DOCTOR OF PHILOSOPHY

Enhanced Resource Discovery Mechanisms for Unstructured Peer-to-Peer Network Environments

Jamal, Azrul Amri

Award date:
2016

Awarding institution:
Bangor University

Enhanced Resource Discovery Mechanisms for Unstructured Peer-to-Peer Network Environments

Azrul Amri bin Jamal

Thesis submitted in partial fulfilment of the requirements for the

degree of PhD

School of Computer Science, Bangor University, United Kingdom

2016

Abstract

This study explores novel methods for resource discovery in unstructured peer-to-peer (P2P) networks. The objective of this study is to develop a lightweight resource discovery mechanism suitable to be used in unstructured P2P networks. Resource discovery techniques are examined and implemented in a simulator with high scalability in order to imitate real-life P2P environments. Simulated topology generator models are reviewed and compared, the most suitable topology generator model is then chosen to test the novel resource discovery techniques.

Resource discovery techniques in unstructured P2P networks usually rely on forwarding as many query messages as possible onto the network. Even though this approach was able to return many resources, the flooding of the network with query messages have an adverse effect on the network. Flooding the network has undesirable consequences such as degenerative performance of the network, waste of network resources, and network downtime. This study has developed alpha multipliers, a method of controlling query message forwarding to deal with the flooding effect of most resource discovery techniques in unstructured P2P networks. The combination of alpha multipliers and breadth-first search (BFS), α-BFS, was able to avoid the flooding effect that usually occurs with BFS. The α-BFS technique also increases the combined query efficiency compared to the original BFS.

Aside from improving a uninformed search technique such as the BFS, this study also examines the network communication cost of several informed resource discovery techniques. Several issues that arise in informed resource discovery techniques, such as false positive errors, and high network communication costs for queries to update search results are discussed. This detailed analysis forms the basis of a lightweight resource discovery mechanism (LBRDM) that reduces the network communication cost by reducing the number of backward updates inside the network when utilising the blackboard resource discovery mechanism (BRDM). Simulations of BRDM and LBRDM show that the lightweight version can also return an almost identical combined query efficiency than the BRDM.

The solution to control query message forwarding in α-BFS, and the removal

of unnecessary exchange of information in LBRDM open a new perspective on simplifying resource discovery techniques. These approaches can be implemented on other techniques to improve the performance of resource discovery.

Acknowledgments

I would like to express the deepest appreciation to my committee chair, Dr. William John Teahan, who has been enthusiastically guiding and encouraging me in this PhD research. Without his guidance and persistent help, this dissertation would not have been possible.

I would like to thank the supervisory committee members for my study, Professor Ludmila Kuncheva and Dr. Ik Soo Lim who has helped me to keep on track with the research. They've guided me on how to stay focus to the research. I would never forget the heartfelt motivation that they have given me.

A big thank you to Malaysian Ministry of Higher Education (MoHE) and Universiti Sultan Zainal Abidin (UniSZA) for sponsoring my studies and expenses in Bangor University. I will not be able to further my studies in United Kingdom without their financial support.

I am thankful to my colleagues for their support, encouragement, and for creating a fun work environment. I would also like to thank my family for being very supportive and understanding towards my PhD research. They are my strength and my biggest motivation.

Nomenclature

APS Adaptive Probabilistic Search

BFS Breadth-First Search

BRDM Blackboard Resource Discovery Mechanism

CAN Content Addressable Network

Chord A protocol and algorithm for a P2P DHT

DFS Depth-First Search

DHT Distributed Hash Table

Int-BFS Intelligent Breadth-First Search

LBRDM Lightweight Blackboard Resource Discovery Mechanism

P2P Peer-to-Peer

PAST Distributed file system layered on top of Pastry

RRW Restricted Random Walk

RW Random Walk

α-BFS Alpha Breadth-First Search

Contents

List of Figures

List of Tables

List of Algorithms

1 Introduction

Chapter Summary

This chapter contains the introduction to the whole study, enhanced resource discovery mechanism for unstructured peer-to-peer network environments. Motivation for the research is also presented in this chapter. Three research objectives have been identified and all the study, research, experiments have been performed in order to fulfil these objectives. The chapter ends with a list of contributions towards published literature that came out from this study.

1.1 Introduction

Resource discovery is one of the most important part in any resource sharing systems [55, 78]. Peer-to-Peer (P2P) is currently the main means of resource sharing in the Internet. Resource discovery plays a more important role in P2P, where the resource is widespread all over the world.

P2P file sharing is often considered as a platform for piracy. In order to fight piracy, creative industries such as the film, music, and software companies have been fighting with P2P establishments. Companies put their effort in polluting the file sharing community in order to discourage piracy [51, 84]. The pollution is done by seeding files with different content to confuse the file identification [54]. Thus, resource discovery does not only focus on finding any resource, but it must be extended to find quality resources [8]. This new approach is known as quality-driven resource discovery.

1.2 Motivation

BitTorrent, is a well-known P2P resource sharing network [20]. The network spans across the world, and it is believed that it is the protocol that uses the most bandwidth in the world. However, BitTorrent is still using Breadth-First Search (BFS) [10] for its

search. BFS consumes a lot of bandwidth, thus network administrators usually block all the resource discovery for the BitTorrent. The research in this dissertation was done to find a better way for resource discovery, where the cost of resource discovery can be reduced marginally without sacrificing much of the search results.

Resource discovery techniques can be categorised as either blind search or knowledge based [99]. Blind, or uninformed search resource discovery techniques are usually lightweight. Examples of uninformed search resource discovery techniques are such as those proposed by Antonini et.al. [9], and Erdil [24]. A uninformed search resource discovery usually has a clever way to manipulate the query message propagation method in order to keep the technique as lightweight as possible. These techniques however rarely show any learning effect over time.

The knowledge based resource discovery techniques such as those proposed by Said & Kojima [94], Al-Dmour & Teahan [4, 5], and Hasanzadeh & Meybodi [36] have a learning effect. The techniques in knowledge based resource discovery will return better search results the longer the techniques are running in the system. The learning based techniques usually require some computing power and/or memory from each node that the query messages visit. A node with less computing resources might struggle to run the search queries over time. A node with a small amount of memory available might find difficulty in keeping track of all the knowledge that the node has gained over time.

Observing the advantages and disadvantages of both the lightweight and learning based resource discovery, it is clear that the field of resource discovery needs to have a technique that has a learning effect but does not tax the network and nodes too much. The idea is to reduce the network or computation cost of a learning based resource discovery technique.

1.3 Thesis Objectives

There are three main objectives of this research.

- To design a lightweight resource discovery technique suitable to be used in unstructured P2P networks.
- To implement the resource discovery techniques onto a P2P simulator.
- To find out the effectiveness of the new resource discovery technique by comparing it with existing approaches.

1.4 Methodology of Research

There are three types of methodologies could be used for research in P2P networks. They are by using mathematical approach, experimenting on actual P2P system, and by conducting computer simulations. Each methodology has their own advantages and disadvantages.

The mathematical approach is usually the first approach being used to develop P2P protocols. Nevertheless, mathematical approach usually relies on assumptions and network simplifications. These are needed to be done, because networks are usually complex, and simplifying the network might omit several important information. Thus making the results obtained using the mathematical approach non-usable in real life P2P network.

The second alternative for P2P research is by implementing the proposed protocols onto actual P2P network. Implementation on real life P2P network requires a lot of computer nodes to really reflect P2P network environments. Having computers turned on to do experiments consume a lot of time, energy, and money. Experiments using limited amount of P2P nodes might not indicate an actual implementation on P2P networks, that usually large in scale. Furthermore, experiments on actual P2P systems might expose the network to security attacks and in some cases, it is possible that the experiments break the network that is supposed to serve other users than the nodes for the experiments alone.

Therefore, P2P network simulations have been selected as the methodology of research. P2P nodes will be generated using P2P network simulator, and experiments are conducted upon it. Nevertheless, there are some limitations for experiments on simulators. The limitation of this methodology are discussed in the following Section 1.5. Details regarding the simulators are discussed further in Section 3.3.

1.5 Scope of Research

There are advantages and disadvantages of methodologies used in P2P network researches. The methodology of simulating the P2P network has been selected to be used in this research. Experiments on P2P network using simulators are not affected to the non-realistic results obtained by mathematical models where they requires researchers to simplify the network being experimented on. The problem with time, energy, and money consumption of experiments done on actual P2P system are also nonexistent when doing the experiments on simulators.

Nonetheless, P2P experiments using simulations have their own disadvantages that requires some trade-offs being made. Conducting experiments on actual P2P system will require the P2P nodes to communicate each other using actual computer networks. By using actual networks, the experiments need to utilise all the seven layers of the Open Systems Interconnection (OSI) model. Thus, experiments conducted using actual P2P system will show how a real network would react when utilising the protocol that being experimented.

Experiments using simulations do not imitate all of the seven OSI layers. This is because a simulator would consume a large number of memory resource in order to replicate all the seven layers of the OSI model. As similar to other P2P simulators, the simulator that being used in this research, PeerSim simulates the most important layer in the OSI model for node to node connection, the network layer.

Another limitation of this research is the memory capacity of the computer running

the simulations. Simulator with the best scalability function was chosen in order to maximise the number of P2P nodes being simulated. In order to achieve high scalability, the simulations were done using cycle-driven simulation engine instead of event-based simulation engine. The event-driven simulation engine process each query based on events occurring on P2P node. The event-driven simulation engine is closer to actual network, however the engine uses a very large amount of memory resources, and consumes a lot of time.

1.6 Thesis Outline

This dissertation is divided into eight chapters. Chapter 1 describes previous work and gives motivation and objectives for the work performed in this dissertation. The literature review is shown in Chapter 2. Chapter 3 discusses the implementation of resource discovery mechanisms on PeerSim, a simulator for peer-to-peer networks. The simulation of unstructured P2P networks and the selection of topology generator model are examined in Chapter 4.

Chapter 5 explores a method of reducing query message forwarding in breadth-first search by implementing alpha multipliers, α-BFS. It is then followed by development of lightweight blackboard resource discovery mechanism (LBRDM) in Chapter 6. The experimental results for α-BFS and LBRDM is presented in Chapter 7. The final chapter concludes the thesis with collective discussion and conclusions of the whole work performed in this dissertation.

1.7 Contributions

Contributions of this research can be divided into two main categories: Contributions towards computer network engineering; and contributions to published literature. Contributions towards computer network engineering section lists the advancement in the field that can be contributed by implementing the developed techniques of this research

onto the computer network engineering field. The contributions to published literature section lists down all the findings from this research that being published to the academia world.

1.7.1 Contributions Towards Computer Network Engineering

This research focuses on reducing the weight of search techniques that rely on query message replication to find resources (eg. Breadth-First Search (BFS), and Blackboard Resource Discovery Mechanism (BRDM)). BFS is a search technique that is still in use nowadays in P2P network applications. BRDM is a technique that is in use in grid computing platform called ParCop for resource discovery. These search techniques were studied and implemented on Power law simulated network so that it will closely reflect a real life network condition. This research contributes to the computer network engineering field by reducing the amount of network usage for applications and protocols that use the BFS and the BRDM as their resource discovery technique.

1.7.2 Contributions to Published Literature

The list of contributions in terms of journal publications are provided as follows.

- A.A. Jamal, W.S. Wan Awang, M.F. Abdul Kadir, A. Abdul Aziz, and W.J. Teahan. Implementation of Resource Discovery Mechanisms on PeerSim: Enabling up to One Million Nodes P2P Simulation. *International Journal of Digital Information and Wireless Communications (IJDIWC)*, 5(1): 14-20, SDIWC, ISSN: 2225-658X, 2015.

- A.A. Jamal and W.J. Teahan, Alpha Multipliers Breadth-First Search Technique for Resource Discovery in Unstructured Peer-to-Peer Networks. *ARPN Journal of Engineering and Applied Sciences (JEAS)*, November 2016. (under review)

- A.A. Jamal and W.J. Teahan, Lightweight Blackboard Resource Discovery Mechanism for Resource Discovery in Unstructured Peer-to-Peer Network. *Journal of Information and Communication Technology*, 2017. (planned)

Contribution in terms of proceedings publication is as follows.

- A.A. Jamal, W.S. Wan Awang, M.F. Abdul Kadir, A. Abdul Aziz, and W.J. Teahan. Implementation of Resource Discovery Mechanisms onto PeerSim. *In The 3rd International Conference on Informatics & Applications (ICIA2014)*, October 2014.

2 Literature Review

Chapter Summary

This chapter contains the literature review of P2P networks and resource discovery. This chapter starts with an introduction and some history regarding the Internet and its usage, the emergence of P2P network and grid computing, followed with the resource discovery techniques in grid computing. Overview of the P2P networks, and the categorisation of the P2P networks are also discussed. This chapter continues with a review of resource discovery mechanisms, where several types of resource discovery mechanisms are discussed and explained.

Summary of Each Section

Introduction	:	Introduction of the chapter
Peer-to-Peer	:	Brief history of P2P network. Categorising P2P networks according to their utilisation & technique, overlay networks, and network topology
Resource Discovery	:	Resource discovery techniques are categorised into two categories and presented. Resource discovery techniques in unstructured P2P network are discussed at the end of the section.

2.1 Introduction

Early computer networks were a point-to-point connection between computers. As the networks grew bigger, more and more computers became connected and not long after the Internet was born. In the initial stage, the Internet was usually built from a number of client-server based connection, where faster computers acted as server and serving files, information and resources to the client computer which usually consisted of a less

powerful computer.

The client-server Internet model was very useful in the early stages of the Internet. Nonetheless, this model has several problems such as network traffic blocking or network bottleneck, server availability problems, and single point of failure. Computer technologies have been keeping on advancing following the Moore's law. Moore's law is the observation that the number of transistors in integrated circuits doubles every year. The observation has been revised so that the number that doubles is now every other year rather than every year [63].

The shrinkage of the transistors inside integrated circuits enable users to have more computational power, more energy efficient machines with the same amount of money [41]. The server class machine of today will be available and obtainable by normal users several years after [72]. Therefore, the Internet's client-server architecture is getting less popular as more Internet users are able to share resources online, a trait that previously only a server can do.

The motivation to share resources to the Internet has contributed to the birth of peer-to-peer (P2P) network [80]. All computers (usually referred to as nodes), that are connected to the P2P network have an equal role. All nodes act as a client and a server at the same time. There is a large amount of information available in a P2P network. Nevertheless, this information is usually distributed across the network and is quite troublesome to find.

Together with the advancement of P2P networks, heterogeneous resources can be distributed geographically. The need for a computational system to be in one place has been eliminated. Thus a new type of computational architecture has emerged, the grid computing [26, 70]. Grid computing utilises the robustness of the Internet and can distribute computing tasks to multiple heterogeneous computers across the Internet. The whole distributed grid system can be viewed as one supercomputer [26]. Having resources distributed across the network requires the system to keep track of the

resources available, and search for new ones. Therefore, the need to develop techniques to find the resources, "resource discovery", became more important [79].

Resource discovery techniques are utilised in distributed computing and resource sharing system, like grids, computational resource sharing, distributed hosting and utility computing [55, 78]. In the early stages of P2P, researchers have been using old search methods as resource discovery techniques. Among the earliest techniques are Breadth-First Search (BFS) [10], Depth-First Search (DFS) [102]. These techniques have been improved over time, and currently many different enhanced techniques have been created in order to find the resource throughout the network.

The following section will discuss the history and basis for P2P networks followed by the different types of resource discovery techniques.

2.2 Peer-to-Peer Networks

The urge to share resources online to some extent emerged at the same time as the development of P2P networks. The availability of affordable computing power also helped the emergence of P2P. Internet users wished to share their resources and in the earliest days of P2P, one network application in particular, Napster [77], had managed to successfully implement a P2P network worldwide. Using this application, Internet users could share their music from their own computer to the whole world [80]. In spite of that, encountering problems with copyright law, Napster did not last long.

After the shutdown of Napster, other resource sharing software based on P2P networks emerged. Applications such as eMule [23], Gnutella [31], Gnutella2 [32], BitTorrent [15] used P2P networks for content distribution. P2P video and audio streaming such as PPLive [87], and Skype [100] also gained rapid growth [78]. Even though it is said that these applications uses a P2P network environment, each P2P network is not identical. They use different protocols and structures that best suit the network requirements in terms of speed or availability according to each software.

2.2.1 P2P Categorisation

There are several ways to classify a P2P network. Researchers classify P2P networks based on its utilisation, techniques, structure of overlay network, and topology. Navimipour & Milani [78] categorised P2P networks into two different overlay networks, centralised, and decentralised. Navimipour et.al [79] classified the P2P networks into four different topologies, that is, unstructured, structured, super-peer, and hybrid.

2.2.1.1 Based on Utilisation and Techniques The most basic categorisation for P2P networks are based on their utilisations. The first well known P2P applications was for file sharing, Napster [77]. However, as has been stated, this has now been shut down by the judiciary because of copyright infringement lawsuit. Napster was then followed by several other file sharing P2P applications such as Gnutella [31], Gnutella2 [32], and KaZaA [49].With file sharing being the first application of P2P networks, some people might have a misconception that all P2P networks are only for file sharing. There are also other utilisations of P2P networks beyond mere file-sharing [1].

Multimedia is another intelligent utilisation of P2P networks. Examples of P2P multimedia applications are P2PTV [27], and Biernacki et. al. [14], that are protocols to share TV broadcasting over the Internet. Skype [100] and Spotify [53] stream videos and audios to the peers using their streaming servers and peer-to-peer network.

P2P networks are also being used just for research such as Chord [101], PAST [92], P-Grid [1], & Coopnet [81]. Another good example of P2P utilisation are for creating currency, Bitcoin [76]. Netsukuku, an alternative from the Internet is also an example of P2P utilisation that is not file sharing [27].

Categorisation of P2P networks according to their utilisation might be the easiest way to classify P2P. Nevertheless, in order to improve the P2P networks, the focus can not just be on their utilisation because each utilisation might use different types of network overlay and/or network topology. In addition, a fair categorisation is difficult

to achieve because a large portion of P2P networks utilisation are for file sharing. Categorisation of P2P networks based on its overlay network and network topology are discussed in following paragraphs.

2.2.1.2 Categorisation Based on Overlay Network Overlay network can be viewed as a computer network that runs above another computer network. The overlay network usually is logical or virtual and the underlying network are considered as physical network. The overlay network usually are easier to see and understand by the network users compared to the physical network. An overlay network is a necessity in P2P networks because in a P2P network, peers can join and leave at any time. If the network uses exactly the same topology as the physical topology, it can affect the reliability and availability of the P2P networks [60].

There are two types of overlay network. Figure 2.1 shows the two types as decentralised overlay topology and centralised overlay topology [61]. In a glance, the centralised overlay network looks like a client-server network where there is a centralised main computer that controls the information of the whole network. A decentralised overlay topology, in contrast, there is no central computer that act as the main server of the whole network.

The type of underlying network topology does not dictate what type of the overlay network topology. Therefore, there can be multiple combinations of both overlay and underlying network.

2.2.1.3 Categorisation Based on Network Topology P2P can be categorised based on their topologies. Navimipour et. al. [79] have grouped the P2P networks based on four topologies, namely, unstructured, structured, super peer, and hybrid. At the same time, P2P network overlay topology can be either centralised or decentralised [78]. Combination of both categorisation techniques will generate 8 types of P2P networks in

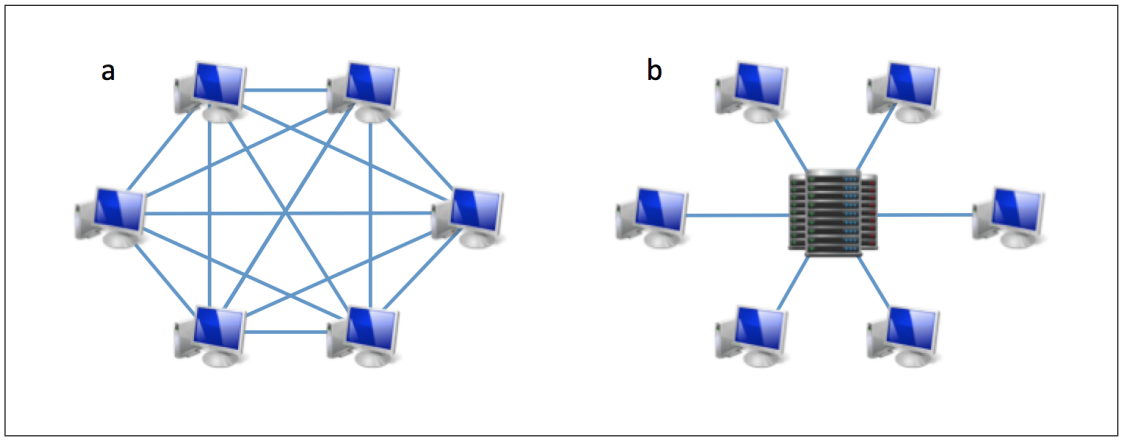

Figure 2.1: (a) Decentralised Overlay Topology. (b) Centralised Overlay Topology.

total. A P2P network can either be an unstructured, structured, super peer, or hybrid topology with either centralised or decentralised overlay network [78, 79].

Unstructured P2P topology can be generated when all the nodes connected with each other randomly. Each node in an unstructured P2P network maintains its anonymity because there is no information regarding the location of nodes. Structured P2P networks are organised into a specific topology. Example of P2P networks with an unstructured P2P network are Napster [77], Gnutella [31].

In contrast to the unstructured networks, the location of all of the nodes in a structured P2P network are stored in a distributed storage location. Usually the locations are stored using distributed hash table (DHT), thus the nodes in a structured P2P network do not maintain their anonymity. Examples of P2P networks with a structured topology are Chord [101], CAN [91]. In a super peer P2P network topology, a node can either be an ordinary peer or a super peer. An ordinary peer and super peer both act as a server and a client. The only difference is that a super peer usually has a lot of neighbours compared to an ordinary peer [78]. Examples of P2P networks that use a super peer topology are KaZaA [49], Gnutella2 [32].

All three categories listed above have their own advantages and disadvantages. A

Table 2.1: Advantages and Disadvantages of P2P Topologies.

Topologies	Advantages	Disadvantages
Unstructured	Offers suitable reliability, robustness, dynamicity, scalability, and removes false positive errors	Suffers from the network-wide broadcast storm problem and low security
Structured	Offers suitable response time, reliability, and load balancing	Suffers from high traffic, low security, and dynamicity
Super Peer	Offers suitable scalability, load balancing, reliability, and dynamicity	Suffers from complex procedure, low robustness, noticeable response time, and single point of failure in each super-peers
Hybrid	Almost offers high reliability, and scalability	Suffers from low load balancing, security, and robustness as well as high overhead

hybrid P2P network topology can also be generated by combining multiple topologies. The hybrid topology will usually use the advantages of its primary P2P topology that it takes. Examples of P2P networks that uses the hybrid topology are Loo et. al. [59], Papadakis et. al. [82]. Table 2.1 shows the advantages and disadvantages of the P2P networks topology. Table 2.2 are some examples of P2P techniques that are being used for various network topologies [78].

2.3 Resource Discovery

Resource discovery is an act of discovering resources available for use. In this research, resource discovery encompasses the discovering resources that are either computational resources or data and storage resources. These resources are needed to be discovered for the use of distributed computing such as in grid computing because resources can join and leave the grid at any given time.

The term "resource" itself, can be used to represent a lot of things. There are times when the term "resource discovery" brings a different meaning to the academic world.

Table 2.2: P2P Techniques According to Their Topologies.

Topologies	P2P Techniques
Unstructured	Napster [77], Gnutella [31], Qu et. al. [88], Mashayekhi & Habibi [66], Shojafar et. al. [96], LARD [104], IAPS [96], DHMCF [69].
Structured	Chord [101], CAN [91], Schmidt & Parashar [95], Merz & Gorunova [68], Giordanelli et. al. [29], Lee et. al. [56], Zhygmanovskyi & Yoshida [113], Si et. al. [97].
Super Peer	KaZaA [49], Gnutella2 [32], Haasn [33], HPRDG [6], Zhang et. al. [112], SPS [58].
Hybrid	Loo et. al. [59], Papadakis et. al. [82], Yang & Yang [109], GAB [111], HybridFlood [12].

Fuhr, for example, noted that the resource discovery needed to be implemented to access and gather information from a vast number of digital libraries [28]. This research discussed various issues regarding information diversity, document formats, indexing methods, database schemas and protocols.

Macgregor & McCulloch uses the term resource discovery to describe the process of finding information in online tags, such as a website's tag cloud [62]. Heckner et. al. explores social tagging, and how to analyse the user keywords for different digital resource type [37]. The study focuses on comparative analysis of social media tags, such as from del.icio.us, flickr, and Youtube. Heckner et. al. [37], Macgregor & McCulloch [62], and Fuhr [28] considered "information" as a resource, therefore the term resource discovery brings another meaning to these researchers. In spite of that, the act of discovering resources in a form of information, can be classified as "information retrieval", "knowledge discovery", or "data mining" [25, 34].

2.3.1 Resource Discovery Categorisation

Discovering computational resources or storage resources are different from finding information, such in "knowledge discovery". Instead of finding information inside a node,

resource discovery of computational or storage resources for grid computing concerns whether the node has a resource or whether it does not have the resource.

Singh et. al. [99] classified resource discovery techniques into two main categories, blind search and knowledge based. Russell and Norvig [93], have categorised search techniques into various categories, such as informed and uninformed searches, adversarial, optimisation, and evolutionary searches. Concerning the first two categories, Singh et. al. [99] named them as blind search and knowledge based. The following sections discusses these search categories.

2.3.1.1 Uninformed (blind) search resource discovery Uninformed or blind search is a category of resource discovery techniques, where the walkers that bring queries do not have any information regarding the network throughout the whole search. The queries will also not be processing any information of the network for their future searches. The query message sent using uninformed search is lightweight to the network and the nodes. The query message can be easily replicated or cloned, and distributed according to its forwarding rules behind the resource discovery techniques that are being used.

Despite the query messages being lightweight, the techniques in this category usually rely on the number of query messages in order to find the resources that they want to find. There are occurrences where the techniques send the same query messages to a node in the P2P network multiple times. This condition is called flooding the network. No matter how lightweight the query message, the number of queries will consume a lot of resources along the way.

Uninformed search resource discovery techniques are usually used in mobile networks, or ad-hoc networks where the computational or storage resources on each nodes are small or limited. Among search techniques that fall into this category are: breadth-first search (BFS); uniform cost; depth-first search; depth-limited; iterative deepening;

and bi-directional [93, 103]. Uninformed search characteristics will be further discussed in Section 2.3.3 and 3.2.

2.3.1.2 Informed (knowledge-based) search resource discovery Informed or knowledge-based search is a category of resource discovery techniques where the techniques utilise some heuristic approach towards finding the resource. Resource discovery techniques that fall within the informed search category take into account the information of the network that the techniques have been working on. Techniques within the informed search category usually generate some information for their own technique's future references.

The query messages of knowledge based resource discovery techniques often is not lightweight, because it contains a lot of information regarding the network. Upon arriving at a new node, several checks need to be done by the node, and sometimes the nodes need to be compared with the information that they already have. After the comparison, the information that the query messages initially brought is updated with the information that the current node has. Further processing by the node needs to be done in order to decide on where to forward the query messages to.

Informed search resource discovery techniques would usually require some computation and/or storage resources from the nodes. Together with good computation and storage management and intelligent decision making, informed search resource discovery techniques can reduce their consumption of network communication resources and subsequently the computational resources. The longer that the informed search resource discovery techniques run on a P2P network, the more they learn about the network. This will lead to a better resource discovery and this phenomena can be considered to be a "learning effect".

Among resource discovery techniques that fall into this category are: best-first search; greedy search; and A* search [103]. All search techniques in this category

utilise some kind of heuristic approach in order to select between all the alternatives that they have. Informed search strategies use problem-specific knowledge to their advantages according to the goal of the search. For example, the greedy search expands nodes closest to the goal, and the A* search expands nodes on the least-cost solution path [103].

The blackboard resource discovery mechanism (BRDM) [4, 5], uses a different kind of informed search. Rather than using a heuristic in the traditional sense, instead, the BRDM utilises the knowledge it has obtained through its past behaviour. This reference to knowledge obtained from past behaviour are also present in other resource discovery techniques, such as the intelligent-BFS (Int-BFS) [46] and adaptive probabilistic search (APS) [106].

Informed search resource discovery techniques are usually used in large and complex interconnected networks, where the nodes in the network have many computation and storage resources to spare for the maintaining of the network. Informed search will be further discussed in Section 2.3.3 and 3.2.

2.3.2 Evaluating Resource Discovery Effectiveness

Scientific researches need to be valid and reproducible by other members of the research community. A survey has been done on P2P research papers to find the type of simulators and evaluation technique used in the researches [74]. Out of 287 papers on P2P networking subjects, 146 papers do not involved any simulation at all. This research rely on mathematical calculations in order to proof the P2P algorithm being developed.

71 papers used simulators but did not specify the simulator being used. 43 papers developed their own simulators. The remaining 27 papers specified the simulators being used, such as, NS-2 (8), Chord (SFS) (7), JavaSim (2), and PeerSim (2) (refer Figure 2.3.2). Among these simulators, PeerSim is the only one that is still being used by

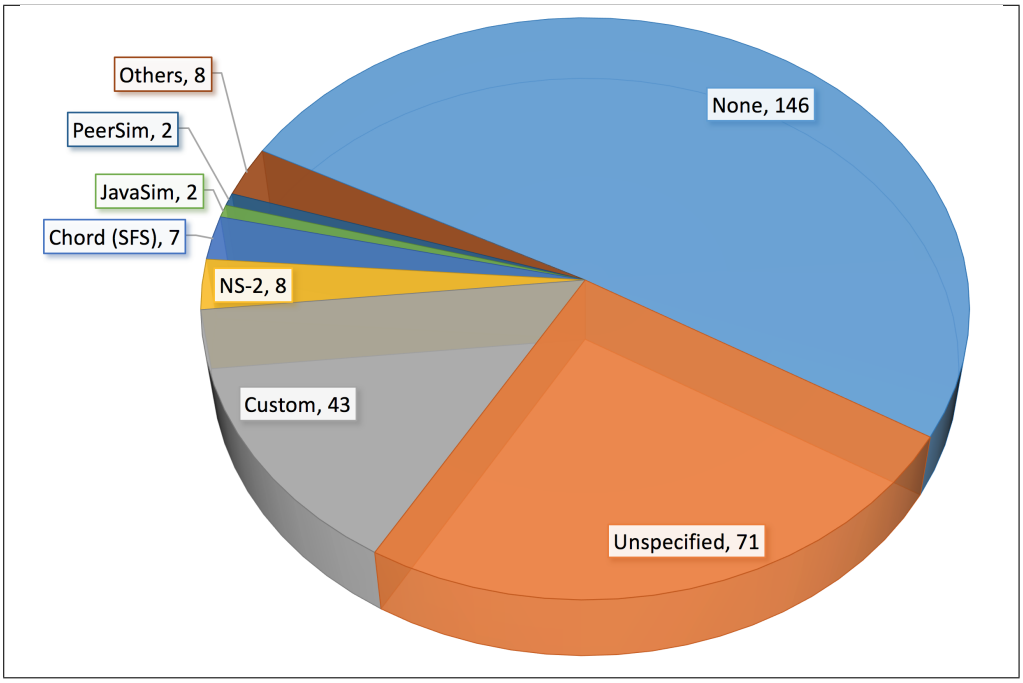

Figure 2.2: Quantitative Survey on the Use of P2P Simulators [74].

researchers for P2P simulations [22]. Resource discovery techniques' effectiveness are calculated by number of nodes being visited by the query message.

