Peer-To-Peer File Sharing in Android Devices

Submitted By

Rutul R. Shah

14MCEN20

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INSTITUTE OF TECHNOLOGY

NIRMA UNIVERSITY

AHMEDABAD-382481

May 2016

Peer-To-Peer File Sharing in Android Devices

Major Project

Submitted in partial fulfillment of the requirements

for the degree of

Master of Technology in Computer Science and Engineering(Networking Technologies)

Submitted By

Rutul R. Shah

14MCEN20

Guided By

Prof. Zunnun Narmawala

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INSTITUTE OF TECHNOLOGY

NIRMA UNIVERSITY

AHMEDABAD-382481

May 2016

Certificate

This is to certify that the major project entitled **"Peer-To-Peer File Sharing in Android Devices"** submitted by **Rutul R. Shah (Roll No: 14MCEN20)**, towards the partial fulfillment of the requirements for the award of degree of Master of Technology in Computer Science and Engineering(Networking Technologies) of Institute of Technology, Nirma University Ahmedabad, is the record of work carried out by him under my supervision and guidance. In my opinion, the submitted work has reached a level required for being accepted for examination. The results embodied in this project, to the best of my knowledge, haven't been submitted to any other university or institution for award of any degree or diploma.

Prof. Zunnun Narmawala
Guide & Associate Professor,
CSE Department,
Institute of Technology,
Nirma University, Ahmedabad.

Prof. Gaurang Raval
Associate Professor,
Coordinator M.Tech - CSE
Institute of Technology,
Nirma University, Ahmedabad

Dr. Sanjay Garg
Professor and Head,
CSE Department,
Institute of Technology,
Nirma University, Ahmedabad.

Dr P.N.Tekwani
Director,
Institute of Technology,
Nirma University, Ahmedabad

Statement of Originality

I, **Rutul R. Shah**, Roll. No. **14MCEN20**, give undertaking that the Major Project entitled "**Peer-To-Peer File Sharing In Android Devices**" submitted by me, towards the partial fulfillment of the requirements for the degree of Master of Technology in **Computer Science & Engineering(Networking Technologies)** of Institute of Technology, Nirma University, Ahmedabad, contains no material that has been awarded for any degree or diploma in any university or school in any territory to the best of my knowledge. It is the original work carried out by me and I give assurance that no attempt of plagiarism has been made. It contains no material that is previously published or written, except where reference has been made. I understand that in the event of any similarity found subsequently with any published work or any dissertation work elsewhere; it will result in severe disciplinary action.

_____

Signature of Student

Date:

Place:

Endorsed by

Prof.Zunnun Narmawala

(Signature of Guide)

Acknowledgements

It gives me immense pleasure in expressing thanks and profound gratitude to **Prof. Zunnun Narmawala**, Associate Professor, Computer Science Department, Institute of Technology, Nirma University, Ahmedabad for his valuable guidance and continual encouragement throughout this work. The appreciation and continual support he has imparted has been a great motivation to me in reaching a higher goal. His guidance has triggered and nourished my intellectual maturity that I will benefit from, for a long time to come.

It gives me an immense pleasure to thank **Dr. Sanjay Garg**, Hon'ble Head of Computer Science and Engineering Department, Institute of Technology, Nirma University, Ahmedabad for his kind support and providing basic infrastructure and healthy research environment.

A special thank you is expressed wholeheartedly to **Dr.P.N.Tekwani**, Hon'ble Director, Institute of Technology, Nirma University, Ahmedabad for the unmentionable motivation he has extended throughout course of this work.

I would also thank the Institution, all faculty members of Computer Engineering Department, Nirma University, Ahmedabad for their special attention and suggestions towards the project work.

See that you acknowledge each one who have helped you in the project directly or indirectly.

- Rutul R. Shah

14MCEN20

Abstract

Peer-to-peer file sharing is very popular on the Internet. But, to use it, one has to be connected to the Internet and it incurs significant data cost. We are developing an android application for peer-to-peer file sharing named Mobile Torrent using which files can be shared between smartphones within a campus without using the Internet. Each application user can share files and can register a request to download a file shared by any other application user even if the source is not directly connected. Initially, we have implemented the same for single hop; i.e. when two smartphones come in the Wi-Fi communication range of each other, they auto-discover each other using Wi-Fi Direct and automatically exchange required files.

Conventions

Typesetting

This thesis is typeset using Latex software.
Font used in this thesis are of Times new roman family.

Referencing

Referencing and citation style adopted in this thesis is ieee transaction(ieeetr).
For electronic references, Last publication date is shown here.

Spelling

The Unites States English Spelling is adopted here.

Units

The Units used in This thesis are based in the International System of Units(SI Units), unless specified.

Abbreviations

DTN	Delay Tolerant Network
ON	Opportunistic Network
P2PN	Peer To Peer Network
P2PFS	Peer To Peer File Sharing
M2MShare	Mobile-to- Mobile File Sharing
ORION	Optimized Routing Independent Overlay Network
MANET	Mobile Ad-Hoc Network
NS-2	Network Simulator-2
RLNC	Random Linear Network Coding
ONE	Opportunistic Network Environment simulator
SDK	Software Development Kit
TCP/IP	Transmission Control Protocol /Internet Protocol
BT	Bit Torrent
PC	Presence Collector
FT	Frequency Threshold
FIFO	First In First Out
TTL	Time To Live
URL	Uniform Resource Locator
URI	Uniform Resource Indetifier

Contents

List of Figures

Chapter 1

Introduction

File sharing provides the mechanism to share different kind of files within two devices. Peer to peer file sharing is a different approach for file sharing.This kind of file sharing became in existence with the cooperation of the users which have the file and those who want the file.Then Bit-torrent came in existence that now works on the no central server requirement but it requires the Internet to connect to the available peers.

Peer to peer file sharing in mobile applications are available now.In this application we are using android operating system. This is currently the most popular Os and widely used and provides mostly all the file sharing mechanisms. This application does not require any Internet support for the file transfer in mobile devices.

