Multiple Criteria Routing Algorithms
in Mesh Overlay Networks

Dissertation
zur Erlangung des akademischen Grades
DOKTOR-INGENIEUR

der Fakultät für
Mathematik und Informatik
der FernUniversität
in Hagen

von
Lada-on Lertsuwanakul, M.Sc.
Bangkok, Thailand

Hagen 2011
Vorsitzender des Prüfungsausschusses: Prof. Dr. Dr. Wolfgang A. Halang

1. Berichterstatter: Prof. Dr.-Ing. habil. Herwig Unger

2. Berichterstatter: Dr. Roman Gumzej
Acknowledgements

It gives me immense pleasure to acknowledge my family and fellow colleagues, and especially Prof. Dr.-Ing. habil. Herwig Unger who tirelessly helped me to prepare this thesis. I appreciate all his contributions of time, ideas, encouragement, supervision and support from the preliminary to the concluding level enabled me to develop an understanding of the subject. And I would like to thank my committee members: Prof. Dr. Dr. Wolfgang A. Halang and Dr. Roman Gumzej for their precious time. Furthermore, I gratefully acknowledge the funding sources that made my Ph.D works possible. I have been funded by the STIBET-Matching Fund between The German Academic Exchange Service (DAAD) and JNC Software Development and Consulting for three years in Germany.

Many thanks to Hauke Coltza with his valuable comments and suggestions make quality of this thesis improve. I thank to my other colleagues, Wolfgang Benner, Werner Schubert, Daniel Berg, Mario Kubek, Coskun Akinalp, Miguel A. Rojas, Jessica Gabski, Nicole Lotz, Barbara Kleine as the members of the Communication Networks Department. Moreover, I am also thankful to all Thai friends: Panchalee Sukjit, Sophida Tuammee, Sunantha Sodsee and Maytiyanin Komkhao. They all have contributed to my personal and professional time at FernUniversität in Hagen. They have been a source of friendships as well as collaboration.

I am grateful for all internship students who were contributed their works as parts of this thesis: Saranratch Koed-im, Pornnapa Piyarujira, Lulu Hou, Roy Glad, Marcio, Dmitriy Ten, Vasileios Georgiou-Sarlhiotis, Phanupong Anantawatchakorn, Noppadol Mututanon, Pongpan Kajadsarapadpai and Yahya Hamad.

Finally, I have to thank my family and friends for all their love, support and encouragement in all my pursuits.

Lada-on Lertsuwanakul
Hagen
November 11, 2011
# Contents

Acknowledgments ........................................ III

Abstract .................................................. VII

1 Introduction ........................................... 1

1.1 Background and Motivation .......................... 1

1.2 Main Contributions ................................. 4

1.2.1 Thermal Field Approach ......................... 4

1.2.2 Multiple Criteria Routing in Overlay P2P Networks .... 5

1.2.3 Self-organizing Overlay Topology ............... 5

1.2.4 Other contributions ............................. 6

1.3 Thesis Outline .................................... 6

2 State Of The Art ...................................... 7

2.1 Quality of Service Routing ....................... 7

2.2 Overlay Networks ................................ 8

2.2.1 Peer-to-Peer Systems .......................... 11

2.2.2 Unstructured P2P Systems ................... 13

2.2.3 Structured P2P Systems ....................... 14

2.2.4 Search and Routing Extension of P2P Systems ... 16

2.3 Small-World Networks ............................ 18

2.3.1 Kleinberg’s Small-World Model ............... 19

2.3.2 Extension of Small-World Networks ........... 20

2.4 Grid structure on top of P2P System ............. 21

2.5 QoS Routing in Overlay Networks ............... 22

2.6 P2PNetSim - Network Simulator .................. 24

3 Multicriteria Routing Algorithm - Criteria Selection .... 28

3.1 Motivation ........................................ 28

3.2 Thermal Field Considering Buffer Utilization .... 31

3.3 Route Decision by Constant Probability .......... 32

3.3.1 Simulation Setup ............................. 33

3.3.2 Simulation Results and Discussion ............ 36
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.4</td>
<td>Route Decision by Probability Functions</td>
<td>38</td>
</tr>
<tr>
<td>3.4.1</td>
<td>Probability Function Considering Remaining Distance</td>
<td>38</td>
</tr>
<tr>
<td>3.4.2</td>
<td>Simulation Results and Discussion</td>
<td>40</td>
</tr>
<tr>
<td>3.5</td>
<td>Summary</td>
<td>43</td>
</tr>
<tr>
<td>4</td>
<td>Multicriteria Routing Algorithm - Criteria Combination</td>
<td>47</td>
</tr>
<tr>
<td>4.1</td>
<td>Route Decision by Weight Function</td>
<td>47</td>
</tr>
<tr>
<td>4.1.1</td>
<td>Weight Function with Thermal Field Approach</td>
<td>48</td>
</tr>
<tr>
<td>4.1.2</td>
<td>Simulation Results and Discussion</td>
<td>49</td>
</tr>
<tr>
<td>4.2</td>
<td>Route Decision by Fuzzy Logic</td>
<td>53</td>
</tr>
<tr>
<td>4.2.1</td>
<td>Fuzzy Logic</td>
<td>53</td>
</tr>
<tr>
<td>4.2.2</td>
<td>Fuzzy Logic with Thermal Field Approach</td>
<td>54</td>
</tr>
<tr>
<td>4.2.3</td>
<td>Simulation Results and Discussion</td>
<td>56</td>
</tr>
<tr>
<td>4.3</td>
<td>Summary</td>
<td>59</td>
</tr>
<tr>
<td>5</td>
<td>Self-Organizing Overlay Topology</td>
<td>61</td>
</tr>
<tr>
<td>5.1</td>
<td>Motivation</td>
<td>61</td>
</tr>
<tr>
<td>5.2</td>
<td>Self-organizing Bridge Algorithm</td>
<td>62</td>
</tr>
<tr>
<td>5.3</td>
<td>Simulation Results and Discussion</td>
<td>64</td>
</tr>
<tr>
<td>5.3.1</td>
<td>Simulation Model</td>
<td>64</td>
</tr>
<tr>
<td>5.3.2</td>
<td>Simulation Results Discussion</td>
<td>65</td>
</tr>
<tr>
<td>6</td>
<td>Extension Algorithms and Related Applications</td>
<td>69</td>
</tr>
<tr>
<td>6.1</td>
<td>Long-Link Notification Algorithm in Small-World Network</td>
<td>69</td>
</tr>
<tr>
<td>6.1.1</td>
<td>Ant Colony Optimization</td>
<td>69</td>
</tr>
<tr>
<td>6.1.2</td>
<td>Pheromone-Based Landmarks Algorithm</td>
<td>70</td>
</tr>
<tr>
<td>6.1.3</td>
<td>Simulation Results and Discussion</td>
<td>73</td>
</tr>
<tr>
<td>6.2</td>
<td>Network Creation Model</td>
<td>76</td>
</tr>
<tr>
<td>6.2.1</td>
<td>Related Works</td>
<td>76</td>
</tr>
<tr>
<td>6.2.2</td>
<td>Game Model</td>
<td>77</td>
</tr>
<tr>
<td>6.2.3</td>
<td>Simulation and Discussion</td>
<td>83</td>
</tr>
<tr>
<td>7</td>
<td>Conclusion</td>
<td>87</td>
</tr>
<tr>
<td>7.1</td>
<td>Key Contributions and Results Evaluation</td>
<td>87</td>
</tr>
<tr>
<td>7.2</td>
<td>Other Contributions</td>
<td>89</td>
</tr>
<tr>
<td>7.3</td>
<td>Outlook and future works</td>
<td>90</td>
</tr>
<tr>
<td>Bibliography</td>
<td></td>
<td>91</td>
</tr>
</tbody>
</table>
Abstract

Overlay networks, especially Peer-to-Peer systems, are emerging technologies and a field of active research and development today. Many different kinds of overlay systems are used to solve problems in processing and handling massively distributed information. Therefore, scalable and flexible information dissemination solutions are crucial. One of the major challenges is to build overlay systems with effective routing mechanisms working efficiently in dynamically changing distributed environments. This is especially true today, when numerous and enormously large data transfers have become vastly popular on the Internet with more than a billion users.

This dissertation presents novel algorithms aimed to improve the quality of service (QoS) of routing in overlay networks by considering multiple criteria, i.e., physical network parameters, content correlation, and overlay topology. To do so, a thermal field approach, inspired by thermal physics, is introduced first and investigated exemplarily using a peer’s buffer utilization to represent its state. After that, several routing decision making mechanisms are presented that make use both of a peer’s state as well as of the current position in the overlay network. The first algorithms take one criteria at a time into account, whereas the following ones take several concurrent criteria into account to make an optimum routing decision.

Besides general network constraints, also the topology itself strongly affects the routing performance. Inspired by Jon Kleinberg’s small-world model, a novel mechanism for organizing grid overlay systems is introduced. It is based on self-organizing shortcuts that evolve dynamically according to network traffic. Making use of the thermal field approach to represent traffic, automatic long-link construction and removal are possible and thus enhancing Kleinberg’s original model of fixed-distance long-links. Moreover, there is no global information required.

All proposed algorithms are able to work effectively in modern overlay networks, since they are distributed, flexible, locally executed and support highly dynamic networks. In the end of this thesis, several ideas of related application utilizing the proposed algorithms will be presented in order to provide additional benefits from this work.
Chapter 1

Introduction

1.1 Background and Motivation

During the last few years, overlay networks have gathered much interest in the research and industrial community due to several advantages compared to existing network infrastructures. An overlay network is a virtual network built on top of another network by end-users or third party service providers. The overlay system aims to provide an application or service which is either difficult or impossible to offer using traditional infrastructures such as the deployment of new services, which can be done without requiring universal changes or coordinated actions. Well known overlay network applications include simple file sharing as well as more sophisticated content delivery networks. In addition, some overlay networks serve to improve quality of service and routing performance. Overlay applications require more complex and specific networking functionalities to support lookup and routing mechanism as the core components of overlay systems in distributed and decentralized networks. The most famous overlay network design principle is the Peer-to-Peer (P2P) system, which is composed of a group of members (peers) participating and collaborating without centralized control for the specific purpose or requirement. Peers are equally privileged and both offer and access system resources. Modern P2P systems provide various features such as a robust routing architecture, efficient searching and routing mechanisms, redundantly storing, load-balancing, as well as implementing trust and authentication. For examples, Internet users use BitTorrent to distribute bulk data such as software updates, data sets, and media files to many nodes [22]; commercial P2P software allows enterprises to distribute news and events to their employees and customers; millions of people use Skype to make video and phone calls [99]; and hundreds of TV channels are available using live streaming applications such as PPLive [98] and the BBC’s iPlayer [96].

The three main components of all P2P systems are neighbor finding, a lookup protocol, and a routing protocol which diverse P2P architecture offers different attributes. Early P2P systems were often not fully decentralized, but were a hybrid architecture between P2P and Client/Server. The object indexes were collected in a centralized location. When a new node joins, it informs the server about its stored objects. Searching an object requires to query the
desired peer’s position from the server. Such architecture is very simple and easy to deploy, but it has the problem of single point failure which only can be eased a little by using several parallel servers. Current P2P systems are totally distributed and can be classified into structured and unstructured systems [68, 104]. In structured P2P systems, a logical topology is organized, as for example a mesh, ring, $d$-dimension torus, and butterfly. These structured topologies often provide distributed hashing table (DHT) functionality to support the lookup process. The stored objects (or their hash-values) are used to identify the relationships between nodes and for searching and routing control. The query guarantees to be successful after some deterministic hops in the ideal case. Such structure is capable of finding data, but a potential bottleneck and a single point of failure and attack can be occur. Moreover, any node leaving the community without notification affects the functioning of the object lookup. In contrast, unstructured P2P also known as ad-hoc systems, peers are connected randomly that can potentially scale very well. They use flooding or random search on the graph to find contents. The query is executed hop-by-hop through the system until it succeeds, is lost or reaches its time-to-live. This type has no single point of failure but the query efficiency may be low. In addition, such broadcasting techniques cause heavy traffic or may be unable to find rare and remote contents.

The main advantage of structured P2P overlay networks lies in their ability to distribute arbitrary contents over a dynamically changing number of participants and still provide efficient lookup mechanisms [68]. Unfortunately, in structured overlays, such as CAN and Grid-like structures, the routing process can cause single peers to have a high message load, since each may have a central or otherwise crucial position in the network so that a lot of messages are routed to or through it. This problem is enforced, whenever a peer manages content that is accessed by a lot of users in the whole network. The peers around such hot-spots are inherently exposed to higher routing load, since a lot of messages need to be routed to and from the hot-spot. Whereas some messages definitely have to be forwarded to the hot-spot, other messages should be routed around it. This not only avoids additional load and possible messages loss in the stressed region, but also decreases the delay time for the redirected messages. On the other hand, the alternative routes should still have a minimum number of hops to make sure that no messages are lost due to time-to-live (TTL) expiries.

To solve these kind of problems, the research community focuses on balancing traffic and load in the system. Several methods have been proposed. As mentioned, load balancing problem is could be solved by duplicating popular items in multiple nodes but it is costly due to a full price of construction and maintenance content in multiple locations. Furthermore, other solutions are to construct effective overlay architecture to prevent load problem as well as to improve satisfaction of QoS routing as shown in [43, 66, 93, 114]. Among all of these solutions, the small-word network model is utilized. The small-world phenomenon is described by the hypothesis that everyone in the world can be reached through a short chain of social acquaintances. The analogy of P2P networks and social networks is clear, when peers are seen as people and connections as relationships. Watts and Strogatz [106] first proposed a
construction algorithm for small-world that created graphs with both a small average path length as well as a high cluster coefficient. Later, Jon Kleinberg proved that routing efficiency can be improved remarkably if nodes know their local neighbors (local-link) and some distant nodes (long-link) chosen at random according to the $d$-harmonic distribution [50]. His models base on a two-dimensional grid ($n \times n$) in which each peer has undirected local links to its neighbors, and directed long links randomly generated. The link distribution is non-uniform and follows a power law, such that short-distance links are preferred over long-distance links. In such structures, Kleinberg shows that using a greedy algorithm and only local information, it is possible to find a route between any two nodes within $O(\log^2 n)$ complexity. In [71, 77, 115], his model is extended and the routing performance is improved. The small-world paradigm is applied not only to unstructured P2P overlays [26, 105], but also to structured P2P [43, 93, 107].

Although several solutions have been proposed, there are still open problems in supporting QoS routing in overlay networks. One of open issues is to find the best route on virtual systems which generally known as a multiple constraint optimal path (MCOP) problem. The constraints, for instance, overall routing delay, the number of hops, and transfer rate, usually are entailed by application specific quality-of-service (QoS) requirements. QoS routing in overlay networks is slightly different from QoS Routing in the TCP/IP infrastructure since overlay networks are composed of virtual links and overlay nodes without direct relation to the underlying physical network.

The challenge addressed in this thesis is to utilize overlay systems for large-scale content searching and routing efficiency. Both overlay infrastructure as well as routing decision methodologies are considered. While several protocols exist to construct overlay systems, often such architectures are fixed or inflexible in dynamic traffic situation. In this thesis, several criteria from both the overlay infrastructure and the peer’s properties are taken into account for organizing the architecture and for routing the messages. Our objective is to improve QoS routing satisfaction supporting especially complex queries or unpopular keywords. The proposed routing algorithms however are not limited to be executed in grid-like overlay P2P system, it can be used in any overlay systems where the distance between any two nodes can be measured or at least predicted.

Grid-oriented structured P2P overlays like CAN and Kleinberg’s small-word model show that the two-dimensional grid topology is efficient for routing. Such structures provide many benefits; redundancy, prevention of deadlocks, network balancing and fault tolerance. Additionally, when using a grid structure’s coordinate system to identify peers, each peer can predict the shortest route to any other peers without prior communication. The coordinate system can be projected by a peer’s content (e.g. Electronic Product Code), keywords or geography (e.g. latitude and longitude). To take benefit from the grid with coordinate system as well as utilizing efficiency of existing P2P overlay protocols, this thesis proposes methodologies executed on a grid overlay network on top of P2P systems. The logical grid with coordinate system could be built on top of decentralized networks using only local knowledge of peers [11, 12] without centralized control.
The main contribution of this thesis is categorized into three parts:

1. A thermal field approach is introduced to represent and distribute a peer’s properties such as bandwidth, buffer size, and content popularity (e.g. PageRank or number of content download). In this thesis, the thermal field approach is used to propagate the buffer utilization level.

2. Multiple criteria routing algorithms considering several constraints from physical layer as well as overlay layers are proposed. The first approach selects a single criteria at each routing step by using pre-defined probabilities. In the second approach, all criteria are combined into a decision function to find an appropriate trade off among the constraints.

3. In addition to routing algorithms, this thesis proposes a self-organizing overlay structure dynamically adapting to the traffic in the system. Shortcuts are constructed across areas with high traffic load and removed, when they are no longer necessary.

Beside above key contributions, this thesis gives additional ideas for overlay P2P systems by extending Kleinberg’s small-world model. An ant colony approach is used to distribute information about existing long-links, and a human behavior model is considering for constructing the long-link in the overlay network.

1.2 Main Contributions

1.2.1 Thermal Field Approach

Cisco’s latest traffic forecast reports that global traffic will reach 966 exabytes per year or 80.5 exabytes per month in 2015 [81]. The traffic will grow at an average rate of 32% from 2010 to 2015 which means it will increase fourfold over the next 5 years. While P2P traffic currently holds the largest share of Internet traffic, it will decrease relatively. Internet video streaming and downloads are beginning to take the largest share of bandwidth, such that the global networks will deliver 7.3 petabytes every 5 minutes in next five years.

The challenge is to design lookup mechanisms and routing protocols for supporting sharing such massive data over the Internet. When modeling such problems, usually available bandwidth and hop count are used as metrics, but only few approaches consider the buffer stage. Still, routing huge amounts of data can overload communication buffers and so lead to packet delay and loss. Hence, in this thesis the thermal field approach is used to communicate buffer utilization level among a peer’s neighbors. A lower temperature indicates that more resources are available and the peer is more capable of handling new data. The temperature gradient then indicates the direction to overloaded peers and thus can be integrated into route decision. The cost of thermal field approach occurs due to peers have to memorize its neighbor’s temperatures. However, it is very small when there are not more than four neighbors in gird-like overlay P2P network.
Those temperature data could be delivered as a piggyback and thus would not generate any remarkable overload.

1.2.2 Multiple Criteria Routing in Overlay P2P Networks

The goal of routing is generally to find the fastest path. However, the shortest path often leads to bottle-necks or to overloaded region. So, we make use of the buffer level information distributed by the thermal field, to decide whether to select the shortest route or the path leading to neighbors with lower buffer level and thus avoiding a long queue. The first set of approaches bases this decision making on pre-defined probabilities. First, a constant probability is used [56]. Then, the idea is adapted to a probability function of the remaining distance to target. The approach is based on the assumption that the closer the message is to its target, the more often the direct path should be used, even if this leads into high traffic areas [57].

Several approaches for multiple constraint based routing have been introduced in this area [37]. Expert systems, swarm intelligent systems, artificial neural networks, and fuzzy logic have been applied for multi-constraints decision making. Several constraints are taking into account concurrently to find an optimal path. In this thesis, weight functions and fuzzy techniques have been chosen [53, 54, 55, 59, 101]. Functions of two constraints, temperature and distance, are used to calculate the weight of the next hop instead of applying only a single constraint value. A weight is assigned to every neighbor based on the neighbor’s distance to target as well as its temperature. The message is then routed to the neighbor with the minimum weight. Each weight is calculated as a linear combination, where the coefficients control the influence of each summand on the total weight. Priority of each criterion is defined by factors.

