

# Trust Management in Partially Decentralized Peer To Peer File Sharing System

Shweta Gupta, Sunita Tiwari

**Abstract:-** Trust management for peers in P2P network is very much important due to its open and anonymous nature. In this paper, we propose a reputation based trust management scheme for partially decentralized peer-to-peer file sharing systems. Reputation data is stored at the super node level. We calculate the reputation value for each file by using fuzzy logic inferences, which can better handle uncertainty, fuzziness, and incomplete information in peer trust reports. We have considered three factors namely: quality, popularity and size of the file shared by a peer and simulated the resulting trust management system.

**Keywords:-** P2P file sharing system, Trust Management, Reputation, Partially Decentralized Systems, Fuzzy logic.

## I. INTRODUCTION

A peer-to-peer (P2P) network is a computer network that does not have fixed clients and servers but a number of peer nodes that function as both clients and servers to the other nodes in the network.

The open and anonymous nature of a P2P network makes it an ideal medium for attackers to spread malicious content. So there should be some techniques present which will let the peers know about the trustworthiness of the shared files and the peer who is sharing the file. So a reputation based trust management system is required, which will store the reputation based on some input factors and the output will be the reputation of the shared file by that peer. In a Peer-to-Peer (P2P) file sharing system, peers communicate directly with each other to exchange information and share files. P2P systems can be divided into several categories:

**Centralized P2P systems:** First generation P2P (e.g. Napster [1] [2]) utilizes the server-client network structure. The central server maintains directories of shared files. Each time a client logs on or off the network, the directory is updated. It provides the highest performance when it comes to locating files. Every individual peer in the network must be registered, which ensures that all searches are comprehensive and execute quickly and efficiently. These systems suffer from a single point of failure, scalability, and censorship problems.

**Completely decentralized P2P systems:** Second generation P2P (e.g. Gnutella protocol [3]) uses a distributed model where there is no central server and every node has equal status. A decentralized framework does not rely on a central server, and is therefore more robust than its centralized counterpart. Disadvantages to the decentralized model appear in the form of prolonged search times. An outgoing search request may need to travel through thousands of users before any results are identified.

**Partially decentralized P2P systems:** Third generation P2P (e.g., KaZaA[4], Morpheus[5] and Gnutella2[6]) employs a hybrid of the central-server and fully decentralized frameworks. In this peers can have different roles. Some peers act as local central indexes for files shared by local peers. These special peers are called “supernodes” or “superpeers”.

Partially decentralized P2P systems have been proposed to reduce the control overhead needed to run the P2P system. They also provide lower discovery time because the discovery process involves only superpeers (figure 1).

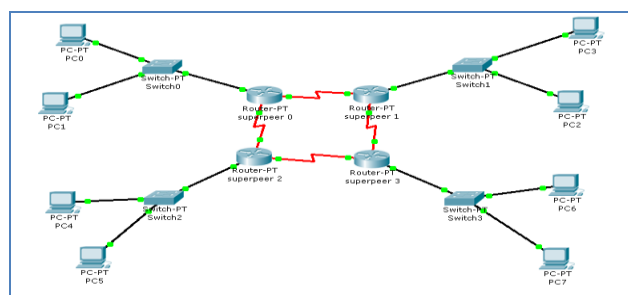


Figure 1: Topology in a partially decentralized P2P system

## II. RELATED WORK

### 1. Reputation in Centralized P2P Systems

eBay [7] uses the feedback profile for rating their members and establishing the members' reputation. Members rate their trading partners with a Positive, Negative or Neutral feedback, and explain briefly why. The reputation is calculated by assigning 1 point for each positive comment, 0 points for each neutral comment and -1 point for each negative comment.

The eBay feedback system suffers from the following issues:

- Single point of failure.
- It is not an easy task to look at all feedbacks for a member in the way the feedback profile is presented on eBay.
- A seller may have a good reputation by satisfying many small transactions even if he did not satisfy a transaction with a higher amount.

