International Journal of Computer Applications (0975 – 8887)
Volume 121 – No.15, July 2015

Implementation of Distributed Variable Chunk based Switching in Peer to Peer Network

Shambhuraj Deshmukh
Assistant Professor
V. V. P. I. E. T. Solapur, India

Sagar Mane
Assistant Professor
V. V. P. I. E. T. Solapur, India

G. A. Patil
Associate Professor & HOD CSE
D. Y. P. C. E. T. Kolhapur, India

ABSTRACT

The Peer-to-peer (P2P) computing has been one of the emerging technologies, particularly contributing in distributed file sharing. Experimental studies show that for a file download, network congestion or service capacity fluctuation takes minutes to several hours. For a P2P n/w one of the fundamental performance metrics is the average download time. The common approach to analyse the average download time is average service capacity. Heterogeneity and fluctuation have significant impact on service capacity and hence averages the download time. Random Chunk Based Switching is one of the file downloading scheme where, the file to be downloaded is divided into many chunks but the shortcomes of this scheme are mentioned below. User sequentially downloads one chunk at a time. If user gets stuck in a low service capacity peer, downloading a fixed amount of bytes from that peer may take a long time. Chunk size remains fixed and it does not change with time.

We have designed and implemented a new scheme Distributed Variable Chunk Based Switching where chunk size changes with time. The downloader will be downloading the file from different peers. If bandwidth available is increased then downloading can complete before specified time. If bandwidth available is decreased then downloader will search another peer with good bandwidth and get it replaced. Our new scheme removes heterogeneity and fluctuation. It also provides a distributed approach to a sequential Random Chunk Based Switching.

Keywords

Service Capacity, P2P, JXTA.

1. INTRODUCTION

The basic idea of P2P network is to have peers participating in file sharing network and a P2P system that is distributed. Every computer (peer) in the network can act as both a server and a client. If a host has a popular file and many peers are requesting it, as soon as one of the downloading hosts finishes the download [4], it can become a server to service other peers in the network. As time passes on due to the increase in the number of servicing peers the service capacity of the entire network also increases.

Research on P2P system considers traffic measurement, workload analysis and behaviour of peers of P2P applications like Gnutella and Napster and it has been observed that there is significant heterogeneity in peers bandwidth, availability and transfer rate. Much of the research is focused on characterizing overall P2P system e.g. request pattern, traffic volume, traffic categorization.

Earlier results show that a file download session in a P2P network is rather long and varies a lot from user to user. Due to the distributed nature of the P2P network, Searching and locating data of interest in the network has been an important issue in the literature.

If we want to minimize the download time for each user, reducing the actual file transfer time would make a noticeable difference. Most of the recent studies, however, have focused on reducing the total download duration. As the measurement study shows that per-user performance in a P2P network may be even worse than that of centralized network architecture. Those results suggest that there is much room for improvement in the P2P system in terms of per-user performance.

However, there have been very few results in minimizing the download time for *each user* in a P2P network. In recent work [2], the problem of minimizing the download time is formulated as an optimization problem by maximizing the aggregated service capacity over multiple simultaneous active links (parallel connections) under some global constraints. There are two major issues in this approach. One is that global information of the peers in the network is required, which is not practical in real world. The other is that the analysis is *based on the averaged quantities*, e.g., average capacities of all possible source peers in the network. The approach of using the average service capacity to analyze the average download time has been a common practice in the literature.

1.1 Service Capacity in P2P Network

The service capacity [5] of a P2P system is modelled into two regimes. One is the transient phase in which the system tries to catch up bursty demands. Both analytical model and trace measurements exhibit the exponential growth of service capacity during the transient phase.

Second is the steady state, the service capacity of a P2P system will scale and track the offered loads, so individual user's performance will not degrade significantly. Both analysis and empirical data suggest that at higher offered loads and with cooperative users, the system performance might improve.

1.2 Our contribution

The main contribution of this paper is that, (1) during downloading if we get stuck in a bad source peer with low bandwidth capacity; there arises a need to switch from low bandwidth peer to high bandwidth peer, (2) to design a distributed approach for chunk based switching, (3) to design a Source Replacement approach.

2. FACTORS OF AVERAGE DOWNLOAD TIME

In this section, we consider the heterogeneity of over different network paths and the fluctuation of the capacity over time for a given source peers.