However, the challenge is to find an optimal proportion between distance and thermal along the path of weight function in a dynamic traffic situation. Thus, fuzzy logic is chosen to handle such problem. Fuzzy logic known to be efficient when dealing with uncertainties. Additional advantages of fuzzy logic are that it is conceptually easy to understand, flexible, and tolerant toward imprecise data. It can model nonlinear functions of high complexity, and it also can be built on top of expert’s experience. Hence, we finally proof that using a fuzzy-based decision making mechanism for multiple-criteria-routing suits best for current overlay systems.

1.2.3 Self-organizing Overlay Topology

Based on the idea that the structure of the overlay itself can dynamically be changed to improve QoS routing, the last main contribution is inspired by Kleinberg’s small-world to implement an algorithm for organizing long distance links automatically [61, 62]. A bridge across an intersection in Bangkok gave the inspiration for a bridge algorithm, in which overloaded peer are the analogue to the traffic junction. In the algorithm, long-range links are added and removed according to network conditions. This novel algorithm creates bridges (long-links),
when an overloaded peer is notified through the thermal field. Both ends of a bridge are assigned to high performance peers represented by low temperature. When bridges have been built crossing an overloaded area, packets are transmitted through those new shortcuts. This way, packets can be delivered faster and with less hop-count. In addition, the whole system has better load balancing. A bridge is removed to reduce maintenance cost, when it is unnecessary.

1.2.4 Other contributions

Finally, additional extensions of the main contributions are presented. First, Kleinberg’s small-world is extended by distributed information about long-links by using ant colony system [58]. Long links are notified when messages have visited the according node. Next, the long-link is generated and deleted according to a node’s character or strategy. A network organization model with economic behavior is proposed [60]. This model is used to understand and predict the network structure from an economic perspective. This system refers to a network where a node represents a customer, a provider, or just a person, while an edge symbolizes trading, delivery method, dealing, or just a relation, and a message or traffic in the network substitutes goods, services, or information.

1.3 Thesis Outline

The remaining parts of this thesis are organized as follows. Chapter 2 gives a state-of-the-art and literature review regarding overlay networks; especially peer-to-peer systems, and quality of service routing. Afterwards, a short description of the P2PNetSim simulator is given, which has been used for all simulations within this thesis.

Next, as the first part of contributions in this thesis, Chapter 3 and 4 present novel multiple criteria routing algorithms taking shortest distance and buffer utilization into account. In Chapter 3, first, the design of the thermal field approach used to distribute buffer utilization is introduced. Then, the routing decision making algorithms that choose a single constraint per hop are shown. In Chapter 4, the algorithms combining distance and buffer levels are discussed. The first of these approach uses a weight function, the second one fuzzy logic is chosen to make the decision.

Chapter 5 presents a novel self-organizing topology algorithm for organizing shortcuts depending on the current traffic situation to improve routing performance. Further enhancements of self-organizing overlay topology then discussed in Chapter 6. Chapter 7 concludes this thesis and gives outlooks on possible future research directions.
Chapter 2

State Of The Art

2.1 Quality of Service Routing

Routing, the determination of a path to transmit data between source and destination, is one of the key functions of all computer networks. To clarify, which path is the optimal for a given situation and application, some Quality of service (QoS) metrics have to be defined and thus help to guarantee the ability of a network to deliver predictable results. QoS routing goals are to select routes able to meet particular QoS requirement and to maximize the network utilization. The complexity in QoS routing comes from considering multiple criteria to meet multiple objectives, which is also called “multi-constrained optimal path problem (MCOP)”. Some common metrics in the computer network field are as follows.

- **Hop Count** simply counts intermediate hops (nodes or devices) which the data passes from one point to another. A path with minimal hop-count is preferred because it reduces the use of network resources as well as the path delay.

- **Bandwidth** is usually referred to the amount of data that can be transported in a given time period in computer science. A bandwidth metric would prefer a higher-bandwidth path over a lower-bandwidth link, but could also used to reserve a specific amount of bandwidth for a particular time.

- **Load** reflects the amount of traffic utilizing the links along the whole path. The best path would be the one with the lowest load.

- **Delay** is a measure of the time necessary for a packet to traverse a route. There are several types of delay such as processing delay, queuing delay, transmission delay and propagation delay.
  - **Processing delay** is the time required to process the arrived packets in a node.
  - **Queuing delay** is the time a packet experience at a queue waiting to be transmitted.
  - **Transmission delay** is the time required to transmit all of the packet bits into the
State Of The Art

- **Propagation delay** is the time required to propagate a bit from the beginning to the end of a link.

All of these delays are fixed values, except for the queuing delays, which are variable.

- **Reliability** measures the likelihood that the link will fail in some way and can be either variable or fixed.

- **Cost** is a more abstract concept to reflect more- or less-preferred routes. Cost may be defined using any policy or link characteristic or may reflect the arbitrary judgment of the network administrator.

- **Throughput** is the rate of successful message delivery over a communication channel. The throughput is usually measured in bits per second (bps), and sometimes in data packets per second or data packets per time slot.

- **Dropped packets**: The nodes or routers might fail to deliver (drop) some packets if either data is corrupted or buffer overflows occur. The receiving application may ask for this information to be retransmitted, possibly causing severe delays in the overall transmission.

In theory, QoS routing with multiple additive metrics is NP-complete. Hence, many developments focus on pseudopolynomial time algorithms, heuristics, and approximation algorithms for the construction of multiconstrained QoS paths [14, 37, 73, 82, 91, 110]. The QoS requirement relies upon application types, so that a lot of efforts have been dedicated to find different QoS models and metrics for different applications according to their needs. Many presented approaches mainly consider available bandwidth or the hop-count [65, 82, 91]. Only few approaches, like [110], take buffer utilization into account. They used fuzzy logic for finding the best path which has the highest degree of the path usability when considering buffer utilization and number of hops.

The DiffServ [29] and InServ [45] are protocols implemented to support modern QoS Routing in packet-switched IP networks. However, they are difficult to deploy in large-scale networks because the physical infrastructure has to be changed to make effective use of them. To overcome such problems, overlay networks has been emerged as an effective methodology.

### 2.2 Overlay Networks

An overlay network is a logical network on top of another (usually TCP/IP) infrastructure, which performs the basic networking functions, e.g., routing and forwarding messages. The overlay provides an infrastructure for one or more applications and takes responsibility for forwarding and handling of application data in different ways from the basic Internet [19]. The overlay systems have become well known with the three influential systems in 1999: The Napster
music-sharing system, the Freenet anonymous data store, and the SETI@home volunteer-based scientific computing projects [86]. More than a decade later, overlay technologies have gone far beyond; it now enjoys significant research attention and increasingly widespread use in open software communities and industry including distribution of data, media, telephony, and scientific computing in both research and commercial areas.

Overlay systems are built in the application layer of the Open Systems Interconnection (OSI) model. Nodes in the overlay can be connected by a single virtual link, which may take many physical links in the network layer. Fig.2.1 presents a layered view on overlay systems [68]. The basic infrastructure under the overlay network is the network layer (e.g. UDP and TCP) which offers the basic primitives of sending and receiving messages (packets). This layer consists of all globally distributed resources that are directly connected to the Internet. These resources include desktop machines, wireless- and sensor-based devices, storage devices, supercomputers, clusters, etc. The Overlay Nodes Management layer covers the management of peers, which include discovery of peers and routing algorithms for optimization. This relates to the custom routing, forwarding, rendezvous, and discovery functions of the overlay architecture. Additionally, this layer offers core services for indexing resources at Internet scale. Routing relates to
the process of building and maintaining routing tables. Forwarding is the process of sending messages toward their destination, and rendezvous is a function that is used to resolve hidden peer issues, e.g., coming from packet filtering and network address translation. Some example overlay protocols are CAN [85], Chord [92] and Viceroy [70] which provide distributed hash table (DHTs), while Gnutella [38] and Freenet [20] provide content-based queries. Other protocols are JXTA [84], P-Grid [1], and Structella [16].

The Feature Management overlay layer introduces additional functions, such as, security and resource management, reliability support, fault tolerance, and aggregated resource availability aspects of maintaining the robustness of P2P systems. Security pertains to the way node identities are assigned and controlled, and message and packets are secured. Resource management is about taking content demand and supply into account and ensuring that certain performance and reliability requirements are met. The Services Specific layer supports the underlying P2P infrastructure and the application-specific components through scheduling of parallel and computation intensive tasks, content and file-management. This layer provides mechanisms for both monitoring and controlling service life cycles. When a service is deployed on top of an overlay, there need to be functions for administering it and controlling various issues such as administrative boundaries, and data replication and access control policies. Finally in the top-most layer, the Application-level layer is concerned with tools, applications, and services that executed on top of the layered overlay architectures which supports for scalable and resilient data discovery and exchange. The applications are implemented with specific functionalities in various domains, as exemplarily shown in Table.2.1 [19, 86].

The most popular overlay network applications are file sharing and huge data distribution systems. Some examples of overlay networks widely used for disseminating data, software, or media content are Napster [78], Gnutella [38] and Bittorent [22]. Other systems are video streaming applications used for streaming media distribution like digital television service over the Internet (IPTV). Some academic efforts with widespread adoption are PPlive [98] and Cool-Streaming [111], some commercial products are BBC’s iPlayer [96] and Skinkers LiveStation [97].

Content delivery networks (CDNs) such as Akamai [2] are systems containing copies of data placed at various nodes of a network in order to improve access to the data by increasing access bandwidth and redundancy and reducing access latency. Data content types often cached in CDNs include web objects, downloadable objects (media files, software, documents), applications, live streaming media, and database queries. RON (Resilient Overlay Networks) [6] is an example of an overlay network for improving effective routing in large-scale networks. RON is part of a larger research agenda on large-scale, robust, Internet-based distributed systems, which spans all areas of resilient routing. RON is an architecture that allows a small group of distributed Internet applications to route packets directly over the Internet or by way of other RON nodes, optimizing application-specific routing metrics. Another major use of overlay network is for making audio and video calls, popularized by the Skype application. Skype exploits the resources of participating nodes to provide seamless audiovisual connectivity to
Table 2.1: Example of Overlay Networks

<table>
<thead>
<tr>
<th>Type</th>
<th>Objective</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>File sharing</td>
<td>Sharing media and data.</td>
<td>Napster, Gnutella, KaZaA, BitTorrent, PPLive, CoolStreaming</td>
</tr>
<tr>
<td>Content-delivery network(CDN)</td>
<td>Content caching to reduce delay and transport costs.</td>
<td>Akamai, Digital Island</td>
</tr>
<tr>
<td>Routing</td>
<td>Reduce routing delays and cost, resiliency, flexibility</td>
<td>Resilient Overlay Network (RON), Akamai SureRoute</td>
</tr>
<tr>
<td>Security</td>
<td>Enhance end-user security and offer privacy protection</td>
<td>Virtual private networks (VPNs), Onion routing (Tor,I2P), Anonymous content storage (Freenet, Entropy), Censorship resistant overlays (Publius, Infranet, Tangler)</td>
</tr>
<tr>
<td>Experimental</td>
<td>Provide testing environment for new technologies.</td>
<td>PlanetLab</td>
</tr>
<tr>
<td>Telephony</td>
<td>Making audio and video calls.</td>
<td>VoIP (Skype)</td>
</tr>
<tr>
<td>Others</td>
<td>Enhance communications</td>
<td>Email, Multicast (MBone), Delay tolerant networks, etc.</td>
</tr>
</tbody>
</table>

its users. Peers assist those without publicly routable IP addresses to establish connections, thus working around connectivity problems due to firewalls and network address translation, without requiring a centralized infrastructure that handles and forward calls.

Overlay applications require more complex and specific networking functionalities on top of the basic IP-based infrastructure, especially to support lookup and routing mechanisms, when the core components of overlay systems are organized in a distributed and decentralized manner. Such Peer-to-Peer (P2P) systems are the most widely known overlay networks and currently have the biggest traffic share on the Internet.

2.2.1 Peer-to-Peer Systems

The earliest design of P2P system, used by Napster [78], actually is a mixture between client/server and P2P, since it maintains a centralized index of all contents available in the system. When the content location is centralized, it is easy to maintain. But when simultaneous huge number of clients requests to a given server increases, the server can become overloaded and is a single point of failure. In contrast, current P2P systems are organized in a totally decentralized manner. The systems mainly are classified based on the distribution of content indices among all members into structured and unstructured types. The diversity of these approaches bases upon different content location mechanisms, but also on different lookup and routing techniques.
Fig. 2.2 shows the classification of decentralized P2P systems which is firstly distinguishing between structured and unstructured systems. Both systems have been surveyed and discussed in many P2P articles [19, 37, 68, 95]. They are divided by topology and lookup methodology. The unstructured P2P does not have a controlled topology. It utilizes, e.g., flooding searches or content-based queries for finding nodes that contain the desired data. In contrast, structured networks built a specific structure that allows for directed searches (e.g. DHTs). They maintain a topology, such that contents are kept at specific locations and thus leading to efficient queries. For both types, more details are given in the next section. In recent developments, new generations of P2P systems have combined both unstructured and structured P2P networks into Hybrid types. For examples, Gnutella2 (G2), FasTrack [83], and Structella [16] are hybrid P2P protocols. The peers form a hierarchic structure in FasTrack protocol which used by the KaZaa [49]. Super-peers are peers with high bandwidth, disk space and processing power, and have volunteered to get elected to facilitate search by caching the meta-data. All queries are forwarded to the super-peers and broadcast among super-peers for searching content. Structella replaces the random graph model of Gnutella with a structured overlay of Pastry, while retaining the search and content placement mechanism to support complex queries. The constrained flooding and random walks are discovery mechanisms.
2.2.2 Unstructured P2P Systems

Unstructured P2P networks are composed of peers joining the network without rules and prior knowledge of topology. This type has no pattern control and is typically based on random graphs following a flat organization. Unstructured P2P applies opportunistic techniques, such as flooding or broadcasting, random walks, or expanding-ring Time-to-Live (TTL) searches in order to identify interesting locations. Those system often rely on keyword-based searching of the queried objects in the network. Such keyword-based systems can simply query keywords, but also keyword frequencies or relation to a normalized keyword-vector. Each peer that is contacted through a search evaluates the query locally on its own content. A flooding request model, even when it is limited or directed, for decentralized P2P overlay systems is quite effective to locate popular data objects. However, it leads to excessive network bandwidth consumption, and remote or unpopular objects may not be found due to the TTL limit. One benefit of unstructured networks is their ability to support variable kinds of query languages, whereas structured overlay generally require additional query-processing layers on top of the basic overlay. Some of the unstructured P2P systems are:

- **The Freenet protocol** [20] does not assign responsibility for data to specific peers, so key-based lookup searches for available cached copies within the so-called Hops-to-live (HTL) limits. Freenet provides anonymity and uses an index scheme where files are identified by content-hash keys and are secured by signed-subspace keys to ensure that only one object owner can write to files, but anyone can read it. Each peer maintains a dynamic routing table, which contains address of other peers and the data key that they are holding. The key features of Freenet are the ability of maintaining locally a set of files in accordance to the maximum disk space allocated by the network operator, and to provide security mechanisms against malicious peers. The routing algorithm is designed to adjust routes over time and to provide efficient performance while using only local knowledge, since peers only have knowledge of their immediate neighbors. Thus, the routing performance is good for popular content. Example applications using Freenet protocol are Freenet [21] and Frost [36].

- **Gnutella** [38] initial version is a decentralized protocol for distributed search on a flat topology. The participants join the system and self-organize in a virtual mesh network. New node first must connect to a known Gnutella node to get lists of some existing nodes to connect with for start-up. To find a desired content, a node issues queries to all neighbors within a given TTL. Gnutella can support partial-match queries. This protocol is very resilient to nodes entering and leaving the system. However, the current flooding-based lookup mechanism is un-scalable, since it generates large load on the network participants.

- **BitTorrent** [22] achieves high level of robustness and resource utilization based on its incentives cooperation technique for file distribution. BitTorrent uses a centralized download management for users. The protocol is design to prevent free-riders by having the
peers choose other peers from which data has been received. Peers with high upload speed will probably also be able to download with a high speed, thus achieving high bandwidth utilization. This system has a centralized location called tracker providing all information about the content such as length, name and hashing information. Then the downloading client uses this information to connect to the peers maintaining the requested files. Example client implementations are µTorrent [76] and Vuze [100].

Although the weaknesses of unstructured type are that they make excessive network bandwidth consumption and they may fail in finding remote unpopular data, they often support more complex queries both exact-match as well as range-match. Thus, unstructured P2P protocol are more often used for file sharing than structured systems.

2.2.3 Structured P2P Systems

In structured P2P, system topology is tightly controlled and contents are placed at specified locations that will make subsequent queries more efficient. The most commonly known structured systems provide Distributed Hash Tables (DHTs) for lookup and routing mechanisms. DHT-based systems assign uniform NodeIDs to the set of peers into a large space of identifiers. Data objects are also assigned unique keys from the same identifier space. Keys are mapped by the overlay network protocol to a unique online peer. Each peer maintains a routing table consisting of its neighboring peers’ NodeIDs and IP addresses. The messages routing are forwarded greedily across overlay paths choosing the closer neighbor to the key in the identifier space. In theory, DHT-based systems can guarantee that any data object can be located in not more than $O(\sqrt{N})$ overlay hops, where $N$ is the number of peers in the system. So the lookup latency in DHT-based P2P overlay networks can be quite high and could adversely affect the performance of the applications running over it.

Another key property of structured P2P overlay network is the system topology or architecture. The overlay system topology relates to the mechanisms choosing neighbors and routes. Frequently used overlay topologies are tree, tori, butterflies, and rings [95]. The important properties of topologies are their degree of distribution and their diameter. Tree, hypercube and ring are examples of non-constant degree topology, while Tori and butterfly are example of constant degree topology. The basic topologies with example protocols are shown as follows.

- **Rings** are popular due to their simplicity. Nodes lie on a one-dimensional cyclic identifier space on which the distance between two nodes is calculated as the clockwise numeric distance on the circle. Efficient routing on a ring is based on a cyclic identifier space of $2^n - 1$ identifier. **Chord** is well-known example of a ring topology [92]. Given an object, the node responsible for storing it can be determined using a hash function that generates a key from it and thus makes it able to identify the node storing the respective object. An object is stored on the first node whose identifier $id$ is equal to or follows the object’s key $k$ in the identifier space. In Chord, each node maintains a routing table
with information for only about \((\log N)\) nodes. In fact, Chord is similar to binary search, where the searching space will be reduced half after a search/routing-hop. So the number of nodes that must be contacted to resolve a query in a \(N\)-node network is in \(O(\log N)\).

- **Tori - CAN** [85] provides a \(d\)-torus structure that is partitioned among nodes such that every node owns a distinct zone within the key space. When a node joins, an already existing node splits its region and transfers half of it to the newly added node. Each node will keep its neighbor node IDs locally, so that routing is performed by forwarding requests to the regions closest to the requested key. In this way, the expected search length is \(O(d\sqrt{N})\) with only constant amount of local information necessary. One of the main advantages of CAN is its efficiency to handle nodes in resilient large-scale networks. However, greedy routing in CAN is quite poor especially when only using few dimensions. Hence, to reduce routing latency, the authors of CAN proposed to increase the number of dimensions used, in order to increase number of direct neighbors per node.