### 2. Reputation in Completely Decentralized Systems

In [8], the EigenTrust algorithm assigns to each peer in the system a global trust value. This value is based on its past uploads and reflects the experiences of all peers with the peer. This scheme is a reactive one; it requires reputation to be computed on demand which requires cooperation from a large number of peers in performing computations. As this is performed for each peer having the requested file with the cooperation of all other peers, this will introduce additional latency as it requires long periods of time to collect statistics and compute a global rating. Most of proposed reputation management schemes for completely decentralized P2P systems are reactive and hence suffer from this drawback.

### 3. Reputation in Partially Decentralized Systems

KaZaA Media Desktop (KMD) [4] uses Integrity Rating Files for rating files and Participation Level for rating peers. Users can rate the files they share based on technical quality and accuracy of their descriptive file data. A file can be given the following ratings: excellent, average, poor, or delete file. In KaZaA, a Participation Level is assigned to each user in the network based on the files that are uploaded from the user and the files the user downloads from other users. The participation Level is:  $(\text{Uploads in MB} / \text{Downloads in MB}) * 100$ . In KaZaA, malicious peers can still upload inauthentic files. This will reduce the peers' satisfaction and waste network resources.

### 4. Trust Management by using Fuzzy Logic

Damiani et al. propose XRep [9], a protocol for trust management in decentralized P2P systems. In XRep peers query about the reputation of resources or peers by using network broadcasts to retrieve votes from neighboring peers.

Cornelli et al. present a trust system, called P2Prep [10], where peers poll the network for reputation opinions on service providers.

XRep [11] proposed by Curtis, Safavi-Naini, Susilo, extends XRep and is designed to protect against the weaknesses of XRep. As in XRep, a distributed polling algorithm is employed. In X2Rep to manage the reputations based on resources and servants. However, it is an improvement on XRep and has the ability to compute the weight of a peer based on past voting experiences.

Many reputation schemes have been proposed in P2P networks. We propose a fuzzy inference system to design P2P reputation based trust management system for partially decentralized system. It generates the reputation values for each shared files by interacting with other peers. The inference system takes the fuzzy voted values from the peers and calculates the reputation for file. Fuzzy logic is effective for calculation of reputation as many factors affecting the reputation are uncertain and has fuzziness. We have taken three factors namely quality, popularity and size of the file share by a peer and simulate the resulting trust management system.

## III. PROPOSED WORK

### A. Fuzzy Model for the Calculation of Reputation

The reputation of a file is decided by three factors: quality, popularity and size of the file shared by a peer. *Quality* of the file shared is the important factor which makes the user to decide whether to download the file or not. Good quality files got good reputation and bad or fake files got very low or no reputation. Second factor, *popularity* of the file also plays an important role in deciding reputation of a file. If a file is good quality, the demand of the file increases in the internet. As a file is more popular it is widely available in the internet. So the third factor, *size* of the shared file decides which file to download. If the file quality is good and the size of one file is less than the size of other file. Then peers download the less size file to save their download bandwidth and time. So all the factors depend upon each other and affect the reputation of the shared file.

Each peer records its own experiences on shared files in its local repository at connected superpeers. A *Shared Repository*, stores a value which describes whether the shared file that the peer has experienced is good or poor.

Reputation updating is done in a regular timely manner or on demand by other peers by following *Update\_Repository ()* algorithm by the tracker. Tracker takes three inputs from peers and the reputation is calculated by the fuzzy inference system, and then stored in the repository.

