Reputation Management System in Peer-to-Peer networks

A *Thesis Submitted*

in Partial Fulfilment of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

by

RUCHIR GUPTA

to the

DEPARTMENT OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY KANPUR

Jul, 2013

CERTIFICATE

It is certified that the work contained in the thesis entitled "**Reputation Management System in Peer-to-Peer networks**" being submitted by **Mr. Ruchir Gupta** has been carried out under my supervision. In my opinion, the thesis has reached the standard fulfilling the requirement of regulation of the Ph.D. degree. The results embodied in this thesis have not been submitted elsewhere for the award of any degree or diploma.

[illegible]

Dr. Yatindra Nath Singh
Professor
Department of Electrical Engineering
Indian Institute of Technology Kanpur
Kanpur, INDIA

23 **Jul., 2013**

Synopsis

Name of the Student	:	**Ruchir Gupta**
Roll Number	:	**10104121**
Degree for which submitted	:	**Ph.D.**
Department	:	**Electrical Engineering**
Thesis Title	:	**Reputation Management System in Peer-to-Peer networks**
Thesis Supervisor	:	**Dr. Yatindra Nath Singh**
Month and year of submission	:	**Jul., 2013**

Peer-to-peer networks generally comprise of rational users. This leads to the problem of free riding, i.e, users want to use resources from network but do not want to contribute back to the network. This problem can be theoretically explained on the basis of *Prisoners' Dilemma*. The problem of free riding can be overcome by providing suitable incentives for sharing. For proper provisioning of incentives, some kind of reputation management system is required.

Peer-to-peer networks don't have any central control or repository. Large size of peer-to-peer networks makes the reputation management more challenging task. Hence reputation management system should perform all the tasks in distributed fashion. When these kind of systems are implemented, peers try to deceive them to take maximum advantage. Whitewash and collusion are two

ways to deceive the reputation management system. Here whitewash implies the tendency of a peer to change its identity to avoid the bad reputation. Whereas collusion means forming groups to deceive the system collaboratively.

Probabilistic allocation based on reputation may be a good option for allocation of resources because in this case nodes that don't have very good reputation about each other, may allocate at least some resource with finite probability. This avoids disconnect between them.

In this thesis, various issues in different reputation management systems have been explored. New algorithms for making a complete reputation management system have been proposed. An algorithm for resource allocation on the basis of reputation is proposed. The main objective of the thesis is to identify the limitations of existing reputation management system and to solve them by proposing new algorithms that are simple and light weight in implementation.

The thesis has been organized in the following six chapters.
Chapter 1 discusses Client Server and peer-to-peer architectures, their differences, advantages and disadvantages. Classification of peer-to-peer networks have also been presented. Problem of free riding, its theoretical justification on the basis of Prisoners' Dilemma and experimental evidences on the basis of different experimental [1, 2, 3] studies are presented.

Chapter 2 discusses the estimation of trust in peer-to-peer file sharing network. Different uncertanities involved in measurement of trust are explained and have been considered for trust estimation in the proposed algorithm. These uncertanities include uncertainty due to load variation, uncertainty due to congestion, uncertainty due to multiple demands. Best Linear Unbiased Estimator (BLUE) [4] is used for estimation. Proposed hypothesis has been verified through simulation.

Chapter 3 discusses the aggregation of trust values from all nodes in the network. Before aggregation, the aggregating node multiplies weight of every node to the trust value shared by that node. These weighted trust values are averaged to get a final aggregated trust value. Nodes perform distributed averaging using a variant of gossip algorithm viz. 'differential gossip algorithm' for scale free networks. An upper bound for convergence time of differential gossip algorithm has also been derived. Robustness of algorithm against collusion has also been analysed theoretically. Simulation results are presented to understand the performance of algorithm.

Chapter 4 discusses the problem of whitewashing and proposes a solution by making initial reputation adaptive. Adaptive initial reputation algorithm has been analysed and it is concluded that this algorithm discourages the whitewashing tendency of users. Payoff for a newcomer and a whitewashing nodes are also calculated in presence of permanent, free and finite cost identities respectively.

Chapter 5 proposes a resource allocation algorithm based on reputation. Algorithms are presented for optimizing the shared capacity, reputation based probabilistic allocation that is optimal for a node, and formation of interest groups on the basis of similarity between interests of nodes. Generally in peer-to-peer network many users have common interests but they may not necessarily form a group. Simulation results are presented to verify the hypothesis.

Chapter 6 discusses the major conclusions of the thesis, and suggest problems that can be investigated further.

Dedicated to

my Family, Teachers

and Friends

&

My Brother

Late Sri Rama Kant Gupta

Acknowledgements

I would like to express my deepest gratitude to my advisor Professor Y N Singh for his constant support, patience, and encouragement throughout my research. I have been very much fortunate to have an advisor who gave me the freedom to explore my own research, and at the same time the guidance to recover when things became turbulent. He taught me what actually research is and how to approach and solve the problems. His patience for my silly acts and questions was really commendable.
I am also thankful to Dr. Vimal Kumar for his help regarding game theory.

I am also grateful to all the faculty members of Department of Electrical Engineering, IIT Kanpur for their support and encouragement. I would like to thank especially Dr. Adrish Banerjee and Dr. S S K Iyer for their support and encouragement.

I would like to thank IIT Kanpur Student Assistantship for making my graduate years financially secure.

This dissertation has also been considerably improved because of inputs from

Abhay Kumar Shah and Gopal Parashari in the form of comments and through discussions. I am very grateful for their help and friendship.

I immensely express my heartiest thanks to my friends Adarsh, Anupam, Gaurav, Ram, Rameshwar, Gagan, Anurag and Satish for their support and help.

Although it is beyond the scope of any acknowledgement for all that I have received from my family members, I take this opportunity to express my heartfelt and affectionate gratitude to them.
My parents Mr. H. R. Gupta and Mrs. Hemant Gupta receive deepest gratitude and love for their inspiration, patience and encouragement.
I also want to thank my bhaiya Mr. Sandeep Gupta, Saumya Bhabhi, Padma bhabhi, didi and Jija ji for their support and encouragement. My nephews Puneet and Ishan and niece kopal, Pankhuri, Priyanshi and Shivangi also deserve an affectionate thank for their valuable support and love.

Finally, I want to thank my wife Rakhi Gupta and son Shreyas for their patience, support and understanding without which this dissertation would not has been completed.

(Ruchir Gupta)

Acronyms and Abbreviations

DHT	Distributed Hash Table
NE	Nash Equilibrium
BLUE	Best Linear Unbiased Estimator
TCP	Transport Control Protocol
UDP	User Datagram Protocol
RTT	Round Trip Time
PA	Preferential Attachment
RMS	Root Mean Squire

List of Symbols

t_{ij}^{k}	trust estimated by node i from node j for k^{th} transaction
$q_{r,ij}$	data rate requested by the node i to node j.
$q_{w,ji}$	willing service rate from node j to node i
r_{ji}	reputation of node i for node j
v_j	arbitrary constant chosen by node j
x	reputation exponent
B_j	Shared bandwidth of node j
$q_{o,ji}$	Offered service rate from node j to node i
n	Time instant
$\tilde{w}_{ji}$	Uncertainty due to demand variation on node j in the network
$Q_{w,ji}$	willing service rate from node j to node i per unit request
$Q_{o,ji}$	Offered service rate from node j to node i per unit request
$\tilde{W}_{ji}$	Uncertainty due to demand variation on node j in the network per unit request
C_{ji}	uncertainty parameter
$C_{ji,1}$	ratio of the total request made by the node to the download capacity of the node
$C_{ji,2}$	ratio of total capacity shared by all the nodes to the total requests made in the network by all the nodes
$\hat{Q}_{w,ji}$	estimated willing service rate from node j to node i per unit request
P $q_{ij,ay}$	accepted data rate
$q_{ij,an}$	not accepted data rate
$q_{a,ji}$	actual service rate from node j to node i
$q_{f,ji}$	feasible service rate from node j to node i
η_i	punishment parameter
W_{max}	maximum window opened by the receiver
b	number of packets acknowledged by a single acknowledgement

p	packet loss probability
T_0	time-out period
β_i	node behaviour parameter
$\mathbf{Cov}$	covariance matrix
Δt	absolute change in reputation
itr	iteration number
N	Number of nodes in the network
P_i	Probability of selection of node *i* by new coming node in PA model
$\mathbf{t}$	Trust Matrix
g_{ij}	Gossip Weight assumed by node *i* for node *j*
y_{ij}	Information to be gossiped
csl	number of steps that a node will continue to gossip after its convergence
R_{global}	global reputation
w_{ij}	Weight given by node *i* to node *j*
a_i	weight calculation parameter
b_{ij}	weight calculation parameter
W_{Iij}	for node I, the ij element in weighted matrix
Rep_{Ij}	globally calibrated local reputation of node *j* for node *I*
ξ	gossip error bound
NS_i	Set of neighbours of node *i*
$g_{n,j}$	evolved gossip weight in nth time instant at node *j*
$c_{n,i,j}$	contribution received from all nodes in nth time instant at node *j*
ψ_n	potential function
d_k	degree of node *k*
γ	network exponent
P_{d_k}	probability that node *k* has degree *d*
r_{ij}	reputation computed by differential gossip in presence of colluding nodes
$\hat{r}_{ij}$	reputation computed by differential gossip in absence of colluding nodes
$R_{ini,i,n}$	initial reputation offered by node *i*
h_i	honesty level of node *i*
$G_{local,i}$	local growth rate w.r.t. node *i*
$d_{average}$	average degree of complete network
$d_{local,i}$	local average degree w.r.t. node *i*
L_i	legitimate departing nodes

A_i	newly arriving nodes
$\acute{w}_i$	number of whitewashing nodes
$\acute{W}_{i,n}$	level of whitewashing at node *i*
$\acute{W}_{i,max}$	maximum whitewashing level
$R_{ini,i,max}$	maximum initial reputation that can be offered by node *i*
$R_{ini,i,min}$	minimum initial reputation that can be offered by node *i*
P_{wa}	Probability by which a whitewashing node will attempt whitewash
$q_{r,i,d}$	download capacity of node *i*
$P_{allocation,i,j}$	Probability of allocation of resource from node *i* to node *j*
$U_{s,i}$	shared capacity of node *i*
χ_{ij}	similarity coefficient between node *i* and node *j*
Ω_{ij}	number of time node *i* node *j* pr vice versa
$score_{ij}$	score for ranking

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Introduction

Exchange of knowledge and information has always been a strong desire of human being. Internet has come up as a good medium for it. Traditionally internet has been operated on client server model. Client server model is defined as a model where one entity acts as server and other entities act as clients. Whenever a client needs something, it asks server for the same. Server provides the requested resource to the client. These kind of systems have problems of scalability and have single point of failure. Initially, as the computers were expensive as well as of low computing capacity client server structure evolved to share expensive and high computing capacity server while using low cost ordinary clients..

Now a days, due to development of integrated circuit technology, computing and storage become cheaper As a result, the difference of computing power between server and clients has reduced. Another consequence of this is that the server cannot serve the same size of client population. Another innovation has happened in the form of optical fiber which provides very high bandwidths making the bandwidth cheaper. The combination of above has made it possible to talk about of almost equal status machines connected with a reliable network, forming peer to peer network. Whenever a peer needs some resource, starts a search among all peers in the network. The peer that have the requested resource, serves the requesting peer. Scalability is not an issue in these kind of networks as serving capacity of network is also growing with increase in the number of peers. As all peers are equal, absence of any particular peer does not affect the system. In this thesis words peer and node are used interchangeably.

1.2 Classification of Peer-to-Peer Networks

Peer-to-peer networks are classified on the basis of query resolution technique. If object location is stored distributively following a specific algorithm, we call it structured peer-to-peer network. Networks that do not follow any such algorithm are called unstructured peer-to-peer networks.

Structured peer-to-peer networks are further classified as centralised, pure and hybrid peer-to-peer networks. These are discussed in the subsections later.

1.2.1 Structured Peer-to-Peer Networks

Structured peer-to-peer networks are the networks where object location is stored distributively following a specific algorithm. This algorithm ensures the faster query response. Structured peer-to-peer networks guarantee the resolution of a query for the resource that is available in the network even for the rarest resource. Structured peer-to-peer networks generally uses distributed hash table (DHT). Example of such networks are Chord [5], Pastry [6], Tapestry [7] etc..

In such networks, objects' locations are stored according to their hash values. A map is created with the hash values of node ids. Objects are mapped to the node that has nearest hash value to object's hash value. If some node wants to search some object, it computes the hash value of that object id and then looks up the node that has id having hash value nearest to hash value of object id. Now it contacts this node and asks about the location of said object. Once it gets the location of object, it contacts the node that has the object and downloads it from there directly. This process is explained in figure 1.1. In this figure, red lines are used for query propagation whereas green lines are used for data transport. These networks require high overhead to maintain graph for efficient query routing.

1.2.2 Unstructured Peer-to-Peer Networks

Unstructured peer-to-peer networks are those that do not follow any specific algorithm for storage of object location. These networks are further classified on the basis of query resolution.

1.2.2.1 Centralised Peer-to-Peer Networks

Centralised Peer-to-Peer networks are the networks that have one or more server in the network. This server is used for indexing services. Indexing servers are infrastructure nodes. When somebody wants to share some object, it informs the

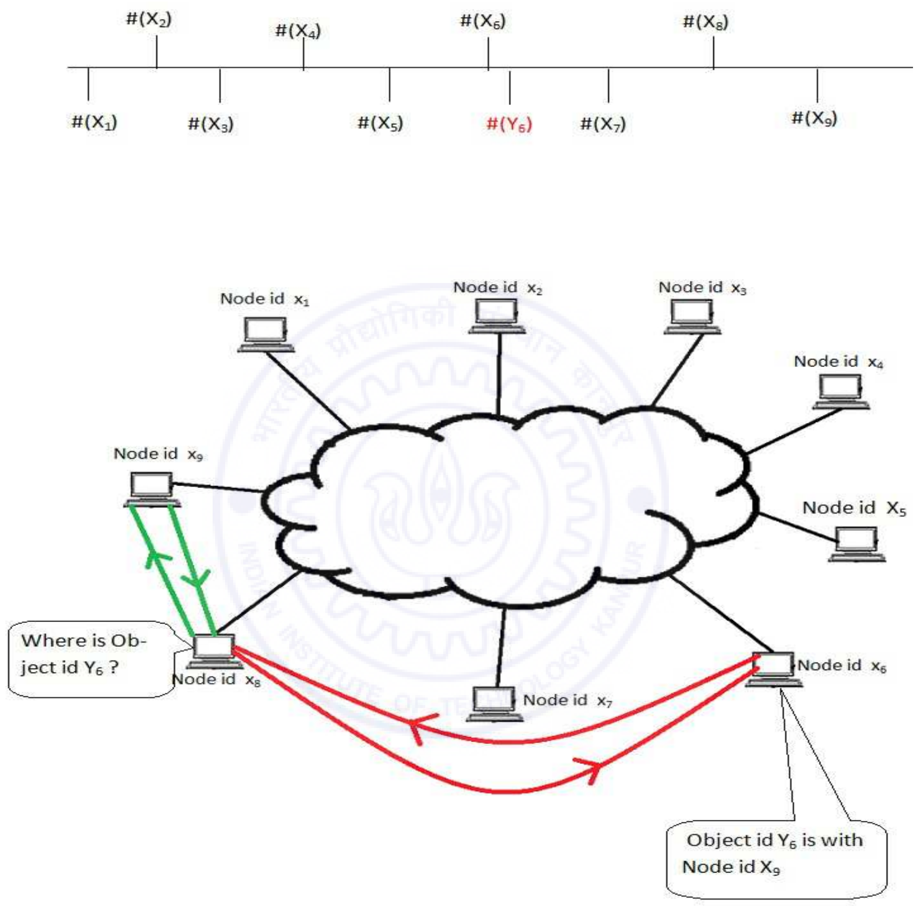

Figure 1.1: Structured Peer-to-Peer networks

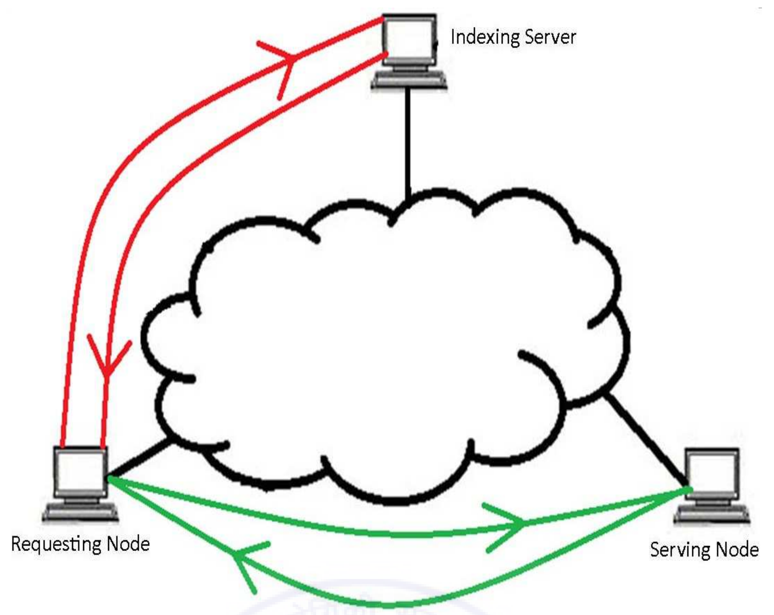

Figure 1.2: Centralised Peer-to-Peer Network

server about the object and its own address. The Server keeps this detail with it. When any node needs an object, it asks indexing server about it. Indexing server, inform the querying node the location of resource on the basis of indexing entries. Once the querying node gets the information about the object's location, it directly contacts the concerned node and gets the object. This process is explained in figure 1.2. Napster [8] is an example of such kind of peer-to-peer network.

1.2.2.2 Pure Peer-to-Peer Networks

Pure peer-to-peer networks are the networks where every node is equal to any other node. These networks can be seen as a replica of human networks. When some node wants to search any objects, it asks its neighbours. If neighbours have that object, they reply to query otherwise they propagate the query to their

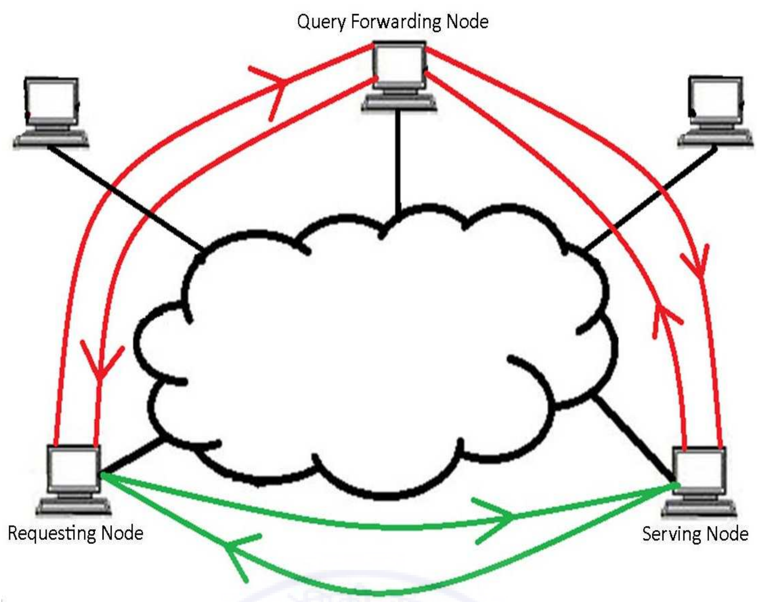

Figure 1.3: Pure Unstructured Peer-to-Peer networks

neighbours and so on. When query reaches to a node that has the queried object, it sends the response to the the query. The response follows the same path in reverse direction, which was followed by the query. After getting the reply of its query, querying node contacts directly to the responding node and gets the object. This process is explained in figure 1.3. These networks are simple in nature and do not guarantee the discovery of resource. There are many methods proposed in literature to make search more efficient in such networks viz. random walk [9], Bubblestorm [10] etc..

1.2.2.3 Hybrid Peer-to-Peer Networks

In hybrid peer-to-peer networks few nodes act as super nodes and others are normal nodes. Super nodes play an important role in making search process efficient. In these networks, super nodes are mutually connected and normal nodes are connected to one of the super nodes. When a node needs some object, it asks a super node for the object. Super node asks other super nodes for that object and in this way node gets the query resolved. Once the query is resolved, node directly takes the object from the replying node. This process is depicted in figure 1.4.

1.3 Free Riding in Peer-to-Peer Networks

Distributed nature of the peer-to-peer networks brings many challenges for system designers. These networks are usually designed keeping in mind that every node is honest and co-operative. It means, if some node takes some resource from the community, it will also reciprocate to the community.

But nodes are the entities operated by rational human beings so they are expected to behave in a selfish manner i.e. they try to maximise their utility. This results in their non co-operative behaviour. This phenomenon is explained by the famous problem of game theory – *Prisoners' Dilemma* which shows that individuals do not cooperate even if cooperation may yield better result. [11].

1.3.1 Game Theoretic Explanation of Free Riding

In this example, there are two prisoners A and B. They have been caught by police and are suspect of a major crime. Police does not have proper evidences against them. Police interrogates them separately. If both of them says that other one is

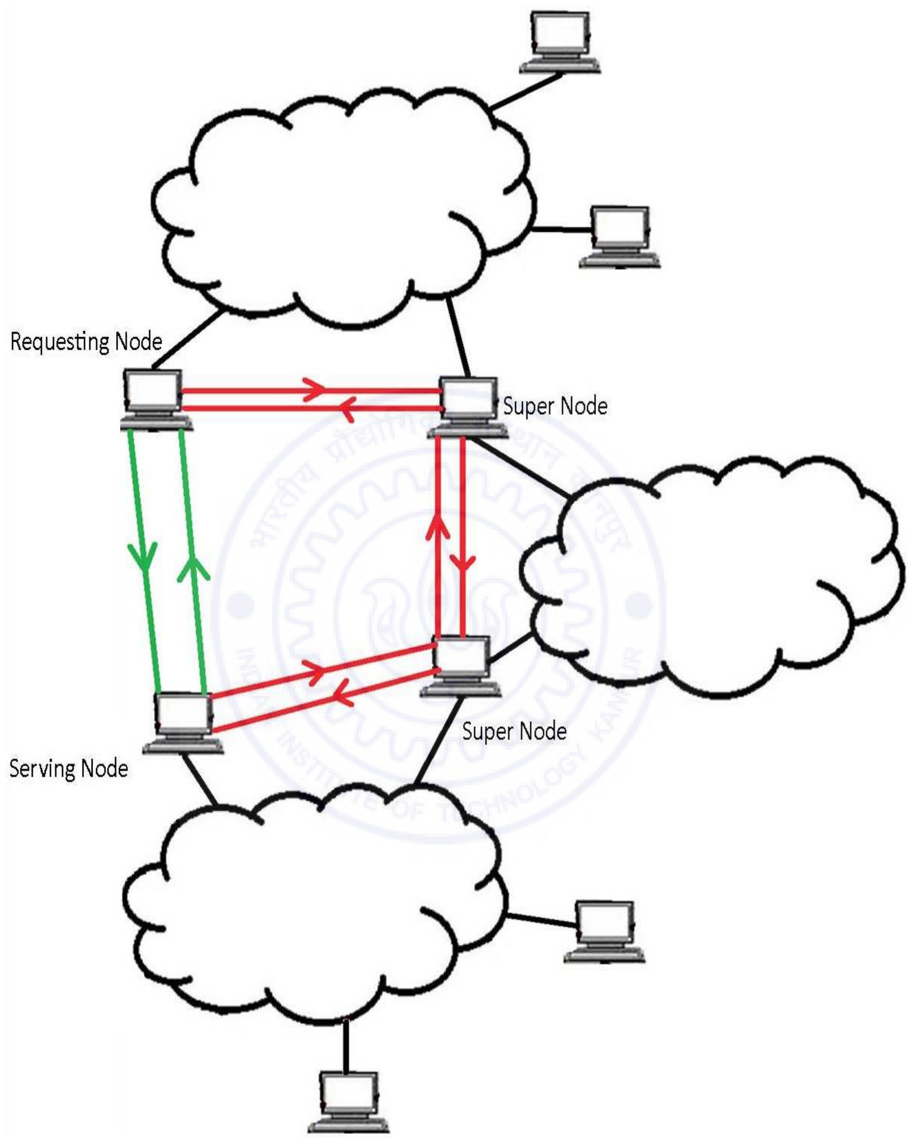

Figure 1.4: Hybrid Unstructured Peer-to-Peer networks

	Cooperate(B)	Defect (B)
Cooperate(A)	(1,1)	(10,0)
Defect(A)	(0,10)	(5,5)

Table 1.1: Payoff in Prisoners' Dilemma

innocent, i.e. cooperate with each other, both will be convicted for a minor crime and get a sentence of only one year. If A says that B is guilty and B says A is innocent, i.e. A defects and B cooperates, A walks free and B gets sentence of 10 years. Similarly if B defects and A cooperates, B will be freed and A will be sentenced for 10 years. If both defect, both of them will be sentenced for 5 years each. These payoffs (Number of years prisoner is sentenced) are shown in table 1.1. First payoff is for player or prisoner A whereas second payoff is for player or prisoner B. Prima Facie it seems that (Cooperate, Cooperate) gives the least punishment so it should be the best strategy and hence both should follow it. When this is analysed for players individually, we find out that it is not the case. Considering the payoff for player A, it can be seen that when B cooperates, defect will free the prisoner A whereas cooperate will put him in jail for one year. When B defects, defect will put the prisoner A in jail for 5 years, whereas cooperate will put him in jail for 10 year. Hence, it can be observed that for player A, defect is the most optimal choice. Applying similar logic while considering the payoff of player B, it can be observed that defect is the most optimal choice for B as well. Therefore, the Nash Equilibrium (NE) of this game is (defect, defect), i.e., when both prisoners deceive each other.

Similarly in a file sharing network, if there are two nodes, their interaction can also be modelled as a two players Prisoners' Dilemma game. If players get a payoff

	Share(B)	Not Share(B)
Share(A)	$(d - s, d - s)$	$(-s, d)$
Not Share(A)	$(d, -s)$	(0,0)

Table 1.2: Payoff in two player File sharing system

of $-s$ for sharing and d for downloading, the payoff table will be as follows. Here nodes are considered as players. When both players share, both gets the payoff of ($d - s$), when one player shares and one does not then sharing player receives the payoff of $(-s)$ as there is nothing to download for him and non-sharing player receives the payoff of (d) as he is not sharing any thing and when nobody shares, no body gets any thing either positive or negative i.e. zero payoff. Analysing as the above case like the prisoners dilemma, it can be observed that equilibrium of the game is (not share, not share) i.e. with payoff (0, 0). It means that NE is the point where none of them shares any resources [12]. This tendency of nodes to draw resources from the network and not giving any thing in return is termed as 'Free Riding'.

1.3.2 Experimental Studies on Peer-to-Peer Networks

Experimental Studies on different peer-to-peer networks have been performed to measure the level of contribution made by the nodes to the network. First study was conducted on Gnutella network [13] in 2000 by Adar *et al.* [1]. In this study, they observed the following points.

1. In Gnutella, 70% users did not share any file.

2. 5% of the hosts shared 70% of all the files.

3. Top 1% hosts shared 25% and top 25% shared 98% of all files.
4. 63% of the peers, who shared some files, never answered a single query. This may be thought as another form of free riding.
5. Free riders were found distributed equally through the network.

Another study in 2002 by Saroiu *et al.* [2] on Nepster and Gnutella networks revealed the following points.

1. 25% of total hosts in Gnutella did not share any file.
2. 75% of total hosts in Gnutella shared 100 or less files and 7% of total hosts shared major portion of total files shared in the network.
3. In Napster, 40% to 60% of hosts shared 5% to 20% of total files.
4. Peers were found to report less bandwidth than they were actually having.

One more study conducted on Gnutella in 2005 by Hughes *et al.* [3] which shows that free riding increased over time since 2000. Their findings were, 85% of hosts shared no file. This number is bigger than that was observed in 2000 by Adar *et al.*. It is also reported that free riding is possible in Bittorrent. [14].

1.4 Solution to Free Riding Problem

In literature, various researchers have suggested different methods to combat free riding. these methods have generally proposed to give some incentive for sharing or disincentive for not sharing or both [15, 16, 17, 18]. These incentives are generally provided in terms of better quality of service to contributing node and degraded quality of service to non contributing nodes when they request for some resource. Few people have also come up with the idea of micro-payments, i.e. every node should pay some amount of money for receiving the resource, to the serving node [19].

To implement incentives or disincentives of any form, one needs methods a system that may keep account of actions of a node. Such systems are termed as reputation management systems [20, 21, 22, 23, 24]. In a peer to peer file sharing network, trust or reputation of a node represents a measure of its co-operative behaviour towards other nodes. A node seeking a resource from another node measures the ratio of received resources to the requested resource after every transaction and uses it to update the trust value. If a node imparts all the requested resource, it's reputation is considered as one and if a node always declines sharing of resources, its reputation is considered as zero.

Eigen-Trust [21] depends largely on pre-trusted peers i.e. peers that are globally trusted, this is scalable to a limited extent. Peer-Trust [20] stores the trust data (i.e. trust values of all the peers in the network) in a distributed fashion. This is performed using a trust manager at every node. In Peer-Trust, hash value of a node id is calculated to identify the peer where the trust value of node will be stored. Song *et.al.* [25] used fuzzy inference to compute the aggregation weights. In Fuzzy-Trust, each peer maintains the local trust value and transaction history of the remote peer. At the time of aggregation each peer asks for the trust value from the qualified peers and combine the received values and locally existing values to compute updated trust values. Power trust [23] depends largely on power nodes. Power nodes are few top reputation nodes in the network. It uses score manager like trust manager in Peer Trust and Look ahead Random Walk for aggregation. Gossip Trust [22] uses gossip algorithm for aggregation. PET [24] categorises services qualities of different transactions into four types and then compute the total average of a node by giving different weight to each category of transaction.