Today, smartphones are becoming very popular. These devices have evolved from very simple mobile phones to devices able to browse the Internet, access and read emails, and watch video stream directly from the net.All these new possibilities created new problems regarding the related requests for connectivity.mobile devices often work in environments with infrastructures that suffer from intermittent connectivity and difficulty in predicting changes in topology.

We are developing an android application for peer-to-peer file sharing named Mobile Torrent using which files can be shared between smartphones within a campus opportunistically without using the Internet. Each application user can share files and can register a request to download a file shared by any other application user

even if the source is not directly connected. Initially, we have implemented the same for single hop; i.e. when two smartphones come in the Wi-Fi communication range of each other, they auto-discover each other using Wi-Fi Direct and automatically exchange required files. Ideally, instead of using Wi-Fi Direct, an ad hoc network of smartphones should be established which is truly peer-to-peer. But, unfortunately, android does not support ad hoc networking. So, we are using Wi-Fi Direct instead.

1.1 Aim of Project

The project aim is to develop P2P file sharing application that allows the sharing of file from peer to peer.We will make android application peer to peer file sharing where peer can share the file.

1.2 Problem Definition

Due to the limited wireless radio range in mobile phone, we can not sustain continuous network connectivity between mobile phone. So if peer is out of wireless radio range then we are not able to share the file from peer to peer in mobile devices.

Peer-to-Peer file sharing application to share the files in an opportunistic network that will work perfectly with one to one communication.But it does not work the multi-hop communication in opportunistic network.

1.3 Objective of Study

We made peer-to-peer file sharing application which can be sharing the files between peer.

we provide a file sharing application with a simple user interface provides good speed to transfer the data between two devices by using the opportunity to transfer the packet as soon as the device come in contact.

1.4 Scope of Work

a. Study Peer-to-Peer file sharing architecture.

b. User interface designing for the application.

c. One to one peer communication and file transfer.

d. Multihop communication and file transfer.

e. Calculating bit rate, packet drop and success rate of the packets transmitted.

1.5 Methodology

a. Developing the flow of the application.

b. Integrate it with the file sharing android application.

c. Measuring the performance of the application.

d. Minimize the battery power, time and memory consumption.

1.6 Thesis Outline

The rest of the thesis is organized as follow:

In Chapter 2, We discuss Literature Survey in detail.The Literature Survey describes Delay Tolerant Networks (DTN) and Peer-to-Peer (P2P) file sharing systems.

In Chapter 3, We discuss proposed solution and the overall design of android application.

In Chapter 4, we discuss implementation details of the application in android.

In Chapter 5, we contain conclusion and presents the future work.

Chapter 2

Literature Survey

2.1 Introduction to DTN

Today,Transmission Control Protocol/Internet Protocol (TCP/IP) mostly used for a connected network on the Internet. TCP/IP protocol are depending upon the low latencies coupled with low bit error rates. So It will allow TCP transmitting the acknowledgement for the message and also receiving acknowledgement for message reliably across the terrestrial network.

A DTN is a network designed to operate effectively in highly-challenged environments where protocols adopted in connected networks (i.e. TCP/IP) fail. The D part in DTN acronym stands for Delay and also we say that D stands for Disruption. Now delay means that end to end latency for data transmission, but packet can store on intermediate node temporarily. Disruption means that connection can not establish due to changing the environment or we can says that connection has been broken down due to quickly changing the aspect of the system.

DTNs need not be solely concerned with deep-space communications but can also be useful in terrestrial networks. In some environment, networks may be subjected to high probability of disruption. One example is in military applications, where adopting DTNs allows the retrieval of critical information in mobile battlefield scenarios

using only intermittently connected network communications. Another more peaceful application is the adoption of DTNs to overcome a major natural disaster. In such a situation terrestrial infrastructures may have been swept away and tolerant protocols must be used to coordinate rescue teams.[1]

In [2], Networks adopted in these situations are characterized by:

- Intermittent connectivity: In TCP/IP protocol required end to end path and end to end communication between source and destination. If no end to end path from source to destination then TCP/IP protocol does not work.

- Variable or long Delay: Long propagation delay between node due to the intermittent connectivity that can be defeat the Internet protocol and application does not return the quickly acknowledgements or data for receiving the message. Data Rates: TCP/IP protocol provide asymmetric bidirectional data rate at the moderate level. In this protocol if asymmetric data rates are large then this protocol defeat on the Internet.

- High Error Rates: Retransmission of entire packet required due to bit error on given link. So few retransmission of packet required for hop by hop rather than end to end retransmission of the packet.

To overcome problems associated with all these factors, DTNs use a old yet affective method used in postal systems since ancient times. In this method, named Store-and-forwarding, nodes physically deliver data to the destination moving from the source location to the destination of the recipient node Replication techniques can be adopted to increase the delivery ratio, copying the carried data and passing it to other nodes following a different physical path.

Messages are stored in storage mediums which can hold them for long periods of time, called persistent storage. This is another difference from the Internet protocols, where routers adopt very short-term storage provided by memory chips to store incoming packets. In Internet routers, messages are queued only for a few milliseconds

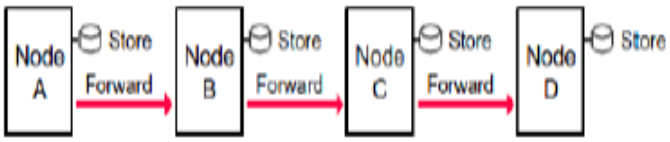

Figure 2.1: Example of the Store-and-Forwarding Technique [3]

while they are waiting for their next hop. In DTNs routers need persistent storage for one or more of the following reasons:

- A communication link to the next hop may not be available for while and during this period message must be stored in the router.
- We can say that asymmetric in term of reliability and speed between the node, So node communicating to each other to send or receive data more reliable and faster than another node.
- Once the message has been transmitted it may be required to retransmitted if the link may be broken or node does not receive forwarded message.

2.2 A Delay/Disruption Tolerant Solution For Mobile-to-Mobile File Sharing (M2MShare)

In DTNs mobility and low node density require new strategies to implement a file sharing application which allow the exchange of multimedia contents between mobile devices.In [4] M2MShare uses Bluetooth to create a Peer-to-Peer overlay network and uses node mobility to reach data content on other local disconnected networks.

