ISSN 2047-3338

Peer to Peer Networks: A Study Using Fragmentation Schemes

Mohd Sarfaraz Ahmed

Department of CSE, Nizam Institute of Engineering and Technology, Deshmukhi, Nalgonda, (A.P.), India

***Abstract*– In P2P file sharing, millions of peers participate in file-sharing community, with each one functioning as a server and contributing resources to the community. In this paper, we concentrate on the same data sharing in client to client system. The provider distributes the data over the P2P network wherein the users retrieve the data by using certain fragmentation schemes. The main thrust of this paper is that peers retrieve the data using fragmentation scheme according to their requirements which could effectively support in processing locality.**

***Index Terms*– P2P Networks, Client System and Fragmentation Schemes**

I. INTRODUCTION

THE general characteristic of client-server architecture is that it involves two processes (i.e., the client and the server) who play different roles with different capabilities and responsibilities. The client process plays the role of a service requestor and on the other hand the server process plays the role of a service provider, waiting passively for the arrival of request coming from the client and provides the desired service in response.

In P2P networks the client and the server play equal roles with equal capabilities and responsibilities, which means that the peer is both the web client as well as a transient web server. It is the web server because it is serving content within Http responses, it is transient because it is intermittently connected to the internet and may get a new IP address each time it reconnects to the internet. In other words, all the peers connected to the network must be capable of running both the client and server sides of file transfer protocol (FTP).

Locating data content is a major problem in P2P systems. To overcome this problem, many mechanisms were proposed in the literature [1], [2]. However these mechanisms were fruitful to some extent but resulted in many shortcomings for locating data. However in [3], the provider splits the data using a certain mathematical approach which involves range of data/files and storagespace. And then these files are compressed in an individual manner. But a point to be noted here is that all files may or may not be compressed.

Moreover, each tasks in [3] like (partitioning, compression, indexing), is the responsibility of provider peer which would require huge computation and storage capabilities, but this is not always what we want. In this paper, we propose a strategy of distributing the data (file types) to the peers connected onto the network. The idea behind this technique is to individually allow the participating peers to process locality of data by introducing fragmentation schemes using which peers locate specific portions of data.

The remaining part of this paper is organized as follows: Section II summarizes the survey of related work, section III describes the reference model architecture of distributed databases, section IV specifies the proposed fragmentation schemes, section V specifies the performance metrics and section VI concludes the paper.

II. RELATED WORK

If peer X is interested in obtaining a particular object, it is difficult to ascertain which peer has the object (file), it desires. One approach for locating content is maintaining a centralized directory as was done by Napster in [1].

A. Centralized Directory

Here, we make use of a large server or a server farm for providing the directory service. Whenever the user launches the P2P application it contacts the directory server and specifically the application running in the peer informs the directory server about its IP address and the number of objects it has in its local disks. In this way the directory server comes to know which peer has what objects to share.

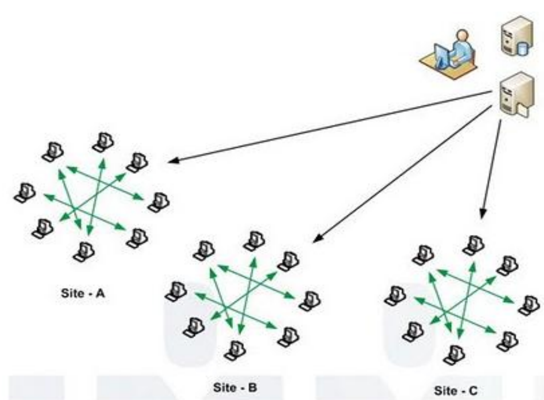

Fig. 1: P2P Networks

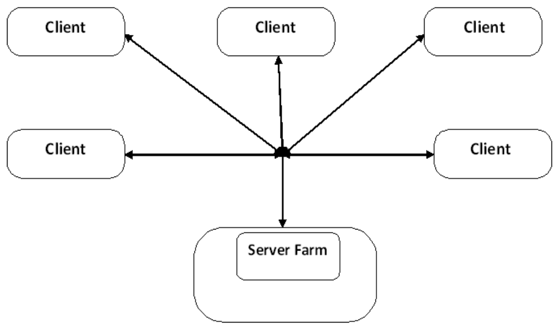

Fig. 2: Peers maintained by server farm

Therefore, the directory servers collects the information from each peer and thereby create a centralized dynamic database that maps each object name to a set of IP addresses.

