

# Global Peer Ranking Methods to Handle the Malicious Peers and Free-Riders in Peer-to-Peer Networks

*A Thesis Submitted*

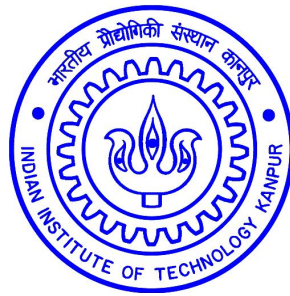
in Partial Fulfilment of the Requirements

for the Degree of

**DOCTOR OF PHILOSOPHY**

*by*

**SATEESH KUMAR AWASTHI**



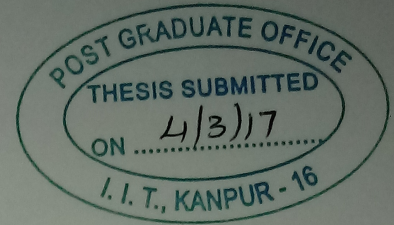
*to the*

**DEPARTMENT OF ELECTRICAL ENGINEERING**

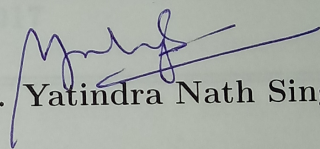
**INDIAN INSTITUTE OF TECHNOLOGY KANPUR**

**March, 2017**

## CERTIFICATE



It is certified that the work contained in the thesis entitled “ **Global Peer Ranking Methods to Handle the Malicious Peers and Free-Riders in Peer-to-Peer Networks**” being submitted by **Mr. Sateesh Kumar Awasthi** 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.

  
Dr. Yatindra Nath Singh

Professor

Department of Electrical Engineering

Indian Institute of Technology Kanpur

Kanpur, INDIA

04<sup>th</sup> March, 2017

# Synopsis

---

---

Name of the Student	: <b>Sateesh Kumar Awasthi</b>
Roll Number	: <b>13104186</b>
Degree for which submitted	: <b>Ph.D.</b>
Department	: <b>Electrical Engineering</b>
Thesis Title	: <b>Global Peer Ranking Methods to Handle the Malicious Peers and Free-Riders in Peer-to-Peer Networks</b>
Thesis Supervisor	: <b>Dr. Yatindra Nath Singh</b>
Month and year of submission	: <b>March, 2017</b>

---

---

The peer-to-peer (P2P) networks leads to a significant amount of traffic on Internet due to its inherent advantages over traditional client-server networks, viz., scalability, robustness and diversity of data. On one hand, its open and anonymous environment gives everyone an opportunity to interact with the others, at the same time, it also brings new security threats. The malicious peers or the peers having conflict of interest can easily put inauthentic contents in the network. This can easily sabotage the system. Further, lacks of central control may lead to the problem of free-riding, i.e., peers download the resources without contributing anything to the networks. This leads the large difference between upload and download amount of resources in the peers. In such a situation, downloading speed for non-free-riders becomes very slow. Thus, efficient methods and policies to discourage the malicious and free-rider peers are needed.

Maintaining the reputation system could be one of the methods to handle the

malicious peers. This method has been studied by many researchers [1], [2], [3] in past. In this method, each peer evaluates the other peers and assigns them some trust value called local trust. These local trust values need to be aggregated in the network. The aggregated local trust is called global trust. The global trust is understood to be the trust, the system as a whole keeps on a peer. This is also called reputation of the peer. For the convergence of aggregation, local trust matrix needs to be stochastic, which requires the normalization of local trust. The process of aggregation of local trust is motivated by Google's PageRank [4] algorithm which is based on the popularity of page on the web. But, trustworthiness and popularity are different notions. In this thesis, we examine the problems with normalization of local trust and proposed a new algorithm, 'Absolute trust' to resolve them. The Absolute trust can rank the peers according to their trustworthiness and can also give their absolute characterization. We proved that the global trust vector will always converge at a certain unique value.

Free-riding can be avoided by implementing an incentive mechanism. For this purpose, many incentive methods [5], [6], [7], [8], [9], [10], have been proposed in recent years. Among these, global approaches are considered better, because peers' cooperation is considered in the whole network. But implementation of a global approach is not trivial. To make the implementation simpler, we proposed a light-weight algorithm based on Biased Contribution Index (BCI). The BCI converge faster than the other existing global incentive mechanism. The BCI is also able to balance the upload and download amount of resources for each peer.

The global incentive mechanisms are based on the iterative calculation of contribution index. We analyzed the problems with iterative calculation and proposed simplified form of BCI named SBCI. The SBCI is very simple to implement in a network. We also proposed and simulated the peer selection method based on well known 'the stability

of marriage problem' [11].

Based on the above, the thesis has been organized in the following seven chapters.

Chapter 1, defines the brief history and introduction of Internet and P2P networks. The classification of P2P networks based on central dependency and overlay network has also been discussed. We have also explained some Distributed Hash Table (DHT) protocols in this chapter. The advantages and challenges in P2P networks have also been discussed. The detail of experimental survey has been presented to identify the problem.

In Chapter 2, we explained some basic definitions and a brief introduction of some models presented in past. The other related work has also been summarized in this chapter.

In Chapter 3, we present the trust aggregation algorithm called Absolute trust. First we define the local trust and after that, we derived, intuitively, the formula for global trust. We have shown that the proposed global trust exists and have unique value. This can be calculated by iterative method and thus, can be implemented in a distributed system. The Absolute trust algorithm is evaluated through simulated experiments and compared with the other existing algorithm. Simulation results have been presented in the same chapter.

In Chapter 4, we present a generalized analysis of convergence of Absolute trust. We have derived the proof of convergence mathematically and gave some numerical example to justify it.

In Chapter 5, we address the problem of unfairness and free-riding in P2P network. We present a mechanism named Biased Contribution Index (BCI). In this mechanism,

the contribution of peers are biased in such a way that they are motivated to download from low contributing peers and upload to high contributing peers. As a result, upload and download amount in each peer gets balanced. We have also given the solution of BCI and justification of fairness in this chapter. The BCI can be calculated by iterative method and can also be implemented in a distributed system. Finally the BCI is evaluated through simulation and compared with the other mechanism.

In Chapter 6, we present a simplified form of BCI named Simplified Biased Contribution Index (SBCI). We consider some design rules and define the formula for SBCI. In this formula, peers are motivated to choose the transacting partner in same way as in BCI. The iterative calculations are not needed in SBCI unlike in BCI and in the other methods. Thus, it is very simple to implement in the network. We have given the mathematical justification for design rules for fairness. The SBCI is also evaluated through simulation in this chapter. We have given two different methods for peer selection and compared the simulation results with the other mechanisms in the same chapter.

In Chapter 7, we conclude the thesis and present some open problems for possible future work.

*Dedicated to*  
*my*  
*Family, Friends*  
*and*  
*Teachers*

# Acknowledgements

I feel immense pleasure in expressing my heartfelt sense of gratitude to my thesis supervisor Dr. Y. N. Singh under whose supervision and inspiring guidance, I carried out my research work. I am really fortunate to have an advisor who gave me the freedom to explore my own research, same time the guidance to recover when things become out of my control. Under his guidance, I learnt how to manage the overall work on priority basis. I am indebted to him for his constant and ungrudging encouragement, valuable suggestions, ingenious ideas and for improving my technical writing. He showed me different ways to approach a research problem and the need to be persistent to accomplish any goal.

I wish to express my deep gratitude to Prof. Govind Sharma, Prof. R. K. Bansal and Prof. Aditya K. Jagannatham for enhancing my basic knowledge of core subjects.

I express my heartiest thanks to Prof. S. N. Singh, Head, Department of Electrical Engineering, IIT Kanpur, for providing me necessary facilities. I am also grateful to all the faculty members and staff for their support and encouragement.

I immensely express my heartiest thanks to my friends Anupam Soni, Kumar Gaurav, Nitin Singha, Pallavi Athe, V Sateeshkrishna Dhuli, Rameshwar Nath Tripathi,



Varsha Lohani, Rajesh Bhatt, Ruchir Gupta, Amit Munjal and Ashutosh Singh for their support and help. I have spent some of the craziest and most memorable moments with them.

I would also like to acknowledge the contribution of all those who are directly or indirectly associated with me. I express my sincere thanks to all of them.

Though it is beyond the scope of any acknowledgement for all that I have received from my family members, I express my heartfelt and affectionate gratitude to them.

The encouragement and support from my wife Swati and our son Sankalp are a powerful source of inspiration and energy. A special thought is devoted to my parents and parents in-laws for their patience and never ending support.

**(Sateesh Kumar Awasthi)**

# List of Abbreviations

P2P	Peer-to-peer
DHT	Distributed Hash Table
SHA	Secure Hash Algorithm
ID	Identity Document
CAN	Content Addressable Network
IP	Internet Protocol
ET	EigenTrust
PT	PowerTrust
AT	Absolute Trust
GC	Global Contribution
TFT	Tit-for-tat
BCI	Biased Contribution Index
SBCI	Simplified Biased Contribution Index
TTL	Time to live
AAD	Average of Absolute Deviation

# List of Symbols

$N$	Number of nodes in the network.
$\mathbf{T}$	Trust matrix with its $ij$ element as the local trust of peer $i$ evaluated by peer $j$ .
$sat(i, j)$	Number of satisfactory transactions, peer $i$ has had with peer $j$ .
$unsat(i, j)$	Number of unsatisfactory transactions, peer $i$ has had with peer $j$ .
$max(a, b)$	Maximum among $a$ and $b$ .
$\mathbf{T}^{norm}$	Normalized trust matrix.
$\mathbf{t}$	Global trust vector.
$\mathbf{T}^{tr}$	Transpose of matrix $\mathbf{T}$ .
$\mathbf{C}$	Incidence matrix corresponding to $\mathbf{T}^{tr}$ .
$n_g$	Number of satisfactory files.
$n_n$	Number of average or neutral files.
$n_b$	Number of unsatisfactory files.
$n_t$	Total number of files.
$w_g$	Weight factor for satisfactory file.
$w_n$	Weight factor for average or neutral file.
$w_b$	Weight factor for unsatisfactory file.
$frac\_sat$	Fraction of satisfactory files.
$frac\_unsat$	Fraction of unsatisfactory files.
$p, q,$	Scalars
$\alpha, \beta$	Scalars
$a, b, c, d$	Scalars
$a_i, b_i$	Scalars
$\mathbf{e}_i$	Row vector with $i^{th}$ entry as 1 and all others are zero.
$\mathbf{e}$	Column vector with each entry as '1'.
$\mathbf{diag}(\mathbf{t})$	Diagonal matrix with $ii^{th}$ element as $t_i$ .
$\mathbf{u}, \mathbf{v}, \mathbf{u}_i, \mathbf{v}_i$	Eigenvectors
$\mathbf{A}, \mathbf{B}, \mathbf{A}_i$	$N \times N$ matrix
$\mathbf{M}, \mathbf{M}_i$	$N \times N$ matrix
$\mathbf{X}, \mathbf{Y}, \mathbf{Z}$	$N \times N$ matrix
$\lambda, \lambda_i, \gamma_i$	Eigenvalues.
$t_{Th}$	Threshold value of Global trust for selection of source peers.

---

$\gamma$	Zipf's Constant
$\mathbf{S}$	Share matrix with its $ij^{th}$ entry as amount of resource shared by peer $i$ to peer $j$ .
$\mathbf{S}(\mathbf{t}_n)$	Share matrix with its $ij^{th}$ entry as amount of resource shared by peer $i$ to peer $j$ at time $t_n$ .
$R$	Biased Upload to Download Ratio
$R(t_n)$	Biased Ratio at time $t_n$
$\mathbf{x}$	Contribution index vector.
$\mathbf{x}(\mathbf{t}_n)$	Contribution index vector at time $t_n$ .

# Contents

List of Figures	xviii
-----------------	-------

List of Tables	xxi
----------------	-----

<b>1 Introduction</b>	<b>1</b>
1.1 Introduction . . . . .	1
1.2 Definition of P2P Networks . . . . .	3
1.3 Classification of P2P Networks . . . . .	3
1.3.1 Centralized P2P Networks . . . . .	3
1.3.2 Decentralized P2P Networks . . . . .	4
1.3.3 Hybrid P2P Networks . . . . .	4
1.3.4 Unstructured P2P Networks . . . . .	5
1.3.5 Structured P2P Networks . . . . .	5
1.4 Distributed Hash Table (DHT) . . . . .	5
1.4.1 Chord Protocol . . . . .	6
1.4.2 CAN Protocol . . . . .	7

---

1.5	Advantages of P2P Networks . . . . .	9
1.6	Problems in P2P Networks . . . . .	10
1.6.1	Malicious Peers . . . . .	10
1.6.2	Free-riders . . . . .	11
1.7	Experimental Studies . . . . .	11
1.8	Solution of Malicious Peers and Free-riders . . . . .	12
1.8.1	Reputation System . . . . .	12
1.8.2	Incentive Mechanism . . . . .	13
1.9	Organization of Thesis . . . . .	14
<b>2</b>	<b>Background and Related Work</b>	<b>15</b>
2.1	Introduction . . . . .	15
2.2	Basic Definitions . . . . .	15
2.3	Related Work . . . . .	16
2.3.1	Related Work in Reputation System . . . . .	16
2.3.2	Related Work in Incentive Mechanism . . . . .	18
2.3.3	EigenTrust Algorithm . . . . .	19
2.3.4	Power Trust . . . . .	20
2.3.5	Flow-Based Reputation . . . . .	21
2.3.6	Global Contribution Approach to Maintain Fairness . . . . .	21
<b>3</b>	<b>Absolute Trust: Algorithm for Aggregation of Trust in Peer-to-Peer</b>	

<b>Networks</b>	<b>24</b>
3.1 Introduction . . . . .	24
3.2 Motivation . . . . .	25
3.3 Proposed Trust Model . . . . .	28
3.3.1 Local Trust . . . . .	29
3.3.2 Absolute Trust: Algorithm for Aggregation . . . . .	30
3.4 Existence and Uniqueness of Global trust . . . . .	35
3.5 Analysis of Algorithm . . . . .	45
3.5.1 Implementation in Distributed System . . . . .	45
3.5.2 Speed of Convergence . . . . .	47
3.5.3 Message Overhead and Time Complexity . . . . .	48
3.6 Experimental Evaluation . . . . .	52
3.6.1 Simulation Setup . . . . .	52
3.6.2 Performance of Absolute Trust for Various Settings of Parameters	55
3.6.3 Comparison of Absolute Trust with Other Reputation Systems .	57
3.7 Conclusion . . . . .	63
<b>4 Generalized Analysis of Convergence of Absolute Trust</b>	<b>65</b>
4.1 Introduction . . . . .	65
4.2 Motivation . . . . .	66
4.3 P2P Model and Absolute Trust . . . . .	66

---

4.4	Analysis of Convergence of Absolute Trust . . . . .	67
4.4.1	Center Point of the Matrix . . . . .	67
4.4.2	Properties of Center Point . . . . .	71
4.4.3	Convergence of Absolute Trust for Large Error in Initial Guess . .	73
4.5	Numerical Results . . . . .	75
4.5.1	Convergence of the Center Point . . . . .	76
4.5.2	Convergence of the Global Trust . . . . .	76
4.6	Conclusion . . . . .	76
<b>5</b>	<b>Biased Contribution Index: A Distributed Mechanism to Ensure Fairness in P2P Networks</b>	<b>80</b>
5.1	Introduction . . . . .	80
5.2	Motivation . . . . .	81
5.3	Network Model and Biased Contribution Index . . . . .	82
5.4	Solution of BCI and Justification For Fairness . . . . .	84
5.4.1	Solution of BCI . . . . .	84
5.4.2	Comparison of BCI With Global Contribution Approach . . . . .	89
5.4.3	Justification For Fairness . . . . .	89
5.5	Analysis of BCI . . . . .	91
5.5.1	Solution Of Free-riding . . . . .	91
5.5.2	Implementation In Distributed System and Time Complexity . . .	92
5.6	Simulation Results . . . . .	95



5.7	Conclusion . . . . .	98
<b>6</b>	<b>Simplified Biased Contribution Index (SBCI) for Fair and Efficient P2P Network</b>	<b>101</b>
6.1	Introduction . . . . .	101
6.2	Motivation . . . . .	102
6.3	Proposed Incentive model . . . . .	105
6.3.1	Design Rules to Ensure the Fair and Efficient P2P Network . . . .	105
6.3.2	Simplified Biased Contribution Index . . . . .	106
6.3.3	Justification For Design Rules . . . . .	107
6.3.4	Justification For Fairness . . . . .	111
6.4	Analysis of Algorithm . . . . .	114
6.4.1	Implementation in Distributed System . . . . .	114
6.4.2	Message Overhead, Storage Capacity and Time Complexity . . . .	114
6.5	Transaction Procedure for Maximum Efficiency . . . . .	115
6.5.1	Simple Procedure For Peer Selection . . . . .	115
6.5.2	College Admission and The Stability of Marriage Based Approach For Peer Selection . . . . .	117
6.6	Experimental Evaluation . . . . .	120
6.6.1	Simulation Setup . . . . .	120
6.6.2	Evaluation Metrics . . . . .	121
6.6.3	Simulation Results of Simple Procedure For Peer Selection . . . .	122

---

6.6.4	Simulation Results of College Admission and The Stability of Marriage Based Approach For the Peer Selection . . . . .	129
6.7	Conclusion . . . . .	136
<b>7</b>	<b>Conclusion and Future Work</b>	<b>137</b>
7.1	Conclusion . . . . .	137
7.2	Major Contributions . . . . .	140
7.3	Future Work . . . . .	140
	<b>Bibliography</b>	<b>142</b>
	<b>List of Publications</b>	<b>153</b>

# List of Figures

1.1	The Chord Protocol for $m = 4$ bit long identifier . . . . .	8
1.2	The CAN protocol for 2-d space . . . . .	9
3.1	Different ways of evaluation . . . . .	31
3.2	Authentic download for different threshold values of global trust ( $t_{Th}$ ) and $\alpha$ . . . . .	56
3.3	Rejected transactions, i.e., good peers are rejecting good peers, for different threshold values of global trust ( $t_{Th}$ ) and $\alpha$ . . . . .	58
3.4	Authentic download for different settings of node population, ( $\alpha = 1/3, t_{Th} = 4.5$ ) . . . . .	59
3.5	Intermediate values of % of authentic download in the presence of pure malicious peers, ( $\alpha = 1/3$ ) . . . . .	59
3.6	Performance in the presence of pure malicious peers, ( $\alpha = 1/3$ ) . . . . .	60
3.7	Performance in the presence of unpredictable peers: 10% of peers are purely malicious and few behave as good peers upto some time then behave maliciously after that, ( $\alpha = 1/3$ ) . . . . .	61
3.8	Half of the unpredictable peers changing their behavior from 2000 <sup>th</sup> transaction and rest from 4000 <sup>th</sup> transaction, ( $\alpha = 1/3$ ) . . . . .	62

3.9	Performance in the presence of malicious collectives, ( $\alpha = 1/3$ ) . . . . .	63
3.10	Standard deviation of load distribution among the peers in different Reputation Systems, ( $\alpha = 1/3$ ) . . . . .	64
5.1	Incentive factor $x_i$ as a function of bias ratio $R_i$ . . . . .	82
5.2	Upload and Download Amount at Each Peer for Simple Model. . . . .	94
5.2	Upload and Download Amount at Each Peer for Simple Model. . . . .	95
5.3	Upload and Download Amount at Each Peer for Adaptive Model. Half of the Free Riders do not share anything at all and rest stoped sharing in the middle of session. . . . .	96
5.4	Upload and Download Amount at Each Peer for Extreme Model. At the beginning of simulation, 10% peers are free-riders. After completion of every 12.5% of total transactions, 10% more peers convert to free-riders. . . . .	97
6.1	Upload and download at each peer at time $t = 0$ . . . . .	111
6.2	Upload and Download Amount at Each Peer for SBCI in Simple Model for Simple Procedure of Peer Selection. . . . .	123
6.2	Upload and Download Amount at Each Peer for SBCI in Simple Model for Simple Procedure of Peer Selection. . . . .	124
6.3	Upload and Download Amount at Each Peer for SBCI in Adaptive Model for Simple Procedure of Peer Selection. . . . .	126
6.3	Upload and Download Amount at Each Peer for SBCI in Adaptive Model for Simple Procedure of Peer Selection. . . . .	127
6.4	Upload and Download Amount at Each Peer for SBCI in Extream Model for Simple Procedure of Peer Selection. At the beginning of simulation, 10% peers are free-riders. After completion of every 12.5% of total transactions, 10% more peers convert themselves to free-riders. . . . .	128

- 
- 6.5 Upload and Download Amount at Each Peer for best case of GC, i.e.,  $\alpha = 0.8, \beta = 0.2$  in different distribution models for Simple Procedure of Peer Selection. . . . . 129
- 6.6 Upload and Download Amount at Each Peer for Different values of  $\alpha$ . Peer selection approach is based on the problem of "College Admission and The Stability of Marriage". Bandwidth distribution is as type 1. . . 131
- 6.6 Upload and Download Amount at Each Peer for Different values of  $\alpha$ . Peer selection approach is based on the problem of "College Admission and The Stability of Marriage". Bandwidth distribution is as type 1. . . 132
- 6.7 Upload and Download Amount at Each Peer for Different values of  $\alpha$ . Peer selection approach is based on the problem of "College Admission and The Stability of Marriage". Bandwidth distribution is as type 2. . . 133
- 6.7 Upload and Download Amount at Each Peer for Different values of  $\alpha$ . Peer selection approach is based on the problem of "College Admission and The Stability of Marriage". Bandwidth distribution is as type 2. . . 134

# List of Tables

2.1	Comparison of Reputation Systems . . . . .	21
2.2	Comparison of Incentive Mechanisms . . . . .	22
3.1	Local trust of peers A, B, C, D and E, zero means there is no interaction between peers till now . . . . .	28
3.2	Convergence Speed and Message Overhead per Peer in Different Repu- tation Systems . . . . .	49
3.3	Values of various parameters which we used in our simulation . . . . .	52
4.1	Center point in each iteration, when initial guess is close to center point .	77
4.2	Center point in each iteration, when initial guess is very far from center point . . . . .	77
4.3	Global trust in each iteration, when initial guess is close to global trust ( $\alpha = 1/3$ ) . . . . .	77
4.4	Global trust in each iteration, when initial guess is very far from global trust ( $\alpha = 1/3$ ) . . . . .	78
4.5	Global trust in each iteration for lesser value of $\alpha$ ( $\alpha = 1/6$ ) . . . . .	78
5.1	BCI for $\alpha = 0.8$ in each iteration . . . . .	87

5.2	BCI for $\alpha = 0.4$ in each iteration . . . . .	87
5.3	Number of iterations, required to converge the BCI and GC [8] for different values of $\alpha$ and $\beta$ . . . . .	88
5.4	AAD for Different Mechanisms and Models . . . . .	99
6.1	AAD and % of Rejections for SBCI in Simple Model for Simple Procedure of Peer Selection . . . . .	122
6.2	AAD and % of Rejections for SBCI in Adaptive Model for Simple Procedure of Peer Selection . . . . .	125
6.3	AAD and % of Rejections for SBCI in Simple Model for Simple Procedure of Peer Selection . . . . .	127
6.4	AAD and % of Rejections for Different Distribution Model in Best case of GC for Simple Procedure of Peer Selection . . . . .	130
6.5	AAD and % of Rejections for SBCI in Simple Model for Stable marriage approach with two different bandwidth peers . . . . .	130
6.6	AAD and % of Rejections for SBCI in Simple Model for Stable marriage approach with ten different bandwidth peers . . . . .	132

# Chapter 1

## Introduction

### 1.1 Introduction

Sharing and exchanging the knowledge and information has always played a vital role, in the process of understanding the nature, thus, in the development of human civilization. Its scope has been expanded with the development of technology. Now we have reached at a place where every person is connected with every other person via the Internet and the whole world has become a global village. The global infrastructure of the Internet, which we see today is the result of efforts and hard work done by many researchers [12], [13], [14].

Leonard Kleinrock of MIT recognized the problem with circuit switching for bursty data communication and introduced the concept of packet switching. He published the first paper [15], on packet switching theory in July 1961 and the first book [16], on the subject in 1964.

The Advanced Research Projects Agency Network (ARPANET) was an early packet switching network funded by the Advanced Research Projects Agency (ARPA) of the



United States, Department of Defense. In October 1969, the first host-to-host message was sent from Kleinrock's Network Measurement Center at University of California, Los Angeles (UCLA) to Stanford Research Institute (SRI).

The early popular applications of the Internet were based on the client-server model in which one port works as a client and the other as a server. Server works as a supplier of resources and the client works as the consumer of resources. In contrast to this, some decentralized structure like Usenet evolved on the Internet which gave birth to peer-to-peer (P2P) model. In this model, ports can work as a client as well as a server.

In its initial days, P2P networks were only used by cooperative researchers to share the information, and by some companies to run the simulation on many computers worldwide to utilize the distributed processing power.

In May 1999, Shawn Fanning of Northeastern University in Massachusetts introduced a popular P2P network named Napster. It was used mainly for file sharing applications. After launching of Napster, P2P networks gained significant popularity as a means to share music files over the Internet. But soon, music companies started a campaign to ban the Napster due to violation of copyright. After a long legal fight, Napster was forced to shut down in 2001.

Napster was a first generation P2P network, which used a centralized server for indexing. In the same era, few more P2P networks, e.g., Kazaa, Gnutella, Gnutella2, Audiogalaxy and iMesh, were emerging. After closure of Napster, Kazaa and Gnutella become most popular P2P networks. These are second generation P2P networks operated in fully decentralized manner.

Now a days, BitTorrent is the most popular file sharing application responsible for

large P2P traffic. As of February 2013, BitTorrent was responsible for 3.35 % of all worldwide bandwidth, more than half of the 6% of total bandwidth dedicated to file sharing [17]. Upto June 2016, BitTorrent accounts for around 5% of the total daily traffic in the North America and Latin America [18].

## 1.2 Definition of P2P Networks

A network, in which each peer can act like client as well as a server, is defined as peer-to-peer or P2P networks in short. In such networks, each peer has equal responsibility and authority. Any peer can initiate the query for communication and in response to this any interested peer can respond. The searching methods for different P2P system could be different, but one thing is common among them that the file or resource is transferred directly between peers.

## 1.3 Classification of P2P Networks

According to dependency on central node, P2P networks can be classified as follows:

### 1.3.1 Centralized P2P Networks

In centralized P2P networks, there is one central node which is used to manage the database of peers in the network and the resource available with them. All peers periodically log into this central server and share the information about the resources which they want to share in the network. This centralized server maps the name of files (or resources) with the current IP address of the peer and resource identifier.

Whenever any peer needs the file, it sends the search request for it to the central server. A central server sends the IP address of corresponding peer and the resource identifiers. Finally, the file can be transferred directly between the peers. An example of such networks are Napster, seti@home, folding@home.

### 1.3.2 Decentralized P2P Networks

In decentralized P2P networks, there is no central node for managing the database of resource index, thus it is also called pure P2P network. In these networks, the nodes are connected with each other either in a random fashion (unstructured P2P), or through a DHT based overlay (structured P2P) networks. Whenever any node wants to connect to the network, it contacts to the bootstrapping nodes. Bootstrapping nodes are always maintained online. Normally, peers exchange their neighbors tables, and each peer updated its neighbor leading to optimization of overlay topology. Bootstrapping node gives the joining peer the IP addresses of one or more existing peers in the network and joining peer updates its neighbor table. Each peer keeps the information about its neighbors for forwarding the queries about any resource.

### 1.3.3 Hybrid P2P Networks

In these types of networks, both centralized and decentralized configurations are used. There are some nodes which act like a centralized indexing server for some set of nodes. These centralized nodes are also called super nodes.

Whenever any node needs the resource (e.g., a file), it sends the request to its super node. Super node flood the query to other super nodes and finally the peer who possesses the file can be located.

Further, based on the overlay network, which is built at the application layer on the top of physical network topology, P2P network can be classified as the unstructured and structured network.

### 1.3.4 Unstructured P2P Networks

In these networks, no specific algorithm is used for searching the query. If any peer needs the file, it sends the query to its neighbors, all neighbors send the query to their neighbors and so on. When the query is resolved, the location of resource holding peer is found, and the file can be downloaded directly from the source node.

### 1.3.5 Structured P2P Networks

These networks use an algorithm for content search. The content can be located in bounded time by routing the query in Distributed Hash Table (DHT) structure. These DHT algorithms are scalable and robust to peer churn, but needed overlay is costly to maintain. In literature, many DHT algorithms have been investigated, e.g., Chord [19], CAN [20], Pastry [21], Tapestry [22].

## 1.4 Distributed Hash Table (DHT)

In this section, we discuss two DHT protocols, Chord [19] and CAN [20].

### 1.4.1 Chord Protocol

The Chord [19] is a distributed protocol, which provides the support of just one operation: given a key, it maps the key to a node. Each node is placed in a virtual ring called Chord ring and given an identifier in clockwise from 0 to  $2^m - 1$ . Each key is identified by m-bit identifier (thus identifier ranges from 0 to  $2^m - 1$ ). The key identifier can be obtained by an hashing algorithm on keyword. SHA-1 [23] is one such popular hash function. Nodes randomly pick an m-bit identifier as node ID such that no two nodes will have same ID. The key identifier space and node identifier space have the same cardinality as both are m-bit identifier.

Each node in the chord ring is responsible for storing some key-value pairs. The key  $k$  is assigned to the node having smallest identifier greater or equal to hash value of key  $k$ . This node is called as root node of key  $k$ .

For efficient lookup, each node maintains a routing table also called a finger table with maximum entries upto  $\log_2(N)$ . For any node  $n$ , the  $i^{th}$  entry of the finger table contains the identifier of the successor (node having smallest identifier greater than or equal to) of node ID  $(n + 2^{i-1})$ , where  $1 \leq i \leq \log_2(N)$ . Whenever any node search for the root node of key, first it checks the nodes in its finger table for key and if it is not found then it passes the query to the node from its finger table which has the highest identifier value less than the hash of key.

Let us understand this process through an example as shown in Fig.1.1. Let the identifier be  $m = 4$  bit long. Thus, there can be maximum 16 nodes and 16 hash value of keys. Let key  $i$  be stored at node  $j$ . Now if node 0 wants the location of key  $k$ , first it will find 4-bit hash value of key  $k$ , let this hash value of key  $k$  be 14. The node checks

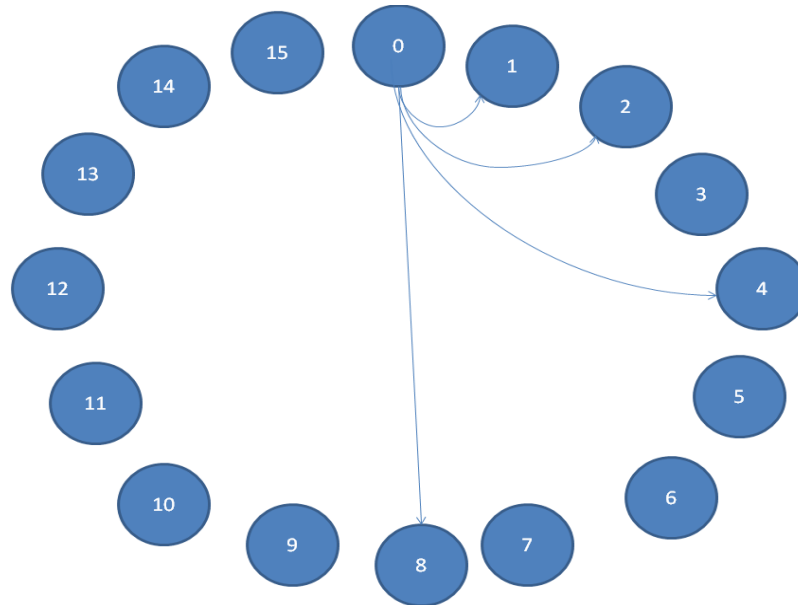
the nodes who are in its finger table. In the finger table of node 0 there are node 1, 2, 4, and node 8 but none of them have identifier 14, then the node 0 will select the node with highest identifier which is less than the key identifier and pass on the query to it. In this case it is node 8. Now node 8 contains node 9, 10, 12 and node 0 in its finger table. Again, none of them have identifier 14, so node 8 will select the node 12. Finally, node 12 will check its finger table for identifier 14 and it will be located at node 14. It can be easily observed that maximum number of queries needed to locate a key will be 4 in this case and in general  $\log_2 N$ .

### 1.4.2 CAN Protocol

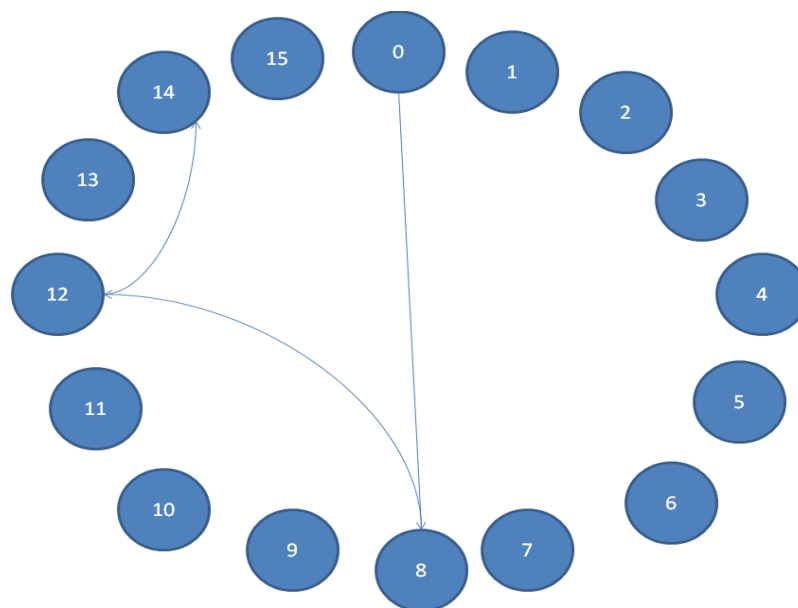
In CAN protocol [20],  $d$ -dimensional coordinate space is dynamically partitioned among all the nodes in the system such that every node is placed in an individual distinct zone within the overall space. This coordinate space is completely a logical space and has no relation with any physical coordinate system.

A pair  $(key, value)$  is stored in this virtual space by uniformly hashing the key to a point P in the space. This pair is stored on the node who owns this point P in coordinate space. Each node in CAN maintains the IP address and virtual coordinate of its each neighbor in the coordinate space. Two nodes in  $d$ -dimension coordinate space are neighbors if their coordinate spans overlap in  $d - 1$  dimension and abut along one dimension. The nodes route the query in CAN using a straight path algorithm. If point P does not lie in the node's zone it forward the query to its neighbor whose coordinate space is closest to the destination.

To understand the routing process in CAN, consider the Fig.1.2. Let there be ten nodes and they own the coordinate zone as shown in Fig.1.2. If node 1 wants to locate



(a) Nodes in finger table of node 0



(b) Lockup mechanism for key 14

Fig. 1.1: The Chord Protocol for  $m = 4$  bit long identifier

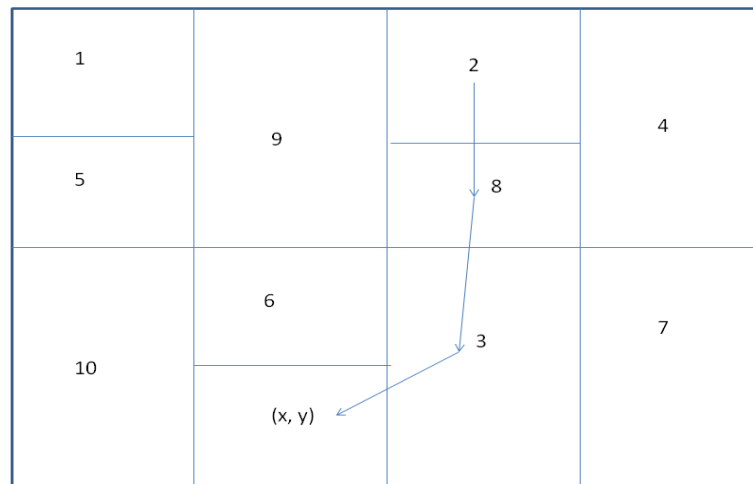


Fig. 1.2: The CAN protocol for 2-d space

the point  $(x, y)$ , it can contact its nearest neighbor to point  $(x, y)$ . In this figure node 2 has three neighbors node 4, node 8 and node 9. So node 2 can choose either node 8 or node 9. Let it selected node 8 then nearest neighbor of node 8 which is closest to point  $(x, y)$  is node 3. So it will forward the query to node 3 and finally node 3 will locate the point  $(x, y)$ .

## 1.5 Advantages of P2P Networks

P2P networks have many advantages over client-server networks which make it popular.

- 1. Diversity of available Data:** In P2P networks, there are diverse peers and each having some data to share. This data is made available to other peers.
- 2. Robustness against single point failure :** Since each peer can act like client as well as server, thus failure of some nodes will have much less affect on the functionality



of the others. If replication of data to be shared is implemented then data loss due to node failure can be avoided.

**3. Scalability:** If any new peer becomes the part of P2P network it enhances the resources in the network, thus total capacity of the network also increases as the consumption increases.

**4. Small initial setup cost:** Since central authority is not needed, thus the network can function with only few initial peers. As the new peers join the system resources also increases. In contrast to this client-server system requires high cost to establish a costly central server to begin the resource sharing network.

**5. Low maintenance cost:** Central server requires high maintenance cost. In P2P networks, there is no central server, thus no costly maintenance is required.

**6. Anonymity of peer:** The peers can interact with each other anonymously if P2P network is configured that way. Though it is not preferred by most of the governments due to security reasons.

## 1.6 Problems in P2P Networks

There are two major issues with P2P networks, the presence of malicious and free-riding peers. The success of P2P networks largely depends on the policies and mechanism by which these two issues are handled.

### 1.6.1 Malicious Peers

There may be some peers who are against the policies of network, e.g., anti piracy agencies do not want music files to be distributed in the network free of cost [24]. Peers with the intention of destroying the network are called the malicious peers.

These malicious users can spread the virus malware/ransomware in the network. Like VBS.Gnutella worm [25] can be introduced by any unknown user in the Gnutella network. It spreads by making a copy of itself in a peers Gnutella program directory, then modifying the Gnutella.ini file to allow sharing of .vbs files.

### 1.6.2 Free-riders

The diversity of data is a primary feature in P2P networks. But its availability depends on if the peers choose to share it. Mostly, peers would like to get more resources while trying to share less [26]. But in such a scenario, no one will get the resources. The peers who share very less or nothing are called free-riders. These free-riders also cause the slow download for others.

## 1.7 Experimental Studies

First study was conducted on Gnutella network by Adar et al. [27] in 2000, leading to following observations:

- 70% users did not share any file.
- 70% of files are shared by 5% of the peers.
- Top 1% peers shared 37% and top 20% shared 98% files.
- Free-riders were distributed uniformly through the network.

Another study was conducted on Gnutella and Napster network by Saroiu et al. [28] in 2002, which shows the following:

- 25% users in Gnutella did not share any file.
- 75% of users in Gnutella shared 100 or less number of files and 7% of peers shares majority of files.
- 40% – 60% users shared 5% – 20% of total files in Napster.

Cuevas et al. [24] conducted another study on popular file sharing P2P network BitTorrent in 2013, which concluded the following:

- 3% of publishers are responsible for 67% of the contents and 75% of download session.
- Antipiracy agencies or malicious users are responsible for 30% of contents and 25% of download session.

## 1.8 Solution of Malicious Peers and Free-riders

The experimental studies show that malicious peers and free-riders are major problems in P2P networks. It attracted the attention of many researchers in the recent past and many solutions have been proposed for these.

### 1.8.1 Reputation System

Managing the reputation through distributed system is one of the methods proposed by many authors. In this method, the past behavior of peers is modeled as trust and it is used by peers to interact with other peers. In most of the existing reputation systems, all the peers evaluate the other peers, based on the past interactions and assign them

some trust value, also called the local trust value. These local trust values are basic information, which are aggregated in the whole network to form the global reputation of the peer. This aggregation process is different for structured and unstructured P2P network. In structured network, the responsibility to manage global reputation through aggregation of local trust is distributed among all the peers. Global reputation is also called global trust. With the help of DHT algorithms, such as Chord [19], CAN [20], Pastry [21], Tapestry [22], the peer managing global trust of a peer can be easily located. The formal definition of local trust and global trust is given in Chapter 2.

In an unstructured network, each peer evaluates the global trust value of peers by collecting the local trust from different peers through a distributed aggregation algorithm, the aggregation can be done either by gossiping protocol or by taking feedback only from few significant peers.

Reputation management system is useful in identifying the malicious peers. Some researchers suggest that it can also be used to prevent the free-riding.

