# STUDY OF INTELLIGENCE PEER SELECTION IN ENHANCING P2P USING ARTIFICIAL NEURAL NETWORK

BY

CHAM HUI

A PROJECT

SUBMITTED TO

Universiti Tunku Abdul Rahman,

In partial fulfillment of the requirements

for the degree of

MASTER OF INFORMATION SYSTEMS

Faculty of Engineering and Science

November 2014

UNIVERSITI TUNKU ABDUL RAHMAN

# REPORT STATUS DECLARATION FORM

Title: **STUDY OF INTELLIGENCE PEER SELECTION IN ENHANCING P2P USING ARTIFICIAL NEURAL NETWORK**

I, CHAM HUI declare that I allow this Final Year Project Report to be kept in

Universiti Tunku Abdul Rahman Library subject to the regulations as follows:

1. The dissertation is a property of the Library.

2. The Library is allowed to make copies of this dissertation for academic purposes

Verified by,

_____ _____

Address:

_____

_____

_____ _____

DR TAY YONG HAUR

Date: _____ Date: _____

# STUDY OF INTELLIGENCE PEER SELECTION IN ENHANCING P2P USING ARTIFICIAL NEURAL NETWORK

BY

CHAM HUI

A PROJECT

SUBMITTED TO

Universiti Tunku Abdul Rahman,

In partial fulfillment of the requirements

for the degree of

MASTER OF INFORMATION SYSTEMS

Faculty of Engineering and Science

November 2014

# DECLARATION OF ORIGINALITY

I declare that this report entitled "**STUDY OF INTELLIGENCE PEER SELECTION IN ENHANCING P2P USING ARTIFICIAL NEURAL NETWORK**" is my own work except as cited in the references. The report has notbeen accepted for any degree and is not being submitted concurrently in candidature for any degree or other award**.**

Signature: _____

Name : _____

Date :_____

# ACKNOWLEDGEMENT

I would like to use this great opportunity to give thanks to everyone who has given their support on my study and research work. Thousand thanks to my supervisor Dr Tay Yong Haur for consistent support, motivation, guidance, and full support on documentation and administration work. He also consistent support throughout my master research. I appreciate his guidance, patient and support in my system development and thesis writing.

Department of Internet Engineering and Computer Science, Faculty of Engineering and Science, Universiti Tunku Abdul Rahman,for their kind advice, helping, and supporting.

Last but not least, I would like to share the achievement of this work of mine to my friends and family especially my parents and my friends Kong Polian, Lau Chia Fong, Yip Fu Cheang, and Cheam Heng Soon. I wouldn't have completed this research without their understanding, help and support.

# ABSTRACT

P2P network technology had consumed a large proportion of total Internet traffic and is significantly increasing during recent years. The application designed for data and information sharing effectively and allows quick dissemination of information to avoid bottlenecks caused frequently on the dedicated servers. Free riding is one of the drawbacks in P2P networks. It occupied the resources shared in the network and with less or without contribution back to the network. This reduces the performance of whole P2P environment and holds unnecessary open connections which might consume the network resources of internet. The main objectives of this project are to build up a better network environment for P2P applications with neural network technology and reduce the overhead cost of P2P networks. The application will train the neural network with details and information provided by peers and prioritizes the peers during the data sharing process. The enhanced P2P client shows the download speed of P2P application accelerated by comparing it with the default P2P client. It also reduced the occurrence of free riding peers in connected peers.

# TABLE OF CONTENTS

# LIST OF FIGURES

# 1.0 INTRODUCTION

## 1.1 Defining P2P network

Peer-to-Peer (P2P) network is a communication model that permits data sharing between multiple users with the same capabilities. In other words, P2P network can be defined as a system that enables individual users (or nodes) to portray as both client and server, contrasting to the traditional client/server network. The system is particularly designed for each node to have the same functions, such as to share computer resources (e.g., files and printers), directly among small-scale groups. Networks that implement P2P can effectively share resources, as well as quick dissemination of information to avoid bottleneck events that occur frequently on the dedicated servers. The method employed by P2P to achieve this is through the distribution of divided data segments in the network for maximum number of nodes to act as servers of the information sharing process. Rapid global Internet growth is one of the reasons that lead to the widespread of resource sharing, thus increasing its practicality in business, in a situation where the nodes could not be identified.

In terms of marketing, P2P networks have gained favor due to the increasing interest of its applications. (Tomoya & Shigeki, 2003) technology to marketing). Traffic measurement data of internet service providers (ISPs) shows that P2P applications occupied a substantial percentage of today's traffic of Internet. The aim of developing these P2P based applications is to maximize the accessibility of data and information to a large amount of users. As the nodes in P2P networks can serve as both client and server, the nodes can assist each other in many ways, such as file searching, file lookup, and anonymous information (I. Clarke, 2001). For instance, Centralized file look up approach P2P networks have transformed to a distributed object query method in file searching. The limitation of distributed object queries is that it could be affected by type of the controlled flooding. Thus, the latter version of P2P networks utilizes the technique of hashing consistently to enhance the file searching efficiency. In spite of such approaches carried out to enhance the operations in a P2P network, there are still issues and some

fundamentals regarding the fundamental cooperative paradigm of information and data exchange remain unresolved. Free riding and the disaster of commons were among the issues. It was reported that approximately 70% of P2P users share nothing within P2P environment, but act as “free-ride” on other users that willing to share in that particular community. As there are only few users who are willing to share or provide services, only about 50% of the entire file searching answer came from the top 1% of data and information sharing nodes. (Limewire). Hence, congestion is more common to occur in nodes which share resources and information, resulting among the disaster of commons problem (Hardin, 1968). As suggested previously, a P2P network should adapt desirable features in order to resolve the issues as described. The desirable properties proposed were (Ma):

1. Fairness: This refers to nodes that already contributed more data or information to the P2P network supposedly gets more resources in the environment of resource sharing.
2. Avoidance of resource wastage: This mechanism will not give too much resource to nodes than the amount this can be consumed. Whenever jamming on path of communication, this kind of mechanism can adjust to overcrowding level and reallocate resources.
3. Maximization of social utility: In some kind of situation, the mechanism does not only make best use of the utilities of users, but also get high cumulative utility for all users.
4. Adaptability and scalability: This mechanism can adjust to circumstances like leave or join of dynamic nodes. Since this mechanism executes at every involved node, the performance should be scalable even when size of network increases.

### 1.1.1 Traditional P2P system

In a P2P file sharing system, nodes are in direct communication with each other in information exchange and file sharing. There are a few categories of P2P systems. One of the categories is centralized P2P systems: The utilization of a centralized control server for system management like Napster. However, these systems experience a single point

of scalability, censorship problems, and failure (A. Oram, 2001). Another category is known as decentralized P2P systems, which could be further separated into partially decentralized systems and totally decentralized systems. For totally decentralized systems, no hierarchical structure in peers, such that all peers contains same function. However, partially decentralized systems like Morphus, Kazaa, Gnutella, peers are permitted to have different functions. For example, there are peers that perform as local central guides for local peers resource sharing. They are known as “supernodes” or “ultrapeers” due to their special role, and they are also assigned dynamically. If there are failure or malicious attacks, these peers can be replaced. On the other hand, proxy search requests for peers that are connected to them. Therefore, queries are delivered to supernodes, but not to the other peers. Depending on the resources available, a particular supernode is capable of supporting 300-500 peers (Gnutella2 Specification.). At the time being, partially decentralized systems could be the most frequently implemented in P2P networks. One of the successful examples is KaZaA, which can support greater than four million simultaneous peers, somemore more than 5281.54 TB of data available for downloading.

In traditional P2P systems, a number of peers which is able to provide the requested file is given to the user. In this case, the user needs to select one peer from the list of many peers to perform the download. It is rather difficult for users to choose the most reliable peer, as files provided by some peers may contain malicious content, and also to prove whether the downloaded file matches to requested file. If it doesn’t, the user have to repeat the process from scratch (Gnutella Protocol Specification v0.4.). In addition to that, limited information is provided to the user, causing more difficulty to choose the appropriate peer. The following briefly depicts the life cycle of a peer in a traditional P2P system:

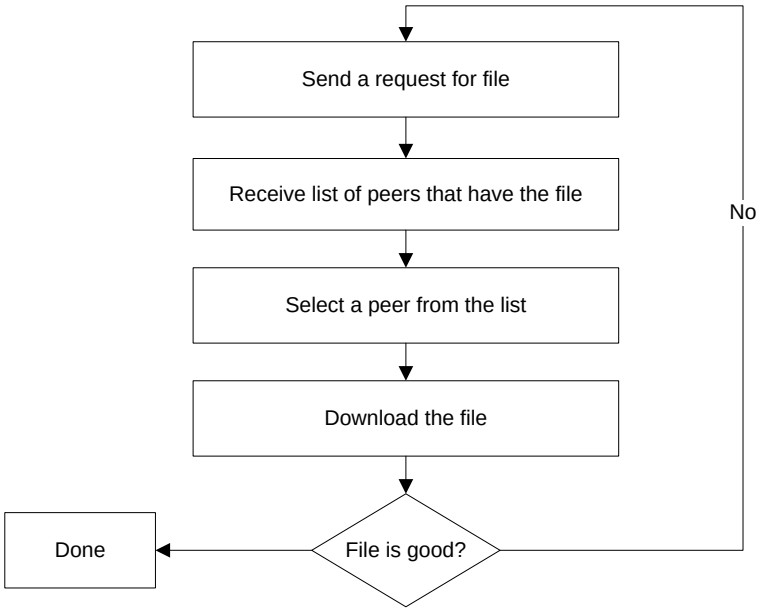



**Figure 1.1 Life cycle of a traditional P2P system**

(Mekouar, 2006)

1. Send message about the file request to other peers and supernode.
2. Receive a number of peers which could provide requested file.
3. Choose one of the peers from the provided list, which is done by user.
4. Download requested file from other peers.

It was suggested which majority of the shared content only supplied by 30% of the peers (E. Adar, 2000). Hence, a mechanism is required to resolve this issue by rewarding these peers, and also to motivate other peers to share their resources. Simultaneously, another mechanism is required to punish peers that provides malicious content or misleading filenames. Alternatively, these peers with malicious behavior must at least be isolated from that particular system. In order to resolve the drawbacks, reputation-based P2P systems were designed. The designation of reputation-based P2P systems is to function as a reputation managing system to evaluate every transaction carried out by peers, then to assign reputation points to peers. Values of reputation are crucial for selection of peers. The following briefly depicts the life cycle of a peer in a reputation-based P2P system:

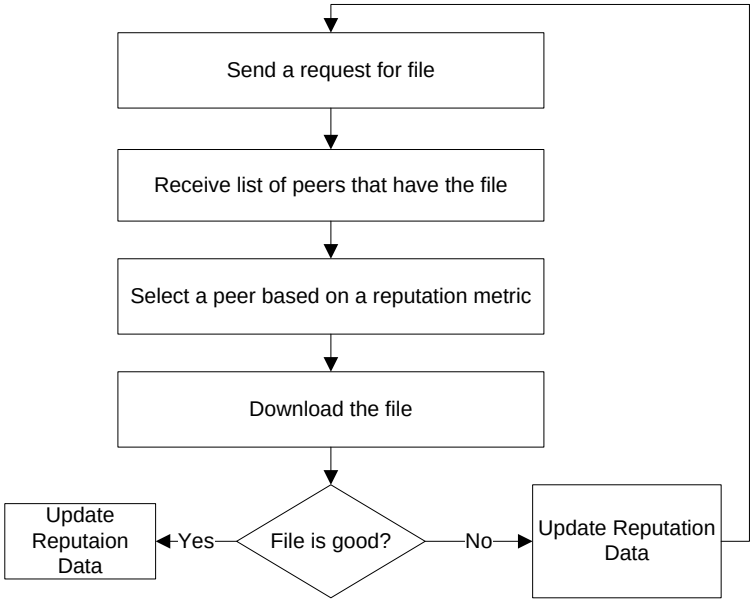



**Figure 1.2 the life cycle of a peer in a reputation-based P2P system**

Life cycle of a traditional P2P system (Mekouar, 2006)

1. Send a message of to request file.

2. Receive a number of peers which could provide requested file.

3. Choose a peer or a list of peers, according on the value on reputation metric.

4. Download requested file.

5. Provide feedback and update value of reputation.

### 1.1.2 Protocol of unstructured P2P network

In an unstructured P2P network, a peer is required open connections with the other peers which existed within that particular network. Right after a connection is established in that network, the user needs to deliver a Query message to its neighboring peers in order to search the network. Next, each of the neighbors will then forward the request to all of its neighbors, and these neighbors also forwards the request, and the cycle continues until request is passed by a predefined amount of "hops" from sender. In order to limit the

dissemination of a Query message, Time-To-Live (TTL) field of query message is responsible to control the number of hops. For each forwarding peer, the value of TTL declines by one. If zero is the TTL's value of the query, then the peers would drop the message rather than proceed to forwarding to its neighboring peers. However, if there is a result for the Query, the peer who accepts the result returns query hit message back to peer that first sends query message. The Query Hit message returned via the opposite direction of the Query that entered through, carrying the port number and IP address of the responding peer. When the time the user is interested to download the file, then it will request related file from provider via direct connection. (Karakaya, 2008)

### 1.1.3 Motivation and contribution

The suggestion of partially decentralized P2P systems to cut down the control overhead required to run the P2P systems is one of the contributions to resolve the drawbacks encountered. In addition to that, this is beneficial as it exhibits the feature of lower discovery time, as only supernodes were involved in the discovery process. A study has proposed a few reputation management systems, concentrating on decentralized P2P systems. For instance, only KaZaA brought in fundamental reputation metric, or in other terms "participation level" for ranked peers. Nevertheless, strategies of reputation management suggested were not applicable in partially decentralized systems, which fully depend on supernodes in controlling message exchange such that no direct management of messages is permitted in peers.