In order to gain insight into intricacies of reputation systems, we can look into what is implemented by on-line commerce portals. In e-commerce portals like e-bay, people sell and buy different things on-line. Buyers and sellers generally do not know each other, so the possibility of cheating or possibility of providing a product or service of inferior quality always exist. To avoid this, e-bay uses a rating based reputation system. After every transaction, user gives a feedback rating to his counterpart and based on these ratings, reputation is decided. This reputation helps users in making decision about transactions [26].

The advantage with e-bay like systems is that these have a central server which keeps all the feedback rating and reputation related data. So, if a user wants to check any other user's reputation, he just asks the central server and receives authentic information. In peer-to-peer file sharing networks, as there is no central server, trust has to be estimated and stored by each node in a distributed fashion.

When reputation management is implemented, nodes try to deceive the system to exploit the network. This results in problem of collusion and whitewashing. Here whitewash implies the tendency of a peer to change its identity to avoid the bad reputation whereas collusion means forming groups to deceive the system collaboratively.

Many authors have suggested that whitewashing can be totally removed if system has permanent identities [27, 28]. Problem of whitewashing due to availability of cheap or free identities was initially pointed out by Friedaman *et.al.* [29]. They proved that considering every newcomer as defector is a better policy than any other static stranger policy. Whereas Feldman *et.al.* [30] proved that newcomers should only be punished if turnover rate is high. They also proved that a legitimate user can only win from a whitewashing user if newcomer will be served well with a very small probability [31]. Yang *et.al.* studied Maze file sharing system and concluded that incentives promote whitewashing [32].

Probabilistic allocation based on reputation may be a good method to allocate the resources because in this case nodes that do not have very good reputation about each other, may be allocated at least some resource with finite probability. This avoids disconnect between them.

In this thesis, we have investigated a new algorithm for making a complete reputation management system. The proposed system is robust against collusion and whitewashing tendency of nodes. Moreover, it poses less overhead on the system. An algorithm for resource allocation on the basis of reputation is also proposed.

Rest of the thesis is organised as follows. Chapter 2 discusses the uncertanities in trust measurement and proposes an estimator for trust estimation. Chapter 3 proposes a method to aggregate the opinions of different peers in the network to get the reputation of a peer. In this method every peer gives higher weights to its trusted peers. The aggregation is performed using a variation of gossip algorithm. An upper bound for convergence time of differential gossip algorithm is also computed. Chapter 4 computes the payoff of cooperative peer and non cooperative peer. It proposes a solution to avoid whitewashing tendency of non cooperative peers when zero cost identities are used in the network. Chapter 5 proposes the reputation based probabilistic resource allocation algorithm. It also proposes the algorithm to form common interest groups. Chapter 6 presents the major conclusions of the thesis and possible future work.

Chapter 2

Trust Estimation in Peer-to-Peer Network Using BLUE

2.1 Introduction

A good trust estimation method is required for efficient reputation management system in peer-to-peer networks. Trust can be estimated in a very simple way as the ratio of received to requested resources but this simple method can not overcome the effect of noise (i.e. uncertainty) in the estimation of trust value. We are proposing a trust estimation method using BLUE (Best Linear Unbiased Estimator) [4]. This method overcomes the noise effects considerably and requires almost same amount of memory and computation.

The remainder of this chapter is organised as follows. Section 2.2 discuses the related work in reputation management while section 2.3 describes the system model. In section 2.4 the estimator for trust using BLUE is derived. Section 2.5 presents the numerical results and section 2.6 concludes the chapter.

2.2 Related Work

Different authors have used different approaches for measurement of trustworthiness of a node. The trust is more if the contributions from the node are more. One approach is to observe the contribution made by the serving node [15, 33, 34, 35]. Second approach is to consider the quality of service obtained from the serving node [33, 36, 37, 38, 21, 20, 23, 22]. Another approach is to take the ratio of the resources received and provided to a node [39, 40, 41]. It may be noted that in this case reputation can be more than one. One more approach is to calculate the ratio of the sum of resources received to the resource requested to that node for last ten transactions [42].

Different methods are employed for measurement using the above said approaches. Eigen-Trust [21] uses sum of positive and negative ratings, Peer-Trust [20] normalises the rating on each transaction whereas Power-Trust [23] uses Bayesian approach to calculate reputation locally.

Mengshu *et.al.* [43] took the ratio of successful transactions to total transactions. PET [24] categorises services qualities of different transactions into four types and then compute the total average of a node by giving different weight to each category of transaction. In [44], different reputation is been calculated for different resources. In Fuzzy-Trust [25], nodes do *fuzzy inference* on parameters to calculate the trust score locally for another node 'x' and then aggregate it with trust scores of the node 'x' as received from other nodes using their weights. In [45], peer maintains a binary vector of m bits. After a transaction, one or zero is added at most significant position of the vector, after shifting right all the previously placed bits by one place. This trust vector is considered as a m bit binary number. To compute the trust, value of the m bit binary number is divided by 2^m. This ensures that the trust value lies in between 0 and 1. But none of the above work considers the uncertainties in the input while estimating the trust value.

2.3 System Model

There is no dedicated server in peer-to-peer networks and peers in this network are rational, i.e. they are only interested in their own welfare. They are connected to each other by an access link followed by a backbone link and then again by an access link to the second node. We are assuming that the network is heavily loaded i.e. every peer has sufficient number of pending download requests, hence these peers are contending for the available transmission capacity. We also assume that every peer is paying the cost of access link as per the use. So, every peer wants to maximises its download and minimise its upload so that it can get maximum utility of its spending and this leads to problem of free riding.

If a node is downloading, some other node has to upload. So the desired condition is, download should be equal to upload for a node. Usually this means that there is no gain. Even in this scenario the node gains due to interaction with others. In any society, even when resources given and taken are same for each participating entity, they have better chance of survival compared to a society where there is no interaction. Thus interaction itself is an incentive for all the peers as it increases their utility.

A node will usually try to get the content and avoid uploading it as this maximises gain for it. Thus free riding becomes optimal strategy. So, a reputation management system needs to be enforced to safeguard the interest of every node by controlling the free riding.

In reputation management system, every node maintains a reputation table. In this table, the node maintains the reputations of the nodes with which it has interacted. Whenever it receives a resource from some node, it adjusts the reputation of that node accordingly. When a node asks for the resource from this node, it checks the reputation table and according to the reputation value of requesting node, it allocates resource to that node. This ensures that every node is facilitated from the network as per its contribution to the network and consequently free riding is discouraged.

For using such a reputation management system, a node needs to estimate the trust value of the nodes interacting with it. This estimation can be made on the basis of requested and actual transfer rates and other parameters after every transaction.

A node can estimate the reputation of nodes with whom it is interacting. But the existing neighbours may not always suffice and it may have to interact with new nodes. At that point it can either use a default initial reputation or some estimate which is maintained via reputation management mechanism. For an efficient p2p network, we would like to use some realistic estimate of reputation of new node. Next chapter describes a reputation management mechanism.

The trust can be estimated by the various methods as described in the literature.

We have modified one such method where a node estimates the trust by

$$t_{ij}^{k} = \frac{Z_{ij}^{k}}{Requ_{ij}^{k}}. \tag{2.1}$$

Here $Requ_{ij}^{k}$ represents the amount of resources requested by node i from node j for k^{th} transaction; Z_{ij}^{k} represents the amount of resource received by node i from node j in k^{th} transaction.

In estimation of trust value by such a method, few important points are missed out. These points are as follows.

1. In peer-to-peer networks, when a peer asks for some resource, it is not guaranteed that it will get the asked resource. So, generally peer asks for larger amount of resource than needed and when it is offered more resource than its requirement, it refuses the extra offered resource. Also, it does not give any credit for this extra offer.

2. Once requesting node decides about the node from which it is going to take data, both of the nodes decide about the rate of data according to their upload and download capacities. But at the same time underlying network may not be able to provide the agreed data rate because of congestion in the network at different routers. So, even if service provider node is willing to give data, it may not be able to send at the committed rate, due to limitations of underlying transport network. This affects the reputation assignment as reputation is assigned on the basis of actual data rate.

3. If some node has already got too many requests, it will not be able to provide the quality of service that it could have provided with lesser load. To avoid

the distortion due to the above fact, the requesting node should estimate the reputation of service provider node considering the load and the previous transactions with that node.

One may argue about having a minimum QoS constraint. If this constraint is put, then a node with the resource will provide the service only when it can provide the QoS above a threshold. In this case also, the requesting node will assume that the serving node is not cooperating and thus will reduce its reputation drastically. While, providing service even with lower QoS indicates that the serving node is willing but it may have bandwidth constraints. Thus the reputation reduction may be subtle and not drastic considering the earlier behaviour. Although it is possible that a serving node may only provide the service if QoS is above a threshold, but in this case the reputation should be estimated with much less successful sample interactions with serving nodes and many more interaction where service was not provided. The successful interaction need to somehow indicate that the serving node is not responding intentionally due to QoS constraint and not due to unwillingness to provide the service.

To resolve the above issues, Node will use equation (2.8) for calculation of trust value (t_{ij}).

Let the data rate $q_{r,ij}$ is requested by the node i. At the serving node, $q_{w,ji}$, the willing service rate from node j to node i, is decided according to point 3 mentioned in earlier paragraph. As $q_{w,ij}$ will change according to $q_{r,ij}$, an estimate of $\frac{q_{w,ji}}{q_{r,ij}}$ is desirable.

Assuming that reputation of node i for node j is r_{ji}, we can see that,

$$q_{w,ji} = f(q_{r,ij}, r_{ji}, v_j).$$

Here, v_j is an arbitrary constant chosen by node j and will range from 0 to 1. As this is a distributed scenario, nodes will not be bound by rules and they will act as per their wish. It is not guaranteed that a node will serve strictly as per the value obtained on the basis of earlier requests made by a node and the reputation of that node. The vj takes care of this tendency. $v_j = 1$ implies that node is serving as per the value obtained from function whereas $v_j = 0$ implies that node is not providing any service. One possible way to decide $q_{w,ji}$ can be,

$$q_{w,ji} = q_{r,ij} \cdot (r_{ji})^x \cdot v_j.$$

Here x is a reputation exponent. Value of x will be same for all the nodes in the network. The details about x are given in section 4.4. Nodes can decide individually on the mechanism of computing $q_{w,ji}$. Generally with higher r_{ji} and $q_{r,ij}$, a higher $q_{w,ji}$ should be chosen. After deciding willing service rate for all the requesting nodes, two different conditions are possible,

$$\begin{aligned} & \sum_i q_{w,ji} \leq B_j, \\ \Longrightarrow \quad & \forall i,\ q_{o,ji} = q_{w,ji}. \\ & \textit{or} \\ & \sum_i q_{w,ji} > B_j \implies q_{o,ji} \leq q_{w,ji} \\ & \textit{such that} \\ & \sum_i q_{o,ji} = B_j. \end{aligned}$$

Here B_j is the shared bandwidth of node j and $q_{o,ji}$ is the offered service rate from node j to node i. Thus we can write

$$\begin{aligned} q_{o,ji}[n] &= q_{w,ji}[n] - \tilde{w}_{ji}[n] \\ \implies \frac{q_{o,ji}[n]}{q_{r,ij}[n]} &= \frac{q_{w,ji}[n]}{q_{r,ij}[n]} - \frac{\tilde{w}_{ji}[n]}{q_{r,ij}[n]} \end{aligned} \tag{2.2}$$

Here n is time instant. $\frac{q_{o,ji}[n]}{q_{r,ij}[n]}$ is observable quantity and $\frac{\tilde{w}_{ji}[n]}{q_{r,ij}[n]}$ is the uncertainty due to demand variation on node j in the network. Equation (2.2) can be written as,

$$Q_{o,ji}[n] = Q_{w,ji}[n] - \tilde{W}_{ji}[n] \tag{2.3}$$

Here $Q_{o,ji}[n]$, $Q_{w,ji}[n]$ and $\tilde{W}_{ji}[n]$ are $\frac{q_{o,ji}[n]}{q_{r,ij}[n]}$, $\frac{q_{w,ji}[n]}{q_{r,ij}[n]}$ and $\frac{\tilde{w}_{ji}[n]}{q_{r,ij}[n]}$ respectively. We can see that $Q_{w,ji}[n] = (r_{ji})^{\alpha} \cdot v_j$ will remain constant over the time, so it can be replaced by $Q_{w,ji}$. Let us assume that mean value of $\tilde{W}_{ji}[n]$ is $\tilde{W}_{ji}$ and the ratio of $\tilde{W}_{ji}$ and $Q_{w,ji}$ is C_{ji} given by

$$C_{ji} = \begin{cases} 1 - \frac{1}{C_{ji,1} \times C_{ji,2}}, & \text{if } C_{ji,1} \times C_{ji,2} > 1. \\ 0, & \text{otherwise.} \end{cases} \tag{2.4}$$

Here

$$C_{ji,1} = \frac{\textit{requests made by the node}}{\textit{download capacity of the node}} \tag{2.5}$$

$$C_{ji,2} = \frac{\textit{total capacity shared by the nodes in the network}}{\textit{total requests made by nodes in the network}}. \tag{2.6}$$

Here $C_{ji,1}$ is the ratio of the total request made by the node to the download capacity of the node i.e. how much node over requested, and $C_{ji,2}$ is the ratio of

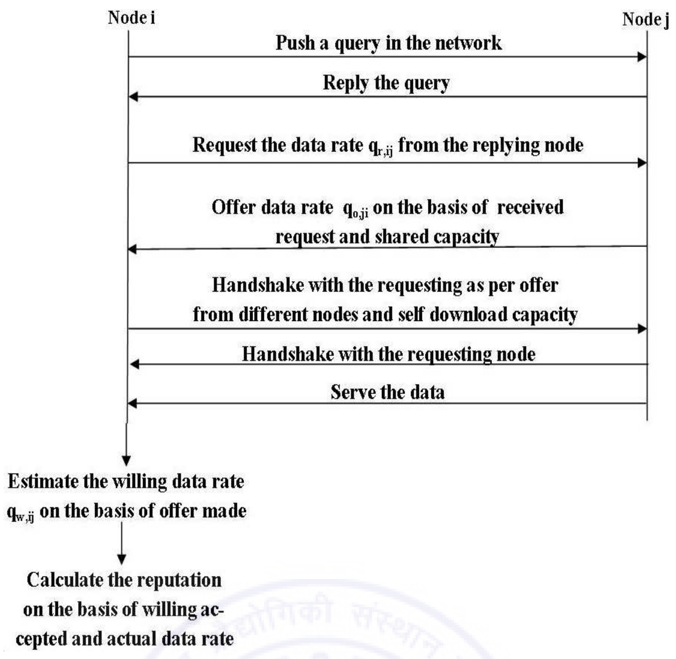
Node i
Node j
Push a query in the network
Reply the query
Request the data rate $q_{r,ij}$ from the replying node
Offer data rate $q_{o,ji}$ on the basis of received request and shared capacity
Handshake with the requesting as per offer from different nodes and self download capacity
Handshake with the requesting node
Serve the data
Estimate the willing data rate $q_{w,ij}$ on the basis of offer made
Calculate the reputation on the basis of willing accepted and actual data rate

Figure 2.1: Process Diagram

total capacity shared by all the nodes to the total requests made in the network by all the nodes. $C_{ji,2}$ is statistically the average upload capacity against the unit request made by a node in the network. Multiplying $C_{ji,2}$ with $C_{ji,1}$ will give us the average upload capacity per unit download capacity of the node. We have assumed that uncertainty have identical distribution for every sample and all the samples are independent, i.e. the noise samples are i.i.d.. Hence, we expect every sample to have same variance. Let us assume this variance be σ. Estimate for $q_{w,ji}$ i.e. $\hat{q}_{w,ji}$ is derived in section 2.4.

Once a node gets an offer from some node, it may act in three ways, first, it may completely accept the offer, second, it may partially accepts the offer or third it may not accept the offer as mentioned in point 1 in this section. So

$$q_{o,ji} = q_{ij,ay} + q_{ij,an}. \tag{2.7}$$

Here $q_{ij,ay}$ and $q_{ij,an}$ are accepted and not accepted data rates respectively. The reputation must be given on the basis of $q_{ij,ay}$ and not $q_{o,ji}$ as it is most likely the actual transfer rate. It is given by,

$$t_{ij} = \left(\frac{q_{a,ji}}{min(q_{ij,ay}, q_{f,ji})}\right)^{1-\eta_i} \times \frac{\hat{q}_{w,ji}}{q_{r,ji}}. \tag{2.8}$$

Here $q_{a,ji}$ is actual service rate i.e. the average rate at which receiver receives the data, $q_{f,ji}$ is feasible service rate (point 2) i.e. the rate at which the TCP Reno algorithm can get the throughput via underlying link with packet loss probability p. Here we are assuming that underlying transport layer is using TCP Reno algorithm. This rate can be computed using the equation (2.10) [46].

Equation 2.8 is the multiplication of two factors viz. $\frac{\hat{q}_{w,ji}}{q_{r,ji}}$ and $\left(\frac{q_{a,ji}}{min(q_{ij,ay},q_{f,ji})}\right)^{1-\eta_i}$. $\frac{\hat{q}_{w,ji}}{q_{r,ji}}$ i.e. the ratio of estimated willing service rate and requested service rate gives the value of trust as per willing service rate. Whereas $\left(\frac{q_{a,ji}}{min(q_{ij,ay},q_{f,ji})}\right)^{1-\eta_i}$ ensures that node is not playing a game by offering more data rate and not serving it. Value of η_i will decide the amount of penalty in case of cheating by the offering node. This means that node will be punished for offering more and actually providing less. The $\eta_i = 0$ means maximum punishment whereas $\eta_i = 1$ means that no punishment is given to the node for the difference in the offered and actual provisioning. The η_i can be decided by a node based on its experience with network. To compute the value of η_i, node i can take the exponential moving average of the ratio $\frac{q_{a,ji}}{min(q_{ij,ay},q_{f,ji})}$, i.e.,

$$\eta_i(n) = \frac{\beta_i(n-1) + \eta_i(n-1)}{2}. \tag{2.9}$$

Here $\beta_i(n) = (\sum_j \frac{q_{a,ji}(n)}{min(q_{ij,ay}(n), q_{f,ji}(n))}) / \sum_j 1$ and $\eta_i(1) = \beta_i(1)$. If node will not accept the offer, the value of $\frac{q_{a,ji}}{q_{ij,ay}}$ will be taken as a limiting case of 1 as both $q_{a,ji} = min(q_{ij,ay}, q_{f,ji}) = 0$.

$$q_{f,ji} \approx \left(\frac{W_{max}}{RTT}, \frac{1}{RTT\sqrt{\frac{2bp}{3}} + T_0 \cdot min\left(1, 3\sqrt{\frac{3bp}{8}}\right) p(1+32p^2)} \right). \tag{2.10}$$

Here $q_{f,ji}$ is the feasible service rate as a function of packet loss probability p; W_{max} is the maximum window opened by the receiver; RTT is the round trip time between the two nodes; T_0 is the time-out period and b is the number of packets acknowledged by a single acknowledgement.

Fig 2.1 shows the process starting from resource query to trust estimation. In this process when a node i needs a resource; it pushes a query in the network for resource. If node j has the asked resource, it replies to node i. Then requesting node requests to node j, the data rate it want. Node j replies to node i as per its shared capacity and request made by node i. After getting all such replies from different nodes like j, node i handshakes with the replying nodes as per their replies and its own download capacity. This process is followed by data transfer. After data transfer, requesting node estimates the willing service rate for each replying node and finally calculates the reputation of these nodes on the basis of actual transfer rate and willing service rate.

2.4 Estimation of Trust

Based on observed values of trust, exact trust value can be estimated using some estimator. We have used Best Linear Unbiased Estimator (BLUE) [4] for this purpose. Taking expectation in eqn (2.3)

$$\begin{aligned} E[Q_{o,ji}[n]] &= E[Q_{w,ji}] - E[\tilde{W}_{ji}[n]], \\ &= Q_{w,ji} - \tilde{W}_{ji}. \end{aligned} \tag{2.11}$$

so the scaled mean will be

$$S[n] = \frac{E[Q_{o,ji}[n]]}{\Theta} \tag{2.12}$$

where $\Theta = Q_{w,ji}$ is the parameter to be estimated

$$S[n] = \frac{Q_{w,ji} - \tilde{W}_{ji}}{Q_{w,ji}} = 1 - \frac{\tilde{W}_{ji}}{Q_{w,ji}}.$$

So

$$S = \left(1 - \frac{\tilde{W}_{ji}}{Q_{w,ji}}\right)\mathbf{1_{1\times M}}. \tag{2.13}$$

Here $\mathbf{1_{1\times M}}$ is a $1 \times M$ matrix of 1s i.e. $[111111.............1]^T$.

The covariance matrix

$$\mathbf{Cov} = \begin{bmatrix} \sigma^2 & 0 & & \\ 0 & \sigma^2 & 0 & \\ & \ddots & \ddots & \\ & & 0 & \sigma^2 \end{bmatrix}_{\mathbf{M\times M}}, \tag{2.14}$$

$$\mathbf{Cov} = \sigma^2 \mathbf{I_{M\times M}}. \tag{2.15}$$

As we know that the BLUE [4] is

$$\hat{Q}_{w,ji} = \frac{\mathbf{S^t Cov^{-1} Q_{o,ji}}}{\mathbf{S^t Cov^{-1} S}} \tag{2.16}$$

substituting the values,

$$\hat{Q}_{w,ji} = \frac{\left(1 - \frac{\tilde{W}_{ji}}{Q_{w,ji}}\right)\mathbf{1^t}\frac{1}{\sigma^2}\mathbf{I_{M\times M} Q_{o,ji}}}{\left(1 - \frac{\tilde{W}_{ji}}{Q_{w,ji}}\right)\mathbf{1^t}\frac{1}{\sigma^2}\mathbf{I_{M\times M}}\left(\mathbf{1} - \frac{\mathbf{\tilde{W}_{ji}}}{\mathbf{Q_{w,ji}}}\right)\mathbf{1}} \tag{2.17}$$

solving, we get

$$\hat{Q}_{w,ji} = \frac{\frac{1}{N}\sum_{k=n-M}^{n} Q_{o,ji}[k]}{1 - \frac{\tilde{W}_{ji}}{Q_{w,ji}}} = \frac{\overline{Q_{o,ji}[n]}}{1 - \frac{\tilde{W}_{ji}}{Q_{w,ji}}}$$

So, we can see that we need to compute the sample mean of all the samples and the ratio of noise mean and parameter mean.

This formulation have a problem that it is difficult to compute the sample mean when the number of samples is large. We can use moving average to estimate $\overline{Q_{o,ij[n]}}$ as,

$$\overline{Q_{o,ji}[n]} = \alpha \cdot Q_{o,ji}[n] + (1 - \alpha) \cdot \overline{Q_{o,ji}[n-1]}. \tag{2.18}$$

Here α depends on the rate of change of behaviour of nodes. Initially $\overline{Q_{o,ji}[1]} = Q_{o,ji}[1]$.

Value of $C_{ji,2}$ is estimated regularly on the basis of its download capacity and total requests made. Whereas value of $C_{ji,2}$ for complete network, is difficult to find. Nodes will gather the capacities shared and requests made by their neighbouring nodes. On the basis of this data, nodes will evaluate the value of $C_{ji,2}$ locally and then exchange this value with its neighbours. The values received from different neighbours will be averaged to get a better estimate of $C_{ji,2}$. This process will be done periodically.

As shown in section 2.2, estimation of trust has been done by a number of ways in literature. We are considering just one of these. But the uncertainties have not been generally considered in all of these. So the method similar to above can be modified for estimating trust from the observables.

2.5 Numerical Results

Performance of estimation method proposed in this chapter has been evaluated for 200 node network. We have considered the discrete time instants for the purpose of measurement and estimation in the simulations. Every slot is termed as an iteration. At the start of every slot every node queries for some resource and after getting the reply it requests for resource from the replying nodes. A node asks for the resource as per its download capacity from the nodes having the resource. Requested nodes allocate their bandwidth as per their reputation

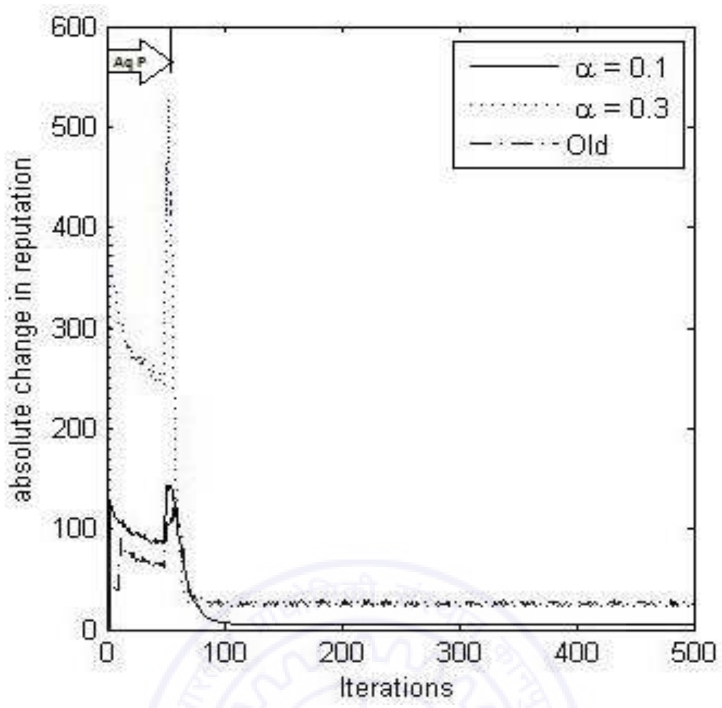

Figure 2.2: absolute change in reputation for homogeneous network

table in a probabilistic manner. Number of requests that a node will serve is fixed. For homogeneous network it is same for every node. Whereas, it is different in heterogeneous network. Finally requesting node and requested nodes will transact after the negotiation of resource transfer rate. At the end of slot each node updates its reputation table as per the quality of transaction. First 50 iterations have been taken as acquaintance period i.e. a node will allocate their bandwidth without referring to the reputation table.

We have plotted the absolute change in reputation estimation with increasing number of iterations up to 500. Here, absolute change in reputation (Δt) is

$$\Delta t(itr) = \sum_{i,j} |t_{i,j,itr} - t_{i,j,itr-1}|. \qquad (2.19)$$

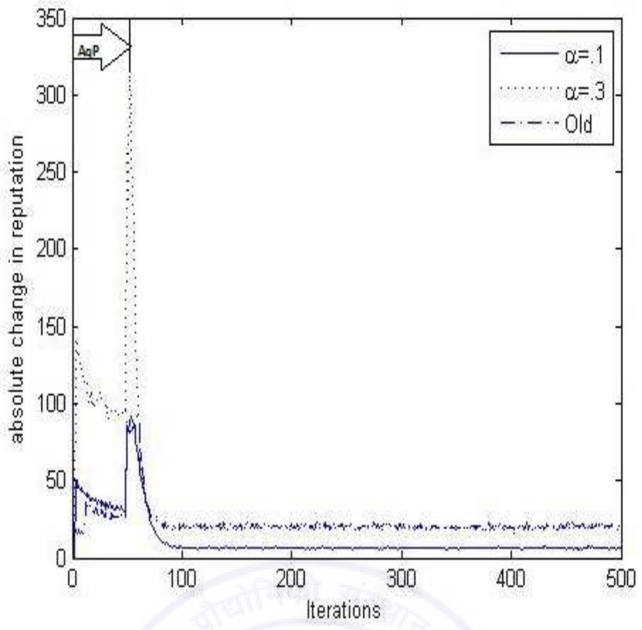

Figure 2.3: absolute change in reputation for heterogeneous network

Here, $t_{i,j,itr}$ and $t_{i,j,itr-1}$ are the reputation of node j after iteration itr and $itr - 1$ respectively, as estimated by node i. The N is total number of nodes.

In figure 2.2, Δt is plotted by taking the average of last ten measurements as proposed in [42] and according to our proposed reputation estimation method for homogeneous network. In figure 2.3, Δt has been plotted for the same methods but for heterogeneous networks. For the above simulations, we have considered 200 nodes once with α=0.1 and then with α=0.3. Homogeneous network means that all the nodes have same download capacity and are ready to provide resources to same number of nodes. Whereas by heterogeneous network, we mean that nodes have different download capacities and are ready to serve different number of nodes.