Since peers connect/disconnect which is a non-trivial problem? To overcome with this scenario, one way to keep track on peers is to send messages periodically to peers to see if they respond. If the peer is no longer connected, it removes its IP address from the database.

Using centralized directory server leads to a number of drawbacks such as:

Single point of failure: If directory server crashes, the entire P2P network applications come to a halt. Even if redundant servers are used, internet connections to server farm may fail, causing the entire applications to crash.

Performance: One centralized server needs to respond to thousands of peers per second, therefore leading to traffic problems.

To overcome these problems, Gnutella network [2] was launched which locates the content completely opposite of that taken by Napster.

B. Gnutella network

Here the application does not use any centralized server for locating the data content. In this network, peers form an abstract logical network called as Overlay Network. As this network may have thousands of participating peers, let us see how these peers send messages in the overlay network for locating the data content.

When peer1 wants to download a song called “My India”, it sends a Gnutella Query message to all of his neighbors, these neighbors in turn forwards this message to all of their neighboring peers until the match is found. This process was referred to as Query Flooding. Once the keyword matches, it sends back to peer1 a Query hit message that contains a file name and file size.

Peer1 Gnutella process may receive many Query hit messages from responding peers. Gnutella sets up a TCP connection with a peer say some Peer20 based on the bandwidth and response time. As in such a huge network with a connection of more than a thousand of peers, request is transferred from peer to peer, thereby dumping a significant amount of traffic onto the internet. We refer to such type of a problem as limited scope query flooding.

To reduce the query traffic and to also reduce the number of peers to be queried, we set a peer count field to some specific limit (say 3 or 7). When the request is transferred from peer1 along with some peer count field, the request is transferred until the peer count value becomes zero. Though the peer count field an approach taken by Gnutella, it may happen that within the limited value of peer count field we may or may not get the desired file.

To overcome this problem, KazaA network [4] was launched, which takes the idea of both [1], [2] resulting in a more powerful P2P system.

C. KazaA network

To overcome such a scenario, KazaA architecture was developed wherein unlike Gnutella and Napster, not all peers are equal. The peers with a higher bandwidth and higher internet connectivity were considered as group leaders and the other peers with less bandwidth were considered as ordinary peers. Therefore, the ordinary peers will be connected to one group leader. This way the group leader maintains the IP addresses of all ordinary peers as well as the objects that these peers make available for sharing thereby creating a database. This way KazaA employs a number of techniques that improves its performance such as:

Request quering: If peer X limits his peer to 4 uploads and his peer is already uploading 4 files when he receives a new upload request from peer Y, he puts peer Y’s request in a local queue. This ensures that each file transferred receives a small amount of bandwidth from sending peers.

Incentives priorities: Here peer X will give priority to those peers, who have in the past uploaded more files than they have downloaded.

Supposing that a user has launched his P2P application and he has some data which he likes to share on to the P2P network. In paper [3], authors proposed a strategy that when peer X wants to share some data set say D, data could be in the form of text, pictures, graphics, video files, software, documents, etc., so that other peers could impose range queries.

In paper [3], in order to make its data to be suitably distributed across the network, peer X (service provider) builds the synopsis of D by first partitioning D using the concept of Clustering technique and then compressing each portion of data in the partition. Peer X also builds a synopsis which consists of index and sub synopsis. These index and sub synopsis are distributed across the network. Therefore peers make access to data by using the index structures.

In the proposed strategy, we thought of bringing a new approach of partitioning the data using fragmentation schemes as a concept used in traditional distributed databases. Here we introduce a reference architecture [5] comprising of Global Schema, Fragmentation Schema, and allocation schema (i.e., allocating file contents to requesting peers).