**Update\_Repository ()**

If (Peer is verified)

- Extract the three inputs Quality, Popularity and Size of the file;
- Input them to fuzzy inference system;
- Collect the output;
- Store in the shared repository;

Else

Update failure;

Retry ();

Rule base for fuzzy system for the calculation of file reputation is shown in figure 2. An example rule in our fuzzy system can look like the following:

*If (Quality == Medium and Popularity == High and Size == Small) then Reputation = Medium*

<b>Quality of the file</b>				
<b>Popularity</b>	LOW      MEDIUM      HIGH			
	LOW	VERY LOW	VERY LOW	LOW
	MEDIUM	VERY LOW	LOW	MEDIUM
	HIGH	LOW	MEDIUM	HIGH
<b>Small Size file</b>				
<b>Quality of the file</b>				
<b>Popularity</b>	LOW      MEDIUM      HIGH			
	LOW	VERY LOW	LOW	MEDIUM
	MEDIUM	LOW	MEDIUM	HIGH
	HIGH	MEDIUM	HIGH	VERY HIGH
<b>Medium Size file</b>				
<b>Quality of the file</b>				
<b>Popularity</b>	LOW      MEDIUM      HIGH			
	LOW	LOW	MEDIUM	HIGH
	MEDIUM	MEDIUM	HIGH	VERY HIGH
	HIGH	HIGH	VERY HIGH	VERY HIGH
<b>Large Size file</b>				

**Figure 2:** Rule Base for Fuzzy System

*Partially Decentralized Peer to Peer file sharing system*

A partially decentralized P2P system has been proposed to reduce the control overhead needed to run the P2P system. They also provide lower discovery time because the discovery process involves only superpeers. In addition to being used for control message exchange, a superpeer will also be used to store reputation data for peers it serves, update this data, and provide it to other superpeers. Since each peer interacts only with one superpeer at a time, the proposed approach achieves more accuracy and reduces message overhead significantly.

**IV. IMPLEMENTATION**

Fuzzy inference system is implemented in **Fuzzy Logic Toolbox Graphical User Interface Tool of Matlab**. In Matlab, create rules using the Rule Editor. Figure: 3 and 4 is a sketch map of Mamdani-type Fuzzy Inference. The Rule Viewer displays a roadmap of the whole fuzzy inference process.

Figure: 5 and 6 gives a 3D simulative sketch map of P2P Fuzzy Inference reputation. By inputting arbitrary Quality, Popularity and Size of the shared file, the figure will map to a fuzzy output value Reputation.