### 1.8.2 Incentive Mechanism

Giving the incentives for sharing the resource is another method which is more efficient to prevent the free-riding in the P2P networks. In recent years, several incentive methods have been studied by the research community. In most of them, each peer evaluates the cooperation of other peers with him in the past and provides the resources for them in same proportion.

In this thesis, we are proposing a new ranking mechanism for both, the reputation system and the incentive mechanism, individually. Our ranking systems can handle the

malicious peers and the free-riders in a more efficient way. These ranking mechanisms can be implemented in a distributed system.

## 1.9 Organization of Thesis

The whole thesis is organized as follows: We have already discussed the brief history of P2P network and problem definition in Chapter 1. In Chapter 2, we will be discussing few of the earlier proposed reputation systems and incentive mechanism in brief and the other related work in same area.

We will introduce our basic trust model named Absolute trust for reputation system in Chapter 3. The trust model will be derived and its convergence will also be proved in the same chapter. The model will also be evaluated through simulation in this chapter. In Chapter 4, we will present a generalized analysis of Absolute trust.

In Chapter 5, we will discuss a distributed incentive method named Biased Contribution Index (BCI) to maintain the fairness and to avoid the free-riding. In Chapter 6, simplified form of the BCI will be presented. In the simplified form, we will also propose the peer selection method using "stable marriage problem approach".

Finally, the thesis will be concluded in Chapter 7. Possible future work will also be discussed in the same chapter.

# Chapter 2

## Background and Related Work

### 2.1 Introduction

As mentioned in chapter 1, malicious peers and free-riders are two major problems in P2P networks. Maintaining the reputation system is one of the methods to handle the malicious peers and implementing an incentive mechanism can avoid free-riding. In this chapter, as a background, we will highlight the related work in this area. We will also explain some of the popular reputation systems and incentive mechanisms proposed in past.

### 2.2 Basic Definitions

Before introducing the related work, let us have some definitions which will be used throughout the thesis.

**Definition 2.2.1.** A numerical value, which models the past behavior of peers in the network, is defined as trust.

**Definition 2.2.2.** The value of trust, evaluated by any peer based on the direct inter-

action it has had with the evaluated peer, is defined as the local trust.

**Definition 2.2.3.** The value of trust, system as a whole keeps on any peer, is defined as a global trust of that peer.

## 2.3 Related Work

In this section we will explore the earlier studies of the reputation system as well as incentive mechanisms to avoid free-riding.

### 2.3.1 Related Work in Reputation System

Reputation system is used to establish the trust among the buyers in e-commerce, e.g., Amazon, Flipkart, Snapdeal, eBay [29]. In all such systems, there is a central authority to keep the record of past experiences of buyers. The experience is used by new buyers for their shopping. Aggregating the feedback in the presence of central authority is simple task, but in a P2P system, which is distributed system, maintaining and aggregating the trust is not trivial.

Aberer and Despotovic [30] proposed a trust model in which only complaints are reported if any, otherwise peers are assumed to be trustworthy. Eigentrust Algorithm [1] is based on Pagerank algorithm [4]. Pre-trusted peers are required to handle the malicious peers in it. In PeerTrust [31] five different factors are defined for evaluation of trustworthiness of the peers. Both Eigentrust [1] and PeerTrust [31] are based on the concept of weighted average. Fuzzy Trust model [32] was proposed by Song *et al.* It is also based on the concept of weighted average, where weight factor is determined by three variables– the peer’s reputation, the transaction date and the transaction

amount. The message complexity in Fuzzy Trust [32] is lesser than in the Eigentrust [1]. PowerTrust [2] is based on assumption of the power law network. In it, local trust is aggregated similarly to the Eigentrust [1] except pre-trusted peers are replaced by most reputable peers in the network. All above trust models [1], [2], [30], [31], [32] are designed for structured network and DHT is used for efficient location of trust holder peers.

In unstructured network, global trust is calculated by floating the query for local trust in the network. The peer, who needs the global trust, waits for the feedback upto some time. Then the calculation of global trust is performed with these limited number of feedback given by some of the peers. Gossip Trust [33] used same metric as in [1] and local trust values are gossiped in the network similarly to randomized gossip algorithm in [34]. In Scalable Feedback Aggregation (SFA) [35], the trustworthiness is calculated by weighted average of local trust and feedback taken by few of the peers. Antonino *et al.* proposed a flow-based reputation [3] which is modified version of [1]. It is only for centralized systems. Wang and Vassileva proposed a Bayesian Trust Model [36] in which, different aspect of peer behavior are modeled in different situations. Damiani *et al.* proposed a system [37] for managing and sharing the server's reputation in which peers poll other peers by broadcasting a request for opinion. In another similar approach [38] Damiani *et al.* considered the reputation of both peers and resources, but credibility of voter was not considered in both the approaches. More details of other related works on reputation systems can be found in [39], [40], [41], [42], [43].



### 2.3.2 Related Work in Incentive Mechanism

Presence of free-riding peers and its impact on fairness in P2P network have been studied earlier also [26], [28]. Several approaches have been proposed by the research community [5], [6], [7], [8], [9], [10], [44], [45], [46],[47], [48], [49], [50].

BitTorrent [51], a most popular file sharing system, used tit-for-tat (TFT) approach to prevent the free-riding. In this approach, a peer cooperates with other peers in the same proportion as they have cooperated with him in the previous round. In each round, every peer updates the contributions of peers in the previous round. To improve the performance, many variants of TFT have been proposed. Garbacki *et al.*, [5], proposed ATFT in which bandwidth is used rather than content to decide the incentives. Dave *et al.*, [52], proposed auction based model to improve the TFT. In this model, peers reward one another with *proportional shares*, [53], of bandwidth. Sherman *et al.*, [9], proposed FairTorrent. It is a deficit based distributed algorithm in which a peer uploads the next data block to the peer, whom it owes the most data as measured by a deficit counter. In Give-to-get [10], peer ranks all its neighbors, based on the amount of data that have been received from them in the last round and then unchokes the top three forwarders. All these mechanisms consider the local and very short history of peers' cooperation.

Global history of peers' cooperation is considered in [6], [7], [8]. In multilevel tit-for-tat (ML-TFT) [7], a peer ranks other peers based on the fraction of download, what he received from them. Its time complexity is much larger for n-step ranking of peers. Feldman *et al.*, [6], proposed a robust incentive technique, which considers the peers' cooperation in the entire network, but it is not trivial to implement in a large network. Its calculation have complexity of  $O(N^3)$ . In Global Contribution (GC) approach [8], a peers' GC point is defined such that all peers are motivated to download from low

contributing peers and upload to high contributing peers. GC point is calculated using iterative methods such as the Jacobi and Gauss-Seidel.

Many authors proposed approaches based on game theory [47], [48], [49], [50] to reduce free-riding. This approach is based on the assumption that the rules of the game are known to all the players. For practically large networks, this may not be true for all the peers.

### 2.3.3 EigenTrust Algorithm

The EigenTrust algorithm was proposed by Sepandar *et al.*, [1] of Stanford university in 2003. The basic idea of EigenTrust is taken from Google's PageRank [4] algorithm.

In this algorithm, each peer  $i$  keeps the record of all the transactions it had with peer  $j$ . Then it calculates the local trust of peer  $j$  as:

$$T_{ij} = sat(i, j) - unsat(i, j), \quad (2.1)$$

where  $sat(i, j)$  and  $unsat(i, j)$  are the number of satisfactory and unsatisfactory transactions respectively, which peer  $i$  has had with peer  $j$ . For the purpose of aggregation, normalized local trust was used instead of local trust, which is defined as:

$$T_{ij}^{norm} = \frac{\max(T_{ij}, 0)}{\sum_{j=1}^N \max(T_{ij}, 0)}, \quad (2.2)$$

here,  $N$  is the number of peers in the network. This normalization process makes the trust matrix,  $\mathbf{T}^{norm}$ , as a row stochastic matrix. Global trust vector,  $\mathbf{t}$ , is calculated as a left principal eigenvector of transpose of normalized trust matrix.

$$\mathbf{t} = (\mathbf{T}^{norm})^{tr} \cdot \mathbf{t} \quad (2.3)$$

It can also be interpreted as the weighted average of normalized local trust of peers where weight factor is given by the global trust of local trust assigning peer. If there is no interaction of peers they give zero local trust to each other. Some pre-trusted peers are assumed to be in the network and they are trusted by all the peers in the network. To include the impact of pre-trusted peers global trust is modified as

$$\mathbf{t} = (1 - a)(\mathbf{T}^{\text{norm}})^{\text{tr}}.\mathbf{t} + a\mathbf{p}, \quad (2.4)$$

here  $\mathbf{p}$  is some distribution over pre-trusted peers and  $a$  is some scalar.

By normalizing the trust matrix, global trust can be calculated by iterative method and it converge at left principal eigenvector of transpose of normalize trust matrix. The major limitation of EigenTrust [1] algorithm is that it gives only ranking of peers without any absolute interpretation of their past history.

### 2.3.4 Power Trust

Power trust was proposed by Zhou *et al.*, [2] in 2007. The basic idea of aggregation of local trust in this algorithm is same as in EigenTrust [1], i.e., using normalized trust matrix to compute weighted average. It used some highly reputed nodes also known as power nodes in place of pre-trusted nodes. These power nodes are searched and elected dynamically in the whole network. Thus, pre-trusted nodes are free to leave the network unlike in EigenTrust [1]. This method used the enhanced trust matrix for the purpose of aggregation of local trust. Enhance trust matrix is the square of trust matrix. It is performed using look-ahead random walk in which, each node in the trust overlay network not only holds its own local trust scores, but also aggregates its neighbors first hand ones.

TABLE 2.1: Comparison of Reputation Systems

S.N.	Reputation System	Normalization	Distributed Implementation
1	EigenTrust	Needed	Possible
2	PowerTrust	Needed	Possible
3	Flow Based	Not Needed	Not Possible
4	Absolute Trust	Not Needed	Possible

Speed of convergence of PowerTrust [2] is higher than EigenTrust [1]. In both, EigenTrust [1] and PowerTrust [2], distributed hash table is used to locate the peer who is calculating and managing the global trust of a peer.

### 2.3.5 Flow-Based Reputation

Flow-Based Reputation was proposed by Simone *et al.*, [3] in 2012. They identified the problems with normalization and hence used basic trust matrix without normalization for the aggregation of local trust. For this purpose they redefined the local trust of a peer by mapping a function from positive (+1), neutral (0) and negative (-1) rating to a value into the range [0, 1].

Global trust vector was taken as the left principal eigenvector of basic trust matrix. It is calculated using power method [54]. Main drawback of this method is that it cannot be implemented in a distributed system.

### 2.3.6 Global Contribution Approach to Maintain Fairness

The Global Contribution Approach for Fairness was proposed by Nishida *et al.*, [8] in 2010. This approach was used to prevent the free-riding in the network and to balance

TABLE 2.2: Comparison of Incentive Mechanisms

S.N.	Incentive Mechanism	Approach	Convergence Speed
1	Tit-for-tat	Local	-
2	ATFT	Local	-
3	FairTorrent	Local	-
4	GC	Global	Slow
5	BCI	Global	Fast
6	SBCI	Global	-

the upload and download amount in each peer.

Global contribution,  $x_i$ , of any peer  $i$  is calculated as

$$x_i = \alpha \frac{\beta \mathbf{e}_i \cdot \mathbf{S} \cdot \mathbf{x} + (1 - \beta) \mathbf{e}_i \cdot \mathbf{S} \cdot \mathbf{e} - \mathbf{e}_i \cdot \mathbf{S}^{\text{tr}} \cdot \mathbf{x}}{\mathbf{e}_i \cdot (\mathbf{S} + \mathbf{S}^{\text{tr}}) \cdot \mathbf{e}} + (1 - \alpha), \quad (2.5)$$

here,  $\mathbf{x}$  is global contribution vector.  $\mathbf{S}$  is share matrix in the network with its  $ij$  element as the amount shared by peer  $i$  to peer  $j$ . The  $\mathbf{e}_i$  is a row vector with its  $i^{\text{th}}$  entry as '1' and  $\mathbf{e}$  is a column vector with each entry as '1'. The parameters  $\alpha$  and  $\beta$  are some scalars decide the initial value of global contribution vector and guaranteed the convergence of it. If no transaction has happened at node  $i$ , then

$$x_i = \frac{2 - \alpha(1 + \beta)}{2 + \alpha(1 - \beta)}.$$

A peer is allowed to take the resources from the network only if its global contribution is higher than a threshold value. Thus, every peer would like to earn more global contribution. It can earn more global contribution if it uploads to high contributing peers and downloads from low contributing peers.

This method can balance the upload and download amount in the network and thus can prevent the free-riding. But this algorithm is very complex to implement in real

---

systems due to its slow speed of convergence. We have proposed a new algorithm, viz., biased contribution index (BCI) and also its simplified form. These algorithms can be implemented in much simpler way in a distributed system.

## Chapter 3

# Absolute Trust: Algorithm for Aggregation of Trust in Peer-to-Peer Networks

### 3.1 Introduction

To mitigate the attacks by malicious peers in P2P networks, several reputation systems have been proposed in the past. In most of them, the peers evaluate other peers based on their past interactions and then aggregate this information in the whole network. However, such an aggregation process requires approximations, in order to converge at some global consensus. It may not be the true reflection of past behavior of the peers. Moreover, such type of aggregation gives only the relative ranking of peers without any absolute evaluation of their past. This is more significant when all the peers responding to a query, are malicious. In such a situation, we can only know that who is better among them without knowing their rank in the whole network.

In this chapter, we are proposing a new algorithm which accounts for the past behavior of the peers and will estimate the absolute value of the trust of peers. Consequently,

we can suitably identify them as good peers or malicious peers. By choosing suitable parameters, our algorithm converges much faster at some global consensus. Due to its absolute nature, it will equally load all the peers in the network. It will also reduce the inauthentic downloads in the network which was not possible in existing algorithms.

## 3.2 Motivation

Due to open and anonymous nature of P2P networks, malicious users can easily sabotage the network by putting inauthentic contents [24]. In such environment, peers don't feel comfortable to establish the communication with unknown users, until they feel them to be trustworthy. Therefore, implementation of reputation system becomes a natural choice. In recent years, many studies have been done on reputation systems to model the past behavior of the peers [1], [2], [3], [30], [31], [32], [33], [35], [36], [37], [38], [55], [56], [57], [58], [59]. But, its implementation in real life system is yet to be done.

Pagerank [4] is the most successful example of reputation system used by google search engine, where reputation of page is decided by its popularity on the web. The popularity is measured by two factors, the number of links that point to the page and from where the links originate. Based on the similar concept, reputation systems have been proposed for peer ranking where most trustworthy peer is considered as the most reputed one [1], [2], [3], [33]. Positive aspect of this is that the consensus is taken from the entire network. But there are four major fundamental problems in it.

First, ranking itself is not adequate to decide the trustworthiness of a peer. For example, when a peer sends a request for a particular file and all the responding peers



are malicious, then ranking system can only tell us who is better among them. We will never know whether they are malicious peers or good peers.

Second, there is a difference between popularity and trustworthiness; a peer can be popular in the network by doing transaction with large number of peers, but may not be providing good quality of service. Whereas a peer is considered to be trustworthy only if it provides good quality of service in each transaction. This can be understood by following example. Let there be five peers in a network - A, B, C, D and E. After some interactions they give some local trust value to each other, as shown in Table 3.1. Local trust value is defined as trust value given by peers to each other based on their direct interactions. More detail on it is given in Section 3.3. After aggregating these local trust values, as per method given in [1], [2], they are ranked as B, E, C, D, A; B is most trustworthy and A is least trustworthy. If it is aggregated following the method in [3], then they are ranked as E, B, D, C, A; E is most trustworthy and A is least trustworthy. But we can see clearly from Table 3.1 that A is making two transactions but both transactions are good as compared to any transaction made by B, C and D. So we cannot conclude that A is less trustworthy as compared to B, C and D.

Third, in these ranking systems, the most reputable peers are always overloaded, even if we use the probabilistic approach to select the source peer for download, i.e., the probability of choosing a download source peer is proportional to its global trust.

And lastly, this type of ranking is performed by normalization of the local trust. Updating the values of normalized local trust to other peers is message consuming task. Because, if trust assigning peer updates the local trust of any one of its interacting peer, then it needs to update the values of all the other interacting peers.

In all other reputation systems, feedback of peers are taken from few significant peers

to estimate the global trust. This does not make the global trust global in true sense. Keeping in view all the above points, a reputation system in P2P network must have the following design objectives:

- Reputation should be true reflection of past behavior.
- Reputation must be aggregated across the whole network.
- The system should be robust to the presence of malicious peers with as many attackers model as possible.
- Load Balance: System should not overload only few peers in network.
- It should be adaptive to peer dynamics.
- Fast Convergence Speed.
- Lower overhead/message complexity.
- Central authority should not be needed.

In this chapter, we are proposing a metric and an aggregation algorithm which truly capture the past behavior of the peers. The proposed algorithm can give the ranking of the peers and can characterize them absolutely as well. Aggregation is done without normalization hence, it automatically meets the above design objectives. It is purely decentralized and does not require any kind of central authority or pre-trusted peers or power nodes. The Absolute Trust is based on the concept of weighted averaging and scaling of local trust. It is calculated recursively in the whole network, till it converges. We will show that it will converge at some unique global value and can be calculated distributively in the whole network by all the peers. Our simulation results show that

TABLE 3.1: Local trust of peers A, B, C, D and E, zero means there is no interaction between peers till now

	A	B	C	D	E
A	0	0.6	0.6	0	0
B	0.3	0	0.3	0.4	0.4
C	0.4	0.4	0	0.2	0.2
D	0.5	0.1	0.1	0	0.5
E	0.7	0.7	0.8	0	0

it gives better authentic download performance and more uniform load distribution among good peers with lesser message complexity.

Rest of the chapter is organized as follows: In Section 3.3, we define the basic trust model and its aggregating algorithm. Section 3.4 discusses the existence and uniqueness of proposed global trust. In Section 3.5, the algorithm is analyzed. Section 3.6 presents the simulation results, and finally in Section 3.7, conclusion and future work is presented.

### 3.3 Proposed Trust Model

In this model of P2P network, the peers are assumed to exchange only files as a resource. With suitable modification, the same model can also be used for other kind of resources. First, we will define the basic trust metric, namely local trust, which is a raw data used for the calculation of global trust. Later, we will give an algorithm for aggregation of the local trusts to generate the global trust. Global trust is the trust, system as a whole keeps on an individual peer.

### 3.3.1 Local Trust

Typically peer's satisfaction after a transaction can be classified as satisfied, average or neutral and unsatisfied. We can also define many other levels, but for simplicity only three levels have been assumed. Let peer  $i$  downloads some files from peer  $j$ , then peer  $i$  can assign a local trust value to peer  $j$  as

$$T_{ij} = \frac{n_g w_g + n_n w_n + n_b w_b}{n_t},$$

where  $n_g, n_n$  and  $n_b$  denote the number of satisfactory, average or neutral and unsatisfactory files respectively. The  $w_g, w_n$  and  $w_b$  denote the weight factor for satisfactory, average or neutral and unsatisfactory files respectively and  $n_t$  denotes the total number of downloaded files.

Without loss of generality, let us consider the weight factor for average or neutral file to be in the middle of weight factors of unsatisfactory file and satisfactory file. Thus,

$$w_n = \frac{w_g + w_b}{2}.$$

On simplification,

$$\begin{aligned} T_{ij} &= \left[ \frac{n_g w_g + (n_t - n_g - n_b) \frac{(w_g + w_b)}{2} + n_b w_b}{n_t} \right] \\ T_{ij} &= \frac{1}{2n_t} [(n_g - n_b + n_t)w_g + (n_b - n_g + n_t)w_b] \\ T_{ij} &= \frac{1}{2} [(frac\_sat - frac\_unsat + 1)w_g + (frac\_unsat - frac\_sat + 1)w_b], \quad (3.1) \end{aligned}$$

where  $frac\_sat, frac\_unsat$  denote the fraction of satisfactory and unsatisfactory files respectively.

This metric will ensure the value of local trust to be within  $w_g$  and  $w_b$ . For example, if peer  $i$  downloads 100 files from peer A and B, out of which, A provides 20 satisfactory

files, 40 unsatisfactory files and rest average files while peer B provides 30 satisfactory files, 60 unsatisfactory files and rest average files. Let the weight factor of good file is 10 and that is for bad file is 1, then  $T_{iA} = 4.6$  and  $T_{iB} = 4.15$ .

Many authors argue that there are many other factors, which can influence the local trust value, e.g., amount of transactions, date of transactions, number of transactions [31], [32], [35], [59]. We agree with their arguments and those can also be considered in our case, but in all the cases, the aggregation process will remain same. In next subsection, we explain the process of aggregation of trust in the network.

### 3.3.2 Absolute Trust: Algorithm for Aggregation

In any evaluation process, there are two parties, one who is evaluating; we will call it the evaluator, and the one who is being evaluated; we will call it the evaluatee. Reliability of evaluation depends on the person who is evaluating it, and varies from person to person. Evaluation is said to be more reliable if it is done by a competent evaluator.

There are three different scenarios in the evaluation as shown in Fig. 3.1. One-to-many: one person is evaluating many persons; many-to-one: many persons are evaluating one person; and one-to-one: different persons are evaluating different persons. In one-to-many scenario, since evaluation is done by only one person, the basis of evaluation can be considered to be uniform. In many-to-one evaluation, since one person is evaluated by many persons so there are chances of contradictions. At the same time, the opinion of any evaluator cannot be ignored. Thus, the best way to resolve the contradiction is to take the weighted average of all the evaluators' opinions, while assigning more weight to a more competent evaluator.

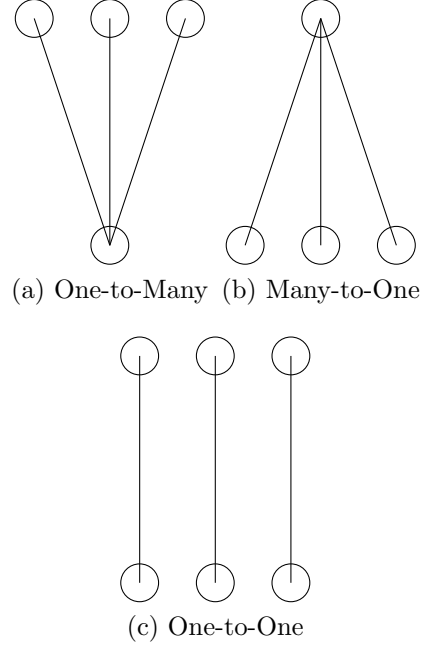


Fig. 3.1: Different ways of evaluation

In one-to-one evaluation, there is no direct comparison of two evaluations because the evaluator and the evaluatee both are different. In order to compare these evaluations, it is essential to make their basis uniform with respect to the evaluator. Again based on the concept that competent evaluator's evaluation will be more accurate, we can bias these evaluations by a weight factor, which must be proportional to the competence of evaluator in some sense. The bias can be given by

$$Eval\_uniform\_out = [(Eval\_value\_in)^p (w_e)^q]^{\frac{1}{p+q}}, \quad (3.2)$$

where,  $Eval\_value\_in$  is evaluation done by an individual evaluator,  $w_e$  is weight factor assigned to this evaluator,  $Eval\_uniform\_out$  is output evaluation based on uniform basis and  $p, q$  are suitably chosen constants.

If  $p = q$ , then  $Eval\_uniform\_out$  is geometric mean of  $w_e$  and  $Eval\_value\_in$ . If we

take  $q = \alpha p$  then

$$Eval\_uniform\_out = [(Eval\_value\_in)(w_e)^\alpha]^{\frac{1}{1+\alpha}}. \quad (3.3)$$

Now let there be  $N$  peers in the network interacting with each other. If any peer  $i$  takes the service of any peer  $j$ , then  $i$  can evaluate trust on  $j$  according to (3.1). Each  $i$  can evaluate all such peer  $j$  independently and there is no need of any modification in the evaluation, because it is one-to-many evaluation. We are aggregating the values of these one-to-many evaluations (local trust) resulting in the estimate of global trust.

Each peer is also providing services to many other peers, and is being evaluated by them. This is many-to-one evaluation. The aggregated trust values after this step will be weighted average of all the local trust estimates. The weight factor can be chosen in many different ways, but global trust of an individual peer will be the best choice to be used as a weight. Many authors argue that a good service provider may not be a good feedback provider [31], [35], [59]. But we argue that until peers are not in the competition, a good service provider will be most likely a good feedback provider. So we have taken global trust of peers as the weight factor for the purpose of aggregation of local trust. Hence global trust,  $t_i$ , of any peer  $i$  is given by

$$t_i = \frac{\sum_{j \in S_i} T_{ji} t_j}{\sum_{j \in S_i} t_j} \quad \forall i, \quad (3.4)$$

here  $S_i$  is a set of peers getting services from peer  $i$ ,  $T_{ji}$  is local trust of peer  $i$  evaluated by peer  $j$ ,  $t_j$  is global trust of peer  $j$ . Equation (3.4) can be rearranged as

$$\begin{aligned} t_i &= \frac{\sum_{j \in S_i} T_{ji} t_j}{\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{t}} \\ &= \sum_{j \in S_i} (\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{t})^{-1} T_{ji} t_j \end{aligned}$$

These set of  $N$  equations can be written in the form of a matrix as

$$\mathbf{t} = [\text{diag}(\mathbf{e}_1 \cdot \mathbf{C} \cdot \mathbf{t}, \mathbf{e}_2 \cdot \mathbf{C} \cdot \mathbf{t}, \dots, \mathbf{e}_N \cdot \mathbf{C} \cdot \mathbf{t})]^{-1} \cdot \mathbf{T}^{\text{tr}} \cdot \mathbf{t},$$

where  $\mathbf{t}$  is global reputation column vector, it is also defined as center point of matrix  $\mathbf{T}^{\text{tr}}$ . For properties and other details of center point of a matrix, see [60].  $\mathbf{T}$  is trust matrix, its  $T_{ij}$  element is local trust value of peer  $j$  assigned by peer  $i$ . The element  $T_{ij}$  is zero if there is no interaction among peer  $i$  and peer  $j$ ,  $\mathbf{e}_i$  is row vector with  $i^{\text{th}}$  entry as 1 and all others are zero,  $\mathbf{C}$  is incidence matrix corresponding to  $\mathbf{T}^{\text{tr}}$ , i.e., if  $T_{ji} > 0$ , then  $C_{ij} = 1$ , else  $C_{ij} = 0$ .  $\mathbf{T}^{\text{tr}}$  is transpose of matrix  $\mathbf{T}$ .

It is clear from (3.4) that value of  $t_i$  will remain between minimum and maximum value of local trust given by peers belonging to  $S_i$ .

Now, in the whole network every peer is evaluated by a different set. If the set can be represented equivalently as a single peer, then it is same as one-to-one evaluation. The basis of this evaluation can be made uniform using (3.2). To give the equivalent global trust of the set, consider a set  $S_i$  of  $m$  peers with global trust values  $t_1, t_2, \dots, t_m$ . The global trust of the set must be dominated by the more trustworthy peers, because we are giving more weight to their opinion. With the notion of weighted average, intuitively, we can define the global trust of the set,  $S_i$ , as

$$t_{s_i} = \frac{\sum_{j \in S_i} t_j^2}{\sum_{j \in S_i} t_j}. \quad (3.5)$$

This equation is similar to (3.4). Here, we are ensuring that global trust of a set will be dominated by the peers having higher global trust value. It will always be in between the minimum and maximum values of global trust of the members of set  $S_i$ . The global trust,  $t_i$ , of a peer  $i$ , can be biased by the global trust,  $t_{s_i}$ , of trust assigning set  $S_i$ , according to (3.2). The modified global trust of peer  $i$  can be written as

$$t_i = [t_i^p t_{s_i}^q]^{\frac{1}{(p+q)}},$$



$$t_i = \left[ \left( \frac{\sum_{j \in S_i} T_{ji} t_j}{\sum_{j \in S_i} t_j} \right)^p \left( \frac{\sum_{j \in S_i} t_j^2}{\sum_{j \in S_i} t_j} \right)^q \right]^{\frac{1}{(p+q)}}. \quad (3.6)$$

Equation (3.6) will be the true reflection of past behavior of peer  $i$  in the whole system. This equation will give us the absolute interpretation of global trust value of any peer. The basis of evaluation has been made uniform by using a biasing factor  $t_{s_i}$ . We can now directly compare the global trust values of any two peers. Equation (3.6) can be rearranged as

$$\begin{aligned} t_i &= \left[ \left( \frac{\sum_{j \in S_i} T_{ji} t_j}{\sum_{j \in S_i} t_j} \right) \left( \frac{\sum_{j \in S_i} t_j^2}{\sum_{j \in S_i} t_j} \right)^{q/p} \right]^{\frac{1}{(1+q/p)}} \\ &= \left[ \left( \frac{(\sum_{j \in S_i} t_j^2)^{q/p}}{(\sum_{j \in S_i} t_j)^{(1+q/p)}} \right) \left( \sum_{j \in S_i} T_{ji} t_j \right) \right]^{\frac{1}{(1+q/p)}} \\ &= \left[ \left( \frac{(\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{diag}(\mathbf{t}) \cdot \mathbf{t})^\alpha}{(\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{t})^{(1+\alpha)}} \right) \left( \sum_{j \in S_i} T_{ji} t_j \right) \right]^{\frac{1}{(1+\alpha)}} \\ &= \left[ \sum_{j \in S_i} \left( \frac{(\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{diag}(\mathbf{t}) \cdot \mathbf{t})^\alpha}{(\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{t})^{(1+\alpha)}} \right) \left( T_{ji} \right) \left( t_j \right) \right]^{\frac{1}{(1+\alpha)}} \end{aligned}$$

There are  $N$  nodes in the network,  $i = 1, 2, \dots, N$ . Thus, set of  $N$  equations can be written in the form of matrix as follows:

$$\mathbf{t} = (\mathbf{D} \cdot \mathbf{T}^{\text{tr}} \cdot \mathbf{t})^{\frac{1}{1+\alpha}}.$$

$\mathbf{D}$  is a diagonal matrix, with its  $ii^{\text{th}}$  element  $d_i$  as  $\left[ \frac{(\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{diag}(\mathbf{t}) \cdot \mathbf{t})^\alpha}{(\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{t})^{(1+\alpha)}} \right]$ ,  $\mathbf{diag}(\mathbf{t})$  is  $N \times N$  diagonal matrix, with its  $ii^{\text{th}}$  element as  $t_i$  and  $\alpha = q/p$ . Rest all variables have the same meaning as mentioned earlier. Power of any vector is defined as the power of its each element. All vectors are defined as a column vector until it is specified.

In the network of  $N$  nodes, there are  $N$  unknowns and  $N$  non-linear equations, hence we cannot state anything directly about the solution of these equations. In next section,

we will show that there exists a unique positive global trust vector, corresponding to these set of equations. The solution can be found by an iterative method.

### 3.4 Existence and Uniqueness of Global trust

We are proposing following Lemmas and Theorems to show the existence and uniqueness of global trust vector. The definitions to be used in this section are given below:

**Definition 3.4.1.** A vector  $\mathbf{v}$  or matrix  $\mathbf{M}$  is said to be positive/non-negative if its each element  $v_i$  or  $M_{ij}$  is positive/non-negative and real.

**Definition 3.4.2.** A vector  $\mathbf{v}'$  / matrix  $\mathbf{M}'$  is said to be less than  $\mathbf{v}''/\mathbf{M}''$  if its each element  $v'_i/M'_{ij}$  is less than  $v''_i/M''_{ij}$ .

**Lemma 3.4.1.** Let  $\mathbf{z}$  be a positive vector in  $\mathbb{R}^N$ , such that  $\mathbf{z} = \mathbf{f}(\mathbf{t})$ , with its  $i^{th}$  element as  $\left[ \frac{t_i(\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{t})}{(\mathbf{e}_i \cdot \mathbf{C} \cdot \text{diag}(\mathbf{t}) \cdot \mathbf{t})} \right]^\alpha t_i$ . Then  $\exists$  at least one pair of positive vectors  $\mathbf{t}'$  and  $\mathbf{t}''$  such that, if  $\mathbf{t}'' > \mathbf{t}'$ , then

$$\mathbf{f}(\mathbf{t}'') > \mathbf{f}(\mathbf{t}'),$$

where  $\alpha$  is an arbitrary rational number.

**Proof.** Let us consider two vectors  $\mathbf{t}' = a\mathbf{e}$  and  $\mathbf{t}'' = b\mathbf{e}$ . Here  $\mathbf{e}$  is a vector with all elements as '1',  $a$  and  $b$  are scalar such that  $b > a > 0$ . Then  $i^{th}$  element of vector  $\mathbf{f}(\mathbf{t}')$

$$f_i(\mathbf{t}') = \left[ \frac{t'_i(\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{t}')}{(\mathbf{e}_i \cdot \mathbf{C} \cdot \text{diag}(\mathbf{t}') \cdot \mathbf{t}')} \right]^\alpha t'_i$$

$$f_i(\mathbf{t}') = \left[ \frac{a(ma)}{(ma^2)} \right]^\alpha a$$

$$f_i(\mathbf{t}') = a$$

here  $m$  is number of '1' in  $i^{th}$  row of incidence matrix  $\mathbf{C}$ .

Similarly

$$f_i(\mathbf{t}'') = b$$

Hence,  $\exists$  a pair of positive vectors  $\mathbf{t}'$  and  $\mathbf{t}''$  satisfying the condition.  $\square$

**Lemma 3.4.2.** *Let  $\mathbf{A}$  and  $\mathbf{B}$  be  $N \times N$  non-negative, irreducible matrices with spectral radius '1', and corresponding eigenvector  $\mathbf{v}$ . Then for any vector  $\mathbf{x}$ ; having at least one component along vector  $\mathbf{v}$ .*

$$\lim_{k \rightarrow \infty} (\mathbf{M}_1 \cdot \mathbf{M}_2 \cdot \mathbf{M}_3 \dots \mathbf{M}_k) \cdot \mathbf{x} = c\mathbf{v}$$

Here  $\mathbf{M}_i$  can be  $\mathbf{A}$  or  $\mathbf{B}$  for all  $i$  from 1 to  $k$  and  $c$  is any scalar.  $\mathbf{A}$  and  $\mathbf{B}$  are such that  $(\mathbf{M}_1 \cdot \mathbf{M}_2 \cdot \mathbf{M}_3 \dots \mathbf{M}_k)$  is also irreducible.

**Proof.** Let the eigenvectors of matrices  $\mathbf{A}$  and  $\mathbf{B}$  be  $\mathbf{v}, \mathbf{v}_2, \mathbf{v}_3, \dots, \mathbf{v}_N$  and  $\mathbf{v}, \mathbf{u}_2, \mathbf{u}_3, \dots, \mathbf{u}_N$ . Then any vector  $\mathbf{x}$ ; having at least one component along vector  $\mathbf{v}$ , can be expressed as

$$\mathbf{x} = a_1\mathbf{v} + a_2\mathbf{v}_2 + \dots + a_N\mathbf{v}_N$$

and

$$\mathbf{x} = b_1\mathbf{v} + b_2\mathbf{u}_2 + \dots + b_N\mathbf{u}_N$$

when this vector will pass through matrices  $\mathbf{A}$  and  $\mathbf{B}$  then it will be

$$\mathbf{A} \cdot \mathbf{x} = a_1\mathbf{v} + a_2\lambda_2\mathbf{v}_2 + \dots + a_N\lambda_N\mathbf{v}_N$$

and

$$\mathbf{B}.\mathbf{x} = b_1\mathbf{v} + b_2\gamma_2\mathbf{u}_2 + \dots + b_N\gamma_N\mathbf{u}_N$$

where  $\lambda_2, \lambda_3, \dots, \lambda_N$  and  $\gamma_2, \gamma_3, \dots, \gamma_N$  are eigenvalues of matrices  $\mathbf{A}$  and  $\mathbf{B}$  respectively. If it will pass through any of  $\mathbf{A}$  and  $\mathbf{B}$  again, then vector  $\mathbf{v}$  will remain as it is and magnitude of all other vectors will decrease because  $1 > |\lambda_2| > |\lambda_3| > \dots > |\lambda_N|$  and  $1 > |\gamma_2| > |\gamma_3| > \dots > |\gamma_N|$  (see [61]). Thus

$$\mathbf{B}.\mathbf{A}.\mathbf{x} = a_1\mathbf{v} + \delta a\mathbf{v} + L.O.M.O.\mathbf{u}_2, \mathbf{u}_3, \dots, \mathbf{u}_N$$

and

$$\mathbf{A}.\mathbf{B}.\mathbf{x} = b_1\mathbf{v} + \delta b\mathbf{v} + L.O.M.O.\mathbf{v}_2, \mathbf{v}_3, \dots, \mathbf{v}_N$$

$L.O.M.O.$  means "lower order magnitude of". Repeating this operation  $k^{th}$  times in any order we will get

$$\lim_{k \rightarrow \infty} (\mathbf{M}_1.\mathbf{M}_2.\mathbf{M}_3.\dots.\mathbf{M}_k).\mathbf{x} = c\mathbf{v}$$

where  $\mathbf{M}_i$  can be  $\mathbf{A}$  or  $\mathbf{B}$  for all  $i$  from 1 to  $k$  □

**Theorem 3.4.1.** *Let  $\mathbf{A}$  and  $\mathbf{B}$  be  $N \times N$  non-negative, irreducible matrices with spectral radius '1', and corresponding eigenvector  $\mathbf{v}$ . Then for any vector  $\mathbf{x}$  in  $\mathbb{R}^N$*

$$\lim_{k \rightarrow \infty} (\mathbf{A} - \mathbf{B})^k.\mathbf{x} = \mathbf{0}$$

**Proof.** In Lemma 3.4.2, let  $\mathbf{M}_i = \mathbf{A}$  for all  $i$ , then

$$\lim_{k \rightarrow \infty} \mathbf{A}^{k-1}.\mathbf{x} \approx a_1\mathbf{v}, \tag{3.7}$$

and if  $\mathbf{M}_i = \mathbf{B}$  for all  $i$ , then

$$\lim_{k \rightarrow \infty} \mathbf{B}^{k-1}.\mathbf{x} \approx a_2\mathbf{v} \tag{3.8}$$

if  $\mathbf{M}_1$  is taken arbitrary  $\mathbf{A}$  or  $\mathbf{B}$ , then

$$\lim_{k \rightarrow \infty} (\mathbf{A}.\mathbf{B}.....\mathbf{B}.\mathbf{A}....(k-1)times).\mathbf{x} \approx a_3 \mathbf{v} \quad (3.9)$$

where  $a_1, a_2$  and  $a_3$  are some scalars, adding 3.7, 3.8 with all combinations of 3.9 will result

$$\lim_{k \rightarrow \infty} (\mathbf{A} - \mathbf{B})^{k-1}.\mathbf{x} \approx b \mathbf{v} \quad (3.10)$$

here  $b$  is a linear combination of  $a_1, a_2$  and all  $a_3$ . Now pre-multiplying (3.10) by  $(\mathbf{A} - \mathbf{B})$ ,

$$(\mathbf{A} - \mathbf{B}).(\mathbf{A} - \mathbf{B})^{k-1}.\mathbf{x} = (\mathbf{A} - \mathbf{B}).b \mathbf{v} = (\mathbf{v} - \mathbf{v})b = \mathbf{0}$$

Hence

$$\lim_{k \rightarrow \infty} (\mathbf{A} - \mathbf{B})^k.\mathbf{x} = \mathbf{0}$$

□

**Theorem 3.4.2.** *Let  $\mathbf{A}, \mathbf{A}_1, \mathbf{A}_2, \dots, \mathbf{A}_m$  be  $N \times N$  non-negative, irreducible matrices, with spectral radius 1,  $\lambda_1, \lambda_2, \dots, \lambda_m$  respectively. If the corresponding eigenvector for all the above matrices be  $\mathbf{v}$ . Then for any vector  $\mathbf{x}$ .*

$$\lim_{k \rightarrow \infty} (\mathbf{A}_1 + \mathbf{A}_2 + \dots + \mathbf{A}_m - \mathbf{A})^k.\mathbf{x} = \mathbf{0}$$

if  $|\lambda_1 + \lambda_2 + \dots + \lambda_m - 1| < 1$

**Proof.** Let

$$\mathbf{M} = (\mathbf{A}_1 + \mathbf{A}_2 + \dots + \mathbf{A}_m)$$

then

$$\mathbf{M}.\mathbf{v} = (\mathbf{A}_1 + \mathbf{A}_2 + \dots + \mathbf{A}_m).\mathbf{v}$$

$$= (\lambda_1 + \lambda_2 + \dots + \lambda_m)\mathbf{v} = \lambda\mathbf{v}$$

hence  $\mathbf{v}$  is also an eigenvector of  $\mathbf{M}$  and corresponding eigenvalue is  $\lambda$ . Matrix  $\mathbf{M}$  is the sum of non-negative, irreducible matrices  $\mathbf{A}_1, \mathbf{A}_2, \dots, \mathbf{A}_m$  therefore  $\mathbf{M}$  is also non-negative and irreducible. So we can conclude that spectral radius of matrix  $\mathbf{M}$  is  $\lambda$ .