It may allow every peer to record information or details on previous interacting experiences with connected peers in decentralized systems, and may also utilize a voting system to call for feedbacks from peers that once requested the file. It was also proposed in the same study to use supernodes to control message exchange, as well as storage of reputation data of peers that it previously served, update and then provide these to other supernodes. Given that each peer could on interact with only one supernode, the suggestion may be more precise, as well as it may be possible to significantly reduce message overhead.

The fact that whenever search request is sent by a supernode on behalf a peer, supernode will receive a number of peers that could provide requested file, including the reputations

value of their supernodes. Hence, voting system is no required because the reputation is representative of all the past experiences from connected peers which once have cooperateed with those peers. On the other hand, the other use of a supernode is to provide service differentiation, with reference to the reputation data of those peers that were under its responsibility. (Mekouar, 2006)

## 1.2 Problem Statement

Free riding has been always a pitfall in P2P networks, as the peer utilizes resources shared in the network without contribution. In other words, a free rider is defined as a peer that exploits the resources shared in P2P networks with no contribution to the network to an acceptable level. In contrast, a contributor is a peer that shares its resources in the network with other peers.

Researches in this area have demonstrated that the amount of free riding is high in P2P networks, suggesting it a crucial threat that affects the operation efficiency in P2P networks (E. Adar, 2000) (Golle, 2001).

There are several consequences of free riding. Scalability problems occurs when a small amount of peers search a large amount of peers, with lots of download requests sent to only a small number of serving peers (Ramaswamy, 2003). Moreover, it also causes the network to function as a client-server like system, posing negative impacts on P2P benefits (Ripeanu, 2002) (Krishna, 2002). As an example, when only a very small community of the peers provide most of the resources, the fault-property of P2P networks might be diminished.

Replacement or introduction of new contents might reduce over time, and the quantity of shared resources might be slow growing. Furthermore, degradation in quality of the search process might occur because of the growing numbers of free rides in the search environment. Peers that have been in the network for a long time may tend to stop searching for interesting files, or leave system together with the resources which they have shared earlier in time (Ramaswamy, 2003) (Gummadi, 2003). Furthermore, the large amount of free riders and their requests will produce quite a number of the P2P network traffic, hence reducing the quality of P2P network services. Likewise, the

underlying free network capacity and network resources are going to be taken up by free riders, causing congestion and unnecessary delay for non-P2P traffic.

### 1.2.1 Free Riding Types

Earlier findings on free riding (Golle, 2001) (Kamvar, 2003) (Andrade, 2004) suggest that P2P networks exhibits specific one kind of free riding, yet there are some studies (E. Adar, 2000) (Markatos, 2002) (Ramaswamy, 2003) on user behavior and P2P network traffic demonstrated that not all kind of free riders possess the same behavior.

As described earlier in a study, there are three kinds of free riding, each with different characteristics. The kinds of free riding were dropper, non-contributor, and consumer, which are non-exhaustive, and there is a possibility to describe new kinds of free riding with dissimilar characteristics. (Karakaya, 2008) These types of free riding are considered sufficient for the development of common framework, and the types of free riding described is believed to occupy a large proportion of all free riders. The following are the detailed description for each type of the free riders:

**1. Non-contributor**

Peers that do not partake in shares uninteresting resources or sharing anything are defined as non-contributor. This kind of free riding might discovered by monitoring peer whom is responsible in counting the Query Hit messages (QHP) that first arises from the neighbor peers and then compares back to them to the amount of Query messages (QTP) which will sent to the neighbors.

**2. Consumer**

Consumer can be defined as peers that may share some resources to the network. Thus, they might not consider as a non-contributor, but instead the services they utilize might go beyond the resources contributed by them. As this affects the long-term steadiness of P2P network and cause unfairness to other peers, it is not desirable behavior.

**3. Dropper**

Peers are defined as droppers when that particular peer drop queries of other peers. Some of the peers may not forward queries in order to prevent the reduction of their connection bandwidth.

## 1.3 Objectives

The main objective of the proposed system is to contribute a better network environment for P2P applications. In recent years, P2P networks occupies a large percentage of the total Internet traffic, which were mainly constituted by P2P applications that involves data sharing or video streaming. As video streaming via P2P applications had been the main attraction to users for searching and viewing video content online, especially with increased popularity of mobile devices nowadays, P2P mechanisms therefore does have its importance in video streaming. The enhancement of P2P applications might assist in providing an optimal network environment for P2P operations to perform efficiently, and also it may resolve the problems encountered in P2P applications. Hence, the proposed system is designated to improvise the peer selection stage, which is one of the steps in P2P system. At this particular step, the client randomly selects some peers from its list of peers to connect and exchange information, a situation that commonly occurs at this stage (Zong, 2008) Therefore, the proposed system will utilize ANN to enhance the peer selection stage to boost the initial speed of P2P applications.

The second objective of the proposed system is to reduce the overhead cost of P2P networks. Usually, P2P applications will tend to establish thousands of network connections simultaneously in order to achieve the maximum speed of data sharing. This introduces unnecessary high latency in the network, and many of the connections are not in use. To avoid such situation, the proposed system is capable to achieve the maximum speed without the need to establish a lot of connections. In the meantime, the initial download speed can be improved via the assistance of ANN.

## 1.4 Scope of Work

### 1.4.1 ANN Based Peer Selection:

We selected ANN as our methodology to study the problem on problem statement. ANN's are capable of capturing high dimensional inputs and generate the relationships between the inputs and outputs from a training set. This is one of the reasons why ANN is proposed in this study. The similarity of the results were also captured in the encoded

internal representation. In brief, ANN is one of the computational tools to solve mixture resolution problems, which is also supported by a wide range of studies (Clarke, 2008) (Vasilescu, 2011). This study will cover on the utility of unsupervised ANN to enhance P2P application.

### 1.4.2 Comparison between current P2P and ANN Enhanced P2P:

This study will also compare between existing P2P software and ANN enhanced P2P software. The comparison will be made based on the perspective of initial connection speed, ratio of established connections used, and the hit rate of quality peers. A total of three P2P software will be selected for comparison as they are well-known, which eases the harvest of their statistical results.

### 1.4.3 Improve epidemic of leechers:

Intentional or unintentional leechers of P2P applications had always been an issue in the P2P society. In general, leecher can be defined as the user that benefits from resources shared by others, but does not share anything in return. Many leechers deliberately disconnect or remove the sharing after the download is completed. To minimize the leech scenario, the proposed system can be used during uploading at the peer selection stage. The method of improving the upload during downloading will be studied in the project.

# 2.0 LITERATURE REVIEW

The literature review present several existing solutions that were suggested to resolve the issues in P2P systems.

## 2.1 Free Rider Aware and Contribution Oriented Reputation System

A public key communications to provide the security and trust of reputation management plus transmit messages within a system was assumed to present in system initialization. Group managers are required as the founder of proposed system. At the same time, the group managers will build up several peers in order to offer file-sharing services in the beginning. It was suggested in a study that there are not less than two groups of managers in a group to maintain system reliability, as the group managers are responsible in the management of the peer's reputation and to perform the extended searching phase across groups. The system operation is proposed to be not affected even if one group manager shuts down. In the case when the amount of peers in particular group is growing at a high rate, some senior peers that have high reputations were allowed to establish a new group. One group can contain many peers based on their types of shared files, IP ranges, or other conditions as well. In summary, the connections between the new peer and the group manager are as follows:

1. A new peer is interested to join the system, he is allowed to select one of the manager and delivers a joining request message to the manager.
2. After receiving joining request message, the manager will validate it, and creates a record in a reputation table. In the meantime, the group manager will assign a initial reputation and the reputation level to that particular peer, and then delivers a notification to the peer.
3. When the peer succeeds in joining the system, the peers is required to categorize their shared files into several authorized according to his will. The peer might provide low authorization level to his data in order to increase his reputation

rapidly. Because on this, other peers will share these data or files constantly and this enhances his own value in reputation.

### 2.1.1 Basic transaction protocol

In the same study carried out, it also describes the operations in a basic transaction protocol. The protocol is if a peer would like to search for a specific file, for instance peer p, he sends a query message to other peers who located in his group. Then peers that have the file will return a query-hit message to peer p that, and peer p will request the group manager for obtain reputation of the providers and investigate whether the requested files are listed as malware. Peer p will then choose a candidate q who owns a good reputation and high bandwidth, and sends a download request message to that peer. When peer q receives the download request message, he will checkup the reputation of peer p by through the group manager. If peer p has a higher reputation than the authorized level of the file, peer q sends back a download reply message to peer p. After peer p downloads the file, he then sends the recommendation for peer q to the group manager. To pay attention that the managers of a particular group are allowed to switch current recommendations and periodically revise the reputation among with each other.

Basically, basic transaction protocol is separated into around four phases. These phases include the searching phase, reputation requesting phase, the downloading phase, and reputation updating phase.

**1. Searching phase**

Peer finds particular file by sending query message to all the peers in his group. Peer that owns that particular file will return a query hit message to the requesting peer.

**2. Reputation requesting phase**

A peer requests the reputation of peers that provides the file of interest from the one of the managers and investigates on whether the files provided are contained in the list of malicious files. For instance, peer p delivers a reputation request to one of the group managers, in which the request contains of all the identities of all the peers that returns the query hit messages during the searching phase. A manager then sends back reputation levels of the peers (peers that return the query hit messages) to peer p. Peer p will check

whether the files were contained in the list of malicious files, which is constantly maintained by the group managers.

**3. Downloading phase**

A peer chooses those candidates that have high reputation levels and then delivers download request to a provider than owns a higher bandwidth among these candidates. The chosen provider will then look into the reputation level of the requesting peer and sends back download reply message whenever requesting peer has a good reputation level to download the file. An encryption key is contained in download reply message, which needed for security conformation during the downloading process of the file. The requesting peer will have to decrypt the download reply message in order to retrieve the encryption key.

**4. Reputation updating phase**

After retrieving the file, the requesting peer reports a recommendation regarding the peer that provides the specific file (service provider) to one of the managers, and service provider delivers service finishing to one of the managers. A more detailed description of this phase is as follows:

As peer p completes the downloading phase, he then delivers a recommendation update message to one of the managers. At the same time, peer q (service provider) then sends service finished message to the manager.

Group manager will verify both recommendation update message and service finish message. In the situation when the recommendation update message regarding peer q is negative, the group manage will request peer q to justify against the negative recommendation of peer p, especially when peer q is reported of spreading malicious file.

### 2.1.2 Extended transaction Protocol

When a high reputation peer, peer p, could not find files during the basic transaction phase, the peer is allowed to call for one of his connected managers for executes extended transaction protocol. The group manager will first check the reputation level of the peer, let’s say if the reputation level is particularly high, for example the group manager j the forwards the request message to group managers of other groups. Upon receiving the message, each group manager of other group managers broadcasts the message to all the peers in their respective groups. Similar to the basic transaction protocol, the group

managers will forward the request message to all the peers of their groups respectively, and then choose peers with better reputation level. Next, the managers will forward all extended query-hit message to the group manager j that initially sends the request message, and then group manager j forwards all extended query-hit messages to the requesting peer. Peer p then selects a candidate, peer q, a peer with the best reputation level, and then directly deliver an extended download request to peer q who is from the other group, which is the service provider. Peer q will then send back an extended download reply message to peer p. After downloading the required file, peer p will then report recommendation to manager j. The manager j will then forward the recommendation to other managers.

Extended transaction protocol can be further separated into three phases, as following:

**1. Extended searching phase**

For those peer with good reputation level is not able to find the desired files in basic transactions phase, the peer is permitted to demand one of his managers to execute next extended searching phase. For instance, peer p delivers an extended search message to one of managers in order for run extended search. The manager looks into reputation level of peer p. Let's say if peer p is of high reputation level, group manager j send the message to the other managers. The other group manager will then broadcast the extended search message to all the peers in his group. Each peer who has file sends to extended query hit message to manager. Managers will check whether the file already marked in one of malicious file inside the list, after that c provider has better reputation level. After it, group managers will return an extended query-hit message to group manager j. Group manager j then confirms all extended query hit messages from other managers, then returns the extended query-hit messages to peer p.

**2. Extended downloading phase**

With similarity to the downloading phase in basic transaction protocol, the only dissimilarity of both phases is peer q isn't permitted to access reputation value of peer p as peer p is of good reputation in order for execute an extended transaction protocol. Peer p chooses the file with best reputation and then sends an extended download request to peer q. Peer q then returns the extended download reply message to peer p. Peer p will decrypt the message in order to for obtain encryption key, which can decrypts file.

**3. Extended reputation updating phase**

With similarity to reputation updating phase inside the basic transaction protocol, peer p returns recommendation of provider q to manager j. Then group manager sends this recommendation to the group manager of peer q. After peer p finishes the extended downloading phase, then delivers a recommendation update message to group manager j. In the meantime, peer q will send service finished message to his own group manager. Group manager j then verifies the recommendation update message, and then forwards it to peer q's group manager. The group manager of peer q will have to verify the recommendation update message from group manager j. Let's say if recommendation update message regarding peer q is negative, the group manager will inform peer q to justify against to the negative recommendation from peer p, particularly with concern to malicious files spread. The manager of peer q then periodically updates the reputation of peer q via a reputation evaluation procedure. (Tseng, 2011)

## 2.2 2 dimensional Reputation Based Policies System

According to a study, the reputation-based policies can be divided into two dimensions: "Provider selection" and "contention resolution". "Provider selection" will be in charge to select the providing peer among peers that offers the same service, whilst "contention resolution" regulates the selection of the peer to be actually served, from all peers that requests service from same peer with little amount of resources. Limited resources assumption of a peer was considered as realistic, and it could have associated with upstream capacity of every peer, it's accessibility of CPU cycles. (Papaioannou, 2004). The dimensions described above was as well depicted in two other publications (Hwang, 2004) (Ma)

**1) Provider selection policies**

The highest reputation is defined as the highest reputation value, with condition that the availability of reputation metric in performance confirmed within P2P system, and reasonable policy for every peer for choose a peer within peers that could offer requested service (Kamvar S. D.-M., 2003). On the other hand, comparable reputation is defined as a policy termed "Peer-Approved" as reviewed in a paper (Ranganathan, 2003). The policy states that peers are allowed to receive services from other peers with lower value

or equal rating. The probability to find a peer to provide the requested service increases if the peer is seen to have improvement in performance, and therefore the reputation value. Nevertheless, this poses a situation that the quality of the received service is doubted, as the requesting peer could choose services among peers that has lower value in reputation, and thus, low performance. Due to this, the study suggests another policy to resolve this problem, termed "Comparable Reputation" (Papaioannou, Reputation-based policies that provide the right incentives in peer-to-peer environments, 2006). In this policy, it was proposed that peers are only can request services from peers which have reputation in equality with them, for instance, by having pre-specific maximum difference.