There are several criteria that can evaluate the effectiveness of a resource discovery techniques, such as: space complexity; time complexity; completeness; optimality; and number of successful searches. Space complexity is a way to measure the amount of memory used at any point in the algorithm. Time complexity is a measurement of time needed for the algorithm to be completed. Both space and time complexity is usually presented as the big O notation [83, 93].

Completeness in resource discovery techniques is whether an algorithm was able to find all resources in the network. If there were some solutions or targets that were not solved or found, an algorithm can be called as incomplete. Optimality is a measurement of algorithms to return the best result obtainable under specific limitations. The number of successful searches as the name suggests is the amount of successful searches that

the resource discovery techniques were able to find. The bigger the number of resources found, the better performing the algorithm is. For this measurement, the number of unsuccessful searches are not taken into consideration.

Space complexity is important in this research due to the limited amount of memory available in running the simulation. Resource discovery techniques with high space complexity would trigger the memory out of bounds error, making the techniques unobtainable for comparison with other techniques. Therefore, with the limitation of memory, the resource discovery techniques in this research are measured based on the number of successful searches of the algorithms.

There are several costs that need to be considered in resource discovery specifically, or network computing generally [19, 47, 98]. The weight of a resource discovery technique can be calculated in two ways, they are: network communication cost; and query efficiency. The former calculation technique are described in detail by Sinclair [98], while the latter is discussed by Lin & Wang [57], and Russell & Norvig [93].

2.3.2.1 Network Communication Cost Sinclair [98] outlined the calculation to find network communication costs (NCC). The proposed network communication cost model divided the network into two types of cost: link cost, and node cost. All shortest route between two nodes are taken into calculation. The equation to calculate the weight of the bi-directional link between nodes i and j are as follows:

$$W_{ij} = 0.5N_i + L_{ij} + 0.5N_j$$

where N_i and N_j are the node effective distances of nodes i and j respectively. L_{ij} is the length of link (i, j) in kilometres. The length of the network link is taken into consideration because data is considered to move at the speed of light in computer networks.

The second cost in consideration is the node cost. Node cost is taken into con-

sideration because longer geographical path may have a lower communication cost if it traverses fewer nodes. The node effective distance of node i can be calculated as follows:

$$N_i = E + n_i F$$

where E and F are the length of links that the node is connected to in order to transfer information across node i.

Based on the NCC calculation technique, Sinclair proposed several network connection between countries in Europe. The research further claimed that the previous NCC of internetwork amongst European countries of 7.22M can be reduced to 2.66M. NCC technique is best used for calculating physical computer network infrastructure. It is not suitable for use of peer-to-peer communication cost where there are a lot of nodes and queries being taken into consideration [98].

2.3.2.2 Query Efficiency Russell & Norvig [93] have outlined two methods to assess a resource discovery technique effectiveness, they are: search cost; and total cost. Search cost is the time complexity for the technique to find the solution. Total cost is the combination of the search cost and the path cost of the solution found [93]. The path cost to the solution can also be considered as the number of message being forwarded and returned in the network.

Lin & Wang [57] has outlined a metrics to calculate the effectiveness of a search algorithm. Efficient algorithm should not generate unnecessary messages and queries that being generated should have a high probability to find the target. The metrics proposed by the research is *Query Efficiency* (QE). Hence QE will be represented as η (common Greek letter to show efficiency), the equation for η is as follows :

$$\begin{aligned} \eta &= \frac{QueryHits}{(QueryMessages/N)} \\ &= \frac{QueryHits}{MessagesPerNode} \end{aligned} \tag{2.1}$$

where N is the number of nodes in the simulation. The query efficiency introduced in Lin & Wang research is suitable for power-law P2P networks.

2.3.2.3 Selected Resource Discovery Effectiveness Evaluation The NCC calculation technique proposed by Sinclair [98] enables researchers to evaluate the cost of optical fibre network connections. The cost relies on the length of the connection and the cost of node for network connections between countries. On the other hand, the η proposed by Lin & Wang [57] calculates the cost of a search query using the amount of query hits, query messages being generated, and the number of simulated nodes. The length of the connection and the cost of nodes being connected is not taken into consideration.

Both the NCC and the η can be used to calculate efficiency, however the former calculates the efficiency of networks, and the latter calculates the efficiency of search query techniques. The network communication cost is more suitable to be used to calculate the best backbone network connection between countries, but is not suitable to calculate the cost of simulated networks, where the length of connection between nodes are not taken into consideration. Henceforth the η is being used as the metric to calculate the efficiency of algorithms introduced in this dissertation.

Equation 2.1 shows the equation of η. The higher the number of query efficiency, the more efficient the query technique is. The efficiency calculated with the calculation technique can not be represented as percentage because there are possibilities of having the efficiency of more than 1.0. For example, the calculation will generate efficiency of

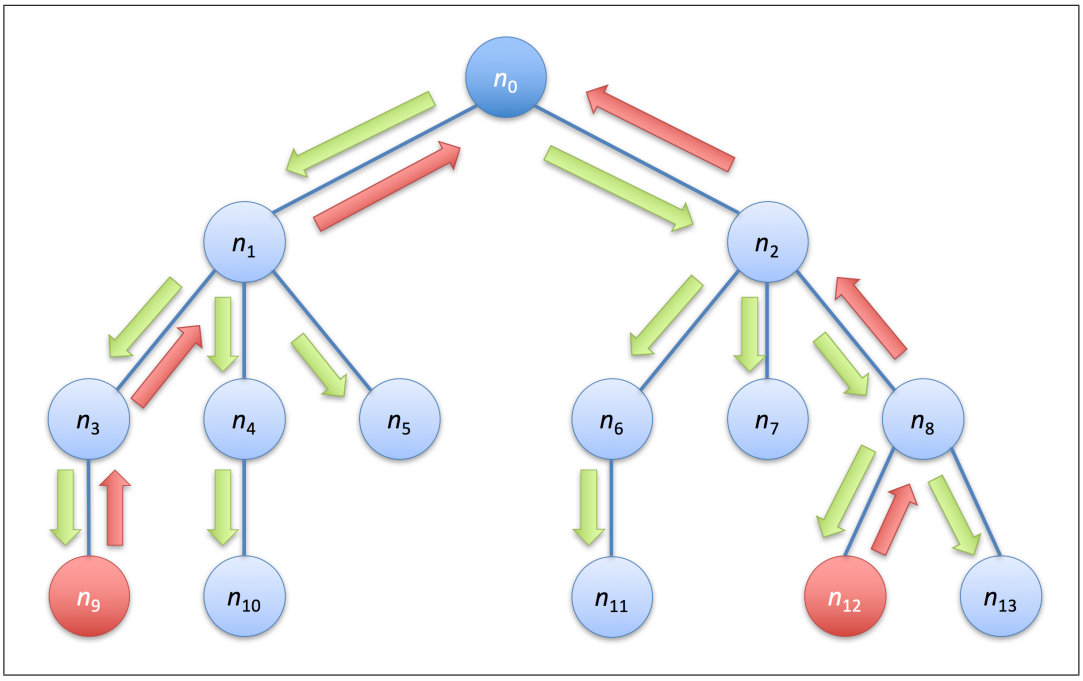

Figure 2.3: Query Messages Forward and Feedbacks (Found Only).

more than one if the number of query message sent by the search algorithm are lower than the number of nodes in the simulated networks, and has at least one query hits. (If the query failed to get any hit, the efficiency will be zero).

Lin & Wang outlined that an efficient query message should not generate excessive and unnecessary queries [57]. These uncontrolled generation of queries will waste the network bandwidth unnecessarily. The research suggest the η by dividing the query hits with the messages per node. Messages per node is the results of dividing query messages with number of nodes in the network. Thus, the η proposed does not take into consideration whether the search algorithm evaluated is using the network for feedback or not, even though the feedback queries uses approximately the same amount of network bandwidth as the query messages going forward.

Calculating the amount of query feedback is important because each information that being send in a network uses the network bandwidth. These query messages

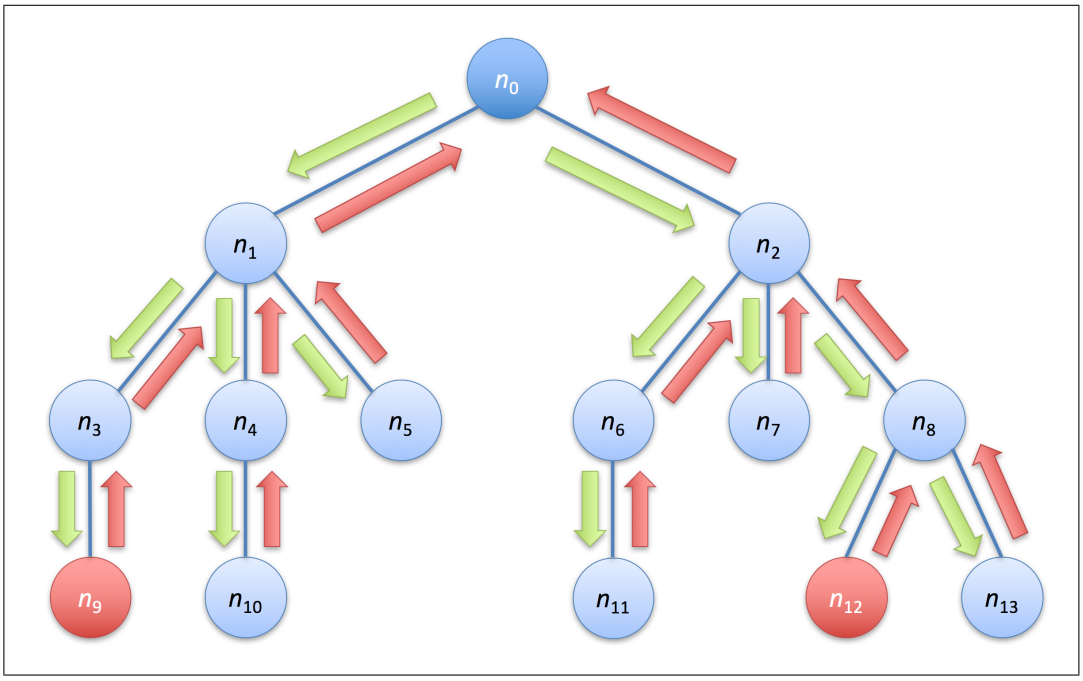

Figure 2.4: Query Messages Forward and Feedbacks (All).

going forward and their feedbacks can accumulate and create queues and latency in the network. Figure 2.3 and Figure 2.4 show the flow of query for search technique that only require queries that have found the target, and search technique that requires all queries to give feedbacks respectively. In both of the figures, node n_0 is the originator of the query, and the nodes being searched is node n_9 and n_{12}. The green and red arrows shows the query messages being forwarded and their feedbacks respectively.

The amount of query messages being forwarded are the same, 13, for Figure 2.3 and Figure 2.4. The amount of query feedbacks for techniques that only require positive feedbacks and techniques that requires all feedback are 6 and 13 respectively. The number of messages sent in the network for Figure 2.3 and Figure 2.4 are 19 and 26 respectively. In this example the technique that require all feedbacks sent 36.84% more messages compared to techniques that only requires positive feedbacks to be sent. The example above is considered small, nevertheless it shows that not only the number of

queries is needed, number of feedbacks is also needed to calculate query efficiency.

Thus, a new efficiency calculation metrics is introduced in this thesis in order to also take into consideration all the network bandwidth communication cost. Thus, this research proposed a new efficiency metrics that take the bandwidth cost of query messages going forward and their feedbacks. The new metrics, η^* (query efficiency with feedback), is based on the equation proposed by Lin [57]. The equation for η^* is as follows:

$$\eta^* = \frac{QueryHits}{\frac{QueryMessagesForward+QueryMessagesFeedback}{N}} \tag{2.2}$$

where N is the number of nodes in the network. The number of query hits is divided with the value of number of messages being sent for that search technique over N.

2.3.3 Resource Discovery in Unstructured P2P Networks

BFS is one of the earliest resource discovery techniques. It derives from mathematical method introduced by Awerbuch and Galager [10]. In the BFS method, the originator of the query sends one walker to each adjacent node [10]. In the BFS resource discovery technique, nodes forward all queries that they have received towards all adjacent nodes. Randić observed this method and considered BFS as flooding the network [90]. Even though BFS is well known for flooding the network, the technique is still widely used in unstructured peer-to-peer (P2P) networks [78].

Kalogeraki et al. introduced Int-BFS, a new resource discovery technique named [46]. Even though Int-BFS added some learning behaviour onto the BFS technique, in the early stages of the learning, the walkers replication behaviour are identical to BFS. Therefore, the problem of walkers from the same query flooding the network might still arise.

Randić proposed a new method called restricted random walk (RRW) [90]. RRW

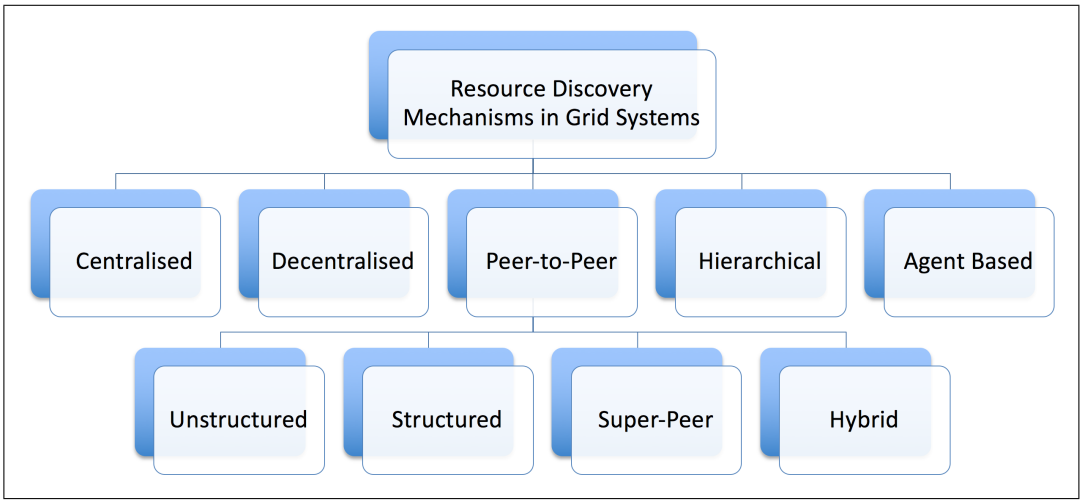

Figure 2.5: Resource Discovery Mechanisms Classification in Grid Computing Systems [79].

reduces the amount of query forwarding by not sending the same query message to adjacent nodes that have previously seen the message. However, if all adjacent nodes have seen the message, RRW will select a random adjacent node to forward the message. The approach introduced by Randić is able to reduce some unnecessary walker replication when compared to the original random walk (RW) technique that was introduced by Gkantsidis et al. [30]. However, the walkers using the RRW technique might be able to forward the query and later replicate exponentially once it has passed the random node.

As shown in Figure 2.5, Navimipour et al. have classified resource discovery mechanisms into five parts, namely, centralised, decentralised, P2P, hierarchical, and agent based. Resource discovery for P2P is further classified as unstructured, structured, super-peer, and hybrid [79]. Unstructured P2P networks are reliable in terms of query correctness, and single point of failure, and can tolerate node dynamicity. Despite that, the complexity of resource discovery algorithms for unstructured P2P networks is $O(N^2)$, where $N$is the number of nodes in the network. Furthermore, the time complexity for unstructured networks has a high order of growth compared to the scale of

the network [79].

Among resource discovery techniques, BFS is the most used resource discovery technique for unstructured P2P networks being used by Napster [105], Iamnitchi [40], Int-BFS [46], Marzolla [65], and Learning Automata-based Resource Discovery (LARD) [104]. Napster [105] and Iamnitchi [40] uses BFS that floods the network with queries. Int-BFS [46] floods the system during its initial stage, but will send less queries once the network has learnt from its previous queries. Marzolla [65] and LARD [104] use BFS, but focus on routing of the unstructured P2P networks in order to minimise the effect of the query flooding.

Navimipour and Milani state that the flooding technique increases the number of walkers exponentially and causes a huge search overhead [78, 79]. Formally, for unstructured networks, the amount of neighbours a node has are represented by k, and the message number of hops as Time-To-Live (TTL). The equation for number of query message forwarding (F_{BFS}) can be written as follows:

$$F_{BFS} = k^{TTL} \tag{2.3}$$

A peer-to-peer node usually has more than 20 neighbours. Therefore, having a 5-hop BFS resource discovery mechanism would forward 20^5 query messages or 3.2 million times. A 10-hop BFS in the network would replicate walkers up to 20^{10}, or 1.024×10^{13} times. These examples are for networks where each node connects on average with other 20 nodes.

In some P2P networks, each node may connect to hundreds of other nodes. In this case, the amount of walker replication is very large, even when considering that the search query was initiated by only one node. In a real world network application, there will be many peers initiating queries. Therefore, implementing a search technique that floods the network will certainly degenerate the network performance and might even

collapse the whole P2P networks.

2.4 Discussion and Conclusions

This chapter contains the literature review for this research. The chapter starts with a brief history of the Internet and its evolution, moving from its initial client-server based architecture towards the use of distributed P2P networks. The transition was fuelled by the advancement of the computer technology, enabling more people to be able to acquire powerful machines able to do what previously only the servers class computer could do.

The innovation of the Internet changing from a centralised source of information, to the development of Web 2.0, where everybody with a device connected to the Internet contributed to the generation of information that was shared across the Internet. This vast amount of information leads to an abundance of resources, which then leads to the requirement to have a suitable resource discovery mechanism in order to find resources that are needed.

The rise of P2P networks coincided with the rise of Napster, a P2P based music sharing platform, which in turn was forced to close after a copyright infringement lawsuit. Nevertheless, other than file sharing, P2P networks were also used in several other applications, such as: multimedia (audio and video) platform; research; alternative Internet; and even currency management.

P2P networks can be categorised based on its utilisation, overlay networks, and the network topology. The overlay networks and the P2P network topology are independent to each other, making it possible to mix between these two criteria in order to develop new types of P2P networks. This is not to mention that the overlay topology and the P2P physical network topology also contain a hybrid of their own.

Resource discovery, a crucial part in P2P networks can also be classified within several categories according to its characteristics. The main categories are, but not

limited to: informed (blind) search and uninformed (knowledge-based).

Uninformed search is basically a resource discovery technique that does not retrieve, process, or store any information regarding the network or onto the network. The query message in uninformed search is lightweight, and the technique relies on the number of query messages forwarded to find its desired resources.

Informed search on the other hand utilises some heuristic approach towards finding the desired resources. The techniques in the informed search category retrieves, processes, and stores information on the network in order to decide an optimised query message forwarding, to return with as many successful searches as possible.

Methods to evaluate the effectiveness of resource discovery techniques are also discussed. Two evaluation techniques, network communication cost (NCC), and query efficiency (QE) were discussed. The NCC is more focused on optical networks between countries, while QE is more suitable for simulated networks. Hence in this dissertation, QE is used as the main evaluation technique. QE is represented as η, a Greek letter usually used to show efficiency. Another new efficiency evaluation technique is introduced, the new metrics, η^*, calculates the query effectiveness with consideration of query message forwarding and feedbacks. This is in contrast to η that only uses query message forwarding in generating the query efficiency.

The discussion continues with resource discovery techniques, but this time focusing on the techniques used in unstructured P2P networks. Among techniques being discussed are the BFS, Int-BFS, RW, and RRW. The discussion also had a brief look at the problem that occurs in unstructured P2P networks that are caused by the most resource discovery techniques; that is flooding of the network with query messages.

3 Implementation of Resource Discovery Mechanisms on Peer-to-Peer Simulator

Chapter Summary

This chapter discusses regarding several resource discovery techniques being used in unstructured P2P networks. Among the resource discovery techniques that have been examined are random walk, restricted random walk, breadth first search, intelligent breadth first search, adaptive probabilistic search, depth first search and the blackboard resource discovery mechanism. Each technique was analysed and the pseudocode of each technique is presented. Several peer-to-peer simulators are listed and reviewed. The chapter ends with the implementation process of several resource discovery techniques onto the most suitable peer-to-peer simulator based on the review.

Summary of Each Section

Introduction	:	Introduction of the chapter
Resource Discovery	:	This section discusses several resource discovery techniques complete with their pseudocodes.
Peer-to-Peer Simulators	:	Various peer-to-peer simulators is being discussed in this section. Strong points and shortcomings of each simulator are also being addressed.
Implementation of the Resource Discovery Mechanisms on PeerSim	:	This section discusses in detail the methodology to implement resource discovery mechanism onto PeerSim. Example of experiment parameter setup is also presented.

3.1 Introduction

Resource discovery is one of the most important parts in any resource sharing systems [55, 78]. Nowadays, the most widely used resource sharing methods are peer-to-peer (P2P) systems. Resources shared in P2P systems are usually spread all over the Internet. Resource discovery plays a vital role in P2P, because it does not possess a central computer storing of all the resources.

A number of researchers have simulated the resource discovery techniques on a P2P network [55, 78]. Lazaro et. al. [55] focuses on resource discovery mechanisms that are used to find computational resources. The paper addresses the main requirements in decentralised resource discovery system such as, search flexibility, scalability, churn and fault tolerance, completeness, accuracy, security, and miscellaneous performance requirements.

Navimipour and Milani [78] analyse and examine resource discovery techniques into four main categories such as, unstructured, structured, super-peer, and hybrid. The main requirements being examined were scalability, dynamicity, reliability, load balancing, response time, and robustness of the resource discovery techniques.

Lazaro et.al. [55], and Navimipour and Milani [78] did a comprehensive study on resource discovery techniques, however did not specify the simulators that was used in their researches. The paper also did not mention the amount of nodes being simulated for the research. Nai'cken et. al. mentioned that most of the researches did not give the reference of the simulator being used, or used a custom simulator built specifically for their research [74].

Several P2P simulators are discussed in this chapter, listing each attributes such as their advantages, disadvantages, architecture, simulations scales, simulated network types, extensibility, and features. These P2P simulators is then compared to each other to find out the best, most suitable simulator to be used for this research experiments.

Justifications about the selection of PeerSim as the P2P simulator for this research are then presented.

Another objective of this chapter is to describe the implementation of several resource discovery mechanisms onto PeerSim [71] in order to facilitate the experimental evaluations described later in this dissertation. PeerSim has been acknowledged by researchers to be having a high level of scalability [22]. Therefore, adding these resource discovery techniques onto PeerSim enables a vast amount of P2P nodes to be simulated making the simulation of the resource discovery mechanisms nearer to the real-life network application.

This chapter is an extension to the proceedings paper in The Third International Conferences on Informatics and Applications (ICIA2014) [42] and to the article in The International Journal of Digital Information and Wireless Communications (IJDIWC) Volume 5 Number 1 [43].

3.2 Resource Discovery Mechanisms

Resource discovery techniques have been studied even before the existence of computer networks. The various techniques being used today are the improvements of previously developed resource discovery methods. Techniques such as Random Walk [30], Restricted Random Walk [90], Breadth-First Search [10], Intelligent BFS [46], Depth-First Search [102], Adaptive Probabilistic Search [106], Blackboard Resource Discovery Mechanism [4] are explained briefly in the following subsections.

3.2.1 Random Walk

Random walk (RW) [30] is a simple method to locate resources. When one node is searching for a resource, the node will check for the resource at its current location. If the resource is not available, the node will send a walker (query) to one adjacent node. The selection of the adjacent node the walker should go next is decided randomly, thus

Algorithm 3.1 Random Walk Pseudocode.

```
00
01 Receive search message;
02 if (this node has the resource) {
03  Reply to originator;
04 }
05 Forward the message to a random adjacent node;
06
```

Algorithm 3.2 Restricted Random Walk Pseudocode.

```
00
01 Receive search message;
02 if (this node has the resource) {
03  Reply to originator;
04 }
05 while (n received the message earlier) do {
06  Randomly select one adjacent node (n);
07 }
08 Forward the message to a random adjacent node;
09
```

the name of the method. The search will be done recursively until it finds the resource that it was looking for. There are no restrictions in RW, therefore there is a possibility that the walker will go back to the node that the walker has been visited previously [30]. The pseudocode for random walk is shown in Algorithm 3.1.

3.2.2 Restricted Random Walk

Restricted random walk (RRW) [90] represents an improvement upon the random walk resource discovery mechanism. It carries most of the criteria of a RW, however, the only differences are when the walker (query) is selecting the adjacent node to go to. RRW's walkers will randomly select an adjacent node that it has never visited before. Therefore, in order to run RRW, the walker will keep track of all of the nodes that it has visited.

The ability to omit the nodes that it has visited makes the RRW a better method than RW because it does not waste the time-to-live (TTL) of its walkers. Algorithm

Algorithm 3.3 Breadth-First Search Pseudocode.

```
00
01 N = number of adjacent nodes;
02
03 Receive search message;
04 if (this node has the resource) {
05  Reply to originator;
06 }
07 for n in range (1:N) {
08  if (n received the message earlier) { }
09  else {
10   Forward the message to n;
11  }
12 }
13
```

3.2 shows the pseudocode for restricted random walk.

3.2.3 Breadth-First Search

Breadth-First Search (BFS) is among the earliest resource discovery techniques that was used in the field of computer networks. In P2P networks, when a node is searching for a resource, it will check itself whether it has the requested resource. If not, the node will query all adjacent nodes for the resources [10]. BFS uses a lot of networking resources by sending a large amount of queries inside the network. This characteristic makes BFS flood the network with queries [90]. Figure 3.2.3a shows the sequence of walkers being generated using the BFS technique. Pseudocode for breadth-first search is shown in Algorithm 3.3.

3.2.4 Intelligent BFS (Int-BFS)

Intelligent BFS (Int-BFS) [46] is an advancement of the BFS [10] searching technique. Int-BFS does not flood the entire network. Instead, it only sends walkers to a fraction of its adjacent nodes. The fraction changes according to the topography and the number of adjacent nodes. In Int-BFS, the node will store query information regarding how

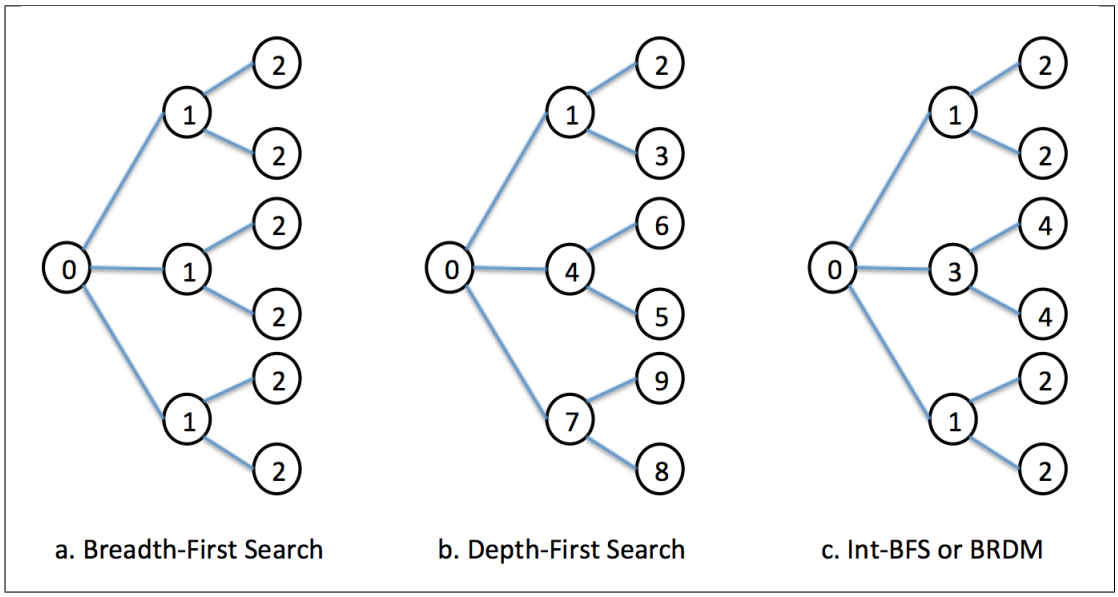

Figure 3.1: Query Message Node Traversal

Algorithm 3.4 Intelligent BFS Pseudocode.

```
00
01 N = number of adjacent nodes;
02 F = Fraction based on topography & number of adjacent nodes;
03
04 Receive search message;
05 if (this node has the resource) {
06  Reply to originator;
07 }
08 for n in range (1:N*F) {
09  while (n received the message earlier) do {
10   Randomly select one adjacent node (n);
11  }
12  Forward the message to n;
13 }
14
```

Algorithm 3.5 Depth-First Search Pseudocode.

```
00
01 N = number of adjacent nodes;
02 count = 0;
03
04 Receive search message;
05 if (this node has the resource) {
06  Reply to originator;
07 }
08 for n in range (1:N) {
09  if (n did not receive the message earlier) {
10   increment count by 1;
11  }
12 }
13 if (count == 0) {
14  Return message to originator;
15 }
16 else {
17  while (n received the message earlier) do {
18   Randomly select one adjacent node (n);
19  }
20  Forward the message to n;
21 }
22
```

many times that the adjacent neighbour has been answering the majority of queries sent by the node.

When a new query arrives, the node will search the stored queries that it has, and forwards it to a set number of neighbours that have answered the most results for the query. Figure 3.2.3c shows the sequence of walkers using Int-BFS. Observe that the technique uses a randomisation technique to select an adjacent node, therefore the results would not be like that in every simulation. Algorithm 3.4 shows the pseudocode for intelligent BFS.

Algorithm 3.6 Adaptive Probabilistic Search Pseudocode.

```
00
01 N = number of adjacent nodes;
02 for n in range (1:N) {
03  P_n=1/N; #probability of n being selected
04
05 Receive search message;
06 if (this node has the resource) {
07  Reply to originator to increase probability being selected;
08 }
09 else {
10  Forward the message randomly using P_n weightage;
11 }
12
```

3.2.5 Depth-First Search

Depth-first search (DFS) [102] can be viewed as the opposite of BFS [10]. Instead of forwarding queries to all adjacent nodes, DFS only forwards one walker to one adjacent node. The selection of the forwarding node is done randomly. The query will continue on forwarding until no other adjacent nodes can be found that it has not visited. When it reaches the end (where there is no other choice), the walker will take one step back to the previous node that it visited, and then continues to go forward to another adjacent node.

Figure 3.2.3b shows the route sequence that a DFS walker would choose during the resource discovery. Observe that the technique uses randomisation therefore the results might differ every simulation cycle. Algorithm 3.5 shows the pseudocode for depth-first search.