Further,  $\mathbf{M}$  can be written as

$$\begin{aligned}\mathbf{M} &= \left[ \frac{\mathbf{M}}{\lambda} + \frac{(\lambda - 1)\mathbf{M}}{\lambda} \right] \\ &= [\mathbf{B} + \mathbf{B}']\end{aligned}$$

here  $\mathbf{B}$  is  $\mathbf{M}/\lambda$  and  $\mathbf{B}'$  is  $(\lambda - 1)\mathbf{M}/\lambda$ . Matrices  $\mathbf{B}$  and  $\mathbf{B}'$  are scalar multiple of non-negative irreducible matrix  $\mathbf{M}$  therefore matrices  $\mathbf{B}$  and  $\mathbf{B}'$  also follow the properties of non-negative irreducible matrices. Hence spectral radius of matrices  $\mathbf{B}$  and  $\mathbf{B}'$  is '1' and  $|\lambda - 1|$  respectively and corresponding eigenvector is  $\mathbf{v}$ .

If  $|\lambda - 1| < 1$  then

$$\lim_{k \rightarrow \infty} \mathbf{B}'^k \cdot \mathbf{x} = \mathbf{0}, \quad (3.11)$$

and from Theorem 3.4.1

$$\lim_{k \rightarrow \infty} (\mathbf{B} - \mathbf{A})^k \cdot \mathbf{x} = \mathbf{0}. \quad (3.12)$$

In fact, when vector  $\mathbf{x}$  is passed through any of  $\mathbf{B}'$  or  $(\mathbf{B} - \mathbf{A})$ , its magnitude decreases, and at  $k \rightarrow \infty$ , it become zero. So in general we can write

$$\lim_{k \rightarrow \infty} (\mathbf{M}_1 \cdot \mathbf{M}_2 \cdot \mathbf{M}_3 \dots \mathbf{M}_k) \cdot \mathbf{x} = \mathbf{0} \quad (3.13)$$

where  $\mathbf{M}_i$  can be any of  $\mathbf{B}'$  or  $(\mathbf{B} - \mathbf{A})$ . Adding all the combinations of (3.13) with (3.11) and (3.12), we will get

$$\lim_{k \rightarrow \infty} (\mathbf{B}' + (\mathbf{B} - \mathbf{A}))^k \cdot \mathbf{x} = \mathbf{0}$$

or

$$\lim_{k \rightarrow \infty} (\mathbf{M} - \mathbf{A})^k \cdot \mathbf{x} = \mathbf{0}$$

hence

$$\lim_{k \rightarrow \infty} (\mathbf{A}_1 + \mathbf{A}_2 + \dots + \mathbf{A}_m - \mathbf{A})^k \cdot \mathbf{x} = \mathbf{0}$$

if  $|\lambda - 1| < 1$  or  $|\lambda_1 + \lambda_2 + \dots + \lambda_m - 1| < 1$ . □

**Theorem 3.4.3.** *Let  $\mathbf{T}$  be  $N \times N$  non-negative, irreducible matrix then,  $\exists$  a positive vector  $\mathbf{t}$  such that*

$$(\mathbf{t})^{1+\alpha} = (\mathbf{D} \cdot \mathbf{T}^{\text{tr}} \cdot \mathbf{t})$$

$\mathbf{D}$  is diagonal matrix, with its  $i^{\text{th}}$  element  $d_i$  as  $[(\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{diag}(\mathbf{t}) \cdot \mathbf{t})^\alpha / (\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{t})^{(1+\alpha)}]$ .

**Proof.** Relation  $(\mathbf{t})^{1+\alpha} = (\mathbf{D} \cdot \mathbf{T}^{\text{tr}} \cdot \mathbf{t})$  can be written as

$$\mathbf{T}^{\text{tr}} \cdot \mathbf{t} = \mathbf{D}^{-1} \cdot (\mathbf{t})^{1+\alpha} = \mathbf{y}$$

where  $y_i = \left[ \frac{(\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{t})^{(1+\alpha)}}{(\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{diag}(\mathbf{t}) \cdot \mathbf{t})^\alpha} \right] t_i^{1+\alpha}$ . Further,  $y_i$  can be written as

$$\begin{aligned} y_i &= (\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{t}) \left[ \frac{(\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{t})^\alpha}{(\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{diag}(\mathbf{t}) \cdot \mathbf{t})^\alpha} \right] t_i^{1+\alpha} \\ &= (\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{t}) \left[ \frac{t_i (\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{t})}{\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{diag}(\mathbf{t}) \cdot \mathbf{t}} \right]^\alpha t_i \\ &= (\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{t}) \cdot f_i(\mathbf{t}) \\ &= \mathbf{e}_i \cdot (f_i(\mathbf{t}) \mathbf{C}) \cdot \mathbf{t} \end{aligned}$$

hence vector  $\mathbf{y}$  can be written as

$$\mathbf{y} = \mathbf{F}(\mathbf{t}) \cdot \mathbf{t}$$

where matrix  $\mathbf{F}(\mathbf{t})$  has nonzero elements at same positions as matrix  $\mathbf{C}$  and therefore, at same position as  $\mathbf{T}^{\text{tr}}$ , its  $ij$  element  $F_{ij}(\mathbf{t})$  is  $f_i(\mathbf{t})$ , hence

$$\mathbf{T}^{\text{tr}}.\mathbf{t} = \mathbf{F}(\mathbf{t}).\mathbf{t}$$

Now,  $\exists$  a positive vector  $\mathbf{t}'$ , such that  $f_i(\mathbf{t}') \leq \min(T_{ij} > 0)$ . For such  $\mathbf{t}'$ ,

$$\mathbf{T}^{\text{tr}}.\mathbf{t}' > \mathbf{F}(\mathbf{t}').\mathbf{t}'.$$

Also  $\exists$  a positive vector  $\mathbf{t}''$ , such that  $f_i(\mathbf{t}'') \geq \max(T_{ij})$ , For such  $\mathbf{t}''$ ,

$$\mathbf{T}^{\text{tr}}.\mathbf{t}'' < \mathbf{F}(\mathbf{t}'').\mathbf{t}''$$

function  $f$  is continuous and from Lemma 3.4.1, there exist a path from  $\mathbf{f}(\mathbf{t}')$  to  $\mathbf{f}(\mathbf{t}'')$  such that if  $\mathbf{t}'' > \mathbf{t}'$ , then  $\mathbf{f}(\mathbf{t}'') > \mathbf{f}(\mathbf{t}')$ . Hence,  $\exists$  a positive vector  $\mathbf{t}$  between  $\mathbf{t}'$  and  $\mathbf{t}''$ , such that

$$\mathbf{T}^{\text{tr}}.\mathbf{t} = \mathbf{F}(\mathbf{t}).\mathbf{t}$$

hence  $\exists$  a positive vector  $\mathbf{t}$ , such that

$$(\mathbf{t})^{1+\alpha} = (\mathbf{D}.\mathbf{T}^{\text{tr}}.\mathbf{t})$$

□

**Theorem 3.4.4.** *Vector  $\mathbf{t}$  in theorem 3.4.3 is unique and can be calculated by an iterative function*

$$\mathbf{t}^k = \phi(\mathbf{t}^{k-1}) = [\mathbf{D}(\mathbf{t}^{k-1}).\mathbf{T}^{\text{tr}}.\mathbf{t}^{k-1}]^{\frac{1}{1+\alpha}},$$

where  $\mathbf{t}^k$  is the value of vector  $\mathbf{t}$  in  $k^{\text{th}}$  iteration and  $\phi$  is the iterative function from  $\mathbb{R}^N \rightarrow \mathbb{R}^N$ .



**Proof.** The iterative function  $\phi(\mathbf{t}^{k-1})$  is

$$\begin{aligned} \mathbf{t}^k &= \phi(\mathbf{t}^{k-1}) \\ &= [\mathbf{D}(\mathbf{t}^{k-1}).\mathbf{T}^{\text{tr}}.\mathbf{t}^{k-1}]^{\frac{1}{1+\alpha}} \\ &= [\text{diag}(\mathbf{d}_1, \mathbf{d}_2, \dots, \mathbf{d}_N).\mathbf{T}^{\text{tr}}.\mathbf{t}^{k-1}]^{\frac{1}{1+\alpha}} \end{aligned} \quad (3.14)$$

where

$$d_i = \left( \frac{(\mathbf{e}_i.\mathbf{C}.\text{diag}(\mathbf{t}^{k-1}).\mathbf{t}^{k-1})^\alpha}{(\mathbf{e}_i.\mathbf{C}.\mathbf{t}^{k-1})^{(1+\alpha)}} \right)$$

The  $i^{\text{th}}$  element of  $\mathbf{t}^k$  will be

$$\begin{aligned} t_i^k &= \left[ \frac{(\mathbf{e}_i.\mathbf{T}^{\text{tr}}.\mathbf{t}^{k-1})(\mathbf{e}_i.\mathbf{C}.\text{diag}(\mathbf{t}^{k-1}).\mathbf{t}^{k-1})^\alpha}{(\mathbf{e}_i.\mathbf{C}.\mathbf{t}^{k-1})^{1+\alpha}} \right]^{\frac{1}{1+\alpha}} \\ &= \left[ \frac{(\mathbf{e}_i.\mathbf{T}^{\text{tr}}.\mathbf{t}^{k-1})^{\frac{1}{1+\alpha}} (\mathbf{e}_i.\mathbf{C}.\text{diag}(\mathbf{t}^{k-1}).\mathbf{t}^{k-1})^{\frac{\alpha}{1+\alpha}}}{(\mathbf{e}_i.\mathbf{C}.\mathbf{t}^{k-1})} \right] \end{aligned} \quad (3.15)$$

Let  $t_i^k$  and  $t_i^{k-1}$  are far from actual solution  $t_i$  by  $\delta t_i^k$  and  $\delta t_i^{k-1}$  respectively, then

$$\begin{aligned} t_i + \delta t_i^k &= \left[ \frac{(\mathbf{e}_i.\mathbf{T}^{\text{tr}}.(\mathbf{t} + \delta \mathbf{t}^{k-1}))^{\frac{1}{1+\alpha}}}{(\mathbf{e}_i.\mathbf{C}.(\mathbf{t} + \delta \mathbf{t}^{k-1}))} \right] [(\mathbf{e}_i.\mathbf{C}.\text{diag}(\mathbf{t} + \delta \mathbf{t}^{k-1}).(\mathbf{t} + \delta \mathbf{t}^{k-1}))^{\frac{\alpha}{1+\alpha}}] \\ &= \left[ \frac{(\mathbf{e}_i.\mathbf{T}^{\text{tr}}.\mathbf{t})^{\frac{1}{1+\alpha}} (\mathbf{e}_i.\mathbf{C}.\text{diag}(\mathbf{t}).\mathbf{t})^{\frac{\alpha}{1+\alpha}}}{(\mathbf{e}_i.\mathbf{C}.\mathbf{t})} \right] \left[ \frac{(1 + \frac{\mathbf{e}_i.\mathbf{T}^{\text{tr}}.\delta \mathbf{t}^{k-1}}{\mathbf{e}_i.\mathbf{T}^{\text{tr}}.\mathbf{t}})^{\frac{1}{1+\alpha}}}{(1 + \frac{\mathbf{e}_i.\mathbf{C}.\delta \mathbf{t}^{k-1}}{\mathbf{e}_i.\mathbf{C}.\mathbf{t}})} \right] \\ &\quad \left[ \left( 1 + \frac{2\mathbf{e}_i.\mathbf{C}.\text{diag}(\mathbf{t}).\delta \mathbf{t}^{k-1}}{\mathbf{e}_i.\mathbf{C}.\text{diag}(\mathbf{t}).\mathbf{t}} + \frac{\mathbf{e}_i.\mathbf{C}.\text{diag}(\delta \mathbf{t}^{k-1}).\delta \mathbf{t}^{k-1}}{\mathbf{e}_i.\mathbf{C}.\text{diag}(\mathbf{t}).\mathbf{t}} \right)^{\frac{\alpha}{1+\alpha}} \right] \end{aligned}$$

If  $\delta \mathbf{t}^{k-1} \ll \mathbf{t}$  then we can neglect the higher order terms of  $\delta \mathbf{t}^{k-1}$ .

$$\begin{aligned} &\approx \left[ \frac{(\mathbf{e}_i.\mathbf{T}^{\text{tr}}.\mathbf{t})^{\frac{1}{1+\alpha}} (\mathbf{e}_i.\mathbf{C}.\text{diag}(\mathbf{t}).\mathbf{t})^{\frac{\alpha}{1+\alpha}}}{(\mathbf{e}_i.\mathbf{C}.\mathbf{t})} \right] \left[ \frac{(1 + \frac{\mathbf{e}_i.\mathbf{T}^{\text{tr}}.\delta \mathbf{t}^{k-1}}{\mathbf{e}_i.\mathbf{T}^{\text{tr}}.\mathbf{t}})^{\frac{1}{1+\alpha}}}{(1 + \frac{\mathbf{e}_i.\mathbf{C}.\delta \mathbf{t}^{k-1}}{\mathbf{e}_i.\mathbf{C}.\mathbf{t}})} \right] \\ &\quad \left[ \left( 1 + \frac{2\mathbf{e}_i.\mathbf{C}.\text{diag}(\mathbf{t}).\delta \mathbf{t}^{k-1}}{\mathbf{e}_i.\mathbf{C}.\text{diag}(\mathbf{t}).\mathbf{t}} \right)^{\frac{\alpha}{1+\alpha}} \right] \end{aligned}$$

Using steady state form of (3.15), i.e.,  $\mathbf{t}^k = \mathbf{t}^{k-1} = \mathbf{t}$

$$= t_i \left[ \frac{(1 + \frac{\mathbf{e}_i \cdot \mathbf{T}^{\text{tr}} \cdot \delta \mathbf{t}^{k-1}}{\mathbf{e}_i \cdot \mathbf{T}^{\text{tr}} \cdot \mathbf{t}})^{\frac{1}{1+\alpha}}}{(1 + \frac{\mathbf{e}_i \cdot \mathbf{C} \cdot \delta \mathbf{t}^{k-1}}{\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{t}})} \right] \left[ \left( 1 + \frac{2\mathbf{e}_i \cdot \mathbf{C} \cdot \text{diag}(\mathbf{t}) \cdot \delta \mathbf{t}^{k-1}}{\mathbf{e}_i \cdot \mathbf{C} \cdot \text{diag}(\mathbf{t}) \cdot \mathbf{t}} \right)^{\frac{\alpha}{1+\alpha}} \right]$$

Using binomial expansion and neglecting higher order terms of  $\delta \mathbf{t}^{k-1}$

$$\begin{aligned} &\approx t_i \left[ \frac{(1 + \frac{\mathbf{e}_i \cdot \mathbf{T}^{\text{tr}} \cdot \delta \mathbf{t}^{k-1}}{(1+\alpha)\mathbf{e}_i \cdot \mathbf{T}^{\text{tr}} \cdot \mathbf{t}})}{(1 + \frac{\mathbf{e}_i \cdot \mathbf{C} \cdot \delta \mathbf{t}^{k-1}}{\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{t}})} \right] \left[ \left( 1 + \frac{2\alpha \mathbf{e}_i \cdot \mathbf{C} \cdot \text{diag}(\mathbf{t}) \cdot \delta \mathbf{t}^{k-1}}{(1+\alpha)\mathbf{e}_i \cdot \mathbf{C} \cdot \text{diag}(\mathbf{t}) \cdot \mathbf{t}} \right) \right] \\ \delta t_i^k &= t_i \left[ \frac{(1 + \frac{\mathbf{e}_i \cdot \mathbf{T}^{\text{tr}} \cdot \delta \mathbf{t}^{k-1}}{(1+\alpha)\mathbf{e}_i \cdot \mathbf{T}^{\text{tr}} \cdot \mathbf{t}})}{(1 + \frac{\mathbf{e}_i \cdot \mathbf{C} \cdot \delta \mathbf{t}^{k-1}}{\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{t}})} \left( 1 + \frac{2\alpha \mathbf{e}_i \cdot \mathbf{C} \cdot \text{diag}(\mathbf{t}) \cdot \delta \mathbf{t}^{k-1}}{(1+\alpha)\mathbf{e}_i \cdot \mathbf{C} \cdot \text{diag}(\mathbf{t}) \cdot \mathbf{t}} \right) - 1 \right] \\ &= t_i \left[ \left( 1 + \frac{\mathbf{e}_i \cdot \mathbf{T}^{\text{tr}} \cdot \delta \mathbf{t}^{k-1}}{(1+\alpha)\mathbf{e}_i \cdot \mathbf{T}^{\text{tr}} \cdot \mathbf{t}} \right) \left( 1 + \frac{2\alpha \mathbf{e}_i \cdot \mathbf{C} \cdot \text{diag}(\mathbf{t}) \cdot \delta \mathbf{t}^{k-1}}{(1+\alpha)\mathbf{e}_i \cdot \mathbf{C} \cdot \text{diag}(\mathbf{t}) \cdot \mathbf{t}} \right) \right. \\ &\quad \left. - \left( 1 + \frac{\mathbf{e}_i \cdot \mathbf{C} \cdot \delta \mathbf{t}^{k-1}}{\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{t}} \right) \right] / \left( 1 + \frac{\mathbf{e}_i \cdot \mathbf{C} \cdot \delta \mathbf{t}^{k-1}}{\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{t}} \right) \end{aligned}$$

Approximating the denominator term  $\left( 1 + \frac{\mathbf{e}_i \cdot \mathbf{C} \cdot \delta \mathbf{t}^{k-1}}{\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{t}} \right) \approx 1$ ,

$$\begin{aligned} \delta t_i^k &\approx t_i \left[ \left( \frac{\mathbf{e}_i \cdot \mathbf{T}^{\text{tr}} \cdot \delta \mathbf{t}^{k-1}}{(1+\alpha)\mathbf{e}_i \cdot \mathbf{T}^{\text{tr}} \cdot \mathbf{t}} \right) + \left( \frac{2\alpha \mathbf{e}_i \cdot \mathbf{C} \cdot \text{diag}(\mathbf{t}) \cdot \delta \mathbf{t}^{k-1}}{(1+\alpha)\mathbf{e}_i \cdot \mathbf{C} \cdot \text{diag}(\mathbf{t}) \cdot \mathbf{t}} \right) \right. \\ &\quad \left. + \left( \frac{\mathbf{e}_i \cdot \mathbf{T}^{\text{tr}} \cdot \delta \mathbf{t}^{k-1}}{(1+\alpha)\mathbf{e}_i \cdot \mathbf{T}^{\text{tr}} \cdot \mathbf{t}} \right) \left( \frac{2\alpha \mathbf{e}_i \cdot \mathbf{C} \cdot \text{diag}(\mathbf{t}) \cdot \delta \mathbf{t}^{k-1}}{(1+\alpha)\mathbf{e}_i \cdot \mathbf{C} \cdot \text{diag}(\mathbf{t}) \cdot \mathbf{t}} \right) - \left( \frac{\mathbf{e}_i \cdot \mathbf{C} \cdot \delta \mathbf{t}^{k-1}}{\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{t}} \right) \right] \end{aligned}$$

Again neglecting higher order terms of  $\delta \mathbf{t}^{k-1}$

$$\begin{aligned} \delta t_i^k &\approx t_i \left[ \left( \frac{\mathbf{e}_i \cdot \mathbf{T}^{\text{tr}} \cdot \delta \mathbf{t}^{k-1}}{(1+\alpha)\mathbf{e}_i \cdot \mathbf{T}^{\text{tr}} \cdot \mathbf{t}} \right) + \left( \frac{2\alpha \mathbf{e}_i \cdot \mathbf{C} \cdot \text{diag}(\mathbf{t}) \cdot \delta \mathbf{t}^{k-1}}{(1+\alpha)\mathbf{e}_i \cdot \mathbf{C} \cdot \text{diag}(\mathbf{t}) \cdot \mathbf{t}} \right) - \left( \frac{\mathbf{e}_i \cdot \mathbf{C} \cdot \delta \mathbf{t}^{k-1}}{\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{t}} \right) \right] \\ &= \left[ \left( \frac{t_i \mathbf{e}_i \cdot \mathbf{T}^{\text{tr}}}{(1+\alpha)\mathbf{e}_i \cdot \mathbf{T}^{\text{tr}} \cdot \mathbf{t}} \right) + \left( \frac{2\alpha t_i \mathbf{e}_i \cdot \mathbf{C} \cdot \text{diag}(\mathbf{t})}{(1+\alpha)\mathbf{e}_i \cdot \mathbf{C} \cdot \text{diag}(\mathbf{t}) \cdot \mathbf{t}} \right) - \left( \frac{t_i \mathbf{e}_i \cdot \mathbf{C}}{\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{t}} \right) \right] \cdot \delta \mathbf{t}^{k-1} \\ &= [\mathbf{X}_i + \mathbf{Y}_i - \mathbf{Z}_i] \cdot \delta \mathbf{t}^{k-1}, \end{aligned}$$

where  $\mathbf{X}_i, \mathbf{Y}_i$  and  $\mathbf{Z}_i$  are  $i^{th}$  row of  $N \times N$  matrices  $\mathbf{X}, \mathbf{Y}$  and  $\mathbf{Z}$  respectively, it can be observed easily,  $\mathbf{X} \cdot \mathbf{t} = \frac{1}{1+\alpha} \mathbf{t}$ ,  $\mathbf{Y} \cdot \mathbf{t} = \frac{2\alpha}{1+\alpha} \mathbf{t}$  and  $\mathbf{Z} \cdot \mathbf{t} = \mathbf{t}$ . Here,  $\mathbf{X}, \mathbf{Y}, \mathbf{Z}, \mathbf{t} > \mathbf{0}$ .

Matrices  $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$  have non zero elements at same positions as matrix  $\mathbf{T}^{tr}$ , hence all these are also irreducible. Therefore spectral radius of  $\mathbf{X}, \mathbf{Y}$  and  $\mathbf{Z}$  will be  $\frac{1}{1+\alpha}$ ,  $\frac{2\alpha}{1+\alpha}$  and 1 respectively. Now,

$$\delta \mathbf{t}^k = (\mathbf{X} + \mathbf{Y} - \mathbf{Z}) \cdot \delta \mathbf{t}^{k-1}$$

If initial error in  $\mathbf{t}$  is  $\delta \mathbf{t}^0$  then

$$\lim_{k \rightarrow \infty} \delta \mathbf{t}^k = \lim_{k \rightarrow \infty} (\mathbf{X} + \mathbf{Y} - \mathbf{Z})^k \cdot \delta \mathbf{t}^0$$

Directly from Theorem 3.4.2

$$\lim_{k \rightarrow \infty} (\mathbf{X} + \mathbf{Y} - \mathbf{Z})^k \cdot \delta \mathbf{t}^0 = \mathbf{0}$$

$$\Rightarrow \lim_{k \rightarrow \infty} \delta \mathbf{t}^k = \mathbf{0}$$

if

$$\begin{aligned} & \left| \frac{1}{1+\alpha} + \frac{2\alpha}{1+\alpha} - 1 \right| < 1 \\ \Rightarrow & \frac{\alpha}{1+\alpha} < 1, \end{aligned}$$

which is true for any  $\alpha > 0$ .

If  $\delta \mathbf{t}^{k-1} > \mathbf{t}$  then  $\delta t_i^k$  will reduce very fast and very soon  $\delta \mathbf{t}^{k-1} \ll \mathbf{t}$  (see [60]). Convergence of algorithm depends upon the value of  $\alpha$ . For lower  $\alpha$ , it will converge faster. Same thing is also verified through simulation in next section.  $\square$

In Theorem 3.4.4, if  $\alpha = q/p$  then  $i^{th}$  element of vector  $\mathbf{t}$  will be

$$t_i = \left( \frac{(\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{diag}(\mathbf{t}) \cdot \mathbf{t})^{\frac{q}{p}}}{(\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{t})^{(1+\frac{q}{p})}} (\mathbf{e}_i \cdot \mathbf{T}^{tr} \cdot \mathbf{t}) \right)^{\frac{1}{1+\frac{q}{p}}}$$

$$\begin{aligned}
&= \left( \left( \frac{\mathbf{e}_i \cdot \mathbf{C} \cdot \text{diag}(\mathbf{t}) \cdot \mathbf{t}}{\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{t}} \right)^q \left( \frac{\mathbf{e}_i \cdot \mathbf{T}^{\text{tr}} \cdot \mathbf{t}}{\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{t}} \right)^p \right)^{\frac{1}{p+q}} \\
t_i &= \left[ \left( \frac{\sum_{j \in S_i} T_{ji} t_j}{\sum_{j \in S_i} t_j} \right)^p \left( \frac{\sum_{j \in S_i} t_j^2}{\sum_{j \in S_i} t_j} \right)^q \right]^{\frac{1}{(p+q)}}.
\end{aligned}$$

This is (3.6), hence the global trust vector exists and has a unique positive value. It can be calculated by iterative method.

## 3.5 Analysis of Algorithm

### 3.5.1 Implementation in Distributed System

Algorithm 3.1 describes, how the requesting peers can select the peer from whom to download. We will call the selected peers as source peers. Each peer can set a threshold value of global trust,  $t_{Th}$ , to decide whether to select a peer as a source or not. If global trust of any peer is less than  $t_{Th}$ , then it should not be selected as a source. The requesting peers initiate a query for resources. Each query is given a TTL value. Whenever a query is forwarded, its TTL value is decremented. When TTL becomes zero, the query is not forwarded anymore. The requesting peer can control the scope of query by choosing TTL value. A requesting peer will wait for a time greater than  $2 \times \text{TTL}$ . If no response is received within waiting period, the query can be made again with larger TTL value.

After getting the responses from the network, a peer can select the most reputed peer as a source and can download the required file. In order to balance the load of the network, a peer can select the set of peers whose global trust is more than the  $t_{Th}$  and then the source peer can be selected probabilistically among them. The probability of selecting any peer as a source can be taken to be proportional to its global trust. This

strategy has twofold effects; one is to allow only the reputed peers to become the source with higher probability, and another is to balance the load among them. However, we selected most reputed peer as a source peer for simplicity. In this process, if all of the responding peers have a global trust less than  $t_{Th}$ , then all of them can be rejected and requesting peer can go for another search by increasing the TTL value of query. There should be an upper limit on TTL, after which peer should stop and terminate the query process.

After selecting the source peer and getting the file from it, a peer can evaluate the quality of file and can give feedback for it. Collection of feedback and calculation of global trust can be performed distributively by following the same approach as in [1], [2]. With the help of DHT algorithms, such as Chord [19], CAN [20], Pastry [21] and Tapestry [22], the peer named trust holder peer, managing feedback and global trust of a peer, can be easily located.

Each trust holder peer can update the global trust as described in Algorithm 3.2. To calculate the global trust value of any peer, trust holder peer needs to know the local trust values of that peer and the current global trust value of trust assigning peers. Trust assigning peers will send the local trust values of source peers to their trust holder peers and trust holder peers will ask the current global trust values of trust assigning peers from their respective trust holder peers. This process is repeated till the convergence of global trust (see Algorithm 3.2). Global trust will converge for any initial value of global trust vector,  $\mathbf{t}^0$ . But we have to ensure that, at least one component of initial global trust vector,  $\mathbf{t}^0$ , must be along the final global trust vector,  $\mathbf{t}$ , (see Section 4). To ensure the security, more than one peer can be configured to manage the global trust of a particular peer.

So far, we have discussed, how to aggregate the global trust from local trust. In peer selection procedure, we are considering only the aggregated global trust. However, global trust is more significant if peer has no past history with any of the responding peer. If peer  $j$  has some past history with any one of them then decision of selection of source peer can be done according to  $\beta t_i + (1 - \beta)T_{ji}$ . It is a convex combination of the global trust of peer and the local trust value assigned by requesting peer to the responding peer in the past. The value of parameter  $\beta$  can be selected by the peer depending on its confidence on the responding peer.

### 3.5.2 Speed of Convergence

The speed of convergence is measured in terms of the number of iterations before global trust vector converges. The range and average value of required number of iterations in different reputation systems are shown in Table 3.2. In general, speed of convergence of EigenTrust [1], and PowerTrust [2], depends on second largest eigenvalue of normalized trust matrix [62]. For Absolute Trust, it depends on the largest eigenvalue of matrix,  $(X + Y - Z)$  and  $\alpha$  (see Section 3.4). Smaller the above eigenvalues, lesser number of iterations will be needed for convergence.

The impact of  $\alpha$  on the speed of convergence of Absolute Trust is clearly evident from Table 3.2. We can see, as  $\alpha$  is decreasing, speed of convergence of Absolute Trust is increasing. For  $\alpha \leq 1/3$ , on an average, it is converging in less than seven iterations.

Lower value of  $\alpha$  means higher value of  $p$  compared to  $q$ . Higher  $p$  means more weightage to first term, which is weighted average of local trust values informed by direct interacting peers. Lower  $q$  means lesser weight to second term, which is the equivalent global trust of trust assigning set  $S$ . But there is trade-off between these

two. First term is used to settle the conflicts among direct trust assigning peers, and second term is used to bias the global trust of peer according to global trust of the trust assigning set. The global trust of each member of set is biased by its trust assigning set respectively, and so on. Hence, second factor is taking the opinion from rest of the network. We cannot neglect the opinion of other peers but we also need faster convergence of the algorithm. Higher speed of convergence implies that a lesser number of messages are needed to update the global trust.

### 3.5.3 Message Overhead and Time Complexity

The summary of overall message overhead in different reputation systems are shown in Table 3.2. In general, message overhead of the algorithm depends on three major factors, i.e., sparsity of trust matrix, speed of convergence and need of normalization of trust matrix. Lower sparsity of trust matrix and lower speed of convergence increase the message overhead. The need of normalization of trust matrix also require more messages, because if local trust of any one source peer is updated then it will affect the local trust of other co-source peers, and the peer needs to send the updated feedback of all the source peers.

In PowerTrust [2], enhanced trust matrix, which is square of trust matrix, is used in place of trust matrix. Squaring of trust matrix increases the speed of convergence but decreases the sparsity of matrix very rapidly. Normalization of trust matrix is needed in both EigenTrust [1] and PowerTrust [2]. As a result, overall message overhead is higher in PowerTrust [2] compared to EigenTrust [1].

In EigenTrust [1], trust matrix is more sparse compared to Absolute Trust, because minimum trust value is given '0' in former and '1' in latter. But due to impact of

TABLE 3.2: Convergence Speed and Message Overhead per Peer in Different Reputation Systems

S.N.	Reputation System	Sparsity of Matrix	Required Iterations		Required Messages per Peer
			Range	Average	
1	ET	High	7 - 12	8.6	13618.24
2	PT	Low	4 - 8	5.85	28892.22
3	AT, $\alpha = 1$	High but $< ET$	9 - 12	11	19039.64
4	AT, $\alpha = 1/2$	High but $< ET$	6 - 9	7.3	14078.93
5	AT, $\alpha = 1/3$	High but $< ET$	5 - 8	6.8	12670.80
6	AT, $\alpha = 1/4$	High but $< ET$	5 - 7	6.3	11947.52
7	AT, $\alpha = 1/5$	High but $< ET$	5 - 7	6.25	11664.90

need of normalization and speed of convergence, we can see that a required number of messages per peer in Absolute Trust,  $\alpha \leq 1/3$ , are lesser than in EigenTrust [1].

Time complexity of calculation of global trust on trust holder peer will be  $O(N \times \text{number of iterations})$  for each update of global trust. The simulation is conducted by varying the number of peers in the network,  $N$ , from 100 to 1000. Our simulation results show that the number of iterations required to converge the global trust is independent of the  $N$ . Hence we can say that time complexity of algorithm is  $O(N)$  per peer for one update.



---

**Algorithm 3.1** For Selection of Source Peer

---

```

1: procedure
2:   Set  $t_{Th}$ 
3:   Set  $TTL$ 
4: top:
5:   Set  $Time\_Counter \geq 2 \times TTL$ 
6:    $i \leftarrow 0$ 
7:   Send the query for required file in Network;
8:   while  $i \leq Time\_Counter$  do
9:     Wait for response from the network;
10:     $i \leftarrow i + 1$ ;
11:  end while
12:  if  $Number\_of\_responding\_peers == 0$  then
13:    if  $TTL \geq (TTL)_{upper}$  then
14:      Terminate the query process;
15:    else
16:      Increase  $TTL$ ;
17:      goto top
18:    end if
19:  else
20:    Get the  $Global\_Trust$  of all the responding peers from their trust holder peer;
21:    Select the peer with maximum  $Global\_Trust$ ;
22:    if  $Global\_Trust \geq t_{Th}$  then
23:      Download the required file from the selected peer;
24:      Evaluate the file;
25:      Send the feedback to trust holder peer of source peer;
26:      Stop;
27:    else
28:      if  $TTL \geq (TTL)_{upper}$  then
29:        Terminate the query process;
30:      else
31:        Increase  $TTL$ ;
32:        goto top
33:      end if
34:    end if
35:  end if
36: end procedure

```

---

---

**Algorithm 3.2** For Updating the Global Trust of Peers
 

---

```

1: Input: Local Trust values of peers
2: Output: Global Trust with trust holder peers
3: procedure
4:   for each peer  $i$  do
5:     forall peer  $j$ , who is selected as source peer do
6:       Evaluate the received file;
7:       Assign the Local Trust value between  $w_b$  to  $w_g$  to peer  $j$ ;
8:       Send the Local Trust to trust holder peer of peer  $j$ ;
9:     end forall
10:    if Peer  $i$  is trust holder peer of peer  $k$  then
11:      forall peer  $j$ , who selected peer  $k$  as a source peer do
12:        Receive the Local Trust values  $T_{jk}$ ;
13:        Locate  $j^{th}$  peer's trust holder peer;
14:      end forall
15:      \\ Initialization of parameters
16:      Set  $p, q, previous\_t_k, threshold, error$ ;
17:      while  $error \geq threshold$  do
18:        Receive the current Global Trust  $t_j$  from their trust holder peer ;
19:        Compute
20:          
$$t_k \leftarrow \left[ \left( \frac{\sum_{j \in S_k} T_{jk} t_j}{\sum_{j \in S_k} t_j} \right)^p \left( \frac{\sum_{j \in S_k} t_j^2}{\sum_{j \in S_k} t_j} \right)^q \right]^{\frac{1}{(p+q)}}$$

21:         $error \leftarrow |t_k - previous\_t_k|$ 
22:         $previous\_t_k \leftarrow t_k$ 
23:      end while
24:    end if
25:  end for
26: end procedure

```

---

TABLE 3.3: Values of various parameters which we used in our simulation

S.N.	Parameter	Description	Value(s)
1	$N$	Number of Peers in the Network	$10^2 - 10^3$
2	$Num\_file$	Number of different files in the Network	$10^3 - 10^4$
3	$Num\_trans$	Total number of transactions in the network	$10^4 - 10^5$
4	$\gamma$	Zipf's Constant	0.4
5	$\alpha = q/p$	See (3.6)	0 - 2
6	$w\_b$	Weight factor for unsatisfactory file	1
7	$w\_g$	Weight factor for satisfactory file	10
8	$t_{Th}$	Threshold value of Global Trust for selection of source peers	1 - 5.5

## 3.6 Experimental Evaluation

Like in [35] and [63], we used NetLogo 5.2 [64], to evaluate the performance of our algorithm. NetLogo is a multi-agent programmable modeling environment, where we can model different agents and can ask them to perform the task in parallel and independently. It is written mostly in Scala, with some parts in Java. We also simulated and compared our results with two popular reputation systems, EigenTrust[1] and PowerTrust [2]. We found that our algorithm is giving better performance in various behavioral conditions of peers in the network. It is explained in the following subsections.

### 3.6.1 Simulation Setup

We simulated a typical P2P network with parameters and distributions taken from real world measurements [24], [28]. We used percentage of authentic downloads, per-

centage of rejected transactions among the good peers and load distribution as standard metric to evaluate and compare the performance of reputation systems. In this model, a peer can issue a query for a particular file. The query propagates in the network and peers can respond to it, if they have that particular file. Peers can ask for only those files which they don't have. Peers can select a source peer according to its global trust and can reject all the possible source peers if none of them are found suitable.

The values of parameters used in simulation are shown in Table 3.3. We have taken 100 to 1000 nodes in the network. The number of nodes can be increased upto any value but the results are expected to remain same, because all the metrics are taken in percentage to facilitate the comparison of results. Files are distributed among the nodes as per Zipf's Law [65], with Zipf constant as 0.4. We have also considered the transient phase and applied the global trust update after every 200 query cycles.

We assumed that 95% of the time, peer behaves as per their defined behavior and rest of the time just opposite to it. The assumption is that a peer can make mistake and sometimes may not behave the way it should. With a peer, it can happen 5% of the time. Simulation is performed for various behavioral conditions in these peers. Each experiment is conducted ten times and then readings are averaged over these multiple trials to remove statistical noise. We have considered four kind of behavior in peers as given below.

#### 3.6.1.1 Good Peers

Good peers provide authentic files and right feedback.

### 3.6.1.2 Pure Malicious Peers

Pure malicious peers provide inauthentic files and wrong feedback. Wrong feedback can be given in many ways [59], however we are considering the case in which, malicious peers give best feedback for other malicious peers and worst feedback for the good peers.

### 3.6.1.3 Unpredictable Peers

These peers behave as good peers for some time. After earning good reputation, they start behaving like malicious peers. Since behavior of these peers change dynamically and it is very difficult to predict their behavior, hence they are called unpredictable peers.

### 3.6.1.4 Malicious Collectives

Malicious collectives are group of peers, those who know each other and increase each others reputation values and decrease it for all others. Malicious collectives always prefer to choose the source peers from their group and increase their reputation by giving maximum weight to their files and minimum to all others' files. If there are more than one malicious groups then malicious collectives prefer to choose the source peer among their own group and for the other groups they behave just like a pure malicious peer.

### 3.6.2 Performance of Absolute Trust for Various Settings of Parameters

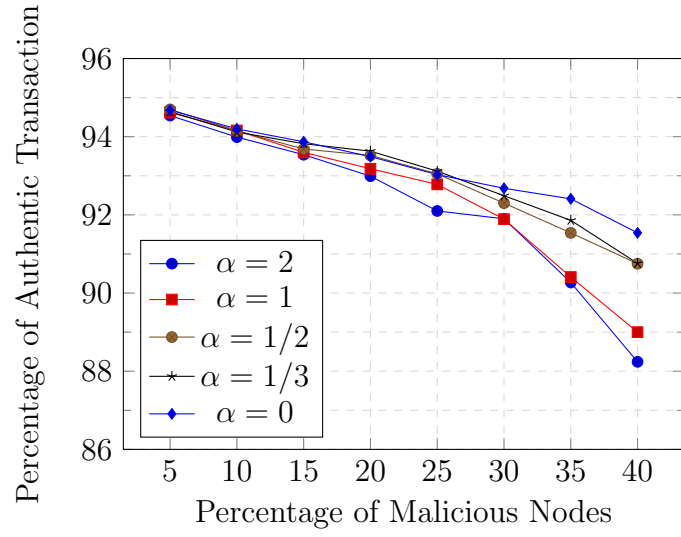
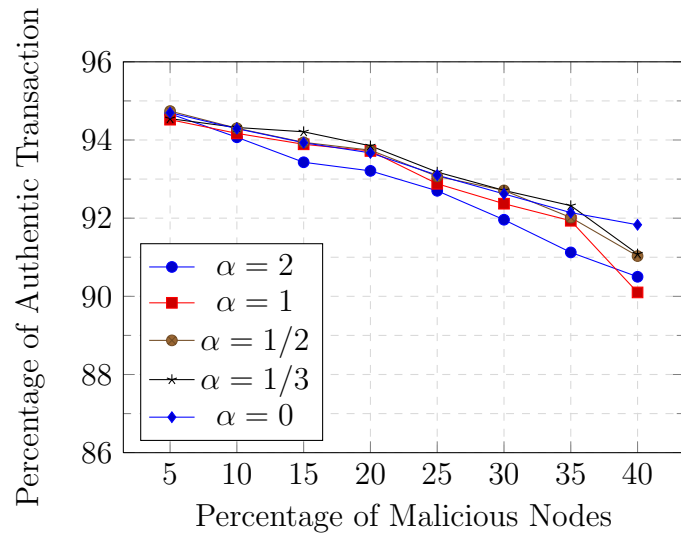
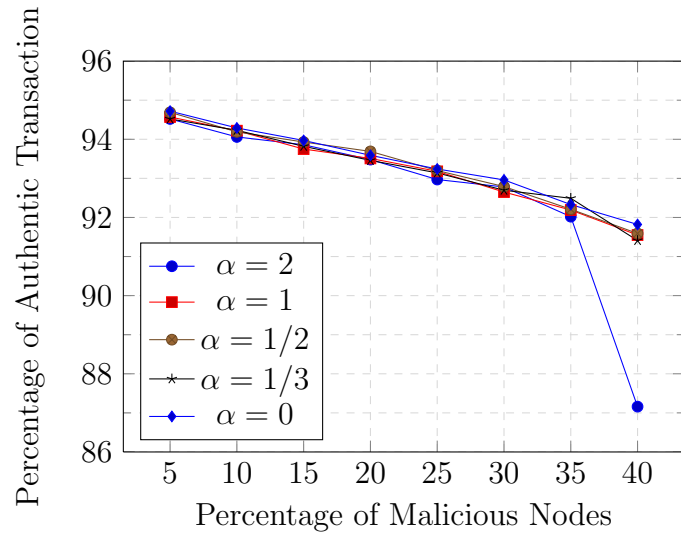
In this subsection, the impact of variation in different parameters on Absolute Trust will be analyzed. For simplicity, we consider the presence of good peers and pure malicious peers. In each experiment, fraction of malicious peers is varied from 5% to 40%.