The basic design of the proposed policy is pairing of level of performance which already supplied by a peer with given level of performance to it. Therefore, layered communities are formed with this policy implemented. Peers that were in the same layer shuffle services of related quality. For instance, in a P2P system, peers with high performance level in the top layer offers high quality services, whereas peers in the lowest layer could offer services that are useless or the worst can be harmful to other peers. Black list, another suggested policy to eliminate peers that possess low reputation values below an acceptable level in a P2P system, especially peers that provides services of low quality in a consistent manner for a certain time period. These peers will be disqualified from group of qualified providing peers. It will improve quality of services for provision to other remaining peers.

**2) Contention resolution policies**

In the actual environment of a P2P system, peer who had already entertained some of the previous requests may receive a new service request. As the peer have finite resources, it has to choose on whether to block or serve or place new request in the queue list, with concern to the quality provided to those that already being served. A contention resolution policy is employed to make this decision. In systems where low-performing peers exceeds the amount of high-performing peers and when provider selection policies are used, competition to get resources that latter often occurs. The applicability of the

suggested model is by assuming time for slotted. Numerous requests may be sent to single peer during starting of every time slot. The suggested model takes into account regarding the issue on limited resources of a single peer, and forces the providing peer to serve a limited amount of peers successfully at single time slot. This policy utilizes the "highest reputation" policy to select peers to be served (Hwang, 2004). Peer that possess the highest-reputation value should be selected for being served by providing peer from peers requests service from latter, which as well competes for the resources. The policy gives full priority to peers with highest-reputation values. A symmetric randomized rule is used to resolve ties. Therefore, a peer that owns a high reputation value had a higher chance for the service to be provided. Nonetheless, the end result of service provision relies on provider selection policy utilized at the P2P system. Let's say if the policy is engaged and there are severe resources contentions, peers have lower reputation values will not be given any services. Due to this, a solution is suggested to resolve this situation, termed "probabilistically fair with respect to reputation", or to be referred as "probabilistically fair". Based on this policy, the peer to be served for the requested service is relies on the following condition:

Of all the peers j who demands for same service from single peer, chance for single peer j will be chosen to receive the service equals $R_i = P_j R_j$, where $R_j$ represents peer j's reputation valuej. Again, peers which have high reputation value are prioritized over others, yet following current policy, peers have low values in reputation are often positive, perhaps a minor chance to receive some services without concerning on who they compete with. In the last stage, in a situation when every peer contending for single resource have same values in reputation, both contention resolution policies are in accord. (Papaioannou, Reputation-based policies that provide the right incentives in peer-to-peer environments, 2006)

## 2.3 FuzzyTrust System Architecture

The FuzzyTrust prototype system was developed to evaluate peer reputation in P2P transactions after investigate the features of transaction data of eBay. FuzzyTrust was constructed via the fuzzy-logic inference method utilized within the sidebar. Above

all, this system has the ability to handle inaccurate or unsure information gathered from connected peers.

### 2.3.1 System Design Requirements

Three important design criteria were suggested, according to the features of eBay's transaction:

1. The consumption of network bandwidth in order to switch local trust scores to hot spots might very high. Hence, reputation system might judge the unbalance transactions within user for e-transactions.

2. In order to handle the minor impact from small users, a reputation system should not practice the same evaluation process for all peers. The frequent users must receive more update more than small users.

3. It is logic to evaluate the large amount of transactions than the minor ones with a skewed transaction amount.

The system functions by operating two major inference steps, which were described as follows:

1. **Local-Score Computation**

   Every peer operate fuzzy inference in the local parameters in order for produce local scores in fuzzy trust. The fuzzy inference mechanism is capable of capturing some uncertainties and could adapt itself, or in other words, self-adjusting. It can adapt itself to record the changes of local parameters, like delivery time, payment time, quality of goods, payment method, and others.

2. **Global Reputation Aggregation**

   The system will gather local trust scores gathered from other peers in order for generate a global reputation for every peer. System uses fuzzy inference to retrieve global reputation aggregation weights. The aggregation weights is determined by using three variables: transaction date, the peer's reputation, and transaction amount. The amount rules of fuzzy inference can expand into few hundreds within a full-scale P2P reputation system. Nevertheless, five frequently

used fuzzy inference rules employed to the prototype FuzzyTrust system construction is listed as following:

1. The aggregation weight will be very large when a reputation of peer is good and amount of transaction is high.
2. The aggregation weight will be very large when transaction amount considered high and transaction time is new.
3. The aggregation weight will be small when the transaction amount considered low or transaction time is very old.
4. The aggregation weight will be medium when reputation of peer is good and amount of transaction is low.
5. The aggregation weight will be very small when reputation of peer is low.

**Implementation of DHT-Based Overlay**

The prototype FuzzyTrust system was implemented on DHT-based P2P overlay network, it has a design like Chord 4, a DHT ring which is capable of secure message transmission and fast trust aggregation. The advantages Chord system is tough to failure, highly scalable, and self-organizing in which takes care of peer joining or leaving from the system.

Two tables were preserved by each peer: First, transaction recorded table for sustain transaction records with remote peers, and the second is local score table to sustain evaluated trust scores of remote peer.

The global aggregation weights were inferred via the fuzzy inference system with reference to the transaction records. Every peer queries and calculate trust scores from the remote peers when operating global reputation aggregation. The system partly queries capable peers which meet an aggregation point in order to encounter the hot-spot issue. (Song, 2005)

# 3.0 METHODOLOGY

This chapter will explain the methodology of system development life cycle (SDLC) which will apply in the project of Optimum Peer Analyze. Some diagram will explain the architecture and design of the system, process and workflow of the system.

The Systems Development Life Cycle (SDLC) in software engineering and information system is the processes to understand how an information system can fulfill the user requirements, designing the system, developing and building it and delivering it to users. The process including planning, analysis, design and implementation, the result in a high quality system that meets customer's expectations, complete within time and cost estimates, works effectively and efficiently with planned schedule, and is inexpensive to maintain and cost effective to enhance.

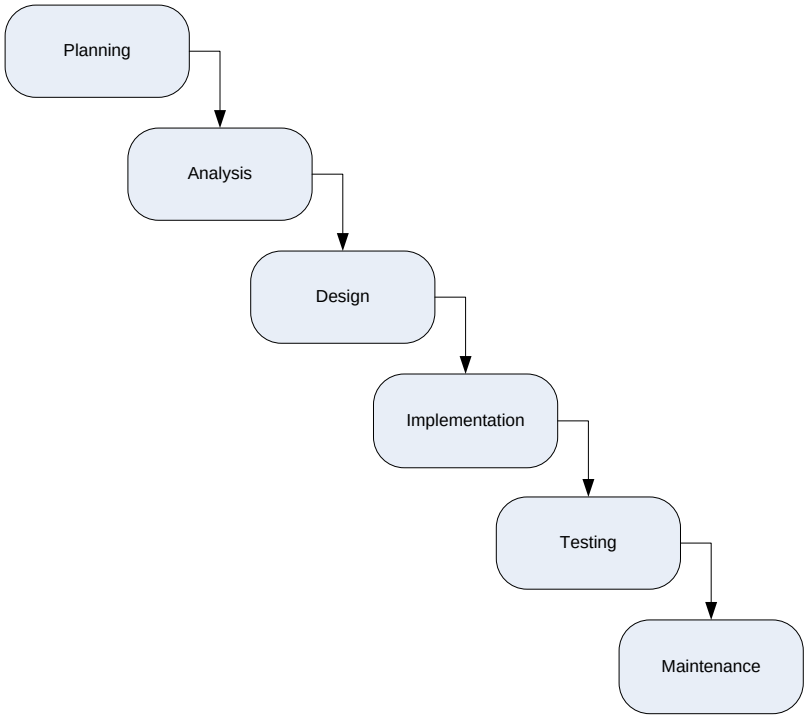



**Figure 3.1 SDLC waterfall model**

The SDLC is a waterfall model and proceed in sequence which the stage move from one to the next step and generally a stage is finished before the next stage begins. Generally there are some basic steps, it started from planning. Planning can be considered to find out the requirement and problem statement which need to be solved. Some basic research

and some literature review need to be done. It provides the basic for acquiring the resources needed to come out a solution. The second step basically is analysis that analysis the current problem and user requirement to create a detail functional requirement document. Then is design phase, it change detail requirement into complete system design document that focus on how to solve the problem with detail functionality design, such as architecture design, process flow, system design, and interface design. The next step is implementation, it will start do the coding part and develop the system. After this is testing part, the system will be testing and debugging. It will be evaluate how useful is the system. The last step is maintenance, how to maintain the system after it delivered.

In the Intelligence Peer Analyzer, it will follow the basic waterfall model in development. Below is the step for develop the system.

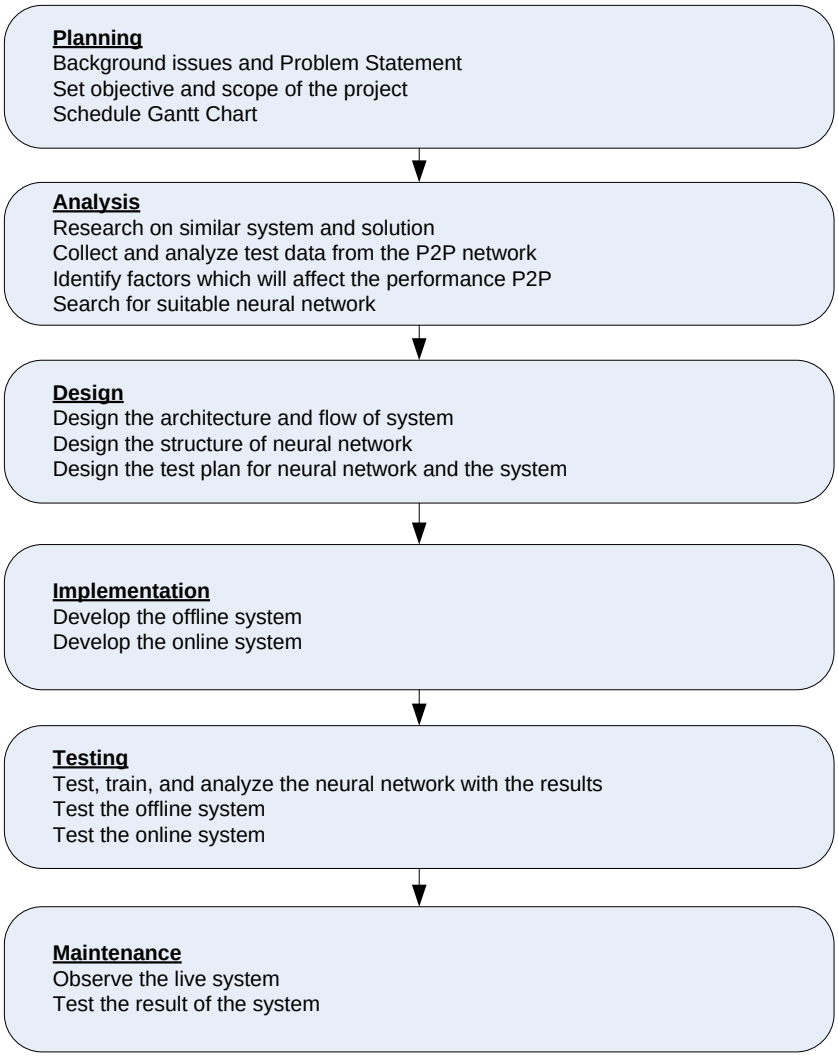



**Figure 3.2 Description of each stage and activities within each stage**

### 3.0.1 Planning

First stage is planning and it start with confirm the project title for the system which is going to develop. Then identify the background issues and the problem statement of the project. Find out the related tools, solution or existing system which can solve the current problem and find out the suitable type of neural network for the project. Then need to set objective, goals and scope for the project. Besides this, author schedule the project plan by using Gantt chart.

### 3.0.2 Analysis

During the analysis stage, literature review is doing research and analysis the techniques or system on similar suggestion to solve the problem and issues occurred at P2P network. Based on the journal and system provided by other journal, I can know how the other researchers solve from different view and have an idea to design and implement my system.

### 3.0.3 Design

The design stage will focus on the method and design for the system. From the literature review and research done on analysis stage, I will design the architecture of the system and the flow of the process and data. At the same time, I also need to design the neural network to process the peer details to do the training and testing. The testing plan also needs to carry out on this stage. The testing plan will evaluate the performance and effectiveness of the system.

### 3.0.4 Implementation

Development and coding are doing on this stage. The offline and online system will develop to train, test, and use the chosen neural network. The offline part will develop before the online part because it will use to train neural network before can use it on online part.