III. REFERENCE ARCHITECTURE

The following section specifies the reference model for distributed database structures. It consists of three components Global Schema (GS), Fragmentation Schema (FS) and Allocation Schema (AS). The components of this reference architecture are described below:

A. Global schema

This component is present at the top level of the architecture representing the data at provider side. Hence in P2P Networks, peers retrieve the data by posing range queries. Here in this section, we make use of a relational model. Using this model, we define Global Schema to be consisting of a set of Global relations.

B. Fragmentation Schema

This component is present at the second level of the architecture. In this the data is partitioned into multiple segments known as fragments. This division is done by using several techniques such as Horizontal Fragmentation or Vertical Fragmentation. The Fragmentation schema contains one to many relationship in which several fragments belong to one global relation but only one global relation belong to one fragment. Here GR_x specifies the fragment of Global Relation GR along with xth fragment.

C. Allocation Schema

This schema is present at the third level of the architecture. This schema is responsible for specifying the location of a particular fragment. A Global relation name along with an index can be used for representing the physical images (locations). For instance, GR^y specifies the physical image of Global relation GR which is located at y.

IV. FRAGMENTATION STRATEGIES

In the proposed strategy, the provider peer distributes the data/file types to the requesting peers based on their IP address and port numbers. The peers in turn retrieve the data using certain fragmentation scheme. Here we use the same approach as discussed in [5].

A. Horizontal Fragmentation

Horizontal fragmentation refers to a fragmentation in which every fragment is specified in terms of a selection operation performed on a global relation. Consider an example represented as:

FT (FNAME, FSIZE, FSTATUS, DOWNLOADSPEED)

From the above relation, FT represents file type is a table name and attributes represent the columns. From the above relation, the horizontal fragmentation can be defined as,

$$FT_1 = SL_{FNAME=\text{"TEXTFILE"}}\ FT$$
$$FT_2 = SL_{FNAME=\text{"IMAGEFILE"}}\ FT$$

If the two values "TEXTFILE" and "IMAGEFILE" belong to the name attribute, then completeness criteria of fragmentation is satisfied.

$$FT = FT_1\ UN\ FT_2$$

In the above FT relation, FT global relation is reconstructed using "UN" operation. Thus the reconstruction criterion of fragmentation is satisfied. Also the disjointness criterion is satisfied because FT_1 and FT_2 are two disjoint fragments.

In the selection operation, a predicate is used which helps in defining a fragment referred to as its "qualification". From the example, the qualifications are defined as:

$$qual_1 = FNAME = \text{"TEXTFILE"}$$
$$qual_2 = FNAME = \text{"IMAGEFILE"}$$

To satisfy the completeness criteria, it must be ensured that the group of qualifications belonging to every fragment is complete. If they are not complete with respect to all fragments, then they must be complete at least with respect to the group of allowed values. To satisfy the reconstruction criteria, union operation is used. Finally to satisfy the disjointness condition, the two qualifications $qual_1$ and $qual_2$ must be mutually exclusive i.e., they must be disjoint.

B. Vertical Fragmentation

In vertical fragmentation, the attributes of a global relation are subdivided in the form of groups. The global relation is projected over each group so as to obtain the fragments. Here we use this type of fragmentation because the data containing common geographical properties are stored in each group of attribute. A rule states that if every attribute is mapped to one or more attributes of fragments then vertical fragmentation is valid. Consider an example, represented as:

FT (FNAME, FSIZE, FSTATUS,
DOWNLOADSPEED)

Using the above relation, a vertical fragmentation can be defined as:

$$FT_1 = PJ_{FNAME,\ FSIZE,\ FSTATUS}\ FT$$
$$FT_2 = PJ_{FNAME,\ FSTATUS}\ FT$$

Since FNAME is a unique key of FT relation, the relation FT can be reconstructed as:

$$FT = FT_1\ JN_{FNAME=FNAME}\ FT_2$$

Thus the reconstruction is done using the join operation. But a point to be noted is that this reconstruction property is not satisfied, since after performing the join operation, the FT relation comprises of a column that contains replicated value of FNAME. This can be avoided by using a projection technique by omitting one column of FNAME.

