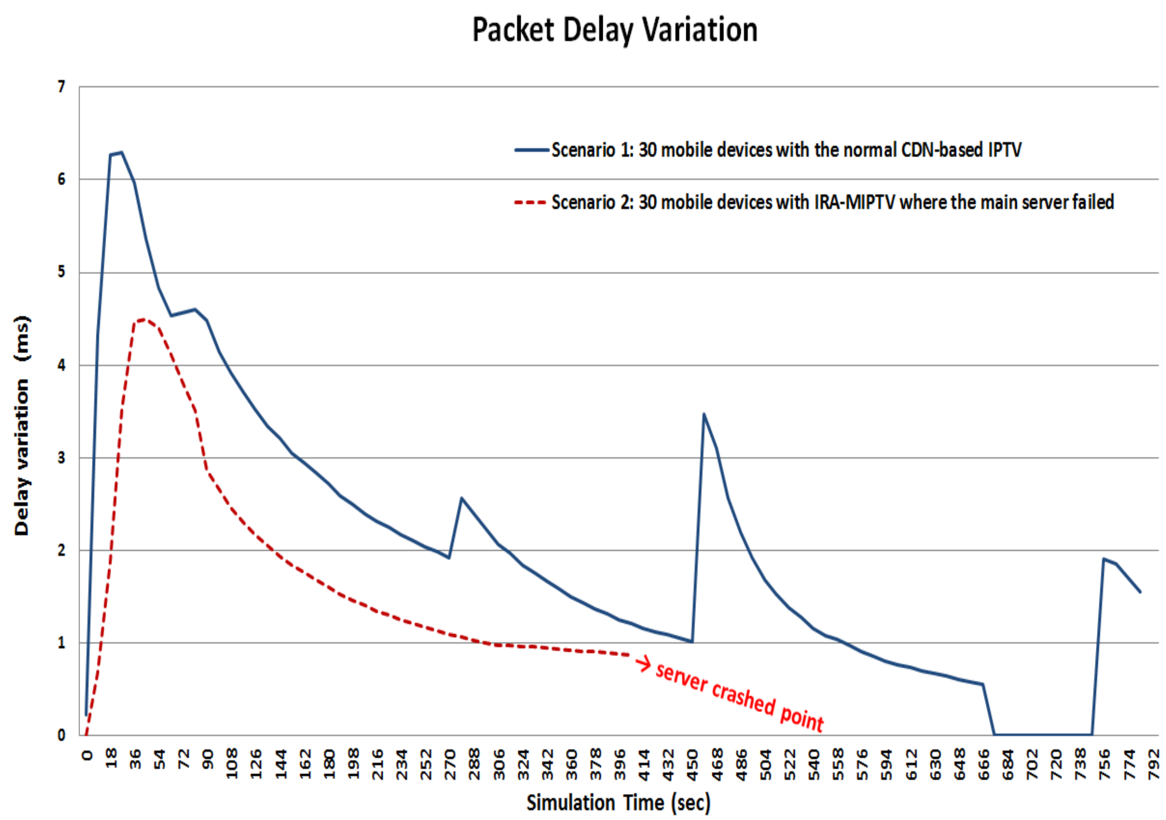


Figure 7.7: Packet Delay Variation (jitter)

The packet delay variation (jitter) results for Scenario 1 and Scenario 2 are shown in Figure 7.7. Scenario 1 is the normal CDN-based IPTV while Scenario 2 is the proposed algorithm with the designated server deliberately crashed during simulation process at exactly 414 secs of the simulation time. The results show that the proposed algorithm reduced averagely 18% of jitter compared to the normal IPTV up to the time when the server crashed. The highest amount of jitter produced by the proposed algorithm is about 4.5ms while the normal IPTV produced up to 6.2ms. Jitter is sensitive to real-time application, particularly video application. High amount of jitter produces poor video QoS and QoE to the end users. This means that the proposed algorithm performed well in improving the general quality of service. Thus, the failure of the designated server rendered the algorithm unusable. Therefore, the algorithm should provide ways where devices can automatically adapt to unforeseen situations and be able to find alternative ways of providing services at all times. This has been dealt with in Scenario 4.

Server load is the representation in bits/sec of the total server load at a point on time. Figure 7.8 show the server load results for Scenario 1(normal CDN-based IPTV) and Scenario 2 (proposed algorithm with sever deliberately crashed). The results show that the proposed algorithm reduced up to about 25% of the total server load compared to the normal IPTV. This shows that the proposed algorithm outperformed the normal IPTV service in reducing server overload, which resulted to good quality of service. However, the server crashed at exactly 414 secs of the simulation time which rendered the algorithm inefficient. Therefore, the algorithm has to be intelligent enough to address the issue of server failure that may occur. Thus, Scenario 4 has taking that into consideration.

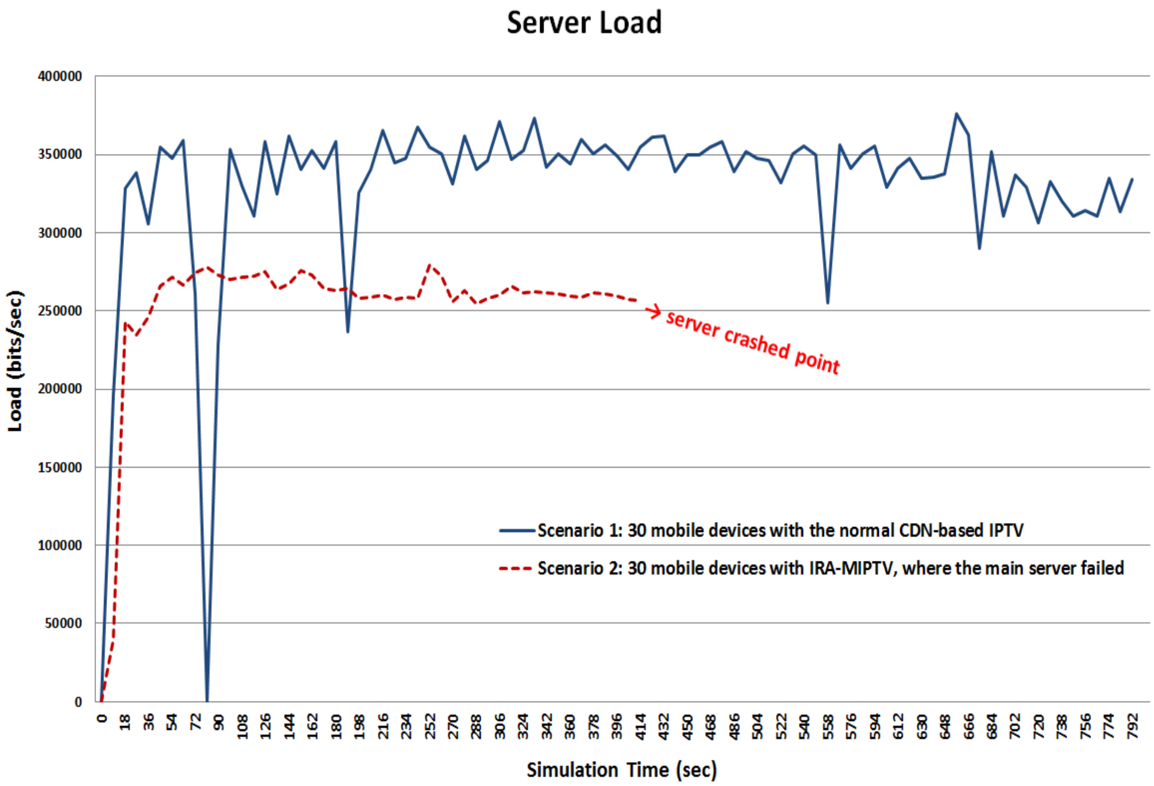


Figure 7.8 Server load

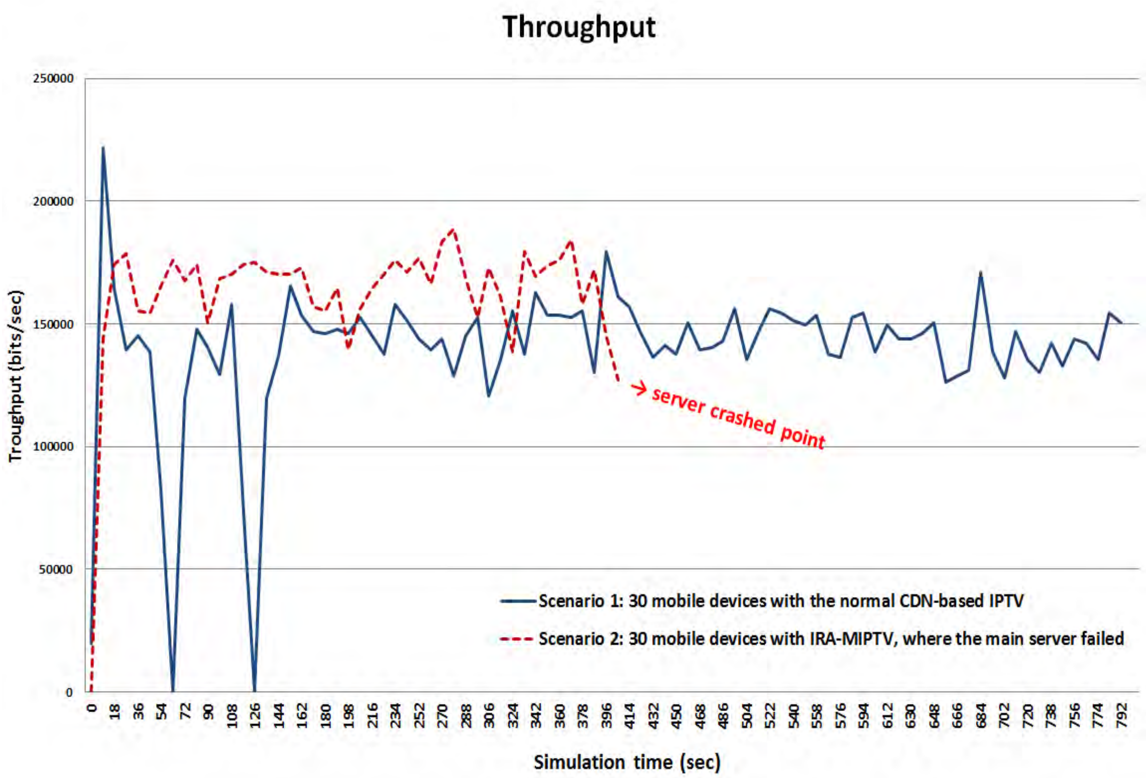


Figure 7.9 Throughput

Throughput is also considered in this study as QoS matrix parameter. Throughput is determined by the amount of packets that are successfully processed from source to the destination in a network. Figure 7.9 illustrates throughput results for both Scenario 1 and 2. Scenario 2 (proposed algorithm) produced high amount of throughput compared with Scenario 1. Thus it can be concluded that the proposed algorithm performed better than the normal IPTV by successfully delivered more packets from source to the destination nodes.

From all the results analysed in these scenarios, it shows that the proposed algorithm outperformed the normal CDN-based IPTV by reducing end-to-end delay, jitter and server load as well as providing high throughput. However, the algorithm lacks the ability to intelligently handle the server breakdown at this stage. This issue has been considered and explained in Scenario 4 below.

Scenario 1 vs Scenario 4

The adaptiveness and effectiveness of the proposed algorithm is tested by comparing Scenario 1 and Scenario 4. As explained earlier, Scenario 1 is the normal CDN-based IPTV while Scenario 4 is the proposed algorithm with the ability of electing designated server and backup designated server and re-assigning the role of designated server to the backup server whenever the designated server fails.

Figure 7.10 shows packets end-to-end delay for Scenarios 1 and 4. The results show that the proposed algorithm reduced averagely 8% of packet end-to-end delay compared to the normal IPTV. The highest amount of packet end-to-end delay produced by the proposed algorithm is about 1.35ms, while the normal CDN-based IPTV produced about 1.9ms. Also as the designated server failed at exactly 522secs of the simulation time, the proposed algorithm was able to reassigned the designated server role to the backup designated server and elect another backup designated server among the other servers. Even though, high amount of packet end-to-end delay (1.35ms) was generated due to this process, but it is not as much as what the normal IPTV produced (1.42ms) at that particular point in time. Thus, the proposed algorithm performed better than the normal IPTV in reducing packets end-to-end delay.

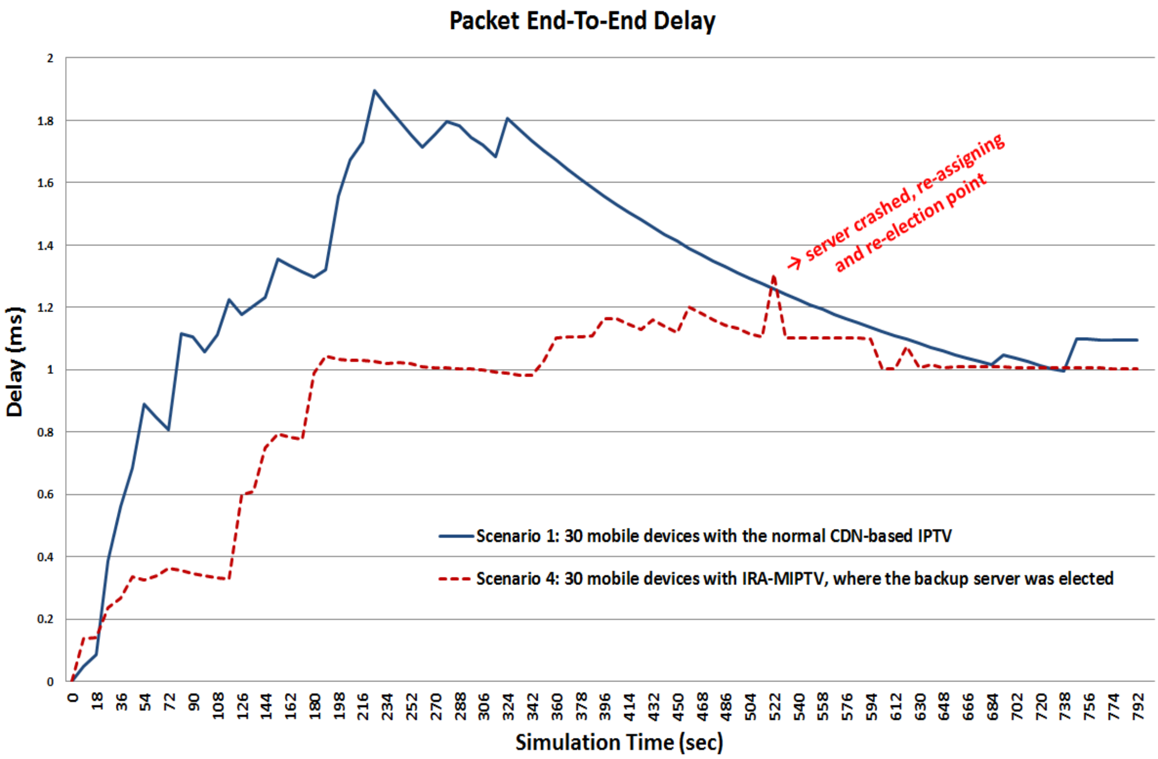


Figure 7.10: Packet End-To-End Delay

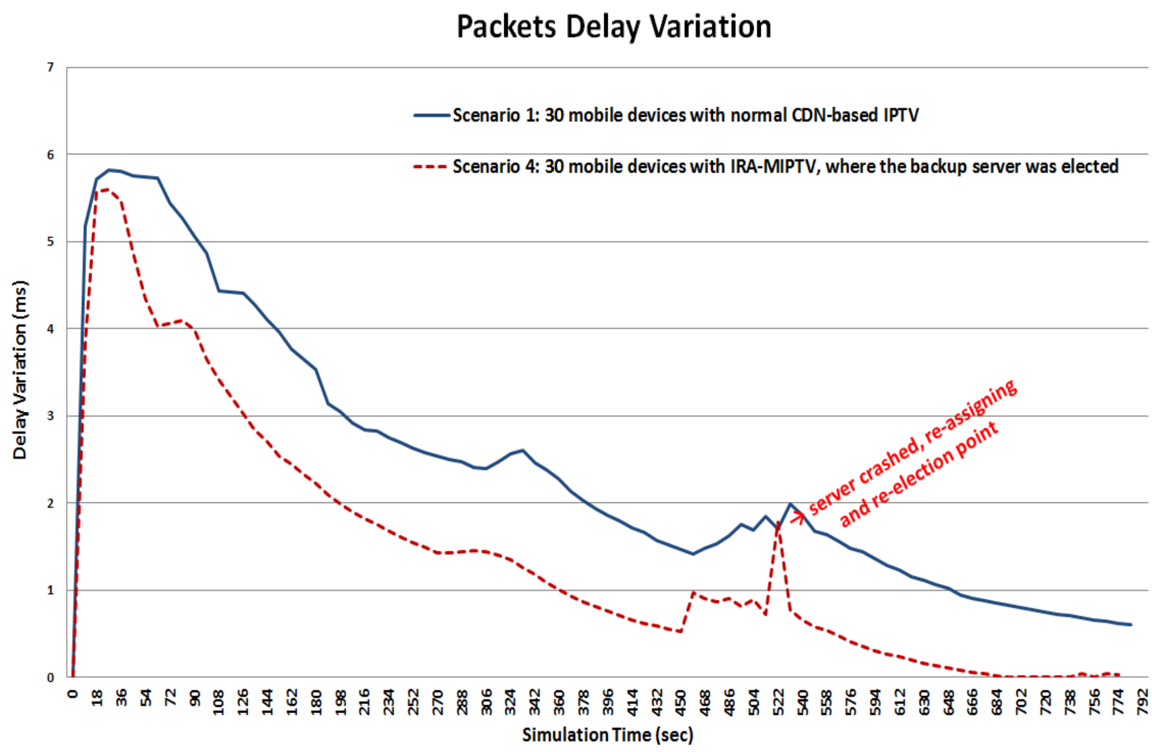


Figure 7:11 Packet Delay Variations

The packet delay variation (jitter) result for Scenario 1 and Scenario 4 is presented in Figure 7.11. The results show that, the Scenario 4 (proposed algorithm) produced less amount of jitter to averagely 18% compared to Scenario 1 (normal IPTV). However, the process of re-assigning the role of designated server to the backup designated server and the re-election of new backup designated server produced additional 1ms delay variation. As it can be seen in the graph, this process happened exactly at 522 secs of the simulation time. The amount of end-to-end delay produced by the proposed algorithm even at that particular point in time, it was not up to what was produced by the IPTV. In general the proposed algorithm performed better than the normal IPTV.

Figure 7:12 shows the server load results for Scenario 1 (normal CDN-based IPTV) and Scenario 4 (proposed algorithm). The results show that the proposed algorithm reduced up to 15% of the total server load compared with the normal IPTV. Similarly, the results show high increase in the server load during server selection and re-election process, which happened exactly at 522 secs of the simulation time. Nonetheless, it normalised within a minute. However, the average reduction of server load to 15% is an indication that the proposed algorithm accomplished as predicted.

The overall throughput results for Scenarios 1 and 4 are presented in Figure 7.13. The results show that Scenario 4 (proposed algorithm) produced high amount of throughput compared with scenario 1 (normal IPTV). This is due to the number of requests clients served on behalf of the servers in addition to that of the servers. Yet, the amount of throughput has dropped down during the designated server failure and re-assignment for about 20 seconds of the simulation time, but it stabilised after that. This process has happened at exactly 522 sec of the simulation time as it can be seen from the graph. However, it can be concluded that the proposed algorithm averagely produced up to 10% increase in throughput than the normal IPTV.

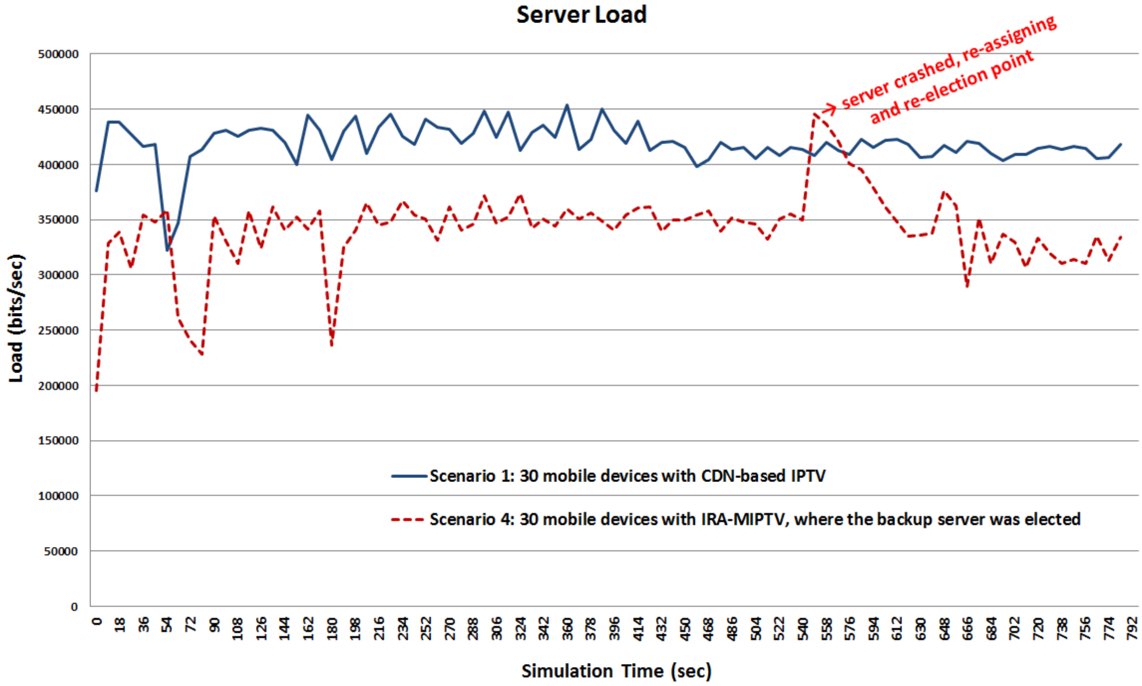


Figure 7.12: Server Load

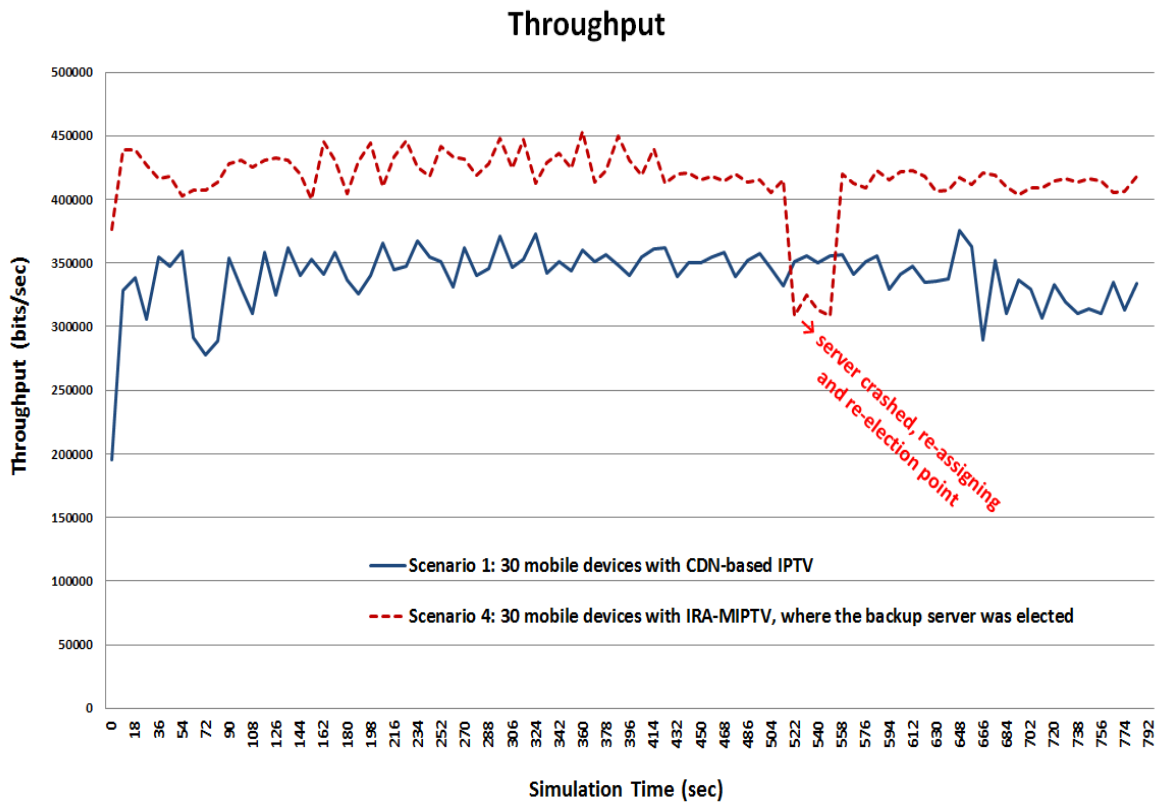


Figure 7.13: Throughput

Link utilization represents the percentage of link usage by all devices and application over the internet. Figure 7:14 displays the average link utilization for Scenario 1 (normal CDN-based IPTV) and Scenario 4 (IRA-MIPTV, where the backup server was elected prior to the designated server failure). The results show that the proposed algorithm has used the link less than the normal IPTV by averagely 4%. This is because some of the packets are served by the client on behalf of the server without going over the Internet. This also confirmed that the proposed algorithm has reduced the bandwidth consumption compared to the normal CDN-based IPTV.