### 3.0.5 Testing

The testing will execute follow the test plan on designed earlier. The testing will evaluate the training and effectiveness of neural network. After it, the system will be tested together with other P2P system.

### 3.0.6 Maintenance

The system will continue running and observe the live running of it within period of time.

## 3.1 System Architecture

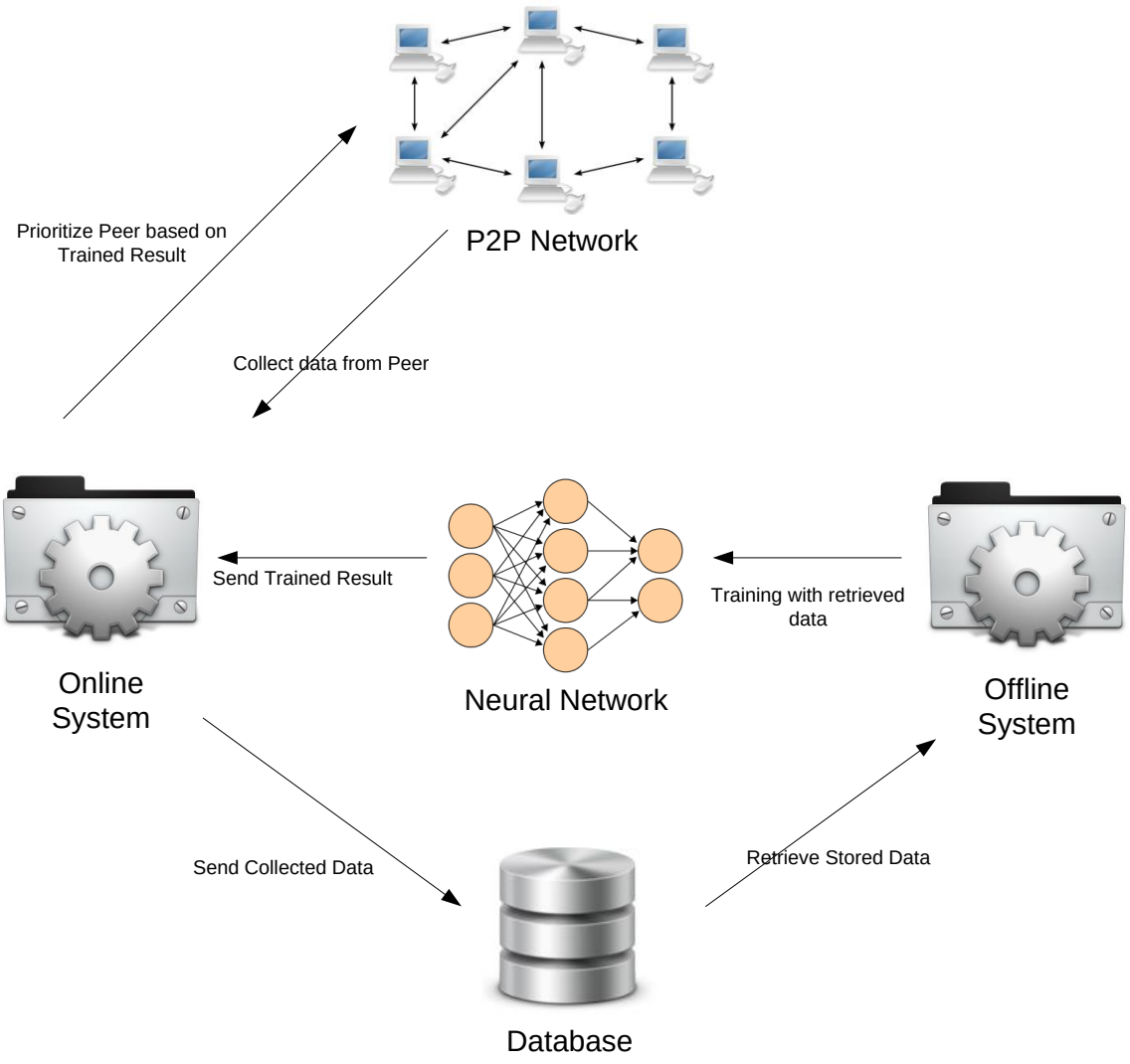



**Figure 3.3 Architecture of System**

Above is the basic architecture of the system. The system separate into 2 part consists of online and offline system. The online system will integrate into P2P client software. It will collect data from peer in P2P network and prioritize the peer in P2P client. The offline system will retrieve stored data and train the neural network. Further process will be discussed on the following paragraph.

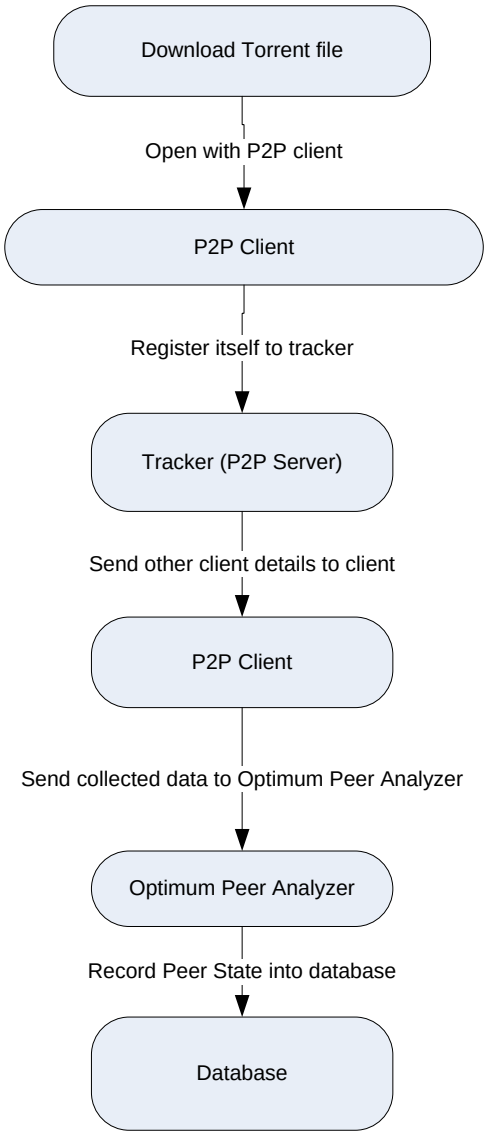



**Figure 3.4 Collect Peer Details Process**

The diagram above show how the data collect from starting. The P2P client opens the downloaded torrent file and registers itself to the tracker. The tracker will send other client details to the client. The client will send the collected data to the Optimum Peer Analyzer. Then the online part of the system will record and reorganized the peer state and data to the database.

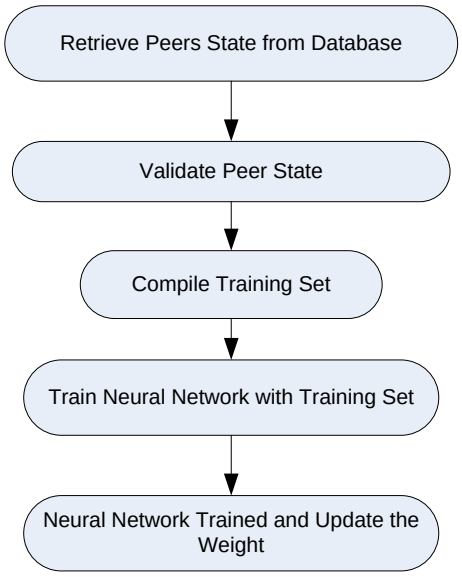



**Figure 3.5 Offline System Training**

The figure show the how the offline part of the system train the neural network. Firstly the OPA will retrieve stored peers state from database daily. Then it will validate the peer state to reduce the noise within the data. Some of the abnormal data will reduce effectiveness and accuracy of the neural network. After that, it will compile the training set and send it to neural network for training. Neural network will be trained by the training set few times. Every time OPA will randomized the order and send it to neural network for training.

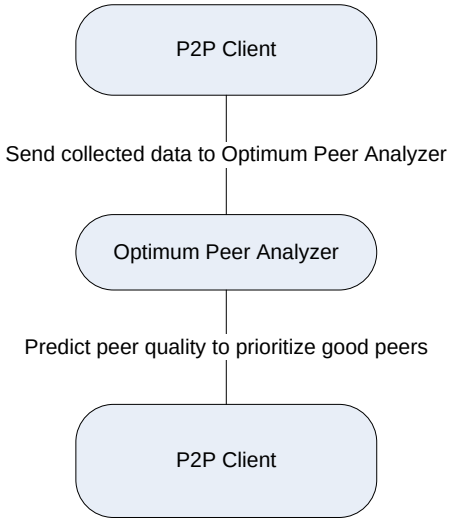



**Figure 3.6 Predict and Prioritize Process**

Lastly, every time P2P client send collected peer data to Online System, it will analyze the peer by using the neural network every few minutes. Then OPA will predict the peer quality and tell P2P client how to prioritize the good peers.

## 3.2 Structure of Neural Network

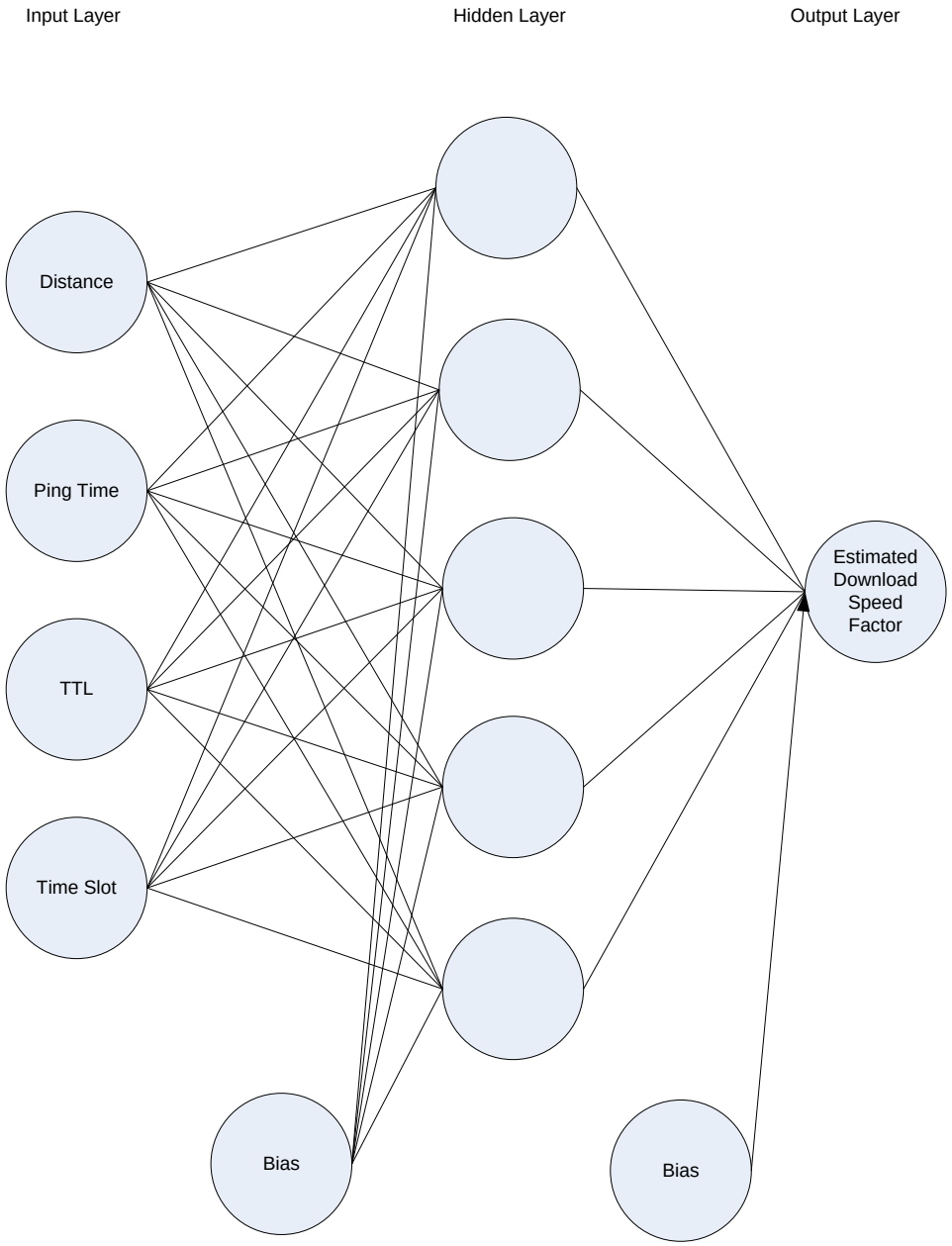



**Figure 3.7 Structure of Neural Network**

The figure shows the structure of neural network. Inside the first layer, it will have 4 input nodes. The distance is estimation of geographical distance between the peers. The ping time is the response time within the peers. The Time to live (TTL) is having a concept similar to hop count. Time slot is divided by 5 minutes for each session in the scale of 1 week. The hidden layer has 5 nodes and out layer has 1 node which is the estimated download speed factor. 2 bias nodes feeding the hidden layer and output layer. This neural network is using back-propagation and feed-forward technique. Feed forward neural network is one of the simplest yet efficient artificial neural networks. The information only moves 1 direction from the input nodes, through the hidden nodes then the output nodes. Back-propagation is a neural network learning method. This is due to the method will calculate gradient of loss function with respects to all weights in the network backward from output layer back to hidden layer. The gradient is provide to optimization method uses it to update weights and try to reduce the lose function.

