



MITIGATION OF FREE RIDING IN
PEER-TO-PEER SYSTEMS

BY

MOHAMMED ONIMISI YAHAYA

A Dissertation Presented to the
DEANSHIP OF GRADUATE STUDIES

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the
Requirements for the Degree of

DOCTOR OF PHILOSOPHY

In

COMPUTER SCIENCE AND ENGINEERING

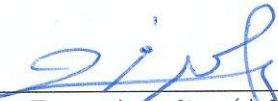
May 2014

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS
DHAHRAN 31261, SAUDI ARABIA


DEANSHIP OF GRADUATE STUDIES

This thesis, written by **MOHAMMED ONIMISI YAHAYA** under the direction of his thesis advisor and approved by his thesis committee, has been presented to and accepted by the Dean of Graduate Studies, in partial fulfillment of the requirements for the degree of **DOCTOR OF PHILOSOPHY IN COMPUTER SCIENCE AND ENGINEERING**.


Dissertation Committee



Dr. Farag Azzedin (Advisor)



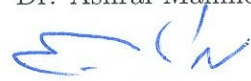
Dr. Moataz Ahmed (Member)




Dr. Nasir Darwish (Member)




Dr. Ashraf Mahmoud (Member)



Dr. Shokri Selim (Member)



Dr. Umar Al-Turki
Department Chairman



Dr. Salim A. Zummo
Dean of Graduate Studies

6/5/14

Date



Mohammed Onimisi Yahaya
2014

Dedication

In memory of my late sister Aisha Jimoh bint Yahaya Osuwa

ACKNOWLEDGMENTS

All praise is due to Allah, the lord and sustainer of the worlds, for his countless favour and seeing me this far in life. I appreciate the support and prayers of my parents, Alh. Yahaya Osuwa and Hajiya Sefinat Osuwa all through my life and especially during this work. To my siblings, I say thank you all for being there for me.

My profound gratitude goes to my advisor, Dr Farag Azzedin for his constructive criticism, guidance and especially the assistance he offered me during the writing process. I thank all my committee members, Dr Moataz Ahmed, Dr Ahsraf Mahmoud, Dr Nasir Darwish and Prof Shokri Selim for their support and comments. Finally, I appreciate the advice and effort of Dr Slim Belheiza of Mathematics Department towards the success of this work.

My special thanks go to my dear wife, Bilkisu Ohunene and our children, Ammatullah Onize, Abdulmuhaimin Onoruoiza and Aisha Ahuoiza for their love, care, understanding and patience throughout the entire PhD program.

To my friends and all the Nigerian and African Community in KFUPM, I wish you all the best the life and hereafter has to offer. I specifically appreciate the effort of my friends and brothers, Hamza Salami, Mohammed Adinoyi, Abdulrauf, Zakaria Jamiu and Adeniran for their time and technical assistance.

TABLE OF CONTENTS

ACKNOWLEDGMENTS	v
LIST OF TABLES	x
LIST OF FIGURES	xi
ABSTRACT (ENGLISH)	xiii
ABSTRACT (ARABIC)	xv
CHAPTER 1 INTRODUCTION	1
1.1 Distributed Systems	2
1.2 Free Riding	7
1.2.1 Free Riding in Distributed Systems	9
1.3 Peer-to-Peer Systems	13
1.3.1 Types of Peer-to-Peer Systems	16
1.3.2 Challenges of Peer-to-Peer Sharing	19
1.3.3 Free Riding in P2P Systems	20
1.3.4 Effects of Free riding in P2P networks	21
1.4 Motivation and Research Problem	23
1.5 Objectives	24
1.6 Contributions	25
1.7 Thesis Overview	26

CHAPTER 2 LITERATURE SURVEY	28
2.1 Overview	28
2.2 Incentive Based Schemes	29
2.3 Reciprocity Based Schemes	33
2.4 Behavioral Based Schemes	36
2.5 Game Theory Based Schemes	41
CHAPTER 3 GAME THEORY	54
3.1 Overview	54
3.2 Game Basics	55
3.3 Game Descriptions	56
3.4 Game Examples	58
3.4.1 Prisoner’s Dilemma	58
3.4.2 Stag Hunt	59
3.4.3 Battle of Sexes	60
3.4.4 Matching Pennies	61
3.5 Game Taxonomy	62
3.5.1 Player-based Game Classification	62
3.5.2 Action-based Game Classification	70
3.5.3 Outcome-based Game Classification	71
3.5.4 Repetition-based Game Classification	73
3.5.5 Termination-based Game Classification	74
3.6 Game Solution Concepts	77
3.6.1 Dominant Strategies	77
3.6.2 Nash Equilibrium	79
CHAPTER 4 BITTORRENT SYSTEM	82
4.1 File Announcement	83
4.2 File Downloading	84
4.2.1 Metadata discovery	84
4.2.2 Peer discovery	86

4.2.3	File dissemination	87
4.3	Peer status update	88
4.4	Illustrative Example of File Sharing	90
4.5	Piece Selection Strategy	91
4.6	Choking Algorithm	93
4.6.1	Regular Unchoke	95
4.6.2	Optimistic Unchoke	102
CHAPTER 5 CHOKING ALGORITHM GAME MODELLING		105
5.1	Overview	105
5.2	Motivation	106
5.3	Motivation for Using Game Theory in Modelling BitTorrent Choking Algorithm	107
5.4	The Game Model	109
5.4.1	Game Model For Initial Seeder	111
5.4.2	Game Model For Leecher	112
5.4.3	Game Model For Experienced Seeder	114
5.4.4	Game Model Every Thirty Seconds	115
5.5	Model Implementation	116
CHAPTER 6 PERFORMANCE EVALUATION		127
6.1	Overview	127
6.2	Performance Metrics	128
6.3	The Simulation Model	129
6.3.1	PeerSim	129
6.3.2	BitPeer	130
6.4	Simulating Free riding in BitTorrent	134
6.5	Simulation Setup	134
6.6	BitTorrent Vulnerability	136
6.6.1	BitTorrent Exploitation	136
6.6.2	Exploitation Root Causes	140

6.7	Game Model Effectiveness	142
6.7.1	Game Model Fairness	143
6.7.2	Game Model Robustness	145
6.7.3	Game Model Agility	147
CHAPTER 7 CONCLUSIONS AND FUTURE DIRECTIONS		154
7.1	Conclusions	154
7.2	Summary of Thesis Contributions	156
7.3	Future directions	157
Bibliography		160
VITAE		185

LIST OF TABLES

1.1	Characteristics of Distributed Systems	7
1.2	Free Riding in Distributed Systems	12
3.1	Payoff matrix for prisoner P_1 and prisoner P_1	56
3.2	Payoff matrix of Prisoners Dilemma	59
3.3	Payoff matrix of stag hunt game	60
3.4	Payoff matrix of Battle of Sexes Game	61
3.5	Payoff matrix of Matching Pennies Game	62
3.6	Example of Players-related Classification of games	75
3.7	Example of Actions-related Classification of games	77
3.8	Example of Moves-related Classification of games	77
3.9	Example of Outcome-related Classification of games	78
3.10	Payoff matrix of Prisoners Dilemma	79
3.11	Payoff matrix of Battle of Sexes Game	81
4.1	BitTorrent messages	90
5.1	Game Model: Initial Seeder to an opponent	112
5.2	Game Model: Leecher to Leecher	114
5.3	Game Model: Leecher to Freerider	114
5.4	Game Model: Experienced Seeder to Leecher	115
5.5	Game Model: Experienced Seeder to Free rider	115
5.6	Number of blocks uploaded by an IS to its neighbors	118
6.1	Simulation design parameters	135

LIST OF FIGURES

1.1	Examples of Distributed Systems.	4
2.1	Types of incentives in P2P system.	29
2.2	Direct and Indirect Reciprocity.	33
2.3	Direct and Indirect Reputation.	37
3.1	Extensive form representation of matching pennies game.	58
3.2	Classification of game.	62
3.3	Player-based classifications of game	63
3.4	Player-identity effect classifications of game.	64
3.5	Number of player classifications of game.	65
3.6	Nature of player classifications of game.	68
3.7	Information availability to player classifications of game.	70
3.8	Action-based classifications of game.	72
3.9	Outcome-based classifications of game.	73
3.10	Repetition-based classifications of game.	74
3.11	Termination-based classification of game.	76
4.1	Phases involved in uploading or downloading a file in BitTorrent.	83
4.2	File announcement steps.	85
4.3	An Example Metadata.	85
4.4	Metadata discovery steps.	86
4.5	Message interleaving during file dissemination.	89
4.6	BitTorrent file sharing process.	92

4.7	Choking algorithm time intervals.	94
4.8	Choking algorithm classification.	94
4.9	Snubbing algorithm time intervals.	98
4.10	Regular unchoking steps by Seeders.	101
4.11	Regular unchoking steps by Leechers.	101
4.12	Regular and Optimistic unchoking time intervals.	104
4.13	Optimistic unchoking steps.	104
6.1	A PeerSim sample configuration file.	131
6.2	Using 5% SEs and 30% FRs: Conversion rate of FRs and LEs. . .	136
6.3	Using 5% SEs and 70% FRs: Conversion rate of FRs and LEs. . .	137
6.4	Using 10% SEs and 30% FRs: Conversion rate of FRs and LEs. . .	138
6.5	Using 10% SEs and 70% FRs: Conversion rate of FRs and LEs. . .	138
6.6	Using 5% SEs: Download time for FRs and LEs.	139
6.7	Using 5% seeders and 30% FRs: Unchoke messages sent by seeders.	140
6.8	Using 5% seeders and 30% FRs: Unchoke messages sent by LEs. . .	141
6.9	Using 5% seeders and 70% FRs: Unchoke messages sent by seeders.	141
6.10	Using 5% seeders and 70% FRs: Unchoke messages sent by LEs. . .	142
6.11	Using 5% ISs and 30% FRs: Conversion rate of FRs and LEs. . .	143
6.12	Using 5% ISs and 70% FRs: Conversion rate of FRs and LEs. . .	144
6.13	Using 10% ISs and 30% FRs: Conversion rate of FRs and LEs. . .	144
6.14	Using 10% ISs and 70% FRs: Conversion rate of FRs and LEs. . .	145
6.15	Using 5% ISs: Download time for FRs.	146
6.16	Using 5% ISs: Download time for LEs.	146
6.17	Using 5% ISs and 30% FRs: Unchoke messages sent by ISs.	147
6.18	Using 5% ISs and 30% FRs: Unchoke messages sent by ESs.	148
6.19	Using 5% ISs and 30% FRs: Unchoke messages sent by LEs.	149
6.20	Using 5% ISs and 70% FRs: Unchoke messages sent by ISs.	149
6.21	Using 5% ISs and 70% FRs: Unchoke messages sent by ESs.	150
6.22	Using 5% ISs and 70% FRs: Unchoke messages sent by LEs.	150

THESIS ABSTRACT

NAME: Mohammed Onimisi Yahaya
TITLE OF STUDY: Mitigation of Free Riding in Peer-to-Peer Systems
MAJOR FIELD: Computer Science and Engineering
DATE OF DEGREE: May 2014

Distributed systems are collection of autonomous computers, connected through a network and distribution middleware, which enables computers to coordinate their activities and to share the resources of the system, so that users perceive the system as a single, integrated computing facility. A peer-to-peer system is a form of distributed system in which all participating nodes have "equal standing", both serving as either a client or server. The design principles of peer-to-peer systems led to some challenges. One of these challenges is the problem of free riding. Free riding has been identified as a threat to the existence of peer-to-peer systems, as it negates the fundamental nature of peer-to-peers systems. Researchers are currently still designing techniques to curb free riding and therefore, there is a need for robust solution to this complex problem. In this dissertation, we investigate distributed systems in order to clearly classify them according to their character-

istics. We also examine each of the distributed systems and identify those that are prone to free riding. Consequently, we proposed a taxonomy for distributed to clearly identify them. In addition, we propose a new free riding based taxonomy for distributed systems. Furthermore, we specifically focused on the problem of free riding in peer-to-peer systems and investigate why it is very challenging. We conducted a review of existing approaches to combating free riding in peer-to-peer systems and identify research gaps between the existing approaches.

As a case study, we conducted an in depth study of BitTorrent system, one of the most popular peer-to-peer file sharing system. In this phase of the thesis, we investigated BitTorrent peer selection algorithms by carrying out experimental analysis to understand BitTorrent exploitation points by free riders. Based on the understanding gained from the analysis, we proposed a game theory-based choking algorithm for BitTorrent to deter free riding. Finally, we carried out extensive simulation to evaluate the performance of our proposed model and compared our results with the existing BitTorrent. The results show that our proposed game based choking algorithm out performed the existing choking algorithm in BitTorrent.

CHAPTER 1

INTRODUCTION

Peer-to-Peer (P2P) systems are experiencing a significant amount of attraction due to the current technological advancements in communication, computation, and storage [1, 2, 3]. The widespread and the success of P2P systems are due to the nature and benefits of collaboration. To name few of these benefits, collaboration nurtures cross-boundary sagacious solutions, increases the overall work done by the collection of shared resources, and creates an environment for mutual gains [1, 2, 3]. This collaborative effort will not succeed if peers start to consume resources more than they contribute. Non-collaborative peers, who do not contribute to the community, are referred to as *free riders* in the literature [1, 4, 5, 6]. The phenomenon of free riding does not just create a tedious task that can be bothersome for the contributors, but also can create a potential performance bottleneck. As a consequence, responses to queries can experience a longer delay. In a collaborative file sharing system, 66% of users do not share files (i.e., free riders) and the top 25% users account for 99% of all downloads [7].

This chapter is organized as follows. Section 1.1 proposes a taxonomy of distributed systems that we argue is essential to distinguish between P2P and other distributed systems while section 1.2 examines the problem of free riding in distributed systems in general. In section 1.4, we clearly define the problem statement examined in this thesis as well as the motivation behind choosing such a problem. Finally, section 1.7 outlines the organization of the thesis.

1.1 Distributed Systems

The need for wide-scale applications, consisting of large number of devices and users, is emerging due to advances in computing and technology [8, 9, 10] such as advances in mobile devices and big data analytics platforms. To support the design and implementation of such applications, distributed systems are needed since centralization is becoming unfeasible due to a number of unavoidable factors:

The client devices increases year over time and thus creating an increasingly complex end systems

The data collected continues to grow exponentially with the use of advanced technology

The analysis of such data is becoming imperative

The requirements of ongoing create a demand for agile and flexible systems

The cost of scaling is becoming too high to handle from both an administrative and infrastructure sides

Single point of failure is unacceptable in an era where end users don't tolerate downtime

The inherent characteristics of reliability, availability, performance, scalability, cost reduction, and non-centralized nature play a vital role in promoting distributed systems [8, 9, 10]. As defined by Tanenbaum, a distributed system is a collection of independent computers that appear to its users as a single coherent system [8]. A distributed system can also be viewed as a collection of autonomous computers, connected through a network and distribution middleware, which enables these computers to coordinate their activities and to share the resources of the system in such a way that users perceive the system as a single, integrated computing facility [11]. Distributed systems have been classified based on interconnection network, memory access, and Operating System [8][11]. As such, distributed systems have been classified as: (a) tightly coupled where processors share clock and memory, run a single operating system, and communicate frequently and (b) loosely coupled, where each processor has its own memory, runs its own operating system, and communicate infrequently.

Examples of distributed systems as shown in figure 1.1 include client-server systems, cluster systems, Grid systems, P2P systems, Cloud systems and volunteer systems. A cluster is a type of parallel or distributed processing system consisting of a collection of interconnected stand-alone computers cooperatively working together as a single integrated computing resource and are located in one geographical location [12]. A Grid computing system is a dependable and coordi-

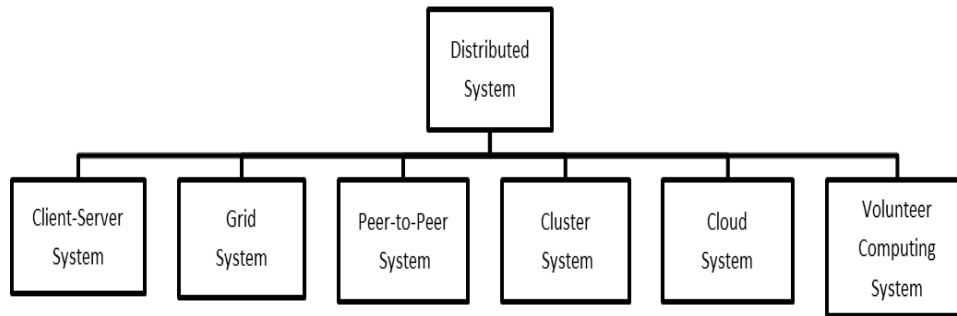


Figure 1.1: Examples of Distributed Systems.

nated collection of computing infrastructure that provide high end computational capabilities [13][14]. P2P systems consist of interconnected nodes able to self organize with the purpose of sharing resources without requiring the intermediation or support of a global centralized server or authority [15][16]. A volunteer system is a voluntary-based initiative to harness a vast unused resources to achieve computational intensive scientific projects such as SETI@home [17], Einstein@home [18] and ABC@home [19]. A Cloud system is a paradigm that provides massive computing infrastructure on pay as per use basis, accessible via the Internet [9]. The resources provided by a Cloud system are metered, hence, this technology is based primarily on economy of scale, so that small organization could rent computing facility and pay as per use other than investing in expensive computing infrastructure.

With the emergence and the popularity of such loosely coupled distributed systems (i.e., Grid systems, P2P systems, Cloud systems, volunteer systems), there is a need for a classification to understand and shed light on the specific

characteristics of such systems. To the best of our knowledge, this is the first attempt for such a classification. Furthermore, our classification is applicable to both tightly and loosely coupled distributed systems.

We provide a node-based taxonomy, where distributed systems are classified according to *Node Ownership*, *Node Controlling Policy*, *Node Management*, and *Node Discovery Mechanism*. Distributed system nodes are either all owned by a global entity or each node is locally owned. Examples of distributed systems that have local node ownership are client-server systems, Grid systems, P2P systems, and volunteer computing systems. In a server-client environment, the server as well as the clients are locally owned. Similarly, nodes in a Grid, a volunteer computing or P2P environments are locally owned. In such environments, there is no global entity that owns all nodes in the environment. On the other hand, distributed systems such as cluster systems and Cloud systems, a global entity owns all the participating nodes.

Node controlling policy refers to a set of rules controlling a node. These policies are necessary for proper functionality of the system. The entity that sets the controlling policy can be either a node itself or a system-wide global entity. In distributed systems such as client-server, Grid and volunteer computing, each node has a single controlling policy that is set locally. As such, each node including the server has its own controlling policy. Systems such as P2P and volunteer computing have no controlling policies while systems such as cluster computing have global entity that sets the controlling policy for the whole system.

In cloud computing, every node has local multiple controlling policies since a node is virtually partitioned into virtual machines and each of these virtual machines has its own controlling policy.

Node management is concerned with the entity that manages the node. A node might manage itself. In this case, the node is managed locally. Examples of distributed systems that are managed locally are client-server systems and P2P systems. On the other hand, there might exist a global entity that manages the node. In such a case, we say that the node is managed globally. In distributed systems such as cluster systems, a global entity manages all the participating nodes. In systems such as Grid, Cloud and volunteer systems, there exists a global entity managing the local nodes but this global management is done under the local policies.

In a distributed system, the process by which a node becomes aware of other nodes is referred to node discovery. The node discovery mechanism to find a participating node in the network may be centralized, distributed or none. In systems such as client-server, cluster and volunteer computing, there is no discover mechanism. Nodes in these systems do not need to discover others. The rest of the distributed system might have a centralized or distributed mechanism for node discovery. As a summary, Table 1.1 classifies the distributed systems according to our taxonomy.

Table 1.1: Characteristics of Distributed Systems

Distributed Environments	Node Ownership	Node Controlling Policy	Node Management	Node Discovery Mechanisms
Client-Server Systems	local	local single controlling policy	local	none
Cluster Systems	global(single ownership)	global single controlling policy	global	none
Grid Systems	local	local single controlling policy	global management under local policies	centralized or distributed
Peer-to-Peer Systems	local	none	local	centralized or distributed
Cloud Systems	local	local multiple controlling policies	global management under local policies	centralized or distributed
Volunteer Computing Systems	local	local single controlling policy	global management under local policies	none

1.2 Free Riding

The burden of giving should not be on the shoulders of specific users. These altruistic users might be very few and this will create bottlenecks resulting in degrading the *quality-of-service* (QoS). As such, if users do not contribute, applications such as distributing multimedia content over the Internet may not survive. Also, burdening the collaborative users with huge requests might force them to be non-collaborative. Furthermore, non-collaborative users negate the advantages of distribution and hence convert a distributed architecture to few scattered centralized nodes. Free riding is consuming more resources than one's contribution [20, 21, 22, 23, 24, 25].

Free riders are peers with selfish behavior with the intent of taking advantages

of other peers in the network. Free riding peers devise means of utilizing the resources of other peers in the network, so as to contribute less or nothing to other peers. Free riders can also share malicious content so as to deceive the environment that they are contributing. Also, free riders can deliberately upload useless files that do not benefit others, in order to hide under that cover for the purpose of downloading files [26].

The free riding problem is not unique to distributed environments alone. The problem of free riding emerged from Economics with regards to public goods. These are goods that are both "non-excludable" and "non-rival". A non-excludable resource means that an individual user cannot effectively exclude others from using it. Non-rival resources are type of resources that their use by an individual does not reduce the availability to others. Some examples of such goods are fresh air, knowledge and street lighting. However, if there is ability to impose restrictions on these goods they become private goods. Economists have studied the free riding problem [27, 28] and *the tragedy of the common* [29], a situation where some selfish individuals refrain from contributing to the common good. Some familiar examples in our society are, overuse of public resources such as over fishing in the deep ocean, pollution of the environment and excessive use of pesticides.

Similarly, shared computer resources in distributed systems have been compared to public and private goods in Economics. In [6], the authors pointed out that the fundamental difference in the characteristics of public goods in Economics

from that of information goods in distributed systems is that, while that the former is non-rival, the latter is affected by the fact that the provider of the content also participate in the sharing and usage of the same resource. Thus, any deficit from both producer and consumer will definitely affect the quality and quantity of the goods as well as overall performance of the system. For instance, sharing in P2P systems is an act similar to the private provision of public goods [30].

1.2.1 Free Riding in Distributed Systems

The problem of free riding in distributed systems has not been largely explored. Most of the earlier work in literature on free riding focuses on P2P system alone. In this section, we investigate further free riding in all distributed systems, so as to find out in addition to P2P networks, which other distributed environment is prone to free riding and what are the characteristics that give rise to the problem if any.

Generally, free riding is consuming resources more than providing resources. In the quest to fully understand free riding in distributed systems, there is a need for a critical analysis of this definition. This definition highlights resource provision as the key element. Each distributed system is made up of service providers and service consumers. They use resources to provide and consume services. These services include computational services, storage services, file sharing services, telephony services and video streaming services. To be able to provide these services, some or all of computer resources such as files, CPU time, storage devices, mem-

ory locations and network bandwidth are utilized. For instance, in a file sharing service, the primary resource needed to provide this service is files.

In view of these, we assert that, if the resources shared to provide the service by any of the distributed systems have public-subscription, then the system is prone to free riding. Otherwise, if resource provision is closed, free riding is not possible. Based on this assertion, we examine and classify distributed systems based on service and resource provision. The summary of this comparison is presented in Table 1.2. We now classify each of the distributed systems with respect to their services and resources provision to determine the possibility of existence of free riding.

Client-Server System: In a client-server system, the resources utilized to provide the service may be public or private. Free riding may occur in this system depending on the scenario of resource provision. We explain in detail each of these cases.

- **Case 1:** Consider a client-server system where clients provide the resources and the server provides the service. As such, the resource provision is public, hence free riding is always possible in this system.
- **Case 2:** Consider a client-server system where the server provides the resources and the services. In this scenario, the existence of free riders is impossible. This is due to the fact that the resource provision is private.
- **Case 3:** Consider a client-server system where the clients and the

server provide the resources while the server provides the service. In this scenario, free riding exist as the provision of resources is both private and public.

Peer-to-Peer System: There are two cases in P2P systems.

- **Case 1: Public P2P system :** In this type of network, both the service and the resources are being provided by any peer in the system. In this case, free riding is always possible.
- **Case 2: Closed P2P System:.** Peers in this system provide both service and resources but the system does not have public-subscription. An example of such system is Skype [31]. Therefore, such a system is free from free riding.

Grid System: Grid is a coordinated hardware and software infrastructure that provides dependable, consistent high end computational power [13]. By this definition, service and resource provision is coordinated and dependable, hence free riding is not possible.

Cluster Systems: In this system, both resources and services are private and under direct control of system owners. Users that subscribe to these system use and adhere to control and management policy set by the owners, hence free riding does not exist in cluster systems.

Cloud Systems: Here, both resources and services are metered by the provider which is private and under direct control of system provider. Users

Table 1.2: Free Riding in Distributed Systems

Distributed Environments	Environments	Resource and Service Provision	Comments
Client-Server Systems		Clients provide resources and server provides service	Free riding always exists.
		Server provides resources and services.	No free riding
		Clients and server provide resources while server provides service.	Free riding sometime exists
Cluster Systems		Resources and services provided by one entity (owner)	No free riding
Grid Systems		Resources and services coordinated and monitored	No free riding
Peer-to-Peer Systems		Both resources and services provided by any peer	Free riding always exists
		Resources and services may be provided by monitored peers	No free riding
Volunteer Computing Systems		Free will contributions	Free riding not a concern
Cloud Systems		Resources and services provided by one entity (owner)	No free riding

that subscribe to this system use and adhere to control and management policy set by the owners, hence free riding does not exist in both systems.

Volunteer Computing System: The resources in this system are provided by the public, but the service and the system is managed by a global entity. Free riding is not a concern in this form of distributed system, since the whole system is based on voluntary contributions from individuals. In this case, resources provided by a donor node is not expected to be dependable and continuous in supply. The system is being coordinated using public resources.

1.3 Peer-to-Peer Systems

Having identified P2P systems as one of the distributed systems with problem of free riding, we concentrate specifically on the features of P2P systems. We discuss the general challenges of sharing in P2P system and focus on the free riding problem and its effects in P2P systems. P2P systems have received significant attention in recent years, due to the advancement in the world of information technology. P2P systems have inspired the design of social networking sites that made large scale interaction of people and businesses possible. The availability of new forms of P2P paradigm such as B2C (Business to Consumer), B2B (Business to Business), B2E (Business to Employee) and B2G (Business to Government) has made simultaneous exchange of resources among numerous users through the traditional client-server model almost impossible. The severe limitations of the client-server model in terms of memory, bandwidth, storage and CPU, led to the current rise in the utilization of P2P systems for storage sharing, processing power sharing and large scale content distribution such as video streaming, file sharing and music sharing. This increase in the use of P2P systems also comes with its challenges, which has made it an area of current research. Most of the earlier research efforts on P2P systems concentrated on the design of efficient search mechanisms, improving resource indexing methodology, performance and scalability issues. The intent of P2P systems for large scale content distribution, communication, distributed computation and collaboration is being threatened by the problem of free riding.

A P2P system is described as a system that relies on computing power and bandwidth of nodes at the ends of a connection rather than concentrating on low number of servers within the network [32]. In [33], the authors defined P2P systems as any network that exhibits the following characteristics: distributed control, self organized and symmetric communication. There are many types of P2P systems, mostly used for large scale content distribution, file sharing, platform sharing, communication, distributed computation and collaboration. The concept of P2P systems is different from that of client-server model in the design principles. Researchers in [34] identified the principles of P2P as principle of sharing, decentralization and self organization. In addition, the authors in [35] identified anonymity, autonomy and open nature as principles of P2P systems. In order to provide a better understanding of P2P paradigm, we list the design principles of P2P systems as follows :

Equal standing : This is a fundamental design principle of any P2P system. It requires that every peer in the networks acts as a servant, that is either as a client or server. This principle, exemplify the P2P paradigm as an alternative to the traditional client/server model which uses servers for service provision, monitoring and access. P2P systems eliminate the need for central servers which are known to constitute a bottleneck to the expansion of the networks as well as creating a single point of failure. Peers' equal standing gives rise to some peculiar characteristics exhibited by these systems due to lack of central control. For instance, high degree of entry

and exit, high degree of autonomy from central server.

Autonomy: Autonomy of an entity refers to the impossibility of external control in an interaction [36]. This freewill to do anything without external influence includes mobility-the will to join and leave the networks at any time. Resource sharing is based on voluntary collaboration, therefore, peers have the freedom to share or not to share, to be selfish or generous. The leeway for peers to change identity, the decision to be trustworthy or untrustworthy, honest or liar; all these actions are under the control of an autonomous peer.

Anonymity : This refers to the desire of a peer in a network to be publicly unknown. In most P2P systems pseudonyms are used for access rather than full authentication. In reality, pseudo-anonymity is achieved with the use of pseudonyms as in the case of ID, bank account number, nickname etc.

Self organizing : P2P systems require that the architecture should adjust and organize itself if a new peer joins or leaves the networks. The rapid growth in P2P is due to its resiliency in absorbing expansion and shrinking due to its self organizing nature. A self organizing network should automatically adapt to entry, exit and failure of nodes [37]

Some level of decentralization: This design principle excludes the need for architectures that require central management. P2P systems eliminate completely or partially the need for central server. The level of decentraliza-

tion depends on the design objective, application and performance requirements. But zero-level decentralization as in client-server model negates the principle of P2P systems.

1.3.1 Types of Peer-to-Peer Systems

In general, there are several classifications of Peer-to-Peer networks [15, 1] such as classification based on structure of the network, degree of centralization, level of access and type of service provided. Structural classifications of the network depends on how the nodes on the over lay networks are connected together. In the following subsection, we present the structured based classification.

Unstructured Peer-to-Peer Systems

In unstructured P2P networks, there is no structure in the overlay network, no specific criteria and algorithm in the arrangement of the nodes. P2P overlay networks comprises of a participating nodes in the networks. There exist a link between any nodes that knows each other. It is a logical networks that are based on ad hoc connections. The overlay networks in unstructured P2P are formed arbitrarily in a flat or hierarchical manner. There is no single point of failure and requires no administrative efforts. For instance a link exist between any two nodes that has a TCP connection. In order to retrieve as many contents as possible, searching in this model is done through different methods such as flooding, random walking [38] and expanding ring(e.g Time to Live in Gnutella) [39]. There are three types of unstructured P2P networks, depending on the degree of centralization.

1. **Pure Unstructured P2P:** In this model, all peers acts as client and servers with no central authority. Communication and exchange of resources between peers are random since there is no central coordination. Search requests from peers are passed to neighbors, if direct neighbor does not have the resource it passed it on until a request hit based on a predefined search depth. For instance, Time-to-Live in Gnutella. When there is a search match for the requested resource, the requesting peer download directly from the peer that has the resource for offer. The overlay network in this model is flat with a single routing layer. There is no central look up server, hence, no single point of failure. However, this system does not scale up to higher population as communication may grow with respect to the number of peers. A typical example of this type of P2P system is Gnutella and it relatives such as iMesh, Freenet and AudioGalaxy.
2. **Hybrid Unstructured P2P:** In this architecture, there are no central servers but there exist clusters of nodes around some peers called super peers. Also known as ultra peers or overlay nodes in some literature. These infrastructure nodes called super nodes create hierarchy in the overlay network as against the flat manner in pure unstructured. The role of a node in this model may vary with time. An ordinary node this time may be a super node after sometimes. Some examples of this model is BitTorrent and Kazaa built upon FasTrack [40].
3. **Centralized Unstructured P2P:** In this model, a centralized directory

that maintains information about peers exist. To access any resources, peers' request is channeled through this central authority. The connections between peers are not determined by the central authority, it is only used for indexing and bootstrapping. For example, in a centralized P2P file sharing networks, when a peer joins the networks, it contact the central server for it request to download file. The server performs the lookup search for the stored files and their holders, it then responds to the requesting peer with the list of all the stored files, the peers and their addresses. The peer then send query to the node that posses his file of interest to download the file. A typical example of this model is Napster.