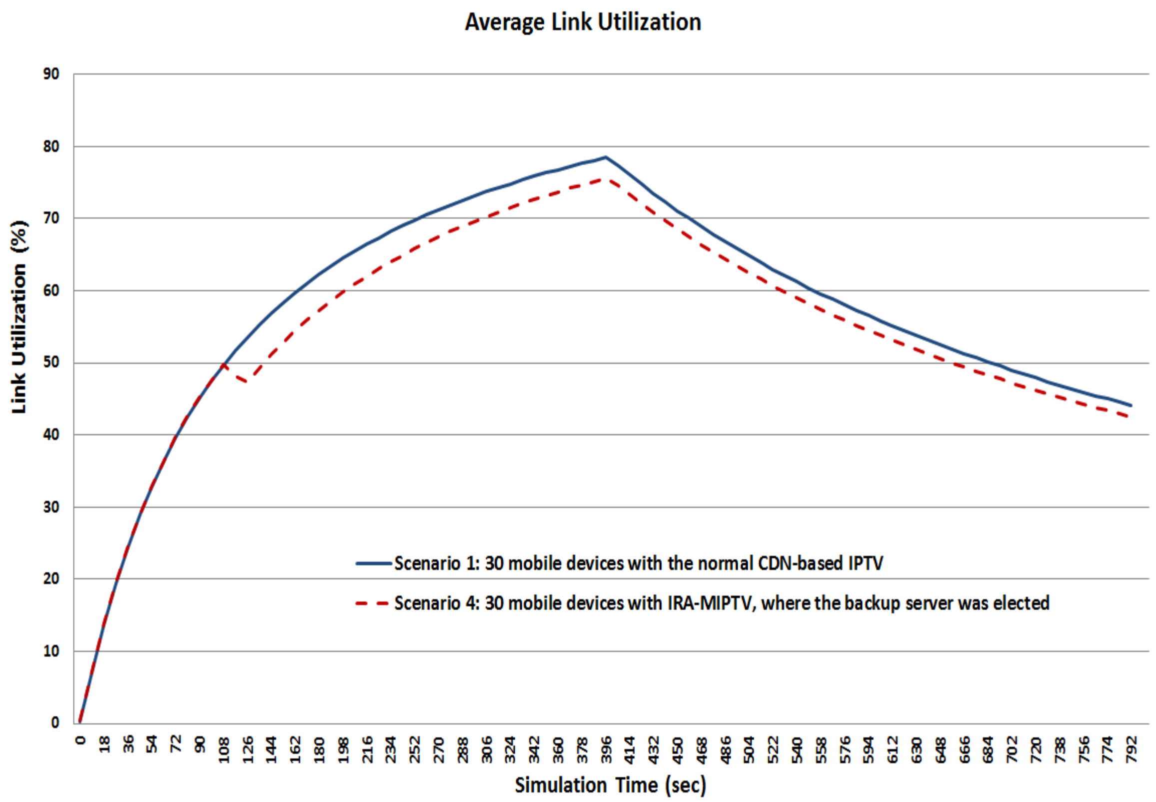


Figure 7.14: Average Link Utilization

For all the results of Scenarios 1 and 4 compared above, the investigation show that the proposed algorithm reduces the amount of end-to-end delay, jitter and server load and bandwidth consumption as well as produces high throughput. It can also be concluded that the proposed algorithm intelligently outperformed the normal IPTV by its ability to learn and elect the designated server and backup designated server based on the server available bandwidth, load and status. Also the re-assigning of designated server responsibilities to backup server when the designated server fails, assured service delivery at all times.

Scenario 3 vs Scenario 4

As explained earlier, you may recall that Scenario 3 is the scenario with the proposed algorithm that intelligently selects a server among the available servers in the network to replace the failed designated server. While Scenario 4 is the scenario that the proposed algorithm intelligently elect a backup designated server in case the designated server fails. The purpose of comparing these scenarios is to measure the additional overheads that may be accrued by the proposed algorithm during server re-assigning process and recommend the best practice. The servers were deliberately crashed at different times to give clear representational behaviour of each server during failure and re-assigning process.

The End-to-End delay results for Scenario 3 and Scenario 4 is illustrated in Figure 7.15. The results show that the proposed algorithm performed well in both scenarios by reducing packet end-to-end delay compared to the normal IPTV. The servers were deliberately crashed at 108 secs and 180 secs in Scenario 3 and Scenario 4 respectively. However, scenario 3 incurred more overhead during server replacement compared with scenario 4. Thus, it is better to elect the backup designated server prior to the failure of the designated server than to replace with any of the available servers after the designated server failed.

Packet delay variation for Scenario 3 and Scenario 4 is presented in Figure 7.16. From the results, it can be observed that the servers were deliberately crashed at 540 secs in Scenario 3 and 557 secs in Scenario 4. Scenario 4 produced less amount of packet delay variation (jitter) during switching from the failed designated server to the elected backup designated server and the re-election of new backup server, compared to Scenario 3 where no backup designated server was elected prior to the designated server failure. Thus, electing the backup designated server before the designated server failed, produced better result.

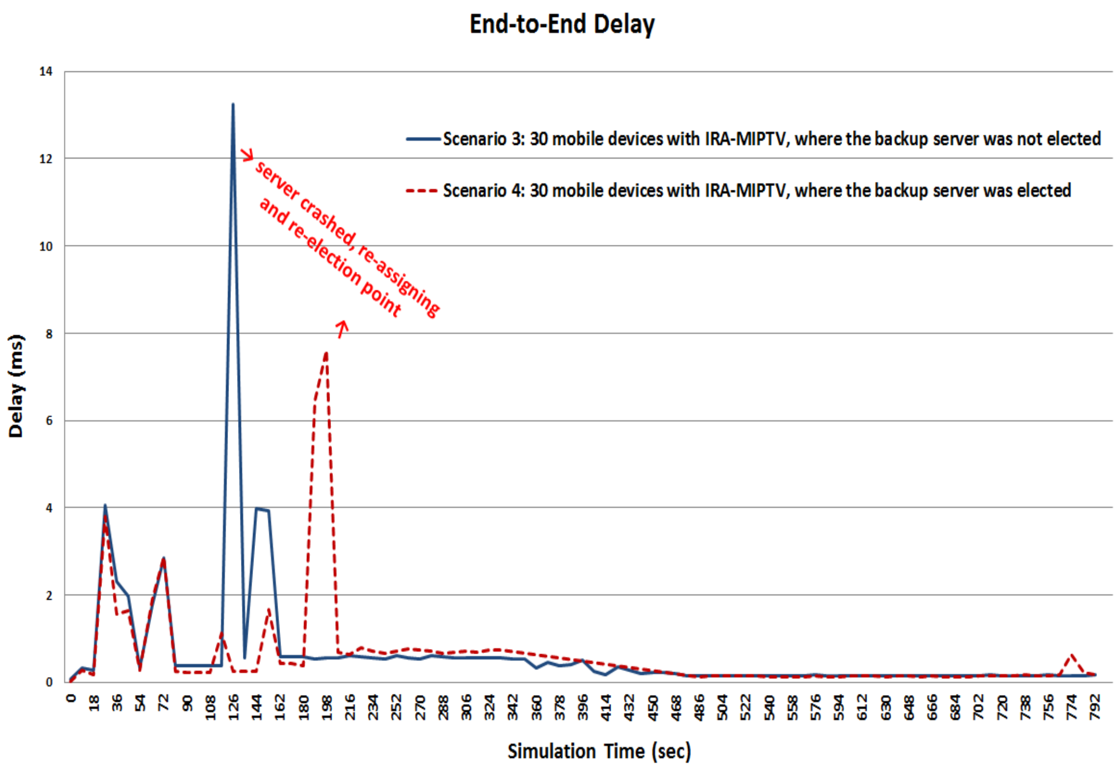


Figure 7.15: Packet End-to-End Delay

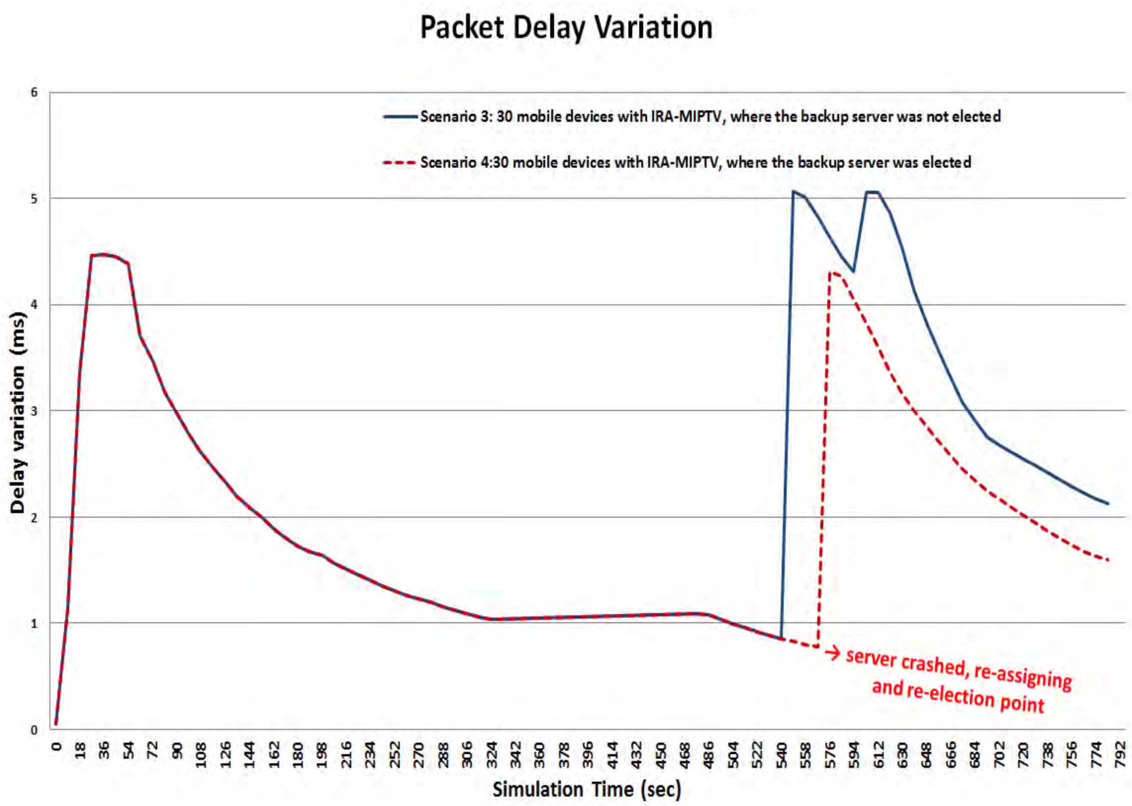


Figure 7.16: Packet Delay Variation

The server load results for Scenario 3 and Scenario 4 is shown in Figure 7.17. The results show that, the server load is the same in both scenarios from the beginning till when the designated server failed at exactly 414 secs of the simulation time. Selecting a server out of available servers to replace the failed server produced significant high server load than electing designated server prior to the designated server failure. Thus, electing backup designated server is more effective than replacing the designated server after it fails.

Figure 7.18 shows the throughput results for Scenario 3 and Scenario 4. The servers were deliberately crashed at 414 secs of the simulation time. The results also confirmed that, the throughput has slightly improved by electing the designated backup server prior to the designated server failure.

In this project, a network of 30 mobile devices was model. The devices were streaming video contents from the main server and the replicative servers. Four different scenarios were created, tested, analysed and evaluated in order to test the effectiveness of the proposed algorithm. Based on the results analysed, the proposed algorithm performed better than the normal IPTV by reducing the end-to-end delay, jitter, server load and increased overall network throughput. Also it has been observed that electing designated server and backup designated server before the main server crashed improved the effectiveness of the algorithm.

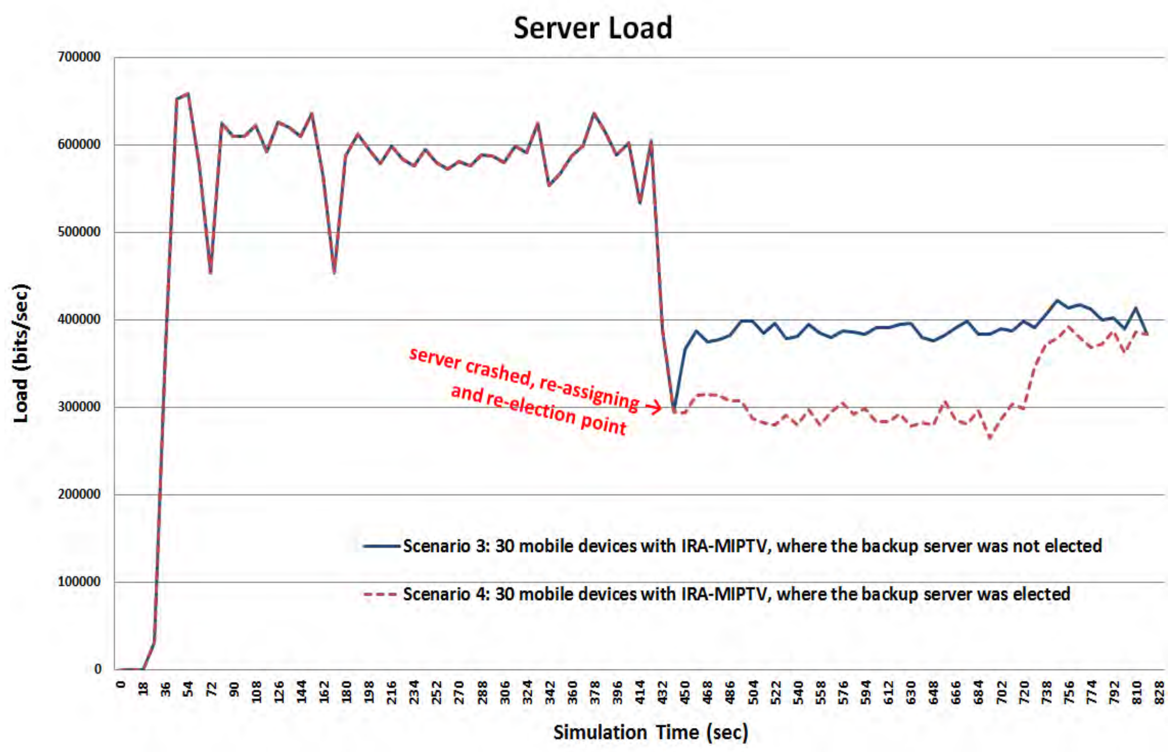


Figure 7.17: Server Load

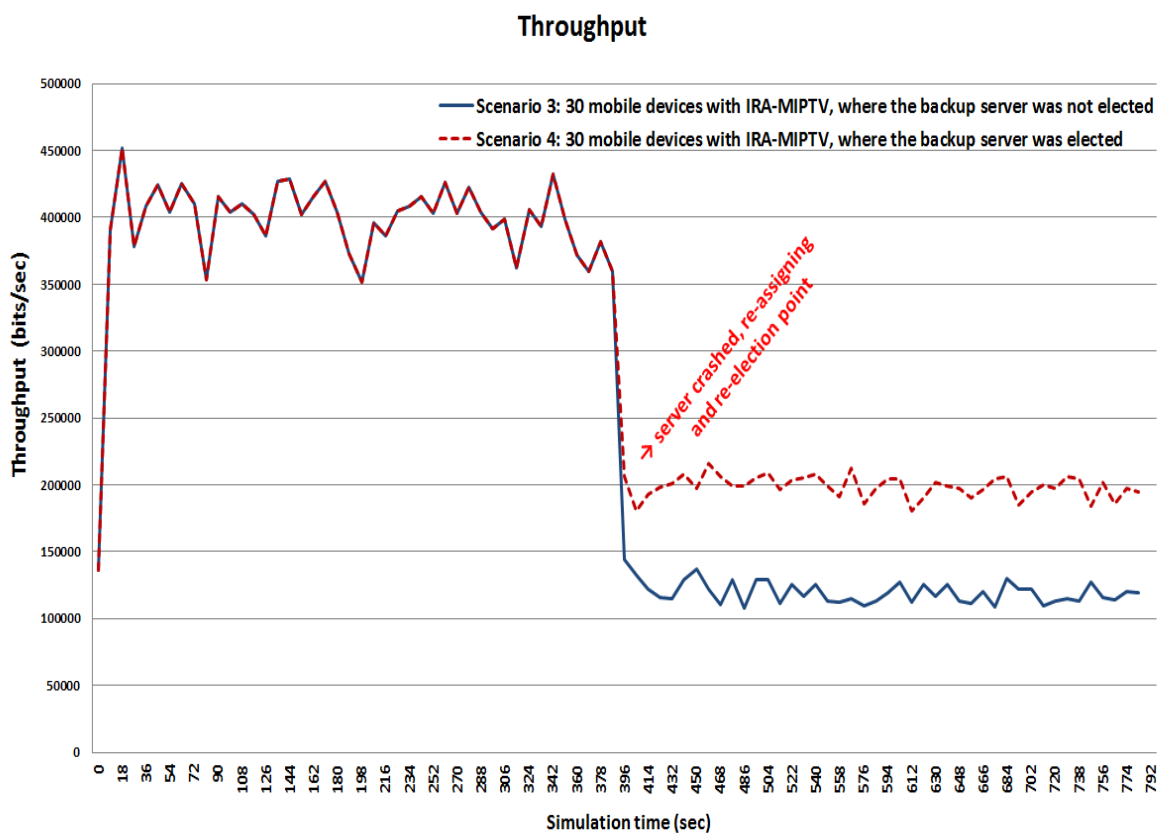


Figure 7.18: Throughput

7.8.2 IRA-MIPTV Project 2: Results Analyses and Evaluation

This project is modelled with 70 mobile devices, consisting of 7 groups with 10 devices in each group. As earlier explained, the difference between Project 1 and Project 2 is the number of clients. Several simulations were conducted with number of client's ranges from 10 to 100, where best results were chosen. At this project, the highest number of devices that the simulation was successfully completed is 70. Thus, 70 mobile devices are considered. Considering the number of devices to 70 is to analyse the network behaviour and the effectiveness of the proposed algorithm as the number of devices increases. The same QoS parameters and scenarios descriptions used in Project 1 were also used in this project.

You may recall that Scenario 1 is the normal CDN-based IPTV, while Scenario 2 is the proposed algorithm where the designated server was deliberately crashed during simulation. Scenario 3 is the proposed algorithm where the designated server was replaced with any available server after it failed. Scenario 4 is the proposed algorithm where the backup designated server was elected prior to the designated server failure.

Scenario 1 vs Scenario 2

The end-to-end delay results for Scenario 1 and Scenario 2 are presented in Figure 7.19. The results show that the proposed algorithm reduced considerable amounts of end-to-end delay compared to the normal IPTV. However, the designated server deliberately crashed at 200 secs of the simulation time. The server's failure rendered the proposed algorithm ineffective. However, adaptive measures were introduced to handle such problems. The issue has been addressed in Scenario 4. When the two projects were compared, Project 2 produced averagely about 56% less end-to-end delay. This is because the numbers of clients that have the packets are greater in a network with many devices than the network with fewer devices. Thus, the proposed algorithm performs better as the number of devices increase.

Figure 7.20 presents the packet delay variation results for Scenario 1 and Scenario 2. The results show that the proposed algorithm reduced about 15% of jitter compared with normal IPTV. When comparing Project 1 with Project 2, the average packet delay variation for Project 2 is around 5.6ms, while Project 1 has about 4.4ms. Considering the number of devices in Project 2 is twice more than that of Project 1, this show that the proposed algorithm performed very well in reducing the amount of jitter during high service request. Therefore, it can be concluded that as the number of devices increases, the effectiveness of the proposed algorithm also increases. However, as the server deliberately crashed, the algorithm render ineffective. Therefore, the proposed algorithm has to be intelligent to handle such situations. This problem has been addressed in Scenario 4 below.