3.2.6 Adaptive Probabilistic Search

Adaptive probabilistic search (APS) [106] is a modification of random walk [30]. In the initial stage, APS works just like RW, where all adjacent nodes have an equal probability to be selected. The probability of an adjacent node increases when it returns a hit for any

Algorithm 3.7 Blackboard Resource Discovery Mechanism Pseudocode.

```
00
01 N = number of adjacent nodes;
02 alpha = fraction based on topography and number of adjacent nodes;
03
04 Receive search message;
05 if (this node has the resource) {
06  Reply to originator #write on originator's n1 blackboard
07 }
08 elsif (the blackboard has the path to the resource) {
09  Reply to originator #write on originator's n1 blackboard
10 }
11 else {
12  for n in range (1:alpha*N) {
13   Forward to the message to alpha*N adjacent nodes;
14  }
15 }
16
17 decrease TTL by 1
18 if (TTL == 0 && does not find the resource) {
19  Reply to originator #write on originator's n2 blackboard
20 }
21
```

query. Alternatively, the probability to be selected decreases if the adjacent node did not return any successful searches [106]. For this reason, APS searching capabilities improve over time. The pseudocode for adaptive probabilistic search is shown in Algorithm 3.6.

3.2.7 Blackboard Resource Discovery Mechanism

The Blackboard resource discovery mechanism (BRDM) was first devised in 2004 as a method that utilises an artificial intelligence knowledge representation technique where a blackboard is used to list all the important information about neighbouring entities [4]. In BRDM, the blackboard is used to record recommended adjacent nodes to forward a future query to. If there is a recommended node, the query will be forwarded to the node; if there are not any recommended nodes, the query will be forwarded to a number of random adjacent nodes.

BRDM forwards queries using walkers. The amount of walkers is decided based on the TTL of the query and the amount of adjacent nodes. The percentage of neighbours to forward the query to can be modified to suit the topology of the P2P networks. BRDM is utilised as one of the efficient scheduling policies for ParCop, a decentralised P2P system [3]. Figure 3.2.3c shows the sequence of walkers using BRDM. Observe that the technique uses randomisation, therefore the results would differ every simulation. Algorithm 3.7 shows the pseudocode for blackboard resource discovery mechanism.

3.2.8 Summary of Resource Discovery Mechanisms

As stated in Section 2.3, all the above mentioned resource discovery techniques can be classified into either uninformed or informed search technique. In uninformed search technique, no data will be stored in the query messages or the nodes. In informed search technique, the information are stored by the query message or the nodes. Old information will be discarded using the least recently used (LRU) algorithm. The techniques can also be classified according to their query message replication. Each technique will be either generate multiple replication of the query message, or it can just forward the query that it received. Table 3.1 shows each resource discovery techniques being discussed above and their characteristics.

3.3 Peer-to-Peer Simulators

There has been a lot of research conducted on various aspects of P2P networks. Like other scientific research, studies regarding P2P systems need to produce solutions that are valid and able to be reproduced by other researchers. There are three methods to achieve this goal, they are: analytical; using simulations; or by performing experiments on actual P2P systems [74].

Non-simulation methods for P2P systems are analytical, and by performing on actual P2P systems. Analytical approach requires the P2P system to be modelled mathem-

Table 3.1: Resource Discovery Techniques

Resource Discovery Techniques	Multiple Query Replication	Single Query Forwarding	Informed Search	Uninformed Search
Random Walk (RW) [30]		✓		✓
Restricted Random Walk (RRW) [90]		✓		✓
Breadth-First Search (BFS) [10]	✓			✓
Intelligent Breadth-First Search (Int-BFS) [46]	✓		✓	
Depth-First Search (DFS) [102]		✓		✓
Adaptive Probabilistic Search (APS) [106]		✓	✓	
Blackboard Resource Discovery Mechanism (BRDM) [4]	✓		✓	
Alpha Breadth-First Search (α-BFS)*	✓			✓
Lightweight Blackboard Resource Discovery Mechanism (LBRDM)*	✓		✓	

*: Techniques developed in this research. Details regarding α-BFS and LBRDM techniques will be discussed in Section 5 and Section 6 respectively.

atically. Nevertheless, real life P2P systems tend to be very complex and robust. In order to produce the mathematical models, there are many assumptions being done to simplify the P2P systems. Therefore, any findings using this method will have a very limited applicability [74].

Experimenting proposed P2P system on an actual system is a difficult thing to do. P2P systems may scale to a large numbers of nodes, therefore, in order to do the experiments on actual systems a large number of nodes are needed. This method requires significant amount of resources, and can be very costly in administration and hardware. In case of studies that need to introduce malicious nodes onto the network, many security issues that require limitations and controls need to be implemented carefully [74].

In spite aforementioned difficulties and high cost of a real world test bed for actual P2P systems, there are researchers that use this method. One example is the test bed PlanetLab [86]. Currently it has 1353 actual nodes spread across 717 sites. All of these nodes are subject to real-world network conditions.

Simulations of P2P networks do not suffer from the difficulties of the analytical and the experiments on actual system. The implementation cost of simulations are just a small fraction when compared to the actual P2P system test bed. The simulations are less complex when compared to the mathematical model used in the analytical method [74]. There are several P2P simulators available for researchers to use for the scientific tests. They are: 3LS [52]; General Peer-to-Peer Simulator (GPS) [110]; Neurogrid [75]; P2PSim [35]; PeerSim [71]; PeerThing [85]; Query Cycle [73]; and RealPeer [38].

Each simulator has their own advantages and disadvantages. Each simulator has some tradeoffs in order to achieve the goal of the P2P simulator [22, 73, 75]. Among concerns regarding the experiments on P2P simulations are as follows: poor documentation; limited functionality; missing statistical data collection; and publications do not clearly specify simulations that they use. This leads to poor reproducibility of results

and analysis and comparison [74].

3.3.1 3LS

3LS or 3 Layered System is an open source simulator for overlay networks that focuses on extensibility and usability. The three architectural layers in 3LS are: network model (bottom layer); protocol model (middle layer); and user model (top layer). Communications in 3LS can only occur with adjacent layers. The simulator also has several queues to handle requests, they are: Outbox; Inbox-for-network-delay; Inbox-for-processor-delay; and Inbox. These queues are implemented to allow simulation of network traffic and CPU delay [52].

3LS integrates with GUI and uses main memory to store each event executed in simulations. Consequently the simulator has a high memory overhead that limits the scalability of the simulated system. At most, 3LS can simulate networks with only a couple of thousand peers on a regular machine. 3LS is well documented, however the simulator is not available on the web, researchers can request for the source code from the authors of the simulator [22].

3.3.2 General Peer-to-Peer Simulator (GPS)

GPS is an event-driven P2P simulator that focuses on modelling P2P protocols as accurate, efficient, realistic, and dynamic as possible. The simulator is written in Java, and maintains its efficiency by modelling the communication at the message level. GPS does not have a fixed synchronous increments, instead the processing and time advance based on occurrence of events. Packet level simulation is not implemented to maintain its performance [74, 110].

GPS is poorly documented, in addition, it has a steep learning curve in order to use the simulator. The simulator excels in extensibility due to the availability of all infrastructures required for P2P simulations. Other advantages of this simulator is that

it is able to functionally model BitTorrent [15] protocol, that seems to be a difficult feat to be done by other simulators discussed in this chapter. In order to be able to functionally simulate BitTorrent protocol, the simulator was designed to be tightly coupled to the BitTorrent protocol. Unfortunately, this simulator design decision has made it difficult for researchers to implement protocols other than BitTorrent onto it [22].

3.3.3 Neurogrid

Neurogrid is a simulator developed to simulate large scale neural. In the early stages of the development, the simulator has been used to compare P2P protocols such as Freenet [18], Gnutella [31] with the Neurogrid protocol [45]. In recent years, with the advancement of computing power, Neurogrid is being used to simulate large scale neural networks[13]. Neurogrid is a single-threaded discrete even simulator able to simulate on structured and unstructured networks [75].

Neurogrid can only simulate overlay layer of the network, this is due to the simulator design that assumes all links between nodes have equal bandwidths. The simplifying assumptions allow the simulator to generate up to 300,000 nodes. Extensive documentations for Neurogrid are available on the Internet, however are disorganised in the form of wiki documentation [22].

3.3.4 P2PSim

P2PSim has a different goal compared to other simulators. The simulator design focuses on three main objectives: to uncomplicate P2P protocol source codes; to make comparison between protocols easy; and to have reasonable performance. P2PSim utilises threads, thus making protocol implementations as similar as possible to their pseudocodes. P2PSim can only simulate structured overlay network topologies, and does not support distributed simulations. Among network topology that are available in P2PSim

are: random graph; end-to-end time graph; G2 graphs; and Euclidean graph [35].

P2PSim simulation can scale up to 3000 nodes for Euclidean constant failure model topology. The protocols in P2PSim can be extended, however limited and poor C++ API documentation of the simulator makes it difficult to be done. Another drawback of the simulator is that it is unable to simulate unstructured or semi-structured P2P routing protocols [22].

3.3.5 PeerSim

PeerSim is a P2P simulator developed under BISON project. Simulations in PeerSim can be done using cycle-driven or event-driven simulation engine. The simulator has been designed to simulate large and dynamic P2P networks dynamic. PeerSim does not have a GUI, simulations parameters are set using a configuration files that are loaded before the simulation [71].

Class packages are offered in PeerSim in accordance with the type of simulations to be done. One of PeerSim's forte is its scalability, the simulator can simulate networks of up to one million nodes. It has a steep learning curve as the simulator has limited amount of documentations available on the Internet. Nevertheless, the source code comment and API documentations are sufficient for researchers to study and use. There are no distributed simulation available in PeerSim, forcing experiments to be done on only one machine [22].

3.3.6 PeerThing

PeerThing is a P2P simulator with architectures that can be categorised into two: system behaviour; and system scenario. The former allows defining the behaviour of each node in the network. This architecture define the whole behaviour of the network based on the behaviour in a single peer. The latter define the number of nodes and their connections' characteristics such as uplink and downlink speeds, delays, loops, and

actions. Once the system behaviour and system scenario have been set, a corresponding XML (eXtended Markup Language) is generated [85].

PeerThing is claimed to be able to generate 2000 nodes for the Gnutella model [31] and 1000 nodes for Napster [77] model. This simulator has a GUI, and simulation results can be saved in comma separated values for further analysis. PeerThing has the best user manual, however the API documentation source code is not well commented. Therefore, making it difficult to extend the simulator for other protocols or functionalities [22].

3.3.7 Query Cycle

The Query Cycle Simulator, or sometimes called P2PSim is a P2P simulator developed by Stanford University for its P2P sociology project. The simulator focuses on accurately simulating user behaviour in P2P file sharing network. Among parameters in the simulators are query activity, download behaviour, and uptime. As its name suggests, each cycle in the Query Cycle Simulator is based on queries generated by the network. The cycle finishes when all peers that submit queries received satisfactory response [73, 89].

Query Cycle Simulator can simulate up to one million nodes, however there are some instance that the simulator does not scale when when simulating more than 1000 peers. The simulator also has GUI present, making it easier to simulate for sociology studies. Nonetheless, this simulator has poor API documentation, thus limiting the possibility to extend this simulator [22, 89].

3.3.8 RealPeer

RealPeer is a P2P system development framework. The system can be executed as a simulator or as a real P2P application. Classes in RealPeer are generic, and the utilisation of plug-in design pattern makes it easily extensible. Researchers can combine or exchange elements of the framework to make it suitable for the intended experiments.

RealPeer is quite scalable, able to simulate up to 20,000 peers [38].

There is no GUI available for RealPeer, all commands need to be written on the command prompt. Simulation results for RealPeer are stored in text files. Nevertheless, the numeric value are stored with no corresponding variable name. RealPeer is still relatively new, however being actively and extensively developed, it is more likely that the aforementioned problems will be addressed due time [22].

3.3.9 Selection of Peer-to-Peer Simulators

P2P simulators that have been discussed in previous Section can be summarised as in Table 3.2. The table lists the usability & documentations, scalability, extensibility, GUI availability, and the programming language being used to develop the simulators. Above mentioned simulators can also be summarised based on their architecture (Table 3.3). Table 3.4 shows comparisons between P2P simulators for researches.

Each of the simulators has its advantages and disadvantages. These traits are usually based on the main objectives of the simulator being built. Among simulators that were developed for specific objectives are Neurogrid, Query Cycle, and P2PSim. Neurogrid's objective is to simulate large scale neural networks, early adaptations of the P2P protocols were only to proof of concept for Neurogrid protocol [13, 45].

Query Cycle is a part of a larger research group, Stanford P2P sociology project. Its main objectives are to mimic distilled and simplified human rules of behaviour towards the P2P networks. These implementations of behaviour are expected to be used to face several issues such as trust, privacy, and economics [89]. P2PSim focuses on making peer-to-peer protocol easy to understand, and convenient. The objective of easy protocol comprehension has lead to tradeoffs with the performance of the simulator [35]. Simulators with narrow objectives might have simplifying assumptions and difficult to extend with new protocols. Therefore, simulators like the Neurogrid, Query Cycle, and P2PSim are not considered to be the simulator for this research.

Table 3.2: Summary of P2P Simulator Analysis [22].

Simulators	Usability / Documentation	Scalability (maximum)	Extensibility	GUI	Language
3LS [52]	N/A	20 nodes	Theoretically extensible but practical implementation is required.	Yes	Java
GPS [110]	Poor documentation.	512 nodes	Poor documentation limits extensibility.	Yes	Java
Neurogrid [75]	Good API documentation and user manual. Source code is not well commented.	300,000 nodes	Designed to be extensible.	Yes	Java
P2PSim [35]	Poor documentation.	3,000 nodes	Limited protocol extensibility and complex handling.	Yes	C++
PeerSim [71]	Detailed API documentation. Lacks detail in user manual.	10^6 nodes	Designed to be extensible.	No	Java
PeerThing [85]	Good user manual. Poor API documentation. Source code is not well commented.	2,000 nodes	Extensible.	Yes	Java
Query Cycle [73]	Poor API documentation and user manual.	10^6 nodes	Limited.	Yes	Java
RealPeer [38]	Well commented source code. Poor API documentation and user manual.	20,000 nodes	Poor API documentation limits extensibility.	No	Java

Table 3.3: P2P Simulator Architecture Comparison [22].

Simulators	Structured / Unstructured	Event / Cycle / Query based	Triggered by	Distributed Simulation	Dynamic Network	Special Features	Accuracy	Efficiency
3LS [52]	Unstructured	Event based	Time scheduling to user specific behaviour.	No	Yes	3 Separate level architecture	Low	Low
GPS [110]	Both	Discrete event based	Occurrence of event.	Yes	Yes	Macroscopic models	High	High
Neurogrid [75]	Both	Discrete event based	Time scheduling.	No	Yes	N/A	Medium	Low
P2PSim [35]	Structured	Discrete event based	Time scheduling.	No	No	Pseudocode-like implementation	Low	Low
PeerSim [71]	Both	Event based and cycle based	Time and random selection.	No	Yes	N/A	Medium	Low (Event) High (Cycle)
PeerThing [85]	N/A	Event based	Time scheduling to user specific behaviour.	Yes	Yes	System behaviour and system scenario definition	High	High
Query Cycle [73]	Unstructured	Query based	Time scheduling.	No	Yes	For file sharing	Low	Low
RealPeer [38]	Unstructured	Discrete event	Time scheduling.	No	No	Development environment	Medium	Medium

Table 3.4: P2P Simulator Comparison.

Simulators	Usability & Documentation	Scalability	Extensibility	Runtime, Status, & GUI	Overall Marks (★)
PeerSim [71]	★★★★	★★★★★	★★★	★★★★	16
Neurogrid [75]	★★★★★	★★★★	★★★	★★★	15
PeerThing [85]	★★★★	★	★★★★	★★★★★	14
Query Cycle [73]	★★	★★★★★	★★	★★★★	13
RealPeer [38]	★★★★	★★	★★★	★★	11
P2PSim [35]	★	★	★★	★★★★	8
GPS [110]	★	★	★★	★★★★	8
3LS [52]	N/A	N/A	★	★★★	4

One of the concerns in P2P simulations are the number of nodes in the network being simulated. P2P simulators with low scalability might come out with protocols that are not suitable for real life implementation of peer-to-peer where the number of nodes is very large. Experiments done on the simulators might not truly reflect the actual implementations of the P2P protocols. Among simulators that has low scalability are 3LS [52], GPS [108], PeerThing [85], and P2PSim [35] with 20, 512, 2000, and 3000 nodes respectively. These simulators were not explored further to be the simulator of this research.

Subsequent to the omissions of several simulators above, the two remaining simulators are PeerSim [71] and RealPeer [38]. Both of the simulators are extensible, and neither have a simulator GUI to simplify the simulations settings. The advantages of PeerSim against RealPeer are PeerSim has a well documented API and its development are more mature and supported by researchers. On the other hand, RealPeer has a better user manual, making it an easier to use compared to PeerSim [22]. PeerSim was chosen for this research, considering that a well documented API is one of the most

important aspect in order to extend a simulator and simulating new protocols.

3.4 Implementation of the Resource Discovery Mechanisms on PeerSim

Several resource discovery techniques have been implemented onto PeerSim. The pseudocode of the RW, RRW, BFS, Int-BFS, DFS, APS, and BRDM are shown in Algorithm 3.1, 3.2, 3.3, 3.4, 3.5, 3.6, and 3.7 respectively. Although the pseudocodes of the resource discovery mechanisms are straightforward, however, the implementation of the techniques onto PeerSim was found to be anything but straight forward.

In order to achieve its high scalability, developers of PeerSim have developed it in Java to fully utilise its object oriented programming approach. The node generator and basic characteristic of the nodes are placed at the base of the PeerSim (`peersim.core`). All of the topology generators are placed in `peersim.dynamics` and the methods to plot the nodes and topologies are placed in `peersim.graph`. Resource discovery techniques are placed in `peersim.extras.gj.isearch`, a dedicated package for search techniques.

Given that the objects that needed to be used for the resource discovery techniques are located in several places in the PeerSim package, a configuration file is needed to run the simulator. An example of the configuration file for BRDM techniques is shown in Algorithm A.1. Lines 01 to 05 of the configuration file is for the setup of the whole environment. It is then followed with the setup of the resource discovery technique from lines 07 to 17. Setup configurations for the topology is stated in lines 19 to 22. Lines 24 to 29 lists all the initial setup configuration for the search. The remainder of the configuration file are several settings for printing the results of the resource discovery techniques.

As mentioned above, the implementation of resource discovery techniques are not straight forward. Resource discovery techniques are needed to be interpreted as object oriented and not as structured as the pseudocodes may suggest. As an example, the

pseudocode for BRDM are shown in Algorithm 3.7, while the real coding for the BRDM in PeerSim is as shown from Algorithm A.2 until Algorithm A.7. Notice that the code for `BRDMProtocol.java` is an extension to `SearchProtocol.java`, a java programming that consists of 650 plus lines of codes.

In order to get parameters from outside the program, values needed to be added in the configuration file. In the case of BRDM, the values of alpha multipliers are listed in the configuration file, to be read by `SearchProtocol.java`. The blackboard implementation of recommended, ($N1$), and unrecommended, ($N2$), lists are declared as `HashMap<Node, ArrayList<Node>>` so that the list can grow according the information received, and are renamed to RL and UL respectively to avoid confusion with nodes numbering.

The amount of message forwarding are dictated by the alpha multipliers. The message forwarding is then decided based on the recommended and unrecommended lists, where in BRDM, the query will be forwarded to the nodes in the recommended list, and the unlisted, while avoiding forwards to the nodes in the unrecommended list. The remainder of the BRDM code are the methods needed to add, search, and update the information in recommended and unrecommended blackboard list.

3.5 Conclusions

There are a lot of P2P simulators available on the Internet. Most of the simulators are developed for certain objectives, thus making the simulator hard to be extended for other protocols. There are times when the objectives limit the scalability and made simplifying assumptions to suite their own research. These features and limitations of P2P simulators have been used as the justification for selecting PeerSim as the simulator of this research.

PeerSim is a very scalable P2P simulator. In order to achieve its high scalability, the simulator is designed to be as object oriented as possible. Components for the

simulation in PeerSim are distributed across the whole PeerSim package. Researchers who want to use PeerSim, would need to choose suitable components of PeerSim based on the specific research and/or application requirements. These components are usually dependent upon each other, requiring users to dig deep just to find a simple component, for example, nodes. To change a simple parameter in PeerSim might require the user to change multiple files that depend on the parameter.

This difficulty does not include the complexity of extensions, polymorphs, and overrides within the PeerSim package. To fully appreciate the scalability of PeerSim, codes implemented in it should neither be structured nor static. Due to the large package of PeerSim, and many restrictions and coding techniques, utilising PeerSim as a P2P simulator will require a steep learning curve. Once a researcher was able to endure the steep learning curve, the testing of P2P techniques and algorithms should be easy. Nevertheless, memory and simulation cycle management of PeerSim is commendable, enabling simulations of up to 1 million nodes, a rare characteristic among P2P simulators.

4 Unstructured P2P Network Topology Simulation

Chapter Summary

This chapter discusses the characteristics of several network topology models. The P2P network topology generator models discussed are as follows: Heuristically optimised trade-offs, regular rooted tree, star, ring lattice, Watts-Strogatz, scale free Barabási-Albert, scale free Dorogovtsev-Mendes, and K-out topology generators. The rules and conditions that need to be followed in order to correctly imitate real life P2P networks are also listed. The chapter ends with the selection of an unstructured P2P network topology generator to be used in all of the experiments beyond this chapter.

Summary of Each Section

Introduction	:	Introduction of the chapter
P2P Network Generator Models	:	Lists all the P2P network generator models. The characteristics of each model is also being presented.
Selection of the Topology Generator Models	:	This chapter discusses the real life characteristics of P2P networks, and some rules and conditions that the network follows. The P2P network generator model that can closely imitate real-life P2P networks is then selected to be used in the resource discovery experiments.

4.1 Introduction

Many real world scenarios follow the power law distribution. Power law is a functional relationship between two quantities, where one of the quantities varies as a power of another. In the example of P2P networks, the two quantities are the number of the nodes and the number of neighbours that each node has [2]. There will be many nodes

that have a small number of neighbours, and there will be a small number of nodes that have many neighbours.

4.2 P2P Network Generator Models

Network topology types are crucial in experimenting with P2P protocols or algorithms. As stated in Section 2.3.2, out of 287 papers in P2P field, there are 146 papers that do not use a simulator, 71 papers uses a simulator but do not specify which simulator being used, and 43 papers developed their custom simulators for the research [74]. Above mentioned researches generate random network or custom made topologies that are suitable for their P2P experiments. Karaoglanoglou & Karatza [48] for example, make use of a "Grid Graph", a custom made topology generator suitable for P2P Grid computing to generate their simulated networks.

There are several topology generator being used to generate a P2P network, such as the Inet Topology Generator, Georgia Tech Internetwork Topology Models (GT-ITM), and Tiers Topology Generator. The Inet topology generator is developed by University of Michigan, used to generate an Internet topology based on some configuration parameter. The developed topology is structured networks. GT-ITM creates flat random graphs, and two types of hierarchical graphs. The Tiers topology generator generates a random graph [16, 17, 67].

A topology with large number of nodes N is needed in order to be able to imitate real life resource discovery in an unstructured P2P network [6]. Furthermore, the simulated networks need to follow the power law to truly reflect unstructured P2P networks. With PeerSim [71], there are several types of networks that can be generated, namely HOT [64], K-out [71], regular rooted tree [7], ring lattice [107], scale free Barabási-Albert [11], scale free Dorogovtsev-Mendes [21], star [39], and Watts-Strogatz [107] topology. All of these topologies can be generated for very large networks (eg. 1,000,000 nodes) and subsequently be tested with the resource discovery mechanism techniques. This chapter

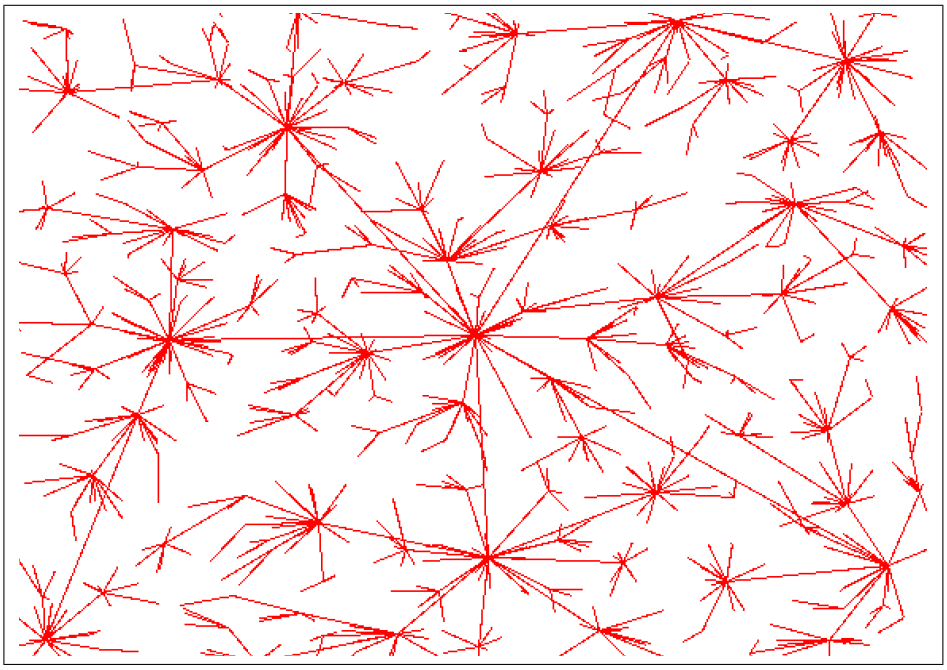

Figure 4.1: Generated Topology using Heuristically Optimised Trade-offs Model (N=1000, α=2).

is an extension to an article submitted to ARPN Journal of Engineering and Applied Sciences [44].

4.2.1 Heuristically Optimised Trade-offs Topology

Mahadevan et al. introduced Heuristically Optimised Trade-offs (HOT) network topology that follows the power law [64]. However, as shown in Figure 4.1, topologies generated by the HOT technique do not resemble unstructured P2P networks. In each topology generated using the HOT technique, there is a central node connected to several other nodes that contains a cluster of nodes connected to it. Even though the central node does not have many neighbours, it shows that the whole network has one point of failure, a characteristic that clearly resembles a structured network.

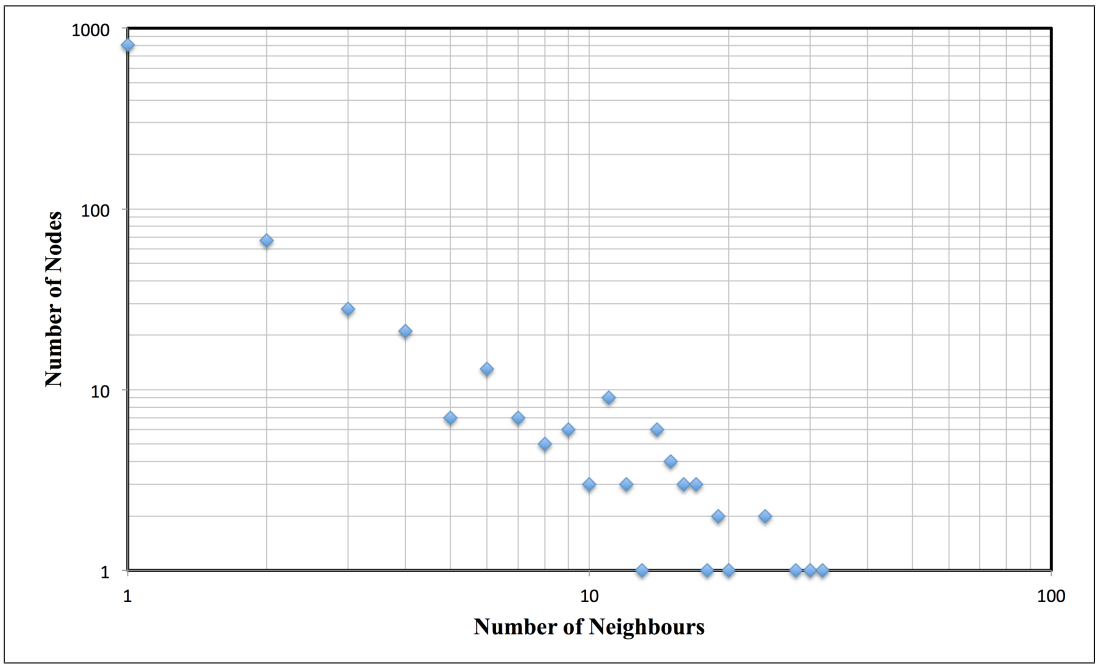

Figure 4.2: Number of Nodes Against Number of Neighbours using Heuristically Optimised Trade-offs Model (N=1000, α=2).

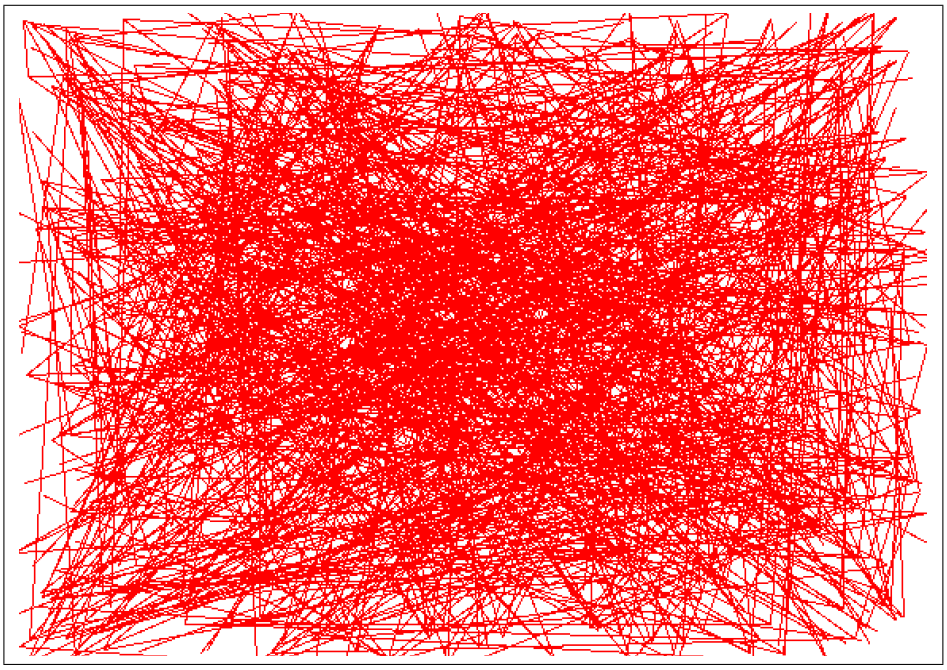

Figure 4.3: Generated Topology using Regular Rooted Tree Model (N=1000, k=2).

HOT technique utilises an integer variable of "out degree", α to generate its topology. The value of α dictates the type of topology that the HOT technique will generate. The most suitable value for α is between 2 and less or equal than the square root of the network size, $(2 \leqslant \alpha \leqslant \sqrt{N})$. Extremely low α, $(1 \leqslant \alpha \ll \sqrt{N})$, will generate topology similar to a Star topology, while α value that is more than the square root of the network size, $(\alpha > \sqrt{N})$, will generate a random graph.