Structured Peer-to-Peer Systems

Structured P2P model is organized following specific criteria, algorithms and topology. Structure P2P system tightly associates placement of files with the structure of the overlay networks. The most common amongst them use distributed hash tables (DHT). DHT provides a look up service with (key, value) stored with unique identifier. This hash table type look up search guarantee locating a specific files if only it is available in the network. They are also referred to as overlay networks. The system may differ in look up service, organization and routing strategy. For instance, the organization may be in form of ring, hypercube, butterfly etc. Furthermore, in this architecture, resources indices are systematically distributed amongst par-

ticipating nodes, so that queries for that resources such as files are directly routed to the peer holding the file. A new entrant into this type of network is assigned a peer set with directory of files keys and peers addresses. Examples of such systems that are based on DHT are Chord [41] and P-Grid [42].

1.3.2 Challenges of Peer-to-Peer Sharing

In spite of these design principles that contributed to the success of P2P systems, there are challenges [43, 44] posed by P2P systems:

Free Riding: This is a phenomena, where a peer contribute less than what it consume. This is the main topic of this thesis, we will expatiate on this in next section of this dissertation.

Identity Management: Persistent identity is not a stringent requirement for participating in P2P sharing, due to design goals of the networks. Thus, the availability of cheap or free pseudonyms would make a selfish or malicious peer, to exit when it is about to be discovered and rejoin with a different name, thereby evading any mechanism put in place to check such misbehavior.

Trust Management: The management of trust and reputations of peers is another major challenge facing sharing in P2P systems. The notion of resource availability, authenticity, access control and fair trading [45] is threatened by various trust and reputation issues such as col-

lusion among nodes. An untrustworthy peer may pollute other peers' contents, that is adding impostor contents just to pollute the popularity of files shared by others.

Security Management: Security challenges are common to all computing system. In P2P system, the design characteristics of high rate of mobility, autonomy and anonymity pose a lot of security challenges. These include transmission of malicious contents such as virus, worms and trojan by malicious peer to deliberately harm others in the networks. Furthermore, there is high level of vulnerability of P2P softwares and risk attached to downloaded content.

1.3.3 Free Riding in P2P Systems

Researchers in [23, 24, 20] have identified the existence of free riding in P2P systems. The statistics in [23] show that 25% of the peers in the network provides about 99% of the resources in Gnutella. In [46], researchers found the existence of free riding in Gnutella 0.6. The authors in [20] corroborated the result of [23] by confirming that 7% of the peers contributed more files than others in file sharing networks. In [24], the authors reported the effects of free riding on the performance of the networks. This results exposed the level and seriousness of the problem, as they found that free riding has increased to 85% in Gnutella. There are other measurement studies to confirm the existence and seriousness of free riding such as [47, 48, 49]. In [47],

the authors performed a measurement study on the traffic of PPlive [50], SOPcast [51] and PPstream [52]. They reported the imbalance between peers' file streaming consumption and contribution level. The existence of low fairness ratio in Joost [53] is reported in [48]. In [49], the researchers presented the result of log analysis from Roxbeam Media Networks [54] and confirmed that some peers consume more than their contributions. There is a need for more research effort to develop robust algorithms and techniques to identify free riders and reduce their effects to the barest minimum or possibly eliminate them.

Free riders exploit design principles of P2P systems as explained in section 1.3 to carry out their intent. The activities of free riders constitute a serious threat to the existence of P2P systems as it affect performance and peer utilization of resources [23, 55]. Free riders do not play by the rules of the game of P2P systems, which is supposed to be a cooperative game. These non-cooperative players subvert the overall objectives of the networks, to satisfy their selfish needs.

1.3.4 Effects of Free riding in P2P networks

Although, most of the popular P2P systems such as BitTorrent, Gnutella, Napster, and KaZaa have mechanisms for discouraging free riders, the problem of free riding still poses a serious threat to the existence and survival of P2P systems. The impact differs from one network to the other. The

effects of free riding range from simply annoying the networks or so severe that could bring down the whole system, depending on the networks [1]. For instance, In [4], the authors posited that free riding affects robustness, expandability and availability of P2P networks. They asserted that a network without a free rider counteracting mechanism will have a short life time. This assertion is supported in [56], as the authors argued that if the degree of free riding exceeds the benefits of contribution, the system can be brought to a standstill.

Furthermore, it is reported in [2, 57] that if the menace of free riding is not checked, P2P networks will be reduced to Client-Server paradigm. As the number of free riders increases, the network performance is degraded by the increase in traffic generated by free riders to the networks. The few altruistic peers will become 'server' to the free riders constituting a bottleneck to the networks.

However, in [6], the authors argued that though without external incentives, the level of contribution might be below the socially desirable optimum, file sharing P2P networks can tolerate some degree of free riding due to altruism of some peers.

In summary, the effect of free riding in P2P systems can not be over emphasized as evident in the following recent research work in the literature, summarized as follows; performance degradation [58], increases system stress, degradation of user experience and denial of service [25, 59, 60], degrades

scalability [61, 62], and destroys the philosophy of P2P file sharing network [63].

1.4 Motivation and Research Problem

Mitigation of free riding in P2P systems is currently an important area of research [64, 62, 25, 58, 65]. This is also confirmed through recent work in literature that described the effects of free riding in P2P systems such as [58, 25, 59, 64, 61]. The recent work of [65, 61] and the earlier work in [66] also focused on solving this crucial and vital problem to the success of P2P resource sharing communities. Recently the problem of unfairness due to free riding in BitTorrent is also reported in [39].

Free riders negate the advantages of P2P systems. Free riders encourage selfish behavior and create bottlenecks. With selfish behavior, resource sharing P2P systems will not survive since resources will be limited and a consequence, scalability is negated as well by having only few resource providers. Since BitTorrent is one of the most popular file sharing protocols implementing P2P resource sharing [67, 68, 39, 62, 69] and it has enabled a powerful community of large content distributions [70, 71, 65], researchers have identified free riding in BitTorrent and proposed solutions to solve the problem of free riding [39, 67, 68, 72]. In this thesis, we study and analyze the BitTorrent choking algorithm, show the exploitation of BitTorrent by free riders, pinpoint the root causes of this exploitation, and propose a solution based

on game theory to mitigate free riding in BitTorrent.

In an attempt to proffer a solution to the problem of free riding in P2P systems, we ask and attempt to answer the following questions.

Does free riding occur in all distributed systems?

What are the characteristics of these distributed systems prone to free riding?

Why is mitigation of free riding in P2P system a challenging task?

How do we model and identify a free rider in P2P systems?

How can we effectively mitigate free riding in P2P systems?

1.5 Objectives

The goal of this dissertation, is to design a robust mechanism for effective identification and control of free riders in a P2P system. Our specific objectives are:

- (a) To investigate distributed systems so as to understand their distinguishing characteristics.
- (b) To investigate all distributed systems so as to identify those that are prone to free riding.
- (c) To investigate and identify the strength and weaknesses of existing free

rider detection techniques, so as to evolve new strategies for improvement.

- (d) To characterize and model interactions with Game Theory amongst different types of peers in a P2P system.
- (e) To formulate robust free riding mitigation mechanism in P2P systems by building a game theoretic based model.
- (f) To design and simulate the proposed mechanism and evaluate the effectiveness of the proposed solution in mitigating the effect of free riders in P2P systems.

1.6 Contributions

The main contributions of this dissertation are as follows:

- (a) An extensive literature survey on existing free riders mitigation approaches.
- (b) Propose a node-based taxonomy of distributed systems. This will provides a deeper understanding of the distinguishing characteristics of different types of distributed systems. This taxonomy will also assist in identifying distributed systems that are prone to free riding.
- (c) Propose a game-based taxonomy of games from the game theory perspective

- (d) Propose and evaluate a novel free riding mitigation mechanism for BitTorrent systems based on Game theoretic modelling.
- (e) An extensive simulation-based evaluation study to investigate the performance of our proposed model using BitTorrent as a case study.

1.7 Thesis Overview

In this thesis, we start by providing a node-based taxonomy for distributed systems and identify which of these distributed systems is prone to free riding. Based on our proposed taxonomy, we identified client-server systems as well as P2P systems as distributed systems prone to free riding. We choose P2P system as our focus of research since P2P protocols are among the successful and popular protocols in usage [67, 68, 39, 62, 69].

Among the P2P protocols, we devoted our research towards BitTorrent since it is one of the most successful P2P protocols today [67, 68, 39]. We thoroughly studied the BitTorrent choking algorithm and identified its vulnerabilities to free riders. Then, we proposed a game-theory-based model to mitigate free riding in BitTorrent. To the best of our knowledge, this is the first attempt in mitigating free riding in BitTorrent using game theory. We conducted extensive performance evaluation experiments to assess our proposed model in mitigating free riders in BitTorrent.

The rest of the thesis is organized as follows. Chapter 2 examines the related literature and gives background information on the different approaches to solve free riding. Chapter 3 presents an overview of game theory and introduces the general concepts of playing a game. This Chapter also proposes a taxonomy for games while Chapter 4 presents and analyzes the BitTorrent system particularly the choking algorithm and its advantages and disadvantages. Chapter 5 models the BitTorrent choking algorithm using game theory and also shows the model implementation. Performance studies are performed in Chapter 6 to investigate the effectiveness of the utility of our proposed game-theory-model. The thesis closes with future work and conclusions in Chapter 7.

CHAPTER 2

LITERATURE SURVEY

2.1 Overview

Free riders' problem in P2P networks has been addressed by many researchers in the literature. Some researchers designed algorithms to encourage cooperation amongst peers, while others tailored such algorithms toward detection and punishment. The objective of all these algorithms is to counteract free riders' activities and hence reduce their effects in P2P systems. In this chapter, we classify the existing algorithms as incentive based, reciprocity based, behavioral based, and game theory based. The following sections present a comprehensive review of mitigating free riding algorithms in P2P systems. In section 2.2, we present incentive based schemes. Section 2.3 reviews the reciprocity based schemes. Behavioral based schemes are surveyed in section 2.4 while, game theory based schemes are reviewed in section 2.5.

2.2 Incentive Based Schemes

Incentives have been identified to encourage cooperation amongst participating peers in P2P networks. The incentives could be monetary or other types of non priced incentives that designers deem fit such as service, delay, network membership, and peer rating [73]. Incentive approaches differ in the type of incentive used and the method of managing the incentives as shown in Figure 2.1. For instance, in monetary

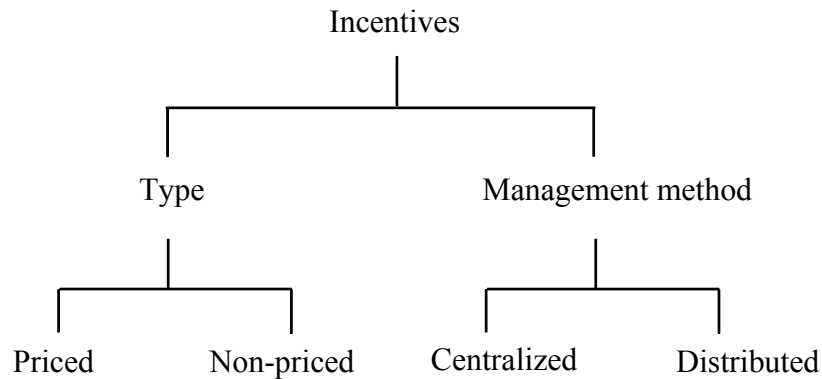


Figure 2.1: Types of incentives in P2P system.

incentive schemes, users are expected to pay for the services received and be paid for services provided in the P2P community. This approach involves the use of virtual currency or digital coins such as XPay [74], KARMA [75], Mojonation [76] and tycoon [10]. The basic idea is that, attaching an economic value to resources in a social community will serve as incentives to encourage peers to contribute more of their resources and get rewarded. It will also serve as punishment for peers that refuse to contribute resources. Pricing is considered to be an effective

means of enhancing peer cooperation in P2P network [77, 78]. Stock market auction based pricing approach is proposed in [79, 3]. While the authors in [80] proposed a lottery based pricing approach. Most monetary incentive based techniques are similar. Some of the fundamental differences are the pricing methods and the exchange mechanisms for the virtual currency between peers in a transaction. In [55], the authors suggested flat rate prices for every peer, but this did not effectively discourage free riding due to the availability of cheap pseudonyms and the difficulty in tracking peers' identity.

In [75], the authors proposed a scalar unit currency called KARMA exchanged between peers while exchanging files. This is a different from [55], in that, a peer does not store the KARMA. The exchange is made by bank-set which is a group of peers that stores the KARMA. The drawback of this technique is that it requires a central administration-entity to manage the KARMA usage.

In Ppay [81], digital coins are exchanged between peers while exchanging resources. These coins are managed by a central authority called a broker. When a peer joins the network, the peer opens an account with the broker and gets an initial credit. Each coin is identified by a coin identifier as well as an identifier of the current coin holder. While exchanging resources, a peer holding a coin transfers the coin to another peer that shared the resource. This transfer is done through the

broker.

The researchers in [82] proposed a micropayment scheme called *flood-gate*. The authors modified BitTorrent and introduced a content provider which controls the tracker. Each peer joining the BitTorrent environment must register and open an account with the content provider. When a peer wants to download a file, it searches the repository operated by the content provider. The content provider contacts the tracker to locate the required file pieces. The token collected from the downloader is given to the provider of that file pieces which can be redeemed at the content provider for future download.

In [7], the authors proposed an incentive mechanism to deter free riding. Credits are computed and assigned to every peer based on their contributions to the network. Credits are managed by a central authority where peers are only allowed to "trade" if they have enough credits. As such, upload of resources increases the credits while download decreases those credits. A free riding peer depletes its credits quickly if it doesn't upload resources.

In [4], the authors presented distributed, monitoring-based, hierarchical structure(DHMS) model. It uses an utility function made up of size of the file, the number of files and the reputation of a peer. This model was implemented in DHMS topology with two classes of peers, super peers and ordinary peers. Their results demonstrate the need for a

robust mechanism to combat free riding. They concluded that without an effective mechanism to mitigate the activities of free riders, P2P network would have a short life span.

In [83], a content management protocol to deter free riders and encourage contribution is proposed. In this approach, a peer monitors the traffic of its neighbors routed through it. In this case, the number of query messages from a peer is compared to query messages from its neighbors. A peer can either be a monitor or be monitored. It counts the query hits of that of its neighbors. If a peer is discovered to be free riding, set of actions such as decrementing the TTL value, ignoring that peers request or even disconnection from the networks depending on the level of free riding is applied on that peer.

In general, incentive based approaches require infrastructure for accounting and micropayment. Thus, there is a need for central authority for administration and control. The cost of exchanging the accounting information is also a serious challenge facing the implementation of these schemes [84, 81]. Moreover, the problems of payment delivery, churn rate variation, cost manipulation and persistent identity are challenging issues that need to be resolved [5].

2.3 Reciprocity Based Schemes

Reciprocity or barter systems involve exchange of services between peers based on their contribution level to each other. The ability of peers to predict future need from other peer is central to this approach. Tit-for-tat [85, 86] in BitTorrent is an example of barter-based approach. As shown in Figure 2.2, there are two types of reciprocity; direct reciprocity and indirect reciprocity. Direct reciprocity is based on repetition of transaction between peers. While indirect reciprocity is based on reputation through recommendations of other peers that have interacted with a particular peer of interest.

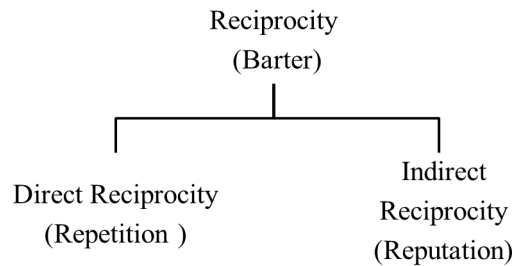


Figure 2.2: Direct and Indirect Reciprocity.

Tit-for-tat has been introduced in BitTorrent environment by the authors in [86]. In such an environment, files are divided into pieces. Tit-for-tat ensures exchange of pieces since peers have to provide in order to receive from other peers. A peer can temporary refuse to upload limits how fast a peer can downloads if it is unwilling to upload files for others. Pure tit-for-tat have been flawed for its inability to

ensure fair exchange of files, especially for new comers in a dynamic system environment like P2P. Other mechanisms are proposed to be used in conjunction with tit-for-tat in BitTorrent to increase its performance. For instance, source coding approach is proposed in [87]. In this scheme, each data block is considered to be a sequence of alphabets. This technique helps in discouraging malicious peers from taking advantage of pure tit-for-tat. Other forms of tit-for-tat are discussed in [88, 89] where a peer is randomly selected so as to allow new comers into the networks. This gives them a chance to download without the need for reciprocation. In the long run they would have to contribute resources to others in the network, by the time they have files to share.

Odd-for-even approach proposed in [90] is a slight modification of tit-for-tat. In this scheme, files are divided into two pieces of odd and even. Peers are expected to negotiate the exchange of pieces, so as to download the pieces that they are not in possession of. If a peer sends its odd piece and does not receive the even piece from its partner, hence that peer is choked for free riding.

In [91], the authors proposed treat-before-trick for BitTorrent-like P2P networks. A peer providing a file in this scheme encrypts the file's pieces with keys and the decrypting keys of each chunk are shared among the peers. Peers are forced to exchange the keys with one another to download their missing chunks, hence making free riders download

files unreadable.

The authors in [92] proposed a quota-based block encryption approach for BitTorrent. In this mechanism, peers are required to exchange encrypted blocks with a randomly generated encryption keys. It uses symmetric-key cryptography in which the encryption and decryption keys are the same. Each peer in a trading session, evaluate two functions to determine whether to send encrypted blocks or decryption keys. These functions are several variables dependent between two leechers i, j . The variables are number of encrypted blocks received and sent, number of decryption key sent and received, as well as their differences. The outcome of the contributiveness function determines if the leecher is to send encrypted block while the optimism function determines if a leecher is to send decryption keys. The authors modelled both contributiveness and optimism function for each peer type described. The proposed peer types in the model are obedient, free rider, conservative strategic and aggressive peers. They compared the quota-based approach with BitTorrent choking algorithm and concluded that quota-based can effectively replace choking algorithm.

The main drawbacks of these approaches are, namely, scalability, peer identity management, fake service and record keeping [5, 84]. Scalability, it does not scale to larger population of peers. As such, there is no guarantee of repetitive transactions as the number of peers in the net-

work increases. The peer identity management, for effective tracking of peers, there is a need for persistent identity. But, the availability of cheap pseudonyms would make tracking peers identity a herculean task. Peers might also elicit services from other peers with the use of fake services. Keeping an accurate record of peers contribution level could undermine this approach.

2.4 Behavioral Based Schemes

In this scheme, peers behavior based on direct interaction such as trust and perception of others are used to design mechanisms to curb free riding. Trust is defined as the confidence that a peer has to ensure that it will be treated fairly and securely, when interacting with another peer [93]. The reputation of a peer is a perception of its behavior based on other peers' observations or the collective information about the peer's past behavior within a specific context at a given time [94]. In other word, reputation is the opinion of peers toward a peer or a group of peers based on intention and social norm [95]. Reputation is contextual in nature but is an integral part of many facets of life such as business, online communities and social networks. Reputation could be derived, inferred or based on direct interactions.

Reputation and trust of peers have been used to enhance cooperation in a P2P systems by detecting and punishing unwanted behavior as

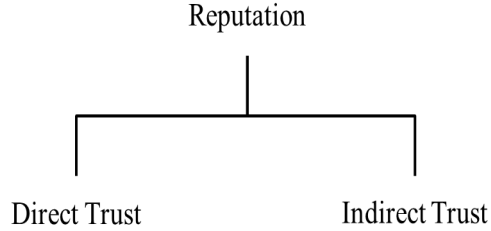


Figure 2.3: Direct and Indirect Reputation.

well as appraising and rewarding wanted behavior. These have also been employed in tracking free riders, which is an example of unwanted behavior in a P2P networks. In this scheme, there is a need to have trust information and past history of peers' behavior to be able to utilize reputation-based approaches. Reputation based scheme can be employed in conjunction with other techniques for identification and control of free riders. Most reputation-based approaches are similar, the difference are mainly in the methodology of trust inference to determine the trustworthiness of a peer, accuracy and complexity [96]. There are several behavioral based systems for identification of free riders and/or for encouraging contribution in P2P systems in the literature. For example, see [67, 97, 98, 99, 100, 101, 102].

In [97], the authors proposed a probabilistic resource allocation to node using their reputation to overcome free riding. Each node resource download is adjusted with its reputation so as to get an appropriate quality of service. A node with low reputation is allowed to download a

finitely low amount of resources. Furthermore, they proposed neighbor selection based on interest and reputation so as to enhance cooperation among peers with similar interest.

Researchers in [98] proposed a reputation based scheme called *Barter-Cast*. In this model, a maxflow network is build in which a node represents a peer while the edges denote the aggregated amount of services. Each peer constructs their local network based on the information exchanged with others. From this network, peers measure the upload and download of other peers. This is based on a reputation metrics designed on the maxflow graph proposed to measure a subjective reputation. In addition, peers build a subjective share history from all the local history exchanged with other peers in the network.

In [99], the authors proposed a free riding mechanism in a hybrid P2P systems. They used super nodes to monitor the behavior of all peers attached to it. Three variable are kept for each peer to measure service provided, requested and rejected. Peers are progressively suspected if there is an increment in its service rejection. The super peer uses these information to queue service provisioning to each peer. Hence, with this behavioural monitoring, free riders can be controlled.

In [67], the authors proposed a demerit point mechanism. In this approach, every peer maintains a list to record its interactions history with other peers in its neighborhood. Each peer is expected to main-

tain a threshold level of upload rate of reciprocation. If the rate is lower than the expected predefined threshold, a demerit point is assigned to that peer. A blacklisted peer will not be selected during the window for random peer selection for download subsequently. Hence, reducing the slots of free riders during the window.

In [103], the authors explored three approaches for trust based incentive. These are, trust aware topology construction, trust-based cost searching and trust-based dynamic topology optimization. They suggested that peers be placed on the P2P topology based on their reputation. Such arrangement they claimed, encourages peers to contribute more so as to be promoted to higher level in order to obtain better service, improved performance and utilization of the networks.

The use of appraisal and blacklist is proposed in [100, 101, 102]. The scheme used the activity set model to describe the activeness of a peer. Each peer maintains a blacklist to record any dealing with free riding peer and an appraisal is posted to every other peers in the network in case of good deal. The authors carried out experiments with three classes of free riders; namely selfish, trustworthy contributors and untrustworthy temporal contributors. The drawback of this approach is the assumption that all peers are honest in their appraisals and blacklisting.

Trust aware topology is proposed in [104] to mitigate the effect of free

riding. The authors present an adaptive P2P overlay topology based on partitioning of peers into super peers and ordinary peers. This partitioning made the computation of direct and recommendation trust of each peer easier, thereby aiding in repelling malicious peers and reducing free riding. Similarly, in [105], the authors proposed a 3-tier hierarchical topology, where peers are classified into three categories, based on performance, availability and contribution to the networks. The classes are superpeers, trusted supervisors and normal peers. Super peers are entrusted with the responsibility of monitoring the normal peers which are treated as new comers to the networks. They are monitored for a while before being "promoted" to the next level based on the behavior, availability and performance. The authors claimed that due to the hierarchy in the node degree of exit and entry, the mechanism does not cause overhead to the networks. Moreover, topologically closer nodes would be encourage to contribute more, hence decrease free riding. However, they assumed trusted supervisors are always available and obedient, which is not realistic in real P2P networks.

There are several challenges facing these approaches for effective performance in the tracking of free riders. These challenges are, whitewashing and sybil attack, the need for centralization, persistent identity and communication cost [106, 5, 107, 108, 84]. The problem of whitewashing and sybil attack. Due to the availability of cheap pseudonyms in

the networks, a peer can always change its name and perform every transaction with a new name. This will thwart the effort to record its unwanted behavior in the networks, even though the peer is free riding. Sybil attack is a situation where a peer generates several names of itself. It then colludes to either falsely accuse an obedient peer to tarnish its image, or erroneously praise a malicious peer in order to boost its reputation. For performance of reputation based approaches, there is a need for some level of centralization such as the central authority to manage the reputation database. This might become a bottleneck to the networks which may cause a single point of failure and vulnerability to attack. Moreover, if there is no persistent identity, it is difficult to track anonymous peers. Also, putting a high cost on identity to reduce whitewashing, will deter genuine new entrants into the networks. Furthermore, the cost of exchanging and storing the reputation information might increase substantially in terms of bandwidth and storage as the number of peers in the network increases.

2.5 Game Theory Based Schemes

Game theory has been found to provide a rich mathematical framework for analysis of complex interactions that occur in all facets of life [109]. Game theory modelling is appealing to researchers of incentives and interactions in P2P networks due to the fact that the difference of cost

and incentives are natural net benefit that can easily be represented as utility function. Also, the rationality assumption of game theory that every players try to maximize their utility tends to fit exactly the situations in P2P system [110]. Each peer in the network can be viewed as a rational game player with his own strategies to maximize his utility of the system. Game theory has been used in the literature in an attempt to represent a peer with rational and diverse interest and model their interactions in a P2P system.

In [66], the researchers presented a game model of interactions between peers in a P2P systems. They assumed the network lifetime is infinitely long, hence discretized time $t = 0, 1, 2 \dots \infty$. Every user is assumed to be rational and pursue maximum payoff. A game is played during each time period. In a game, nodes request service for themselves, and decide whether to serve others or not. The authors assumed all peers in the network are selfish. The contribution reputation of a peer is directly proportional to what a peer can download in a given time period. They studied the reputation changes of each peers as time progresses. A constant ratio of benefit to cost is assumed in the simulation of their utility function and for the computation of peer's sharing probability. The authors analyzed the pure strategy and mixed strategy equilibria of the game. In the pure strategy game analysis, the authors posited that the only Nash equilibrium in this game is (Don't serve, Don't

serve). Which means that no service is provided to the network at all. The authors claimed that this inhibit free riding as free riders has to upload to be able to download equal measure.

The drawbacks of this work are; first, the use of binary value (0 = no contribution and 1 = contribution) in every time period. This does not cater for the size of the resources downloaded or uploaded by peers. Second, the assumption that all players are selfish. This is highly unlikely, since there are altruists that will always give resources to others irrespective of what they get from the system. Third, the cost and benefit may change as game progresses in real P2P systems. Fourth, how the service information between peers are managed is not considered. Finally, repetition history is needed to be able to compute the reputation.

In [65], the authors used game theory to model interactions between any two peers in an unstructured P2P systems. They identified three types of peers, namely, as altruist, in-between (altruist and free riders) and free riders. They modelled the utility function $U_i =$ utility derived by peer i as a net gain, which is the difference between the benefit and cost function ($f_1(.)$ and $g_1(.)$). An extra benefit function $f_2(Q_u)$ is assigned to altruists. These are based on subjective parameters such as self esteem, self satisfaction, reputation, and additional TTL. The utility function is based only on the size of file. The game model was

analyzed offline based on their assumptions. The model was simulated using a predefined threshold, repetition history and complete information. They studied the strategies between two interacting altruists, two interacting free rider and between a free rider and an altruist. A modified game model that uses reputation of resource contributions of a peer is used to designed what they termed *co-opetion framework* to enhance cooperation among peers. The researchers used fairness index to evaluate the performance of their mechanism.

Though, this work is an improvement over [66] which uses binary value for reputation measure per unit time, but there are shortcomings. These includes; the ratio used to differentiate service do not discriminate effectively between peers as the sizes of files increases. The supply of reputation information by the requesting peer to the providing peer is prone to manipulation. The mechanism does not provide a window for free riders to recover once their reputation goes beyond a certain predefined threshold. The assumption that there is no illegal activities, lying of peers is unrealistic. The real time variation of cost and benefit is not modelled. Moreover, the authors do not consider activeness and inactiveness of peers in their work. This may give rise to dormant network, if peers are allowed to stay in the network for a long time period without participating in network activities.

In [61], the authors presented a mathematical framework of incentives

in P2P networks based on game theory. They modelled P2P network as a discrete time system. They referred to peers' utility as points. A peer gains α points for receiving service and losses β point for providing service. The number of services provided over time is used to differentiate between different types of peers. Also, they proposed two learning models for peers, namely, current best learning and opportunistic learning. In current best learning model, a peer chooses its strategy at the end of every time slot using an adaption rate, in which peers learn from others to make decision on sharing. On the hand, opportunistic learning model, in this case, a peer randomly chooses a "teacher" to learn from in the network to make decision on sharing. Moreover, the authors classified peers in P2P networks into cooperators, defectors and reciprocators. Thereafter, they presented incentive models, which are; mirror incentive, an analogy of tit-for-tat, proportional incentive, where a peer serves requester with a probability equals to the peer's contribution to consumption ratio. In linear incentive model, a constant generosity matrix is used to classify peers. A check in sharing history by peer will determine whether to share or not. The authors analyzed the robustness of each incentive model analytically. One of the proposed model, the proportional incentive policy is used to incentivized BitTorrent. Based on their experiments, they deduced that free riders adapt their strategy by changing from defective to reciproca-

tive. Furthermore, the researchers showed a relationship between their model and evolutionary game model, specifically, the pairwise contest population game. They concluded that there should be limit to degree of altruism so as to encourage sharing. However, the use of constant values to represent peers gain and loss limits this approach.

The authors in [64] proposed a stage game of chunk sharing in a multimedia P2P sharing as a gift giving game [111]. They use asymmetric interest of peers to support their choice of gift giving game instead of Prisoners dilemma. The game is modelled as follows; it has two actions, $\{Serve = S, Notserve = NS\}$, If $a = S$, the server incurs cost c and the client receives a benefit of r . If $a = NS$, both r and c equals 0. They considered constant r and c for every chunk. They claimed it can be extended to consider peer dependent and time varying cost. With the assumption that $r > c$, and the ability of peers to have multiple connections, the social utility U is the sum of all utility derived from all connections. If $a = NS$, the short term equilibrium = 0 which is undesirable for the utility of all system to be 0. This model is a single stage game. To cater for subsequent interactions, the authors adopted a repeated game model. Where each player can be identified with their reputations. A trusted third party is assigned to manage the trust information. Moreover, they analyzed the repeated game based on social norm and one shot deviation principle. The researchers concluded

that based on their simulation and analysis, no peer gains by deviating from socially acceptable norm. However, the true cost c may vary from one peer to the other and may vary with time. Also, the assumption that the benefit r is constant limits this model, though it may vary depending on its content and peer, since benefit is highly subjective.

The authors in [106] adapted the prisoner dilemma to model incentives in P2P networks. Each game consists of two players which can either defect or cooperate. To present a social dilemma, one player is the client and the other is the server in the game. The decision of the server is meaningful in determining the outcome of the game. A player can be a server in one game and a client in another game. The authors used a constant points to represent the payoff of each player.

In [112], the authors represent the interaction of peers in P2P network as a game theoretic model. They identified three forms of equilibrium and concluded that though social optimality of public goods cannot be reached if external incentives are not provided to the peers, some sharers will be inherently generous to provide the contents needed to maintain the networks. Hence, file sharing networks could reach equilibrium in the presence of free riding, but the degree of free riding that can be tolerated is not ascertained. They claimed that forcing all peers in the networks to share may not yield the desired social optimal outcome. To some peers, the cost of sharing may outweigh the benefits.

The authors in [113] proposed a model of interactions between peers in a P2P systems as a two player game. They present the repeated form of the game and classified nodes as Enthusiastic, Selfish and Rational. They proposed a graded punishment for peers based on their strategy. However, there is no measure to safeguard the reputation from attack or cheating.