The main idea of this protocol is to use a store-delegate-and-forward model. This is similar to the store-and-forward strategy widely used in DTNs, but M2MShare adds the delegate part to the model. The result is an asynchronous communication model where a peer delegates an unaccomplished task to other peers in the overlay network.

Delegations are used to extend search and diffusion area for a shared file, while M2MShare dynamically establishes forward routes along the destination path by exploiting social relations existing between peers users. This is done by limiting delegations only to frequently encountered peers, which users can pass on by the same geographical location at the same time frequently enough. The history of previous encounters is then used as heuristic evaluation of whether a peer is a good candidate for delegations. This allows assigning tasks only to nodes that we should meet again in the future, so they can return the result of the task back to us.

In a Peer-to-Peer application it is also fundamental to have a high availability of shared files, to ensure the recovery of a searched file. This implies the need for a high number of simultaneously connected nodes, which we saw was not so obvious in a mobile network.

To overcome these limitations, M2MShare uses an asynchronous communication strategy in which a client peer, in search for a file, can delegate to another peer, a servant, the task of searching for the file and returning it to the requester. Delegation system is at the root of M2MShare. This permits widely extending the area of where to look for the searched file, in a network composed of spread out and poorly connected

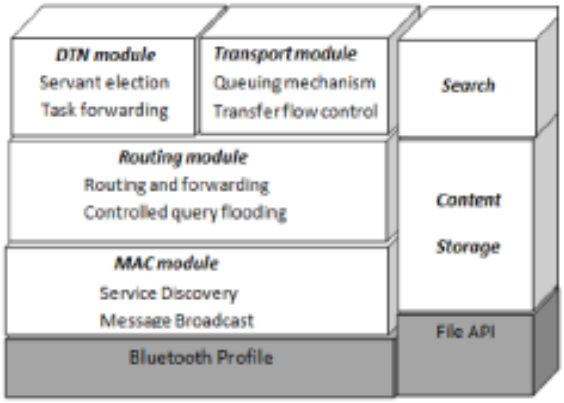

Figure 2.2: M2MShare Protocol Stack
[3]

nodes.

Before proceeding further it would be useful to explain in what a delegation action consists of. A delegation is the action performed by a client peer which asks a servant peer to execute some unaccomplished task of its own. This unaccomplished task can be a query, composed of several keywords, for which the client wants to receive a list of files which satisfy it. To accomplish this task, the servant schedules it for later execution and searches among shared files in encountered nodes for some file satisfying the query. When the pending task is completed, the servant creates a new task type, a forward task. The objective of this kind of task is to return the output of a pending task to the requester node.

One task can be delegated by the client to several servant peers and this allows using nodes mobility as an advantage instead of a negative factor.

All delegated tasks also have a Time To Live (TTL) value. If a delegated task is not completed and its result has not been returned to the client within its TTL value the task expires, it is deleted and not scheduled again for execution in the servant.

2.3 Performance Evaluation With Realistic Mobility of a File Sharing DTN Protocol

M2MShare protocol providing following modules. 1. Search Module 2. DTN Module 3. Transport Module 4. Routing Module and 5. MAC Module.

2.3.1 DTN Module

M2MShare protocol use store-delegate-forward model. DTN module is useful for intermittent connectivity among the node. In Mobile Network, every node has limited connection range and high mobility. So we can not establish an end to end connection between source node and destination node. To overcome this limitation,M2MShare uses asynchronous communication strategy in which client peer, search for file, can delegate to another peer.

A delegation part of M2MShare model is performed where peer ask to a servant peer to execute some unaccomplished task of its own. This unaccomplished task can be include query, composed of several keywords and which the client wants to receive a list of files. So the objective of this task is to return output to the requestor node. The task can be delegated to by client peer to several servant peer. This is not good strategy for to delegate unaccomplished task to every servant peer.So we will select servant peer for complete those task.[5]

2.3.2 Servant Election

In mobile environment, where routing path dynamically created during the execution.Before creating the path we need to choose which peer can be elected as servant. One possible solution to elect the servant, delegate task to every met node that would be expensive in bandwith usage and energy consumption. This is not best solution to elect the servant. So M2MShare strategy is elect the servant only node that we expect to meet again.

Now we will check how frequently node meet to another node in servant election. M2MShare uses an active daemon called Presence Collector. This daemon traces the number of encounter node, scanning at regular intervals for other nodes within communication range. When the number of encounters with another node exceeds the value of a parameter called Frequency Threshold, that node is elected as servant.

2.3.3 Search Module

Before a user can choose what file to download from the network, it must know what files are shared by other users. Every peer maintains a local repository of files shared with other users. Each file included file name, dimension and an hash value that univocally identifies that file in the network. There are two kind of query:1)unique file query 2)keyword query

2.3.4 Transport Module

Each mobile device must simultaneously handle incoming requests for upload and delegations from other peers. If requests are parallel executed then each would be separate execution flow. Each mobile devices limited resources, this could cause substantial memory usage,probably crash the application. So this module manages task queuing and execute limited number of them simultaneously.

Queuing Central mechanism which manages the queues. It also imposes limits on the maximum number of tasks a single peer can handle. Tasks execution is scheduled by a component called Scheduler, which extracts tasks from related queues and executes them. Inside each queue tasks are inserted and fetched with a FIFO policy. The Scheduler includes four execution flows used to implement inter-queue policies and to assign different priorities to different task types. Some tasks can use several execution flows, with a maximum number of flows for each task. This permits, for example,simultaneous downloading of pieces of the same file from different sources.