4.2.2 Reg Rooted Tree Topology

Regular rooted tree [7], as the name suggests, is a rooted tree topology using basic rules and configurations. This topology has a single point of failure, the root, therefore networks generated using the regular rooted tree model can be considered as structured. Regular rooted trees have the characteristic of any tree topology, that is the topology does not contain any loops. Although this characteristic might be useful in establishing network routes, it will not represent a real-life unstructured P2P network topology.

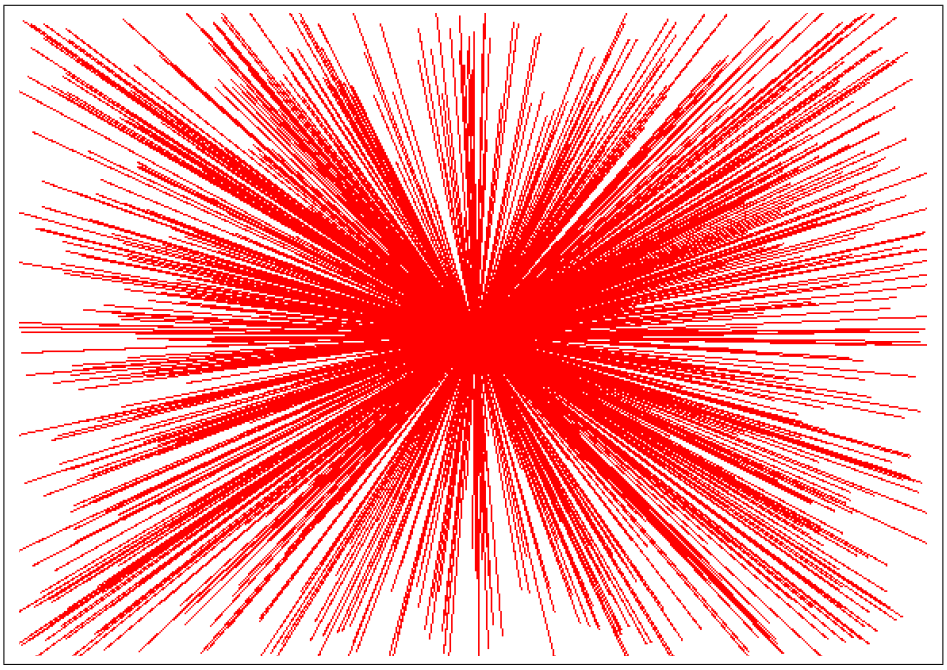

Figure 4.4: Generated Topology using Star Model (N=1000).

4.2.3 Star Topology

Star topology is one of the most basic computer network topologies. It consists of one super node, and other nodes in the network have only one connection, connecting to the super node. The one super node is the single point of failure of the network topology. If the super node is down or broken, the whole network collapses, because other nodes are no longer connected to each other. Iamnitchi and Foster suggest avoiding using unrealistically optimistic topology configurations such as star topology (Figure 4.4) [39].

4.2.4 Ring Lattice Topology

The ring lattice is a highly regular topology [107]. The topology is created by arranging n nodes in a circle, and joining each nodes to its k nearest neighbours, where n is the number of nodes and k is a small constant. Ring lattice topology is resilient. In

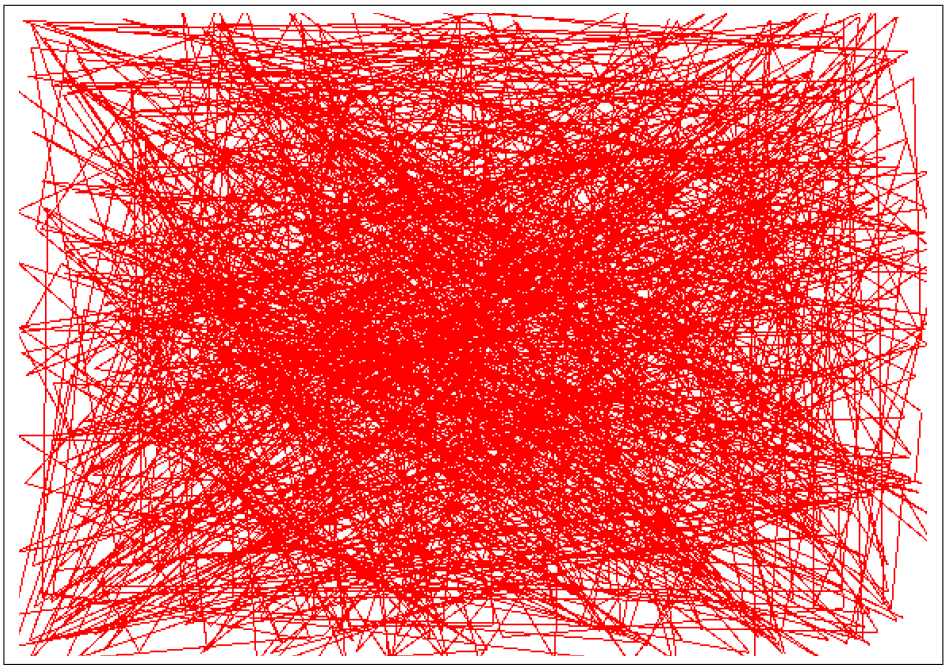

Figure 4.5: Generated Topology using Ring Lattice Model (N=1000, k=2).

order to isolate the network into two parts, the minimum number of connections to be disconnected are $2k$. Therefore, the higher the k, the more resilient the network topology will be. Even though ring lattice topology has a high availability and does not have a single point of failure, the network has a clear structure and therefore does not represent an unstructured network.

4.2.5 Watts-Strogatz Topology

Watts-Strogatz introduced a topology based on the ring lattice topology [107]. The topology starts similarly as the ring lattice, with every node being arranged to form a circle. This topology however does have another variable for its randomness, $(\beta \mid \beta = 0 \rightarrow 1)$. When the value of β is equal to 0, the Watts-Strogatz technique will wire the network using the k nearest neighbour identical with the ring lattice topology. A slight increase in the value of p will generate a small-world network. When the value of β is equal to 1, an entirely random graph will be generated.

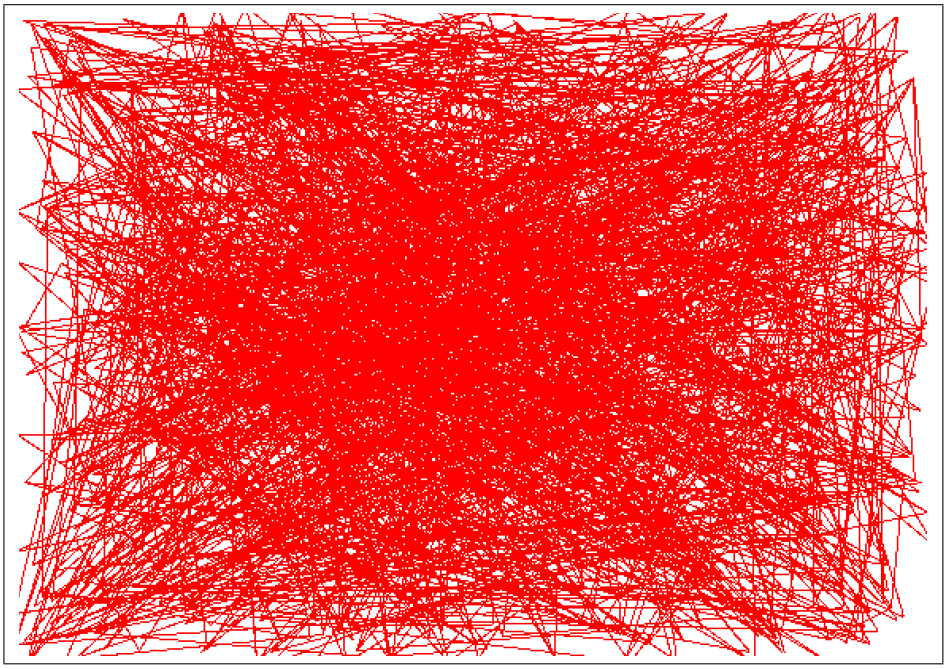

Figure 4.6: Generated Topology using Watts-Strogatz Model (N=1000, k=2, β=0.2).

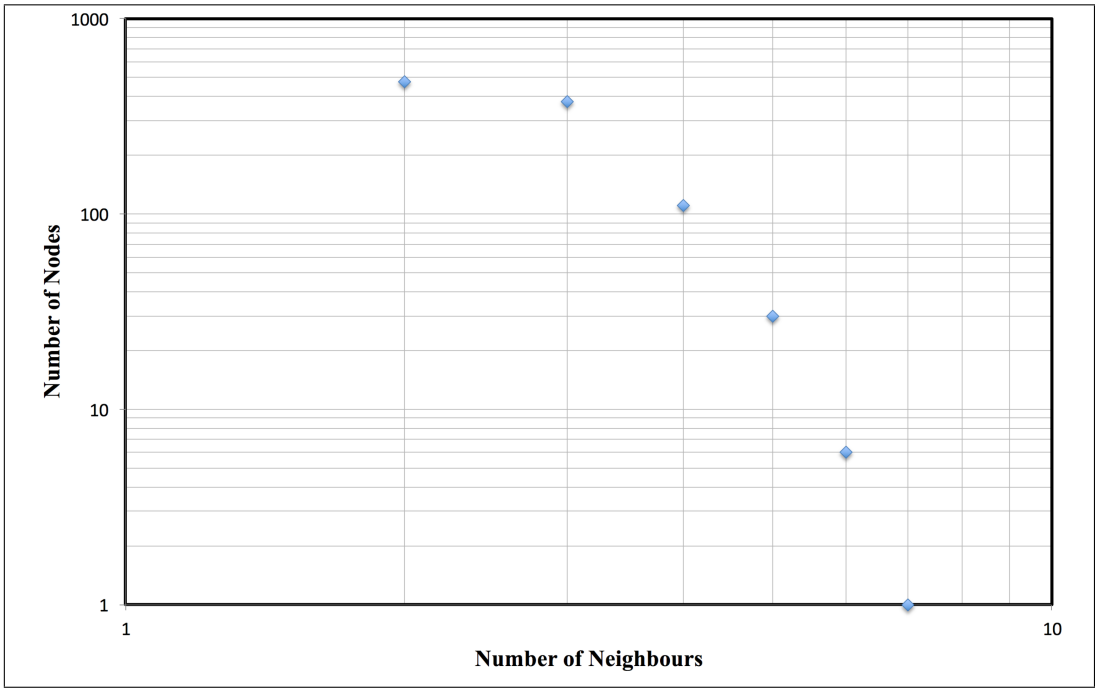

Figure 4.7: Number of Nodes Against Number of Neighbours using Watts-Strogatz Model (N=1000, k=2, β=0.2).

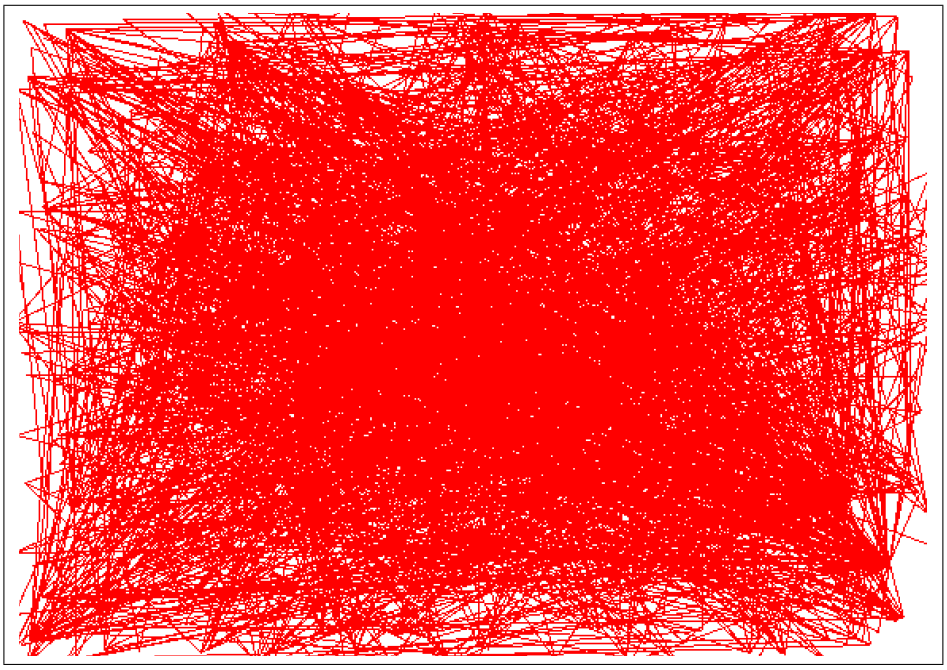

Figure 4.8: Generated Topology using Scale Free Barabási-Albert Model (N=1000, k=2).

The value of β and k for the Watts-Strogatz model needs to be carefully set in order to get a suitable topology that follows the power law topology [107]. Having a high value of β might generate a random graph, which does not show a clear obedience to the power law. Another disadvantage of the topology is that the generated topology is not suitable for rescaling. A slight change by either increasing or decreasing the number of nodes in the network affects the whole simulated network.

4.2.6 Scale Free Barabási-Albert Topology

The Scale Free Barabási-Albert (BA) model has been introduced by Albert-László Barabási and Réka Albert [11]. Prior to Barabási and Albert, the mathematicians, Paul Erdös and Alfréd Rényi had proposed a scale free network model. However, topologies generated by the Erdös-Rényi model do not follow the power law distribution. The scale Free Barabási-Albert model can be considered the first scale free model that follows the

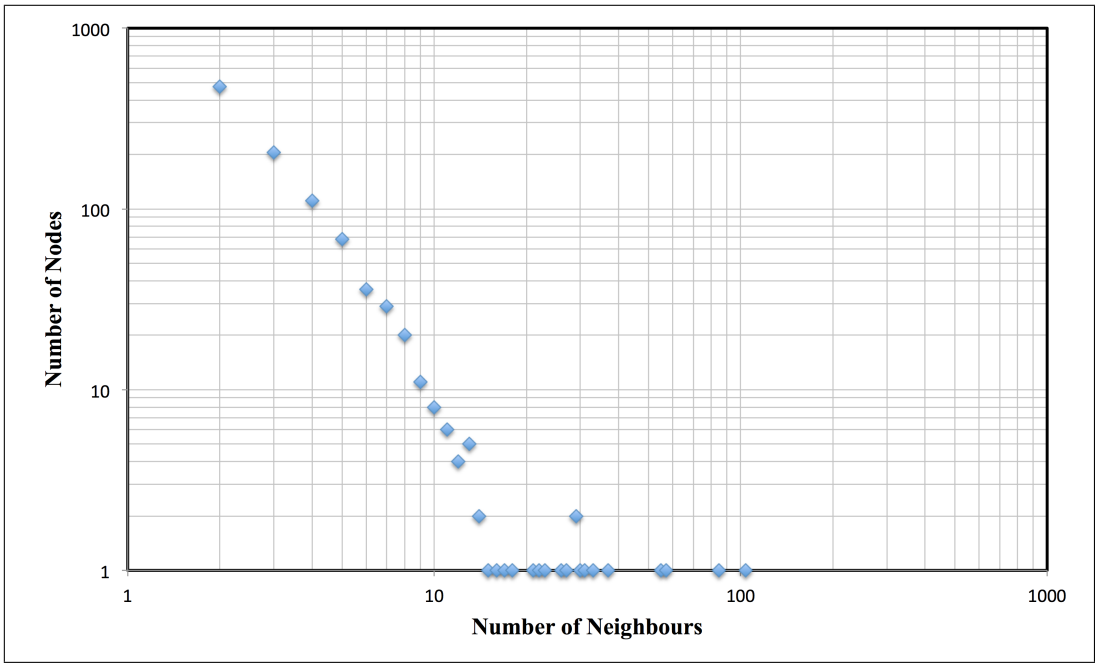

Figure 4.9: Number of Nodes Against Number of Neighbours using Scale Free Barabási-Albert Model (N=1000, k=2).

power law. Nodes can leave and new nodes can be added to the network while obeying the power law.

Figure 4.8 shows a topology of 1000 nodes generated using the Barabási-Albert model. The topology generation starts with k nodes, and every subsequent node added to the Barabási-Albert topology will be connected with k random neighbours that are already in the network. Thus, the longer a node is in the generated network, the higher the possibility it is connected with new nodes.

4.2.7 Scale Free Dorogovtsev-Mendes Topology

The Scale Free Dorogovtsev-Mendes [21] is another scale free model to generate an unstructured P2P network topology. This model is an incremental technique, and starts with k nodes. All subsequent nodes to be added to the Dorogovtsev-Mendes topology are connected to the two ends of the network. As the name suggests, scale free Dorogovtsev-

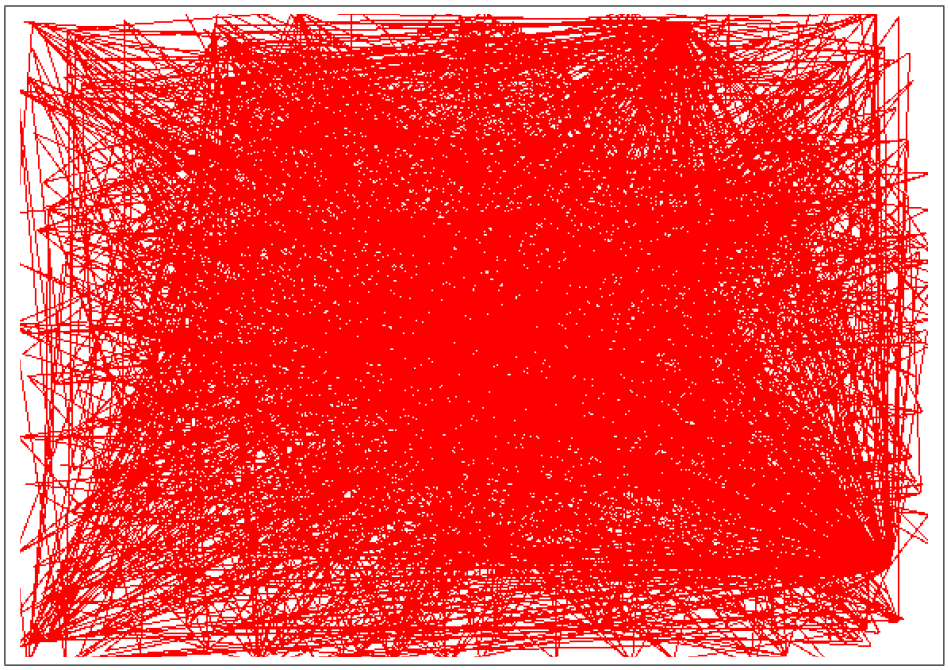

Figure 4.10: Generated Topology using Scale Free Dorogovtsev-Mendes Model (N=1000, k=2).

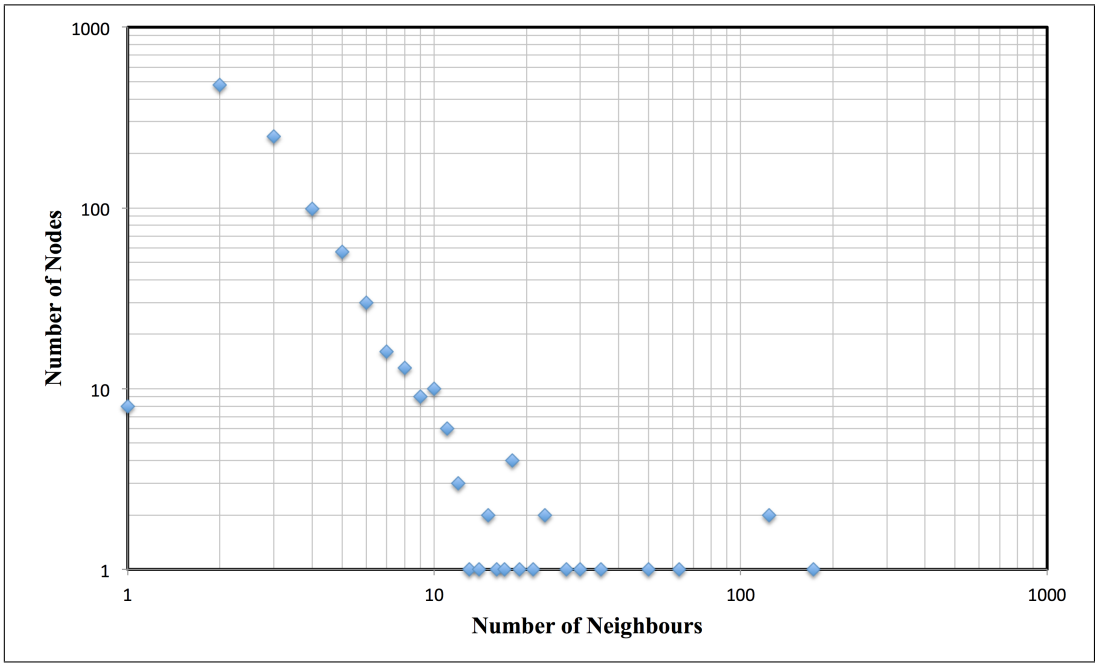

Figure 4.11: Number of Nodes Against Number of Neighbours using Scale Free Dorogovtsev-Mendes Model (N=1000, k=2).

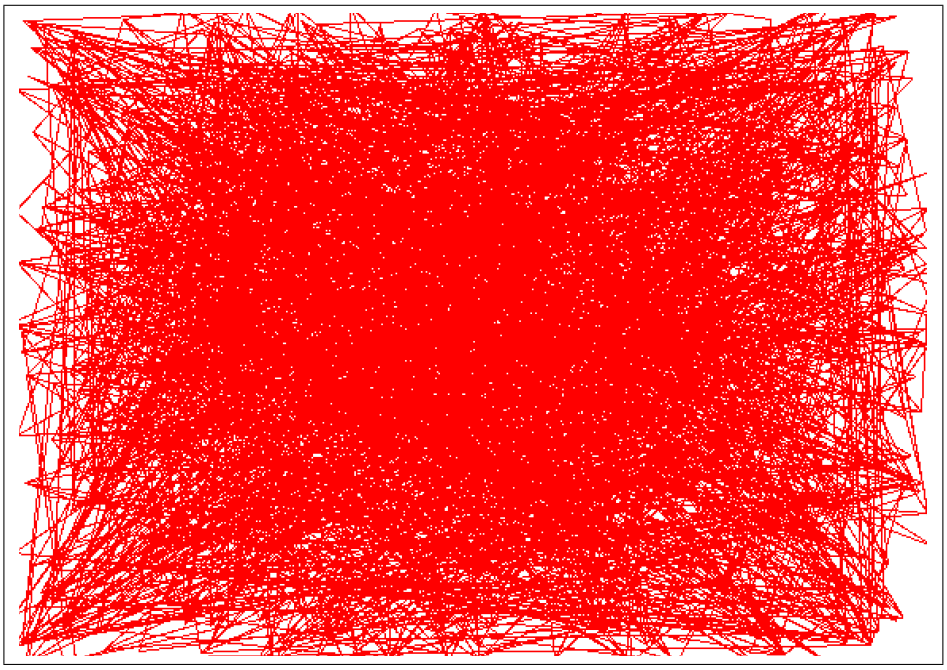

Figure 4.12: Generated Topology using K-out Model (N=1000, k=2).

Mendes is a scale free model, which means that the network scale can be freely extended just by adding new nodes. Nodes in the scale free model join and leave the whole network more easily without disrupting the whole characteristics (whether it obeys power law, and whether it is structured, or unstructured) of the network model. The model generates an unstructured network topology that obeys the power law.

4.2.8 K-Out Topology

The K-out model [71] is an original topology model by PeerSim that is based on the Barabási-Albert model [11]. The only differences are during the initial state of the topology generation. The initial number of nodes in a Barabási-Albert model is equal to the value of k. K-out on the contrary starts the topology generation from 1 node, and the number of neighbours to be connected is $N-1$, where N is the number of nodes in the topology. Every subsequent node added to the K-out topology will be connected with k random neighbours that are already in the network. The K-out model is also a

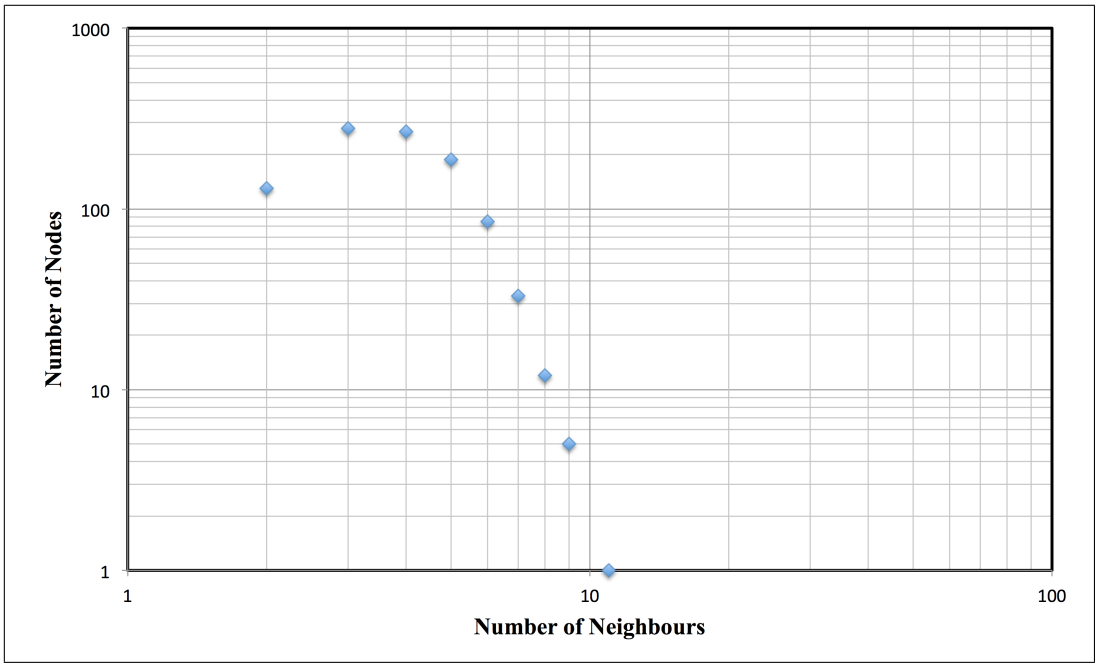

Figure 4.13: Number of Nodes Against Number of Neighbours using K-out Model (N=1000, k=2).

scale free network topology generator model.

4.3 Selection of Topology Generator Models

Table 4.1 summarises all the topology generator model discussed in this chapter. Table 4.3 lists all the variables utilised in the generation of simulated topology of each technique. In order to correctly simulate the BFS technique, we need to generate simulated network topologies that are unstructured, close to real world scenarios (that obey the power law), and scale free. We have eliminated ring lattice [107], regular rooted tree [7], star [39] and HOT [64] from our consideration because the network topologies that they generate are structured. We also omitted the Watts-Strogatz [107] model because of its non-scale free characteristic and the difficulty to set the model to simulate network topologies that follow the power law.

Figures 4.8, 4.10, and 4.12 show the generated topology of 1,000 nodes with k value

Table 4.1: Summary of Topology Generator Models

Topology Model	Scale free	Power law	Unstructured	Structured
Scale free Barabási-Albert [11]	✓	✓	✓	
Scale free Dorogovtsev-Mendes [21]	✓	✓	✓	
K-out [71]	✓		✓	
Watts-Strogatz [107]		✓	✓	
Ring lattice [107]				✓
Regular rooted tree [7]				✓
Star [39]				✓
HOT [64]		✓		✓

of 2 using scale free Barabási-Albert [11], scale free Dorogovtsev-Mendes [21], and K-out [71] respectively. K-out, scale free Barabási-Albert, and scale free Dorogovtsev-Mendes are the only three generators that are able to produce topologies that follow the power law, and are also scale free. Being scale free makes it easy to add and remove nodes while maintaining the node and number of neighbours power law correlation. It is hard to differentiate between the topologies generated by the three models because there is no significant structure, a trait that is the most important of being an unstructured topology.

However, it is easy to differentiate between the models once the number of neighbours on each node is taken into consideration. Figures 4.9, 4.11, and 4.13 show the logarithmic scaled graph of the number of nodes against number of neighbours on each node for the scale free Barabási-Albert, scale free Dorogovtsev-Mendes, and K-out models, respectively. Even though the K-out model is only dissimilar to the Barabási-Albert during the initial part of the topology generation, the graph shows that the K-out model does not generate a power law topology. The maximum number of neighbours in the K-out model for 1000 nodes is only 11.

Both scale free Barabási-Albert [11] and scale free Dorogovtsev-Mendes [21] graphs

Table 4.2: Topology Generators' Variables

Topology Generator	Variable(s)	Value	Description
Heuristically Optimised Trade-offs [64]	α	Small integer	Out-degree. $1 \leqslant \alpha \ll \sqrt{N}$: Star topology. $2 \leqslant \alpha \leqslant \sqrt{N}$: Clustered topology. $\alpha > \sqrt{N}$: Random topology.
Reg Rooted Tree [7]	k	Small integer	Number of outgoing links of nodes in the tree.
Star [39]	N/A	N/A	N/A
Ring Lattice [107]	k	Small integer	Number of neighbouring nodes being connected. (All nodes are initially arranged in one circle)
Watts-Strogatz [107]	k	Small integer	Number of neighbouring nodes being connected. (All nodes are initially arranged in one circle).
	β	Real number $(0 \rightarrow 1)$	Probability of rewiring. $\beta = 0.0$: Similar to Ring Lattice topology. $\beta = 1.0$: Random topology
Scale Free Barabási-Albert [11]	k	Small integer	Two purposes 1. Number of initial nodes to be generated. 2. Number of random neighbours each node would connect during each node generation.
Scale Free Dorogovtsev-Mendes [21]	k	Small integer	Initial number of nodes being generated.
K-Out [71]	k	Small integer	Number of random neighbours each node would connect during each node generation.

N: Number of nodes generated for the topology

show that the models generate a power law network. However, when examined closely, there are three outliers in Figure 4.11. These outliers do not seem significant, however, when considering that the graphs are logarithmic scaled, meaning that the differences in values for the outliers is more significant. In the generated topologies, the Dorogovtsev-Mendes model has two nodes with 124 neighbours, and one node with 173 neighbours. These values differ greatly with the node with the 4th most neighbours, having only 63 neighbours. There are also eight nodes in the scale free Dorogovtsev-Mendes model that contain only one neighbour, which is quite odd considering every node in the topology should at least have two neighbours, because every new node added to the network should connect to the two ends of the topology.

Even though K-out [71] is a scale free model, the network topologies that it generates do not follow the power law. Thus, the model is removed from our list of topologies we have experimented with. This leaves the remaining two, scale free Barabási-Albert [11] and scale free Dorogovtsev-Mendes [21]. We opted against using scale free Dorogovtsev-Mendes because it generates some irregular nodes, such as nodes with too many neighbours compared to other nodes, and nodes with only one neighbour that the model should not have generated. Based on the findings regarding the generation of unstructured P2P networks above, from here henceforth, only the scale free Barabási-Albert model will be used to test the new α-BFS resource discovery mechanism in the next chapter.