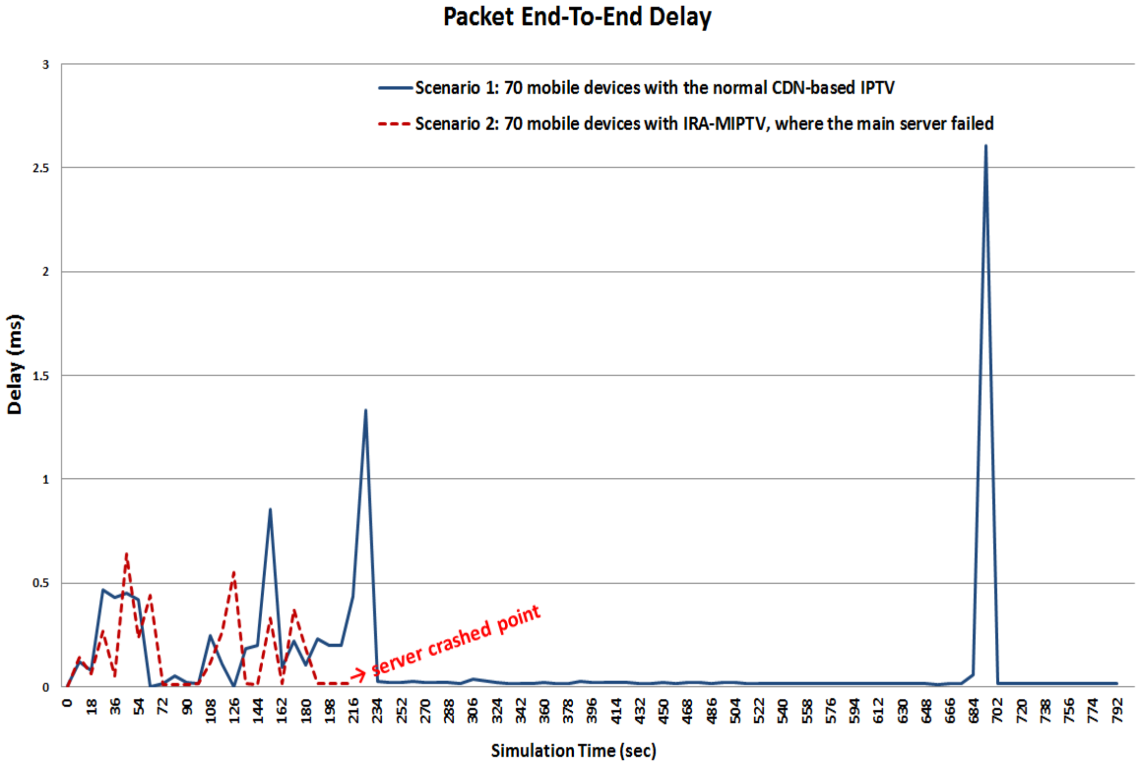


Figure 7.19: Packet End-To-End

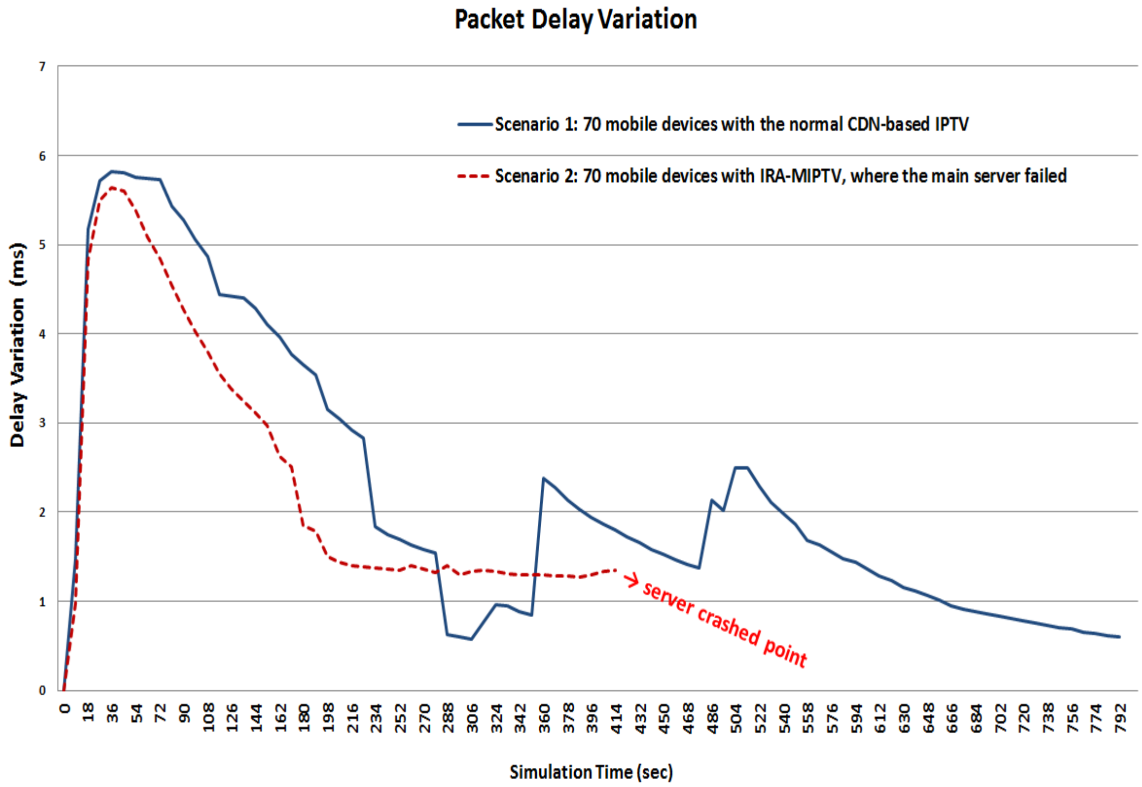


Figure 7.20: Packet Delay Variation

The server load results for Scenario 1 and Scenario 2 are shown in Figure 7.21. The results show that the proposed algorithm reduced averagely about 16% of the server load compared with the normal IPTV. However, as the designated server crashed, the proposed algorithm was rendered ineffective. Thus, the algorithm has to be intelligent enough to handle such unforeseen circumstances. This issue has been addressed in Scenario 4 of this project. When comparing the two projects, Project 1 accrued two times the server load than Project 2. This is because the more devices there are on the network, the higher the tendency of getting the requested packets within the clients. Hence, the proposed algorithm performs better in reducing server load as the number of devices increase.

Figure 7.22 presents the throughput results for Scenario 1 and Scenario 2. The results show that the proposed algorithm produced more throughput compared to the normal IPTV although the server failure made the algorithm ineffective. Thus, the issue has been addressed in Scenario 4. However, when Project 1 was compared with Project 2, the results show that Project 2 produced higher throughput than in Project 1. This is due to the number of devices in Project 2 being two times more than that of Project 1, so it is expected to have higher requests and to deliver more packets.

Based on the results analysed in both Projects, it has been observed that the proposed algorithm performed better than the normal IPTV in end-to-end delay, jitter and server load reduction as well as increasing the amount of throughput. After thorough evaluation, the results revealed that as the number of devices increase, the effectiveness of the algorithm also increases.

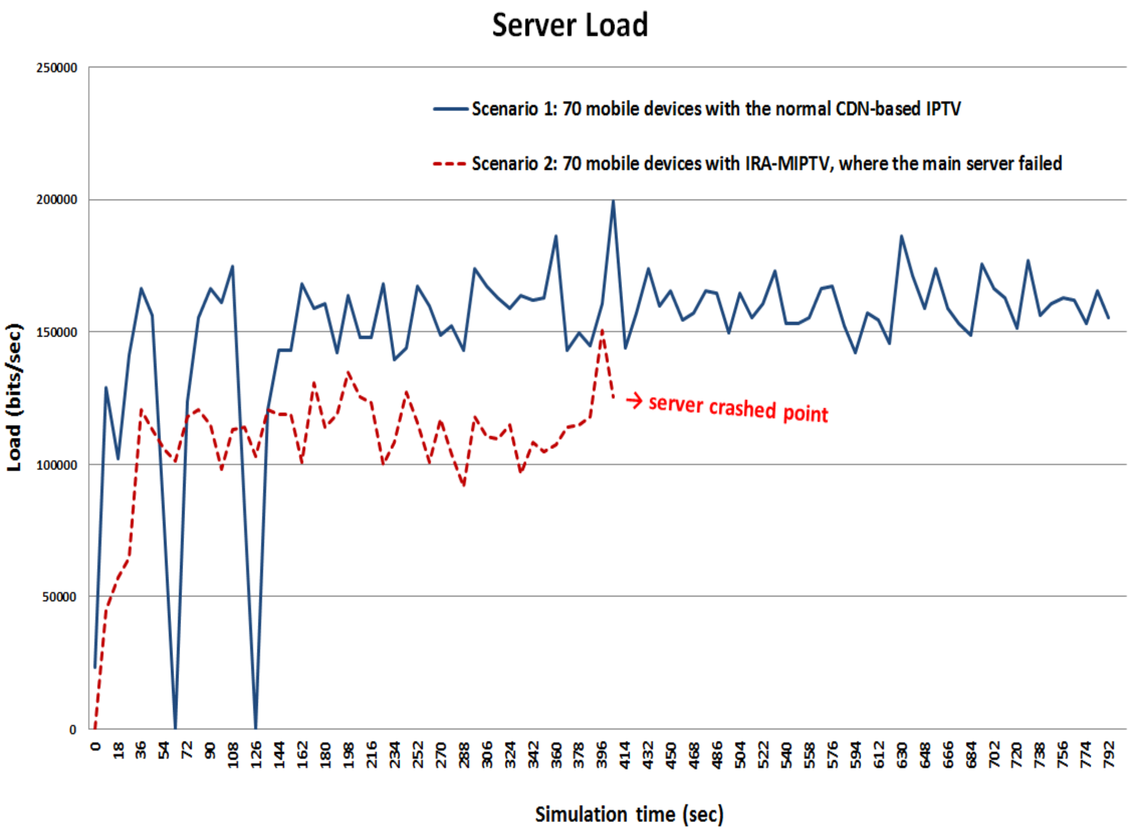


Figure 7.21: Server Load

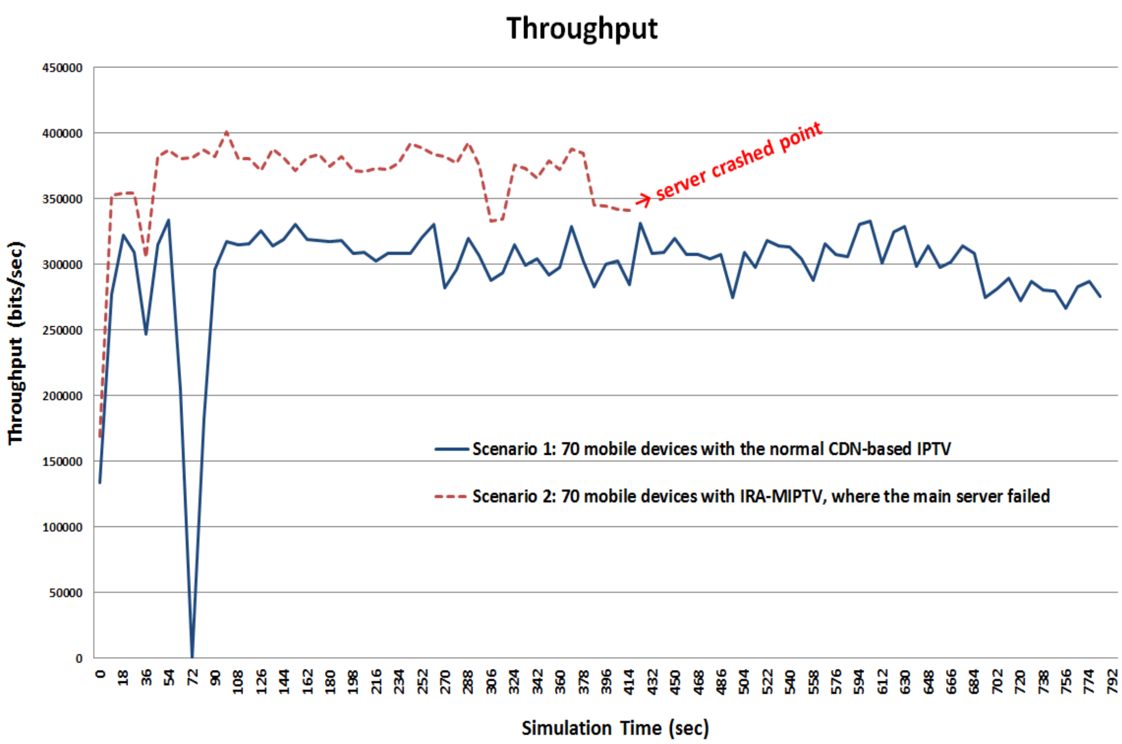


Figure 7.22: Throughput

Scenario 1 vs Scenario 4

The issue of the server failure discussed in Scenario 2 affected the efficiency of the proposed algorithm. This problem has been addressed in Scenario 4 and is discussed in detail as this stage of the experiment. You may recall that Scenario 1 is the normal CDN-based IPTV and Scenario 4 is the proposed intelligent algorithm, where designated and backup servers are elected. Figure 7.23 shows the packet-end-to-end delay results for Scenario 1 and 4. The results show that normal IPTV produced averagely 2.7ms end-to-end delay while the proposed algorithm produced averagely 1.4 end-to-end delay. This confirmed that the normal IPTV produced almost twice end-to-end delay compared to the proposed algorithm. Also the amount of end-to-end delay produced by the proposed algorithm during the time server crashed, re-assigned and backup server re-elected is not up to what the normal IPTV produced. This also indicates that the election of the designated and backup designated server, as well as re-assigning the role of designated server to the backup server when the designated server failed, does not affect the efficiency of the proposed algorithm. Furthermore, when comparing Project 1 with Project 2, the proposed algorithm produced 1.4ms of end-to-end delay in Project 2, while Project 1 produced averagely 1.1ms. However, considering the number of devices in project two, the proposed algorithm reduced the amount of end-to-end delay. Thus, this also proved that the proposed algorithm performs better as the number of devices increase.

Figure 7.24 presents the packet delay variations for Scenario 1 and Scenario 4. The results show that the proposed algorithm reduced averagely 40% packet delay variation compared with the normal IPTV. Although the proposed algorithm produced high packet delay variation for about 2.4 milliseconds at the point of assigning designated server roles and re-election of backup designated server, but at the average of 2ms produced by the proposed algorithm compared to 3.2ms produced by the normal IPTV, shows that the algorithm performed well. Similarly, when comparing Project 1 and Project 2, the proposed algorithm produced slightly higher delay variation in Project 2. However, that does not affect the general efficacy of the algorithm, and it can be addressed by packet buffering.

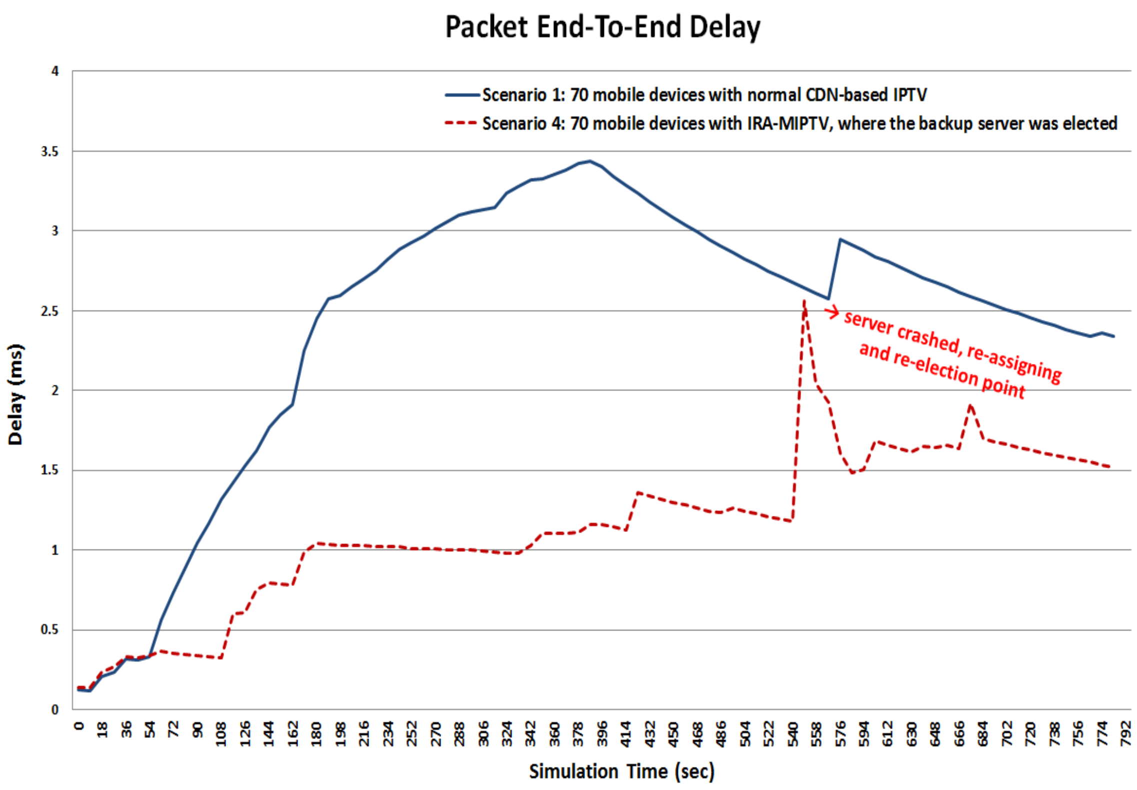


Figure 7.23: Packet End-To-End Delay

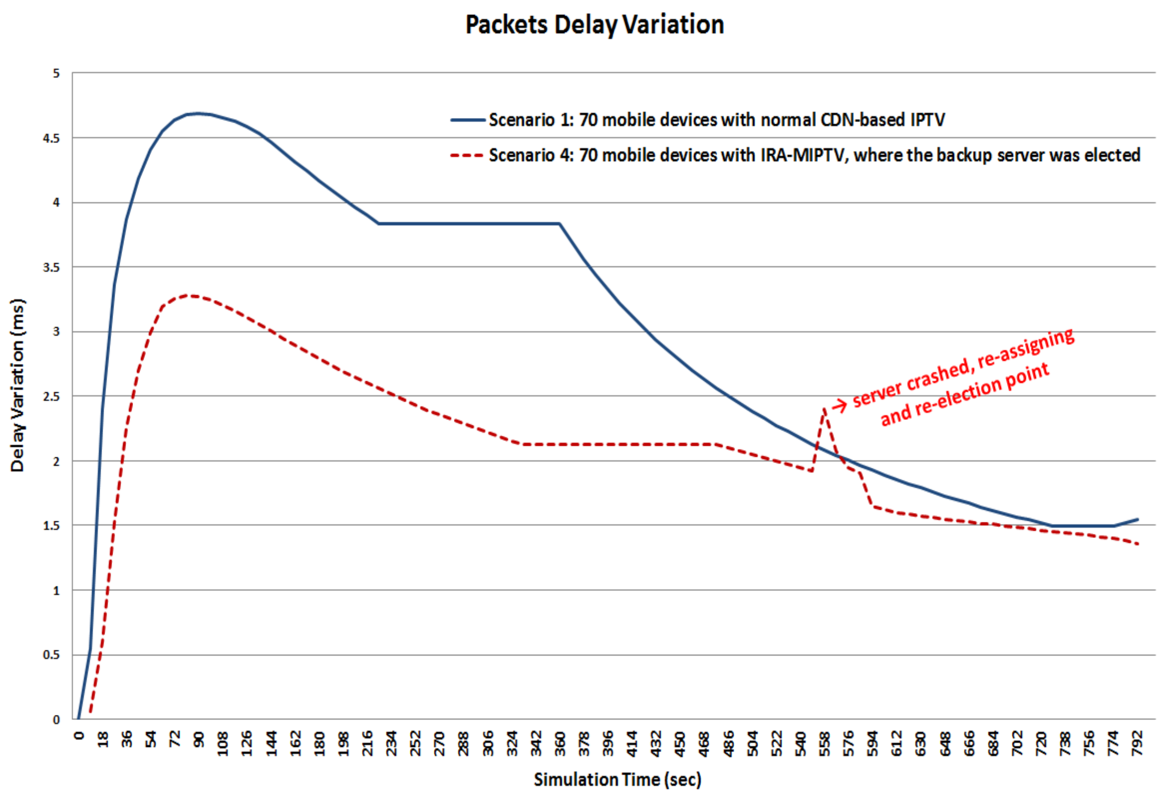


Figure 7.24: Packet Delay Variation

The server load results for Scenario 1 and 4 are presented in figure 7.25. The results show that the proposed algorithm reduced a substantial amount of server load compared with the normal IPTV. The proposed algorithm reduced, on average, about 40% of the server load compared with the normal IPTV. Likewise, when comparing Project 1 with Project 2, the proposed algorithm reduced the total server load to about 10% in Project 2 compared with Project 1. Therefore, the effectiveness of the proposed algorithm increases as the number of devices increase. Hence the proposed algorithm outperformed the normal IPTV in both projects.

Figure 7.26 shows the throughput results for Scenarios 1 and 4. The results show that the proposed algorithm produced on average about 50% higher throughput than the normal IPTV. While comparing the two projects, Project 2 produced more throughput than Project 1. Thus the efficiency of the proposed algorithm increases as the number of devices increase.

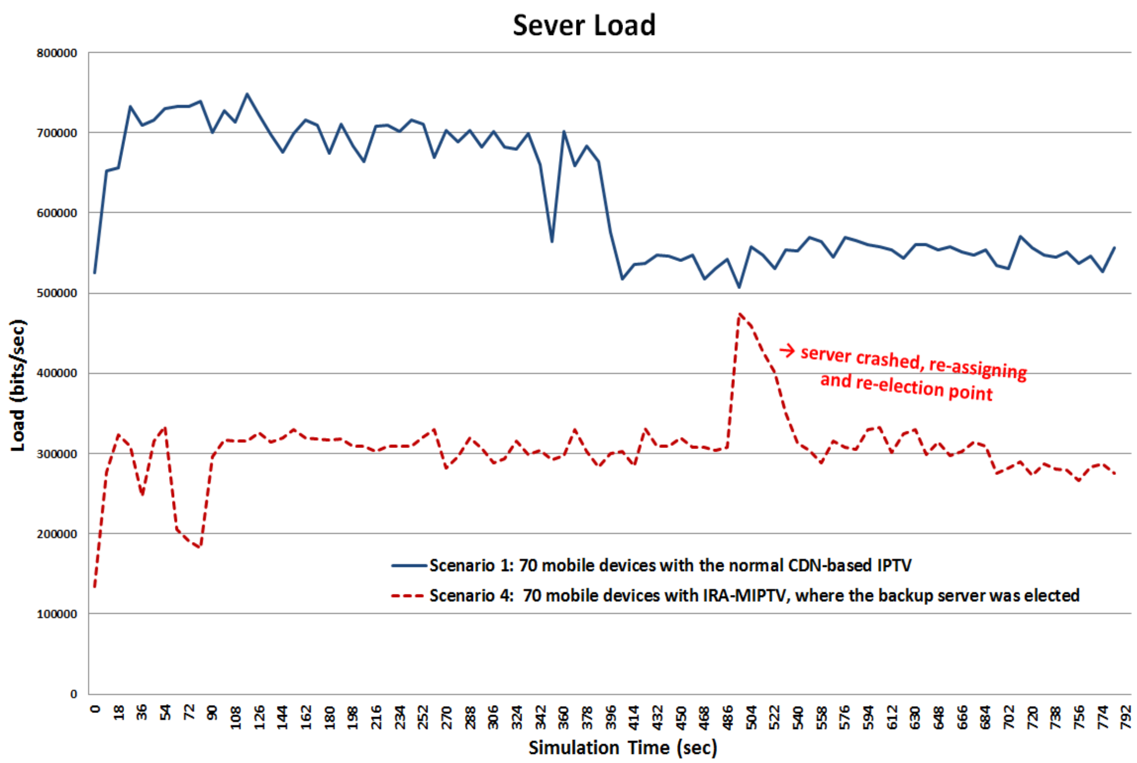


Figure 7.25: Server Load

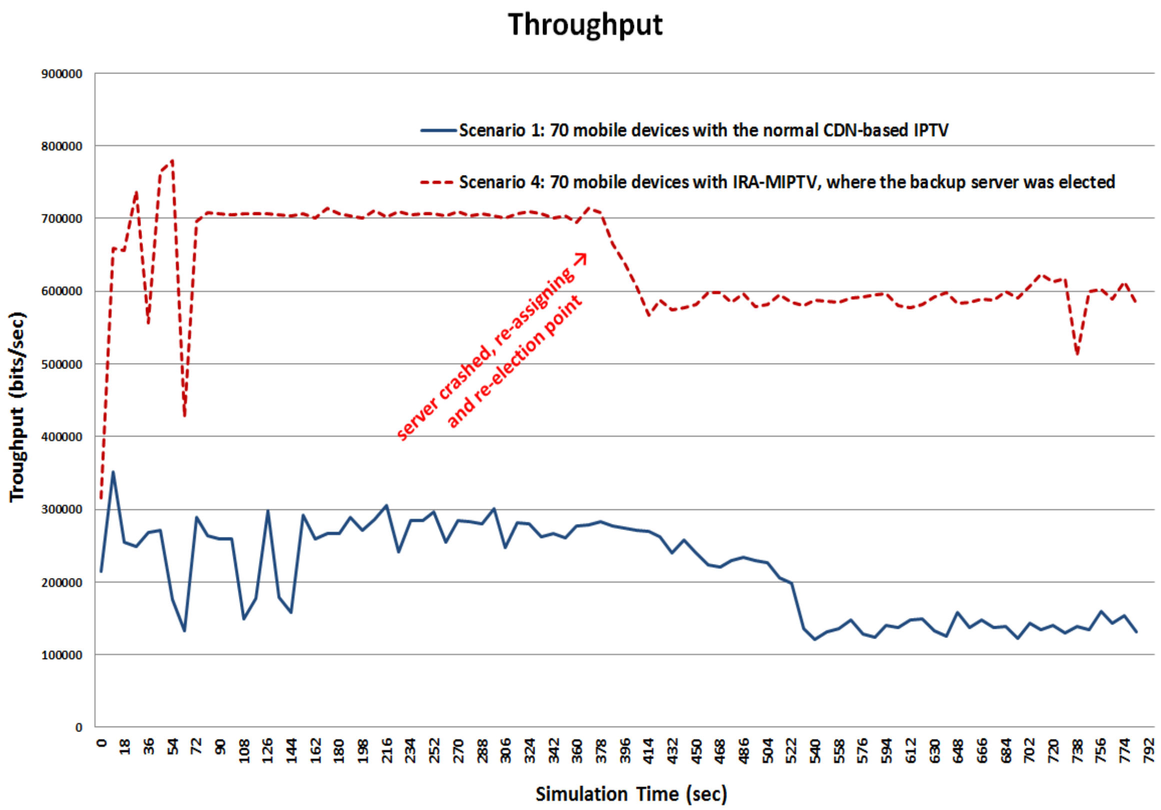


Figure 7.26: Throughput

Link utilization results for Scenario 1 and 4 are presented in Figure 7.27. The results show that the proposed algorithm has the highest link utilization at the beginning of the simulation but reduced significantly towards the end of the simulation. This is due to the number of request some of the clients served on behalf of the server using MANET protocol, without using IP over to the internet. The proposed algorithm has averagely 57% of like utilization while the normal IPTV has averagely 65%. This confirmed that the proposed algorithm reduced the bandwidth consumption compared to the normal IPTV.