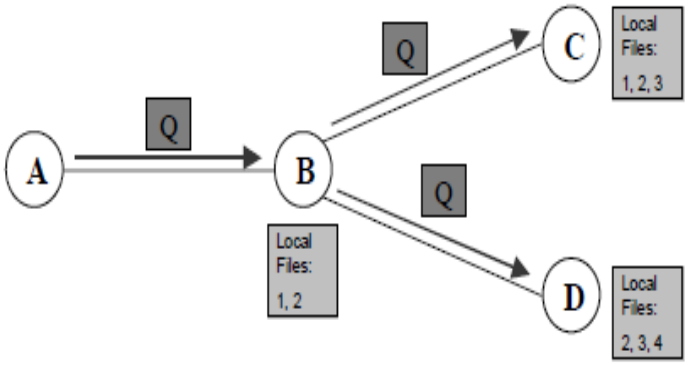

Figure 2.3: Node A Floods a QUERY Message With Keywords Matching to Files 1 to 4

[6]

2.3.5 Routing Module

M2MShare uses this module for query routing using Relay method used in ORION protocol and query has been broadcasting using flooding mechanism used in AODV protocol. Optimized Routing Independent Overlay Network (ORION) comprises search algorithm and transfer protocol used for routing all type of message like query, response and file transmission required in P2P file sharing application.

ORION Protocol

This protocol basically used for routing the different type of message such query,response and file transmission in peer to peer file sharing application.ORION combines the network layer of route discovery and application layer processing.Orion protocol does not depend on the deploying any MANET protocol.ORION protocol comprises the searching algorithm and basic transfer protocol.

In ORION protocol,each mobile device containing local repository which consist set of file stored in file system.This protocol also provide two routing table.1)response routing table store the node from query message received and to return response to requester node.2)File routing table is used for to store next hop for file transfer based

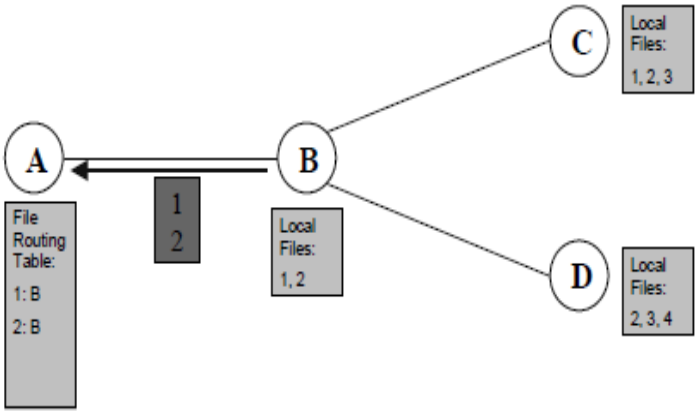

Figure 2.4: Node B Sends a RESPONSE Message With Identifiers of Local Files
[6]

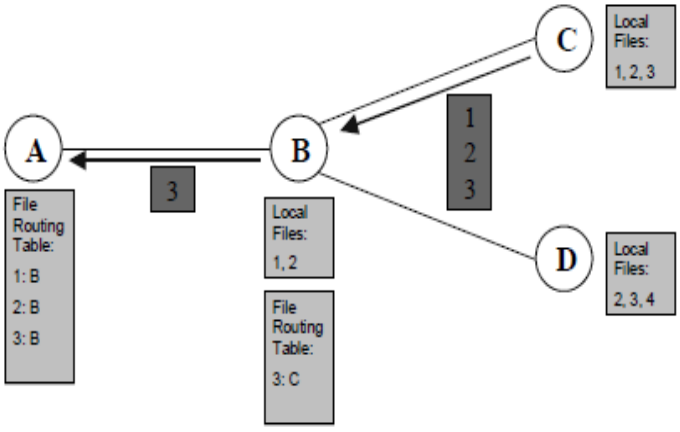

Figure 2.5: Node C Sends a RESPONSE Message With Identifiers of Local Files, Node B Filters and Forwards RESPONSE of C
[6]

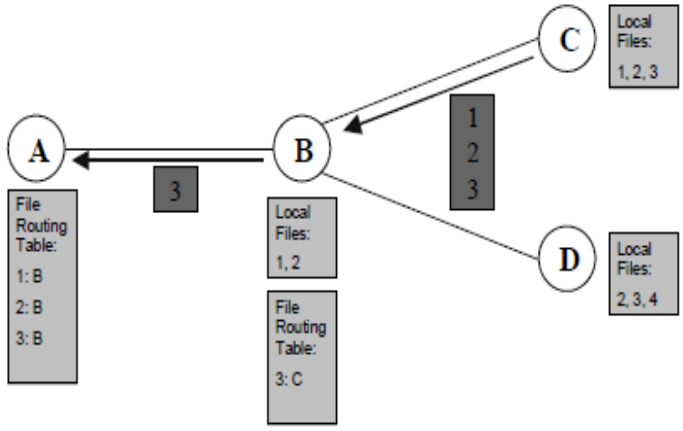

Figure 2.6: Node D Sends a RESPONSE Message With Identifiers of Local Files; Node B Filters and Forwards the RESPONSE message
[6]

on file identifier.

We can see the fig 2.3 all mobile node see in circle and all the node are connected by line and each mobile node contain local repository and file routing table.Now,node A query message are broadcasting using flooding mechanism and each node searching in own local file system.After searching local repository node B send response message contain file identifier 1,2 to node A.similarly node C send response message contain file 1,2,3 to node A's direction.Before forward message to node A,node B examines those result.

If file are unknown to node B then it is stored file in node B file routing table.node B sends a reduced RESPONSE message to node A containing only the new file identifiers(see fig:2.5).similarly node D response message contain query matching file to node A's direction.But node B received those file and reducing the response meassge and send to node A.Because node B already forwarded response to file 1,2,3.so node B send only file 4 to node A (see fig:2.6)

2.3.6 MAC Module

In this module provide fundamental service called presence collector which gathering the information about mobile devices. So we can determine which mobile node are frequently meet and have a expectation to meet again in future. Presence Collector service is active running which periodically scan the network for in reach areas devices. Presence collector service to track encounters between other nodes to realize the election strategy selected for the current mobile node.When another mobile node exceeds a value named Frequency Threshold, it can be elected as a servant and receive unaccomplished tasks as delegations from the current node.