4.4 Conclusions

Several P2P network topology generators, namely HOT [64], K-out [71], regular rooted tree [7], ring lattice [107], scale free Barabási-Albert [11], scale free Dorogovtsev-Mendes [21], star [39], and Watts-Strogatz [107] have been tested. The simulated network topologies are then compared to see characteristics of each network topology generators.

The summary of findings are shown in Table 4.1. From the tests where the number

of nodes are set to be 1,000, the HOT [64], regular rooted tree [7], ring lattice [107], and star [39] show the characteristics of being structured. Whereas the K-out [71], scale free Barabási-Albert [11], scale free Dorogovtsev-Mendes [21], and Watts-Strogatz [107] show the characteristics of being unstructured.

Among the structured network topology generators, only the HOT [64] generator models follows the power law, a crucial criterion to simulate real life network topology. In the unstructured P2P network topology generator, all except the K-Out [71] follows the power law distribution. Scale free Barabási-Albert [11], scale free Dorogovtsev-Mendes [21], and K-Out [71] shows a scale free characteristics, another criterion needed to closely imitate real life network topology.

Among all eight network topology generator models, only the scale free Barabási-Albert [11] and scale free Dorogovtsev-Mendes [21] that fulfil the criteria of being unstructured, follow the power law, and scale free. Further observations on the graphs of both topology model generators show that there are some outliers in scale free Dorogovtsev-Mendes [21] model. Therefore, it is found that the best unstructured P2P network generator to test the resource discovery techniques is the scale free Barabási-Albert [11] network model generator.

5 Alpha Breadth First Search

Chapter Summary

This chapter discusses the Alpha breadth first search, an improved version of breadth-first search. It implements alpha multipliers, a set of 5 multipliers that dictate the number of message forwarding from each node. This chapter discusses the two main techniques that have been implemented on alpha breadth first search, that is alpha multipliers and restricted random walk with null exception. Both of the techniques are aimed at reducing message forwarding by eliminating unnecessary duplicate query messages from the network.

Summary of Each Section

Introduction	:	Introduction of the chapter
Alpha Breadth First Search Overview	:	This section discusses regarding the problem of the breadth first search, a basic flooding method that is still widely used until today, and the need to overcome the problem.
Alpha Multipliers	:	Alpha multipliers, the multiplier that controls the duplication of query message forwarding are discussed in this section.
Restricted Random Walk with Null Exception	:	A simple but essential step of eliminating unnecessary and duplicating query message in order to control breadth-first search flooding is discussed in this section.
Experimental Setup	:	Describes network topology simulation setup and query behaviours setup that are required for the experiments.
Experimental Results	:	All experimental steps and results for α-BFS and BFS are listed.

5.1 Introduction

This chapter will explain a new technique that consists of two new walker replication rules that significantly decreases the amount of walker replications while maintaining good search results. The first rule consists of implementing alpha multipliers. The second rule is on determining the forwarding of walkers to nodes that have seen the query before. Details regarding the two rules of walker replication in alpha breadth-first search (α-BFS) will be discussed further in the following sections.

This section is a continuation of an article journal published in International Journal

of Digital Information and Wireless Communications, that focuses on testing resource discovery techniques on simulated P2P networks with one million nodes [43]. Some parts of this chapter are also published in the ARPN Journal of Engineering and Applied Sciences [44].

5.2 Alpha Breadth First Search Overview

The breadth-first search technique is widely used in resource discovery in unstructured P2P networks. Although the technique usually gets the most successful hits, it does this with a very high cost to the network by flooding it with many replicated query messages. The message flooding, no matter how small the message, can degenerate the performance of the whole network, and at worst, could bring down the network altogether.

The idea of alpha breadth-first search (α-BFS) is to contain the message flooding to an acceptable level, while maintaining the same amount of successful searches. We took two approaches in order to achieve this goal. The first approach is by implementing alpha multipliers; these change according to the number of hops the query message has done. The second approach is to control the neighbour selection so that the message forwarding does not consider the neighbours that have already seen the message.

5.3 Alpha Multipliers

Alpha multipliers are a set of multipliers that dictate how many replications a query message can make of itself. The amount of replications are based on two variables at each stage of the message forwarding. They are the number of neighbours that the node x has, $L(n_x)$, and the alpha multipliers ($\alpha_{hops} \mid 0.0 \leqslant \alpha_x \leqslant 1.0 \,\&\, hops = 1, 2, 3, 4, 5, ...$) that are based on the number of hops that the query message has take so far.

α_{hops} is a real number ranging from 0.0 to 1.0, and $hops$ is the number of hops that the query message is about to execute. For example the value for the first until the fifth

hops' alpha multipliers can be set as follow: $\alpha_1 = 1.0$, $\alpha_2 = 0.8$, $\alpha_3 = 0.6$, $\alpha_4 = 0.4$, and $\alpha_5 = 0.2$. Number of query message forwarding for each node is equal to the number of current alpha multiplied by the number of neighbours that the node has (eg. number of forwards for the first hop of origin node is $\alpha_1 * L(n_o)$).

There are possibilities that the number of forwards fell below 1 (eg. number of adjacent neighbour is 2, and the current alpha multiplier is 0.4). In order to maintain continuation of the search, the number of forwards will be reset to 1. If not, the query message may finish earlier than it should have been, eliminating the chance to find the resource needed.

Let n_o be the node where the query messages originate from. The number of query message that are forwarded $F_{\alpha-BFS}$ for TTL of 5 and above in α-BFS are as follows:

$$F_{\alpha-BFS} = \alpha_5 \cdot L\left(\alpha_4 \cdot L\left(\alpha_3 \cdot L\left(\alpha_2 \cdot L\left(\alpha_1 \cdot L\left(n_o\right)\right)\right)\right)\right). \tag{5.1}$$

Using the above mentioned set of alpha multiplier values on networks with 20 neighbours per node, say then the messages sent is reduced from 3.2 million messages to just 122,880 messages only as shown in following calculation:

$$\begin{aligned} F_{\alpha-BFS} &= (1.0 \times 20) \times (0.8 \times 20) \times (0.6 \times 20) \times (0.4 \times 20) \times (0.2 \times 20) \\ &= 20 \times 16 \times 12 \times 8 \times 4 \\ &= 122,880. \end{aligned}$$

The pseudocode for the QF calculation is provided in Algorithm 5.1.

Algorithm 5.1 Determining $QF_{\alpha-BFS}$ Value Pseudocode.

```
01
02 n = number of neighbours;
03 alpha[5] = [1.0, 0.8, 0.6, 0.4, 0.2];
04 hops = number of hops;
05
06 QF = n * alpha[hops-1];
07 round QF to nearest integer.
08
09 if (QF is less than 1) {
10  QF is set to 1;
11 }
12
```

5.4 Restricted Random Walk With Null Exception

PeerSim [71] has two types of neighbour selection for the purpose of query forwarding. One is random walk (RW) [30], and the other one is restricted random walk (RRW) [90]. Both RW and RRW decide to forward or replicate any query message randomly. The only differences is that RRW uses a method named `selectFreeNeighbor` that will forward to one of the free neighbours, that is, neighbour nodes that did not receive the query message earlier. However, if there is no free neighbour available, it will still select and return one non-free neighbour to forward the message. We consider that the query message forwarding to a non-free neighbour is unnecessary and a waste of network resource. Algorithm 5.2 depicts the pseudocode of message forwarding that is being used by RRW.

The message forwarding method that is being used by RRW returns an ID of a neighbour (`neighbourID`) even though there is no free neighbour available. We have altered the message forwarding selection method to only return a free neighbour's `neighbourID`. The return value is set to `null` if there is no free neighbour available. Once the search protocol received the `null` value, no message forwarding will be done. The query message will stop on that node. The new RRW with `null` exception is run after the

Algorithm 5.2 RRW Message Forwarding Pseudocode.

```
01
02 receive query message.
03
04 if (there are free neighbours) {
05  select one of the free neighbours;
06 } else {
07  select one of the neighbours;
08 }
09 return neighbourID;
10
```

Algorithm 5.3 RRW with Null Exception Pseudocode.

```
01
02 receive query message.
03
04 if (there are free neighbours) {
05  select one of the free neighbours;
06  return neighbourID;
07 } else {
08  return null;
09 }
10
```

calculation of $F_{\alpha-BFS}$, therefore, it will overwrite the outcome of the $F_{\alpha-BFS}$ if there is no free neighbours available. The new method is named, `selectFreeNeighborOnly`. Algorithm 5.3 shows the pseudocode of the restricted random walk with null exception.

5.5 Experimental Setup

The experimental setup for the experiments can be divided into two parts, namely the topology setup, and the query behaviour setup. The former setup focuses on the topology of the simulated network being experimented on. The setup focuses on the generation of the topology such as the type of topology generator, the direction of connections, variables, and random seed. The latter setup focuses on setup affect the behaviours of the query such as alpha multipliers, query forwarding and replication, number of initial query, and time-to-live.

5.5.1 Topology Setup

The BFS and α-BFS techniques have been tested according to these parameters: one million nodes distributed and wired using the scale free Barabási-Albert model; undirected connection; k variable of two; and is run of 20 cycles. The experimental setup parameters are shown in Table 5.1.

The experiments were done using three different random seed in order to get multiple results using the listed sets of alpha multipliers. The first random seed is 1234567890, a standard seed being used in PeerSim simulations. The second and third random seed is the first 10 and the following 10 decimal places of π respectively. The value of 22/7 up until the 20^{th} decimal places is 3.14159265358979323846, therefore the value for the second and third random seeds are 1415926535 and 8979323846 respectively.

Figure 5.1, Figure 5.2, and Figure 5.3 show the distribution logarithmic graph of the number of neighbours against number of nodes in the generated topology using 1234567890, 1415926535, and 8979323846 as the random seed respectively. As expected,

the graph obeys the power law and does not show any outliers in the data. Number of neighbours of nodes have a mean ($\bar{x}$) of 3.999992 across all three random seeds. Number of neighbours standard deviation (σ) for random seed of 1234567890, 1415926535, and 8979323846 are 8.4006762, 8.6408129, and 8.2416443 respectively.

Table 5.1: Experiment Topology Setup for α-BFS, BRDM, & LBRDM.

Parameters	Value
Topology generator model	Scale free Barabási-Albert
Number of nodes (N)	1,000,000
Number of initial query	1
Number of cycles run	20
k	2
Random Seeds	1234567890, 1415926535, and 8979323846
Time To Live (TTL)	5, 10, and 20

5.5.2 Query Behaviour Setup

The alpha multipliers are a set of five-tuple numbers that act as multipliers to find the number of query message replication needed on each step of the search. The five-tuple numbers can be of any combination of numbers ranged from zero to one. Let the alpha multipliers be numbers with one decimal place, the five-tuple numbers can have ${}_{10}\mathrm{C}_5$, 252 combinations.

Several patterns or orders of alpha multipliers such as, fixed numbers, ascending order, descending order, division, and logarithmic has been chosen for the experiments. The numbers and their patterns are as shown in Table 5.2. Fixed numbers pattern is where the numbers are all the same from α_1 to α_5. Ascending is when the value of α_1 keeps on increasing until α_5. Ascending pattern means that smaller number of queries are forwarded nearer to the originator, and the query forwarding increases when away

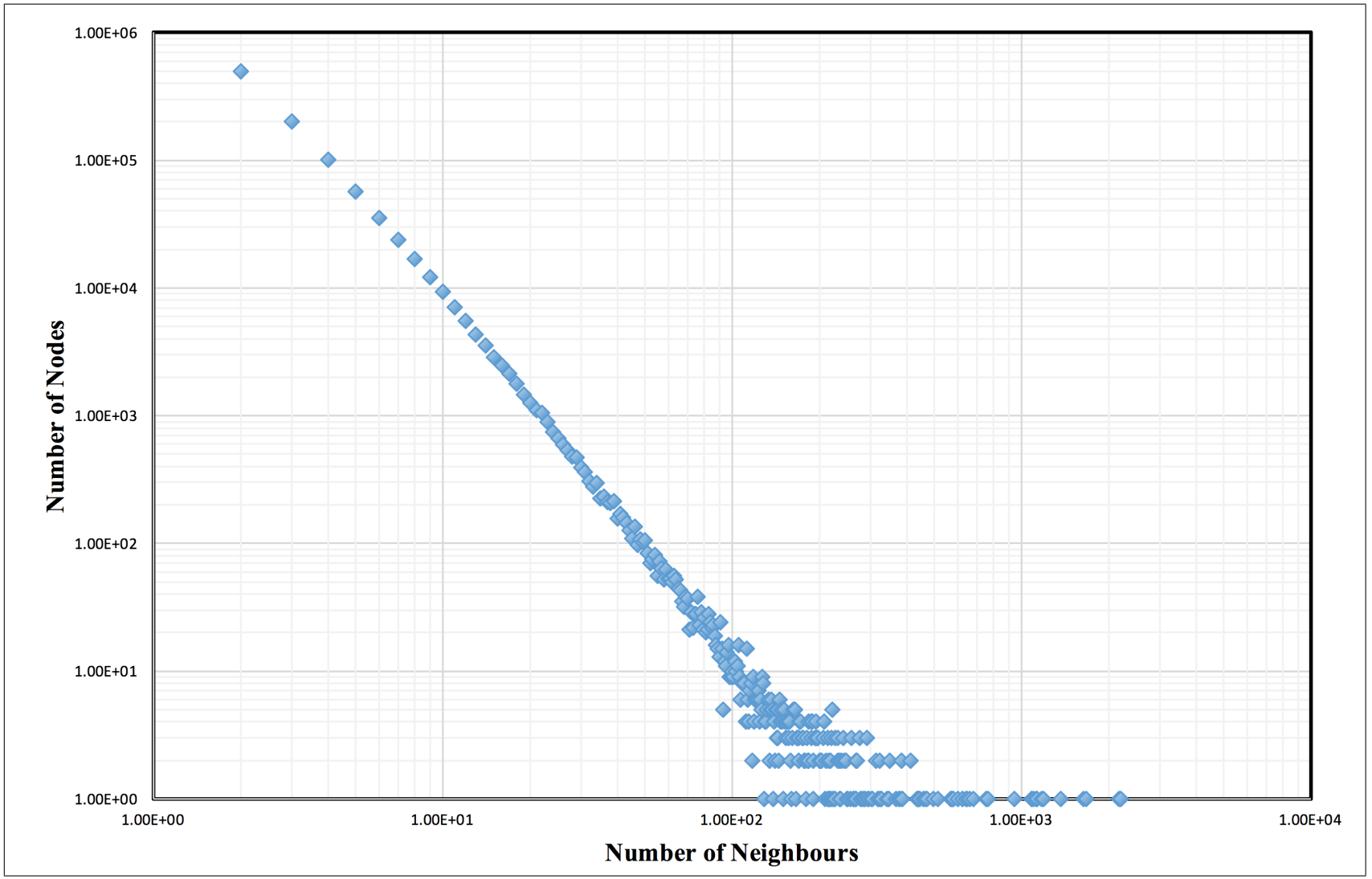

Figure 5.1: Number of Nodes Against Number of Neighbours (Random seed=1234567890, N=1 million, k=2).

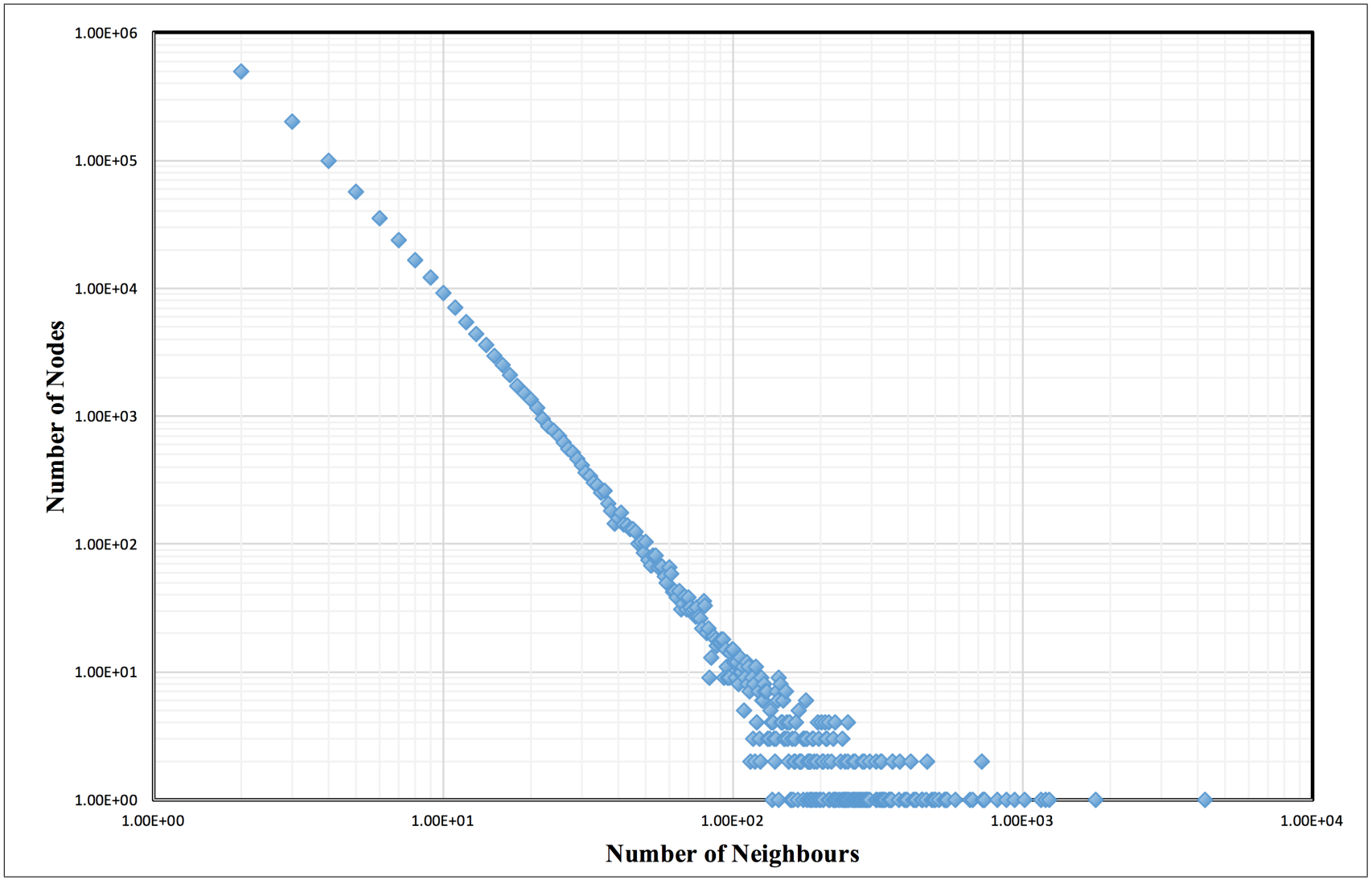

Figure 5.2: Number of Nodes Against Number of Neighbours (Random seed=1415926535, N=1 million, k=2).

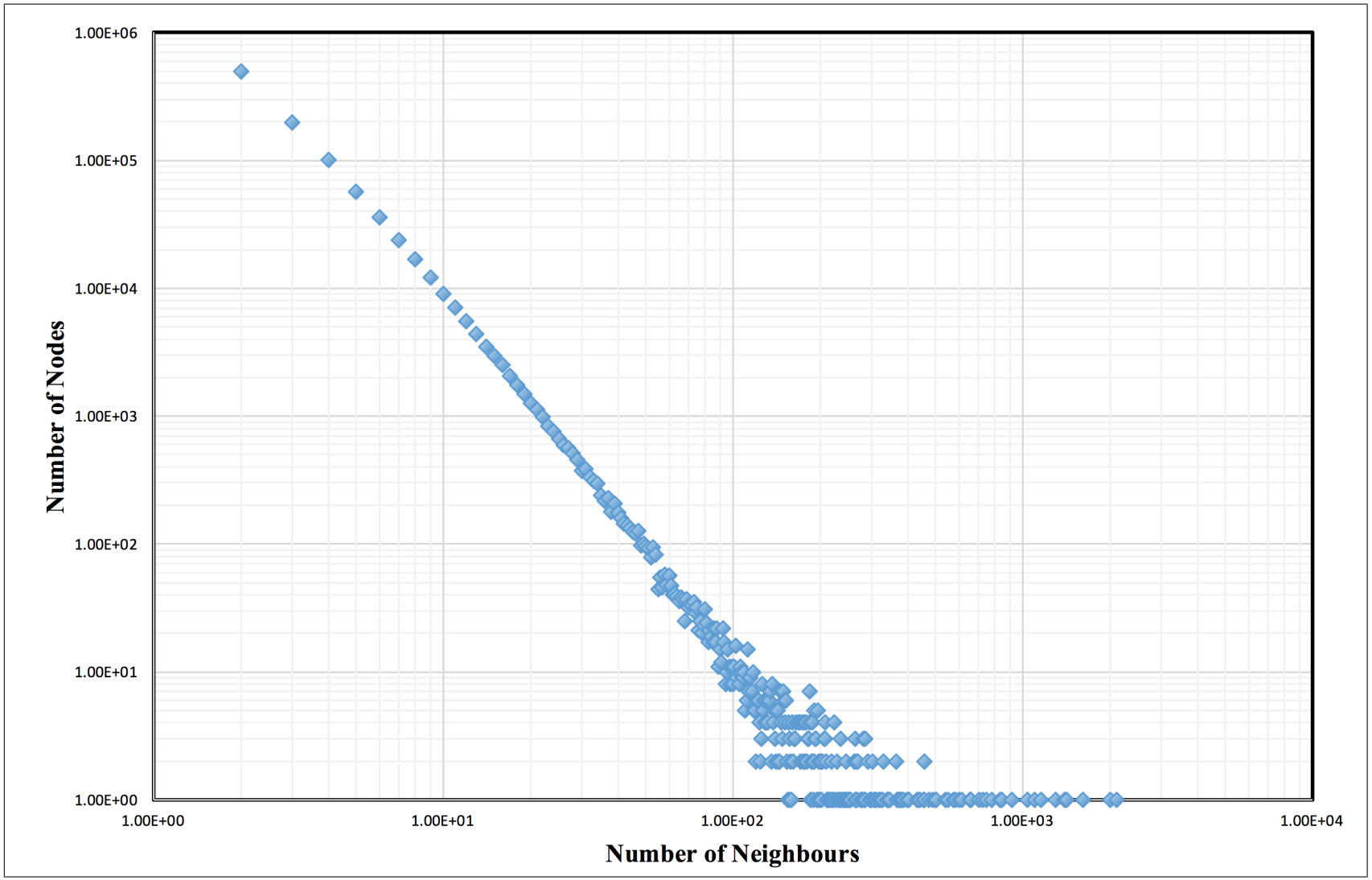

Figure 5.3: Number of Nodes Against Number of Neighbours (Random seed=8979323846, N=1 million, k=2).

Table 5.2: Alpha Multipliers' Patterns and Values

Set	Pattern	α_1	α_2	α_3	α_4	α_5
A	Fixed*	1.0	1.0	1.0	1.0	1.0
B	Descending	1.0	0.8	0.6	0.4	0.2
C	log_{10}	1.0	0.9030	0.7782	0.6021	0.3010
D	Descending	1.0	0.8	0.4	0.2	0.1
E	Division**	1.0	0.5	0.25	0.125	0.0625
F	Descending	0.8	0.5	0.3	0.1	0.0
G	Division	0.8	0.4	0.2	0.1	0.05
H	Descending ***	0.5	0.4	0.3	0.0	0.0
I	Division**	0.5	0.25	0.125	0.0625	0.03125
J	Ascending	0.2	0.4	0.6	0.8	1.0
K	Fixed	0.8	0.8	0.8	0.8	0.8
L	Fixed	0.6	0.6	0.6	0.6	0.6
M	Fixed	0.5	0.5	0.5	0.5	0.5
N	Fixed	0.4	0.4	0.4	0.4	0.4
O	Fixed	0.3	0.3	0.3	0.3	0.3

* Equivalent to BFS.
** Inspired by six degrees of separation techniques [50].
*** Used by Al-Dmour and Teahan for unstructured P2P networks [4].

from the originator. Descending pattern is the exact opposite of Ascending pattern.

Division pattern is when the alpha multipliers are half of the previous alpha multipliers. This results with five-tuples that keeps decreasing as the number of hops increases. The division pattern is inspired with the six degrees of separation studies [50]. The alpha multipliers in this pattern can also be summarised as follows, $\alpha_x = 2^{-x+1}$.The log_{10} pattern is when the number of alpha multipliers are decided with the decreasing number of 10 base log. ($log_{10}10 = 1$, $log_{10}8 = 0.9030$, $log_{10}6 = 0.7782$, $log_{10}4 = 0.6021$, $log_{10}2 = 0.3010$).

All experiments started with one initial query. The experiments are repeated three times with the change of the TTL parameter. The TTL are set to 5, 10, and 20. Query efficiency are calculated using Equation 2.1 that has been discussed thoroughly in Section 2.3.2. The maximum number of successful searches (the number of successful searches when all queries finished their TTL) for each iteration of the experiments are also being recorded.

5.6 Experimental Results

Table B.1, Table B.2, and Table B.3 list all the query efficiency (η) and the maximum number of successful searches for three different random seeds, 1234567890, 1415926535, and 8979323846 respectively. On each table, the top 10 query efficiency are marked with green, and the top 10 maximum successful searches are marked with cyan. It can be observed from results across the random seeds that the alpha multipliers that generate high number of query efficiency do not have a high number of successful searches, and vice versa. This suggests that at some settings, the query forwarding techniques have high efficiency with the expense of not finding many resources in the network. In resource discovery, having high query efficiency and high number of successful searches are considered equally important.

In order to find the best alpha multiplier values for unstructured P2P networks,

Table 5.3: Query Efficiency and Maximum Successful Searches Mean According to Random Seed

Alpha Multipliers					Query Efficiency and Max. Successful Searches Average (%)								
					Random Seed: 1234567890			Random Seed: 1415926535			Random Seed: 8979323846		
α_1	α_2	α_3	α_4	α_5	TTL5	TTL10	TTL20	TTL5	TTL10	TTL20	TTL5	TTL10	TTL20
1.0	1.0	1.0	1.0	1.0	27.25	50.65	NA	31.61	54.28	54.35	29.20	57.20	56.99
1.0	0.8	0.6	0.4	0.2	38.59	58.80	64.21	40.14	63.46	66.69	34.29	53.03	58.56
1.0	0.9030	0.7782	0.6021	0.3010	35.69	57.09	62.27	38.61	62.72	63.34	34.11	58.25	60.34
1.0	0.8	0.4	0.2	0.1	42.88	53.14	59.86	43.74	57.87	64.47	36.14	47.22	51.25
1.0	0.5	0.25	0.125	0.0625	45.31	50.57	54.25	47.16	52.53	57.20	40.69	40.50	43.07
0.8	0.5	0.3	0.1	0.0	45.48	50.38	52.60	47.63	52.41	56.41	36.68	40.08	40.95
0.8	0.4	0.2	0.1	0.05	49.07	50.14	52.50	47.86	52.60	55.10	37.08	37.10	42.35
0.5	0.4	0.3	0.0	0.0	45.34	49.14	50.77	46.37	50.85	54.09	39.75	37.95	38.88
0.5	0.25	0.125	0.0625	0.03125	47.80	50.63	50.37	50.30	51.15	52.68	50.38	44.83	39.95
0.2	0.4	0.6	0.8	1.0	23.86	57.88	59.08	29.45	60.41	61.00	23.99	56.89	57.98
0.8	0.8	0.8	0.8	0.8	26.58	57.33	57.79	32.79	58.04	58.05	28.84	58.26	58.37
0.6	0.6	0.6	0.6	0.6	29.86	60.16	62.14	35.76	62.97	63.82	30.43	57.45	59.77
0.5	0.5	0.5	0.5	0.5	33.12	59.77	63.93	35.31	62.76	64.72	32.17	54.35	59.70
0.4	0.4	0.4	0.4	0.4	37.22	56.82	63.68	40.01	61.11	66.96	33.17	49.32	56.04

both query efficiency (η) and the maximum number of successful searches importance are weighted the same. All of the values of query efficiency and the maximum successful searches are converted into percentage by dividing it with the maximum value of the parameter. For example, the maximum value of query efficiency for the random seed of 1234567890 is 220,228. Therefore, all query efficiency for that set of experiments is divided by 220,228 and multiplied by 100%. The results of converting the query efficiency and maximum successful searches into percentage are as shown in Table B.4 and Table B.5 respectively. The top average from each random seed experiments are marked with green.

After converting query efficiency and maximum successful searches into percentage, corresponding results from Table B.4 and Table B.5 are added and the mean is calculated. The average are shown in Table 5.4. Numbers that are marked with green colour are the ones with the highest result on both query efficiency and maximum successful searches. All three of them are with the TTL of 20. The five-tuple alpha multipliers corresponding to the top three are Set $\mathbf{B} = \{1.0, 0.8, 0.6, 0.4, 0.2\}$, Set $\mathbf{M} = \{0.5, 0.5, 0.5, 0.5, 0.5\}$, and Set $\mathbf{N} = \{0.4, 0.4, 0.4, 0.4, 0.4\}$ with combined efficiency of 63.15%, 62.78%, and 62.23% respectively. Graph for α-BFS for all of the alpha multipliers set (Set **A** to Set **N**) with TTL of 20 is shown in Figure 5.4.

5.7 Discussion and Conclusions

This chapter suggested two methods on reducing the query message forwarding for uninformed search resource discovery techniques that rely on flooding the network to find resources. The first method is by restricting the query message forwarding so that the resource discovery technique will only forward query a message to a fraction of the current node's neighbours. The second method is to avoid resending the same query message to neighbours that have seen the message.