2.1 Heterogeneity of Service Capacity

In a P2P network, the service capacities from different source peers are different. There are many reasons for this heterogeneity. On each peer side, physical connection speeds at different peers vary over a wide range. On the network side, peers are geographically located over a large area and each logical connection consists of multiple hops. Hence, we assume that those factors mainly determine the long-term average of the service capacity over a given source peers.

2.2 Correlations in Service Capacity

While the long-term average of the service capacity is mainly governed by topological parameters, the actual service capacity during a typical session is never constant, but always fluctuates over time. There are many factors causing this fluctuation. *First*, the number of connection a source peer allows is changing over time, which creates a fluctuation in the service capacity for each user. *Second*, some user applications running on a source peer, such as games, may throttle the CPU and impact the time amount of capacity it can offer. *Third*, temporary congestion at any link in the network can also reduce the service capacity of all users utilizing that link.

3. EFFECT OF EXISTING METHOD FOR FILE DOWNLOADS IN P2P NETWORK

In this section, we briefly discuss all the existing methods that are used to download a file from P2P networks. And we design our new Distributed Variable Chunk Based Switching. Generally, if a downloader relies on a single source peer for its entire download, then it results in high download time. Since the service capacity of each source peer is different and fluctuates over time, utilizing different source peers either simultaneously within one download session would be a good idea to diversify the risk. The existing methods are (i) Parallel Downloading; (ii) Random chunk-based switching; (iii) Periodic Switching.

3.1 Parallel downloading

Here a file gets divided into equal size of chunks and single file is allowed to download in parallel. Parallel downloading is better than single downloading.

3.2 Random chunk based switching

If we get stuck in a source peer with very low service capacity, downloading a fix amount of bytes from that source peer may take a long time. Switching based on chunk can reduce the correlation in service capacity between chunks and hence reduce average download time. But there is negative impact of spatial heterogeneity on average download time.

3.3 Random Periodic Switching

Instead of waiting to get the complete chunk, we randomly switch between the source peers based on time. There are two schemes in this method (i) Permanent Connection (ii) Random Periodic Switching.

It is observed that both the spatial heterogeneity and the temporal correlation in the service capacity can significantly increase the average download time of the users in the network.

3.4 Dynamically Distributed Parallel Periodic Switching

Dynamically Distributed Parallel Periodic Switching (D2PS) that effectively [1] removes correlations in the capacity fluctuation and the heterogeneity in space, thus greatly reducing the average download time. There are two schemes in this method.

3.4.1 Dynamically Distributed Parallel Permanent Connection (D2P)

Here the downloader randomly chooses multiple source peers and divides the file randomly into chunks and download happens in parallel for the fixed time slot and source selection function does not change for that fixed time slot.

3.4.2 Dynamically Distributed Parallel Periodic Switching (D2PS)

Here the downloader randomly chooses multiple source peers and divides the file randomly into chunks and downloads in parallel for each randomly selected time slot.

3.5 Limitation of the Existing methods

1. Average service capacity alone is not sufficient to describe each user's average performance.
2. Source selection function randomly selects source peer but that source may be with low bandwidth.
3. A source has more bandwidth but downloader cannot utilize this bandwidth.
4. Chunk size is fixed for a fixed time slot and doesn't changes with time.
5. Random chunk based switching is sequential approach.

4. DISTRIBUTED VARIABLE CHUNK BASED SWITCHING

4.1 System Information

As shown in Fig.1 every peer in the group contains Data packet, Control Packet and Peer Index. Here we are using two connections (Rendezvous Process) between any two peers. *Data Packet connection:* where only data packets get transmitted between peers. *Control Packet connection*: where only control packets get transmitted between peers. Control packet (Fig. 2) contains Peer ID, Bandwidth of that peers, and number of source it contain and number of groups it has joined. Control packet helps to decide the source selection and chunk size. Control packets gets exchanged between the peers after a fixed time slot (Fig. 3) and each peer is maintains a Peer Index (Fig.4) where information regarding all peers is maintained from the control packets history.

Here Rendezvous process between any two peers uses Rendezvous Protocol which is designed to propagate messages between peers within a group and minimizes network traffic also minimizes burden on the peers and it increases bandwidth

Data Packet	Control Packet	Peer Index

Fig. 1 Peer Diagram

4.2 Chunk size decision module

After the downloader has received the control packet, we decide the chunk size of the peers (Fig. 5). The downloader will be connected to no. of peers in the group and the downloader will be downloading the file in parallel from these different peers. If bandwidth available is increased then downloading can complete before specified time. If bandwidth available is decreased then downloader will search another peer with good bandwidth and get it replaced (Fig. 6). After downloading all chunks from the all sources, the system will check whether the entire file got downloaded or not.