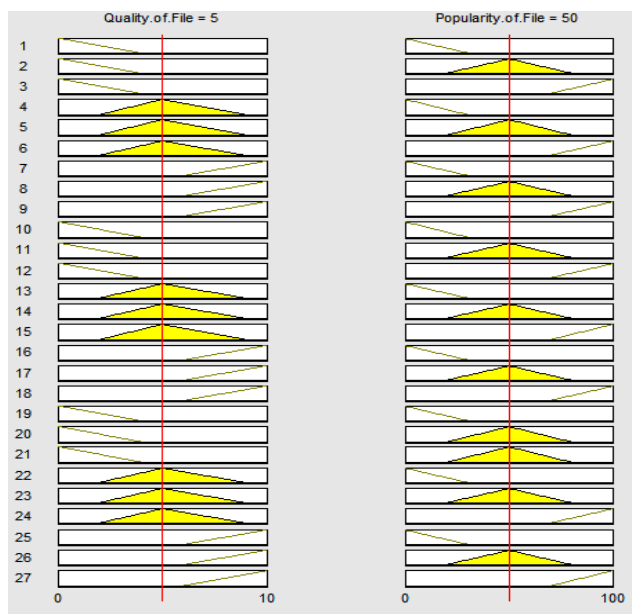


Figure 3: A sketch map of Mamdani-type Fuzzy Inference

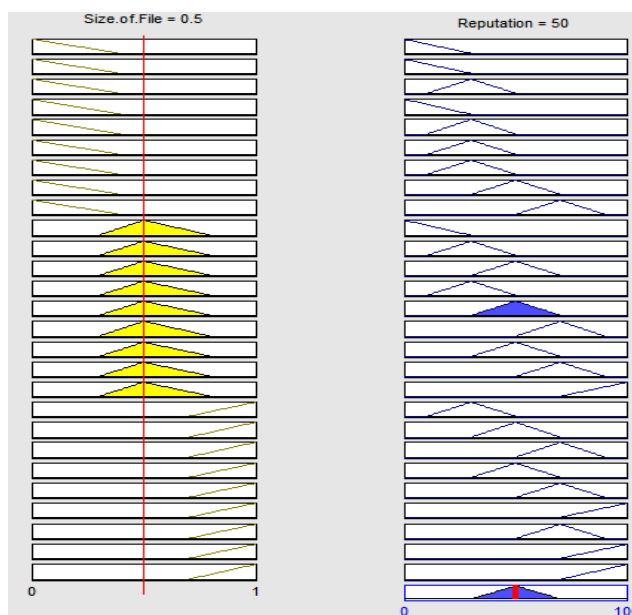


Figure 4: A sketch map of Mamdani-type Fuzzy Inference

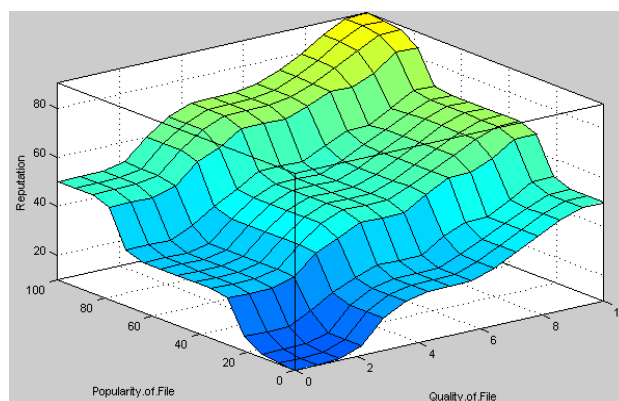
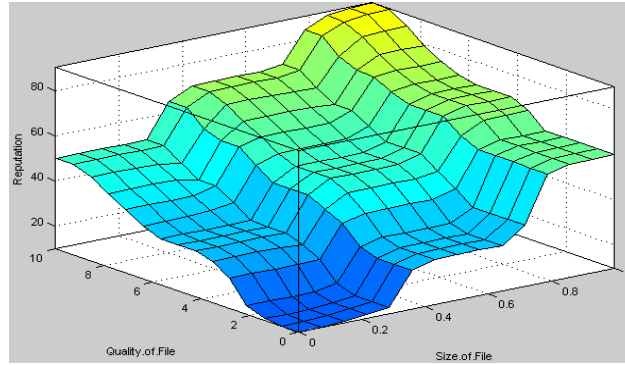


Figure 5: A 3D simulative sketch map of P2P Fuzzy Inference Reputation plotting Upload bandwidth Vs. No. of files shared