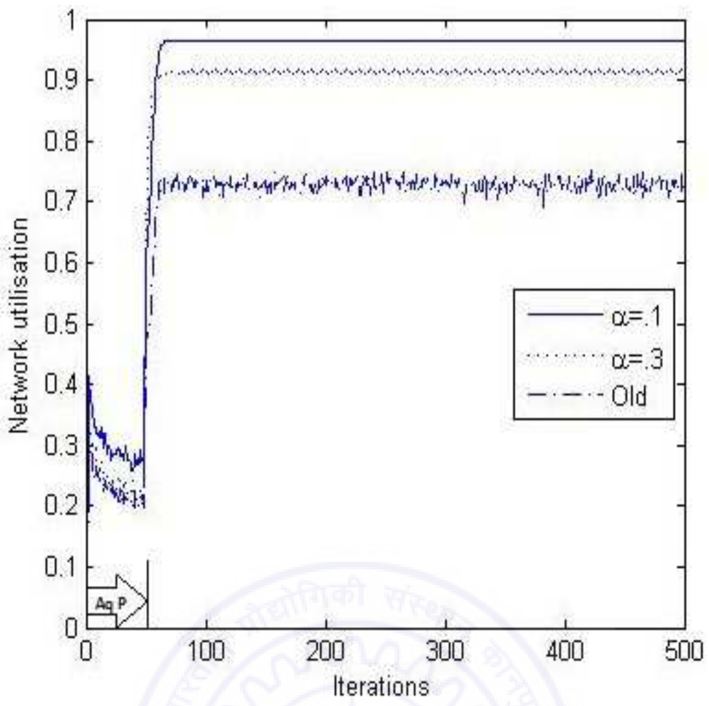

Figure 2.4: Network utilisation by a homogeneous network of 200 nodes

This is evident in both figure 2.2 and 2.3 that using estimator, the change in reputation is less. This implies that by use of proposed estimator, reputation can be estimated more accurately.

In figure 2.4 and 2.5, the utilisation level of shared resources in the network is plotted for homogeneous and heterogeneous networks of 200 nodes for $\alpha = 0.1$ and 0.3 respectively.

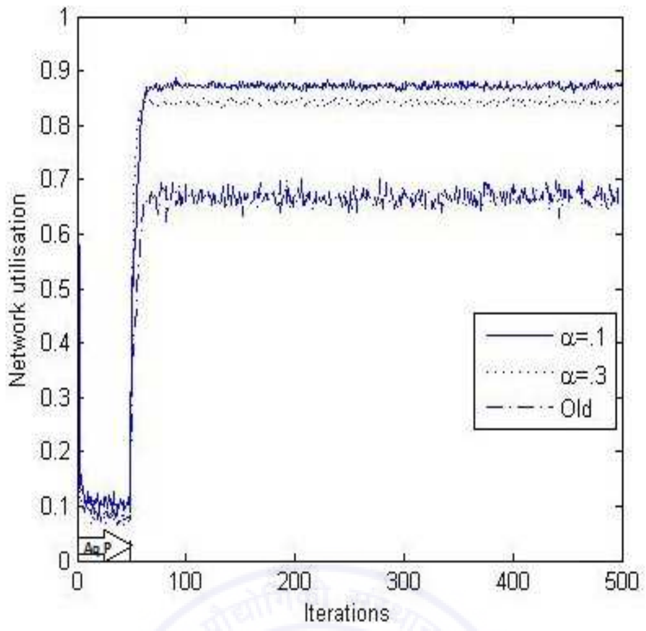

Figure 2.5: Network utilisation by a heterogeneous network of 200 nodes

2.6 Conclusion

In peer-to-peer networks, free riding is a major problem that can be overcome by using reputation management system. In this chapter we have proposed an estimation technique using BLUE.

The proposed trust estimator considers uncertainties in the trust estimation. The absolute change in trust values in the trust table of nodes is considerably less than the older techniques. It implies that reputation can be estimated more accurately using this estimator. The better estimation of reputation will lead to a better counter measures against free riding in a peer-to-peer system.

Chapter 3

Reputation Aggregation in Peer-to-Peer Network

3.1 Introduction

Good methods for trust aggregation need to be designed for an effective reputation management system. Aggregation of trust generally consumes a lot of time and memory. It becomes even more difficult when number of nodes is large. Moreover, existing methods assume that the reputation of a peer must have a global value, i.e. peers behave uniformly with all other peers, but this is not true at least in the case of selfish nodes. In this work, the trust estimated by a node directly, trust reported by neighbours and trust averaged using a variation of gossip algorithm [47] are aggregated to make trust vector at each node.

This algorithm has been simulated to work with networks having power law degree distribution, i.e. networks formed by Preferential Attachment (PA) model. The PA graphs are introduced in [48] and formally defined in [49]. According to

[49], G_N^m is converted from G_{N-1}^m when a new node joins the network with m edges. The joining node chooses the node i with probability P_i. Here
$P_i = \frac{degree\ of\ node\ i\ before\ this\ connection\ is\ made}{sum\ of\ degree\ of\ all\ the\ nodes\ before\ this\ connection\ is\ made}$.

Remainder of this chapter is organised as follows. Section 3.2 discusses the related work in reputation management. Section 3.3 describes the system model. Section 3.4, proposes differential gossip trust algorithms for aggregation of trust. Section 3.5 presents the analysis and numerical results. Section 3.6 concludes the chapter.

3.2 Related Work

Many methods [21, 20, 25, 23, 22] have been proposed for reputation aggregation in the literature. Eigen-Trust [21] depends largely on pre-trusted peers i.e. peers that are globally trusted, this is scalable to a limited extent. Peer-Trust [20] stores the trust data (i.e. trust values of all the peers in the network) in a distributed fashion. This is performed using a trust manager at every node. In Peer-Trust, hash value of a node id is calculated to identify the peer where the trust value of node will be stored. Song *et.al.* [25] used fuzzy inference to compute the aggregation weights. In Fuzzy-Trust, each peer maintains the local trust value and transaction history of the remote peer. At the time of aggregation each peer asks for the trust value from the qualified peers and combines the received values and locally existing values to compute updated trust values. Power trust [23] depends largely on power nodes. Power nodes are few top reputation nodes in the network. It uses score manager like trust manager in Peer Trust and Look ahead Random Walk for aggregation. Gossip Trust [22] uses gossip algorithm for aggregation. Generally earlier works [21, 20, 23, 22] assume that reputation of a peer must have a global value, i.e. peers behave uniformly with all peers. But this is not the case due to nodes being selfish in nature. A peer behaves with different decency levels with different peers. The same holds for the opinion as well, i.e. peer gives different weights to opinions of different peers. These aspects have been taken into account in our algorithm called Differential Gossip Trust. The current work combines local, reported value from trusted neighbours and global trust using a new variation of gossip to define a novel trust aggregation algorithm.

3.3 System Model

In this work we are studying a peer-to-peer network. Typically there will be millions of nodes in a peer-to-peer network. These nodes are connected by a network graph generated by *PA process* G_N^m [50] for $m \geq 2$. Generally nodes will have small number of neighbours. Here by neighbour, we mean, that these nodes' addresses is stored with the node.

There is no dedicated server in this network. Peers in this network are rational, i.e. they are only interested in their own welfare. They are connected to each other by an access link followed by a back bone link and then again by an access link to the second node. We are assuming that the network is heavily loaded i.e. every peer has sufficient number of pending download requests, hence these peers are contending for the available transmission capacity. We also assume that every peer is paying the cost of access link as per the use. So, every peer wants to maximise it's download and minimise it's upload so that it can get maximum utility of it's spending and this leads to problem of free riding.

If a node is downloading, some other node has to upload. So the desired condition is that the download should be equal to upload for a node. Usually this means that there is no gain. Even in this scenario the node gains due to interaction with others, as the chance of survival increases. Thus, interaction itself is an incentive. A node will usually try to get the content and avoid uploading it as this maximises gain for it. Thus free riding becomes optimal strategy. So a reputation management system need to be enforced to safeguard the interest of every node by controlling the free riding.

In a reputation management system, every node maintains a reputation table. In this table, the node maintains the reputations of the nodes with whom it has interacted. Whenever it receives a resource from some node, it adjusts the reputation of that node accordingly. When another node asks for the resource from this node, it checks the reputation table and according to the reputation value of the requesting node, it allocates resource to the other node. This ensures that every node is facilitated from the network as per its contribution to the network and consequently free riding is discouraged.

For using such a reputation management system, the nodes need to estimate the trust value of the nodes interacting with them. There are number of ways to estimate reputation [51]. We assume that trust value observed by node i for node j can be defined as t_{ij}.

3.4 Aggregation of Trust

Whenever a node needs a resource, it asks from its neighbours; if they have the resource, the node gets the answer of its query. If neighbours do not have it, they forward the query to their neighbours and so on. The node that have the resource, replies back to the requesting node. The requesting node now asks for the resource from the node having the resource. The answering node provides the resource now directly according to the reputation of the node.

If a node receives a request from another node that is not its neighbour, the reputation of that node needs to be estimated some how in order to decide the quality of service to be provisioned. If two nodes are going to transact for the first time they should have reputation of each other with them. This can be done by getting the reputation of node from neighbours and then using it to make an initial estimate. When for a node, multiple trust values are received, we need an aggregation mechanism to get the trust value. Trust value should always lie in between zero and one.

For the whole network, we can define a trust matrix **t**, a matrix of dimensions $N \times N$. Here t_{ij} represents the trust value of j as maintained by i based on direct interaction. This matrix is generally sparse in nature as generally a node will have very small number of neighbours as compared to total number of nodes in the network. It may be noted that t_{ij} is estimated based on transaction between nodes i and j and can be called as local trust value. These trust values will be propagated and aggregated by all the nodes in the network. The trust estimate which should be actually used will be based on aggregation of local trust values and trust estimates received from neighbours. For the reputation information received from direct neighbours, the weights can be assigned based on neighbours' reputation.

3.4.1 Differential Gossip Trust

We have modified the gossip based information diffusion algorithm to form a innovative differential gossip information diffusion algorithm. It allows faster diffusion of trust values to enable faster estimation of global trust vectors at all the nodes. The algorithm can be divided into two parts. In first part, we will discuss about the method of information diffusion whereas in second part, we will discuss about the information that is to be diffused.

3.4.1.1 Differential Gossip Algorithm

Gossip Algorithms are used for information spreading in large decentralised networks. These algorithms are random in nature as in these algorithms, nodes randomly choose their communication partner in each information diffusion step. These algorithms are generally simple, light weight and robust for errors. They have less overhead compared to deterministic algorithms [47], [52]. Gossip algorithms are also used for distributed computation like taking average of numbers stored at different nodes. These algorithms are suitable for computation of reputation vector in peer-to-peer networks [22].

There are three types of gossip algorithms: push, pull and push-pull. In push kind of algorithms, in every gossip step nodes randomly choose a node among its neighbours and push their information to it. Whereas in pull algorithms, nodes take the information from the randomly selected neighbouring nodes. Both these processes happen simultaneously in the push-pull based algorithms.

In [22], push gossip algorithm as given in [47] has been used. This algorithm considers that network is a complete graph but in reality peer-to-peer networks will be generally based on *Preferential Attachment*(PA) model [49], [50]. In networks based on PA model, the nodes can be divided into two types [53] viz. power nodes and low degree nodes. Power nodes are the nodes that have high degree (of order of N^ϵ). The degree of low degree node is of the order of $log_2(N)$. Low degree nodes form a linear sub graph within the network with diameter of the order of $log_2(N)$. Here N is the number of nodes in the network, and ϵ is a constant greater than zero.

Chierichetti *et.al.* [53] stated that in PA model based networks, push or pull alone cannot spread the information with a reasonably fast speed. If push model is implemented and the information is with a power node, it will take many rounds in pushing information to low degree nodes. If pull model is being used, and the information is with low degree node, it will again take many rounds for a power nodes to pull the information. This phenomenon will be more evident in average computation using push gossip algorithm or pull gossip algorithm as information with every node needs to be distributed to every other node.

To avoid this problem we propose differential push gossip algorithm. In this algorithm, every node makes different number of pushes in a single gossiping step. Here, we make three assumptions, i) every node has a unique identification number known to every other node. So, if some node pushes some information about some other node, receiving node knows that this information is about which particular node; ii) time is discrete; and iii) every node knows about the starting time of gossip process.

All nodes that have some feedback about a single node, gossip their feedback about that single node. All the nodes estimate the global reputation of the single node based on the outcome of gossip. Let the feedback about the j^{th} node by node i be y_{ij}. If it does not have any feed back about j it keeps the value of y_{ij} as 0. Every node that have feed back about node j assumes the gossip weight g_{ij} as 1 and rest of the nodes assume the gossip weight as zero. It is done so because as a result of gossip every node coverages to the ratio of summation of all gossiped values and summation of all gossip weight, i.e. $\frac{\sum_i y_{ij}}{\sum_i g_{ij}}$. If only one node will assume gossip weight as 1 and rest will assume as 0, it is evident that nodes will converge to summation of gossiped values.

Every node also pushes its degree to the neighbouring nodes. Thus all the nodes can calculate the average degree of their neighbours. Every node has y_{ij} and g_{ij} as information to be gossiped. Let us call this pair as gossip pair. The ratio of gossip pair is tracked in every step to decide on convergence. Every node, first calculates the ratio k of its degree and average degree of its neighbours. As k will be a real number hence it is rounded off to nearest integer if $k \geq 1$. For all other cases $k = 1$. The node chooses k nodes randomly in its neighbourhood and sends $(\frac{1}{k+1} y_{ij}, \frac{1}{k+1} g_{ij})$ as gossip pair to all randomly selected k nodes and itself. In this algorithm every node receives contribution from other nodes. In this process node starts with no value and in every step receives contribution from some new nodes and once it receives contribution from all the nodes in the network, it gets the aggregated value. By pushing to itself, node is retaining the contributions that it has received till the start of current round. The node is also considered as one of the neighbour of itself.

After receiving all gossip pairs from different nodes including itself, the node sums up all the pairs. This summation now becomes new gossip pair. The ratio of this gossip pair is the value that node has evolved in this step. If at least one gossip pair has been received from a node other than itself, the condition of convergence will be checked between the ratios of this step and last step with a predefined error constant. If convergence condition is satisfied, it means a node need not run the gossip process any more for its convergence. But once convergence is achieved by a particular node, convergence of other nodes is not assured. Hence, if a node will stop gossiping, convergence of its neighbours may suffer. To avoid this problem, once a node gets converged, it will announce among all its neighbours that it has achieved convergence. Every neighbour will note this announcement. When a node will achieve convergence and listens about the convergence of all of its neighbours it will stop gossiping.

When a round of gossiping starts it takes some time to complete. After gossiping, nodes get a value that is used till the next new value is converged upon at the end of next round. After the end of a round, next round of gossip will start after some time. The time difference between two rounds will depend upon the change in behaviour of nodes in the network and number of new nodes coming in the network. For simplicity, this time difference has been taken as constant. In reality, this should be dynamically adjusted.

3.4.1.2 Differential Reputation Aggregation

When some node *j* requests resource from some node *i*, node *i* needs the reputation of node *j* so that it can decide the quality of service to be offered to node *j*. There are two possible conditions between node *i* and node *j*. First, node *j* may have served node *i* earlier and hence node *i* has some trust value about node *j*. In this case there is no problem for node *i* and node *i* will serve as per the reputation available with it. Second, node *i* and node *j* are unknown to each other. In this case node *i* needs general reputation about node *j*. For this, aggregation of reputation is needed.

Aggregation of reputation should not be resource intensive and free from collusion and whitewashing. We can have two kind of options for this. First by gossiping, all the nodes can reach on consensus about the reputation of *j* [22]. Second all the nodes exchange their reputation tables about the nodes they have interacted with and this process should always be running, like executing a routing protocol at network layer. First process is more vulnerable to collusion where as second process is resource intensive. As we can see that unstructured peer-to-peer network is very similar to human network. So observing human network and identifying a solution from it will be a better idea. In human network when we need the reputation value of some body we rely on personal experience with him. If we don't have any personal experience with him, we rely on two things. First the general perception about him which we receive from gossip flowing around and second the information given by our friends if they have any direct interaction with him. We combine these two and act accordingly.

Nodes follow the same kind of approach, in our proposed algorithm. The nodes gather opinion of their neighbours and combine it with the opinion, obtained from general gossip after weighing neighbours opinion according to confidence in neighbours. In general it can be said that a node gives weight to every node in the network. The nodes that have not interacted with it are given weight as 1 where as those which have interacted are given weight according to the confidence in them (always ≥ 1).

Let us consider that there are N nodes in the network. Every node periodically calculates the trust value of other node on the basis of quality of service provided by that node against the requests made. Let us assume that t_{ij} is the trust value measured by node i for node j. Here t_{ij} $(1 \leq i, j \leq N)$ will always lie between 0 and 1 such that $t_{ij} = 1$ will represent complete trust in node j whereas $t_{ij} = 0$ will represent no trust in node j.

If a node has not transacted with a node its trust value will also remain 0 with that node. This value is taken as 0 to avoid the white washing attack. Initially this value can be taken higher than zero and can be dynamically adjusted thereafter as per the level of whitewashing in the network. In this work, we have not studied this aspect.

We will discuss the algorithm in four different variations. In the first variation, global reputation aggregation for a single node will be discussed. In the second variation, we will discuss globally calibrated local reputation aggregation for this single node. In third variation, global reputation aggregation for all the nodes simultaneously will be discussed, and finally in fourth variation globally calibrated local reputation aggregation for all the nodes simultaneously will be discussed.

In the first variation to keep things simple we are assuming that weights of all nodes for every node are 1. This leads us to calculation of global reputation ($\mathbf{R_{global}}$) of a node. This can be equivalently represented using matrix vector multiplication as follows,

$$\mathbf{R_{global}}(n+1) = \frac{1}{N}(\mathbf{t}^{\mathbf{T}}(n) \times \mathbf{1_{N\times1}}) \tag{3.1}$$

Here n is the time instant and r_i is global reputation of i^{th} node and is also i^{th} element in $\mathbf{R_{global}(n+1)}$ column vector. $\mathbf{1_{N\times1}}$ is a vector of N $1's$, i.e. $[1111.......]^T$. Differential Gossip algorithm is used for doing this computation in distributed fashion. This process is shown in algorithm 1. Although each node considers the average of feedback from every node in the network, it is desirable to assign different weights to the direct feedback received from neighbour nodes. The direct feedback from a node is based on the direct interaction which it has experienced. The weights can be assigned by a node on the basis of number and quality of transactions made with the node providing the feedback. The trust value of a node is a good representation of quality and number of transactions. The weights for different nodes can be derived on the basis of the trust values of these nodes. Same idea is used in second variation of algorithm where nodes are calculating globally calibrated local reputation vector. So in second variations, we propose the weight w_{ij} to be of the form:

$$w_{ij} = a_i^{b_{ij}\cdot t_{ij}}. \tag{3.2}$$

Here a_i and b_{ij} are two parameters that a node can decide on its own. First

Algorithm 1 Global Reputation aggregation for a single node

```
Require: $t_{ij}$ (The reputation estimated by node $i$ for node $j$ only on the basis of
  direct interaction) for $1 \le i \le N$, gossip error tolerance $\xi$
Ensure: Global Reputation of node $j$ ($R_j$)
  if $i$ has some reputation value about $j$ then
    Assume weight $g_{ij} = 1$, and $y_{ij} = t_{ij}$
  else
    Assume weight $g_{ij} = 0$, and $y_{ij} = 0$
  end if
  Push self degree to neighbouring node
  Take the average of neighbour degree
  Calculate the ratio of its degree and average of neighbour degree ($k_i \leftarrow \frac{degree\ of\ i}{average\ neighbour\ degree}$)
  Round off $k_i$ to nearest integer for $k_i \ge 1$ else take $k_i = 1$
  $m \leftarrow 1$ {Initialise Gossip Step}
  $u \leftarrow \frac{y_{ij}}{g_{ij}}$ for nodes having $g_{ij} \ne 0$.
  repeat
    ($y_{rj}, g_{rj}$) are all pairs (of gossip weight and gossip value) received by the node
    $i$ in the previous step
    $y_{ij} \leftarrow \sum_{r \in R} y_{rj}$; $g_{ij} \leftarrow \sum_{r \in R} g_{rj}$ {update gossip pairs}
    {R is the set of nodes sending the gossip to $i$}
    choose $k_i$ random nodes in its neighbourhood
    send gossip pair ($\frac{1}{k_i+1} y_{ij}, \frac{1}{k_i+1} g_{ij}$) to all $k_i$ nodes and also to itself
    $m \leftarrow m + 1$ {increment the gossip step}
    if $|R| > 1$ then
      if $|\frac{y_{ij}}{g_{ij}} - u| \le \xi$ then
        Inform all neighbours about self convergence
      end if
    end if
    $u \leftarrow \frac{y_{ij}}{g_{ij}}$
  until Self convergence and all neighbours' convergence has happened
  output $\frac{y_{ij}}{g_{ij}}$
```

parameter can be adjusted according to the overall quality of service received by the node from the network, whereas second parameter can be adjusted according to the recommendation of a particular neighbour and quality of service from the network. So the second parameter will be adjusted for every neighbour independently. For this work, a_i and b_{ij} has been taken as constant for every node for simplicity. This form of weight provides higher weight to the opinion of nodes that have good relation with aggregating node. The level of discrimination among the opinions are decided with the value of a_i and b_{ij}. These values are proposed to be decided by the node on the basis of its experience. These kinds of weights also ensure that every nodes opinion is taken into account with some weight.Salient features of this scheme are as follows.

- Even if a node has no neighbourhood relation with the estimating node, its feedback will still get some consideration.
- If a node has bad reputation with estimating node, its feedback will have weight close to the node which have no neighbourhood relation with the estimating node.
- Nodes with higher reputation will be given higher weights and it will help in making better quality of service groups.
- Values of a_i and b_{ij} can be dynamically adjusted by nodes as per their requirement. Though in this work, a_i and b_{ij} have been taken as constants.
- Collusion will be significantly reduced.

A weighted trust matrix, that is different at every node, is formed by multiplying trust values with weights i.e. for node I, the ij element in weighted matrix will be W_{Iij} such that,

$$W_{Iij} = w_{Ii} \times t_{ij}. \tag{3.3}$$

A node gives high weight to the feedback given by nodes according to the quality of service provided by them. This leads to the calculation of globally calibrated local reputation vector. It is a collection of the reputations of all the nodes in the network according to received feedback about the node under consideration and the weight of node giving the feedback to calculating node. It means if some node I is calculating globally calibrated local reputation vector, the j^{th} element of this vector will be

$$Rep_{I,j} = \frac{\sum_i W_{Iij}}{\sum_i w_{Ii}}. \tag{3.4}$$

Now globally calibrated local reputation at node I, $\mathbf{R_I}$ can be equivalently represented as the matrix vector multiplication

$$\mathbf{R_I}(n+1) = \frac{1}{S_I}(\mathbf{W_I^T}(n) \times \mathbf{1_{N\times 1}}). \tag{3.5}$$

Here $S_I = \sum_i w_{Ii}$. It is interesting to see that if we consider the weights of all nodes as 1 in (3.5), this equation degenerates to (3.1).

Each node will have four different kind of data about other nodes - first t_{ij} i.e. the trust value as result of direct interaction, second y_{ij}, the intermediate variable for gossiping and third g_{ij}, gossiping weight. In the start of every gossiping round y_{ij} assume the value of t_{ij} whereas g_{ij} will be 1 only for one value of i, and 0 for all others. This will happen for all i and j. Fourth value Rep_{ij} is obtained after gossiping and consideration of neighbours opinion. It will also be maintained at every node. Apart from these four entities, we also wants to count the total number of nodes opining about node j. For this purpose every node that have opined about node j will assume $count_{ij} = 1$ and others will assume $count_{ij} = 0$ and hence in process of gossip these all 1′s will sum up and we will get the count.

Equation (3.4) can be alternatively represented as,

$$\begin{aligned} Rep_{I,j} &= \frac{\sum\limits_{i \in NS_i} W_{Iij} + \sum\limits_{i \notin NS_i} W_{Iij}}{\sum\limits_{i \in NS_i} w_{Ii} + \sum\limits_{i \notin NS_i} w_{Ii}}, \\ &= \frac{\sum\limits_{i \in NS_i} w_{Ii} \times t_{ij} + \sum\limits_{i \notin NS_i} w_{Ii} \times t_{ij}}{\sum\limits_{i \in NS_i} w_{Ii} + \sum\limits_{i \notin NS_i} w_{Ii}}, \\ &= \frac{\sum\limits_{i \in NS_i} (w_{Ii} - 1) \times t_{ij} + \sum\limits_{i \notin NS_i} (w_{Ii} - 1) \times t_{ij} + \sum\limits_{i} t_{ij}}{\sum\limits_{i \in NS_i} (w_{Ii} - 1) + \sum\limits_{i \notin NS_i} (w_{Ii} - 1) + \sum\limits_{i} 1}. \end{aligned}$$

Here NS_i is the set of neighbours of node i. As neighbourhood between two nodes is based upon the interaction between them so for non neighbour nodes the weight will be 1. Using this fact,

$$Rep_{I,i} = \frac{\sum\limits_{i \in NS_i} (w_{Ii} - 1) \times t_{ij} + \sum\limits_{i} t_{ij}}{\sum\limits_{i \in NS_i} (w_{Ii} - 1) + \sum\limits_{i} 1}. \tag{3.6}$$

As we want to compute at each node, the globally calibrated local reputation of a node j, the reputation of node j from its neighbours on the basis of direct interaction will be needed. Whereas in the gossip algorithm, after every step, the value at the node keeps on changing as it gets added to the values pushed by other nodes and is distributed after division to neighbours. So after few steps the values converge when incoming and outgoing values statistically balance each other. Hence after first step of gossiping its difficult to get the value of t_{ij} from a neighbour for the nodes with whom it has direct interaction, by gossiping process.

If a node is participating in the process of gossip for the first time about node j or reputation of node j at this node has changed considerably since start of previous round of gossip, this node will inform the reputation to all of its neighbours before the start of next gossiping round (figure 3.1). This will be done by all the nodes. After this process every node has opinion of its neighbours about node j (If a node does not inform reputation of j, the already available earlier value will be considered). If node will not hear from a node since long, it will assume that this node is no longer present and hence it will drop its feedback after some time.

Now these reputations will be multiplied by $(W_{Ii} - 1)$ as required in equation (3.6) and summed up as value $\hat{y}_{Ij}$. Now normal gossip will be done as in algorithm 1 with a difference that only one node will be given gossip weight 1 and rest will be given 0 gossip weight and the nodes that have reputation information about node under consideration will also push 1 so that these nodes can be counted. This will lead to the summation of all reputation values available and total number of nodes giving these reputation values. Now reputation can be calculated using (3.6)[algorithm 2].

In third variation we want to aggregate the global reputation of all nodes simultaneously. This algorithm is quite similar to algorithm 1 except few changes. Unlike algorithm 1, node will push complete vector $\mathbf{y_i}$ which consists of feedback from node about all other nodes it has transacted with. Similarly, instead of single gossip weight g_{ij}, node will send vector $\mathbf{g_i}$. A node id will also be attached with

Algorithm 2 Globally calibrated local Reputation aggregation for a single node

```
Require: Feedback matrix t, gossip error tolerance $\xi$.
Ensure: Globally calibrated local Reputation of node $j$ ($r_{ij}$)
  Assume weight $g_1 = 1$
  Node $i$ do
  if $i \neq 1$ then
    $g_{ij} = 0$
  end if
  if $i$ has some reputation value about $j$ then
    take $count_{ij} = 1$, and $y_{ij} = t_{ij}$
  else
    take $count_{ij} = 0$, and $y_{ij} = 0$
  end if
  calculate $w_j$ for all neighbours of $i$ by formula $w_{ij} = a_i^{b_{ij} t_{ij}}$
  if Node $i$ is participating first time in gossiping process then
    Push feedback due to direct interaction about the node under consideration
    to all neighbours
  else
    if Feedback about the node under consideration has changed by more than
    some constant $\Delta$ then
      Push the new feedback to all the neighbours
    end if
  end if
  Push self degree $d_i$ to neighbouring nodes
  for k=1 to $d_i$ do
    Calculate $\hat{y}_{ij} \leftarrow (w_k - 1) \times feedback\ from\ node\ k\ about\ j$
  end for
  Take the average of neighbour degree
  Calculate the ratio of its degree and average of neighbour degree ($k_i \leftarrow \frac{degree\ of\ i}{average\ neighbour\ degree}$)
  Round off $k_i$ for $k_i \geq 1$ else take $k_i = 1$
  End do
  $m \leftarrow 1$ and $u \leftarrow \frac{y_{ij}}{g_{ij}}$ {Initialise Gossip Step}
  Algorithm Continued...
```

Algorithm Algorithm 2 (continued)

```
repeat
  Do for node $i$
  $u \leftarrow \frac{y_{ij}}{g_{ij}}$ and $fcount \leftarrow \frac{count_i}{g_{ij}}$
  $(y_{rj}, g_{rj}, count_{rj})$ are all pairs of gossip weight, gossip value for reputation and count received by node $i$ in the previous step
  $y_{ij} \leftarrow \sum_{r \in R} y_{rj}; g_{ij} \leftarrow \sum_{r \in R} g_{rj}; count_{ij} \leftarrow \sum_{r \in R} count_{rj}$ {update gossip pairs}
  choose $k_i$ random nodes in its neighbourhood
  send gossip pair $(\frac{1}{k_i+1} y_{ij}, \frac{1}{k_i+1} g_{ij})$ to all $k_i$ nodes and itself
  $m \leftarrow m + 1$ {increment the gossip step}
  if $|R| > 1$ then
    if $|\frac{y_{ij}}{g_{ij}} - u| \leq \xi$ then
      Inform all neighbours about self convergence
    end if
  end if
  $u \leftarrow \frac{y_{ij}}{g_{ij}}$
until Self convergence and all neighbours' convergence has happened
$Rep_{ij} \leftarrow \frac{\hat{y}_{ij} + \frac{y_{ij}}{g_{ij}}}{\sum(w_k - 1) + \frac{count_i}{g_{ij}}}$
output $Rep_{ij}$
```

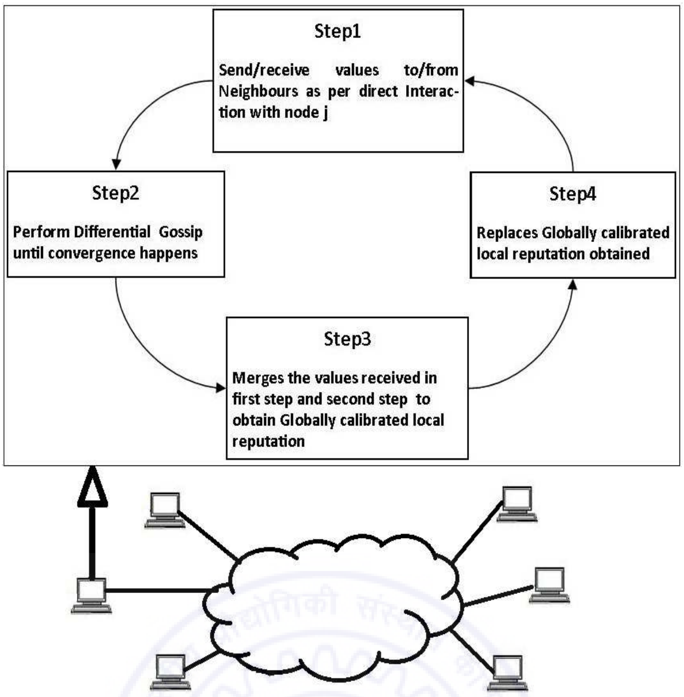
Step1
Send/receive values to/from Neighbours as per direct Interaction with node j
Step2
Perform Differential Gossip until convergence happens
Step3
Merges the values received in first step and second step to obtain Globally calibrated local reputation
Step4
Replaces Globally calibrated local reputation obtained

Figure 3.1: Process Diagram for computation of Globally Calibrated Local Reputation

every pair of y_{ij} and g_{ij} so that receiving node can distinguish among gossip pairs. So, in fact, node pushes gossip trio consisting of y_{ij}, g_{ij} and node id. The convergence of algorithm is checked by the following condition:

$$\sum_j \left| \frac{y_{ij}(n)}{g_{ij}(n)} - \frac{y_{ij}(n-1)}{g_{ij}(n-1)} \right| \leq N\xi \tag{3.7}$$

Where n is the time instant and ξ is the permissible error bound.