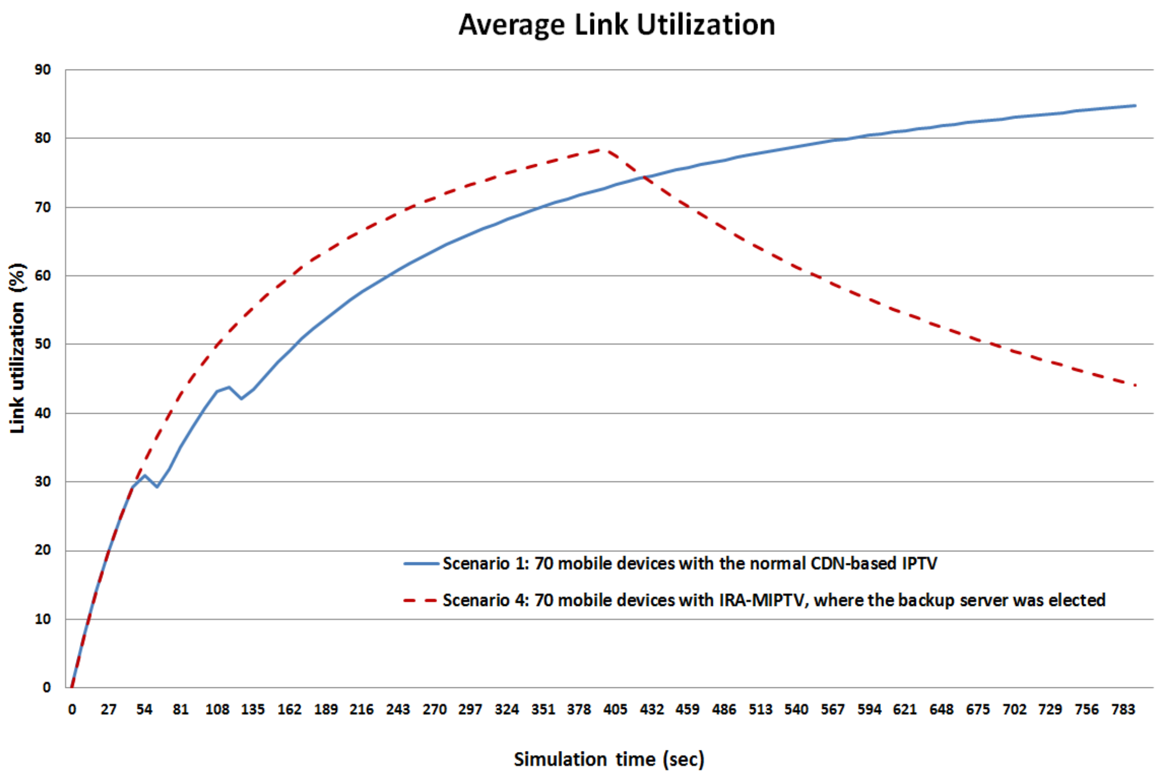


Figure 7.27: Average Link Utilization

From all the results analysed in these scenarios, the evaluation confirmed that the proposed algorithm outperformed the normal IPTV in all the QoS parameters considered in this study. Also, it has been evidenced that the efficiency of the proposed algorithm increases as the number of devices increase in the network.

Scenario 3 vs Scenario 4

In order to evaluate any overheads the proposed algorithm may have accrued during the process of electing a designated server and backup designated server, Scenarios 3 and 4 are compared and analysed. You may recall that Scenario 3 is the proposed algorithm that intelligently selected a server among the available servers in the network to replace the failed designated server, while Scenario 4 is the proposed algorithm that intelligently elects a designated server and backup designated server prior to the designated server failure.

The packet end-to-end delay results for Scenario 3 and 4 are presented in Figure 7.28. The results show that, after the designated server failed, in the process of replacing the failed server, Scenario 4 reduced averagely 51% packet end-to-end delay compared with Scenario 3. Thus, it is recommended to elect the backup server prior to the designated server failure. When comparing the two projects, Project 2 produced slightly more end-to-end delay than Project 1. Considering the number of clients in Project 2 is two times more than that of Project 1, it shows that the proposed algorithm performed well as the number of clients increased.

Figure 7.29 presents the packet delay variation results for Scenarios 3 and 4. The results show that replacing a designated server after it failed, generated a higher amount of jitter to averagely 18% compared to electing the backup designated server prior to the designated server failure. Thus, the proposed algorithm performed better when the designated server and the backup designated server were elected. Similarly, when comparing

Project 1 and Project 2, there was averagely 13% increase in packet delay variation in Project 2 than in Project 1. This is due to the number of clients in Project 2 is twice more in Project. However, considering increase of only 13% of the jitter with 130% increase in the number of clients, this shows that the proposed algorithm performed well as the number of client's increases.

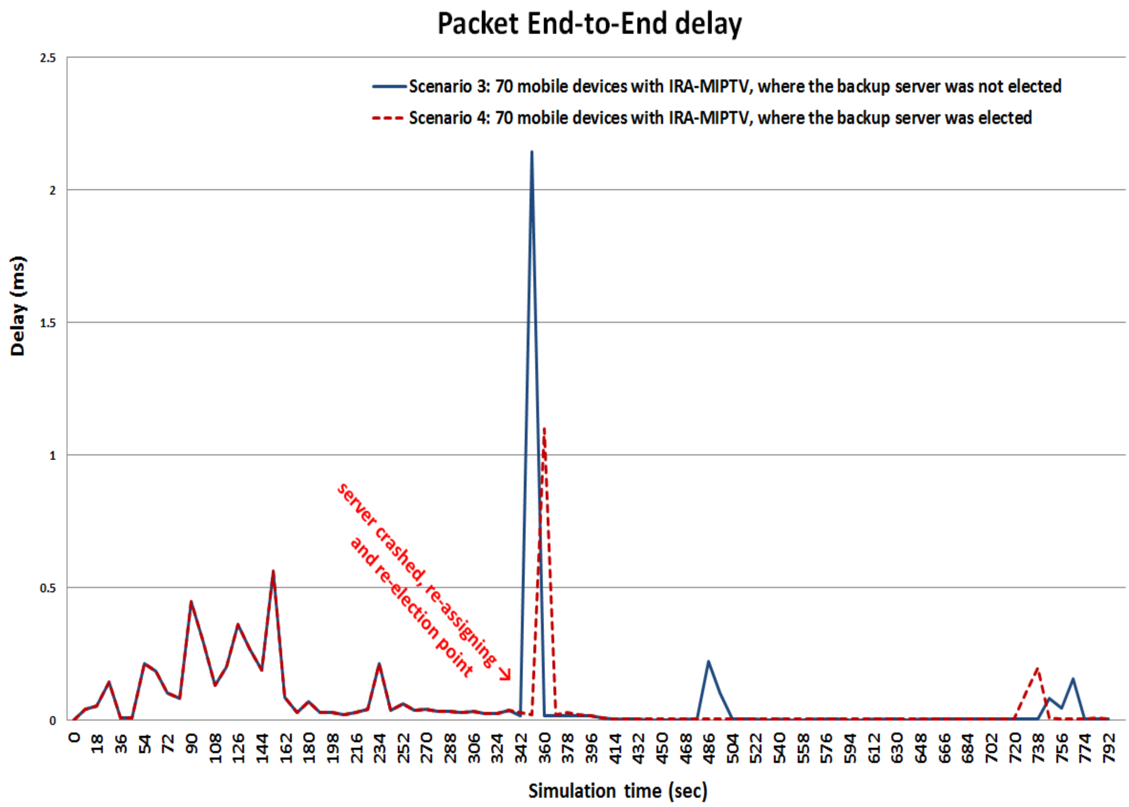


Figure 7.28 Packet End-to-End Delay

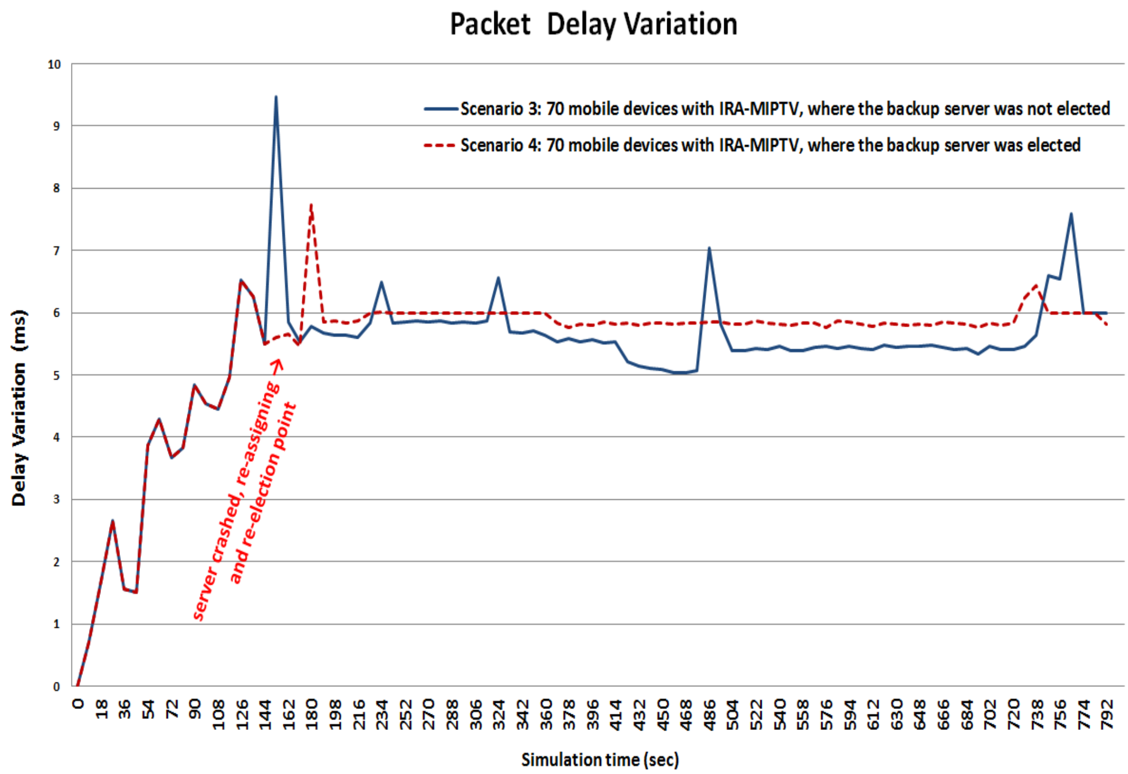


Figure 2.29: Packet Delay Variation

The server load results for Scenarios 3 and 4 are presented in Figure 7.30. The results show that the process of replacing the designated server from the available server after it failed produced averagely 18% higher server loads than electing a backup server prior to the designated server failure. Therefore it is recommended to elect designated and backup designated servers prior to the designated server failure. Also when comparing the two projects, Project 2 reduced 56% more server load than Project 1. However considering the number of clients in Project 2 is 130% more than that of Project 1, this indicates that the proposed algorithm performed well as the number of clients increases.

Figure 7.31 presents the throughput results for Scenario 3 and Scenario 4. The results show slight increases in throughputs of 8% averagely when the designated server and the backup designated server are elected prior to the designated server failure. Thus, it is recommended that the backup server is elected prior to the designated server failure.

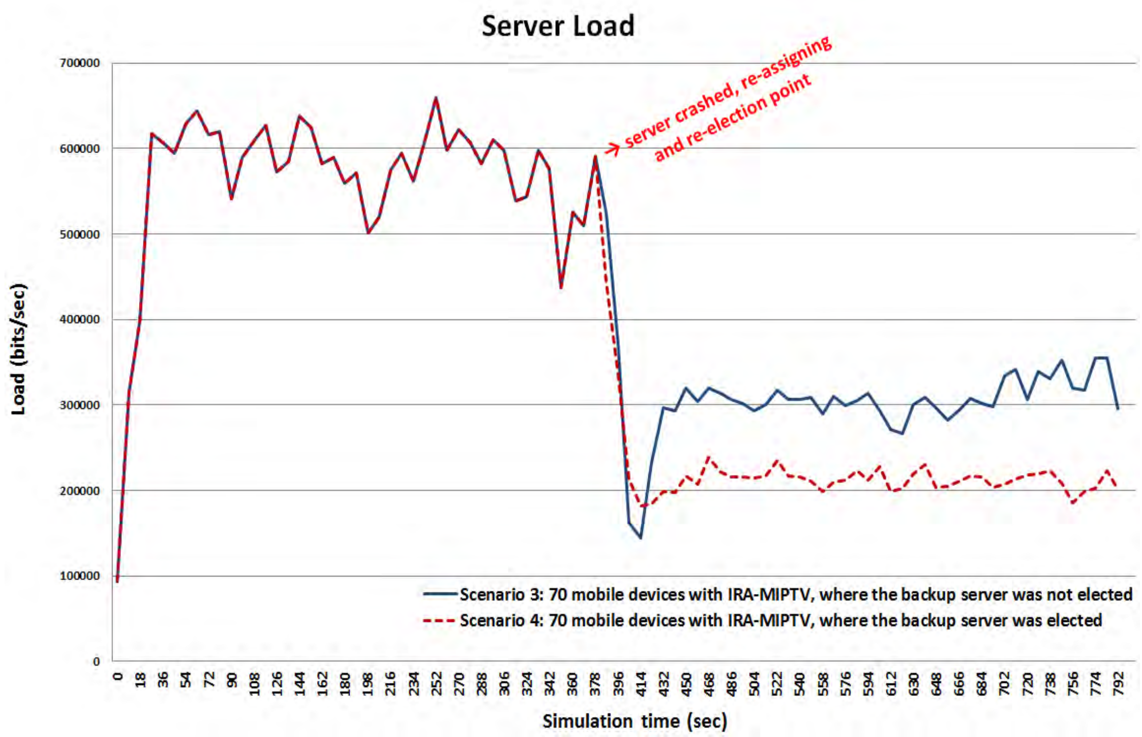


Figure 7.30: Server load

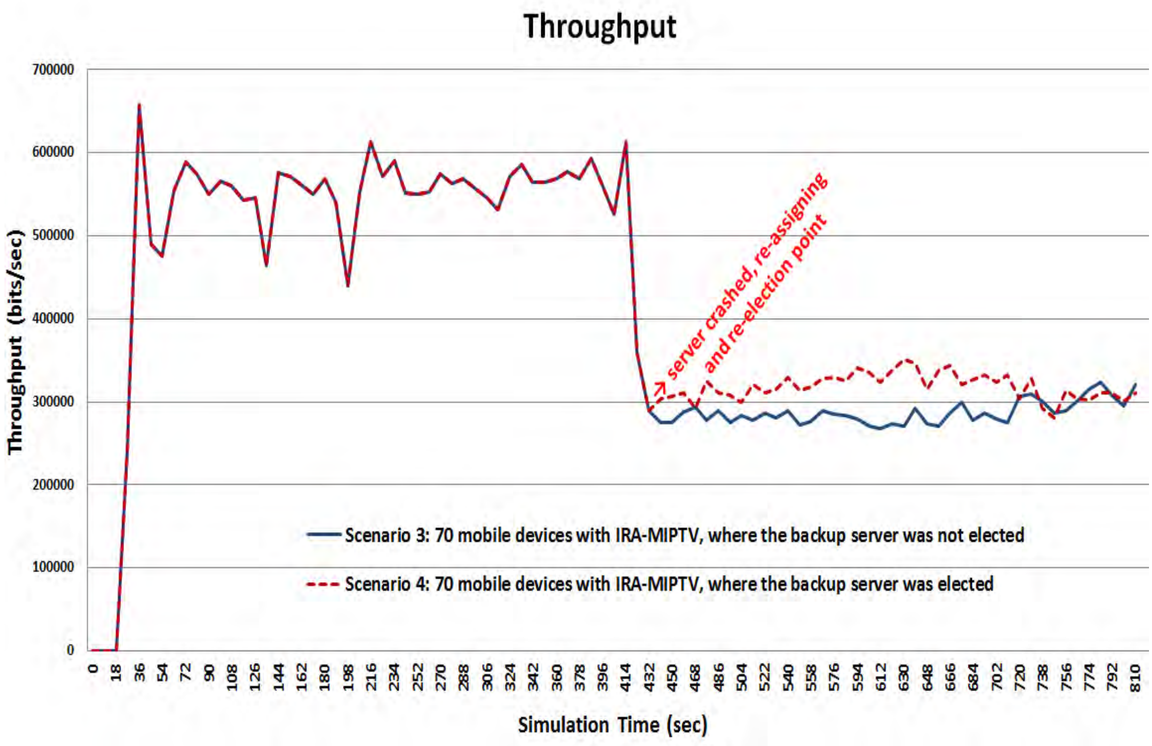


Figure 7.31: Throughput

7.9 Conclusion

This chapter introduced, tested and evaluated the proposed algorithm called Intelligent Routing Algorithm for mobile IPTV (IRA-MIPTV). The proposed algorithm intelligently elects a designated server and backup designated server to provide a backup mechanism in case the designated server fails. MD5 hash algorithm was also used for faster packet searching. Two projects and four scenarios were created and analysed in order to evaluate the effectiveness of the proposed algorithm as the number of clients increased in the network. Project 1 was modelled with 30 mobile devices, while Project 2 was modelled with 70 mobile devices.

IRA-MIPTV is the complete intelligent routing algorithm for mobile IPTV that included the entire previous algorithm discussed in the chapters above. The QoS parameters considered included packet end-to-end delay, packet delay variation (jitter), server load, and throughput as well as link utilization. The analyses of the results shows that the proposed algorithm outperformed the normal IPTV by reducing considerable amounts of packet end-to-end delay, jitter, and server load. It also produced more throughput than the normal IPTV. In the previous algorithms results discussed, jitter was high in all the scenarios with proposed algorithm. This was due to the time taken to search for packets in the clients. However, the used of hash algorithm clearly show that, it provided servers with faster packet searching, which in turn solved the jitter problem. When comparing the two projects, the results show that the efficiency of the proposed algorithm increases as the number of clients increases in all the QoS parameters considered.

Chapter 8: Discussion, Conclusion and Future work

8.1 Discussion:

IPTV can be described as a system through which television services and video on demand are delivered through controlled Internet protocol network using streaming technologies that managed and support quality of service (QoS), security, interactivity, and reliability. The IPTV is not just about transmitting digital television services over internet technology, but it is about reinventing television to better achieve the required goals and creating a video-centric next-generation Internet accessible on any device, be it mobile phone, computer, or smart TV, at any time and place the consumer chooses. This makes demand for IPTV service in wireless networks higher and is expected to continue increasing over time.

To provide IPTV services with the necessary guarantee quality of service and quality of experience to the end users, required minimum bandwidth and resources must be met by the server in order to serve all the service requests received from the clients at all times. As the number of service requests increases the bandwidth consumption and server load also increases. However, the server bandwidth is limited and can totally be exhausted by the numerous clients' requests, which can overload the server resources, capacity and congest the network. Thus, results to poor quality of service.

Providing IPTV services with required QoS has been a serious concern to the service providers, especially during high service request demand. Several researches were conducted and implemented in order to improve the general IPTV QoS. In this research, previous related studies were reviewed and found that emphasis were given to how client nodes and network devices communicate. However, IPTV services totally rely on servers in providing services to the clients. Thus, the server must have the required bandwidth and capacity to serve clients' requests. For this reason, this study emphasised on the server node. The study is aimed at addressing the problems associated with delivery of video content over the Internet, at the server node. The problems addressed include server overload, bandwidth consumption, packet end-to-end delay and jitter, as well as ensuring the general quality of service in IPTV.

To achieve the set aims and objectives, three research questions were identified, which are:

- Can techniques be found and put in place to reduce bandwidth consumption and workload at the server node in IPTV?
- How can quality of service be enhanced in IPTV?
- How can other protocols be adopted to enhance IPTV quality of service?

To answer these questions, a systematic approach was applied in the designing, developing and evaluating of the proposed algorithm called Intelligent Routing Algorithm for Mobile Internet Protocol Television. The proposed algorithm used IP, MANET and OSPF protocol techniques as well as considering CDN approach.

The proposed algorithm was designed in four different stages, to provide clear understanding and allow modifications from one stage to the other. In each stage, the results obtained were presented and published in a top IEEE conference and journal in some case. Valuable comments and feedbacks were obtained and used in the modification of proposed algorithm.

Deploying real IPTV network is not only time consuming, expensive and require different technical skills but also involve government policies and approval. To deploy such infrastructure even if all the resources are available, it requires government approval and licence, which may take longer time due to government policies and bureaucracy. Therefore, in this study simulation method was adopted, where the IPTV network was deployed. The algorithm was carefully designed, developed, implemented and tested in OPNET/Riverbed Modeler simulator. Riverbed Modeler supports real-world data into its simulation environment, which make this study to be closer to the real application. Live video data was captured during 2014 winter Olympics programme and the captured data were used in all the simulation scenarios. This gave more authenticity to simulation results as the research targeted and captured real data during high service request period and the data were used in all the simulation scenarios.

To validate the efficiency of the proposed algorithm as the number of clients' increases, two projects were created and simulated in all the implementation stages. Project 1 was created with few numbers of clients, while Project 2 was created with greater number of clients.

The proposed algorithm implementation and results obtained in various stages are explained as follows:

At the first stage of the proposed algorithm, a unicast scheme was applied in serving only VoD request. Therefore, all the VoD service requests received were served through one to one communication between the server and client. At this stage MANET routing algorithm was adopted that allowed ad-hoc communication between clients in packet forwarding. The implementation of the proposed algorithm and the simulation results obtained at this stage were presented in details in chapter 4. The results shown that the proposed algorithm reduced considerable amounts of server bandwidth consumption, workload and packet end-to-end delay in both projects. The results also shown that as the number of client increased, the efficiency of the proposed algorithm also increased. This confirmed that the proposed algorithm performed well during high request demand. However, the proposed algorithm produced higher jitter compared to the normal IPTV. Eventhough, the jitter problem can be address by packet buffering at the client storage, it is still not reasonable for real-time application.

The results obtained at this stage were presented at IEEE computer socity conference 2014 in USA and the exteded version was sent to the International Journal of Service Computing, see attached appendix. The feedback obtined were considered in improving the quality of the proposed algorithm.