**Figure 6:** A 3D simulative sketch map of P2P Fuzzy Inference Reputation plotting seeding time Vs. No. of files shared

## V. SIMULATION AND PERFORMANCE ANALYSIS

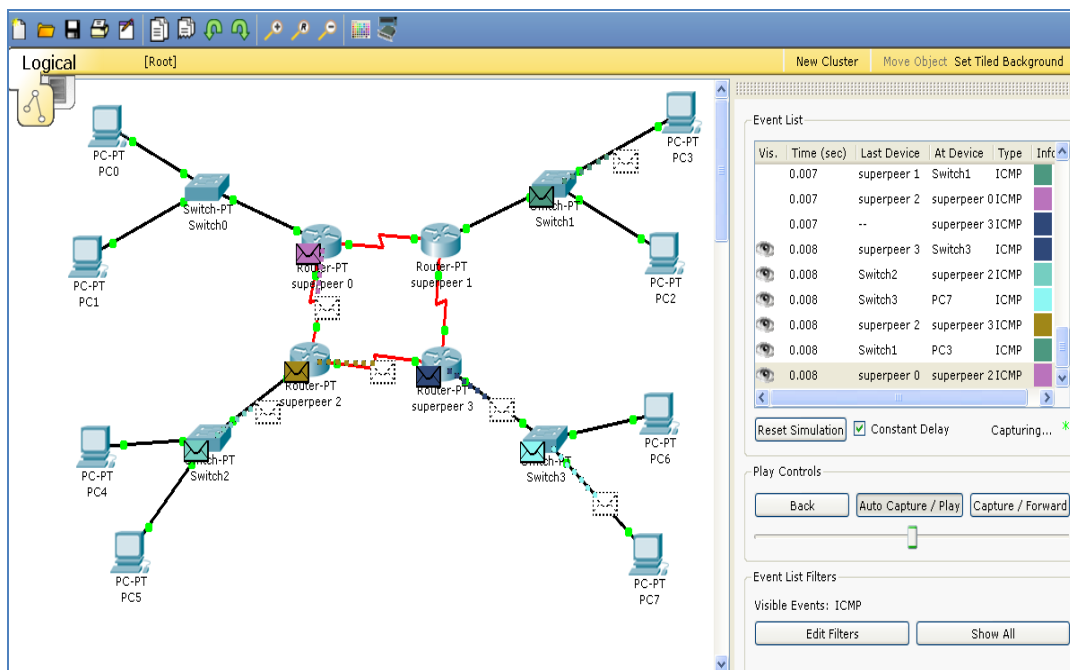
The **Cisco Packet Tracer 5.1** was used for simulation of partially decentralized P2P system. A reputation value is calculated by using fuzzy inference system. This value is stored at Superpeers.

### A. Simulation Mode

In *Simulation Mode*, you can observe the paths that packets take and inspecting them in detail. The *Event List* window records what happens as your PDU propagates through the network. An event can be defined as any instance of a PDU that is generated in the network. The Event List keeps track of all such PDU instances. It is shown in Figure 7.

### B. Simulation Parameter

- We simulate a system with 12 peers and 4 super peers.
- No. of PDU packets (file) are 10.
- File sizes are uniformly distributed between 10MB and 150MB.

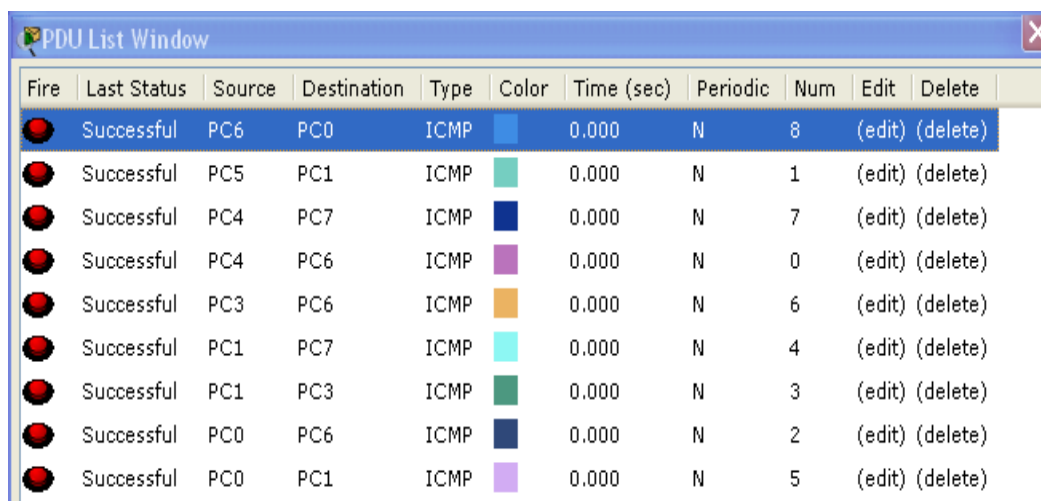


**Figure 7:** Simulation mode and Event list

### C. Simulation Result

Our Proposed method is assisting peers in selecting the most trustworthy peer to download file. The **PDU (Protocol Data Units) List** tracks all of the PDUs you created for the current scenario. Each PDU in the list has the following fields: Fire, Last Status, Source, Destination, Type, Color, Time, Periodic, Num, Edit, and Delete (Figure 8). The PDU List shows the successful transfer of file from source to destination and Num field in PDU list shows number of hops or count, a file travel to reach the destination. Our reputation management

scheme is simple, proactive and has minimal overhead in terms of computation, infrastructure, storage and message complexity.