In the fourth variation we want to aggregate the globally calibrated local reputation of all nodes simultaneously. This algorithm is quite similar to second variation expect that we will use the third variation for gossiping process. Moreover in this variation nodes will push full vector $\mathbf{t_i}$, in place of only t_{ij} for node *j*.

3.5 Analysis of Algorithm

3.5.1 Analysis of convergence of Gossip Algorithm

In this section we will study the time needed by nodes to converge the average of local direct estimate values at different nodes. First, we will study the spreading of gossip in power law network. Then, we will study the diffusion speed of gossip. Based on these results, we will find the time of convergence.

Chierichetti *et.al.* [53] proved that in PA based graph, $\{G_N^m\}$ for $m \geq 2$ alone push or alone pull will fail for spreading of gossip. They also proved that push-pull will succeed in $O((log_2 N)^2)$ where low degree nodes push information to low degree nodes and power nodes and pull information from power node.

But in a peer-to-peer networks its difficult to identify the power nodes. Moreover pulling the information is more expensive the pushing the information. So we have proposed to use differential push gossip in place push pull gossip. In the fallowing theorem we are proving that differential push gossip will take same time as push pull gossip.

Theorem 3.5.1. *Gossip will spread with high probability in a PA based graph, $\{G_N^m\}$ for $m \geq 2$, within $O((log_2 N)^2)$ time using differential-push (by high probability we mean, $1 - o(1)$, where $o(1)$ goes to zero as N increases).*

Proof. Differential push means that every node will push its data to different number (k_i) of nodes instead of one node. Here k_i is the ratio of nodes degree and average degree of all its neighbours.

Nodes in $\{G_N^m\}$ can be classified into three types [53]. First type is high degree nodes (i.e. power nodes); let us call this set of nodes as H. Second type is low degree nodes (with degree $O(log_2 N)$) which joined in the first half of the process, let us call this type as W and third type is low degree nodes which joined after W; lets call this type as V.

Let us call the set $\{G_N^m\} \setminus H$ as H′. H′ may be thought as the combination of few finite connected components that are further connected with each other using some node(s) from H.

Members of set H′ are low degree nodes with degree of the order of $log_2 N$ and diameter of each connected component will also be of the order of $log_2 N$ [53]. It can be seen that if gossip originates in one of these components, it will spread in that component easily within $O((log_2 N)^2)$ [**?**] if normal push is used. In case of differential push, it will be even faster.

Reaching from one component to another component using normal push may take longer time as now it has to be pushed by a node from H having high degree. Hence normal push will take longer time in spreading the gossip as high degree nodes will also be making only one push to a node chosen uniformly at random in one time slot [53].

Hence if a high degree node, i.e. the member of H makes pushes to 'k' nodes chosen uniformly at random in one time slot, where k may be the ratio of the degree of high degree node to the degree of nodes from H′, information will also spread in H within $O((log_2 N)^2)$ steps.

Once information is reached to every node of H, it will reach to remaining components of H′ and spread there in another $O((log_2 N)^2)$ steps. Therefore, it can be seen that in a PA graph, information will spread within $O((log_2 N)^2)$ steps to all nodes with high probability.

There are very few nodes which belong to H. Hence the nodes from set H will have neighbours from H′ with high probability. Therefore k can be approximated as the ratio of degree of node to the average degree of its neighbours.

□

In our case few nodes have information (reputation of a node) that has to be averaged and this average has to be spread to all nodes. So we will prove the convergence for the case where every node has information that has to be averaged. Our case is a special case of this case.

Let at $n = 0$, each node has a number. For *jth* node this number is $d_{0,j}$. So the objective of gossip is to have $\frac{\sum_j d_{0,j}}{N} = d_{avg}$ at each node after some rounds of gossip. The number of rounds needed should be least possible. For $n = 0$, the gossip weight at each node will be unity. After n steps, let the node j have the evolved number as $d_{n,j}$ and evolved gossip weight as $g_{n,j}$.

To study the time taken in the convergence of algorithm we assume that each node j maintains a vector $\mathbf{c_j^m}$. The dimension of this vector will be $1 \times N$. This vector will record the contribution received from every node including itself about the node m. So initially at $n = 0$ each node will have a vector in which $(N - 1)$ elements will be zero and one element that is for itself will be one. If in the process of first step of gossiping, only the node i chooses node j for pushing the gossip about m, then the contribution by i to j i.e. $c^m_{n,i,j}$ will be recorded in the i^{th} element of contribution vector of node j. So after first step of gossip the contribution vector of j will contain two non-zero elements one received from i and one pushed to itself. We assume here that only push has been received by node j. In case of l pushes being received the vector will have $l + 1$ non zero entries. Now node j will choose some node o. Node j will push the complete contribution divided by $p + 1$(if p push gossip is under consideration) to node o. Now node o will do vector addition of all received vectors including the one received from itself. The resultant vector will be the new contribution vector. This process will be repeated at all the nodes.

So it can be said that $d^m_{n,j} = \sum_i c^m_{n,i,j} \cdot d^m_{0,i}$ such that $g^m_{n,j} = \sum_i c^m_{n,i,j}$. When a node will receive same amount of contribution from all the node at that time the ratio of evolved number ($d^m_{n,j}$) and evolved gossip weight ($g^m_{n,j}$) will be the average of all the numbers.

Theorem 3.5.2. *Uniform Gossip diffuses with differential push in PA based graph within* $O((log_2 N)^2 + log_2 \frac{1}{\xi})$ *time with high probability such that contributions at all nodes will be* ξ *uniform after this amount of time, i.e.* $max_i|\frac{c^m_{n,i,j}}{||\mathbf{c^m_{n,j}}||_1} - \frac{1}{N}| \leq \xi \ \forall j$ *where* $||\mathbf{c_{n,j}}||_1 = \sum_{\mathbf{i}} \mathbf{c_{n,i,j}}$

We can see the property of *mass conservation* (proposition 3.5.3) [47] holds in this case as well.

Proposition 3.5.3. *Under the differential Push protocol with Uniform Gossip, the sum of all of i^{th} node's contributions at all nodes j is $\sum_j c^m_{n,i,j} = 1$ and hence the sum of all weights is $\sum_j g^m_{n,j} = N$*

proof of theorem 3.5.2. As we know when a node will get equal contribution from every node it will reach the average value. Taking variance around the mean value of the contributions from all the nodes at a particular node will give the level of convergence at one node. If we sum these variances for all the nodes, we will get the idea about the convergence of network. We are just referring to reputation of node m in this proof, and super script m have not been explicitly shown. It means $c^m_{nij} = c_{nij}$ and $g^m_{nj} = g_{nj}$. The variance at node j will be $E(c_{n,i,j} - \overline{c_{n,i,j}})^2$ i.e. $\frac{1}{N}\sum_i (c_{n,i,j} - \frac{g_{n,j}}{N})^2$. As this quantity is small, we can drop $\frac{1}{N}$. This will still give the idea about the convergence. Further adding the variance at all the nodes gives us the idea about further convergence in the whole network. We call this as potential function ψ_n

$$\psi_n = \sum_{j,i} \left(c_{n,i,j} - \frac{g_{n,j}}{N} \right)^2 \tag{3.8}$$

Let us study ψ for p-push gossip, i.e. when every node is making p pushes to p nodes. Here we are assuming that node chooses every node including itself for push independently. Here f(k)=j means that a node k chooses a node j and pushes gossip pair. The $d_{n,j}$ and $g_{n,j}$ are divided by $(p+1)$; one part is always retained by the node and remaining are used for p push.

$$\begin{aligned}
c_{n+1,i,j} &= \frac{1}{p+1}c_{n,i,j} + \frac{1}{p+1}\sum_{k:f(k)=j} c_{n,i,k}. && (3.9)\\
g_{n+1,j} &= \sum_i c_{n+1,i,j} && (3.10)\\
&= \sum_i \frac{1}{p+1}c_{n,i,j} + \sum_i \frac{1}{p+1}\sum_{k:f(k)=j} c_{n,i,k}\\
&= \frac{1}{p+1}g_{n,j} + \frac{1}{p+1}\sum_{k:f(k)=j} g_{n,k}\\
As\\
\psi_{n+1} &= \sum_{j,i}\left(c_{n+1,i,j} - \frac{g_{n+1,j}}{N}\right)^2, && (3.11)
\end{aligned}$$

substituting the values of $c_{n+1,i,j}$ and $g_{n+1,j}$, we get

$$\begin{aligned}
\psi_{n+1} &= \sum_{j,i}\left(\frac{1}{p+1}\left(c_{n,i,j} - \frac{g_{n,j}}{N}\right) + \sum_{k:f(k)=j}\frac{1}{p+1}\left(c_{n,i,k} - \frac{g_{n,k}}{N}\right)\right)^2\\
&= \frac{1}{(p+1)^2}\sum_{j,i}\left(c_{n,i,j} - \frac{g_{n,j}}{N}\right)^2\\
&\quad + \frac{1}{(p+1)^2}\sum_{j,i}\sum_{k:f(k)=j}\left(c_{n,i,k} - \frac{g_{n,k}}{N}\right)^2\\
&\quad + \frac{2}{(p+1)^2}\sum_{j,i}\sum_{k:f(k)=j}\left(c_{n,i,j} - \frac{g_{n,j}}{N}\right)\left(c_{n,i,k} - \frac{g_{n,k}}{N}\right)\\
&\quad + \frac{1}{(p+1)^2}\sum_{j,i}\sum_{\hat{k}}\sum_{k:f(k)=f(\hat{k})=j,k\neq\hat{k}}\left(c_{n,i,k} - \frac{g_{n,k}}{N}\right)\\
&\quad \left(c_{n,j,\hat{k}} - \frac{g_{n,\hat{k}}}{N}\right)
\end{aligned}$$

As we know each node makes p copies $(c_{n,i,k} - \frac{g_{n,k}}{N})^2$ and sends it to randomly to itself and other neighbouring nodes. Each node j will get $(c_{n,i,k} - \frac{g_{n,k}}{N})^2$ from neighbours. When seeing over j, this is the total contribution received. This should be equal to total contribution distributed by all the nodes, i.e.

$\sum_j$ *contibution distributed by each node*

$= \sum_j p \cdot (c_{n,i,j} - \frac{g_{n,j}}{N})^2$

$= p \sum_j (c_{n,i,j} - \frac{g_{n,j}}{N})^2$

$$\begin{aligned}
\psi_{n+1} &= \frac{1}{(p+1)^2} \sum_{j,i} \left(c_{n,i,j} - \frac{g_{n,j}}{N}\right)^2 + \frac{p}{(p+1)^2} \sum_{j,i} \left(c_{n,i,j} - \frac{g_{n,j}}{N}\right)^2 \\
&+ \frac{2}{(p+1)^2} \sum_{j,i} \sum_{k:f(k)=j} \left(c_{n,i,j} - \frac{g_{n,j}}{N}\right)\left(c_{n,i,k} - \frac{g_{n,k}}{N}\right) \\
&+ \frac{2}{(p+1)^2} \sum_{j,i} \sum_{k \neq \hat{k}:f(k)=f(\hat{k})=j} \left(c_{n,i,k} - \frac{g_{n,k}}{N}\right)\left(c_{n,j,\hat{k}} - \frac{g_{n,\hat{k}}}{N}\right) \\
&= \frac{1}{(p+1)}\psi_n + \frac{2}{(p+1)^2} \\
&\sum_{j,i,k:f(k)=j} \left(c_{n,i,j} - \frac{g_{n,j}}{N}\right)\left(c_{n,i,k} - \frac{g_{n,k}}{N}\right) + \frac{2}{(p+1)^2} \\
&\sum_{j,i} \sum_{\hat{k}} \sum_{k:f(k)=f(\hat{k}),\hat{k}\neq k} \left(c_{n,i,k} - \frac{g_{n,k}}{N}\right)\left(c_{n,i,\hat{k}} - \frac{g_{n,\hat{k}}}{N}\right)
\end{aligned}$$

Now we have to calculate the probability of a node being neighbour of another node whose degree is d_k. So first we calculate the number of groups possible without fixing any neighbour and after that we fix one neighbour and calculate the number of groups. By taking the ratio of these two things, we get the desired probability.

We know that node will choose a node randomly among its neighbours. Let us assume that the degree of said node is d_k with probability P_{d_k}. If we form the groups of d_k nodes, total $\binom{N}{d_k}$ groups can be formed. If we fix one node in groups i.e. j, total $\binom{N-1}{d_k-1}$. So the probability that j is neighbour of k is $\frac{\binom{N-1}{d_k-1}}{\binom{N}{d_k}} \cdot P_{d_k}$ and the probability to choose j by node k when j is neighbour of k will be $\frac{1}{\binom{d_k}{1}}$. So

$$\begin{aligned} P[f(k) = j] &= \frac{\binom{N-1}{d_k-1}}{\binom{N}{d_k}} \cdot P_{d_k} \cdot \frac{1}{\binom{d_k}{1}} \\ &= \frac{P_{d_k}}{N} \qquad (3.12) \\ \textit{Similarly} & \qquad (3.13) \\ P[\hat{k} \neq k, f(k) = f(\hat{k}) = j] &= \frac{P_{d_k}}{N} \cdot \frac{P_{d_{\hat{k}}}}{N} \qquad (3.14) \end{aligned}$$

The P_{d_k} is the probability that a node has degree d_k. For networks generated by PA Model,

$$P(d_k) \sim d_k^{-\gamma}.$$

Here γ is network exponent. Hence,

$$\begin{aligned}
E[\psi_{n+1} \mid \psi_n] &= \frac{1}{(p+1)}\psi_n + \frac{2}{(p+1)^2}\sum_{j,k,i}\left(c_{n,i,j} - \frac{g_{n,j}}{N}\right) \\
&\left(c_{n,i,k} - \frac{g_{n,k}}{N}\right)P[f(k) = j] \\
&+\frac{1}{(p+1)^2}\sum_{i,j}\sum_{\hat{k}}\sum_{k:\hat{k}\neq k}\left(c_{n,i,k} - \frac{g_{n,k}}{N}\right) \\
&\left(c_{n,i,\hat{k}} - \frac{g_{n,\hat{k}}}{N}\right)P[\hat{k} \neq k, f(k) = f(\hat{k}) = j] \\
&= \frac{1}{p+1}\psi_n + \frac{2}{(p+1)^2}\sum_{j,k,i}\left(c_{n,i,j} - \frac{g_{n,j}}{N}\right) \\
&\left(c_{n,i,k} - \frac{g_{n,k}}{N}\right)\frac{P_{d_k}}{N} + \frac{N}{(p+1)^2} \\
&\sum_{i}\sum_{\hat{k}}\sum_{k}\left(c_{n,i,k} - \frac{g_{n,k}}{N}\right)\left(c_{n,i,\hat{k}} - \frac{g_{n,\hat{k}}}{N}\right)\frac{P_{d_k}}{N} \\
&\cdot\frac{P_{d_{\hat{k}}}}{N} - \frac{N}{(p+1)^2}\sum_{i,k}\left(c_{n,i,k} - \frac{g_{n,k}}{N}\right)^2\frac{P^2_{d_k}}{N^2} \\
&= \frac{1}{p+1}\psi_n + \frac{2}{(p+1)^2 N}\sum_{i}\sum_{j}\left(c_{n,i,j} - \frac{g_{n,j}}{N}\right) \\
&\sum_{k}\left(c_{n,i,k} - \frac{g_{n,k}}{N}\right)P_{d_k} \\
&+\frac{1}{(p+1)^2 N}\sum_{i}\sum_{k}\left(c_{n,i,k} - \frac{g_{n,k}}{N}\right)P_{d_k} \\
&\sum_{\hat{k}}\left(c_{n,i,\hat{k}} - \frac{g_{n,\hat{k}}}{N}\right)P_{d_{\hat{k}}} \\
&-\frac{1}{(p+1)^2 N}\sum_{k,i}\left(c_{n,i,k} - \frac{g_{n,k}}{N}\right)^2 P^2_{d_k}
\end{aligned}$$

Let us assume that the maximum value of P_d is $P_{d_{max}}$ and minimum value is $P_{d_{min}}$

and the difference of $P_{d_{max}}$ and $P_{d_{min}}$ is K_c, then

$$\begin{aligned}
E[\psi_{n+1} \mid \psi_n] \;\leq\; & \frac{1}{p+1}\psi_n \\
& + \frac{2}{(p+1)^2 N} \sum_i \left(\sum_j c_{n,i,j} - \sum_j \frac{g_{n,j}}{N} \right) \\
& \sum_k \left(c_{n,i,k} - \frac{g_{n,k}}{N} \right) P_{d_k} \\
& + \frac{1}{(p+1)^2 N} \sum_i \left(\sum_k c_{n,i,k} P_{d_{max}} - \sum_k \frac{g_{n,k}}{N} P_{d_{min}} \right) \\
& \left(\sum_{\hat{k}} c_{n,i,\hat{k}} P_{d_{max}} - \sum_{\hat{k}} \frac{g_{n,\hat{k}}}{N} P_{d_{min}} \right) \\
& - \frac{1}{(p+1)^2 N} \sum_{i,k} \left(c_{n,i,k} - \frac{g_{n,k}}{N} \right)^2 P_{d_k}^2
\end{aligned}$$

Applying mass conservation $\sum_j c_{n,i,j}$, $\sum_k c_{n,i,k}$, $\sum_j \frac{g_{n,j}}{N}$ and $\sum_k \frac{g_{n,k}}{N}$ will be equal to 1, so the second term will become zero and third term will become $N \cdot k_c^2$. Forth term is always non negative so removing this term will not affect the bound. So

$$\begin{aligned}
E[\psi_{n+1} \mid \psi_n] \;\leq\; & \frac{1}{p+1}\psi_n + \frac{1}{(p+1)^2 N} \cdot N \cdot K_c^2 \\
& - \frac{1}{(p+1)^2 N} P_{d_k}^2 \sum_{k,i} \left(c_{n,i,k} - \frac{g_{n,k}}{N} \right)^2 \\
\leq\; & \frac{1}{p+1}\psi_n + \frac{K_c^2}{(p+1)^2} - \frac{1}{(p+1)^2 N} P_{d_k}^2 \psi_n \\
\leq\; & \frac{1}{p+1}\psi_n + \frac{K_c^2}{(p+1)^2}
\end{aligned}$$

In the last line we use the fact that $\frac{1}{(p+1)^2 N} P_{d_k}^2 \psi_n$ will always remain non negative. We know that d_{min} is 2. If we consider the value of γ to be 2, maximum value of

K_c^2 will be $\frac{1}{16}$ considering $P_{d_{max}}$ to be zero,

$$E[\psi_{n+1} \mid \psi_n] \leq \frac{1}{p+1}\psi_n + \frac{1}{16 \cdot (p+1)^2} \tag{3.15}$$

Now we will calculate the value of ψ_0. We know that initially the contribution vector $\mathbf{c_j}$ contains only single non-zero value i.e. contribution received from it self and that value is 1, rest all $N-1$ elements are 0. So

$$\begin{aligned}
\psi_0 &= \sum_{j,i}\left(c_{0,i,j} - \frac{g_{0,j}}{N}\right)^2 \\
&= \sum_{j}[\left(c_{0,1,j} - \frac{g_{0,j}}{N}\right)^2 + \left(c_{0,2,j} - \frac{g_{0,j}}{N}\right)^2 + \ldots + \\
&\quad \left(c_{0,j,j} - \frac{g_{0,j}}{N}\right)^2 + \ldots + \left(c_{0,N,j} - \frac{g_{0,j}}{N}\right)^2] \\
&= \sum_{j}[\left(0 - \frac{1}{N}\right)^2 + \left(0 - \frac{1}{N}\right)^2 + \ldots + \\
&\quad \left(1 - \frac{1}{N}\right)^2 + \ldots + \left(0 - \frac{1}{N}\right)^2] \\
&= \sum_{j}[1 + \frac{1}{N^2} - \frac{2}{N} + (N-1) \cdot \frac{1}{N^2}] \\
&= \sum_{j}[1 - \frac{1}{N}] = N - 1
\end{aligned} \tag{3.16}$$

Now we will substitute the value of ψ_0 in (3.15). This will give us the bound on

ψ_n

$$\begin{aligned}
E[\psi_1 \mid \psi_0] &\le \frac{1}{p+1}\psi_0 + \frac{1}{16\cdot(p+1)^2} && (3.17)\\
E[\psi_1] &\le \frac{1}{p+1}(N-1) + \frac{1}{16\cdot(p+1)^2} \\
\textit{Similarly} & && (3.18)\\
E[\psi_2] &\le \frac{1}{p+1}(\frac{1}{p+1}(N-1) + \frac{1}{16\cdot(p+1)^2}) \\
&\quad + \frac{1}{16\cdot(p+1)^2} \\
&\le \frac{N-1}{(p+1)^2} + \frac{1}{16(p+1)^3} + \frac{1}{16(p+1)^2} \\
E[\psi_n] &\le \frac{N-1}{(p+1)^n} + \frac{1}{16(p+1)^{n+1}} \\
&\quad + \frac{1}{16(p+1)^n} + \ldots + \frac{1}{16(p+1)^2} \\
&\le (N-1)\cdot(p+1)^{-n} + \frac{1}{16\cdot(p+1)p} \\
&\quad - \frac{1}{16p(p+1)^{n+1}} \\
&\le (N-1)\cdot(p+1)^{-n} + \frac{1}{16\cdot(p+1)p}
\end{aligned}$$

It can be seen that right hand side of the above equation is maximum when p=1. It means potential function is decaying at the slowest rate for p=1. So time taken in convergence for p=1 will be maximum. For normal push, algorithm will act as upper bound for differential push algorithm i.e. combination of different values of positive integer p's. So taking p=1;

$$\begin{aligned}
E[\psi_n] &\le (N-1)\cdot 2^{-n} + \frac{1}{32} \\
&\le (N-1)\cdot 2^{-n}\cdot k_d && (3.19)
\end{aligned}$$

Here K_d is an integer constant that is greater than the ratio of maximum value of $(N-1)\cdot 2^{-n}$ and $\frac{1}{32}$. This can be seen that K_d will depend on the number of steps required for convergence.

After gossiping for $n = log_2(N-1) + log_2 k_d + log_2\frac{1}{\xi}$ steps,

$$
\begin{aligned}
E[\psi_n] &\leq (N-1)\cdot 2^{-(log_2(N-1))}\cdot 2^{-(log_2(k_d))} \\
&\quad \cdot 2^{-(log_2(\xi))}\cdot k_d \\
E[\psi_n] &\leq \xi \qquad (3.20)\\
\sum_{j,i}\left(c_{n,i,j} - \frac{g_{n,j}}{N}\right)^2 &\leq \xi
\end{aligned}
$$

If summation of some non-negative numbers are less than ξ than individually each number must be less than ξ, i.e. $|c_{n,i,j} - \frac{g_{n,j}}{N}| \leq \xi^{\frac{1}{2}}$ for all nodes i.

If we consider weight as an information to be spread among the nodes, according to theorem (3.5.1), information will reach to all nodes in a power law network with high probability, in $n = (log_2)^2 N$ rounds. After these n rounds every node will receive at least 2^{-n} weight. So applying union bound [54] over weight spreading and potential decay event (We have seen potential is decaying at every step) and dividing with $g_{n,j}$ results that $|\frac{c_{n,i,j}}{g_{n,j}} - \frac{1}{N}| \leq \xi$ at steps $O((log_2 N)^2 + log_2 N + log_2 k_d + log_2\frac{1}{\xi})$ i.e. within $O(log^2 N + log\frac{1}{\xi})$ steps with high probability. □

On the basis of these two theorems, it can be seen (as in [47]), that with high probability relative error in average estimation and sum estimation (if only one node is given weight one and others are given zero) will be bounded by ξ after $O((log_2)^2 N + log_2\frac{1}{\xi})$ gossip steps.

3.5.2 Analysis of Collusion

As we have said that in our proposed system a node may get trust values about a node by three possible ways, first by direct interaction, second from neighbours and third by gossiping. First can not be affected by collusion. We are assuming that second will also not be affected by collusion as neighbours have a definite level of trust for each other. We are considering the collusion because of the third way.

For analysis of collusion, we will calculate the difference of real reputation and estimated reputation of a node x by some node o in the presence of collusion using our proposed method, we will compare it with the method proposed in [22]. Let us assume that the network is formed by the member nodes of set $\mathbb{N}$. There is a subset $\mathbb{C}$ of set $\mathbb{N}$ such that member nodes of set $\mathbb{C}$ are involved in collusion. The cordiality of sets $\mathbb{N}$ and $\mathbb{N}$ are assumed to be N and C respectively. We also assume that members nodes of set $\mathbb{C}$ are colluding in groups with a group size of G. By colluding in a group we mean that if some node is the member of that group then group members of colluding group will report its reputation as 1. Whereas for others nodes they will report the reputation value as 0. Let us say that real reputation of a node x is R^x_{real}, and estimated reputation is R^x_{estc}, if x is a colluding node, R^x_{estnc}, if x is not a colluding node.

$$R^x_{real} = \frac{\sum\limits_{i \in \mathbb{N}} t_{xi}}{N}. \tag{3.21}$$

Here t_{xi} is trust value of the node x at node i.