#### 3.6.2.1 Authentic Download for Different Settings of $\alpha$ & $t_{Th}$

To see the impact of variation of  $\alpha$  and different threshold values of global trust, experiment is conducted for three different values of  $t_{Th}$ , i.e., 4.5, 5, 5.5, in each case,  $\alpha$  is varied from 0 to 2. Results for percentage of authentic downloads are shown in Fig. 3.2(a), (b) and (c). We can observe that percentage of authentic download is higher for lower value of  $\alpha$ . For any  $\alpha$ , if value of  $t_{Th}$  increases, percentage of authentic download also increases slightly. It is more evident at higher percentage of malicious nodes.

#### 3.6.2.2 Rejected Transactions for Different Settings of $\alpha$ and $t_{Th}$

Consider a situation when all the responding peers are not malicious peers but their global trust is slightly lesser than  $t_{Th}$ . They are rejected because of higher value of  $t_{Th}$ , which leads to unsuccessful transaction. Therefore, the value of  $t_{Th}$  should not be very high. The impact on percentage of rejections between good peers, i.e., good peers rejected by good peers, is shown in Fig. 3.3(a), (b) and (c). Again in each case  $\alpha$  is varied from 0 to 2. We can see from the figure that the percentage of rejection is higher when the value of  $t_{Th}$  is higher. This difference is more clearly seen at higher percentage of malicious peers. For  $t_{Th} = 4.5$ , percentage of rejection is less than 1% at  $\alpha \geq 1/3$ .

(a)  $t_{Th} = 4.5$ (b)  $t_{Th} = 5$ (c)  $t_{Th} = 5.5$ Fig. 3.2: Authentic download for different threshold values of global trust ( $t_{Th}$ ) and  $\alpha$

Therefore, the value of  $t_{Th}$  should not be very high and should not be very low. Ideally, it should be in the middle of 1 and 10, i.e., 5.5.

### 3.6.2.3 Authentic Download for Different Settings of $N$

To justify the statement about the number of peers, made in previous subsection, we performed the experiments for different number of peers. Results are shown in Fig. 3.4. It is evident from figure that percentage of authentic downloads are almost in same range irrespective of the number of peers in the networks.

### 3.6.2.4 Authentic Download During Process of Execution

To see the performance of our algorithm for intermediate states, we performed the experiments for  $\alpha=1/3$  and  $t_{Th}=1$  & 4.5. Percentage of authentic downloads are measured after every 200 transactions. Results are shown in Fig. 3.5. It is evident from figure that except in transient phase, percentage of authentic downloads always increase as the number of transactions increases. It imply as system learns more, authentic transaction happen more.

### 3.6.3 Comparison of Absolute Trust with Other Reputation Systems

In this subsection, we compared the performance of our algorithm with other reputation systems [1], [2]. Results show that our algorithm performs better under various behavioral conditions of peers.



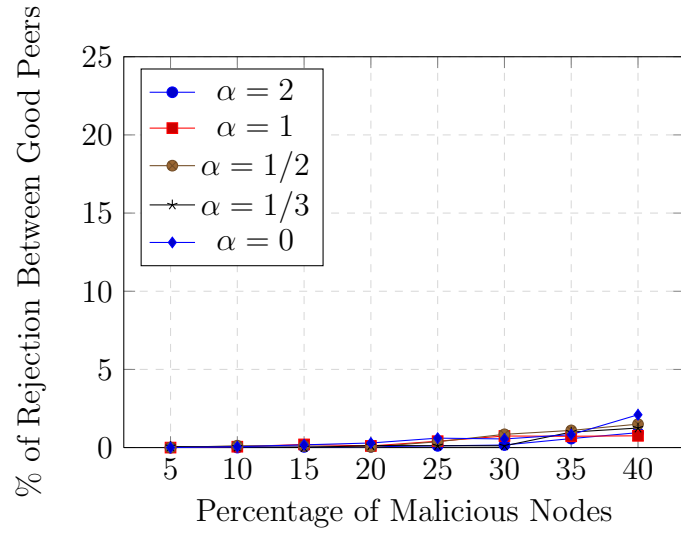
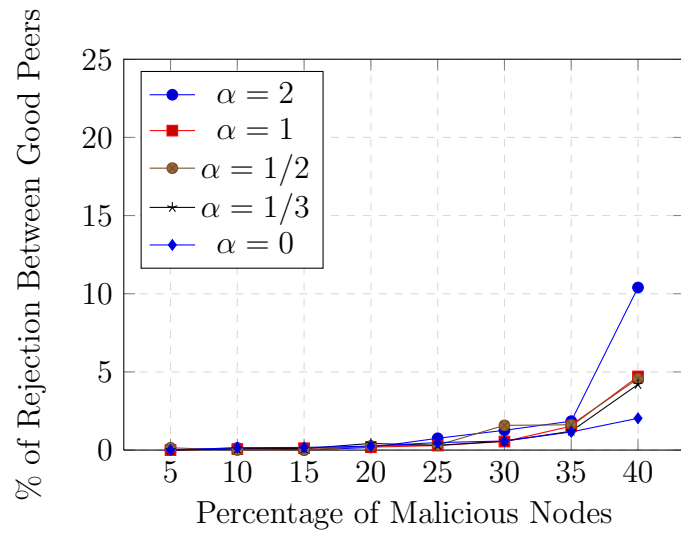
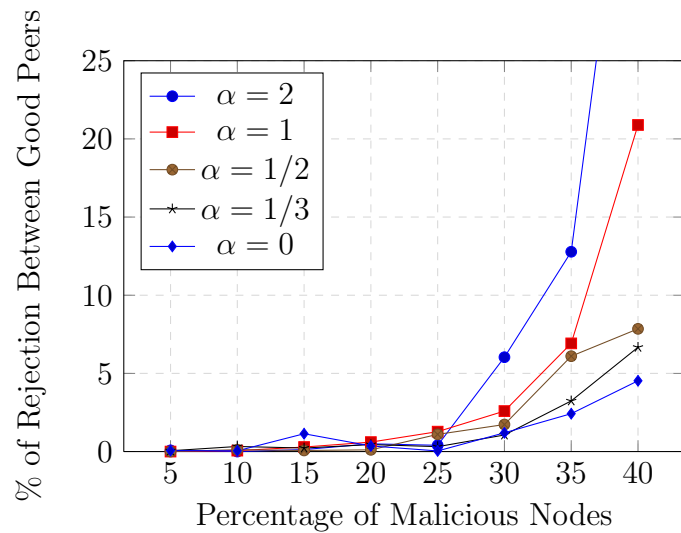
(a)  $t_{Th} = 4.5$ (b)  $t_{Th} = 5$ (c)  $t_{Th} = 5.5$ 

Fig. 3.3: Rejected transactions, i.e., good peers are rejecting good peers, for different threshold values of global trust ( $t_{Th}$ ) and  $\alpha$

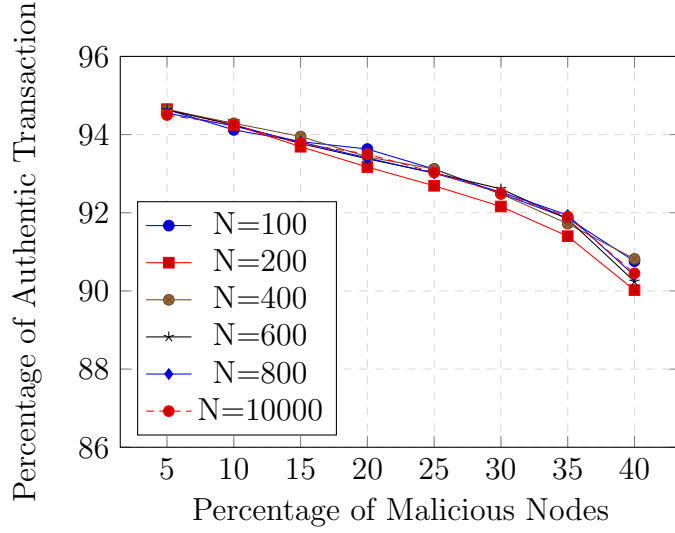


Fig. 3.4: Authentic download for different settings of node population, ( $\alpha = 1/3, t_{Th} = 4.5$ )

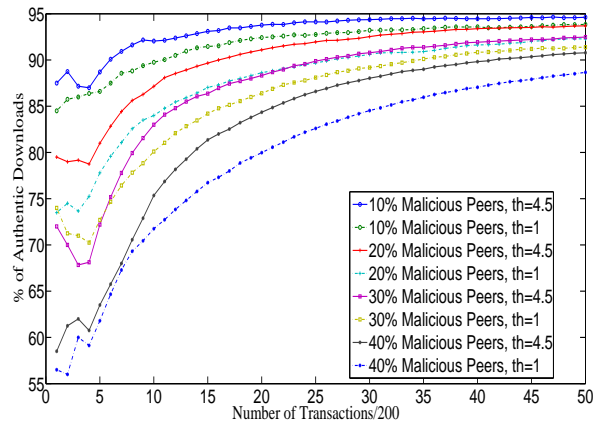


Fig. 3.5: Intermediate values of % of authentic download in the presence of pure malicious peers, ( $\alpha = 1/3$ )

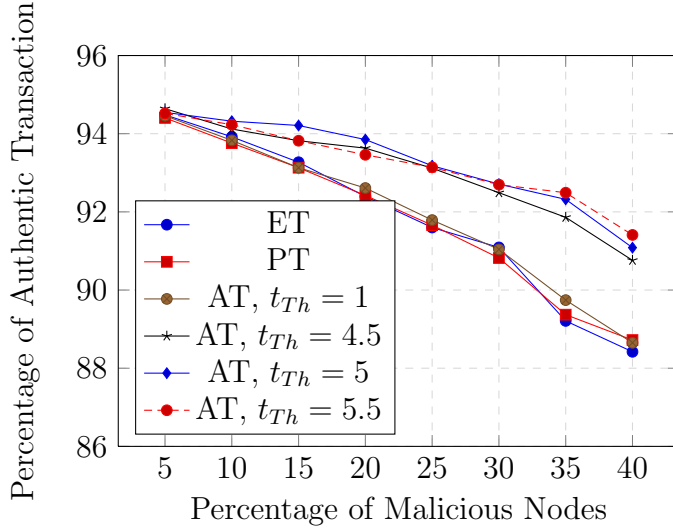


Fig. 3.6: Performance in the presence of pure malicious peers, ( $\alpha = 1/3$ )

### 3.6.3.1 Performance in the Presence of Pure Malicious Peers

In this model, there are some good and some malicious peers in the network. Results for percentage of authentic downloads are plotted in Fig. 3.6. It can be observed from this figure that the percentage of authentic download is higher in Absolute Trust for  $t_{Th} \geq 4.5$ , as compared to the EigenTrust [1] and PowerTrust [2]. Here we can see that maximum percent of authentic download is 95% because 5 % of time peers behave just opposite of their defined behavior.

### 3.6.3.2 Performance in the Presence of Unpredictable Peers

The simulation is performed in the presence of unpredictable peers. Initially, there are 10% of purely malicious peers and few unpredictable peers in the network. We increased the percentage of unpredictable peers from 5% to 30% and plotted the results in Fig. 3.7. We can see from this figure that the Absolute Trust performs significantly better. We can also see that PowerTrust [2] is vulnerable to unpredictable malicious

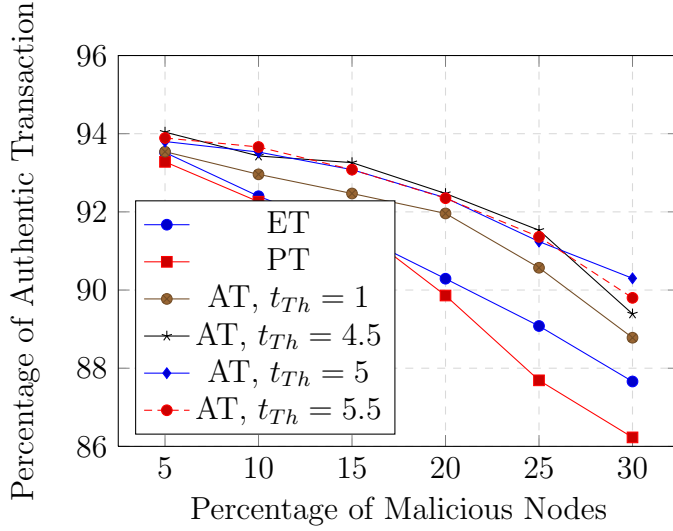


Fig. 3.7: Performance in the presence of unpredictable peers: 10% of peers are purely malicious and few behave as good peers upto some time then behave maliciously after that, ( $\alpha = 1/3$ )

peers attack. This is because in some cases, unpredictable malicious peers can earn the good reputation upto some time and can be elected as power nodes in the network, and then they can start misusing this reputation. Since power nodes play a major role in PowerTrust [2], if wrong nodes are elected as power nodes, they can damage the system to a greater extent. Chances of unpredictable peers getting elected as a power node is higher when percentage of malicious nodes are more. We can observe this from Fig. 3.7, when percentage of unpredictable peers reaches 25%, the authentic download decreases rapidly.

To see the adaptability of Absolute Trust in the presence of unpredictable peers, we plotted the results as function of time in Fig. 3.8. In this experiment, half of the unpredictable peers changed their behavior from 2000<sup>th</sup> transaction and rest from 4000<sup>th</sup> transaction. We can observe from the figure that percentage of authentic download decreases after 2000<sup>th</sup> and 4000<sup>th</sup> transaction and then gradually increases. These

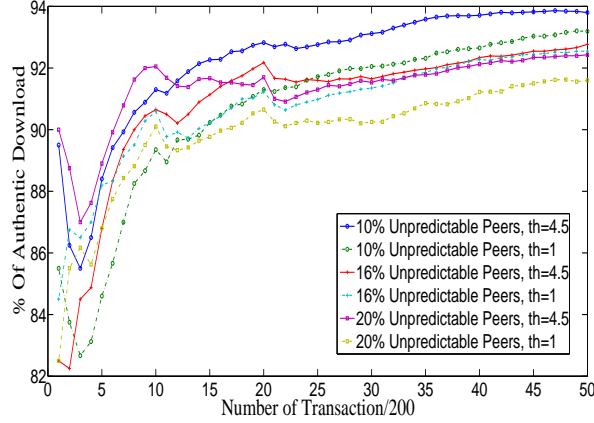


Fig. 3.8: Half of the unpredictable peers changing their behavior from 2000<sup>th</sup> transaction and rest from 4000<sup>th</sup> transaction, ( $\alpha = 1/3$ )

unpredictable peers are identified by Absolute Trust and suppressed immediately.

### 3.6.3.3 Performance in the Presence of Malicious Collectives

We have also performed the simulation in the presence of malicious collectives. Practically speaking, percent of peers making malicious collective cannot be more than 5% to 10% however, there can be many number of malicious groups. Keeping this thing in view, we kept 5% of the peers in one group and a number of groups are increased from 1 to 8. Results of simulations are plotted in Fig. 3.9. We can observe that Absolute Trust is performing slightly better than the rest of the two.

### 3.6.3.4 Analysis of Load Distribution Among the Peers

Among all the responding peers, a peer is selected as a source peer if its global trust is more than the threshold value and higher as compared to others. In the Absolute Trust, we are calculating the global trust of peers such that it is not reducing the global trust of others. Whereas in relative ranking, peers are competing with each other for

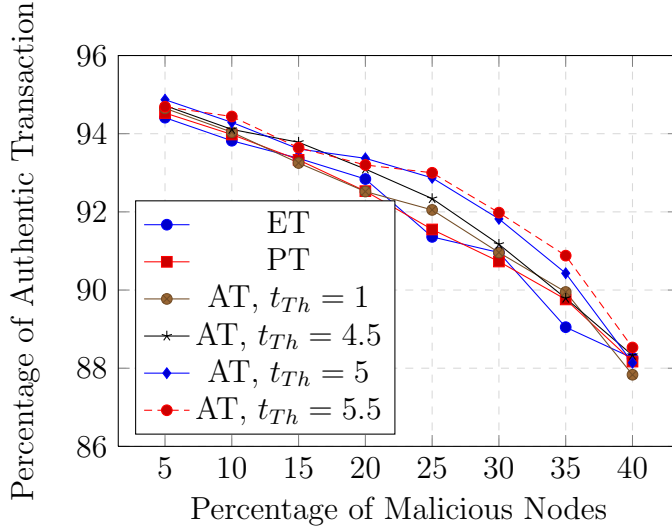


Fig. 3.9: Performance in the presence of malicious collectives, ( $\alpha = 1/3$ )

global trust, i.e., if global trust of any peer is increased by some fraction, it will decrease the global trust of other peers. Hence, relative difference between the global trusts will always be higher in case of relative ranking.

In simulation, we calculated the load of each peer, i.e., number of times a particular peer is selected as a source peer. Then we calculated the standard deviation of load among all the peers. Simulation is performed in the presence of pure malicious peers, and standard deviation of load is calculated only among the good nodes, because majority of the load have to be shared by good peers only. Malicious peers are increased from 5% to 40%. The results of simulation are plotted in Fig. 3.10. We can see, the standard deviation of load distribution is minimum in Absolute Trust.

## 3.7 Conclusion

In this chapter, we presented an algorithm for aggregation of local trust in P2P network. We have seen that our algorithm is able to fulfill all design considerations

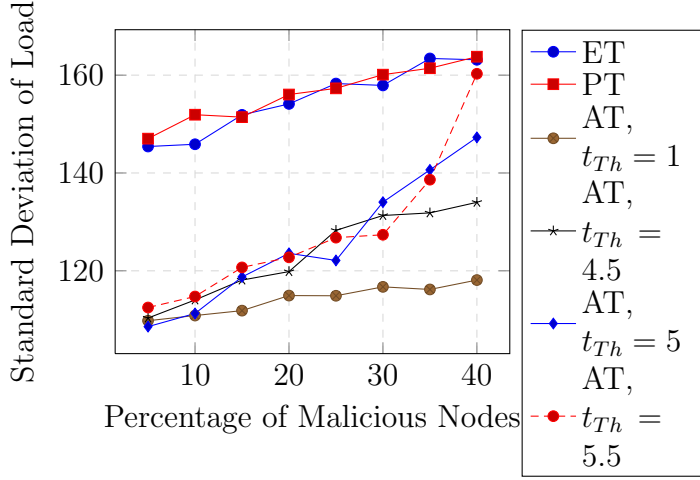


Fig. 3.10: Standard deviation of load distribution among the peers in different Reputation Systems, ( $\alpha = 1/3$ )

mentioned in the Section 3.2. Aggregation is done without normalization, hence it is true reflection of past behavior of peers in network. The calculation of global trust is done by iterative method and it converges at some unique value. For lesser value of  $\alpha$ , the algorithm converges much faster. The updates have to be sent about only those peers whose local trust value is changing. Hence, a lesser number of messages are required to update the global trust. This algorithm can be implemented in a distributed system where no central authority is present. We have presented the results for simulation and it shows that this algorithm is robust against the various attacks like individual malicious, unpredictable malicious and collective malicious peers. Lastly, we have shown through simulations that because peers are not competing with each other for higher value of global trust, thus, the load on good peers is more uniform compared to the relative ranking mechanism.

## Chapter 4

# Generalized Analysis of Convergence of Absolute Trust

### 4.1 Introduction

Open and anonymous nature of peer-to-peer networks provides an opportunity for malicious peers to behave unpredictably in the network. This leads to the lack of trust among peers. To control the behavior of peers in the network, a reputation system can be used. In the reputation system, aggregation of trust is a primary issue. Algorithm for aggregation of trust should be designed such that, it can converge to a certain finite value. Absolute Trust is one such trust aggregation algorithm, which was introduced in the previous chapter. In this chapter, we present the generalized analysis of convergence of the Absolute Trust algorithm <sup>1</sup>.

---

<sup>1</sup>**S. K. Awasthi** and Y. N. Singh, “Generalized analysis of Absolute Trust in peer-to-peer Networks” *IEEE Communication Letters*, vol. 20, no. 7, pp. 1345-1348, July 2016.



## 4.2 Motivation

Reputation systems are being studied [1], [2], [31], [32] to prevent the attacks by rogue peers. Absolute Trust models one such system. This model characterizes the past behavior of peers in the network, and can be implemented as a truly distributed system. In this model, peers evaluate each other locally, and the local trust for each other is aggregated in the whole network. The aggregated trust is called global trust, and is evaluated recursively.

In recursive solution of any equation, error in each iteration must reduce and for large number of iteration it should tend to zero. This will guarantee the uniqueness of the solution. It was shown in previous chapter that if the error in global trust is less compared to the actual solution, then it will converge to zero. But analysis for large error was not presented. In this chapter, we will show that in any step, if error is very large compared to the actual solution then it will converge much faster in that step.

## 4.3 P2P Model and Absolute Trust

Let there be  $N$  peers in a peer to peer network. In this network, peer  $i$  can be evaluated by peer  $j$  based on service provided by peer  $i$  in the past. Evaluated value  $T_{ji}$  can be represented by a number from one to ten. One is for worst service and ten is for best service. If there is no interaction between peers,  $T_{ji}$  will be zero.  $T_{ji}$  is called the local trust of peer  $i$ , evaluated by peer  $j$ . All local trust values evaluated by various nodes can be aggregated in the whole network. Aggregated global trust of peer  $i$  can

be given by (see previous chapter)

$$t_i = \left[ \left( \frac{\sum_{j \in S_i} T_{ji} t_j}{\sum_{j \in S_i} t_j} \right)^p \cdot \left( \frac{\sum_{j \in S_i} t_j^2}{\sum_{j \in S_i} t_j} \right)^q \right]^{\frac{1}{p+q}}. \quad (4.1)$$

Here,  $S_i$  is the set of peers evaluating the service of peer  $i$ ,  $t_j$  is global trust of peer  $j$ , and  $p, q$  are suitably chosen constants. Equation (4.1) can be rearranged as follows:

$$\begin{aligned} t_i &= \left[ \left( \frac{\mathbf{e}_i \mathbf{T}^{\text{tr}} \mathbf{t}}{\mathbf{e}_i \mathbf{C} \mathbf{t}} \right)^p \left( \frac{\mathbf{e}_i \mathbf{C} \cdot \text{diag}(\mathbf{t}) \cdot \mathbf{t}}{\mathbf{e}_i \mathbf{C} \mathbf{t}} \right)^q \right]^{\frac{1}{p+q}} \\ &= \left[ \left( \frac{\mathbf{e}_i \mathbf{T}^{\text{tr}} \mathbf{t}}{\mathbf{e}_i \mathbf{C} \mathbf{t}} \right)^{\frac{1}{1+\alpha}} \left( \frac{\mathbf{e}_i \mathbf{C} \cdot \text{diag}(\mathbf{t}) \cdot \mathbf{t}}{\mathbf{e}_i \mathbf{C} \mathbf{t}} \right)^{\frac{\alpha}{1+\alpha}} \right] \end{aligned}$$

Here,  $\mathbf{t}$  is global trust vector. Its  $i^{\text{th}}$  entry is a global trust of peer  $i$ .  $\text{diag}(\mathbf{t})$  is a diagonal matrix with its  $ii$  entry as  $t_i$ .  $\mathbf{T}$  is trust matrix with its element  $T_{ij}$  as a local trust of peer  $j$ , evaluated by peer  $i$ .  $\mathbf{T}^{\text{tr}}$  is transpose of matrix  $\mathbf{T}$ .  $\mathbf{C}$  is incidence matrix corresponding to matrix  $\mathbf{T}^{\text{tr}}$ , i.e.  $C_{ij} = 1$  if  $T_{ji} > 0$  otherwise  $C_{ij} = 0$ .  $\mathbf{e}_i$  is the row vector with its  $i^{\text{th}}$  entry as '1' and all the other entries as zero. Here,  $\alpha = q/p$ .

## 4.4 Analysis of Convergence of Absolute Trust

### 4.4.1 Center Point of the Matrix

**Definition 4.4.1.** Center point of non-negative matrix  $\mathbf{T}^{\text{tr}}$  can be defined as the column vector  $\mathbf{t}$ . Where, its  $i^{\text{th}}$  element  $t_i$  will be  $(\mathbf{e}_i \mathbf{T}^{\text{tr}} \mathbf{t} / \mathbf{e}_i \mathbf{C} \mathbf{t})$ .

**Definition 4.4.2.** A non-negative matrix  $\mathbf{M}$  is said to be mutually exclusive with a non-negative matrix  $\mathbf{N}$ , if  $M_{ij} > 0$  then  $N_{ij} = 0$ . It also implies that when  $N_{ij} > 0$  then  $M_{ij} = 0$ .

**Lemma 4.4.1.** *Center point of any non-negative, irreducible matrix  $\mathbf{T}^{\text{tr}}$  is unique and can be calculated by an iterative function*

$$\mathbf{t}^k = \phi_1(\mathbf{t}^{k-1}) = [\text{diag}(d_1, d_2, \dots, d_N)]^{-1} \cdot \mathbf{T}^{\text{tr}} \cdot \mathbf{t}^{k-1},$$

where  $d_i = \mathbf{e}_i \mathbf{C} \mathbf{t}$ .

**Proof.**  $i^{\text{th}}$  element of iterative function  $\phi_1(\mathbf{t}^{k-1})$  is

$$t_i^k = \frac{(\mathbf{e}_i \mathbf{T}^{\text{tr}} \cdot \mathbf{t}^{k-1})}{(\mathbf{e}_i \mathbf{C} \cdot \mathbf{t}^{k-1})} \quad (4.2)$$

Let  $t_i^k$  and  $t_i^{k-1}$  are, far from actual solution  $t_i$  by  $\delta t_i^k$  and  $\delta t_i^{k-1}$  respectively, then

$$\begin{aligned} t_i + \delta t_i^k &= \left[ \frac{(\mathbf{e}_i \mathbf{T}^{\text{tr}} \cdot (\mathbf{t} + \delta \mathbf{t}^{k-1}))}{(\mathbf{e}_i \mathbf{C} \cdot (\mathbf{t} + \delta \mathbf{t}^{k-1}))} \right] \\ &= \left[ \frac{(\mathbf{e}_i \mathbf{T}^{\text{tr}} \cdot \mathbf{t})}{(\mathbf{e}_i \mathbf{C} \cdot \mathbf{t})} \right] \left[ \frac{(1 + \frac{\mathbf{e}_i \cdot \mathbf{T}^{\text{tr}} \cdot \delta \mathbf{t}^{k-1}}{\mathbf{e}_i \cdot \mathbf{T}^{\text{tr}} \cdot \mathbf{t}})}{(1 + \frac{\mathbf{e}_i \cdot \mathbf{C} \cdot \delta \mathbf{t}^{k-1}}{\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{t}})} \right]. \end{aligned}$$

From definition 4.4.1,  $(\mathbf{e}_i \mathbf{T}^{\text{tr}} \mathbf{t} / \mathbf{e}_i \mathbf{C} \mathbf{t})$  is  $i^{\text{th}}$  element of center point of matrix  $\mathbf{T}^{\text{tr}}$ . So we can write,

$$t_i + \delta t_i^k = t_i \cdot \left[ \frac{(1 + \frac{\mathbf{e}_i \cdot \mathbf{T}^{\text{tr}} \cdot \delta \mathbf{t}^{k-1}}{\mathbf{e}_i \cdot \mathbf{T}^{\text{tr}} \cdot \mathbf{t}})}{(1 + \frac{\mathbf{e}_i \cdot \mathbf{C} \cdot \delta \mathbf{t}^{k-1}}{\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{t}})} \right]$$

$$\delta t_i^k = t_i \cdot \left[ \frac{(1 + \frac{\mathbf{e}_i \cdot \mathbf{T}^{\text{tr}} \cdot \delta \mathbf{t}^{k-1}}{\mathbf{e}_i \cdot \mathbf{T}^{\text{tr}} \cdot \mathbf{t}})}{(1 + \frac{\mathbf{e}_i \cdot \mathbf{C} \cdot \delta \mathbf{t}^{k-1}}{\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{t}})} - 1 \right]$$

$$\delta t_i^k = \frac{t_i}{(1 + \frac{\mathbf{e}_i \cdot \mathbf{C} \cdot \delta \mathbf{t}^{k-1}}{\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{t}})} \cdot \left[ \frac{\mathbf{e}_i \cdot \mathbf{T}^{\text{tr}} \cdot \delta \mathbf{t}^{k-1}}{\mathbf{e}_i \cdot \mathbf{T}^{\text{tr}} \cdot \mathbf{t}} - \frac{\mathbf{e}_i \cdot \mathbf{C} \cdot \delta \mathbf{t}^{k-1}}{\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{t}} \right]$$

$$\begin{aligned}
&= \frac{1}{\left(1 + \frac{\mathbf{e}_i \cdot \mathbf{C} \cdot \delta \mathbf{t}^{k-1}}{\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{t}}\right)} \cdot \left[ \frac{t_i \mathbf{e}_i \cdot \mathbf{T}^{\text{tr}}}{\mathbf{e}_i \cdot \mathbf{T}^{\text{tr}} \cdot \mathbf{t}} - \frac{t_i \mathbf{e}_i \cdot \mathbf{C}}{\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{t}} \right] \cdot \delta \mathbf{t}^{k-1} \\
&= \frac{1}{\left(1 + \frac{\mathbf{e}_i \cdot \mathbf{C} \cdot \delta \mathbf{t}^{k-1}}{\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{t}}\right)} \cdot \left[ \mathbf{A}_i - \mathbf{B}_i \right] \cdot \delta \mathbf{t}^{k-1} \\
&= f_i(\delta \mathbf{t}^{k-1}) \cdot \left[ \mathbf{A}_i - \mathbf{B}_i \right] \cdot \delta \mathbf{t}^{k-1}
\end{aligned}$$

where  $\mathbf{A}_i$  and  $\mathbf{B}_i$  are  $i^{\text{th}}$  row of  $N \times N$  matrices  $\mathbf{A}$  and  $\mathbf{B}$  respectively. It can be observed easily that

$$\mathbf{A} \cdot \mathbf{t} = \mathbf{t}$$

and

$$\mathbf{B} \cdot \mathbf{t} = \mathbf{t}.$$

Matrices  $\mathbf{A}, \mathbf{B}$  have non zero elements at the same positions as in matrix  $\mathbf{T}^{\text{tr}}$ , hence  $\mathbf{A}, \mathbf{B}$  are also irreducible. Therefore spectral radius of  $\mathbf{A}, \mathbf{B}$  will be '1'(see[61]).

Now

$$f_i(\delta \mathbf{t}^{k-1}) = \frac{1}{\left(1 + \frac{\mathbf{e}_i \cdot \mathbf{C} \cdot \delta \mathbf{t}^{k-1}}{\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{t}}\right)}$$

If  $\delta \mathbf{t}^{k-1} \ll \mathbf{t}$  then  $f_i(\delta \mathbf{t}^{k-1}) \approx 1$

Hence,

$$\delta \mathbf{t}^k = [\mathbf{A} - \mathbf{B}] \cdot \delta \mathbf{t}^{k-1}$$

$$\lim_{k \rightarrow \infty} \delta \mathbf{t}^k = \lim_{k \rightarrow \infty} [\mathbf{A} - \mathbf{B}]^k \delta \mathbf{t}^0 = \mathbf{0}$$

(see Theorem 3.4.1 in previous chapter) here  $\delta \mathbf{t}^0$  is the initial error in  $\mathbf{t}$ .

Now, for the case when  $\delta \mathbf{t}^{k-1} > \mathbf{t}$  then  $f_i(\delta \mathbf{t}^{k-1}) < 1$ , hence in each step,  $\delta t_i^k$  will decrease more rapidly. If  $\delta \mathbf{t}^{k-1} \approx \mathbf{t}$  then  $f_i(\delta \mathbf{t}^{k-1}) \approx 1/2$ , in this case, each  $\delta t_i^k$  will be reduced to more than half of its value in previous step.

Hence we can conclude that center point of any non negative, irreducible matrix can be calculated by the above iterative function. Error in each step will depend upon the error in previous step. In any step, error will reduce very fast, if current value of  $\mathbf{t}$  is far from actual solution and after large iterations, the error,  $\delta \mathbf{t}^k$ , will become zero.  $\square$

**Lemma 4.4.2.** *If vector  $\mathbf{t}$  is the center point of matrix  $\mathbf{C}.\text{diag}(\mathbf{t})$ , then center point will lie on the vector  $\mathbf{e}$ . It can be calculated by iterative function*

$$\mathbf{t}^k = \phi_2(\mathbf{t}^{k-1}) = [\text{diag}(d_1, d_2, \dots, d_N)]^{-1} \cdot \mathbf{C}.\text{diag}(\mathbf{t}^{k-1}).\mathbf{t}^{k-1}$$

where  $\mathbf{e}$  is a vector with each element as '1' and  $d_i = \mathbf{e}_i \mathbf{C} \mathbf{t}^{k-1}$ .

**Proof.** The iterative function can be written as

$$\begin{aligned} \mathbf{t}^k &= \mathbf{M}(\mathbf{k} - 1).\mathbf{t}^{k-1} \\ &= \mathbf{M}(\mathbf{k} - 1).\mathbf{M}(\mathbf{k} - 2) \dots \mathbf{M}(0).\mathbf{t}^0 \end{aligned}$$

where  $i^{th}$  row of matrix  $\mathbf{M}(\mathbf{k})$  is  $\left( \frac{\mathbf{e}_i \mathbf{C}.\text{diag}(\mathbf{t}^k)}{\mathbf{e}_i \mathbf{C}.\mathbf{t}^k} \right)$ . We can easily prove that for matrix  $\mathbf{M}(\mathbf{k})$ , sum of its each row is one. Hence it has '1' as an eigen value. The corresponding eigen vector will be  $\mathbf{e}$ . For positive initial guess of  $\mathbf{t}^0$ , all matrices  $\mathbf{M}(\mathbf{k})$  will be non-negative and irreducible. So we can conclude that spectral radius of all  $\mathbf{M}(\mathbf{k})$  is '1'. Hence (see Lemma 3.4.2 in previous chapter)

$$\lim_{k \rightarrow \infty} \mathbf{t}^k = \lim_{k \rightarrow \infty} \mathbf{M}(\mathbf{k} - 1).\mathbf{M}(\mathbf{k} - 2) \dots \mathbf{M}(0).\mathbf{t}^0 = \mathbf{e}$$

$$\lim_{k \rightarrow \infty} \mathbf{t}^k = \lim_{k \rightarrow \infty} \phi_2(\mathbf{t}^{k-1}) = \mathbf{e}$$

□

### 4.4.2 Properties of Center Point

**Property 4.4.1.** If center point of matrix  $\mathbf{M}$  is  $\mathbf{t}$  then center point of matrix  $k\mathbf{M}$  will be  $k\mathbf{t}$ . Where  $k$  is any arbitrary scalar.

**Proof.** Let  $t_i$  is  $i^{th}$  element of  $\mathbf{t}$  then

$$t_i = \frac{\mathbf{e}_i \cdot \mathbf{M} \cdot \mathbf{t}}{\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{t}}$$

or

$$kt_i = \frac{\mathbf{e}_i \cdot k\mathbf{M} \cdot \mathbf{t}}{\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{t}}$$

or

$$(kt_i) = \frac{\mathbf{e}_i \cdot (k\mathbf{M}) \cdot (k\mathbf{t})}{\mathbf{e}_i \cdot \mathbf{C} \cdot (k\mathbf{t})}$$

hence  $k\mathbf{t}$  is center point of matrix  $k\mathbf{M}$

□

**Property 4.4.2.** If all entries of the  $N \times N$  matrix  $\mathbf{M}$  are positive ( $M_{ij} > 0, \forall i, j$ ) then center point,  $\mathbf{t}$ , will lie on the principle eigen vector of matrix.

**Proof.** If all the entries of the matrix  $\mathbf{M}$  are positive then

$$\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{t} = \sum_{j=1}^N t_j = \lambda \quad \forall i$$

Here  $\lambda$  will be a constant. Further,

$$t_i = \frac{\mathbf{e}_i \cdot \mathbf{M} \cdot \mathbf{t}}{\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{t}} = \frac{\mathbf{e}_i \cdot \mathbf{M} \cdot \mathbf{t}}{\lambda}.$$

Hence  $\mathbf{M}\mathbf{t} = \lambda\mathbf{t}$ . Hence  $\lambda$  will be spectral radius and  $\mathbf{t}$  will be principle eigen vector of matrix  $\mathbf{M}$  (see[61])  $\square$

**Property 4.4.3.** If center point of  $m$  non-negative, irreducible and **mutually exclusive** matrices  $\mathbf{M}_1, \mathbf{M}_2, \dots, \mathbf{M}_m$  are same, then the center point of their sum will also be the same.

**Proof.** Let the incidence matrix corresponding to matrices  $\mathbf{M}_1, \mathbf{M}_2, \dots, \mathbf{M}_m$  are  $\mathbf{C}_1, \mathbf{C}_2, \dots, \mathbf{C}_m$  respectively. Now if  $\mathbf{M} = \mathbf{M}_1 + \mathbf{M}_2 + \dots + \mathbf{M}_m$  then incidence matrix corresponding to matrix  $\mathbf{M}$  will be  $\mathbf{C} = \mathbf{C}_1 + \mathbf{C}_2 + \dots + \mathbf{C}_m$  because matrices  $\mathbf{M}_1, \mathbf{M}_2, \dots, \mathbf{M}_m$  are mutually exclusive. Let  $i^{th}$  element of center point of matrices  $\mathbf{M}_1, \mathbf{M}_2, \dots, \mathbf{M}_m$  be  $t_i$  then

$$t_i = \frac{\mathbf{e}_i \cdot \mathbf{M}_j \cdot \mathbf{t}}{\mathbf{e}_i \cdot \mathbf{C}_j \cdot \mathbf{t}} \quad \forall j$$

or

$$t_i \mathbf{e}_i \cdot \mathbf{C}_j \cdot \mathbf{t} = \mathbf{e}_i \cdot \mathbf{M}_j \cdot \mathbf{t}$$

taking summation on both side w.r.t.  $j$

$$\sum_{j=1}^m t_i \mathbf{e}_i \cdot \mathbf{C}_j \cdot \mathbf{t} = \sum_{j=1}^m \mathbf{e}_i \cdot \mathbf{M}_j \cdot \mathbf{t}$$

$$t_i \mathbf{e}_i \cdot \left( \sum_{j=1}^m \mathbf{C}_j \right) \cdot \mathbf{t} = \mathbf{e}_i \cdot \left( \sum_{j=1}^m \mathbf{M}_j \right) \cdot \mathbf{t}$$

$$t_i \mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{t} = \mathbf{e}_i \cdot \mathbf{M} \cdot \mathbf{t}$$

or

$$t_i = \frac{\mathbf{e}_i \cdot \mathbf{M} \cdot \mathbf{t}}{\mathbf{e}_i \cdot \mathbf{C} \cdot \mathbf{t}}$$

hence  $\mathbf{t}$  is also the center point of matrix  $\mathbf{M}$ .

□

### 4.4.3 Convergence of Absolute Trust for Large Error in Initial Guess

It was stated in previous chapter that global trust in (4.1) can be calculated by iterative function

$$\mathbf{t}^k = \phi(\mathbf{t}^{k-1}) = [\mathbf{diag}(d_1, d_2, \dots, d_N) \mathbf{T}^{\text{tr}} \cdot \mathbf{t}^{k-1}]^{\frac{1}{1+\alpha}}$$

where  $d_i = \left[ \frac{(\mathbf{e}_i \mathbf{C} \cdot \mathbf{diag}(\mathbf{t}^{k-1}) \cdot \mathbf{t}^{k-1})^\alpha}{(\mathbf{e}_i \mathbf{C} \mathbf{t}^{k-1})^{(1+\alpha)}} \right]$ . Proof was derived only for the small error that is, if initial guess of  $\mathbf{t}$  is very close to the actual solution. However global trust can be calculated for any positive initial guess. In this subsection, we will show that it will converge faster in any step if error is large compared to  $\mathbf{t}$ .