- **Butterfly** is a \(k\)-ary \(n\)-fly network consisting of \(k^n\) source nodes, \(n\) stages of \(k^n - 1\) switches, and \(k^n\) destination nodes. The butterfly contains a binary tree with its root in the first level, where the leaves are the nodes in the last level. The weakness of basic butterfly topology is there is only one path from a source to a given destination. **Viceroy** operates in a butterfly network but adapts to be self-organizing and robust in joining and leaving nodes. The Viceroy network is a composition of an approximate butterfly network and a connected ring of predecessor and successor links. Therefore, each server has three outgoing links to chosen long-range contacts - up, left and right links, and two ring connections, one to its successor and one to its predecessor. Up, left and right connections are considered the ‘butterfly’ links and the predecessor and successor are denoted as ‘ring’ links. Locating resources is done in \(O(\log N)\) hops.

- **Tree** topologies, node identifiers comprise of the leave nodes in a binary tree. The distance between any two nodes is the height of the smallest subtree containing both nodes. Each node contains \(\log N\) neighbors in a routing table, where the \(i^{th}\) entry refers to a node of distance \(i\). The plaxton-style is based on this geometry. An example protocol is **Tapestry** [113]. The core location and routing mechanisms of Tapestry are similar to basic Plaxton. But Tapestry aims to improve the capability to detect, circumvent and recover from failures through maintaining periodically updated cached content. To detect link and server failures during normal operations, Tapestry can rely on TCP timeouts. Additionally, each Tapestry node uses back-pointers to send periodic heartbeats as UDP packets to nodes for which it is a neighbor. By checking the ID of each node, a message arrives at, we can quickly detect faulty or corrupted neighbor tables. Tapestry assigns multiple roots to each object. These roots are used during the publishing process to insert location information into the system. When locating an object, Tapestry performs the same hashing process with the target object ID, generating a set of roots to search.

Overlays that perform query routing in DHT-based systems have strong theoretical foundations,
guaranteeing that a key can be found if it exists and they do not capture the relationships between an object name and its content. However, they have some problems in terms of data object lookup latency. First, peers route messages to the next overlay hop that can be located very far away with regard to the physical topology of the underlying IP network. This can result in high network delays and unnecessary long-distance network traffics, from a deterministic short overlay path of $O(\log N)$. Moreover, the DHT-based systems assume that all peers equally participate in hosting published data objects or their location information. This leads to bottlenecks at low capacity peers. P2P DHT-based overlay systems are sensitive to security breach from malicious peers’ attacks. One simple attack on DHT-based overlay system is that the malicious peer returns wrong data objects to the lookup queries. Malicious peers may be able to corrupt, deny access or response to lookup queries of replicas of a data object so that replicas may be stored on illegitimate peers.

2.2.4 Search and Routing Extension of P2P Systems

The comparison of example P2P overlay networks which summarize the previous section shows in Table 2.2. The system properties and routing performance are compared as well as security issues.

The basic structured systems like CAN and Chord are DHT-based systems are more efficient at many tasks and have strong theoretical base for guaranteeing routing time. The key identifier is built from semantic object relationships between its name and its content or metadata. They use precise placement algorithms and specific routing protocols to make searching efficient. However, global system-wide parameters are required to build and control the structure. Other challenges of structured P2P are lookup latency, bottleneck from low capacity peers or high popular data, and security issues (e.g. man-in-middle and Trojan attacks). In addition, DHT-based system cannot support complex queries or keyword searches which is often required in file sharing applications more than exact-match key searches. To implement DHT-based systems support searching of rare data, they incur significantly higher overhead than unstructured networks. Consequently, unstructured overlay networks are more commonly used in mass-market file sharing applications. And such approaches are quite effective to locate popular data objects due to the power-law property. However, flooding and random walks approaches in unstructured systems are not scalable as the load on each peer grows linearly with the total number of queries and the system size. Thus, the overall system performance is not effective when handling a high rate of aggregated or unpopular queries or when the system grows.

Due to the disadvantages mentioned above, the research community focuses on improving P2P systems in terms of lookup and routing performance in many areas. Several extensions of unstructured P2P overlay aim to include flow control, dynamic geometric topology adaptation, on-hop replication, peer heterogeneity, etc [17, 69]. The original flooding approach has been replaced by multiple random walks in Gnutella in order to reduce network traffic [69]. Chen et al. [17] introduce a self-learning semantic search method (SLPS), which learns similarity of
Table 2.2: Characteristic of Structured P2P Overlay Networks [68]

<table>
<thead>
<tr>
<th>Protocol</th>
<th>CAN</th>
<th>Chord</th>
<th>Freenet</th>
<th>Gnutella</th>
</tr>
</thead>
<tbody>
<tr>
<td>Topology / model</td>
<td>Muti-dimensional ID coordinates space.</td>
<td>Ring; uni-directeral and circular Node ID space.</td>
<td>Flat and Ad-hock network of peers.</td>
<td></td>
</tr>
<tr>
<td>Lookup protocol</td>
<td>DHT-key,value pair to map a point P in the coordinate space using uniform hash function.</td>
<td>DHT-Matching Key and NodeID.</td>
<td>Keys, Descriptive Text String search.</td>
<td>Query flooding.</td>
</tr>
<tr>
<td>System parameter</td>
<td>$N$-Number of peers, $d$-number of dimensions</td>
<td>$N$-Number of peers</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Routing performance</td>
<td>$O(d \sqrt{N})$</td>
<td>$O(\log N)$</td>
<td>Guarantee to locate data using Key search until Hops-to-live (HTL) limits</td>
<td>No guarantee to locate data; Good performance for popular content.</td>
</tr>
<tr>
<td>Routing state</td>
<td>$2d$</td>
<td>$\log N$</td>
<td>Constant</td>
<td>Constant</td>
</tr>
<tr>
<td>Peer join/leave</td>
<td>$2d$</td>
<td>$(\log N)^2$</td>
<td>Constant</td>
<td>Constant</td>
</tr>
<tr>
<td>Security</td>
<td>Low level. Suffer from man-in-middle and Trojan attacks.</td>
<td></td>
<td>Low; Threats: flooding, malicious content, attack on query, and denial of service attacks.</td>
<td></td>
</tr>
</tbody>
</table>

peers from historic search results, and computes relations, which can be used to locate content more efficiently.

In addition, a lot of research works want to improve routing and load balancing of structured overlay networks [13, 39, 41, 88, 94]. An example simple idea is to replicate the same data items in multiple nodes by using different hash functions. The routing performance remarkably increases, but at the price of a full replication of the content in more than one location. Sahin et al. [88] improve CANs routing performance by maintaining information about the load of neighbored peers both during a node’s join process as well as periodically afterwards. In [39], a scheme is introduced to partition the key space according to the node performance (e.g. storage or capacity), when nodes join and leave. A semantic overlay network, named pSearch, uses the feature vectors of documents as coordinates in CAN space to support semantic retrieval [94]. Similar to Meghdoot [41], Gupta et al. implement a content-based publish/subscribe system
using DHTs by distributing events and subscriptions based on their attribute values in a CAN-like space. Boukhelef and Kigagawa [13] propose RCAN, an extended basic CAN, by adding long links which are directed clockwise to form small rings along each dimension (multi-ring) to improve routing performance while minimizing the maintenance overhead caused by churn.

In summary, many research works focus to improve lookup and routing performance by adapting structural properties. Gummadi et al. analyzed, how overlay topology affects routing performance in terms of proximity and flexibility [40]. They compared routing along a hypercube (when the dimension is taken to be $\log n$) of CAN, a ring structure of Chord, a butterfly network of Viceroy, and tree-like structure of PRR’s algorithms. On their assumption, these geometries have different degrees of flexibility in choosing neighbors and next-hop paths. And they found out that, the ring allows the greatest flexibility, and achieves the best resilience and proximity performance.

To enhanced topology, the small-world model from Jon Kleinberg [50] is an often cited model that has been applied to improve P2P systems performance in addition to Watts and Strogatz’s model [106]. Kleinberg introduced a two-dimensional torus grid with additional long-range links and proved that such topology is able to forward message from any source node to target within limited time using a simple greedy routing algorithm. Therefore, in the next section, details of Kleinberg’s small-world as well as some extensions to improve the performance of the original model and P2P systems will be discussed.

### 2.3 Small-World Networks

The small-world phenomenon was first proposed by Stanley Milgram in 1967 [74]. He set out to measure the “smallness” of the world. He wanted to know if it was really true that any two people could be connected through a short chain of acquaintances. To conduct this experiment, he gave volunteers living in Omaha, Nebraska, a letter addressed to a stockbroker from outside Boston, asking them to forward it to him, with the constraint that the letter could only ever pass between people who were on a first name basis. The results of his experiment were generally seen as proof that we really do live in a small world - for the letters that arrived successfully, the average number of steps was just six. Since then, a number of network models have been proposed as frameworks in which to study the problem analytically. One of the most refined of these models was formulated in 1998 by Watts and Strogatz [106]. They proposed a model for creating small-world graphs based on a class of random networks that are strongly connected locally and additionally have some random long-range contacts. They claimed that such a model captures two key parameters of social networks: a simple underlying structure that represents of most edges, and a few edges that do not follow that structure created by a random process. In 2000, Jon Kleinberg [50] introduced a family of small-world models building on the model of Watts and Strogatz, and proofed that one model of the families is a decentralized algorithm which is able to find a short path within finite steps.
2.3.1 Kleinberg’s Small-World Model

Kleinberg’s model [50] is based on a two-dimensional grid \((n\times n)\) with undirected local links to direct neighbors and some directed long link randomly generated, as can be seen in Fig. 2.3. The shortcut is constructed from nodes \(u\) to random endpoint \(v\) with a probability proportional to \(d(u,v)^{-2}\), the inverse square of the lattice distance of \(u\) and \(v\). Links have a non-uniform distribution that prefer closer node over distant ones, such that the distribution follows a power-law.

![Figure 2.3: Kleinberg’s Small-World Network [50]](image)

In such structures, Kleinberg shows that a greedy algorithm is able to find the route between any two nodes within \(O(\log^2 N)\) time. The route decision making process in each node uses only local information: (i) the lattice structure (two-dimensional torus), (ii) the coordinates of existing long-range neighbors and (iii) the coordinate of the target. The decision node does not know the long-range neighbor of any node that has not yet been visited. Kleinberg extended the 2D lattice model to a tree network and a general group structure model in [51]. However, the exact topology information is required to calculate the probability of connecting a long link with a remote node.

Although Kleinberg’s small-world models can provide an effective routing performance, there are some disadvantages.

- Small-world construction needs to have global knowledge about all existing nodes to compute the probabilities for long-range contacts. It is very difficult to get this global information especially in practical distributed networks.

- Long-range links are fixed at the construction time. The small-world topology does not change since it has been built.

- Routing using greedy algorithm in such static topology allows for unbalanced load and hop-spots in the system.

Hence, there are several recent approaches that analyzed and extended Kleinberg’s model, not
only according to the routing algorithm but also regarding the overlay architecture.

### 2.3.2 Extension of Small-World Networks

Kleinberg’s small-world model has been extended in many perspectives. First, the routing time boundary could later be tightened to $\theta(\log^2 N)$ [8, 71]. Martel and Nguyen [71] used additional knowledge of the graph they show that the expected path length can be reduced to $O(\log^{1+1/k} N)$ for a general $k$-dimensional model ($k \geq 1$). Barriere et al. [8] use a ring rather than the mesh while preserving other features of the model. In [52] and [77] routing performance is improved by using neighbor’s information to gain better routing performance when using a greedy algorithm. Zeng and Hsu [109] extended Kleinberg’s small-world model by adding two more random links in each node chosen uniformly and randomly within $(\log N)^{2/d}$ Manhattan distance from the respective node, where $d$ is the dimension of the model. Then the optimal expected bound for routing decreases to $O(\log N)$. Additionally, the network has a better fault tolerance. Zou et al. [115] claimed that Kleinberg’s model needs to use global information to form the structure. Consequently, they proposed to use cached long distance links instead of fixed ones. The structure is refined as more queries are handled by the system.

In addition, the long-range link concept from the small-world model is applied to P2P architecture. In [43, 44, 66, 67], the problems of structured P2P networks were addressed by maintaining an additional multilayer topology. Hongjun Lui et al. [66] used a rigorous binary tree code algorithm to improve search capability by organizing peer nodes into different peer groups. These group-based systems were built on top of structured P2P networks, acquired the efficiency of structured P2P protocols and achieved an enhanced performance. In [43], a small-world routing model (SWRM) is proposed, which is based on Chord for solving three issues: high lookup failure rate, low effective proximity routing, and poor proximity neighbor selection by classifying nodes into super and common nodes. Yuhua Lui et al. [67] applied the small-world model to unstructured P2P networks for information discovery, peer nodes are freely linked to each other in inter-groups and not every peer node needed to be connected to remote groups and also could easily find the information in remote peer nodes through some leader peer nodes. It reduced the average distance as well as information losses. Hui et al. [44] have built a small-world on top of existing structured P2P networks by classifying peer nodes into clusters, which achieved improved lookup performance over existing protocols.

Since the CAN topology is similar to the small-world lattice structure, additionally introduced long-range links could be effectively used. Sun has introduced SCAN, a structured P2P overlay that augments the CAN overlay with long links based on Kleinberg’s small-world model in a $d$-dimensional Cartesian space [93]. His construction of the long link model does not need the estimation of network size like Kleinberg’s model does. He proofed that queries in multi-dimensional data space achieve $O(\log N)$ hops by equipping each node with $O(\log N)$ long links and $O(d)$ short links. Xu and Zhang [107] introduced eCAN for achieving $O(\log N)$ hops by building express-ways in the CAN overlay. The self-organized expressway (long link)
construction depends on the joining process of nodes. Zhuge and Sun [114] present a virtual ring, which is built on-top of the base topology according to the distance metric, then they build small-world long links in the virtual ring and map the links back onto the real network to construct the small-world routing table for achieving logarithmic greedy routing. The virtual ring method was applied to the based topologies of $d$-torus (CAN) with Manhattan distance and other base topologies including the unbalanced $d$-torus and the ring topology with tree distance.

### 2.4 Grid structure on top of P2P System

In the previous section, Kleinberg’s small-world model was presented that shows that the grid topology is efficient for routing. Mesh or Grid-like topologies have been widely used in communication networks. The functions of routing algorithms in general are the provision of the fastest path, prevention of deadlocks, low latency insurance, network utilization balancing, and fault tolerance. Guided by these classical methods, grid-like structures provide multiple paths which have the same hop count. An example of deterministic routing method in grid is $XY$-routing where packets are routed along $X$ direction until reaching the $X$ value of the target and then route the packet in $Y$ direction to the target. Such basic method can be refined to the partial-adaptive routing algorithms, for instance, “West-First”, “North-Last”, and “Negative-First” approaches [28]. These methods change packet routes dynamically by using a function that reacts immediately on network traffic, but in predefined conditions they use the deterministic ways.

The main advantage of grid-like overlay system lies in their ability to distribute arbitrary contents over a dynamically changing number of participants and still provide efficient lookup mechanisms. Especially, the inherent coordinate system enables each peer to predict the shortest route to target without prior communication. The coordinate system can arbitrarily be defined as, for example, by grouping content, by hashing, by geographic location, by using the Network Coordinate System [75], or by Electronic Product Code (EPC) [11]. In addition, several techniques have been proposed for representing Internet hosts such as database (e.g. $IP2LL$ [79]), DNS (e.g. GeoTrack [80]), clustering (e.g. GeoCluster [80]) and delay measurement (e.g. GeoPing [80]). The generated map can be changed dynamically according to the overlaying application’s requirements. The distance between peers can be measured in Euclidean or Manhattan distance.

To take benefit from grid structure for searching of and routing to unpopular data and partial keywords, the proposed routing algorithms in this thesis are executed in a grid overlay network on top of existing P2P systems as presented in Fig.2.4. The basic network is today’s Internet infrastructure. The middle layer is any peer-to-peer system that could be structured or unstructured. The topmost layer is the grid topology with its coordinate system.

The logical grid with coordinate system can easily be built on top of any decentralized network
using only local knowledge of each node [12]. In this case, this approach is superior to basic structured P2P systems which needed strictly and centralized control hash function to maintain overlay topology. The second overlay layer allows to adapt topology while P2P overlay layer retains its architecture. The multiple overlay system is able to support complex and inexact-match key query when the node identifier (XY value) in grid easily adapts according to the application requirement, while structured P2P layer retains to support other overlay properties.

There are some research works have multiple overlay structures similar to the proposed idea [12, 24, 112]. Berg et al. [12] have introduced a mesh-like structure on top of a P2P system that uses RFID for node identification. Each peer carries a logical 2D node identifier, where X is the respective RFID chip’s current location and Y is the information about manufacturers. The Brocade is proposed as a secondary overlay on top of P2P networks in order to improve point to point routing performance on P2P overlay networks [112]. The brocade layer uses Tapestry to direct messages to the super-node nearest to the destination node’s location. In [24], Coltzau proposes the P2Life infrastructure, a 2-dimensional grid topology, built on top of CAN system as a lookup protocol, in which each physical peer act as both NVME-peer providing object information and lookup-peer for handling lookup queries.

2.5 QoS Routing in Overlay Networks

The main motivation behind the deployment of overlay networks is to overcome many limitations of the current physical architecture. Some of major concerns include the lack of security, QoS guarantees, mobility support, and end-to-end service guarantees. Out of these issues, overlay networks have emerged as a viable solution by providing third-party service providers and users some solutions for the mentioned needs at a flexible scale and without requiring universal
change or coordination for the deployment. As discussed earlier, the overlay routing, especially to satisfy QoS, is slightly different from IP-based routing protocols. Physical infrastructure or IP-based layer protocol can directly measure a link’s capacity (e.g. available bandwidth) and the computational capacity of a node itself. If this knowledge is available, the routing can guarantee the required QoS. In contrast, the overlay routing is limited to the overlay topology, e.g. overlay link capacity and overlay node capacity. An overlay link generally is composed of several physical links, so that the overlay link capacity can only be estimated by end-to-end measurement techniques. For this reason, there cannot be any absolute QoS guarantee in distributed overlay networks.

Some proposals have been made to provide QoS guarantees according to specific application requirements. SON, the service overlay network, aims to provide QoS guarantees in the inter-domain scale through an overlay network [31]. QoS is provided by purchasing bandwidth from ISPs via bilateral SLAs with certain end-to-end QoS guarantees. The use of SON allows the deployment of QoS-sensitive applications in the network without the overhead required for the physical network layer. However, the big challenge is the big cost for purchasing bandwidth in order to provide adequate bandwidth to support QoS requirements. Similar to SON, QRON also aims to provide an application sufficient bandwidth over overlay networks [64]. QRON organizes the overlay nodes using a hierarchical clustering and naming scheme. Two routing mechanism on top of the overlay are used: modified shortest distance path (MSDP) and proportional bandwidth shortest path (PBSP), which aims to select paths with the best available bandwidth in addition to the shortest overlay path. QoSMap [89] is an overlay construction mechanism which computes high quality overlay networks for applications having stringent constraints on hop-degrading QoS metrics and provides resilience against the Internet’s unpredictable network behavior. The overlay link considers latency and packet loss, and provides a backup overlay path.