2.4 A Special Purpose Peer-to-Peer File Sharing System for Mobile Ad Hoc Networks

In MANET, we require the algorithm for transmit the queries and return the search result related to queries and also download those file which is matching to the query.so we represent special purpose approach for P2p file sharing to MANET denoted ORION protocol.ORION [7] is another protocol based on an application layer overlay network for peer-to-peer file sharing in an opportunistic network. In an overlay network, overlay routes are set up on demand by using the searching algorithm and redundant transfer path on per file basis. ORION protocol combines the network layer and application layer for routing all type of messages like queries, response, and file transmission. Benefits of using ORION protocol are reduced control overhead, low overhead file transfer and increased probability of successful file transfer. ORION protocol is simulated in the NS-2 simulator and it enables more reliable file transfer with low overhead.[6].

2.5 A Survey on Peer-to-Peer File Sharing Using Network Coding in Delay Tolerant Networks

Network coding can be used as a tool to improve the delivery probability of a file to an interested user. In network coding, a forwarding node linearly combines two or more incoming packets into one or more outgoing packets. In peer-to-peer file sharing, a file is divided into chunks of equal size. Ideally, nodes should forward chunks such that all chunks have the same number of copies in the network. But, it is not possible to keep track of the number of copies of a chunk in the network. In network coding, each combined chunk contributes the same to the eventual delivery of the file because network coding does not require delivery of the specific chunk but it requires enough number of independent chunks. A review of network coding based peer-to-peer file sharing in DTN is available in [8]. We intend to implement network coding in our android application in future. In[9] A variant of Network Coding that is Random Linear Network Coding (RLNC) is examined for peer-to-peer file transfer for reliability and robustness.

2.6 Android

Android provides Wi-FI Direct in the versions above 4.0. This version provides multi tasking environment, deep interactivity and powerful new ways for communication and sharing.It also supports Wi-Fi Direct for more reliable connectivity without any Internet or tethering. It lets us connect directly to the nearby peers. Android has different components to provide different functionality.Basically, android application is written based on java programming language. When we compile the java code contain the data and resource file into the APK file using android SDK tools. In APK file including the content which is required by the user use this application and installing the application on the device.

Components of Android Application: There are different components of the an-

droid application, each of them serve a different purpose. The life cycle of these components defines how they created and how to get destroyed. So here some details about the components[10].

- Activities:Activities interact with the user so activity presents you the screen to interact with.Activities in the system manage on the stack whenever the activity started its placed on the top and come in the running stack the previous activity cannot be resumed until the top running activity got finished. Activity is implemented as the subclass of activity. Activity has different methods.The full lifetime of the activity is from oncreate() to ondestroy(), visible lifetime is from onstart() to onstop() and foreground lifetime is from onpause() to onresume().

- Services:Services run in the background to perform the operations. if any other application is running on the system the service continues to run in the background.services perform interprocess communication. to create the service we need to implement the subclass of the service class. Some methods are present in the service class we need to override to create the service. StartService(), StopService() are the methods to invoke services.

- Content Providers:Content provider manages the access to the data. Its provides the interface which connects the data in one process and code running in another process. There is no need to create the content provider for the application if there is a need to share data, it could be established with the implementation of the subclass of the content provider.

- Broadcast Receivers:To receive the system wide broadcast announcements as well as for providing the response towards it. we use the broadcast receiver. To use this we need to implement the subclass of the broadcast receiver.

- Manifest:This should be present at the root of the directory. it defines the permissions that are required for the application. This defines the API level require for the application. This application works on the API level Equals to

or greater than 14. With all that it defines the software as well as the hardware requirements of the application.

- Resources:Application interface requires images as well as layouts these are defined in resources. In this project we are using two layouts file.xml and main.xml.The layout files are defined in XML. layout files are present as the user interface for the user. It consists of different tabs, buttons and text fields.

- Generated Java Files:There are two autogenerated java files present in the system. First is R.java and second is Buildconfig.java. R.java file contains the unique identifiers for the resources for the easy access of the elements. while buildconfig.java has a Debug constant that automatically set according to built type.

2.7 Summary of Literature Survey

Literature survey confines that there is so much research is going in the direction of efficient and fast file sharing, and for that, lots of efforts and new techniques are arising day by day. these techniques are rather based upon the routing or the Bluetooth and different features that could be useful.while talking on android phones it provides different features such as Wi-Fi Direct and Bluetooth and most of the people those who are having smartphones are using android. hence, the applications it supports are also a way higher than any other phone operating system. So with the help of such and adaptive operating system, it is quite efficient to build a network. A network that has no need for network connection as well as no need for the infrastructure to perform file sharing action.

From the Literature survey , we conclude that no actual implementation of peer-to-peer file sharing in opportunistic network is available. So, we propose one such implementation in this thesis. As smartphones are very common and android is the most popular smartphone operating system, we have implemented the application in

android.

2.8 Our Choice

Hence using the previous work the base of the new technique could be judged at the ground level and new technique could be build. File sharing between peer-to-peer has been done in ONE simulator for one to one communication.We are trying to make peer-to-peer file sharing application for multi-hop communication in android platform.