Let  $t_i^k$  and  $t_i^{k-1}$  are, far from actual solution  $t_i$  by  $\delta t_i^k$  and  $\delta t_i^{k-1}$  respectively, then

$$t_i + \delta t_i^k = \left[ \frac{(\mathbf{e}_i \mathbf{T}^{\text{tr}} \cdot (\mathbf{t} + \delta \mathbf{t}^{k-1}))}{(\mathbf{e}_i \mathbf{C} (\mathbf{t} + \delta \mathbf{t}^{k-1}))} \right]^{\frac{1}{1+\alpha}} \left[ \frac{(\mathbf{e}_i \mathbf{C} \cdot \mathbf{diag}(\mathbf{t} + \delta \mathbf{t}^{k-1}) \cdot (\mathbf{t} + \delta \mathbf{t}^{k-1}))}{(\mathbf{e}_i \mathbf{C} (\mathbf{t} + \delta \mathbf{t}^{k-1}))} \right]^{\frac{\alpha}{1+\alpha}}$$

If error  $\delta \mathbf{t}^{k-1} > \mathbf{t}$  then we can approximate  $\mathbf{t} + \delta \mathbf{t}^{k-1} \approx \delta \mathbf{t}^{k-1}$  hence

$$\delta t_i^k = \left[ \frac{(\mathbf{e}_i \mathbf{T}^{\text{tr}} \cdot \delta \mathbf{t}^{k-1})}{(\mathbf{e}_i \mathbf{C} \cdot \delta \mathbf{t}^{k-1})} \right]^{\frac{1}{1+\alpha}} \cdot \left[ \frac{(\mathbf{e}_i \mathbf{C} \cdot \mathbf{diag}(\delta \mathbf{t}^{k-1}) \cdot \delta \mathbf{t}^{k-1})}{(\mathbf{e}_i \mathbf{C} \cdot \delta \mathbf{t}^{k-1})} \right]^{\frac{\alpha}{1+\alpha}}$$

Using Young's Inequality [66], i.e.

$$c.d \leq \frac{1}{1+\alpha} c^{1+\alpha} + \frac{\alpha}{1+\alpha} d^{(1+\alpha)/\alpha};$$



and taking

$$c = \left[ \frac{(\mathbf{e}_i \mathbf{T}^{\text{tr}} \cdot \delta \mathbf{t}^{\mathbf{k}-1})}{(\mathbf{e}_i \mathbf{C} \cdot \delta \mathbf{t}^{\mathbf{k}-1})} \right]^{\frac{1}{1+\alpha}}$$

and

$$d = \left[ \frac{(\mathbf{e}_i \mathbf{C} \cdot \text{diag}(\delta \mathbf{t}^{\mathbf{k}-1}) \cdot \delta \mathbf{t}^{\mathbf{k}-1})}{(\mathbf{e}_i \mathbf{C} \cdot \delta \mathbf{t}^{\mathbf{k}-1})} \right]^{\frac{\alpha}{1+\alpha}},$$

We can write

$$\delta t_i^{\mathbf{k}} \leq \frac{1}{1+\alpha} \left[ \frac{(\mathbf{e}_i \mathbf{T}^{\text{tr}} \cdot \delta \mathbf{t}^{\mathbf{k}-1})}{(\mathbf{e}_i \mathbf{C} \cdot \delta \mathbf{t}^{\mathbf{k}-1})} \right] + \frac{\alpha}{1+\alpha} \left[ \frac{(\mathbf{e}_i \mathbf{C} \cdot \text{diag}(\delta \mathbf{t}^{\mathbf{k}-1}) \cdot \delta \mathbf{t}^{\mathbf{k}-1})}{(\mathbf{e}_i \mathbf{C} \cdot \delta \mathbf{t}^{\mathbf{k}-1})} \right].$$

$$\delta \mathbf{t}^{\mathbf{k}} \leq \frac{1}{1+\alpha} \phi_1(\delta \mathbf{t}^{\mathbf{k}-1}) + \frac{\alpha}{1+\alpha} \phi_2(\delta \mathbf{t}^{\mathbf{k}-1}).$$

Here  $\phi_1(\cdot)$  and  $\phi_2(\cdot)$  are as defined in Lemma 4.4.1 and in Lemma 4.4.2 respectively.

$$\delta \mathbf{t}^{\mathbf{k}} \leq \frac{1}{1+\alpha} \mathbf{M}'_1 \cdot \delta \mathbf{t}^{\mathbf{k}-1} + \frac{\alpha}{1+\alpha} \mathbf{M}'_2 \cdot \delta \mathbf{t}^{\mathbf{k}-1} \quad (4.3)$$

It is convex combination of iterative function  $\phi_1$  and  $\phi_2$ .  $i^{\text{th}}$  row of matrix  $\mathbf{M}'_2$  is  $\left[ \frac{(\mathbf{e}_i \mathbf{C} \cdot \text{diag}(\delta \mathbf{t}^{\mathbf{k}-1}))}{(\mathbf{e}_i \mathbf{C} \cdot \delta \mathbf{t}^{\mathbf{k}-1})} \right]$  and sum of each row is one. Hence  $\infty$ -norm of matrix  $\mathbf{M}'_2$  is  $|\mathbf{M}'_2|_{\infty} = 1$  using the property of norm [54]

$$|\mathbf{M}'_2 \cdot \delta \mathbf{t}^{\mathbf{k}-1}|_{\infty} \leq |\mathbf{M}'_2|_{\infty} \cdot |\delta \mathbf{t}^{\mathbf{k}-1}|_{\infty} = |\delta \mathbf{t}^{\mathbf{k}-1}|_{\infty} \quad (4.4)$$

Function  $\phi_1$  is converging function toward center point. It is shown in Lemma 4.4.1 that in any step, if  $\mathbf{t}^{\mathbf{k}}$  is very far from the center point then it will tend towards the center point very rapidly. Hence for  $\delta \mathbf{t}^{\mathbf{k}-1} > \mathbf{t}$

$$\phi_1(\delta \mathbf{t}^{\mathbf{k}-1}) < \delta \mathbf{t}^{\mathbf{k}-1}$$

or

$$|\mathbf{M}'_1.\delta\mathbf{t}^{k-1}|_\infty < |\delta\mathbf{t}^{k-1}|_\infty \quad (4.5)$$

Now taking  $\infty$ -norm on both side of (4.3)

$$|\delta\mathbf{t}^k|_\infty \leq \left| \frac{1}{1+\alpha}\mathbf{M}'_1.\delta\mathbf{t}^{k-1} + \frac{\alpha}{1+\alpha}\mathbf{M}'_2.\delta\mathbf{t}^{k-1} \right|_\infty$$

$$\leq \frac{1}{1+\alpha}|\mathbf{M}'_1.\delta\mathbf{t}^{k-1}|_\infty + \frac{\alpha}{1+\alpha}|\mathbf{M}'_2.\delta\mathbf{t}^{k-1}|_\infty$$

Using (4.4) and (4.5),

$$|\delta\mathbf{t}^k|_\infty < \frac{1}{1+\alpha}|\delta\mathbf{t}^{k-1}|_\infty + \frac{\alpha}{1+\alpha}|\delta\mathbf{t}^{k-1}|_\infty = |\delta\mathbf{t}^{k-1}|_\infty$$

Hence

$$|\delta\mathbf{t}^k|_\infty < |\delta\mathbf{t}^{k-1}|_\infty$$

Therefore error in every step will decrease. In any step, it will decrease faster if  $\mathbf{t}^k$  is very far from actual solution. Speed of convergence depends upon  $\alpha$ . For lower  $\alpha$ , impact of  $\phi_2$  will be lower and  $\phi_1$  will dominate the speed of convergence. Hence for smaller  $\alpha$  speed of convergence will be high.

## 4.5 Numerical Results

In order to verify what has been discussed in the earlier sections, we have taken the values of a trust matrix  $\mathbf{T}$  as

$$\begin{bmatrix} 0 & 5 & 6 & 6 \\ 8 & 0 & 5 & 5 \\ 5 & 6 & 0 & 2 \\ 0 & 4 & 0 & 0 \end{bmatrix}.$$

The values of center points and the global trusts in each iteration, are calculated and shown in Table 4.1, 4.2, 4.3, 4.4 and 4.5.

### 4.5.1 Convergence of the Center Point

The convergence of center point of matrix  $\mathbf{T}^{\text{tr}}$  is shown in Table 4.1 and 4.2. In Table 4.1, initial guess  $\mathbf{t}^0 = [1 \ 2 \ 3 \ 4]^{\text{tr}}$ . It is close to the center point, and in Table 4.2, initial guess  $\mathbf{t}^0 = [100 \ 300 \ 200 \ 100]^{\text{tr}}$ , which is significantly far from center point. But we can see in both the cases that convergence happen in seven iterations. In the latter case, error is very large in  $0^{\text{th}}$  step and becomes less than  $\mathbf{t}$  within one step.

### 4.5.2 Convergence of the Global Trust

Impact of initial guess and parameter  $\alpha$  on the convergence of global trust is shown in Table 4.3, 4.4 and 4.5. In Table 4.3 and 4.4  $\alpha$  is taken as  $1/3$  but initial guess is different. Again we can see that for large initial guess of global trust, it takes only one more iteration to converge to final value. It converge very fast when error ( $\delta\mathbf{t}$ ) is very large compare to global trust ( $\mathbf{t}$ )

In Table 4.3 and 4.5 initial guess is taken same but  $\alpha$  is different and we can see that for  $\alpha = 1/6$  it converges only in eight iterations.

## 4.6 Conclusion

In this chapter, we analyzed the convergence of global trust given by (4.1). We have shown that in recursive calculation of global trust, error will decrease even if initial guess is very far off from the actual solution. In any step, convergence is faster if error

TABLE 4.1: Center point in each iteration, when initial guess is close to center point

$i$	1	2	3	4
$\mathbf{t}^0$	1.0000	2.0000	3.0000	4.0000
$\mathbf{t}^1$	6.2000	4.8750	5.3333	3.6667
$\mathbf{t}^2$	6.4327	5.1096	5.5598	4.4027
$\mathbf{t}^3$	6.4367	5.0706	5.5573	4.4008
$\mathbf{t}^4$	6.4313	5.0705	5.5594	4.4002
$\mathbf{t}^5$	6.4310	5.0707	5.5592	4.3994
$\mathbf{t}^6$	6.4311	5.0708	5.5591	4.3994
$\mathbf{t}^7$	6.4311	5.0708	5.5591	4.3994

TABLE 4.2: Center point in each iteration, when initial guess is very far from center point

$i$	1	2	3	4
$\mathbf{t}^0$	100.0000	300.0000	200.0000	100.0000
$\mathbf{t}^1$	6.8000	5.2500	5.2500	4.1667
$\mathbf{t}^2$	6.5000	5.0668	5.5643	4.4827
$\mathbf{t}^3$	6.4298	5.0654	5.5620	4.4050
$\mathbf{t}^4$	6.4299	5.0706	5.5593	4.3987
$\mathbf{t}^5$	6.4310	5.0708	5.5591	4.3993
$\mathbf{t}^6$	6.4311	5.0708	5.5591	4.3994
$\mathbf{t}^7$	6.4311	5.0708	5.5591	4.3994

TABLE 4.3: Global trust in each iteration, when initial guess is close to global trust ( $\alpha = 1/3$ )

$i$	1	2	3	4
$\mathbf{t}^0$	1.0000	2.0000	3.0000	4.0000
$\mathbf{t}^1$	4.9893	4.4051	3.9876	3.2749
$\mathbf{t}^2$	5.8801	4.8299	5.3148	4.4838
$\mathbf{t}^3$	6.0621	5.1110	5.5131	4.6039
$\mathbf{t}^4$	6.1418	5.1543	5.5636	4.6490
$\mathbf{t}^5$	6.1550	5.1683	5.5802	4.6631
$\mathbf{t}^6$	6.1593	5.1717	5.5834	4.6657
$\mathbf{t}^7$	6.1603	5.1725	5.5844	4.6665
$\mathbf{t}^8$	6.1605	5.1727	5.5846	4.6667
$\mathbf{t}^9$	6.1606	5.1728	5.5847	4.6667
$\mathbf{t}^{10}$	6.1606	5.1728	5.5847	4.6667

TABLE 4.4: Global trust in each iteration, when initial guess is very far from global trust ( $\alpha = 1/3$ )

$i$	1	2	3	4
$\mathbf{t}^0$	100.0000	300.0000	200.0000	100.0000
$\mathbf{t}^1$	16.9093	12.1379	13.7913	11.3982
$\mathbf{t}^2$	7.6469	6.5685	7.1369	5.9630
$\mathbf{t}^3$	6.5419	5.4824	5.9029	4.9291
$\mathbf{t}^4$	6.2499	5.2477	5.6685	4.7377
$\mathbf{t}^5$	6.1829	5.1918	5.6048	4.6834
$\mathbf{t}^6$	6.1662	5.1775	5.5897	4.6710
$\mathbf{t}^7$	6.1620	5.1740	5.5859	4.6678
$\mathbf{t}^8$	6.1610	5.1731	5.5850	4.6670
$\mathbf{t}^9$	6.1607	5.1729	5.5847	4.6668
$\mathbf{t}^{10}$	6.1606	5.1728	5.5847	4.6667
$\mathbf{t}^{11}$	6.1606	5.1728	5.5847	4.6667

TABLE 4.5: Global trust in each iteration for lesser value of  $\alpha$  ( $\alpha = 1/6$ )

$i$	1	2	3	4
$\mathbf{t}^0$	1.0000	2.0000	3.0000	4.0000
$\mathbf{t}^1$	5.4761	4.6006	4.5168	3.4374
$\mathbf{t}^2$	6.1901	5.0137	5.4742	4.5092
$\mathbf{t}^3$	6.2506	5.1176	5.5682	4.5424
$\mathbf{t}^4$	6.2693	5.1288	5.5767	4.5486
$\mathbf{t}^5$	6.2714	5.1304	5.5790	4.5510
$\mathbf{t}^6$	6.2716	5.1307	5.5793	4.5511
$\mathbf{t}^7$	6.2717	5.1307	5.5793	4.5512
$\mathbf{t}^8$	6.2717	5.1307	5.5793	4.5512

---

is larger. We have shown that speed of convergence depends upon the value of  $\alpha$ . For smaller  $\alpha$  it will converge faster.

## Chapter 5

# Biased Contribution Index: A Distributed Mechanism to Ensure Fairness in P2P Networks

### 5.1 Introduction

Unfairness, a large difference between uploads and downloads at any peer, is a fundamental problem in peer-to-peer (P2P) networks. Motivation for resource sharing in peers can be achieved by a proper incentive mechanism. In this chapter, we are proposing a mechanism to rank the peers based on their resource contributions. Contributions of peers are biased in such a way that the amount of uploads and downloads at each peer get balanced naturally. Faster convergence of this mechanism makes it simple and light-weight to implement in a distributed system. The results indicate that the proposed algorithm does achieve the objective of fairness in the network.

## 5.2 Motivation

The diversity of data is a primary feature of P2P networks. But its availability depends on the choice of peers if they want to share it. Mostly, peers would like to get more resources while trying to share less. But in such a scenario, no one will get the resources. The peers are motivated to share the resources if they are also getting it from the network. Ideally, in a P2P network, each peer should get the same amount what it is sharing, in long-term average. This is a fair situation and a mechanism to ensure it is needed.

In literature, many mechanisms have been studied for aforementioned purpose [5], [6], [7], [8], [9], [10]. Among these, global approaches [6], [7], [8] are expected to perform better compared to localized approaches [5], [9], [10]. Because in global approaches, shared history of peers in the entire network is taken into consideration, which gives a wider view of a peers' cooperative behavior. But global approaches require large storage and bandwidth which makes its implementation complex. Therefore, the light-weight algorithm is required to make it simple.

In this chapter, we are proposing a simple, light-weight global approach called biased contribution index (BCI). It is expected to fulfill the desired objective. Rest of the chapter is organized as follows: Section 5.3 presents the network model and introduction of biased contribution index. Solution of biased contribution index is given in Section 5.4. Analysis of proposed algorithm is given in Section 5.5. Section 5.6 shows the numerical results, and in Section 5.7, conclusion is presented.



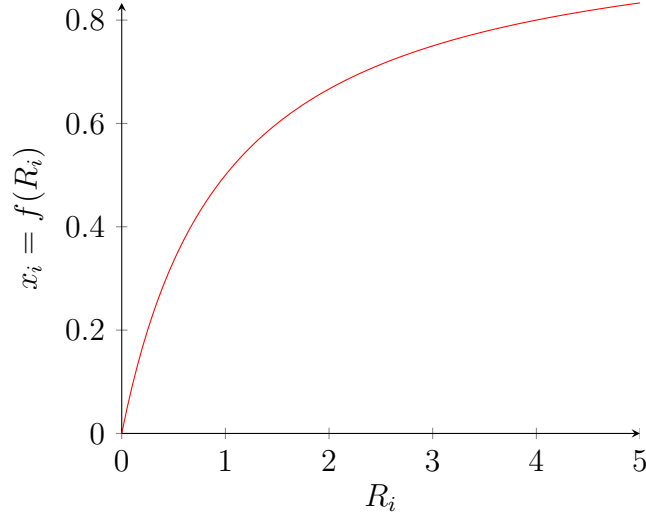


Fig. 5.1: Incentive factor  $x_i$  as a function of bias ratio  $R_i$

### 5.3 Network Model and Biased Contribution Index

In a P2P network, peers share their resources with each other and their contribution is evaluated globally, i.e., based on resources shared with all other peers. A simple metric, which can best reflect the contribution of peers in the network, could be the ratio of its total upload to the network to the total download from it. But, to balance the upload and download amount in each peer, we need to motivate the peers to upload more to the peers who contribute more and to download more from the peers who contribute less. This can be done by biasing this ratio by some incentive factor. Let this incentive factor be,  $x_i$ , for peer  $i$ . We define, the biased upload to download ratio for this peer  $i$  as

$$R_i = \frac{\mathbf{e}_i \cdot \mathbf{S} \cdot \mathbf{x}}{\mathbf{e}_i \cdot \mathbf{S}^{\text{tr}} \cdot \mathbf{x}}. \quad (5.1)$$

Here,  $\mathbf{x}$  is a column vector containing incentive factors. We assume  $N$  peers in the network.  $\mathbf{S}$  is the  $N \times N$  share matrix; its  $ij^{\text{th}}$  element represents the amount of resource shared by peer  $i$  to peer  $j$ .  $\mathbf{S}^{\text{tr}}$  is transpose of matrix  $\mathbf{S}$ . Here,  $\mathbf{e}_i$  is the row vector with

its  $i^{th}$  entry as '1' and all other entries as zero. In order to give higher incentive to the peers who are contributing more, let us define the incentive factor,  $x_i$ , of peer  $i$  as a monotonically increasing function of biased ratio  $R_i$ , i.e.,

$$x_i = \frac{R_i}{1 + R_i} = \frac{\mathbf{e}_i \cdot \mathbf{S} \cdot \mathbf{x}}{\mathbf{e}_i \cdot \mathbf{S} \cdot \mathbf{x} + \mathbf{e}_i \cdot \mathbf{S}^{tr} \cdot \mathbf{x}}. \quad (5.2)$$

We call this incentive factor as the biased contribution index (BCI), as it is calculated using biased ratio.

Now to start the process of sharing, we need to give some initial value of BCI to all the peers. Let us define a parameter  $\alpha \in (0, 1)$  to decide the initial value of BCI. Later we will see that the parameter  $\alpha$  decides the speed of convergence. The BCI is modified to include this parameter  $\alpha$ , as given below.

$$x_i = \begin{cases} \alpha \frac{\mathbf{e}_i \cdot \mathbf{S} \cdot \mathbf{x}}{\mathbf{e}_i \cdot \mathbf{S} \cdot \mathbf{x} + \mathbf{e}_i \cdot \mathbf{S}^{tr} \cdot \mathbf{x}} + (1 - \alpha), & \text{if } \mathbf{e}_i \cdot \mathbf{S} \cdot \mathbf{x} + \mathbf{e}_i \cdot \mathbf{S}^{tr} \cdot \mathbf{x} \neq 0. \\ (1 - \alpha/2), & \text{otherwise.} \end{cases} \quad (5.3)$$

Here,  $(1 - \alpha/2)$  is the initial value of BCI, when neither upload nor download has happened at the node.

Peers are allowed to take the resources from the network only if their BCI is above a certain threshold value. Therefore, every peer will try to increase its BCI, so that it can get the required amount of resources whenever needed.

It can be observed easily from (5.3) that a peer's BCI will be higher if

- 1). Its contribution,  $S_{ij}$ , is higher,
- 2). It shares more of its resources with higher contributing peers (higher  $x_j$ ), and
- 3). It takes more of the services from lower contributing peers (lower  $x_j$ ).

Therefore, intuitively, we can say that this metric can assure fairness in the whole network. A mathematical justification for this is given in next section.

In (5.3), BCI of any peer is expressed in the terms of BCI of other peers. In the network of  $N$  nodes, there will be  $N$  unknown BCIs and  $N$  nonlinear equations. It will be difficult to say anything about the solution of these equations. In the next section, we will show that these equations can be solved by a suitable iterative function.

## 5.4 Solution of BCI and Justification For Fairness

### 5.4.1 Solution of BCI

If  $\mathbf{e}_i \cdot \mathbf{S} \cdot \mathbf{x} + \mathbf{e}_i \cdot \mathbf{S}^{\text{tr}} \cdot \mathbf{x} \neq 0$ , (5.3) can be written as

$$x_i = \frac{\mathbf{e}_i \cdot [\mathbf{S} + (1 - \alpha) \mathbf{S}^{\text{tr}}] \cdot \mathbf{x}}{\mathbf{e}_i \cdot [\mathbf{S} + \mathbf{S}^{\text{tr}}] \cdot \mathbf{x}} = \frac{\mathbf{e}_i \cdot \mathbf{S}' \cdot \mathbf{x}}{\mathbf{e}_i \cdot \mathbf{S}'' \cdot \mathbf{x}} \quad (5.4)$$

Here,  $\mathbf{S}' = \mathbf{S} + (1 - \alpha) \mathbf{S}^{\text{tr}}$  and  $\mathbf{S}'' = \mathbf{S} + \mathbf{S}^{\text{tr}}$ . Since  $i = 1, 2, \dots, N$ , this set of  $N$  equations can be written in matrix form as

$$\mathbf{x} = \mathbf{diag}[d_1, d_2, \dots, d_N] \cdot \mathbf{S}' \cdot \mathbf{x}$$

Here,  $\mathbf{diag}[\dots]$  is diagonal matrix with its  $ii^{\text{th}}$  element,  $d_i$ , as  $1/(\mathbf{e}_i \cdot \mathbf{S}'' \cdot \mathbf{x})$ . In order to solve the above equations, we propose the following Lemmas.

**Lemma 5.4.1.** *The BCI vector  $\mathbf{x} \in [(1 - \alpha), 1]^N$*

**Proof.** When any peer  $i$  only takes the resources from the network and does not contribute any thing, then  $\mathbf{e}_i \cdot \mathbf{S} \cdot \mathbf{x} = 0$  and  $\mathbf{e}_i \cdot \mathbf{S}^{\text{tr}} \cdot \mathbf{x} \neq 0$ . In this case, BCI,  $x_i$ , of peer  $i$  will be minimum and is given by,  $\alpha \cdot 0 + (1 - \alpha) = (1 - \alpha)$ .

When the peer  $i$  only contributes the resources to the network without taking any thing, then  $\mathbf{e}_i \cdot \mathbf{S} \cdot \mathbf{x} \neq 0$  and  $\mathbf{e}_i \cdot \mathbf{S}^{\text{tr}} \cdot \mathbf{x} = 0$ . In this case, BCI,  $x_i$ , of peer  $i$  will be maximum and is given by,  $\alpha \cdot 1 + (1 - \alpha) = 1$ .

In all other cases, it will be in between these values. Hence  $\mathbf{x} \in [(1 - \alpha), 1]^N$ .  $\square$

**Lemma 5.4.2.** *Let  $(\mathbf{S} + \mathbf{S}^{\text{tr}})$  be  $N \times N$  non negative, irreducible matrix, then the BCI vector  $\mathbf{x}$ , in the above expression can be calculated by an iterative function*

$$\mathbf{x}^k = \phi(\mathbf{x}^{k-1}),$$

where  $i^{\text{th}}$  element of iterative function  $\phi(\mathbf{x}^{k-1})$  is  $[\mathbf{e}_i \cdot \mathbf{S}' \cdot \mathbf{x}^{k-1} / (\mathbf{e}_i \cdot \mathbf{S}'' \cdot \mathbf{x}^{k-1})]$ .

**Proof.** The  $i^{\text{th}}$  element of iterative function  $\phi(\mathbf{x}^{k-1})$  is

$$x_i^k = \frac{\mathbf{e}_i \cdot \mathbf{S}' \cdot \mathbf{x}^{k-1}}{\mathbf{e}_i \cdot \mathbf{S}'' \cdot \mathbf{x}^{k-1}}.$$

Let  $x_i^k$  and  $x_i^{k-1}$  are, far from actual solution  $x_i$  by  $\delta x_i^k$  and  $\delta x_i^{k-1}$  respectively, then

$$x_i + \delta x_i^k = \frac{\mathbf{e}_i \cdot \mathbf{S}' \cdot (\mathbf{x} + \delta \mathbf{x}^{k-1})}{\mathbf{e}_i \cdot \mathbf{S}'' \cdot (\mathbf{x} + \delta \mathbf{x}^{k-1})} = \frac{\mathbf{e}_i \cdot \mathbf{S}' \cdot \mathbf{x}}{\mathbf{e}_i \cdot \mathbf{S}'' \cdot \mathbf{x}} \left[ \frac{1 + \frac{\mathbf{e}_i \cdot \mathbf{S}' \cdot \delta \mathbf{x}^{k-1}}{\mathbf{e}_i \cdot \mathbf{S}' \cdot \mathbf{x}}}{1 + \frac{\mathbf{e}_i \cdot \mathbf{S}'' \cdot \delta \mathbf{x}^{k-1}}{\mathbf{e}_i \cdot \mathbf{S}'' \cdot \mathbf{x}}} \right].$$

Using (5.4),

$$x_i + \delta x_i^k = x_i \left[ \frac{1 + \frac{\mathbf{e}_i \cdot \mathbf{S}' \cdot \delta \mathbf{x}^{k-1}}{\mathbf{e}_i \cdot \mathbf{S}' \cdot \mathbf{x}}}{1 + \frac{\mathbf{e}_i \cdot \mathbf{S}'' \cdot \delta \mathbf{x}^{k-1}}{\mathbf{e}_i \cdot \mathbf{S}'' \cdot \mathbf{x}}} \right]$$

or

$$\begin{aligned} \delta x_i^k &= x_i \left[ \frac{1 + \frac{\mathbf{e}_i \cdot \mathbf{S}' \cdot \delta \mathbf{x}^{k-1}}{\mathbf{e}_i \cdot \mathbf{S}' \cdot \mathbf{x}}}{1 + \frac{\mathbf{e}_i \cdot \mathbf{S}'' \cdot \delta \mathbf{x}^{k-1}}{\mathbf{e}_i \cdot \mathbf{S}'' \cdot \mathbf{x}}} - 1 \right] \\ &= \frac{1}{\left(1 + \frac{\mathbf{e}_i \cdot \mathbf{S}'' \cdot \delta \mathbf{x}^{k-1}}{\mathbf{e}_i \cdot \mathbf{S}'' \cdot \mathbf{x}}\right)} \left[ \left( \frac{x_i \mathbf{e}_i \cdot \mathbf{S}'}{\mathbf{e}_i \cdot \mathbf{S}' \cdot \mathbf{x}} \right) - \left( \frac{x_i \mathbf{e}_i \cdot \mathbf{S}''}{\mathbf{e}_i \cdot \mathbf{S}'' \cdot \mathbf{x}} \right) \right] \cdot \delta \mathbf{x}^{k-1} \\ &= f_i(\delta \mathbf{x}^{k-1}) \cdot [\mathbf{A}_i - \mathbf{B}_i] \cdot \delta \mathbf{x}^{k-1}, \end{aligned}$$

where  $f_i(\delta \mathbf{x}^{k-1})$  is  $1 / \left(1 + \frac{\mathbf{e}_i \cdot \mathbf{S}'' \cdot \delta \mathbf{x}^{k-1}}{\mathbf{e}_i \cdot \mathbf{S}'' \cdot \mathbf{x}}\right)$ . Here,  $\mathbf{A}_i$  and  $\mathbf{B}_i$  are  $i^{\text{th}}$  row of  $N \times N$  matrices  $\mathbf{A}$  and  $\mathbf{B}$  respectively. It can be observed about matrices  $\mathbf{A}$  and  $\mathbf{B}$  that,  $\mathbf{A} \cdot \mathbf{x} = \mathbf{x}$  and  $\mathbf{B} \cdot \mathbf{x} = \mathbf{x}$ .

The matrices  $\mathbf{A}$  and  $\mathbf{B}$  are derived from irreducible matrices  $\mathbf{S}'$  and  $\mathbf{S}''$  respectively, hence matrices  $\mathbf{A}$  and  $\mathbf{B}$  will also be irreducible. Elements of vector  $\mathbf{x}$  are positive (see Lemma 5.4.1), so for non negative matrix  $\mathbf{S}$ , matrices  $\mathbf{A}$  and  $\mathbf{B}$  will also be non negative. Therefore, spectral radius of matrices  $\mathbf{A}$  and  $\mathbf{B}$  will be '1' and corresponding eigenvector will be  $\mathbf{x}$  (see [61]).

If  $\delta\mathbf{x}^{k-1} \ll \mathbf{x}$ , then  $f_i(\delta\mathbf{x}^{k-1}) \approx 1$ . Hence,

$$\delta\mathbf{x}^k = [\mathbf{A} - \mathbf{B}].\delta\mathbf{x}^{k-1} = [\mathbf{A} - \mathbf{B}]^k.\delta\mathbf{x}^0$$

$$\lim_{k \rightarrow \infty} \delta\mathbf{x}^k = \lim_{k \rightarrow \infty} [\mathbf{A} - \mathbf{B}]^k.\delta\mathbf{x}^0 = \mathbf{0}$$

(see Theorem 3.4.1 in Chapter 3), and if  $\delta\mathbf{x}^{k-1} > \mathbf{x}$ , then  $f_i(\delta\mathbf{x}^{k-1}) < 1$ . Hence in this case,  $\delta x_i^{k-1}$  will decrease more rapidly till  $\delta\mathbf{x}^{k-1} \ll \mathbf{x}$ .

Hence, by the aforementioned iterative function, we will finally converge to  $\mathbf{x}$ .  $\square$

To understand the calculation of BCI through example, consider the share matrix,  $\mathbf{S}$

$$\begin{bmatrix} 0 & 100 & 50 & 20 \\ 20 & 0 & 30 & 40 \\ 10 & 40 & 0 & 50 \\ 50 & 10 & 60 & 0 \end{bmatrix}.$$

The number of iterations required to converge the BCI were estimated for two different values of  $\alpha$ . For  $\alpha = 0.8$ , BCI in each step is shown in Table 5.1. We can see that it converges in seven iterations. For  $\alpha = 0.4$  (see Table 5.2), it converges only in five iterations. Thus, the impact of  $\alpha$  is clearly evident. Smaller the  $\alpha$ , faster the convergence of the estimate.

TABLE 5.1: BCI for  $\alpha = 0.8$  in each iteration

$i$	1	2	3	4
$\mathbf{x}^0$	0.6000	0.6000	0.6000	0.6000
$\mathbf{x}^1$	0.7440	0.5000	0.5333	0.6174
$\mathbf{x}^2$	0.7266	0.4823	0.5161	0.6373
$\mathbf{x}^3$	0.7202	0.4861	0.5170	0.6379
$\mathbf{x}^4$	0.7207	0.4870	0.5177	0.6371
$\mathbf{x}^5$	0.7210	0.4869	0.5177	0.6370
$\mathbf{x}^6$	0.7210	0.4868	0.5177	0.6370
$\mathbf{x}^7$	0.7210	0.4868	0.5177	0.6370

TABLE 5.2: BCI for  $\alpha = 0.4$  in each iteration

$i$	1	2	3	4
$\mathbf{x}^0$	0.8000	0.8000	0.8000	0.8000
$\mathbf{x}^1$	0.8720	0.7500	0.7667	0.8087
$\mathbf{x}^2$	0.8690	0.7465	0.7634	0.8124
$\mathbf{x}^3$	0.8685	0.7468	0.7634	0.8124
$\mathbf{x}^4$	0.8686	0.7468	0.7635	0.8124
$\mathbf{x}^5$	0.8686	0.7468	0.7635	0.8124

Justification of faster convergence of BCI for smaller values of  $\alpha$ , can be given from the proof of Lemma 5.4.2. Speed of convergence depends upon the matrix  $(\mathbf{A} - \mathbf{B})$ . Lesser the difference between elements of matrices  $\mathbf{A}$  and  $\mathbf{B}$  higher will be the speed of convergence. Matrices  $\mathbf{A}$  and  $\mathbf{B}$  are derived from matrices  $\mathbf{S}'$  and  $\mathbf{S}''$ , therefore, the same thing is true for matrices  $\mathbf{S}'$  and  $\mathbf{S}''$ . For  $\alpha = 0$ ,  $\mathbf{S}' = \mathbf{S}''$  hence it converges in single step. As  $\alpha$  increases difference between matrices  $\mathbf{S}'$  and  $\mathbf{S}''$  increases and thus, the difference between matrices  $\mathbf{A}$  and  $\mathbf{B}$  also increases, as a result, the speed of convergence increases.

TABLE 5.3: Number of iterations, required to converge the BCI and GC [8] for different values of  $\alpha$  and  $\beta$ .

	Number of iteration required in GC[8]	Number of iteration required in BCI
$\alpha = 0.9$	$\beta = 0.8, \text{Iterations} = 9$	$\text{Iterations} = 8$
	$\beta = 0.5, \text{Iterations} = 10$	
	$\beta = 0.2, \text{Iterations} = 10$	
$\alpha = 0.8$	$\beta = 0.8, \text{Iterations} = 10$	$\text{Iterations} = 7$
	$\beta = 0.5, \text{Iterations} = 8$	
	$\beta = 0.2, \text{Iterations} = 11$	
$\alpha = 0.7$	$\beta = 0.8, \text{Iterations} = 9$	$\text{Iterations} = 7$
	$\beta = 0.5, \text{Iterations} = 8$	
	$\beta = 0.2, \text{Iterations} = 9$	
$\alpha = 0.6$	$\beta = 0.8, \text{Iterations} = 8$	$\text{Iterations} = 6$
	$\beta = 0.5, \text{Iterations} = 7$	
	$\beta = 0.2, \text{Iterations} = 8$	
$\alpha = 0.5$	$\beta = 0.8, \text{Iterations} = 7$	$\text{Iterations} = 5$
	$\beta = 0.5, \text{Iterations} = 6$	
	$\beta = 0.2, \text{Iterations} = 8$	
$\alpha = 0.4$	$\beta = 0.8, \text{Iterations} = 7$	$\text{Iterations} = 5$
	$\beta = 0.5, \text{Iterations} = 6$	
	$\beta = 0.2, \text{Iterations} = 7$	
$\alpha = 0.3$	$\beta = 0.8, \text{Iterations} = 6$	$\text{Iterations} = 4$
	$\beta = 0.5, \text{Iterations} = 5$	
	$\beta = 0.2, \text{Iterations} = 6$	
$\alpha = 0.2$	$\beta = 0.8, \text{Iterations} = 5$	$\text{Iterations} = 3$
	$\beta = 0.5, \text{Iterations} = 5$	
	$\beta = 0.2, \text{Iterations} = 6$	

### 5.4.2 Comparison of BCI With Global Contribution Approach

Similar research was done in [8], where GC is expressed as first order polynomials of other GC. A Coefficient matrix of these linear equations is made diagonally dominant, as a result, it can be solved by iterative methods such as the Jacobi, Gauss-Seidel methods.

We compared the number of iterations required for convergence of the BCI with that of global contribution [8]. For fair comparison, we considered the same share matrix,  $\mathbf{S}$ , given above, for both BCI and GC [8]. In latter case, we have taken the different values of  $\alpha$  and  $\beta$  as defined in [8]. Results are shown in Table 5.3. We can observe that the number of iterations required for convergence of BCI is always lesser than that of the GC [8].