More specific to routing algorithm in overlay systems, some research works have introduced novel routing algorithms such as [48, 63]. A novel path selection algorithm, named Modified Least-Cost Path (MLCP) was introduced in [48]. They consider an available link bandwidth using active end-to-end probing measurement techniques to find an optimal route and balance the load of the system. The Greedy-EF, a heuristic algorithm to find service paths to route multimedia data flow while meeting the applications resource requirements and specific QoS constraints, is proposed in [63]. The Greedy-EF uses an aggregate function to evaluate the QoS conditions for each service instance, and a hop-by-hop service selection approach to explore the proper service path. The algorithm considers CPU workload, bandwidth, response time, and availability to evaluate the QoS conditions of the service instance. They claim that their proposed algorithm has lower computation complexity than most of the existing service selection algorithms. The upper bound on the time complexity is $O(M^2K)$, where $M$ is the maximum number of instances per service in the overlay network, and $K$ is the number of services to be composed.

To our knowledge, routing on overlay networks by considering buffer utilization together with
other criteria from different layers has not, so far, received special attention. Therefore, this thesis introduces a novel set of routing algorithms considering buffer utilization objecting to improve QoS Routing in overlay networks. Because we believe that the transport of very big files will enforce buffer to overload, which will lead to additional and unnecessary packet loss and unbalanced network load. In addition to the novel routing mechanisms, a self-organizing overlay architecture is introduced that creates Kleinberg-like long-links according to dynamically changing traffic situations.

All proposed algorithms in this thesis had been implemented in a network simulator to analyze their performance. The P2PNetSim network simulation tool has been chosen, which will be introduced next.

2.6 P2PNetSim - Network Simulator

In order to analyze and investigate any new P2P algorithm’s performance in decentralized systems under real network condition, P2PNetSim developed at the department of Communication Networks at FernUniversität in Hagen (Germany), is utilized [23]. P2PNetSim supports large scale network simulations (maximum 2 million peers) and can be run on up to 256 simulation nodes. P2PNetSim is able to:

- simulate a TCP/IP network with an IP address pace, limited bandwidth and latencies giving the developers the possibility to structure the nodes into subnets like in existing IPv4 networks
- built up any underlying hardware structure and to establish a variable time depending background traffic
- setup a peer structure, where the programmer can concentrate on programming the P2P functionality and use a broad library of standard P2P functions like broadcasts

P2PNetSim can be used through the graphical user interface (GUI) of the Simulation Controller as shown in Fig.2.5 and Fig.2.6, which is a GUI used to set up, run and control simulations. Each Simulator is responsible for one class A IP-subnet, its subnets and for all Peers within this subnet. Communication between peers within one subnet is confined to the corresponding a simulator. This hierarchical structure, which is based on real-life IP network architecture, gives P2PNetSim its highly scalable character.

P2PNetSim is Java-based, which makes the implementation of peer behavior simple and platform independent. A peer is designed by inheriting from existing peer prototypes that provide basic communication and logging facilities and an event system which allows it to keep track of the state of simulation and perform analysis processes on this. To run a simulation on P2PNetSim, two components are required:
1. **Peer class** - this java class is used to implement a peer’s actions. P2PNetSim provides a basic class, which the developer can override for specifying the required behavior. In this thesis, every peer is configured to execute the same main actions as shown in Fig.2.7. At the initialization step, the configurations of peer and simulation parameters are set. Then two key functions are executed: receiving and forwarding process. Finally, the system checks the condition to stop the simulation, otherwise, runs peer process again with ongoing simulation time.
2. **Network configuration**- For small networks, the setup can be done manually using the GUI. For big networks, P2PNetSim provides a more convenient method based on XML. An XML configuration file contains simulation parameters and the properties of all peers within the simulation. A part of an example XML configuration is shown in Fig.2.8.
Figure 2.8: Network configuration in XML file
Chapter 3

Multicriteria Routing Algorithm - Criteria Selection

P2P Overlay network are very popular especially for sharing contents. For such very big and complex systems it is important to have efficient management mechanisms, especially for the lookup and routing algorithms, which are the main functions of P2P systems. The complexity of routing is to choose the “best” path from a multi-dimensional perspective, particularly when several paths are available. The difficulty in QoS routing comes from the existence of multiple criteria to maximize the system utilization and performance, as well as to minimize congestion and cost. Hence, our idea to develop novel routing algorithms executed in a P2P overlay network to improve QoS routing.

We are interested especially in grid topologies according to Kleinberg’s small-world networks [50]. The delivery time for a message to be sent between any two nodes using a greedy algorithm and only local knowledge is limited to $O(\log^2 n)$. And inspired by the idea to use temperature representing the peer’s properties. The combination of grid and temperature motivated us to introduce new routing algorithms executing in an overlay networks. The grid topology is the system structure and the temperature represents a peer’s property.

This chapter gives details on how the idea of temperature is applied to properties of peers in an overlay network. The grid structure with its inherent coordinate system is introduced. Then, the first multi-criteria routing algorithms are proposed, in which the routing decision is executed hop-by-hop while randomly choosing a constraint to take into account. First, the constraint-choosing probability is constant, and after that, it is determined by different probability functions.

3.1 Motivation

Temperature is a property of a material, and thus depends on the material, whereas heat is a form of energy existing on its own. Heat is a flow of energy from a higher-temperature object to
another lower-temperature object. It is the temperature difference between the two neighboring objects that causes this heat transfer. There are three different ways that heat can be transferred from one substance/place to another; Conduction, Convection, and Radiation [42]. Conduction is the transfer of heat by intermolecular collisions which is the most common way of transferring heat between two solids or liquids, or within a single solid or liquid. While conduction involves molecules passing their kinetic energy to other molecules, convection involves the molecules themselves moving from one place to another as it e.g. happens in gases. Radiation takes place when the source of heat is some form of electromagnetic wave, such as a microwave or sunlight. Finally, the temperature gradient is a quantity that describes direction and rate of temperature changes around a particular location.

Inspired by thermal physics, a temperature value is applied to every peer in order to represent its physical network status or activities, such as bandwidth, delay or buffer usage, query frequency of content or content key words. The assumption is that peers or members cooperate among others and contribute to the community or network performance like a body composed of particles. In physics, the temperature of the object is the average speed of the particles. Using this, the temperature is the intensity of activities or changes of peers in an overlay network. The temperature of one peer spreads to its neighbors and therefore propagates its property over the network. The temperature gradient allows messages to be routed either directly to or away from a hot area. However, there are some differences between temperature in physics and in P2P overlay, as shown in Table 3.1.

Table 3.1: The comparison of Thermal approach in reality and overlay network

<table>
<thead>
<tr>
<th></th>
<th>Thermal physics</th>
<th>Overlay Network</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Object</strong></td>
<td>Solid, liquid, gases</td>
<td>Network/system</td>
</tr>
<tr>
<td><strong>Component</strong></td>
<td>Particle</td>
<td>Node/peer</td>
</tr>
<tr>
<td><strong>System</strong></td>
<td>Close and Open</td>
<td>Open only</td>
</tr>
<tr>
<td><strong>Heat Conduction</strong></td>
<td>By material</td>
<td>By messages</td>
</tr>
<tr>
<td><strong>Heat Radiation and convection</strong></td>
<td>Have</td>
<td>Don’t have</td>
</tr>
<tr>
<td><strong>Meter</strong></td>
<td>Thermometer</td>
<td>Mathematic Function</td>
</tr>
</tbody>
</table>

An object can be solid, liquid or gases composed of many particles. In the same way, an overlay systems is composed of many nodes or peers. Messages or packets are forwarded through networks, which is similar to heat conduction in an object. A high temperature of any part of an object affects a spatial spread of heat in the object around that part. The temperature of the affected objects is sinking with the distance to the heat source. When several hot areas overlap, the temperatures rises. When there is no more heat source, the temperature of the system sinks. The heat-sensor (thermometer) that is used to measure/define the temperature on a node is a mathematic function. However, due to the special properties in a network, a peer is able to memorize a neighbor’s temperature from last access. In such manner, any messages
arriving at this node can contain the latest temperature of the sender, so that the current node can store this information and use it for following route decisions.

The temperature field analogon has already been used in earlier approaches. In 2004, Unger and Wulff [102] use a temperature field to locate nodes managing contents of common interest in P2P networks. Each node has a temperature, which is an index for the activity of that node. The heat of each node radiated toward its direct neighbors and therefore influenced their temperature as well. Whenever the content of a node was accessed or updated, its temperature was increased, whereas during periods of inactivity, the temperature was decreased exponentially to align with the temperatures of the surrounding neighbors. In 2007, Baumann et al. [9] introduced the HEAT routing algorithm for large multi-hop wireless mesh networks to increase routing performance. HEAT used anycasts instead of unicasts to make better use of the underlying wireless network, which used anycasts by design. HEAT relied on a temperature field to route data packets toward the Internet gateways. Every node was assigned a temperature value, and packets were routed along increasing temperature values until they reached any of the Internet gateways, which were modeled as heat sources. It was a distributed protocol to establish such temperature fields which did not require flooding of control messages. Rather, every node in the network determined its temperature considering only the temperature of its direct neighbors, which rendered the protocol particularly scalable to the network size.

In this thesis, the buffer utilization level is presented by the temperature value as an example. The hotter a node is, the less buffer space for storing new messages is available on it. The overloaded buffer problem leads to an increased packet loss-ratio and to longer delay times. The route decision therefore should avoid following temperature gradient directly to hotter areas. However, the main objective of any routing approach is the fastest delivery of messages from source to target. Thus, the distance is also crucial to find a short(est) way, although it maybe not the fastest one. Our idea is that the mesh structured overlay network with coordinate system gives a lot of benefits to routing algorithms due to its fault-tolerant properties and inherent coordinate system. So, the multi-criteria routing algorithms consider either the shortest way (given by coordinate system) or lowest buffer usage (following the low temperature gradient) to find the optimal path efficiently. Packet success-delivery ratio and load balancing are expected to be remarkably better, and the overall delay time is expected to decrease. The number of hops from source to target may increase because of a slightly longer route, but still, the delay time will decrease, because waiting in long queue is avoided. However, the proposed algorithms are not restricted to grid-like structures, the proposed multi-criteria routing algorithms in this thesis could be applied to all other structures where the IDs source and target can be used to determine the distance of them.
3.2 Thermal Field Considering Buffer Utilization

In the presentation of the algorithms, the temperature ($\theta$) indicates the usage level of a peer’s incoming- and outgoing message buffer. The temperature of a node $c$ is referred to as $\theta_c$. The possible values of $\theta_c$ are in the range from 0 to 1, where 0 denotes an empty buffer and a value of 1 indicates that the buffer is full. The temperature is calculated as shown in Eq.3.1.

$$\theta_c = \frac{\text{Message in buffer}}{\text{Buffer size}}, \quad 0 \leq \theta_c \leq 1 \quad (3.1)$$

The peer structure is presented in Fig.3.1. Each node has a list of direct neighbors and each neighbor’s temperature value, as well as the time, at this temperature has been taken. This time is important, because it is used to estimate the current temperature at decision time. Our assumption is that the neighbor’s temperature decreases exponentially similar to Web properties. So if the neighbor’s temperature feeds to decision node, an update temperature is used. In contrast, an estimate temperature is measured exponential decrement when there is no temperature fed from that neighbor.

To reduce complexity, each node only uses one message buffer, which is organized in a First-in-first-out (FIFO) manner in our works. Hence, the temperature of that buffer is equal to the temperature of the node. Since the routing decision strongly depends on $\theta_c$ being up to date, the temperature is recalculated with every message that enters or leaves a buffer. Additionally, the messages themselves act as temperature-carriers, conveying a node’s temperature from one peer to another until they either reach their target or expire. This underlines the analogy to convectional processes in thermal physics, where temperature is conveyed by rapidly moving particles.

Each node keeps account of the temperature of its neighbors. Let $N(c)$ be the set of neighbors of $c$, let $k$ be the number of neighbors with $1 \leq k \leq 4$, since the degree of distribution in the mesh structure is 4. Let $i$ be the index of each neighbor $N_i$ in $N(c)$ where $1 \leq i \leq k$. In addition, let $i$ be the number of messages sent from $N_i$ to $c$. Now, there are two cases to update a neighbor’s temperature, $\theta(N_i)$ in $c$’s dataset:
1. Whenever $c$ receives a message from neighbor $N_i$, containing that node’s temperature $\theta_i$, the previously stored value, is overwritten:

$$\theta(N_i) = \theta_i, \quad if \phi_i > 0$$

(3.2)

2. If no message is sent from $N_i$ to $c$, $\theta(N_i)$ is decreased exponentially over time with a configurable time constant of $\alpha$:

$$\theta(N_i) = \theta(N_i) \cdot \exp^{-\alpha t}, \quad if \phi_i = 0$$

(3.3)

Object to improve routing performance in grid overlay networks, thermal field approach is applied for multi-criteria routing algorithms. The hop-by-hop routing decision mechanism is used to decide to forward messages to either the neighbor with the shortest path to target or the one with the lowest temperature. First, a constant probability of choosing the lowest buffer route is discussed in Section 3.3. In Section 3.4, an adaptive probability routing relying on the remaining distance to target is introduced.

### 3.3 Route Decision by Constant Probability

The idea of the path decision by thermal field is to find a path with low buffer usage. But if the path selection chooses only low buffers, it will take very long routes to target in case that a lot of overloaded areas are located on that way. Hence, the balance between deterministic and adaptive routing is critical. Due to the optimal point to choose constraints have not been identified, then a random method is used.

The path selection is adapted upon a predefined probability to follow the temperature field, $P_\theta$. The route decision selects a low-buffer route with probability $P_\theta$, and a direct route with $1 - P_\theta$. If $P_\theta$ is high, the low buffer routes are used more often. This might lead to increased hop-counts but less buffer waiting time. On the other hand, a small value for $P_\theta$ indicates, that the direct route is preferred, which might lead to more message losses due to overloaded nodes along the shortest way. So, the optimal balance between route and network load balanced depends on a suitable probability.

Each peer has only information about its neighbors, so that the route has to be selected step by step, from each single node to its next neighbor until a message reached its target location. The information in each message required to do the routing is source, target and previous node (sender). The sender information is required in order to avoid to send messages backwards. The flow chart of how the routing decision mechanism by using a predefined probability is shown in Fig.3.2.

Every neighbor is taken into account when deciding for the next hop. First, the distance between each neighbor and the packet’s target is calculated as well as the according neighbor’s
In the end, one out of two neighbors can be chosen; either the one which is closest to target or the one with the lowest buffer usage. Which of them is chosen, is decided by $P_\theta$ as described above.

### 3.3.1 Simulation Setup

The experiments are setup to analyze the routing performance when thermal approach is applied. The scenario is defined on the hypothesis that the delivery success ratio increases and the delay time decreases when messages are routed through unoverloaded path. Since most of the Internet activities and properties follow power-law distributions, the simulation setups in this thesis also do. The simulation focuses on the network situation that a majority of members accesses popular data located in a specific area at the same time. As well as the majority of messages target are located in one area.

The performance of the proposed routing algorithms is examined in the P2PNetSim network simulator as discussed in Section 2.6. All simulations to analyze routing algorithms in Chapter 3 and 4 are set with the same network configuration and traffic pattern as follows.

**Network** - the network is organized into a two-dimensional grid structure composed of 10,000 nodes.
nodes (100x100) that we though it is big enough to represent an overlay network. In Fig.3.3, a square represents a peer in coordinate $x$ and $y$ from 0 to 99. The grid overlay system is simulated over an IPv4 network. A node is connected to its neighbors in all four directions, with the exception that nodes on the borders only have two or three neighbors. The coordinate of a node serves as its ID. The outgoing-bandwidth is limited for all the peers randomly in some area of the grid to represent the overload situation. The small-outgoing-bandwidth nodes are colored in Fig.3.3.

![Two-dimensional grid network of 10,000 nodes (100x100)](image)

Figure 3.3: Two-dimensional grid network of 10,000 nodes (100x100)

**Traffic pattern** - messages are generated randomly by all network nodes from simulation time 0 to simulation time 300. Each message has a randomly defined source and target, the number of generated and received messages follows a power-law distribution. The source nodes are located all over the network, but target area is limited behind the overloaded region mentioned above. The system handles data packets in First-In-First-Out (FIFO) manner both according to queuing and forwarding. Fig.3.4 and Fig.3.5 show the receiving and forwarding process in the thermal field approach. The difference among the proposed algorithms is the decision logic in “Find next node” functions.

The simulations generate 84,000 packets in total that have to be routed through the network. Each message is forwarded through the network until one of the following three cases happens: message reaches its target, the TTL reaches zero, or a message is dropped due to an overloaded buffer. The traffic pattern has been saved and reused over several simulations to ensure comparability of simulation results over multiple simulations.

**Simulation metric** - the metrics are as follows.

- *Packet delivery ratios* are
  
  1. Success Ratio: The number of messages that have reached their target successfully in relation to the total number of messages that have been generated in the simulation.
2. Loss Ratio: The number of messages that are lost or dropped due to buffer overflows in relation to the total number of messages that have been generated in the simulation.

3. Expired Ratio: The number of messages that dropped due to TTL expiry before reaching the target in relation to the total number of messages that have been generated in the simulation.

- *Average hop-count*: The average number of overlay hops that a message has to pass until
reaching its destination.

- **Average waiting time**: The average time a message has to wait in queues. The delay time of a message is the sum of waiting times of every queue the message has passed. The delay can be caused by low throughput and overloaded areas.

- **Average routing time**: The average of the time necessary to transport a message from source to target. Since processing delay and propagation delay are not concerned in this thesis, this is the sum of hop-count (each is 1 timestep) and queuing delay time as described above.

- **Number of overloaded nodes** which temperature is greater than 70% ($\theta_c > 0.7$). The assumption is that with this level, a node is starting to become overloaded if more messages arrive.

### 3.3.2 Simulation Results and Discussion

The routing performance of routing considering thermal field by constant probability from $(P_00.1)$ to $(P_00.9)$ are discussed and compared to the shortest path algorithm. Their results are categorized into two groups of metrics; packet delivery ratio and routing time.

![Figure 3.6: Results of packet delivery ratio by constant probabilities](image)

Fig. 3.6 shows packet delivery ratio results according to the three packet-delivery metrics described above. The first column from the left is the results of the shortest path algorithm. In this traffic pattern, 74% of the packets can reach their target, while the loss ratio due to overloaded buffer is 26%. It is clear that using shortest path, no TTL expiries occur. The messages are forwarded to the target directly which is confirmed by Fig.3.7, where the graph
of shortest path algorithm shows the lowest hop-count at 94 steps. The average waiting time is 18 steps so the average routing time is 112 steps.

The results of the adaptive routing are better than shortest path algorithm in a lot of cases. The success ratios are rising with higher values for $P_\theta$ until a maximum is reached for values of $P_\theta = 0.4$ and 0.5. These results support our assumption that if messages are forwarded avoiding overloaded nodes, the packet loss due to overloaded buffers is decreased. However, with higher values for $P_\theta$, the number messages losses due to expired TTL is going up, too, because longer paths are chosen when a lot of nodes are overloaded. For $P_\theta = 0.9$, the success ratio is decreased to only is 4%, while loss due to overloaded is 29% and loss due to expired TTL is 67%. Hence, a range of optimal values for considering thermal field $P_\theta$ in the current traffic pattern is between 0.4 and 0.5. The success ratio is 88% which is 14% higher than using the shortest path algorithm.

When the value for $P_\theta$ is too high, the avoiding of overloaded regions increases the number of nodes that have to handle a single message, remarkably. That leads to the situation, that the avoiding itself creates new overloaded nodes. Hence, $P_\theta$ should not exceed a value of 0.7.