In the above fragmentation, both FT_1 and FT_2 relations contain similar attribute FNAME. To avoid this replication, we apply reconstruction with the help of another projection operation as:

$$FT = FT_1\ JN_{FNAME=FNAME}\ PJ_{FNAME,\ FSTATUS}\ FT_2$$

C. Mixed Fragmentation

A mixed fragmentation is a combination of both horizontal fragmentation and vertical fragmentation. This type of fragmentation can be obtained by applying horizontal fragmentation, only after applying vertical fragmentation of relation FT as:

$$FT_1 = PJ_{FNAME,\ FSIZE,\ FSTATUS,\ DOWNLOADSPEED}\ FT$$
$$FT_2 = PJ_{FNAME,\ FSTATUS,\ LABEL,\ ADDEDON}\ FT$$

The mixed fragmentation for the given relation is:

$$FT_1 = SL_{FSIZE<=1000MB}\ PJ_{FNAME,\ FSIZE,\ FSTATUS,\ DOWNLOADSPEED}\ FT$$

$$FT_2 = SL_{802< FSIZE <= 1000}\ PJ_{FNAME,\ FSTATUS,\ LABEL,\ ADDEDON}\ FT$$

$$FT_3 = PJ_{FNAME,\ FSIZE,\ FSTATUS,\ DOWNLOADSPEED}\ FT$$

Using the join operation, the global relation FT can be reconstructed as:

$$FT = UN\ (FT_1, FT_2)\ JN_{FNAME=FNAME}\ PJ_{FNAME,\ FSTATUS,\ FSIZE}\ FT_3$$

In this way provider peers distribute the data over the network, whereas receiving peers retrieve the data by specifying an appropriate fragmentation scheme according to his/her requirement. As of paper [3], the authors proposed a strategy of partitioning the data, compressing each portion of data in the partition, and then scattering the data over to the network which could provide an overhead on the provider. In this paper, we tried to bring a simple approach which provides a clear understanding in the below section that follows:

Table I: List of File Types provided by Distributor (FTypes)

Fname	Fsize	Fstatus	Download speed	Date	Time
Text Files	4 Mb	On	11.9 kb/s	3-8-2011	2:17:00
Java Files	120 Mb	Off	21.4 kb/s	3-8-2011	3:06:01
Image Files	8 Mb	Downloading	14.3 kb/s	3-8-2011	1:18:00
Document File	220 Mb	On	12.6 kb/s	3-8-2011	4:14:09
. Class Files	8 Mb	Downloading	12.9 kb/s	3-8-2011	2:17:02
Video Files	802 Mb	Downloading	22.8 kb/s	3-8-2011	2:17:00

V. PERFORMANCE METRICS

The Analytical approach for achieving the Fragmentation for different file types is discussed. The system is finally tested using 3 different types of fragmentation schemes, which provides efficient retrieval of data by the receiving peers, thereby processing locality.

Table I describes the different file types given by the provider to the requesting peers. The receiving peers on the other hand retrieve the desired files by posing an appropriate fragmentation scheme as follows:

A. Horizontal Fragmentation

Select * from FTypes where Fname="TEXTFILES";
Or
$$FTypes_1 = SL_{FNAME="TEXTFILE"}\ FTypes$$

Table II: Retrieving Text Files using Horizontal Fragmentation

Fname	Fsize	Fstatus	Download Speed	Date	Time
Text Files	4 Mb	On	11.9 kb/s	3-8-2011	2:17:00

In this way data of all columns could be retrieved using Horizontal fragmentation with select operation as in Table II.

B. Vertical Fragmentation

Select Fname, Fstatus from FTypes;
Or
$$FTypes = PJ_{Fname,\ Fstatus}\ FTypes$$

Table III: Retrieving Text Files using Vertical Fragmentation

Fname	Fstatus
Text Files	On
Java Files	Off
Image Files	Downloading
Document Files	On
. Class Files	Downloading
Video Files	Downloading

In this way data of specific or required columns can be retrieved by performing a projection operation on columns as seen in Table III.