Chapter 3

Proposed Approach

3.1 Proposed Solution

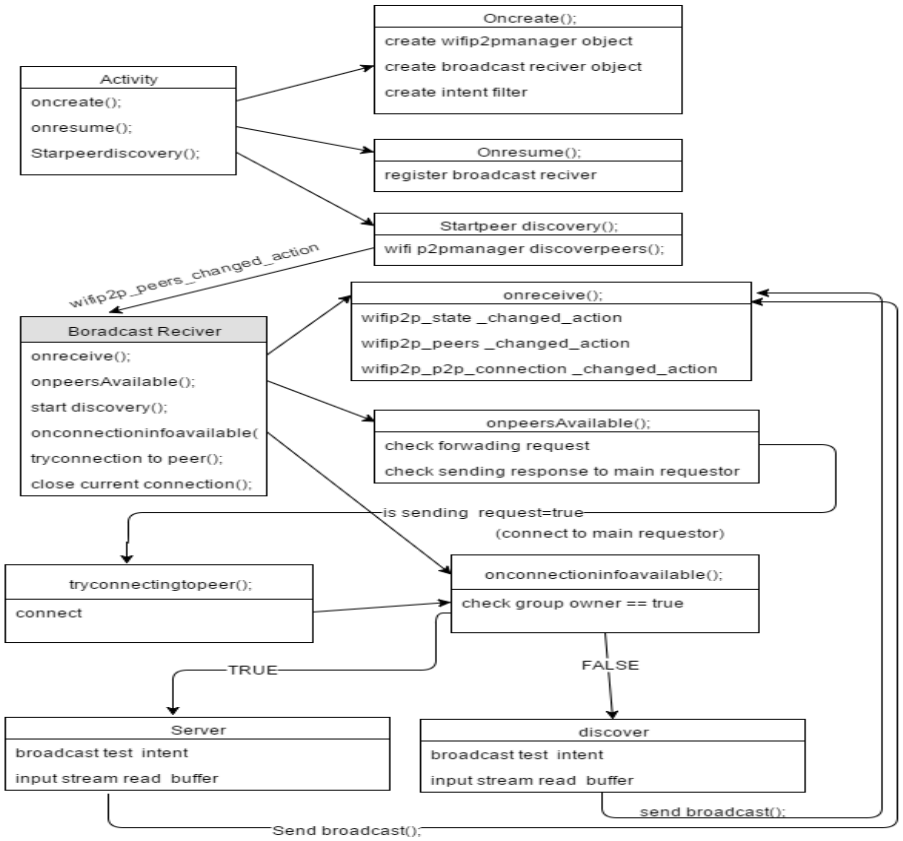

Figure 3.1: Design Of Application

File sharing provides a facility to the users to share files they have with other users. Peer-to-peer file sharing provides the advantage that a node can act as a server as well as a client. It can be the source of a file and at the same time, it can download a file from other nodes as a client. In this application, we have used Wi-Fi Direct. Wi-Fi Direct does not require any wireless access point to connect with each other. This is relevant for applications that share data among the users such as file sharing, photo sharing or multi-layer gaming. In the following subsections, we discuss the implementation of the application in detail. The overall design of the application is depicted in Fig. 3.1.

3.2 Application Flow

3.2.1 Activity Class

This activity includes so many phases. Normally an android activity life cycle looks like that. In this application we have TorrentActivity.java class that extends the activity class. This application have only this activity. It initialize various components of the application, like WifiP2pManager, Channel, IntentFilter and WifiDirectBroadcastReceiver.Now the working of different class.

As we know the activity includes such life cycle our activity also has different methods define.

- onCreate(): Two different objects are created of different classes. First is WifiP2pManger class, Second is Broadcast Receiver class.With that Channel and intent filter is created.

- Onresume(): We register Broadcast receiver. Description of broadcast receiver is defined next.

- StartPeerDiscovery(): When this method is invoked WifiP2pManger.discoverpeer() method is called. This passes an intent to the Broadcast Receiver class.

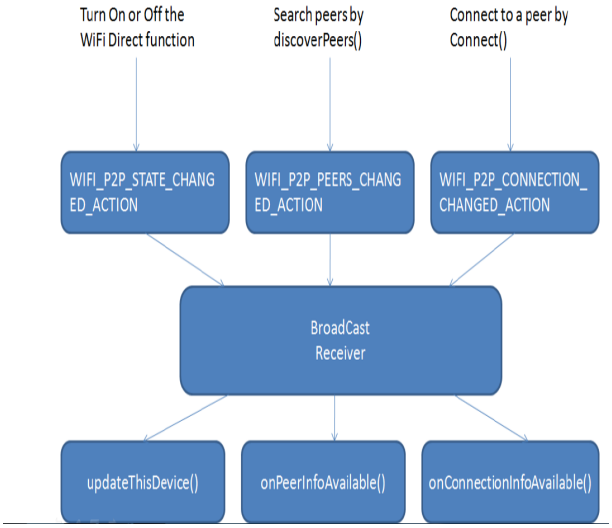

Figure 3.2: Broadcast Receiver Class Intents and Methods

3.2.2 Broadcast Receiver Class

We have to a show recipient class in light of the fact that a Broadcast Receiver permits to accept goals show by the Android framework, so requisition can react to intents that we are intrigued by. The essential steps for making a telecast recipient to handle Wi-Fi P2p expectations are as takes after:

- Make a class that enlarges the Broadcast receiver class. For the class constructor, we must have parameters for the Wifip2pmanager, Wifip2pmanager . channel, and the action that this show collector will be enlisted in. This permits the telecast beneficiary to send redesigns to the action and in addition have entry to the WiFi fittings and a correspondence station if necessary. Wifip2pmanager is really essential and here are the strategies, interfaces and the goals characterized under that.

WifiP2pManager

The primary class we need to work with is WifiP2pManager, which you can acquire by calling getSystemService (WIFI P2P SERVICE). The WifiP2pManager includes APIs that allows to:

- Initialize your provision for P2p associations by calling initialize().
- Discover close-by gadgets by calling autodiscoverpeers().
- Start a P2p association by calling autoconnect().
- The Wifip2pmanager.actionlistener interface permits get callbacks when an opera-tion, for example, uncovering companions or interfacing with them succeeds or falls at.
- Wifip2pmanager.peerlistlistener interface permits get data about uncovered companions. The callback gives a Wifip2pdevicelist, from which you can recover a Wifip2pdevice object for every gadget inside reach and get data, for example, the gadget name, address, gadget sort, the WPS arrangements the gadget helps, and that's just the beginning.
- The Wifip2pmanager.groupinfolistener interface permits accept data around a P2p bunch. The callback gives a Wifip2pgroup object, which gives bunch data, for example, the manager, the system name, and pass phrase.
- Wifip2pmanager.connectioninfolistener interface permits accept data about the current association. The callback gives a Wifip2pinfo object, which has data, for example, whether a gathering has been structured and who is the gathering holder.
- We have to have admittance to the distinctive assets of the framework henceforth we have to get consents, for example, ACCESS WIFI STATE,CHANGE WIFI STATE.
- A channel that interfaces the requisition to the Wifi p2p structure. An occasion of Channel is acquired by doing an approach initialize(context, Looper, Wifip2pmanager.channellistener.