![Figure 3.7: Results of routing time by constant probabilities](image)

Next, the resulting routing times are presented in Fig.3.7. As discussed earlier, the shortest path algorithm always have the lowest hop-count, since messages are forwarded only to the neighbor who is closest to the target. The improvements of packet success ratio described above are on the cost of a higher routing time. The average hop-count and the average routing time increase corresponding to higher values for $P_\theta$. The average waiting time in queues almost has the same absolute value as shortest path although more nodes are visited, so that the waiting time per node on the way is reduced. That supports our idea to route messages avoiding an overloaded area in order to decrease loss ratio and delay.
The results of routing algorithm shows that the adaptive routing algorithm considering overloaded area are surely able to improve QoS routing performance. The route decision with constant probability decreases loss ratio and waiting time. The balance of considering shortest path and overloaded area is crucial. Too high values for $P_\theta$ lead to very long routes and increase the loss ratio due to expired TTL. On the other hand, the loss ratio due to overloaded buffers is high when the probability to choose a low-temperature path is too low. In the discussed traffic pattern, the optimal probability to consider both criteria roughly equally. These results have lead us to a new idea. A constant probability might not be suitable for every decision location. Especially, when messages are very close to the target, an adaptive path might cause an unnecessary longer route. Therefore, the probability considering thermal field relying on the remaining distance are next presented.

### 3.4 Route Decision by Probability Functions

In the previous section, the adaptive routing approach was applied for a constant probability routing algorithm. The general idea in this section is to move a message more directly on the shortest path, when it is closer to its destination. Hence, the probability to follow the thermal field is now calculated in a function depending on the distance to target. The path selection step is related to a probability for using temperature data, $P_\theta$. Each node randomly selects a low-buffer route according to $P_\theta$, otherwise selects a direct route.

#### 3.4.1 Probability Function Considering Remaining Distance

The remaining distance is the proportion of the distance from current location to target and the distance from source to target, and is called $\Omega$. When a source node $s$ sends a packet to a destination node $t$, the distance between original peer and a target peer is $d_{st}$. At a current node $c$ that is going to decide for a path to forward message to, the distance between current node and target node is $d_{ct}$. The remaining distance is calculated as in Eq.3.4.

$$\Omega = \frac{\text{Distance from current to target}}{\text{Distance from source to target}} = \frac{d_{ct}}{d_{st}}$$  \hspace{1cm} (3.4)

The various adaptive probability functions ($AP_\theta$) discussed now, are both linear and exponential functions of the relative remaining distance. They all follow the main idea that messages closer to the target should be routed more directly. The most crucial question is now, which function is able to find an optimal probability at any decision location. The discussion starts with two exponential functions and goes on with a combination of linear and exponential functions.

Adaptive Probability 1 ($AP_\theta^1$): $AP_\theta^1$ is an exponential cumulative distribution function (cdf). The probability for using thermal field is decrease exponentially when messages come closer to their target. At the starting point ($\Omega = 1$), the probability is $e^{-1}$. If the decision
location is very close to target, $AP_\theta$ is nearly zero. Hence, messages are forwarded almost directly to the target. If a message is located farther away from target, the probability of using the thermal field is higher.

$$\text{Probability of } AP_\theta^1(\Omega, \lambda) = e^{(-\frac{\lambda}{\pi})}$$  \hspace{1cm} (3.5)

**Adaptive Probability 2 ($AP_\theta^2$):** $AP_\theta^2$ is an exponential cumulative distribution function (cdf). It is similar to ($AP_\theta^1$), but in general provides a higher chance to use the thermal field at the same distance, especially close to target, where $\Omega$ is almost zero.

$$\text{Probability of } AP_\theta^2(\Omega, \lambda) = 1 - e^{(-\lambda(\Omega))}$$  \hspace{1cm} (3.6)

**Adaptive Probability 3 ($AP_\theta^3$):** The idea of $AP_\theta^3$ is to combine functions for selecting direct and indirect paths. The routing path is split into two functions. At the starting point ($\Omega = 1$), the probability considering thermal field is one, i.e. the message is first routed to a “cold” area. When messages come closer to the target, the probability decreases linearly. On the other hand, when decision node is farther away from the target than it was at the starting point (i.e. $\Omega > 1$), the probability of using temperature also decreases, but exponentially. This exponential function is the inverse of $AP_\theta^1$. So, when messages are routed too far away from the target, they should be brought back to the direct route.

$$\text{Probability of } AP_\theta^3 = \begin{cases}  
\Omega & d_{c2t} \leq d_{s2t} \\
1 - e^{(-\frac{\Omega}{\pi})} & d_{c2t} > d_{s2t}
\end{cases}$$  \hspace{1cm} (3.7)

The graph of probability from three adaptive probability functions are displayed in Fig.3.8. At the starting point, $AP_\theta^3$ has the highest probability, followed by $AP_\theta^2$ and $AP_\theta^3$ respectively.

![Figure 3.8: The graph of adaptive probability functions when $\lambda = 1$](image)
The flow chart of the routing decision making by adaptive probabilities using thermal field is shown in Fig. 3.9. There are three options identified in simulation constant to identify decision function. The main routing process is the same as already discussed before and shown in Fig. 2.7.

![Flow chart of route decision by probability function using thermal field](image)

Figure 3.9: Flow of route decision by probability function using thermal field

Again, every neighbor is a potential next hop. The distance between each neighbor and the packet’s target as well as the neighbor’s temperature is taken into account. As before, a decision between two neighbors has to be made: either the one with the lowest temperature or the one with the shortest path to target. Now, the globally known probability function is used to find out with which probability the low-temperature path shall be followed.

### 3.4.2 Simulation Results and Discussion

The simulation environment is the same as in the previous simulation as discussed in Section 3.3.1. The routing performance is again analyzed against the shortest path algorithm in two
groups of metrics: packet delivery ratio and routing time. The comparison results are presented in Fig.3.10.

As expected, in Fig.3.10a, the exponential functions show superior routing performance compared to the shortest path algorithm in terms of delivery success. The higher success ratio corresponds to higher probability considering thermal field. Still, the combined function shows worse results than shortest path. The results of $AP^3$ show that the loss ratio due to expired TTL is 29% and loss due to overloaded nodes is 4%. Such result implies that a lot of messages have followed the temperature criteria more than shortest distance criteria. It may be because at the starting point or the distance equivalent to starting point has very high probability to select adaptive way. So, messages are forwarded to longer routes. Although the probability using thermal field decreases exponentially, when the decision node is far away from the target, so that messages are leaded to the shortest way, but when coming closer again to the original distance, the probability of following the thermal field again is very high. So, the idea of this combined function does not work efficiently in the analyzed traffic pattern.

![Figure 3.10: Simulation results by probability functions](image)

(a) Packet delivery ratio
(b) Routing time

The comparison results of routing times are shown in Fig.3.10b. As expected, the routing time is increased according to hop-count, when considering the adaptive way. But the average waiting times from all probability functions are similar to that of the shortest path algorithms, even a bit less when using $AP^1$. From these results, it is confirmed that multi-criteria routing using thermal field can improve routing performance at the balance of considering shortest path and low buffer path.

In addition to packet delivery ratio and routing time, the load balancing of the network has been analyzed. The graph of number of overloaded nodes in the system by simulation time is shown in Fig.3.11.

The red line is the result of the shortest path algorithm, whereas blue, green and orange dotted lines represent the results of the adaptive probability functions respectively. The number of overloaded nodes from shortest path algorithm rise early and fast up to the highest peak.
When system stops generating new messages at simulation time 300, the graph drops notably until no more overloaded nodes exist at about simulation time 400. The trends of exponential probability function graphs are similar to graph of shortest path. When probability using thermal field is higher making graphs are slower rise and less sharpen dropped. The graph of $AP_3^3$ rises only slowly and also slows down slower compared to the other approaches. An interesting detail is that the graph of $AP_3^3$ shows many peaks. This may be, because messages are first routed away from the target, but are routed back again and again, when they get too far away. So, the messages are going back and forth and therefore produce singe nodes to be overloaded again and again.

Finally, Fig.3.12 and Fig.3.13 show the buffer utilization status of the community (100x100) of routing both by constant probability (see previous section) and probability functions. The colored dots represent the buffer level of each node. Grey represents low temperature (buffer level), while Yellow, Orange, and Pink denote higher temperature. Red indicates a full buffer.

In Fig.3.12, three constant probability values have selected; $P_\theta$ is 0.2, 0.5 and 0.8. The application captured buffer usage status at the simulation time-steps 100, 200, 300, 400, and 500. Fig.3.12a shows the temperature of the network, when the probability using thermal field is 0.2. At time 100, the picture shows some yellow dots and only few red dots, whereas many grey nodes are arranged in a star-like manner. The situation is more clear at time 200 and 300, when a lot of fully loaded nodes appear in the area of limited bandwidth. Again, many grey dots are appear similar to the starting situation, because messages route more directly to the target. Finally, at time 500, there are no more messages in the system. In Fig.3.12c, the buffer situation is shown when using a probability of 0.8 for using the thermal route. The number of red dots is small, while a lot more orange and pink dots (i.e. loaded, but not overloaded nodes) can be seen in comparison to Fig.3.12a. Also, the grey dots (mildly loaded nodes) are
distributed much more in the network. This strongly indicates the load-balancing that results from avoiding overloaded nodes. However, at simulation time 500, there are still a lot of messages in the system, since they take longer paths on the low buffer way. The balance between adaptive and deterministic way is $P_\theta^0.5$ as seen in Fig.3.12b. Grey and yellow dots are distributed in the network, however they are located in the central area with limited bandwidth. The three probabilities obviously produce very different network load, best seen at time 200 and 300. The number of overloaded nodes is inversely proportional to the constant probability of using the thermal field.

Fig.3.13 presents simulation network diagrams of the three probability functions. For each function, the sequence of pictures is to be read from left to right. The application made a snapshot of the buffer usage status at the simulation time-steps 100, 200 and 300. The colors represent buffer levels of the nodes in the same way as described above. In Fig.3.13a, at time 100, messages are concentrated in the center of network and produce a few red dots (overloaded nodes), whereas Fig.3.13b displays no overloaded nodes when considering the thermal field approach more. At time 200 of both Fig.3.13a and Fig.3.13b, the number of overloaded nodes is strongly increased at the limited throughput area, which is even more the case at time 300. Fig.3.13b also shows some overloaded nodes, but less and instead has more mildly loaded nodes (gray) than Fig.3.13a. Again, the routing towards low temperature areas improves load balancing. The area, in which mildly loaded nodes occur, is much bigger and much more distributed over the network in Fig.3.13b than in Fig.3.13a. In contrast, the diagrams of $AP_\theta^3$ in Fig.3.13c show much less overloaded peers. There are many more grey and yellow dots distributed over a wider area in the network at time 100. At time 200 and 300, messages are forwarded avoiding overloaded nodes, so that highly loaded nodes are spread around the network and away from the limited throughput area. The load balancing is better at the beginning but it leads to a lot of message losses due to the limited time-to-live. The messages are spread around the network although some overloaded peers can be observed.

3.5 Summary

In this chapter, the temperature of any object, an average speed of the particles in thermal physics, was implied to the intensity of the activities or status of peers in an overlay network. The thermal field approach was introduced in order to represent physical network status and activities in peer, for instance, bandwidth, delay or buffer usage, query frequency of content or content key words. The buffer utilization was chosen as an example in this thesis to be represented by the temperature value. The thermal filed considering buffer utilization has been used for multi-criteria routing in a grid overlay network to improve routing performance.

The multi-criteria routing algorithms by selecting a single criteria at each routing step have been introduced. First, a constant probability for route decision has been discussed. The route has been randomly selected hop-by-hop and either forwards into the direction of the shortest
path or to the lowest temperature. Every route decision in any location of the system is based on the same chance to select either lowest-load or shortest path. However, a suitable probability must be found in order to balance out adaptive and deterministic rules. So, the optimal path is found, when shortest way and adaptive approach are taken into account with about the same probability. Next, the random routing relying on one constant probability value was adapted to use probability functions, instead. The idea is when packets move closer to the target, the probability to route them directly is higher. The different degree of adaptively depends on the remaining relative distance to target.

All proposed algorithms have been examined in P2PNetSim, and where compared to a greedy shortest-path approach. The results of multi-criteria routing by selecting criteria show that the loss ratio decreases, when messages avoid being forwarded through overloaded peers but the delay time is not clearly improved. The load balancing results of multi-criteria routing methods using thermal field are better.

The leading question for the next chapter is, in how far both criteria can be considered at the same time? Which method is suitable for routing decisions with concurrent multiple criteria in P2P overlay networks?
Figure 3.12: Network diagrams present buffer status from a global probability
Figure 3.13: Network diagrams present buffer status from Probability Function algorithms
Chapter 4

Multicriteria Routing Algorithm - Criteria Combination

In the previous chapter, routing mechanisms have been discussed that randomly select a criterion to take into account at each routing step. In this chapter, the decision mechanism concurrently considers both criteria, so that a multi-criteria decision function is needed to find an appropriate trade-off between distance and load. The problem of determining a QoS route that satisfies two or more path constraints is known to be an NP-hard problem. Hence, multi-constrained QoS path algorithms have focused on heuristics and approximation algorithms. In this thesis, a weight function and a fuzzy logic setup are analyzed.

4.1 Route Decision by Weight Function

The shortest path problem is a fundamental and classical problem, for which a variety of solutions have been proposed over time [33, 103]. The shortest path problem with positive and negative weights is the problem of finding the shortest distance from a specified source node to all the nodes in the graph. Bellman and Ford introduced an algorithm which can find the shortest paths on a weighted directed graph and supports both positive and negative edge weights [10]. The Bellman-Ford algorithm is a label-correcting algorithm that computes single-source shortest paths in a weighted digraph, where some of the edge weights might be negative, within \( O(V \cdot E) \) time, where \( V \) and \( E \) are the number of vertices and edges respectively [4]. The algorithm in its basic structure is similar to Dijkstra’s approach, but, instead of greedily selecting the minimum-weight node which was not yet processed and is in relaxing state, it simply relaxes all the edges \( V - 1 \) times. The repetitions allows minimum distances to be accurately propagated throughout the graph, since, without the negative cycles, the shortest path touches each node only once. Unlike the greedy approach, which depends on certain structural assumptions derived from positive weights, this straightforward approach can be applied in more general cases [4].
Though the Bellman-Ford algorithm expects complete knowledge about the graph, it was still the inspiration for the decentralized algorithm that is presented in this chapter. Here a function of two constraints, temperature and distance, is used to calculate the weight of all possible next hops. The weight is assigned to every neighbor based on that neighbor’s distance to target and temperature. The message is then routed to the neighbor with the minimum weight. Each weight is calculated as a linear combination, where the coefficients control the influence of each summand on the total weight. The priority of each criterion is defined by two coefficients; factor $\alpha$ (Alpha) for distance and $\beta$ (Beta) for temperature.

### 4.1.1 Weight Function with Thermal Field Approach

The path selection process is related to the weights or influential factors. Again, let all $N_i$ with $1 \leq i \leq 4$ be the neighbors of the current node $c$ and let $t$ be the target node of the message that is to be routed. Additionally, let $d(N_i, t)$ be the euclidean distance from neighbor $N_i$ to the target node and let $d(s, t)$ be the euclidean distance from the source to the target node. The distance in the weight function is now normalized by Eq.4.1

$$d(N_i) = \frac{d(N_i, t)}{d(s, t)} \quad (4.1)$$

Finally, let $\theta(N_i)$ be the temperature of neighbor $N_i$. The weight of the edge to the neighbor $N_i$ is now calculated as Eq.4.2.

$$f(N_i, t) = \alpha \cdot d(N_i) + \beta \cdot \theta(N_i) \quad \text{where} \quad \alpha + \beta = 1 \quad (4.2)$$

After calculating the weight of each edge going out from the current node $c$, the next hop can be selected to be the node with the lowest weight. Since $\beta = 1 - \alpha$, the coefficient $\alpha$ alone already defines the influence of the remaining distance to target as well as the load of the next hop on the total weight and therefore on the routing decision. Higher values for $\alpha$ let the node select a more direct path while taking the risk to lose the message due to buffer overflows. On the other hand, lower values for $\alpha$ result in selecting a low buffer route, which on the other hand may lead to long routing times. Thus, finding the optimal path with both respecting distance and network load is the result of finding an optimal value for $\alpha$.

The flow chart of the routing decision making by using a weight function is shown in Fig.4.1. First, the distance between the neighbor and the packet’s target as well as the neighbor’s temperature are calculated. Next, each neighbor’s weight is calculated, where $\alpha$ and $\beta$ are defined as global constants. Lastly, the neighbor with the lowest weight can be selected.
4.1.2 Simulation Results and Discussion

To analyze the performance of the routing algorithm using weight function considering distance and temperature field, this section presents the results of simulations with varying influence of each criterion. The routing performance as a function of the coefficients ($\alpha$ and $\beta$) is analyzed. The results are again compared to the shortest path algorithm. The network and traffic patterns are the same as already discussed in Section 3.3.1.

The results of packet delivery ratio can be seen in Fig.4.2 and the results according to routing time are shown in Fig.4.3. The shortest path algorithm reaches a delivery success ratio of 75% with an average routing time of 112 time-steps. The hop-count and waiting time according to overloaded buffers is 94 and 18 time-steps, respectively. The experimental results of weight functions using thermal field approach are discussed for nine cases with varying coefficients. The columns from left to right show the results of ($\alpha0.9;\beta0.1$), ($\alpha0.8;\beta0.2$), to ($\alpha0.1;\beta0.9$).

The best results in terms of packet delivery ratio are reached for ($\alpha0.9;\beta0.1$), the loss ratio due to overloaded buffer is only 3% and the success ratio is very high with 97%. The loss ratio decreases 23% in comparison to the shortest path in the same traffic environment. In addition, the total routing time is 109 steps which is less than shortest path. The hop-count is only 4 steps higher, but the waiting time due to long queue was decreased 7 steps compared to shortest path. Hence, in average, the total routing time for this configuration is 3 steps lower than for
shortest path approach. This result obviously supports our idea that considering the buffer utilization helps to reduce loss ratio and waiting time, especially, when both criteria (distance and buffer usage status) are concurrently taken into account.

The multi-criteria routing algorithm using weight function gives very interesting results in this traffic pattern, however, a higher factor considering thermal field approach shows higher loss ratio and longer routing time accordingly. Although loss ratio from weight function increases corresponding to factor of temperature, it is less than the results of shortest path. The expired ratio also rises as temperature factor. Such situation is contrast to multi-criteria routing algo-
algorithm by selecting criteria as presented in previous chapter. The probability to select criteria allows higher chance to forwarded very long path then average waiting time decrease. Contrast to results of routing time of weight function, more temperature factor shows the waiting time in long buffer queue increase. The reason is the combination of criteria allows the packet forwarded into higher temperature and less distance. Thus the loss ratio, expired ratio and waiting time increase corresponding to temperature factor in this traffic pattern. The discussion on this results will be more clear from the results of the number of overloaded nodes in Fig.4.4 and the network diagrams of three selected weights in Fig.4.5.