## 3.3 Training Neural Network

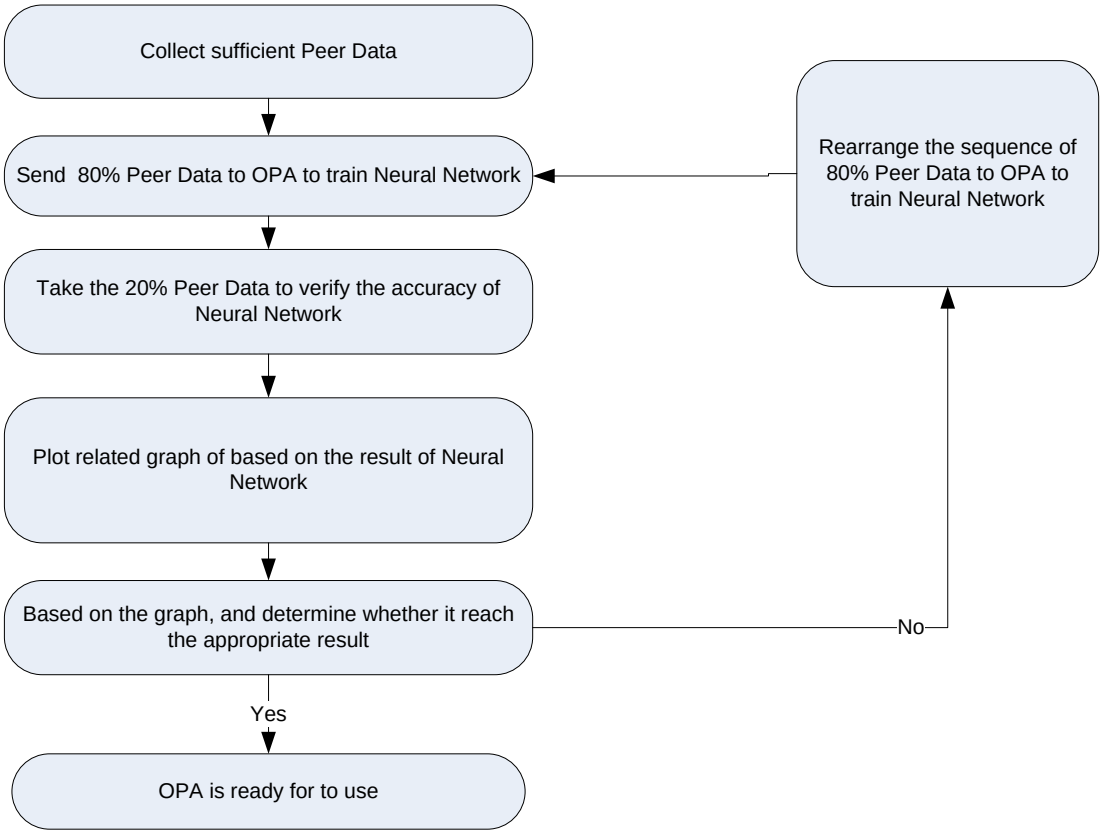


**Figure 3.8 Training Neural Network Method**

The figure illustrates the method for training neural network. First peer data is collected by downloading the file via P2P client. Before training, each training set is validated for a complete session, incomplete session will be ignored. Then 80% of training set will be sending to OPA to train the neural network. The remaining 20% will be using to verify the accuracy of prediction. The tested result will plot on the graph to determine whether OPA is ready to go live. If it didn't reach the acceptable error range, the training set will be shuffle and send back to OPA again for training until it reaches the desired result or the max epoch is being hit. After this stage, it means neural network training is completed.

## 3.4 Test Plan

To test the efficiency and accuracy of Neural Network, I will draft a mean absolute error vs epoch graph to test the result provided neural network. If the graph showing a convergence pattern, this means the neural network is working as expected. Else it means the setting of neural network is wrong or insufficient parameters. After it, 2 download performance profiles which are the normal P2P client and P2P client embedded with neural network will be compared. The comparison performance of P2P clients is using a download speed vs time graph. It will show the effectiveness of P2P with neural network.

# 4.0 RESULT AND DISCUSSIONS

## 4.1 Experiment Setup

To prepare the data for neural network, I need to collect sufficient data for training and testing. I prepare a computer installed with P2P client Vuze. The computer will use to collect data of peers when P2P client download the files. For creating the fairness for time period data, I started the download with P2P client from Monday of first week till the Sunday of Second week. After the data of peers collected, I filtered the noise in the peers' data. Some of the peer data has null value or abnormal value. Then I separated the data into 2 groups which is training dataset and testing data set. Lastly, I executed the training of neural network and test plan based on previous chapter to provide the following results. Then, I prepared 2 computers with the same specifications and setup in the same network. One of the computers is installed with Vuze and another computer is also installed with Vuze together with the neural network peer priotizer. Next, I prepared 20 different torrent files for testing the performance of neural network peer priotizer. After the preparation of torrent file, I started the testing for performance. The download for each torrent in both computers starts at the same time, so that I can simulate the similar P2P condition for both computers. The download speed vs time data of both computers were collected and recorded in appendix A.

## 4.2 Neural network

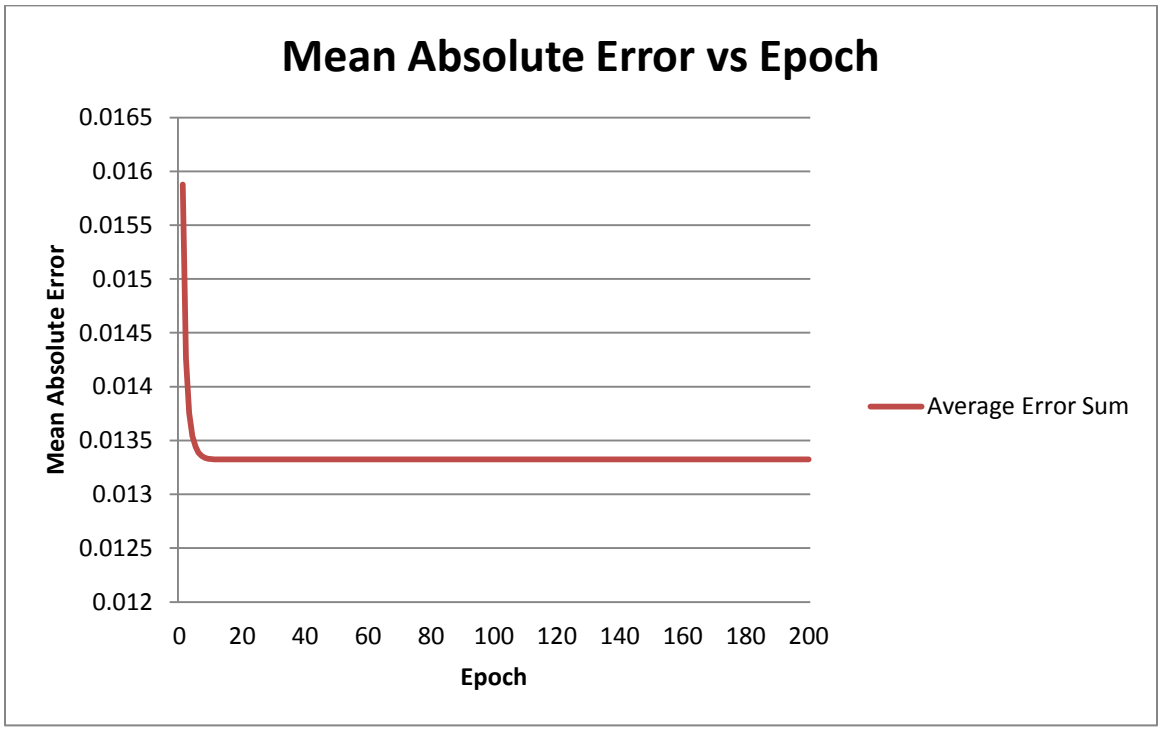



**Figure 4.1 Mean Absolute Error vs Epoch**

The figure above is the mean absolute error vs epoch graph. Below is some basic information of the testing:

1. Training Records : 9717
2. Testing Records:1944
3. Learning Rate: 0.9
4. Momentum: 0.04
5. Epoch: 200
6. Hit Rate: ±10KB/s

The graph is showing a convergence pattern. It reached the minimum mean absolute error on the 16$^{th}$ epoch. It shows the neural network is learning as expected and trying minimizes the prediction error. Although the error translates to actual download speed is around 16KB/s, however from the hit rate graph, prediction with 10KB/s is already cover around 40% of the testing training set. This shows the neural network has the ability to predict the result to certain degree of accuracy.

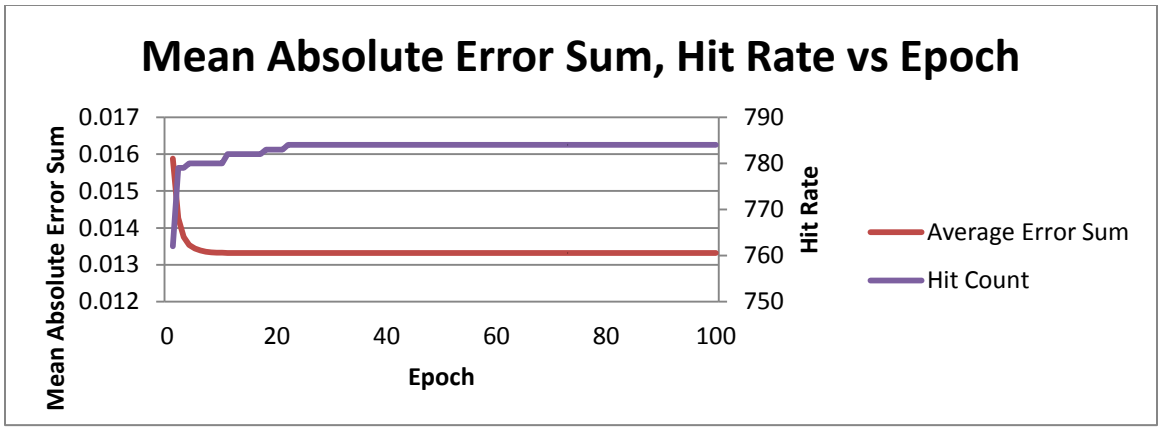



**Figure 4.24 Mean Absolute Error Sum, Hit Rate vs Epoch**

The hit rate is the predicted results which fall on the plus minus 10KB/s of target value.

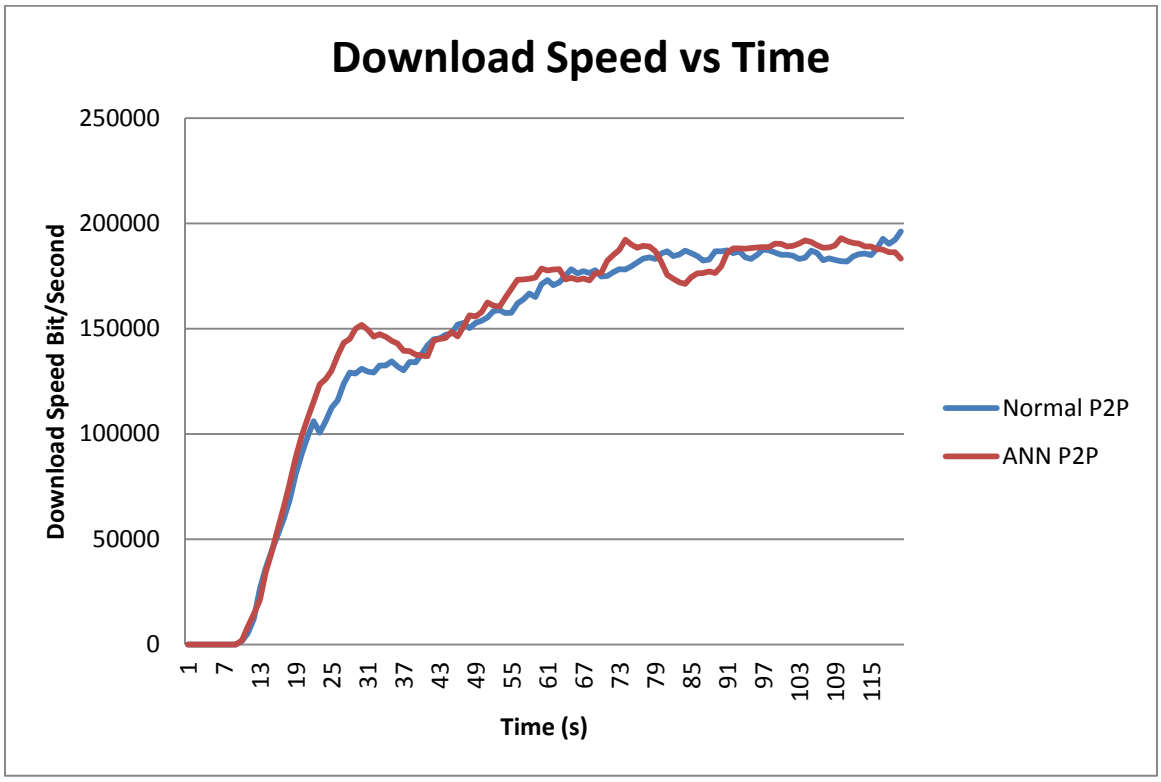



**Figure 4.3 Download Speed vs Time**

The torrent is selected based on 50 or below seeds and 200 or below peers. The seed to peers ratio is around 1:4. For each download is downloaded 2 times to get the average download speed. Total 2 PC is download concurrent in the same network.

The appendix A are the data of download speed vs time table with default P2P and data of download speed vs time with Neural Network P2P. The figure above is the average performance of ANN P2P vs Normal P2P. During the initial stage, ANN P2P show higher acceleration during time 20 to 29 seconds. It shows 20kb/s higher than the normal P2P. It show the ANN has an impact on the initial download performance.