Peer ID Bandwidth No. of source No. of Group

Fig. 2 Parameters of Control Packet

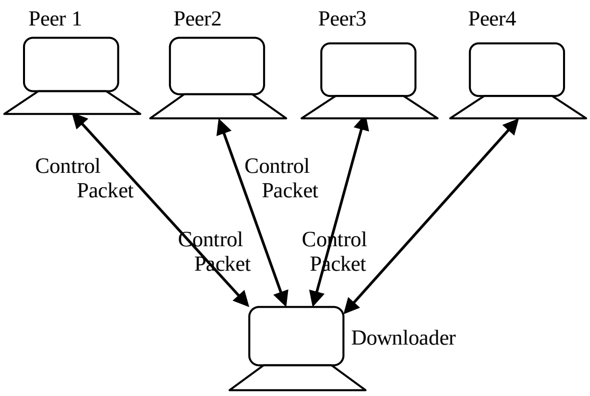


Fig. 3 Control Packet Flow Diagram

Node1	Bandwidth No. of sources No. of Groups
Node2	Bandwidth No. of sources No. of Groups
Node3	Bandwidth No. of sources No. of Groups
	..
	..
	..
Node n	Bandwidth No. of source No. of Group

Fig. 4 Peer Index Diagram

5. METHODOLGY USED

Following methodology are used to implement the proposed system.

5.1 JXTA

JXTA (Juxtapose) is an open source peer-to-peer protocol specification. The JXTA protocols are independent of any programming language, and multiple implementations exist for different environments. The protocols can be implemented using any language, thus allowing heterogeneous devices to exist and communicate with one another in a huge peer-to-peer system.

5.2 Content Management Service (CMS)

Content Management Service is designed to allow peers to share their content and retrieve shared content by other peers. CMS libraries are used for sharing purpose. CMS specifies a protocol for searching and retrieving content. The purpose of the CMS is to keep an accurate record of all media files stored on the local peer that are eligible to be shared.

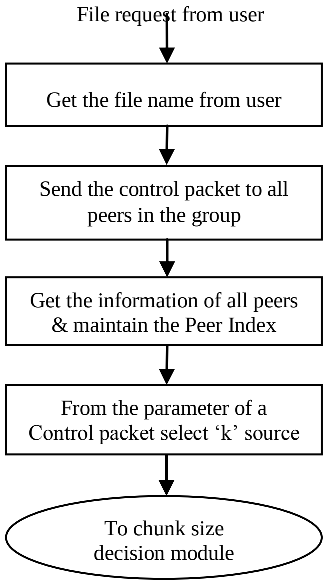


Fig. 5 Peer Index Diagram

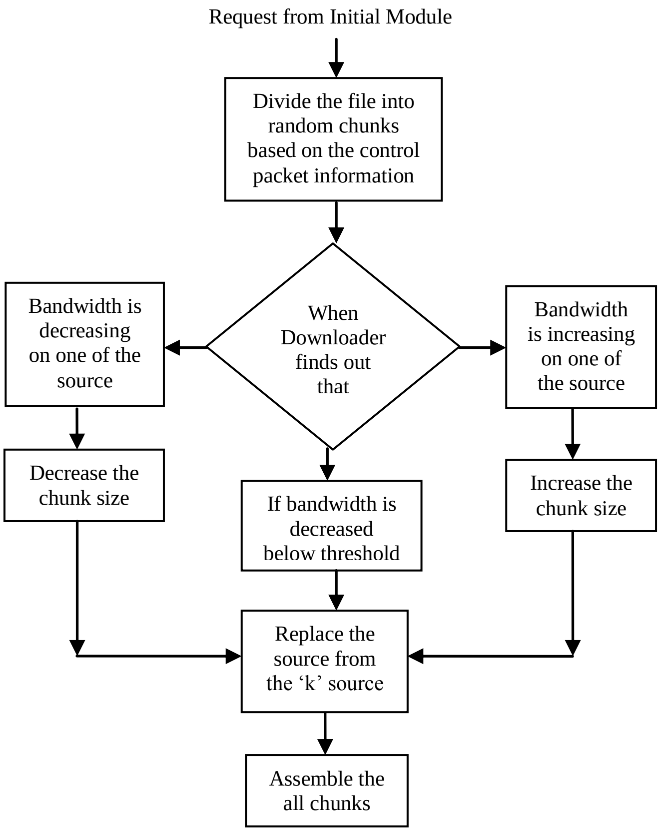


Fig.6 Chunk Size Decision Module

This protocol augments the basic jxta pipe functionality to allow a peer to download content from a peer and ensure that the content is retrieved in a reliable fashion.