If x is not a colluding node then,

$$R^x_{estnc} = \frac{\sum\limits_{i\in\mathbb{N}\setminus\mathbb{C}} t_{xi}}{N}. \tag{3.22}$$

If x is a colluding node then,

$$R^x_{estc} = \frac{\sum\limits_{i\in\mathbb{N}\setminus\mathbb{C}} t_{xi} + G}{N}. \tag{3.23}$$

So the expected value of reputation estimate ($E[R^x_{est}]$) will be

$$\begin{aligned} E[R^x_{est}] &= \frac{C}{N}\left(\frac{\sum\limits_{i\in\mathbb{N}\setminus\mathbb{C}} t_{xi} + G}{N}\right) + \left(1 - \frac{C}{N}\right)\left(\frac{\sum\limits_{i\in\mathbb{N}\setminus\mathbb{C}} t_{xi}}{N}\right) \\ &= \frac{GC}{N^2} + \frac{\sum\limits_{i\in\mathbb{N}\setminus\mathbb{C}} t_{xi}}{N} \end{aligned} \tag{3.24}$$

So difference in real reputation and expected value of estimated reputation by node *o* for node *x* (ΔR^{ox}_{old}) will be,

$$\Delta R^{ox}_{old} = -\frac{GC}{N^2} + \frac{\sum\limits_{i} \in \mathbb{C}\mathbf{t_{xi}}}{N}. \tag{3.25}$$

Now, we incorporate the trust based weighted opinion of neighbours. Lets us assume that w_{oi} is the weight given to the opinion of node *i* by node *o*. It may be noted $w_{oi} \geq, \forall i$ (*equation* 3.2) So the real reputation of node *x* for node *o* will be,

$$R^x_{real} = \frac{\sum\limits_{i} \in \mathbb{N}\mathbf{t_{xi}} + \sum\limits_{\mathbf{i}} \in \mathbb{N}(\mathbf{w_{oi}} - \mathbf{1})\mathbf{t_{xi}}}{N + \sum\limits_{i=1}^{i\in\mathbb{N}} (w_{oi} - 1)}. \tag{3.26}$$

If x is not a colluding node then,

$$R^x_{estnc} = \frac{\sum\limits_{i\in\mathbb{N}\setminus\mathbb{C}} t_{xi} + \sum\limits_{i\in\mathbb{N}} (w_{oi} - 1)t_{xi}}{N + \sum\limits_{i\in\mathbb{N}} (w_{oi} - 1)}. \tag{3.27}$$

And if x is a colluding node then,

$$R^x_{estc} = \frac{\sum\limits_{i\in\mathbb{N}\setminus\mathbb{C}} t_{xi} + \sum\limits_{i\in\mathbb{N}}(w_{oi}-1)t_{xi} + G}{N + \sum\limits_{i\in\mathbb{N}}(w_{oi}-1)}. \tag{3.28}$$

So the expected value of reputation estimate ($E[R^x_{est}]$) will be

$$\begin{aligned} E[R^x_{est}] &= \frac{C}{N}\left(\frac{\sum\limits_{i\in\mathbb{N}\setminus\mathbb{C}} t_{xi} + \sum\limits_{i\in\mathbb{N}}(w_{oi}-1)t_{xi} + G}{N + \sum\limits_{i\in\mathbb{N}}(w_{oi}-1)}\right) \\ &\quad + \left(1-\frac{C}{N}\right)\left(\frac{\sum\limits_{i\in\mathbb{N}\setminus\mathbb{C}} t_{xi} + \sum\limits_{i\in\mathbb{N}}(w_{oi}-1)t_{xi}}{N + \sum\limits_{i\in\mathbb{N}}(w_{oi}-1)}\right) \\ &= \frac{GC}{N\cdot(N + \sum\limits_{i\in\mathbb{N}}(w_{oi}-1))} \\ &\quad + \left(\frac{\sum\limits_{i\in\mathbb{N}\setminus\mathbb{C}} t_{xi} + \sum\limits_{i\in\mathbb{N}}(w_{oi}-1)t_{xi}}{N + \sum\limits_{i=1}^{i\in\mathbb{N}}(w_{oi}-1)}\right). \end{aligned} \tag{3.29}$$

So difference in real reputation and expected value of estimated reputation by node o for node x (ΔR^{ox}_{new})will be,

$$\begin{aligned} \Delta R^{ox}_{new} &= -\frac{GC}{N\cdot(N + \sum\limits_{i=1}^{N}(w_{oi}-1))} + \frac{\sum\limits_{i=N-C+1}^{C} t_{xi}}{N + \sum\limits_{i=1}^{N}(w_{oi}-1)} \\ &= \frac{N}{(N + \sum\limits_{i=1}^{N}(w_{oi}-1))}\cdot\left(-\frac{GC}{N^2} + \frac{\sum\limits_{i=N-C+1}^{C} t_{xi}}{N}\right) \\ &= \frac{N}{(N + \sum\limits_{i=1}^{N}(w_{oi}-1))}\cdot\Delta R^{ox}_{old} \end{aligned} \tag{3.30}$$

3.5.3 Numerical Results

Performance of algorithm for reputation aggregation for peer to peer file sharing is evaluated by simulation as well. A power law network has been built using *Preferential Attachment* model.

The simulation experiments has been conducted for 100 to 50000 nodes. Performance of differential algorithm has been evaluated in terms of number of iterations (to assess the rate of convergence) required to converge within a particular degree of aggregation error. Number of packets per node per gossip step that are required to be transmitted for convergence have also been calculated, so that network overhead can be assessed. Algorithm is also tested against collusion and node churn.

Figure 3.2 shows the number of gossip steps required for different error bounds for different number of nodes. This is clearly evident that number of gossip steps is increasing with a rate much less than $O(log_2N)^2$ i.e. said bound is followed for all the cases.

Figure 3.3 and 3.4 are the simulation results against the problem of churn. Figure 3.3 shows the number of gossip steps required with different packet loss probability for 10000 nodes. Figure 3.4 shows the number of gossip steps required with different number of nodes with $\xi = 0.00001$. Peer to peer networks operate above TCP layer, i.e. these kind of networks assume a reliable bit pipe between sender and receiver. So peer to peer network suffers by packet loss only when some node leaves the network i.e. due to churning. Here the assumption is when a node leaves during gossip process, it hands over the gossip pair vectors to some other node so mass conservation still applies. Whenever a node pushes gossip pair to this absent node, the pushing node doesn't receive any acknowledgement. It implies that node has left the network and the pushed gossip pair got dropped. In such cases pushing node pushes the gossip pair to itself so that mass conservation still applies. We can see a small increment in the number of gossip steps with the increase in the packet loss probability.

As in the case of packet loss, the gossip pair returns back to pushing node itself, one step for this pair goes waste. Hence, we observe a small increment in the number of gossip steps. We also did simulations for the case when node is not handing over to any other node before leaving the network. In this case also, network converges in almost same number of steps but on a different value. As we have not studied it theoretically, we have not presented it in the thesis.

Figure 3.5 and 3.6 are the simulation results against the problem of collusion. Figure 3.6 plots the average RMS error in estimated reputation with the variation of percentage of users that are colluding individually i.e. they are not forming any group Figure 3.5 plots the average RMS error in the estimated reputation with the variation of group sizes of colluding users for different percentages of colluding users in the network.

Here average RMS error is defined as follows,

$$Average\ RMS\ error = \frac{1}{N}\sum_{i}\sqrt{\frac{\sum_{j}((r_{ij}-\hat{r}_{ij})/r_{ij})^2}{N}} \qquad (3.31)$$

Here r_{ij} is the reputation of node i at node j computed by differential gossip in

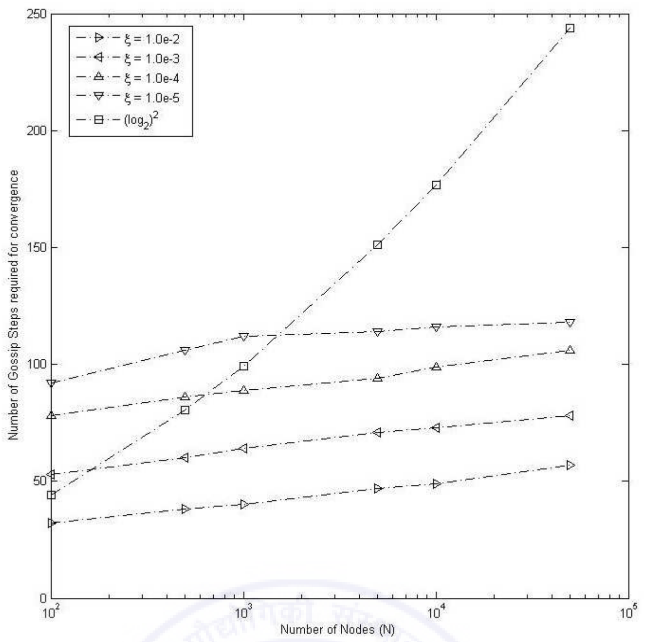

Figure 3.2: Gossip step counts with different number of nodes (N) and different error bounds ξ

presence of colluding nodes, whereas $\hat{r_{ij}}$ is the computed reputation if colluding nodes would not have been there. This is clearly evident that effect of collusion on reputation computation by differential gossip is quite less even with very high percentage of colluding users. The colluding group size is making a small difference in differential gossip reputation computation.

Table 3.1 shows the number of message transfer required by a node in one gossip step. It can be seen that this is decreasing slightly with the increase in number of nodes. This is happening because as number of gossip steps increases the overhead incurred in the beginning gets distributed and a node is less burdened as the number of total nodes increases. Similar thing happens when a lower value of ξ is chosen.

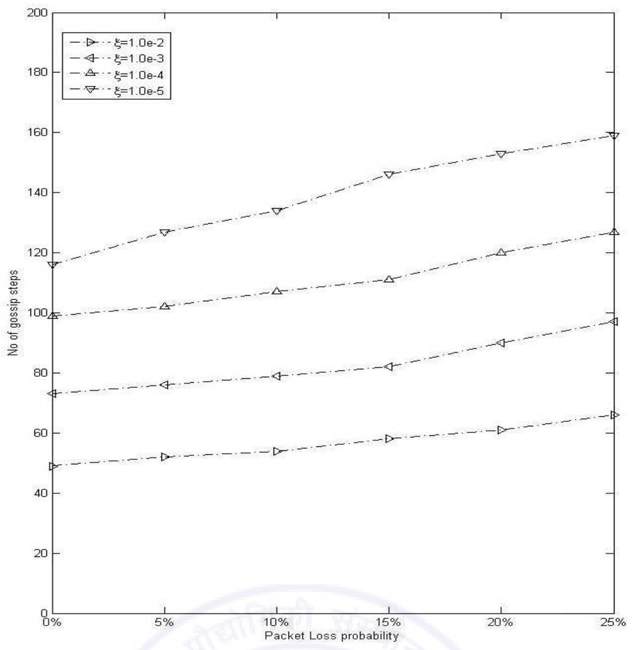

Figure 3.3: Gossip step counts for N=10000 with different error bounds ξ for different packet loss probability

3.6 Conclusion

In peer-to-peer networks, free riding is a major problem that can be overcome by using reputation management system. A reputation management system includes two processes, first estimation of reputation and second aggregation of reputation. In this chapter we have proposed an aggregation technique by modifying push gossip algorithm to differential push gossip algorithm.

The proposed aggregation technique aggregates the trust values from different nodes in a power law network. This technique does not require the identification of power nodes. As identification of power nodes in distributed setting is hard, this makes algorithm implementable. This algorithm is also robust against

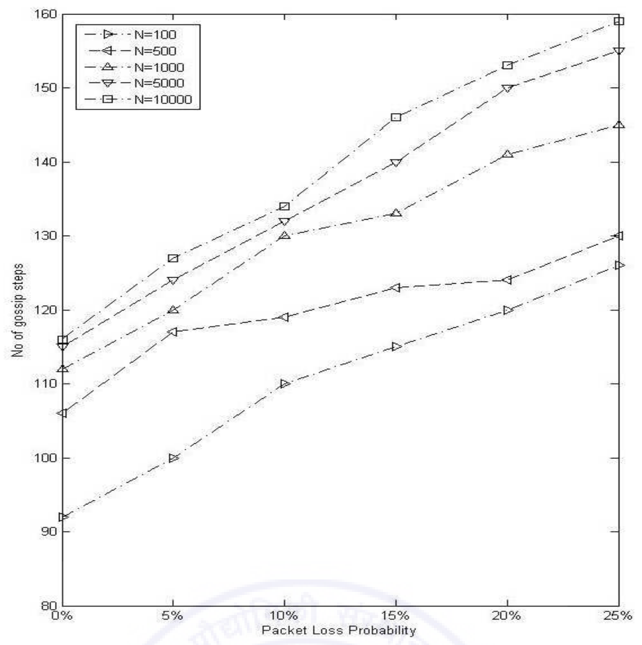

Figure 3.4: Gossip step counts with different number of nodes(N) for $\xi = 1.0e-5$ for different packet loss probability

churning as can be seen in figure 3.3 and 3.4. Proposed technique aggregates the reputation in a differential manner. This is done by considering the feedback of trusted nodes with a higher weight. This leads to robustness against collusion as evident from figure 3.5 and 3.6.

Proposed algorithm is presented to avoid the problem of free riding but this can also be used to avoid malicious users in the network just by change the method of estimation of a_i and b_{ij}.

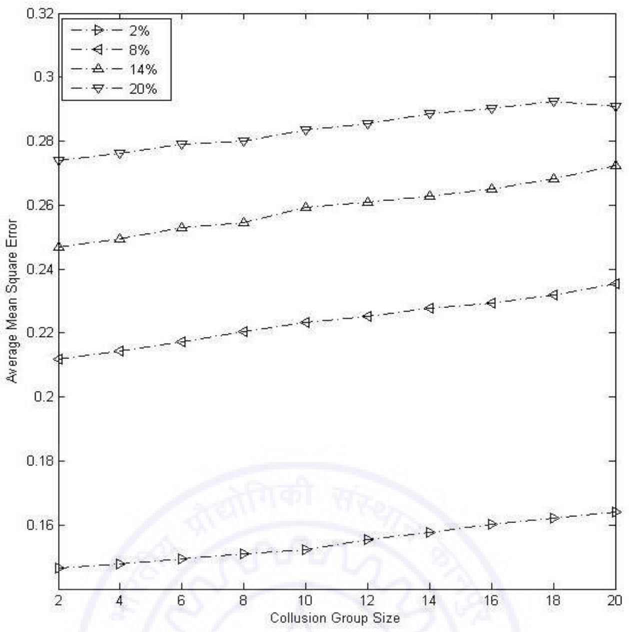

Figure 3.5: Average RMS error with different size colluding groups for different percentage of colluding peers

	ξ=0.01	ξ = 0.001	ξ = 0.0001	ξ = 0.00001
N=100	1.212	1.203	1.195	1.188
N=500	1.199	1.194	1.189	1.183
N=1000	1.178	1.159	1.157	1.148
N=10000	1.156	1.139	1.124	1.122
N=50000	1.152	1.132	1.119	1.112

Table 3.1: Number of Message Per Node per Step Transmitted Due to Gossiping

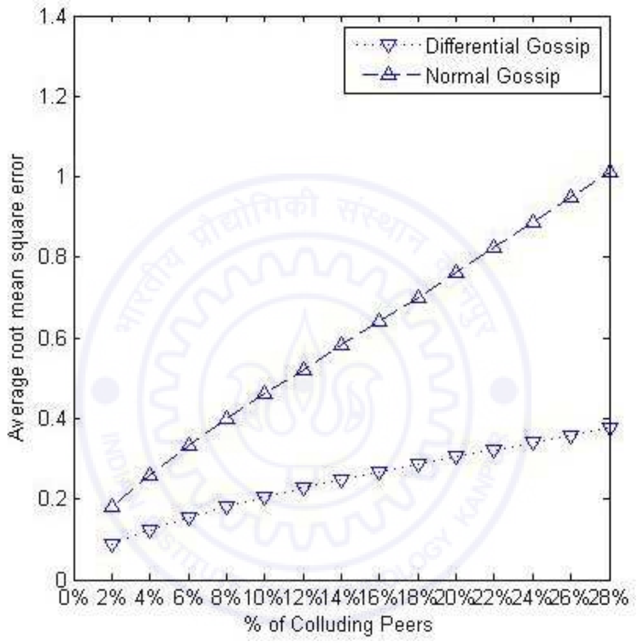

Figure 3.6: Average RMS error with individual peers for different percentage of colluding peers

Chapter 4

Whitewashing in Unstructured Peer-to-Peer Network

4.1 Introduction

On implementation of reputation management system, selfish users start white-washing and colluding [55]. When a node has bad reputation in the network then to avoid disincentives, it leaves the network and returns back with a new identity as a new comer to the network. This is termed as whitewashing. Whitewashing problem becomes more challenging as it is difficult to differentiate between a legitimate newcomer and a whitewasher. This makes whitewashing a big problem in peer-to-peer networks. Although the problem of whitewashing can be solved using permanent identities, it may take away the right of anonymity for users. Hence, any reputation system should be able to overcome these problems.

In this chapter we have analysed the pay-off and penalty for cooperative and defector node respectively in a reputation management system. We have also proposed a novel algorithm to avoid this problem when network uses free temporary identities. In this algorithm, the initial reputation is adjusted according to the level of whitewashing in the network.

Remainder of this chapter is organised as follows. Section 4.2 discusses the related work. Section 4.3 describes the system model. Section 4.4 computes the pay-off in the reputation game. In section 4.5, the problem of whitewashing is discussed along with an algorithm as its solution and its analysis, Section 4.6 presents the numerical results and section 4.7 concludes the chapter.

4.2 Related Work

Many authors have suggested that whitewashing can be totally removed if system has permanent identities [27, 28]. Problem of whitewashing due to availability of cheap or free identities was initially pointed out by Friedaman *et.al.* [29]. They proved that considering every newcomer as defector is a better policy than any other static stranger policy. Whereas Feldman *et.al.* [30] proved that newcomers should only be punished if turnover rate is high. They also proved that a legitimate user can only win from a whitewashing user if newcomer will be served well with a very small probability [31]. Yang *et.al.* studied Maze file sharing system and concluded that incentives promote whitewashing [32].

Lai *et.al.* suggested in [30, 17] that newcomers should be served as per the behaviour shown by them at that time. But this is more prone to collusion and moreover in a resource sharing network, nodes generally form a group where they serve more. In that case observing behaviour will not work well. Similar kind of approaches are proposed for new service providers' reputation among consumers in on-line communities [56].

Anceaume *et.al.* [57] have suggested to charge an entry fee from every newcomer in such a way that it should avoid whitewashing and should not discourage newcomers. Chen *et.al.* claims that a whitewashing node will continue to have same kind of habits even after whitewash so it should be identified and punished on the basis of its habits [58]. Zuo *et.al.* suggested that newcomers should only be allowed to access the resource from newcomers and low trust value nodes [59].

Yu *et.al.* [60] proposed that any user that has served even once should have a higher value of reputation than a newcomer and it should rise faster initially and slower later on as a result of cooperation.

4.3 System Model

In this chapter, we are studying a pure unstructured peer-to-peer network. Peers in this network are connected by an access link followed by a backbone link and then again by an access link to the second node. We are assuming that the network is heavily loaded i.e. every peer has sufficient number of pending download requests, hence these peers are contending for the available transmission capacity. We also assume that every peer is paying the cost of access link as per the usage. So, every peer wants to maximises its download and minimise its upload so that it can get maximum utility for its money spend in access link.

Considering nodes to be purely rational does not answer many questions about nodes' behaviour [30]. Feldman *et.al.* [30] proposed a model where rationality depends upon the type of node or the level of generosity i.e. every node will free ride or contribute as per its type. On the similar lines, we propose the level of honesty regarding whitewashing behaviour of nodes. If h_i be the honesty level of i^{th} node then i will whitewash provided,

$$R_{ini} \geq h_i \tag{4.1}$$

Here R_{ini} is the initial reputation given to a newcomer node. This Equation states that any node will only whitewash when it will get the initial reputation more than its honesty level. It means any node i will only whitewash when $R_{ini} \geq h_i$. In other words, every node is a potential white washer as per its level of honesty. If pay-off by whitewash is greater than its honesty level, then it will whitewash otherwise it will remain an honest node. As network will contain all kind of nodes with equal probability, honesty levels of nodes are assumed to be distributed uniformly between 0 to 1.

Reputation can be measured in number of ways. We measure the reputation of a node after a transaction as,

$$Reputation\ of\ node\ i = \frac{Resource\ provided\ by\ node\ i}{Resource\ requestd\ to\ node\ i}. \tag{4.2}$$

4.4 Pay-off in Reputation Game

In this section, we will see the pay-off of an agent in a network where a reputation management system has been implemented with different type of identities viz.

permanent identities and temporary identities. Permanent identities can also be termed as ∞ cost identities as these can not be changed. Social security number in USA is one such example. Whereas temporary identities can be classified into two types, finite cost identities and zero cost identities on the basis of cost of assigning a new identity.

Here it is important to note that in an interaction even if cost incurred in serving other nodes' request and the value received from other nodes in response to its request, are same, the node still gains by a small amount δ. This comes because after every transaction the chance of survival satisfaction for a node increases.

4.4.1 Payoff of Agents with Permanent Identities

Let us assume that there are N nodes in the network. Reputation of these nodes is distributed between 0 and 1 with some arbitrary distribution having mean μ. When a new node joins the network, it is assigned an initial reputation value of R_{ini}. We are assuming that reputation aggregation is happening after every round of transactions and in this process, the nodes that have transacted with the node will jointly form a reputation that will be spread across the network. It means that every node knows the reputation of other node after every round. For simplicity we assume that every node requests for same amount of resource i.e. c. We also assume that network is large and hence the mean reputation of the network will not change considerably and probability of allocation against a node's request will be

$$P(allocation) \propto (reputation)^x$$

Taking proportionality constant to be 1 for simplicity

$$P(allocation) = (reputation)^x. \tag{4.3}$$

As we have stated, the expected reputation of nodes will be μ. Any node can ask for a resource from the entering node. The probability that a particular node will ask for a resource is $\frac{1}{N}$ assuming that all nodes are equally likely to ask for a resource. Let us assume that every node can ask c amount of resource. So, if the node receives one request, the expected service, it has to do, will be

$$\begin{aligned} E[Serv(1)] &= \frac{1}{N} \cdot \mu^x \cdot c + \frac{1}{N} \cdot \mu^x \cdot c + ... + \frac{1}{N} \cdot \mu^x \cdot c \\ &= \mu^x \cdot c. \end{aligned} \tag{4.4}$$

Whereas the node will get service according to its reputation. So the expected return for this node if it makes a single request will be

$$E[ret(1)] = c \cdot (R_{ini})^x + \delta. \tag{4.5}$$

First we will see the pay-off of cooperative node. If a node serves m requests and makes $\acute{m}$ requests, the expected pay-off of this node at the end of the first round will be

$$E_c[payoff(1)] = -m\mu^x \cdot c + \acute{m}c \cdot (R_{ini})^x + \delta. \tag{4.6}$$

The expected reputation of an entering node after first round of transaction, if reputation is measured using equation (4.2), is

$$\begin{aligned} E_c[R(1)] &= \frac{m\mu^x \cdot c}{mc} \quad , \\ &= \mu^x. \end{aligned} \tag{4.7}$$

The reputation of a node i is estimated by few other nodes. The nodes with whom interaction does not take place use R_{ini} as the default reputation. When aggregating the reputation [51], only the estimated reputations are aggregated, not the default value R_{ini}.

Similarly pay-off and reputation of this node at the end of second round will be,

$$\begin{aligned} E_c[payoff(2)] &= -2m\mu^x \cdot c + \acute{m} \cdot (\mu^x)^x \cdot c \\ &\quad +\acute{m} \cdot (R_{ini})^x c + 2\delta \\ E_c[R(2)] &= \frac{\mu^x + \mu^x}{2} = \mu^x \end{aligned}$$

Similarly, pay-off and reputation of this node at the end of k^{th} round will be

$$\begin{aligned} E_c[payoff(k)] &= -km\mu^x \cdot c + (k-1)\acute{m} \cdot (\mu^x)^x \cdot c \qquad (4.8) \\ &\quad +\acute{m} \cdot (R_{ini})^x c + k\delta \\ E_c[R(k)] &= \frac{\mu^x + \ldots + \mu^x}{k} = \mu^x \qquad (4.9) \end{aligned}$$

Now let's consider defecting node. In the first round it will get service as per R_{ini}. But after that, it will not get any service because every body will know that the node with this identity is not going to serve. It can not change its permanent identity as this will cost him ∞. So the expected pay-off of defecting node, with same request profile, after k rounds will be

$$E_d[payoff(k)] = \acute{m} \cdot (R_{ini})^x c + \delta. \qquad (4.10)$$

If we take $m = \acute{m}$, as this should be the condition statistically, it can be seen that for $x \geq 1$, cooperative node will always remain in negative pay-off if it has a R_{ini} less than μ. Moreover for any value of R_{ini} with $x \geq 1$, the defector node will always remain in advantage. So value of x should be less than 1. Now let's compare (4.8) and (4.10) to get the number of rounds required after which cooperative node will be in advantage over defector node.

$$\acute{m} \cdot (R_{ini})^x c + \delta \leq -km\mu^x \cdot c + (k-1)\acute{m} \cdot (\mu^x)^x \cdot c + \acute{m} \cdot (R_{ini})^x c + k\delta$$

On solving, we get

$$k \geq \frac{\acute{m}(\mu^x)^x + \frac{k\delta}{c}}{\acute{m}(\mu^x)^x + \frac{k\delta}{c} - m\mu^x}. \tag{4.11}$$

Considering δ to be small and minimizing the value of k with respect to x, it turns out that for $x = \frac{1}{2}$, k will be minimum. So for $x = \frac{1}{2}$, k will be

$$k \geq \frac{\acute{m}\mu^{\frac{1}{4}}}{\acute{m}\mu^{\frac{1}{4}} - m\mu^{\frac{1}{2}}} \tag{4.12}$$

Here it is interesting to note that $\acute{m} = m$ (that should be the condition under equilibrium) and $\mu = 1$ value of k turns out to be ∞. This happens because now newcomer node can not improve its reputation any more above $\mu = 1$ and if node is cooperative, every time its positive pay-off becomes equal to its negative pay-off. Only δ gain is the gain that comes as an advantage to him over the defector. Secondly if every body has reputation equal to 1 that means it's an ideal system and does not need any reputation management.

4.4.2 Payoff of Agents with Zero Cost Identities

In case of free identities, the pay-off for a cooperative node will remain same as in the case of permanent identities i.e. (4.8). But for defector node, it will change because now defector node can white wash and change its identity. So the defector's expected pay-off after first round will be,

$$E_w[payoff(1)] = \acute{m} \cdot c \cdot (R_{ini})^x + \delta.$$

Its reputation will become 0 as it will not serve but it will change its identity as it will not cost any thing for it. So its reputation will again become R_{ini}. Pay-off of defector after second round will be,

$$\begin{aligned} E_w[payoff(2)] &= \acute{m} \cdot c \cdot (R_{ini})^x + \delta + \acute{m} \cdot c \cdot (R_{ini})^x + \delta \\ &= 2\acute{m} \cdot c \cdot (R_{ini})^x + 2\delta. \end{aligned}$$

Similarly pay-off after k rounds will be,

$$E_w[payoff(k)] = k\acute{m} \cdot c \cdot (R_{ini})^x + k\delta. \tag{4.13}$$

Comparing (4.8) and (4.13) to get the number of rounds required after which cooperative node will be on advantage over defector node.

$$k\acute{m} \cdot (R_{ini})^x c + k\delta \le -km\mu^x \cdot c + (k-1)\acute{m} \cdot (\mu^x)^x \cdot c + \acute{m} \cdot (R_{ini})^x c + k\delta$$

On solving we get,

$$k \ge \frac{\acute{m}(\mu^x)^x - \acute{m}(R_{ini})^x}{\acute{m}(\mu^x)^x - m\mu^x - \acute{m}(R_{ini})^x}. \tag{4.14}$$

Fig 4.1 is showing the initial reputation that can be awarded to newcomer for different values of x in such a way that node cannot whitewash. We want to provide the maximum value of initial reputation For $\mu = 0.5$ and $m = \acute{m}$, it seems that the maximum possible value of R_{ini} giving which ensures that a cooperative node will win against a defector node is 0.036 at $x = 0.75$ (figure 4.1).

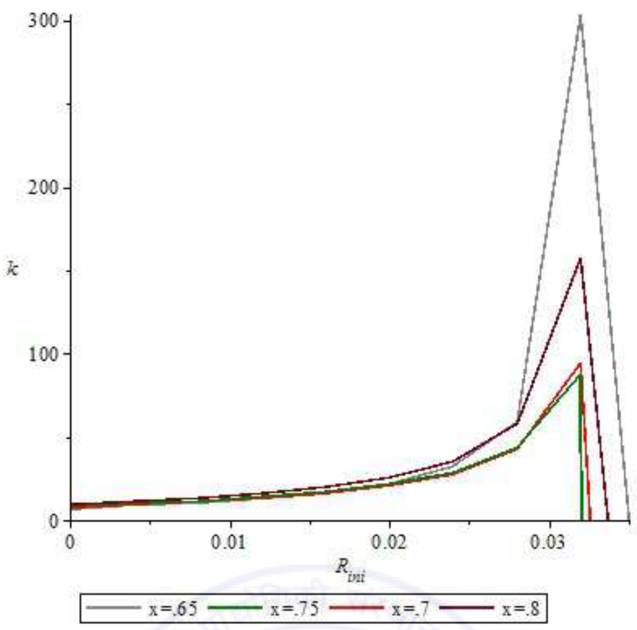

Figure 4.1: Variation in Initial reputation, number of round and x

4.4.3 Payoff of Agents with Finite Cost Identities

Now we will consider an intermediate case where identities have a fixed finite cost, i.e. z. It means that every time when a node will enter in the network with a new identity, it will get a pay-off of $-z$. The expected pay-off of the cooperative node in the end of first round will be

$$E_c[payoff(1)] = -m\mu^x \cdot c + \acute{m}c \cdot (R_{ini})^x - z + \delta. \tag{4.15}$$

The expected reputation of entering node after first round of transaction if reputation is measured using equation (4.2), is given by

$$\begin{aligned} E_c[R(1)] &= \frac{m\mu^x \cdot c}{mc} \\ &= \mu^x. \end{aligned} \tag{4.16}$$

Similarly pay-off and reputation of this node at the end of second round will be

$$\begin{aligned} E_c[payoff(2)] &= -2m\mu^x \cdot c + \acute{m} \cdot (\mu^x)^x \cdot c \\ &\quad +\acute{m} \cdot (R_{ini})^x c - z + 2\delta \\ E_c[R(2)] &= \frac{u^x + \mu^x}{2} = \mu^x \end{aligned}$$

Similarly pay-off and reputation of this node at the end of k^{th} round will be,

$$\begin{aligned} E_c[payoff(k)] &= -km\mu^x \cdot c + (k-1)\acute{m} \cdot (\mu^x)^x \cdot c \\ &\quad +\acute{m} \cdot (R_{ini})^x c - z + k\delta \end{aligned} \tag{4.17}$$

$$E_c[R(k)] = \frac{\mu^x + ... + \mu^x}{k} = \mu^x. \tag{4.18}$$

The defector's expected pay-off after first round will be,

$$E_w[payoff(1)] = \acute{m} \cdot c \cdot (R_{ini})^x - z + \delta.$$

Its reputation will become 0 as it will not serve but it will change its identity as it will not cost any thing for it. So its reputation will again become R_{ini}. Pay-off of defector after second round will be,

$$\begin{aligned} E_w[payoff(2)] &= \acute{m} \cdot c \cdot (R_{ini})^x - z + \acute{m} \cdot c \cdot (R_{ini})^x - z + 2\delta \\ &= 2\acute{m} \cdot c \cdot (R_{ini})^x - 2z + 2\delta. \end{aligned}$$

Similarly pay-off after k rounds will be,

$$E_w[payoff(k)] = k\acute{m} \cdot c \cdot (R_{ini})^x - kz + k\delta. \tag{4.19}$$

Comparing (4.17) and (4.19) to get the number of rounds required after which cooperative node will be on advantage over defector node, we get

$$k\acute{m} \cdot c \cdot (R_{ini})^x - kz + k\delta \leq km\mu^x \cdot c + (k-1)\acute{m} \cdot (\mu^x)^x \cdot c + \acute{m} \cdot (R_{ini})^x c - z + k\delta. \quad (4.20)$$

$$k \geq \frac{\acute{m}(\mu^x)^x - \acute{m}(R_{ini})^x + \frac{z}{c}}{\acute{m}(\mu^x)^x - m\mu^x - \acute{m}(R_{ini})^x + \frac{z}{c}}. \quad (4.21)$$

4.5 Problem of Whitewashing

When in a peer-to-peer network, a reputation management system is implemented, selfish nodes start applying different strategies to counter the rules of the system for their benefit. Whitewashing is one such strategy.