However, IPTV services include VoD, live programme and time-shifted programme. To offer this services unicast and multicast are used, therefore, at this stage, the research questions were partially answered. In oder to deliver all IPTV services, the proposed algorithm was extended to handle the multicast scheme.

At the second stage of the algorithm, multicast scheme was added to accommodate the live and time-shipted programmes. The detailed implementation and analyses of the results at this stage were discussed in Chapter 5. Similarly, at this stage, in both projects, the results shown that packet end-to-end delay, server load and bandwidth consumption have been reduced significantly by the proposed algorithm compared with the normal IPTV. Similarly, the effectiveness of the proposed algorithm increased as the number of client increased. This also confirmed that the techniques adopted by the proposed algorithm yielded positive results even during high request period, which is the main focus of this research. However, the jitter produced by the proposed algorithm is also higher than the amount produced by the normal IPTV. However, there is slight improvement on the amount of jitter produced at this stage compared to the previous stage, yet, it is still not satisfactory.

The results obtained at this stage were also published at the IEEE conference, see attached appendix. The comments and feedback received from reviewers and participants were considered at the next stage of the proposed algorithm.

At the third stage, the proposed algorithm was extended to include the CDN architecture. Service providers are using CDN approach to improve faster service delivery and reduce server workload. Chapter 6 detailed the implementation and analyses of results of the proposed algorithm at this stage.

The results obtained shown that the proposed algorithm performed well in adapting to different server load states and select the best server or client to serve an incoming requests. It also evidenced that the proposed algorithm outperforms the normal IPTV system in server load reduction, high throughput and low amount of end-to-end delay in both projects. The results shown that as the number of clients increased, the efficacy of the proposed algorithm also increased. However, as was the case in all previous stages, the average delay variation is relatively high in the scenarios with the proposed algorithm. Nonetheless, when compared with previous stages, the amount of jitter is enhanaced. The research questions were literally answered at this stage. However, the issue of jitter is still a major concern. Similarly, the proposed algorithm should be intelligent to provide servers with the ability to take decision when the main server failed.

The results obtained at this stage were submitted and accepted at the IEEE conference, in Thailand which will be presented in Feb. 2016. See attached appendix. The reviewer's comments were also considered at the final stage of the algorithm.

The fourth and the final stage of the algorithm implementation considered the improvement of the proposed algorithm to handle the jitter issue and intelligently elect main server and backup server in case of server failure. MD5 hash algorithm was used in the proposed algorithm to improve faster packet searching process and provide basic security measures on the packets transmissions across the network. The hash algorithm implemented, solved the jitter problem by providing the servers a faster packet searching in the database. Detailed implementation and the results analyses of the proposed algorithm was discussed in Chapter 7.

OSPF election techniques were also adopted at this stage. It provided the algorithm with the ability to elect a main server and a backup server. The backup server is a mechanism to address the main server failure.

The results obtained at this stage in both projects shown that the proposed algorithm outperformed the normal IPTV by reducing considerable amounts of packet end-to-end delay, jitter and server load. It also produced more throughput than the normal IPTV. The jitter has been reduced to the minimum when compared to previous stages. It also shown that the efficiency of the proposed algorithm increased as the number of clients increased.

The results were also submitted and accepted at another IEEE conference, which will be presented in March 2016, in India. See attached appendix.

8.2 Conclusion

In general, the results obtained in all stages confirmed that the proposed algorithm reduced significant amounts of server bandwidth consumption, workload and packet end-to-end delay. However, packet delay veriation (jitter) produced by the proposed algorithm at stage 1 through stage 3 was issue of concerned. But, at the last stage, the high jitter problem was solved by the used of hash algoritm for faster packet searching. Therefore, it can be concluded that the proposed algorithm also reduced the amount of jitter significantly.

Consequently, the used of IP and MANET protocols, as well as using clients to served some of the IPTV service request, have proved to be a good technique to reduce bandwidth consumption and workload at the source node. The used of hash algorithm and adoption of OSPF election methods by the proposed algorithm enhanced the general IPTV quality of service. It also produced an intelligent algorithm that adapted to different server sircumstances.

Based on the analyses of results obtained, it can be concluded that, as the as the number of clients increases the efficiciency of the algorithm also increases.

Finally, all the research questions identified in this research have been answered.

8.3. Limitation and Difficulties Encountered

The new algorithm proposed in this research project is limited to IPTV services, where mobile clients are streaming video content from video streaming servers over the Internet. However, the algorithm can be extended to other services that are deployed over the Internet. The extension of the algorithm has been discussed in the future work section. During the course of this research work several difficulties were encountered, some of the major ones are explain in the following:

a. Time: The scenarios were simulated for 1 hour each, however the simulation took between 5 to 8 hours to complete.
b. Software: In some cases, the software used for the simulation (OPNET/Riverbed) crashed during simulation, after waiting for several hours for the simulation to complete.
c. Software License: OPNET require purchased license to work. Purchasing and renewal of the license took several weeks due to the University and OPNET bureaucracies.
d. Training: even though, I have basic knowledge on how to use the software (OPNET/Riverbed) initially, and I received intensive training to incorporate the new algorithm in it.

8.4 Contribution to knowledge

As mentioned in Chapter 1, the expected original contribution to knowledge of this research work can be verified and confirmed in the following points:

1. The leading originality of this research work is the systematic designing, developing and evaluating of the new intelligent routing algorithm for mobile IPTV called Intelligent Routing Algorithm for Mobile IPTV (IRA-MIPTV). Due to complexity of the algorithm, it was developed in four different stages. The first stage deals with

the unicast scheme only to provide video-on-demand service. The second stage combines the use of unicast and multicast schemes to deliver video-on-demand and live video programme services. The third stage is the extension of the first and second stages, where Content Delivery Network (CDN) architecture was added for load balancing across multiple servers. The fourth/final stage is the combination of all the three stages. The intelligent part of the algorithm was also added at this stage, where designated server and backup designated server are elected to address the issue of server failure and provide service delivery at all times.

2. Integrating the new algorithm into the standard server manager process in OPNET/Riverbed Modeler and the simulation of the IPTV network also provide the originality of the research work.
3. Based on the results obtained, the proposed algorithm adapts to different server capacities, such as bandwidth, number of request at a point in time and the location of the requesting client, to serve and an incoming request with the required quality of service. The algorithm also improved the IPTV general quality of service by providing high throughput and reduced bandwidth consumption, workload, packet end-to-end delay and jitter.

8.5 Future work

This research is limited to IPTV services where video streaming packets are transmitted from source to the destination over the Internet. However, the proposed algorithm can be extended to other services delivered over the Internet. The new proposed algorithm uses common protocol techniques and network architecture (such as MANET, IP, OSPF and

CDN) that are being used in todays computer communication. Therefore the algorithm can simply be extended to other services to improve in the QoS without requiring any changes on the underlying computer communication architecture. The QoS matrices considered in this reseach are Server overload, bandwidth consumption, packet end-to-end delay, jitter and throughput. Security was not included in the reseach focus area, eventhough, MD5 hash algorithm was used and provided basic level of security. In future work, security measures such as encryption mechanisms will be implemented to enhanced secure service delivery, so that the proposed algorithm can be suitable when extended to other services that require high level security measures.

Appendix A: Table of Data

Project 1 (30 mobile devices)

Intelligent Routing Algorithm for Mobile IPTV
Scenario 1 and 2

Load			End-to-End Delay			Throughput			Packet Delay variation (Jitter)		
Time (sec)	Scenario 1	Scenario 2	Time (sec)	Scenario 1	Scenario 2	Time (sec)	Scenario 1	Scenario 2	Time (sec)	Scenario 1	Scenario 2
0	56.88888889	88.8888889	0	#N/A	5.3793103	0	133937.78	169408	0	0.22111117	0.0052373
9	195751.1111	39427.5556	9	1.5264443	2.5478469	9	276849.78	352593.778	9	4.32832101	0.6840782
18	328280.8889	243185.778	18	3.3423545	2.2925258	18	322670.22	354464	18	6.27506879	1.886924
27	338567.1111	234769.778	27	3.9177609	2.5919428	27	309521.78	354464	27	6.29464532	3.5215294
36	305838.2222	245991.111	36	3.9265859	4.7046385	36	246926.22	304846.222	36	5.96751955	4.4670327
45	354464	265628.444	45	4.5935714	5.4910448	45	315189.33	382517.333	45	5.36303429	4.4917969
54	347861.3333	271182.222	54	2.0317267	2.8865854	54	333891.56	387192.889	54	4.83825374	4.4004484
63	359139.5556	266563.556	63	5.6590747	7.3722517	63	204846.22	380647.111	63	4.53435565	4.119727
72	260952.8889	274044.444	72	3.4291996	4.0742996	72	56.888889	381582.222	72	#N/A	3.8075854
81	56.88888889	277784.889	81	0	3.6168395	81	182403.56	387192.889	81	4.59568746	3.5143654
90	228224	273109.333	90	3.4552102	4.0216561	90	296430.22	382517.333	90	4.48747395	2.8657513
99	353528.8889	270304	99	3.6991206	4.7002567	99	317059.56	401162.667	99	4.14602301	2.6609307
108	330151.1111	271239.111	108	4.0831117	4.0438757	108	315189.33	380647.111	108	3.91879029	2.4612678
117	310456.8889	272174.222	117	7.1690352	6.1618981	117	316124.44	380647.111	117	3.71290292	2.3042586
126	358204.4444	274922.667	126	4.0731087	5.0590102	126	325475.56	371296	126	3.52197342	2.1683164
135	324540.4444	263758.222	135	3.4844607	4.543014	135	314254.22	388128	135	3.33839148	2.0534865
144	361944.8889	267498.667	144	4.3171642	4.5365854	144	318872.89	381525.333	144	3.21026249	1.9398884
153	340380.4444	275857.778	153	7.9985549	5.3695652	153	330094.22	371239.111	153	3.04484469	1.8409922
162	352536.8889	273052.444	162	7.4091791	6.1860768	162	318872.89	381525.333	162	2.94196786	1.7573572

171	341315.5556	264636.444		171	5.2041565	5.0487805	171	317937.78	383395.556	171	2.83610027	1.67874
180	358147.5556	262766.222		180	5.1956124	5.0292683	180	317002.67	374979.556	180	2.72010569	1.6029308
189	236583.1111	264636.444		189	7.0954015	5.0195122	189	317937.78	382460.444	189	2.59706847	1.5318215
198	325418.6667	258090.667		198	5.8980613	5.0316687	198	308586.67	371239.111	198	2.49668	1.4661222
207	340380.4444	259025.778		207	5.2898109	5.0298963	207	309521.78	370304	207	2.39841765	1.4052624
216	365628.4444	259960.889		216	5.3378049	4.9829268	216	302976	373109.333	216	2.31660001	1.3473693
225	345056	257155.556		225	5.4658537	5.0420475	225	308586.67	372174.222	225	2.25166611	1.2997991
234	347861.3333	259025.778		234	5.3382084	5.081147	234	308586.67	376849.778	234	2.17218861	1.2569816
243	367498.6667	258090.667		243	5.4109756	5.0085366	243	308586.67	391811.556	243	2.10700073	1.2118
252	354407.1111	279598.222		252	5.4036585	4.9792683	252	320743.11	389006.222	252	2.04428432	1.1690609
261	350666.6667	272117.333		261	5.3813301	4.9835466	261	330094.22	383395.556	261	1.99086256	1.1288821
270	331029.3333	256220.444		270	5.3601463	5.0177153	270	282403.56	382460.444	270	1.92590841	1.092825
279	361888	262766.222		279	5.4003656	5.0207065	279	296430.22	376849.778	279	2.56029606	1.060354
288	340380.4444	254350.222		288	5.1965916	5.0115924	288	319808	392746.667	288	2.39761626	1.0281874
297	345991.1111	258090.667		297	5.2870201	5.0091408	297	306716.44	375914.667	297	2.23136203	0.9976527
306	371239.1111	259960.889		306	5.1507016	5.5300245	306	288014.22	332899.556	306	2.06933784	0.970025
315	346926.2222	265571.556		315	5.2212066	6.7010444	315	293624.89	334769.778	315	1.96683326	0.97562
324	352536.8889	261571.556		324	5.0937881	6.5273413	324	315132.44	375914.667	324	1.84636059	0.96562
333	373109.3333	262571.556		333	6.0490683	6.5073413	333	299235.56	373109.333	333	1.75958857	0.95562
342	342250.6667	261571.556		342	5.4146341	6.5273413	342	303911.11	365628.444	342	1.67527574	0.94562
351	350666.6667	260571.556		351	5.455542	6.4973413	351	291754.67	378720	351	1.59220978	0.93562
360	344120.8889	259571.556		360	5.3223443	6.4773413	360	297365.33	372174.222	360	1.50416223	0.92562
369	360017.7778	258571.556		369	5.4	6.4573413	369	329159.11	388071.111	369	1.43147415	0.91562
378	350666.6667	261571.556		378	5.4109756	6.4473413	378	302976	384330.667	378	1.36876176	0.90562
387	356277.3333	260571.556		387	5.4040219	6.4273413	387	283338.67	345056	387	1.31232952	0.897562
396	348796.4444	259571.556		396	5.3239951	6.4373413	396	300170.67	344056	396	1.25633316	0.887562
405	340380.4444	257571.556		405	4.9951279	6.4173413	405	302976	342056	405	1.206685	0.87562
414	354407.1111	256571.556		414	5.007326	6.4073413	414	284273.78	341056	414	1.15495686	
423	360952.8889			423	5.1553931		423	331029.33		423	1.11972985	

432	361888			432	5.0164534		432	308586.67		432	1.08803214	
441	339445.3333			441	5.0707317		441	309521.78		441	1.05531869	
450	349731.5556			450	5.0774863		450	319808		450	1.02059049	
459	349731.5556			459	5.0018282		459	307651.56		459	3.46948147	
468	354407.1111			468	4.9037759		468	307651.56		468	3.10651593	
477	358147.5556			477	4.8073171		477	303911.11		477	2.57097374	
486	339445.3333			486	4.7890244		486	307651.56		486	2.19948206	
495	351601.7778			495	4.7897623		495	274922.67		495	1.91522089	
504	347861.3333			504	6.2336329		504	309521.78		504	1.68813276	
513	345991.1111			513	4.9872029		513	297365.33		513	1.52285904	
522	331964.4444			522	5.0164534		522	317937.78		522	1.37753863	
531	350666.6667			531	4.9829268		531	314197.33		531	1.27067951	
540	355342.2222			540	4.8994516		540	313262.22		540	1.15842844	
549	349731.5556			549	5.410625		549	303911.11		549	1.07580159	
558	255285.3333			558	7.1322537		558	288014.22		558	1.04114282	
567	356277.3333			567	5.9528243		567	316067.56		567	0.97103067	
576	341315.5556			576	5.5902439		576	307651.56		576	0.91053025	
585	350666.6667			585	5.6229408		585	305781.33		585	0.85848296	
594	355342.2222			594	5.5581254		594	330094.22		594	0.80650818	
603	329159.1111			603	5.5790116		603	332899.56		603	0.76923626	
612	341315.5556			612	5.597561		612	301105.78		612	0.73525517	
621	347861.3333			621	5.8469641		621	324483.56		621	0.69767886	
630	334769.7778			630	5.880706		630	329159.11		630	0.66624813	
639	335704.8889			639	5.7585366		639	298300.44		639	0.6398226	
648	337575.1111			648	5.7071385		648	314197.33		648	0.61046416	
657	375914.6667			657	5.6928702		657	297365.33		657	0.58240389	
666	362823.1111			666	5.7		666	302040.89		666	0.55739659	
675	289884.4444			675	6.7073643		675	314197.33		675	1.8194E-05	
684	351601.7778			684	5.7664234		684	308586.67		684	1.0434E-05	

693	310456.8889			693	5.8353659		693	274922.67		693	8.8844E-06	
702	336640			702	5.8171846		702	281468.44		702	7.6337E-06	
711	329159.1111			711	5.8243902		711	289884.44		711	6.4722E-06	
720	306716.4444			720	5.8244973		720	272117.33		720	5.675E-06	
729	332899.5556			729	5.7949969		729	287079.11		729	5.1151E-06	
738	319808			738	5.820841		738	280533.33		738	4.7683E-06	
747	310456.8889			747	5.8390244		747	279598.22		747	4.6868E-06	
756	314197.3333			756	5.8025594		756	266506.67		756	1.90927259	
765	310456.8889			765	5.6965265		765	283338.67		765	1.85244529	
774	334769.7778			774	5.4917733		774	287079.11		774	1.69420332	
783	313262.2222			783	5.4		783	275857.78		783	1.54763339	
792	333834.6667			792	5.4734918		792	#N/A		792	#N/A	
801	#N/A			801	#N/A							

Project 1 (30 mobile devices)

Intelligent Routing Algorithm for Mobile IPTV

Scenario 1 and 4

Load			End-to-End Delay			Throughput			Packet Delay variation (Jitter)			Link Utilization		
Time (sec)	Scenario 1	Scenario 4	Time (Sec)	Scenario 1	Scenario 4	Time (sec)	Scenario 1	Scenario 4	Time (sec)	Scenario 1	Scenario 4	Time (sec)	Scenario 1	Scenario 4
0	376291.5556	195751.1111	0	0.0002555	0.0006986	0	195751.11	376291.56	0	0	0	0	0.2650739	0.4378142
9	438624	328280.8889	9	0.0498667	0.1387205	9	328280.89	438624	9	5.1759206	3.828321014	9	7.0651004	7.2178635
18	438567.1111	338567.1111	18	0.086081	0.1389918	18	338567.11	438567.11	18	5.7109381	5.575068792	18	13.703308	13.845159
27	427402.6667	305838.2222	27	0.3860644	0.235225	27	305838.22	427402.67	27	5.8250688	5.594645316	27	19.45642	19.588815
36	416181.3333	354464	36	0.5614491	0.2666428	36	354464	416181.33	36	5.8098544	5.467519555	36	24.490394	24.614514
45	418051.5556	347861.3333	45	0.6843287	0.3341599	45	347861.33	418051.56	45	5.750527	4.863034289	45	28.932136	29.048954
54	322670.2222	359139.5556	54	0.89063	0.3249603	54	359139.56	402670.22	54	#N/A	4.338253745	54	32.88035	32.990679
63	346983.1111	260952.8889	63	0.8437548	0.3380842	63	290952.89	406983.11	63	5.7346495	4.034355645	63	36.412963	36.517486
72	406830.2222	240785.8889	72	0.8090243	0.3638902	72	277453.89	406830.22	72	5.437074	#N/A	72	39.592315	39.691611
81	413319.1111	228224	81	1.1162006	0.3555161	81	288224	413319.11	81	5.2775182	4.095687457	81	42.468872	42.563439
90	428337.7778	353528.8889	90	1.1045892	0.3473071	90	353528.89	428337.78	90	5.059415	3.987473949	90	45.083923	45.174192
99	431143.1111	330151.1111	99	1.0585424	0.3401863	99	330151.11	431143.11	99	4.8692815	3.646023015	99	47.471579	47.557923
108	425532.4444	310456.8889	108	1.1126343	0.3336724	108	310456.89	425532.44	108	4.4358739	3.418790289	108	49.660263	49.743009
117	431143.1111	358204.4444	117	1.2244185	0.3272131	117	358204.44	431143.11	117	#N/A	3.212902915	117	51.673852	48.146033
126	433013.3333	324540.4444	126	1.1773254	0.5990333	126	324540.44	433013.33	126	4.4076025	3.021973421	126	53.53255	47.405825
135	431086.2222	361944.8889	135	1.2057779	0.6092045	135	361944.89	431086.22	135	4.2827147	2.838391478	135	55.253567	49.353757
144	419864.8889	340380.4444	144	1.2315147	0.7497702	144	340380.44	419864.89	144	4.1048236	2.710262493	144	56.851654	51.162552
153	400227.5556	352536.8889	153	1.3565661	0.7939857	153	352536.89	400227.56	153	3.9632024	2.544844689	153	58.339528	52.846602
162	445112.8889	341315.5556	162	1.3346298	0.7849309	162	341315.56	445112.89	162	3.7701439	2.441967864	162	59.72821	54.418382
171	431086.2222	358147.5556	171	1.3151185	0.7767153	171	358147.56	431086.22	171	3.64965	2.336100275	171	61.0273	55.888756
180	404903.1111	236583.1111	180	1.2982744	0.989756	180	336583.11	404903.11	180	3.5386342	2.220105687	180	62.245197	57.267233

189	430151.1111	325418.6667	189	1.3202531	1.0439702	189	325418.67	430151.11	189	3.1470474	2.097068472	189	63.389282	58.562165
198	444177.7778	340380.4444	198	1.5582377	1.0349056	198	340380.44	444177.78	198	3.0454732	1.996679999	198	64.466068	59.780925
207	409578.6667	365628.4444	207	1.673627	1.0293109	207	365628.44	409578.67	207	2.9188626	1.898417648	207	65.481323	60.930041
216	433891.5556	345056	216	1.7310743	1.0289894	216	345056	433891.56	216	2.8336674	1.816600006	216	66.440175	62.015318
225	446048	347861.3333	225	1.8945837	1.026272	225	347861.33	446048	225	2.8331067	1.751666108	225	67.347197	63.041931
234	425475.5556	367498.6667	234	1.8463645	1.019793	234	367498.67	425475.56	234	2.7515706	1.672188605	234	68.206482	64.014512
243	417994.6667	354407.1111	243	1.8005118	1.0240181	243	354407.11	417994.67	243	2.7010775	1.607000727	243	69.0217	64.937217
252	441372.4444	350666.6667	252	1.756839	1.0189957	252	350666.67	441372.44	252	2.6361368	1.544284317	252	69.796158	65.813786
261	433891.5556	331029.3333	261	1.7152956	1.0112038	261	331029.33	433891.56	261	2.5839121	1.490862562	261	70.532837	66.647596
270	432021.3333	361888	270	1.7566089	1.007161	270	361888	432021.33	270	2.5386241	1.425908408	270	71.234436	67.441701
279	418929.7778	340380.4444	279	1.7953357	1.0076007	279	340380.44	418929.78	279	2.502452	1.430296058	279	71.903402	68.198871
288	428280.8889	345991.1111	288	1.783001	1.0043974	288	345991.11	428280.89	288	2.478595	1.446162634	288	72.541961	68.921624
297	447918.2222	371239.1111	297	1.7447197	1.0027845	297	371239.11	447918.22	297	2.406784	1.453136203	297	73.15214	69.612254
306	424540.4444	346926.2222	306	1.72007	0.999073	306	346926.22	424540.44	306	2.39365	1.446933784	306	73.735789	70.272858
315	446983.1111	352536.8889	315	1.6848381	0.9922644	315	352536.89	446983.11	315	2.46769	1.406833264	315	74.294602	70.90535
324	412384	373109.3333	324	1.8060443	0.9874617	324	373109.33	412384	324	2.564382	1.346360593	324	74.830131	71.511488
333	429216	342250.6667	333	1.7704476	0.9827816	333	342250.67	429216	333	2.6064367	1.259588573	333	75.343802	72.092887
342	435761.7778	350666.6667	342	1.7363759	0.9808929	342	350666.67	435761.78	342	2.456383	1.175275741	342	75.836926	72.651029
351	424540.4444	344120.8889	351	1.7034637	1.0266885	351	344120.89	424540.44	351	2.3838406	1.092209779	351	76.310712	73.187283
360	453528.8889	360017.7778	360	1.6719754	1.1032412	360	360017.78	453528.89	360	2.2795347	1.004162233	360	76.766275	73.702912
369	413319.1111	350666.6667	369	1.6414149	1.1064858	369	350666.67	413319.11	369	2.1375604	0.931474147	369	77.204647	74.199084
378	422670.2222	356277.3333	378	1.6121767	1.1059764	378	356277.33	422670.22	378	2.0341272	0.868761764	378	77.626783	74.676879
387	449788.4444	348796.4444	387	1.5841251	1.1082779	387	348796.44	449788.44	387	1.9370281	0.812329523	387	78.033569	75.137299
396	430972.4444	340380.4444	396	1.5569464	1.1628177	396	340380.44	430972.44	396	1.8629232	0.756333157	396	78.425827	75.581276
405	418816	354407.1111	405	1.5308226	1.1631882	405	354407.11	418816	405	1.7986525	0.706685005	405	77.343549	74.548847
414	439388.4444	360952.8889	414	1.5054992	1.1452362	414	360952.89	439388.44	414	1.7213613	0.654956857	414	76.010039	73.263522
423	412270.2222	361888	423	1.4810939	1.1279335	423	361888	412270.22	423	1.6600778	0.619729849	423	74.721733	72.021768
432	419751.1111	339445.3333	432	1.4574445	1.1616271	432	339445.33	419751.11	432	1.577416	0.588032142	432	73.476371	70.821405
441	420686.2222	349731.5556	441	1.4344947	1.1403938	441	349731.56	420686.22	441	1.5238543	0.555318686	441	72.27184	69.660398