### 5.4.3 Justification For Fairness

**Lemma 5.4.3.** *If BCI of all the peers is same, then the amount of resources contributed will be same as what is taken from the network for each peer.*

**Proof.** Let  $\mathbf{e}$  be column vector with each element as '1' and  $a$  be a scalar, then the BCI vector,  $\mathbf{x}$ , can be written as  $a\mathbf{e}$ . Now, if  $\mathbf{e}_i.\mathbf{S}.\mathbf{x} + \mathbf{e}_i.\mathbf{S}^{\text{tr}}.\mathbf{x} \neq 0$ , the (5.3) can be rearranged as

$$x_i(\mathbf{e}_i.\mathbf{S}.\mathbf{x} + \mathbf{e}_i.\mathbf{S}^{\text{tr}}.\mathbf{x}) = \mathbf{e}_i.\mathbf{S}.\mathbf{x} + (1 - \alpha)\mathbf{e}_i.\mathbf{S}^{\text{tr}}.\mathbf{x} \quad \forall i.$$

The above relation can be written in the form of a matrix as follows.

$$\text{diag}(\mathbf{x}).(\mathbf{S} + \mathbf{S}^{\text{tr}}).\mathbf{x} = \mathbf{S}.\mathbf{x} + (1 - \alpha)\mathbf{S}^{\text{tr}}.\mathbf{x}.$$



Here  $\mathbf{diag}(\mathbf{x})$  is  $N \times N$  diagonal matrix with its  $ii^{th}$  element as  $x_i$ . Now if  $\mathbf{x} = a\mathbf{e}$ , then

$$a\mathbf{I}(\mathbf{S} + \mathbf{S}^{\text{tr}}).a\mathbf{e} = a\mathbf{S}.\mathbf{e} + (1 - \alpha)a\mathbf{S}^{\text{tr}}.\mathbf{e},$$

$$\Rightarrow a^2(\mathbf{S} + \mathbf{S}^{\text{tr}}).\mathbf{e} = a(\mathbf{S}.\mathbf{e} + (1 - \alpha)\mathbf{S}^{\text{tr}}.\mathbf{e}).$$

Pre-multiplying by  $\mathbf{e}^{\text{tr}}$  on both sides,

$$a^2\mathbf{e}^{\text{tr}}.(\mathbf{S} + \mathbf{S}^{\text{tr}}).\mathbf{e} = a(\mathbf{e}^{\text{tr}}.\mathbf{S}.\mathbf{e} + (1 - \alpha)\mathbf{e}^{\text{tr}}.\mathbf{S}^{\text{tr}}.\mathbf{e}),$$

$$\Rightarrow a^2(\mathbf{e}^{\text{tr}}.\mathbf{S}.\mathbf{e} + \mathbf{e}^{\text{tr}}.\mathbf{S}^{\text{tr}}.\mathbf{e}) = a(\mathbf{e}^{\text{tr}}.\mathbf{S}.\mathbf{e} + (1 - \alpha)\mathbf{e}^{\text{tr}}.\mathbf{S}^{\text{tr}}.\mathbf{e}).$$

For any matrix  $\mathbf{S}$ ,  $\mathbf{e}^{\text{tr}}.\mathbf{S}.\mathbf{e}$  will be the sum of all of its elements. Hence,  $\mathbf{e}^{\text{tr}}.\mathbf{S}.\mathbf{e} = \mathbf{e}^{\text{tr}}.\mathbf{S}^{\text{tr}}.\mathbf{e} = T$ , and above expression can be written as

$$a^2(T + T) = a(T + (1 - \alpha)T).$$

Since  $a \in [(1 - \alpha), 1]$  and  $T \neq 0$ , hence  $a = (1 - \alpha/2)$ . Now, substitute  $\mathbf{x} = (1 - \alpha/2)\mathbf{e}$  in (5.3)

$$(1 - \alpha/2) = \alpha \frac{(1 - \alpha/2)\mathbf{e}_i.\mathbf{S}.\mathbf{e}}{(1 - \alpha/2)(\mathbf{e}_i.\mathbf{S}.\mathbf{e} + \mathbf{e}_i.\mathbf{S}^{\text{tr}}.\mathbf{e})} + (1 - \alpha)$$

$$\Rightarrow \alpha/2 = \alpha \frac{\mathbf{e}_i.\mathbf{S}.\mathbf{e}}{(\mathbf{e}_i.\mathbf{S}.\mathbf{e} + \mathbf{e}_i.\mathbf{S}^{\text{tr}}.\mathbf{e})}.$$

Since  $\alpha \neq 0$ , hence,  $\mathbf{e}_i.\mathbf{S}.\mathbf{e} = \mathbf{e}_i.\mathbf{S}^{\text{tr}}.\mathbf{e}$  for each peer  $i$ . □

**Lemma 5.4.4.** *If resources contributed and resources taken from the network in each peer are same, then the BCI of all peers will be same.*

**Proof.** If  $\mathbf{e}_i.\mathbf{S}.\mathbf{e} = \mathbf{e}_i.\mathbf{S}^{\text{tr}}.\mathbf{e}$  for each peer  $i$ . Then,  $\mathbf{S}.\mathbf{e} = \mathbf{S}^{\text{tr}}.\mathbf{e} \Rightarrow \mathbf{e}^{\text{tr}}.\mathbf{S}^{\text{tr}} = \mathbf{e}^{\text{tr}}.\mathbf{S}$ . Now, again if  $\mathbf{e}_i.\mathbf{S}.\mathbf{x} + \mathbf{e}_i.\mathbf{S}^{\text{tr}}.\mathbf{x} \neq 0$ , then

$$\mathbf{diag}(\mathbf{x}).(\mathbf{S} + \mathbf{S}^{\text{tr}}).\mathbf{x} = \mathbf{S}.\mathbf{x} + (1 - \alpha)\mathbf{S}^{\text{tr}}.\mathbf{x}.$$

Pre-multiplying by  $\mathbf{e}^{\text{tr}}$  on both sides,

$$\begin{aligned}
\mathbf{e}^{\text{tr}} \cdot \text{diag}(\mathbf{x}) \cdot (\mathbf{S} + \mathbf{S}^{\text{tr}}) \cdot \mathbf{x} &= \mathbf{e}^{\text{tr}} \cdot \mathbf{S} \cdot \mathbf{x} + (1 - \alpha) \mathbf{e}^{\text{tr}} \cdot \mathbf{S}^{\text{tr}} \cdot \mathbf{x} \\
\Rightarrow \mathbf{x}^{\text{tr}} \cdot (\mathbf{S} + \mathbf{S}^{\text{tr}}) \cdot \mathbf{x} &= \mathbf{e}^{\text{tr}} \cdot \mathbf{S} \cdot \mathbf{x} - (\alpha/2) \mathbf{e}^{\text{tr}} \cdot \mathbf{S}^{\text{tr}} \cdot \mathbf{x} + (1 - \alpha/2) \mathbf{e}^{\text{tr}} \cdot \mathbf{S}^{\text{tr}} \cdot \mathbf{x} \\
\Rightarrow \mathbf{x}^{\text{tr}} \cdot (\mathbf{S} + \mathbf{S}^{\text{tr}}) \cdot \mathbf{x} &= \mathbf{e}^{\text{tr}} \cdot \mathbf{S} \cdot \mathbf{x} - (\alpha/2) \mathbf{e}^{\text{tr}} \cdot \mathbf{S} \cdot \mathbf{x} + (1 - \alpha/2) \mathbf{e}^{\text{tr}} \cdot \mathbf{S}^{\text{tr}} \cdot \mathbf{x} \\
\Rightarrow \mathbf{x}^{\text{tr}} \cdot (\mathbf{S} + \mathbf{S}^{\text{tr}}) \cdot \mathbf{x} &= (1 - \alpha/2) \mathbf{e}^{\text{tr}} \cdot \mathbf{S} \cdot \mathbf{x} + (1 - \alpha/2) \mathbf{e}^{\text{tr}} \cdot \mathbf{S}^{\text{tr}} \cdot \mathbf{x} \\
\Rightarrow \mathbf{x}^{\text{tr}} \cdot (\mathbf{S} + \mathbf{S}^{\text{tr}}) \cdot \mathbf{x} &= (1 - \alpha/2) \mathbf{e}^{\text{tr}} \cdot (\mathbf{S} + \mathbf{S}^{\text{tr}}) \cdot \mathbf{x} \\
\Rightarrow [\mathbf{x}^{\text{tr}} - (1 - \alpha/2) \mathbf{e}^{\text{tr}}] \cdot (\mathbf{S} + \mathbf{S}^{\text{tr}}) \cdot \mathbf{x} &= 0
\end{aligned}$$

One can observe that  $(\mathbf{S} + \mathbf{S}^{\text{tr}})$  is non negative matrix and  $\mathbf{x} > \mathbf{0}$ , thus  $(\mathbf{S} + \mathbf{S}^{\text{tr}}) \cdot \mathbf{x} \neq \mathbf{0}$ .

Hence,  $\mathbf{x} = (1 - \alpha/2) \mathbf{e}$ . □

To understand the fair situation in network, consider another share matrix,  $\mathbf{S}$ , as follows

$$\begin{bmatrix} 0 & 20 & 30 & 0 \\ 40 & 0 & 30 & 0 \\ 0 & 20 & 0 & 80 \\ 10 & 30 & 40 & 0 \end{bmatrix} .$$

We can see in this example that sum of total upload and download for all the peers is different, i.e., 100 for peer 1, 140 for peer 2, 200 for peer 3 and 160 for peer 4. But total upload at each peer is same as total download, as a result, BCI of each peer is same. For  $\alpha = 0.8$ , it is 0.6 for each peer.

## 5.5 Analysis of BCI

### 5.5.1 Solution Of Free-riding

To start the process of sharing, initially, each peer is allowed to take some resources from the network. Therefore, initial BCI of  $(1 - \alpha/2)$  for each peer is justified. But

as soon as BCI is updated, free-rider's BCI will reach at minimum level. As for any free-rider  $i$ ,  $\mathbf{e}_i \cdot \mathbf{S} \cdot \mathbf{x} = 0$ , hence from (5.3),  $x_i = (1 - \alpha)$ . This will disqualify them from taking any resources from the network in the future, until they acquire sufficient BCI.

### 5.5.2 Implementation In Distributed System and Time Complexity

Distributed estimation of BCI can be implemented following the same approach as in [1], [8], [67]. Maintaining the record of uploads, downloads and calculation of BCI of a peer, can be assigned to any other peer, named index manager. The index manager can be located using distributed hash table (DHT) such as, Chord [19], CAN [20], Pastry [21] or Tapestry [22]. Whenever any peer needs the BCI of other peers, it can send the query to the respective index manager. For more secure and accurate estimation of BCI, more than one index managers can be configured. If there is any conflict about the BCI of a peer, it can be settled by majority vote of index managers. In this way we can also avoid the collusion among peers.

Each index manager can update the BCI as explained in algorithm 5.1. For calculation of BCI of any peer, index manager needs to know the resource contribution and consumption by that peer and the BCI of peers with whom it has transacted. The calculation is iterated and it converges in fewer iterations. If the number of iterations required to converge the algorithm are less, then the required number of update messages per unit time will also be less. Therefore, the algorithm can be implemented with lesser overhead.

Our simulation result shows that the number of iterations required to converge the BCI remain same irrespective of the order of the matrix. Therefore, we can say that

the time complexity of the algorithm is  $O(N)$  per peer.

---

**Algorithm 5.1** For Updating the BCI of Peers

---

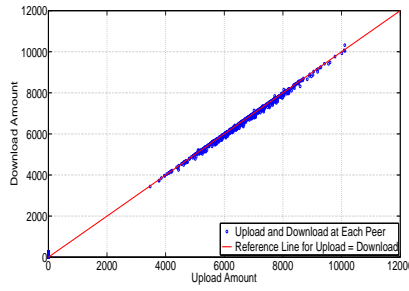
```

1: Input: Amount of upload and download of peers
2: Output: BCI with index managers
3: procedure
4:   for each peer  $i$  do
5:     forall peer  $j$ , who is selected as source peer do
6:       Download the resource
7:       Send the value of resource to the index manager of peer  $j$ ;
8:     end forall
9:     forall peer  $j$ , who selected peer  $i$  as source peer do
10:      Upload the resource
11:      Send the value of resource to the index manager of peer  $j$ ;
12:    end forall
13:    if Peer  $i$  is index manager of peer  $k$  then
14:      forall peer  $j$ , who transacted with peer  $k$  do
15:        Receive the value of resource uploaded  $S_{kj}$ ;
16:        Receive the value of resource downloaded  $S_{jk}$ ;
17:        Locate  $j^{th}$  peer's index manager;
18:      end forall
19:      \\ Initialization of parameters
20:      Set  $\alpha$ ,  $previous\_x_k$ ,  $threshold$ ,  $\mathbf{x}(\mathbf{0}) = (\alpha/(1 - \alpha))\mathbf{e}$ ,  $error$ ;
21:      while  $error \geq threshold$  do
22:        Receive the current BCI  $x_j$  from their index manager peer;
23:        Compute
24:        
$$x_k \leftarrow \alpha \left( \frac{\mathbf{e}_k \cdot \mathbf{S} \cdot \mathbf{x}}{\mathbf{e}_k \cdot \mathbf{S} \cdot \mathbf{x} + \mathbf{e}_k \cdot \mathbf{S}^{tr} \cdot \mathbf{x}} \right) + (1 - \alpha)$$

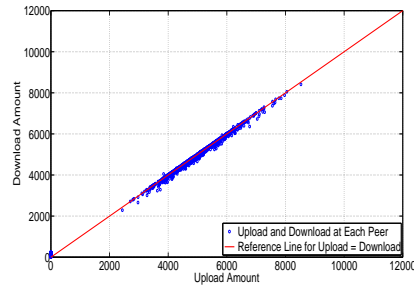
25:         $error \leftarrow |x_k - previous\_x_k|$ 
26:         $previous\_x_k \leftarrow x_k$ 
27:      end while
28:    end if
29:  end for
30: end procedure

```

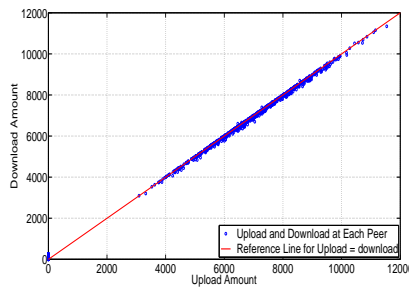
---



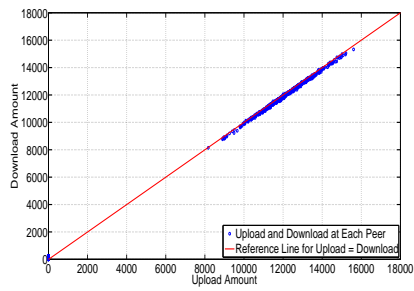
(a) GC,  $\alpha = 0.8, \beta = 0.2$ , Free-riders = 30%



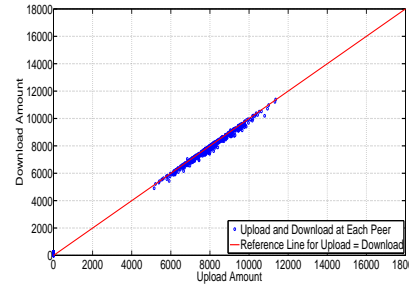
(b) BCI,  $\alpha = 0.8$ , Free-riders = 30%



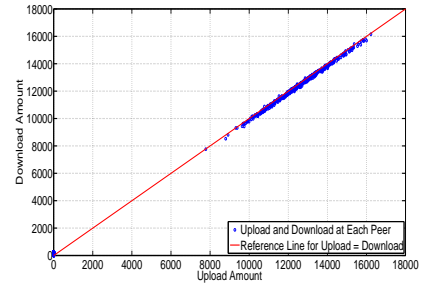
(c) BCI,  $\alpha = 0.4$ , Free-riders = 30%



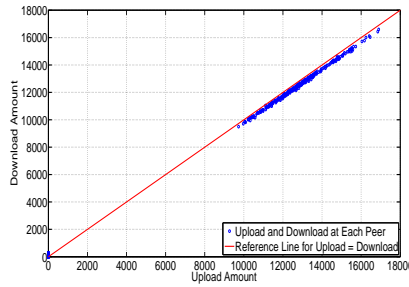
(d) GC,  $\alpha = 0.8, \beta = 0.2$ , Free-riders = 50%



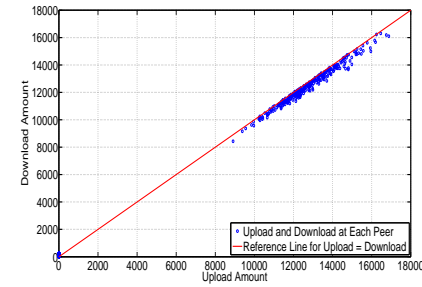
(e) BCI,  $\alpha = 0.8$ , Free-riders = 50%



(f) BCI,  $\alpha = 0.4$ , Free-riders = 50%



(g) GC,  $\alpha = 0.8, \beta = 0.2$ , Free-riders = 70%



(h) BCI,  $\alpha = 0.8$ , Free-riders = 70%

Fig. 5.2: Upload and Download Amount at Each Peer for Simple Model.

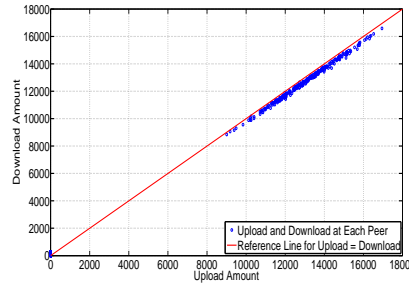
(i) BCI,  $\alpha = 0.4$ , Free-riders = 70%

Fig. 5.2: Upload and Download Amount at Each Peer for Simple Model.

## 5.6 Simulation Results

We simulated a large network with 1000 peers in NetLogo 5.2, [64], for three different peer distribution models, i.e., Simple Model, Adaptive Model and Extreme Model.

In Simple Model, a certain percentage of peers are considered to be free-riders and they do not share anything at any point of time in the simulation and rest behave as normal peers. The free-riders are taken as 30%, 50% and 70% in simulation.

In Adaptive Model, there are few adaptive free-riding peers. Half of these free-riding peers do not share anything during the whole simulation. Remaining half behave as normal peers till midway of simulation, and thereafter convert themselves to free-riders. In simulation, these free-riders are taken as 40% and 60%.

In Extreme Model, at the beginning of simulation, 10% peers are free-riders. After completion of every 12.5% of total transactions, 10% more peers convert to free-riders. Thus, at the end of simulation, there will be 80% free-riders.

The simulation is performed till completion of 100000 transactions. Downloader peer is selected randomly, and it requests for random value of data between 1 to 255 units.

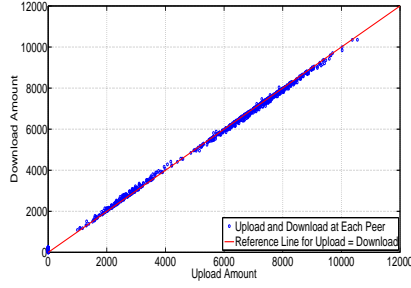
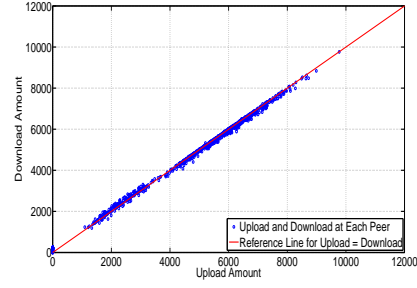
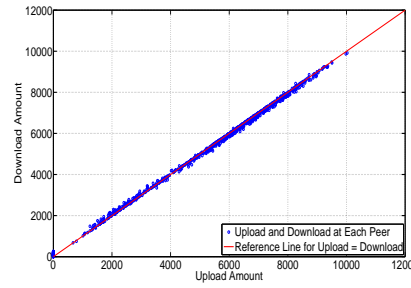
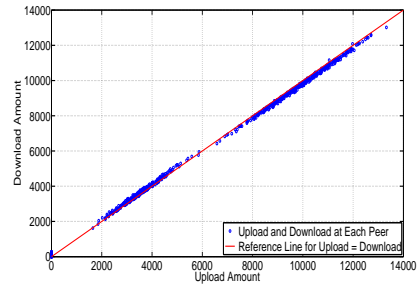
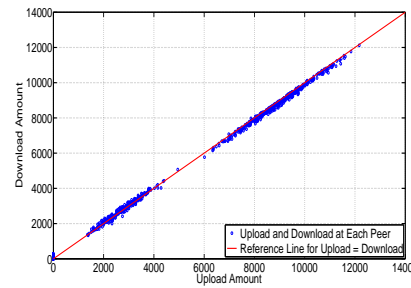
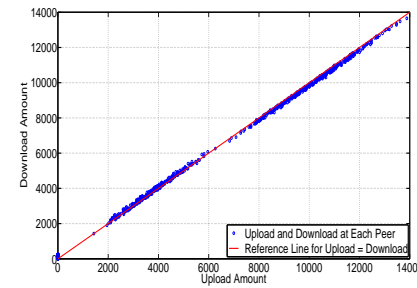
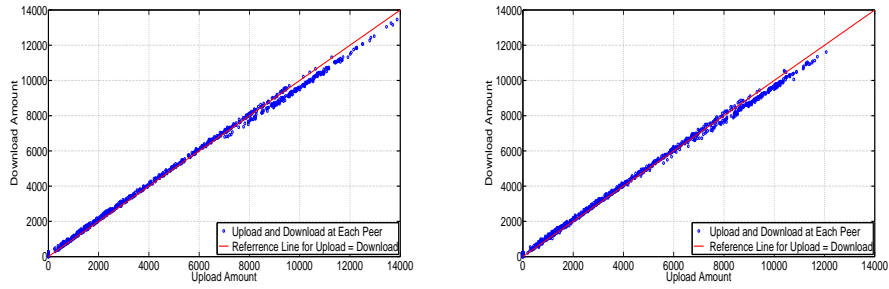
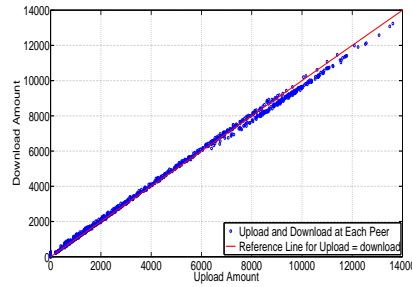
(a) GC,  $\alpha = 0.8, \beta = 0.2$ , Free-riders = 40%(b) BCI,  $\alpha = 0.8$ , Free-riders = 40%(c) BCI,  $\alpha = 0.4$ , Free-riders = 40%(d) GC,  $\alpha = 0.8, \beta = 0.2$ , Free-riders = 60%(e) BCI,  $\alpha = 0.8$ , Free-riders = 60%(f) BCI,  $\alpha = 0.4$ , Free-riders = 60%

Fig. 5.3: Upload and Download Amount at Each Peer for Adaptive Model. Half of the Free Riders do not share anything at all and rest stopped sharing in the middle of session.



(a) GC,  $\alpha = 0.8, \beta = 0.2$ , Free-riders = 80%      (b) BCI,  $\alpha = 0.8$ , Free-riders = 80%



(c) BCI,  $\alpha = 0.4$ , Free-riders = 80%

Fig. 5.4: Upload and Download Amount at Each Peer for Extreme Model. At the beginning of simulation, 10% peers are free-riders. After completion of every 12.5% of total transactions, 10% more peers convert to free-riders.



For any query, we expect 10% of the total node population to respond for providing the resources. Uploader always prefer uploading to high BCI peer and downloader always prefer to download from a low BCI peer.

We compared the performance of BCI with the best case of GC [8], i.e.,  $\alpha = 0.8, \beta = 0.2$ . The upload and download amount at each peer for different mechanisms in Simple Model, Adaptive Model and Extreme Model are shown in Fig. 5.2, Fig. 5.3 and in Fig. 5.4 respectively. We can see from these figures that each mechanism is able to balance the upload and download amount at each peer.

To get, the deeper picture of inside, we define a metric, average absolute deviation (AAD) of upload to download ratio from one as:

$$AAD = (1/N) \sum_{i=1}^N |1 - \mathbf{e}_i \cdot \mathbf{S} \cdot \mathbf{e} / \mathbf{e}_i \cdot \mathbf{S}^{\text{tr}} \cdot \mathbf{e}|.$$

Ideally, total upload at each peer must be same as total download thus,  $AAD = 0$ . Therefore, lesser the  $AAD$ , better will be the performance of mechanism. The  $AAD$  corresponding to above figures are shown in Table 5.4. We can see from this table that the BCI for  $\alpha = 0.4$  performs slightly better in Adaptive Model and in Extreme Model.

## 5.7 Conclusion

In this chapter, we proposed a new mechanism, the biased contribution index (BCI), to maintain the balance between total upload and download by a peer in the network. With the help of mathematical justification and simulation results, we have shown that the mechanism is able to achieve this objective. We compared our algorithm with another existing approach [8]. With the help of a numerical example, we have shown

TABLE 5.4: AAD for Different Mechanisms and Models

S.N.	Mechanism	Model	Free-riders	AAD
1	GC, Best Case	Simple	30%	0.3059
2	BCI, $\alpha = 0.8$	Simple	30%	0.3100
3	BCI, $\alpha = 0.4$	Simple	30%	0.3057
4	GC, Best Case	Simple	50%	0.5053
5	BCI, $\alpha = 0.8$	Simple	50%	0.5085
6	BCI, $\alpha = 0.4$	Simple	50%	0.5052
7	GC, Best Case	Simple	70%	0.7072
8	BCI, $\alpha = 0.8$	Simple	70%	0.7069
9	BCI, $\alpha = 0.4$	Simple	70%	0.7072
10	GC, Best Case	Adaptive	40%	0.2124
11	BCI, $\alpha = 0.8$	Adaptive	40%	0.2148
12	BCI, $\alpha = 0.4$	Adaptive	40%	0.2106
13	GC, Best Case	Adaptive	60%	0.3146
14	BCI, $\alpha = 0.8$	Adaptive	60%	0.3144
15	BCI, $\alpha = 0.4$	Adaptive	60%	0.3117
16	GC, Best Case	Extreme	80%	0.1370
17	BCI, $\alpha = 0.8$	Extreme	80%	0.1361
18	BCI, $\alpha = 0.4$	Extreme	80%	0.1320

that our algorithm converges in lesser number of iterations. It also performs better in Adaptive Model and in Extreme Model. This approach can also be implemented in a distributed system and is much simpler than the other existing approaches.

## Chapter 6

# Simplified Biased Contribution Index (SBCI) for Fair and Efficient P2P Network

### 6.1 Introduction

To balance the load and to discourage the free-riding in peer-to-peer (P2P) networks, many incentive mechanisms and policies have been proposed in recent years. Global peer ranking is one such mechanism. In this mechanism, peers are ranked based on a metric called contribution index. Contribution index is defined in such a manner that peers are motivated to share the resources in the network. Fairness in the terms of upload to download ratio in each peer can be achieved by this method. However, calculation of contribution index is not trivial. It is computed distributively and iteratively in the entire network and requires strict clock synchronization among the peers. A very small error in clock synchronization may lead to wrong results. Furthermore, iterative calculation requires a lot of message overhead and storage capacity, which makes its implementation more complex. In this chapter, we are proposing a simple incentive mechanism based on the contributions of peers, which can balance the upload

and download amount of resources in each peer. It does not require iterative calculation, therefore, can be implemented with lesser message overhead and storage capacity without requiring strict clock synchronization. This approach is efficient as there are very less rejections among the cooperative peers. It can be implemented in a truly distributed fashion with  $O(N)$  time complexity per peer.

## 6.2 Motivation

Peer-to-peer (P2P) networks gained a significant popularity in the last decade and are now responsible for a large fraction of internet traffic [18], [68]. The popularity of these networks is due to their inherent advantages over traditional client-server model, e.g., the diversity of available data, scalability, robustness and cost effectiveness. The initial setup cost for these networks is very small because costly central servers are not needed. However, lack of central control leads to the problem of unfairness in these networks, i.e., large difference between upload and download amount at any peer. In such a situation, many peers free-ride and contribute very less or nothing which results in slow downloads for other peers [26]. Therefore, designing and implementing an efficient incentive policy to motivate the peers to share the resources becomes important.

In recent years, many incentive policies have been proposed to maintain the fairness in P2P networks [5], [6], [7], [8], [10], [69]. In these policies, peers' cooperative behavior in the network is evaluated and resources are given to them in proportion to their cooperation.

In [5], [7], [10], peers' cooperation is evaluated locally, i.e., peer cooperate with only those peers who had cooperated with them in the past. To start the process of sharing,

a small amount of data is given to every peer. In such scenario, free-riders can always find a new peer to download their desired data. Also, the cooperative peers are not allowed to download more than this small amount of data from a new peer even though they have uploaded the large amount of data to some other peers [8].

In [6], [8], [69] peers' cooperative behavior in the entire network is taken into consideration. For this purpose, in [6], every peer keeps the record of each transaction which has happened in the entire network. It makes the implementation of algorithm very complex. In comparison to this, [8], [69] are simpler approaches. In these approaches, peers are ranked in the entire network. The rank of the peer is determined by the contribution index. It is estimated using two factors, resources contributed by the peer in the network and contribution index of peer with whom it is transacting. Estimation of contribution index is performed by iterative methods and can be implemented in a distributed fashion. These approaches are able to balance the amount of upload and download of resources in the network. However, there are some fundamental problems in its implementation.

First, in each iteration, index managers, i.e., peers who are managing the contribution index of other peers, need the current contribution index of peers from other peers. If clocks of the peers are not synchronized, then the peers who are reporting the contribution index of peers may report the contribution index of the previous iteration, which may lead to the wrong estimates [70].

Second, updating the contribution index in each iteration requires a lot of message overhead. This is more important when the number of iterations required to converge the algorithm is large. If new transactions happen in the network, then contribution index need to be updated. Even one transaction, between any two peers, can affect the

contribution index of all the peers in the network.

And lastly, index managers need to keep the record of all past transactions of a peer for whom they are estimating the contribution index. This needs a large amount of storage capacity. Keeping all these points in view, a simple incentive policy is required, which can ensure the following:

- It should balance the upload and download amount of resources at each peer.
- There must be minimum rejections among the cooperative peers.
- Cooperation of peers must be considered in the entire network.
- Lower message overhead and storage capacity is desirable.
- It should be robust to peer dynamics.
- It should be implementable in truly distributed system.

In this chapter, we are proposing an incentive policy, which considers peers' cooperation in the entire network. We are assigning the contribution index to each peer. It is a simplified form of the Biased Contribution Index (BCI), which was discussed in previous chapter. We call it Simplified Biased Contribution Index (SBCI). It also depends on the cooperation of peers in sharing the resources and in balancing the load in the network. SBCI is updated at regular time intervals. At any time, SBCI is calculated using previous SBCI and the cooperation made by the peers during this period, i.e., in between previous update to current update. In the estimation of SBCI, no iterative calculation is required, hence it automatically solves the first and second problems. Once the peers' cooperation is modeled in terms of SBCI, it need not store the history of peers' transactions, hence, it also solves the last problem. Our simulation results show

that SBCI can balance the upload and download amount at each peer with minimum rejections among cooperative peers. Hence it meets all the above design considerations.

Rest of the chapter is organized as follows. The proposed incentive model is introduced in Section 6.3. Section 6.4 covers the analysis of algorithm. The transaction procedure for maximum efficiency is introduced in Section 6.5. Evaluation of algorithm, through simulation is discussed in Section 6.6. Finally, the chapter is concluded in Section 6.7.

## **6.3 Proposed Incentive model**

### **6.3.1 Design Rules to Ensure the Fair and Efficient P2P Network**

Let us make some design rules to ensure the design considerations mentioned in Section 6.2.

- 1). If any peer only downloads the resources from the network then its SBCI must be zero.
- 2). If it only uploads to the network ( at least once to other than free-rider) then its SBCI must be 1.
- 3). Uploading to the free-riders should not increase the SBCI.
- 4). Uploading to any other peer should always increase the SBCI.
- 5). Download should always decrease the SBCI.
- 6). Peers must be motivated to upload to high contributing peers.
- 7). Peers must be motivated to download from low contributing peers.



### 6.3.2 Simplified Biased Contribution Index

Let there be  $N$  peers in a P2P network. Further, we considered time evolution in discrete instances. A time instance is represented by  $t_n$ , and if an event happened in the time interval,  $(t_{n-1}, t_n]$ , it is considered to happen at  $t_n$ . At any time,  $t_n$ , let the share matrix in the entire network be  $\mathbf{S}(\mathbf{t}_n)$ . Where its  $ij$  element is the amount of resource shared by peer  $i$  to peer  $j$  at time  $t_n$ , i.e., in  $(t_{n-1}, t_n]$ . The bias ratio,  $R_i(t_n)$ , for peer  $i$  at time  $t_n$  can be defined in the similar way as defined in previous chapter.

$$R_i(t_n) = \frac{\mathbf{e}_i \cdot \mathbf{S}(\mathbf{t}_n) \cdot \mathbf{x}(\mathbf{t}_n)}{\mathbf{e}_i \cdot \mathbf{S}^{\text{tr}}(\mathbf{t}_n) \cdot \mathbf{x}(\mathbf{t}_n)} \quad (6.1)$$

Here,  $\mathbf{x}(\mathbf{t}_n)$  is the SBCI vector of peers at time  $t_n$ .  $\mathbf{S}^{\text{tr}}(\mathbf{t}_n)$  is transpose of matrix  $\mathbf{S}(\mathbf{t}_n)$  and  $\mathbf{e}_i$  is a row vector with its  $i^{\text{th}}$  entry as 1 and all others as zero. Now, let us define the SBCI,  $x_i(t_n)$ , of peer  $i$  as a monotonically increasing function of the bias ratio at time  $t_{n-1}$ .

$$\begin{aligned} x_i(t_n) &= \frac{R_i(t_{n-1})}{1 + R_i(t_{n-1})} \\ &= \frac{\mathbf{e}_i \cdot \mathbf{S}(\mathbf{t}_{n-1}) \cdot \mathbf{x}(\mathbf{t}_{n-1})}{\mathbf{e}_i \cdot \mathbf{S}(\mathbf{t}_{n-1}) \cdot \mathbf{x}(\mathbf{t}_{n-1}) + \mathbf{e}_i \cdot \mathbf{S}^{\text{tr}}(\mathbf{t}_{n-1}) \cdot \mathbf{x}(\mathbf{t}_{n-1})} \end{aligned} \quad (6.2)$$

If any peer  $i$  does not upload anything in the network at time  $t_{n-1}$ , then  $\mathbf{e}_i \cdot \mathbf{S}(\mathbf{t}_{n-1}) \cdot \mathbf{x}(\mathbf{t}_{n-1}) = 0$ . But if it download something from the network at this time, then  $\mathbf{e}_i \cdot \mathbf{S}^{\text{tr}}(\mathbf{t}_{n-1}) \cdot \mathbf{x}(\mathbf{t}_{n-1}) \neq 0$  only if  $\mathbf{x}(\mathbf{t}_{n-1}) \neq 0$ . Therefore, to make the denominator in (6.2) nonzero for zero uploading and nonzero downloading, let us replace  $\mathbf{e}_i \cdot \mathbf{S}^{\text{tr}}(\mathbf{t}_{n-1}) \cdot \mathbf{x}(\mathbf{t}_{n-1})$  by  $\alpha \mathbf{e}_i \cdot \mathbf{S}^{\text{tr}}(\mathbf{t}_{n-1}) \cdot \mathbf{x}(\mathbf{t}_{n-1}) + (1 - \alpha) \mathbf{e}_i \cdot \mathbf{S}^{\text{tr}}(\mathbf{t}_{n-1}) \cdot \mathbf{e}$ . Here,  $\alpha \in (0, 1)$  is constant and  $\mathbf{e}$  is a column vector with each element as 1. Hence, (6.2) will be:

$$\begin{aligned} x_i(t_n) &= [\mathbf{e}_i \cdot \mathbf{S}(\mathbf{t}_{n-1}) \cdot \mathbf{x}(\mathbf{t}_{n-1})] / [\mathbf{e}_i \cdot \mathbf{S}(\mathbf{t}_{n-1}) \cdot \mathbf{x}(\mathbf{t}_{n-1}) + \\ &\quad \alpha \mathbf{e}_i \cdot \mathbf{S}^{\text{tr}}(\mathbf{t}_{n-1}) \cdot \mathbf{x}(\mathbf{t}_{n-1}) + (1 - \alpha) \mathbf{e}_i \cdot \mathbf{S}^{\text{tr}}(\mathbf{t}_{n-1}) \cdot \mathbf{e}]. \end{aligned} \quad (6.3)$$

SBCI in the above equation is estimated using the transactions, which are happening only at time  $t_{n-1}$ . If we consider all the past transactions, then SBCI can be modified as:

$$\begin{aligned} x_i(t_n) = & (1 - \beta_i(t_{n-1}))x_i(t_{n-1}) + \\ & \beta_i(t_{n-1})[\mathbf{e}_i \cdot \mathbf{S}(\mathbf{t}_{n-1}) \cdot \mathbf{x}(\mathbf{t}_{n-1})] / [\mathbf{e}_i \cdot \mathbf{S}(\mathbf{t}_{n-1}) \cdot \mathbf{x}(\mathbf{t}_{n-1}) + \\ & \alpha \mathbf{e}_i \cdot \mathbf{S}^{\text{tr}}(\mathbf{t}_{n-1}) \cdot \mathbf{x}(\mathbf{t}_{n-1}) + (1 - \alpha) \mathbf{e}_i \cdot \mathbf{S}^{\text{tr}}(\mathbf{t}_{n-1}) \cdot \mathbf{e}]. \end{aligned} \quad (6.4)$$

If peer  $i$  does not participate in any transaction at time  $t_{n-1}$ , then  $x_i(t_n)$  should be  $x_i(t_{n-1})$ . Parameter  $\beta_i(t_{n-1})$  can be decided by the fraction of transaction, which are happening at time,  $t_{n-1}$ , at node  $i$ , and can be defined as:

$$\beta_i(t_{n-1}) = \begin{cases} 0, & \text{if } A_{u_i} = 0. \\ \frac{\mathbf{e}_i \cdot [\mathbf{S}(\mathbf{t}_{n-1}) + \mathbf{S}^{\text{tr}}(\mathbf{t}_{n-1})] \cdot \mathbf{e}}{\mathbf{e}_i \cdot [\mathbf{S}_{\text{comp}}(\mathbf{t}_{n-1}) + \mathbf{S}^{\text{tr}}_{\text{comp}}(\mathbf{t}_{n-1})] \cdot \mathbf{e}}, & \text{otherwise} \end{cases} \quad (6.5)$$

Here,  $A_{u_i} = \mathbf{e}_i \cdot \mathbf{S}(\mathbf{t}_{n-1}) \cdot \mathbf{x}(\mathbf{t}_{n-1}) + \mathbf{e}_i \cdot \mathbf{S}^{\text{tr}}(\mathbf{t}_{n-1}) \cdot \mathbf{e}$ . The  $\mathbf{S}_{\text{comp}}(\mathbf{t}_{n-1})$  is a complete share matrix with its  $ij$  element as the amount of resources shared by peer  $i$  to peer  $j$ , till time  $t_{n-1}$ . To start the process of sharing, the SBCI vector can be initialized as,  $\mathbf{x}(\mathbf{0}) = \alpha / (1 + \alpha) \mathbf{e}$ , later we will see that this choice of initialization will balance the upload and download amounts in the network.

### 6.3.3 Justification For Design Rules

If any peer  $i$ , does not upload anything and only download the resources from the network at time  $t_{n-1}$ , then  $\mathbf{e}_i \cdot \mathbf{S}(\mathbf{t}_{n-1}) \cdot \mathbf{x}(\mathbf{t}_{n-1}) = 0$  and  $\mathbf{e}_i \cdot \mathbf{S}^{\text{tr}}(\mathbf{t}_{n-1}) \cdot \mathbf{e} \neq 0$ , hence, from (6.4),

$$x_i(t_n) = (1 - \beta_i(t_{n-1}))x_i(t_{n-1})$$

Let us assume that the peer did not upload anything in the network till time  $t_n$ , and started downloading the resource first time at time  $t_m$ , then from (6.5),  $\beta_i(t_m) = 1$ ,

hence

$$x_i(t_n) = (1 - \beta_i(t_{n-1}))(1 - \beta_i(t_{n-2})) \dots (1 - \beta_i(t_m))x_i(t_m) = 0.$$

Therefore, **if any peer  $i$ , only downloads from the network then its SBCI will be zero.**

At time  $t_{n-1}$ , if any peer  $i$  uploads only to the free-riders, i.e., peers who only download without uploading anything in the network, then  $\mathbf{e}_i \cdot \mathbf{S}(\mathbf{t}_{n-1}) \cdot \mathbf{x}(\mathbf{t}_{n-1}) = 0$ , if it does not download anything at this time,  $t_{n-1}$ , then  $\mathbf{e}_i \cdot \mathbf{S}^{\text{tr}}(\mathbf{t}_{n-1}) \cdot \mathbf{e} = 0$ . Therefore,  $A_{u_i} = 0$ , and hence from (6.5),  $\beta_i(t_{n-1}) = 0$ , and from (6.4)

$$x_i(t_n) = x_i(t_{n-1})$$

Therefore, **uploading to the free-riders will not increase the SBCI.**

At time  $t_{m-1}$ , if any peer  $i$ , only uploads the resources in the network (at least one of the downloader should be other than free-rider) and does not download anything from it then,  $\mathbf{e}_i \cdot \mathbf{S}(\mathbf{t}_{m-1}) \cdot \mathbf{x}(\mathbf{t}_{m-1}) \neq 0$  and  $\alpha \mathbf{e}_i \cdot \mathbf{S}^{\text{tr}}(\mathbf{t}_{m-1}) \cdot \mathbf{x}(\mathbf{t}_{m-1}) + (1 - \alpha) \mathbf{e}_i \cdot \mathbf{S}^{\text{tr}}(\mathbf{t}_{m-1}) \cdot \mathbf{e} = 0$ . Hence from (6.4),

$$x_i(t_m) = (1 - \beta_i(t_{m-1}))x_i(t_{m-1}) + \beta_i(t_{m-1})$$

Let it is first time when the peer  $i$  makes any transaction in the network, then from (6.5),  $\beta_i(t_{m-1}) = 1$ . Hence,

$$x_i(t_m) = 0 \cdot x_i(t_{m-1}) + \beta_i(t_{m-1}) = \beta_i(t_{m-1}) = 1$$

Now, at time  $t_m$ , if it does not participate in any transaction then

$$x_i(t_{m+1}) = x_i(t_m) = 1$$

if it uploads to only free-riders and does not download anything then again

$$x_i(t_{m+1}) = x_i(t_m) = 1$$

if it uploads to at least one of the peer other than free-rider without downloading anything then,

$$\begin{aligned} x_i(t_{m+1}) &= (1 - \beta_i(t_m))x_i(t_m) + \beta_i(t_m) \\ &= (1 - \beta_i(t_m))1 + \beta_i(t_m) = 1. \end{aligned}$$

Hence, from mathematical induction, we can say that this is true for any  $n$  thus,  $x_i(t_n) = 1$

**Therefore, if any peer  $i$ , only uploads to the network (at least once to other than free-rider) then its SBCI will be 1.**

If any peer  $i$ , uploads the resources to non-free-rider peer, at time  $t_{n-1}$ , then

$$\mathbf{e}_i \cdot \mathbf{S}(\mathbf{t}_{n-1}) \cdot \mathbf{x}(\mathbf{t}_{n-1}) \neq 0.$$

If it does not download anything at this time then,

$$\alpha \mathbf{e}_i \cdot \mathbf{S}^{\text{tr}}(\mathbf{t}_{n-1}) \cdot \mathbf{x}(\mathbf{t}_{n-1}) + (1 - \alpha) \mathbf{e}_i \cdot \mathbf{S}^{\text{tr}}(\mathbf{t}_{n-1}) \cdot \mathbf{e} = 0.$$

Hence, from (6.4),

$$x_i(t_n) = (1 - \beta_i(t_{n-1}))x_i(t_{n-1}) + \beta_i(t_{n-1})$$

It is a convex combination of 1 and  $x_i(t_{n-1})$  hence,

$$x_i(t_{n-1}) < x_i(t_n) < 1 \quad \forall \beta_i(t_{n-1}) \in (0, 1)$$

**Therefore, uploading to the peer other than free-rider will always increase the SBCI.**

If any peer  $i$ , downloads the resource from the network at time  $t_{n-1}$ , then  $\mathbf{e}_i \cdot \mathbf{S}^{\text{tr}}(\mathbf{t}_{n-1}) \cdot \mathbf{e} \neq 0$ , hence  $A_u \neq 0$ , therefore, from (6.5),  $\beta_i(t_{n-1}) > 0$ . If it does not upload anything in the network at this time, then  $\mathbf{e}_i \cdot \mathbf{S}(\mathbf{t}_{n-1}) \cdot \mathbf{x}(\mathbf{t}_{n-1}) = 0$ , hence from (6.4)

$$\begin{aligned} x_i(t_n) &= (1 - \beta_i(t_{n-1}))x_i(t_{n-1}) + \beta_i(t_{n-1}) \cdot 0 \\ &= (1 - \beta_i(t_{n-1}))x_i(t_{n-1}) \end{aligned}$$

hence,

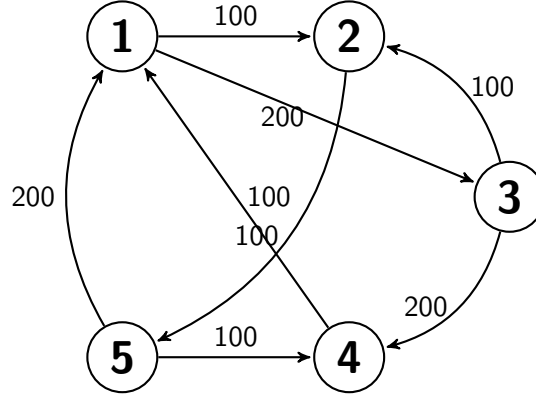
$$x_i(t_n) < x_i(t_{n-1})$$

Therefore, **download will always decrease the SBCI.**

It can be concluded from the above discussion that high contributions will lead to high SBCI. Now, observing directly the (6.4), if peers will upload the resources to high SBCI peers then, they will earn more SBCI. Therefore, **peers will be motivated to upload the resources to high contributing peers.**

It can also be observed from (6.4) that peers will lose less SBCI, if they will download from a low SBCI peer. Therefore, **peers will be motivated to download from low contributing peers.**

Let us understand the SBCI and its computation through an example. Let there be five peers A, B, C, D and E in a P2P network as shown in Fig. 6.1. If  $\alpha = 0.9$ , then initial SBCI of all the peers will be  $\alpha/(1 + \alpha) = 0.4737$ . At time  $t = 0$ , let them share the resources as shown in figure, i.e.,  $S_{12}(0) = 100, S_{13}(0) = 200, S_{25}(0) = 100, S_{32}(0) = 100, S_{34}(0) = 200, S_{41}(0) = 100, S_{51}(0) = 200, S_{54}(0) = 100$  and all others are zero. Since, it is initial step, hence, for all  $i$ ,  $\beta_i(0) = 1$ . Using (6.4), SBCI vector at time  $t = 1$ , can be calculated as,  $\mathbf{x}(1) = [0.4737, 0.3103, 0.5745, 0.2308, 0.7297]^t$ .