C. Mixed Fragmentation

Select Fname, Fsize from FTypes where DownloadSpeed<14.0;
Or
$$FTypes = SL_{downloadspeed < 14.0}\ PJ_{Fname,\ Fsize}\ FTypes$$

Table IV: Retrieving File Types using Mixed Fragmentation

Fname	Fsize
Text Files	4 Mb
Document Files	220 Mb
. Class Files	8 Mb

Table IV depicts mixed fragmentation which can be achieved using both horizontal and vertical fragmentation.

In this way, data is distributed by the provider peer among the network. This type of data distribution among peers is highly applicable only for some confidential networks. This paper in some way resembles [1], wherein a centralized directory was proposed by Napster, which helps in locating the content easily. But in this paper, we distribute the data unevenly onto the P2P network. Peers locate the data by using an appropriate fragmentation schemes depending on his application requirement which helps in processing locality.

The P2P system in this regard also maintains a high degree of availability by storing multiple copies of same data, in this way peers can switch to an alternative copy when the one that is to be accessed under normal conditions is unavailable.

Reliability can also be achieved by distributing the same data over different peers. This helps to recover the data from remaining peers when the current peer suffers from crashes or data destruction.

VI. CONCLUSION

Data distribution in the provider side is concentrated. Data is scattered onto the network, whereby the data is partitioned by using fragmentation schemes as discussed in this paper which helps in processing locality. Compression is also a major hindrance which can be overcome by using an appropriate compression scheme at peer side. This paper forms a major factor for corporations on a private and confidential network, thereby providing fast and accurate answers, which helps in improving efficiency in peer-to-peer systems.

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompanies the successful completion of any task would be incomplete without the mention of the people who made it possible and whose encouragement and guidance have crowned my efforts with success. I extend my deep sense of gratitude to Principal *Dr. D. V. Ravi Shankar*, Nizam Institute of Engineering and Technology, Deshmukhi. I am indebted to *Dr. S. M. Afroz*, Professor and HOD, Computer Science and Engineering, Nizam Institute of Engineering and Technology, Deshmukhi, for his support and encouragement. I am indebted to *Mr. M. S Qaseem*, HOD, Information Technology, Nizam Institute of Engineering and Technology, Deshmukhi, for his constant support and guidance throughout my paper. I thank *Dr. Salman Abdul Moiz*, Research Scientist at Center for Development of Advanced Computing (CDAC), Bangalore, India, for giving his helpful comments and sharing his ideas in carrying out this research work. Finally, I express my deep sense of gratitude to one and all who have helped me in successfully completing this task. Furthermore I would like to thank my family specially my mom *Mrs. Noorjahan* and my friends for their moral support and encouragement.

REFERENCES

[1] Computer Networking "A Top-Down Approach Featuring the Internet" Third Edition by James F. Kurose & Keith W. Rose, http://www.Napster.com.

[2] Computer Networking "A Top-Down Approach Featuring the Internet" Fourth Edition by James F. Kurose & Keith W. Rose, http://www.Gnutella.com.

[3] "Managing Multidimensional Historical Aggregate Data in Unstructured P2P Networks" by Filipo Fur faro, Giuseppe Massimiliano Mazzeo and Andrea Pugliese, IEEE Transactions on Knowledge and Engineeering, Vol. 22, 2010.

[4] Computer Networking "A Top-Down Approach Featuring the Internet" Third Edition by James F. Kurose & Keith W. Rose, KazaA Networks, http://www.KazaA.com

[5] Distributed Databases, "Principles & Systems" by Stefano Ceri and Giuseppe Pelagatti.

Mohd Sarfaraz Ahmed has received his Bachelors Degree B.Tech in Information Technology from Syed Hashim College of Science and Technology, Affiliated to JNTU, Hyderabad and M.Tech in Software Engineering from Nizam Institute of Engineering and Technology, Deshmukhi, Nalgonda Dist, Affiliated to JNTU Hyderabad, AP India. Presently he is working as an Asst. Professor in the Department of Computer Science Engineering, Nizam Institute Of Engineering and Technology, Deshmukhi. His area of interests includes Web Technologies, Artificial Intelligence, Software Requirements and Estimation, DataMining, Peer-to-Peer Networks (Email: mohdsarfarazahmed@yahoo.com).