6. SYSTEM ARCHITECTURE

Fig. 7 describes the Distributed Variable Chunk Based Switching in Peer to Peer Network. It allows to make peer to host itself on the network and publish the list of files for other peer to search and download.

When a peer initializes the service it tries to search a peer GroupID which is configured in it. If a peer doesn't find GroupID it declares himself as Server Peer. The peer who finds the GroupID they join the Server Peer and become Client Peer. With the support of CMS Interfaces and methods both client and server publish the data file content for searching.

Both can make searching of publish files providing an input file name and download that file by selecting the IP address of the searched peer.

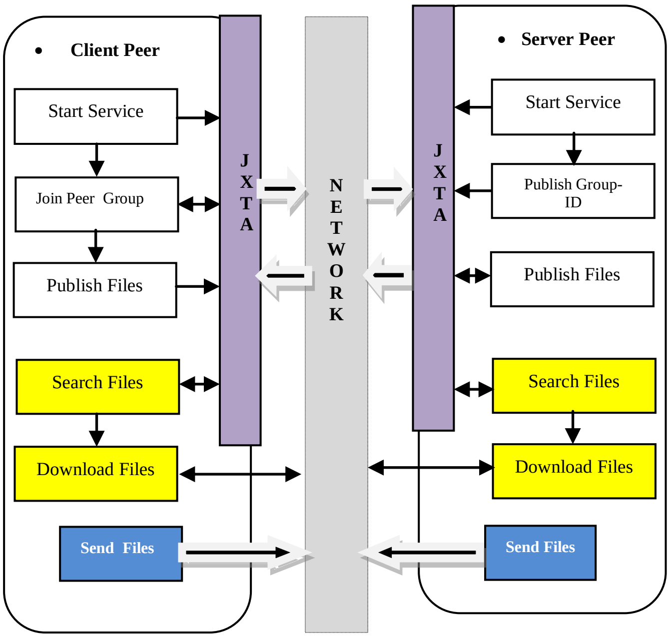


Fig. 7 System Architecture

6.1 System Modules

As per the Client and Server peer functions we categorized the following modules for implementing the Distributed Variable Chunk Based Switching in Peer to Peer Network:

A. Server Peer
B. Client Peer
C. File Publishing
D. Search File
E. Download Files
F. Send Data Files

6.1.1 Server Peer

This module implements server peer functions. It is responsible for publishing the GroupID and maintaining the client peer list. It receives the client join and leave request message and update peer list and distribute to all other peers in the network. It publishes it & data files for searching with support Content Management Service (CMS) interfaces and listeners. It can search and download a file published in multiple peers in group and also allow other peers to download its published data files.

6.1.2 Client Peer

This module implements client peer functions. It is responsible for searching the GroupID published by the server peer and send the join the request for joining. It receives the updated peer list distributed by the server peer. It publishes it & data files for searching with support Content Management Service (CMS) interfaces and listeners. It can search and download a file published in multiple peers in group and also allow other peers to download its published data files.

6.1.3File Publishing

This module implements CMS interface and methods for data file publishing. The Content Management Service (CMS) enables a peer to share or publish data—such as text documents, graphics files, sound files, and other media—with remote peers. To maintain consistency with the specification, the CMS relies on advertisements to provide information about the media or files. A peer is going to share or publish, and relies on JXTA pipes to transfer the content. The CMS is an excellent example of a service that has been built into the JXTA system. A peer can choose to use the service or not, depending on its desire to share content within a peer group. By using the CMS, the peer is relieved of the housekeeping details behind sharing the content and making it available for discovery and transfer.

6.1.4 Search Files

This module implements the published data file searching functions. A peer provides an input file name query for searching the published content. The query initiate a new *ListRequestor* object that passes the query to the current peer group and searches the file name string. *ListRequestor* is a derived class from *CachedListContentRequest*, which is defined in the CMS package.

6.1.5 Download File

This module implements the searched file downloading functions. A peer download a data file by selecting the IP address of the publishing peer obtained during search function. It established a socket connection with the select peer and send file download message. It buffers the data packets received and on completions stores the files. On failure of download completion it switches to next peer available in search peer list to continue the download. It repeats the process until the file download completes.