Fig. 6.1: Upload and download at each peer at time  $t = 0$ 

Now, let peer 1 needs the data amount of 100 units and all the four peers responded to his query, then peer 1 will select the peer with least SBCI as an uploader, in this case, peer 4 has least SBCI. After this transaction, let SBCI vector is updated at  $t = 2$ . For  $t = 1$ ,  $S_{41}(1) = 100$  and all others are zero. Hence for this time,  $\beta_1(1) = 1/7$ ,  $\beta_2(1) = \beta_3(1) = \beta_5(1) = 0$  and  $\beta_4(1) = 1/5$ . Hence, updated SBCI vector will be,  $\mathbf{x}(2) = [0.4060, 0.3103, 0.5745, 0.3846, 0.7297]^{tr}$ .

#### 6.3.4 Justification For Fairness

**Lemma 6.3.1.** *At any time  $t_{n-1}$ , if upload and download at each peer is same and SBCI vector,  $\mathbf{x}(\mathbf{t}_{n-1}) = \alpha/(1 + \alpha)\mathbf{e}$ , then SBCI vector,  $\mathbf{x}(\mathbf{t}_n) = \mathbf{x}(\mathbf{t}_{n-1})$ .*

**Proof.** Let upload and download for any peer  $i$  at time  $t_{n-1}$  be  $T_i(t_{n-1})$ , then

$$\mathbf{e}_i \cdot \mathbf{S}_i(\mathbf{t}_{n-1}) \cdot \mathbf{e} = \mathbf{e}_i \cdot \mathbf{S}_i^{tr}(\mathbf{t}_{n-1}) \cdot \mathbf{e} = T_i(t_{n-1}).$$

Since,  $\mathbf{x}(\mathbf{t}_{n-1}) = \alpha/(1 + \alpha)\mathbf{e} = a\mathbf{e}$ , here,  $a = \alpha/(1 + \alpha)$ , hence from (6.4),

$$\begin{aligned} x_i(t_n) &= (1 - \beta_i(t_{n-1}))a + \\ &\beta_i(t_{n-1})[a\mathbf{e}_i \cdot \mathbf{S}(\mathbf{t}_{n-1}) \cdot \mathbf{e}] / [a\mathbf{e}_i \cdot \mathbf{S}(\mathbf{t}_{n-1}) \cdot \mathbf{e} + \alpha a\mathbf{e}_i \cdot \mathbf{S}^{\text{tr}}(\mathbf{t}_{n-1}) \cdot \mathbf{e} + (1 - \alpha)\mathbf{e}_i \cdot \mathbf{S}^{\text{tr}}(\mathbf{t}_{n-1}) \cdot \mathbf{e}] \\ &= (1 - \beta_i(t_{n-1}))a + \beta_i(t_{n-1})[aT_i(t_{n-1})] / [T_i(t_{n-1})(a + \alpha a + (1 - \alpha))] \\ &= (1 - \beta_i(t_{n-1}))a + \beta_i(t_{n-1})a / (a(1 + \alpha) + (1 - \alpha)) \end{aligned}$$

Put  $a = \alpha/(1 + \alpha)$ , hence

$$x_i(t_n) = (1 - \beta_i(t_{n-1}))a + \beta_i(t_{n-1})a = a \quad \forall i$$

□

**Lemma 6.3.2.** *If SBCI vector at any two successive time instances,  $t_{n-1}$  and  $t_n$ , is same and lie on vector  $\mathbf{e}$ , then upload and download at time  $t_{n-1}$  will be same in each peer.*

**Proof.** Let  $\mathbf{x}(\mathbf{t}_n) = \mathbf{x}(\mathbf{t}_{n-1}) = a\mathbf{e}$ , where  $a$  is any constant, then from (6.4)

$$\begin{aligned} a &= (1 - \beta_i(t_{n-1}))a + \\ &\beta_i(t_{n-1})[a\mathbf{e}_i \cdot \mathbf{S}(\mathbf{t}_{n-1}) \cdot \mathbf{e}] / [a\mathbf{e}_i \cdot \mathbf{S}(\mathbf{t}_{n-1}) \cdot \mathbf{e} + \alpha a\mathbf{e}_i \cdot \mathbf{S}^{\text{tr}}(\mathbf{t}_{n-1}) \cdot \mathbf{e} + (1 - \alpha)\mathbf{e}_i \cdot \mathbf{S}^{\text{tr}}(\mathbf{t}_{n-1}) \cdot \mathbf{e}]. \end{aligned}$$

Manipulating above, we get

$$\begin{aligned} a\beta_i(t_{n-1}) &= a\beta_i(t_{n-1})[\mathbf{e}_i \cdot \mathbf{S}(\mathbf{t}_{n-1}) \cdot \mathbf{e}] / [a\mathbf{e}_i \cdot \mathbf{S}(\mathbf{t}_{n-1}) \cdot \mathbf{e} + \\ &\alpha a\mathbf{e}_i \cdot \mathbf{S}^{\text{tr}}(\mathbf{t}_{n-1}) \cdot \mathbf{e} + (1 - \alpha)\mathbf{e}_i \cdot \mathbf{S}^{\text{tr}}(\mathbf{t}_{n-1}) \cdot \mathbf{e}]. \end{aligned}$$

For nonzero  $a\beta_i(t_{n-1})$ ,

$$a\mathbf{e}_i \cdot \mathbf{S}(\mathbf{t}_{n-1}) \cdot \mathbf{e} + (a\alpha + 1 - \alpha)\mathbf{e}_i \cdot \mathbf{S}^{\text{tr}}(\mathbf{t}_{n-1}) \cdot \mathbf{e} = \mathbf{e}_i \cdot \mathbf{S}(\mathbf{t}_{n-1}) \cdot \mathbf{e}.$$

Solving,

$$(a\alpha + 1 - \alpha)\mathbf{e}_i.\mathbf{S}^{\text{tr}}(\mathbf{t}_{n-1}).\mathbf{e} = (1 - a)\mathbf{e}_i.\mathbf{S}(\mathbf{t}_{n-1}).\mathbf{e} \quad \forall i \quad (6.6)$$

Since,  $i = 1, 2, \dots, N$ , hence this set of  $N$  equations can be written in the form of matrix as follows,

$$(a\alpha + 1 - \alpha)\mathbf{S}^{\text{tr}}(\mathbf{t}_{n-1}).\mathbf{e} = (1 - a)\mathbf{S}(\mathbf{t}_{n-1}).\mathbf{e}$$

Pre-multiplying by  $\mathbf{e}^{\text{tr}}$  on both sides,

$$(a\alpha + 1 - \alpha)\mathbf{e}^{\text{tr}}.\mathbf{S}^{\text{tr}}(\mathbf{t}_{n-1}).\mathbf{e} = (1 - a)\mathbf{e}^{\text{tr}}.\mathbf{S}(\mathbf{t}_{n-1}).\mathbf{e}$$

for any matrix,  $\mathbf{S}(\mathbf{t}_{n-1})$ ,  $\mathbf{e}^{\text{tr}}.\mathbf{S}(\mathbf{t}_{n-1}).\mathbf{e}$  will be the sum of all of its elements, hence  $\mathbf{e}^{\text{tr}}.\mathbf{S}(\mathbf{t}_{n-1}).\mathbf{e} = \mathbf{e}^{\text{tr}}.\mathbf{S}^{\text{tr}}(\mathbf{t}_{n-1}).\mathbf{e} = T$  hence,

$$(a\alpha + 1 - \alpha)T = (1 - a)T$$

since  $T \neq 0$  hence,

$$a = \frac{\alpha}{1 + \alpha}$$

Substituting the value of  $a$  in (6.6)

$$\left(\frac{\alpha^2}{1 + \alpha} + 1 - \alpha\right)\mathbf{e}_i.\mathbf{S}^{\text{tr}}(\mathbf{t}_{n-1}).\mathbf{e} = \left(1 - \frac{\alpha}{1 + \alpha}\right)\mathbf{e}_i.\mathbf{S}(\mathbf{t}_{n-1}).\mathbf{e}$$

or

$$(1 + \alpha)\mathbf{e}_i.\mathbf{S}^{\text{tr}}(\mathbf{t}_{n-1}).\mathbf{e} = (1 + \alpha)\mathbf{e}_i.\mathbf{S}(\mathbf{t}_{n-1}).\mathbf{e}$$

since  $\alpha \in (0, 1)$ , hence

$$\mathbf{e}_i.\mathbf{S}^{\text{tr}}(\mathbf{t}_{n-1}).\mathbf{e} = \mathbf{e}_i.\mathbf{S}(\mathbf{t}_{n-1}).\mathbf{e} \quad \forall i$$

Hence, upload and download at time  $t_{n-1}$  will be same in each peer  $i$ .

□



## 6.4 Analysis of Algorithm

### 6.4.1 Implementation in Distributed System

SBCI of each peer can be calculated distributively as shown in Algorithm 6.1. Each peer's SBCI can be calculated and managed by some other peer in the network. We call it index manager and the peer whose SBCI is being calculated by this peer is called its daughter peer. The index manager peer can be located using distributed hash table (DHT) such as Chord [19], CAN [20], Pastry [21] and Tapestry [22]. Each peer  $i$  will send the values of resources uploaded and downloaded to and from other peer  $j$  to the index manager of peer  $j$ . An index manager peer will collect the values of resources uploaded and downloaded by its daughter peer  $k$ , to other peers. Each index manager will locate the index manager of peer  $j$  and will receive the current SBCI,  $x_j(t_{n-1})$ , of peer  $j$ .

Now each index manager possesses all the things to calculate the SBCI of its daughter peer using (6.4). The  $\beta_k(t_{n-1})$  can be calculated using (6.5). If  $A_u = 0$  then it is zero, otherwise it is just a ratio of the current transaction amount to the total transaction amount made by peer  $k$ , till time  $t_{n-1}$ . The total amount of transactions can be updated by adding the current amount of transaction with the previous total amount of transactions.

### 6.4.2 Message Overhead, Storage Capacity and Time Complexity

In this method, the SBCI is calculated directly while in other similar approaches [8], [69] iterative calculations are required. Therefore, the total number of messages

required to calculate the SBCI, in this method will be  $I_1$  and  $I_2$  times lesser than [8] and [69] respectively. Where  $I_1$  and  $I_2$  is the number of iterations required to converge the algorithm in [8] and [69] respectively.

In this algorithm, index manager needs to store only two information about its daughter peer, i.e., current SBCI and total amount of transaction till  $t_{n-1}$ . While in [8], [69], all transaction history of its daughter peer, i.e., amount of transaction, ID of peer with whom it transacted and whether it was upload or download, are required to be stored. Therefore, the required amount of storage is reduced very much.

Time complexity of algorithm for one update can be calculated directly from (6.5). It will be  $O(N)$  per peer which is same as in [8] and [69].

## 6.5 Transaction Procedure for Maximum Efficiency

### 6.5.1 Simple Procedure For Peer Selection

All the peers are rational and aware of the fact that, if they will share their resources with peer having high SBCI, then their SBCI will be higher, and if they will download from a low SBCI peer then they will lose less SBCI. Therefore, the simple peer selection procedure for any peer  $i$  is to download from low SBCI peer and to upload to high SBCI peer, as far as possible, as shown in Algorithm 6.2.

**Algorithm 6.1** For Updating the SBCI of Peers

---

```

1: Input: Amount of upload and download of peers
2: Output: SBCI with index managers
3: procedure
4:   for each peer  $i$  do
5:     forall peer  $j$ , who is selected as source peer do
6:       Download the resource
7:       Send the value of resource to the index manager of peer  $j$ ;
8:     end forall
9:     forall peer  $j$ , who selected peer  $i$  as source peer do
10:      Upload the resource
11:      Send the value of resource to the index manager of peer  $j$ ;
12:    end forall
13:    if Peer  $i$  is index manager of peer  $k$  then
14:      forall peer  $j$ , who transacted with peer  $k$  do
15:        Receive the value of resource uploaded  $S_{kj}(t_{n-1})$ ;
16:        Receive the value of resource downloaded  $S_{jk}(t_{n-1})$ ;
17:        if  $t = 0$  then
18:          \ Initialization of parameters
19:          Set  $\mathbf{x}(0) = (\alpha/(1 + \alpha))\mathbf{e}$ ;
20:        else
21:          Locate  $j^{th}$  peer's index manager;
22:          Receive the current SBCI,  $x_j(t_{n-1})$  of peer  $j$  from these index
managers ;
23:        end if
24:      end forall
25:      \ Initialization of the amount of total transactions
26:      Set  $Ttr_k(0) = 0$ 
27:      Compute
28:       $A_{u_k} = \mathbf{e}_k \cdot \mathbf{S}(\mathbf{t}_{n-1}) \cdot \mathbf{x}(\mathbf{t}_{n-1}) + \mathbf{e}_k \cdot \mathbf{S}^{tr}(\mathbf{t}_{n-1}) \cdot \mathbf{e}$ 
29:      if  $A_{u_k} = 0$  then
30:         $\beta_k(t_{n-1}) = 0$ 
31:      else
32:         $\delta Ttr_k(t_{n-1}) = \mathbf{e}_k \cdot (\mathbf{S}(\mathbf{t}_{n-1}) + \mathbf{S}^{tr}(\mathbf{t}_{n-1})) \cdot \mathbf{e}$ 
33:         $Ttr_k(t_{n-1}) = Ttr_k(t_{n-2}) + \delta Ttr_k(t_{n-1})$ 
34:         $\beta_k(t_{n-1}) = \frac{\delta Ttr_k(t_{n-1})}{Ttr_k(t_{n-1})}$ 
35:      end if
36:      Compute  $x_k(t_n)$  using (6.4)
37:      Save  $x_k(t_n)$ 
38:      Save  $Ttr_k(t_{n-1})$ 
39:    end if
40:  end for
41: end procedure

```

---

---

**Algorithm 6.2** Simple Procedure for Peer Selection

---

```

1: procedure
2:   if Peer  $i$  needs a resource then
3:     Send the request for resource;
4:     Get the SBCI of responding peers from their respective index managers;
5:     Select the source peer having minimum SBCI;
6:     Download the resource;
7:     Send the value of resource to the index manager of source peer;
8:   end if
9:   if Peer  $i$  get a request for a resource then
10:    Get the SBCI of requesting peer from their respective index managers;
11:    if SBCI of all requesting peer is less than the threshold then
12:      Reject all the requesting peers;
13:    else
14:      Select the peer with maximum SBCI;
15:      Upload the resource;
16:      Send the value of resource to the index manager of downloading peer;
17:    end if
18:  end if
19: end procedure

```

---

## 6.5.2 College Admission and The Stability of Marriage Based Approach For Peer Selection

### 6.5.2.1 Preliminaries

College Admission and the stability of Marriage is a well-known problem, introduced by Gale and Shapley [11]. In its most popular variants, there are two disjoint sets of cardinality,  $n$ . One set is representing the men and the other one is representing the women. Each person has a different order of preference for his or her marriage partner. There are several ways by which one-to-one pairing can be done. But a pairing is said to be stable, if there is no pair both of whom prefer each other to their actual partner.

Gale and Shapley [11] provide the solution and the algorithm for stable pairing. They also proved that there always exists a stable match for such type of problem. In

this algorithm, one of the group proposes his or her first preference, another group can reject the proposal or can keep it on hold until they get a better option. If any member from the proposing group get rejected, he or she tries on next preference. This process continues until proposing group is not rejected or rejected by all of his or her preferred partners.

If a proposal is given by men, then they get the better preferred partner as compared to any other stable pairing, hence it is called man optimal stable matching, the other way around women optimal stable matching.

#### 6.5.2.2 Application in Peer Selection

We considered the situation where there are many uploaders and many downloaders for a resource. In order to earn the high SBCI, uploader would like to upload the resource to high SBCI peers, thus they have certain preferences for downloaders. On the other hand, for downloaders the resource and the SBCI both matter. Therefore, downloader may prefer the higher bandwidth uploader over low SBCI uploader. Thus, downloder have a different preference order for uploaders.

In this situation, all uploaders and downloaders preference order can be collected at a certain node. We call it the resource manager node. This node can be found by hashing the resource identifier and finding corresponding root node in DHT network. On this node, the stable marriage algorithm can be used to pair the uploaders and downloaders. A message to each pair will be sent after pairing, so that they can start the process of transaction. Detail of peer selection procedure in this situation is shown in Algorithm 6.3.

---

**Algorithm 6.3** College Admission and The Stability of Marriage Based Approach for Peer Selection

---

```

1: procedure
2:   if Peer  $i$  needs a resource then
3:     Send the request for resource;
4:     Get the SBCI of responding peers from their respective index managers;
5:     Learn about the bandwidth of responding peers;
6:     Make the order of preference for uploader;
7:     Send the order of preference to the resource manager;
8:     Get the ID of uploader partner from resource manager;
9:     Download the resource;
10:    Send the value of resource to the index manager of uploading peer;
11:  end if
12:  if Peer  $i$  get a request for a resource then
13:    Get the SBCI of requesting peers from their respective index managers;
14:    Remove the peers, having SBCI less than the threshold;
15:    Make the order of preference according to their SBCI;
16:    Send the order of preference to the resource manager;
17:    Get the ID of downloader partner from resource manager;
18:    Upload the resource;
19:    Send the value of resource to the index manager of downloading peer;
20:  end if
21:  if Peer  $i$  is the resource manager then
22:    Get the order of preference from uploaders;
23:    Get the order of preference from downloaders;
24:    Run the Stable marriage algorithm;
25:    Send the ID of partner to each peer;
26:  end if
27: end procedure

```

---

## 6.6 Experimental Evaluation

As in [35] and [63], we used NetLogo 5.2 [64], to evaluate the performance of our algorithm. NetLogo is a multiagent programmable modeling environment where we can model different agents and can ask them to perform the task in parallel and independently. It is written mostly in Scala, with some parts in Java.

### 6.6.1 Simulation Setup

We simulated a typical P2P network with parameters and distributions taken from real world measurements as in [24], [71]. In this network, peers can send a query for the resource. We assumed that ten percent of peers respond to this query. After selecting the source peer according to the procedure described in Section 6.5, resource is downloaded. We assumed the amount of resources requested by downloading peers varies randomly between 1 unit to 255 units. After downloading the resource, SBCI of peer is updated by an index manager using (6.4). Any peer whose SBCI is less than the threshold value is rejected and cannot download the resources from the network. We assumed the threshold value of SBCI to be  $\alpha/(1 + \alpha)$ .

The number of nodes in the network is taken as 1000, which is reasonable size. However, the number of nodes can be increased up to any number, but this will not affect the results. Because, evaluation metrics are normalized with respect to the number of nodes. The initial value of SBCI of all the peers are taken as  $\alpha/(1 + \alpha)$ . We conducted the experiment for  $\alpha = 0.9, 0.6$  and  $0.3$ . Percentage of free-riders were varied from 10% to 80%.

The simulation is performed for three different peer distribution models, i.e., Simple,

Adaptive and Extreme Model.

In Simple Model, free-riders vary from 10% - 70%. These free-riders do not share anything at any point of time in the simulation.

In Adaptive Model, free-riders vary from 20% - 60%. Half of these free-riding peers do not share anything during the whole simulation. Remaining half behave as normal peers till midway of simulation, and thereafter convert themselves to free-riders.

In Extreme Model, at the beginning of simulation, 10% peers are free-riders. After completion of every 12.5% of total transactions, 10% more peers convert themselves to free-riders. Thus, at the end of simulation, there will be 80% free-riders. The simulation was run upto 100000 transactions.

### 6.6.2 Evaluation Metrics

We plotted the graph between the total upload and download amounts of each peer for all the models. To get the deeper picture, we also calculated the average absolute deviation (AAD) of upload to download ratio from one, in any model as:

$$AAD = (1/N) \sum_{i=1}^N |1 - \mathbf{e}_i \cdot \mathbf{S}_{\text{comp}}(\mathbf{t}_n) \cdot \mathbf{e} / \mathbf{e}_i \cdot \mathbf{S}_{\text{comp}}^{\text{tr}}(\mathbf{t}_n) \cdot \mathbf{e}|.$$

If upload amount for each peer is same as download amount, then the value of  $AAD$  in the network will be zero. The larger value of  $AAD$  implies, the larger difference between upload and download and thus, lesser fairness in the network.

Network is said to be efficient if free-riders are not allowed to download anything, without affecting the transactions between non-free-rider peers. At any time, if SBCI of any cooperative peer is less than the threshold then it will also get rejected. This is not



TABLE 6.1: AAD and % of Rejections for SBCI in Simple Model for Simple Procedure of Peer Selection

S.N.	$\alpha$	Free-riders	AAD	% of Rejections
1	0.9	10%	0.103772	1.817
2	0.9	30%	0.303675	0.127
3	0.9	50%	0.505119	0.023
4	0.9	70%	0.707233	0.008
5	0.6	10%	0.103667	0.064
6	0.6	30%	0.303652	0.009
7	0.6	50%	0.505459	0.006
8	0.6	70%	0.707117	0.003
9	0.3	10%	0.103844	0.009
10	0.3	30%	0.303671	0.002
11	0.3	50%	0.505088	0.001
12	0.3	70%	0.707069	0

a desired state in the network. Therefore, we calculated the percentage of rejections among cooperative peers, i.e., cooperative peers rejecting the request of cooperative peers. For efficient algorithm, percentage of rejections must be minimum.

For comparison, we also simulated the GC for its best case [8], i.e.,  $\alpha = 0.8$  and  $\beta = 0.2$ . The parameters  $\alpha$  and  $\beta$  are taken to be same as in [8]. For fair comparison, we kept the threshold value for peer selection as  $(2 - \alpha(1 + \beta))/(2 + \alpha(1 - \beta))$ . We kept maximum value of threshold in both GC as well as in SBCI. Rest of the settings for GC are same as in SBCI.

### 6.6.3 Simulation Results of Simple Procedure For Peer Selection

We conducted the simulation experiment for simple procedure of peer selection, as explained in Section 6.5. Bandwidth of all the peers is assumed to be same. For

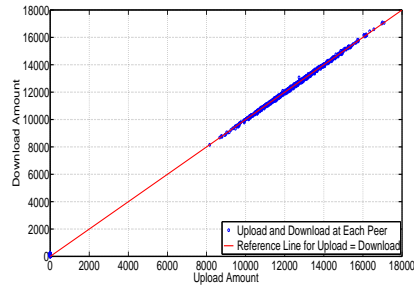
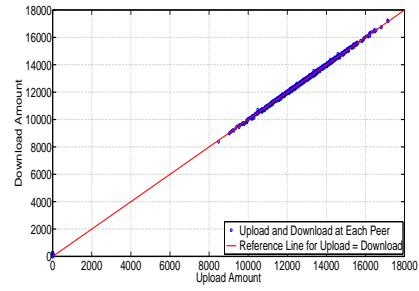
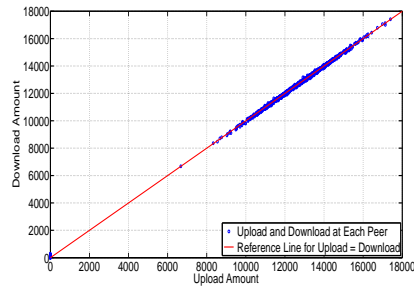
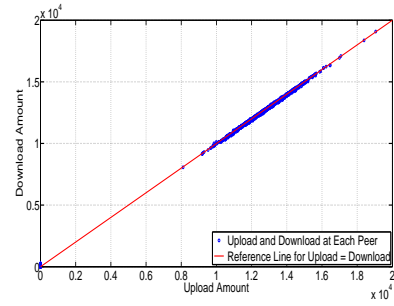
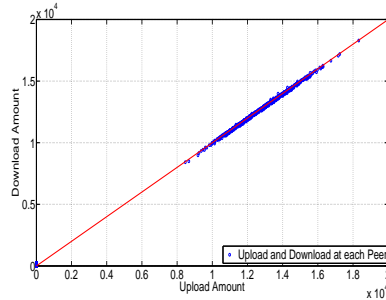
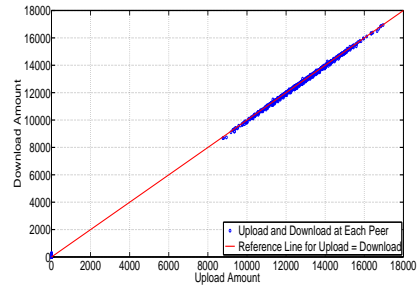
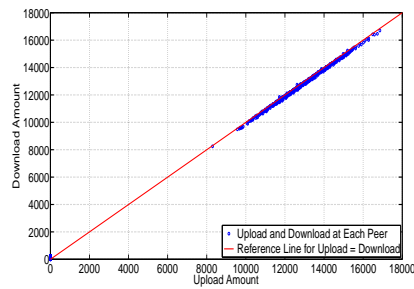
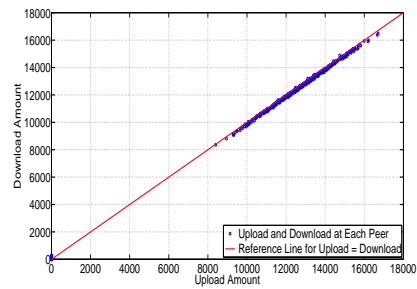
(a) Free Riders = 10 %,  $\alpha = 0.9$ (b) Free Riders = 10 %,  $\alpha = 0.6$ (c) Free Riders = 10 %,  $\alpha = 0.3$ (d) Free Riders = 30 %,  $\alpha = 0.9$ (e) Free Riders = 30 %,  $\alpha = 0.6$ (f) Free Riders = 30 %,  $\alpha = 0.3$ (g) Free Riders = 50 %,  $\alpha = 0.9$ (h) Free Riders = 50 %,  $\alpha = 0.6$ 

Fig. 6.2: Upload and Download Amount at Each Peer for SBCI in Simple Model for Simple Procedure of Peer Selection.

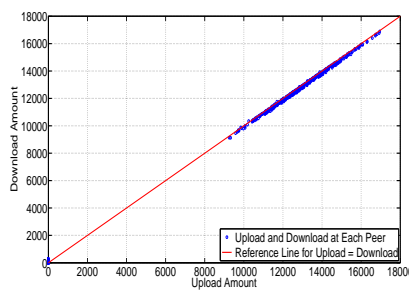
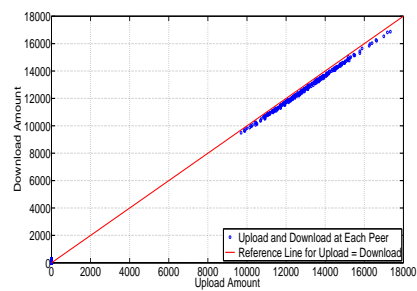
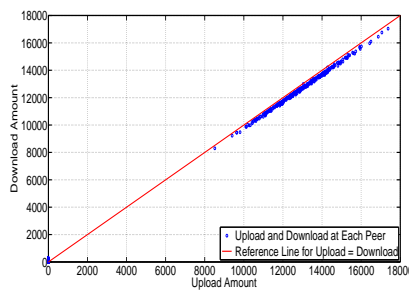
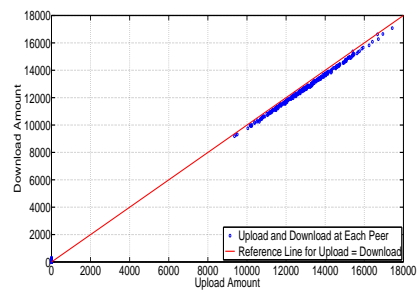
(i) Free Riders = 50 %,  $\alpha = 0.3$ (j) Free Riders = 70 %,  $\alpha = 0.9$ (k) Free Riders = 70 %,  $\alpha = 0.6$ (l) Free Riders = 70 %,  $\alpha = 0.3$ 

Fig. 6.2: Upload and Download Amount at Each Peer for SBCI in Simple Model for Simple Procedure of Peer Selection.

TABLE 6.2: AAD and % of Rejections for SBCI in Adaptive Model for Simple Procedure of Peer Selection

S.N.	$\alpha$	Free-riders	AAD	% of Rejections
1	0.9	20%	0.118469	1.87
2	0.9	40%	0.236766	0.606
3	0.9	60%	0.354868	0.198
4	0.6	20%	0.175055	0.054
5	0.6	40%	0.351099	0.028
6	0.6	60%	0.527092	0.011
7	0.3	20%	0.200618	0.020
8	0.3	40%	0.401786	0.011
9	0.3	60%	0.600657	0.008

simple model, simulation results for SBCI are shown in Fig. 6.2. Corresponding *AAD* and percentage of rejections among cooperative peers are shown in Table 6.1. We can observe from this figure that in initial transactions, free-riders got some resources after that their SBCI become zero, which disqualify them in taking any resources from the network. For all other peers, upload to download ratio is very close to the reference line, thus algorithm is able to maintain the fairness in the network. We can observe from Table 6.1 that the percentage of rejections among the cooperative peers are more for higher values of  $\alpha$ . Because for higher values of  $\alpha$ , threshold value of SBCI will be higher. But its impact on *AAD* is not very significant in this model.

In Adaptive Model, free-riders earn the SBCI and thereafter use this SBCI to download maximum resources from the network. Simulation results for this model are shown in Fig. 6.3. Corresponding *AAD* and percentage of rejections among cooperative peers are shown in Table 6.2. We can observe from this figure that for higher  $\alpha$ , algorithm performs better. For  $\alpha = 0.9$ , even in the presence of a large number of free-riders, the algorithm is able to balance the upload and download amount in the network. We

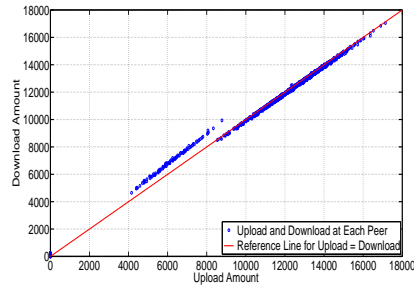
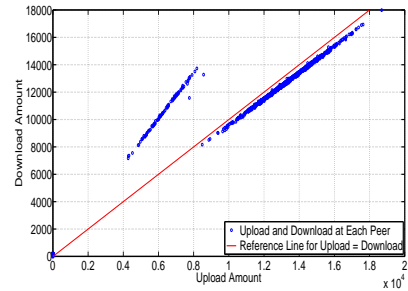
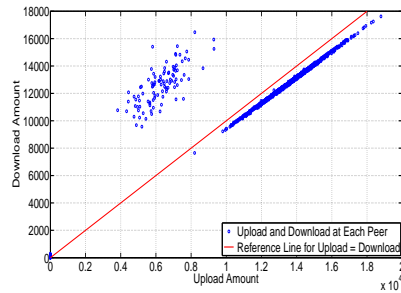
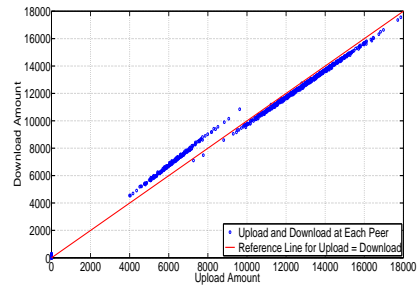
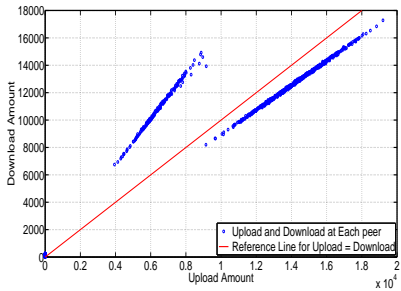
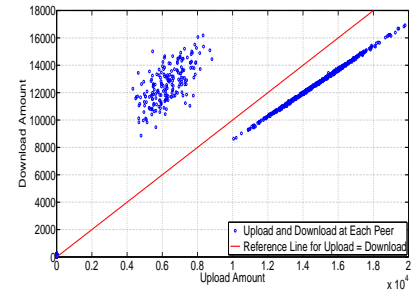
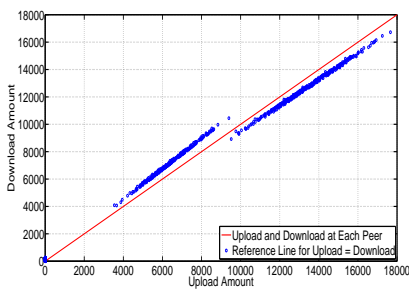
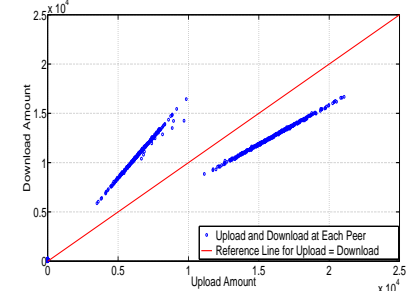
(a) Free Riders = 20 %,  $\alpha = 0.9$ (b) Free Riders = 20 %,  $\alpha = 0.6$ (c) Free Riders = 20 %,  $\alpha = 0.3$ (d) Free Riders = 40 %,  $\alpha = 0.9$ (e) Free Riders = 40 %,  $\alpha = 0.6$ (f) Free Riders = 40 %,  $\alpha = 0.3$ (g) Free Riders = 60 %,  $\alpha = 0.9$ (h) Free Riders = 60 %,  $\alpha = 0.6$ 

Fig. 6.3: Upload and Download Amount at Each Peer for SBCI in Adaptive Model for Simple Procedure of Peer Selection.

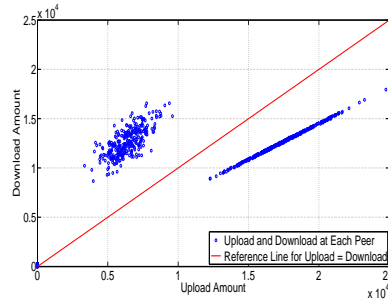
(i) Free Riders = 60 %,  $\alpha = 0.3$ 

Fig. 6.3: Upload and Download Amount at Each Peer for SBCI in Adaptive Model for Simple Procedure of Peer Selection.

TABLE 6.3: AAD and % of Rejections for SBCI in Simple Model for Simple Procedure of Peer Selection

S.N.	$\alpha$	Free-riders	AAD	% of Rejections
1	0.9	80%	0.211228	1.794
2	0.6	80%	0.423116	0.068
3	0.3	80%	0.567303	0.010

can also observe from Table 6.2 that for higher  $\alpha$  the percentage of rejection among cooperative peers is higher but corresponding *AAD* is very less. Thus, impact of  $\alpha$  is clearly evident.

And finally, we conducted the simulation for SBCI in Extreme Model. Results for upload and download at each peer are shown in Fig. 6.4. Corresponding *AAD* and percentage of rejections among cooperative peers are shown in Table 6.3. We can observe from the figure that for  $\alpha = 0.9$  the algorithm is able to balance the upload and download amount in the network. For  $\alpha = 0.9$ , at the cost of less than 2% of rejections among the cooperative peers, algorithm is able to maintain *AAD* as 0.211228.

We also reported the simulation results of GC for all peer distribution models in Fig. 6.5. Corresponding *AAD* and percentage of rejections among cooperative peers

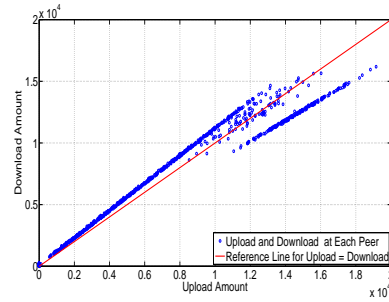
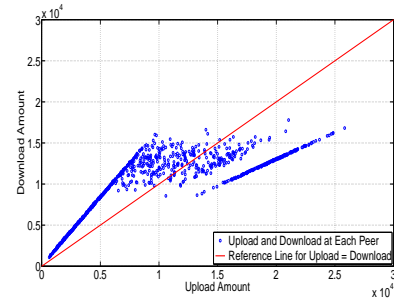
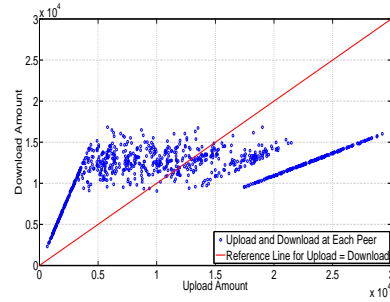
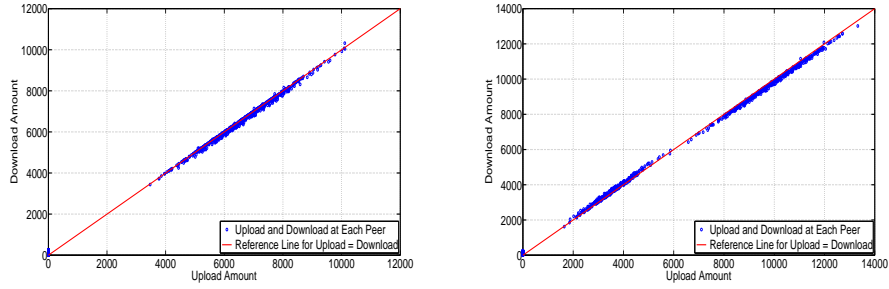
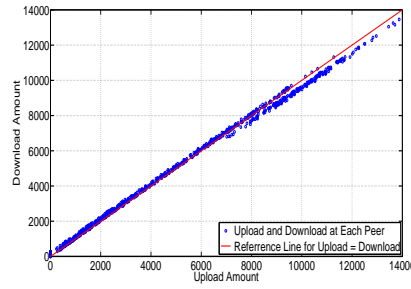
(a) Free Riders = 80%,  $\alpha = 0.9$ (b) Free Riders = 80%,  $\alpha = 0.6$ (c) Free Riders = 80%,  $\alpha = 0.3$ 

Fig. 6.4: Upload and Download Amount at Each Peer for SBCI in Extream Model for Simple Procedure of Peer Selection. At the beginning of simulation, 10% peers are free-riders. After completion of every 12.5% of total transactions, 10% more peers convert themselves to free-riders.



(a) Simple Model, Free Riders = 30% (b) Adaptive Model, Free Riders = 60%



(c) Extreme Model, Free Riders = 80%

Fig. 6.5: Upload and Download Amount at Each Peer for best case of GC, i.e.,  $\alpha = 0.8, \beta = 0.2$  in different distribution models for Simple Procedure of Peer Selection.

are reported in Table 6.4. We can see from the figure that GC can also balance the upload and download amounts in each peer. In Adaptive Model and in Extreme Model GC can maintain better fairness compared to SBCI but the percentage of rejections among cooperative peers are higher in GC for all the models. Thus, it is less efficient compared to SBCI.