![Image](image.png)

**Figure 4.4: Number of overloaded nodes by simulation time using weighted function**

In Fig.4.4, the results of multi-criteria routing by weight function display that the number of overloaded nodes start increasing later in comparison to the shortest path approach. This is especially true for high values of $\alpha$, i.e. $(\alpha0.9;\beta0.1)$ and $(\alpha0.8;\beta0.2)$, indicated by the purple and pink lines, which are both ascending slowly and descend earlier than the black line for shortest path. The peaks of these two graphs at simulation time 300 are also lower compared to shortest path. When looking at the network diagrams in Fig.4.5a, it can also be seen that the weighted-function approach strongly distributes the load in the system. There are only few overloaded nodes (red dots) at simulation time 200 and 300, and at simulation time 400, no overloaded nodes remain in the system. This is the best load balancing so far, compared to the simulations of the previous chapter. The multi-criteria algorithm considering both distance and buffer status by weight function therefore shows best overall routing performance until now, when the coefficients $(\alpha0.9;\beta0.1)$ are used.

As mentioned earlier, one interesting point of the results corresponds to the temperature coefficient $\beta$. When $\beta$ is increased, the loss ratio, expired ratio, waiting time also increase accordingly. In this case, the graphs of number of overloaded nodes also rise fast and high as shown by the
blue, red and green lines in the network diagram in Fig.4.4. This influence can also be seen in the network diagrams for \((\alpha_{0.5};\beta_{0.5})\) and \((\alpha_{0.2};\beta_{0.8})\) in Fig.4.5b and Fig.4.5c. When \(\beta\) is high, messages are forwarded to nodes with lower load, so their overall path length is longer. The more messages are routed on such longer paths, the more nodes are affected by them and therefore have a higher load over time.

![Network diagrams](image.png)

(a) Weight \((\alpha_{0.9};\beta_{0.1})\)

(b) Weight \((\alpha_{0.5};\beta_{0.5})\)

(c) Weight \((\alpha_{0.2};\beta_{0.8})\)

Figure 4.5: Network diagrams present buffer status from weight function algorithm

Although the results of this approach are very promising, still a single coefficient to configure
the behaviour might not serve efficiently in dynamic situations. So, to make the route decision method more capable of adapting to dynamic and unpredictable situations, a fuzzy logic approach is discussed in the next section.

4.2 Route Decision by Fuzzy Logic

The weight function routing algorithm is able to improve routing performance remarkably, but the difficulty is to find an optimal proportion between distance and load along the path, where the traffic situation may change rapidly. Thus, we assume that fuzzy logic is an appropriate technique able to handle this problem. Fuzzy logic is efficient to deal with uncertainties situation. So, in this section, a fuzzy logic decision method is discussed.

4.2.1 Fuzzy Logic

Fuzzy Logic was introduced by Zadeh [108] and allows a computer to reason the same way as people do, i.e. not always precise. People think and reason using linguistic terms such as “hot” and “fast”, rather than in precise numerical terms “90 degrees” and “200 km/hours”, respectively. The fuzzy set theory models the interpretation of imprecise and incomplete sensory information as perceived by the human brain. Thus, it represents and numerically manipulates such linguistic information in a natural way via membership functions and fuzzy rules. Some advantages of fuzzy logic are that they are conceptually easy to understand, flexible, and tolerant in regard to imprecise data. It can model nonlinear functions of high complexity, and also can be built on top of expert’s experience.

A key feature of the fuzzy logic technique is to handle uncertainties and non-linearity. In regard to traffics and applications emerged from physical networks and P2P systems, fuzzy logic is a very attractive concept for route decision making [47, 110]. A fuzzy logic system comprises basically three elements: Fuzzification, Knowledge base (rule and function), and Defuzzification. Fig.4.6 shows the generalized block diagram of a fuzzy system.

![Figure 4.6: Fuzzy System](image)
The function of the fuzzification is to determine the degree of membership of a crisp input in a fuzzy set. The fuzzy rule base is used to present the fuzzy relationship between input-and output fuzzy variables. The output of the fuzzy rule base is determined based on the degree of membership specified by the fuzzifier. The defuzzification is used to convert outputs of the fuzzy rule base into crisp values.

### 4.2.2 Fuzzy Logic with Thermal Field Approach

In our approach, temperature and distance are input data used to find a suitable next hop neighbor. In the approach described now, three parameters from neighbors are considered: the neighbor’s temperature, its distance to target, and the neighbor type. These three selection parameters make the route reflect the network status and the nodes’ ability to reliably deliver the network packet. The distance is defined current packet-holder position compares to source and target. The neighbor type is calculated as the proportion of the distance from current position to target, to the distance of neighbor position to target. Let \((x_c, y_c)\) be the current peer, \((x_s, y_s)\) be the source node, \((x_t, y_t)\) be the destination, and \((x_{Ni}, y_{Ni})\) be the neighbor \(i\) of the current node. Three parameters are used; distance from source to target \((d_{S2T})\), distance from current position to target \((d_{C2T})\), and distance from neighbor to target \((d_{N2T})\).

The euclidean distance is used to determine \((d_{S2T})\), \((d_{C2T})\), and \((d_{N2T})\). The input variable “distance” and “neighbor type” are calculated by Eq.4.3 and Eq.4.4 respectively.

\[
\text{Distance} = \frac{d_{C2T}}{d_{S2T}} \quad (4.3)
\]

\[
\text{NeighborType} = \frac{(d_{N2T}) - (d_{C2T})}{(d_{N2T}) + (d_{C2T})} \quad (4.4)
\]

The steps involved in the calculation of the neighbor preference rate are designed in a Fuzzy Interference System (FIS). The input variables to be fuzzified are temperature (buffer usage status), distance, and neighbor type. The terms “Empty”, “Few”, “Half”, “Almost”, and “Full” are used to describe the temperature field. “Close”, “StartPoint”, “Far”, and “VeryFar” are used to explain the relation of distance and “Closer” and “Farer” describe neighbor types. In Fig.4.7, the membership functions of the input variables are shown.

Fig.4.8 shows the membership functions of the only output, neighbor rate. The message will be sent to the neighbor who has the highest rate. Five terms of neighbor rate are defined from lowest to highest; “VeryBad”, “Bad”, “Fair”, “Good”, and “VeryGood”.

A fuzzy rule table is designed to determine what control actions take place under which input conditions to find an optimal path as shown in Table.4.1.

There are 42 rules defined for this fuzzy system. Some examples are:
### Table 4.1: Fuzzy Rule Table

<table>
<thead>
<tr>
<th>Distance</th>
<th>Neighbor</th>
<th>Thermal</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Empty</td>
</tr>
<tr>
<td>Close</td>
<td>Close</td>
<td>VeryGood</td>
</tr>
<tr>
<td></td>
<td>Far</td>
<td>Good</td>
</tr>
<tr>
<td>StartPoint</td>
<td>Close</td>
<td>VeryGood</td>
</tr>
<tr>
<td></td>
<td>Far</td>
<td>Good</td>
</tr>
<tr>
<td>Far</td>
<td>Close</td>
<td>VeryGood</td>
</tr>
<tr>
<td></td>
<td>Far</td>
<td>Fair</td>
</tr>
<tr>
<td>Farer</td>
<td>Close</td>
<td>VeryGood</td>
</tr>
<tr>
<td></td>
<td>Far</td>
<td>VeryBad</td>
</tr>
<tr>
<td>VertFar</td>
<td>Close</td>
<td>VeryGood</td>
</tr>
<tr>
<td></td>
<td>Far</td>
<td>VeryBad</td>
</tr>
</tbody>
</table>

- **R1**: IF distance IS Close AND Neighbor IS Close AND thermal IS Empty THEN neighbor_rate IS VeryGood;
- **R2**: IF distance IS Close AND Neighbor IS Far AND thermal IS Empty THEN neighbor_rate IS Good;
- **R42**: IF distance IS VeryFar AND Neighbor IS Far THEN neighbor_rate IS VeryBad;

The defuzzification is the process of conversion of the fuzzy output set into a single number. The method “Center of Gravity” (COG) is chosen as show in Eq. 4.5, where $x_i$ is the element...
and \( \mu(x_i) \) is its membership function. COG method is the most widely defuzzification strategy, which is reminiscent of the calculation of the expected value of probability distributions.

\[
\text{NeighborRate} = \frac{\sum_{i=1}^{n} X_i \cdot \mu(x_i)}{\sum_{i=1}^{n} \mu(x_i)} \quad (4.5)
\]

Fig. 4.9 shows the flowchart of the routing decision mechanism by fuzzy logic. The additional configuration file is required in fuzzy control process due to specifications of the java library [18]. The simulation constants must identify the location of fuzzy control (.fcl) file. All other requirements are same as in the previous algorithms.

![Flowchart of routing decision by fuzzy logic](image)

Figure 4.9: Flow of route decision by fuzzy logic

Route decision with fuzzy logic calculates the proportion of distance from current location to target and distance from source to target, and the neighbor type. Next, the neighbor rate of each neighbor determined by using distance, neighbor type and the neighbor’s temperature. Finally, the neighbor with the maximum rate is chosen as next hop.

### 4.2.3 Simulation Results and Discussion

The multi-criteria routing algorithm by fuzzy logic is analyzed in this section. The simulation environment is described in Section 3.3.1. This simulation uses the “jFuzzyLogic” library for the fuzzy components. *JFuzzyLogic* is an Open Source Fuzzy Logic library and FCL language implementation [18]. The package written in java is maintained and developed by Pablo Cingolani. The fuzzy control file composes of membership functions of input and output variables and the according fuzzy rules.
To analyze the results of multiple criteria routing using fuzzy logic, we selected the best results of all previous algorithms for comparison. The selected weight functions are \((0.9:0.1)\) and \((0.8:0.2)\). Additionally, the probability functions 1 and 2, as well as the constant probability of 0.4 and 0.5 have been chosen. Also, the shortest path algorithm is examined. Fig.4.10 and Fig.4.11 show the comparisons of packet delivery ratio and routing time, respectively. All routing methods considering thermal field approach have superior success delivery ratios than the shortest path approach. The fuzzy logic shows the best result of success delivery ratio (99%) but routing time is a single step higher compared to the weighted \((0.9:0.1)\) approach. Both fuzzy logic and weight function considering temperature field can reduce the waiting time (11 and 12 steps) compared to the shortest path algorithm, although the hop-count is a few steps higher. In contrast, the greedy method shows longer waiting time (18 steps) while multi-criteria routing algorithm considering buffer usage results in lower waiting time.

The results of both packet delivery ratio and routing time support that the concurrent multi-criteria route decision is better than the shortest path algorithm, and also better than the criteria selection multi-criteria routing methods as described in the previous section. Moreover, the route decision by fuzzy logic allows a very good load balancing in the system as presented in Fig.4.12 and Fig.4.13. In Fig.4.12, the number of overloaded nodes from shortest path are displayed by the black line, and the red line presents the number of overloaded nodes when using fuzzy logic. The different dotted and dashed lines in between show the numbers of overloaded nodes when weight function, probability function, and constant probability considering thermal field approach are used. It is clear that the fuzzy logic is able to find a better route than all other methods without using global constants. Messages avoid overloaded peers with only a bit longer routes and have less waiting time. The number of peers which have to handle messages is spread to a wider area, so that the messages queues have no higher load as shown in Fig.4.13.
An example in this traffic pattern is analyzed in order to compare the approaches more detailed. When using the shortest path approach, up to 101 nodes are overloaded during the simulations (red and pink dots), whereas only 22 nodes are overloaded when using fuzzy logic and 31, when using a weight function ($\alpha_{0.9}:\beta_{0.1}$). So, the number of overloaded nodes is decreases by more than 75% by fuzzy logic and 70% by weight function. Messages are transmitted to buffer-free nodes nearby the highly-loaded peers, so that a lot of loaded, but not overloaded peers can bee seen around overloaded nodes (yellow and grey dots). Instead of forwarding messages into an already highly loaded area, the proposed multi-criteria routing algorithms effectively find an
alternative traffic-free path using only local knowledge and hop-by-hop decision.

In summary, the concurrent multi-criteria routing approach is efficient in finding optimal routes that improve QoS routing performance. The loss ratio decreases about 25%, the waiting time is reduced by 40% and the number of overloaded peers is decreased by 75% at compared to the shortest path algorithm. Even though the weight function ($\alpha0.9;\beta0.1$) shows outstanding results similar to fuzzy logic, fuzzy logic is superior, because global constant or configuration for all dynamic situations is not required.

![Network diagrams present buffer status from fuzzy logic and shortest path algorithm](image)

4.3 Summary

In this chapter, the second part of multiple criteria decision making has been presented. The temperature from thermal field approach and relative distances from coordinate system have been considered concurrently to find an optimal path. First, the routing process is carried out by considering both relative distance and buffer usage level in a weight function. The weight presents the priority among the determined constraints. The optimal path appears on a
minimum weight route. The different proportion of considering temperature and distance have been analyzed. The results support our hypothesis that considering both criteria in a routing decision concurrently shows better results than the sequentially executed decision making.

The fuzzy technique is last method for route decision presented in this thesis, chosen due to its efficiency to handle uncertainties and non-linearity situations like in today’s P2P systems. The simulation results sustain our idea that fuzzy logic provides the best routing performance compared to using weight function as well a the criteria selection methods. Fuzzy logic shows the highest success ratio and decreases the delay time remarkably, with only a bit higher number of hops. The routing time is less than that in shortest path approaches, even though a bit higher than that of the best weight function result.

Simulation results of the proposed routing algorithms considering thermal field sustain our idea that routing methods considering buffer utilization improve routing performance. The packet dropped ratio and queuing delay time decreases, and the load balancing in the system is strongly improved.
Chapter 5

Self-Organizing Overlay Topology

The overlay topology is the pattern of how interconnections of network elements (links, nodes, etc.) are managed. It can be represented as a graph, in which the edges describe existing communication channels between nodes and thus represent possible ways of data transmission. Structures can be distinguished by different criteria, e.g., degree of decentralization, model, distribution of power, and functions. The architectural design of overlay networks aims to support specific functions and operations of various overlaying applications and systems. For example, CAN is designed to provide distributed hashtables using a multi-dimensional Cartesian coordinate space on a multi-torus, Chord is connected using a uni-directional and circular keyspace, and Pastry and Tapestry use a Plaxton-style global mesh network [68].

On the hypothesis that the structural design and resulting patterns of the overlay network can improve QoS routing by dynamically changing the structure, the Bridge algorithms are now presented. Originally, Jon Kleinberg introduced a model of small-world network [50], a grid structure with long-range links (or shortcuts), and proofed that on such an architecture, the delivery time for routing a message between any two nodes with greedy algorithms and only local knowledge, has an upper limit of $O(\log^2 n)$. In this chapter, a dynamically changing grid structure is discussed that uses knowledge from the thermal field to add additional links for shortcutting congested areas. Long-links are maintained relying on the thermal field with the objective to decrease loss ratio and balance the load in the system. Long links are added and changed dynamically according to the current network conditions.

5.1 Motivation

The proposed method generates shortcuts, called “Bridges”, in an overlay system depending on dynamically changing network traffic. We expect to be able to increase transmission reliability by reducing loss ratio and delay time due to overloaded buffers, as well as to decrease overall routing time. Again, a thermal field is used to represent buffer usage levels of the network nodes.
A bridge across an intersection in Bangkok, one of the world’s biggest cities with a large traffic congestion problem as shown in Fig.5.1, provided the inspiration for the proposed bridge algorithm. When the traffic situation at an interjunction becomes unmanageable, adding a shortcut in form of a bridge provides an alternative to ease traffic jams. The biggest advantage of a bridge is that it breaks up the existing traffic congestion. Additionally, it allows cars to get through the area unimpeded and also reduced the number of cars on the road. When comparing this real-world scenario to communication networks, an intersection could be the analogon for a highly loaded peer (high buffer utilization) leading to loss of messages and increased delay times. A bridge therefore would be generated by providing a shortcut link crossing the congested network area. In difference to real bridges, such virtual shortcuts can be maintained, built and removed with only a low cost. When a link is not necessary any more, it can easily be removed from the system.

The self-organizing bridges algorithm starts when a node heats up to a predefined value. Two types of agents are used to work cooperatively to build bridges overpassing high-traffic peers. The first type of agent is used to find candidate-nodes for building virtual links. The second one works for distribute the information of possible bridge starting points through the neighborhood. It is possible, that links are built but are seldomly used. In this case, they should be removed from the system. In our proposal, a counter is assigned to every bridge that is used as a indicator for how long the link has not been used. So, unnecessary links can easily be removed, when this counter reaches a predefined threshold.

### 5.2 Self-organizing Bridge Algorithm

As a base, an $n$-dimensional grid structured network is established. The network is identified with a set of lattice, a $m \times n$ square, $(i, j) : i \in 1, 2, \cdots, m, j \in 1, 2, \cdots, n$. The lattice distance
between two nodes \((i, j)\) and \((k, l)\) is the Manhattan distance \(d((i, j), (k, l)) = |k - i| + |l - j|\). A node in the community has four bi-directional edges to its direct neighbors. The algorithm is executed without any centralized control. As in the previous sections, the temperature indicates the usage level of a peer’s incoming- and outgoing message buffer. The temperature of a node \(c\) is referred to as \(\theta_c\). The temperature is calculated as shown in Eq.3.1. Each node only uses a single buffer, which is organized in a FIFO manner. Hence, the temperature of that buffer is equal to the temperature of the node. Since the adaptation of the grid strongly depends on \(\theta_c\) being up to date, the temperature is recalculated when messages enter or leave a buffer.

1. Initialization process, important terms:

- **Heat level**, \(\theta_H\), is a temperature level that designed as a starting point of overloaded buffer.
- **Cool level**, \(\theta_L\), is the maximum temperature level a node should have to be a suitable bridge start- or endpoint.
- **Heat time**, \(\tau\), is the time period, through which an area should be on heat level \(\theta_H\) or above to indicate the need for a dynamic bridge.
- **Hot Peer** is a peer, which temperature is above or equal to the heat level for at least the heat time or longer.
- **Jumper Peer** is a peer which has a temperature lower than the cool level.
- **Seeker** is an agent responsible for searching jumper peers around a hot peer.
- **Builder** is an agent who works in distributing jumper peer’s information to others nodes for creating bridges over heat node.

2. The self-organizing bridges algorithm starts when a hot peer has emerged. An example is shown in Fig.5.2a, in which node \(u\) is constantly heated up to a heat level of \((\theta_u \geq \theta_H)\).

![Figure 5.2: Bridge Algorithm](image)

(a) Traffic peer \(u\) finds suitable nodes for generating long links  
(b) Jumper nodes create long distance links to each other  
(c) Many bridges are built over a congested area

3. At Hot Peer \(u\),
• Four seekers are generated.
• The seekers are forwarded to four directions until they find jumper nodes \( w, x, y, \) and \( z \), such that \( \theta_w, \theta_x, \theta_y, \) and \( \theta_z \leq \theta_L \).
• The seekers die either when they find a jumper node or their life time exceeds a predefined TTL value.

4. When jumper nodes \( w, x, y, \) or \( z \) are assigned, each jumper node generates an acknowledgment and sends it back to the original heat node \( u \) to inform it about routing and jumper node details.

5. The second agent, \( Builder \), is launched when node \( u \) receives acknowledgments from four seekers within the double seeker life time. Otherwise, the heat node resends a seeker into the missing seeker’s direction.

6. Once jumper peers receive information from a Builder agent, bridges are created as shown in diagram Fig.5.2b.

7. Finally, nodes \( w, x, y, \) and \( z \) send acknowledgments back to the original heat node \( u \) to confirm that the bridge creation process is completed. The network diagram in Fig.5.2c shows a scenario, in which many heat nodes exist. When new shortcuts are created, the peer also creates a counter that indicates, for how long that long link has not been used. Once this counter reaches some \( \delta t \), the according link is removed from the system.