6.1.6 Send Data File

This module implements the distributed variable chunk based data sending. It initiates the process on receiving the file download message and file name from the downloader peer. It decides the data chunk size need to send based on the current bandwidth of the sender peer. If the bandwidth of the sender peer goes below the threshold bandwidth configured, it stops sending data packets.

7. IMPLEMENTATION

To experiment the proposed architecture, we implemented the modules which consists of two peers as Server Peer and Client Peer.

7.1 Server Peer

Server peer implements the functionality of a centralized server and publish a GroupID for clients for joining. It keeps & monitoring the clients who joins and leave the network periodically. It also implements the methods for data file publishing, searching, sending and downloading.

7.2 Client Peer

Client peer implements the functionality similar to the server peer but instead of publishing a GroupID they join the existing Group. It also implements the methods for data file publishing, searching, sending and downloading. To perform the described functions of peers some classes and methods are mentioned bellow

- First it executes a program on both server and client peer which initiates services of the peer like client service, JXTA service and CMS Services which implements a program for initialization of peer list. Server and Client Peer Listener are used for providing services which handles all the message communication between the peer and an algorithm is used for computing chunk data size for sending to the downloader. Methods are used to send the updated peer list periodically to the peers in the Group, update the peer list that are joining and leaving which is used only by server peer, to join a peer in Group by sending join message, to publish the share file data contents which uses Content Management Service (CMS) Library for sharing purposes, to launch CMS service, for creating and publishing new GroupID in case of failed to find any existing GroupID, for launch JXTA and search for group, for generating random bandwidth value which provides random bandwidth value for evaluation of the proposed system.
- It implements the methods to download the search file. It implements a method to start the downloading process by sending a "*FDW*" message to the sender peer. It implements the process to continue the download from other peers in case download failure from the current peer.
  Here chunk size is calculated as

  Buffer = *systembwd* * 300 bits

  As per above formula we get number of bits per bandwidth which are stored into buffer. *systembwd* is random bandwidth generated in this module. Also it is considered as for 1mp bandwidth 300 bits are send by the machine. Chunk size is stored into data packet. As system bandwidth changes, buffer size also changes. So we get variable chunk size as

  Data packet = buffer / 1024

  Here, bytes are converted into KB by dividing it by 1024. After sending bytes it checks the bandwidth again, changes the buffer size and data packet also.
- One configuration file is maintained which provides five parameters such as, Server Peer Address, Server Peer Listening Port, Bandwidth Threshold Limit for disconnection and Number of retry the sender need to perform in case of repetitively failure of the bandwidth limit failure. (i.e. number of retry). This configuration file is made available to all Client Peers.

8. EXPERIMENTAL SETUP

8.1 Installing JXTA

The primary web site for JXTA is *www.jxta.org*, where one can find all information on downloading an easy installation package.

Each of these installs includes

- The myJXTA application
- JXTA libraries
- The JXTA shell application

Most JXTA applications will require the libraries to compile and execute correctly. We need to place the binaries JAR directly in the lib directory of the installation.

8.2 Client – Server Network

To experiment the proposed architecture, local area network (LAN) is used where 'n' number of computers are connected in the network and implemented programming code is copied at all machines. Here, for communication purpose system has adopted port 6060 in between client – server network.

8.3 Peer to Peer Network

It consists of two peers as Server Peer and Client Peer. Any one machine is treated as server by providing input "1" at runtime and other machines are treated as client by providing input "2" at runtime in peer to peer network. First client – server network is established and then JXTA services run for that network then it becomes peer to peer network.

Server Peer in P2P network need to be initializing first and then Client Peers are initialized. All the peers are identified by their IP addresses.

8.4 Result and Output

Output on command prompt is as follow and Figure 8 show GUI of system

```
D:\>java CB.MainApp 1
<INFO  2013-08-31 14:42:41,265
NullConfigurator::<init>:146> JXTA_HOME = D:\.jxta
<INFO  2013-08-31 14:42:41,265
NullConfigurator::resetFromResource:361> D:\.jxta
\PlatformConfig already exists
<INFO  2013-08-31 14:42:41,546
NullConfigurator::adjustLog4JPriority:440> Log4J [user
default] requested, not adjusting logging priority
[===== Group Details =====]
jxta:Cred ( xml:space="preserve" xmlns:jxta="http://jxta.org"
type="jxta:NullCred" ) :
      PeerGroupID : urn:jxta:uuid-
4E0742B0E54F4D0ABAC6809BB82A311E02
      PeerID : urn:jxta:uuid-
59616261646162614A7874615032503302B2966E22764BA9
B408A784664A0FD903
      Identity : nobody

Waiting.........
Waiting.........
Downloading File : TC.rar

Connecting Peer :192.168.1.132
Peer Connected : Downloading.....
Waiting.........
Downloading Peer :192.168.1.132 Connection Failed

Connecting Peer :192.168.1.133
Peer Connected : Downloading.....
Peer Connected : Download Complete
```