#### 6.6.4 Simulation Results of College Admission and The Stability of Marriage Based Approach For the Peer Selection

We also conducted the experiment for college admission and the stability of marriage based approach for the peer selection. For simplicity, we considered only Simple model. Bandwidth of peers is assumed to be different, so that they can also include the band-



TABLE 6.4: AAD and % of Rejections for Different Distribution Model in Best case of GC for Simple Procedure of Peer Selection

S.N.	Model	Free-riders	AAD	% of Rejections
1	Simple	30%	0.3059	33.657
2	Adaptive	60%	0.3146	16.27
3	Extreme	80%	0.1370	19.269

TABLE 6.5: AAD and % of Rejections for SBCI in Simple Model for Stable marriage approach with two different bandwidth peers

S.N.	$\alpha$	Free-riders	AAD	% of Rejections
1	0.9	10%	0.102265	1.0114
2	0.9	30%	0.301927	0.287
3	0.9	50%	0.501482	0.0244
4	0.9	70%	0.701471	0.0026
5	0.6	10%	0.102312	0.046
6	0.6	30%	0.30196	0.0134
7	0.6	50%	0.50149	0.0034
8	0.6	70%	0.701432	0.0008
9	0.3	10%	0.102367	0.0114
10	0.3	30%	0.301967	0.0036
11	0.3	50%	0.501484	0.0016
12	0.3	70%	0.701441	0

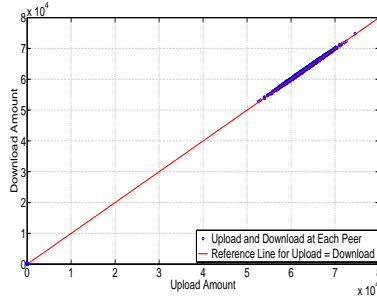
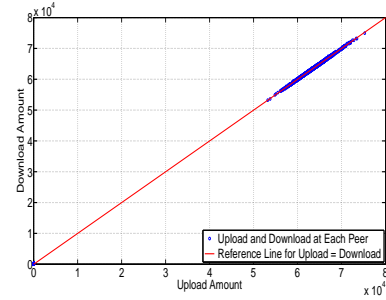
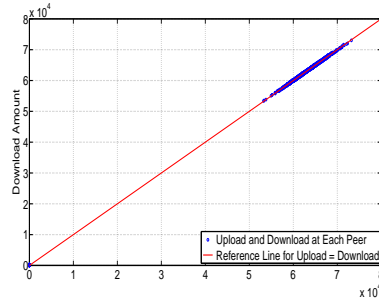
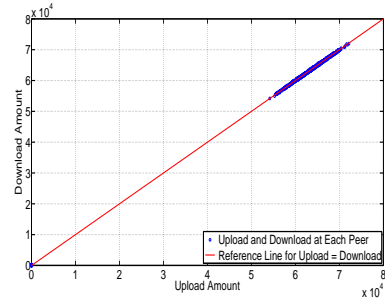
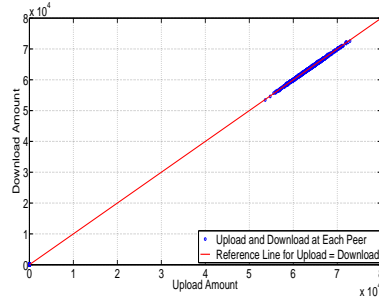
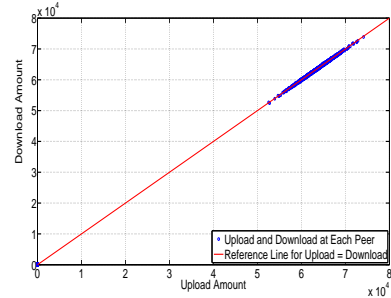
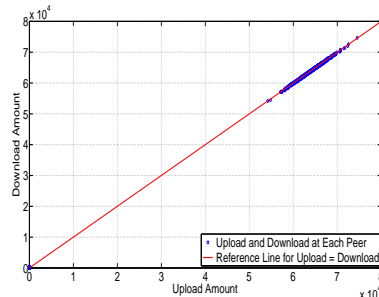
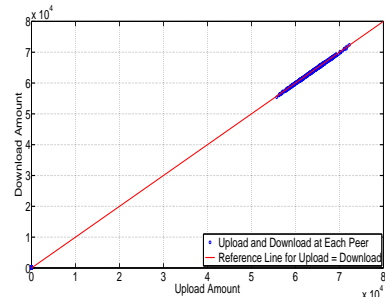
(a) Free Riders = 10%,  $\alpha = 0.9$ (b) Free Riders = 10%,  $\alpha = 0.6$ (c) Free Riders = 10%,  $\alpha = 0.3$ (d) Free Riders = 30%,  $\alpha = 0.9$ (e) Free Riders = 30%,  $\alpha = 0.6$ (f) Free Riders = 30%,  $\alpha = 0.3$ (g) Free Riders = 50%,  $\alpha = 0.9$ (h) Free Riders = 50%,  $\alpha = 0.6$ 

Fig. 6.6: Upload and Download Amount at Each Peer for Different values of  $\alpha$ . Peer selection approach is based on the problem of "College Admission and The Stability of Marriage". Bandwidth distribution is as type 1.

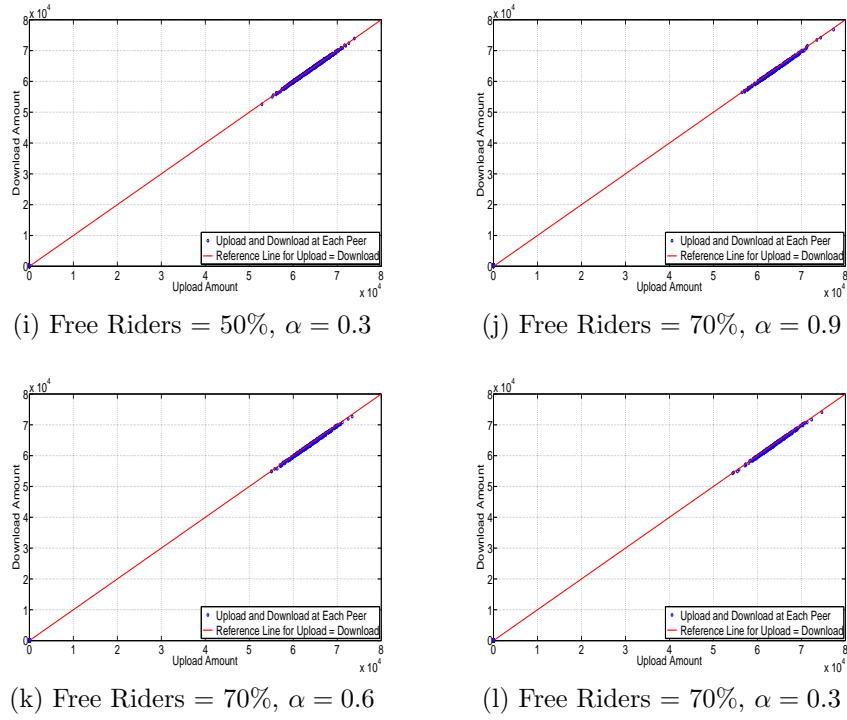


Fig. 6.6: Upload and Download Amount at Each Peer for Different values of  $\alpha$ . Peer selection approach is based on the problem of "College Admission and The Stability of Marriage". Bandwidth distribution is as type 1.

TABLE 6.6: AAD and % of Rejections for SBCI in Simple Model for Stable marriage approach with ten different bandwidth peers

S.N.	$\alpha$	Free-riders	AAD	% of Rejections
1	0.9	10%	0.131826	6.1534
2	0.9	30%	0.322511	4.2034
3	0.9	50%	0.515015	2.3352
4	0.9	70%	0.709604	0.9604
5	0.6	10%	0.128717	0.4344
6	0.6	30%	0.327149	0.2746
7	0.6	50%	0.520278	0.1396
8	0.6	70%	0.710663	0.0348
9	0.3	10%	0.130575	0.1214
10	0.3	30%	0.323463	0.0652
11	0.3	50%	0.514418	0.024
12	0.3	70%	0.70845	0.0054

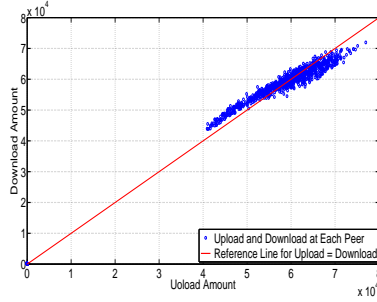
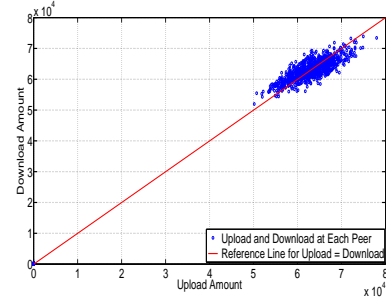
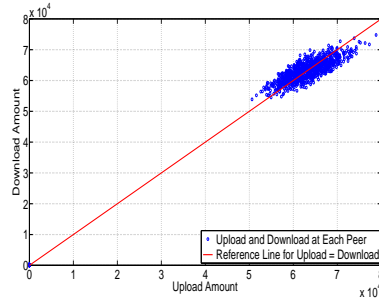
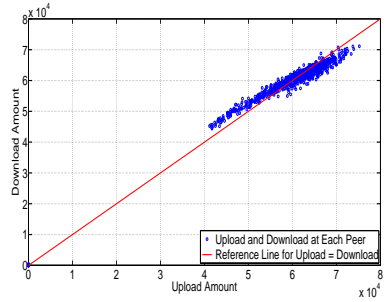
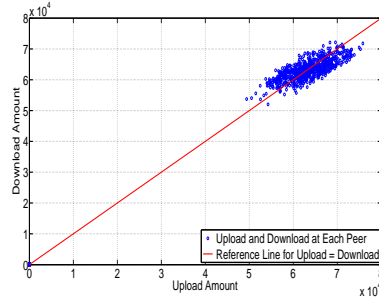
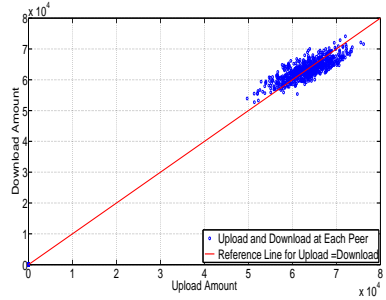
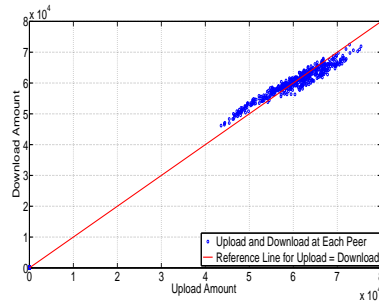
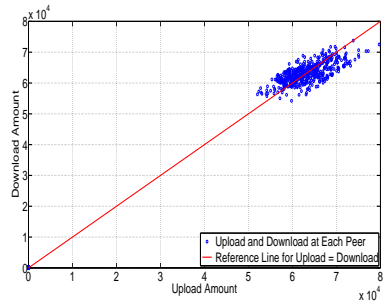
(a) Free Riders = 10%,  $\alpha = 0.9$ (b) Free Riders = 10%,  $\alpha = 0.6$ (c) Free Riders = 10%,  $\alpha = 0.3$ (d) Free Riders = 30%,  $\alpha = 0.9$ (e) Free Riders = 30%,  $\alpha = 0.6$ (f) Free Riders = 30%,  $\alpha = 0.3$ (g) Free Riders = 50%,  $\alpha = 0.9$ (h) Free Riders = 50%,  $\alpha = 0.6$ 

Fig. 6.7: Upload and Download Amount at Each Peer for Different values of  $\alpha$ . Peer selection approach is based on the problem of "College Admission and The Stability of Marriage". Bandwidth distribution is as type 2.

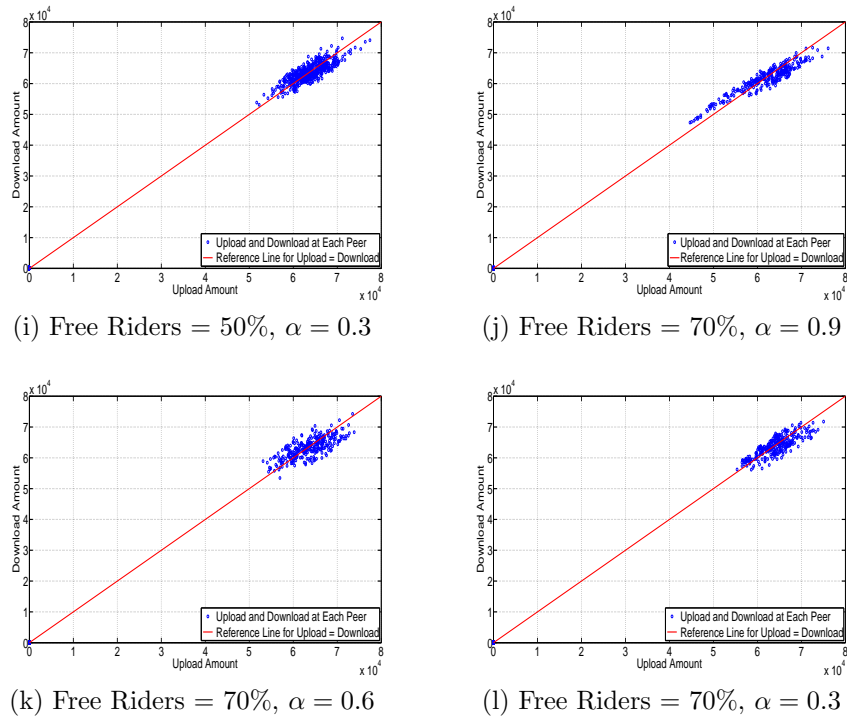


Fig. 6.7: Upload and Download Amount at Each Peer for Different values of  $\alpha$ . Peer selection approach is based on the problem of "College Admission and The Stability of Marriage". Bandwidth distribution is as type 2.

width, as a criteria for peer selection. Selection of peer for downloading and uploading is done according to Algorithm 6.3. The stable match for uploader and downloader is made downloader optimal. To observe the impact of heterogeneity, we simulated the Simple Model for two different types of bandwidth distributions, i.e., type 1 and type 2.

In type 1, half of the peers have bandwidth 10 units and the rest have 20 units. Simulation results for this type are shown in Fig. 6.6. Corresponding *AAD* and percentage of rejections among cooperative peers are shown in Table 6.5. We can see from the figure that upload and download amount increases in each peer compared to simple procedure. Because each peer, who request for resources, is getting some option for downloading. Uploads and downloads in each peer are close to the reference line and corresponding *AAD* are lesser compared to simple procedure. Thus, the algorithm is able to balance the upload and download amount in each peer.

In type 2, 10% of the peers have bandwidth 10 units, next 10% of the peers have bandwidth 20 units, next 10% of peers have bandwidth 30 units and so on. In this way, last 10% of peers will have bandwidth 100 units. Simulation results for this type are shown in Fig. 6.7. Corresponding *AAD* and percentage of rejections among cooperative peers are shown in Table 6.6. We can observe from this figure that upload and download amounts for most of the peers are far from reference line and corresponding *AAD* are also higher. Thus, the impact of heterogeneity is clearly evident. It also supports the argument that if we will select the source peer according to bandwidth rather than SBCI, we will loose the fairness in the network.

## 6.7 Conclusion

In this work, we proposed a new algorithm to make the P2P network fair and efficient. The algorithm ranks the peers based on their simplified biased contribution index (SBCI) which can vary from 0 to 1. Estimation of SBCI is based on two factors, the resources contributed by the peer and the SBCI of peer with whom it is transacting. We propose the design rules to make the network fair and efficient. With the help of mathematical justification, we have shown that our algorithm can fulfill all the design objectives and is able to maintain the fairness in the network. This algorithm can be implemented in the truly distributed fashion. Since, no iterative calculation is needed, it can be implemented with lesser message overhead and storage capacity.

We proposed two different peer selection approaches, namely simple procedure and college admission and the stability of marriage based approach. Simulation results show that the algorithm is able to suppress the free-riders in highly free-riding environment. The algorithm is also able to suppress the dynamic free-riders, i.e., those who change their behavior dynamically.

# Chapter 7

## Conclusion and Future Work

In this thesis, we addressed two major problems in P2P networks, i.e., presence of malicious peers and free-riders. We studied the existing literature in this area and found that an efficient ranking mechanism can solve the above problems. We proposed the Absolute trust algorithm to handle the malicious peers and to suppress the free-riders. We had also proposed Biased Contribution Index (BCI) and its simplified form, SBCI to further improve the performance. In this chapter, we are presenting the conclusions of the reported research work, major contributions and possible future directions of investigations.

### 7.1 Conclusion

We presented the Absolute trust algorithm for the aggregation of local trust. In this algorithm, aggregation is done without normalization hence it reflects the true past behavior of peers. We have shown that the global trust can be calculated by an iterative method, thus the mechanism for Absolute trust can be implemented in a distributed system. The speed of convergence of this algorithm depends upon the parameter  $\alpha$  as



explained in Chapter 3. Lesser the value of  $\alpha$ , higher will be the speed of convergence.

In this algorithm, trust assigning peer sends the value of local trust without normalization. If this peer updates the local trust of any one of the source peers, the local trust of other source peers will not be affected. Therefore, it need not send the local trust of all other source peers at every update thus, lesser number of update messages is required. This has reduced the message complexity of the solution.

We also evaluated the performance of algorithm through simulation experiments. We have shown that our algorithm is robust against various malicious behavior, e.g., individual malicious, unpredictable malicious and collective malicious. We considered the two metrics, the percentage of authentic transactions and percentage of rejections to evaluate the performance of the algorithm. We have shown that our algorithm performs better than the other existing algorithms in the literature. We have also shown through simulations that because peers are not competing with each other for higher value of global trust, the load on good peers is more uniform compared to the other methods described in the existing literature.

In Chapter 4, we generalized the proof of convergence of Absolute trust algorithm. We have shown mathematically that the error in global trust will converge to zero if it is calculated by iterative method as given in Chapter 3. The initial value of global trust can be very far off from the actual solution. In any step, convergence is faster if error is larger. We have also shown the dependency of speed of convergence on the parameter  $\alpha$ . With the help of suitable numerical example, we have given the justification for mathematical proof.

In Chapter 5, we studied the problem of unfairness and free-riding in P2P networks. We proposed a new algorithm - Biased Contribution Index (BCI). In this algorithm,

we defined the BCI of peers in the form of second order polynomial of BCI of other peers with which it interacted. We have shown that BCI can be calculated by iterative method, thus it can be implemented in a distributed system. With the help of mathematical justification and simulation results, we have shown that BCI can achieve the fairness and can suppress the free-riders in the network. We compared the speed of convergence of BCI with another existing approach in the literature, i.e., Global Contribution Approach [8]. We found that the speed of convergence of BCI is faster than that of the GC [8].

In Chapter 6, we studied the problems with iterative methods. As a solution, we presented simplified form of BCI named Simplified Biased Contribution index (SBCI). In SBCI, we rank the peers by assigning them the value of SBCI between 0 to 1. The estimation of SBCI is based on two factors, the resources contributed by the peer and the SBCI of the peers with whom it has transacted. We considered some basic design rules to achieve the fairness in network. With the help of mathematical justification we have shown that SBCI can achieve the objective of fairness in the network.

For the selection of peer for resource sharing, we proposed two methods, simple procedure, and the college admission and the stability of marriage approach. We also evaluated the SBCI through simulation. For evaluation we considered, the average absolute deviation (AAD) of upload to download ratio from unity and percentage of rejections among cooperative peers as the standard metric for evaluating. We have shown that the SBCI perform better than the other existing approach.

## 7.2 Major Contributions

The introduction of new concept of a center point of a non-negative irreducible matrix, as explained in Chapter 4, is a major contribution of the thesis. Based on this concept we introduced the Absolute trust algorithm and Biased Contribution Index (BCI). Both of these algorithms performed better than the existing one. Another major contribution of the thesis is the introduction of simplified form of BCI named Simplified Biased Contribution Index (SBCI). It is more practical to implement in the real P2P system.

## 7.3 Future Work

The ranking methods, which we presented here, are assumed to be implemented in an structured P2P networks and DHT algorithm is used to locate the peers. To investigate the algorithm when implementation is done in unstructured P2P networks, can be a good future work.

The ranking of peers are updated after certain number of transactions in the network. If we increase the frequency of update, the complexity of solution will increase and if we decrease it the accuracy of the solution is compromised. Thus, optimum value of frequency of update and how it can be maintained adaptively, needs further investigation.

In Chapter 6, for peer selection procedure, college admission and the stability of marriage based approach was introduced. The optimum number of heterogeneous peers, in which upload and download balance can be achieved, needs further attention.

---

Secure communication of trust value and contribution index need to be added in the model. The impact of security breaches on the performance need to be looked into before any realistic implementation.

# Bibliography

- [1] S. D. Kamvar, M. T. Schlosser and H. Garcia-Molina, “The EigenTrust Algorithm for Reputation Management in P2P Networks,” *Proc. of the 12th international conference on World Wide Web*, ser. WWW’03. New York, USA: ACM, pp. 640-651, 2003, .
- [2] R. Zhou and K. Hwang, “PowerTrust: A Robust and Scalable Reputation System for Trusted Peer-to-Peer Computing,” *IEEE Trans. Parallel and Distributed Systems*, vol. 18, no. 4, pp. 460-473, April 2007.
- [3] A. Simone, B. Skoric and N. Zannone, “Flow-Based Reputation: More than Just Ranking,” *International Journal of Information Technology and Decision Making*, vol. 11, no. 3, pp. 551-578, December 2012.
- [4] L. Page, S. Brin, R. Motwani and T. Winograd , “The PageRank Citation Ranking: Bringing Order to the Web,” Technical report, Stanford Digital Library Technologies Project, 1998.
- [5] P. Garbacki and D. H. J. Epema, “An Amortized Tit-For-Tat Protocol for Exchanging Bandwidth Instead of Content in P2P Networks,” *First Int’l Conf. Self-Adaptive and Self-Organizing Systems*, pp. 119-228, 2007.

- [6] M. Feldman, K. Lai, I. Stoica, and J. Chuang, “Robust Incentive Techniques for Peer-to-Peer Networks,” *Proc. Fifth ACM Conf. Electronic Commerce*, pp. 102-111, May 2004.
- [7] Q. Lian, Y. Peng, M. Yang, Z. Zhang, Y. Dai, and X. Li, “Robust Incentives via Multi-Level Tit-for-Tat,” *Concurrency and Computation: Practice & Experience*, vol. 20, pp. 167-178, 2008.
- [8] H. Nishida and T. Nguyen, “A Global Contribution Approach to Maintain Fairness in P2P networks” *IEEE Trans. Parallel and Distributed Systems*, vol.21, No.6, pp. 812-826, June 2010.
- [9] A. Sherman, J. Nieh, and C. Stein, “FairTorrent: A Deficit-Based Distributed Algorithm to Ensure Fairness in Peer-to-Peer Systems” *IEEE/ACM Trans. Networking*, vol. 20, no. 5, pp.1361-1374, October 2012.
- [10] J. J. D. Mol, J. A. Pouwelse, D. H. J. Epema, and H. J. Sips, “Give-to-Get: Free-Riding Resilient Video-on-Demand in P2P Systems,” *Proc. Multimedia Computing and Networking*, pp. 681804-1-681804-8, 2008.
- [11] D. Gale and L. S. Shapley, “College admissions and the stability of marriage,” *The American Mathematical Monthly*, vol. 69, no. 1, pp.9-15, January 1962.
- [12] L. Kleinrock, “An early History of the Internet,” *IEEE Communication Magazine*, vol. 48, no. 8, pp. 26-36, August 2010.
- [13] L. Kleinrock, “History of the Internet and its Flexible Future,” *IEEE Wireless Communication* , vol. 15, no. 1, pp. 8-18, February 2008.
- [14] B. M. Leiner, V. G. Cerf, D. D. Clark, R. E. Kahn, L. Kleinrock, D. C. Lynch, J. Postel, L. G. Roberts, S. Wolff, “Brief History

- of the Internet,” <http://www.internetsociety.org/internet/what-internet/history-internet/brief-history-internet>
- [15] L. Kleinrock Leonard, “Information Flow in Large Communication Nets”. RLE Quarterly Progress Report. Massachusetts Institute of Technology. July 1961.
- [16] L. Kleinrock, Communication Nets; Stochastic Message Flow and Delay, McGraw-Hill, New York, 1964.
- [17] Paloalto Networks, [Online]<http://researchcenter.paloaltonetworks.com/app-usage-risk-report-visualization>
- [18] Sandvine, Waterloo, Canada, “Sandvine June 2016 global Internet phenomena report,” 2016 [Online]. Available: <https://www.sandvine.com/trends/global-internet-phenomena/>
- [19] I. Stoica, R. Morris, D. Nowell, D. Karger, M. Kaashoek, F. Dabek and H. Balakrishnan, “Chord: A Scalable Peer-to-Peer Lookup Protocol for Internet Applications,” *ACM SIGCOMM Computer Comm. Rev.*, vol. 31, no. 4, pp. 149-160, 2001.
- [20] S. Ratnasamy, P. Francis, M. Handley, R. Karp and S. Shenker, “A scalable content-addressable network,” *Proc. of ACM SIGCOMM '01 conference on Applications, technologies, architectures, and protocols for computer communications*, pp. 161-172, August 2001.
- [21] A. Rowstron and P. Druschel, “Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems,” *Proc. of the Middleware'01 IFIP/ACM International Conference on Distributed Systems Platforms Heidelberg*, pp. 329-350, 2001.

- [22] B. Y. Zhao, L. Huang, J. Stribling, S. C. Rhea, A. D. Joseph and J. D. Kubiatowicz, "Tapestry: A resilient global-scale overlay for service deployment," *IEEE Journal on Selected Areas in Communications*, vol. 22, no. 1, pp.41-53, January 2004.
- [23] "Secure Hash Standard," U.S. Dept. Commerce/NIST, National Technical Information Service, Springfield, VA, FIPS 180-1, Apr. 1995.
- [24] R. Cuevas, M. Kryczka, A. Cuevas, S. Kaune, C. Guerrero, and R. Rejaie, "Unveiling the Incentives for Content Publishing in Popular BitTorrent Portals," *IEEE/ACM Trans. Networking*, vol. 21, no. 5, pp.1421-1435, October 2013.
- [25] VBS.Gnutella Worm. <http://securityresponse.symantec.com/avcenter/venc/data/vbs.gnutella.html>.
- [26] M. Karakaya, I. Korpeoglu, and O. Ulusoy, "Free riding in peer-to-peer networks," *IEEE Internet Comput.*, vol. 13, no. 2, pp. 92-98, March-April 2009.
- [27] E. Adar and B. A. Huberman, "Free riding on Gnutella," *First Monday* vol. 5, no. 10, Octbor 2000.
- [28] S. Saroiu, P. K. Gummadi, and S. 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.
- [29] eBay website. [www.ebay.com](http://www.ebay.com).
- [30] K. Aberer and Z. Despotovic, "Managing Trust in a Peer-to-Peer Information System," *Proc. of the 10th International Conference on Information and Knowledge Management (ACM CIKM)*, New York, USA, pp. 310-317, 2001.



- [31] L. Xiong and L. Liu, "Peertrust: Supporting Reputation-Based Trust for Peer-to-Peer Ecommerce Communities," *IEEE Trans. Knowledge and Data Eng.*, vol. 16, no. 7, pp. 843-857, July 2004.
- [32] S. Song, K. Hwang, R. Zhou and Y. K. Kwok, "Trusted P2P Transactions with Fuzzy Reputation Aggregation," *IEEE Internet Computing*, vol. 9, no. 6, pp. 24-34, November-December 2005.
- [33] R. Zhou, K. Hwang and M. Cai, "Gossiptrust for Fast Reputation Aggregation in Peer-to-Peer Networks," *IEEE Trans. Knowledge and Data Eng.*, vol. 20, no. 9, pp. 1282-1295, September 2008.
- [34] S. Boyd, A. Ghosh, B. Prabhakar and D. Shah, "Randomized Gossip Algorithms," *IEEE Trans. On Information Theory*, vol. 52, no. 6, pp. 2508-2530, June 2006.
- [35] Xiaoyong Li, Feng Zhou and Xudong Yang, "Scalable Feedback Aggregating (SFA) Overlay for Large-Scale P2P Trust Management," *IEEE Trans. Parallel and Distributed Systems*, vol. 23, no. 10, pp.1944-1957, October 2012.
- [36] Y. Wang and J. Vassileva, "Bayesian Network-based Trust Model in Peer-to-Peer Networks," *Proc. IEEE/WIC International Conference on Web Intelligence*, pp. 372-378, 2003.
- [37] E. Damiani, S. D. C. di Vimercati, S. Paraboschi and P. Samarati, "Managing and Sharing Servants' Reputations in P2P Systems, " *IEEE Trans. Knowledge and Data Eng.*, vol. 15, no. 4, pp. 840-854, July/August 2003.
- [38] E. Damiani, S.D.C. di Vimercati, S. Paraboschi, P. Samarati and F. Violante, "A Reputation-Based Approach for Choosing Reliable Resources in Peer-to-Peer

- Networks,” *Proc. Ninth ACM Conf. Computer and Comm. Security*, pp. 207-216, 2002.
- [39] Sergio Marti and Hector Garcia-Molina, “Taxonomy of Trust: Categorizing P2P Reputation Systems,” *Computer Networks*, vol. 50, no. 4, 15 Pages 472 - 484, March 2006.
- [40] A. Jsang, R. Ismail and C. Boyd, “A Survey of Trust and Reputation Systems for Online Service Provision,” *Decision Support Systems*, vol. 43, no. 2, pp. 618-644, 2007.
- [41] F. Hendriks, K. Bubendorfer and R. Chard, “Reputation systems: A survey and taxonomy,” *Journal of Parallel and Distributed Computing*, vol. 75, pp. 184-197, January 2015.
- [42] Wattana Viriyasitavat and Andrew Martin, “A Survey of Trust in Workflows and Relevant Contexts,” *IEEE Communication Surveys and Tutorials*, vol. 14, no. 3, pp. 911-940, THIRD QUARTER 2012
- [43] O. A. Wahab, J. Bentahar, H. Otrok and A. Mourad, “A survey on trust and reputation models for Web services: Single, composite and communities,” *Decision Support Systems*, vol. 74, Issue C, pp. 121-134, June 2015.
- [44] M. Feldman, C. Papadimitriou, J. Chuang, and I. Stoica, “Free-Riding and White-washing in Peer-to-Peer Systems,” *Proc. ACM SIGCOMM Workshop Practice and Theory of Incentives in Networked Systems (PINS)*, pp. 228-236, August 2004.
- [45] K. Eger and U. Killat, “Fair Resource Allocation in Peer-to-Peer Networks (Extended Version),” *Computer Comm.*, vol. 30, no. 16, pp. 3046-3054, November 2007.

- [46] H. Park and M. van der Schaar, "A Framework for Foresighted Resource Reciprocation in P2P Networks," *IEEE Trans. Multimedia*, vol. 11, no. 1, pp. 101-116, January 2009.
- [47] R. Izhak-Ratzin, H. Park and M. van der Schaar, "Online Learning in BitTorrent Systems," *IEEE Tran. on Parallel and Distributed system*, vol. 23, no. 12, pp. 2280-2288, December 2012.
- [48] J. Park and M. van der Schaar, "A Game Theoretic Analysis of Incentives in Content Production and Sharing Over Peer-to-Peer Networks," *IEEE Journal of Selected Topics in Signal Processing*, vol. 4, no. 4, August 2010.
- [49] Y. Chen, B. Wang, W. Sabrina Lin, Y. Wu, and K. J. Ray Liu, "Cooperative Peer-to-Peer Streaming: An Evolutionary Game-Theoretic Approach," *IEEE Trans. on Circuit and Systems for Video Technology*, vol. 20, no. 10, pp. 1346-1357, October 2010.
- [50] R. T. B. Ma, S. C. M. Lee, J. C. S. Lui and D. K. Y. Yau, "Incentive and Service Differentiation in P2P Networks: A Game Theoretic Approach," *IEEE/ACM Trans. Networking*, vol. 14, no. 5, pp. 978-991, October 2006.
- [51] B. Cohen, "Incentives build robustness in BitTorrent," presented at the 1st Workshop Econ. Peer-to-Peer Syst., Jun. 2003.
- [52] D. Levin, K. LaCurts, N. Spring, and B. Bhattacharjee, "BitTorrent is an auction: Analyzing and improving BitTorrent's incentives," in *Proceedings of the ACM SIGCOMM*, pp. 243-254, August 2008.
- [53] F. Wu and L. Zhang. Proportional response dynamics leads to market equilibrium. In ACM STOC, 2007.

- [54] Denis Serre, *Matrices Theory and Applications*, Springer-Verlag New York, Inc., 2002.
- [55] A. Jsang, R. Hayward and S. Pope, "Trust Network Analysis with Subjective Logic," *Australian Computer Society, Twenty-Ninth Australasian Computer Science Conference(ACSC2006)*, Hobart, Tasmania, Australia, vol. 48, January 2006.
- [56] Yan Wang and Lei Li, "Two-Dimensional Trust Rating Aggregations in Service-Oriented Applications," *IEEE Trans. On Service Computing*, vol. 4, no. 4, pp. 257-271, October/December 2011.
- [57] X. Wang, L. Liu and Jinshu Su, "RLM: A General Model for Trust Representation and Aggregation," *IEEE Trans. On Service Computing*, vol. 5, no. 1, pp. 131-143, January/March 2012.
- [58] S. Wang, Z. Zheng, Z. Wu, M. R.Lyu and F. Yang, "Reputation Measurement and Malicious Feedback Rating Prevention in Web Service Recommendation Systems," *IEEE Trans. On Service Computing*, vol. 8, no. 5, pp. 755-767, September/October 2015.
- [59] Ahmet Burak Can and Bharat Bhargava, "SORT: A Self-ORganizing Trust Model for Peer-to-Peer Systems," *IEEE Trans. On Dependable and Secure Computing*, vol. 10, no. 1, pp. 14-27, January/February 2013.
- [60] S. K. Awasthi and Y. N. Singh, "Generalized Analysis of Convergence of Absolute Trust in Peer to Peer Networks," *IEEE Communications Letters*, vol. 20, no. 7, July 2016.
- [61] E. Seneta, *Non-negative Matrices and Markov Chains*, 2nd ed. Springer-Verlog, New York Heidelberg Berlin, 1981.

- [62] Taher H. Haveliwala and Sepandar D. Kamvar, "The Second Eigenvalue of the Google Matrix," Technical report, Stanford University, 2003.
- [63] Z. Liang and W. Shi, "Analysis of ratings on trust inference in open environments," *Performance Evaluation*, vol. 65, no. 2, pp. 99-128, 2008.
- [64] Uri Wilensky, <https://ccl.northwestern.edu/netlogo/>, 2015.
- [65] Lada A. Adamic and Bernardo A. Huberman, "Zipf's law and the Internet," *Glottometrics* 3, 143-150, 2002.
- [66] Robert B. Ash, *Probability and Measure Theory*, 2nd ed. Chapter 2 pp. 84-85, Academic Press, 2000.
- [67] S. K. Awasthi and Y. N. Singh, "Absolute Trust: Algorithm for Aggregation of Trust in Peer to Peer Network", <http://arxiv.org/abs/1601.01419>
- [68] J. S. Otto, M.A. Sanchez, D. R. Choffnes, F.E. Bustamante, and G. Siganos, "On Blind Mice and the Elephant - Understanding the Network Impact of a Large Distributed System," *Proc. of ACM SIGCOMM, conference 2011*. pp. 110- 121, 2011.
- [69] S. K. Awasthi and Y. N. Singh, "Biased Contribution Index: A Simpler Mechanism to Maintain Fairness in Peer to Peer Networks," <https://arxiv.org/pdf/1606.00717.pdf>, June 2016.
- [70] V. S. Borkar, R. Makhijani and R. Sundaresan, "Asynchronous Gossip for Averaging and Spectral Ranking," *IEEE Journal Of Selected Topics in Signal Processing*, vol. 8, no. 4, pp. 703-716, August 2016.
- [71] Matei Ripeanu, Adriana Iamnitchi, and Ian Foster, "Mapping the Gnutella Network: Properties of Large-Scale Peer-to-Peer Systems and Implications for System

- Design,” *IEEE Internet Computing Journal special issue on peer-to-peer networking*, vol. 6(1), 50-57, January/February 2002.
- [72] S. Saroiu, K. P. Gummadi, and S. D. Gribble, “Measuring and analyzing the characteristics of napster and gnutella hosts,” *Multimedia Syst.*, vol. 9, no. 2, pp. 170-184, Aug. 2003.
- [73] E. K. Lua, J. Crowcroft, M. Pias, R. Sharma, and S. Lim, “A survey and comparison of peer-to-peer overlay network schemes,” *IEEE Commun. Surveys Tutorials*, vol. 7, no. 2, pp. 72-93, 2005.
- [74] C. T. Kelley, *Iterative Methods for Linear and Nonlinear Equations*, Society for Industrial and Applied Mathematics, Philadelphia, 1995.
- [75] H. Zhang, A. Goel and R. Govindan, “Making Eigenvector-Based Reputation Systems Robust to Collusion,” *Proc. Third Workshop Economic Issues in P2P Systems*, pp. 92-104, June 2003.
- [76] E. Khosrowshahi Asl, J. Bentahar, H. Otrok, R. Mizouni, “Efficient Community Formation for Web Services,” *IEEE Trans. On Service Computing*, vol. 8, no. 4, pp. 586-600, July/August 2015.
- [77] H. Park, and M. van der Schaar, “On the Impact of Bounded Rationality in Peer-to-Peer Networks,” *IEEE Signal Processing Letters*, vol. 16, no. 8, pp.675-678, August 2009.
- [78] H. Park, and M. van der Schaar, “Evolution of Resource Reciprocation Strategies in P2P Networks,” *IEEE Trans. on Signal Processing*, vol. 58, no. 3, pp. 1205-1218, March 2010.

- [79] R. Gupta and Y. N. Singh, "Reputation Aggregation in Peer-to-Peer Network Using Differential Gossip Algorithm," *IEEE Trans. Knowledge and Data Eng.*, vol. 27, no. 10, pp. 2812-2823, October 2015.
- [80] S. K. Awasthi and Y. N. Singh, "Simplified Biased Contribution Index (SBCI): A Mechanism to Make P2P Network Fair and Efficient for Resource Sharing," <https://arxiv.org/pdf/1702.07992.pdf>, February, 2017.
- [81] T. Chen, F. Wu and S. Zhong, "FITS: A Finite-Time Reputation System for Cooperation in Wireless Ad Hoc Networks," *IEEE Trans. on Computers*, vol. 60, no. 7, pp. 1045-1056, July 2011.
- [82] G. Liu, H. Shen and L. Ward, "An Efficient and Trustworthy P2P and Social Network Integrated File Sharing System" *IEEE Trans. on Computers*, vol. 64, no. 1, pp. 54-70, January 2015.
- [83] S. K. Awasthi and Y. N. Singh, "Reputation System and Incentive Mechanism to Handle the Malicious Peers and Free-riders in Peer-to-Peer Networks," *ICEIT MCNC 2017*, 16-17 Feb 2017, New Delhi, pp.112-116.

# List of Publications

## Journals:

1. **S. K. Awasthi** and Y. N. Singh, "Absolute trust: Algorithm for aggregation of trust in peer-to-peer networks." *IEEE Trans. on Service Computing* (Revision Submitted)
2. **S. K. Awasthi** and Y. N. Singh, "Generalized analysis of Absolute Trust in peer-to-peer Networks" *IEEE Communications Letters*, vol. 20, no. 7, pp. 1345-1348, July 2016.
3. **S. K. Awasthi** and Y. N. Singh, "Biased Contribution Index: A Simpler Mechanism to Maintain Fairness in Peer to Peer Networks" *IEEE Trans. on Communications* (Submitted for Review)
4. **S. K. Awasthi** and Y. N. Singh, "Simplified Biased Contribution Index (SBCI): A Mechanism to Make P2P Network Fair and Efficient for Resource Sharing" *IEEE Trans. on Mobile Computing* (Submitted for Review)

## Conferences:

1. Anurag Dwivedi, **S. K. Awasthi**, Ashutosh Singh and Y. N. Singh, "Flash Crowd Handling in P2P Live Video Streaming Systems," *ICEIT MCNC 2015*, 16-17 April 2015, New Delhi.



2. **S. K. Awasthi** and Y. N. Singh, “Reputation System and Incentive Mechanism to Handle the Malicious Peers and Free-riders in Peer-to-Peer Networks,” *ICEIT MCNC 2017*, 16-17 Feb 2017, New Delhi, pp.112-116. (**Given Best Conceptual Paper Award**)