In [114], a model of incentive with strategy and Nash equilibrium in game theory is proposed. The strategy of a peer is its level of contribution and its utility is payoff which is expected benefit from other peers. The authors proposed service differentials based incentive scheme where a peer provides service to a peer based on his contribution to improve the performance of the system. They show that two equilibria exist in this form of game. Similarly, in [55], the utility function is made up some variables such as download amount, network variety, disk space used, bandwidth, altruism and financial transfer. With the assumption that every peer is economically rational, they weigh the financial benefit and the cost of sharing. However, there is a need for a fair exchange mechanism to be agreed upon by the two peers in interaction in order for the transaction to end successfully.

In [115], the researchers observed that the incentive mechanisms in BitTorrent do not deter free riders enough. As free riders are able to download as fast as non free riders. They proposed a modification of

the choking algorithm as an iterated prisoners dilemma's game. Every peer maintains a record of upload amount and download amount in all its connections. The difference of these two variables is multiplied to a constant factor called nice factor. This is the factor a peer can afford to risk for cooperation. The constant is assumed to be equal to the size of a block. The nice factor is then adjusted to have the following properties;(i) retaliatory, which means defect if the opponent defects. (ii) Nice, that is never defect first.(iii) Forgiving, which connotes cooperation if the opponent starts cooperating after defection.(iv) clear, which means inform the opponent of reciprocal behaviour. They claimed their modified incentive mechanism is more robust against free riding. However, the effect of seeders are not considered.

The researchers in [116] proposed a model to analyze the relationship between user selfishness and system stability in file sharing P2P networks. The model is based on Prisoners dilemma game in which players have two sets of actions, defector and cooperator. The authors modeled what they called caching game to represent P2P networks. In this game, each player is a file holder with two sets of actions, caching and no caching. The game is an asymmetric complementary game with complete information. In this regard, taking a different strategy of the opponent will yield different payoffs. The strategy selection for each player is controlled by the ratio of number of cooperators to the total

number of peers in the system. Furthermore, the authors attempted to cater for heterogeneity by relating their model to Evolutionary game theory. In this case, they concluded that a dominant strategy displaces weak strategy in the network. They assumed that every node is aware of the behavior of all other nodes in the system. From their model and analysis, they concluded that there is a relationship between users cooperation and system robustness.

However, the requirement that players pass on their strategies to their offsprings as time goes on does not correctly capture peers characteristics. Since, in P2P systems every peer is an autonomous entity on its own. The assumption that every player has the knowledge of the whole system is not realistic. The possibility of one strategy displacing other strategy with time no matter how high is the payoff is not realistic.

In [117], the authors modeled the resource sharing in P2P with Evolutionary game theory with the claim that Nash equilibrium is hard to compute and does not capture the highly dynamic nature of P2P networks. The authors posited that the strategy of trial and error by peers to share or not in which they learn over time is better. Since, sharing is performed in a rational and uncertain manner but with mutual interest. Moreover, they compared the resource sharing in P2P network to stag hunt [118]. They concluded from their model that there is a correlation between evolution and payoff matrix.

In [110], the authors adapted oligopoly game to model incentive in a P2P storage system. They represents the total desirable resource as α , the market clearing price in oligopoly was defined as the difference between α and the total contribution of resources. It is observed that α and c are time variant quantities. They tailored the value of α and c in the game to reflect P2P system. However, the authors posited that is difficult to compute the total resource required offline. However, the system dynamics and interaction mode in real P2P systems affect this modeling. For instance, in storage sharing, CPU sharing networks, each peers contribution level affects the global resource availability. Hence, if total contribution is optimal, further contribution of the same resource would yield less utility. Generally, utility functions may vary with time, due to variation in cost and benefit [110]. Similarly, it is noted that in a P2P storage system, the cost may not be significantly that of the storage but more of that of bandwidth consumed by peers accessing the storage. There is cause-effect dynamics in storage P2P systems between storage and bandwidth which is unknown. Hence, representing this parameters and compute their value to reflect a real system remain a daunting task.

In [39, 119, 120], the authors proposed a buddy incentive protocol for BitTorrent. The buddy protocol is modelled as an infinitely repeated game. In this model, leechers are only made to unchoke their buddies,

that is peers with the same download and upload rate. Randomly giving a peer chance to download is only performed by leechers if it does not have the data required by its buddies. The analytical game model models the distribution of contents between leechers. The strategy choice for all the players are independent player, that is a peer that runs the original BitTorrent. A free rider, a peer that does not upload to others and a buddy aware peers. They formulated the utility function of leechers to be $U = d$ that is the average download rate for all leechers. Where all players have different utilities according to their behaviors. They prove that Nash equilibrium exist in their game model. Similarly, the authors extended the utility functions in [119] to include honest peer and dishonest peer and model same as game as presented in [120]. They concluded that unfairness is reduced from their simulations. However, seeders impact is not considered in both approaches.

In conclusion, we observed in our review that a good number of factors may pose challenges to a generic game model for P2P systems. First, heterogeneity of environments. Users of P2P systems are scattered over a vast geographically distributed over the world. These users have different computational power, storage capability and bandwidth. Second, the presence of Altruists in the system also affect the rationality assumptions. Altruists are peers that contribute to the common good irrespective of their gain from the system. Third, variable factors such

as cost, benefit variation, peer variational behavior and decisions under uncertainties which are common in real P2P system poses challenges to the design of utility functions. Fourth, interactions in real P2P system is more complex than this model as peers cannot accurately predict the need from another peer in the future. Fifth, heterogeneity of interest. Each user in the system has different interest, need and objectives. For instance, peer x may be interested in a resource owned by peer y, while peer y may want resource from z. Peer z may be interested in resource of x. Furthermore, behaviorally, peer x may be an altruist, peer y may be a free rider while z may have behavior in between these two. Hence, there is a need to take the architecture of the P2P system into consideration in an attempt to model peer interaction as a game.

CHAPTER 3

GAME THEORY

3.1 Overview

In this chapter, we present an overview of game theory. We introduce the general concepts of playing a game and classify games based on some criteria such as the amount of information available to players, the manner in which the players choose their actions during the game, how the game is being represented and number of players involved in the game. This overview serves as a background to provide a better understanding of our proposed game model. The remaining sections of this chapter are arranged as follow. Section 3.2 presents the basics of a game and its constituents. In section 3.3, game descriptions are presented while some examples of games are discussed in section 3.4. In section 3.5 a taxonomy of games is presented and section 3.6 concludes this chapter with some game solution concepts.

3.2 Game Basics

Games are played in numerous aspects of our life but we might not be aware that we are playing a game [121]. Games are being played in politics, market, war and daily interactions. For instance, a person decision to choose restaurant A instead of B due to price and quality of food offered, is playing a game. Economic activities such as auction, bargaining and price competition between firms offering the same service or producing the same product are all games. Games are modelled using game theory. Game theory has been found to provide a rich mathematical framework for analysis of complex interactions between players. In general, game theory models decision making in an interaction between rational players. The rationality assumption shows that a player has preferences and is aware of the consequences of his actions. Each game must have one or more players. A player is a general entity of interest which may be an individual, organization, country, or a node in a network. A game must also have a non empty set of actions. The overall plan of actions taken by a player is known as strategy. The set of actions taken by a player in a game is determined by the player's strategy. For instance, one can view the strategy of a game as a complete algorithm to play the game, while the actions of the game are each step in the algorithm. The result of a particular action in a game is known as payoff. Based on the strategy, an action or set of

Table 3.1: Payoff matrix for prisoner P_1 and prisoner P_2

	P_2 (Confess)	P_2 (Do not confess)
P_1 (Confess)	$(f_1^{P_1}, f_1^{P_2})$	$(f_2^{P_1}, f_2^{P_2})$
P_1 (Do not confess)	$(f_3^{P_1}, f_3^{P_2})$	$(f_4^{P_1}, f_4^{P_2})$

actions are taken in order to increase the payoff. A game should be played according to the rules that specify how the game can proceed.

For example, consider a classical example of a game known as *prisoners dilemma*. In this game, two crime suspects are remanded in prison for interrogation. The players are prisoner (P_1) and prisoner (P_2). The strategy of each prisoner is to spend less time in prison. Each prisoner must take an action for the game to evolve. The set of actions available to each prisoner are *confess* or *not confess* of the crime. The rules of the game specify that prisoners are kept in separate cells and questioned separately. The rules also specify the payoff a prisoner gets for taking a particular action. Once a game is played by all players, the outcome is jointly determined. As shown in Table 3.1, if P_1 confesses and P_2 confesses, the outcome of the game is $(f_1^{P_1}, f_1^{P_2})$. As such, the game has four outcomes depending on the actions taken by the players.

3.3 Game Descriptions

In game theory, games are described using different representations. These representations include normal form and extensive form. These two representations differ in the type of game described, the amount

and depth of information needed to be retained for convenience in analysis and to reduce computational complexity. A normal form game is a representation of game that does not capture the notion of sequence and time of the action. It is a game structure that is mathematically simple, abstract and analytically convenient [122]. A normal form game is represented with a payoff matrix. An example of a normal form representation of game is as shown in Table 3.1. In the normal form representation, some information such as action sequence and time about the game may be lost but focuses our understanding on dominant strategies, Nash equilibria and their payoffs [123]. For instance, consider a game involving two players, if the moves per player is finite, this can be reduced to a game with one move per player. The extensive form game is an explicit representation of a game that captures the dynamics of interactions of players and made the temporal structure explicit. This complete description of a game is represented by game tree revealing information such as; (i) complete descriptions of the set of players (ii) the move of each player (iii) the time of the move of each player and the choices made by those players (iii) complete description of the players' knowledge as at the time of taking an action in the game and (iv) the players payoff functions. Every extensive-form game can be converted to an equivalent normal form game but this conversion process may result into an exponential increase in the size of the representation,

hence, making it computationally intractable [124]. An example of the extensive form of matching pennies game is shown in Figure 3.1.

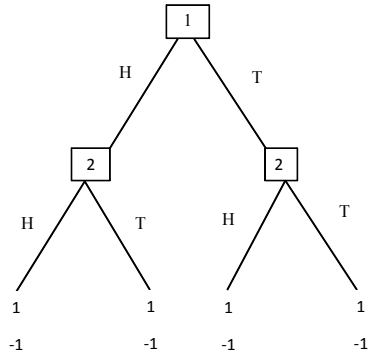


Figure 3.1: Extensive form representation of matching pennies game.

3.4 Game Examples

In this section, we present some classical game models to further illustrate the usage of game theory to model decision making. These game models are widely used in the literature and they serve as classical example to explain game theory concepts.

3.4.1 Prisoner's Dilemma

The description of the prisoner's dilemma game is as follows. Two individuals are arrested over a serious crime. Each is known to be guilty of a minor crime prior to this time, but there is no evidence to convict them of the serious crime. The suspects are separated and interrogated differently. During interrogation, each of them is told that

if he testifies about the other's guilt, he will get an amnesty of reduced sentence for the crime committed. This game is modeled as follows. If both suspects did not confess they both get a reduced sentence of 1 year and if neither of them confesses, they both get a sentence of 3 years. If a suspect testifies against the other, the suspect gets 0 year and the other gets 4 years. The payoff matrix for this game is presented in Table 3.2.

Table 3.2: Payoff matrix of Prisoners Dilemma

	Prisoner2(Confess)	Prisoner2(Dont confess)
Prisoner1(Confess)	(3,3)	(0,4)
Prisoner1(Dont confess)	(4,0)	(1,1)

3.4.2 Stag Hunt

This game describes an interaction where conflict and cooperation may exist. Two hunters go out hunting for three possible preys which are, a large stag and two hares. The only way to get a stag is for both hunters to work together. However if one goes for the stag and the other goes for a hare, the hunter that goes for the stag will end up with nothing while the hunter that go for hares will get the two hare for sure. If both hunters go for hares, each will get one; and therefore will have dinner regardless of what the other has chosen. Although hunting a hare produces less meat, it is potentially advantageous because it is really easy. The assumptions of this model are; (i) there is no communications

between the two hunters (ii) the stag yields more food than the hare (iii) in any given scenario, at least one hunter is guaranteed food (iv) both hunters are rational and equally informed (v) each hunter can only bring equipment for one type of animal. The payoff of this game is as shown in Table 3.3. It should be noted that the numbers in the payoff matrix represents the value of the meat from the animals. If both hunters hunt for stag, they both get 40. If one hunter goes for stag alone, he will get 0, while the hunter that went for hare gets 20. If both hunters hunt hare, they both get 10.

Table 3.3: Payoff matrix of stag hunt game

	Hunter2(Hunt stag)	Hunter2(Hunt hare)
Hunter1(Hunt stag)	(40, 40)	(0, 20)
Hunter1(Hunt hare)	(20, 0)	(10, 10)

3.4.3 Battle of Sexes

Battle of sexes is a two person game. Both players need to coordinate their activities but they have different preferences. In this model, there is a conflicting interest, but both husband and wife has to agree on mutual gain. The game is described as follows. A couple plan to meet at a cinema in the evening to watch a TV program. Both are separated and driving to the cinema. They both forget on the plan and not sure of the show to watch. Both cannot communicate but they prefer to be together and watch the same show. The husband prefers football

than opera and the wife likes opera much better than football. If the husband watches the football with her, his payoff is 2; but if he watches the opera with her, his payoff is 1; and if he goes to either of the places without his wife, his payoff is 0. Similarly, if the wife goes to watch the opera with her husband, her payoff is 2. But her payoff is 0 if she ends up at the opera without him. If the woman joins the husband to watch football, her payoff is 1. But her payoff is 0 if she ends up at the football game without him. This scenario is represented as a game with payoff matrix shown in Table 3.4.

Table 3.4: Payoff matrix of Battle of Sexes Game

	Husband(Opera)	Husband(Football)
Wife(Opera)	(2, 1)	(0, 0)
Wife(Football)	(0, 0)	(1, 2)

3.4.4 Matching Pennies

Matching pennies game is an attack-defense game with the following description. Two people hold a coin and simultaneously choose to reveal either Head or Tail on their coins. Player 1 loses his coin to player 2 if both match, otherwise, player 1 wins player 2' s coin. The payoff matrix is shown in Table 3.5. One can observe clearly in this game that it a real game of conflict. The gain of player 1 is the loss of player 2 and vice versa. It is an example of two person, zero-sum game.

Table 3.5: Payoff matrix of Matching Pennies Game

	Player2(Head)	Player2(Tail)
Player1(Head)	$(-1, +1)$	$(+1, -1)$
Player1(Tail)	$(+1, -1)$	$(-1, +1)$

3.5 Game Taxonomy

Games can be classified based on the player, action, outcome, repetition and termination as shown in Figure 3.2. For example, player-based classification can be based on different criteria. These criteria could be the number of players involved, the nature of players and the type of information available to players.

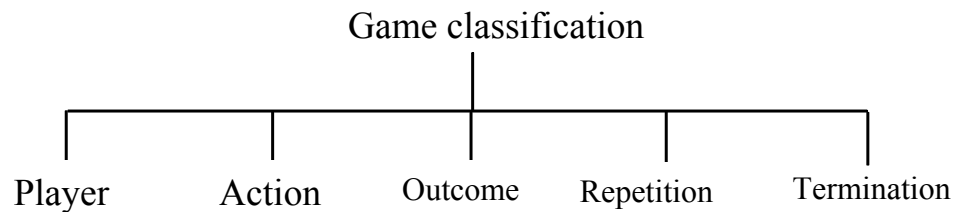


Figure 3.2: Classification of game.

3.5.1 Player-based Game Classification

This game classification is based on "who is playing the game"-the player. The criteria related to this classification are number of players, the effect of the identity of players on the outcome of the game, the nature of cooperation type of players and the information availability to players in the game as shown in Figure 3.3.

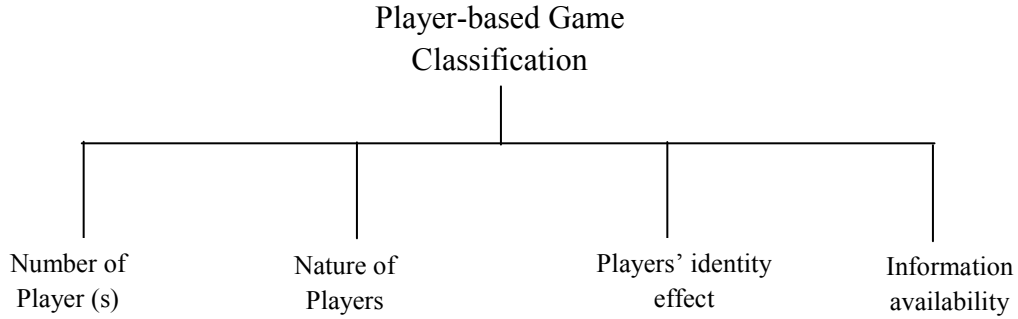


Figure 3.3: Player-based classifications of game

Game Classification Based on Identity Effects of Players

A symmetric game is a game in which the payoffs depend only on the other actions employed and not on the player playing them [125]. The identities of the player do not affect the payoffs. In other word, the identities of the players can be changed without changing the payoff. An example of a symmetric game is prisoner’s dilemma described in section 3.4. Formally, a normal-form game is symmetric if the players have identical strategy spaces($S_1 = S_2 = \dots = S_I = S$) and $u_i(s_i, s_{-i}) = u_j(s_j, s_{-j})$, for $s_i = s_j$ and $s_{-i} = s_{-j}$ for all $i, j \in 1, \dots, I$. (s_{-i} denotes all the strategies in profile s except for s_i .) Thus, we can write $u(t, s)$ for the payoff to any player playing strategy t in profile s . We denote a symmetric game by the tuple $[I, S, u(\cdot)]$ [125]. On the contrary, asymmetric games are games in which the strategies are not identical for both players. Though, it may be possible sometimes for an asymmetric game to have an identical strategy for both players. In asymmetric game the identity of the players affect the payoff. For in-

stance, if both prisoners in the prisoner's dilemma game have different payoffs for choosing the same action, then, it becomes an asymmetric game.

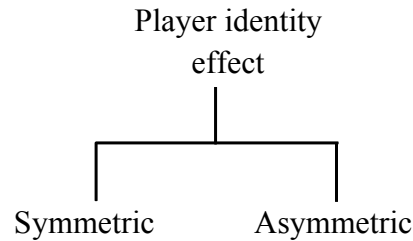


Figure 3.4: Player-identity effect classifications of game.

Game Classification Based on Number of Players

There are different types of games that may involve one, two or more players interacting in a strategic setting as shown in Figure 3.5.

Two Player Game : This is also referred to as a two person game. It involves two players interacting in a strategic environment. These are the most common category of game studied in game theory. A special case of this class of games is N-Player games. Some common examples of two player games are prisoners dilemma, stag hunt and battle of sexes.

N-Player Game : This is also called multi player game. This involves three or more players interacting in a strategic setting. The analysis of this form of game is complex due to multiple and complex decision made by several players. Most N-player games

can be reduced to a two player game for simplicity of analysis. Formally, N-person game is a set of n players or positions each player with a finite set of pure strategies and corresponding to each player a payoff function P_i which maps the set of all n-tuple of pure strategies into real number [126]. An example of N-player game is an n-player coordination game.

One-Player Game : In game theory, game occurs when there is conflict of interest amongst the players. In one person game there is no conflict of interest. Only a single player take decisions based on its strategy to achieve it goal. This form of game is of no much interest in game theory, but it may be interesting in probabilistic and complexity term. A one player game is model as a decision problem.

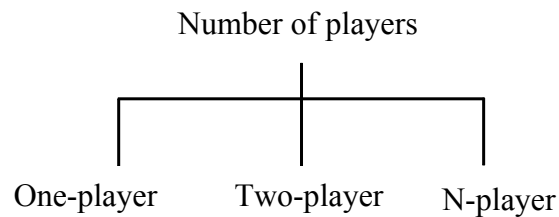


Figure 3.5: Number of player classifications of game.

Game Classification Based on Nature of Cooperation

As shown in Figure 3.6, the nature of cooperation of players involved in a game can be used to classify game into cooperative, non cooperative and coordination game. *Non-Cooperative Game* : Non-cooperative

game is a class of games in which players make decisions independently. Though players may choose to cooperate, cooperation must be self enforced. As such, there is no external enforcement for each player to plan actions with other group of players and take consensus-based decisions. Also, a game is said to be non-cooperative if it is impossible for the players to communicate in any way. In other word, if each player pursues self interest that may partly conflict with the interest of other player, we said the game is non cooperative [127]. Furthermore, in this type of games, players cannot make a credible commitment to cooperate in their strategies. The basic model of interest in this form of games is the individual player-including his interest, beliefs, preference and possible actions [124]. Thus, non cooperative game theory deals with the study of interactions of competing rational individuals. Example of non-cooperative game is prisoners' dilemma game since both prisoners do not share information in taking their decisions. Both take independent decisions without necessarily considering the opponent.

Cooperative Game : In cooperative games also known as coalition games [124], the term cooperative implies that players have complete freedom of communication and complete information on the structure of the game. There is a possibility of making an enforced agreement binding on one or both players to a certain policy. Co-ordination with other members in the group of players is required in coopera-

tive games [126]. The basic modelling unit in this class of game is the group. In this case, the outcome of any chosen strategy is a joint decision by the group. Formally, cooperative games involve a set of players, denoted by $N = \{1, \dots, N\}$ who seek to form cooperative groups, i.e., coalitions, in order to strengthen their positions in the game. Any coalition $S \subseteq N$ represents an agreement between the players in S to act as a single entity [128]. Example of cooperative game is coordination game. In this case, players need to take a joint decisions, since the game is between group of players.

Coordination Game : This is a class of games somewhat in-between cooperative and non cooperative games. In these games, there exist some element of cooperation and competition. This class of games usually have multiple equilibria. Coordination games are formalization of coordination problems. There are variations in coordination games depending on the level of conflict of interest and preference with reference to the payoffs. In *pure (common interest) coordination game*, both players prefer the same action. Hence, their preference will yield a higher payoff than the other action in which they do not have same preference. Another forms of coordination game is the one in which both players get a better payoff if they coordinate, but less if they do not. This is the case in the stag hunt game, see section 3.4. In general, it is better for both hunters to coordinate their efforts, since the action

pair (stag, stag) yield a higher payoff than the action pair (hare, hare), but there is a risk. On the other hand, the actions (hare, hare) provide less risk due to uncertainty of each hunter's action. The strategy pair (hare, hare) provides a higher expected payoff. As such, the payoff for both hunters are the same if they coordinate their activities but differ if they do not. Lastly, the conflicting interest coordination game also refers to as Battle of Sexes as described in section 3.4. In this case, both players have to agree on a mutual gain, though some compromises exists.

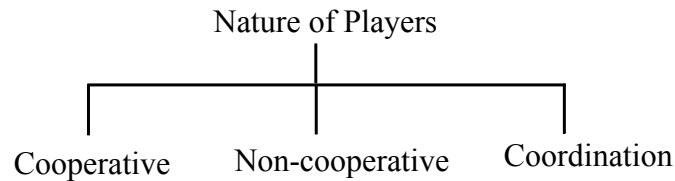


Figure 3.6: Nature of player classifications of game.

Game Classification Based on Information Availability

Complete Information Game: The payoff functions and the strategy sets of this type of game is a common knowledge. Hence, decisions are based on the available knowledge to all the players in the game. The players know each element in the game definition [129]. In incomplete information game, none of the players knows the type and action of the other until both of them makes their move [129]. In the other word, the game being played is not a common knowledge. A game is

said to be a common knowledge if the players know the structure of the game and they are all aware that other player knows the structure of the game and so on [130]. The difference between complete and incomplete information game is that players in incomplete information game or at least some of them, lack full information about the structure of the game as define by it normal form [131]. Furthermore, by suitable modelling and transformation all forms of incomplete information game can be reduced to complete but imperfect information game.

Perfect Information Game : A game in which the state of the game and all moves are known to all players are said to have perfect information. The player with the move knows the full history of the game so far. That is, the moves previously made by all other players. For example, a game without chance element like Chess is a perfect information game. On the other hand, in *imperfect information game* , the players do not know the full history of previous moves made by others.

To clearly distinguish between the often confused concept of complete and perfect, incomplete and imperfect information game. The author in [129] differentiates them as follows, complete and incomplete information deals with the amount of information the players have about the rules of the game. On the other hand, perfect and imperfect information deals with the amount of information each player has of other players and their moves. Game classification based on information

available to players is illustrated in Figure 3.7.

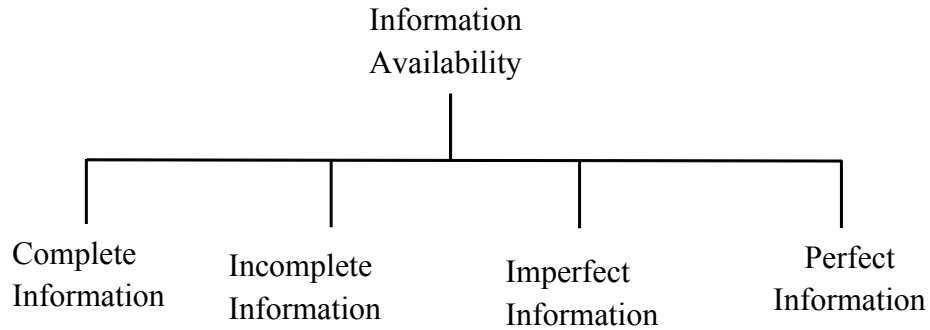


Figure 3.7: Information availability to player classifications of game.

3.5.2 Action-based Game Classification

Games can also be classified based on the features of the set of actions. The size of the action set which can either be finite or infinite. Also, the occurrence of these actions and the payoffs associated with the outcome as illustrated in Figure 3.8. A player in a game may choose a pure action and apply it or randomize it, thus, resulting in pure or mixed strategy game. Moreover, games may have a known outcome, that is deterministic or may occur with known a probability distributions. *Pure Strategy and Mixed Strategy* , In this game, each player selects a single action and apply it. The columns or rows of the payoff matrix represent both an action and a pure strategy while in mixed strategy, players randomize over the set of available actions according to some probability distribution [130]. Formally, A mixed strategy for player $n \in P$ is a probability distribution say $\xi^n = (\xi_i^n)_{i \in S_n}$ over his pure

strategies in S_n . That is, $(i \in S_n) \xi_i^n$ and $\sum_{i \in S_n} \xi_i^n = 1$ [132]. The prediction outcome of a mixed strategy is stochastic, hence less accurate than the pure strategy. As such, the outcome is computed as expected utility. Pure strategy can be seen as a special case of mixed strategies. We will discuss this concept further in section 3.6 with battle of sexes.

Finite game is a type of game that each player has a finite number of actions alternatives to choose, otherwise, the game is an infinite game. As such, the player has a continuous action set, this might be a vector of alternatives. Finite games are also known as matrix games.

Deterministic game : If the players' actions uniquely determine the outcome of the game, as captured in the objective functions, hence the game is deterministic. On the other hand, if the objective function of at least one player depends on an additional variable (state of nature) with a known probability distribution, then the game is a *stochastic game* [133].

3.5.3 Outcome-based Game Classification

As shown in Figure 3.9, games can be classified based on their outcome. In this class, we have constant-sum and non-zero sum game. *Constant-sum game*: This is a type of game in which the sum of payoff of every outcome of each player is constant. It is a game of pure competition. Poker game is an example of constant-sum game, since the total

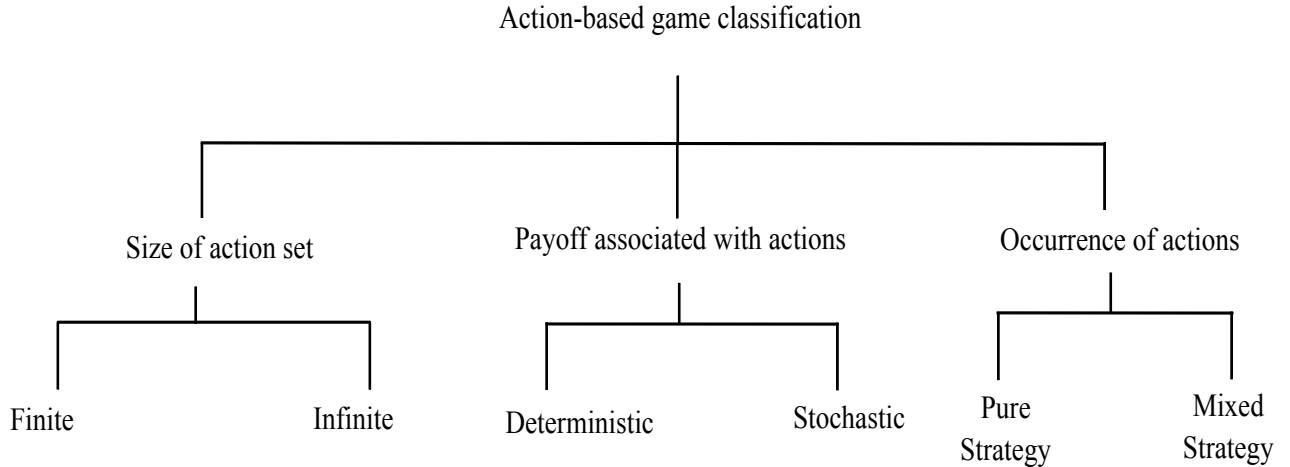


Figure 3.8: Action-based classifications of game.

wealth of both players add up to constant. *Zero-sum game*, is a class of constant-sum game in which the total gain for both players sum to zero. Formally, given a two person game with one move per player. If each player selects independently and simultaneously from the finite set of action plan that results into a payoff for each player. Let i, j be their independent choices, then the game is described by a real matrix $X_{ij} = (x_{ij})$ and $Y_{ij} = (y_{ij})$. Where x_{ij} is the payoff for player 1 and y_{ij} is the payoff for player 2. The game is zero sum if and only if $x_{ij} + y_{ij} \equiv 0$ [123]. Thus, zero sum game is a real game of conflict, a player benefits only at the expense of the opponent, hence, the gain of one player equal the loss of the other player. Example of such game are chess, poker, matching pennies. On the other hand, in *non zero sum game*, the gain of one player may not necessarily be the loss of the other player. As such, the payoff may not add up to zero. For instance,

in prisoner dilemma, the sum of both players payoff may be greater or less than zero, depending on the actions of both players.

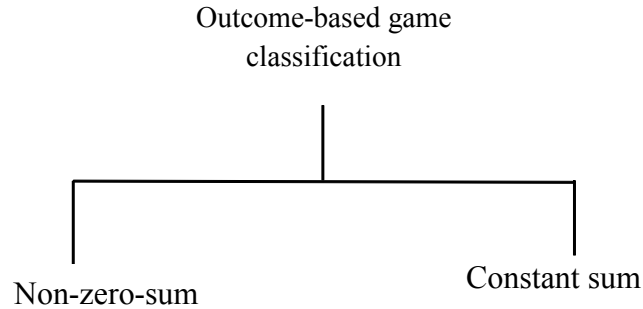


Figure 3.9: Outcome-based classifications of game.

3.5.4 Repetition-based Game Classification

Repeated Games : This is sometime referred to as supergame. It is a scenario in which the same stage of strategic form of a game is played at each time t for some duration of T periods. It is an infinitely repeated game if $T = \infty$ [134, 135]. A stage game is a one-period simultaneous move game of complete information. Repeated game is used to study interactions that goes on over time. Repeated game could be of perfect monitoring. This means that at the end of every time period, every player is aware of the action chosen by all other players at that period. But in imperfect monitoring, the actions of other players are not observable. *Unrepeated (one) shot games* are games played once without repetition.

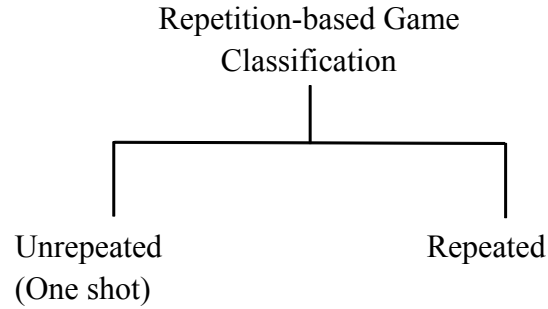


Figure 3.10: Repetition-based classifications of game.