To analyze the algorithm’s performance as well as the impact that bridges have on the system’s load, the next section discusses simulation results gathered from running the proposed algorithm on a static grid structure.

5.3 Simulation Results and Discussion

The following simulations are made to proof the hypothesis that the self-organizing shortcut algorithm that manages shortcuts across overloaded areas can decrease traffic overload, decrease loss ratio and queuing delay, and provide a better traffic load. The experiments are designed to focus on the loss ratio effect resulting from bridges. In addition, the number of overloaded nodes as a function of the number of bridges will be discussed.

5.3.1 Simulation Model

Networks - the networks are organized in a two-dimensional grid structure composed of 10,000 nodes \( (100 \times 100) \). Nodes are connected to their neighbors in all four directions, except nodes on the borders. The coordinate of a node serves as its ID. The grid overlays a simulated IPv4 network. The buffer sizes and outgoing bandwidths are limited for all the peers randomly
following a power-law distribution. There are three types of messages in the simulation; data packet, algorithm agents and acknowledgments. The system handles both data packets and algorithm agents in First-In-First-Out (FIFO) manner, and keeps them in the same buffer as all other messages, while the acknowledgments are handled with higher priority due to their importance for reducing network load.

Traffic pattern - traffic is generated randomly by all network nodes from starting time until simulation time 1000. The sending probabilities and intensities are distributed exponentially over all generating source nodes, as well as the number of messages that can be sent per simulation time-step. All simulations generate 325,000 messages in total. The routing algorithm uses a greedy method to forward data messages to its target. Messages are routed to the neighbor with the lowest distance to target. The distance is measured using the Euclidean Distance of current and target node ID.

Self-organizing Bridges Parameter - four parameters are defined; hot value, hot period, suitable thermal value, and unused period as discussed before. The simulation is configured to use \( \theta_H = 0.7, \tau = 10 \text{ time-steps}, \theta_L = 0.1, \) and \( \Delta t = 20 \text{ time-steps}. \)

Performance metric - the metrics are as follows.

- Packet delivery ratios (success, loss and expired): The packets that successfully reach their target. Packet loss (dropped) refers to packets that cannot be handled due to overloaded buffers of a peer on the route. Expired packets are those packets that reach their time-to-life before they reach their target.

- Average hop-count: The average of number of hops that are necessary to deliver a packet.

- Average delay time: The average delay time that a message has wait in buffers during routing process.

- Average routing time: The routing time is the sum of hop-count and delay time. It is the total time needed to route the message from source to target.

- Number of overloaded nodes, which temperature is greater than 70\% \( (\theta_c \geq 0.7). \)

- Number of bridges (long-links) in the system.

5.3.2 Simulation Results Discussion

First, the packet delivery ratio and routing performance are analyzed as presented in Fig.5.3a. The success ratio of self-organizing bridges network shows 19\% better result than routing in the static network. The bridges obviously help decreasing the loss ratio. Messages are routed over overloaded areas, so that no more messages are filling the buffers of already highly loaded peers.

The routing time is analyzed in Fig.5.3b, in which the comparison of average hop-count and
waiting time is presented. The hop-count from a self-organizing bridges network is a bit (1.7 time-steps) less than that in the static network. But the average delay time decreases remarkably from 24.05 time-steps to 10.62 time-steps. The average routing time decreases also remarkably from 114 time-steps to 99 time-steps. From the results, the bridges or virtual links built across overloaded peers make more messages reach their destination and additionally reduce the number of highly loaded peers.

The graphs in Fig.5.4 support the results gathered from Fig.5.3a. Using the bridge algorithm, the number of message losses is strongly reduced. One can see that it took roughly until simulation time 100 until the bridges have been built, so that in the beginning, almost the same number of messages are lost as in the static case. Nevertheless, after simulation time 200, enough bridges have been created to strongly reduce the number of message losses.

Next, the load balancing results of grid networks are compared in Fig.5.5. The network traffic
intensity is presented by the number of hot nodes at every simulation time-step. Before the self-organizing bridges algorithm starts, the numbers of hot nodes from both networks are same. Once the algorithm is executed, the number of overloaded node notably decreases as shown by the red-dotted line. The bridge clearly balances the load in the network.

The self-organizing bridges network is analyzed in deeper detail in Fig.5.6. The graphs present the number of bridges together with the number of heated nodes as a function of simulation time. The number of overloaded nodes depends on the number of bridges. After the algorithm has been started, the number of bridges increases dramatically until roughly simulation time 200 is reached, from which on the system of bridges is organized. The number of bridges slightly increases the same amount as the number of hot nodes decreases.

An example of self-organizing grid network is shown in Fig.5.7, which is captured at simulation
time 120, 250, 500, and 1000. The center of network is enlarged presenting how the self-organizing grid looks like. At time 120, the self-organizing bridge algorithm has started so that many complete-short links appear. After that, some unnecessary links were removed as can be seen at simulation time 250. Later, longer bridges are constructed to release traffic instead of short bridges. The longest length of a bridge is 10 steps, while the shortest and average length of bridges is 2 steps. The short bridges are constructed across single hot nodes, whereas long bridges are built across a whole overloaded area.
Chapter 6

Extension Algorithms and Related Applications

This dissertation mainly contributes to novel algorithms applied to overlay networks in order to improve satisfaction of quality of service (QoS) routing considering multiple criteria. The physical network parameters and peer’s correlation are considered for routing methods, and the overlay topology has been designed in a self-organized manner relying on traffic situation as discussed previously.

In this chapter, the extension ideas related to main works are presented. First, Kleinberg’s small world is extended by distributed long-link information using Ant Colony System [58], contrast to original model that shortcuts is notified when messages visited node. Next, the long-link is generated and deleted according to node’s character or strategy. A network organization model with economic behavior is proposed [60]. This model is used to understand and predict the network structure from economic perspective. This network or system refers to a network where a node represents customer, provider, or a person, while an edge symbolizes trading, delivery method, dealing, or involvement, and a message or traffic in the network substitutes good, service, or information.

6.1 Long-Link Notification Algorithm in Small-World Network

6.1.1 Ant Colony Optimization

The idea behind this concept is to map biological ant’s behaviors to useful optimization techniques in the networking and telecommunication area. Already decades ago, the Ants Colony Optimization (ACO) was introduced by Dorigo et al. [30]. A colony of ants performs fantastic tasks; finding shortest path between their nest and food, and sharing information to other ants
by using pheromone trails. Researches have transformed such behavior into many different optimization algorithms. Sim and Sun [90] have elaborated a survey and present some examples of how the Ant Colony Optimization has been used for routing and load balancing in networks, such as in the Ant-based control (ABC) system, AntNet and some extensions. Additionally, they compared ant-based algorithms and genetic algorithms to traditional routing approaches.

An outstanding ant algorithm for routing is AntNet, as presented by Caro and Dorigo in [15]. In this approach, two ant groups, forward and backward ants, collaborate in building routing tables that adapt to current traffic with the intention to optimize the performance of the whole network. The SemNet presented by Michlmayr [72] is an example of an AntNet extension. She adopted the AntNet strategy for searching content in distributed systems. The ant system also is applicable for self-organized and self-adaptive routing as Rojas et al. introduced in [87]. They applied an Ant system to find alternative routes under congestion situations using a colorized pheromone.

### 6.1.2 Pheromone-Based Landmarks Algorithm

Our idea, a greedy algorithm is used for route decision as in Kleinberg’s model. When a long-range link is found and used to transfer a packet, an ant colony system is executed to propagate this possibly shorter path to surrounding peers as shown in Fig.6.1. The ants work as the messenger to broadcast long-range links to the environment by spreading pheromone randomly until they die, while the color of pheromone represents long link details.

![Figure 6.1: The color ants spread around when long-link has been used.](image)

The pheromone amount is dropped increasing exponentially by hop count, and the pheromone value is decreased exponentially by time in case no ants has visited. Each peer keeps only its own local neighbors and its shortcut. Furthermore, each node also keeps track of the colored pheromones from ants that have passed, as seen in Fig. 6.2. The according pheromone table consists of color, amount and neighbor who have sent the ants.
The color of a pheromone is set up according to HSV (Hue, Saturation and Value) color model [34]. HSV model, also known as the hex-cone color model, presents a type of color space as displayed in Fig.6.3(a). The model has three components: hue, saturation and value. Hue is the actual color that is measured in angular degrees counter-clockwise around the cone starting and ending at red = 0 or 360 (such as yellow = 60, green = 120, etc.) as shown in Fig.6.3(b). Saturation describes the grey scale of the color, identified by a value between 0 and 1. When the value is “0”, the color is grey and when value is “1”, the color is the primary color. The Value controls the brightness of the color and varies with the color saturation. It ranges from 0 to 1. When the value is ‘0’ the color will be totally black. With the increase in the value, the color space brightness up and shows various colors.

We apply HSV color model for identifying a pheromone’s color; Hue keeps the direction of the remote node in relation to the jumper node. Saturation keeps the length of the long-range link calculated in lattice distance. The Value by default is one. In Kleinberg’s small-world model, color pheromone is able to spread overlapping due to all nodes have their own long-link. At any node which has to decide for the next routing step of a packet, the next node to forward packet to is selected by the shortest distance to the message’s target. It is able to choose a node among its neighbors (its local neighbors and its long links) and remote nodes of long-links nearby, which it can identify through its pheromone table.
The algorithm is as follows:

1. At a simulation time \( t \), a packet is generated randomly toward a destination node \( d \) at source node \( s \) by a given constant probability.

2. While traveling toward their destination nodes, the packets memorize their source, target, and traveling paths.

3. At each node \( k \), each packet headed to destination \( d \) selects the next node \( n \) to move to.
   
   (a) Consider the closest neighbors \( n_1 \) to its targets and do not use the previous sender.
   
   (b) Consider the pheromone table, compare destination location \( d \) with nearby long-links, as defined in color pheromone, and select best neighbor \( n_2 \).
   
   (c) Compare \( n_1 \) and \( n_2 \) and then choose the one closer to target.

4. At each node \( k \), if the shortcut has been used the color pheromone algorithm starts.

   (a) Additional definition, node \( k \) which has long link that founded by packet at \( t \) is called *Jumper Node*. The opposite side of shortcut is called *Remote Node*.

   (b) A group of ants is generated at jumper node \( k \). The number of ants in the group and the ant ages are predefined as global constants. The starting age of ant is always zero when they are born.

   (c) The system calculates pheromone color following the HSV color model based on the direction of the remote node in comparison to the jumper node and link distance.

5. At each simulation time step, data packets and ants are forwarded. The ant has higher priority than the data packet, since it is small and objects to drop pheromone, only.

6. The ants characteristics:

   (a) They walk around jumper node using paths of non-uniform random distribution and are not allowed to return to their sender or jumper direction.

   (b) The color pheromone \( (\sigma) \) is dropped at every node they pass until they die, i.e. they reached the maximum ant age.

   (c) The dropped amount of pheromone is increased according to the distance from jumper node or ant age, as in Eq.\( 6.1 \).

\[
\sigma' = \sigma \cdot [1 - \exp^{-\lambda t_{\text{ant}}}] 
\]  

(6.1)

7. The ants update pheromone value when they visit peers that already have pheromone information (keys are color and neighbor ID)

8. When no ants visit at peer, the pheromone is decayed exponentially as a function of the “ant-free” time, \( t_{\text{ant-free}} \), as Eq.\( 6.2 \).
\[
\sigma' = \begin{cases} 
\sigma \cdot [1 - \exp^{-\lambda \text{ant-free}}], & \sigma \geq 0.0001 \\
0, & \sigma < 0.0001 
\end{cases}
\] (6.2)

The method described above ensures that the color pheromone is generated and ant colony distributes long link information, as well as it can be used by nodes for finding fastest route decision. Some experimental results are given in the next section.

6.1.3 Simulation Results and Discussion

Simulation Environment

A small-world network is simulated; two-dimensional grid sized 10,000 nodes (100x100). The long-range links were random added between any two nodes with a probability proportion to lattice distance: \(d(u,v)^{-2}\). All the links are bidirectional graphs noted that this point is different from Kleinberg’s model. There are ten percentages of source nodes that uniformly-randomly distribution in network which aim to generate a message send to their target in every simulation time with a constant probability value. This network is used in all following experiments. All simulations were generated 50,000 data messages and run until all packets reached their targets. The system handles the data packet in First-In-First-Out manner. Each packet overhead contains only source ID, target ID, and path that passed. Each data message has 500 TTL time units. The color pheromone parameter setting for exponential functions as mentioned in Eq.6.1 and 6.2, the lambda (\(\lambda\)) value for calculating volume of pheromone drop and amount of pheromone evaporation were set to one in all experiments.

The simulation has taken into account the two scenarios. First case aims to analyze the influence of number of ants per group and time-to-life of color ants. The jumper node spread groups of 10, 15, and 20 ants at each time, and TTL of ants were 3, 4, and 5 time unit. The probability of packet generate in each time unit was 0.3 for all source peers. Second scenario intends to take the congestion into account then we defined the probability of packet generate to 0.3, 0.5 and 0.7. The jumper node spread two types of ants; ten three-aged-ants and fifteen four-aged-ants.

Results Discussion

In the first scenario in Fig.6.4, compare routing performance by the average routing time (Z axis) between greedy algorithm and color pheromone algorithm. The parameters of color pheromone are set. Three different numbers of ants per group (Y axis) are 10, 15, and 20 and three different values of ant age or TTL (X axis) are 3, 4, and 5. From graph, the average routing time of greedy algorithm is 7.23 time unit, showed in small net. It is only one value due to its routing time dependent from color pheromone parameters. The second net, big one presents the average routing time of color pheromone form nine combination cases. The best
result of color pheromone method, the average routing time is 7.20 when ant age is three and numbers of ant per group are ten and twenty which is better than greedy one

![Figure 6.4: The average routing time compares greedy algorithm with color pheromone algorithm when number of ants per group are 10, 15, and 20, and ant age are 3, 4, and 5. The probability to generate packet is fixed at 0.3.](image)

In contrast, some two peaks in the graph show higher routing time than greedy approach; the values are 7.28 and 7.26 time unit, when ant age is five. This result came up because pheromone amount is not up-to-date. The pheromone trails decayed exponentially when time that none of ants pass the nodes. In case the packets decided to follow pheromone but the pheromone is evaporated while moving to the jumper node. The packets can decide to select local neighbors instead of going to jumper node although previous few steps finding the jumper station if pheromone is not enough. Such that situation makes delivery time longer than usual. This situation is more explicit when we use small group of ants. Number of ants is not enough to spread through the area and not enough to maintain the pheromone until next packet need. Then it makes time higher according to the influence of color pheromone.

The life time of ants implies to circle of pheromone spread area. The number of ant per group implies to opportunity to visit every node around within area. The short life ants present better performance than long-life one. And big ant group in cooperation are better than small group. Of course, big group leads to more processing time and resource consumption than small one, then the optimal point should be considered depend on network size and resources.

The second scenario in Fig.6.5 and Fig.6.6, this test intend to analyze influence of congestion effect to color pheromone. Fig.6.5, we tested with two color pheromone parameter sets, ten ants per group with three steps of age and fifteen ants per group with four steps of age compare to three probability values for generating message per time unit; 0.3, 0.5, and 0.7. And, of course, compare average routing time to greedy algorithm as presented in small net diagram. All results of color pheromone present better performance than greedy algorithm in these cases. The best result of color pheromone (big net) comes from probability value is 0.7 and set ten three-aged-ants per group that shows average routing time 7.18 time unit. So the average routing time increase when probability to generate packet decrease. Also longer-life ants lead
Figure 6.5: The average routing time compares greedy algorithm with color pheromone algorithm when the probability to generate packet are 0.3, 0.5 and 0.7. Compare with two set of ants; ten of three-aged-ants and fifteen of four-aged-ants.

Figure 6.6: The average routing time compare greedy algorithm (Straight line) with color pheromone algorithm ('X' marks with lines) of ten three-aged-ant group. This graph observed packets that have same source and target peers.

to longer routing time.

Fig.6.6 demonstrates the results of greedy and color pheromone routing methods comparing of messages that generated from same source send to same target. The straight line is actual routing time by greedy method. And blue 'X' mark is the mean of routing time by color pheromone with standard deviation line. Graph shows there are some messages by color pheromone using longer route than greedy, however, more messages are forwarded faster.
6.2 Network Creation Model

Today we have several popular social networking websites, such as Twitter, Facebook, YouTube and MySpace that facilitate commercial interests. Their users can generate news, stories, events and products that may be of interest for the world wide community. For instance, when customers and providers look for profitable prospects online, the commercial products can be advertised directly to their targeted consumers, and customers can also search for their desired goods easily. Both are benefiting from the network. Moreover, the decision of a customer whether or not to buy a product or use a service is often influenced by the choices of their neighbor (friends, socials, or professionals). In such cases, we can apply the idea to a network where a node represents customer, provider, or a person, while an edge symbolizes trading, delivery method, dealing, or involvement, and a message or traffic in the network substitutes good, service, or information.

Therefore, it is a challenge to understand the success of these networks in game theory. How do interactions of selfish members lead to an efficient network structure? How do they form their behavior with economic drive? And how can we design the rules to optimize social objectives, while individuals are trying to reach their own goal? Game theory provides a general framework of these models which facilitate a study of efficiency and stability in network formation. The network game model has been introduced a decade ago, both in theoretical and empirical aspects. Many of previous studies use the Nash equilibrium as a concept solution and referred the networks in correspondence to these equilibrial stages as being stable. To evaluate overall quality of the networks, they compared the worst and best cases to the optimal social value. In particular, they are interested in Price of Anarchy and Price of Stability. Price of Anarchy is the ratio of the worst possible equilibrium to the social optimality. It measures how well people perform when they act selfishly while being controlled by central authorities [3, 25, 32]. In contrast, Price of Stability is a ratio of the best possible equilibrium to the best value of the social optimal [7, 27]. Furthermore, network games have been used in regard to problems of routing, load balancing, facility locations, security, and so on [5, 46].

6.2.1 Related Works

Many recent theoretical works on network games focused their study on two values; Price of Anarchy and Price of Stability of the network. Fabrikant et al. [32] introduced network creation game model where links were generated by unilateral actions of players and links cost was one-sided. They found out an upper bound on the Price of Anarchy for all the costs of establishing any undirected edges. Each player recognized quality of a network as the sum of distances to all other nodes. Players aimed to minimize their cost function that combined both network quality and building costs. They attempted to minimize the sum of building costs and distance to all other players. In addition, in [25, 27] considered wider range of the sum of link’s building cost in the unilateral game. Corbo and Parkes [25] examined the bilateral network game when two
players had to agree to establish a link between them. Recently, Demaine et al. [27] extended boundary of the sum game when cost paid by builder. In addition, they introduced a new concept of cost, the max game. The usage cost in the max game was the maximum distance to all other nodes which was different from previous works where the usage cost was the sum of all distances. The max game captured the worst-case scenario, instead of average-case behavior of routing as featured in the sum game.

Anshelevich et al. [7] worked on the Price of Stability for network design where the building cost of each edge was divided equally amongst users who used the connections. It was the best possible case comparing to the Nash equilibrium. They showed that the Price of Stability for network design with respect to the fair cost allocation was $O(\log k)$, where $k$ is number of users. Fiat et al. [35] also studied Price of Stability of undirected network with sharing cost. However, they considered only in the case where there was an agent in every vertex. Their Price of Stability was $O(\log \log k)$.