450	415075.5556	349731.5556	450	1.4123052	1.1198057	450	349731.56	415075.56	450	1.4644102	0.520590488	450	71.106166	68.536843
459	398243.5556	354407.1111	459	1.3908065	1.1998376	459	354407.11	418243.56	459	1.4121899	0.969481473	459	69.977496	67.448957
468	404789.3333	358147.5556	468	1.370058	1.1805748	468	358147.56	414789.33	468	1.4757325	0.906515933	468	68.884098	66.395067
477	419751.1111	339445.3333	477	1.3498568	1.1617639	477	339445.33	419751.11	477	1.5323353	0.870973738	477	67.824343	65.373604
486	413205.3333	351601.7778	486	1.3303268	1.1434404	486	351601.78	413205.33	486	1.6187958	0.89948206	486	66.796701	64.383095
495	415075.5556	347861.3333	495	1.3113155	1.1334204	495	357861.33	415075.56	495	1.7501565	0.815220891	495	65.799735	63.422153
504	405724.4444	345991.1111	504	1.2929132	1.1168558	504	345991.11	405724.44	504	1.6923478	0.888132764	504	64.832092	62.489475
513	415075.5556	331964.4444	513	1.2750921	1.1043166	513	331964.44	415075.56	513	1.8413001	0.722859037	513	63.892497	61.58383
522	408529.7778	350666.6667	522	1.2576685	1.3075567	522	350666.67	308243.66	522	1.7084662	1.777538632	522	62.979747	60.704061
531	415075.5556	355342.2222	531	1.2409037	1.1011234	531	355342.22	325075.56	531	1.9863653	0.770679507	531	62.092708	59.849074
540	413205.3333	349731.5556	540	1.2245493	1.1012246	540	349731.56	313205.33	540	1.858973	0.658428439	540	61.230309	59.017837
549	408529.7778	445285.3333	549	1.208619	1.1012874	549	355285.33	308529.78	549	1.6818972	0.575801591	549	60.391538	58.209374
558	419751.1111	436277.3333	558	1.1931064	1.101264	558	356277.33	419751.11	558	1.6356998	0.541142825	558	59.575436	57.422761
567	412270.2222	421315.5556	567	1.1779923	1.1013881	567	341315.56	412270.22	567	1.5542254	0.471030668	567	58.781097	56.657124
576	409464.8889	400666.6667	576	1.1632708	1.1014451	576	350666.67	409464.89	576	1.4839542	0.410530246	576	58.007661	55.911635
585	422556.4444	395342.2222	585	1.1488641	1.1005405	585	355342.22	422556.44	585	1.4439421	0.358482958	585	57.254315	55.18551
594	415075.5556	379159.1111	594	1.1348335	1.1000869	594	329159.11	415075.56	594	1.3603226	0.306508179	594	56.520285	54.478004
603	421621.3333	361315.5556	603	1.1237021	1.0015136	603	341315.56	421621.33	603	1.2855995	0.269236257	603	55.804839	53.788409
612	422556.4444	347861.3333	612	1.1104781	1.0019054	612	347861.33	422556.44	612	1.2313769	0.23525517	612	55.107278	53.116054
621	417880.8889	334769.7778	621	1.0974165	1.0759433	621	334769.78	417880.89	621	1.1556466	0.19767886	621	54.426942	52.4603
630	406659.5556	335704.8889	630	1.0848208	1.0074205	630	335704.89	406659.56	630	1.1135973	0.166248135	630	53.763198	51.82054
639	407594.6667	337575.1111	639	1.0724191	1.0158542	639	337575.11	407594.67	639	1.0685317	0.139822601	639	53.115449	51.196196
648	416945.7778	375914.6667	648	1.0602932	1.0054797	648	375914.67	416945.78	648	1.0179128	0.110464162	648	52.483122	50.586718
657	411335.1111	362823.1111	657	1.048639	1.0090174	657	362823.11	411335.11	657	0.9510711	0.082403892	657	51.865674	49.99158
666	420686.2222	289884.4444	666	1.0372175	1.0095439	666	289884.44	420686.22	666	0.9106635	0.05739659	666	51.262585	49.410282
675	418816	351601.7778	675	1.0260359	1.0090409	675	351601.78	418816	675	0.8815054	0.04521	675	50.673359	48.842348
684	409592.8889	310456.8889	684	1.017566	1.0085568	684	310456.89	409592.89	684	0.8541456	0.02132	684	50.097526	48.287321
693	403854.2222	336640	693	1.0467303	1.0080639	693	336640	403854.22	693	0.8332013	0.010234	693	49.534632	47.744767
702	409464.8889	329159.1111	702	1.0357918	1.007511	702	329159.11	409464.89	702	0.8075641	0.003343	702	48.984247	47.21427

711	409464.8889	306716.4444	711	1.025057	1.0074334	711	306716.44	409464.89	711	0.7802803	0.004322	711	48.445959	46.695432
720	414140.4444	332899.5556	720	1.0145622	1.007058	720	332899.56	414140.44	720	0.7547628	0.004222	720	47.919372	46.187873
729	416010.6667	319808	729	1.0043553	1.0065281	729	319808	416010.67	729	0.7280848	0.00412	729	47.40411	45.691229
738	413205.3333	310456.8889	738	0.9943893	1.0060775	738	310456.89	413205.33	738	0.7075338	0.004098	738	46.899811	45.205152
747	416010.6667	314197.3333	747	1.0984631	1.0057614	747	314197.33	416010.67	747	0.6864679	0.03902	747	46.406129	44.729308
756	414140.4444	310456.8889	756	1.0974961	1.005236	756	310456.89	414140.44	756	0.6582386	0.0038	756	45.922732	44.263378
765	405724.4444	334769.7778	765	1.0965579	1.0046488	765	334769.78	405724.44	765	0.6392445	0.037	765	45.449302	43.807054
774	406659.5556	313262.2222	774	1.0956369	1.0040985	774	313262.22	406659.56	774	0.6199773	0.036	774	44.985533	43.360044
783	417880.8889	333834.6667	783	1.0947366	1.0035064	783	333834.67	417880.89	783	0.6030953	0.035	783	44.531134	42.922063
792	#N/A	#N/A	792	1.0938414	1.0032085	792	#N/A	#N/A	792	#N/A	#N/A	792	44.085823	42.492843

Project 1 (30 mobile devices)

Intelligent Routing Algorithm for Mobile IPTV

Scenario 3 and 4

Load			End-to-End Delay			Throughput			Packet Delay variation (Jitter)		
Time (sec)	Scenario 3	Scenario 4	Time (sec)	Scenario 3	Scenario 4	Time (sec)	Scenario 3	Scenario 4	Time (sec)	Scenario 3	Scenario 4
0	0	0	0	0.0793936	0.015345003	0	135840	135840	0	0.0596552	0.05965519
9	56.88888889	56.888889	9	0.322498	0.287083957	9	390933.33	390933.33	9	1.1469468	1.14694678
18	56.88888889	56.888889	18	0.2660369	0.180035069	18	451715.56	451715.56	18	3.3563194	3.35631942
27	30979.55556	30979.556	27	4.0696035	3.793473075	27	377841.78	377841.78	27	4.4579709	4.45797091
36	368618.6667	368618.67	36	2.3043906	1.55695906	36	408643.56	408643.56	36	4.4684974	4.46849741
45	652764.4444	652764.44	45	1.9760235	1.642722162	45	424597.33	424597.33	45	4.4520979	4.45209794
54	658375.1111	658375.11	54	0.3290283	0.283864541	54	404024.89	404024.89	54	4.3866428	4.38664281
63	576028.4444	576028.44	63	1.7646835	1.885113057	63	425532.44	425532.44	63	3.7103221	3.71032214
72	454520.8889	454520.89	72	2.8609445	2.870611044	72	410570.67	410570.67	72	3.4725649	3.47256487
81	625646.2222	625646.22	81	0.3802396	0.236746738	81	353528.89	353528.89	81	3.1789351	3.17893507
90	610684.4444	610684.44	90	0.3709419	0.231229281	90	415189.33	415189.33	90	2.9827132	2.98271317
99	610684.4444	610684.44	99	0.3774991	0.22497608	99	404024.89	404024.89	99	2.7839814	2.78398138
108	622840.8889	622840.89	108	0.3902868	0.223813408	108	410570.67	410570.67	108	2.6184661	2.6184661
117	592917.3333	592917.33	117	0.3715253	1.126459946	117	402154.67	402154.67	117	2.4623621	2.46236207
126	626524.4444	626524.44	126	13.259026	0.257405873	126	386257.78	386257.78	126	2.3304723	2.33047233
135	620035.5556	620035.56	135	0.546562	0.238572257	135	427402.67	427402.67	135	2.1924176	2.19241762
144	610684.4444	610684.44	144	3.9755104	0.242303983	144	429216	429216	144	2.0911761	2.0911761
153	636867.5556	636867.56	153	3.9192231	1.672511994	153	402097.78	402097.78	153	1.9942727	1.99427273
162	565799.1111	565799.11	162	0.5839158	0.434899399	162	415189.33	415189.33	162	1.8911098	1.89110979
171	454520.8889	454520.89	171	0.5877104	0.44154962	171	427345.78	427345.78	171	1.804243	1.80424303
180	587249.7778	587249.78	180	0.5851631	0.369951622	180	403968	403968	180	1.726296	1.72629598
189	612497.7778	612497.78	189	0.5279289	6.453241145	189	373109.33	373109.33	189	1.6761782	1.6761782

198	594730.6667	594730.67	198	0.5596682	7.61071226	198	351601.78	351601.78	198	1.6437134	1.64371344
207	578833.7778	578833.78	207	0.5572529	0.685303182	207	395552	395552	207	1.5639768	1.56397676
216	598471.1111	598471.11	216	0.6159695	0.630871203	216	386200.89	386200.89	216	1.5077661	1.50776607
225	583509.3333	583509.33	225	0.5888135	0.776618372	225	404903.11	404903.11	225	1.4549162	1.45491617
234	576963.5556	576963.56	234	0.5566166	0.716605082	234	408643.56	408643.56	234	1.3996999	1.39969994
243	594730.6667	594730.67	243	0.5399461	0.654132023	243	415189.33	415189.33	243	1.3530555	1.35305554
252	579768.8889	579768.89	252	0.6084768	0.71571322	252	403032.89	403032.89	252	1.3080943	1.30809425
261	572288	572288	261	0.5476443	0.752777377	261	426410.67	426410.67	261	1.2684012	1.26840117
270	581639.1111	581639.11	270	0.5407491	0.732090238	270	403032.89	403032.89	270	1.2287611	1.22876114
279	576028.4444	576028.44	279	0.6082879	0.701580671	279	422670.22	422670.22	279	1.1950696	1.19506957
288	589120	589120	288	0.5763194	0.650965254	288	403968	403968	288	1.1580676	1.15806759
297	587249.7778	587249.78	297	0.5646372	0.694343297	297	391811.56	391811.56	297	1.1229931	1.12299306
306	579768.8889	579768.89	306	0.5481794	0.700750661	306	398357.33	398357.33	306	1.0926315	1.09263151
315	598471.1111	598471.11	315	0.5539444	0.680177716	315	361888	361888	315	1.063573	1.06357297
324	591925.3333	591925.33	324	0.5493488	0.727722617	324	405838.22	405838.22	324	1.0341191	1.03411907
333	625589.3333	625589.33	333	0.5552171	0.741313661	333	393681.78	393681.78	333	1.0341191	1.03411907
342	554520.8889	554520.89	342	0.532597	0.000012	342	432021.33	432021.33	342	1.0341191	1.03411907
351	567612.4444	567612.44	351	0.5361723	0.000011	351	398357.33	398357.33	351	1.0341191	1.03411907
360	587249.7778	587249.78	360	0.3342297	0.000002	360	372174.22	372174.22	360	1.0341191	1.03411907
369	598471.1111	598471.11	369	0.4545246	0.000005	369	360017.78	360017.78	369	1.1229931	1.06357297
378	636810.6667	636810.67	378	0.3832587	0.000033	378	381525.33	381525.33	378	1.1229931	1.06357297
387	615303.1111	615303.11	387	0.4140516	0.000001	387	360017.78	360017.78	387	1.1229931	1.06357297
396	589120	589120	396	0.500144	0.000002	396	144007.11	206659.56	396	0.000012	0.000008
405	602211.5556	602211.56	405	0.2529789	0.000003	405	132785.78	180476.44	405	0.000012	0.000008
414	533948.4444	533948.44	414	0.1665184	0.000008	414	121564.44	192632.89	414	#N/A	#N/A
423	605016.8889	605016.89	423	0.3407652	0.000001	423	115953.78	198243.56	423	0.8875872	0.83053701
432	389006.2222	389006.22	432	0.2785248	0.000004	432	115018.67	201048.89	432	0.8875872	0.83053701
441	295495.1111	295495.11	441	0.2023352	0.000002	441	129045.33	207594.67	441	0.8875872	0.83053701
450	366563.5556	293624.89	450	0.230221	0.000005	450	137461.33	197308.44	450	1.0961871	1.09618707

459	388071.1111	314197.33	459	0.2138314	0.000003	459	121564.44	216010.67	459	1.0961871	1.09618707
468	374979.5556	315132.44	468	0.1982139	0.00004	468	110343.11	206659.56	468	1.0961871	1.09618707
477	377784.8889	314197.33	477	0.1449745	0.000001	477	129045.33	199178.67	477	1.0961871	1.09618707
486	382460.4444	307651.56	486	0.1338719	0.125276032	486	107537.78	199178.67	486	1.0807499	1.08074989
495	398357.3333	307651.56	495	0.1432496	0.137024784	495	129045.33	205724.44	495	1.0360667	1.03606671
504	398357.3333	287079.11	504	0.1487103	0.147791445	504	129045.33	208529.78	504	0.9933863	0.99338631
513	385265.7778	282403.56	513	0.1405586	0.139873169	513	111278.22	196373.33	513	0.9572098	0.95720978
522	396487.1111	279598.22	522	0.1439103	0.1333114	522	125304.89	203854.22	522	0.9215826	0.92158259
531	378720	291754.67	531	0.1520731	0.144781732	531	116888.89	205724.44	531	0.8875872	0.88758721
540	381525.3333	279598.22	540	0.1403647	0.125269366	540	125304.89	207594.67	540	0.8579761	0.85797606
549	394616.8889	297365.33	549	0.1387889	0.123990127	549	113148.44	199178.67	549	5.0622795	0.83053701
558	385265.7778	280533.33	558	0.1379757	0.126141379	558	112213.33	190762.67	558	5.0076182	0.80453876
567	380590.2222	294560	567	0.1367334	0.123974943	567	115018.67	212270.22	567	4.8335092	0.77869946
576	387136	305781.33	576	0.1604209	0.147087074	576	109408	186087.11	576	4.6383326	4.31085149
585	386200.8889	292689.78	585	0.1415933	0.131507911	585	113148.44	197308.44	585	4.4482687	4.26668086
594	384330.6667	298300.44	594	0.1419976	0.127645035	594	119694.22	204789.33	594	4.3065735	4.05089569
603	390876.4444	284273.78	603	0.1515865	0.133894397	603	127175.11	204789.33	603	5.0601293	3.82916136
612	391811.5556	284273.78	612	0.1457625	0.13451279	612	112213.33	180476.44	612	5.0605791	3.60177477
621	394616.8889	292689.78	621	0.1577253	0.137204294	621	125304.89	189827.56	621	4.8606593	3.35822652
630	396487.1111	278663.11	630	0.1536462	0.126910133	630	116888.89	201984	630	4.5344081	3.16652383
639	379655.1111	282403.56	639	0.1495514	0.136051662	639	125304.89	199178.67	639	4.1391091	2.99942448
648	376849.7778	280533.33	648	0.1522278	0.138116137	648	113148.44	197308.44	648	3.8204878	2.8468969
657	382460.4444	307651.56	657	0.1571393	0.128213187	657	111278.22	189827.56	657	3.561078	2.70756953
666	391811.5556	285208.89	666	0.1451802	0.133798193	666	120629.33	196373.33	666	3.3156546	2.5787881
675	398357.3333	281468.44	675	0.1511507	0.131716152	675	108472.89	204789.33	675	3.0789892	2.45316757
684	383395.5556	296430.22	684	0.1409027	0.130774756	684	130044.44	206659.56	684	2.9012583	2.34686231
693	384330.6667	264636.44	693	0.1368544	0.126497072	693	121564.44	185152	693	2.7550586	2.25002698
702	389941.3333	286144	702	0.1507101	0.139687948	702	121564.44	194503.11	702	2.6741046	2.16764234
711	388071.1111	303911.11	711	0.1634315	0.157482025	711	109408	200113.78	711	2.610008	2.09183965

720	399356.4444	298300.44	720	0.144334	0.157502148	720	113148.44	197308.44	720	2.5487653	2.01567579
729	390876.4444	346926.22	729	0.1395671	0.14326	729	115018.67	206659.56	729	2.4883222	1.94059268
738	406773.3333	373109.33	738	0.1541748	0.163859971	738	113148.44	204789.33	738	2.4214873	1.86540126
747	422670.2222	378720	747	0.1521154	0.155607383	747	127175.11	184216.89	747	2.3503339	1.80281641
756	413319.1111	392746.67	756	0.1665635	0.151151697	756	115953.78	201984	756	2.2871568	1.73752297
765	417994.6667	379655.11	765	0.1587185	0.173708563	765	114083.56	186087.11	765	2.2256487	1.67858401
774	412384	368433.78	774	0.1549069	0.624698629	774	120629.33	197308.44	774	2.1738674	1.63511213
783	400227.5556	373109.33	783	0.1447458	0.210601129	783	119694.22	194503.11	783	2.1236701	1.59552123
792	402097.7778	387136	792	0.159447	0.177726063	792	#N/A	#N/A	792	#N/A	#N/A

Table of Data: Project 2 (70 mobile devices)

Intelligent Routing Algorithm for Mobile IPTV

Scenario 1 and 2

Load			End-to-End Delay			Throughput			Packet Delay variation (Jitter)		
Time (sec)	Scenario 1	Scenario 2	Time (sec)	Scenario 1	Scenario 2	Time (sec)	Scenario 1	Scenario 2	Time (sec)	Scenario 1	Scenario 2
0	23466.667	56.888889	0	0.0069177	0.00304129	0	133937.778	169408	0	0.0002558	6.51E-05
9	129198.22	45006.222	9	0.1222543	0.14068115	9	276849.778	352593.78	9	1.4745332	0.9737335
18	101984	57098.667	18	0.0803846	0.06374436	18	322670.222	354464	18	5.1759206	4.8270972
27	141258.67	64579.556	27	0.4667982	0.26776824	27	309521.778	354464	27	5.7109381	5.4992136
36	166506.67	120686.22	36	0.4327929	0.0527284	36	246926.222	304846.22	36	5.8250688	5.6408296
45	156220.44	113205.33	45	0.4521612	0.64102567	45	315189.333	382517.33	45	5.8098544	5.606099
54	76679.111	105667.56	54	0.4182021	0.23800205	54	333891.556	387192.89	54	5.750527	5.3787991
63	56.888889	101048.89	63	0.0015372	0.43911037	63	204846.222	380647.11	63	#N/A	5.0813185
72	123491.56	117880.89	72	0.0138695	0.01262663	72	56.8888889	381582.22	72	5.7346495	4.8351287
81	155285.33	120686.22	81	0.0521244	0.01251533	81	182403.556	387192.89	81	5.437074	4.5463894
90	166506.67	115075.56	90	0.022146	0.01263997	90	296430.222	382517.33	90	5.2775182	4.2697952
99	160896	98243.556	99	0.0153327	0.01362713	99	317059.556	401162.67	99	5.059415	4.0321094
108	174865.78	113205.33	108	0.2479328	0.11611575	108	315189.333	380647.11	108	4.8692815	3.7952325
117	82346.667	114083.56	117	0.1116143	0.26218067	117	316124.444	380647.11	117	4.4358739	3.5533101
126	56.888889	102919.11	126	0.0015376	0.55032221	126	325475.556	371296	126	#N/A	3.3865227
135	120686.22	120686.22	135	0.1851896	0.01384889	135	314254.222	388128	135	4.4076025	3.2366939
144	143128.89	118816	144	0.2000157	0.01347333	144	318872.889	381525.33	144	4.2827147	3.1151696
153	143072	118759.11	153	0.8565664	0.33202292	153	330094.222	371239.11	153	4.1048236	2.968506
162	168320	100992	162	0.0972321	0.01359881	162	318872.889	381525.33	162	3.9632024	2.6227134
171	158968.89	130915.56	171	0.2187935	0.37080583	171	317937.778	383395.56	171	3.7701439	2.5079538
180	160839.11	114083.56	180	0.1053398	0.18254646	180	317002.667	374979.56	180	3.64965	1.8574431

189	142136.89	118759.11	189	0.22953	0.01642376	189	317937.778	382460.44	189	3.5386342	1.7850845
198	163644.44	134656	198	0.1981035	0.01763929	198	308586.667	371239.11	198	3.1470474	1.5076463
207	147747.56	125304.89	207	0.2015651	0.01642864	207	309521.778	370304	207	3.0454732	1.4409899
216	147747.56	123434.67	216	0.4342019	0.01462037	216	302976	373109.33	216	2.9188626	1.39405
225	168320	100056.89	225	1.3332786		225	308586.667	372174.22	225	2.8336674	1.38405
234	139331.56	108472.89	234	0.0245347		234	308586.667	376849.78	234	1.8331067	1.37405
243	144007.11	127175.11	243	0.0217771		243	308586.667	391811.56	243	1.7515706	1.36405
252	167384.89	115953.78	252	0.02009		252	320743.111	389006.22	252	1.7010775	1.35405
261	159904	100992	261	0.0270233		261	330094.222	383395.56	261	1.6361368	1.39436
270	148682.67	116888.89	270	0.0227788		270	282403.556	382460.44	270	1.5839121	1.36444
279	152423.11	103797.33	279	0.0204597		279	296430.222	376849.78	279	1.5386241	1.329022
288	143072	91640.889	288	0.0213766		288	319808	392746.67	288	0.6235829	1.39405
297	173930.67	117824	297	0.0175406		297	306716.444	375914.67	297	0.6017512	1.30345
306	167384.89	110343.11	306	0.0348684		306	288014.222	332899.56	306	0.5808669	1.3401004
315	162709.33	109408	315	#N/A		315	293624.889	334769.78	315	#N/A	1.35405
324	158968.89	115018.67	324	0.0234735		324	315132.444	375914.67	324	0.9612017	1.33871
333	163644.44	96316.444	333	0.0158264		333	299235.556	373109.33	333	0.950531	1.309802
342	161774.22	108472.89	342	0.0142016		342	303911.111	365628.44	342	0.8911759	1.298325
351	162709.33	104732.44	351	0.0173088		351	291754.667	378720	351	0.8401389	1.298952
360	186087.11	107537.78	360	0.0208934		360	297365.333	372174.22	360	2.3838406	1.29405
369	143072	114083.56	369	0.0165101		369	329159.111	388071.11	369	2.2795347	1.2865422
378	149617.78	115018.67	378	0.0182131		378	302976	384330.67	378	2.1375604	1.2888004
387	144942.22	117824	387	0.0244136		387	283338.667	345056	387	2.0341272	1.277805
396	160839.11	150552.89	396	0.019254		396	300170.667	344056	396	1.9370281	1.3012405
405	199178.67	125248	405	0.0195339		405	302976	342056	405	1.8629232	1.3356405
414	144007.11		414	0.0195123		414	284273.778	341056	414	1.7986525	1.344405
423	157098.67		423	0.0202439		423	331029.333		423	1.7213613	
432	173930.67		432	0.0155613		432	308586.667		432	1.6600778	
441	159904		441	0.0160756		441	309521.778		441	1.577416	