3.5.5 Termination-based Game Classification

The outcome of a game is determined at the end of the game. The question here is what determine the end of the game ?. The end of a game could be determine by time and action. Games can also be classified in the dimension of how does the game terminate as illustrated in Figure 3.11. If a game is duration specified game, this is time constrained game in which the expiry of this designer specified duration marks the end of the game. As in the game of soccer, two 45 minutes make a complete 90 minutes regulation time. Also, the end of a game could be determined by moves taken by either one or both players in a two-player game. The action that determines the end of the game may be taken simultaneously by both players or sequentially one after the other.

Sequential (Dynamic) Game: The action is taken sequentially until a particular point in the game or the player choose not to move further.

Players in dynamic games are aware of earlier actions of other players

Table 3.6: Example of Players-related Classification of games

Players-related Classification				
Game/Criteria	No. of players	Nature of players	Players identity	Information availability
Prisoner' Dilemma	2	Non-cooperative	Symmetric	Imperfect
Battle of Sexes	2	Coordination	Symmetric	Imperfect
Matching Pennies	2	Non-cooperative	Symmetric	Imperfect
Stag Hunt	2	Coordination	Symmetric	Imperfect

before taking an action. The order of play are strictly observed hence, there is possibility of learning since players are aware of others' actions. The simplest case of a sequential game is a two player game with one move each per player. In this case, the first player takes an action and the second player takes it action after observing the action of the first player. Examples of sequential games are auction, bargaining and price competition.

Simultaneous (Static) Game: The actions in this category of game are applied by all players simultaneously. The players might not take the action exactly simultaneously but the second player is not aware of the action of the first player while taking its action. Example of simultaneous game is prisoners' dilemma described in section 3.4.

As a summary, we classify the four classical games discussed in Section 3.4 to fit our taxonomy. The classifications of these four games are illustrated in Tables 3.6 - 3.9.

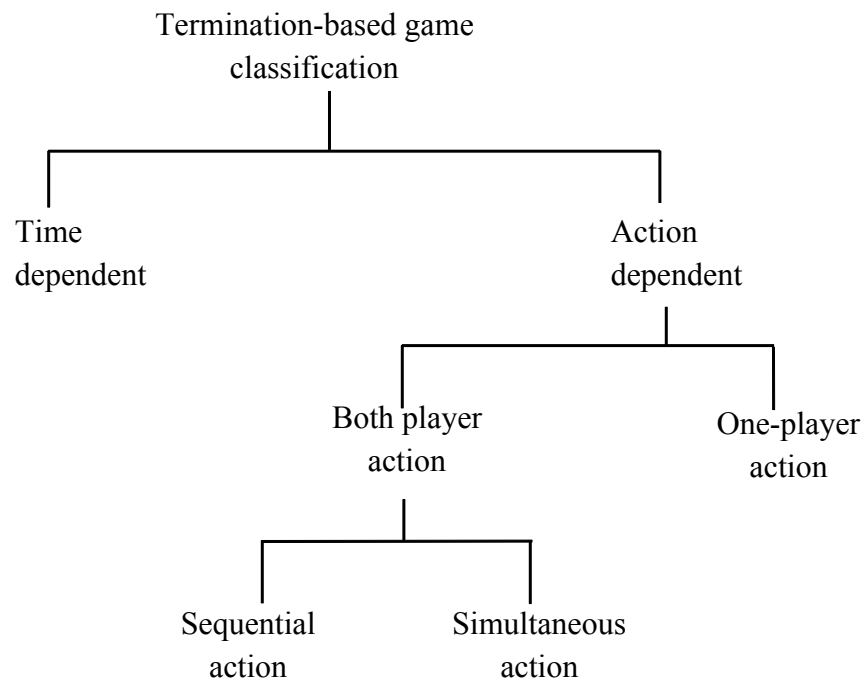


Figure 3.11: Termination-based classification of game.

Table 3.7: Example of Actions-related Classification of games

Actions-related Classification						
Game\Criteria	Size of actions		Occurrence of actions		Payoff associated with actions	
	finite	infinite	pure	mixed	deterministic	stochastic
Prisoner' Dilemma	✓		✓	✓	✓	
Battle of Sexes	✓		✓	✓	✓	
Matching Pennies	✓		✓	✓	✓	
Stag Hunt	✓		✓	✓	✓	

Table 3.8: Example of Moves-related Classification of games

Moves-related Classification		
Game	Sequential	Simultaneous
Prisoner' Dilemma		✓
Battle of Sexes		✓
Matching Pennies		✓
Stag Hunt		✓

3.6 Game Solution Concepts

3.6.1 Dominant Strategies

A strategy profile is an n-tuple $S = (s_1, \dots, s_n)$, one strategy for each player and $U_i(S)$ is the payoff for player i if the strategy profile is S.

A strategy s_i is said to be dominant over another strategy s_j , if the player will always do better by using s_i than s_j . That is, it always give a higher payoff irrespective of the opponents actions. A strategy $s_i \in S_i$ is strictly dominated for player i if there exists another strategy $s_i^* \in S_i$ such that for all $s_{-i} \in S_{-i}$, we have,

$$u_i(s_i^*, s_{-i}) > u_i(s_i, s_{-i}) \tag{3.6.1}$$

Table 3.9: Example of Outcome-related Classification of games

Outcome-related Classification		
Game	Constant-sum	Non-zero sum
Prisoner Dilemma		✓
Battle of Sexes		✓
Matching Pennies	✓	
Stag Hunt		✓

According to equation 3.6.1, we say that s_i^* strictly dominates s_i .

A strategy is said to be dominated if it yields a payoff less than or equal to the payoff of some other strategy of the opponent. A strategy $s_i \in S_i$ is a strictly dominant strategy for player i if it strictly dominates every other strategy in S_i . A strategy $s_i \in S_i$ is weakly dominated for player i if there exists another strategy $s_i^* \in S_i$ such that for all $s_{-i} \in S_{-i}$, we have,

$$u_i(s_i^*, s_{-i}) \geq u_i(s_i, s_{-i}) \quad (3.6.2)$$

As shown in equation 3.6.2, s_i^* weakly dominates s_i . A strategy $s_i \in S_i$ is a weakly dominant strategy for player i if it weakly dominates every other strategy in S_i .

Let us illustrate these concepts with an example. Refer to the payoff matrix of prisoner's dilemma in section 3.6. The strictly dominant strategy for prisoner 1 is *confess*, conversely, *Dont confess* is strictly dominated by *confess* for prisoner 1. As observe from the payoff matrix, *confess* will yield (3, 0) compared to *Dont confess* that will yield a payoff (4, 1) irrespective of prisoner 2's action. Similarly, a strictly

dominant strategy for prisoner 2 is *confess* while the strategy *Dont confess* is strictly dominated by *confess* for prisoner 2.

Table 3.10: Payoff matrix of Prisoners Dilemma

	Prisoner2(Confess)	Prisoner2(Dont confess)
Prisoner1(Confess)	(3,3)	(0,4)
Prisoner1(Dont confess)	(4,0)	(1,1)

3.6.2 Nash Equilibrium

Nash equilibrium is an important concept in game theory analysis. It is an action profile a^* with the property that no player i can do better by choosing an action different from a_i^* , given that every other player j adhere to a_j^* [109]. Formally, the action profile a^* is a Nash equilibrium if, for every player i and every action a_i of player i , a^* is at least as good according to player i 's preferences as the action profile (a^*, a_i^*) in which player i chooses a_i while every other player chooses a_j^* . Equivalently, for every player i ,

$$u_i(a^*) \geq u_i(a_i, a_i^*) \quad \forall \text{ action } a_i \text{ of player } i \quad (3.6.3)$$

Where u_i is a payoff function that represent player i ' preference. In other word, each player strategy is a best response to the strategies of others, no player has an incentive to deviate to an alternative strategy [130]. This is the stable point of the game in which each player predicts the actions of others correctly. Hence, each player cannot

gain by changing his strategy, keeping the strategies of other fixed. Let us illustrate this concept with an example. Refer to the Prisoners Dilemma [109] in section 3.6. In this case, the best choice for each player is to confess. But note that irrespective of whatever suspect 1 does, it is better for suspect 2 to confess ($3 < 4$) and ($0 < 1$). The strategy (confess, confess) yield 3 years jail term for both and strategy (Dont confess, Dont confess) yield 1 year behind bar for both. The Nash equilibrium of this game is that both confess. No single player can gain by unilaterally changing his strategy.

There are game with none, one or multiple equilibria. In discussing Nash equilibrium, we have to consider pure and mixed strategy separately. Some game might have an equilibrium in it mixed strategy but non in it pure strategy form. Example of a game with one Nash equilibrium is the prisoner' dilemma. Some may not have Nash equilibrium in it pure strategy form such as in the case of matching pennies, see section 3.4. In this game no player can predict the response of the other player. However, equilibrium exists in the mixed strategy form. Others might have multiple equilibrium as in the stag hunt game discussed in section 3.4. It has two equilibria. The Nash equilibria in this game are the action profiles (stag, stag) and (hare, hare). These can be referred to as payoff dominant equilibrium and risk-dominant equilibrium respectively.

Table 3.11: Payoff matrix of Battle of Sexes Game

	Husband(Opera)	Husband(Football)
Wife(Opera)	2, 1	0, 0
Wife(Football)	0, 0	1, 2

Mixed Strategy Nash Equilibrium

We will explain this concept using the Battle of Sexes game discussed in section 3.4. Suppose both husband and wife randomize their preferences. This means that they agree between them to alternate watching opera and football. Let us view the game from the wife point of view, and focus on how she evaluates her two pure strategies of definitely watching football or definitely watching opera. Suppose the probability of husband watching opera is p and football $1-p$. The expected utility of the wife's action U_w is $U_w(\text{Football}, S_h) = 0 \cdot p + 1(1 - p)$ and $U_w(\text{opera}, S_h) = 0(1 - p) + 2p$, where S_h is the husband's strategy. If the wife randomize her two actions. It resulted into same utility. $0 \cdot p + 1(1 - p) = 0(1 - p) + 2p$, so $p = \frac{1}{3}$. The husband's mixed strategy is $S_h(\text{Opera}) = \frac{1}{3}$ and $S_h(\text{football}) = \frac{2}{3}$.

Similarly, the wife's mixed strategy S_w is $S_w(\text{Opera}) = \frac{2}{3}$ and $S_w(\text{football}) = \frac{1}{3}$. At equilibrium, P (wife get 2, husband get 1) = $\frac{2}{3} \cdot \frac{1}{3} = \frac{2}{9}$ P (wife get 1, husband get 2) = $\frac{1}{3} \cdot \frac{2}{3} = \frac{2}{9}$ p(both get 0) = $\frac{1}{3} \cdot \frac{1}{3} + \frac{2}{3} \cdot \frac{2}{3} = \frac{5}{9}$. Thus, the expected utility for each player is $\frac{2}{3}$.

CHAPTER 4

BITTORRENT SYSTEM

BitTorrent is one of the most popular file sharing protocol implementing P2P technology [69]. It has enabled a powerful community of large content distributions [70, 71]. The BitTorrent system comprises of the following components, tracker(s), peers and files. The *tracker* is a server that coordinates sharing amongst interacting peers. The tracker coordinates the sharing process by keeping track of file owners and peers currently sharing files. A peer can either download or upload a file. A peer with the complete file is known as a *seeder*, while a peer with incomplete file is called a *leecher*. The set of all peers sharing a particular file is known as *swarm*. For each file, there is only one swarm kept by the tracker.

The whole file sharing process using BitTorrent can be divided into three phases, file announcement, file downloading and peer status update as shown in Figure 4.1. The file announcement phase involves

metadata creation which is the process of generating a metadata file by the owner of the file. The metadata provides all the necessary information to share the file. The file downloading phase is subdivided further into metadata discovery, peer discovery and file dissemination. Metadata discovery involves the process of searching for the metadata file. The process of peers finding each other to share a file is called peer discovery. File dissemination includes peer selection and piece selection, while peer status update involves providing updates to the tracker by peers during file sharing. Subsequent subsections explain each of these processes in detail.

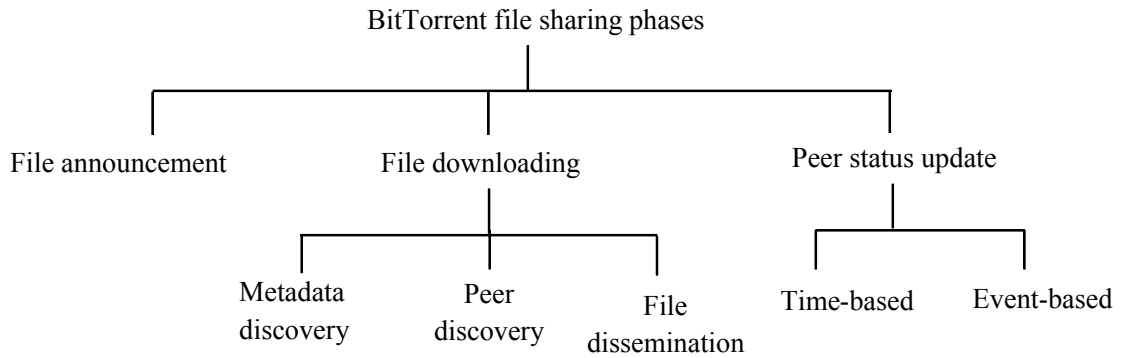


Figure 4.1: Phases involved in uploading or downloading a file in BitTorrent.

4.1 File Announcement

The aim of this phase is to make the file available for sharing by the file owner. For instance, if peer x wants to share a file using BitTor-

rent, peer x divides the file into pieces of equal sizes. Typically, these pieces are of sizes 16KB, 128KB, 256KB and 512KB. Afterwards, peer x creates a *.torrent* file that contains filename, the number of pieces, the length of a piece, creation date, and creator's name. The hash of each piece is also included in the *.torrent* so that the download can be verified piece by piece. The *.torrent* also contains the URL(s) of the tracker(s) obtained by searching tracker sites. Some examples of tracker sites can be found in [136, 137]. Once the *.torrent* file is created, it is uploaded to one or more torrent sites as well as to one or more trackers. Torrent sites and tracker sites can be found using a search engine. An example of metadata is shown in Figure 4.3. Figure 4.2 summarizes the steps involved in file announcement.

4.2 File Downloading

File downloading can be broken down into three phases, metadata discovery, peer discovery and file dissemination. Each of these phases is explained in detail in the following subsections.

4.2.1 Metadata discovery

When a peer needs to download a particular file, the peer retrieves the *.torrent* file by using any search engine and searching for torrent sites. Examples of torrent sites are Torrentz [138] and piratebay [139].

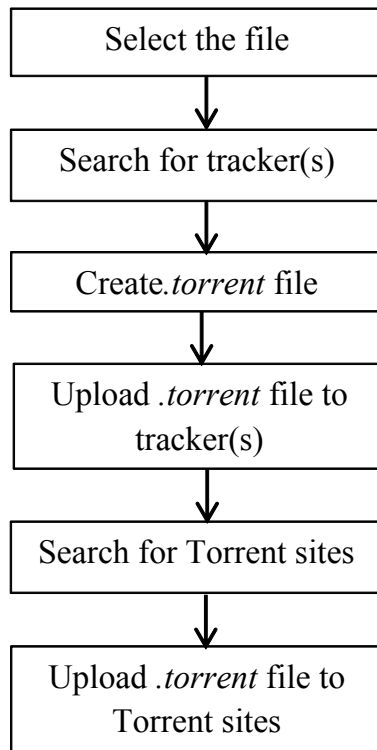


Figure 4.2: File announcement steps.

```
'announce': 'http://bigfoot1942.sectori.org:6969/announce',  
            'udp://tracker.ccc.de.80/announce'  
            'udp://tracker.openbittorrent.com/announce'  
'info':  
  {  
    'name': 'Ebira2013-AVESEQ01.DAT',  
    'piece length': 816 x 128kB,  
    'length': 101MB,  
    'pieces hash': '8ee6abbfa6161fb9439bff40f0b36eeef5e710 '  
    'created on': 9/22/2013 5.36:42pm  
    'created by': utorrent/3310  
  }
```

Figure 4.3: An Example Metadata.

Lastly, the peer contacts the tracker by sending a download request. Information contained in the download request include total amount uploaded, total amount downloaded, the number of bytes left to be downloaded. The download request also contains event description with these possible values; (a) *completed*, which is sent when the client finishes downloading the complete file, (b) *started*, which is sent by a client when it begins downloading and (c) *stopped*, which is sent when a client stops downloading. Figure 4.4 summarizes the steps involved in metadata discovery.

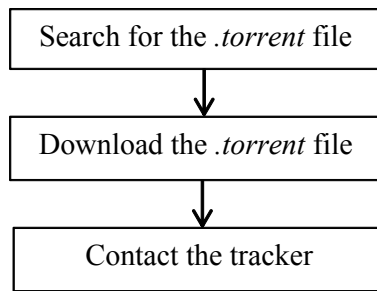


Figure 4.4: Metadata discovery steps.

4.2.2 Peer discovery

Peer discovery is done through the tracker which keeps statistics about peers. A peer sends a download request to the tracker. The reason for this download request is that a peer wants to join an existing swarm. In this case, the peer will send *announce* request to the tracker. Moreover, the peer may also send *announce* request to the tracker to ask for more peer set. This occurs if the number of peers in the peer set drops below

a set threshold, typically 20 [140].

The tracker responds to the peer with a set of peers known as *peer set* containing randomly selected 50 peers [141]. It should be noted that a peer set is a subset of the swarm and a peer in the peer set is known as a neighbor. The peer set is sent as a "text/plain" document containing all the necessary information required by the peer to download the file. The information contained in the "text/plain" document includes; (a) *peer key*, which is a dictionary containing map list of peers. Each map represents a peer with *peer-id*, *peer-ip* , *peer-port-number*, (b) *tracker-id*, (c) *complete* which indicates the number of peers having the complete file and (d) *incomplete* which indicates the number of peers with the incomplete file. Consequently, the peer contacts these peers specified in the peer set to initiate connection and commence file dissemination.

4.2.3 File dissemination

In BitTorrent, the dissemination of files is done directly by peers without the knowledge of the tracker. After receiving the peer set from the tracker, peers connect to each other by sending *handshake* messages. This is followed by sending *bitfield* message to announce the pieces initially each peer possesses. Consequently, each peer announces a successfully downloaded and verified piece through a *have* message

which is sent to all of its neighbors.

Once a provider announces a piece, the provider selects interested requesters by sending *unchoke* message. If the interested requester decides to get the piece, the upload process starts by sending a *piece* message to the requester.

At the requester side, the requester sends an *interested* message to the provider if the requester is interested in any piece announced by the provider. Once the requester receives an *unchoke* message from the provider, then the requester sends a *request* message for a particular piece and the download process starts. It should be noted that in case of a situation where a peer sends many requests for the same piece, it sends a *cancel* message as soon as it gets the piece. Furthermore, a *keep alive* message is sent at regular interval to keep the connections between two peers alive. The interleaving of messages between a provider and a requester is illustrated in Figure 4.5. A summary of messages and their descriptions during file dissemination, is as shown in Table 4.1

4.3 Peer status update

All peers in a swarm need to contact the tracker to update their status. This update may be time-based or event-based triggered. In the time-based update, each peer contacts the tracker periodically, typically every 30 minutes, to update the tracker of their status [141]. This

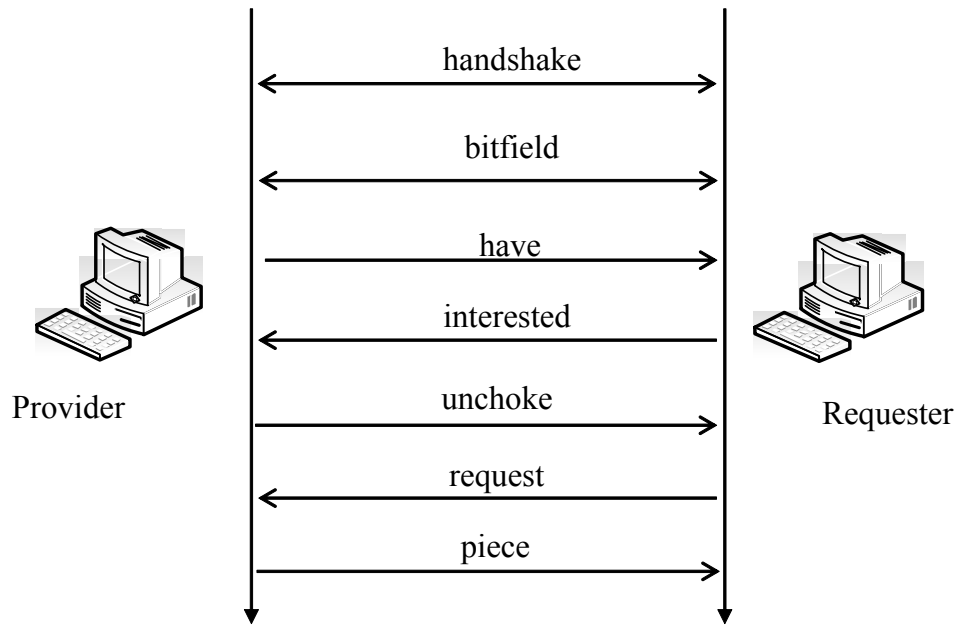


Figure 4.5: Message interleaving during file dissemination.

status update might include updating a peer’s active status. In case of a situation in which an existing peer in the swarm does not contact the tracker after a predefined time, usually 45 minutes, the tracker removes the peer from the swarm assuming the peer left the system [39]. In addition, in order to avoid overwhelming the tracker with requests and updates, a predefined minimum interval of requests can be enforced. This enables the tracker to update its statistics and maintain system up-to-date active list. Furthermore, event-based update occurs when a peer sends an update due to an occurrence of some events such as *stopped*, indicating the peer stopped downloading. The event-based update will also occur when a leecher sends a *complete* event to tell the tracker to change its status from a leecher to a seeder.

Table 4.1: BitTorrent messages

Message	Description/usage
choke	Notifying receiver of temporary refusal to upload by the sender.
unchoke	A notification message to indicate the willingness of the sender peer to send a piece.
interested	Indicates the sender is interested in the receiver's piece.
not interested	Shows that sender is not interested in the receiver's piece.
have	To inform others of file pieces the sender has successfully downloaded and verified.
bitfield	Sent immediately after a successful handshake. It contains bitmap, using single bit to represent piece availability.
request	Used by the sender to request a piece.
piece	This follows a request message which contains piece index, length and the data requested.
cancel	This is used to cancel request made.
handshake	The first message to be exchanged by both peers.

4.4 Illustrative Example of File Sharing

The interaction process is depicted in Figure 4.6 and summarized into the following listed steps.

- (a) Peers use a search engine to search for torrent site(s) in order to locate the *.torrent* file
- (b) Peers download the *.torrent* file from torrent site(s)
- (c) Peers contact the tracker for a peer set
- (d) The tracker sends the peer set
- (e) Peers select their neighbors and establishes connections with them
- (f) When connection is established, the file dissemination process can take place

We use the following example to illustrate the file sharing process using BitTorrent. Let us assume that peer 1 and peer 2 are interacting where peer 2 is sharing file F_x and peer 1 is interested to download F_x . First, peer 2 divides F_x into pieces and creates a *.torrent* file with the file metadata. Peer 2 announces the *.torrent* file to one or more torrent sites and trackers. When peer 1 retrieves the *.torrent* file and sends a download request to the tracker, the tracker adds peer 1 as a leecher to the swarm of F_x . The tracker replies to peer 1 with a peer set that contains only peer 2. Let us assume that while peer 1 is downloading from peer 2, a new peer (peer 3) joins the swarm of F_x . Peer 3 will get a peer set that contains peer 1 and peer 2. Hence, peer 3 can download pieces of F_x from peer 1 or peer 2. Once a peer 1 or peer 2 completely downloads F_x , the tracker changes their status to seeders in the swarm.

4.5 Piece Selection Strategy

In BitTorrent, files have different number of pieces depending on file size and piece size. Peers download files piece by piece, hence there is a need for a piece selection strategy. Selecting a suitable piece to download is very important to the performance of the network. For instance, if every peer in a peer set is downloading one piece owned by only one peer, this will definitely slow down the entire download

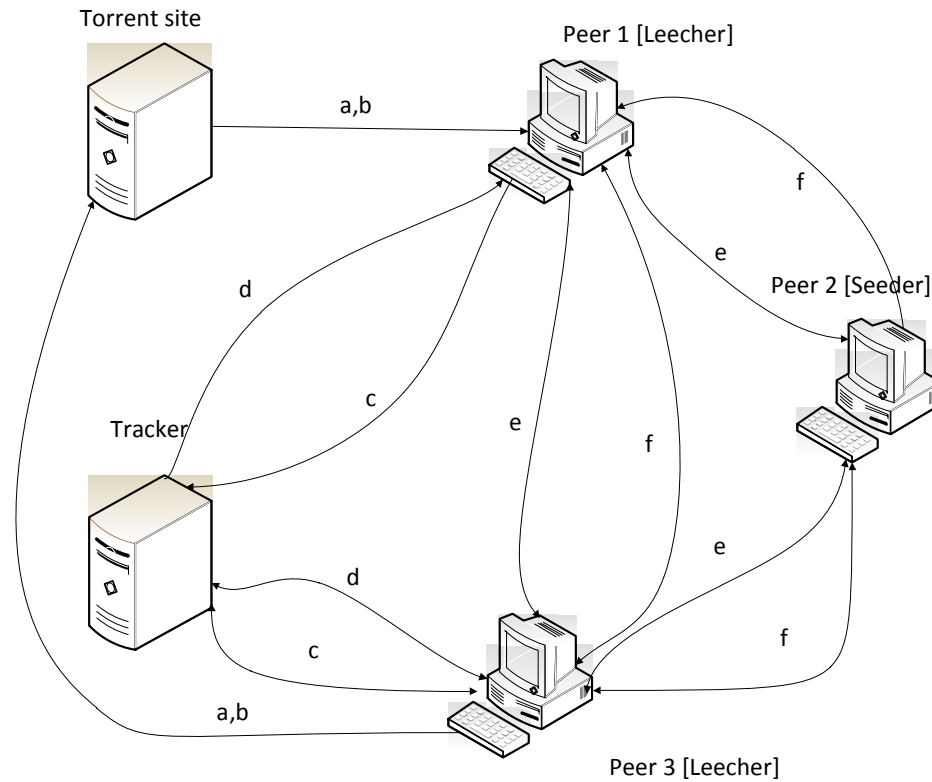


Figure 4.6: BitTorrent file sharing process.

process. There are piece selection mechanisms applying different selection criteria, namely; rarest piece first, strict priority, random piece first and end game.

- i. **Rarest piece first:** Every peer maintains the list of all the copies of each piece in its peer set. A rarest piece is the piece with the fewest number of copies or owners in a peer set [141]. A peer using this mechanism selects next rarest piece to download so that the peer has the piece many other peers want to download. This will also increase the possibility of peers offering pieces throughout the downloading process and leave the common pieces to a later time for download.

- ii. **Strict priority:** This mechanism is to ensure that once a piece is selected, all the remaining blocks from that piece must be downloaded next before downloading other pieces. This will speed up the download process since the verification can be done faster.
- iii. **Random piece first:** A new peer that joins the swarm needs to have a complete piece as soon as possible. This facilitates the new peer to start as early as possible exchanging of pieces with other peers.

4.6 Choking Algorithm

Since BitTorrent is a resource sharing environment, peers have to decide with whom to interact. This is made through the choking algorithm. Peers can randomly or intelligently select other peers to exchange pieces with. In a BitTorrent environment, seeders and leechers have different strategies which are applied at fixed intervals [86]. Choking is a term used in BitTorrent meaning a temporary refusal to upload pieces to a peer. This is to encourage other peers to share and cooperate.

As shown in Figure 4.7, every peer in the BitTorrent environment applies the choking algorithm at fixed time intervals to select peers and give them chance to download. To achieve this, the algorithm is divided into regular unchoke and optimistic unchoke as shown in Figure 4.8. The regular unchoke is performed differently by seeders and leechers

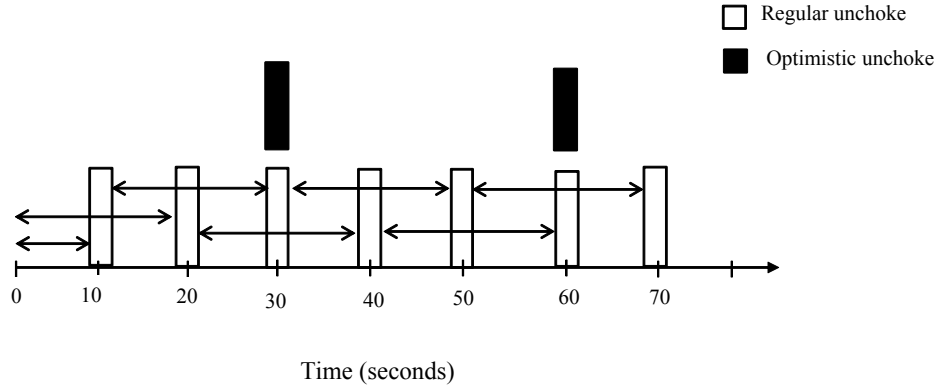


Figure 4.7: Choking algorithm time intervals.

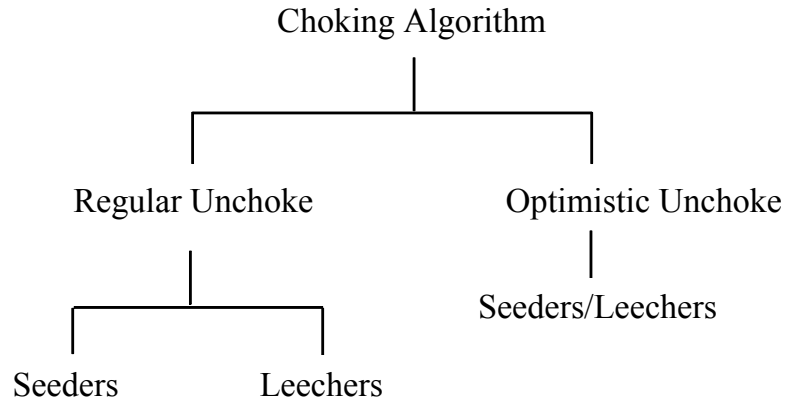


Figure 4.8: Choking algorithm classification.

where as optimistic unchoke is performed in the same manner by seeders and leechers. The regular unchoke is done every 10 seconds, and it looks back to the last 20 seconds, whereas optimistic unchoke is done every 30 seconds as illustrated in Figure 4.7.

The objective of regular and optimistic unchoke during every time interval is to give a chance to unchoke other peers. A peer can unchoke upto three peers during regular unchoke and one peer during optimistic unchoke.

Regular and optimistic unchoke is done in phases, namely candidate selection, candidate sorting criteria, best candidate selection and unchoke decision making. In the candidate selection phase, a peer selects candidates from its peer set to be unchoked, where as in candidate sorting criteria, the candidates selected are sorted based on either the download or upload rate. Best candidate selection is used to select the best candidates from the sorted candidates while during unchoke decision making phase, a peer decides whether to unchoke the selected best candidates.

4.6.1 Regular Unchoke

Regular unchoke is done by seeders and leechers every 10 seconds. In applying the regular unchoke, seeders and leechers use the same candidate selection phase but they use different candidate sorting criteria, different best candidate selection algorithm and different unchoke decision making process.