Fire	Last Status	Source	Destination	Type	Color	Time (sec)	Periodic	Num	Edit	Delete
	Successful	PC6	PC0	ICMP		0.000	N	8	(edit)	(delete)
	Successful	PC5	PC1	ICMP		0.000	N	1	(edit)	(delete)
	Successful	PC4	PC7	ICMP		0.000	N	7	(edit)	(delete)
	Successful	PC4	PC6	ICMP		0.000	N	0	(edit)	(delete)
	Successful	PC3	PC6	ICMP		0.000	N	6	(edit)	(delete)
	Successful	PC1	PC7	ICMP		0.000	N	4	(edit)	(delete)
	Successful	PC1	PC3	ICMP		0.000	N	3	(edit)	(delete)
	Successful	PC0	PC6	ICMP		0.000	N	2	(edit)	(delete)
	Successful	PC0	PC1	ICMP		0.000	N	5	(edit)	(delete)

Figure 8: PDU List Window

## VI. CONCLUSION

In this paper, we propose a fuzzy inference system to design reputation management scheme for partially decentralized peer to peer systems. Reputation data is stored at the superpeer level to take advantage of the use of superpeer in partially decentralized P2P systems. It generates the reputation values for each shared files by interacting with other peers. The inference system takes the fuzzy values of quality, popularity and size of the shared file from the peers and calculates the reputation for file. Fuzzy logic is effective for calculation of reputation as many factors affecting the reputation are uncertain and has fuzziness.

## REFERENCES

- [1]. A.Oram, Peer-to-Peer: Harnessing the Power of Disruptive Technologies, O'Reilly Books, 2001, Ch. 2, pp. 21-37.
- [2]. Napster, <http://www.napster.com/>.
- [3]. Gnutella Protocol Specification v0.4, <http://www9.limewire.com/>.
- [4]. KaZaA, <http://www.kazaa.com/>.
- [5]. Morpheus, <http://www.morpheus.com/>.
- [6]. Gnutella2 Specialization, <http://www.gnutella2.com/>.
- [7]. EBay, <http://www.ebay.com/>.
- [8]. S. D. Kamvar, M. T. Schlosser, H. Garcia-Molina, The EigenTrust Algorithm for Reputation Management in P2P Networks, in: The 12th International World Wide Web Conference, Budapest, Hungary, 2003.
- [9]. E. Damiani, S. De Capitani di Vimercati, S. Paraboschi, P. Samarati and F. Violante, A reputation-based approach for choosing reliable resources in peer-to-peer networks. Proceedings of the 9th ACM conference on Computer and Communications Security, pages 207–216, November 2002.
- [10]. F. Cornelli, E. Damiani, S.D. di Vimercati, S. Paraboschi, P. Samarati Choosing reputable servents in a P2P network, in: Proceedings of the 11<sup>th</sup> International Conference on World Wide Web, Honolulu, 2002.
- [11]. Nathan Curtis, R. Safavi-Naini, and W. Susilo, X2Rep: Enhanced trust semantics for the XREP protocol. Applied Cryptography and Network Security. Second International Conference, ACNS2004. Proceedings. LNCS3089, pages 205–219, 2004.

**Shweta Gupta**, M.tech Student, Computer Science Department, Krishna Engineering College, Mohan Nagar, Ghaziabad, India.

**Sunita Tiwari**, Associate Professor, Computer Science Department, Krishna Engineering College, Mohan Nagar, Ghaziabad, India.