450	165514.67		450	0.0208361		450	319808		450	1.5238543	
459	154293.33		459	0.0175496		459	307651.556		459	1.4644102	
468	157098.67		468	0.0206026		468	307651.556		468	1.4121899	
477	165514.67		477	0.0188086		477	303911.111		477	1.3757325	
486	164579.56		486	0.0162313		486	307651.556		486	2.1323353	
495	149617.78		495	0.0188485		495	274922.667		495	2.0187958	
504	164579.56		504	0.01926		504	309521.778		504	2.5015654	
513	155228.44		513	0.0176013		513	297365.333		513	2.4923478	
522	160839.11		522	0.0155528		522	317937.778		522	2.28413	
531	172995.56		531	0.0175803		531	314197.333		531	2.1084662	
540	153358.22		540	0.0178553		540	313262.222		540	1.9863653	
549	153358.22		549	0.017397		549	303911.111		549	1.858973	
558	155228.44		558	0.017276		558	288014.222		558	1.6818972	
567	166449.78		567	0.0170712		567	316067.556		567	1.6356998	
576	167384.89		576	0.0177705		576	307651.556		576	1.5542254	
585	152423.11		585	0.0175656		585	305781.333		585	1.4839542	
594	142136.89		594	0.0171684		594	330094.222		594	1.4439421	
603	157098.67		603	0.0165057		603	332899.556		603	1.3603226	
612	154293.33		612	0.015143		612	301105.778		612	1.2855995	
621	145877.33		621	0.0140806		621	324483.556		621	1.2313769	
630	186087.11		630	0.0177248		630	329159.111		630	1.1556466	
639	171125.33		639	0.0161691		639	298300.444		639	1.1135973	
648	158968.89		648	0.0157325		648	314197.333		648	1.0685317	
657	173930.67		657	0.0128403		657	297365.333		657	1.0179128	
666	158968.89		666	0.0150041		666	302040.889		666	0.9510711	
675	153358.22		675	0.0151864		675	314197.333		675	0.9106635	
684	148682.67		684	0.0606446		684	308586.667		684	0.8815054	
693	175800.89		693	2.6082184		693	274922.667		693	0.8541456	
702	166449.78		702	0.0151931		702	281468.444		702	0.8332013	

711	162709.33		711	0.0181653		711	289884.444		711	0.8075641	
720	151488		720	0.0152548		720	272117.333		720	0.7802803	
729	176736		729	0.0159238		729	287079.111		729	0.7547628	
738	156163.56		738	0.0141059		738	280533.333		738	0.7280848	
747	160839.11		747	0.0160876		747	279598.222		747	0.7075338	
756	162709.33		756	0.0175681		756	266506.667		756	0.6864679	
765	161774.22		765	0.0186745		765	283338.667		765	0.6582386	
774	153358.22		774	0.0166633		774	287079.111		774	0.6392445	
783	165514.67		783	0.0157005		783	275857.778		783	0.6199773	
792	155228.44		792	0.0156188		792	#N/A		792	0.6030953	

Table of Data: Project 2 (70 mobile devices)

Intelligent Routing Algorithm for Mobile IPTV

Scenario 1 and 4

Load			End-to-End Delay			Throughput			Packet Delay variation (Jitter)			Link Utilization		
Time (sec)	Scenario 1	Scenario 4	Time (Sec)	Scenario 1	Scenario 4	Time (sec)	Scenorio 1	Scenario 4	Time (sec)	Scenario 1	Scenario 4	Time (sec)	Scenario 1	Scenario 4
0	525781.3	133937.8	0	0.12114	0.13872	0	215260.4	315381.3	0	0.000259	#N/A	0	0.249448	0.064599
9	652764.4	276849.8	9	0.118181	0.138992	9	350723.6	658375.1	9	0.551794	0.0596552	9	7.504679	7.437854
18	656448	322670.2	18	0.207297	0.235225	18	254407.1	655569.8	18	2.40278	0.603301	18	14.11149	14.04944
27	733184	309521.8	27	0.231907	0.266643	27	248739.6	737802.7	27	3.368128	1.5209738	27	19.83739	19.77947
36	708871.1	246926.2	36	0.319932	0.33416	36	267498.7	556448	36	3.869721	2.2552231	36	24.84755	24.79326
45	715416.9	315189.3	45	0.312208	0.32496	45	271239.1	764977.8	45	4.187095	2.6978779	45	29.26828	29.21718
54	730378.7	333891.6	54	0.328199	0.338084	54	175857.8	779939.6	54	4.402615	2.9902479	54	30.9642	33.14956
63	733184	204846.2	63	0.559052	0.36389	63	132842.7	427402.7	63	4.546486	3.1897329	63	29.3353	36.66801
72	733184	190888.9	72	0.735411	0.355516	72	289006.2	696714.7	72	4.633597	3.2548066	72	31.81653	39.83461
81	739672.9	182403.6	81	0.89256	0.347307	81	263758.2	707936	81	4.676409	3.2790019	81	35.06336	42.69962
90	700455.1	296430.2	90	1.039951	0.340186	90	259960.9	707000.9	90	4.685883	3.2689953	90	38.01503	45.30419
99	727573.3	317059.6	99	1.167902	0.333672	99	260017.8	705073.8	99	4.678448	3.2429696	99	40.71002	47.68227
108	713546.7	315189.3	108	1.317711	0.327213	108	149674.7	706065.8	108	4.656423	3.2047206	108	43.18044	49.86217
117	748145.8	316124.4	117	1.425032	0.599033	117	177728	706065.8	117	4.625254	3.1596241	117	43.80359	51.86768
126	721962.7	325475.6	126	1.527164	0.609205	126	297422.2	706065.8	126	4.584043	3.1098197	126	42.11941	53.71893
135	697592.9	314254.2	135	1.625241	0.74977	135	178606.2	705130.7	135	4.535476	3.0578632	135	43.52088	55.43304
144	676085.3	318872.9	144	1.764483	0.793986	144	158033.8	704138.7	144	4.46225	3.0037728	144	45.53799	57.02472
153	699463.1	330094.2	153	1.84681	0.784931	153	291754.7	706008.9	153	4.388746	2.9500907	153	47.41599	58.50662
162	715360	318872.9	162	1.909809	0.776715	162	259025.8	701333.3	162	4.314459	2.8969897	162	49.16879	59.88974
171	709749.3	317937.8	171	2.250382	0.989756	171	266506.7	713489.8	171	4.240973	2.8440486	171	50.80851	61.18362
180	674215.1	317002.7	180	2.451188	1.04397	180	266506.7	706944	180	4.169471	2.7920583	180	52.34575	62.39663

189	710684.4	317937.8	189	2.577032	1.034906	189	288949.3	704138.7	189	4.099874	2.7413078	189	53.78981	63.53612
198	683566.2	308586.7	198	2.596587	1.029311	198	271182.2	701333.3	198	4.03126	2.6928928	198	55.14894	64.60859
207	663928.9	309521.8	207	2.653188	1.028989	207	286144	710684.4	207	3.964232	2.6472763	207	56.4304	65.61977
216	707879.1	302976	216	2.699103	1.026272	216	304846.2	702268.4	216	3.897854	2.6021388	216	57.64066	66.57478
225	708814.2	308586.7	225	2.750988	1.019793	225	241258.7	709749.3	225	3.83354	2.5583639	225	58.78551	67.47816
234	701333.3	308586.7	234	2.820022	1.024018	234	284273.8	705073.8	234	3.83354	2.5159236	234	59.8701	68.334
243	716295.1	308586.7	243	2.885171	1.018996	243	284273.8	706944	243	3.83354	2.474582	243	60.89907	69.14595
252	710684.4	320743.1	252	2.927393	1.011204	252	296430.2	706944	252	3.83354	2.4345275	252	61.8766	69.9173
261	668604.4	330094.2	261	2.969848	1.007161	261	254350.2	704138.7	261	3.83354	2.3956849	261	62.80644	70.65103
270	703203.6	282403.6	270	3.013867	1.007601	270	285208.9	709749.3	270	3.83354	2.3581088	270	63.692	71.34981
279	688241.8	296430.2	279	3.054424	1.004397	279	282403.6	704138.7	279	3.83354	2.3216782	279	64.53637	72.0161
288	703203.6	319808	288	3.098182	1.002784	288	279598.2	706944	288	3.83354	2.2864717	288	65.34236	72.65209
297	681696	306716.4	297	3.117729	0.999073	297	300170.7	704138.7	297	3.83354	2.2522777	297	66.11253	73.25982
306	701333.3	288014.2	306	3.13494	0.992264	306	246869.3	700398.2	306	3.83354	2.2190634	306	66.84921	73.84113
315	682631.1	293624.9	315	3.146683	0.987462	315	281468.4	706008.9	315	3.83354	2.1868796	315	67.55455	74.3977
324	679825.8	315132.4	324	3.240171	0.982782	324	280533.3	709749.3	324	3.83354	2.1556767	324	68.2305	74.93109
333	698528	299235.6	333	3.279387	0.980893	333	261831.1	706008.9	333	3.83354	2.1253643	333	68.87885	75.4427
342	660131.6	303911.1	342	3.317204	1.026689	342	267441.8	700398.2	342	3.83354	2.1253643	342	69.50128	75.93384
351	563758.2	291754.7	351	3.326274	1.103241	351	260323.6	703203.6	351	3.83354	2.1253643	351	70.09929	76.40573
360	701333.3	297365.3	360	3.353542	1.106486	360	276792.9	694787.6	360	3.83354	2.1253643	360	70.6743	76.85946
369	658318.2	329159.1	369	3.382247	1.105976	369	278663.1	713489.8	369	3.693187	2.1253643	369	71.22762	77.29608
378	683566.2	302976	378	3.420473	1.108278	378	282403.6	707879.1	378	3.562723	2.1253643	378	71.76044	77.71652
387	664654.2	283338.7	387	3.438379	1.162818	387	276792.9	664864	387	3.44109	2.1253643	387	72.27389	78.12167
396	575914.7	300170.7	396	3.399368	1.163188	396	274216.9	637632	396	3.327465	2.1253643	396	72.769	78.51236
405	517937.8	302976	405	3.342735	1.145236	405	270915.6	606277.3	405	3.221083	2.1253643	405	73.24673	77.43224
414	535704.9	284273.8	414	3.287949	1.127934	414	269980.4	566563.6	414	3.121296	2.1253643	414	73.708	76.09721
423	537575.1	331029.3	423	3.234777	1.361627	423	262423.1	588071.1	423	3.027499	2.1253643	423	74.15362	74.80742
432	547861.3	308586.7	432	3.183296	1.340394	432	239331.6	574979.6	432	2.939175	2.1253643	432	74.5844	73.56063
441	545991.1	309521.8	441	3.133538	1.319806	441	257098.7	577784.9	441	2.855851	2.1253643	441	75.00105	72.35472

450	541315.6	319808	450	3.085289	1.299838	450	240266.7	582460.4	450	2.777123	2.1253643	450	75.40426	71.18771
459	547861.3	307651.6	459	3.038624	1.280575	459	224369.8	598357.3	459	2.702622	2.1253643	459	75.79466	70.05775
468	517002.7	307651.6	468	2.993541	1.261764	468	221201.8	598357.3	468	2.632015	2.1253643	468	76.17287	68.96309
477	530094.2	303911.1	477	2.949599	1.24344	477	229045.3	585265.8	477	2.565006	2.1253643	477	76.53944	67.90212
486	542250.7	307651.6	486	2.907087	1.23342	486	233720.9	596487.1	486	2.501327	2.0982807	486	76.89491	66.8733
495	506716.4	474922.7	495	2.865614	1.261686	495	229331.6	578720	495	2.440738	2.0721902	495	77.23976	65.87519
504	558147.6	459521.8	504	2.825314	1.244317	504	226240	581525.3	504	2.38302	2.0462871	504	77.57447	64.90644
513	546926.2	427365.3	513	2.786313	1.227557	513	205304.9	594616.9	513	2.327972	2.0206066	513	77.89948	63.96577
522	531029.3	400937.8	522	2.748406	1.211123	522	198682.7	585265.8	522	2.27541	1.9952876	522	78.2152	63.05197
531	554407.1	350197.3	531	2.711364	1.195225	531	136526.2	580590.2	531	2.225175	1.9703177	531	78.52203	62.16391
540	552536.9	313262.2	540	2.675568	1.179874	540	120629.3	587136	540	2.177112	1.9457102	540	78.82033	61.30053
549	569368.9	303911.1	549	2.640734	2.563969	549	131850.7	586200.9	549	2.131084	1.9215383	549	79.11046	60.46079
558	564693.3	288014.2	558	2.606773	2.043881	558	136526.2	584330.7	558	2.086963	2.3978209	558	79.39275	59.64376
567	545056	316067.6	567	2.573704	1.924451	567	147747.6	590876.4	567	2.044635	2.0745596	567	79.66752	58.84851
576	569368.9	307651.6	576	2.947288	1.605405	576	128110.2	591811.6	576	2.003993	1.9517292	576	79.93505	58.07418
585	565628.4	305781.3	585	2.910633	1.486884	585	124369.8	594616.9	585	1.964935	1.9019153	585	80.19563	57.31997
594	560952.9	330094.2	594	2.874881	1.505136	594	140266.7	596487.1	594	1.927375	1.6492106	594	80.44954	56.5851
603	558147.6	332899.6	603	2.839972	1.681905	603	137461.3	579655.1	603	1.891226	1.6204202	603	80.69701	55.86883
612	553472	301105.8	612	2.80603	1.659433	612	147747.6	576849.8	612	1.85641	1.5957806	612	80.9383	55.17047
621	544120.9	324483.6	621	2.772914	1.63742	621	148682.7	582460.4	621	1.822854	1.5855163	621	81.17363	54.48936
630	560017.8	329159.1	630	2.740572	1.615854	630	132785.8	591811.6	630	1.790491	1.5696406	630	81.40322	53.82485
639	560952.9	298300.4	639	2.709046	1.647972	639	125304.9	598357.3	639	1.759259	1.5594021	639	81.62728	53.17636
648	553472	314197.3	648	2.678246	1.644417	648	158968.9	583395.6	648	1.729098	1.5454739	648	81.846	52.54331
657	557212.4	297365.3	657	2.648178	1.654391	657	137461.3	584330.7	657	1.699957	1.5383059	657	82.05957	51.92515
666	551601.8	302040.9	666	2.618794	1.634741	666	147747.6	589941.3	666	1.671782	1.5282932	666	82.26818	51.32137
675	547861.3	314197.3	675	2.590034	1.915568	675	137461.3	588071.1	675	1.644527	1.5157592	675	82.472	50.73147
684	554471.1	308586.7	684	2.561967	1.696964	684	138588.4	599356.4	684	1.618149	1.5088269	684	82.67118	50.15498
693	534769.8	274922.7	693	2.534589	1.67911	693	122499.6	590876.4	693	1.592605	1.4954676	693	82.86589	49.59144
702	530094.2	281468.4	702	2.508064	1.661743	702	144007.1	606773.3	702	1.567856	1.4856428	702	83.05626	49.04042

711	570304	289884.4	711	2.48201	1.644658	711	134656	622670.2	711	1.543866	1.4758332	711	83.24246	48.50152
720	556277.3	272117.3	720	2.45647	1.62813	720	140266.7	613319.1	720	1.520601	1.4648333	720	83.42461	47.97433
729	546926.2	287079.1	729	2.431466	1.611775	729	129980.4	617994.7	729	1.498027	1.4526924	729	83.60284	47.45847
738	545056	280533.3	738	2.407049	1.595761	738	138396.4	512384	738	1.498027	1.4478803	738	83.77727	46.9536
747	551601.8	279598.2	747	2.383178	1.580236	747	133720.9	600227.6	747	1.498027	1.4323884	747	83.94804	46.45935
756	536640	266506.7	756	2.359856	1.564883	756	159904	602097.8	756	1.498027	1.4240203	756	84.11525	45.9754
765	545991.1	283338.7	765	2.341438	1.549846	765	143072	589941.3	765	1.498027	1.4134367	765	84.27901	45.50142
774	527288.9	287079.1	774	2.361329	1.535064	774	153358.2	613319.1	774	1.498027	1.4005608	774	84.43943	45.03712
783	556277.3	275857.8	783	2.338809	1.520855	783	131850.7	583395.6	783	1.520594	1.3816025	783	84.5966	44.5822
792	#N/A	#N/A	792	#N/A	#N/A	792	#N/A	#N/A	792	1.54246	1.3618875	792	84.75064	44.13638

Table of Data: Project 2 (70 mobile devices)

Intelligent Routing Algorithm for Mobile IPTV

Scenario 3 and 4

Load				End-to-End Delay			Throughput			Packet Delay variation (Jitter)		
Time (sec)	Scenario 3	Scenario 4		Time (sec)	Scenario 3	Scenario 4	Time (sec)	Scenario 3	Scenario 4	Time (sec)	Scenario 3	Scenario 4
0	93632	93632		0	0.0003534	0.0003534	0	56.888889	56.888889	0	0	0
9	315253.33	315253.33		9	0.0408382	0.0408382	9	56.888889	56.888889	9	0.7090123	0.7090123
18	398414.22	398414.22		18	0.0555426	0.0555426	18	56.888889	56.888889	18	1.7070003	1.7070003
27	617230.22	617230.22		27	0.1436998	0.1436998	27	241507.56	241507.56	27	2.6615334	2.6615334
36	605952	605952		36	0.0076935	0.0076935	36	657440	657440	36	1.5599634	1.5599634
45	593852.44	593852.44		45	0.0081533	0.0081533	45	490055.11	490055.11	45	1.5101041	1.5101041
54	629386.67	629386.67		54	0.2156482	0.2156482	54	475036.44	475036.44	54	3.8604181	3.8604181
63	644348.44	644348.44		63	0.1846266	0.1846266	63	554577.78	554577.78	63	4.2923131	4.2923131
72	616295.11	616295.11		72	0.1051316	0.1051316	72	588241.78	588241.78	72	3.6613585	3.6613585
81	620035.56	620035.56		81	0.0815778	0.0815778	81	575150.22	575150.22	81	3.831475	3.831475
90	541429.33	541429.33		90	0.4468224	0.4468224	90	549902.22	549902.22	90	4.8431655	4.8431655
99	589176.89	589176.89		99	0.3044444	0.3044444	99	565799.11	565799.11	99	4.5436521	4.5436521
108	608814.22	608814.22		108	0.1330015	0.1330015	108	560188.44	560188.44	108	4.4521227	4.4521227
117	627516.44	627516.44		117	0.2031523	0.2031523	117	543356.44	543356.44	117	4.9558541	4.9558541
126	572344.89	572344.89		126	0.3604648	0.3604648	126	546104.89	546104.89	126	6.5211174	6.5211174
135	584501.33	584501.33		135	0.2737026	0.2737026	135	464807.11	464807.11	135	6.2498419	6.2498419
144	637745.78	637745.78		144	0.1906606	0.1906606	144	576085.33	576085.33	144	5.4863789	5.4863789
153	624654.22	624654.22		153	0.5638024	0.5638024	153	572344.89	572344.89	153	9.4644776	5.5989388
162	582574.22	582574.22		162	0.0893483	0.0893483	162	562058.67	562058.67	162	5.8455791	5.6612756
171	589120	589120		171	0.031721	0.031721	171	549845.33	549845.33	171	5.5297347	5.4612756
180	559196.44	559196.44		180	0.0716694	0.0716694	180	569482.67	569482.67	180	5.7829696	7.725
189	571352.89	571352.89		189	0.0289704	0.0289704	189	540494.22	540494.22	189	5.6652439	5.8555319

198	501219.56	501219.56		198	0.0304151	0.0304151	198	440437.33	440437.33	198	5.6414634	5.8626793
207	518986.67	518986.67		207	0.0229452	0.0229452	207	552650.67	552650.67	207	5.628432	5.8369963
216	575093.33	575093.33		216	0.0302461	0.0302461	216	613432.89	613432.89	216	5.6067073	5.8663819
225	593795.56	593795.56		225	0.0412197	0.0412197	225	572288	572288	225	5.830264	5.9990786
234	562001.78	562001.78		234	0.2146186	0.2146186	234	590990.22	590990.22	234	6.4872682	6.0025078
243	605952	605952		243	0.0399952	0.0399952	243	551715.56	551715.56	243	5.8389262	6.0025078
252	659253.33	659253.33		252	0.0617435	0.0617435	252	549845.33	549845.33	252	5.8463883	6.0025078
261	598471.11	598471.11		261	0.0383373	0.0383373	261	552650.67	552650.67	261	5.8645929	6.0025078
270	621848.89	621848.89		270	0.0413204	0.0413204	270	574158.22	574158.22	270	5.8446069	6.0025078
279	605952	605952		279	0.0336741	0.0336741	279	562936.89	562936.89	279	5.8591463	6.0025078
288	582574.22	582574.22		288	0.0359321	0.0359321	288	569482.67	569482.67	288	5.8353156	6.0025078
297	610627.56	610627.56		297	0.0319295	0.0319295	297	557326.22	557326.22	297	5.8537477	6.0025078
306	597536	597536		306	0.0359594	0.0359594	306	546104.89	546104.89	306	5.8335366	6.0025078
315	538624	538624		315	0.0251977	0.0251977	315	532078.22	532078.22	315	5.8646341	5.9948218
324	543299.56	543299.56		324	0.025079	0.025079	324	572288	572288	324	6.5674001	5.9948218
333	597536	597536		333	0.0390595	0.0390595	333	585379.56	585379.56	333	5.6985384	5.9948218
342	576028.44	576028.44		342	0.0175461	0.0280425	342	564807.11	564807.11	342	5.6653459	5.9948218
351	437632	437632		351	2.1447399	0.0219236	351	563872	563872	351	5.7130673	5.9948218
360	525532.44	525532.44		360	0.0160627	1.0981594	360	569482.67	569482.67	360	5.6359756	5.9948218
369	509635.56	509635.56		369	0.016846	0.0219941	369	577898.67	577898.67	369	5.5338208	5.8336887
378	590990.22	590990.22		378	0.0169649	0.0293857	378	568547.56	568547.56	378	5.5865854	5.7640244
387	524597.33	443242.67		387	0.0161719	0.0231066	387	593795.56	593795.56	387	5.5350824	5.8096981
396	372174.22	336640		396	0.0192041	0.0188491	396	562001.78	562001.78	396	5.5682927	5.7931666
405	162709.33	213205.33		405	0.0086836	0.0100225	405	526467.56	526467.56	405	5.5195008	5.8438313
414	144007.11	181411.56		414	0.0055459	0.0038751	414	613432.89	613432.89	414	5.5377129	5.8135283
423	232842.67	184216.89		423	0.0056618	0.0048077	423	359082.67	359082.67	423	5.2170732	5.8279439
432	296430.22	199178.67		432	0.0034576	0.0057338	432	289884.44	289884.44	432	5.1439024	5.7914634
441	293624.89	197308.44		441	0.0032508	0.0060779	441	274922.67	303911.11	441	5.1073171	5.8244973
450	319808	216945.78		450	0.0031842	0.0059443	450	275857.78	306716.44	450	5.0822669	5.8280488