Leechers Regular Unchoke

Each leecher maintains a list of its download rate from its neighbours. During regular unchoke, every leecher runs regular unchoke algorithm as shown in Algorithm 1. A leecher selects all the interested candidates from its peer set as illustrated in lines 1 - 6. The leecher then sorts

the candidates based on their upload rate to the leecher in the last 20 seconds and selects the top three uploaders as shown in line 7. It should be noted that in regular unchoke, a leecher needs to select the top three uploaders. If the leecher cannot select the top three uploaders due to unavailability of record in the last 20, the leecher then complements the top uploaders list with randomly choosing from its peer set as illustrated in lines 9 - 11. As such, the leecher uses tit-for-tat to reciprocate with others. After selecting the top three uploaders, the leecher finally makes the unchoke decision as illustrated in lines 12 - 19. The unchoke decision is applied to all the selected top three uploaders. The leecher checks whether it is snubbed by any of the top three uploaders. A peer considers itself snubbed by its neighbor if the peer downloaded from the neighbour but not in the last 60 seconds. If the leecher is snubbed by any of its top three selected uploaders, the leecher chokes that particular uploader, otherwise the leecher unchoke the uploader. The leecher then chokes the remaining peers in its peer set. Each leecher runs the snubbing algorithm in the last snub point as illustrated in Figure 4.9.

Seeders Regular Unchoke

Each seeder maintains a list of its upload rate to peers in its peer set. During regular unchoke, a seeder selects all the interested candidates from its peer set. Then, the seeder sorts the candidates based on its

Algorithm 1 Leecher regular unchoke algorithm

Frequency: Every 10 seconds

Input: List of all peers in the peer set

Output: Unchoke zero to three peers and choke the rest

```
//Candidate selection//
1: CandidateSelected  $\leftarrow$  {}
2: for all peers in the peer set do
3:   if a peer is interested then
4:     CandidateSelected  $\leftarrow$  CandidateSelected + peer
5:   end if
6: end for
// Candidate sorting criteria//
7: Sort all peers in CandidateSelected based on their upload rate to me in the
   last 20 seconds
//Best candidate selection//
8: Select upto three top uploaders from CandidateSelected
9: if the number of interested peers is less than  $< 3$  then
10:   Complement the remaining by randomly selecting from the peer set
11: end if
// Unchoke decision making//
12: for every peer among the three best candidates selected do
13:   if the peer did not snub me then
14:     Unchoke peer
15:   else
16:     Choke peer
17:   end if
18: end for
19: Choke the rest of the peers in the peer set
```

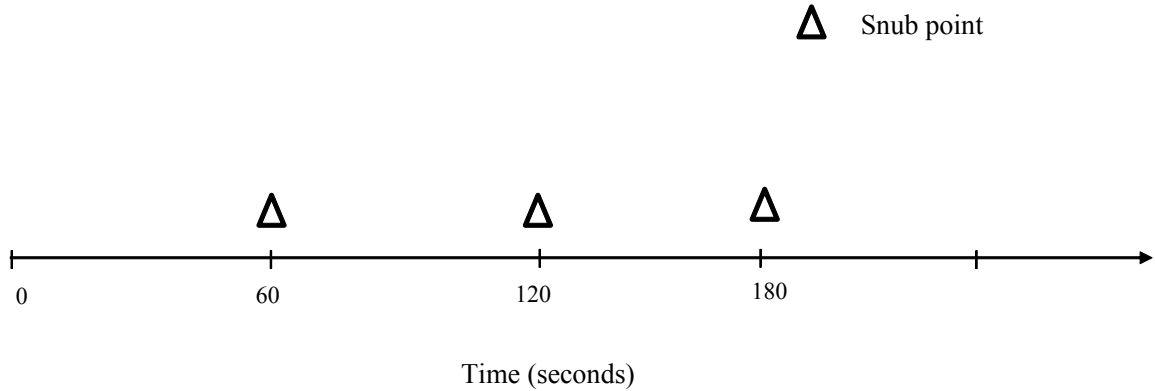


Figure 4.9: Snubbing algorithm time intervals.

upload rate to them in the last 20 seconds. Thereafter, the seeder selects the top three downloaders as its best candidates. However, in case the top candidates selected by the seeder are less than three, the seeder complements the best candidate list with randomly choosing a peer from its peer set. This may occur if the seeder does not have three upload records of the candidate selected in the last 20 seconds, or the interested candidates are less than three. During the unchoke decision process, the seeder unchokes the three top candidates. It should be noted that seeders do not apply snubbing algorithm during their unchoke decision making phase because they do not require to download from any peer in their peer set. Also, seeders select based on its upload rate to those top downloaders and unchokes them. The objective of the seeder is to search for fast downloaders in the last 20 seconds so as to distribute its pieces as fast as possible to many leechers to become seeders and will reduce the load on original seeders. The regular unchoke algorithm applied by seeders is shown in Algorithm 2.

As a summary, the phases of regular unchoke for seeders and leechers are illustrated in Figures 4.10 and 4.11.

Algorithm 2 Seeder regular unchoke algorithm

Frequency: Every 10 seconds

Input: List of all peers in the peer set

Output: Unchoke three peers and choke the rest

```
//Candidate selection//
1: CandidateSelected  $\leftarrow$  {}
2: for all peers in the peer set do
3:   if a peer is interested then
4:     CandidateSelected  $\leftarrow$  CandidateSelected + peer
5:   end if
6: end for

//Candidate sorting criteria//
7: Sort all peers in CandidateSelected based on their download rate from me in
   the last 20 seconds

//Best candidate selection//
8: Select three top downloaders from CandidateSelected
9: if the number of interested peers is less than  $< 3$  then
10:   Complement the remaining by randomly selecting from the peer set
11: end if

//Unchoke decision making //
12: Unchoke the three best candidates selected
13: Choke the rest of the peers in the peer set
```

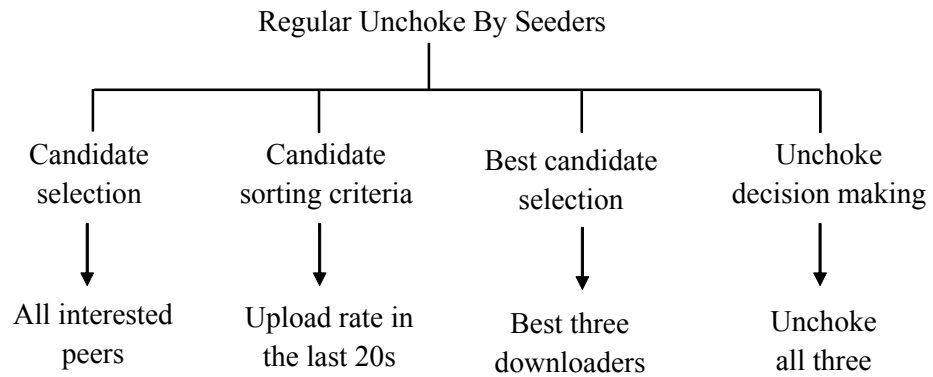


Figure 4.10: Regular unchoking steps by Seeders.

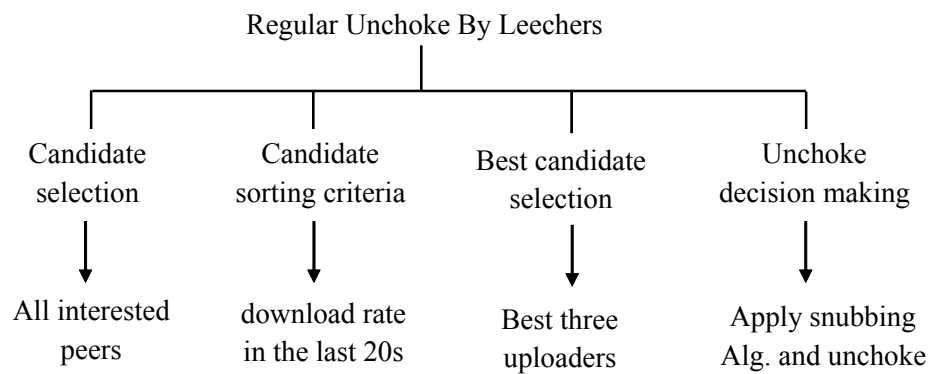


Figure 4.11: Regular unchoking steps by Leechers.

4.6.2 Optimistic Unchoke

Optimistic unchoke is run by seeders as well as leechers in the same manner. Optimistic unchoke refers to random selection of one peer to unchoke. This helps in searching for a better peer to compete for one of the best candidates. Optimistic unchoke also facilitates the bootstrapping of new peers joining the swarm without any piece to exchange with others [39].

Algorithm 3 Seeder and leecher optimistic unchoke algorithm

Frequency: Every 30 seconds

Input: List of all choked peers in the peer set

Output: Unchoke one peer and choke the rest

```
//Candidate selection//  
1: CandidateSelected  $\leftarrow$  {}  
2: for all peers in the peer set do  
3:   if a peer is choked then  
4:     CandidateSelected  $\leftarrow$  CandidateSelected + peer  
5:   end if  
6: end for  
7: Randomly select one peer from CandidateSelected  
//Unchoke decision making //  
8: Unchoke the selected peer
```

Optimistic unchoke is carried out in every 30 seconds by both seeders and leechers as shown in Algorithm 3. The one peer selected during the optimistic unchoke is added to the unchoked top candidates only for the next 10 seconds. After that, regular unchoke is applied as shown in Figure 4.12. The peer has no candidate initially from time 0. Between the 10th and 20th seconds, the peer has three candidates from the regular unchoke at time 10. This continues until the 30th seconds, when the first optimistic unchoke is run by the peer to add one additional peer in addition to the three from the regular unchoke ran at time 20. As from 30th second till the 40th seconds when the next regular unchoke will be run, the peer has four connections. Once the peer runs the next regular unchoke at 40th seconds, three peers are unchoked and the rest will be choked. Hence, the peer's connection returns to three until the next optimistic unchoke and the cycle continues.

As shown in Figure 4.13, in optimistic unchoke, the candidate is selected from all the choked peers in the peer set. There is no sorting criteria as the best candidate is selected randomly from the candidate list, hence candidate sorting criteria is not applicable (NA). Finally, this selected peer is unchoked. The summary of optimistic unchoke steps is presented in Figure 4.13.

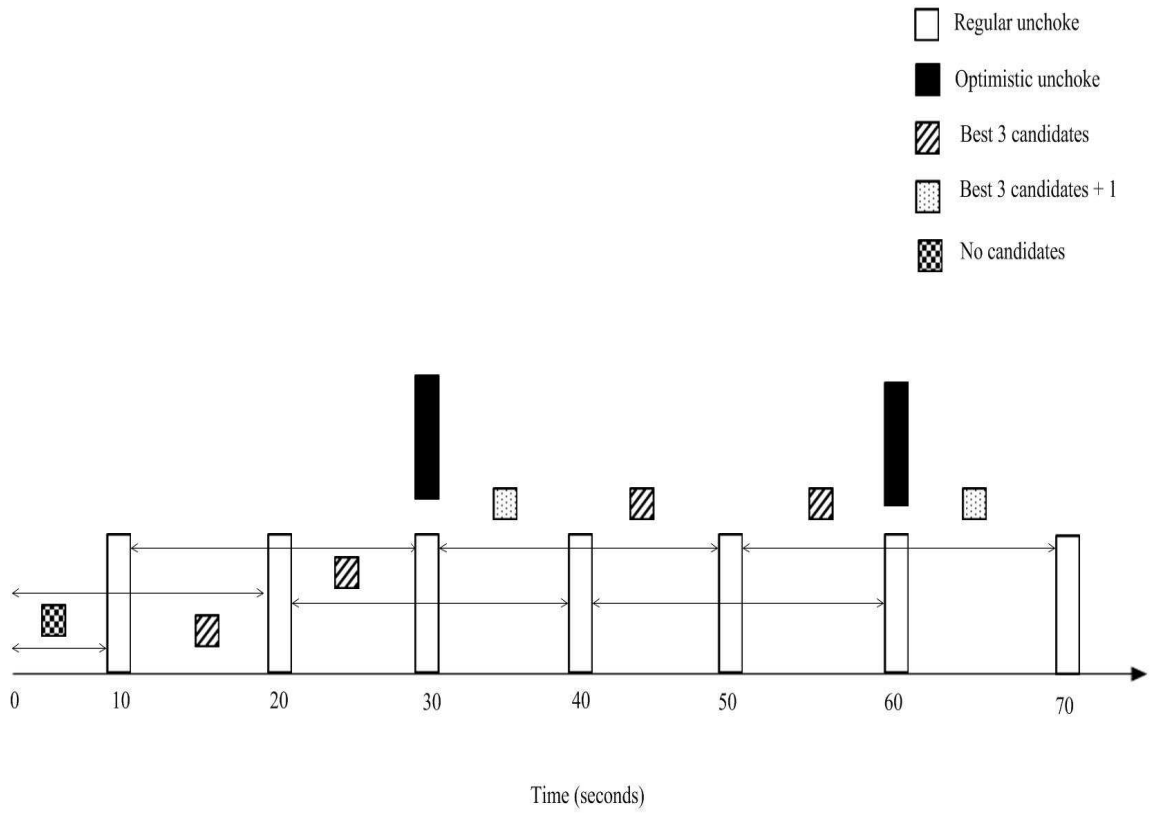


Figure 4.12: Regular and Optimistic unchoking time intervals.

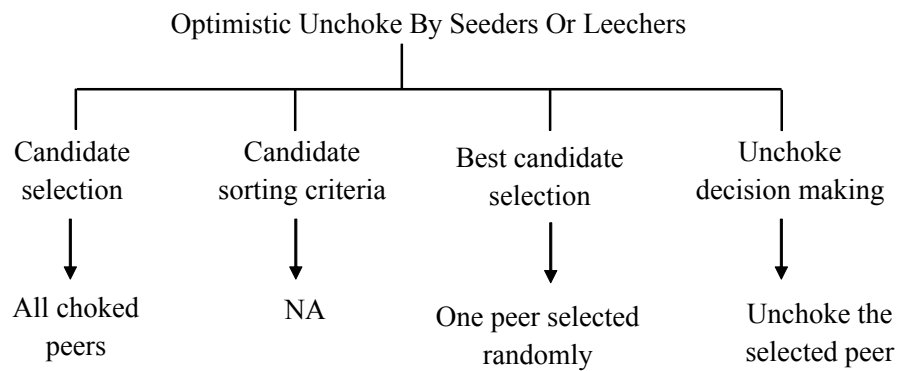


Figure 4.13: Optimistic unchoking steps.

CHAPTER 5

CHOKING ALGORITHM GAME MODELLING

5.1 Overview

The goal of choking algorithm is to improve the performance of BitTorrent [86, 67, 68]. Peers should maximize the utility of available resources, motivate others to contribute and resist free riding behavior [39]. In this chapter, we use game theory to model the choking algorithm so that peers can determine their best strategy to interact with others. The overall objective of our game model is to inhibit free riding, increase fairness and encourage resource contribution. This is necessary to ensure a fair and stable BitTorrent system.

5.2 Motivation

Despite the popularity and the continuous usage of BitTorrent as a file sharing system [142, 67, 68], it is bedeviled with some challenges. Developers and researchers alike are exerting efforts to address these shortcomings and improve the performance of one of the most successful P2P protocols today [67, 68, 39]. Among researchers, free riding in BitTorrent has been a widely acknowledged problem [39, 67, 68, 72]. Free riders misuse the BitTorrent environment through both optimistic and regular unchoke.

The optimistic unchoke algorithm has been fingered as one of the weaknesses of BitTorrent [67, 68, 143, 115, 144, 120] since selecting a peer to unchoke is random and does not consider reciprocity. On the other hand, the optimistic unchoke is vital and crucial in BitTorrent and is needed for better performance [144]. This is because new peers get their initial pieces through the optimistic unchoke. Moreover, optimistic unchoke is the opportunity to shop for better peers in terms of bandwidth and pieces. Because optimistic unchoke is run by seeders and leechers, free riders take advantage of seeders and leechers alike during the optimistic unchoke.

Free riders also exploit regular unchoke executed by seeders. Analyzing Algorithm 2 in Section 4.6.1, free riders can be members in the candidateSelected set and also can be selected among the three best

candidates either due to their download rate or due to the random selection. Since, a seeder always unchokes the three best candidates, free riders will be unchoked if they are members in the three best candidates.

Furthermore, free riders exploit regular unchoke executed by leechers. Analyzing Algorithm 1 in Section 4.6.1, free riders can be members in the candidateSelected set. Free riders can also be selected among the three best candidates either: (a) due to zero upload rate of all the peers in the candidateSelected set or (b) due to the random selection to complement the three best candidates. Once a free rider is selected amongst the three best candidates, a free rider has an advantage over a non-free rider. A free rider never snubs whereas a non-free rider might snub a leecher. Therefore, a free rider among the three best selected peers is always unchoked by a leecher whereas a non-free rider among the three best selected peers might be choked or unchoked.

5.3 Motivation for Using Game Theory in Modelling BitTorrent Choking Algorithm

In a P2P environments, there are peers and resources. Due to the nature of P2P environment, peers compete with each others to for resource and attempt to maximize their profit and minimize their

cost. Therefore, P2P is an environment with players that compete for resources. These scenario fits well in game theory. Resources in P2P environments such as BitTorrent have been likened to common goods in Economics [6]. Game theory is the most used tool in the study of interactions between rational players in Economics. Similarly, game theory provides a rich analytical framework for researchers of P2P interactions and incentives as a modeling and analysis tool. Recently, the increase in the application of game theory in P2P modeling can be attributed to the need to have a mathematical framework for the design and analysis of the growing P2P online community. Researchers [110, 117, 66, 116, 64, 61, 65] claimed that game theory is a natural choice for this task of modeling these vast and complex decision making interactions due to the following. (i) the structure of utility functions in game theory that allows benefit to be modelled in terms of gain and cost. In addition, game theory provides avenue for incorporation of other approaches. This suitability of game theory modeling that allow plug in of other approaches strengthened its position amongst other approaches. For instance, trust, reputation, incentives and reciprocity are deeply studied and used in game theory. Similarly, they are all required in the study and modeling of P2P systems. (ii) the availability of existing game models that have been studied extensively in game theory which can easily be adapted to the economics

of P2P (iii) the uncertainties that exist in economics which are best modeled with game theory also arise in P2P interactions. These uncertainties in decision making in P2P can also be suitably modeled with the aid of game theory (iv) the rationality assumption that every player will always pursue their maximum gain in game theory suits P2P environment (v) game theory framework provides formal representation of interactions in P2P systems (vi) game theory allows the analysis of P2P system equilibrium and therefore can stand on its own in the model and study of P2P interactions.

5.4 The Game Model

In this section, we model the choking algorithm as a game. In our model, we assume that peers are rational. This means that every peer would pursue its maximum profit. This is in accordance with the assumptions of game theory [65].

Peers gain or incur cost when downloading resources they want or uploading resources others have requested. Each peer gains when receiving resources. We express the downloaded quantity of resources as Q_d . Therefore, the gain function is expressed as $g(Q_d)$. In the same manner, peers incur cost when uploading resources of quantity Q_u . Hence, the cost function is expressed as $c(Q_u)$. As such, the payoff is expressed as $g(Q_d) - c(Q_u)$ and depends on whether a peer has chosen to upload

or download resources.

As shown in Figure 4.7, Section 4.6, a peer performs regular unchoke and optimistic unchoke every 10 and 30 seconds, respectively. As such, a peer plays a game every 10 and 30 seconds as long as the peer is part of the BitTorrent environment. It should be noted that every third ten-second period, a peer plays two games, one is the ten-second game and the other one is the thirty-second game.

In our model, each peer can be one of four types, Initial Seeder (IS), Experienced Seeder (ES), Leecher (LE), and Free Rider (FR). The ISs are peers having the complete file since the time of joining the peer set and are always sharing with their neighbors. Therefore, ISs do not send download requests and they are only uploaders. Peers having an incomplete file (i.e., missing pieces of the file) are called LEs. That is, LEs send download requests to download the missing pieces and might share the pieces they have. When LEs get the complete file, they are called ESs. The ESs do not send download requests and they might share with their neighbors. The last peer type is FRs. This type of peers does not share with any other peer but sends download requests to get pieces from others. Therefore, they have a selfish behavior.

Free riders exploit the lack of reciprocation with seeders, since seeders only upload to others. Therefore, once a free rider becomes one of the three best candidates, it will keep on downloading without con-

tributing. To overcome this shortcoming, we were motivated to divide seeders into ISs and ESs. This is done to minimize the effect of free riders through seeders since both ISs as well as ESs allocate their bandwidth to others but they use different selective strategies to inhibit free riding.

5.4.1 Game Model For Initial Seeder

Every 10 seconds, ISs play the following game. In this game, ISs play against all interested neighbors in the peer set. The strategy of ISs is to upload to neighbors such that the standard deviation among the neighbors in terms of the number of uploaded blocks is minimized. The formula of standard deviation is shown in 5.4.1.

$$SD = \sqrt{\frac{1}{N-1} \sum_{i=1}^{N-1} (x_i - \bar{x})^2} \quad (5.4.1)$$

Where \bar{x} is the mean of the number of blocks downloaded by each neighbor, N is the size of the peer set and x_i is the number of blocks downloaded by neighbor i .

Since ISs are always sharing, their bandwidth is basically allocated for others. Hence, ISs ignore the cost incurred when uploading to others. Therefore, $c(Q_u) = 0$. Since, self satisfaction is the gain for ISs, the payoff function for ISs is expressed as $g(Q_u)$.

In a nutshell, the game played by ISs has the following components:

Table 5.1: Game Model: Initial Seeder to an opponent

	Opponent (Choke)	Opponent (Unchoke)
Initial Seeder (Choke)	(0,0)	(0,0)
Initial Seeder (Unchoke)	$(g(Q_u), g(Q_d))$	$(g(Q_u), g(Q_d))$

Set of player = {interested neighbors as well as the IS}

Strategy = upload to neighbors such that the standard deviation among the neighbors in terms of the number of uploaded blocks is minimized

Set of actions = {choke, unchoke}

Payoff function = $g(Q_u)$

An IS can play against any other peer type except ISs and ESs. Since an IS does not require to download from others, it will treat all of its opponents the same. Table 5.1 shows the game model of an IS with an opponent. If the IS unchokes an opponent, the IS will gain by assuring that its payoff function is achieved. Otherwise, the IS will choke the opponent. On the other hand, if the opponent is unchoked by the IS, the opponents payoff is $g(Q_d)$. Otherwise, the opponent will gain nothing.

5.4.2 Game Model For Leecher

Every 10 seconds, LEs play the following game. In this game, the set of players are all the interested peers in the peer set. These interested peers may include other LEs and FRs. The strategy used by LEs is tit-

for-tat. When a LE uploads, the LE incurs a cost expressed as $c(Q_u)$. A gain is expressed as $g(Q_d)$ a LE get when downloading. Tables 5.2 shows the game model when LEs play against LEs. When both LEs choke each other, no one gains and no one incurs cost. Also, when both LEs unchoke each other, both LEs gain is proportional to the quantity downloaded. Consequently, both LEs incur the associated cost of uploading to the other. However, if one LE unchokes the other LE while the other chokes the same LE in return, the unchoked LE will gain by downloading without incurring a cost. In that case, the LE that carry out the unchoke will incur the cost of uploading.

Furthermore, as shown in Table 5.3, in a game with a FR, if a LE chokes a FR, both LE and FR gain nothing. This occurs irrespective of whether the FR chokes or unchokes the LE. The LE's gain is always 0 since FRs do not upload to others. However, if a LE unchokes a FR, the FR gains by downloading from the LE while the LE incurs the cost of uploading to a FR.

The game components played by a LE are as follows:

Set of player = {interested neighbors as well as the LE}

Strategy = tit-for-tat

Set of actions = {choke, unchoke}

Payoff function = $g(Q_d) - c(Q_u)$

Table 5.2: Game Model: Leecher to Leecher

	Leecher (Choke)	Leecher (Unchoke)
Leecher (Choke)	(0,0)	$(g(Q_d), c(Q_u))$
Leecher (Unchoke)	$(c(Q_u), g(Q_d))$	$(g(Q_d) - c(Q_u), g(Q_d) - c(Q_u))$

Table 5.3: Game Model: Leecher to Freerider

	Free rider (Choke)	Free rider (Unchoke)
Leecher (Choke)	(0,0)	(0,0)
Leecher (Unchoke)	$(c(Q_u), g(Q_d))$	$(c(Q_u), g(Q_d))$

5.4.3 Game Model For Experienced Seeder

Every 10 seconds, ESs play the following game. In this game, ESs play against all interested neighbors in the peer set. The strategy of ESs is tit-for-tat. Since ESs do not download, their bandwidth is basically allocated for others but ESs' strategy dictates that if there is no reciprocation history with the opponent, then the opponent should not use the bandwidth. As such, ESs incur cost when uploading to FRs. Therefore, $c(Q_u)$ is only ignored if uploading to LEs. When uploading, ESs will gain $g(Q_u)$.

In a nutshell, the game played by ESs has the following components:

Set of player = {interested neighbors as well as the ES}

Strategy = tit-for-tat

Set of actions = {choke, unchoke}

Payoff function = $g(Q_u) - c(Q_u)$

An ES can play against any other peer type except ISs and ESs. Table 5.4 shows the game model of an ES with a LE. If the ES unchokes a

LE, the ES will gain by assuring that its payoff function is achieved. Otherwise, the ES will choke the LE. On the other hand, if the LE is unchoked by the ES, the LEs payoff is $g(Q_d)$. Otherwise, the LE will gain nothing.

Similarly, Table 5.5 shows the game model of an ES with a FR. If the ES unchokes a FR, the FR will gain by downloading from the ES. The FR's payoff is $g(Q_d)$. In addition, the ES incurs an associated cost for uploading to a FR. The ES's payoff is $g(Q_u) - c(Q_u)$. On the other hand, when both ES and FR choke each other, both gain nothing. Though, the ES does not need to download.

Table 5.4: Game Model: Experienced Seeder to Leecher

	Leecher (Choke)	Leecher (Unchoke)
Experienced Seeder (Choke)	(0,0)	(0,0)
Experienced Seeder (Unchoke)	$(g(Q_u), g(Q_d))$	$(g(Q_u), g(Q_d))$

Table 5.5: Game Model: Experienced Seeder to Free rider

	Free rider (Choke)	Free rider (Unchoke)
Experienced Seeder (Choke)	(0,0)	(0,0)
Experienced Seeder (Unchoke)	$(g(Q_u) - c(Q_u), g(Q_d))$	$(g(Q_u) - c(Q_u), g(Q_d))$

5.4.4 Game Model Every Thirty Seconds

When an IS wants to play a thirty-second game, the set of players consist of all choked neighbors in the peer set plus the IS himself. The IS's strategy is to discover better peers to become seeders as soon as possible. This strategy also assists in bootstrapping new peers. The

IS randomly selects a choked peer and unchoke it. This is a game of chance in which the outcome might be a failure or success. The IS gains if the outcome of the random selection is a success, that is, the selected peer is new or will be among the top downloaders, otherwise the outcome is a failure.

In addition, LEs and ESs play a game every 30 seconds which is basically the same as the game they play in every ten-seconds. The only difference is the set of players, which are all choked neighbors in their peer set.

5.5 Model Implementation

The proposed choking algorithm game is played by all peers in every 10 and 30 seconds in the life time of their interactions. To apply their strategies, players need information. The information base for players in this game is the number of blocks downloaded or uploaded from their opponents. This is proportional to the number of choke and unchoke messages. This is similar to the technique used in [145] that monitors the *have* messages from the regular BitTorrent in their study of effect of locality in reducing BitTorrent traffic. Each player choose an action to apply based on its strategy. The number of blocks uploaded to the target peer by the source peer and the number of blocks downloaded by the source peer from the target peer are the information peers used

in this game. With these information available to all players in the network, it serves as the information base for them to make their decision during the game.

Initial Seeder Game Implementation

The IS is an upload only player that does not receive pieces from other peers. To attain its' strategy of minimizing the standard deviation in term of number of blocks uploaded to peers in its peer set, an IS carry out the following steps during every 10 seconds as illustrated in Algorithm 4. An IS maintains the records of uploaded blocks to all peers in its peer set. It selects all interested peers in its peer set as candidates as illustrated in lines 1 - 4. Then, the IS sorts them based on the number of blocks uploaded to these candidate during the sharing process this is illustrated in line 7. Thereafter, it selects three neighbors with the least number of blocks as best candidates as illustrated in line 8. However, in case interested candidates are less than three, the IS complements the best candidates by randomly selecting a peer from its peer set as shown in line 9 - 11. This arises if the IS does not have three upload records of the candidates selected or the interested candidates are less than three. In the unchoke decision making phase, the IS unchokes the three top candidates selected and choke the rest peers in its peer set as illustrated in line 12 and 13 respectively. Let us illustrate this process with an example. Consider the numbers in

Table 5.6 as the number of blocks uploaded by an IS to its interested peers in its peer set $P_1, P_2 \dots P_8$ so far. During this game, the IS will select peer P_4, P_5 and P_3 as the three best candidates and unchokes them.

Table 5.6: Number of blocks uploaded by an IS to its neighbors

Peer	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9
No of blocks uploaded	100	50	5	0	68	0	78	62	12

During the thirty-second period, the ISs need to play a game to select one of its neighbors. As presented in Algorithm 5, the IS does not have a sharing experience with any peer in its peer set and hence, it always select one choked peer in its peer set and apply the action unchoke. This strategy gives new peers the chance to get some pieces from the IS. This action of IS is the same as the optimistic unchoke for seeders in original BitTorrent. However, FRs will also get unchoked through this window, but the effect is minimal since the number of IS in the network are always few compared to the number of LEs.

Algorithm 4 Initial Seeder Ten-second game

Frequency: Every 10 seconds

Input: List of all peers in the peer set

Output: Unchoke three peers and choke the rest

```
//Candidate selection//  
1: CandidateSelected  $\leftarrow$  {}  
2: for all peers in the peer set do  
3:   if a peer is interested then  
4:     CandidateSelected  $\leftarrow$  CandidateSelected + peer  
5:   end if  
6: end for  
  
//Candidate sorting criteria//  
7: Sort all peers in CandidateSelected based on the number of blocks downloaded  
   from me  
  
//Best candidate selection//  
8: Select three peers from CandidateSelected such that the standard deviation  
   among the peers in terms of the number of uploaded blocks is minimized  
9: if the number of interested peers is less than  $< 3$  then  
10:   Complement the remaining by randomly selecting from the peer set  
11: end if  
  
//Unchoke decision making//  
12: Unchoke the three best candidates selected  
13: Choke the rest of the peers in the peer set
```

Algorithm 5 Initial Seeder Thirty-second Game

Frequency: Every 30 seconds

Input: List of all choked peers in the peer set

Output: Unchoke one peer and choke the rest

```
//Candidate selection//  
1: CandidateSelected  $\leftarrow$  {}  
2: for all peers in the peer set do  
3:   if a peer is choked then  
4:     CandidateSelected  $\leftarrow$  CandidateSelected + peer  
5:   end if  
6: end for  
7: Randomly select one peer from CandidateSelected  
//Unchoke decision making //  
8: Unchoke the selected peer
```

Experienced Seeder Game Implementation

An ES uses tit-for-tat as the strategy. Though presently they do not require to download, ESs use their past experience in deciding which action to take. During the ten-second game, an ES selects three of its interested peers in its peer set as candidates. Then, it checks if it has an upload information for each of the selected candidate. Thereafter, the ES game is as shown in Algorithm 6. If the selected peer does not have any interaction with the ES, then the ES unchoke the peer. This

is to give new comers the opportunity to get pieces. Otherwise, the ES has downloaded from the peer while the ES was a LE, the ES unchokes the peer.