# 5.0 CONCLUSION

As a conclusion, the Neural Network based P2P system OPA has an impact on the earlier download performance based on the performance graph. This implies that accuracy of 40% hit rates of approximately 10kb/s is able to reduce the resource in order to get effective peers or good peers. This means that the overhead and the idle connection is possible to be reduced due to the capability of the neural network to predict, which the accuracy is within the range of approximately 16kb/s, and covers 90% of the peers based on the statistics collected and around 40% of total is based on approximately 10kb/s. This will pose a positive effect on video streaming application. Since OPA is the supplement system to the P2P client, it does not monopolized or cut off user connection but instead suggests prioritizing the user connections which predicted as an effective peers. So that the early download performance will be improved. Due to the simplicity of the neural network and the input parameters chosen, the accuracy of the neural network falls within 16kb/s. It is possible to further optimize with other pin point data, such as the actual physical location and user account of the client etc.

Since the internet network status is unstable and very volatile, the learning rate must be sufficiently high to capture the recent changes. Hence, the expected pattern captured is by scheduling system which embedded with most of the P2P client to specify the download and upload periods. Human behavior or their working hours may form a certain patterns of download period. In the future enhancement, community user rating system based on automatic collection regarding the upload and download ratio may be integrated into the system. The human routine pattern may be captured by the neural network implicitly to enhance the accuracy of prediction.

# BIBLIOGRAPHY

A. Oram. (2001). *Peer-to-Peer: Harnessing the Power of Disruptive Technologies.* O'Reilly Books.

Andrade, N. B. (2004). Discouraging free riding in a peer-to-peer cpu-sharing grid. *Proceedings. 13th IEEE International Symposium* (pp. 129-137). High performance Distributed Computing.

C, C. (2008). *Development of an automated identification system for nanocrystal encoded micropheres in flow cytometry.* Cranfield University.

Clarke, C. (2008). *Development of an automated identification system for nano-crystal encoded microspheres in flow cytometry.*

E. Adar, B. H. (2000). *Free riding on Gnutella.* TechRep.

*Gnutella Protocol Specification v0.4.* (n.d.). Retrieved from http://www9.limewire.com

*Gnutella2 Specification.* (n.d.). Retrieved from http://www.gnutella2.com

Golle, P. L.-B. (2001). Incentives for sharing in peer-to-peer networks. *Electronic Commerce* , 75-87.

Gummadi, K. P. (2003). Measurement, modeling, and analysis of a peer-to-peer file-sharing workload. *ACM SIGOPS Operating Systems Review* , 314-329.

Hardin, G. (1968). The tragedy of the commons. *science* , 162(3859), 1243-1248.

Hwang, J. &. (2004). Agent-based modeling for differentiated admission in P2P systems using evolutionary game theory focused on ownership reputation. *Proc. of the Workshop on Economics of Peer-to-Peer Systems.*

I. Clarke, O. S. (2001). Freenet:A distributed anonymous information storage and retrieval.

Kamvar, S. D.-M. (2003). The eigentrust algorithm for reputation management in p2p networks. *Proceedings of the 12th international conference* (pp. 640-651). World Wide Web .

Kamvar, S. Y.-M. (2003). *Addressing the non-cooperation problem in competitive P2P networks.* 1st NetEcon.

Karakaya, M. K. (2008). Counteracting free riding in Peer-to-Peer networks. *Computer Networks* , 52(3), 675-694.

Krishna, S. S. (2002). *A measurement study of peer-to-peer file sharing systems.*

Limewire. (n.d.). a Gnutella client software.

Ma, R. T. *A game theoretic approach to provide incentive and service differentiation in P2P networks.* ACM.

Markatos, E. P. (2002). Tracing a large-scale peer to peer system: an hour in the life of gnutella. *Cluster Computing and the Grid, 2002. 2nd IEEE/ACM International Symposium* , 65-65.

Mekouar, L. I. (2006). Peer-to-peer's most wanted: malicious peers. *Computer Networks* , 50(4), 545-562.

Papaioannou, T. G. (2004). Effective use of reputation in peer-to-peer environments. *Effective use of reputation in peer-to-peer environments. In Cluster Computing and the Grid* (pp. 259-268). IEEE.

Papaioannou, T. G. (2006). Reputation-based policies that provide the right incentives in peer-to-peer environments. *Computer Networks* , 563-578.

Ramaswamy, L. &. (2003). Free riding: A new challenge to peer-to-peer file sharing systems. *Proceedings of the 36th Annual Hawaii International Conference* (p. 10). System Sciences.

Ranganathan, K. R. (2003). To share or not to share: An analysis of incentives to contribute in collaborative file sharing environments. *Workshop on Economics of Peer-to-Peer Systems* .

Ripeanu, M. F. (2002). *Mapping the gnutella network: Properties of large-scale peer-to-peer systems and implications for system design.* arXiv preprint cs/0209028.

Song, S. H. (2005). Trusted P2P transactions with fuzzy reputation aggregation. *Internet Computing IEEE* , 24-34.

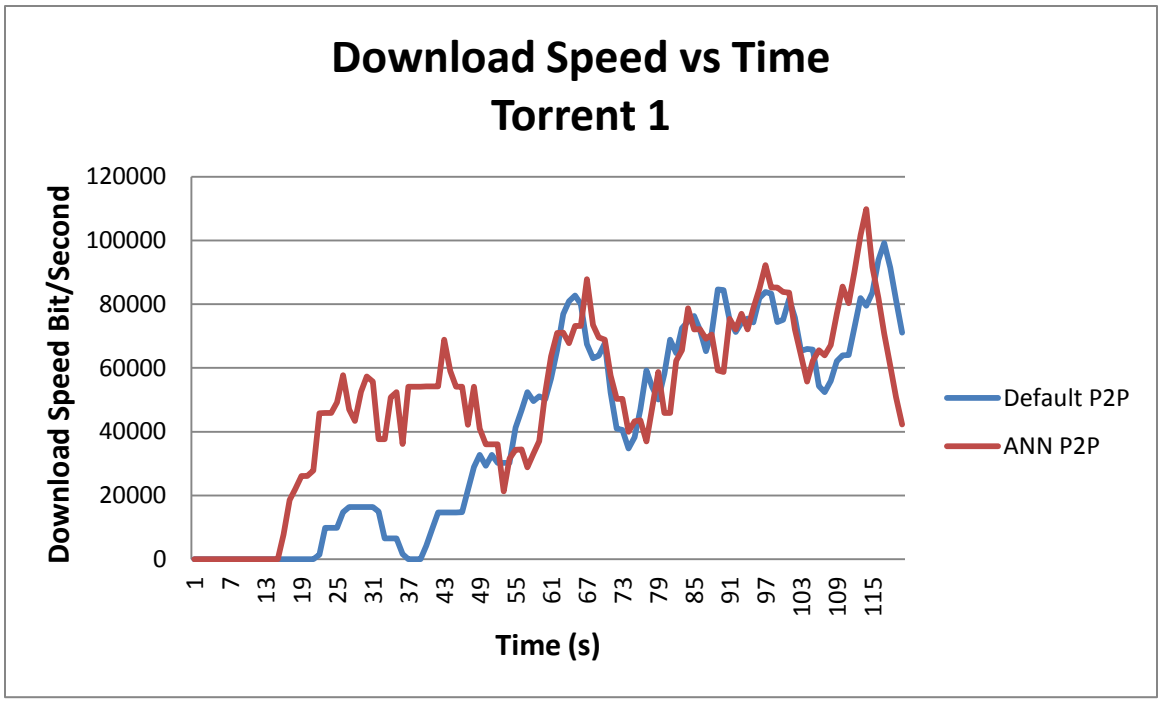
Tomoya, K., & Shigeki, Y. (2003). Application of P2P (peer-to-peer) technology to marketing,. *Cyberworlds, 2003. Proceedings. 2003 International Conference* (pp. 372,379,). IEEE.

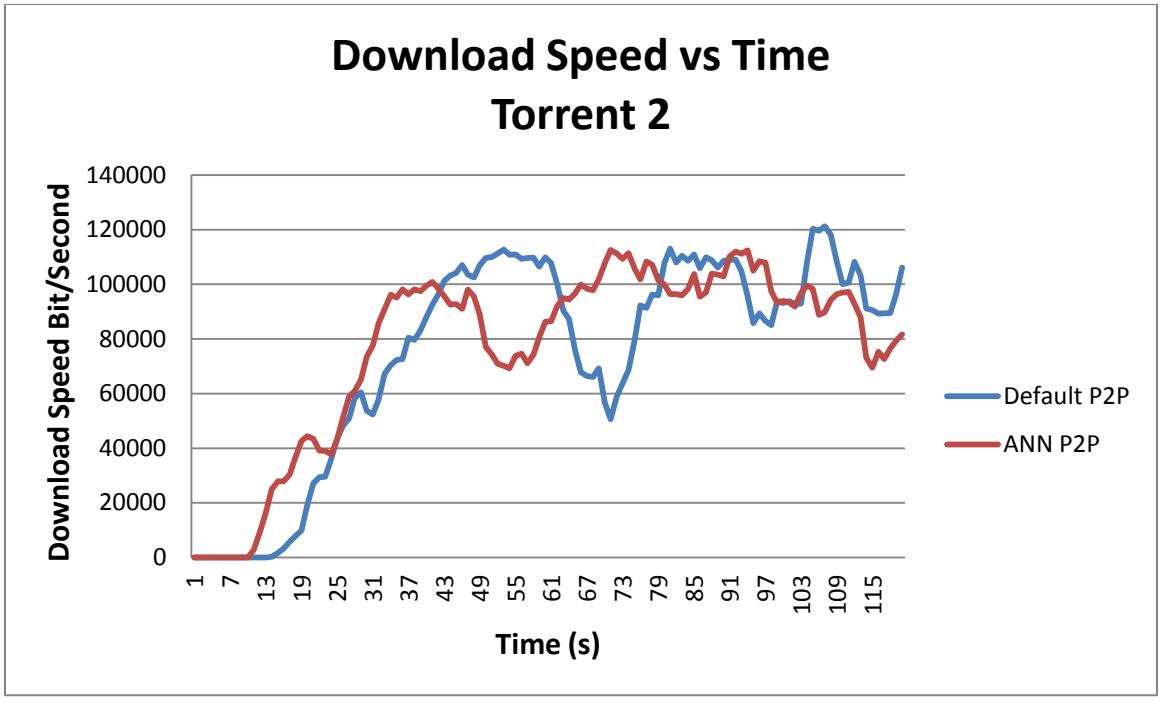
Tseng, Y. M. (2011). A free-rider aware reputation system for peer-to-peer file-sharing networks. *Expert Systems with Applications* .

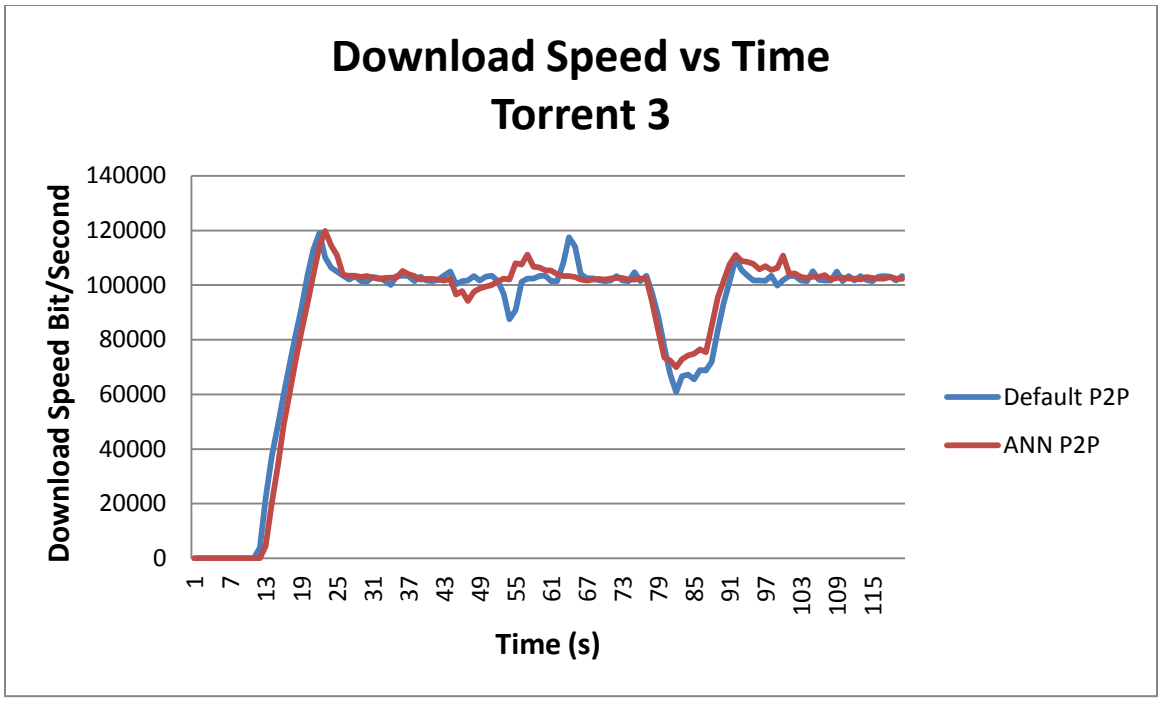
Vasilescu, J. M. (2011). Analysis of seawater pollution using. *Romainian Journal of Physics* , 56, 530-539.