The query message forwarding restrictions are achieved by implementing alpha mul-

Table 5.4: Query Efficiency and Maximum Successful Searches Mean

Alpha Multipliers					Query Efficiency η and Max. Successful Searches Average (%)		
α_1	α_2	α_3	α_4	α_5	TTL5	TTL10	TTL20
1.0	1.0	1.0	1.0	1.0	29.35	54.04	55.67
1.0	0.8	0.6	0.4	0.2	37.68	58.43	63.15
1.0	0.9030	0.7782	0.6021	0.3010	36.14	59.35	61.98
1.0	0.8	0.4	0.2	0.1	40.92	52.74	58.53
1.0	0.5	0.25	0.125	0.0625	44.39	47.87	51.51
0.8	0.5	0.3	0.1	0.0	43.26	47.62	49.99
0.8	0.4	0.2	0.1	0.05	44.67	46.61	49.98
0.5	0.4	0.3	0.0	0.0	43.82	45.98	47.91
0.5	0.25	0.125	0.0625	0.03125	49.49	48.87	47.67
0.2	0.4	0.6	0.8	1.0	25.77	58.39	59.35
0.8	0.8	0.8	0.8	0.8	29.40	57.88	58.07
0.6	0.6	0.6	0.6	0.6	32.02	60.19	61.91
0.5	0.5	0.5	0.5	0.5	33.53	58.96	62.78
0.4	0.4	0.4	0.4	0.4	36.80	55.75	62.23
0.3	0.3	0.3	0.3	0.3	39.04	51.29	58.22

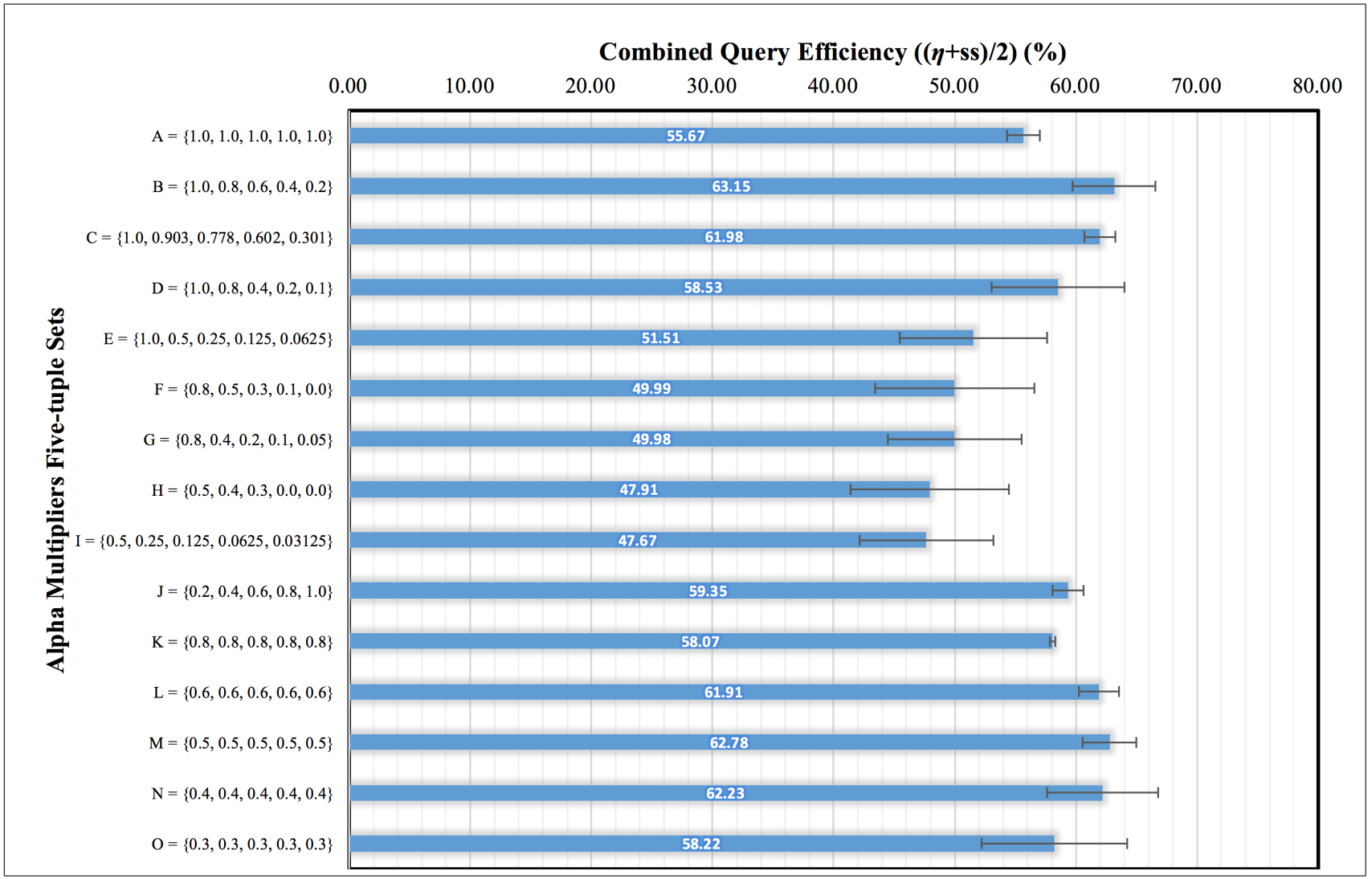

Figure 5.4: Combined Query Efficiency ($(\eta+ss)/2$) (N=1 million, k=2, TTL=20).

tipliers, five-tuple numbers between 0.0 to 1.0. Number of query message replication and forwarding can be controlled at the first five hops of the query. Query message can be forwarded but will not be replicated in the subsequent hops.

The restricted random walk with null exception is a change in the code to return a `null` value if there is no neighbouring node that has never seen the query message. Once the node received the `null` exception, it will not forward the query message. This method is to reduce unnecessary query message forwards. These two methods, α multipliers and restricted random walk with null exception is still considered within uninformed search resource discovery technique because they do not require the nodes to store any information regarding the search.

Several values of alpha multipliers are tested to find out the best option for unstructured P2P network. The alpha multipliers were set as either fixed, descending order, ascending order, logarithmic, and division. All these multipliers are then tested onto three sets of random seeds, and three values of TTL in order to find the best parameters for resource discovery in unstructured P2P networks. The results are shown in two type of measurements, they are query efficiency (η) and maximum successful searches. Results of the experiments show that set of alpha multipliers with high query efficiency do not have as much successful searches as the query that has lower query efficiency.

Considering that both measurements are equally important in resource discovery, each experiment results are converted into percentage values. These values are added together and averaged to come out with a single measurement (combination of query efficiency and number of maximum successful searches). Combination of alpha multipliers that return the top three results are Set $\mathbf{B} = \{1.0, 0.8, 0.6, 0.4, 0.2\}$, Set $\mathbf{M} = \{0.5, 0.5, 0.5, 0.5, 0.5\}$, and Set $\mathbf{N} = \{0.4, 0.4, 0.4, 0.4, 0.4\}$ with combined efficiency of 63.15%, 62.78%, and 62.23% respectively. All three combinations shows their best results with TTL of 20.

From the experiments, it is known that the BFS technique that floods the network

has the maximum number of successful searches. Nevertheless, the technique has the worst query efficiency compared to any other alpha multipliers combination. BFS resource discovery technique has a combined efficiency results of 55.67%. It is observed that the best combination of alpha multipliers, Set **B** has increased the combined efficiency of BFS by 7.48%.

6 Lightweight Blackboard Resource Discovery Mechanism

Chapter Summary

This chapter discusses the Blackboard Resource Discovery Mechanism (BRDM). Some issues regarding the resource discovery technique are also listed and explained. The main issue, network cost, are highlighted. Network cost calculation methods are also discussed in this chapter. The solutions to the issues are then discussed leading to a new resource discovery technique. The new lightweight technique is called lightweight BRDM (LBRDM).

Summary of Each Section

Introduction	:	Introduction of the chapter
BRDM Overview	:	Overview of the blackboard resource discovery mechanism. Discusses the main idea, characteristics and query message handling of the resource discovery technique.
BRDM Issues	:	Addresses the issues of previous researches in BRDM: Small simulation environment, unrecommended lists type I error, and high network cost of unsuccessful searches.
Improving BRDM: Foundations of Lightweight BRDM	:	The solution of the issues listed in previous section are listed. Also presents the calculation of the resource discovery costs to support the argument.
Lightweight BRDM	:	This section discussed the approach taken in order to reduce the network communication cost of BRDM.
Experimental Setup	:	Describes network topology simulation setup and query behaviours setup that are required for the experiments.
Experimental Results	:	All experimental steps and results for LBRDM and BRDM are listed.

6.1 Introduction

The Blackboard Resource Discovery Mechanism (BRDM) is a technique used in grid computing to find resources. The technique was first coined by Al-Dmour and Teahan [4], and is used in the enhanced ParCop [3]. In BRDM, all nodes in the P2P network

has two blackboards to store information regarding the surround neighbouring nodes.

The blackboard mechanism is based on research in the field of artificial intelligence [5]. In BRDM, the mechanism is used to keep track of recommended and unrecommended neighbours for query message forwarding. The recommended and unrecommended neighbour lists are independent of each other.

6.2 BRDM Overview

In BRDM, the search starts with one query message, mes, from the originator node, n_o. The search query is then forwarded to a fraction of its adjacent neighbours. Upon receiving the message, the neighbouring nodes that received the query messages will then forward it to a fraction of their neighbours. Each time the messages are cloned and forwarded, the message's Time-To-Live, TTL, is reduced by one. This action of forwarding will be done recursively until the message's TTL has expired [5].

In the early stages of searching using the BRDM technique, the network is flooded with query messages. The recommended and unrecommended lists are empty. Once the node forwarded the query messages, it will wait for the reply from the neighbouring nodes that it has already sent the query messages to. If the neighbouring node finds the source that the search intended, or the neighbouring node knows a path towards the source, the neighbouring node will be added onto the recommended list located in the forwarding node [5].

On the other hand, if after the search TTL has depleted, but still the neighbouring node could not return a successful searches, the neighbouring node will be added to the unrecommended list. The longer BRDM is run on the network, the more that it can learn from it. The size of recommended and unrecommended list will increase over time. Subsequently, when BRDM receives a new query message, it will then choose neighbouring nodes that appears in its own recommended list. If there is no recommended list, BRDM will look into the unrecommended list. The node will then forward the

query message to its neighbouring nodes that are not in the unrecommended list [5].

The more nodes in the recommended and unrecommended lists, the more intelligent the BRDM technique will be. The number of successful searches will increase over time, without the need to forward as much query messages as in the earlier stages. This demonstrates a learning effect, where the technique learns about its surrounding and intelligently decides where to and where not to forward [5].

6.3 BRDM Issues

BRDM has been shown by Al-Dmour and Teahan to produce good results compared to other resource discovery mechanisms [5]. Nonetheless, there are several issues in BRDM that need to be addressed. These include a small simulation environment, unrecommended list type I errors being produced, and high network costs for unsuccessful searches.

In order to explore the BRDM technique and develop improvements, the network has been set as recommended in the literature published in International Journal of Digital Information and Wireless Communications (IJDIWC) entitled "Implementation of Resource Discovery Mechanisms on PeerSim: Enabling up to One Million Nodes P2P Simulation" [43]. The general simulation parameters that were used are described in the following paragraphs.

Let $L(n_x)$ be the set of nodes to which a node n_x is connected. $l(n_x)$ is the set of the nodes to which the query will be forwarded from node n_x. The association between $L(n_x)$ and $l(n_x)$ are as shown in the following equation:

$$l\left(n_x\right) \subseteq L\left(n_x\right). \tag{6.1}$$

Let n_o be where the query message that originates from. F_x is the number of query messages that have been forwarded by the x resource discovery technique. The number

of query messages forwarded on each step l are equal to the number of neighbouring nodes L on each step. Therefore, the number of query messages forwarded for a TTL of 5 can be stated as follows:

$$F_x = l\left(l\left(l\left(l\left(l\left(n_o\right)\right)\right)\right)\right).$$

The number of query messages forwarded for multiple TTL can be simplified as followed:

$$F_x = l_{TTL} \circ l_{TTL-1} \circ l_{TTL-2} \circ ... \circ l\left(n_o\right) \tag{6.2}$$

$$= \prod_{0}^{TTL} l\left(n_o\right). \tag{6.3}$$

As shown in the Equation 6.3, it can be observed that the number of query message forwarding, QF, increases geometrically. Clearly, the amount of query message forwarding can be reduced significantly if the number of messages that are forwarded is reduced at each step of the resource discovery.

6.3.1 Small Simulation Environment

The comparison for the BRDM was done using a simulator that placed the nodes in 100 by 100 plots. Therefore at most, only 10,000 nodes can be tested. In order to simulate a real life P2P environment, this resource discovery mechanism needs to be tested with a larger network. In order to clear up this issue, BRDM and several other resource discovery mechanism were implemented on PeerSim so that they could be tested with up to one million simulated unstructured P2P nodes [42, 43]. The small number of nodes used for the previous BRDM simulations [4] was because of the limitation of computing

power that was available at the time the work was published which restricted the size of the networks that were used to simulate the resource discovery techniques that have exponential growth.

6.3.2 Unrecommended List Type I Error

Al-Dmour and Teahan's technique utilises two blackboards on each node in the P2P network. One blackboard, $N1$ lists all of the recommended neighbouring nodes, and the other blackboard, $N2$ lists all of the unrecommended nodes. As shown in the equations above, reducing message forwarding on each step will have a significant reduction of the whole query message forwarding, QF. The unrecommended list is an intelligent way of reducing query message forwarding on each stage [4].

All neighbouring nodes that are included in the unrecommended list will not be sent any query message. Nevertheless, in the early steps of message forwarding by BRDM, not all neighbouring nodes are forwarded with the message. Thus, the query message might not find the results, even though the resources might be nearby. The neighbouring nodes without successful hits will be included in $N2$, and would not be forwarded with any query message. This error is called a Type I error.

6.3.3 High Network Cost for Unsuccessful Searches

Intelligent breadth first search (Int-BFS) [46] increases the probability of choosing a neighbour based on the neighbour's action of forwarding a message. Adaptive probabilistic search (APS) [106] increases the probability of choosing a neighbour based on whether the neighbour returns any successful searches, and vice versa. In contrast, BRDM puts neighbouring nodes in the recommended or unrecommended lists based on successful searches [4].

As shown in Equation 6.3, the number of query messages that are forwarded is $QF = \prod_0^{TTL} l\,(n_o)$. Let ss be the number of query messages that return successful

searches, and us be the number of query messages that return unsuccessful searches. Upon any successful or unsuccessful search, the query message needs to traverse through all nodes that it has visited and inform the nodes regarding the status of the search. Similar to forwarding query messages, traversing backward, B, would also cost the network some network resource. The network cost for informing all previous nodes regarding both successful (B_s) and unsuccessful searches (B_u) can be equated as follows:

$$B_s = \sum_{0}^{QF} ss \times number\,of\,nodes\,traversed \tag{6.4}$$

$$B_u = \sum_{0}^{QF} us \times TTL. \tag{6.5}$$

Note that in the Equations 6.4 and 6.5, successful searches will travel up until they find the resources, while unsuccessful searches have to traverse in the nodes up till the TTL is depleted before they can report to the originator. Given that all queries forwarded should be either successful or unsuccessful, and the length that the message needs to go back to inform the originator is the same as the length that it took to go forward, relations between F, B_s, and B_u can be formulated as follows:

$$F = B_s + B_u. \tag{6.6}$$

Therefore, in resource discovery techniques that require queries to return the status of the search upon the end of TTL, such as APS [106] and BRDM [4], the total number of messages travelling, $T_{APS/BRDM}$ inside the P2P system are as follows:

$$\begin{aligned} T_{APS/BRDM} &= F_{APS/BRDM} + B_{APS/BRDM} \\ \because B_{APS/BRDM} &= B_s + B_u \end{aligned}$$

Using Equation 6.6, substitute the value of $B_s + B_u$:

$$B_{APS/BRDM} = F_{APS/BRDM} \tag{6.7}$$

$$\begin{aligned} \therefore T_{APS/BRDM} &= 2\left(F_{APS/BRDM}\right) \\ &= 2\left[\prod_{0}^{TTL} l\left(n_o\right)\right]. \end{aligned} \tag{6.8}$$

In a uninformed search resource discovery such as BFS, the amount of query message forwarding can overwhelm to the whole system. The backward traversing that would also cost the network as much as the query message forwarding is also going to make the situation even worse. The total network cost would be double the uninformed search.

6.4 Improving BRDM: Foundations of Lightweight BRDM

As mentioned above, there are three things that need to be addressed regarding BRDM; namely, small simulation environment, unrecommended lists type I error, and high network cost for unsuccessful searches. Two enhancements have been devised in order to tackle the problems stated in the previous section. These are now described.

6.4.1 Increasing the Simulation Environment Size

Al-Dmour and Teahan have compared resource discovery techniques for unstructured P2P networks [4]. Nevertheless, possibly due to the computing power restrictions at the time, the simulations were done for only 1000 nodes. In order to simulate the resource discovery technique to be closer to real life unstructured P2P networks, the techniques were implemented on a highly scalable PeerSim [22, 71]. The simulation for this research was done using the maximum settings, that is usually up to one million nodes.

6.4.2 Eliminating Type I Error

Type I error occurs when the algorithm produces false positives. In BRDM, there are two blackboards on each node, recommended nodes $N1$ and unrecommended nodes $N2$. If a query message successfully searches for the resource, it will return its finding to the whole nodes that it traverses through. The findings will be updated in each node's $N1$. There is no possibility for a type I or type II error for this blackboard update.

Notwithstanding this, if a query message did not find the resource that it has been searching for, it will also update all the nodes that the message traverses to. Nonetheless, in BRDM, not all neighbours are selected to be forwarded the query message. Therefore, there is a possibility that a node neighbouring to the resource was not find because the message was not forwarded to the neighbour. By updating the unsuccessful searches back to the originator and all the nodes the message traverses to, the node will be included in $N2$. Being included in $N2$ means that it will not be selected for future resource discovery message forwarding. This error is called type I error, a false positive.

The easiest way to eliminate this type I error is to disable the unsuccessful searches blackboard update. Eliminating the update means that each node will only need one blackboard list, that is the recommended list $N1$.

6.4.3 Increasing BRDM Query Efficiency

As discussed in Section 2.3.2, Lin & Wang [57] propose query efficiency, η (Equation 2.1) as a way to evaluate the efficiency of query techniques. However the metrics proposed above does not take into consideration the number of feedback, which equally important because the feedbacks use the network bandwidth and ultimately may flood the network. A new metric, query efficiency with feedback, η^* (Equation 2.2) is proposed to take into consideration the number of feedbacks in calculating the efficiency of resource discovery techniques. Substituting the number of messages sent (Equation 6.3) into η^* (Equation

2.2) will produce following equation:

$$\begin{aligned} \eta^* &= \frac{QueryHits}{\frac{QueryMessagesForward+QueryMessagesFeedback}{N}} \\ &= \frac{QueryHits}{\frac{\left[\prod_0^{TTL} l(n_o)\right]+B_s+B_u}{N}} \end{aligned} \tag{6.9}$$

where B_s and B_u are the feedbacks for successful and unsuccessful searches respectively. n_o is the origin node of the query and N is the number of nodes in the network.

Number of query messages sent and feedbacks differs between resource discovery techniques. Obtaining a balance between query replication and its feedback is crucial to make sure the resource discovery technique remains efficient. In BRDM, all queries sent would have to generate a feedback query towards its originator, making the number of query messages sent and feedback to be the same (Equation 6.7). Substituting $B_s + B_u = \prod_0^{TTL} l\left(n_o\right)$ to Equation 6.9, the η^* calculation for BRDM is as follows:.

$$\eta^* = \frac{QueryHits}{\frac{2\left[\prod_0^{TTL} l(n_o)\right]}{N}}$$

Resource discovery techniques that require all queries to send feedbacks have equal number of query messages sent and feedbacks. The number of query messages sent is equal to or bigger than the positive feedbacks ($\prod_0^{TTL} l\left(n_o\right) \geqslant B_s$). Therefore, by eliminating the need to give feedbacks by unsuccessful query, the query efficiency of the resource discovery technique should increase, if not equal to techniques that requires all queries to send feedbacks.

$$
\begin{aligned}
\prod_{0}^{TTL} l\left(n_o\right) &\geqslant B_s \\
\therefore \frac{QueryHits}{\frac{2\left[\prod_0^{TTL} l(n_o)\right]}{N}} &\leqslant \frac{QueryHits}{\frac{\left[\prod_0^{TTL} l(n_o)\right]+B_s}{N}} \\
\eta^*_{AF} &\leqslant \eta^*_{SoF}
\end{aligned}
$$

where η_{AF} and η_{SoF} are query efficiency for resource discovery technique that require all feedbacks and techniques that only require successful feedbacks respectively.

Therefore, it has been proven that techniques that require all successful and unsuccessful messages queries to return their finding through the network will incur a greater cost and less query efficiency when compared to techniques that only requires successful searches to return their findings. In pursuance of fulfilling the lightweight resource discovery that is based on BRDM, the need to return queries that are unsuccessful searches will be eliminated. It will significantly reduce the cost of network of BRDM, which results in message reduction and higher query efficiency. The new lightweight technique, Lightweight Blackboard Resource Discovery Mechanism (LBRDM), utilises only one blackboard on each node, recommended nodes $N1$ compared to two blackboards on BRDM, recommended nodes $N1$ and unrecommended nodes $N2$.

6.5 Lightweight BRDM

With reference to Section 6.4.2 and 6.4.3, there are several things that are needed to be done. In order to eliminate the type I error (issue in Section 6.4.2) of the unrecommended nodes $N2$, there are two ways to handle it. The first method is by eliminating the $N2$ altogether. The second method is to forward query messages to nodes that are in $N2$ even though it is listed as unrecommended.

Section 6.4.3 discusses the network cost of resource discovery techniques that require

both successful and unsuccessful searches to return their search results η^*_{AF}. The number of query messages going forward and backward is very substantial, and might overwhelm the network. As a result of η^*_{AF} network cost is almost twice the η^*_{SoF}, it might be a good idea not to use the return query of the unsuccessful searches. Ultimately, disabling the unrecommended nodes $N2$ blackboard from the BRDM technique would solve two issues of the BRDM, the type I error and the high network cost issue.

6.6 Experimental Setup

In order to compare the experiments performed on the BFS and α-BFS, the BRDM and LBRDM techniques have also been tested according to the same topology setup parameters: one million nodes distributed and wired using the scale free Barabási-Albert model; undirected connection; k variable of two; and is run of 20 cycles. The nodes and neighbours distribution for the random seed of 1234567890, 1415926535, and 8979323846 are shown in Figure 5.1, Figure 5.2, and Figure 5.3 respectively. The experimental setup parameters are shown in Table 5.1.

The query behaviour setup are one initial query and the TTL of 20. Three of the best five-tuple alpha multipliers discovered in Section 5 are used in all of the experiments for BRDM and LBRDM. The three alpha multipliers set are Set $\mathbf{B}$ = {1.0, 0.8, 0.6, 0.4, 0.2}, Set $\mathbf{M}$ = {0.5, 0.5, 0.5, 0.5, 0.5}, and Set $\mathbf{N}$ = {0.4, 0.4, 0.4, 0.4, 0.4}, as shown in Table 5.2. Query efficiency (η) are calculated using Equation 2.1 that has been discussed thoroughly in Section 2.3.2. Query efficiency with feedback (η^*) (Equation 6.9) that was discussed in Section 6.4.3 were calculated to find the efficiency of informed resource discovery techniques. The maximum number of successful searches (the number of successful searches when all queries finished their TTL) for each iteration of the experiments are also being recorded.

Table 6.1: Average Query Efficiencies (η and $\eta*$) and Successful Searches (ss)

Techniques	Alpha Multipliers	η & ss	η^* & ss
BRDM	Set **B**	89.30	70.65
	Set **M**	80.40	62.84
	Set **N**	68.15	47.60
LBRDM	Set **B**	84.12	86.31
	Set **M**	84.01	86.28
	Set **N**	69.02	70.86

6.7 Experimental Results

Table B.7 shows the results for BRDM and LBRDM resource discovery techniques when tested on three different random seed and three different sets of alpha multipliers. The results are then averaged across the random seed and then are shown in Table 6.1. It can be observed that in two out of three sets of alpha multipliers (Set **M** and **N**), the LBRDM has a better combined query efficiency (η & ss) than BRDM. This shows that the LBRDM technique can maintain good results even though it only require one blackboard on each node instead of two blackboards in BRDM.

In combined query efficiency star that took into consideration of network communication cost, η^* & ss, the LBRDM has a higher percentage of efficiency compared to the BRDM in all three sets of alpha multiplier being experimented. This result is expected because the LBRDM is designed to use less network resources by eliminating the requirements to return unsuccessful results to the originator of the query. The comparison chart for BRDM and LBRDM in combined query efficiency and query efficiency star metrics are shown in Figure 6.1 and 6.2 respectively.

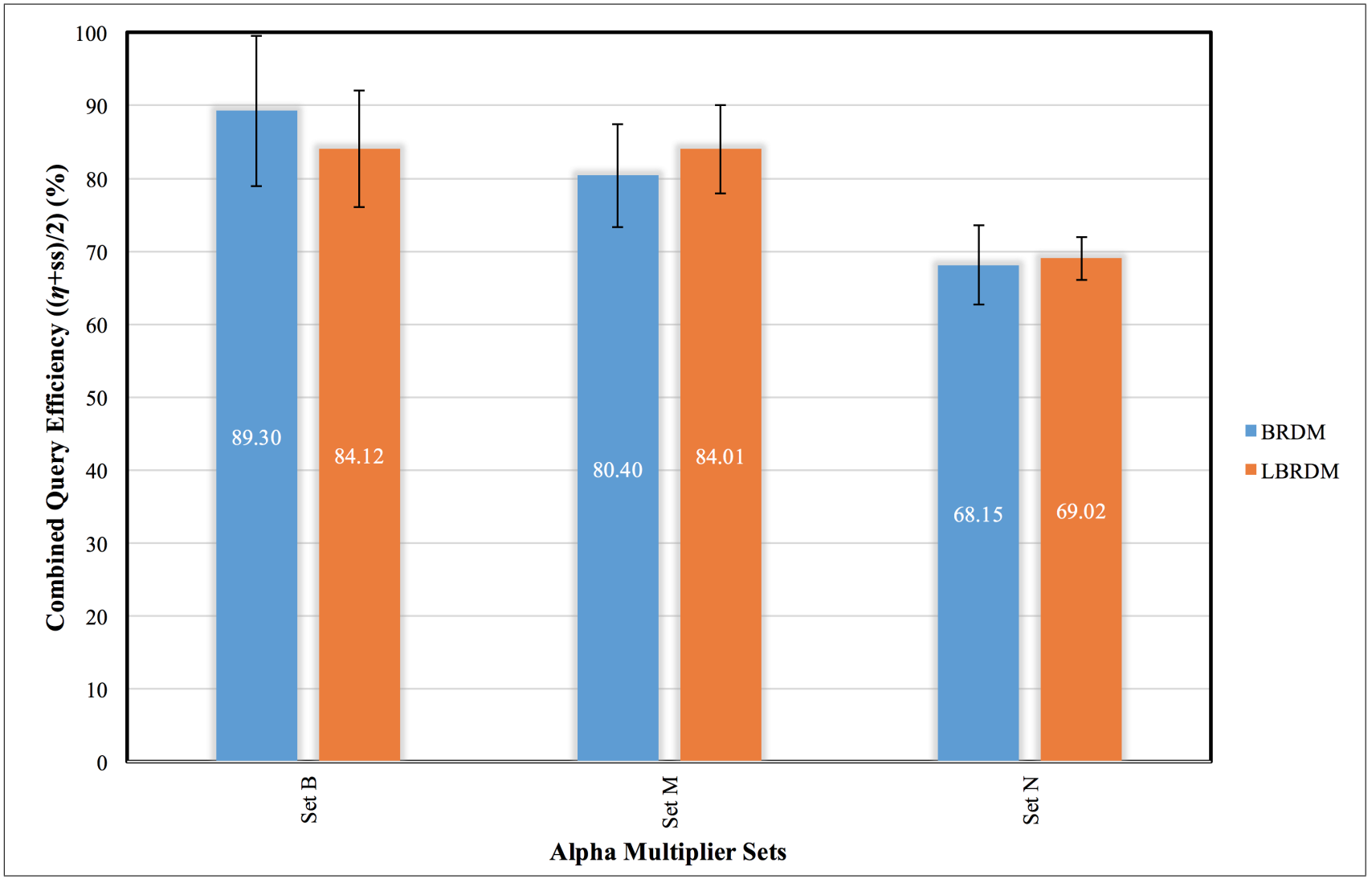

Figure 6.1: Combined Query Efficiency ($(\eta+ss)/2$) (N=1 million, k=2, TTL=20).

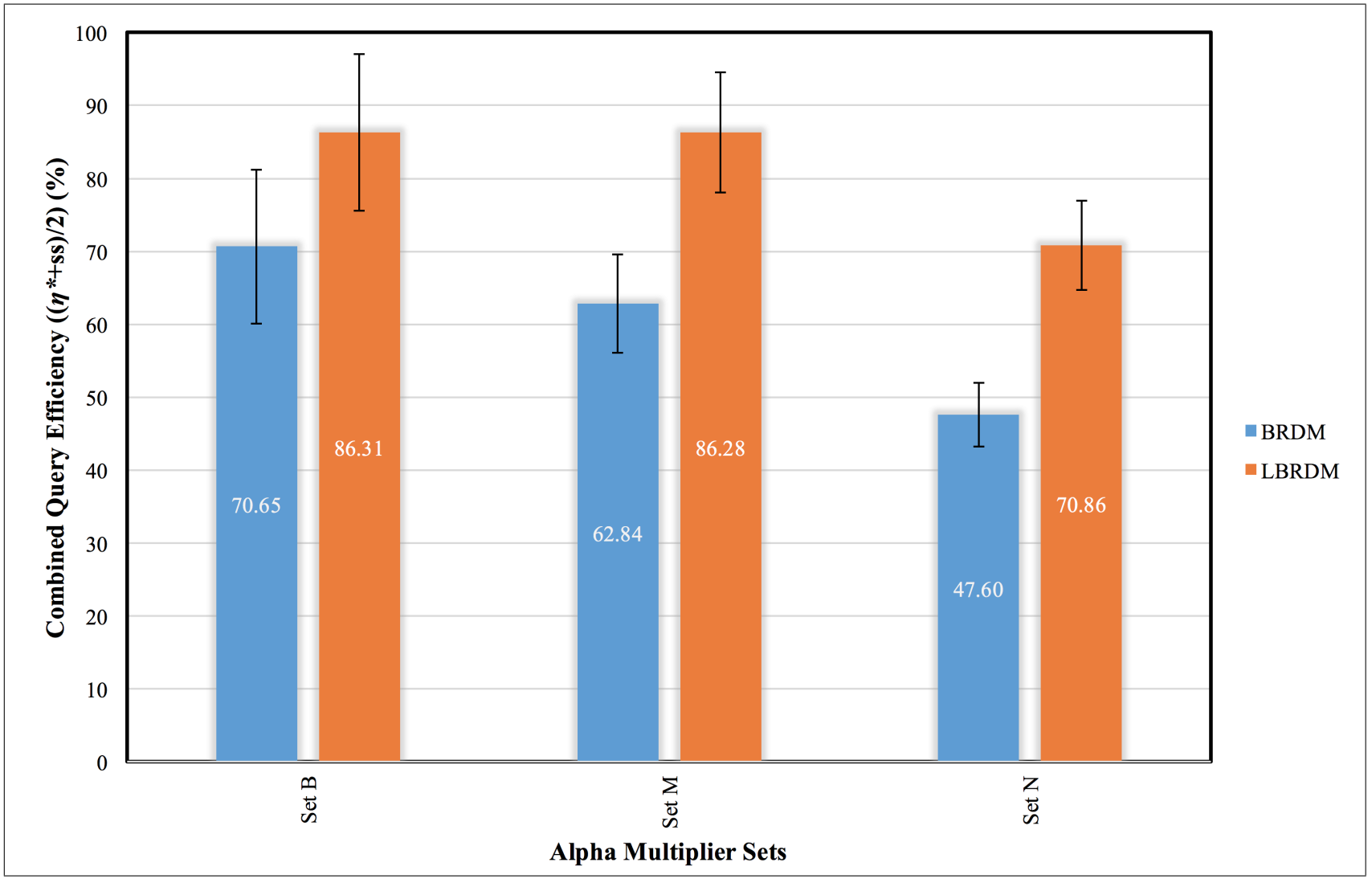

Figure 6.2: Combined Query Efficiency Star ($(\eta^*+ss)/2$) (N=1 million, k=2, TTL=20).