459	303911.11	206659.56		459	0.003266	0.0052849	459	288014.22	310456.89	459	5.0341463	5.8170732
468	319808	238453.33		468	0.0038595	0.0066486	468	293624.89	292689.78	468	5.0378049	5.8280488
477	314197.33	221621.33		477	0.003575	0.0062168	477	277728	325418.67	477	5.0707317	5.8243902
486	306716.44	215075.56		486	0.2225702	0.0053843	486	289884.44	311392	486	7.0322785	5.8427788
495	301105.78	215075.56		495	0.102713	0.0051731	495	275857.78	307651.56	495	5.8192331	5.8463415
504	292689.78	214140.44		504	0.0041304	0.0057572	504	283338.67	300170.67	504	5.3963415	5.8206223
513	300170.67	216945.78		513	0.0042172	0.0054337	513	278663.11	321678.22	513	5.3967093	5.8060976
522	317002.67	234712.89		522	0.0040779	0.0054841	522	286144	311392	522	5.4179378	5.8604509
531	306716.44	216945.78		531	0.0031871	0.0063845	531	280533.33	315132.44	531	5.4076782	5.8244973
540	306716.44	216010.67		540	0.0042028	0.0059491	540	289884.44	329159.11	540	5.4512195	5.8097561
549	308586.67	210400		549	0.0034001	0.0054673	549	272117.33	314197.33	549	5.3963415	5.802439
558	289884.44	199178.67		558	0.0033027	0.0058679	558	276792.89	317937.78	558	5.3890244	5.8354662
567	310456.89	209464.89		567	0.0039741	0.0065566	567	289884.44	328224	567	5.4478976	5.8390244
576	299235.56	212270.22		576	0.00381	0.0059989	576	285208.89	329159.11	576	5.4655278	5.7695122
585	305781.33	223491.56		585	0.0038067	0.0068552	585	283338.67	325418.67	585	5.4259598	5.8683729
594	314197.33	212270.22		594	0.0031862	0.0061645	594	279598.22	341315.56	594	5.454878	5.85
603	293624.89	227232		603	0.003379	0.0058089	603	271182.22	335704.89	603	5.4182927	5.8134146
612	271182.22	199178.67		612	0.0035358	0.0050281	612	268376.89	323548.44	612	5.4043849	5.7841463
621	266506.67	201984		621	0.0041277	0.0057525	621	273052.44	337575.11	621	5.4691885	5.8316046
630	300170.67	219751.11		630	0.0034606	0.0063973	630	271182.22	351601.78	630	5.4442413	5.809872
639	308586.67	230037.33		639	0.003157	0.0062449	639	292689.78	346926.22	639	5.4545455	5.802439
648	296430.22	203854.22		648	0.0042896	0.0065275	648	273052.44	315132.44	648	5.4552102	5.820841
657	282403.56	204789.33		657	0.0041769	0.0051431	657	271182.22	338510.22	657	5.4768293	5.7878049
666	294560	210400		666	0.0039084	0.0053372	666	286144	344120.89	666	5.4439024	5.8463415
675	307651.56	216945.78		675	0.0037877	0.0057916	675	300170.67	321678.22	675	5.4036585	5.8280488
684	302040.89	215075.56		684	0.003448	0.0064018	684	278663.11	327288.89	684	5.4296161	5.8170732
693	297365.33	203854.22		693	0.0044576	0.0055886	693	287079.11	332899.56	693	5.3437688	5.7635379
702	334769.78	207594.67		702	0.0038082	0.0070416	702	279598.22	324483.56	702	5.454878	5.8390244
711	341315.56	213205.33		711	0.0039837	0.0067376	711	274922.67	332963.56	711	5.4098361	5.7997573

720	306716.44	217880.89		720	0.0039841	0.0058279	720	306716.44	303911.11	720	5.408039	5.8280488
729	338510.22	219751.11		729	0.0039407	0.1069555	729	309521.78	328224	729	5.461867	6.2375712
738	331029.33	222556.44		738	0.0073384	0.1990954	738	301105.78	292689.78	738	5.6361401	6.4335463
747	352536.89	208529.78		747	0.0835005	0.0077386	747	287079.11	280533.33	747	6.5962733	6.4335463
756	319808	185152		756	0.0477507	0.0064485	756	289884.44	313262.22	756	6.544	6.4335463
765	317002.67	199178.67		765	0.156359	0.0056938	765	301105.78	303911.11	765	7.5911083	5.9890378
774	354407.11	201984		774	0.004268	0.0067801	774	315132.44	302040.89	774	6	5.9890378
783	354407.11	222556.44		783	0.0038198	0.0078963	783	324483.56	310456.89	783	6	5.9890378
792	295495.11	201048.89		792	0.0043766	0.0056507	792	307651.56	311392	792	6	5.8097561
801	#N/A	#N/A		801	#N/A	#N/A	801	294560	301105.78	801	#N/A	#N/A

Appendix B: List of papers published during this study

1. B. A. Abubakar, M. Petridis, D. S. Gill, and S. M. Gheytassi, "Unicast Bandwidth Efficiency Routing Algorithm for Mobile Devices," in *2014 IEEE International Conference on Mobile Services*, 2014, vol. 2, no. 1, pp. 8–15.

2. B. A. Abubakar, M. Petridis, D. S. Gill, and S. M. Gheytassi, "Effective Resource Utilization Routing Algorithm for IPTV," in *2nd World Symposium on Web Applications and Networking (WSWAN)*, 2015, pp. 1–7.

3. B. A. Abubakar, M. Petridis, D. S. Gill, and S. M. Gheytassi, "A Novel Routing Algorithm For Video-On-Demand On Mobile Devices," *International Journal of Services Computing (ISSN 2330-4472).* Accepted, in press

4. B. A. Abubakar, M. Petridis, D. S. Gill, and S. M. Gheytassi, "Adaptive CDN-Based Bandwidth Conserving Algorithm for Mobile IPTV," *The Eighth International Conference on Advanced Computational Intelligence*, 2016. Accepted, in press

5. B. A. Abubakar, M. Petridis, D. S. Gill, and S. M. Gheytassi, "Intelligent Routing Algorithm for Mobile IPTV," *The Sixth International Conference on Communication Systems and Network Technologies*, 2016. Accepted, in press

References

[1] S. Park and S. Jeong, "Mobile IPTV: Approaches, challenges, standards, and QoS support," *Internet Comput. IEEE*, no. June, 2009.

[2] F. E. Retnasothie, M. K. Ozdemir, T. Yucek, H. Celebi, J. Zhang, and R. Muththaiah, "Wireless IPTV over WiMAX: challenges and applications," *WAMICON 2006*, 2006.

[3] M. ŠKrbic, N. ŠEcic, and M. Varatanovic, "A Unicast-Based IPTV Service Control," *2010 Fifth Int. Conf. Syst. Networks Commun.*, pp. 278–282, Aug. 2010.

[4] H. Tran, a. Mellouk, J. Perez, S. Hoceini, and S. Zeadally, "QoE-based Server Selection for Content Distribution Networks," *IEEE Trans. Comput.*, vol. Early Acce, no. 11, pp. 2803–2815, 2013.

[5] C. Hellberg, T. Boyes, and D. Greene, *Broadband network architectures: designing and deploying triple-play services*. Prentice Hall, 2007.

[6] inCode Telecom group Inc., "The ' Quad Play ' — T he First Wave of the Converged Services Evolution," no. 3, pp. 1–10, 2006.

[7] M. Garcia, J. Lloret, M. Edo, and R. Lacuesta, "IPTV Distribution Network Access System Using WiMAX and WLAN Technologies," pp. 35–44, 2009.

[8] ITU-Y (2008/09), "IPTV functional architecture," *Geneva, ITU-T (August 2008)*, 1910.

[9] C. Hu, Y. Liu, L. Wang, and M. Song, "Power Allocation in Cellular Systems with Multicast and Unicast Hybrid Service," *2012 8th Int. Conf. Wirel. Commun. Netw. Mob. Comput.*, pp. 1–4, Sep. 2012.

[10] Cisco System Inc, "IP Multicast Technology Overview," 2002. [Online]. Available: http://www.cisco.com/en/US/tech/tk828/tech_white_papers_list.html.

[11] L. Harte, *Introduction to Data Multicasting, IP Multicast Streaming for Audio and Video Media Distribution*. Althos Publishing, 2008.

[12] S. Venaas, “IPV4 Multicast Address Space Registry,” *Internet Assigned Numbers Authority*, 2013. [Online]. Available: http://www.iana.org/assignments/multicast-addresses/multicast-addresses.xhtml. [Accessed: 21-May-2014].

[13] B. Williamson, *Developing IP Multicast Networks*, Volume 1. Indianapolis, USA: Cisco Press, 2002.

[14] R. Jain, “Media Vision Quality of Experience,” *Ieee Comput. Soc.*, pp. 95–96, 2004.

[15] D. Ferrara and L. Galluccio, “MACRO: an integrated MAC/routing protocol for geographic forwarding in wireless sensor networks,” *... 2005. 24th Annu. ...*, 2005.

[16] J. Lee, E. Lee, S. Park, H. Park, and S.-H. Kim, “Sink-Initiated Geographic Multicasting Protocol in Wireless Sensor Networks,” *2010 24th IEEE Int. Conf. Adv. Inf. Netw. Appl.*, pp. 910–916, 2010.

[17] K. Tian, B. Zhang, H. Mouftah, Z. Zhao, and J. Ma, “Destination-Driven On-Demand Multicast Routing Protocol for Wireless Ad Hoc Networks,” *2009 IEEE Int. Conf. Commun.*, pp. 1–5, Jun. 2009.

[18] A. Mizumoto, “Cost-conscious geographic multicast on manet,” *Sens. Ad Hoc ...*, vol. 00, no. c, 2004.

[19] L. Galluccio, G. Morabito, and S. Palazzo, “GEographic multicast (GEM) for dense wireless networks: protocol design and performance analysis,” *... /ACM Trans. Netw. (...*, vol. 21, no. 4, pp. 1332–1346, 2013.

[20] X. Xiang, Z. Zhou, and X. Wang, “Robust and scalable geographic multicast protocol for mobile ad-hoc networks,” *INFOCOM 2007. 26th IEEE ...*, pp. 2301–2305, 2007.

[21] K. Chen and K. Nahrstedt, "Effective location-guided tree construction algorithms for small group multicast in MANET," ... . *Twenty-First Annu. Jt. Conf. ...*, vol. 00, no. 1, pp. 1180–1189, 2002.

[22] T. Lang, "MobiHoc Poster : Position-Based Multicast Routing for Mobile Ad-Hoc Networks," vol. 7, no. 3, pp. 53–55, 2003.

[23] D. Silvestre and T. Vazao, "A Geographic Unicast Routing Algorithm using no Location Service," *9th IEEE Int. Symp. Netw. Comput. Appl. (NCA), 2010*, pp. 280–285, Jul. 2010.

[24] C. Augustine, "Adaptive Hybrid Transmission mechanism for on-demand mobile IPTV over WMN," *2012 Int. Conf. Adv. Eng. Sci. Manag.*, pp. 557–564, 2012.

[25] A. Canovas and D. Bri, "Vertical WLAN handover algorithm and protocol to improve the IPTV QoS of the end user," *... (ICC), 2012 IEEE ...*, pp. 1901–1905, 2012.

[26] S. Ko, S. Oh, and K. Chung, "An efficient QoS negotiation mechanism for IPTV service in heterogeneous networks," *Comput. Sci. Inf. ...*, pp. 575–579, 2009.

[27] V. T. Srikanth, "Simulation-based approach to performance study of routing protocols in MANETs and ad-hoc Networks," *paper.ijcsns.org*, vol. 11, no. 9, pp. 111–115, 2011.

[28] Y. Qin, "A look-ahead unicast routing algorithm in MANETs," *Veh. Technol. Conf. 2005. VTC 2005*, vol. 00, no. c, 2005.

[29] A. Acharya, A. Misra, and S. Bansal, "A label-switching packet forwarding architecture for multi-hop wireless LANs," ... *ACM Int. Work. Wirel. Commun.*, pp. 33–40, 2002.

[30] W. Jia and L. Wang, "UNCLE: A unified unicast and multicast label forwarding architecture in MANETs," *2012 IEEE Glob. Commun. Conf.*, pp. 5711–5716, 2012.

[31] P. Kuppusamy, "A study and comparison of OLSR, AODV and TORA routing protocols in ad hoc networks," in *3rd International Conference on Electronics Computer Technology (ICECT), 2011*,

2011, pp. 143–147.

[32] C. Perkins and P. Bhagwat, “Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers,” *ACM SIGCOMM Comput. Commun. …*, pp. 234–244, 1994.

[33] P. Jacquet, P. Muhlethaler, T. Clausen, a. Laouiti, a. Qayyum, and L. Viennot, “Optimized link state routing protocol for ad hoc networks,” *Proceedings. IEEE Int. Multi Top. Conf. 2001. IEEE INMIC 2001. Technol. 21st Century.*, pp. 62–68.

[34] D. Johnson and D. Maltz, “Dynamic source routing in ad hoc wireless networks,” *Mob. Comput.*, 1996.

[35] C. Perkins and E. Royer, “Ad-hoc on-demand distance vector routing,” *WMCSA’99. Second IEEE Work. Mob. Comput. Syst. Appl.*, 1999.

[36] T. Yélémou, “Improving ZRP performance by taking into account quality of links,” in *2012 IEEE Wireless Communications and Networking Conference (WCNC 2012)*, 2012, pp. 2956–2960.

[37] K. Lim and A. Datta, “An In-depth Analysis of the Effects of IMEP on TORA Protocol,” *Wirel. Commun. Netw. …*, pp. 3051–3056, 2012.

[38] G. Rajkumar and R. Parthiban, D.Kasiram, “Optimized QoS Metrics and Performance Comparison ofDSR and AODV Routing Protocols,” in *IEEE-International Conference On Advances In Engineering, Science And Management (ICAESM -2012)*, 2012, pp. 251–255.

[39] A. Kanthe, D. Simunic, and R. Prasad, “Comparison of AODV and DSR on-demand routing protocols in mobile ad hoc networks,” *… Technol. Trends …*, pp. 1–5, Dec. 2012.

[40] N. Sharma, S. Rana, and R. Sharma, “Provisioning of Quality of Service in MANETs performance analysis & comparison (AODV and DSR),” *Comput. Eng. …*, vol. 1, pp. 243–248, 2010.

[41] L. Leemis and S. Park, *Discrete-event simulation: A first course*, no. December. 2006.

[42] Z. Lu and H. Yang, *Unlocking the Power of OPNET Modeler*. Cambridge: Cambridge University Press, 2012.

[43] L. Kleinrock and R. Finkelstein, “Time dependent priority queues,” *Oper. Res.*, vol. 15, no. 1, pp. 104–116, 1967.

[44] OPNET Technologies, “OPNET Modeler 17.5-PL6 Documentation,” 2014. [Online]. Available: https://support.riverbed.com/bin/support/static//doc/opnet/OnlineHTML/Modeler/modeler_17.5_PL6/wwhelp/wwhimpl/js/html/wwhelp.htm. [Accessed: 10-Mar-2014].

[45] M. J. De Smith, *Statistical Analysis Handbook*. 2015.

[46] M. J. De Smith, *Statistical Analysis Handbook*. 2015.

[47] L. Bello, K. a. Anang, P. Bakalis, P. B. Rpajic, and T. I. Eneh, “Sensitivity of DSR Protocol Performance to Propagation Loss Models at Higher Microwave Frequencies,” *2012 UKSim 14th Int. Conf. Comput. Model. Simul.*, pp. 561–565, 2012.

[48] S. Yang, R. He, Y. Wang, S. Li, and B. Lin, “OPNET-based Modeling and Simulations on Routing Protocols in VANETs with IEEE 802 . 11p,” 2014, no. Icsai, pp. 536–541.

[49] K. Kucuk, N. Bandirmali, and A. Kavakl, “Modeling of the Modified Ssle in Opnet for Large Scale Wireless Sensor,” 2012, no. October, pp. 12–14.

[50] M. Fazeli and H. Vaziri, “Assessment of Throughput Performance Under OPNET Modeler Simulation Tools in Mobile Ad Hoc Networks (MANETs),” *2011 Third Int. Conf. Comput. Intell. Commun. Syst. Networks*, pp. 328–331, 2011.

[51] J. Dorleus, R. Holweck, Z. Ren, H. Li, H. L. Cui, and J. Medina, “Modeling and simulation of fading and pathloss in OPNET for range communications,” *Proc. - 2007 IEEE Radio Wirel. Symp. RWS*, pp. 407–410, 2007.

[52] OPNET Technologies, “OPNET Modeler,” 2014. [Online]. Available: www.opnet.com.

[53] BBC, “Live Winter Olympics,” 2014. [Online]. Available:

http://www.bbc.co.uk/sport/winter-olympics/2014.

[54] B. A. Abubakar, M. Petridis, D. S. Gill, and S. M. Gheytassi, "Unicast Bandwidth Efficiency Routing Algorithm for Mobile Devices," in *2014 IEEE International Conference on Mobile Services*, 2014, vol. 2, no. 1, pp. 8–15.

[55] B. A. Abubakar, M. Petridis, D. S. Gill, and S. M. Gheytassi, "Effective Resource Utilization Routing Algorithm for IPTV," in *2nd World Symposium on Web Applications and Networking (WSWAN)*, 2015, pp. 1–7.

[56] H.-S. Jang, J.-G. Kim, and K.-S. Cho, "A Dynamic Routing Algorithm in Multicast Communication Networks," *2008 International Conference on Convergence and Hybrid Information Technology*. 2008.

[57] H. Wang, J. He, T. Li, S. Zhang, and Z. Sun, "Heuristic gradient based multicast routing policy for dynamic network," *2011 Int. Conf. Multimed. Technol. ICMT 2011*, pp. 4936–4939, 2011.

[58] B. A. Abubakar, D. S. Gill, M. Petridis, and S. M. Gheytassi, "Effective Resource Utilization Routing Algorithm for IPTV," in *2015 2nd World Symposium on Web Applications and Networking (WSWAN)*, 2015, pp. 1–7.

[59] a. Cabellos-Aparicio, F. J. Garcia, and J. Domingo-Pascual, "A Novel Available Bandwidth Estimation and Tracking Algorithm," *NOMS Work. 2008 - IEEE Netw. Oper. Manag. Symp. Work.*, pp. 87–94, 2008.

[60] B. Meskill, A. Davy, and B. Jennings, "Server selection and admission control for IP-based video on demand using available bandwidth estimation," *2011 IEEE 36th Conf. Local Comput. Networks*, pp. 255–258, 2011.

[61] V. Ribeiro, M. Coates, R. Riedi, S. Sarvotham, B. Hendricks, and R. Baraniuk, "Multifractal cross-traffic estimation," *ITC Spec. Semin. IP Traffic Meas.*, pp. 1–15, 2000.

[62] B. Melander, M. Bjorkman, and P. Gunningberg, "A new end-to-end probing and analysis method for estimatingbandwidth bottlenecks," *Globecom '00 - IEEE. Glob. Telecommun. Conf. Conf. Rec. (Cat. No.00CH37137)*, vol. 1, no. 1, pp. 415–420, 2000.

[63] M. Jain and C. Dovrolis, "Pathload: a Measurement Tool for Available Bandwidth Estimation," *Proc. PAM2003*, 2002.

[64] N. Hu and P. Steenkiste, "Evaluation and characterization of available bandwidth probing techniques," *IEEE J. Sel. Areas Commun.*, vol. 21, no. 6, pp. 879–894, 2003.

[65] V. J. Ribeiro, R. H. Riedi, R. G. Baraniuk, J. Navratil, and L. Cottrell, "pathChirp : Efficient Available Bandwidth Estimation for Network Paths," *4th Int. Work. Passiv. Act. Netw. Meas. PAM 2003*, pp. 1–11, 2003.

[66] S. Ekelin, M. Nilsson, E. Hartikainen, a. Johnsson, J.-E. Mangs, B. Melander, and M. Bjorkman, "Real-Time Measurement of End-to-End Available Bandwidth using Kalman Filtering," *2006 IEEE/IFIP Netw. Oper. Manag. Symp. NOMS 2006*, 2006.

[67] J. Strauss, D. Katabi, and F. Kaashoek, "A measurement study of available bandwidth estimation tools," *Proc. Conf. Internet Meas. Conf. - IMC '03*, pp. 39 – 44, 2003.

[68] C. P. Low, H. Yu, J. M. Ng, Q. Lin, and Y. Atif, "An Efficient Algorithm for the Video Server Selection Problem," *Electron. Eng.*, pp. 1329–1333.

[69] H. Nam, K. Kimt, D. Calin, and H. Schulzrinnet, "Towards Dynamic Network Condition-Aware Video Server Selection Algorithms over Wireless Networks," in *2014 IEEE Symposium on Computers and Communication (ISCC)*, 2014.

[70] H. Huang, P. Xia, S. H. G. Chan, G. Shi, and H. Zhang, "Joint optimization of content replication and server selection for video-on-demand," *IEEE Int. Conf. Commun.*, pp. 2065–2069, 2012.

[71] H. Abdi, A. Isazadeh, and L. M. Khanli, "A Novel Context-Aware Server Selection Algorithm For Mobile Thin Client Computing," pp. 383–388, 2013.

[72] H. Chang, H. Liu, Y. Leung, and X. Chu, “Minimum Latency Server Selection for Heterogeneous Cloud Services,” in *Globecom 2014 - Symposium on Selected Areas in Communications: GC14 SAC Cloud Network*, 2014, pp. 2276–2282.

[73] “VLC Media Player.” [Online]. Available: http://www.videolan.org/vlc/download-windows.en_GB.html. [Accessed: 10-Feb-2014].

[74] Cisco System Inc, “OSPF Design Guide,” 7039, 2005.

[75] G. Michael T. and R. Tamassia, *Algorithm Design: Foundations, Analysis and Internet Examples*, 2nd Editio., vol. Paperback. John Wiley & Sons Ltd., 2007.

[76] S. Meng, L. Liu, and J. Yin, “Scalable and reliable IPTV service through collaborative request dispatching,” *ICWS 2010 - 2010 IEEE 8th Int. Conf. Web Serv.*, pp. 179–186, 2010.

[77] G. Kandavanam, D. Botvich, and S. Balasubramaniam, “An optimization based approach to maximizing QoS assurance for IPTV triple play services on the internet backbone,” *Proc. - Conf. Local Comput. Networks, LCN*, no. October, pp. 406–413, 2009.

[78] G. Kandavanam, R. Mallipeddi, D. Botvich, S. Balasubramaniam, and P. N. Suganthan, “Achieving high robustness and performance in QoS-aware route planning for IPTV networks,” *Inf. Sci. (Ny).*, vol. 269, pp. 217–237, 2014.

[79] F. Wang, Z. Wang, Y. Li, and L. Zeng, “Reliable multi-path routing with bandwidth and delay constraints,” *2010 Int. Conf. Multimed. Technol. ICMT 2010*, 2010.

[80] P. Rattanawadee, R. Natchaphon, and S. Chaiyachet, “The Transmission Time Analysis of IPTV Multicast Service in SDN / OpenFlow Environments,” in *12th International Conference on Information Technology (ECTI-CON)*, 2015, pp. 1–5.

[81] T. Brookes, “What All This MD5 Hash Stuff Actually Means [Technology Explained],” 2011. [Online]. Available: http://www.makeuseof.com/tag/md5-hash-stuff-means-technology-explained/. [Accessed: 12-Aug-2015].

[82] TechTarget, “MD5 definition.” [Online]. Available: http://searchsecurity.techtarget.com/definition/MD5. [Accessed: 11-Aug-2015].