Algorithm 6 Experienced Seeder Ten-second Game

Frequency: Every 10 seconds

Input: List of all peers in the peer set

Output: Unchoke zero to three peers and choke the rest

```

//Candidate selection//
1: CandidateSelected  $\leftarrow$  {}
2: for all peers in the peer set do
3:   if a peer is interested then
4:     CandidateSelected  $\leftarrow$  CandidateSelected + peer
5:   end if
6: end for
//Candidate sorting criteria//
7: Sort all peers in CandidateSelected based on their download rate from me in
   the last 20 seconds
//Best candidate selection//
8: Select up to three top downloaders from CandidateSelected
9: if the number of interested peers is less than  $< 3$  then
10:  Complement the remaining by randomly selecting from the peer set
11: end if
//Unchoke decision making //
12: for every peer among the three best candidates selected do
13:   if my upload to the selected peer is zero then
14:     Unchoke peer
15:   else if my download from the selected peer is not zero then
16:     Unchoke peer
17:   else
18:     Choke peer
19:   end if
20: end for
21: Choke the rest of the peers in the peer set

```

The ES uses the same strategy and the same payoff function in the ten-second game for the thirty-second game as depicted in Algorithm 7 but with different set of players. The ESs use their experience during their interaction as LEs to apply their strategy. An ES selects one of its

choked neighbors randomly then checks if it has an upload information for the selected peer. If this is not the case, the ES unchokes the peer. This is to give room to new peers to download. Otherwise, the ES checks if it has downloaded from the peer while it was a LE if this is the case, the ES unchokes the peer. If all fails, the ES selects a new neighbor randomly and leave the earlier selected neighbor as choked. This is done not to waste this cycle of 30 seconds interval without unchoking a peer.

Algorithm 7 Experienced Seeder Thirty-second Game

Frequency: Every 30 seconds

Input: List of all choked peers in the peer set

Output: Unchoke one peer and choke the rest

```

    //Candidate selection//
1: CandidateSelected  $\leftarrow$  {}
2: for all peers in the peer set do
3:   if a peer is choked then
4:     CandidateSelected  $\leftarrow$  CandidateSelected + peer
5:   end if
6: end for
7: Randomly select one peer from CandidateSelected
    //Unchoke decision making //
8: while the selected peer is choked do
9:   if my upload to the selected peer is zero then
10:    Unchoke peer
11:   else if my download from the selected peer is not zero then
12:    Unchoke peer
13:   else
14:    Select another peer from CandidateSelected
15:   end if
16: end while

```

Leecher Game Implementation

Leechers are peers that still need one or more pieces in order to download the complete file and they use tit-for-tat strategy to exchange pieces.

During the ten-second game, a LE selects three of its interested peers in its peer set as candidates. Then, it checks if it has interacted with each of the selected candidates. In the unchoke decision making process, the first thing that the LE checks for is if the selected peer snubbed the LE. If that is the case, then the LE chokes the selected peer. If the LE does not have any interaction with the selected peer, then the LE unchokes the peer. This is to enable new peers to bootstrap. On the other hand, if the LE has downloaded from the peer, then the LE unchokes the peer because the LE's strategy is tit-for-tat. Otherwise, the LE has not downloaded from the peer before and is not meeting the peer for the first time, then the LE chokes the peer. The LE's ten-second game is as shown in Algorithm 8.

As shown in Algorithm 9, the thirty-second game played by LE is similar to the ten-second game with the same strategy, the same set of actions and payoff function. However, the game is played with a different set of players. During the thirty-second game, a LE selects one of its choked neighbors randomly. The LE then checks if it has not uploaded to the selected peer before. If this is the case, the LE

unchokes the peer assuming the peer is new. This initial unchoke is to bootstrap new peers as well as the first action in the LE tit-for-tat strategy. On the other hand, if this is not a new peer to the LE, it checks if it has downloaded from the peer, if this is the case, the LE unchokes the peer. Otherwise, the LE chokes the peer and select another choked peer for a new round of game. The LE selects another peer so as not to waste this 30 seconds time period with out unchoking a peer.

Algorithm 8 Leecher Ten-second Game

Frequency: Every 10 seconds

Input: List of all peers in the peer set

Output: Unchoke zero to three peers and choke the rest

```
//Candidate selection//
1: CandidateSelected  $\leftarrow$  {}
2: for all peers in the peer set do
3:   if a peer is interested then
4:     CandidateSelected  $\leftarrow$  CandidateSelected + peer
5:   end if
6: end for

// Candidate sorting criteria//
7: Sort all peers in CandidateSelected based on their upload rate to me in the last 20
seconds

//Best candidate selection//
8: Select up to three top uploaders from CandidateSelected
9: if the number of interested peers is less than  $< 3$  then
10:   Complement the remaining by randomly selecting from the peer set
11: end if

// Unchoke decision making//
12: for every peer among the three best candidates selected do
13:   if the peer snubbed me then
14:     Choke peer
15:   else if my upload to the selected peer is zero then
16:     Unchoke peer
17:   else if my download from the selected peer is not zero then
18:     Unchoke peer
```

Algorithm 9 Leecher Thirty-second Game

Frequency: Every 30 seconds

Input: List of all choked peers in the peer set

Output: Unchoke one peer and choke the rest

```
//Candidate selection//
1: CandidateSelected  $\leftarrow$  {}
2: for all peers in the peer set do
3:   if a peer is choked then
4:     CandidateSelected  $\leftarrow$  CandidateSelected + peer
5:   end if
6: end for
7: Randomly select one peer from CandidateSelected
//Unchoke decision making //
8: while the selected peer is choked do
9:   if my upload to the selected peer is zero then
10:    Unchoke peer
11:   else if my download from the selected peer is not zero then
12:    Unchoke peer
13:   else
14:    Select another peer from CandidateSelected
15:   end if
16: end while
```

CHAPTER 6

PERFORMANCE EVALUATION

6.1 Overview

We conducted a series of performance evaluation experiments to assess our proposed game model of the BitTorrent choking algorithm. We started the performance evaluation process by examining the BitTorrent choking algorithm to show the vulnerability of the BitTorrent choking algorithm. This is discussed in subsection 6.6.1, we further examined the causes of such vulnerability subsection 6.6.2. Section 6.7 evaluates our proposed game model through a series of experiments to show its effectiveness as compared with the BitTorrent choking algorithm.

6.2 Performance Metrics

The objectives of a free riding mitigation mechanism are to ensure cooperation among participating peers and also ensure fairness amongst all peers in the system [39, 65]. Therefore, we focused our performance evaluation experiments on the following:

To show the vulnerability of the BitTorrent choking algorithm.

To understand the root causes of the BitTorrent choking algorithm exploitation.

To evaluate the effectiveness of our proposed game model in inhibiting free riding in BitTorrent.

With these objectives in mind, we conducted our evaluation experiments using the following metrics.

Conversion Rate: We define the conversion rate at time t of peer type x , $CR_x(t)$, as the number of peer type x converted to seeders, N_x^s over the number of peer type x , N_x . The conversion rate is computed as shown in equation 6.2.1.

$$CR_x(t) = \frac{N_x^s}{N_x} \times 100 \quad (6.2.1)$$

Number of Unchoke Messages: This is a measure of the actual number of unchoke messages sent by peer type x to peer type y and is expressed as UN_x^y .

Download Time: This measures the download time for a peer type and is calculated as follows:

$$DT_x = \frac{\sum_{i=1}^{i=N_x} DT_i}{N_x} \quad (6.2.2)$$

Where DT_x is the download time for peer type x and N_x is the total number of type x peers.

6.3 The Simulation Model

There is a need for a robust simulator to simulate a P2P environment, analyze and modify the BitTorrent protocol. We conducted our experiments using PeerSim [146], one of the most popular P2P simulators [147, 148, 149, 150] and BitPeer [151], a BitTorrent module for PeerSim. In addition to being open source, PeerSim in conjunction with BitPeer have been used recently in several published articles such as [147, 148, 149, 150, 69, 152, 153, 154].

6.3.1 PeerSim

PeerSim is a simulation engine for P2P systems. It is designed to be highly robust. This is achieved through the use of modular programming approach that offers pluggable components for researchers. It provides easier interfaces for design, analysis, protocol modification and

algorithms in P2P systems. PeerSim architecture comprises of Nodes, Control, and Protocols. A node is a container for protocol. It defines the behavior of every peer in the network with the use of protocol stack. The protocol stack manages the state and actions of each node. For example, one can simulate an overlay network or a distributed algorithm. We utilize the protocol simulation feature of PeerSim and simulated the BitTorrent protocol using BitPeer. PeerSim offers interfaces called Controls used to perform global initialization, observation and performance analysis. An example of these controls is the Observer used to collect statistics for performance analysis. Controls can also be used to modify the state of other entities such as adding and removing a node. The exogenous and design parameters for each experiment are specified using a configuration file. The configuration file is an ASCII text file that made up of three main parts, namely, general setup, protocol definition and control definition. A sample configuration file is shown in figure 6.1.

6.3.2 BitPeer

BitPeer [151] is an event driven BitTorrent protocol that runs on top of PeerSim. We will use the entries in the sample configuration file as shown in figure 6.1 to explain each of the BitPeer components and associated parameters. Note that lines begin with `#` are used as com-

```

1  #Config file for BitTorrent extension
2  random.seed 1234567890
3  simulation.endtime 10^9
4  simulation.logtime 10^3
5
6  simulation.experiments 1
7
8  network.size 101
9  network.node peersim.core.GeneralNode
10
11 protocol.urt UniformRandomTransport
12 protocol.urt.mindelay 10
13 protocol.urt.maxdelay 400
14 protocol.bittorrent peersim.bittorrent.BitTorrent
15 protocol.bittorrent.file_size 100
16 protocol.bittorrent.max_swarm_size 80
17 protocol.bittorrent.peerset_size 50
18 protocol.bittorrent.duplicated_requests 1
19 protocol.bittorrent.transport urt
20 protocol.bittorrent.max_growth 20
21
22 init.net peersim.bittorrent.NetworkInitializer
23 init.net.protocol bittorrent
24 init.net.transport urt
25 init.net.newer_distr 80
26 init.net.seeder_distr 15
27
28 control.observer peersim.bittorrent.BTObserver
29 control.observer.protocol bittorrent
30 control.observer.step 10000
31
32 control.dynamics peersim.bittorrent.NetworkDynamics
33 control.dynamics.protocol bittorrent
34 control.dynamics.newer_distr 60
35 control.dynamics.minsize 10
36 control.dynamics.tracker_can_die 1
37 control.dynamics.step 100000
38 control.dynamics.transport urt
39 control.dynamics.add 0
40 control.dynamics.remove 0

```

Figure 6.1: A PeerSim sample configuration file.

ment and ignored by the simulator. The entries in the configuration file are in the following format

```
<init|protocol|observer|dynamics>[parameter      name]<parameter value>
```

Lines 2 - 9 provide the general simulation and network setup. The network size is the total number of nodes in the network including the tracker. The simulation time and the number of experiments are set in line 3 and 6 respectively. Line 9 specifies that the protocol uses the general node in PeerSim. Second, the protocol definitions are specified in lines 11 - 20. These include some BitTorrent specific parameters. The protocol uses the uniform random transport delay in PeerSim with minimum and maximum delay of 10 and 400 ms, respectively. The protocol name is bittorrent. The bittorrent specific parameters included are file size, maximum swarm size, peer set size, duplicated request and maximum growth.

file size: This is the total size of the file to be shared by peers in the BitTorrent environment.

maximum swarm size: This is the maximum number of collaborating peers.

peer set size: This is the number of neighbors sent by the tracker to every node that joins the BitTorrent environment and indicates the number of neighbors a peer has access to.

duplicated request: This specifies the number of duplicate request messages a peer can send during unchoke. For instance, if the value is 3, it indicates that when a peer is unchoked, it can send requests to 3 different peers for the same piece.

maximum growth: This specifies the limit of allowable expansion to the network in a dynamic network. For example, if the network size is 100 and max growth is 50, then this shows that the network can be expanded to accommodate 150 nodes.

Lines 22 - 26 specify the network initialization parameters. All initialization lines begin with the key word *init*. The bittorrent network initializer, where the tracker and all the peers are initialized is specified in line 22. The newer and seeder distributions specify the initialization of peers with zero and complete pieces respectively at the start of the simulation. Finally, lines 28 - 40 define the controls which are either used to observe or modify the objects in the network. There are two types of controls specified, observer and dynamics. The bittorrent protocol observer is specified in line 28. The protocol to observe and frequency of observation are specified in line 29 and 30, respectively. The dynamic control starts from line 32 to 40.

6.4 Simulating Free riding in BitTorrent

Free riders have been defined as non contributing peers [39, 65, 67, 120]. A free riding behavior can be exhibited by peers in a P2P system in different ways. For example, in [65], a peer declares its reputation to others who decide whether the peer is a free rider. This approach assumes that peers are honest. In our work, we simulated FRs as follow: (a) FRs do not announce to other peers in the peer set any piece they have. This means that, free riders do not send *have* messages to others in their peer set, (b) At the time of handshake with others, FRs do not announce to others, through their bitfield, the initial pieces they have. Consequently, the *bitfield* messages sent by FRs contain only zeros.

6.5 Simulation Setup

We extended the BitTorrent protocol [151] by changing the choking algorithm as described in chapter 5, section 5.5. The design parameters used in the simulation are listed in Table 6.1. The term randomly generated over a range[a,b] means that the number is generated using a discrete (integer-valued) uniform distribution over a, a+1, ..., b inclusive. This is written as $U[a,b]$. Whereas the term randomly generated over [x,y,z] means that the number is generated using a discrete (integer-valued) uniform distribution over x, y and z inclusive.

Table 6.1: Simulation design parameters

Design parameter Definition	Design parameter value
How many times the simulation is repeated for each point in the graphs	5
Number of trackers	1
Number of peers	100
Number of seeders	[5%,10%]
Number of free riders	[30%, 70%]
Bandwidth allocated to peers	U[640Kbps, 1Mbps, 2Mbps, 4Mbps]
File size	100MB
Number of pieces	390
Peer set size	50
Piece size	256KB
Number of pieces allocated to leechers and free riders	U[10% - 90%]

The design parameters repetition, number of tracker and number of peers are set deterministically at 5, 1, and 100, respectively. The number of seeders is set deterministically at [5%,10%] of the number of peers. The objective of varying the number of seeders is to examine the effect of seeders on the performance of the model. The number of free riders is set deterministically at [30%,70%] of the total number of peers. The objective these two extreme values is to assess the agility of the model as the number of free riders increase. Allocated bandwidth to peers is randomly generated over [640Kbps, 1Mbps, 2Mbps, 4Mbps]. File size, number of pieces, peer set size and piece size are set deterministically at 100MB, 390, 50 and 256KB, respectively. Allocated number of pieces to LEs and FRs is randomly generated over the range [10%, 90%] of the number of pieces.

6.6 BitTorrent Vulnerability

6.6.1 BitTorrent Exploitation

The purpose of this section is to show how FRs exploit BitTorrent. We first, run experiments to show that indeed FRs exploit BitTorrent. Then, we examine the root causes of such exploit.

Figure 6.2 shows the conversion rate of LEs and FRs using 5% SEs and 30% FRs. The figure clearly shows that FRs are converting to SEs at a faster rate than LEs. This is unfair since these FRs do not contribute to the BitTorrent community and they are only 30 compared to 65% LEs.

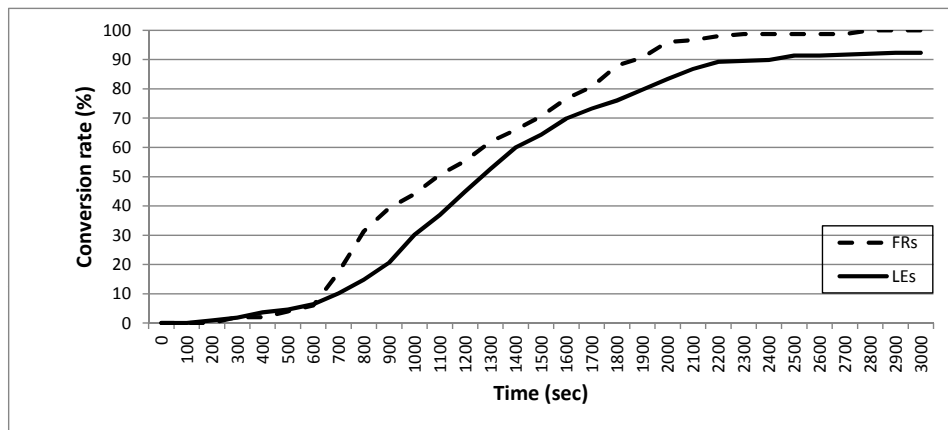


Figure 6.2: Using 5% SEs and 30% FRs: Conversion rate of FRs and LEs.

Figure 6.3 shows the conversion rate of LEs and FRs using 5% SEs and 70% FRs. FRs are converting to SEs also at a faster rate than LEs converting to SEs since FRs are the majority 70% compared to 25% LEs. This is because SEs initially will most likely select FRs since they

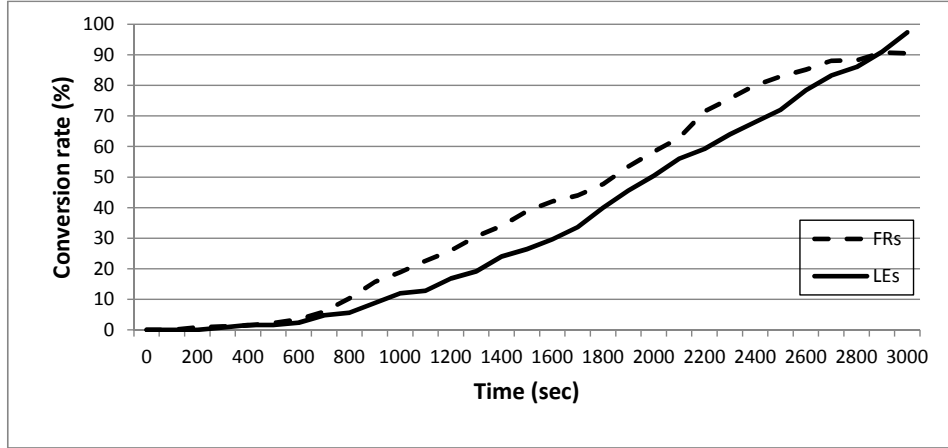


Figure 6.3: Using 5% SEs and 70% FRs: Conversion rate of FRs and LEs.

are the majority. Once FRs are selected by SEs, they will be always selected since their download rate from SEs will be among the top candidates. Therefore, SEs will be selecting FRs and this delays LEs further. Although LEs can exchange pieces among each other, they are only 25% of the community and do not have all the pieces.

We observe that the conversion rate of FRs are consistently higher than that of LEs even with 10% SEs as shown in Figures 6.4 and 6.5. The conversion rate for FRs when FRs are 30% is faster than when FRs are 70% because resource providers are 70% when FRs are 30% and only 30% when FRs are 70%.

Furthermore, we conducted experiments to measure the download time of FRs and LEs. By measuring the download time of FRs and LEs, we can compare their download times. Even if the download time of FRs and LEs are the same, we can conclude that the system is not fair since again FRs do not contribute to others. Furthermore,

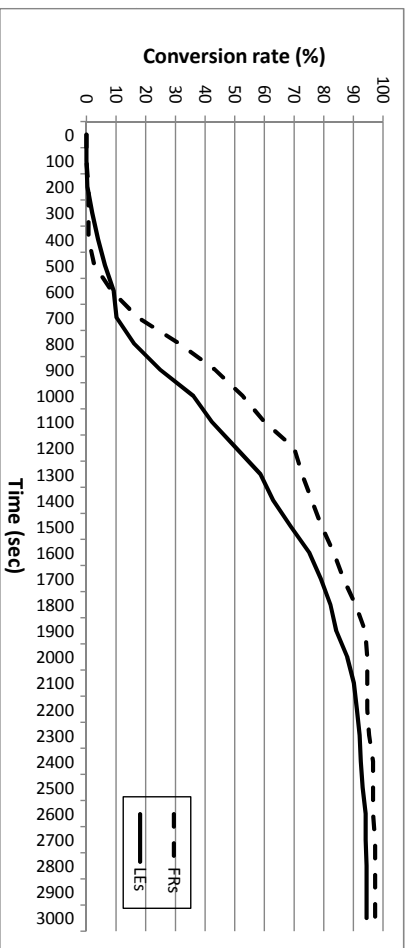


Figure 6.4: Using 10% SEs and 30% FRs: Conversion rate of FRs and LES.

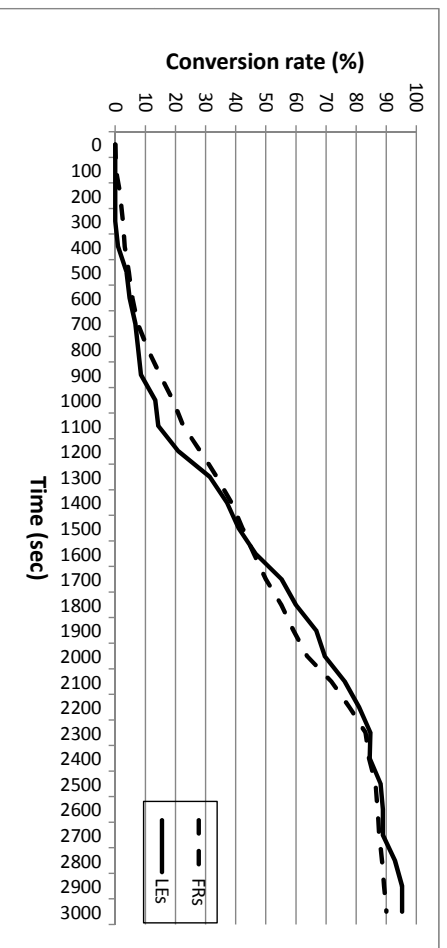


Figure 6.5: Using 10% SEs and 70% FRs: Conversion rate of FRs and LES.

download time can be used to measure system robustness as proposed in [119]. The authors in [119] define *Robustness* as the degree to which a system is able to deliver good service despite the presence of FRs. In a robust system, increasing the numbers of FRs should not be able to significantly affect the performance of LEs. Figure 6.6 shows that BitTorrent is not robust. That is, increasing the number of FRs delays the download time of LEs.

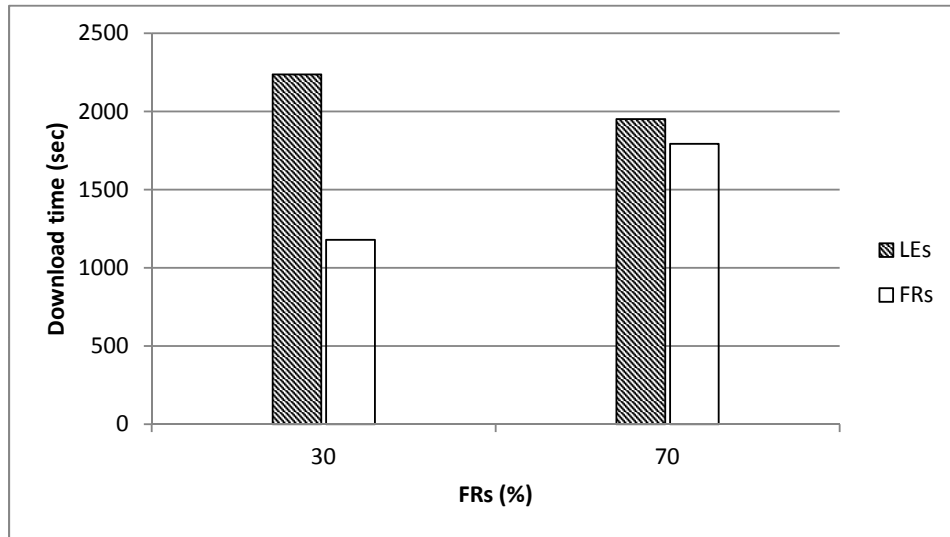


Figure 6.6: Using 5% SEs: Download time for FRs and LEs.

In summary, we conclude that it is unfair for FRs that do not upload pieces to others to surpass LEs in their rate of conversion or in their download time. FRs have an advantage over LEs since FRs utilize all their bandwidth for download while LEs use their bandwidth for download and upload. If this problem is not addressed, selfish behavior will be encouraged which negates the principle of P2P systems.

6.6.2 Exploitation Root Causes

Figure 6.7 shows the number of unchoke messages sent by SEs to FRs and LEs. Though, we observe that SEs are unchoking LEs at a higher rate than FRs. SEs sent almost 24000 unchoke messages to LEs and about 5000 to FRs. But there are 65% LEs and 30% FRs. This shows that FRs exploit the system.

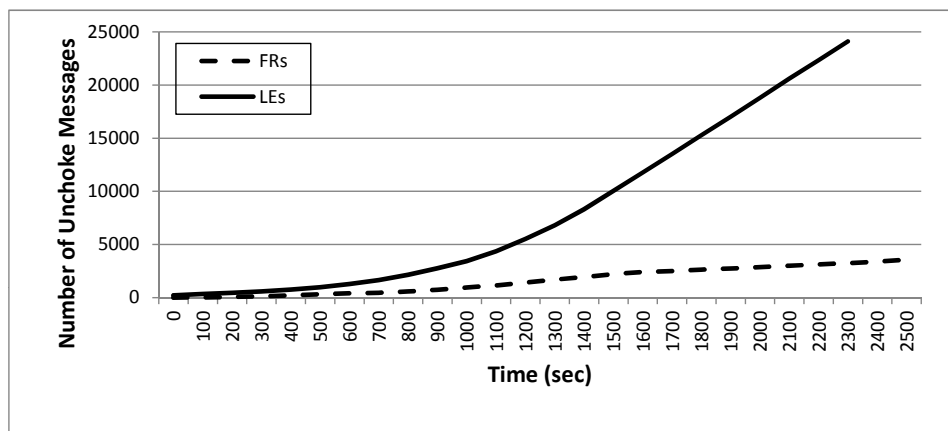


Figure 6.7: Using 5% seeders and 30% FRs: Unchoke messages sent by seeders.

As shown in figure 6.8, LEs sent about 19000 unchoke messages to other LEs. However, FRs received about 4000 unchokes from LEs. Although, LEs use tit-for-tat in the existing BitTorrent, FRs are able to download from LEs.

Figure 6.9 shows the number of unchoke messages sent by SEs to FRs and LEs. SEs are unchoking FRs at a higher rate than LEs. SEs sent almost 7500 unchoke messages to FRs and just above 3000 to LEs since there are 70% FRs and 25% LEs. Since FRs are the majority, there is a significant reduction in the total number of unchoke messages due to

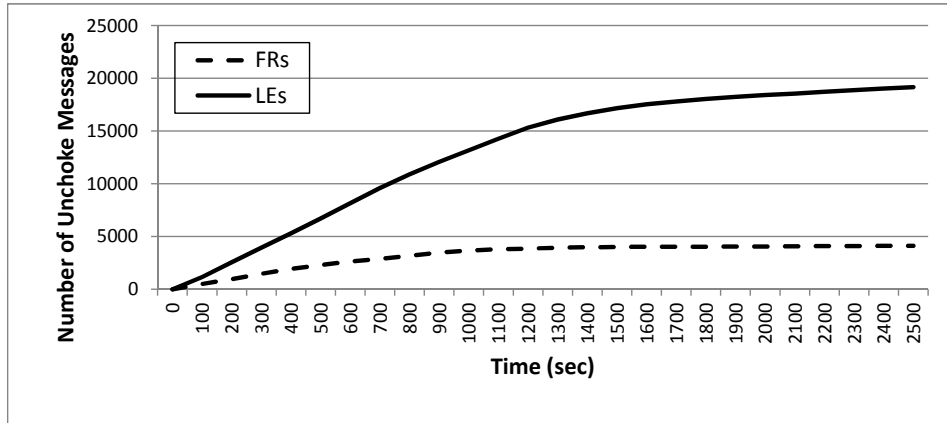


Figure 6.8: Using 5% seeders and 30% FRs: Unchoke messages sent by LEs.

the fact that FRs do not send unchoke messages.

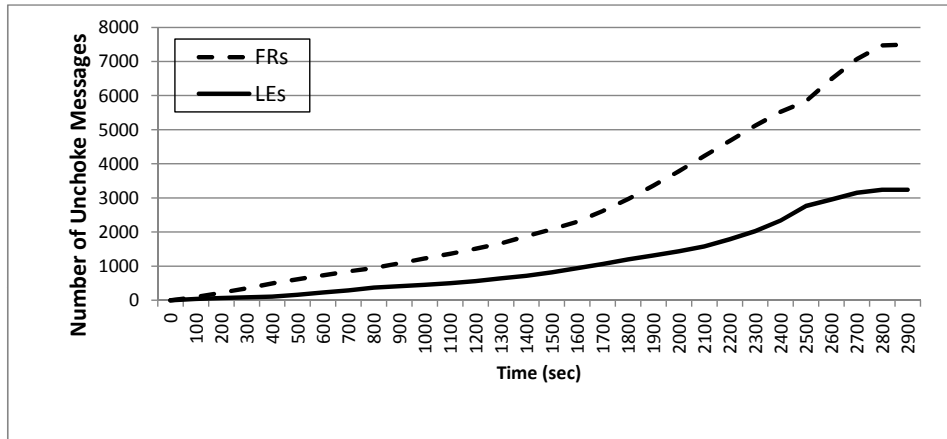


Figure 6.9: Using 5% seeders and 70% FRs: Unchoke messages sent by seeders.

As shown in figure 6.10, LEs are sending unchoke messages to FRs. We observe that LEs sent about 7000 unchoke messages to FRs. Though FRs are the majority with 70% while the 25% LEs received 6000 unchoke messages. LEs are expected to exchange pieces through reciprocation but FRs are able to exploit LEs.

In summary, we observe that FRs receive unchoke messages from both LEs and FRs. These results reveal the exploitation points for FRs. FRs

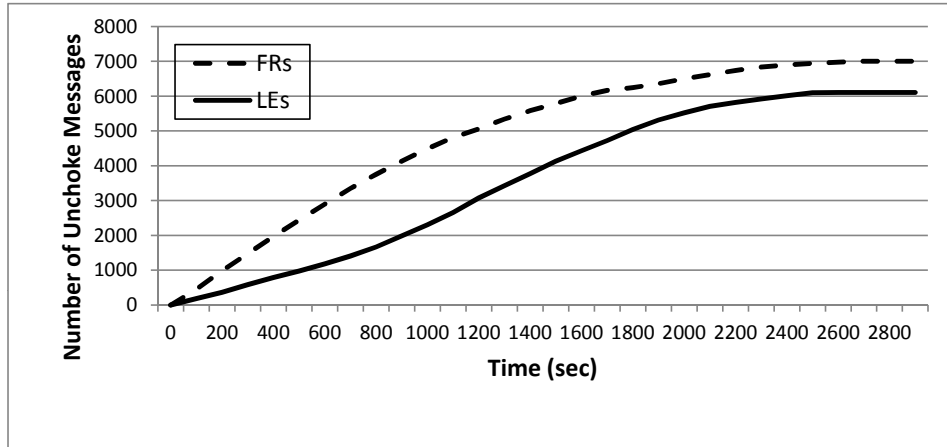


Figure 6.10: Using 5% seeders and 70% FRs: Unchoke messages sent by LEs.

that do not contribute resources to others are still able to download from both LEs and SEs.

6.7 Game Model Effectiveness

Here, we show that our proposed model based on game theory is effective in mitigating FRs in terms of fairness, robustness and agility. In this section, we will use the following notations. FRs before (FRs-B), FRs after (FRs-A), LEs before (LEs-B) and LEs after (LEs-A). Here, the word *before* means before applying our proposed game model (i.e., using the BitTorrent module, BitPeer, with FRs injected but no modifications to the choking algorithm). Similarly, the word *after* means after applying our game model modifications to BitPeer.