6.8 Discussion and Conclusions

This chapter listed several problems that has been identified on the tests and experiments done with previous resource discovery techniques. The first problem is regarding small simulation size of the experiments that have been done, which might not truly reflect the real life P2P networks. This research has implemented the resource discovery techniques on a highly scalable P2P simulator, PeerSim [71]. The implementation was discussed in detail in Section 3 of this dissertation.

The chapter continues with the discussion regarding the approaches taken to eliminate type I error and reduce the number of network communication cost on informed resource discovery techniques. Type I error happens when the resource discovery technique demerits nodes that did not return with successful searches, even though not all nodes that are connected to the node being degraded have been previously checked. This happens when TTL is implemented, and the resource discovery technique only selects a fraction of its neighbouring node to forward the query messages with.

This study also identified the problem of some informed resource discovery techniques that requires all forwarded query messages to return their findings whether it is successful or not. Return of successful searches to the nodes that the query message previously traversed on is useful. Nevertheless returning information of unsuccessful searches, where the amount of unsuccessful searches is bigger than successful searches might be unnecessary and a waste of network communication resource.

This study has developed a new lightweight BRDM that requires the technique to only return successful searches information instead of returning both successful and unsuccessful searches. The new technique being developed is lightweight, thus it is named as lightweight BRDM, or LBRDM. Both the BRDM and LBRDM techniques were implemented onto PeerSim (refer Section 3). The experimental setup for the informed resource discovery techniques was set to be the same as what have been found in the

uninformed resource discovery experiments. Three sets of five-tuple alpha multipliers that returned the best results for the uninformed resource discovery experiments are used in these experiments.

From the experiments, it is found that even though the LBRDM does not have the same amount of information as the BRDM, the LBRDM was able to have combined query efficiency that is comparable to BRDM. This shows that some of the information being shared and distributed in BRDM is not as necessary as it was thought it was. The action to remove information communication and one of the blackboards on each node in the network do not degenerate the resource discovery efficiency.

7 Discussion & Conclusions

Chapter Summary

This chapter is divided to several sections: Discussion; Summary of Chapters; Contributions; Review of Aims and Objectives; Conclusions; and Future works. The discussion is where all the findings and observations during the whole study are discussed. It is then followed with summary of all the chapters in this dissertation. Contributions to the academic publications are also listed in the contributions section. Review of aims and objectives are also listed, right before the conclusion of this dissertation. Future works are also included at the end of this chapter.

Summary of Each Section

Introduction	:	Introduction of the chapter.
Discussion	:	This section contains the summary of all the discussions in this dissertation.
Summary of Chapters	:	Summaries of each Chapters in this dissertation are included in this section.
Contributions	:	Contributions from this dissertation are listed.
Research Limitations	:	Limitations of this research are listed.
Review of Aims and Objectives	:	This section lists all the main objectives of this dissertation and the outcomes from this dissertation.
Conclusions	:	This section contains the conclusions for all the research and experiments performed in this dissertation.
Future Works	:	All future work related to this dissertation is listed in this section.

7.1 Discussion

This study starts with the research motivation of reducing the network communication cost for resource discovery mechanisms for unstructured P2P network environments. This is because it is difficult to find resources in a large unstructured network, and users usually rely on techniques that flood the network with query message. For that reason, three main objectives of this research were: to design a lightweight resource discovery technique; implement it on a P2P simulator; and examines the effectiveness of the newly developed techniques.

Through the literature review, the characteristics of peer-to-peer networks were discussed and categorised according to its utilisation, overlay networks, and network topology. Research regarding resource discovery techniques were explored, and categorised the techniques into two types: uninformed search techniques and knowledge based techniques. This study also identified several problems with certain resource discovery techniques. The breadth-first search technique, is the best in the unstructured P2P networks in terms of finding the most amount of resources in the network but: with a high price of flooding the network with query messages.

In order to study how the resource discovery techniques find resources, the techniques needed to be implemented on a P2P network. Nevertheless, acquiring a big platform just to test P2P techniques would be too expensive. Therefore, the P2P techniques are implemented on PeerSim, a highly scalable P2P network simulator. PeerSim's forte, high scalability, was able to be achieved by programming it in object oriented approach. The object oriented approach has a large package, able to run many types of P2P experiments, such as: resource discovery; load balancing; and peer aggregation. Nonetheless, this vast library of simulation capability leads to a steep learning curve of the whole simulator ecosystem. Implementing a resource discovery technique on PeerSim requires a good programming capability.

To closely imitate the real unstructured P2P network, the network topology generator models need to follow certain rules, the simulated networks need to: follow the power law; unstructured; and are scale free. Several P2P network generators, namely, HOT, regular rooted tree, star, ring lattice, Watts-Strogatz, scale free Barabási-Albert, scale free Dorogovtsev-Mendes, and K-out were tested. Up of all eight topology generator models, only two models, scale free Barabási-Albert, and scale free Dorogovtsev-Mendes were able to fulfil the criteria that have been set. The scale free Barabási-Albert was finally selected for the simulations, this is due to the finding that the scale free Dorogovtsev-Mendes, at generating 1000 nodes has generated 2 nodes that can be considered as outliers from the power law.

Resource discovery techniques are divided into two categories: uninformed and informed searches. For the uninformed search, BFS, the most widely used resource discovery on unstructured P2P networks has been scrutinised. The flooding of the network with query messages deemed to be the main problem of the technique. The number of query message forwarding for BFS can be summarised as, $F_{BFS} = L\left(L\left(L\left(L\left(n_o\right)\right)\right)\right)$, or $F_{BFS} = \prod_0^{TTL} L\left(n_o\right)$, where $L(n_o)$ is the number of originator's neighbouring node. From the equation, it can be observed that number of query messages forwarding in BFS increases exponentially.

Two methods have been devised in order to overcome the problem. The first one was by implementing alpha multipliers, a method to restrict number of query message forwarding according to the number of hops that the query message has taken. The second one was to implement a regulation, not to forward the same message to neighbouring nodes that have seen the query. The implementation of alpha multipliers and the forwarding regulation on BFS is named α-BFS.

Multiple values of five-tuple alpha multipliers are tested to come out with the best three set of alpha multipliers. They are, Set $\mathbf{B} = \{1.0, 0.8, 0.6, 0.4, 0.2\}$, Set $\mathbf{M} = \{0.5, 0.5, 0.5, 0.5, 0.5\}$, and Set $\mathbf{N} = \{0.4, 0.4, 0.4, 0.4, 0.4\}$ with combined efficiency

of 63.15%, 62.78%, and 62.23% respectively. The BFS has a combined query efficiency of 55.67%. The experiments show that by implementing the Set **B**, **M**, and **N** alpha multipliers of the α-BFS, the combined query efficiency were increased 7.48%, 7.11%, and 6.56%.

For the category of informed resource discovery, the BRDM has been examined thoroughly. Several issues have been addressed, such as: small simulation environment; type I error in unrecommended lists; and high network communication cost needed to send the information of unsuccessful searches. This study has implemented the BRDM onto PeerSim to increase the number of network simulated for the experiments to 1 million nodes. The resource discovery query efficiency is calculated as follows: $\eta^* = \frac{QueryHits}{\left[\Pi_0^{TTL}\, l(n_o)+B_s+B_u\right]/N}$, where TTL is the time to live, $l(n_o)$ is the number of query message forwards, B_s and B_u are the number of message traversing in order to inform successful searches and unsuccessful searches respectively.

Acknowledging the problem, the requirement to return information of unsuccessful searches is dropped, as an approach to reduce network communication cost and at the same time increases the query efficiency. Subsequently, unrecommended blackboard lists, $N2$, are also dropped from the BRDM because of no information of unsuccessful search will be received in the technique. These actions were able to tackle the remaining two issues of BRDM: type I error in unrecommended lists; and high network communication cost. The withdrawal of unrecommended list is considered as changing the BRDM to be lightweight, thus the developed technique is name lightweight BRDM, or LBRDM.

The BRDM and LBRDM are experimented side-by-side using the same experimental setup. The combined query efficiency (η) of LBRDM shows a better result in two out of three alpha multiplier sets. The sets where the results of LBRDM is better than BRDM are Set **M**, and **N** of the alpha multipliers with BRDM: 80.40% and LBRDM: 84.01%, and BRDM: 68.15% and LBRDM: 69.02% combined query efficiency respectively. The

set where the BRDM outperformed the LBRDM is the Set **B**, where the BRDM with 89.30% and LBRDM with 84.12% combined query efficiency. Nevertheless the results show that the LBRDM does not suffer from any setback compared to BRDM even though it does not incorporate the use of information from the unsuccessful searches and the non-existent of one of the blackboard list ($N2$).

7.2 Summary of Chapters

This dissertation starts with the first chapter, containing the introduction to the whole study, enhanced resource discovery mechanism for unstructured peer-to-peer network environments. Motivation for the research is also written in this chapter. Three research objectives have been identified and all the study, research, experiments have been performed in order to fulfil these objectives. The chapter ends with a list of contributions towards published literature that came out from this study.

Chapter 2 contains the literature review of P2P networks and resource discovery. This chapter starts with an introduction and some history regarding the Internet and its usage, the emergence of P2P network and grid computing, followed with the resource discovery techniques in grid computing. Overview of the P2P networks, and the categorisation of the P2P networks are discussed. This chapter continues with a review of resource discovery mechanisms, where several types of resource discovery mechanisms are discussed and explained. The resource discovery evaluation techniques are also discussed at the end of this chapter.

The following chapter, Chapter 3, several P2P simulators have been discussed and compared in order to find the best simulator to run the experiments. The remaining of the chapter discusses the implementation of several resource discovery techniques being used in unstructured P2P networks in a P2P simulator called PeerSim. Resource discovery techniques that have been implemented are random walk, restricted random walk, breadth first search, intelligent breadth first search, adaptive probabilistic search,

depth first search and the blackboard resource discovery mechanism. Pseudocode of each resource discovery technique that was implemented on PeerSim are also shown. The code implemented for the BRDM resource discovery technique is presented and discussed at the end of this chapter.

Chapter 4 discusses the characteristics of several network topology models. The P2P network topology generator models discussed are as follows: HOT, regular rooted tree, star, ring lattice, Watts-Strogatz, scale free Barabási-Albert, scale free Dorogovtsev-Mendes, and K-out topology generators. The rules and conditions that need to be followed in order to correctly imitate real life P2P networks are also listed. The chapter ends with the selection of an unstructured P2P network topology generator to be used in all of the experiments beyond Chapter 4.

The subsequent chapter, Chapter 5, discussed regarding the Alpha breadth first search, an improved version of breadth-first search. It implements alpha multipliers, a set of 5 multipliers that dictate the number of message forwarding from each node. This chapter discusses the two main techniques that have been implemented on alpha breadth first search, that is alpha multipliers and restricted random walk with null exception. Both of the techniques are aimed at reducing message forwarding by eliminating unnecessary duplicate query messages from the network. Experiments were done to find the best set of alpha multipliers that are able to outperform the BFS resource discovery technique using the combined query efficiency metrics.

Chapter 6 discusses the Blackboard Resource Discovery Mechanism (BRDM). Some issues regarding the resource discovery technique are also listed and explained. The main issue, network cost, are highlighted. Network cost calculation methods are also discussed in this chapter. The solutions to the issues are then discussed leading to a new resource discovery technique. The new lightweight technique is called lightweight BRDM (LBRDM). The LBRDM is then compared to BRDM in sets of experiments, and the LBRDM shows almost no setbacks compared to the BRDM even though it uses

less information and resource in the resource discovery.

Chapter 7 is the final chapter of this dissertation. The summary of previous discussions in the dissertation are shown in this chapter. It also lists the contribution of this study towards the field of computer science and engineering, generally, and computer networks, specifically. The chapter also contains limitations of this research and review of aims and objectives of the research. The conclusion of the whole research is stated towards the end of this chapter, before closing the whole dissertation with future works that can be done in relation to this research.

7.3 Contributions

This research has manage to come out with several contributions towards the computer networks engineering knowledge. The two most significant contributions are the alpha multipliers and the prove that unsuccessful searches are unimportant. The five-tuple alpha multipliers are considered significant due to the fact that the query efficiency can be increased compared to flooding the unstructured P2P networks in order to find resources. Three sets of alpha multipliers with the best results were pointed out and can be used to reduce the flooding of the networks with query messages.

Some informed resource discovery mechanisms rely on information regarding unsuccessful searches. The importance of the information is trivial. There is also possibility that the information may generate a Type I error that will reduce the resource discovery query efficiency. Experiments conducted in this research have proven that without the information of unsuccessful searches, the resource discovery mechanism can still produce an almost equivalent combined query efficiency. This research has proven that the exchange of information regarding unsuccessful searches to be unimportant and negligible. By removing the information, resource discovery can be more efficient without the unnecessary information exchange and storage. This results to a less usage of computing and network resources of the P2P network.

Other contributions of this research are the identification and classification of the resource discovery mechanism to be used in unstructured P2P networks. This research also scrutinised several P2P simulators and found out the best simulator to experiment with very large scale P2P network. Simulated network topology generators were also tested and the most consistent and most realistic (closely imitate real-life P2P network) topology generator has been found. Finally, this research also contributed to the implementation of several resource discovery mechanism onto PeerSim, where some of the mechanisms required some change in the main search protocol class of the simulator.

7.4 Research Limitations

Resource discovery techniques can be tested and experimented using mathematical models, on actual P2P systems, and on P2P simulators. All experiments for the resource discovery in this thesis were done on a P2P simulator named PeerSim. There are several limitations of doing experiments using PeerSim simulators such as: the resource discovery technique can be tested on limited number of Open Systems Interconnection (OSI) layers; limited amount of P2P nodes based on the memory capacity of the computer.

The OSI layers are important to show how users can communicate with each other using the network. The OSI layer model has seven layers, namely: application; presentation; session; transport; network; data-link; and physical layer. P2P research on actual P2P system would have to be tested on all seven layers. By utilising PeerSim for all of the experiments in this research, the simulator only simulates the network and physical layer of the OSI model. Nevertheless, the two layers simulated by PeerSim are the layers that are crucial for resource discovery.

The second limitation for this research is the number of simulated nodes in PeerSim is reliant to the capacity of the memory of the computer running the simulation. PeerSim is developed using Java programming language, therefore all simulations and experiments for this research were done using Java Virtual Machine (JVM) with memory allocation

of 16GB. There are some experiments that were not completed, for example, one of the experiment for the BFS only managed to complete the first out of 20 cycles being set for the simulation. This limitation can be overcome with future technologies and the increment of memory capacity.

7.5 Review of Aims and Objectives

There are three main objectives of this research. The review of the objectives are as shown in Table 7.1

Table 7.1: Review of Aims and Objectives

Research Objectives	Final Outcomes
To design a lightweight resource discovery technique suitable to be used in unstructured P2P networks.	Two lightweight resource discovery techniques were designed for unstructured P2P networks.
To implement resource discovery techniques onto a P2P simulator.	Several resource discovery techniques were implemented onto PeerSim. Two newly designed resource discovery techniques are also implemented onto the simulator.
To find out the effectiveness of the new resource discovery techniques by comparing it with existing approaches.	Experiments have been done on the new resource discovery techniques and compared to existing approaches.

7.6 Conclusions

The α-BFS and LBRDM were able to reduce network communication cost of BFS and BRDM respectively. The α-BFS was able to do so by controlling the number of query message forwarding. Apart from low network communication cost, the technique was able to generate a better (approximately twice) percentage of successful searches when

compared to the number of message forwarding. The LBRDM was able to reduce the network communication cost by eliminating the need for the query message to return unnecessary information, for the case of BRDM, is the information of unsuccessful searches.

The implementation of α-multipliers can be easily done to other uninformed search resource discovery techniques, as it only requires the number of hops that the query message has taken. For the LBRDM approach of eliminating unnecessary information and type I error, there are other resource discovery techniques that requires such information. Using the same approach towards the techniques might be able to reduce network communication cost of those techniques.

7.7 Future Work

In this research, several resource discovery techniques are examined and tested. Two newly developed resource discovery techniques are also designed, and tested on a simulator. In BRDM and LBRDM, the number of forwarded query messages is determined based on the number of hops. After the number has been obtained, there are several ways to divide the value. In BRDM, the number is devised so that it will forward to nodes in the recommended lists, and the remaining to the nodes that are unlisted (neither listed in the recommended nor the unrecommended list).

If the number of nodes in the recommended lists is bigger than the number of forwarded query messages it will only forward to one of its unlisted neighbour. This affects the number of resources found, because the protocol does not encourage the query message to go to new and unknown nodes. The handling of the number of forwarded query messages can be changed to develop a new resource discovery technique, that will try to find resources in new and unknown nodes. For example, instead of setting the number of alpha multipliers to a set of fixed numbers, the alpha multipliers can be set to have variable numbers depending on the condition of the query or the nodes, such as,

has the query message found resources or based on how many percent of the adjacent node being forwarded the message.

Another variable that should be experimented upon are the time-to-live (TTL). This research has proven that the longer the TTL, the higher the query efficiency of the resource discovery technique. Nonetheless, the longer the TTL, the higher the number of query being sent, which might lead to unnecessary query forwarding. A mechanism should be devised to alter the TTL of each query message. For example, if the query message has found a resource, the TTL can be either elongated or shortened. Justification of elongating the TTL is to find more resources in adjacent nodes, because finding one resource in a node does not mean that its adjacent nodes do not have the resource that the query searched for. Justification of the latter is no other than to decrease network communication cost and increase the query efficiency.

All of the techniques mentioned in this dissertation, the BFS, α-BFS, BRDM, and LBRDM are the techniques used in unstructured P2P networks. Considering that there are other P2P network topologies aside from unstructured, such as: the structured; super peer; and hybrid, implementing the approaches done in α-BFS and LBRDM may or may not be effective. Nonetheless, there is no harm in trying it in a simulator. In order to make an easy adaptation towards other types of topologies, the adaptation work should focus on hybrid versions of unstructured P2P networks. Another approach is by implementing the α-BFS onto all fields that utilise the BFS as their search technique.

Aside from resource discovery in P2P networks based on its physical topologies alone, there are also overlay networks, where the nodes are connected logically. Logical networks opens up to a different type of resource discovery, where nodes can connect and disconnect with each other based on the resource discovery techniques being used. Both the physical topologies and the overlay topologies are interesting choices to implement the approaches taken in this dissertation, in the hope of improving successful searches and reduce the network communication costs.

There are two ways to implement the resource discovery on overlay networks. The first method is by implementing the resource discovery technique onto P2P simulator that simulate more OSI layers than PeerSim. The second method is by implementing the resource discovery technique onto real-life P2P system. Implementing to the real-life P2P system seems to be a better choice because it will lead to a real-life implementation of the resource discovery mechanisms being developed in this research. The second approach will also make it easier for other researchers to adapt and utilise the resource discovery mechanism in actual P2P resource discovery problems and application.

References

[1] K. Aberer, P. Cudré-Mauroux, A. Datta, Z. Despotovic, M. Hauswirth, M. Punceva, and R. Schmidt. P-grid: A self-organizing structured P2P system. *ACM Special Interest Group on Management of Data (SIGMOD)*, 32(3):29–33, 2003.

[2] L.A. Adamic and B.A. Huberman. Power-law distributions of the world wide web. *Science Magazine*, 287(2115), March 2000.

[3] N.A. Al-Dmour and M.S. Saraireh. ParCop with new capabilities and efficient scheduling policies. *Leonardo Journal of Sciences*, (18):11–26, January-June 2011.

[4] N.A. Al-Dmour and W.J. Teahan. The blackboard resource discovery mechanism for P2P networks. In *International Conference on Parallel and Distributed Computing Systems (IASTED)*, MIT, Cambridge, MA, USA, November 9-11 2004.

[5] N.A. Al-Dmour and W.J. Teahan. The blackboard resource discovery mechanism for distributed computing over peer-to-peer networks. In *Proceedings of International Conference on Parallel and Distributed Computing and Networks (IASTED)*, 2005.

[6] H. Ali, M. Ahmed, and et. al. HPRDG: A scalable framework hypercube-P2P-based for resource discovery in computational grid. In *Computer Theory and Applications (ICCTA), 2012 22nd International Conference on*, pages 2–8. IEEE, 2012.

[7] N. Alon. A parallel algorithmic version of the local lemma. In *Proceedings of the 32nd Annual Symposium on Foundations of Computer Science*, pages 586–593, 1991.

[8] B. Amann, L. Berti-Equille, Z. Lacroix, and M.E. Vidal. Challenges of quality-driven resource discovery. In *Resource Discovery, Lecture Notes in Computer Science*, volume 6799, pages 181–189. Springer Berlin Heidelberg, January 2012.

[9] M. Antonini, S. Cirani, G. Ferrari, P. Medagliani, M. Picone, and L. Veltri. Light-weight multicast forwarding for service discovery in low-power IoT networks. In *Software, Telecommunications and Computer Networks (SoftCOM), 2014 22nd International Conference on*, pages 133–138. IEEE, 2014.

[10] B. Awerbuch and R.G. Gallager. A new distributed algorithm to find breadth first search trees. In *IEEE Transactions on Information Theory*, volume IT-33, pages 315–322, May 1987.

[11] A.L. Barabási and R. Albert. Emergence of scaling random networks. In *Science*, volume 286, pages 509–512. American Association for the Advancement of Science (AAAS), 1999.

[12] H. Barjini, M. Othman, and H. Ibrahim. An efficient hybridflood searching algorithm for unstructured peer-to-peer networks. In *Information Computing and Applications*, pages 173–180. Springer, 2010.

[13] B.V. Benjamin, P. Gao, E. McQuinn, S. Choudhary, A.R. Chandrasekaran, J.M. Bussat, R. Alvarez-Icaza, J.V. Arthur, P.A. Merolla, and K. Boahen. Neurogrid: A mixed-analog-digital multichip system for large-scale neural simulations. *Proceedings of the IEEE*, 102(5):699–716, 2014.

[14] A. Biernacki, T. Bauschert, and T.M. Knoll. BitTorrent based P2P IPTV traffic modelling and generation. 2009.

[15] BitTorrent. http://www.bittorrent.com/, 2012.

[16] J. Chen, C.C. Wang, F.C.D. Tsai, C.W. Chang, S.S. Liu, J.J. Guo, W.J. Lien, J.H. Sum, and C.H. Hung. The design and implementation of WiMAX module for NS-2 simulator. In *Proceedings from the 2006 workshop on NS-2: the IP network simulator*, page 5. ACM, 2006.

[17] Q. Chen, F. Schmidt-Eisenlohr, D. Jiang, M. Torrent-Moreno, L. Delgrossi, and H. Hartenstein. Overhaul of IEEE 802.11 modeling and simulation in NS-2. In *Proceedings of the 10^{th} ACM Symposium on Modeling, analysis, and simulation of wireless and mobile systems*, pages 159–168. ACM, 2007.

[18] I. Clarke, S.G. Miller, T.W. Hong, O. Sandberg, and B. Wiley. Protecting free expression online with freenet. *IEEE Internet Computing*, 6(1):40–49, January 2002.

[19] A.J. Conejo, J. Contreras, D.A. Lima, and A. Padilha-Feltrin. Zbus transmission network cost allocation. *IEEE Trans. Power Syst*, 22(1):342–349, 2007.

[20] R. Cuevas, M. Kryczka, A. Cuevas, S. Kaune, C. Guerrero, and R. Rejaie. Unveiling the incentives for content publishing in popular BitTorrent portals. In *IEEE/ACM Transactions of Networking*. IEEE Press, 2012.

[21] S.N. Dorogovtsev and J.F.F. Mendes. Evolution of networks. *Advances in Physics*, 51(4):1079–1187, 2002.

[22] M. Ebrahim, S. Khan, and S.S.U.H. Mohani. Peer-to-peer network simulators: an analytical review. *arXiv preprint arXiv:1405.0400*, 2014.

[23] eMule. http://www.emule-project.net/, 2010.

[24] D.C. Erdil. Proxy-based cloud resource sharing. In *Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 2011 International Conference on*, pages 463–468. IEEE, 2011.

[25] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. From data mining to knowledge discovery in databases. *AI magazine*, 17(3):37, 1996.

[26] I. Foster, C. Kesselman, and S. Tuecke. The anatomy of the grid: Enabling scalable virtual organizations. *International Journal of High Performance Computing Applications*, 15(3):200–222, 2001.

[27] G.A. Fowler and S. McBride. http://www.appliancedesign.com/articles/88301-newest-export-from-china-pirated-pay-tv-9-2. *Wall Street J*, 2005.

[28] N. Fuhr. Resource discovery in distributed digital libraries. 1999.

[29] R. Giordanelli, C. Mastroianni, and M. Meo. Bio-inspired P2P systems: The case of multidimensional overlay. *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, 7(4):35, 2012.

[30] C. Gkantsidis, M. Mihail, and A. Saberi. Random walks in peer-to-peer networks. In *INFOCOM*, volume 1, pages 120–130, Hong Kong, 2004.

[31] Gnutella. http://rfc-gnutella.sourceforge.net/, 2010.

[32] Gnutella2. Gnutella2 developer network, http://g2.trillinux.org/, 2012.

[33] M.I. Haasn. Semantic technology and super-peer architecture for internet based distributed system resource discovery. *International Journal of New Computer Architectures and their Applications (IJNCAA)*, 1(4):848–865, 2011.

[34] J. Han, M. Kamber, and J. Pei. *Data mining: concepts and techniques: concepts and techniques*. Elsevier, 2011.

[35] J.B. Harris. *A Scalable & Extensible Peer-to-Peer Network Simulator*. PhD thesis, Carleton University Ottawa, 2005.

[36] M. Hasanzadeh and M.R. Meybodi. Grid resource discovery based on distributed learning automata. *Computing*, 96(9):909–922, 2014.

[37] M. Heckner, T. Neubauer, and C. Wolff. Tree, funny, to_read, google: what are tags supposed to achieve? a comparative analysis of user keywords for different digital resource types. In *Proceedings of the 2008 ACM workshop on Search in social media*, pages 3–10. ACM, 2008.

[38] D. Hildebrandt, L. Bischofs, and W. Hasselbring. RealPeer - a framework for simulation-based development of peer-to-peer systems. In *Parallel, Distributed and Network-Based Processing, 2007. (PDP'07). 15th EUROMICRO International Conference on*, pages 490–497. IEEE, 2007.

[39] A. Iamnitchi and I. Foster. On fully decentralised resource discovery in grid environments. *Grid Computing-GRID 2001*, (51-62), 2001.

[40] A. Iamnitchi, I. Foster, and D. Nurmi. A peer-to-peer approach to resource discovery in grid environments. *IEEE High Performance Distributed Computing*, July 2002.

[41] Intel Corporation. Over 6 decades of continued transistor shrinkage, innovation. Whitepapers, Intel Corporation, Santa Clara, California, May 2011.

[42] A.A. Jamal, W.S. Wan Awang, M.F. Abdul Kadir, A. Abdul Aziz, and W.J. Teahan. Implementation of resource discovery mechanisms onto PeerSim. In *Proceedings of the 3rd International Conference on Informatics & Applications (ICIA2014)*, Kuala Terengganu, Malaysia, October 2014.

[43] A.A. Jamal, W.S. Wan Awang, M.F. Abdul Kadir, A. Abdul Aziz, and W.J. Teahan. Implementation of resource discovery mechanisms on PeerSim: Enabling up to one million nodes P2P simulation. *International Journal of Digital Information and Wireless Communications (IJDIWC)*, 5(1):14–20, 2015.

[44] A.A. Jamal and W.J. Teahan. Alpha multipliers breadth-first search technique for resource discovery in unstructured peer-to-peer networks. *ARPN Journal of Engineering and Applied Sciences*, (under review), 2016.

[45] S. Joseph. Neurogrid: Semantically routing queries in peer-to-peer networks. In *Web Engineering and Peer-to-Peer Computing*, pages 202–214. Springer, 2002.

[46] V. Kalogeraki, D. Gunopulos, and D. Zeinalipour-Yazti. A local search mechanism for peer-to-peer networks. In *Proceedings of the 11th International Conference on Information and Knowledge Management (CIKM)*, pages 300–307, McLean, Virginia, USA, 2002. ACM.

[47] A. Kapoor. Total cost of application ownership (tca). Whitepapers, The Tolly Group, June 1999.

[48] K. Karaoglanoglou and H. Karatza. Resource discovery in a Grid system: Directing requests to trustworthy virtual organizations based on global trust values. *Journal of Systems and Software*, 84(3):465–478, 2011.

[49] KaZaA. http://www.kazaa.com/, 2010.

[50] J. Kleinberg. The small-world phenomenon: An algorithmic perspective. In *Proceedings of the 32nd annual ACM symposium on Theory of computing*, pages 163–170. ACM, 2000.

[51] J. Kong, W. Cai, L. Wang, and Q. Zhao. A study of pollution on BitTorrent. In *The 2nd International Conference on Computer and Automation Engineering (ICCAE)*, volume 3, pages 118–122. IEEE, February 2010.

[52] N. Kotilainen, M. Vapa, T. Keltanen, A. Auvinen, and J. Vuori. P2PRealm-peer-to-peer network simulator. In *11th International Workshop on Computer-Aided*

Modeling, Analysis and Design of Communication Links and Networks, pages 93–99. IEEE, 2006.

[53] G. Kreitz and F. Niemelä. Spotify - large scale, low latency, P2P music-on-demand streaming. In *Peer-to-Peer Computing (P2P), 2010 IEEE Tenth International Conference on*, pages 1–10. IEEE, 2010.

[54] E.S. Kyper and R.H. Blake. An investigation of the intention to share files over P2P networks. *AMCIS 2009 Proceedings*, page 738, 2009.

[55] D. Lazaro, J.M. Marques, J. Jorba, and X. Vilajosana. Decentralized resource discovery mechanisms for distributed computing in peer-to-peer environments. *ACM Computing Surveys (CSUR)*, 45(4):54, 2013.

[56] G. Lee, Y.C. Chen, and C.C. Lee. Supporting similarity range queries efficiently by using reference points in structured p2p overlays. In *Advances in Intelligent Systems and Applications-Volume 1*, pages 645–652. Springer, 2013.

[57] T. Lin and H. Wang. Search performance analysis in peer-to-peer networks. In *Peer-to-Peer Computing (P2P), 2003 IEEE Third International Conference on*, pages 204–205. IEEE, 2003.

[58] M. Liu, E. Harjula, and M. Ylianttila. An efficient selection algorithm for building a super-peer overlay. *Journal of Internet Services and Applications*, 4(1):1–12, 2013.

[59] B.T. Loo, R. Huebsch, I. Stoica, and J.M. Hellerstein. The case for a hybrid P2P search infrastructure. In *Peer-to-Peer Systems III*, pages 141–150. Springer, 2005.

[60] S.H. Lu, K.C. Li, K.C. Lai, and Y.C. Chung. A scalable P2P overlay based on arrangement graph with minimized overhead. *Peer-to-Peer Networking and Applications*, 7(4):497–510, 2014.

[61] E.K. Lua, J. Crowcroft, M. Pias, R. Sharma, and S. Lim. A survey and comparison of peer-to-peer overlay network schemes. *Communications Surveys & Tutorials, IEEE*, 7(2):72–93, 2005.