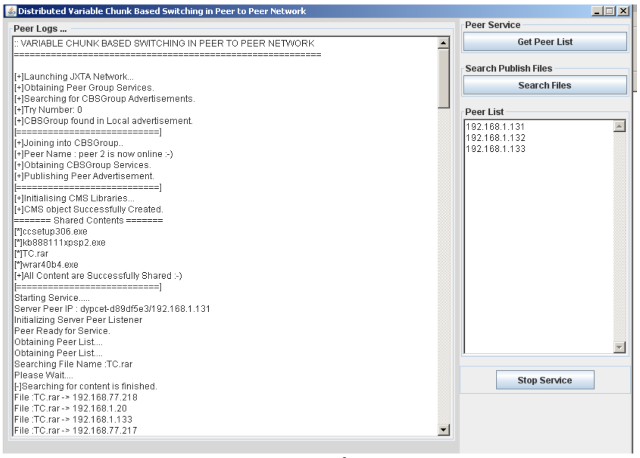


Fig. 8 GUI of System

9. RESULT ANALYSIS & OBSERVATION

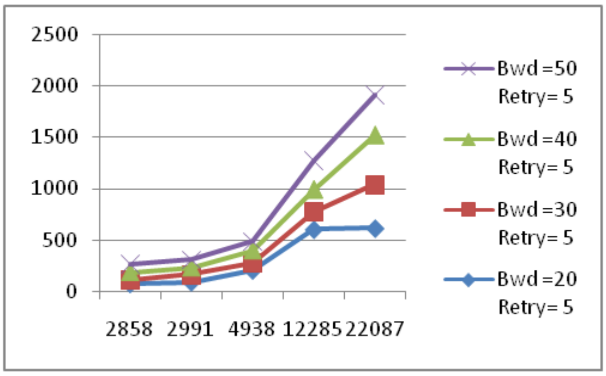
Following results were taken at various threshold values at its various number of retry at the time of file downloading. Here following tables and graphs contains file size in kilobytes (kb), time require to download a file is in seconds and number of up loaders of the file.

9.1 File Size Vs Bandwidth for maximum 5 number of up loaders

At the time of file downloading in the system, bandwidth threshold is kept from 20 kbps to 50 kbps and the number of retry in case of repetitively failure of the bandwidth limit is kept 5 and 10. Here, number of up loaders of any file is maximum 5. In this different situation, time require to download a file in seconds is calculated and it is shown in following Table 1, 2 and Graph 1,2.

File Size	Bwd =20	Bwd =30	Bwd =40	Bwd =50
2858	79	34	75	80
2991	95	71	67	79
4938	210	72	119	87
12285	605	173	214	286
22087	616	427	480	392

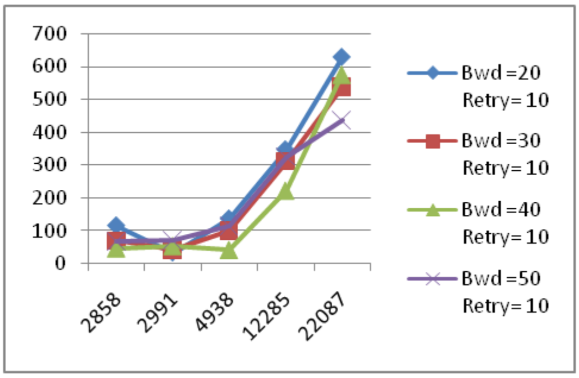
Table 1: Time required for File Size Vs Bandwidth for Retry value of bandwidth is 5



Graph 1: Time required for File Size Vs Bandwidth for Retry value of bandwidth is 5

File Size	Bwd =20	Bwd =30	Bwd =40	Bwd =50
2858	69	64	77	67
2991	64	67	74	65
4938	105	117	121	124
12285	336	301	322	334
22087	530	503	537	438

Table 2: Time required for File Size Vs Bandwidth for Retry value of bandwidth is 10