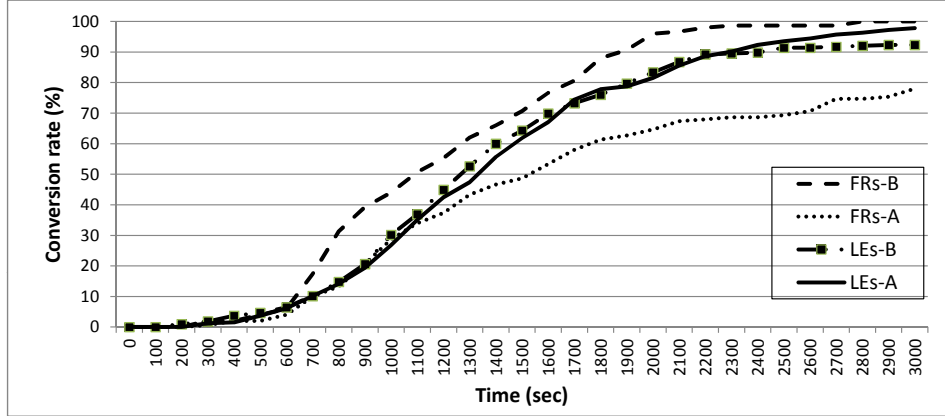


Figure 6.11: Using 5% ISs and 30% FRs: Conversion rate of FRs and LEs.

6.7.1 Game Model Fairness

Figure 6.11 shows the conversion rate of free riders before (FRs-B) and free riders after (FRs-A) using our game model. At time 1800 seconds, almost 90% of FRs converted to seeders but after applying our game model, only 60% of them converted to seeders. As such 30% of them were delayed. This is done to FRs without much negative effect on LEs.

Figure 6.12 shows the conversion rate of free riders before (FRs-B) and free riders after (FRs-A) using our game model. At time 1800 seconds, almost 50% of FRs converted to seeders but after applying our game model, only 25% of them converted to seeders. As such 25% of them were delayed. This is done to FRs much less negative effect on LEs. It should be noted that the conversion rate (in general) is much lower than figure 6.11 since providers are 40% less than figure 6.11.

Figure 6.13 has the same pattern as of figure 6.11. FRs are seriously

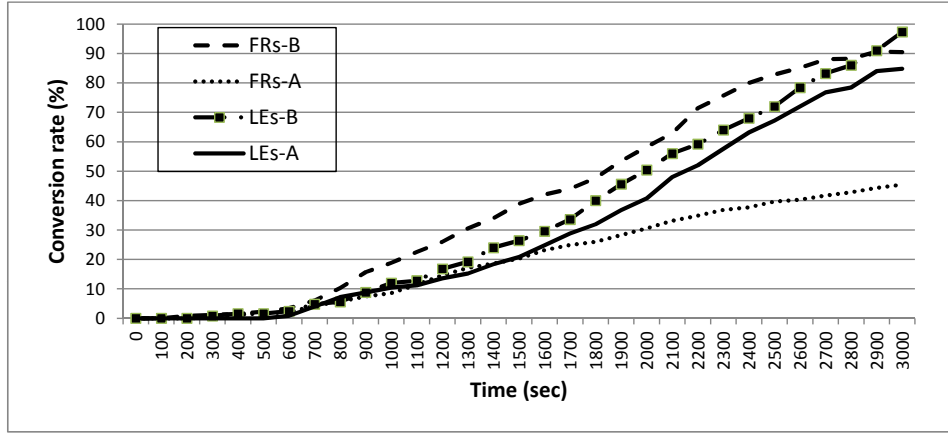


Figure 6.12: Using 5% ISs and 70% FRs: Conversion rate of FRs and LEs.

delayed while LEs are not much impacted even with 10% ISs. The same can be said about figure 6.14 with 70% FRs.

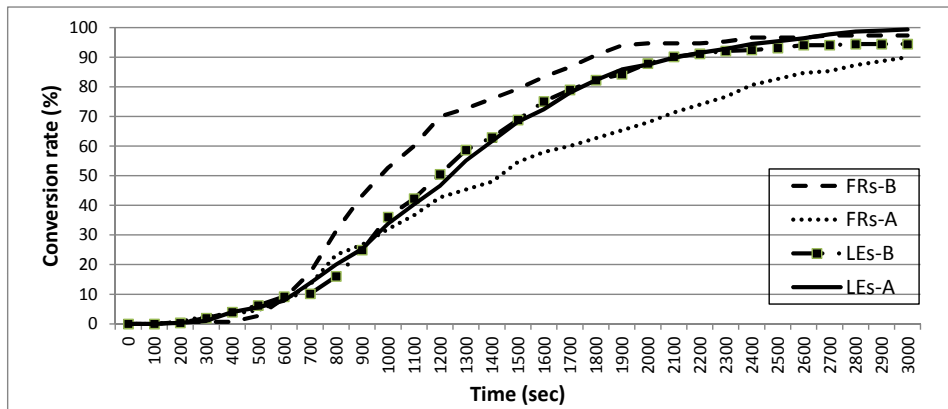


Figure 6.13: Using 10% ISs and 30% FRs: Conversion rate of FRs and LEs.

In summary, our proposed game model shows its effectiveness from the fairness point of view in delaying the conversion of FRs without much impact on LEs for two extreme cases when the BitTorrent community has 30% and 70% FRs.

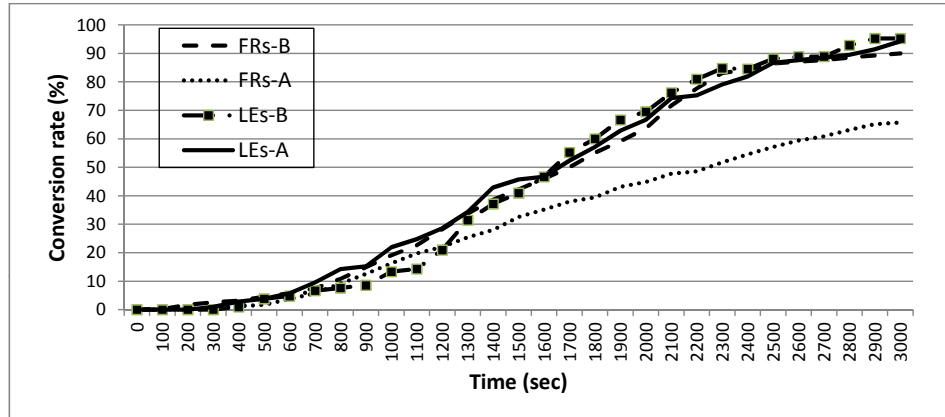


Figure 6.14: Using 10% ISs and 70% FRs: Conversion rate of FRs and LEs.

6.7.2 Game Model Robustness

To show the robustness of our proposed game model, we evaluated download time with 30% and 70% FRs and compared the results with what was obtained in subsection 6.6.1. From figure 6.15, it can be drawn that FRs are delayed in downloading the file when the game model is applied even with 70% FRs. We also compared the results of our proposed solution with the results obtained in figure 6.16 to examine the impact of our solution on LEs. As shown in the figure, LEs are not affected when FRs are 30% of the BitTorrent community but rather their download time improved significantly. On the other hand, when FRs are 70%, the download time is increased, although by not much, for LEs. This is due to the fact that 70% of the community are only consumers. Therefore, LEs have only 30% of the community and yet FRs are competing with them for obtaining service from 30% providers.

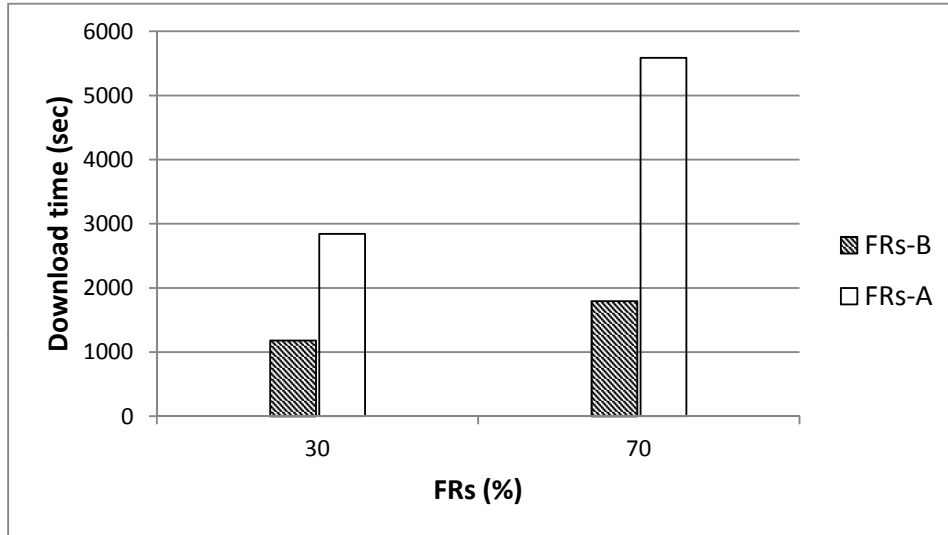


Figure 6.15: Using 5% ISs: Download time for FRs.

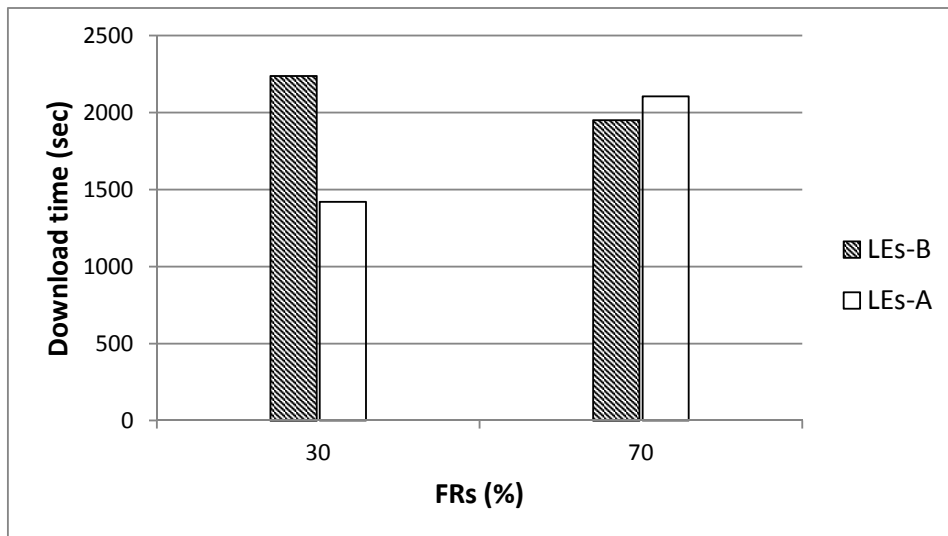


Figure 6.16: Using 5% ISs: Download time for LEs.

6.7.3 Game Model Agility

To understand the agility of our proposed game model, we analyzed the number of unchoke messages sent to FRs and LEs by ISs, ESs and LEs. Let us examine this when there are 5% ISs with 30% FRs and also with 70% FRs.

Game Model Agility with 30% FRs

Figure 6.17 shows the number of unchoke messages sent by ISs to FRs and LEs. ISs are unchoking LEs at a higher rate than FRs. ISs sent almost 4000 unchoke messages to LEs and only about 500 to FRs since there are 65% LEs and only 30% FRs. This is also due to the fact that ISs are selecting their candidates based on minimizing the standard deviation in terms of number of blocks given to these candidates. Therefore, FRs are delayed by ISs since they are the minority.

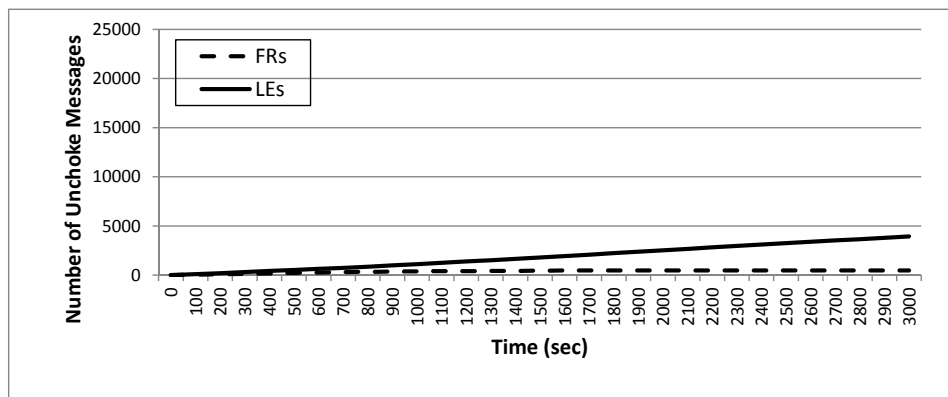


Figure 6.17: Using 5% ISs and 30% FRs: Unchoke messages sent by ISs.

As shown in figure 6.18, ESs are almost refusing to unchoke FRs since

ESs use their experience when they were LEs. For sure when an ES was a LE, it did not download anything from a FR. Therefore, FRs are paying the price for not sharing. On the other hand, ESs unchoke LEs at an increasing rate since as time increases, the number of ESs increases and the number of LEs decreases.

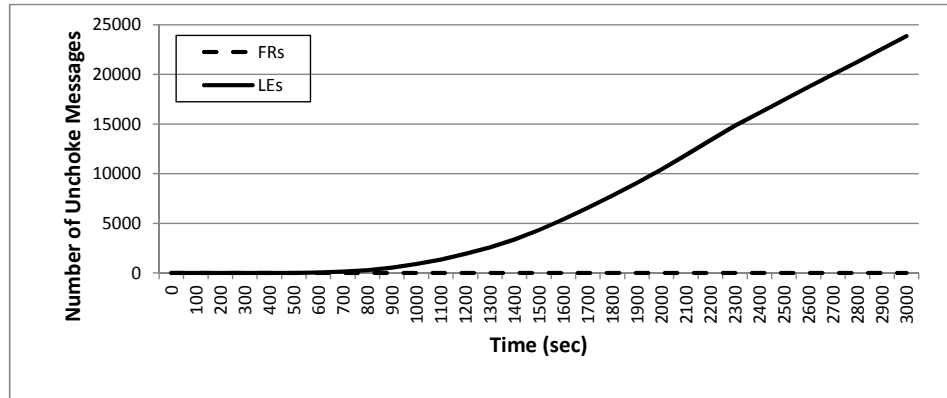


Figure 6.18: Using 5% ISs and 30% FRs: Unchoke messages sent by ESs.

Figure 6.19 shows that LEs are unchocking LEs at a sharp increasing rate. At time advances, the number of LEs decrease and this increase rate slows down. LEs are unchocking LEs because they apply a tit-for-tat strategy in order to cooperate with others.

Game Model Agility with 70% FRs

Figure 6.20 shows the number of unchoke messages sent by ISs to FRs and LEs. ISs sent almost 1800 unchoke messages to LEs and about 2800 to FRs. ISs sent more unchoke messages to FRs than LEs. This is expected, since there are 25% LEs and 70% FRs. ISs select their candidates based on minimizing the standard deviation in terms of

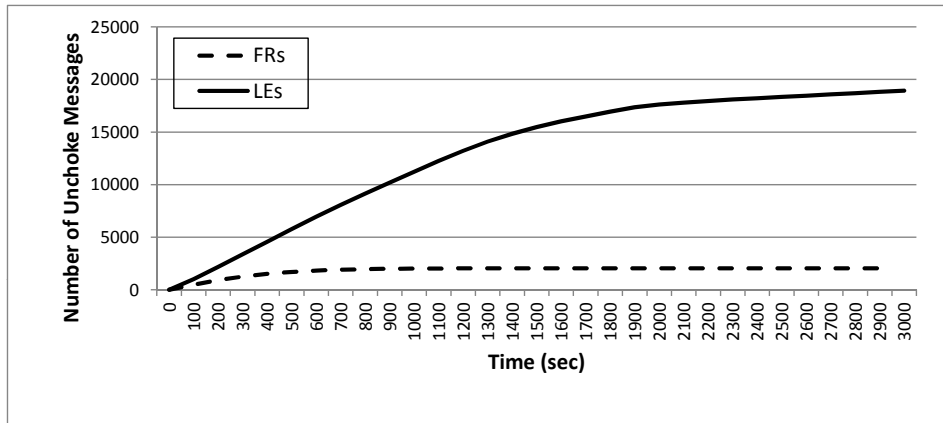


Figure 6.19: Using 5% ISs and 30% FRs: Unchoke messages sent by LEs.

number of blocks given to these candidates. Therefore, FRs are the majority it is expected that they get higher unchoke messages than LEs.

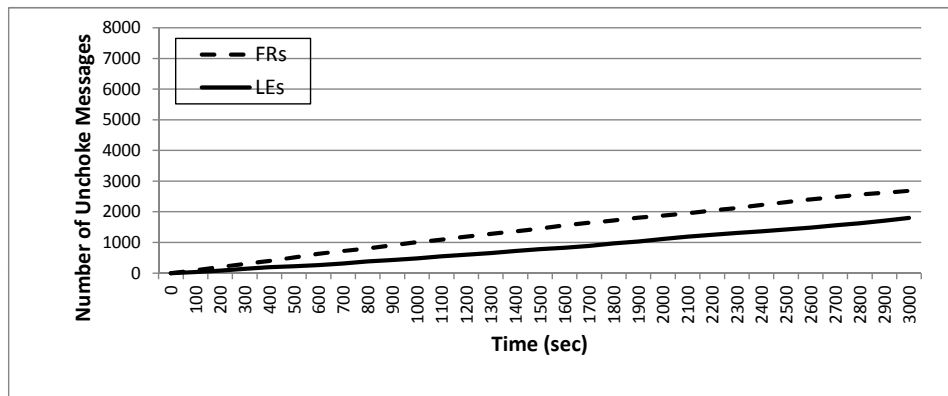


Figure 6.20: Using 5% ISs and 70% FRs: Unchoke messages sent by ISs.

As shown in figure 6.21, ESs almost refused to unchoke FRs. The number of unchoke messages sent by ESs to FRs is almost 0. As expected, ESs use their experience while they were LEs and refused to unchoke FRs. It is known that ES did not download from the FRs while the ES was a LE. Therefore, FRs are paying the price for not uploading

to the LE that turns ES. On the other hand, ESs unchoke LEs at an increasing rate since as time increases, the number of ESs increases and the number of LEs decreases.

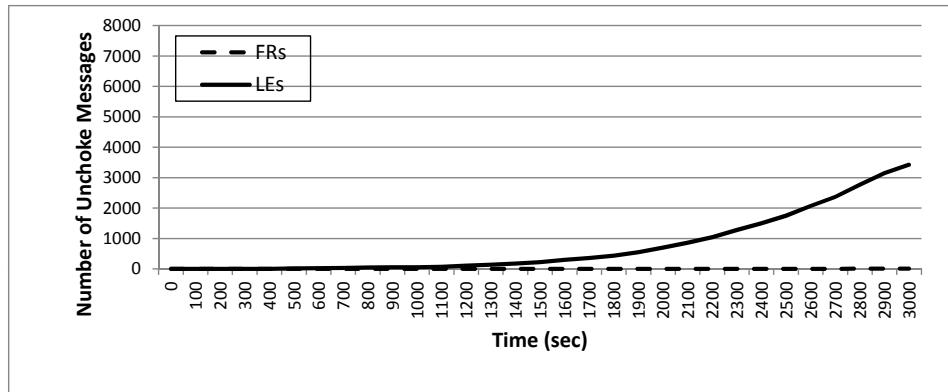


Figure 6.21: Using 5% ISs and 70% FRs: Unchoke messages sent by ESs.

Figure 6.22 shows that LEs are unchocking LEs at a sharp increasing rate as well as FR initially. At time advances, the number of unchoke messages remains constant at 3000 for FRs. FRs are the majority in this case, they got unchoked by LEs initially but with time the LEs stop unchocking them. On the other hand LEs are unchocking LEs because they apply a tit-for-tat strategy in order to cooperate with others.

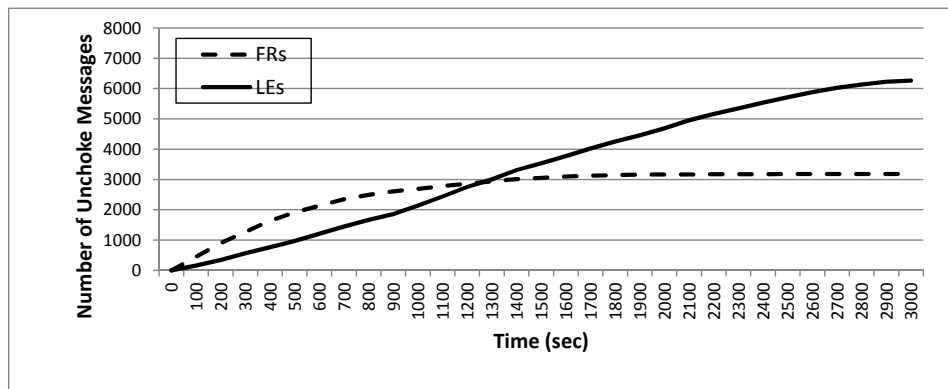


Figure 6.22: Using 5% ISs and 70% FRs: Unchoke messages sent by LEs.

In summary, we compare Figure 6.7 having the existing choking algorithm with SEs to figure 6.17 and figure 6.18 where we separated SEs into ISs and ESs.

Considering the number of unchoke messages sent by SEs to FRs. The unchoke messages in figure 6.7 is much higher than that in figure 6.17 and figure 6.18 combined. It is obvious from the results in figure 6.18 that ESs are punishing FRs for not sharing as the number of unchoke messages from ESs to FRs is almost 0. This punishment received by FRs from ESs are due to the experience of the ESs while they were LEs. This shows the effectiveness of our proposed game model for ES. However FRs do get some unchoke messages from ISs, since ISs do share their blocks with the objective of minimizing the standard deviation in terms of block distribution. Furthermore, the number of unchoke messages sent by SEs to LEs is lesser in 6.7 compared to the combined unchoke messages of ISs and ESs to LEs as shown in figures 6.17 and 6.18. As for LEs, the number of unchoke message received from both ISs and ESs are higher. This is due to the fact that our game model uses tit-for-tat for ES, since LEs are sharing they will get unchoke messages from ES.

Figure 6.8 and Figure 6.19 shows the number of unchoke messages sent by LEs before and after modifications to FRs. We observe from the results that LEs are punishing FRs by cutting down their unchokes

messages by half from about 4100 to 2000. This is because FRs do not upload resources to the LEs. Also, the number of unchoke messages from LEs to FRs seems to be high because there could be several unchoke before the complete pieces are downloaded. But if it happens that LE unchokes a FR initially, as time progresses then the LE applied its experience when it turns to ES and stop unchoking FRs.

On the other hand, the number of unchoke messages sent by LEs to LEs before and after as shown in Figures 6.8 and 6.19 are almost the same. This is an indication that our game model does not hamper the tit-for-tat strategy that is required of all peers.

Similarly using 5% IS and 70% FRs, we compared the number of unchoke messages sent by SEs to LEs and FRs. Figure 6.9 shows the number of unchoke messages sent to LEs and FRs before applying our game model.

We observe from Figures 6.20 and 6.21 are the number of unchoke messages sent by ISs and ESs respectively to FRs. Note that the unchoke messages sent by SEs to FRs in Figure 6.9 is higher due to the fact that after modifications, the SEs are divided into ISs and ESs. Also, we observe from the results that ESs are punishing FRs for not sharing as the number of unchoke messages from ESs to FRs are almost 0 shown in figure 6.21. However FRs do get some unchoke messages from ISs, since ISs block distribution is based on the objective of min-

imizing the standard deviation. FRs are the majority 70%, hence they got unchoked by ISs which is expected. on the other hand, the number of unchoke messages sent by SEs before to LEs in figure 6.9 increases from 3100 to about 5300. The increase results from additional messages from ESs to LEs which are not in the existing algorithm.

As shown in figure 6.10, LEs are sending unchoke messages to FRs. We observe that LEs sent about 7000 unchoke messages to FRs compared to around 3000 in Figure 6.22. Though FRs are the majority with 70% while the 25% are LEs. FRs unchoke messages were reduced by 50% after implementing our game model. Furthermore, we observe that majority of the unchoke messages sent by LEs to FRs were at the initial stage of the sharing interaction. As time increases, the FRs could not receive more unchoke from LEs, hence the value become constant as shown in Figure 6.22. On the other hand, the number of unchoke messages sent by LEs before and after remain approximately the same. This is as a result of tit-for-tat strategy to which LEs adhered to.

CHAPTER 7

CONCLUSIONS AND FUTURE DIRECTIONS

This chapter summarizes the conclusion drawn from the work carried out in this thesis in section 7.1. Section 7 summarizes the contributions of this thesis while section 7.3 suggests area of possible improvement and future directions.

7.1 Conclusions

In our study of free riding in distributed systems, we conclude that free riding does not occur in all distributed systems. Free riding arises in distributed systems in which the resources shared have public subscriptions. Based on our study, client-server system in which the resources are provided by the clients and P2P systems are the two distributed

systems where public subscription resources lead to free riding. But free riding is not a major issue in client-server due to the availability of servers that can easily be tailored towards the control of free riding in this system. On the other hand, free riding in P2P system is a concern and it is a challenging problem to solve due to lack of node management and node control policy.

In a resource rich P2P system with high number of altruist. For instance, a BitTorrent system with a high number of seeders, free riding of few peers should not be of major concern. On the other hand, in an environment with an average number of resource providers, there is a need for an effective free riding mitigation techniques.

On the use of game theory to study free riding in P2P systems. Game theory offers some advantages by providing a framework that other free riding mitigation approaches such as reciprocity and reputation can be incorporated to games theory approaches to provide a better free riding mitigation capability. Moreover, in using game theory to model free riding solutions, it better to model the game based on architecture specific as we did in BitTorrent. A generic game model for P2P systems might not capture some architecture specific behavior.

7.2 Summary of Thesis Contributions

Through out this work, we conducted a study on the problem of free riding in distributed systems in general. A further study is carried out specifically on free riding in BitTorrent system. The thesis contributions are summarized as follows:

Taxonomy of distributed systems : We studied in detail the characteristics of distributed systems. As a result of our study, we compared distributed systems and proposed a node-based to clearly classify distributed systems based on node ownership, node controlling policy, node management and node discovery mechanism. This analysis and comparison resulted into the first contribution of this thesis which is a taxonomy of distributed system to clearly differentiate them. We believe this taxonomy will improve our understanding of distributed system.

Game taxonomy : We present a detail background on games and discusses game basic and components. Based on this background work, we proposed a taxonomy of games. This taxonomy will serve as a reference knowledge not only to researchers in this field of study but also to researchers that may be interested in game theory in their respective discipline.

Choking algorithm game model : We present a detail insight into the mechanisms and algorithms in BitTorrent system and pro-

posed a novel free riding mitigation techniques for BitTorrent based on game theory. Our experiments show that the proposed model performs better than the existing choking and optimistic unchoking in BitTorrent.

Performance evaluation : In this phase, we carried out extensive simulation studies to understand the original BitTorrent protocol behavior. In our study, we conducted experiments to highlight BitTorrent’s exploitations by free riders as well as pointing out the vulnerability points that made free riding possible. Moreover, we extended the BitPeer module by simulating a free riding behavior in BitTorrent. Finally, we implemented our proposed game model and evaluated its performance.

7.3 Future directions

One of the future directions is investigating a tracker level game model. In this game model, the tracker, in addition to its peer discovery role is also involved in the management and coordination of the game and its information. Involving the tracker in this game is to cater for global objective or community welfare such as availability of resources and fairness in sharing. But, a careful study of this model poses lots of challenges to a BitTorrent system. Amongst such drawbacks are;

Increase in the degree of centralization, which may lead to a single

point of failure. The management of the game model ceases as soon as the tracker goes down.

Increase in the number of messages exchanged. This results from the extra game information that needs to be updated by the tracker.

Despite the aforementioned challenges that may face involving the tracker in the game, the idea is worth exploring further. Implementing a tracker based game model might lead to intelligent peer set assignment and better resource utilization.

Another future direction is to extend our proposed game model to work in a pure P2P environment. As it stands now, resource discovery is done in a centralized manner while resource dissemination is done in a P2P fashion. In a pure P2P environment, there is no centralization point hence resource discovery needs to be done in a distributed manner. Another issue is managing the status of peers. For example, when a peer becomes a seeder, where should this change of status be kept. That is managing the swarm as well as ensuring that the peer set is randomly chosen from the swarm.

Furthermore, identity management can be examined to study the effect of identity changes on our proposed model. A free rider can spawn multiple identities in order to download from experienced seeders and leechers because experienced seeders and leechers will unchoke new

peers. Therefore, a free rider can exploit this point by having multiple identities. Multiple identities can also help a free rider to contact the tracker multiple times and therefore increases its peer set beyond the standard peer set size. This allows the free rider to contact a large number of experienced seeders and leechers and hence the free rider will have access to more resource providers.

REFERENCES

- [1] M. Karakaya, “Counteracting Free Riding in Pure Peer-to-Peer Networks,” *PhD Thesis, Bilkent University Turkey*,, 2008.
- [2] M. Ripeanu and I. Foster, “Mapping the gnutella network: Properties of large-scale peer-to-peer systems and implications for system design,” *IEEE Internet Computing Special issue on Peer-to-Peer Networking*, 2002. [Online]. Available: <http://arxiv.org/abs/cs/0209028>
- [3] D. Hausheer and B. Stiller, “Economics of peer-to-peer systems.” *In DOCIS documents in computing and information science.*, 2003.
- [4] J. Tian, L. Yang, J. Li, and R. Tian, “A Peer-To-Peer incentive scheme for overcoming free riding,” *Journal of Electronics (China)*, vol. 27, no. 1, pp. 60–67, May 2010. [Online]. Available: <http://www.springerlink.com/index/10.1007/s11767-009-0018-2>
- [5] M. Karakaya, I. Korpeoglu, and E. Adar, “Free Riding in Peer-to-Peer Networks,” *IEEE Internet Computing*, 2009.
- [6] R. Krishnan and M. D. Smith, “The Virtual Commons : Why

- Free-Riding Can Be Tolerated in File Sharing Networks,” *In International Conference on Information Systems*, 2008.
- [7] M. D. Samuel and R. Balakrishnan, “Trading of grade based incentives to avoid free riding and starvation in p2p network.” in *ICDCIT*, 2013, pp. 347–360.
- [8] A. S. Tanenbaum and V. S. Maarten, *Distributed Systems: Principles and Paradigms*. Prentice Hall, second edition, 2007.
- [9] C. M. Dan, “Cloud computing: Theory and practice.” *Lecture Notes, Computer Science Division, Department of Electrical Engineering Computer Science, University of Central Florida, Orlando, FL 32816, USA*, 2012.
- [10] K. Lai, B. Huberman, and L. Fine, “Tycoon: A Distributed Market-based Resource Allocation Systems”, HP Labs Technical Report cs.DC/0404013, <http://arxiv.org/abs/cs.DC/0404013>, 2004.
- [11] E. Wolfgang, “Distributed system principles,” *Lecture Notes, Software Engineering Group, University College London*, 1997.
- [12] R. Buyya, *High Performance Cluster Computing: Systems and Architectures*. Prentice Hall PTR, NJ, 1999.
- [13] I. Foster, “What is the grid? a three point checklist,” June 2002. [Online]. Available: <http://www-fp.mcs.anl.gov/foster/Articles/WhatIsTheGrid.pdf>

- [14] I. Foster, C. Kesselman, and S. Tuecke, “The anatomy of the grid: Enabling scalable virtual organizations,” *Int. J. High Perform. Comput. Appl.*, vol. 15, no. 3, pp. 200–222, Aug. 2001. [Online]. Available: <http://dx.doi.org/10.1177/109434200101500302>
- [15] S. Androutsellis-theotokis and D. Spinellis, “A Survey of Peer-to-Peer Content Distribution Technologies,” *ACM Computing Survey*, vol. 36, no. 4, pp. 335–371, 2004.
- [16] A. Oram, “Peer-to-Peer: Harnessing the power of disruptive technologies,” *O’Reilly & Associates*, 2001.
- [17] D. P. Anderson, J. Cobb, E. Korpela, M. Lebofsky, and D. Werthimer, “Seti@home: an experiment in public-resource computing,” *Commun. ACM*, vol. 45, no. 11, pp. 56–61, Nov. 2002. [Online]. Available: <http://doi.acm.org/10.1145/581571.581573>
- [18] P. Reinhard and M. Bernd, “Einstein@home - gravitational waves for everybody,” *Einstein Online*, vol. 4, p. 1015, 2010. [Online]. Available: <http://www.einstein-online.info/spotlights/EaH>
- [19] abcathome, “abcathome.com,” 2012. [Online]. Available: <http://abcathome.com/>
- [20] S. Saroiu and P. Gummadi, “A measurement study of peer-to-peer file sharing systems,” in *Multimedia Computing and Networking*,

2002. [Online]. Available: <http://research.microsoft.com/en-us/um/people/ssaroiu/publications/mmcn/2002/mmcn.html>
- [21] S. Saroiu, K. P. Gummadi, R. J. Dunn, S. D. Gribble, and H. M. Levy, “An analysis of internet content delivery systems,” in *Proceedings of the 5th symposium on Operating systems design and implementation - OSDI '02*. New York, USA: ACM Press, 2002, p. 315. [Online]. Available: <http://portal.acm.org/citation.cfm?doid=1060289.1060319>
- [22] S. Saroiu, “Measurement and Analysis of Internet Content Delivery Systems,” *PhD Thesis, University of Washington*, 2004.
- [23] E. Adar and B. A. Huberman, “Free Riding on Gnutella,” *First Monday*, pp. 1–22, 2000.
- [24] D. Hughes, “Free Riding on Gnutella Revisited : The Bell Tolls ?” *IEEE Distributed Systems Online*, vol. 6, no. 6, pp. 1–18, 2005.
- [25] M.-V. Belmonte, M. Díaz, J.-L. Pérez-De-La-Cruz, and A. Reyna, “Coins: Coalitions and incentives for effective peer-to-peer downloads,” *J. Netw. Comput. Appl.*, vol. 36, no. 1, pp. 484–497, Jan. 2013. [Online]. Available: <http://dx.doi.org/10.1016/j.jnca.2012.04.010>
- [26] L. Ramaswamy, “Free Riding : A New Challenge to Peer-to-Peer File Sharing,” in *36th Hawaii International Conference on*

System Science, 2003.