[62] G. Macgregor and E. McCulloch. Collaborative tagging as a knowledge organisation and resource discovery tool. *Library review*, 55(5):291–300, 2006.

[63] C. Mack et. al. Fifty years of Moore's law. *Semiconductor Manufacturing, IEEE Transactions on*, 24(2):202–207, 2011.

[64] P. Mahadevan, C. Hubble, D. Krioukov, B. Huffaker, and A. Vahdat. Orbis: Rescaling degree correlations to generate annotated Internet topologies. *ACM SIGCOMM Computer Communication Review*, 37(4):325–336, August 2007.

[65] M. Marzolla, M. Mordacchini, and S. Orlando. Peer-to-peer systems for discovering resources in a dynamic grid. *Parallel Computing*, 33(4):339–358, 2007.

[66] H. Mashayekhi and J. Habibi. Combining search and trust models in unstructured peer-to-peer networks. *The Journal of Supercomputing*, 53(1):66–85, 2010.

[67] S. McCanne, S. Floyd, K. Fall, K. Varadhan, et al. Network simulator NS-2, 1997.

[68] P. Merz and K. Gorunova. Fault-tolerant resource discovery in peer-to-peer grids. *Journal of Grid Computing*, 5(3):319–335, 2007.

[69] S.L. Mirtaheri and M. Sharifi. DHMCF: An efficient resource discovery framework for pure unstructured peer-to-peer systems. *Computer Networks*, 59:213–226, 2014.

[70] A. Mirzaee and P. Rahimzadeh. A agent-based decentralized algorithm for resource semantic discovery in economic grid. In *Communication Software and Networks (ICCSN), 2011 IEEE 3rd International Conference on*, pages 306–311. IEEE, 2011.

[71] A. Montresor and M. Jelasity. PeerSim: A scalable P2P simulator. In *Proceedings of the 9th International Conference on Peer-to-Peer (P2P'09)*, pages 99–100, Seattle, WA, September 2009.

[72] G.E. Moore. Lithography and the future of Moore's law. In *Advances in Resist Technology and Processing XII*, volume 2438, pages 2–17, June 1995.

[73] S. Naićken, A. Basu, B. Livingston, and S. Rodhetbhai. A survey of peer-to-peer network simulators. In *Proceedings of the 7th Annual Postgraduate Symposium*, volume 2, Liverpool, UK, May 2006.

[74] S. Naićken, A. Basu, B. Livingston, S. Rodhetbhai, and I. Wakeman. Towards yet another peer-to-peer simulator. In *Proceedings of The Fourth International Working Conference on Performance Modelling and Evaluation of Heterogeneous Networks (HET-NETs), Ilkley, UK*, 2006.

[75] S. Naićken, B. Livingston, A. Basu, S. Rodhetbhai, I. Wakeman, and D. Chalmers. The state of peer-to-peer simulators and simulations. *ACM SIGCOMM Computer Communication Review*, 37(2):95–98, 2007.

[76] S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system. *Consulted*, 1(2012):28, 2008.

[77] Napster. http://www.napster.com/, 2004.

[78] N.J. Navimipour and F.S. Milani. A comprehensive study of the resource discovery techniques in peer-to-peer networks. In *Peer-to-Peer Network Applications*. Springer Science+Business Media New York, March 2014.

[79] N.J. Navimipour, A.M. Rahmani, A.H. Navin, and M. Hosseinzadeh. Resource discovery mechanisms in grid systems: A survey. *Journal of Network and Computer Applications*, 41:389–410, 2014.

[80] T. O'Reilly. What is web 2.0: Design patterns and business models for the next generation of software. *Communications & strategies*, (1):17, 2007.

[81] V.N. Padmanabhan, H.J. Wang, P.A. Chou, and K. Sripanidkulchai. Distributing streaming media content using cooperative networking. In *Proceedings of the 12th international workshop on Network and operating systems support for digital audio and video*, pages 177–186. ACM, 2002.

[82] H. Papadakis, P. Trunfio, D. Talia, and P. Fragopoulou. Design and implementation of a hybrid P2P-based grid resource discovery system. In *Making Grids Work*, pages 89–101. Springer, 2008.

[83] C.H. Papadimitriou. *Computational complexity*. John Wiley and Sons Ltd., 2003.

[84] H. Park, J. Yang, J. Park, S.G. Kang, and J.K. Choi. A survey on peer-to-peer overlay network schemes. In *2008 10th International Conference on Advanced Communication Technology*, volume 2, pages 986–988, 2008.

[85] PeerThing. http://www.se.uni-oldenburg.de/en/download/pgp2p/userguide.pdf, 2006.

[86] PlanetLab. http://www.planet-lab.org/, 2014.

[87] Softonic PPLive. http://www.pplive.en.softonic.com/, 2010.

[88] W. Qu, W. Zhou, and M. Kitsuregawa. Sharable file searching in unstructured peer-to-peer systems. *The Journal of Supercomputing*, 51(2):149–166, 2010.

[89] Query Cycle. http://p2p.stanford.edu/index.html, 2015.

[90] M. Randić. Restricted random walks on graphs. In *Theoretica Chimica Acta 92*, number 2, pages 97–106, 1995.

[91] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A scalable content-addressable network. 2001.

[92] A. Rowstron and P. Druschel. Storage management and caching in PAST, a large-scale, persistent peer-to-peer storage utility. In *ACM SIGOPS Operating Systems Review*, volume 35, pages 188–201. ACM, 2001.

[93] S. Russell and P. Norvig. *Artificial intelligence: A modern approach*, volume 25. Prentice-Hall, Egnlewood Cliffs, 3^{rd} edition, 2010.

[94] M.P. Said and I. Kojima. S-MDS: Semantic monitoring and discovery system for the grid. *Journal of Grid Computing*, 7(2):205–224, 2009.

[95] C. Schmidt and M. Parashar. Flexible information discovery in decentralized distributed systems. In *High Performance Distributed Computing, 2003. Proceedings. 12th IEEE International Symposium on*, pages 226–235. IEEE, 2003.

[96] M. Shojafar, J.H. Abawajy, Z. Delkhah, A. Ahmadi, Z. Pooranian, and A. Abraham. An efficient and distributed file search in unstructured peer-to-peer networks. *Peer-to-Peer Networking and Applications*, 8(1):120–136, 2015.

[97] H. Si, Z. Chen, Y. Deng, and L. Yu. Semantic web services publication and OCT-based discovery in structured P2P network. *Service Oriented Computing and Applications*, 7(3):169–180, 2013.

[98] M.C. Sinclair. Minimum cost topology optimisation of the cost 239 european optical network. In *Artificial Neural Nets and Genetic Algorithms*, pages 26–29. Springer, 1995.

[99] Y.P. Singh, R. Rathi, J. Gajrani, and V. Jain. Two levels TTL for unstructured P2P network using adaptive probabilistic search. *International Journal of Scientific & Engineering Research*, 3(1):1–4, January 2012.

[100] Skype. http://www.skype.com/, 2010.

[101] I. Stoica, R. Morris, D. Karger, M.F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. *ACM SIGCOMM Computer Communication*, pages 149–160, August 2001.

[102] R. Tarjan. Depth-first search and linear graph algorithms. In *12^{th} Annual Symposium on Switching and Automata Theory*. IEEE Computer Society, October 1971.

[103] W.J. Teahan. Informed search, artificial intelligence and intelligent agents. University Lecture, Bangor University, 2015.

[104] J.A. Torkestani. A distributed resource discovery algorithm for P2P grids. *Network and Computer Applications*, 35(6):2028–2036, 2012.

[105] P. Trunfio, D. Talia, H. Papadakis, P. Fragopoulou, M. Mordacchini, M. Pennanen, K. Popov, V. Vlassov, and S. Haridi. Peer-to-peer resource discovery in grids: Models and systems. *Future Generation Computer Systems*, 23(7):864–878, 2007.

[106] D. Tsoumakos and N. Roussopoulos. Adaptive probabilisitic search for peer-to-peer networks. In *Proceedings of the 3^{rd} International Conference on Peer-to-Peer Computing (P2P)*, page 102. IEEE Computer Society, 2003.

[107] D.J. Watts and S.H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393:440–442, June 1998.

[108] B. Yang and H. Garcia-Molina. Improving search in peer-to-peer networks. In *Proceedings of the 22nd International Conference on Distributed Computing Systems (ICDCS)*, page 5. IEEE Computer Society, 2002.

[109] M. Yang and Y. Yang. An efficient hybrid peer-to-peer system for distributed data sharing. *Computers, IEEE Transactions on*, 59(9):1158–1171, 2010.

[110] W. Yang and N. Abu-Ghazaleh. GPS: a general peer-to-peer simulator and its use for modeling BitTorrent. In *Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, 2005. 13th IEEE International Symposium on*, pages 425–432. IEEE, 2005.

[111] M. Zaharia and S. Keshav. Gossip-based search selection in hybrid peer-to-peer networks. *Concurrency and Computation: Practice and Experience*, 20(2):139, 2008.

[112] W. Zhang, S. Zhang, M. Cai, and Y. Liu. A reputation-based peer-to-peer architecture for semantic service discovery in distributed manufacturing environments. *Concurrent Engineering*, page 1063293X12457402, 2012.

[113] A. Zhygmanovskyi and N. Yoshida. Peer-to-peer network for flexible service sharing and discovery. In *Multiagent System Technologies*, pages 152–165. Springer, 2013.

A Appendices (Source Codes)

A.1 PeerSim iSearch Configuration File

Algorithm A.1 `config-isearch-BRDM.txt`

```
00
01 random.seed 1234567890
02 simulation.cycles 20
03 control.0 peersim.cdsim.Shuffle
04
05 network.size 1000000
06
07 include.protocol search
08 protocol.topology peersim.core.IdleProtocol
09 protocol.topology.capacity 10
10 protocol.search peersim.extras.gj.isearch.BRDMProtocol
11 protocol.search.ttl 10
12
13 protocol.search.alpha1 0.8
14 protocol.search.alpha2 0.5
15 protocol.search.alpha3 0.3
16 protocol.search.alpha4 0.1
17 protocol.search.alpha5 0.0
18
19 init.0 peersim.dynamics.WireScaleFreeBA
20 init.0.protocol search
21 init.0.k 2
22 init.0.undir
23
24 init.1 peersim.extras.gj.isearch.SearchDataInitializer
25 init.1.protocol search
26 init.1.keywords 1000
27 init.1.query_nodes 1
28 init.1.query_interval 1
29 init.1.max_queries 1
30
31 control.0 peersim.extras.gj.isearch.SearchObserver
32 control.0.protocol search
33 control.0.verbosity 1
34
```

A.2 PeerSim iSearch Protocol Java Codes

Algorithm A.2 `BRDMProtocol.java` (Part 1 of 6)

```
000
001 package peersim.extras.gj.isearch;
002 import java.util.ArrayList;
003 import java.util.HashMap;
004 import java.util.Iterator;
005
006 import peersim.config.Configuration;
007 import peersim.core.Linkable;
008 import peersim.core.Node;
009
010 public class BRDMProtocol extends SearchProtocol {
011  public static final String PAR_WALKERS = "walkers";
012
013  public static final String PAR_ALPHA1 = "alpha1";
014  public static final String PAR_ALPHA2 = "alpha2";
015  public static final String PAR_ALPHA3 = "alpha3";
016  public static final String PAR_ALPHA4 = "alpha4";
017  public static final String PAR_ALPHA5 = "alpha5";
018
019  protected int walkers;
020  protected double alpha1, alpha2, alpha3, alpha4, alpha5;
021
022  protected HashMap<Node, ArrayList<Node>> RL;
023  protected HashMap<Node, ArrayList<Node>> UL;
024
025  public BRDMProtocol(String prefix) {
026   super(prefix);
027   walkers = Configuration.getInt(prefix + "." + PAR_WALKERS, 1);
028
029   alpha1 = Configuration.getDouble(prefix + "." + PAR_ALPHA1, 0.0);
030   alpha2 = Configuration.getDouble(prefix + "." + PAR_ALPHA2, 0.0);
031   alpha3 = Configuration.getDouble(prefix + "." + PAR_ALPHA3, 0.0);
032   alpha4 = Configuration.getDouble(prefix + "." + PAR_ALPHA4, 0.0);
033   alpha5 = Configuration.getDouble(prefix + "." + PAR_ALPHA5, 0.0);
034
035   RL = new HashMap<Node, ArrayList<Node>>();
036   UL = new HashMap<Node, ArrayList<Node>>();
037  }
038
```

Continued in Algorithm A.3.

Algorithm A.3 `BRDMProtocol.java` (Part 2 of 6)

Continued from Algorithm A.2.

```
038
039  public void process(SMessage mes) {
040   boolean match = this.match(mes.payload);
041
042   int amountOfForwards=0;
042
043   if (match){
044    this.notifyOriginator(mes);
045    updateAllRouteTaken(mes, "RL");
046   } else {
047    updateAllRouteTaken(mes, "UL");
048   }
049
050   if(mes.hops==0){
051    amountOfForwards=(int)(this.degree()*alpha1);
052   }else if(mes.hops==1){
053    amountOfForwards=(int)(this.degree()*alpha2);
054   }else if(mes.hops==2){
055    amountOfForwards=(int)(this.degree()*alpha3);
056   }else if(mes.hops==3){
057    amountOfForwards=(int)(this.degree()*alpha4);
058   }else if(mes.hops==4){
059    amountOfForwards=(int)(this.degree()*alpha5);
060   }else{
061    amountOfForwards=1;
062   }
063
064   if(amountOfForwards<1){
065    amountOfForwards=1;
066   }
067
068   mes.addCurrentNode(whoAmI);
069   ArrayList<Node> sentNode = new ArrayList<Node>();
070
```

Continued in Algorithm A.2.

Algorithm A.4 `BRDMProtocol.java` (Part 3 of 6)

Continued from Algorithm A.3.

```
070
071   if (RL.containsKey(whoAmI) && !RL.get(whoAmI).isEmpty()) {
072    Iterator<Node> it = RL.get(whoAmI).iterator();
073
074    while (it.hasNext()) {
075     Node cNode = it.next();
076
077     for (int i = 0; i < this.degree(); i++) {
078      Node fwdNode = this.getNeighbor(i);
079
080      if (cNode.equals(fwdNode)) continue;
081      if (!sentNode.contains(fwdNode) && RL.get(cNode).contains(fwdNode)){
082       this.forward(fwdNode, mes);
083       sentNode.add(fwdNode);
084       amountOfForwards--;
085      }
086     }
087    }
088   }
089
090   if (UL.containsKey(whoAmI) && !UL.get(whoAmI).isEmpty()) {
091    Iterator<Node> it = UL.get(whoAmI).iterator();
092
093    while (it.hasNext()) {
094     Node cNode = it.next();
095
096     for (int i = 0; i < amountOfForwards && i < this.degree(); i++){
097      Node fwdNode = this.selectFreeNeighbor(mes);
098
099      if (cNode.equals(fwdNode)) continue;
100      if (!sentNode.contains(fwdNode) && UL.get(cNode).contains(fwdNode)){
101       this.forward(fwdNode, mes);
102       sentNode.add(fwdNode);
103       amountOfForwards--;
104      }
105     }
106    }
107   }
108
```

Continued in Algorithm A.5.

Algorithm A.5 `BRDMProtocol.java` (Part 4 of 6)

Continued from Algorithm A.2.

```
108
109   if(amountOfForwards<1){
110    amountOfForwards=1;
111   }
112
113   for(int i=0; i<amountOfForwards; i++){
114    Node neighbor = this.selectFreeNeighborOnly(mes);
115
116    if(neighbor!=null && !sentNode.contains(neighbor)) {
117     this.forward(neighbor, mes);
118    }
119   }
120   sentNode.clear();
121  }
122
123  public void nextCycle(peersim.core.Node node, int protocolID) {
124   super.nextCycle(node, protocolID);
125   int[] data = this.pickQueryData();
126
127   if (data != null) {
128    System.err.println("RUN NEXT CYCLE");
129    SMessage m = new SMessage(node, this.whoAmI, SMessage.QRY, 0,
data);
130
131    for (int i = 0; i< this.walkers && i < this.degree(); i++) {
132     this.send((Node) this.getNeighbor(i), m);
133    }
134   }
135  }
136
```

Continued in Algorithm A.6

Algorithm A.6 `BRDMProtocol.java` (Part 5 of 6)

Continued from Algorithm A.5.

```
137  public boolean addULNeighbor(Node current, Node neighbor) {
138   if (UL.containsKey(current)) {
139    if (RL.containsKey(current)) {
140     if (!UL.get(current).contains(neighbor) &&
!RL.get(current).contains(neighbor)) {
141      UL.get(current).add(neighbor);
142      return true;
143     }
144    } else {
145     if (!UL.get(current).contains(neighbor)) {
146      UL.get(current).add(neighbor); return true;
147     }
148    }
149   } else {
150    ArrayList<Node> tmp = new ArrayList<Node>();
151    tmp.add(current);
152    UL.put(current, tmp);
153    return true;
154   }
155   return false;
156  }
157
158  public boolean addRLNeighbor(Node neighbor, Node current) {
159   if (RL.containsKey(current)) {
160    if (!RL.get(current).contains(neighbor)) {
161     RL.get(current).add(neighbor);
162     if (UL.containsKey(current)) {
163      if (UL.get(current).contains(neighbor)) {
164       UL.get(current).remove(neighbor.getIndex());
165      }
166     }
167     return true;
168    }
169   } else {
179    ArrayList<Node> tmp = new ArrayList<Node>();
180    tmp.add(neighbor);
190    RL.put(current, tmp);
191    return true;
192   }
193   return false;
194  }
```

Continued in Algorithm A.7

Algorithm A.7 `BRDMProtocol.java` (Part 6 of 6)

Continued from Algorithm A.6.

```
195
196  public boolean findNeighborUL(Node neighbor) {
197   if (UL.containsKey(whoAmI) && UL.get(whoAmI).contains(neighbor)) {

198    return true;
199   }
200   return false;
201  }
202
203  public boolean findNeighborRL(Node neighbor) {
204   if (RL.containsKey(whoAmI) && RL.get(whoAmI).contains(neighbor)) {
205    return true;
206   }
207   return false;
208  }
209
210  public boolean updateAllRouteTaken(SMessage msg, String type) {
211   if (msg.routeTaken.isEmpty()) {
212    System.err.println("BRDM Protocol - Update Ancestor Node Failed
(routeTaken is empty)");
213    return false;
214   }
215
216   Iterator<Node> itNode = msg.routeTaken.iterator();
217   Node p = this.whoAmI;
218   while (itNode.hasNext()) {
219    Node c = itNode.next();
220    if (c.equals(p)) continue;
221
222    if (type.equals("RL")){
223     addRLNeighbor(p, c);
224    } else {
225     addULNeighbor(p, c);
226    }
227    p = c;
228   }
229   return true;
230  }
231 }
232
```

B Appendices (Simulation Results)

Table B.1: α-BFS Query Efficiency (η) and Maximum Successful Searches (Random Seed: 1234567890)

Alpha Multipliers					Query Efficiency (η)			Max. Successful Searches		
α_1	α_2	α_3	α_4	α_5	TTL5	TTL10	TTL20	TTL5	TTL10	TTL20
1.0	1.0	1.0	1.0	1.0	29,572	2,847	NA*	89,244	217,309	NA*
1.0	0.8	0.6	0.4	0.2	142,787	159,763	162,837	26,837	97,894	118,389
1.0	0.9030	0.7782	0.6021	0.3010	124,938	151,336	154,508	31,813	98,790	118,165
1.0	0.8	0.4	0.2	0.1	176,860	192,567	198,069	11,827	40,957	64,715
1.0	0.5	0.25	0.125	0.0625	195,293	208,909	211,882	4,215	13,662	26,727
0.8	0.5	0.3	0.1	0.0	196,631	211,053	210,632	3,642	10,689	20,755
0.8	0.4	0.2	0.1	0.05	213,646	213,204	215,221	2,458	7,557	15,814
0.5	0.4	0.3	0.0	0.0	197,842	211,537	213,615	1,815	4,848	9,878
0.5	0.25	0.125	0.0625	0.03125	209,613	220,228	216,013	894	2,724	5,771
0.2	0.4	0.6	0.8	1.0	55,969	77,292	76,291	48,492	175,299	181,474
0.8	0.8	0.8	0.8	0.8	47,628	58,900	58,806	68,510	191,055	193,140
0.6	0.6	0.6	0.6	0.6	86,672	104,253	103,987	44,270	158,595	167,448
0.5	0.5	0.5	0.5	0.5	115,465	130,602	132,387	30,004	130,892	147,221
0.4	0.4	0.4	0.4	0.4	145,440	168,535	172,338	18,233	80,658	106,694
0.3	0.3	0.3	0.3	0.3	175,938	187,146	192,646	8,764	47,145	74,012

* Insufficient memory error

Table B.2: α-BFS Query Efficiency (η) and Maximum Successful Searches (Random Seed: 1415926535)

Alpha Multipliers					Query Efficiency (η)			Max. Successful Searches		
α_1	α_2	α_3	α_4	α_5	TTL5	TTL10	TTL20	TTL5	TTL10	TTL20
1.0	1.0	1.0	1.0	1.0	21,554	19,359	19,542	117,680	219,306	219,449
1.0	0.8	0.6	0.4	0.2	132,253	145,298	147,149	46,921	136,499	148,888
1.0	0.9030	0.7782	0.6021	0.3010	96,551	95,869	94,974	75,092	181,564	185,191
1.0	0.8	0.4	0.2	0.1	174,667	188,937	193,412	21,249	69,313	93,937
1.0	0.5	0.25	0.125	0.0625	204,372	213,097	216,337	7,246	22,258	39,613
0.8	0.5	0.3	0.1	0.0	206,911	216,298	219,228	6,820	18,623	33,309
0.8	0.4	0.2	0.1	0.05	210,253	221,944	220,772	4,573	13,929	26,076
0.5	0.4	0.3	0.0	0.0	204,196	217,827	222,853	3,952	10,271	19,577
0.5	0.25	0.125	0.0625	0.03125	223,810	223,871	224,531	2,021	5,687	11,765
0.2	0.4	0.6	0.8	1.0	64,802	71,545	71,166	65,918	195,215	198,193
0.8	0.8	0.8	0.8	0.8	48,299	46,934	46,121	96,718	208,875	209,709
0.6	0.6	0.6	0.6	0.6	92,451	96,662	95,402	66,595	181,892	186,873
0.5	0.5	0.5	0.5	0.5	108,858	119,898	116,764	48,600	158,275	169,918
0.4	0.4	0.4	0.4	0.4	152,854	171,126	173,822	26,226	100,966	124,003
0.3	0.3	0.3	0.3	0.3	177,820	194,307	199,556	14,054	58,084	85,586

Table B.3: α-BFS Query Efficiency (η) and Maximum Successful Searches (Random Seed: 8979323846)

Alpha Multipliers					Query Efficiency (η)			Max. Successful Searches		
α_1	α_2	α_3	α_4	α_5	TTL5	TTL10	TTL20	TTL5	TTL10	TTL20
1.0	1.0	1.0	1.0	1.0	1,115	1,149	1,135	2,471	5,563	5,549
1.0	0.8	0.6	0.4	0.2	4,240	4,489	4,569	859	2,770	3,329
1.0	0.9030	0.7782	0.6021	0.3010	3,218	3,344	3,330	1,551	4,149	4,391
1.0	0.8	0.4	0.2	0.1	5,182	5,626	5,448	408	1,330	1,903
1.0	0.5	0.25	0.125	0.0625	6,272	5,867	5,766	153	415	771
0.8	0.5	0.3	0.1	0.0	5,667	5,885	5,629	129	355	631
0.8	0.4	0.2	0.1	0.05	5,791	5,565	6,001	87	247	527
0.5	0.4	0.3	0.0	0.0	6,221	5,774	5,664	85	196	376
0.5	0.25	0.125	0.0625	0.03125	7,977	7,013	6,113	42	97	182
0.2	0.4	0.6	0.8	1.0	1,896	2,182	2,164	1,347	4,808	4,942
0.8	0.8	0.8	0.8	0.8	1,747	1,869	1,845	1,990	5,179	5,208
0.6	0.6	0.6	0.6	0.6	2,949	3,098	3,083	1,329	4,232	4,500
0.5	0.5	0.5	0.5	0.5	3,840	3,952	4,052	901	3,291	3,817
0.4	0.4	0.4	0.4	0.4	4,497	4,774	4,868	554	2,158	2,840
0.3	0.3	0.3	0.3	0.3	4,790	5,398	5,388	259	1,133	1,799

Table B.4: Query Efficiency in Percentage (Random Seed: 1234567890, 1415926535, and 8979323846)

Alpha Multipliers					Query Efficiency (η) Percentage (%)								
					Random Seed: 1234567890			Random Seed: 1415926535			Random Seed: 8979323846		
α_1	α_2	α_3	α_4	α_5	TTL5	TTL10	TTL20	TTL5	TTL10	TTL20	TTL5	TTL10	TTL20
1.0	1.0	1.0	1.0	1.0	13.43	1.29	NA	9.60	8.62	8.70	13.97	14.40	14.23
1.0	0.8	0.6	0.4	0.2	64.84	72.54	73.94	58.90	64.71	65.54	53.15	56.27	57.28
1.0	0.9030	0.7782	0.6021	0.3010	56.73	68.72	70.16	43.00	42.70	42.30	40.35	41.91	41.74
1.0	0.8	0.4	0.2	0.1	80.31	87.44	89.94	77.79	84.15	86.14	64.95	70.53	68.30
1.0	0.5	0.25	0.125	0.0625	88.68	94.86	96.21	91.02	94.91	96.35	78.62	73.55	72.28
0.8	0.5	0.3	0.1	0.0	89.29	95.83	95.64	92.15	96.33	97.64	71.03	73.77	70.56
0.8	0.4	0.2	0.1	0.05	97.01	96.81	97.73	93.64	98.85	98.33	72.60	69.76	75.22
0.5	0.4	0.3	0.0	0.0	89.84	96.05	97.00	90.94	97.01	99.25	77.98	72.38	71.01
0.5	0.25	0.125	0.0625	0.03125	95.18	100.00	98.09	99.68	99.71	100.00	100.00	87.91	76.63
0.2	0.4	0.6	0.8	1.0	25.41	35.10	34.64	28.86	31.86	31.70	23.77	27.35	27.13
0.8	0.8	0.8	0.8	0.8	21.63	26.74	26.70	21.51	20.90	20.54	21.90	23.43	23.13
0.6	0.6	0.6	0.6	0.6	39.36	47.34	47.22	41.17	43.05	42.49	36.97	38.83	38.65
0.5	0.5	0.5	0.5	0.5	52.43	59.30	60.11	48.48	53.40	52.00	48.13	49.55	50.79
0.4	0.4	0.4	0.4	0.4	66.04	76.53	78.25	68.08	76.21	77.42	56.38	59.84	61.02
0.3	0.3	0.3	0.3	0.3	79.89	84.98	87.48	79.20	86.54	88.88	60.05	67.67	67.55

Table B.5: Maximum Successful Searches in Percentage (Random Seed: 1234567890, 1415926535, and 8979323846)

Alpha Multipliers					Max. Successful Searches Percentage (%)								
					Random Seed: 1234567890			Random Seed: 1415926535			Random Seed: 8979323846		
α_1	α_2	α_3	α_4	α_5	TTL5	TTL10	TTL20	TTL5	TTL10	TTL20	TTL5	TTL10	TTL20
1.0	1.0	1.0	1.0	1.0	41.07	100.00	NA	53.63	99.93	100.00	44.42	100.00	99.75
1.0	0.8	0.6	0.4	0.2	12.35	45.05	54.48	21.38	62.20	67.85	15.44	49.79	59.84
1.0	0.9030	0.7782	0.6021	0.3010	14.64	45.46	54.38	34.22	82.74	84.39	27.88	74.58	78.93
1.0	0.8	0.4	0.2	0.1	5.44	18.85	29.78	9.68	31.59	42.81	7.33	23.91	34.21
1.0	0.5	0.25	0.125	0.0625	1.94	6.29	12.30	3.30	10.14	18.05	2.75	7.46	13.86
0.8	0.5	0.3	0.1	0.0	1.68	4.92	9.55	3.11	8.49	15.18	2.32	6.38	11.34
0.8	0.4	0.2	0.1	0.05	1.13	3.48	7.28	2.08	6.35	11.88	1.56	4.44	9.47
0.5	0.4	0.3	0.0	0.0	0.84	2.23	4.55	1.80	4.68	8.92	1.53	3.52	6.76
0.5	0.25	0.125	0.0625	0.03125	0.41	1.25	2.66	0.92	2.59	5.36	0.75	1.74	3.27
0.2	0.4	0.6	0.8	1.0	22.31	80.67	83.51	30.04	88.96	90.31	24.21	86.43	88.84
0.8	0.8	0.8	0.8	0.8	31.53	87.92	88.88	44.07	95.18	95.56	35.77	93.10	93.62
0.6	0.6	0.6	0.6	0.6	20.37	72.98	77.06	30.35	82.89	85.16	23.89	76.07	80.89
0.5	0.5	0.5	0.5	0.5	13.81	60.23	67.75	22.15	72.12	77.43	16.20	59.16	68.61
0.4	0.4	0.4	0.4	0.4	8.39	37.12	49.10	11.95	46.01	56.51	9.96	38.79	51.05
0.3	0.3	0.3	0.3	0.3	4.03	21.69	34.06	6.40	26.47	39.00	4.66	20.37	32.34

Table B.6: Informed Searches Query Efficiencies (η and $\eta*$) and Successful Searches (ss)

Technique	Alpha Multipliers	Rand. Seed: 1234567890			Rand. Seed: 1415926535			Rand. Seed: 8979323846		
		η	η^*	ss	η	η^*	ss	η	η^*	ss
BRDM	Set **B**	171,933	85,966	12,053	159,654	79,827	29,397	4,833	2,417	348
	Set **M**	159,605	79,802	10,479	158,856	79,428	18,122	4,386	2,193	417
	Set **N**	178,458	89,229	4,399	181,149	90,574	8,564	5,503	2,752	246
LBRDM	Set **B**	166,911	143,036	8,320	169,993	145,294	19,403	4,783	4,760	513
	Set **M**	158,115	136,528	9,605	160,942	138,631	20,674	4,436	4,416	515
	Set **N**	181,224	153,421	4,218	186,712	157,336	10,058	4,898	4,875	288

Table B.7: Combined Query Efficiencies (η and $\eta*$) and Successful Searches (ss)

Techniques	Alpha Multipliers	Rand. Seed: 1234567890		Rand. Seed: 1415926535		Rand. Seed: 8979323846	
		η & ss	η^* & ss	η & ss	η^* & ss	η & ss	η^* & ss
BRDM	Set **B**	97.4	78.0	92.8	75.4	77.7	58.6
	Set **M**	87.5	69.5	73.4	56.1	80.3	63.0
	Set **N**	67.5	47.3	63.1	43.3	73.9	52.1
LBRDM	Set **B**	80.6	81.1	78.5	79.2	93.3	98.6
	Set **M**	83.5	84.3	78.3	79.2	90.3	95.3
	Set **N**	67.5	67.5	67.1	67.1	72.5	78.0