If cheap identities are available in the system, a rational node does not cooperate i.e. it free rides and changes its identity when its reputation goes low. When a normal node encounters such node, it thinks that this is a legitimate newcomer and hence serves it according to that. Hence, a white washer utilizes the resources of system even without cooperating.

Whitewashing can be totally avoided if we have permanent identities as mentioned in the previous section. But having such permanent identities implies that nobody is anonymous [29]. One way to get permanent identities without losing anonymity is to use Gossip-Based Computation of Aggregate Information. This requires a trusted central certification authority and every user has to take a signed certificate.

If permanent identities are not to be used, there is no way by which one can differentiate between legitimate newcomer and a whitewasher. Whitewashing can be avoided by keeping a low initial reputation value R_{ini}. But as we have seen in the previous section, this leads to a very small value and discourages legitimate new comers to join the system because of initial hardship in the system.

Keeping some cost of identities seems to be a good idea. But it is difficult to decide the cost of an identity. If the cost is kept higher, the whitewashing will be discouraged but on the other hand legitimate new comers will also be discouraged. This will reduce the efficiency of the system. Moreover, distribution of the earning due to the cost of identities among the existing users, will be another problem. Alternatively a policy may be made that any newcomer will not be served initially or will be served with poor quality of service. It will be served properly only once it has earned some credit by serving some existing nodes i.e. a newcomer will be provided small or 0 initial reputation. This will lead to initial hardship to legitimate newcomers and hence will decrease system performance.

We propose that when a new node joins the network, it should be given an initial reputation value R_{ini}. As it's difficult to identify whether a new joining node is a a whitewasher or legitimate newcomer, all we can do is to make decision on the basis of level of whitewash in the network. So this initial value should keep on changing on the basis of level of whitewashing in the network. It means that if whitewashing level is low, the R_{ini} value will be kept high; where as if whitewashing level increases, R_{ini} value will be decreased adaptively.

As whitewashing nodes can not be identified and hence can not be counted so we need to estimate the level of whitewashing on the basis of growth in network at regular basis. For this, we need to follow an algorithm for estimation of whitewashing level. In this algorithm, we count the number of nodes at regular basis and find the increase in the size of network in distributed fashion. This can be easily done using gossiping techniques. If reputation management system is based on [51], gossiping is already happening at regular intervals. So there will not be any additional overhead.

As we are considering network based on PA model [61], the newcomer node will join the network preferably attaching to the existing nodes with high degree. Hence, growth rate at different places within the network will be different.

Once a node finds the total number of nodes in the network at two consecutive gossiping instances, it can locally calculate the overall growth rate of network. This estimate of growth rate must be adjusted to get an estimate of local growth rate due to the reason mentioned above. This can be done using,

$$G_{local,i} = \frac{d_{local,i}}{d_{average}} \cdot \frac{N_n}{N_{n-1}}. \tag{4.22}$$

Here $G_{local,i}$ is local growth rate of the network with respect to node i; $d_{average}$ is the average degree of complete network; $d_{local,i}$ is the local average degree with respect to node i; and N_n, N_{n-1} are the network size in current and previous iteration respectively. Overall average degree of the network can be found out using gossip. By adding one more parameter, i.e. self degree of node, to gossiping will give the sum of degrees of nodes in the network and hence average degree can be estimated by every node. As gossip is already happening, adding new parameter in the gossip will not lead to too much increase in overhead. The local average degree can be computed by taking average of neighbours' degree.

The nodes will also monitor the recently departed nodes. It will categorize theses into two types on the basis of Expected initial reputation. Expected initial reputation is the average of maximum initial reputation and minimum initial reputation. First type of nodes have higher or equal reputation than the expected initial reputation. These nodes will be considered as legitimate departing nodes as no node will whitewash if it already has a reputation higher than the expected initial reputation. Second type of nodes are with lower reputation than the expected initial reputation. These will be considered as potential whitewasher nodes. A node shares following information about its neighbourhood - number of nodes at the end of previous iteration, number of departing nodes and newly arriving nodes since last iteration. If there is no whitewashing, then newly arriving nodes (A_i) should be equal to sum of legitimate departing nodes (L_i) and expected localized growth, i.e.,

$$A_i = L_i + N_{i,n-1} \cdot (G_{local,i} - 1).$$

If whitewashing happens, then

$$A_i > L_i + N_{i,n-1} \cdot (G_{local,i} - 1).$$

The difference is expected to be number of whitewashing nodes $\acute{w}_i$. Thus

$$A_i - \acute{w}_i = L_i + N_{i,n-1} \cdot (G_{local,i} - 1).$$

We can define the level of whitewashing $\acute{W}_{i,n}$ by node i as

$$\acute{W}_{i,n} = \frac{\sum\limits_{j \in NS_i} A_j - N_{j,n-1} \cdot (G_{local,j} - 1) - L_j}{\sum\limits_{j \in NS_i} N_{j,n}}. \tag{4.23}$$

In this formula, $N_{i,n}$ is the sum of degree of the neighbours of a node at the current iteration and NS_i is the set of neighbours of node i.

We assume that node will vary the value of initial reputation with respect to level of whitewashing in quadratic fashion. Let $R_{ini,i,max}$, $R_{ini,i,n}$ and $R_{ini,i,min}$ be the initial reputation values at zero whitewashing, current whitewashing level ($\acute{W}_{i,n}$) and maximum possible whitewashing level at node i respectively. We propose the initial reputation $R_{ini,i,n}$ as,

$$R_{ini,i,n} = \begin{cases} \tilde{R}_{ini,i,n} & \text{if } \tilde{R}_{ini,i,n} \geq R_{ini,i,min}. \\ R_{ini,i,min}, & \text{otherwise.} \end{cases} \tag{4.24}$$

where

$$\tilde{R}_{ini,i,n} = \left(1 - \frac{\acute{W}_{i,n}}{\acute{W}_{i,max}}\right)^2 \cdot R_{ini,i,max}$$

The $\acute{W}_{i,max}$ is the maximum possible whitewashing level at the node i. Initially it will be taken as the ratio of number of nodes that have honesty level between 0 and $R_{ini,n,max}$ and total nodes. Hence, it turns out to be $(R_{ini,n,max} \cdot N)/N = R_{ini,n,max}$. Later on node will try to estimate it by taking the maximum whitewashing level seen over last $\acute{n}$ rounds such that

$$\acute{W}_{i,max} = \max_{last\ \acute{n}\ rounds} \acute{W}_{i,n}. \tag{4.25}$$

The value of $\acute{n}$ may be dynamically adjusted by the node as per the change observed in whitewashing level. For this work, $\acute{n}$ has been taken as constant for every node for simplicity. The value of this constant has been taken as 10 for simulation.

The value of $R_{ini,i,min}$ will be taken such that a cooperative user may win over whitewashing user after few rounds. As evident from the figure 4.1, number of rounds required to win a cooperative user increases at a very high rate above a certain $R_{ini,i}$ value. It may be taken as $R_{ini,i,min}$.

The value of $R_{ini,i,max}$ will be decided on the basis of the cooperation received from legitimate newcomers. A node will average the reputations of newcomers nodes after every gossiping round. In this average computation, it will consider the nodes that have seen three gossiping rounds. This average may be taken as $R_{ini,i,max}$.

Timing diagram of whole process that a node will undergo is described in figure 4.2. We can observe the following from this figure. Any node will periodically allocate resource to the requesting nodes. In this process, a node will first reply to the received queries, then on the basis of number of queries received and network growth rate, it will estimate the level of whitewashing. After this estimation, node will allocate resources to the requesting nodes. The new nodes that will join the network after whitewash level estimation will be served resources after the next whitewash level estimate. Nodes can estimate the growth rate by interpolating the previous gossips until the next round of gossip is completed.

4.5.1 Example for Differential Gossip Algorithm

We will consider a network of 10 nodes. Figure 4.3 shows the topology of the network. Let us assume that minimum initial reputation is 0.03 and maximum initial reputation is 0.30. Let node number 7 and 8 be two nodes with honesty level between .03 and .30 with honesty level 0.127 and 0.224 respectively. Table 4.3 shows the initial reputation offered in each round to both the nodes after every

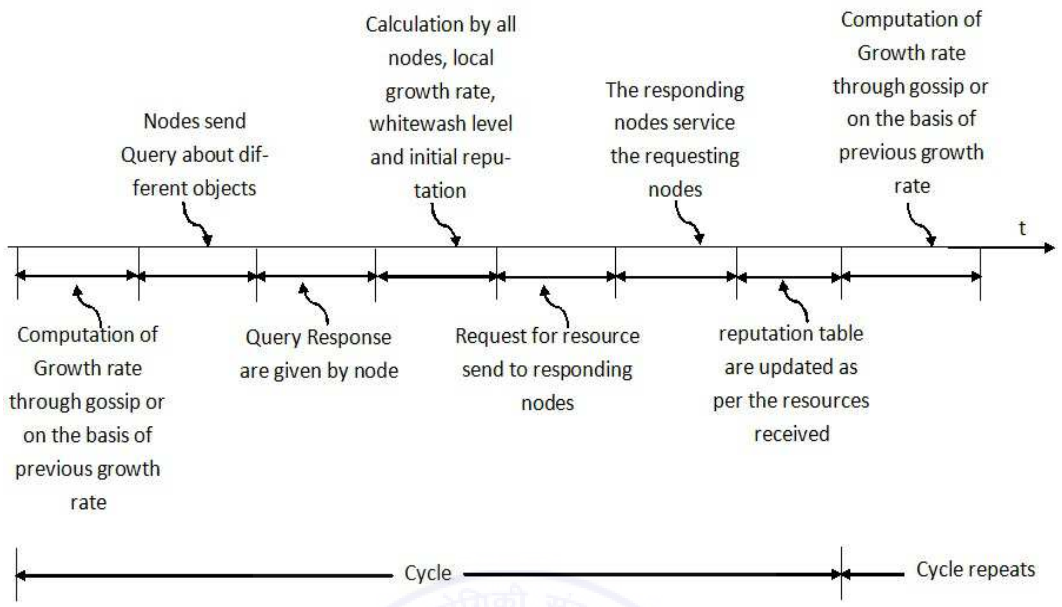
Calculation by all nodes, local growth rate, whitewash level and initial reputation
Computation of Growth rate through gossip or on the basis of previous growth rate
Nodes send Query about different objects
The responding nodes service the requesting nodes
t
Computation of Growth rate through gossip or on the basis of previous growth rate
Query Response are given by node
Request for resource send to responding nodes
reputation table are updated as per the resources received
Cycle
Cycle repeats

Figure 4.2: Timing Diagram

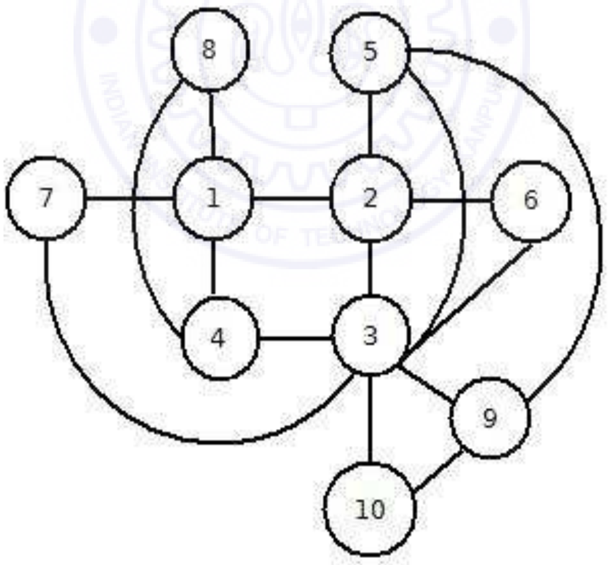

Figure 4.3: Topology of the example network

iteration at each node.

	Honesty level	R_{ini} Offered (1)	R_{ini} Offered (2)
Node 7	0.127	0.192	0.08
Node 8	0.224	0.192	0.08

Table 4.1: R_{ini} Offered to whitewashing nodes

4.5.2 Analysis of Algorithm

While analysing the algorithm, we need to understand that a node has two choices, first, to be cooperative and the second to be non-cooperative. While being non-cooperative, a node have the choice to whitewash to maximize his gain. Usually non-cooperative users will whitewash as that is the only way it can gain from the system. A user will be non-cooperative and hence will also whitewash if its honesty level is lesser than the current initial reputation allotted to any new-comer node. As the current initial reputation level will always lie in between the maximum initial reputation $R_{ini,i,max}$ and minimum initial reputation $R_{ini,i,min}$, we can state the following. All nodes having honesty level higher than $R_{ini,i,max}$, will never whitewash and thus will be always be cooperative. All nodes having honesty level lesser than $R_{ini,i,min}$ will always whitewash. All the nodes having honesty level between $R_{ini,i,max}$ and $R_{ini,i,min}$ will choose to be either cooperative or non cooperative (thus be whitewasher) depending on expected current value of initial reputation.

If in the system, the current value of initial reputation is maintained such that the nodes who are whitewashing in every round will certainly loose to cooperative nodes, the nodes will have no choice but to be cooperative.

The nodes who choose to be non-cooperative, have a choice of whitewashing

B

A		Do whitewash	Don't whitewash
	Do whitewash	$(R_{ini,i,min}, R_{ini,i,min})$	$(\acute{R}, 0)$
	Don't whitewash	$(0, \acute{R})$	$(0, 0)$

Table 4.2: Payoff of a whitewasher - pure strategy

or not. In case whitewashing nodes are less, the whitewashing level will be less and thus $R_{ini,i,n}$ will be set to higher value and vice-versa. Let us understand, what should be natural choice of non-cooperative node - to whitewash or not to whitewash.

To understand the process that is happening with the nodes, we will start with the simple case when only two players have their honesty level below the maximum initial reputation and above the minimum initial reputation. There are three possibilities in this case. First, no node whitewashes, second, one of the nodes whitewash and third when both nodes whitewash. Here, the maximum whitewash level is $\frac{2}{N}$. Hence, in first case $R_{ini,i,n}$ will be $R_{ini,i,max}$, in second case, it will be $\frac{R_{ini,i,max}}{4}$ and in third case, it will be $R_{ini,i,min}$ according to (4.24). Therefore the game matrix will be as follows, Here $\acute{R} = \frac{R_{ini,i,max}}{4}$. By this matrix, it is evident that {Do whitewash, Do whitewash} is the weakly dominant strategy and hence both players will enter in every round and will get a payoff of $R_{ini,i,min}$.

$R_{ini,i,min}$ is a payoff that is too low to whitewash as explained earlier, hence it remains no more lucrative option for such users. Consequently they do not whitewash and their payoff becomes zero.

This zero payoff discourages their non-cooperative, whitewashing tendency and consequently forces them for cooperation. Similar thing will occur in κ player game, i.e. when κ players have their honesty level below the maximum initial reputation. In that case, game will be as follows.

Player set P={nodes with honesty level below maximum initial reputation}

Actions$\in\{S_1, S_2, ..., S_\kappa\}$
here S_i={Do whitewash, Don't whitewash} $\forall i \in \{1, 2, 3, ..., \kappa\}$

$$\begin{aligned} Payoff &= u_i((v_1, ..., v_\kappa)) \\ &= \left(1 - \frac{\sum_j v_j}{\kappa}\right)^2 \cdot R_{ini,n,max} \text{if } R_{ini,i,n} \geq h_i. \end{aligned}$$

Nodes will not whitewash if $R_{ini,n} < h_i$. Here u_i is the pay-off of i^{th} player and v_j is the action performed by j^{th} player such that, $v_j = 1$ if player does the whitewash and 0 otherwise. In this case also, it is evident that the weakly dominant strategy for all the players is to whitewash. This leads to a payoff of $R_{ini,i,min}$. As said earlier $R_{ini,i,min}$ is a payoff that is too low to whitewash and it remains no more lucrative option for these users. Consequently they do not whitewash and their payoff becomes zero. This zero payoff discourages their non-cooperative, whitewashing tendency and consequently forces them for cooperation.

		B	
		Round 1, (p)	Round 2, (1-p)
A	Round 1, (q)	(0,0)	$(\acute{R},\acute{R})$
	Round 2, (1-q)	$(\acute{R},\acute{R})$	(0,0)

Table 4.3: Payoff of a whitewasher - mixed strategy

Users may also randomize their whitewashing action in each period, i.e. they may whitewash in a period with a particular probability. It is easy to observe that if there are κ players then they will randomized over utmost κ rounds to maximize their randomization benefit. Let's examine a two player case when both are randomizing over two periods. The game matrix will be as follows. Here, when both the players are whitewashing simultaneously, their payoff will become $R_{ini,i,min}$ and that will become zero because of reason that both the nodes will find whitewashing non-lucrative and will decide not to whitewash. Let us compute the payoff for player A for whitewashing in round 1 when player B is whitewashing in round 1 and round 2 with probabilities p and $1-p$ respectively,

$$payoff(A,1) = p \cdot 0 + (1-p) \cdot \acute{R}. \tag{4.26}$$

Similarly for entering in round 2,

$$payoff(A,2) = (p) \cdot \acute{R} + (1-p) \cdot 0. \tag{4.27}$$

As player A is randomizing, pay-off in both the round should be same, so on equating both the equation we get $p = \frac{1}{2}$ and in similar fashion $q = \frac{1}{2}$. So the equilibrium probability distribution for p and q both is $\{\frac{1}{2}, \frac{1}{2}\}$.

Now let us examine a three player (A,B,C) case. There are two possibilities – when players are randomizing over two rounds and when players are randomizing for three rounds. Let us assume that, the initial reputations given in some round when 1, 2 or 3 players are entering are $\acute{R}_1$, $\acute{R}_2$ and $\acute{R}_3$ respectively such that $\acute{R}_1 > \acute{R}_2 > \acute{R}_3$. The exact values of $\acute{R}_1$, $\acute{R}_2$ and $\acute{R}_3$ can be calculated using (4.24),

$$\acute{R}_1 = \left(1 - \frac{1}{3}\right)^2 \cdot R_{ini,i,max} = \frac{4}{9} R_{ini,i,max} \tag{4.28a}$$

$$\acute{R}_2 = \left(1 - \frac{2}{3}\right)^2 \cdot R_{ini,i,max} = \frac{1}{9} R_{ini,i,max} \tag{4.28b}$$

$$\acute{R}_3 = \left(1 - \frac{3}{3}\right)^2 \cdot R_{ini,i,max} = R_{ini,i,min} \tag{4.28c}$$

As we know that players have different level of honesty, we assume that A has highest level of honesty and will only whitewash for $\acute{R}_1$, B will whitewash for $\acute{R}_1$ and $\acute{R}_2$ and C will whitewash for all values of initial reputation.

Lets us first consider randomization over three rounds. Probabilities for whitewashing in round 1, 2 and 3 are $p_A, q_A, r_A; p_B, q_B, r_B$ and p_C, q_C, r_C for players A,B and C respectively.

Equating the pay-off of player A for whitewashing in round 1, 2 or 3,

$$(1 - p_B)(1 - p_C)\acute{R}_1 = (1 - q_B)(1 - q_C)\acute{R}_1 = (1 - r_B)(1 - r_C)\acute{R}_1 \tag{4.29}$$

Here it may be noted that player A will only whitewash when neither B nor C are whitewashing. Similarly for player B

$$\begin{aligned}
&(1 - p_A)p_C\acute{R}_2 + (1 - p_C)p_A\acute{R}_2 + (1 - p_A)(1 - p_C)\acute{R}_1 = \\
&(1 - q_A)q_C\acute{R}_2 + (1 - q_C)q_A\acute{R}_2 + (1 - q_A)(1 - q_C)\acute{R}_1 = \\
&(1 - r_A)r_C\acute{R}_2 + (1 - r_C)r_A\acute{R}_2 + (1 - r_A)(1 - r_C)\acute{R}_1
\end{aligned} \tag{4.30}$$

Here it may be noted that player B will only whitewash when only one more or no player is whitewashing.

Similarly for player C

$$\begin{aligned}
&\acute{R}_3 p_B p_A + (1-p_B)p_A\acute{R}_2 + (1-p_A)p_B\acute{R}_2 + \\
&(1-p_B)(1-p_A)\acute{R}_1 = \acute{R}_3 q_B q_A + (1-q_B)q_A\acute{R}_2 + \\
&(1-q_A)q_B\acute{R}_2 + (1-q_B)(1-q_A)\acute{R}_1 = \acute{R}_3 r_B r_A + \\
&(1-r_B)r_A\acute{R}_2 + (1-r_A)r_B\acute{R}_2 + (1-r_B)(1-r_A)\acute{R}_1 \qquad (4.31)
\end{aligned}$$

Here it may be noted that player C will whitewash in all conditions. And

$$p_A + q_A + r_A = p_B + q_B + r_B = p_C + q_C + r_C = 1 \qquad (4.32)$$

Solving above set of equations we get $p_A = p_B = p_C = q_A = q_B = q_C = r_A = r_B = r_C = \frac{1}{3}$. Hence, when randomizing over 3 rounds,the payoff of player A will be $(\frac{4}{9}\acute{R}_1)$, for player B the payoff will be $(\frac{4}{9}\acute{R}_1 + \frac{4}{9}\acute{R}_2)$, and for player C the payoff will be $(\frac{4}{9}\acute{R}_1 + \frac{2}{9}\acute{R}_2 + \frac{1}{9}\acute{R}_2)$. Using equation (4.28), the value of payoff for A, B and C will be $(\frac{16}{81}R_{ini,i,max})$, $(\frac{20}{81}R_{ini,i,max})$ and $(\frac{18}{81}R_{ini,i,max} + \frac{1}{9}R_{ini,i,min})$ respectively.

Similarly if these three players will randomize over two periods with probabilities $p_A, q_A, p_B, q_B, p_C, q_C$ respectively, the probabilities will turn out to be $\frac{1}{2}$. Hence, when randomizing over 2 rounds, the payoff of player A will be $(\frac{1}{4}\acute{R}_1)$, for player B the payoff will be $(\frac{1}{4}\acute{R}_1 + \frac{1}{2}\acute{R}_2)$ and for player C the payoff will be $(\frac{1}{4}\acute{R}_1 + \frac{1}{2}\acute{R}_2 + \frac{1}{4}\acute{R}_2)$. Using equation (4.28), the value of payoff for A, B and C will be $(\frac{4}{36}R_{ini,i,max})$, $(\frac{6}{36}R_{ini,i,max})$ and $(\frac{6}{36}R_{ini,i,max} + \frac{1}{4}R_{ini,i,min})$ respectively.

It is evident from payoffs shown above that players will prefer to randomize over three rounds and not less than three.

Now let us examine for κ players $(A_1, A_2, ..., A_\kappa)$ randomizing over $\acute{\kappa}$ rounds.

Let us assume that, the initial reputations given in some round when 1 to κ players are entering are $\acute{R_1}$ to $\acute{R_\kappa}$ respectively such that $\acute{R_1} > \acute{R_2} > ... > \acute{R_\kappa}$. The exact values of $\acute{R_1}$, $\acute{R_2}$ etc. can be calculated using (4.24). As we know that players have different level of honesty, we assume that A_1 has highest level of honesty and will only accept $\acute{R_1}$, and A_κ will accept all values of initial reputation.

Let us assume that probability for whitewashing in i^{th} round by A_j is p_{ji}. Equating the pay-off of player A_1 for entering in all $\acute{\kappa}$ rounds,

$$\begin{aligned} & (1-p_{21})(1-p_{31})...(1-p_{\kappa 1})\cdot\acute{R_1} \\ = \; & (1-p_{22})(1-p_{32})...(1-p_{\kappa 2})\cdot\acute{R_1} \\ = \; & ... = (1-p_{2\acute{\kappa}})(1-p_{3\acute{\kappa}})...(1-p_{\kappa\acute{\kappa}})\cdot\acute{R_1} \end{aligned} \tag{4.33}$$

Similarly for player A_2

$$\begin{aligned} & \acute{R_2}\cdot[(p_{11})(1-p_{31})...(1-p_{\kappa 1}) + (1-p_{11})(p_{31})...(1-p_{\kappa 1}) \\ & +... + (1-p_{11})(1-p_{31})...(p_{\kappa 1})] + \acute{R_1}\cdot(1-p_{11})(1-p_{31}) \\ & ...(1-p_{\kappa 1}) \\ = \; & \acute{R_2}\cdot[(p_{12})(1-p_{32})...(1-p_{\kappa 2}) + (1-p_{12})(p_{32})...(1-p_{\kappa 2}) + ... + \\ & (1-p_{12})(1-p_{32})...(p_{\kappa 2})] + \acute{R_1}\cdot(1-p_{12})(1-p_{32})...(1-p_{\kappa 2}) \\ = \; & ... = \acute{R_2}\cdot[(p_{1\acute{\kappa}})(1-p_{3\acute{\kappa}})...(1-p_{\kappa\acute{\kappa}}) + (1-p_{1\acute{\kappa}})(p_{3\acute{\kappa}})...(1-p_{\kappa\acute{\kappa}})\acute{R_2} \\ & +... + (1-p_{1\acute{\kappa}})(1-p_{3\acute{\kappa}})...(p_{\kappa\acute{\kappa}})] + \\ & \acute{R_1}\cdot(1-p_{1\acute{\kappa}})(1-p_{3\acute{\kappa}})...(1-p_{\kappa\acute{\kappa}}) \end{aligned} \tag{4.34}$$

and for player A_κ

$$\begin{aligned}
& \acute{R}_\kappa \cdot (p_{11})(p_{21})...(p_{(\kappa-1)1}) + \acute{R}_{\kappa-1} \cdot [(1-p_{11})(p_{21})...(p_{(\kappa-1)1}) + \\
& (p_{11})(1-p_{21})...(p_{(\kappa-1)1}) + (p_{11})(p_{21})...(1-p_{(\kappa-1)1})] + ... + \\
& \acute{R}_2 \cdot [(p_{11})(1-p_{21})...(1-p_{(\kappa-1)1}) + (1-p_{11})(p_{21})... \\
& (1-p_{(\kappa-1)1}) + ... + (1-p_{11})(1-p_{21})...(p_{(\kappa-1)1})] + \\
& \acute{R}_1 \cdot (1-p_{11})(1-p_{21})...(1-p_{(\kappa-1)1}) \\
= \; & \acute{R}_\kappa \cdot (p_{12})(p_{22})...(p_{(\kappa-1)2}) + \acute{R}_{\kappa-1} \cdot [(1-p_{12})(p_{22})...(p_{(\kappa-1)2}) \\
& +(p_{12})(1-p_{22})...(p_{(\kappa-1)2}) + (p_{12})(p_{22})...(1-p_{(\kappa-1)2})] \\
& +\acute{R}_2 \cdot [(p_{12})(1-p_{22})...(1-p_{(\kappa-1)2}) + (1-p_{12})(p_{22})...(1-p_{(\kappa-1)2}) \\
& +... + (1-p_{12})(1-p_{22})...(p_{(\kappa-1)2})] \\
& +\acute{R}_1 \cdot (1-p_{12})(1-p_{22})...(1-p_{(\kappa-1)2}) \\
= \; & ... = \acute{R}_\kappa \cdot (p_{1\acute{\kappa}})(p_{2\acute{\kappa}})...(p_{(\kappa-1)\acute{\kappa}}) + \acute{R}_{\kappa-1} \cdot [(1-p_{1\acute{\kappa}})(p_{2\acute{\kappa}})...(p_{(\kappa-1)\acute{\kappa}}) \\
& +(p_{1\acute{\kappa}})(1-p_{2\acute{\kappa}})...(p_{(\kappa-1)\acute{\kappa}}) + (p_{11})(p_{21})...(1-p_{(\kappa-1)1})] \\
& +... + \acute{R}_2 \cdot [(p_{1\acute{\kappa}})(1-p_{2\acute{\kappa}})...(1-p_{(\kappa-1)\acute{\kappa}}) \\
& +(1-p_{1\acute{\kappa}})(p_{2\acute{\kappa}})...(1-p_{(\kappa-1)\acute{\kappa}}) + ... + \\
& (1-p_{1\acute{\kappa}})(1-p_{2\acute{\kappa}})...(p_{(\kappa-1)\acute{\kappa}})] + \\
& \acute{R}_1 \cdot (1-p_{1\acute{\kappa}})(1-p_{2\acute{\kappa}})...(1-p_{(\kappa-1)\acute{\kappa}}) \qquad (4.35)
\end{aligned}$$

We will also have,

$$\sum_{i=1}^{\acute{\kappa}} p_{ji} = 1; \quad \forall j. \qquad (4.36)$$

Solving above set of equations i.e. (4.33,4.34,4.35,4.36), it turns out that $\forall j, i;\ \ p_{ji} = \frac{1}{\acute{\kappa}}$. Applying similar logic as with three player case we can see that it is optimal for every player to randomize on κ rounds.