- In the telecast recipient, check for the plans onreceive().there are distinctive goals that could be accepted. we have to perform essential activities relying upon the goal that is accepted. Firstly the purpose WIFI P2P STATE CHANGED ACTION will be accepted and afterward Onpeersavailable() we have to check if the associate is sending the solicitation or the this is the first ask for in the event that its an unique ask for then the reaction might be distinctive. After that tryconnectingpeers() system conjures. This will prompt interface the gadgets.

- If the association will secured then We have to check whether the gadget is the gathering holder or not. This might be resolved with the utilization of On connection data Available. In the event that the gadget is the gathering manager then the that will be exchanged to the server class and if not then the reaction will go the revelation class.

3.2.3 Server and Discovery Class

This class will create a sever and client socket at the specified port and blocks it until it receives the connection response. This class extends the AsyncTask class This will make it run in background with the other methods running.When the client invokes the accept method the connection will establish. Working of the server and client class.

a. We have created some custom intents in the activity. TEST intent is one of them. This intent makes the connection process validate and then the connect button will appear on the screen. The TEST intent of the server class will be received by the client class and vice-verse.

b. TEST intent will create the UserInput class for client and server.

Chapter 4

Experiments

4.1 Initialize Application

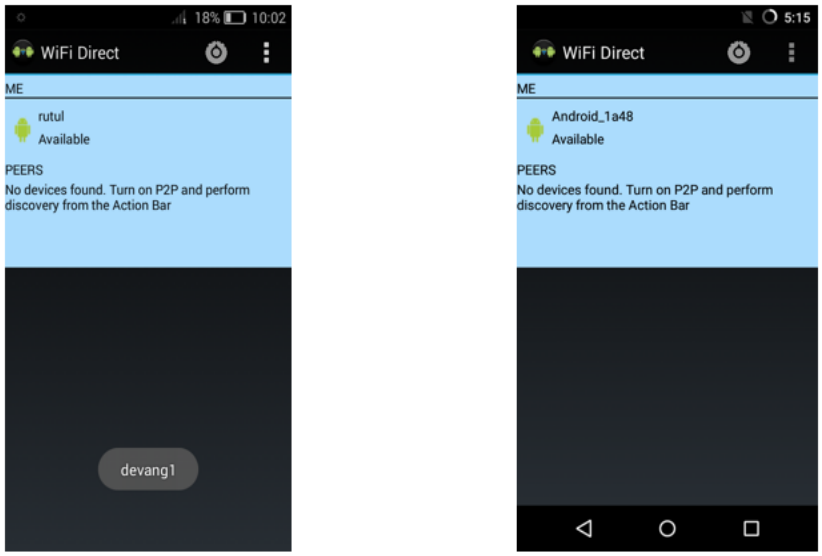

Figure 4.1: Starting the Application

For using WiFi APIs, the application must be able to access the hardware and the cell phone device should have the support of WiFi P2P protocol. We can check this at broadcast receiver when it receives the WiFi P2P STATE CHANGED ACTION intent. In the onCreate() method, take the instance of WiFiP2pManager and call

initialize() to register the application with the WiFi P2P framework. This method returns a WiFIP2pManager.Channel, which helps to connect the application to the WiFi P2P framework. Create an intent filter and add the same intents for which the broadcast receiver checks. In the onResume() method register the broadcast receiver and unregister it in the onPause() method of the current activity. Now use the WiFi P2P features by calling the methods in WiFIP2pManager by implementing the application.Fig. 4.1 shows screen shots of the application when it is started.

4.2 Auto Discovered Peers

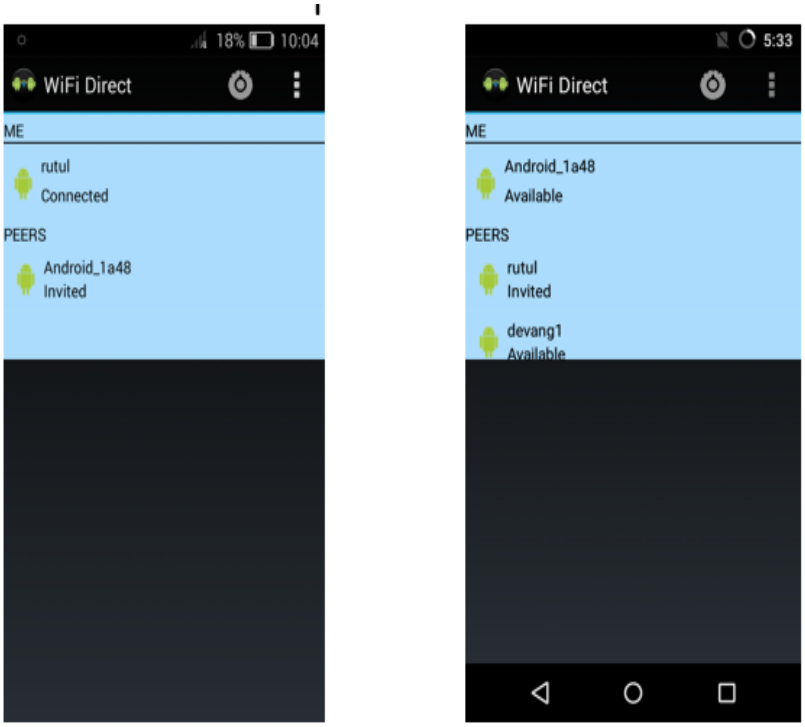

Figure 4.2: Discovered Peers

For discovering the peers which are available within the range for connection, call Autodiscoverpeers(). This call is asynchronous and a success/failure is communicated to the application via onSuccess() and onFailure(), if WifiP2pManager.ActionListener is already created. The on- Success() method would notify only about the success of

peer discovery and would not provide any information about the discovered peers.Fig. 4.2 shows discovered peers.