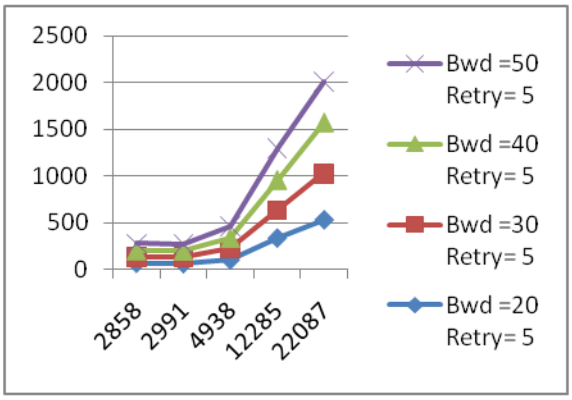
Graph 2: Time required for File Size Vs Bandwidth for Retry value of bandwidth is 10

9.2 File Size Vs Bandwidth for maximum 10 number of up loaders

At the time of file downloading in the system, bandwidth threshold is kept from 20 kbps to 50 kbps and the number of retry in case of repetitively failure of the bandwidth limit is kept 5. Here, number of up loaders of any file is maximum 5. In this different situation, time require to download a file in seconds is calculated and it is shown in following Table 3,4 and Graph 3,4.

File Size	Bwd =20	Bwd =30	Bwd =40	Bwd =50
2858	116	72	49	67
2991	34	44	56	72
4938	137	103	44	114
12285	347	312	224	325
22087	628	537	576	436

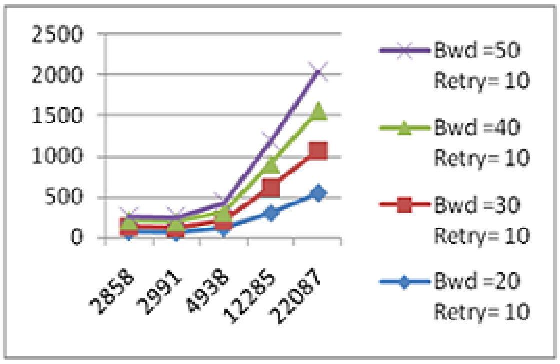
Table 3: Time required for File Size Vs Bandwidth for Retry value of bandwidth is 5



Graph 3: Time required for File Size Vs Bandwidth for Retry value of bandwidth is 5

File Size	Bwd =20	Bwd =30	Bwd =40	Bwd =50
2858	71	57	81	55
2991	54	61	78	57
4938	112	97	101	129
12285	297	317	288	289
22087	548	522	489	491

Table 4: Time required for File Size Vs Bandwidth for Retry value of bandwidth is 10



Graph 4: Time required for File Size Vs Bandwidth for Retry value of bandwidth is 10

9.3 Same Downloader Vs Different Up loaders

Here a single file is downloaded by a single downloader from different up loaders and time required to download that file in seconds is calculated. Obtained results are shown in following Table 5.

- Threshold Value for bandwidth = 40 kbps
- Number of Retry = 10
- File Size = 5553 kb

Time Require	Name of up loader
22	192.168.0.12
284	192.168.0.13
47	192.168.0.114
194	192.168.0.10
77	192.168.0.215
177	192.168.0. 116
40	192.168.0.118

Table 5: Time Vs Up loader

9.4 Different Downloader Vs Same Uploaders

Here a single file is downloaded from different downloaders but it is up loaded by single up loader and time required to download that file in seconds is calculated. Obtained results are shown in following Table.

- Threshold Value for bandwidth = 40 kbps
- Number of Retry = 10
- File Size = 4938 kb

Name of up loader	Time Require
192.168.0.12	37
192.168.0.13	37
192.168.0.114	38
192.168.0.215	37
192.168.0. 116	40
192.168.0.118	40

Table 6: Up loaders Vs Time

9.5 Observation

Following are the observations during the experimentation

1. The implemented system of Peer to Peer network is tried to carry out in Local Area Network for 20 to 30 computers at one time and there are no issues about location of the computers.
2. It is identified that, in file downloading process time require to download a file depends upon number of up loaders, i.e. as number of up loader increases, time require to download a file decreases
3. It is identified that, time require to download a file depend upon the up loader of that file and not on the downloader.
4. It is identified that time require to download a file depends upon traffic in the network, processing power of the computers, viruses in the network and distance between two computers.