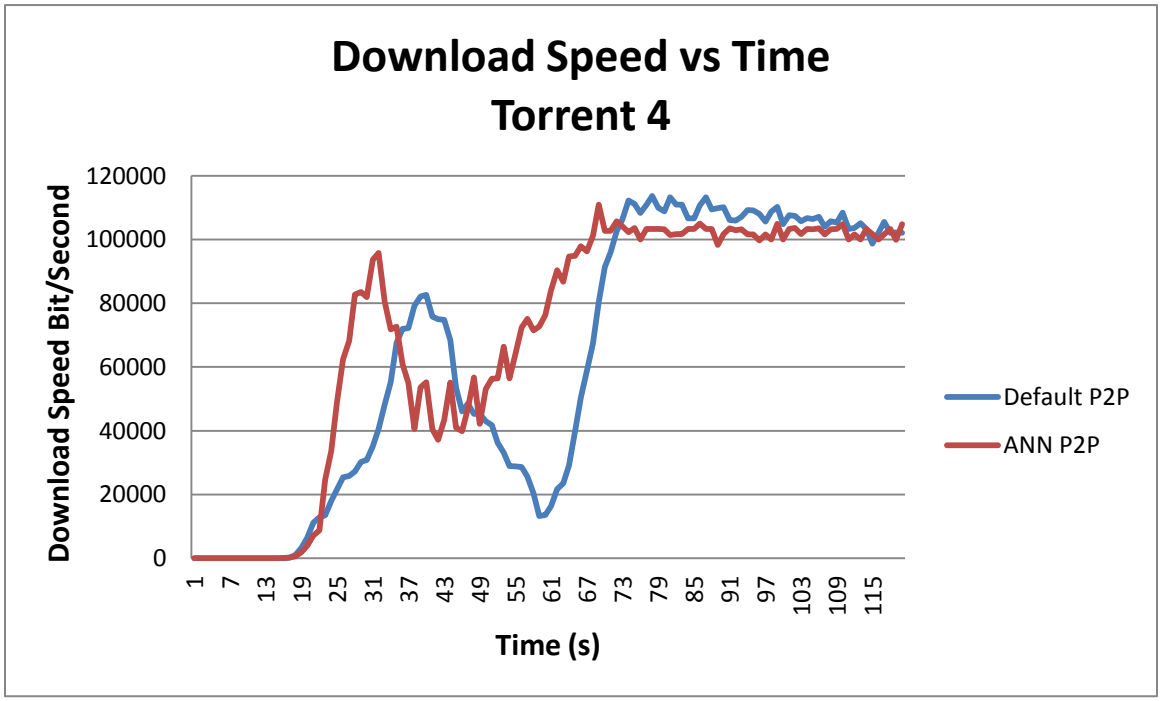
Zong, N. (2008). Survey and Problem Statement of P2P Streaming. *IETF PPSP* .

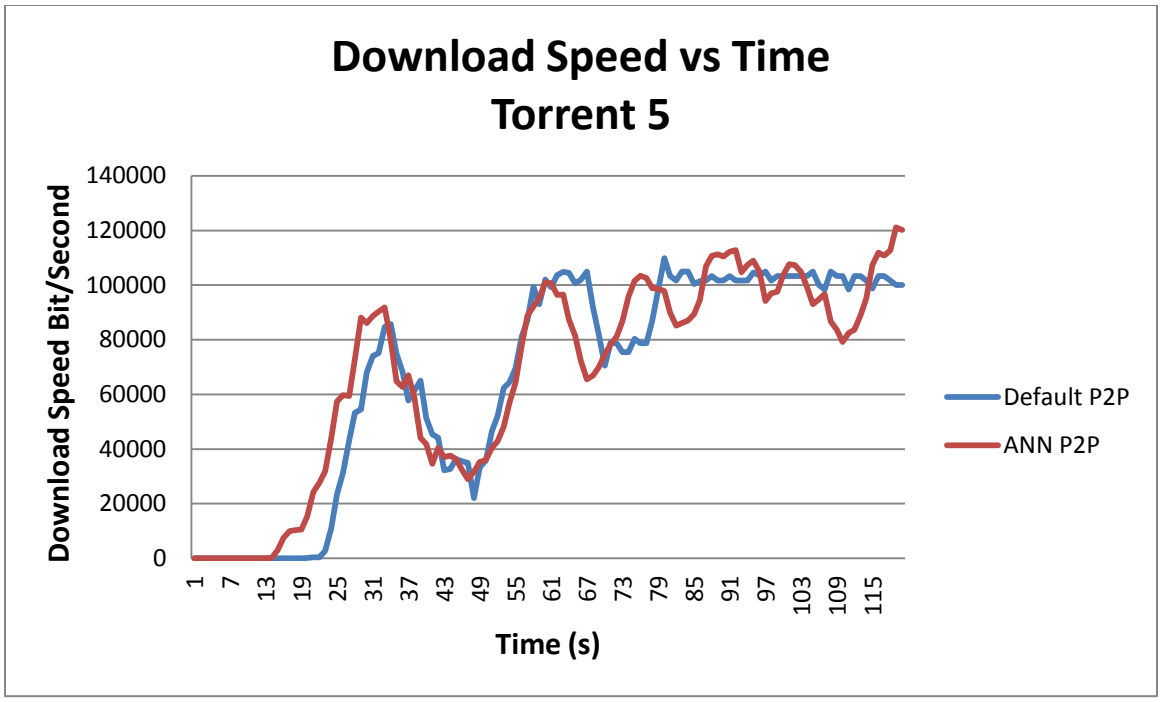
# APPENDIX A

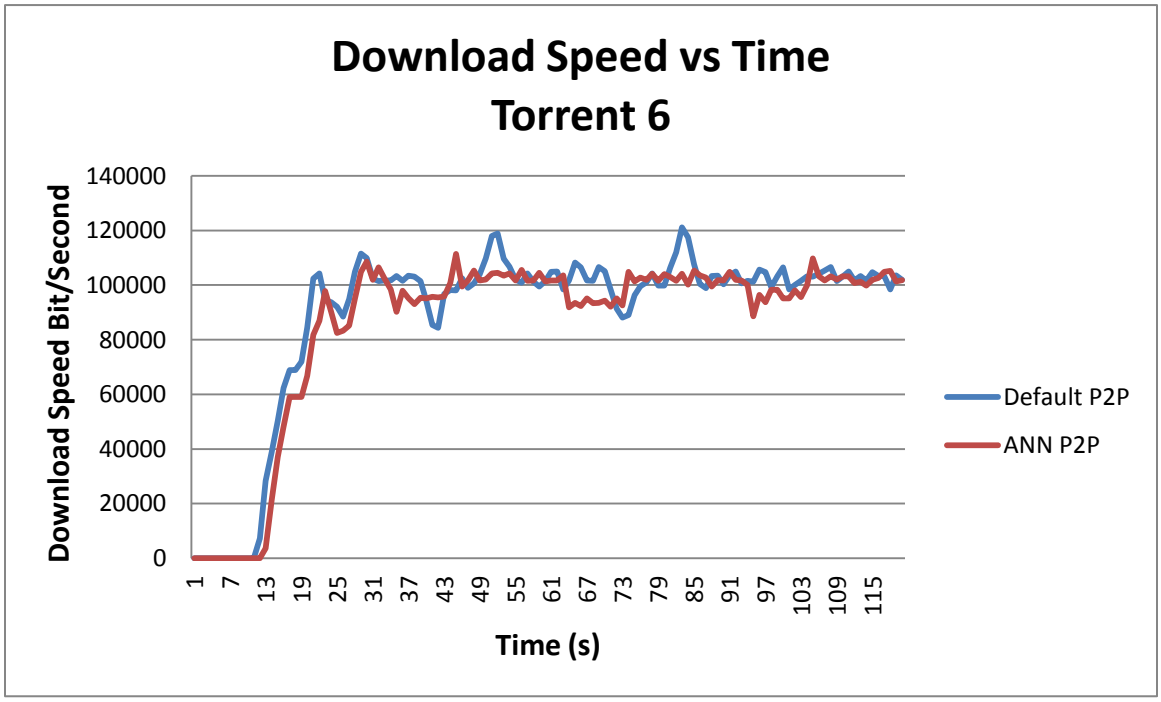

Download Speed vs Time
Torrent 1
Download Speed Bit/Second
120000
100000
80000
60000
40000
20000
0
1
7
13
19
25
31
37
43
49
55
61
67
73
79
85
91
97
103
109
115
Time (s)
Default P2P
ANN P2P

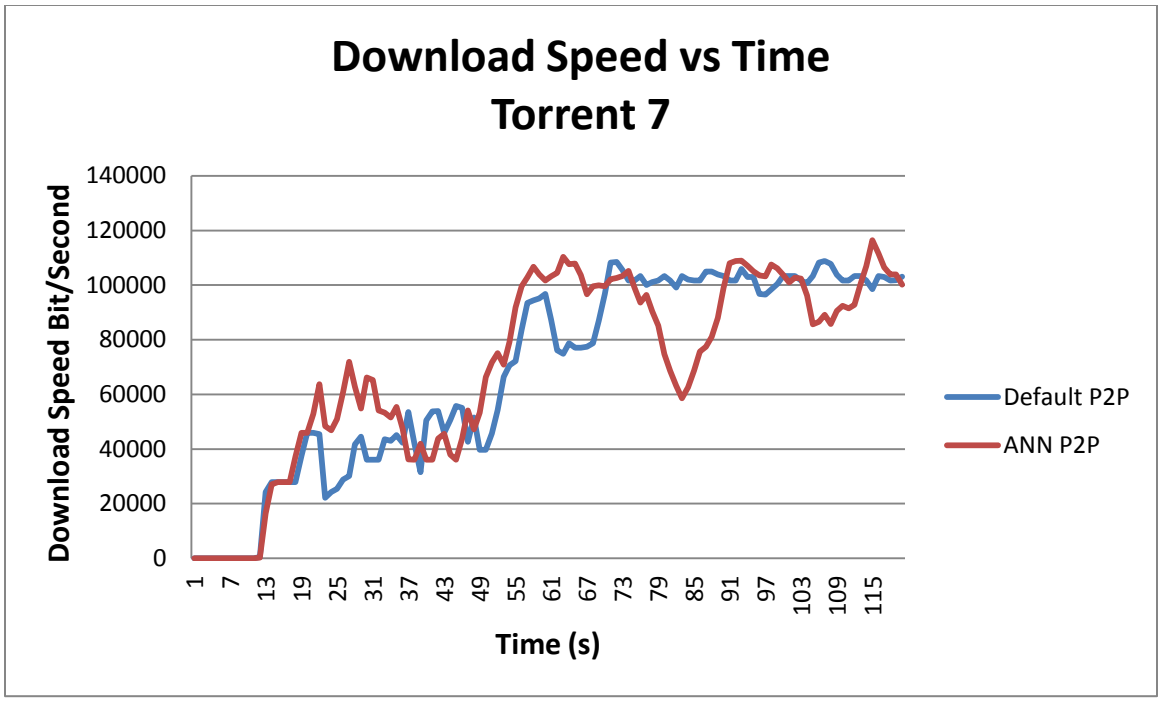

Download Speed vs Time
Torrent 2
Download Speed Bit/Second
140000
120000
100000
80000
60000
40000
20000
0
1
7
13
19
25
31
37
43
49
55
61
67
73
79
85
91
97
103
109
115
Time (s)
Default P2P
ANN P2P

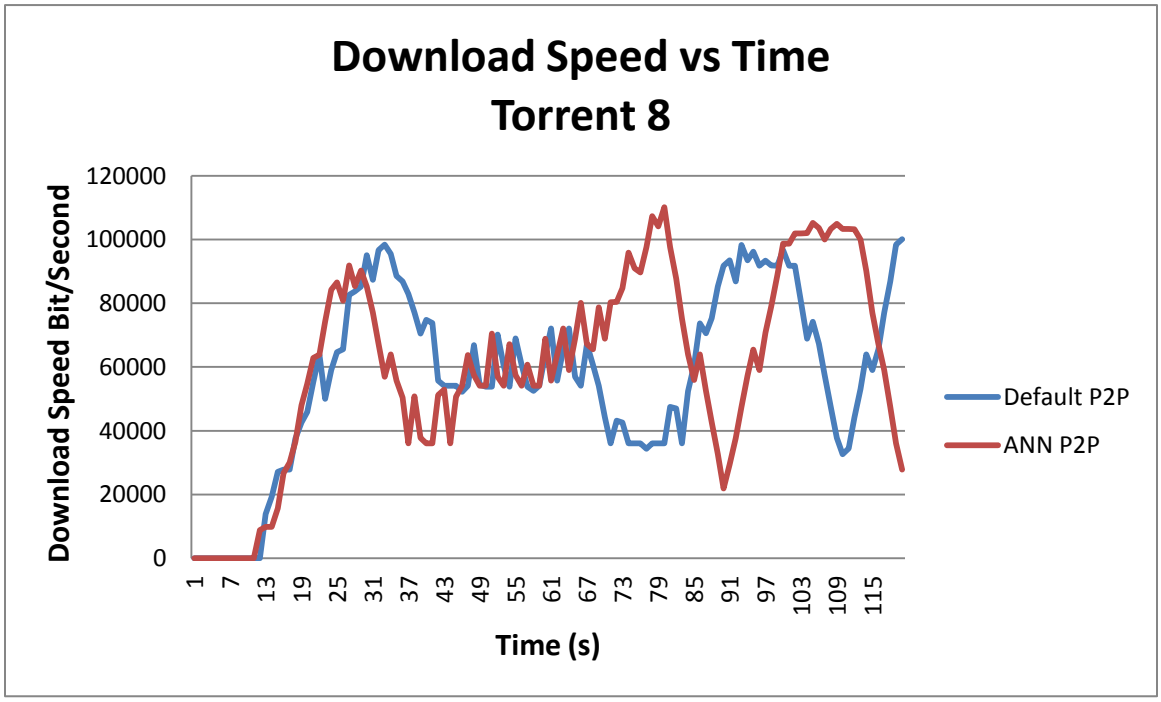

Download Speed vs Time
Torrent 3
Download Speed Bit/Second
140000
120000
100000
80000
60000
40000
20000
0
1
7
13
19
25
31
37
43
49
55
61
67
73
79
85
91
97
103
109
115
Time (s)
Default P2P
ANN P2P