This kind of randomization is only possible when whitewashing users whitewash in a coordinated manner. Coordination is required because otherwise the value of κ will remain unknown to whitewashing users. If they whitewash in such a way, the whitewashing level will be reduced κ times.

It is interesting to note that when all users randomize over $\acute{\kappa}$ rounds, a node can unilaterally increase its pay-off by decreasing the number of rounds, it is going to randomize. It will get maximum pay-off when every other node will randomize over $\acute{\kappa}$ rounds and it will whitewash in every period. As every node is interested in its own pay-off, every node will adapt the strategy to whitewash in every round and hence all will get zero pay-off as shown earlier and consequently they will be discouraged for whitewashing.

In figure 4.8, we have plotted simultaneously, the fraction of users having honesty level below $\frac{N_w}{N}$ and initial reputation as function of whitewashing level $\frac{N_w}{N}$. The intersection of these two graphs gives the stable operating point.

As our algorithm periodically estimates maximum whitewashing level, once system attains stability, the minimum reputation whitewashing level will be reduced. This gives new stable operating point Y (as shown in figure 4.8(b)) point. This process will be repeated again and again and over the time, the operating point adjusted to keep the whitewashing level to a negligible value.

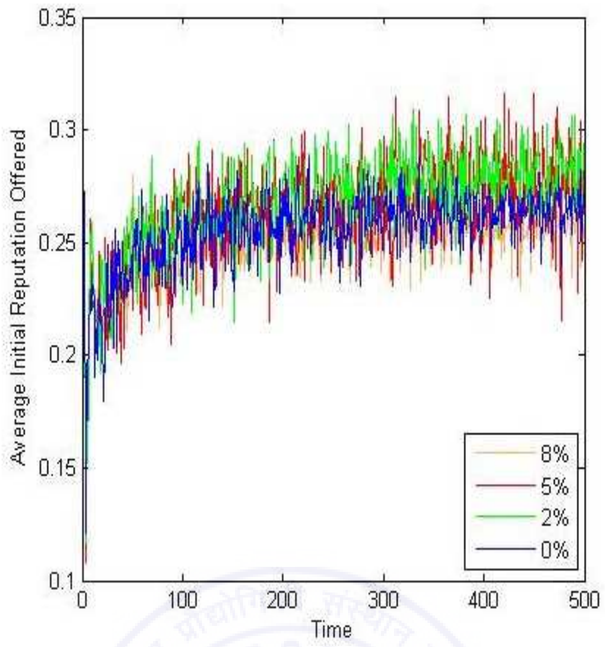

Figure 4.4: Average Initial reputation offered to newcomer nodes

4.6 Numerical Results

Performance evaluation of the proposed method to avoid whitewash has been done for different kind of networks, viz. growing scale-free network based on BA model starting with 1000 nodes with growth rate of 0% per 10 iteration, 2% per 10 iteration, 5% per 10 iteration, 8% per 10 iteration and regular network with 1000, 5000 and 10000 nodes. By growing network, we mean in such a network after every 10 iterations some percentage of new nodes join the network.

We have considered the discrete time instants for the purpose of measurement and estimation in the simulations. Every slot is termed as an iteration. Value of $R_{ini,i,max}$ and $R_{ini,i,min}$ for all nodes has been taken as 0.5 and 0.03 respectively. Thus,

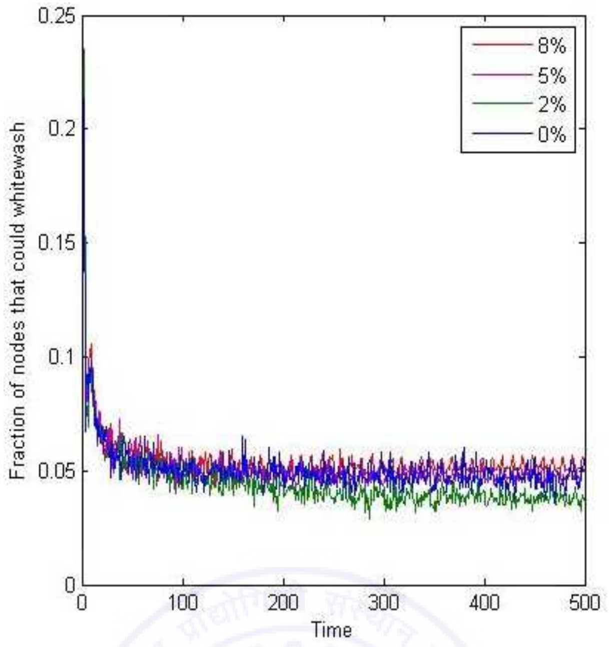

Figure 4.5: The fraction of nodes that whitewash

50% of the nodes are taken to be potential whitewasher initially.

In the first iteration, all potential whitewasher nodes attempt to whitewash to randomly chosen nodes. Every node calculates the level of whitewash and according to that it offers the initial reputation. Nodes, that attempt for whitewash, go for it as per their honesty level and offered initial reputation. From next round onwards, potential whitewashing nodes attempt for whitewash probabilistically and the probability for attempting whitewash for a node depends on the number of times that node is able to whitewash, i.e.,

$$P_{wa} = \frac{number\ of\ times\ nodes\ was\ able\ to\ whitewash}{total\ number\ of\ times\ node\ attempted\ for\ whitewash}. \tag{4.37}$$

Here P_{wa} is the probability by which a node will attempt whitewash. This process is repeated again and again up to 500 iteration.

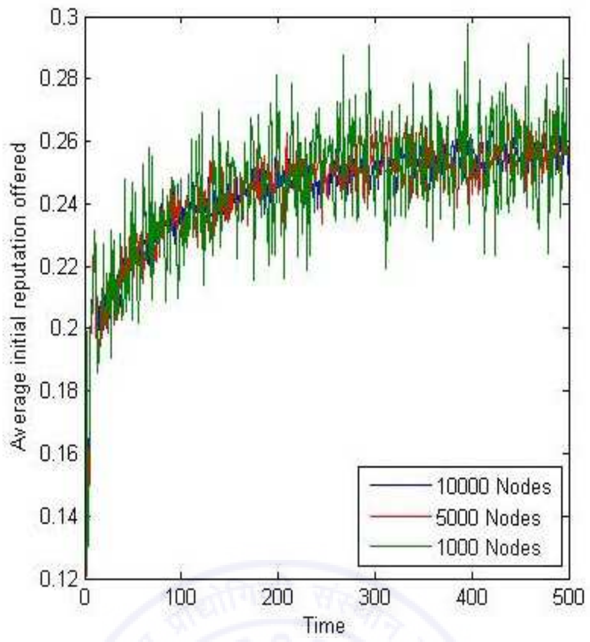

Figure 4.6: Average Initial reputation offered to newcomer nodes

Figure 4.5 and figure 4.7 present the fraction of nodes that could whitewash in every iteration for growing scale-free network and regular networks respectively. Figure 4.4 and figure 4.6 present the average initial reputation offered by nodes in every iteration for growing scale-free network and regular networks respectively.

It is evident in figures that after some iterations, very few nodes could white wash in spite of a reasonable initial reputation being offered to newcomer nodes. This can also be observed from figures that proposed method works for different kind of networks.

In figure 4.8, we have plotted simultaneously, the fraction of users having honesty level below $\frac{N_w}{N}$ and initial reputation as function of whitewashing level $\frac{N_w}{N}$. The intersection of these two graphs gives the stable operating point. The

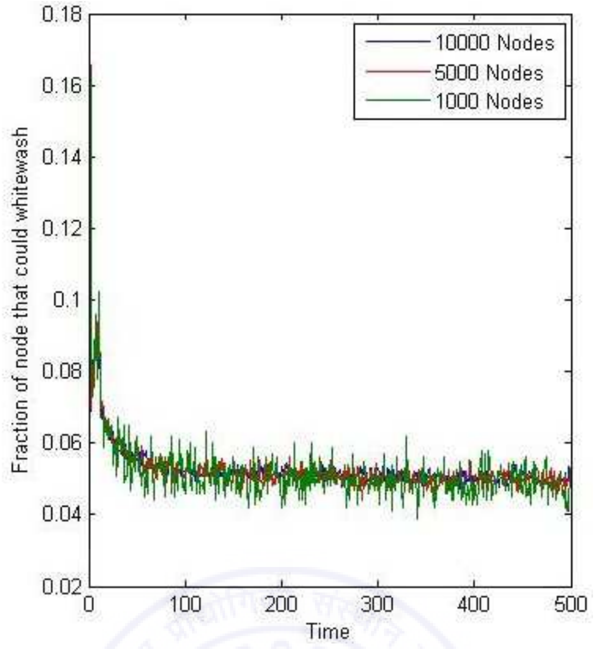

Figure 4.7: The fraction of nodes that whitewash

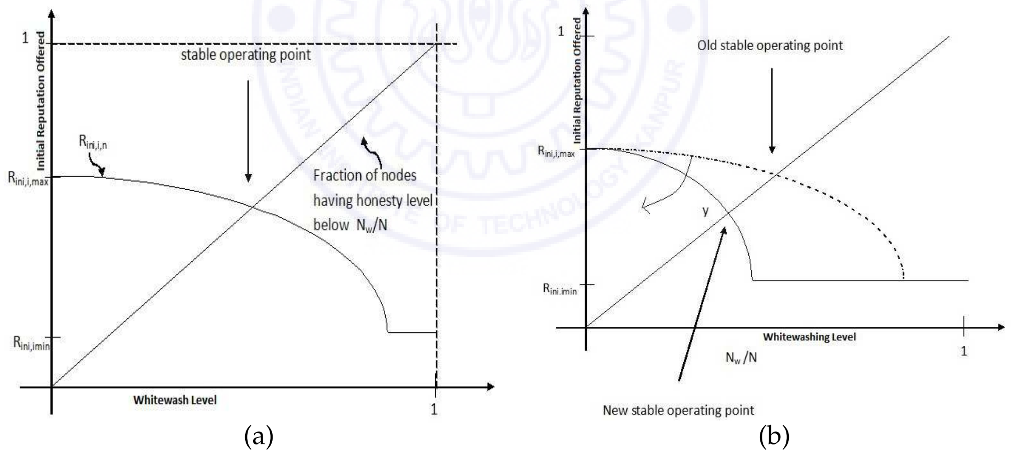
Initial Reputation Offered
1
stable operating point
$R_{ini,i,n}$
$R_{ini,i,max}$
Fraction of nodes having honesty level below N_w/N
$R_{ini,imin}$
Whitewash Level
1
(a)
Initial Reputation Offered
1
Old stable operating point
$R_{ini,i,max}$
y
$R_{ini,imin}$
Whitewashing Level
N_w/N
1
New stable operating point
(b)

Figure 4.8: Change in Maximum whitewashing level

reduction in whitewashing level with time can be explained as follows. As shown in figure 4.8(a), the system will operate at $\frac{N_W}{N}$ where the two curves cross each other. As all the nodes periodically estimate the maximum whitewashing level based on an observation window. The new estimates will be lesser than the earlier ones. This is indicated by the solid line curve towards the left of earlier $R_i ni, i, n$ vs. $\frac{N_W}{N}$ graph (figure 4.8(b)). Thus the operating point moves towards left with time. This keeps on happening till the whitewashing level reaches to negligible value. It may be noted that minimum reputation is 0.03, so all users having honesty level less than 0.03 will always whitewash. Thus whitewashing level will never be less than 0.03.

4.7 Conclusion

Whitewashing tendency of selfish users is a big problem in implementation of reputation management system in peer-to-peer networks. In this chapter, we have proposed a method to avoid the problem of whitewashing. This method adjusts initial reputation value as per the level of whitewash. It is proved theoretically and by simulation that the proposed method is able to avoid whitewash.

Chapter 5

Probabilistic Resource Allocation in Peer-to-Peer Networks

5.1 Introduction

In previous chapters, we have proposed a reputation management system. This system computes the reputation of different nodes in the network so that white-washing and collusion can be avoided. The reputation of a node is computed on the basis of trust values of the node as reported by different nodes that had direct interaction with it. We have also proposed a method to consider the uncertainties involved in measurement of trust.

The reputation computation should be such that it gives enough credit to the nodes those are contributing more to the network. If some node is requested for smaller size resource, it should not be at loss. We are proposing a probabilistic resource allocation method in this chapter. It uses reputation of users while allocating the resources. This kind of allocation ensures that a node with low reputation gets resources with some small finite probability. This will avoid the disconnect between two nodes that have low reputation for each other. This is important when resources are distributed in the network.

The incentive systems proposed in literature do not consider the interests of peers i.e. the neighbourhood of a peer is not made up on the basis of its interests. Although, gnutella servant keeps few last query replying peers in the cache but it is limited to that only. This system is rather inefficient because two nodes may have many interests in common while not being neighbours. Whereas two neighbour nodes may have very few common interests. Hence, the better option is to make neighbourhood on the basis of similarity of interests and reputation.

In this chapter, we have proposed an algorithm for the optimization of shared capacity of a node (section 5.3), a method to compute the reputation (section 5.4), probabilistic resource allocation based on reputation (section 5.4) and server selection according to interests of node and reputation (section 5.5). Finally numerical results are given to verify the hypotheses (section 5.6).

5.2 Related Work

Various groups have suggested different techniques for resource allocation in peer-to-peer networks. Kung *et.al.* [16] proposed selection of a peer for allocation of resource according to its contribution to the network and usage of resources. In the same context, Feldman *et.al.* [55] proposed a new term – generosity of the node. It is estimated as the ratio of the service provided by the node to the service received by the node. Nodes will be served as per their estimated generosity. Banerjee *et.al.* in [44] proposed that a node will calculate the expected utility function for requesting node and on that basis it will decide if service has to be provided or not. In [62, 63] the resource allocation algorithm for single link limited capacity systems has been proposed. These papers considers network as a market and proposes that second price auction leads to optimality. [62] assumes that resource is available everywhere except at the requesting node. Thus, a node is not required to have interaction with many nodes. Ma *et.al.* [64] proposed progressive water filling algorithm on the basis of marginal utility for allocation of resources among different requesting nodes. The base of bucket for water filling is proposed to be varying according to the contribution of requesting node. Ma *et.al.* [65] proposed to allocate the resource to requesting node on the basis of their contribution and requirement of bandwidth. Yan *et.al.* in [40] proposed a ranking based resource allocation scheme. Resource allocation is done according to utility and ranking of requesting peer to ensure max-min fairness.

Social networks are formed on the basis of interests of users. This fact is been capitalised to improve query search as well as recommendation network in peer-to-peer networks [66, 67, 68, 69]. In [66] BitTorrent traces are studied and it is concluded that interest based grouping of peers results in an efficient system. It also proposes a DHT based system to implement this kind of group formation. Wang *et.al.* [68] proposed interest based online social communities that are headed by super nodes and nodes join the communities according to their interests. These communities will have a trust relationship among its members. In [69] a friend network is proposed on the basis of similarity of interests.

5.3 Capacity Sharing

As every node in the network is rational, hence it will try to share minimum amount of resource to increase its pay-off. If there is a reputation management system implemented in the network, nodes are compelled to share the resources to get the quality of service from the network. Nodes, being rational, will try to optimize the amount of shared resources. We propose a method for nodes to optimize the shared capacity to get the required quality of service from network. In this method, nodes will initially share some amount of resource. This amount of shared resource will be periodically reviewed and adjusted for optimality.

Initially a node will share the capacity as per its perceived download requirement. By perceived download requirement, we mean a rough estimate of its average download requirement. This need not to be accurate as it will be updated later on. But, it should neither be too low to ruin the reputation of a newcomer node nor be too high to cause a cost penalty. If no estimate is available, initially half of total download capacity can be shared.

Node will tweak the value of its shared capacity by periodically increasing and decreasing it by some amount δ to get the optimal point where it will get maximum advantage. While doing so, node will follow the following method.

1. If decrease in sharing capacity does not decrease significant average download, it implies that node is sharing more than required resource and hence it should decrease it.
2. If increase in sharing capacity increases significant average download, it

implies that node is sharing less than required resource and hence it should increase it.

3. If decrease in sharing capacity decreases significant average download, it implies that either node was on optimal point (if it was preceded by an increase) so it should get back to that point or is now sharing even lesser than what it should have shared.

4. If increase in sharing capacity does not increases significant average download, it implies that either node was on optimal point (if it was preceded by an decrease) or is now sharing even more than what is required.

This process is shown in algorithm 4.

ϵ is a parameter that is kept for overcoming the effect of demand variation in the network. When node observes high variation of the demand in the network, value of ϵ will be increased.

5.4 Reputation Based System

Network is only meaningful if nodes are interacting with each other and contributing to each others' interest. We have a network of nodes that are rational in nature. Such nodes contribute in the network only when they have some incentive for doing so. To avoid this problem, a reputation based incentive system can be used. In such a system nodes keep the record of behaviour o f other nodes observed by itself or on the basis of recommendation of different nodes. This kind

Algorithm 4 Shared upload capacity adjustment of a node

```
$k = 0$; and $A(k) = -1$ {k is the instant when node reviews its sharing capacity
and A(k) is the indicator variable which shows the action taken at a particular
k}
repeat
   $D_k \leftarrow$ average data download for $kT$ to $(k+1)T$
   $U_s = U_s + \delta \cdot A(k)$ {$U_s$ is shared capacity of the node}
   if $|D_k - D_{k-1}| \leq \epsilon$ then
      if $A(k) = A(k-1)$ or A(k)=0 then
         $A(k+1) \leftarrow -1$
      else
         if $A(k) = 1$ && $A(k-1) = -1$ then

            $A(k+1) \leftarrow 0$
         end if
      end if
   else
      if $A(k) = -1$ && $A(k-1) = 1$ && $D_k < D_{k-1}$ then

         $A(k+1) \leftarrow 1$
      end if
      if $D_k > D_{k-1}$ then
         $A(k+1) \leftarrow 1$
      end if
   end if
   $k \leftarrow \mod_5(k+1)$
until Node is in the network
```

of system forces rational nodes to contribute to the network. To implement this kind of a system, we need to formulate a way for estimation of reputation and a way for allocation of resource according to the estimated reputation.

5.4.1 Reputation Management System

Ideally, reputation should be the measure of cooperative behaviour of a node which is an abstract quantity and it is a private information of a node. So, it is difficult to measure the cooperative behaviour of a node and we can only measure its implications with some degree of uncertainty. However, it can be estimated with certain accuracy on the basis of behaviour observed by a node.

There could be a number of ways to observe the behaviour of a node. One such method may be to use the ratio of received data rate to requested data rate. The advantage of such technique is that if some node is asking for less amount of data, the serving node will not earn a bad reputation. Moreover this kind of reputation remains between 0 and 1 as given in 2.3, [70], i.e.,

$$t_{ij} = \left(\frac{q_{a,ji}}{min(q_{ij,ay}, q_{f,ji})}\right)^{1-\eta_i} \times \frac{\hat{q}_{w,ji}}{q_{r,ji}}, \tag{5.1}$$

hence it is easy to handle. Here t_{ij} is the reputation of node j for node i, $q_{r,ij}$, $\hat{q}_{w,ji}$, $q_{a,ji}$, $q_{f,ji}$ and $q_{ay,ji}$ are the requested, estimated willing, actual, feasible and accepted service rates respectively. The disadvantage of this kind of system is that it does not takes into account the amount of request. It means that if a node is asking for less amount of resource from a node and more amount of resource from another node and both are fulfilling node's demand, both will get similar gain in reputation. However, the node that was requested more resource, had to pay more in comparison to the other one.

This problem can be taken care of by giving different weights to different transactions as per the amount of resource requested by that node. Weights should be such that these should range between 0 to 1 and biggest service request should get maximum weight. Requesting node has a fixed download capacity that is generally the maximum of its download requirement.

A node can calculate the reputation of a node with following formulation,

$$t_{ij} = \left(\frac{q_{a,ji}}{min(q_{ij,ay}, q_{f,ji})}\right)^{1-\eta_i} \times \frac{\hat{q}_{w,ji}}{q_{r,ji}} \times \frac{q_{r,ji}}{q_{r,i,d}}. \tag{5.2}$$

Here $q_{r,i,d}$ is the download capacity of node i. It may be noted that we have multiplied the factor $\frac{q_{r,ji}}{q_{r,i,d}}$ to the t_{ij} as estimated by equation 2.8 in chapter 2.

Keeping the download capacity ($q_{r,i,d}$) instead of requested resource ($q_{r,ji}$) in denominator, we can overcome the above mentioned problem. However, $q_{r,i,d}$ is quite large compared to $q_{r,ji}$, this will make reputation values very low. Apart from it, every node has a different value for $q_{r,i,d}$ and it will be a problem for aggregation because a node having same kind of behaviour with two nodes of different download capacities will have different value for the t_{ij}. To over come these problems nodes will multiply their reputation table with ($q_{r,i,d}/Q_{r,d}$). Here $Q_{r,d}$ is the universal scaling factor known by all the nodes.

However, this will only work well if a node is only interested in resources that have size of same order. If some node 'A' is rich with resources that are small in size, this node will have a very small value of reputation for a node that is requesting for all size of resources because 'A' would have been asked only for small amount of resource. Consequently, when 'A' asks for small amount of resource, it will get the resource with a very small probability. This issue will be further investigated in next subsection.

5.4.2 Probabilistic Resource Allocation

In probabilistic resource allocation, node probabilistically decides if it will provide the resource to requesting node or not. It means, when a node 'A' requests for resource from node 'B', node 'B' checks the reputation table and converts reputation of 'A' to its effective reputation. Here, by effective reputation we mean reputation that is adjusted according to the requested amount of resource. To calculate this value, node multiplies the reputation value with ratio of its download capacity to requested amount of resource, i.e.

$$t_{ij,effective} = t_{ij} \times \frac{q_{r,i,d}}{q_{r,ij}}. \tag{5.3}$$

Now in the proposed system, if a node is asking similar amount of resource as it supplied, it will be given same quality of service. If it asks for smaller resource than it supplied, it gets even better quality of service whereas if opposite happens, it gets a poor quality of resource.

Once node i gets the effective reputation of a node j, it selects the node j with probability proportional to its reputation. It means node i generates a random number. If this generated number is smaller than the reputation of requesting node i.e. node j here, multiplied by a constant (ν_i), requesting node is selected to provide the resource. ν_i is the constant that ensures the requirement of selected nodes remains around the shared capacity so that it can be optimally utilised.

Mathematically,

$$P_{allocation,i,j} = \begin{cases} \acute{P}_{allo} & \text{if } \acute{P}_{allo} < 1. \\ 1, & \text{otherwise.} \end{cases} \tag{5.4}$$

Where,

$$\acute{P}_{allo} = (t_{ij,effective})^x \cdot v_i$$

Here $P_{allocation,i,j}$ is the probability by which j will be allocated the resource. Node will be selected for resource allocation if,

$$rand \leq P_{allocation,i,j} \tag{5.5}$$

Here *rand* is the random number generated by the node and x is the reputation exponent. It is used because a low reputation node can only increase its reputation if it serves with $x < 1$. Its value has been calculated in [71].

Nodes will dynamically and periodically adjust the value of v_i to get the optimality. To do so, node will measure the utilised part of its shared capacity and fulfilment level of demand of selected nodes. If it is not able to utilise its shared capacity regularly, it increases the value of v_i and if demand of selected node is not fulfilled over the time, value of v_i is decreased.

After the selection of nodes, the shared capacity is distributed among selected nodes. If the total demand of requesting nodes is less than the shared capacity of serving node, every node is allocated resource as per their requirement. If this total demand is greater than the shared capacity, node needs to use some kind of allocation algorithm such that, the serving node can get maximum advantage and nodes can not play game by asking for resources greater than their requirement.

A node will be maximum benefited when it has highest chance of getting selected for resource allocation i.e. by maximizing its $P_{allocation}$ for a future time when it will need some resource, according to equation (5.2). If serving node is doing this calculation, it can be assumed that $q_{a,ij} = min(q_{ji,ay}, q_{f,ij})$. Hence equation reduces to,

$$t_{ji,effective} = \frac{\hat{q}_{w,ij}}{q_{r,ij}} \times \frac{q_{r,ij}}{q_{r,ji}}. \tag{5.6}$$

$\hat{q}_{w,ij}$ can be replaced by $q_{o,ij}$ as we are discussing about allocation in a particular round where number of nodes, their demands and shared capacity has already been fixed. Hence equation (5.6) reduces to,

$$t_{ji,effective} = \frac{q_{o,ij}}{q_{r,ji}}. \tag{5.7}$$

$q_{r,ji}$ can only be predicted statistically. We can observe that if a node 'A' is asking less amount of resource to node 'B' then 'B' can only get less amount of resource because if it will demand for bigger resource, its effective reputation will come down and hence it will not be selected for service by node 'A'. Hence if A is asking lesser resource from B, that implies B is asking lesser resource from A.

For simplicity v can be taken as 1. Therefore, the optimisation problem a node needs to solve becomes,

$$\max \sum_j (t_{ji,effective})^x \implies \max \sum_j \left(\frac{q_{o,ij}}{q_{r,ij}}\right)^x. \tag{5.8}$$

Such that

$$\begin{aligned}\sum_j q_{o,ij} &= U_{s,i}\\ q_{o,ij} &\leq q_{r,ij};\quad \forall j\\ \sum_j q_{r,ji} &> U_{s,i}\end{aligned}$$

Here x is a constant. Its value lies between 0 and 1. For our case its value is 0.75.

This is a difficult optimisation problem. For that, we need to observe the function a^x. Here a varies from 0 to some finite value and x is a constant between 0 and 1 as mentioned above. Two facts are easy to observe, first it is a monotonically increasing function and second its rate of change is monotonically decreasing function. Therefore it is evident that initially our objective function will get the maximum increment if resource is allocated to node that corresponds to smallest $q_{r,ij}$. After some allocation, increase in the value of objective function will decrease and now it will be more for any other node that has requested more data than first one. Now it will be beneficial to allocate data to this second node. After some allocation, any other node may result in more increment and this process continues till the resource allocation is complete.

On the basis of reason mentioned above, we propose an algorithm for allocation. First a node i decides about the minimum unit of allocation. Let us call this Δ_i. On the basis of Δ_i and the amount of total resource shared ($U_{s,i}$), i calculates the total number of allocation units ($U_{su,i}$) such that $U_{su,i} = \frac{U_{s,i}}{\Delta_i}$. Now, i constructs an allocation array U_{sua} that has the dimension of $U_{su,i} \times \acute{N}$. Here $\acute{N}$ is the number of nodes selected for data allocation by equation (5.5) such that

$$U_{sua}(k,j) = ((k)^x - (k-1)^x) \cdot \left(\frac{\Delta_i}{q_{r,ij}}\right)^x. \tag{5.9}$$

Here k and j are row and column indices of U_{sua}. Elements of U_{sua} are sorted and indices of top $U_{su,i}$ element are stored in a vector of dimension $1 \times U_{su,i}$. The number of times any particular node comes in this vector will be allocated the same number of units.

In this kind of allocation, nodes asking for less amount of data will be given data first. Hence if a node asks for more data than its requirement, it loses the allocation part. This kind of allocation will also fulfil our second requirement.

As requests will be coming temporally in arbitrary fashion, it is necessary to define a policy followed by a node for provisioning the service. If node will serve the request as and when it comes, node will always remain busy in doing so. Moreover, nodes that has got the bandwidth, will get allocation again and again. If node will service the requests periodically, there is a chance with finite probability that a low reputation node will get the service while a high reputation node may keep on waiting.

Hence, a node should have a dynamic policy about serving instants. It means that when total reputation of requesting nodes crosses a certain threshold, node will serve the accumulated requests. If over a certain period of time total reputation of requesting nodes does not cross the threshold, node will serve the requests accumulated by this time. While summing up the reputation of requesting nodes, it is ensured that high reputation nodes get preferred to the nodes of lower reputation. Whenever, a node serves new requests, node will first do the selection process for newcomer nodes and then it will redistribute the resources among newly selected nodes and already existing nodes.

5.5 Server Selection

5.5.1 Common Interest Groups

In peer-to-peer file sharing network, different users have common interests. For a user, it is beneficial to make neighbours that share interests with him and ready to serve him. Therefore, a node should adopt a strategy to form its neighbourhood according to similar interests with good reputation nodes.

Interest is an abstract notion so classification of nodes on the basis of interest is difficult. Even if it is done, this will be a very large set that will be difficult to handle. Therefore, interest group should be formed on the basis of files, users requested or provided. However, users with different interests may request same file. For example, a song may be liked for different reasons like music, singer or lyrics. But, if two users are requesting for more and more similar files, probably they may have some common interests. As the number of similar files grows, probability of two peers choosing file due to same interest increases while choosing it for different interests decreases.

Therefore, we propose that a node will compute the similarity coefficient of the other nodes in the network. The similarity coefficient (χ_{ij}) of node j will be calculated by node i using

$$\chi_{ij} = \begin{cases} v_{ij} \cdot log_{base_i}(\Omega_{ij} + 1) & \text{if } \Omega_{ij} < base_i. \\ v_{ij}, & \text{otherwise.} \end{cases} \tag{5.10}$$

Here Ω_{ij} is the number of times node i has queried to node j or vice-versa, v_{ij} is the ratio of answered queries to total queries between node i and node j. $base_i$ will be dynamically adjusted periodically as per the accuracy of similarity coefficient of the node. It means if the selected neighbours can not answer sufficient number of requests, value of $base_i$ will be increased.