4.3 Auto Connecting Peers

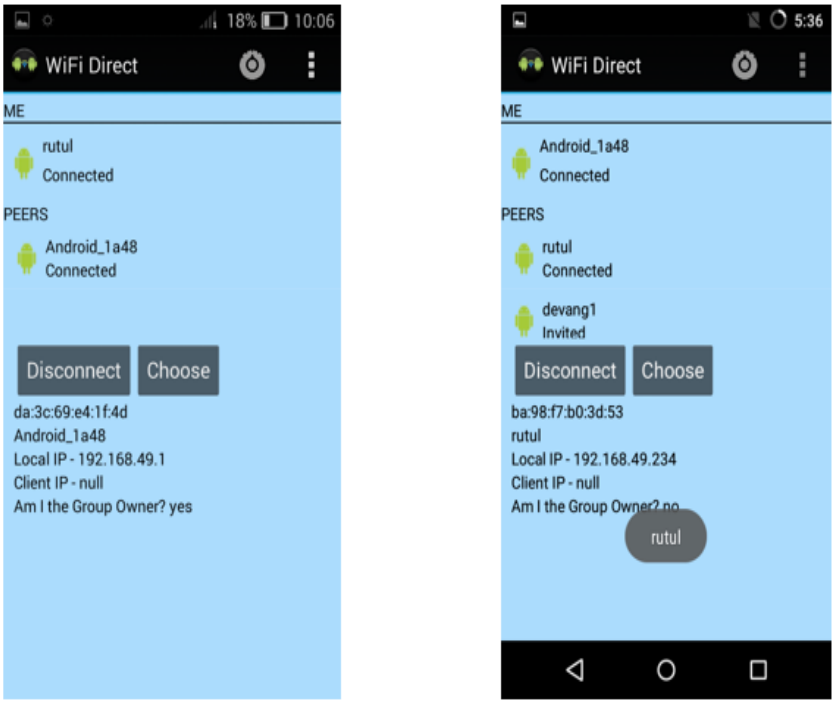

Figure 4.3: Connecting to Peers

When we call the Autodiscoverpeers() method which is obtaining the list of possible peers in peers wifi range than one device want to connect to another device we call autoconnectpeers() method connect only first devices among the list of devices. Now, autoconnectpeers() method needed a WifiP2pConfig object that containing the information about the which device want to connect.Fig. 4.3 shows connected peers.Once two devices are connected, they exchange shared files with each other and then disconnect themselves. Please note that in the entire process, there is no user intervention.

Chapter 5

Conclusion and Future Scope

5.1 Conclusion

This application will make with a motive to research about the peer to peer file sharing application that can work on android and without any Internet and centralization could perform file sharing with better results. we are using M2MShare protocol and using this protocol we will make peer to peer file sharing application in android.we have done designing of this application and this application efficiently transferring the data between mobile devices.The application is built successfully and all the aspects of android, Wi-Fi Direct are studied and implemented.Hence, we can say that the goal is achieved.The Project could be real time project and would be beneficial for the users to transfer file in case Bluetooth or another access point facility is unavailable.We finally have an application that performs data transfer on an efficient transfer rate and for longer distance. The results are discussed primarily that shows that there are still a lot that could be done in this area and in this application only. We studied the limitations of Wi-Fi Direct but at the same time explored the benefits that we could take with this technology. Peer to peer file sharing with that has opened new doors for various data transfer application.

Peer-to-peer file sharing between smartphones is explored in the literature only

through simulation. We have developed an actual android application for the same. For opportunistic networking, it is required that phones should be able to discover other phones when they come in the communication range. A phone should also be able to connect to a neighbour phone and transfer required content. All these should be done automatically without any user intervention. We are successful in achieving the same using Wi-Fi Direct.

5.2 Future Scope

We would develop the application in which the file sharing could be done with the least delay and with no centralized server and storage facility.We provide a file sharing application with a simple user interface provides good speed to transfer the data between two devices by using the opportunity to transfer the packet as soon as the device comes in contact.Currently, our application transfers a single file to an immediate neighbour successfully. The next step is to implement network coding based multicopy forwarding algorithm while dividing a file into chunks.

References

[1] K. Fall, "A delay-tolerant network architecture for challenged internets," in *Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, pp. 27–34, ACM, 2003.

[2] A. Bujari and C. E. Palazzi, "Opportunistic communication for the internet of everything," in *Consumer Communications and Networking Conference (CCNC), 2014 IEEE 11th*, pp. 502–507, IEEE, 2014.

[3] C. E. Palazzi and A. Bujari, "A delay/disruption tolerant solution for mobile-to-mobile file sharing," in *Wireless Days (WD), 2010 IFIP*, pp. 1–5, IEEE, 2010.

[4] C. E. Palazzi, A. Bujari, and E. Cervi, "P2p file sharing on mobile phones: Design and implementation of a prototype," in *Computer Science and Information Technology, 2009. ICCSIT 2009. 2nd IEEE International Conference on*, pp. 136–140, IEEE, 2009.

[5] A. Bujari, C. E. Palazzi, and D. Bonaldo, "Performance evaluation of a file sharing dtn protocol with realistic mobility," in *Wireless Days (WD), 2011 IFIP*, pp. 1–6, IEEE, 2011.

[6] A. Klemm, C. Lindemann, and O. P. Waldhorst, "A special-purpose peer-to-peer file sharing system for mobile ad hoc networks," in *Vehicular Technology Conference, 2003. VTC 2003-Fall. 2003 IEEE 58th*, vol. 4, pp. 2758–2763, IEEE, 2003.

[7] A. Duran and C.-C. Shen, "Mobile ad hoc p2p file sharing," in *Wireless Communications and Networking Conference, 2004. WCNC. 2004 IEEE*, vol. 1, pp. 114–119, IEEE, 2004.

[8] V. U. Parikh and Z. Narmawala, "A survey on peer-to-peer file sharing using network coding in delay tolerant networks,"

[9] M. Yang and Y. Yang, "Peer-to-peer file sharing based on network coding," in *Distributed Computing Systems, 2008. ICDCS'08. The 28th International Conference on*, pp. 168–175, IEEE, 2008.

[10] "Wifi direct:peer to peer file sharing, http://developer.android.com/training/connect-devices-wirelessly/wifi-direct.html."