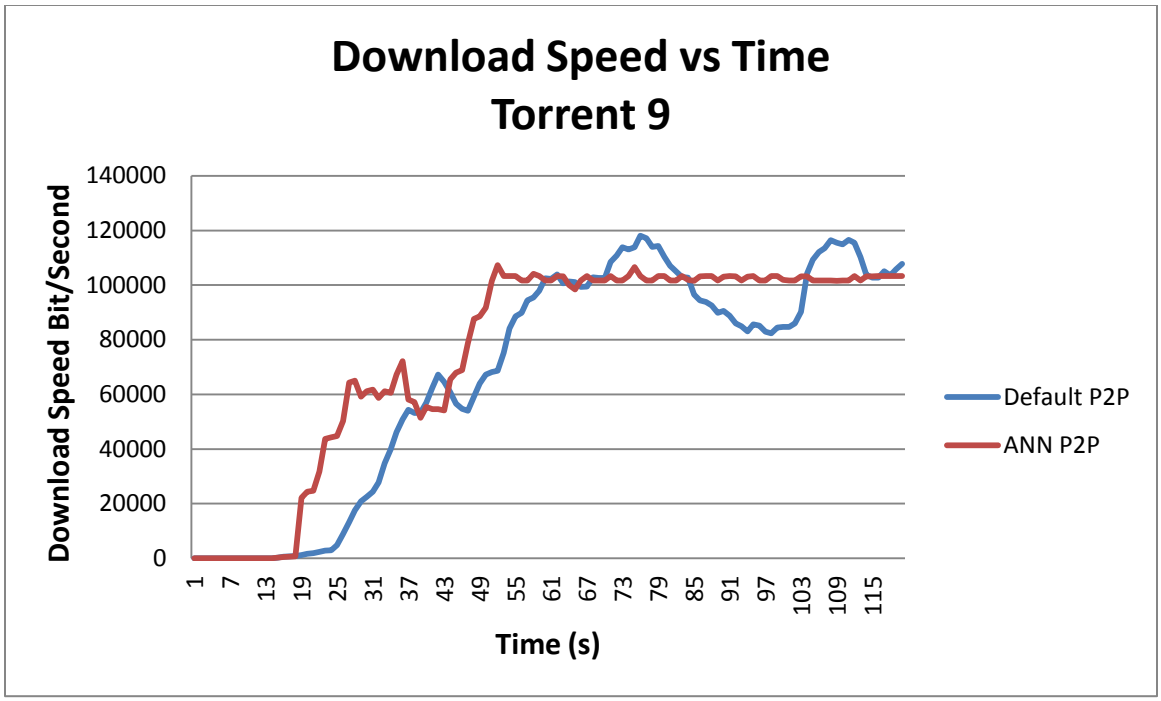

Download Speed vs Time
Torrent 4
Download Speed Bit/Second
120000
100000
80000
60000
40000
20000
0
1
7
13
19
25
31
37
43
49
55
61
67
73
79
85
91
97
103
109
115
Time (s)
Default P2P
ANN P2P

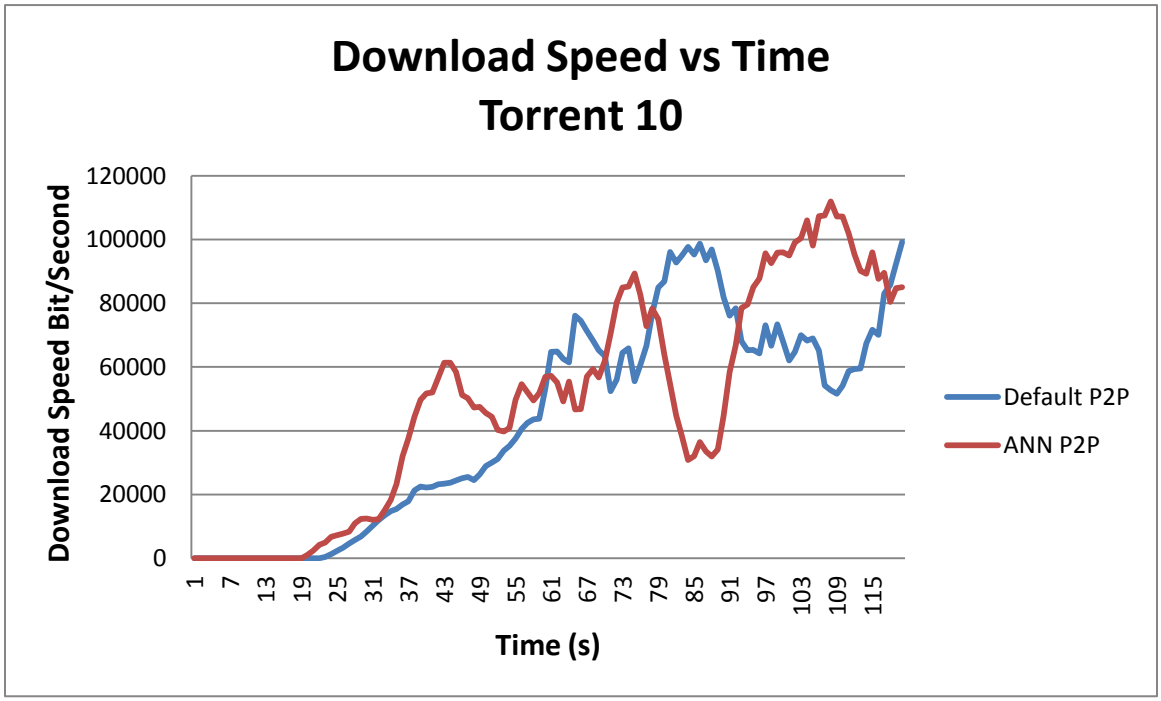

Download Speed vs Time
Torrent 5
Download Speed Bit/Second
140000
120000
100000
80000
60000
40000
20000
0
1
7
13
19
25
31
37
43
49
55
61
67
73
79
85
91
97
103
109
115
Time (s)
Default P2P
ANN P2P


Download Speed vs Time
Torrent 6
Download Speed Bit/Second
140000
120000
100000
80000
60000
40000
20000
0
1
7
13
19
25
31
37
43
49
55
61
67
73
79
85
91
97
103
109
115
Time (s)
Default P2P
ANN P2P


Download Speed vs Time
Torrent 7
Download Speed Bit/Second
140000
120000
100000
80000
60000
40000
20000
0
1
7
13
19
25
31
37
43
49
55
61
67
73
79
85
91
97
103
109
115
Time (s)
Default P2P
ANN P2P


Download Speed vs Time
Torrent 8
Download Speed Bit/Second
120000
100000
80000
60000
40000
20000
0
1
7
13
19
25
31
37
43
49
55
61
67
73
79
85
91
97
103
109
115
Time (s)
Default P2P
ANN P2P


Download Speed vs Time
Torrent 9
Download Speed Bit/Second
140000
120000
100000
80000
60000
40000
20000
0
1
7
13
19
25
31
37
43
49
55
61
67
73
79
85
91
97
103
109
115
Time (s)
Default P2P
ANN P2P


Download Speed vs Time
Torrent 10
Download Speed Bit/Second
120000
100000
80000
60000
40000
20000
0
1
7
13
19
25
31
37
43
49
55
61
67
73
79
85
91
97
103
109
115
Time (s)
Default P2P
ANN P2P

# APPENDIX B

## Retrieve Peer Input Source Code

```
Calendar tCurrentDateTime = Calendar.getInstance(); // get current time
String tIPAddress = tPeer.getIp();
String tRegionCode = null;
ConnectionStatus tConnectionStatus = ConnectionStatus.ping(tIPAddress);
int tCurrentTimeSection = hashTimeSectionOfWeek(tCurrentDateTime);
int tRegionHashCode = -1;
int tPredictedDownSpeed = -1;
int tRoundTripTime = -1;
int tUsedTTL = -1;
long tIPHashCode = -1;

if (isDatabaseReady && connectDatabase()) {
  try {
    dbResultSet = dbStatement.executeQuery("select * from `"
        + DEFAULT_SCHEMA + "`.`" + TRAINING_SET_TABLE + "` where "
        + "`idTrainingSet` >= " + initialTrainingSetID
        + " AND `TimeSection` = " + tCurrentTimeSection + " AND `IP` =
'"
        + tIPAddress + "';");
```

```
          if (dbResultSet.next() && dbResultSet.getString(1) != null) {
            tPredictedDownSpeed = dbResultSet.getInt("PredictedDownSpeed");
          }
        } catch (SQLException sqlEx) {
          log.log(LoggerChannel.LT_ERROR, "Error while trying to query
database. ", sqlEx);
        } finally {
          closeDatabase();
        }

        if (tPredictedDownSpeed != -1) {
          return tPredictedDownSpeed;
        }
      }

      tIPHashCode = hashIP(tIPAddress);
      if (tIPHashCode != -1) {

      } else {
        log.log(LoggerChannel.LT_WARNING, "IP format is not supported or
invalid. ");
        return -1;
      }
```

```
      if (!pif.getUtilities().getLocationProviders().isEmpty()) {
        try {
          tRegionCode = pif.getUtilities().getLocationProviders().get(0)
              .getISO3166CodeForIP(InetAddress.getByName(tIPAddress));
        } catch (UnknownHostException uhEx) {
          log.log(LoggerChannel.LT_ERROR, "Error while trying to query for
country code. ", uhEx);
          return -1;
        }
      } else {
        log.log(LoggerChannel.LT_WARNING, "There is no location provider. ");
        return -1;
      }

      tRegionHashCode = hashRegion(tRegionCode);
      if (tRegionHashCode != -1) {

      } else {
        log.log(LoggerChannel.LT_WARNING, "There is no location info of region
" + tRegionCode + ". ");
        return -1;
      }
```

```
if (tConnectionStatus != null) {
  tRoundTripTime = tConnectionStatus.getRoundTripTime();
  tUsedTTL = tConnectionStatus.getUsedTimeToLive();
}

double[] tInputValues = {
    (double) tCurrentTimeSection / (TIME_SECTIONS_PER_WEEK - 1),
    (double) tRegionHashCode / 50000.0,
    (double) tRoundTripTime / 500.0,
    (double) tUsedTTL / 255.0};

// process inputs
if (tInputValues[1] > 1.0) {
  tInputValues[1] = 1.0;
}

if (tInputValues[2] < 0.0 || tInputValues[2] > 1.0) {
  tInputValues[2] = 1.0;
}

if (tInputValues[3] > 1.0) {
  tInputValues[2] = 1.0;
}
```

## ANN Structure Source Code

```
private int numInput;

private int numHidden;

private int numOutput;


private double[] inputs;

private double[][] ihWeights; // input-to-hidden

private double[] ihSums;

private double[] ihBiases;

private double[] ihOutputs;


private double[][] hoWeights; // hidden-to-output

private double[] hoSums;

private double[] hoBiases;

private double[] outputs;


private double[] oGrads; // output gradients for back-propagation

private double[] hGrads; // hidden gradients for back-propagation


private double[][] ihPrevWeightsDelta; // for momentum with back-propagation
```

```
private double[] ihPrevBiasesDelta;


private double[][] hoPrevWeightsDelta;

private double[] hoPrevBiasesDelta;


private double learningRate; // controls the magnitude of the increase in the

// change in weights, found by trial and error.

private double momentum; // to discourage oscillation, found by trial and

// error
```

## Predict Result Source Code

```
 /**
  * Compute the output values for the neural network.
  *
  * @param xValues Array of the input values
  * @return Array of the output values
  * @throws IllegalArgumentException Input values is null or its length does
  * not match with the number of input nodes defined
  */
 public double[] ComputeOutputs(double[] xValues)
    throws IllegalArgumentException {
  if (xValues == null || xValues.length != numInput) {
    throw new IllegalArgumentException(
       "Input values is null or its length does not match with the number of input nodes defined. ");
  }

  for (int i = 0; i < numHidden; i++) {
    ihSums[i] = 0.0;
  }
```

```
for (int i = 0; i < numOutput; i++) {
  hoSums[i] = 0.0;
}

for (int i = 0; i < numInput; i++) {
  // copy x-values to inputs
  this.inputs[i] = xValues[i];
}

for (int j = 0; j < numHidden; j++) {
  // compute input-to-hidden weighted sums
  for (int i = 0; i < numInput; i++) {
    ihSums[j] += this.inputs[i] * ihWeights[i][j];
  }
}

for (int i = 0; i < numHidden; i++) {
  // add biases to input-to-hidden sums
  ihSums[i] += ihBiases[i];
}
```

```
for (int i = 0; i < numHidden; i++) {
  // determine input-to-hidden output
  ihOutputs[i] = SigmoidActivationFunction(ihSums[i]);
}

for (int j = 0; j < numOutput; j++) {
  // compute hidden-to-output weighted sums
  for (int i = 0; i < numHidden; i++) {
    hoSums[j] += ihOutputs[i] * hoWeights[i][j];
  }
}

for (int i = 0; i < numOutput; ++i) {
  // add biases to input-to-hidden sums
  hoSums[i] += hoBiases[i];
}

for (int i = 0; i < numOutput; ++i) {
  // determine hidden-to-output result
  this.outputs[i] = SigmoidActivationFunction(hoSums[i]);
}
```

```
  // could define a GetOutputs method instead
  double[] result = new double[numOutput];
  result = Arrays.copyOf(this.outputs, this.outputs.length);

  return result;
}
```

## Prioritize Peer Source Code

```
            public void eventOccurred(PeerManagerEvent event) {
              if (event.getType() == PeerManagerEvent.ET_PEER_ADDED) {
                // New peer added


                if (predict(event.getPeer()) > 15.0) {
                  event.getPeer().setPriorityConnection(true);
                }
              } else if (event.getType() == PeerManagerEvent.ET_PEER_REMOVED)
{
                log.log("Peer removed. " + (event.getPeer() == null ? "Peer is
null. " : "Peer data still available. "));
              }
            }
```