5.5.2 Inclusion of Reputation in Neighbourhood Formation

As discussed earlier, for server selection, a node need to form its neighbourhood using interests and reputation. This can be done by combining reputation (t_{ij}) and similarity coefficient (χ_{ij}) for node j. The combined score can be used to rank the other nodes in the network. This rank can be used to select the server i.e. where to send queries.

The combining can be done as follows.

$$score_{ij} = \alpha \cdot \chi_{ij} + (1 - \alpha) \cdot t_{ij}. \tag{5.11}$$

Here α is a combination coefficient between 0 and 1. Value of α will depend upon the stability of common interest network. If a node has newly joined the network, it has to build the interest network hence α will be taken high. Once it has a stable interest network, value of α will be decreased to have more contribution of reputation in the score.

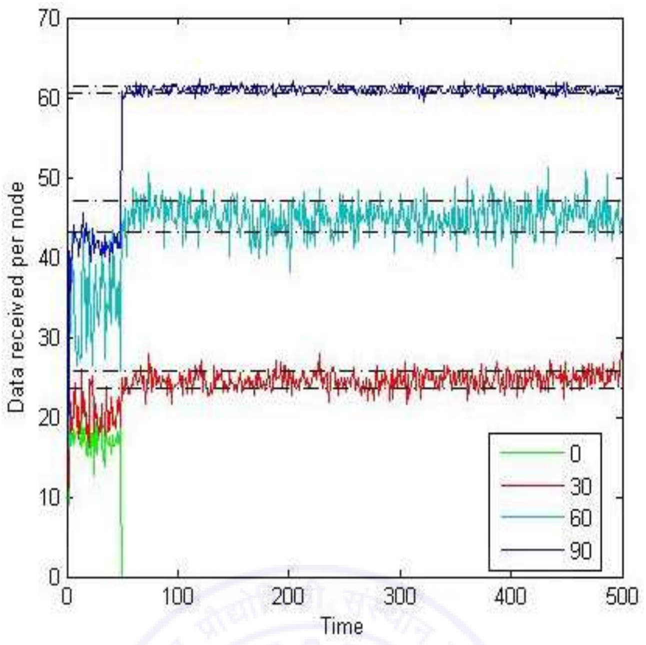

Figure 5.1: Average data received per node within the same network. Different graphs show nodes with different shared capacities

5.6 Numerical Results

We have done performance evaluation of reputation system and resource allocation system for a network of 200 nodes. We have also evaluated interest based group formation algorithm for a network of 1000 nodes. We have considered the discrete time instants for the purpose of measurement and estimation in the simulations. Every slot is termed as an iteration. First 50 iterations have been taken as an acquaintance period i.e. a node will allocate their bandwidth without referring to the reputation table.

Figure 5.1 presents the average data received by nodes sharing different amount of resource to the network. Here, it is evident from figure that the node that is

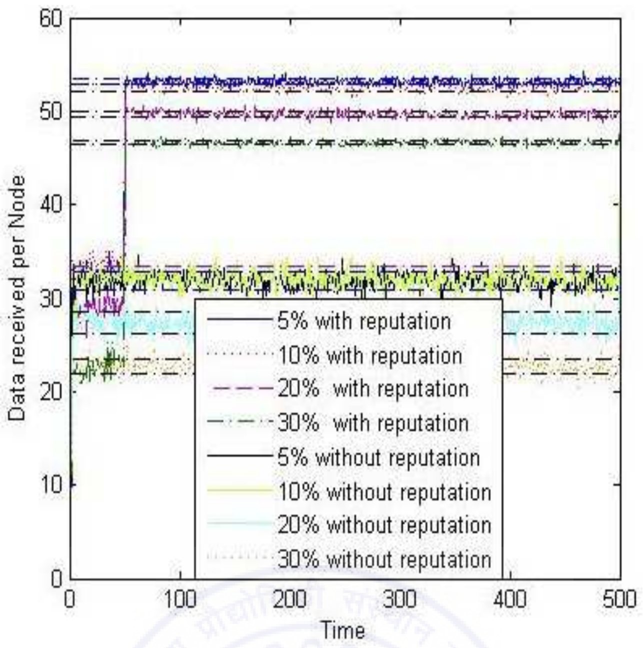

Figure 5.2: System performance with different percentage of free riders.

sharing more data, is getting better quality of service. Figure 5.2 shows the performance of system in presence of different percentage of free riders. We can see in figure that from 5% to 10% decay in system performance is almost negligible. After that, system performance decreases by small amount. So we can say that system performance does not deteriorate much due to free riders.

Figure 5.3 shows the data received by peers asking for different amount of data in the network. Here, BS represents the nodes that request for the amount of resource as per its requirement whereas GS1 and GS2 represents the nodes that requests the amount of resource multiple time to their requirement. GS2 requests more times than GS1. Here it can be seen that nodes making request as per their requirement are getting better quality of service whereas nodes that are trying to exploit network by making requests multiple times are not getting that kind of

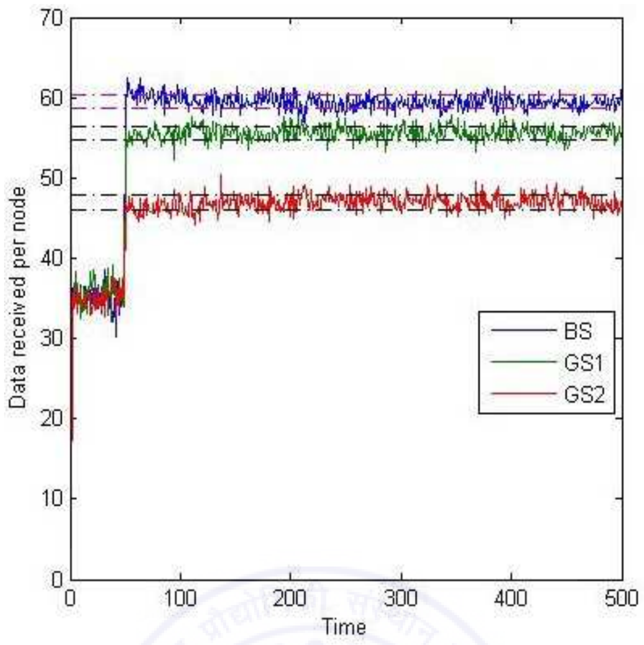

Figure 5.3: Data received by nodes with different strategies

quality of service. This discourages the tendency of exploitation of by making multiple time requests.

Figure 5.4 shows the average number of nodes queried required for resolution of query in interest based and non-interest based network. Here, it can be seen that, if node forms interest groups, its query gets resolved in much lesser number of hops than number of hops in other case.

5.7 Conclusion

In this chapter, we have discussed allocation of resource by node on the basis of reputation. Allocation has been done probabilistically, i.e., requesting node

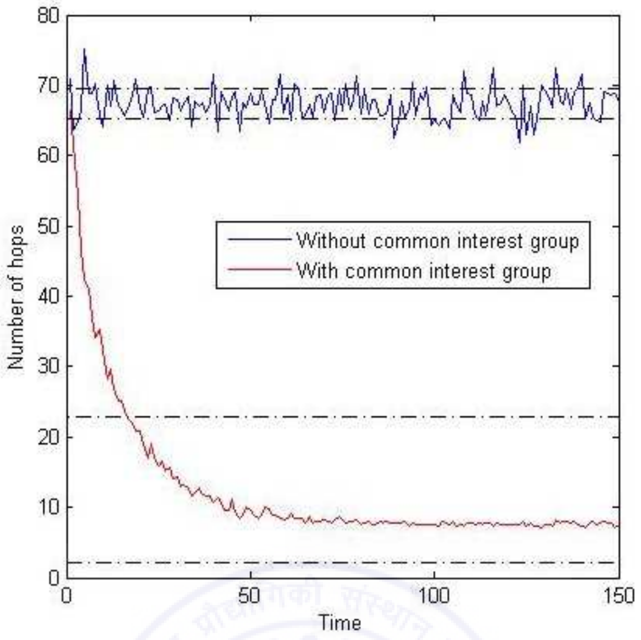

Figure 5.4: Average number of nodes queried required for query resolution

has been offered resource with a probability proportional to its reputation. If total demand of selected nodes is more than offering node's shared capacity, allocation will be done to optimise the gain in reputation of offering node. An algorithm has been proposed for the same. This algorithm also ensures that nodes do not request more than their actual demand. An algorithm for formation of common interest group and shared capacity optimisation has also been proposed. Numerical results show that proposed algorithms work as per the requirement.

Chapter 6

Conclusions and Future Works

6.1 General Conclusions

Good scalability and absence of single point of failure has made peer-to-peer architecture very attractive for system designers. But, distributed nature of peer-to-peer networks poses many challenges. Free riding is one such challenge. Designing a mechanism which provides incentives to non free riders and disincentives to free riders is a solution to this problem and this incentive mechanism should be based on reputation of a peer. Reputation management systems are used to maintain such reputations. On implementation of such system free riders start deceiving the system by activities like collusion and whitewashing. In this thesis a reputation management system was proposed that can overcome the problem of free riding, whitewash and collusion. This chapter discusses the major conclusions of the work presented in the thesis.

6.2 Estimation of Trust

Generally measurement of trust value involves uncertanities that are not considered by existing reputation management systems. These uncertanities include, congestion effect, non consideration of offers that were made but not accepted and varying allocations by serving node because of variation in load. This chapter takes care these uncertanities and proposes a estimator based on BLUE.

Simulation results show that taking care of uncertanities by proposed estimation algorithm leads to better estimation of trust and greater utilisation of resources.

6.3 Aggregation of Trust

Distributed nature of peer-to-peer networks and their large size make aggregation process complex and resource consuming. This chapter describes a variation of gossip algorithm i.e. differential gossip algorithm that aggregates the information at different nodes in scale-free network based on PA model in $(log_2N)^2$ time. Here N is the number of nodes in the network. The said algorithm also gives different weights to opinion of nodes in the network on the basis of their relationship with aggregating node. This avoids collusion in the reputation management.

The said bound has been proved theoretically and numerical results vindicate it. Analysis of collusion has also been done and numerical results show the reduction in collusion.

6.4 Avoiding Whitewashing

Implementation of reputation management system to discourage free riding induces non cooperative users to whitewash so that they can exploit the system. This chapter describes an algorithm to avoid whitewashing tendency of non cooperative users. In this algorithm initial reputation is proposed to vary as per the level of whitewash in the network. Theoretical analysis of algorithm suggests that it can discourage the whitewashing tendency of users substantially. Numerical results confirm the same thing for different kind of networks.

6.5 Resource Allocation

This chapter describes a resource allocation system that allocates probabilistically on the basis of reputation of a node. The proposed system also ensures that if some node is serving large resource it should get more credit for it. At the same time if some node is serving small resource and asking for small resource, it should also be taken care of. Algorithm also discourages the nodes for making unnecessarily big requests, greater than their requirement. This chapter also proposes an algorithm for forming common interest groups so that query of nodes can be resolved in less number of hops. Shared capacity optimisation for a node is also proposed in this chapter. Further, numerical results are presented that verify the performance of algorithm.

6.6 Future Works

This thesis mainly concentrate on design of reputation management system for peer-to-peer networks to avoid the problem of free riding. Reputation management system for peer-to-peer networks involves many issues. In this thesis, we have selected few of them. Rest of the issues can be a part of further study. In specific, following issues can be further explored.

(1) While estimating trust, we are not aware of probability distribution that uncertainty will follow so we have used BLUE. The probability distribution of uncertainty can be modelled by observing real system and consequently a better trust estimator can be obtained.

(2) Weights given to different nodes while aggregation are w_{ij} such that $w_{ij} = a_i^{b_{ij} \cdot t_{ij}}$. Here the values of a_i and b_{ij} are taken as constant in this work. The dynamic evaluation and adjustment of a and b on the basis of quality of service received by node from network can further improve the algorithm, and could be investigated.

(3) In gossiping it is assumed that nodes know the start of gossip. This is difficult to ensure. The proposed algorithm may also work for asynchronous case with minor modifications. These modifications and a theoretical proof for the algorithms can be pursued further.

(4) More efficient techniques for gossiping can be designed like push-then-pull gossip. This may improve the efficiency of the system.

(5) It is assumed that no peer is behind proxy. But generally this is not the case. To avoid this problem few peers that are not behind proxy must act as reflector. This will lead to some cost penalty to these peers. Hence there should be an incentive mechanism for peer acting like reflectors as well.

(6) Verification of the proposed reputation management system in the real life network is important. This can be taken up to test the system performance on different size of networks. If the observed behaviour does not match the theoretical prediction, further theoretical study will be needed.
(7) Pay-off analysis of cooperative and non cooperative peers in more general setting can also be done.

Bibliography

[1] Eytan Adar and Bernardo A. Huberman. Free riding on gnutella. *First Monday*, 5:2000, 2000.

[2] Stefan Saroiu, P. Krishna Gummadi, and Steven D. Gribble. A measurement study of peer-to-peer file sharing systems. In *Multimedia Computing and Networking (MMCN)*, 2002.

[3] Daniel Hughes, Geoff Coulson, and James Walkerdine. Free riding on gnutella revisited: The bell tolls? *IEEE Distributed Systems Online*, 6(6):1, June 2005.

[4] Steven M. Kay. *Fundamentals of statistical signal processing: estimation theory*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1993.

[5] Ion Stoica, Robert Morris, David Karger, M. Frans Kaashoek, and Hari Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. *SIGCOMM Comput. Commun. Rev.*, 31(4):149–160, August 2001.

[6] Antony I. T. Rowstron and Peter Druschel. Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. In *Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms*

Heidelberg, Middleware '01, pages 329–350, London, UK, UK, 2001. Springer-Verlag.

[7] Ben Y. Zhao, Ling Huang, Jeremy Stribling, Sean C. Rhea, Anthony D. Joseph, and John D. Kubiatowicz. Tapestry: A resilient global-scale overlay for service deployment. *IEEE Journal on Selected Areas in Communications*, 22:41–53, 2004.

[8] Anthony J.Howe. Napster and gnutella: a comparison of two popular peer-to-peer protocols, 2002.

[9] Christos Gkantsidis, Milena Mihail, and Amin Saberi. Random walks in peer-to-peer networks: algorithms and evaluation. *Perform. Eval.*, 63(3):241–263, March 2006.

[10] Wesley W. Terpstra, Jussi Kangasharju, Christof Leng, and Alejandro P. Buchmann. Bubblestorm: resilient, probabilistic, and exhaustive peer-to-peer search. *SIGCOMM Comput. Commun. Rev.*, 37(4):49–60, August 2007.

[11] Martin J. Osborne. *Introduction to Game Theory: International Edition*. Oxford University Press, 2009.

[12] YangBin Tang, HuaiMin Wang, and Wen Dou. Trust based incentive in p2p network. In *Proceedings of the E-Commerce Technology for Dynamic E-Business, IEEE International Conference*, CEC-EAST '04, pages 302–305, Washington, DC, USA, 2004. IEEE Computer Society.

[13] Gnutella website. http://www.gnutella.com/, January 2007.

[14] Michael Sirivianos, Jong H. Park, Rex Chen, and Xiaowei Yang. Free-riding in BitTorrent Networks with the Large View Exploit. In *IPTPS*, 2007.

[15] Chiranjeeb Buragohain, Divyakant Agrawal, and Subhash Suri. A game theoretic framework for incentives in p2p systems. In *Proceedings of the 3rd International Conference on Peer-to-Peer Computing*, P2P '03, pages 48–, Washington, DC, USA, 2003. IEEE Computer Society.

[16] HT Kung and Chun-Hsin Wu. Differentiated admission for peer-to-peer systems: Incentivizing peers to contribute their resources. In *Proceedings of the 1st Workshop on Economics of Peer-to-Peer Systems*.

[17] Kevin Lai, Michal Feldman, Ion Stoica, and John Chuang. Incentives for cooperation in peer-to-peer networks. In *Workshop on Economics of Peer-to-Peer Systems*, 2003.

[18] T. G. Papaioannou and G. D. Stamoulis. An incentives' mechanism promoting truthful feedback in peer-to-peer systems. In *Proceedings of the Fifth IEEE International Symposium on Cluster Computing and the Grid - Volume 01*, CCGRID '05, pages 275–283, Washington, DC, USA, 2005. IEEE Computer Society.

[19] Philippe Golle, Kevin Leyton-Brown, and Ilya Mironov. Incentives for sharing in peer-to-peer networks. In *Proceedings of the 3rd ACM conference on Electronic Commerce*, EC '01, pages 264–267, New York, NY, USA, 2001. ACM.

[20] Li Xiong and Ling Liu. Peertrust: Supporting reputation-based trust for peer-to-peer electronic communities. *IEEE Trans. on Knowl. and Data Eng.*, 16(7):843–857, July 2004.

[21] Sepandar D. Kamvar, Mario T. Schlosser, and Hector Garcia-Molina. The eigentrust algorithm for reputation management in p2p networks. In *Proceedings of the 12th international conference on World Wide Web*, WWW '03, pages 640–651, New York, NY, USA, 2003. ACM.

[22] Runfang Zhou, Kai Hwang, and Min Cai. Gossiptrust for fast reputation aggregation in peer-to-peer networks. *IEEE Transactions on Knowledge and Data Engineering*, 20(9):1282–1295, 2008.

[23] Runfang Zhou and Kai Hwang. Powertrust: A robust and scalable reputation system for trusted peer-to-peer computing. *IEEE Trans. Parallel Distrib. Syst.*, 18(4):460–473, April 2007.

[24] Zhengqiang Liang and Weisong Shi. Pet: A personalized trust model with reputation and risk evaluation for p2p resource sharing. In *Proceedings of the Proceedings of the 38th Annual Hawaii International Conference on System Sciences - Volume 07*, HICSS '05, pages 201.2–, Washington, DC, USA, 2005. IEEE Computer Society.

[25] Shanshan Song, Kai Hwang, Runfang Zhou, and Yu-Kwong Kwok. Trusted p2p transactions with fuzzy reputation aggregation. *IEEE Internet Computing*, 9(6):24–34, November 2005.

[26] http://www.ebay.com.

[27] M. Castro, P. Druschel, A. Ganesh, A. Rowstron, , and D. S. Wallach. Security for structured peer-to-peer overlay networks. In *Multimedia Computing and Networking*, MMCN '02, 2002.

[28] Rafit Izhak-Ratzin, Hyunggon Park, and Mihaela van der Schaar. Online learning in bittorrent systems. *IEEE Trans. Parallel Distrib. Syst.*, 23(12):2280–2288, December 2012.

[29] Eric J. Friedman and Paul Resnick. The social cost of cheap pseudonyms. *Journal of Economics and Management Strategy*, 10:173–199, 2000.

[30] Michal Feldman, Christos Papadimitriou, John Chuang, and Ion Stoica. Free-riding and whitewashing in peer-to-peer systems. In *Proceedings of the ACM SIGCOMM workshop on Practice and theory of incentives in networked systems*, PINS '04, pages 228–236, New York, NY, USA, 2004. ACM.

[31] Michal Feldman and John Chuang. The evolution of cooperation under cheap pseudonyms. In *Proceedings of the Seventh IEEE International Conference on E-Commerce Technology*, CEC '05, pages 284–291, Washington, DC, USA, 2005. IEEE Computer Society.

[32] Mao Yang, Zheng Zhang, Xiaoming Li, and Yafei Dai. An empirical study of free-riding behavior in the maze p2p file-sharing system. In *Proceedings of the 4th international conference on Peer-to-Peer Systems*, IPTPS'05, pages 182–192, Berlin, Heidelberg, 2005. Springer-Verlag.

[33] Weihong Wang and Baochun Li. To play or to control: a game-based control-theoretic approach to peer-to-peer incentive engineering. In *Proceedings of the 11th international conference on Quality of service*, IWQoS'03, pages 174–192, Berlin, Heidelberg, 2003. Springer-Verlag.

[34] Rob Sherwood, Seungjoon Lee, and Bobby Bhattacharjee. Cooperative peer groups in nice. *Comput. Netw.*, 50(4):523–544, March 2006.

[35] Qiao Lian, Yu Peng, Mao Yang, Zheng Zhang, Yafei Dai, and Xiaoming Li. Robust incentives via multi-level tit-for-tat: Research articles. *Concurr. Comput. : Pract. Exper.*, 20(2):167–178, February 2008.

[36] Debojyoti Dutta, Ashish Goel, Ramesh Govindan, and Hui Zhang. The design of a distributed rating scheme for peer-to-peer systems. In *WORKSHOP ON ECONOMICS OF PEER-TO-PEER SYSTEMS*, 2003.

[37] Thanasis G. Papaioannou and George D. Stamoulis. Reputation-based policies that provide the right incentives in peer-to-peer environments. *Comput. Netw.*, 50(4):563–578, March 2006.

[38] Sergio Marti and Hector Garcia-Molina. Limited reputation sharing in p2p systems. In *Proceedings of the 5th ACM conference on Electronic commerce*, EC '04, pages 91–101, New York, NY, USA, 2004. ACM.

[39] Nazareno Andrade, Francisco Brasileiro, Walfredo Cirne, and Miranda Mowbray. Discouraging free riding in a peer-to-peer cpu-sharing grid. In *Proceedings of the 13th IEEE International Symposium on High Performance Distributed Computing*, HPDC '04, pages 129–137, Washington, DC, USA, 2004. IEEE Computer Society.

[40] Yonghe Yan, Adel El-atawy, and Ehab Al-shaer. Ranking-based optimal resource allocation in peer-to-peer networks. In *26th Annual IEEE International Conference on Computer Communications (INFOCOM)*, 2007.

[41] Yingjie Xia, Guanghua Song, Yao Zheng, and Mingzhe Zhu. R2p: A peer-to-peer transfer system based on role and reputation. In *Proceedings of the First*

International Workshop on Knowledge Discovery and Data Mining, WKDD '08, pages 136–141, Washington, DC, USA, 2008. IEEE Computer Society.

[42] Anna Satsiou and Leandros Tassiulas. Reputation-based resource allocation in p2p systems of rational users. *IEEE Trans. Parallel Distrib. Syst.*, 21(4):466–479, April 2010.

[43] Hou Mengshu, Lu Xianliang, Zhou Xu, and Zhan Chuan. A trust model of p2p system based on confirmation theory. *SIGOPS Oper. Syst. Rev.*, 39(1):56–62, January 2005.

[44] Dipyaman Banerjee, Sabyasachi Saha, Sandip Sen, and Prithviraj Dasgupta. Reciprocal resource sharing in p2p environments. In *Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*, AAMAS '05, pages 853–859, New York, NY, USA, 2005. ACM.

[45] A. A. Selcuk, E. Uzun, and M. R. Pariente. A reputation-based trust management system for p2p networks. In *Proceedings of the 2004 IEEE International Symposium on Cluster Computing and the Grid*, CCGRID '04, pages 251–258, Washington, DC, USA, 2004. IEEE Computer Society.

[46] Jitendra Padhye, Victor Firoiu, Don Towsley, and Jim Kurose. Modeling tcp throughput: a simple model and its empirical validation. *SIGCOMM Comput. Commun. Rev.*, 28(4):303–314, October 1998.

[47] David Kempe, Alin Dobra, and Johannes Gehrke. Gossip-based computation of aggregate information. In *Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science*, FOCS '03, pages 482–, Washington, DC, USA, 2003. IEEE Computer Society.

[48] Albert-László Barabási and Réka Albert. Emergence of Scaling in Random Networks. *Science*, 286(5439):509–512, October 1999.

[49] Béla Bollobás, Oliver Riordan, Joel Spencer, and Gábor Tusnády. The degree sequence of a scale-free random graph process. *Random Struct. Algorithms*, 18(3):279–290, May 2001.

[50] Stefan Saroiu, Krishna P. Gummadi, and Steven D. Gribble. A Measurement Study of Peer-to-Peer File Sharing Systems. In *Proceedings of Multimedia Computing and Networking 2002 (MMCN '02)*, San Jose, CA, USA, January 2002.

[51] Ruchir Gupta and Yatindra Nath Singh. Reputation aggregation in peer-to-peer network using differential gossip algorithm. *CoRR*, abs/1210.4301, 2012.

[52] Stephen Boyd, Arpita Ghosh, Balaji Prabhakar, and Devavrat Shah. Randomized gossip algorithms. *IEEE/ACM Trans. Netw.*, 14(SI):2508–2530, June 2006.

[53] Flavio Chierichetti, Silvio Lattanzi, and Alessandro Panconesi. Rumor spreading in social networks. *Theoretical Computer Science*, 412(24):2602–2610, 2011.

[54] Geoffrey R. Grimmett. *Probability: An Introduction*. Oxford University Press, Oxford, UK, 1986.

[55] Michal Feldman, Kevin Lai, Ion Stoica, and John Chuang. Robust incentive techniques for peer-to-peer networks. In *Proceedings of the 5th ACM conference*

on Electronic commerce, EC ’04, pages 102–111, New York, NY, USA, 2004. ACM.

[56] Zaki Malik and Athman Bouguettaya. Reputation bootstrapping for trust establishment among web services. *IEEE Internet Computing*, 13(1):40–47, January 2009.

[57] Emmanuelle Anceaume, Maria Gradinariu, and Aina Ravoaja. Incentives for p2p fair resource sharing. In *Proceedings of the Fifth IEEE International Conference on Peer-to-Peer Computing*, P2P ’05, pages 253–260, Washington, DC, USA, 2005. IEEE Computer Society.

[58] Jianguo Chen, Huijuan Lu, and Stefan D. Bruda. A solution for whitewashing in p2p systems based on observation preorder. In *Proceedings of the 2009 International Conference on Networks Security, Wireless Communications and Trusted Computing - Volume 02*, NSWCTC ’09, pages 547–550, Washington, DC, USA, 2009. IEEE Computer Society.

[59] C. H. Zuo, J. F. Zhou, and H. C Feng. A novel multi-level trust model to improve the security of p2p network. In *3rd IEEE International Conference on Computer Science and Information Technology*, (ICCSIT), pages 100–104, 2010.

[60] Xiao Yu and Satoshi Fujita. Whitewash-aware reputation management in peer-to-peer file sharing systems. In *The 2012 World Congress in Computer Science, Computer Engineering, and Applied Computing*, WORLDCOMP’12, 2012.

[61] Béla Bollobás, Oliver Riordan, Joel Spencer, and Gábor Tusnády. The degree sequence of a scale-free random graph process. *Random Struct. Algorithms*, 18(3):279–290, May 2001.

[62] Anna Satsiou and Leandros Tassiulas. Reputation-based resource allocation in p2p systems of rational users. *IEEE Trans. Parallel Distrib. Syst.*, 21(4):466–479, April 2010.

[63] M. Meo and F. Milan. A rational model for service rate allocation in peer-to-peer networks. In *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*, pages 1–5, 2006.

[64] Richard T. B. Ma, Sam C. M. Lee, John C. S. Lui, and David K. Y. Yau. Incentive and service differentiation in p2p networks: a game theoretic approach. *IEEE/ACM Trans. Netw.*, 14(5):978–991, October 2006.

[65] Huiye Ma and Ho fung Leung. A demand and contribution based bandwidth allocation mechanism in p2p networks: A game-theoretic analysis. In *Advanced Information Networking and Applications, 2006. AINA 2006. 20th International Conference on*, volume 1, pages 1005–1010, 2006.

[66] Guoxin Liu, Haiying Shen, and L. Ward. An efficient and trustworthy p2p and social network integrated file sharing system. In *Peer-to-Peer Computing (P2P), 2012 IEEE 12th International Conference on*, pages 203–213, 2012.

[67] Tzu-Ming Wang, Wei-Tsong Lee, Tin-Yu Wu, Hsin-Wen Wei, and Yu-San Lin. New p2p sharing incentive mechanism based on social network and game theory. In *Advanced Information Networking and Applications Workshops (WAINA), 2012 26th International Conference on*, pages 915–919, 2012.

[68] Qingjie Wang, Jianrong Wang, Jian Yu, Mei Yu, and Yan Zhang. Trust-aware query routing in p2p social networks. *Int. J. Commun. Syst.*, 25(10):1260–1280, October 2012.

[69] R. Farahbakhsh, N. Crespi, A. Cuevas, S. Adhikari, M. Mani, and T. Sanguankotchakorn. socp2p: P2p content discovery enhancement by considering social networks characteristics. In *Computers and Communications (ISCC), 2012 IEEE Symposium on*, pages 000530–000533, 2012.

[70] Ruchir Gupta and Yatindra Nath Singh. Trust estimation in peer-to-peer network using blue. *CoRR*, abs/1304.1649, 2013.

[71] Ruchir Gupta and Yatindra Nath Singh. A reputation based framework to avoid free-riding in unstructured peer-to-peer network. *CoRR*, abs/, 2013.

LIST OF PUBLICATIONS

Communicated Papers:

1. **Ruchir Gupta** and Yatindra Nath Singh, "Reputation Aggregation in Peer-to-Peer Network Using Differential Gossip Algorithm", (Submitted for review on 5th April 2013 to *IEEE Transactions on Knowledge and Data Engineering*), (Paper available on Cornell University Research Repository since 16th October 2012, http://arxiv.org/abs/1210.4301).

2. **Ruchir Gupta** and Yatindra Nath Singh, "Trust Estimation in Peer-to-Peer Network Using BLUE", (Submitted for review on 19th May April 2013 to *IEEE Transactions on Knowledge and Data Engineering*), (Paper available on Cornell University Research Repository since 5th April 2013, http://arxiv.org/abs/1304.1649).

3. **Ruchir Gupta** and Yatindra Nath Singh, "Avoiding Whitewashing in Unstructured Peer-to-Peer Resource Sharing Network", (Submitted for review on 18th July 2013 to *IEEE Transactions on Knowledge and Data Engineering*), (Paper available on Cornell University Research Repository since 18th July 2013, http://arxiv.org/abs/1307.5057).