10. CONCLUSION & FUTURE WORK

In this paper we have implemented our new scheme Distributed Variable Chunk Based Switching which removes Heterogeneity and Fluctuation also provide a distributed approach to a sequential Random Chunk Based Switching.

The implemented system can be carried out on wireless network as well as on internet also. The work can be extended for parallel downloading as well as random periodic switching algorithm. Processing power of computer, traffic between the networks, end to end delay can also change the chunk size of file in downloading process.

11. REFERENCES

[1] **"**Distributed Variable Chunk Based Switching in Peer to Peer network**",** by Shambhuraj Deshmukh and Amit Chougule in IEEE International Conference on Information and Computer Networks (ICICN 2011), January 26-28, 2011, Guiyang, China.

[2] "Minimizing File Download Time in Stochastic Peer-to-Peer Networks", by Yuh- Ming Chiu and Do Young Eun IEEE/ACM TRANSACTIONS ON NETWORKING, VOL. 16, NO. 2, APRIL 2008.

[3] "Dynamically Distributed Parallel Permanent Periodic Switching- D2PS for Minimizing File Download Time in Peer-to- Peer Networks", by M. Shymala devi, S. Pushpalatha in IEEE International Advance Computing Conference (IACC 2009), March 2009.

[4] D. Qiu and R. Srikant, "Modelling and Performance Analysis of Bit-torrent-like Peer-to-Peer Networks," in Proceedings of ACM Sigcomm, Aug. 2004.

[5] "Service Capacity of Peer to Peer Networks", by X. Yang and G. deVeciana in Proc. IEEE INFOCOM, Mar. 2004, pp. 2242–2252.

[6] "Measurement, Modeling, and Analysis of a Peer-to-Peer File Sharing Workload," by K. P. Gummadi, R. J. Dunn, and S. Saroiu in Proc. ACM Symp. Operating Systems Principles (SOSP), 2003.

[6] Jxta Programmers Guide_v2.5

[7] JXTA by Brendon J. Wilson

[8] Java Programming Language by James Gosling

[9] M. Adler, R. Kumar, K. Ross, D. Rubenstein, D. Turner, and D. D. Yao, "Optimal peer selection in a free-market peer-resource economy," in *Proc. Workshop on Economics of Peer-to-Peer Systems (P2PEcon)*, Cambridge, MA, Jun. 2004.

[10] M. Adler, R. Kumar, K. Ross, D. Rubenstein, T. Suel, and D. D. Yao, "Optimal peer selection for P2P downloading and streaming," in *Proc. IEEE INFOCOM*, Miami, FL, Mar. 2005, pp. 1538–1549.

[11] D. S. Bernstein, Z. Feng, and B. N. Levine, "Adaptive peer selection," in *Proc. Int.Workshop on Peer-to-Peer Systems (IPTPS)*, Berkeley, CA, Feb. 2003.

[12] S. G. M.Koo, K. Kannan, and C. S. G. Lee, "A genetic-algorithm-based neighbor-selection strategy for hybrid peer-to-peer networks," in *Proc.IEEE Int. Conf. Computer Communications and Networks (ICCCN 2004)*, Rosemont, IL, Oct. 2004, pp. 469–474.

[13] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, "A Scalable Content Addressable Network," Univ. California, Berkeley, Tech. Rep. TR-00-010, 2000.

[14] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for Internet applications," in *Proc. ACM SIGCOMM*, 2001.

[15] B. Y. Zhao, L. Huang, J. Stribling, S. C. Rhea, A. D. Joseph, and J. D. Kubiatowicz, "Tapestry: A resilient global-scale overlay for service deployment," *IEEE J. Sel. Areas Commun.*, vol. 22, no. 1, pp. 41–53,Jan. 2004.

[16] J. Byers, J. Considine, M. Mitzenmacher, and S. Rost, "Informed content delivery across adaptive overlay networks," in *Proc. ACM SIGCOMM*, 2002.

[17] C. Gkantsidis and P. R. Rodriguez, "Network coding for large scale content distribution," in *Proc. IEEE INFOCOM*, Miami, FL, Mar. 2005, pp. 2235–2245.

[18] "Peer-to-peer content distribution: Using client PC resources to store and distribute content in the enterprise" Intel Corp., Tech. Rep., Sep. 2003 [Online]. Available:http://www.intel.com/it/digital-enterprise/peer-peer-content- distribution.pdf

[19] K. K. Ramachandran and B. Sikdar, "An analytic framework for modeling peer to peer networks," in *Proc. IEEE INFOCOM*,Mar. 2005, pp. 215–269.