- [27] J. Sweeny, “An experimental investigation of the free-rider problem,” *Social Science Research*, vol. 2, 1973.
- [28] G. Marwell and R. Ames, “Experiments in the provision of public goods: resources, interest, group size, and the free-rider problem,” *American Journal of Sociology*, vol. 84, 1979.
- [29] G. Hardin, “The Tragedy of the Commons,” *Science*, no. 162, pp. 1243–1248, 1968.
- [30] B. Theodore, B. Lawrence, and R. V. Hal, “On the private provision of public goods,” *Journal of Public Economics*, no. 29, pp. 25–49, 1986.
- [31] S. A. Baset and H. Schulzrinne, “An analysis of the Skype peer-to-peer Internet telephony protocol,” in *Proc. of the 25th IEEE International Conference on Computer Communications*, ser. INFOCOM '06, Barcelona, Spain, April 2006.
- [32] P. Pradeep, N. Kumar, R. S. Shekar, Reddy, and C. Krishna, “Preventive measures for malware in p2p networks.” *International Journal of Engineering Research and Applications (IJERA)*, vol. 2, no. 1, pp. 391–400, Feb. 2012.
- [33] M. Roussopoulos, M. Baker, and D. S. H. Rosenthal, “2 P2P or Not 2 P2P ?” in *IPTPS'04 Proceedings of the Third international conference on Peer-to-Peer Systems*, 2004, pp. 1–6.

- [34] K. Aberer and M. Hauswirth, “An Overview on Peer-to-Peer Information Systems,” in *18th International Conference on Data Engineering*, 2002, pp. 1–14.
- [35] X. Fan, M. Li, J. Ma, Y. Ren, H. Zhao, and Z. Su, “Behavior-based reputation management in p2p file-sharing networks,” *J. Comput. Syst. Sci.*, vol. 78, no. 6, pp. 1737–1750, Nov. 2012. [Online]. Available: <http://dx.doi.org/10.1016/j.jcss.2011.10.021>
- [36] J. Veijalainen, “Autonomy , Heterogeneity , Trust , Security , and Privacy in Mobile P2P Environments,” *International Journal of Security and Its Applications*, vol. 1, no. 1, pp. 57–72, 2007.
- [37] A. Rowstron and P. Druschel, “Pastry : Scalable , decentralized object location and routing for large-scale peer-to-peer systems,” in *In IFIP/ACM International Conference on Distributed System platforms*, 2001, pp. 329–350.
- [38] C. Gkantsidis, M. Mihail, and A. Saberi, “Random walks in peer-to-peer networks: algorithms and evaluation,” *Perform. Eval.*, vol. 63, no. 3, pp. 241–263, Mar. 2006. [Online]. Available: <http://dx.doi.org/10.1016/j.peva.2005.01.002>
- [39] I. R. Rafit, “Improving the BitTorrent Protocol Using Different Incentive Techniques,” *PhD Thesis, University of California, Los Angeles*, 2010.
- [40] FastTrack, “www.fasttrack.nu,” 2012. [Online]. Available:

<http://www.fasttrack.nu>

- [41] I. Stoica, R. Morris, D. Liben-nowell, D. R. Karger, M. F. Kaashoek, and F. Dabek, “Chord : A Scalable Peer-to-peer Lookup Protocol for Internet Applications,” *IEEE Transactions on Networking*, vol. 11, pp. 1–14, 2003.
- [42] K. Aberer, “P-Grid: A Self-Organizing Access Structure for P2P Information Systems,” in *Sixth International Conference on Cooperative Information Systems*, 2001.
- [43] F. G. Mármol and G. M. Pérez, “Security threats scenarios in trust and reputation models for distributed systems,” *Computers & Security*, vol. 28, no. 7, pp. 545–556, Oct. 2009. [Online]. Available: <http://linkinghub.elsevier.com/retrieve/pii/S0167404809000534>
- [44] D. Banerjee, S. Saha, and S. Sen, “Reciprocal resource sharing in P2P environments,” in *Proceedings of the fourth International Conference on Autonomous and Multiagent System*, 2005. [Online]. Available: <http://dl.acm.org/citation.cfm?id=1082603>
- [45] N. Daswani, H. Garcia-Molina, and B. Yang, “Open problems in data-sharing peer-to-peer systems,” in *Proceedings of the 9th International Conference on Database Theory*, ser. ICDT '03. London, UK, UK: Springer-Verlag, 2002, pp. 1–15. [Online]. Available: <http://dl.acm.org/citation.cfm?id=645505.656446>

- [46] A. Asvanund, K. Karen, Clay. Ramayya, and D. S. Micheal, “An Emperical Analysis of Network externalities in Peer-to-Peer Music Sharing networks,” *Information System Research*, vol. 15, no. 2, pp. 155–174, 2004.
- [47] T. Silverston and O. Fourmaux, “Towards an incentive mechanism for peer-to-peer multimedia live streaming systems,” in *8th International Conference on Peer-to-Peer Computing*,, 2008, pp. 6–9. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4627269
- [48] Y. J. Hall, P. Piemonte, and M. Weyant, “Joost : A Measurement Study By analyzing the Joost client application ,” *Technical Report Carnegie Mellon University*, 2007.
- [49] S. Agarwal, J. P. Singh, and S. Dube, “Analysis and Implementation of Gossip-Based P2P Streaming with Distributed Incentive Mechanisms for Peer Cooperation,” *Advances in Multimedia*, vol. 2007, pp. 1–12, 2007. [Online]. Available: <http://www.hindawi.com/journals/am/2007/084150/abs/>
- [50] PPlive, “www.pplive.com,” 2012. [Online]. Available: <http://www.PPlive.com>
- [51] SOPCast, “www.sopcast.com,” 2012. [Online]. Available: <http://www.sopcast.com>
- [52] PPStreamTV, “www.ppstream.com,” 2012. [Online]. Available:

<http://www.ppstream.com>

- [53] Joost, “www.joost.com,” 2012. [Online]. Available: <http://www.joost.com>
- [54] RoxbeamMediaNetwork, “www.roxbeam.com,” 2012. [Online]. Available: <http://www.roxbeam.com>
- [55] P. Golle, K. Leyton-brown, I. Mironov, and M. Lillibridge, “Incentives for Sharing in Peer-to-Peer Networks,” *Electronic Commerce*, 2001.
- [56] J. Babaioff, M. Chuang and M. Feldman, “Incentives in Peer-to-Peer Systems,” in *Algorithmic Game Theory*, N. Nisan, T. Roughgarden, and V. Tardos, Eds. Cambridge University Press, 2007, ch. 23.
- [57] Z. Yu and H. Y. Bai, Yanping, “Free riding inhibition mechanism based on user behavior in p2p file-sharing system,” *China Communications*, vol. 9, no. 12, p. 36, 2012.
- [58] K. Tsai, K. Ho, and H. Weimin, “Tolerant Tit-for-Tat and Fibonacci Transmission Scheme,” in *Recent Advances in Computer Science and Information Engineering*, Q. Zhihong, C. Lei, and H. Y. Weilian Su, Tingkai Wang, Eds. Springer-Verlag Berlin Heidelberg 2012, 2012, vol. 4.
- [59] Y. Li, D. Gruenbacher, and C. M. Scoglio, “Reward only is not enough: Evaluating and improving the fairness policy of the p2p

- file sharing network emule/edonkey,” *Peer-to-Peer Networking and Applications*, pp. 40–57, 2012.
- [60] A. Satsiou and L. Tassiulas, “Trust-based exchange of services to motivate cooperation in P2P networks,” *Peer-to-Peer Networking and Applications*, vol. 4, no. 2, pp. 122–145, Apr. 2010. [Online]. Available: <http://www.springerlink.com/index/10.1007/s12083-010-0069-z>
- [61] B. Q. Zhao, J. C. S. Lui, and D. Chiu, “A mathematical framework for analyzing adaptive incentive protocols in p2p networks,” *IEEE/ACM Trans. Netw.*, vol. 20, no. 2, pp. 367–380, Apr. 2012. [Online]. Available: <http://dx.doi.org/10.1109/TNET.2011.2161770>
- [62] K. Zhang and N. Antonopoulos, “A novel bartering exchange ring based incentive mechanism for peer-to-peer systems,” *Future Gener. Comput. Syst.*, vol. 29, no. 1, pp. 361–369, Jan. 2013. [Online]. Available: <http://dx.doi.org/10.1016/j.future.2011.06.005>
- [63] Y. M. Tseng and F. G. Chen, “A free-rider aware reputation system for peer-to-peer file-sharing networks,” *Expert Systems with Applications*, vol. 38, no. 3, pp. 2432–2440, Mar. 2011. [Online]. Available: <http://linkinghub.elsevier.com/retrieve/pii/S0957417410008158>
- [64] Z. Yu and v. S. Mihaela, “Peer-to-peer multimedia sharing based

- on social norms,” *Signal Processing: Image Communication*, vol. 27, pp. 383–400, 2012.
- [65] J. S. Hua, D. C. Huang, S M Yen, and C. W. Chena, “A dynamic game theory approach to solve the free riding problem in the peer-to-peer networks,” *Journal of Simulation*, vol. 6, pp. 43–55, 2012.
- [66] R. Gupta and A. K. Somani, “Game theory as a tool to strategize as well as predict nodes behavior in peer-to-peer networks,” in *Proceedings of the 11th International Conference on Parallel and Distributed Systems - Volume 01*, ser. ICPADS '05. Washington, DC, USA: IEEE Computer Society, 2005, pp. 244–249. [Online]. Available: <http://dx.doi.org/10.1109/ICPADS.2005.157>
- [67] S. Manoharan and T. Ge, “A demerit point strategy to reduce free-riding in bittorrent.” *Computer Communications*, pp. 875–880, 2013.
- [68] V. Atlidakis, M. Roussopoulos, and A. Delis, “Enhanced-bit: Unleashing the potential of the unchoking policy in the bittorrent protocol,” *J. Parallel Distrib. Comput.*, vol. 74, no. 1, pp. 1959–1970, Jan. 2014. [Online]. Available: <http://dx.doi.org/10.1016/j.jpdc.2013.08.008>
- [69] A. Adam, B. Zoltan, Acs Attila, and T. Lukovszki, “Towards a faster bittorrent,” *Annales Univ. Sci. Budapest., Sect. Comp.*,

- vol. 37, pp. 65 – 80, 2012.
- [70] T. Vinko, F. Santos, N. Andrade, and M. Capota, “On swarm-level resource allocation in bittorrent communities,” *Optimization Letters*, vol. 7, no. 5, pp. 923–932, 2013.
- [71] N. Andrade, E. Santos-Neto, F. V. Brasileiro, and M. Ripeanu, “Resource demand and supply in bittorrent content-sharing communities,” *Computer Networks*, vol. 53, no. 4, pp. 515–527, 2009.
- [72] N. Liogkas, R. Nelson, and E. Kohler, “Exploiting bittorrent for fun (but not profit),” in *Proc. of IPTPS*, 2006. [Online]. Available: <http://www.iptps.org/papers-2006/Liogkas-BitTorrent06.pdf>
- [73] R. Krishnan and M. Smith, “The economics of peer-to-peer networks,” *Heinz Research Showcase, Carnegie Mellon University*, 2002. [Online]. Available: <http://repository.cmu.edu/heinzworks/52/>
- [74] Y. Chen, R. Sion, and B. Carbunar, “Xpay: practical anonymous payments for tor routing and other networked services,” in *WPES’09*, 2009, pp. 41–50.
- [75] V. Vishnumurthy, S. Chandrakumar, and G. Emin, “KARMA : A Secure Economic Framework for Peer-to-Peer Resource Sharing,” *1st Workshop on Economics of Peer-to-peer systems, Berkeley, CA, June 2003.*, 2003.
- [76] B. Wilcox-O’Hearn, “Experiences Deploying a Large-Scale Emer-

- gent Network.” in *IPTPS 2002*, 2002.
- [77] B. Stiller, P. Reichl, and S. Leinen, “Pricing and Cost Recovery for Internet Services : Practical Review , Classification , and Application of Relevant Models Pricing Models in Practice,” *Netmonics:-Economic Research and Electronic Networking*, vol. 2, no. 1, 2000.
- [78] R. Cocchi, S. Shenker, D. Estrin, and L. Zhang, “Pricing in Computer Networks : Motivation , Formulation , and Example,” *ACM/IEEE Transaction on Networking*, pp. 1–27, 1993.
- [79] P. Dasgupta and V. Kalogeraki, “Auctioning strategies in an agent enabled peer-to-peer marketplace.” in *sixth international conference on artificial intelligence. Nevada Las Vegas*, 2002.
- [80] M. Zghaibeh, “A Lottery-based pricing scheme for peer-to-peer networks,” in *IEEE International Conference on Telecommunication Systems*, 2008, pp. 903–908. [Online]. Available: <http://www.springerlink.com/index/53N3422058811LM3.pdf>
- [81] B. Yang and H. Garcia-molina, “PPay : Micropayments for Peer-to-Peer Systems,” in *Proceedings of the 10th ACM conference on Computer and communications security*, no. 3, 2003.
- [82] S. Nair, E. Zentveld, B. Crispo, and A. Tanenbaum, “Floodgate: A micropayment incentivized p2p content delivery network,” in *Computer Communications and Networks, 2008. IC-*

- CCN '08. Proceedings of 17th International Conference on*, Aug 2008, pp. 1–7.
- [83] M. Karakaya, “A connection management protocol for promoting cooperation in Peer-to-Peer networks,” *Computer Communications*, vol. 31, pp. 240–256, 2008.
- [84] M. Feldman and J. Chuang, “Overcoming free-riding behavior in peer-to-peer systems,” *ACM SIGecom Exchanges*, vol. 5, no. 4, pp. 41–50, Jul. 2005. [Online]. Available: <http://portal.acm.org/citation.cfm?doid=1120717.1120723>
- [85] A. Legout, S. Antipolis, N. Liogkas, and E. Kohler, “Clustering and Sharing Incentives in BitTorrent Systems,” in *ACM SIGMETRIC 2007 International Conference on Measurement and Modeling of Computer System*, 2007.
- [86] B. Cohen, “Incentives Build Robustness in BitTorrent,” in *P2PECON Workshop Berkeley*, 2003.
- [87] T. Locher, S. Schmid, and R. Wattenhofer, “Rescuing Tit-for-Tat with Source Coding,” in *P2P '07 Proceedings of the Seventh IEEE International Conference on Peer-to-Peer Computing*, 2007.
- [88] Z. Ma, “A novel optimistic unchoking algorithm for bittorrent,” *Consumer Communications and Networking Conference, CCNC 2009*, pp. 1–4, Jan. 2009. [Online]. Available: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4784859>

- http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4784859
- [89] D. Qiu, “Performance Improvements of Peer-to-Peer File Sharing,” *INTECH Multimedia*, 2003.
- [90] W. Miao and Z. Yujun, “Odd for Even: A Selfishness Prevention File Swap Scheme for BitTorrent,” in *GLOBECOM 2010, IEEE Global Telecommunications conference*, 2010, pp. 1–6. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5685234
- [91] K. Shin and D. Reeves, “Treat-before-trick: Free-riding prevention for BitTorrent-like peer-to-peer networks,” in *IPDPS '09 Proceedings of the 2009 IEEE International Symposium on Parallel&Distributed Processing*. Ieee, May 2009, pp. 1–12. [Online]. Available: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5161007>
http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5161007
- [92] J. Wang, R. Shen, C. Ullrich, H. Luo, and C. Niu, “Resisting free-riding behavior in bittorrent,” *Future Gener. Comput. Syst.*, vol. 26, no. 8, pp. 1285–1299, Oct. 2010. [Online]. Available: <http://dx.doi.org/10.1016/j.future.2009.05.014>
- [93] B. Zhu, S. Jajodia, and M. S. Kankanhalli, “Building trust in peer-to-peer systems: a review,” *Int. J. Secur. Netw.*, vol. 1, no. 1/2, pp. 103–112, Sep. 2006. [Online]. Available:

<http://dx.doi.org/10.1504/IJSN.2006.010827>

- [94] F. Azzedin and M. Maheswaran, “Trust modeling for peer-to-peer based computing system,” in *Int. Parallel and Distributed Processing Symposium*, 2002, pp. 291–297.
- [95] M. Lik, “Computational Models of Trust and Reputation: Agents, Evolutionary Games, and Social Networks,” *PhD Thesis, Massachusetts Institute of Technology*, 2002.
- [96] Z. Qing, Y. Ting, and I. Keith, “A Classification Scheme for Trust Functions in Reputation-Based Trust Management,” *In ISWC04, 3rd International Semantic Web Conference, Workshop on Trust, Security, and Reputation on the Semantic Web (2004)*, 2004.
- [97] R. Gupta and Y. N. Singh, “A reputation based framework to avoid free-riding in unstructured peer-to-peer network,” *CoRR*, vol. abs/1307.6658, 2013.
- [98] M. Meulpolder, J. Pouwelse, D. H. J. Epema, and H. Sips, “Bartercast: A practical approach to prevent lazy freeriding in p2p networks,” in *Parallel Distributed Processing, 2009. IPDPS 2009. IEEE International Symposium on*, May 2009, pp. 1–8.
- [99] M. Amad, D. Assani, A. Meddahi, and A. Boudries, “A pragmatic and scalable solution for free riding problem in peer to peer networks.” in *Modeling Approaches and Algorithms for Advanced Computer Applications*, 2013, pp. 135–144.

- [100] F. A. Azzedin, "Classifying and Tracking Free Riders in Multimedia-Based," *Journal of Universal Computer Science*, vol. 16, no. 10, pp. 1368–1387, 2010.
- [101] A. Farag and K. Salman, "Towards Trustworthy Peer-To-Peer Environments: An Appraisal Analysis Approach," *Journal of Next Generation Information Technology*, vol. 1, no. 1, pp. 61–76, May 2010. [Online]. Available: http://www.aicit.org/jnit/paper_detail.html?q=7
- [102] F. Azzedin, "Trust-Based Taxonomy for Free Riders in Distributed Multimedia Systems," in *International Conference on High Performance Computing and Simulations*, 2010, pp. 362–369.
- [103] Y. Tang and H. Wang, "Trust Based Incentive in P2P Network," in *Proceedings of the IEEE International Conference on E-commerce Technology for Dynamic E-business*, 2004, pp. 302–305.
- [104] L. Guo, S. Yang, L. Guo, K. Shen, and W. Lu, "Trust-aware Adaptive P2P Overlay Topology Based on," in *Grid and Cooperative Computing, 2007. GCC 2007. Sixth International Conference on*, vol. 1, no. 2006, 2007.
- [105] B. K. Shrivastava, G. B. Kumar, G. Khataniar, and D. Goswami, "A 3-Tier Hierarchical Peer-to-Peer System with Fair Incentive

- Scheme to Avoid Free Riders,” in *15th International Conference on Advanced Computing and Communications (ADCOM 2007)*. Ieee, Dec. 2007, pp. 688–693. [Online]. Available: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4426047>
- [106] K. I. S. Feldman, M Lai and J. Chuang, “Robust incentive techniques for peer-to-peer network,” in *EC 04*. ACM Springer-Verlag, 2004.
- [107] M. Feldman, “Free-Riding and Whitewashing in Peer-to-Peer Systems ,” in *Proceedings of the ACM SIGCOMM workshop on Practice and theory of incentives in networked systems*, 2004.
- [108] M. Feldman and J. Chuang, “The Evolution of Cooperation under Cheap Pseudonyms,” *Seventh IEEE International Conference on E-Commerce Technology (CEC’05)*, pp. 284–291, 2005. [Online]. Available: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1524056>
- [109] M. J. Osborne, *An Introduction to Game Theory*. Oxford University press, 2003.
- [110] W. Wang and B. Li, “To play or to control: a game-based control-theoretic approach to peer-to-peer incentive engineering,” in *In International Workshop on Quality of Service*. ACM Springer-Verlag, 2003, pp. 174–192.
- [111] J. Philip and D. K. Levine, “Evolution and information in a gift-

- giving game,” *Journal of Economic Theory*, vol. 100, pp. 1–21, 2001.
- [112] R. Krishnan, M. Smith, and Z. Tang, “The impact of free-riding on peer-to-peer networks,” in *the 37th Annual Hawaii International Conference on System Sciences*, 2004, pp. 1–10. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1265472
- [113] W. Li, J. Chen, and B. Zhou, “Game theory analysis for graded punishment mechanism restraining free-riding in p2p networks,” in *Proceedings of the 2011 International Symposium on Computer Science and Society*, ser. ISCCS ’11. Washington, DC, USA: IEEE Computer Society, 2011, pp. 262–266. [Online]. Available: <http://dx.doi.org/10.1109/ISCCS.2011.78>
- [114] C. Buragohain and D. Agrawal, “A game theoretic framework for incentives in P2P systems,” in *International conference on Peer-to-Peer Computing*, vol. 93106, no. 1, 2003, pp. 45–56. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1231503
- [115] M. Jun, Seung, Ahamad, “Incentives in BitTorrent Induce Free Riding ,” in *P2PECON05*, 2005.
- [116] Y. Matsuda, M. Sasabe, and T. Takine, “Evolutionary game theory-based evaluation of p2p file-sharing systems in heteroge-

- neous environments,” *Int. J. Digital Multimedia Broadcasting*, 2010.
- [117] Q. Zhang, H.-F. Xue, and X. dong Kou, “An evolutionary game model of resources-sharing mechanism in p2p networks,” in *Proceedings of the Workshop on Intelligent Information Technology Application (IITA'07), Zhang Jiajie, China, 2-3 December, 2007*, 2007, pp. 282–285.
- [118] S. Brian, *The Stag hunt and Evolution of Social Structure*. Cambridge University Press, 2003.
- [119] R. Izhak-Ratzin, N. Liogkas, and R. Majumdar, “Team incentives in bittorrent systems,” in *Computer Communications and Networks, 2009. ICCCN 2009. Proceedings of 18th International Conference on*, Aug 2009, pp. 1–8.
- [120] R. Izhak-Ratzin, “Collaboration in bittorrent systems,” in *Proceedings of the 8th International IFIP-TC 6 Networking Conference*, ser. NETWORKING '09. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 338–351. [Online]. Available: <http://dx.doi.org/10.1007/978-3-642-01399-727>
- [121] S. Bernard, “Is life a game we are playing,” *Ethics*, vol. 77, no. 3, pp. 209–219, 1967.
- [122] P. Morris, *Introduction to Game Theory*. Springer New York, 1994.

- [123] T. E. S. Raghavan, “Zero-sum two-person games,” R. J. Aumann and S. Hart, Eds. Elsevier Science, 1994, ch. Handbook on Game Theory, pp. 736–759.
- [124] Y. Shoham and K. Leyton-Brown, *Multiagent System: Algorithmic, Game Theoretic and Logic Foundation*. New York,: Cambridge University Press,, 2009.
- [125] C. Shih-Fen, V. Daniel, M Reeves Yevgeniy, and P. W. Michael, “Notes on equilibria in symmetric games,” in *International Joint Conference on Autonomous Agents & Multi Agent Systems, 6th Workshop On Game Theoretic And Decision Theoretic Agents*, 2004.
- [126] J. Nash, “Non-cooperative games,” *The Annals of Mathematics*, vol. 54, pp. 286–295, 1951.
- [127] L. Pavel, *Game Theory for Control of Optical Networks, Static and Dynamic Game Theory: Foundations and Applications*. Springer Science, 2012.
- [128] W. Saad, Z. Han, M. Debbah, and A. Hjrunghes, “Coalitional game theory for communication networks: A tutorial,” in *IEEE Signal Processing Magazine, Special Issue on Game Theory*, 2009, pp. 77–97.
- [129] J. C. Harsanyi, “Games with incomplete information played by bayesianplayers, parts i, ii, and iii,” *Management Science*, vol. 14,

no. 3, 1967.

- [130] E. David and K. Jon, *Networks, Crowds, and Markets: Reasoning about a Highly Connected World*. Cambridge University press, 2010.
- [131] H. C. John, “Games with incomplete information,” *Nobel Lectures*, 1994.
- [132] W. Robert, “Computing equilibria of n-person games,” *Society for Industrial and Applied Mathematics Journal*, vol. 21, no. 1, pp. 80–87, 1971.
- [133] T. Basar, *Lecture Notes on Non-Cooperative Game Theory*. Electrical and Computer Engineering, University of Illinois at Urbana Champaign, 2010.
- [134] D. Fudenberg and T. Tirole, *Game Theory*. MIT press, 1991.
- [135] G. J. Mailath and S. Larry, *Repeated games and reputations : long-run relationships*. Oxford University Press,, 2006.
- [136] trackon, “<http://www.trackon.org/>,” 2014. [Online]. Available: <http://www.trackon.org/>
- [137] opentracker, “<https://opentrackers.org/>,” 2014. [Online]. Available: <https://opentrackers.org/>
- [138] torrentz, “<http://torrentz.eu/tor/torrentz+org-q>,” 2013. [Online]. Available: <http://torrentz.eu/tor/torrentz+org-q>

- [139] thepiratebay, “<http://thepiratebay.sx/>,” 2013. [Online]. Available: <http://thepiratebay.sx/>
- [140] R. L. Xia and J. K. Muppala, “A survey of bittorrent performance,” *Commun. Surveys Tuts.*, vol. 12, no. 2, pp. 140–158, Apr. 2010. [Online]. Available: <http://dx.doi.org/10.1109/SURV.2010.021110.00036>
- [141] A. Legout, G. Urvoy-Keller, and P. Michiardi, “Rarest First and Choke Algorithms Are Enough,” in *ACM SIGCOMM/USENIX IMC’2006*, Rio de Janeiro, Brazil, Oct. 2006. [Online]. Available: <http://hal.inria.fr/inria-00091678>
- [142] H. Wang, F. Wang, J. Liu, K. Xu, and D. Wu, “Torrents on twitter: Explore long-term social relationships in peer-to-peer systems,” *Network and Service Management, IEEE Transactions on*, vol. 10, no. 1, pp. 95–104, March 2013.
- [143] T. Locher, P. Moor, S. Schmid, and R. Wattenhofer, “Free Riding in BitTorrent is Cheap,” in *HotNets 2006*, 2006, pp. 85–90.
- [144] M. Zghaibeh and F. Harmantzis, “Revisiting free riding and the tit-for-tat in bittorrent: A measurement study,” *Peer-to-Peer Networking and Applications*, vol. 1, no. 2, pp. 162–173, 2008. [Online]. Available: <http://dx.doi.org/10.1007/s12083-008-0013-7>
- [145] S. L. Blond, A. Legout, and W. Dabbous, “Pushing

- bittorrent locality to the limit,” *Computer Networks*, vol. 55, no. 3, pp. 541 – 557, 2011. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S1389128610003051>
- [146] A. Montresor and M. Jelasity, “PeerSim: A scalable P2P simulator,” in *Proc. of the 9th Int. Conference on Peer-to-Peer (P2P’09)*, 2009.
- [147] I. R. Khondkar, “A Cost-effective Distributed Architecture for Content Delivery and exchange over emerging Wireless Technologies,” *PhD Thesis, George Mason University*, 2013.
- [148] R. Centeno, H. Billhardt, and R. Hermoso, “Persuading agents to act in the right way: An incentive-based approach,” *Eng. Appl. Artif. Intell.*, vol. 26, no. 1, pp. 198–210, Jan. 2013. [Online]. Available: <http://dx.doi.org/10.1016/j.engappai.2012.10.001>
- [149] H. Salah and M. Eltoweissy, “Towards a personalized trust management system,” in *Innovations in Information Technology (IIT), 2012 International Conference on*, March 2012, pp. 373–378.
- [150] Y. Zhu and Y. Yang, “Research on It-based communication between peers in bt system,” in *Consumer Electronics, Communications and Networks (CECNet), 2012 2nd International Conference on*, April 2012, pp. 1980–1983.
- [151] F. Fabrizio and M. Pedrolli, “A bittorrent module for peersim,”

in *University of Trento*, 2008.

- [152] Z. B. Ahmed, S. A. May, and Z. H. Hala, “Adaptive sliding piece selection window for bittorrent systems,” *Advances in Multimedia - An International Journal (AMIJ)*, vol. 2, no. 1, pp. 18–29, Feb 2011.
- [153] G. G. Konstantinos, “Challenges in Cooperation between Internet Service Providers and Peer-to-Peer Applications,” *Msc Thesis, George Mason University*, 2010.
- [154] D. Carra, R. Lo Cigno, and E. Biersack, “Stochastic graph processes for performance evaluation of content delivery applications in overlay networks,” *Parallel and Distributed Systems, IEEE Transactions on*, vol. 19, no. 2, pp. 247–261, Feb 2008.

Vitae

Mohammed Onimisi Yahaya was born on November 2, 1978 in Ipaku, Adavi Local Government Area of Kogi State, Nigeria. He completed his B.Tech (First class honour) and MSc in Computer Science from Abubakar Tafawa Balewa University of Technology (ATBU), Bauchi, Nigeria in 2004 and 2009 respectively. Mr. Onimisi has worked as a lecturer in Mathematical Sciences program of ATBU from 2005 to 2009. He joined King Fahd University of Petroleum & Minerals (KFUPM), Dhahran, Saudi Arabia in September 2009. Mr. Onimisi just completed his PhD in Computer Science and Engineering (CSE) at KFUPM in 2014.

Mohammed Onimisi is a citizen of Nigeria and can be contacted at:

Permanent Address: Osuwa's Compound Ogaminana-Okene
Road Ipaku, Adavi Local Government Area Kogi State, Nigeria

Contact Address : Mathematical Sciences Program Abubakar
Tafawa Balewa University P.O. Box 0248, Bauchi, Nigeria

Email: *mdonimisi@gmail.com*