The network formation model with economic behavior is proposed in this thesis aim to study how such the success networks are formed. In the model, a set of $n$ players was initially connected in a ring-like structure. Their objective was to form an individual network for a high revenue. The winner is the richest player of the game. Messages are forwarded from one node to another through the network applying greedy algorithm (shortest path). Messages have to pay for using the node’s link. Then a node or player will be rich when he or she is able to form an efficient network. Competition emerges when only one path is selected for traveling. However, link’s maintenance cost has to be paid in each round. So some links have to be removed when player lacks of money. The player’s strategy of network formation model was categorized in three main actions:

1. Building a new link at a proper time
2. Building a new link to a targeted player
3. Removing a link from the network when maintenance money is insufficient

To evaluate network performance, account balance of player and number of links were measured. The overall system performance was analyzed by the average routing time of all messages.

6.2.2 Game Model

Common aspect of the model, a group of players with basic connection wanted to establish new relations to other players by forming long-distance links in order to make more profits. Messages are randomly generated and forwarded through the networks to their random targets in every round. Players in the game earned money from messages through their links and price depended on link distance. The longer the link, the more money produced to the link owner. However, players must pay the building cost as the first investment and the maintenance cost as for keeping the links in every game round. Hence, a long link was useful when it was built and
regularly used for messages transferring. Mission of the players was to construct an effective network for high income generation. Key success factor was to make more income and pay less maintenance cost by building efficient links. The destination of direct link was chosen upon player’s strategy, such as randomly or statistically selected. The statistical data in this paper represented the frequency of messages passed to destination nodes. The building cost of long links depended on the link’s distance between origin and destination, which was the same as the price of using that long link. The building cost was one time payment but player would earn money every time the link had been used to forward a message. In addition, long link had the maintenance cost per round which player must pay whether or not it had been used. If player decided to build many unused long links, they had to pay big amount of maintained cost. In the end, when players did not have enough money to maintain their links, they had to delete some of those long links.

1. **Topology**

The game was composed of a set of \( n \) players or nodes \( 1, 2, ..., n \) who were connected in a network. The network was defined as a graph \( G = (V,E) \) and comprised of a set of vertexes (players) together with a set of edges (neighbors). The neighbor of player \( i \) was specified by a subset \( n_i \) of \( \{1, 2, ..., n\} - \{i\} \) which corresponded to whom player \( i \) was connected by a link. This contribution, ring-like structure was preliminary form of the network. The link or edge was categorized into basic link and long link. Fig. 6.7 presents an example of network diagram.

![Ring-like Network](image)

**Figure 6.7: Ring-like Network**

- **Basic link** is non-direct edge and connects two neighbors whose locations are next to each other according to ring structure (blue line without arrow in Fig. 6.7). Let edge \( i, j \) be a basic link of player \( i \) and \( j \), then both \( i \in n_j \) and \( j \in n_i \). The distance or hop-count from player \( i \) to \( j \) is always equal to one, \( d(i,j) = 1 \). The basic link is initially built and not allowed to be removed from the network during the game. The usage cost is one unit per link and the maintenance cost is zero.

- **Long link** is direct edge and connects two nodes whose locations are at least two steps away (red line with an arrow in Fig. 6.7). Let edge \( i, k \) be a long link of player
i and k, then k ∈ ni and d(i, k) ≥ 2. The long link is constructed in the game depending on player’s strategy. The player has to pay money for building a long link based upon distance between players or link length. In addition, player must pay one unit for maintenance cost per link per round. Lastly, the usage cost depends on the link length, and that player earns more money comparing to the case of basic link.

2. **Game Rules**

**Traffics and Messages** In each game round, network traffic is generated randomly following the power-law distribution in receiving and forwarding messages by all players. When a message is generated, a routing cost to send this message from an origin to a target is originally assigned by the longest routing time. For each forwarding process, the message has to pay for the routing cost to player whom their links are used in every step. A greedy algorithm is used for routing the messages meaning that messages are transferred via the shortest path under remaining message’s money using hop-by-hop calculation. Messages are created repeatedly until it reaches the maximum amount. The game is ended when all messages are forwarded to their targets or become expired due to insufficient money.

**Player’s Action Definition** Three main actions contributed in player’s strategies of the proposed model are:

1. Generating and routing traffics as described in the previous section

2. Establishing and maintaining their own network considering two queries: *when do they need to create a long link?* and *which player or node to be destined for the new link?*

   (a) **Time to create a long link:** player calculates his or her budget amount (B) in every round. When money balance (A) is higher than the budget (B), building process starts.

   (b) **Destination of a long link:** the destination of a long link can be assigned randomly or by using statistical data. The statistical information in this paper shows frequency of message’s targets and frequency counted by original message’s source to message’s destination.

3. Deleting links when money is insufficient. A long link is kept as long as the player can pay the maintenance cost or else it is removed. This decision can be based on unused period (τ) or ratio of usage of each long link (ω) or randomly selected. When player’s money is not enough, then a randomly selected link, a link of longest unused period, or a link of the lowest ratio of usage will be removed.

3. **Player Strategies**

   **Strategy variable definition:**
The strategy of each player is the key to win the game. Players must define their own policy in creating effective long link, paying less construction cost and spending less maintenance cost, to achieve the goal. The variables of player’s strategy are described and defined below.

1. Parameters influenced by players
   - *Account balance*, $A(t)$, is a remaining amount at time $t$. While players have to spend their money on long link construction and maintenance, they earn some back when their links are used by messages.
   - *Budget*, $B(t)$, is the maximum amount of money prepared for building a long link at time $t$.
   - *Number of existing long link*, $\alpha(t)$, is a quantity of long links left at time $t$ after being generated and taken away during time $0$ to $t-1$.
   - *Reserved time*, $R$, is a period of time that players want to keep their long links and they hold some money for paying maintenance cost.
   - *Unused long link period*, $\tau$, is a period of time that a long link has not been used to forward a message. This criterion is for removing a long link in order to reduce the maintenance cost.
   - *Ratio of long link usage*, $\omega$, is a portion of used time to life time of each long link. This ratio is used to determine which link to be removed, and to identify the applicability of those long links.

2. Parameters independent from players
   - *Forwarding income*, $I(t)$, is total amount of money that player earns when messages are forwarded through basic or long links at time $t$. The forwarding price depends on link length.
   - *Construction cost of long link*, $C(t)$, is total amount of money needed to build long links at time $t$.
   - *Maintenance cost of long link*, $M$, is amount of money to be paid for keeping a long link at a time.

**Player’s Strategy:** Twelve strategies of players are studied. The parameters described in the previous section are functioned to determine the combined behaviors of the three actions into twelve cases.

1. *Time to create a long link:* creating time is factored by the budget in this model. A new long link will be created when player’s account balance is greater than or equal to budget at time $t$. Eq.6.3 presents account balance calculation.
\[ A(t) = A(t-1) + I(t) - [(\alpha(t) \cdot M) + C(t)] \quad (6.3) \]

And budget for constructing a new long link is presented in Eq.6.4.

\[ B(t) = A(t) - [R \cdot (\alpha(t) \cdot M)] \quad (6.4) \]

Players are categorized into Risky and Safety for time to create a long link.

- **Risky player** - a long link will be created when player has enough money and ready to spend the whole amount. Then \( B(t) = A(t) \) when \( R = 0 \).

- **Safety player** - a long link will be created when player has remaining money after a safety period. Then \( B(t) = A(t) - (R \cdot \alpha(t)) \) when \( M = 1 \) and \( R > 0 \).

2. **Destination of a long link**: players have two sets of data of other nodes. They are collected from messages passed through node and used for destination selection of long link.

   - **Random node** - player randomly chooses another player in the network whose location settled within budget.

   - **High Statistic node** - player selects nodes having highest target recorded from all the messages passed through node and located within budget.

3. **Deletion of a long link**: players having insufficient money for maintenance cost have to remove their long links. There are three long link selecting methods.

   - **Delete Randomly** - select existing long links randomly without considering any data.

   - **Delete Longest-Unused** - select a longest long link that has not been used.

   - **Delete Least-Used** - select a link of the lowest ratio of usage.

Examples of player’s strategy in twelve scenarios are as follows.

**Case 1**: “Risky player - Random node - DeleteRandomly” refers to a player who creates a long link when he or she has just enough money and randomly selects destination of that long link within budget. When player does not have enough money to maintain the links, he or she randomly removes a long link.

**Case 2**: “Risky player - Random node - DeleteLongest-Unused” refers to a player who creates a long link when he or she has just enough money and randomly selects destination of that long link within budget. When player does not have enough money to maintain the links, he or she removes a longest unused long link.

... 

**Case 12**: “Safety player - High statistic node - DeleteLess-Used” refers to a player who creates
a long link when he or she has remaining money after cost deduction in a safety period. Player builds a long link to another player who is the most frequent targeted locating within budget. When player does not have enough money, he or she removes a link of the lowest ratio of usage.

At each round of the game, the building long link process starts when player’s budget is greater than zero. Then, the destination is chosen according to their budget and strategy. When multiple destinations with the same creating cost fit to player’s strategy, one node is randomly picked. If the selected destination has already had a long link, the lower candidate is selected respectively.

4. How To Play The Game

Basic instructions for playing the game are predefined as the followings. In each round, time \( (t) \):

1. Player checks total number of messages generated. If it reaches maximum number, go to step (4).

2. Player randomly generates a message according to his or her generating probability, as well as randomly assigns target to the message.

3. Push the new message to the forwarding queue (or outgoing buffer).

4. The maintenance process starts. Player pays the maintenance cost of existing long links. If there is not enough money \( (A(t) \leq \alpha(t) \cdot M) \), a long link must be deleted. The deleted link selection is based on player’s strategy. Run deletion process repeatedly until there is enough money for paying maintenance cost \( (A(t) \geq \alpha(t) \cdot M) \). Next, player’s account is deducted by number of long links when \( M = 1 \). If balance is zero, \( A(t) = 0 \), go to step (6).

5. The building process starts. Player calculates the budget for this round using Eq.6.4. If player has enough money \( (B(t) > 0) \), a new long link is created according to his or her budget and strategy. Then player’s account is deducted upon link’s distance.

6. The process of receiving and forwarding messages starts.

(a) If incoming message buffer is not empty and current player is not a message’s target, then push message to forwarding queue.

(b) If outgoing message buffer is not empty, then repeatedly forwards a message to the next neighbor who is closest to the message’s target, considering the remaining money for sending messages, until the buffer is empty. A message may not be sent via the best route if it does not have enough money to spend for routing through best path. In the end, message’s account is deducted before using the link.
7. When there are no more messages processed in the network, the game is over. If not, repeat step (1) to start the next round.

At the end of the game, the richest player is the winner. In addition, the achievement of network creation game is not only measured by player success. The network structure can also be evaluated and average routing time of all messages identifies the quality of player’s strategy. Experiments from above mentioned model and result discussions are presented in the next Section.

6.2.3 Simulation and Discussion

Setup

The proposed game was modeled by changing different player’s strategies in the network simulator. Player’s profile was set using combination of game parameters stated in the Section 6.2.2 and the simulation results of different strategies are discussed.

Network Setup - There are 30 nodes preliminary formed as a ring topology in IP4 network for different scenarios. The buffer sizes and bandwidths were unlimited for all the nodes. The system handled data forwarding in First-In-First-Out (FIFO) manner.

Traffic - it made by all nodes randomly generated ten messages per simulation time according to predefined forwarding probability, and each message was randomly defined a target according to receiving probability, following the power-law distribution.

Economic - In each game, the network generated totally 1,500 messages. The game ended when message procedures were completed. The routing cost for new messages was set for the longest distance in the ring structure from source to target. So, the maximum routing time of messages is 15 time-steps in 30-players network. The usage cost of basic link, as well as the maintenance cost of long link was each equal to one unit. The usage cost of long link is defined by the link length.

Results Discussion

The results of the game were presented into two parts; influence of dominances and structural convergence. First, the influenced parameter is measured and presented in forms of average routing time, number of added and deleted long links, and average player’s account balance. The routing time represented quality of player’s strategies to construct individual network that affected the whole community performance. The stability of the network is indicated by number of added and deleted long links. And final account balance of individual player showed financial performance by strategies.

Fig. 6.8 shows average routing time of twelve cases. The graph is presented in four different
groups of deletion decision methods. It shows that deletion decision had less influence on the average routing time than that of creation method. The destination selection had the most influence on routing performance as shown in the graphs of High-statistic-node which had the lowest routing time. The Risky player possessed superior results in routing performance than Safety player.

In Fig. 6.9, the number of added long links and number of removed long links are compared in each scenario. Fig. 6.10 shows the comparison of average income and average remaining balance at the end of the game per node. The worst case was identified in RiskyPlayer - RandomNodes players as shown in both graphs. A lot of long links were built and removed during the game. Their network changed dynamically and remaining balance was zero. They used up their income
in building and maintaining their long links, but they could not make much money from those investments. The long links were useless in the cases of Del.longest-unused and Del.less-used players where the representing graph bars are very high. These confirm our conclusion that the deletion methods are less significant.

To sum the first part, the most influential parameter is the strategy to select destination of long links. When players incorporate statistical data in their decision to build long links, the network performance is good; low routing time, high stability, and high account balance. The Risky or Safety behavior has less influence compared to destination selection methods. In addition, different deletion methods do not affect the network performance and player’s financial status. The simulation results of all three deletion methods were similar although other parameters were changed.

In the second part, structural convergence was analyzed by investigating network changes
through out simulation time. We fixed that players in these simulations used only longest-unused method for deletion as the deletion decision had no influence shown in previous part. So, the following cases were divided according to player’s long link creation strategies, into four characters: Risky player - Random node, Risky player - High statistic node, Safety player - Random node, and Safety player - High statistic node. The numbers of added and deleted long links and the average routing time by message-generated time were measured.

![Figure 6.12: Number of deleted long links by simulation time](image)

In Fig. 6.11 and 6.12, the number of added long links and deleted long links were shown respectively. In Fig. 6.11, the graphs of added long links that players randomly created increases rapidly from time 2 to 20, then increases slowly to the end of the simulation. There was no stable period. In contrast, the graphs of added long links that players selected high statistic node grows up to approximately 120 links then the number of added long links starts to be stable at simulation time 10. The graphs imply that stability of the network occurs when players use High-statistic-node strategy in both Risky and Safety types. Fig.6.12 shows that Safety players built good structures where long links were useful. The number of deleted long links from Safety players was lower than those of the Risky ones. At simulation time 50 when message generation process stopped, the Random player graphs raise steeply because there were no more messages using the links and paying money for, then a lot of long links were removed during last set of messages routing. In contrast, the graphs of High statistic node increased slightly at the end because players were already rich and they had effective long links. The rich players are able to pay maintenance cost for a long period of time. The effective long links also made the game ended quicker.
Chapter 7

Conclusion

7.1 Key Contributions and Results Evaluation

This thesis presents novel algorithms in overlay networks that improve the satisfaction of quality of service routing (QoSR) by considering multiple criteria, i.e., physical network parameters, content correlation, and the overlay topology. This thesis focuses on using a peer’s buffer level as second criterion in addition to the shortest number of hops. The main assumption judging this is that when transporting large datasets over a well-connected overlay, buffer overflows and resulting message losses may occur much more often. A thermal field approach was therefore introduced to propagate a peer’s buffer level. The approach is also suitable to propagate any kind of property, such as bandwidth, buffer size, or content popularity (e.g., PageRank or number of download) at minimum costs for maintenance and message delivery in grid-like structures.

Next, routing algorithms have been presented that consider several constraints from both the physical and the overlay layer. Using the thermal field to communicate the buffer utilization level, the temperature gradient indicates the direction to overloaded regions, which gives benefit to the routing decision. A grid-like overlay has been used, because it provides both an easy mechanism to find the shortest path and allows for a large amount of alternative paths to select, when a region is overloaded. Several methods for routing in grid overlay networks have been introduced and discussed in this thesis.

1. In the first approach, at each routing step a single criterion is selected by a pre-defined probability. The route is randomly selected hop-by-hop and either follows the shortest path or the lowest temperature route. The idea behind this is to explore the network step-wise and find low-temperature regions, but never leave the shortest path too much while doing so. So, finding appropriate values for the system-wide criteria-selection probability is the key problem. The experimental results show that the adaptive routing algorithm is able to improve routing performance remarkably, since loss ratio and overall waiting time are decreased. The optimal value for the criteria-selection probability has been found by
Conclusion

1. Simulation and lies at about 0.5, i.e. if both criteria are taken into account with the same weight, the optimal tradeoff is found. A probability too high leads to very long routes and increases the loss ratio due to expiring TTLs. On the other hand, the loss ratio due to routing into overloaded area is high when the probability is too low.

2. A constant probability might not be suitable for all traffic conditions and message locations. Especially when a message is close to its target, an adaptive path leads to unnecessary longer distances. Hence the random routing relying on one constant probability value has been extended by using probability functions dependent on the remaining distance to target. The idea is that when packets move closer to the target, routing directly (i.e. by shortest path) should have more weight than low-temperature paths. If on the other hand the message is still far away from the target, exploring low-temperature paths might be more helpful and even more important that directly going onto the shortest route. The proposed algorithms have been examined comparing them to a greedy algorithm. The results show that loss ratio decreases as before, when it is avoided to forward messages through overloaded peers, but the delay time is not clearly improved. However, the proposed algorithm shows better balance of load in the network.

3. The temperature from thermal field approach and the relative distance to target are concurrently considered to find an optimal path. A function combining both constraints, temperature and distance, has been used to calculate a weight of each available next hop, based on each neighbor’s distance to target and temperature. The message is then routed to the neighbor with the minimum weight. Each weight is calculated as a linear combination, where the coefficients control the influence of each summand on the total weight. The simulation results showed much improvement in terms of delivery ratio comparing to the shortest path. The best results are achieved by setting \(\alpha=0.9:\beta=0.1\). The success ratio is obviously high and the loss ratio is much decreased. At the same time, the total routing time is also reduce. However, the results of higher factor considering thermal field (\(\beta\)) show higher loss ratio and longer routing times correspondingly. That contrasts to multi-criteria routing algorithms by selecting criteria, for which the probability to select low-buffer path allows higher likelihood to forwarded through a very long path then to decrease average waiting time. The reason for this are the packets being forwarded into higher temperature and less distance when using the weighted function.

4. The difficulty of making a routing decision using a weight function is to find an optimal proportion between distance and temperature along the path, where the traffic situation is dynamic and different in each area. Thus, fuzzy logic has been chosen next to address the inherent uncertainties and non-linearity. The temperature and relative distances have been designed in a fuzzy set and fuzzy rules to find the best neighbor. The simulation results show that this approach provides the best routing performance compared to the previously discussed weighted function and single-criterion selection methods. Fuzzy logic shows the highest success ratio and decreased delay time on cost of only a bit higher number of hops. The overall routing times are better than those when using shortest
path, even though they are a bit higher than the best weighted function results.

On the hypothesis that dynamic structures and patterns of the overlay can also improve QoS routing in addition to the routing algorithms, the *Bridge* algorithm, a self-organizing overlay structure that adapts to the traffic in the system, was introduced. Kleinberg’s small-world network provides a limited routing time in 2D torus structure. However, such architecture is fixed and requires global knowledge to construct the system, which is difficult in distributed and decentralized networks. So, in this thesis, bridges or long-links are built and removed dynamically considering the traffic situation in the system. The thermal field approach is utilized to represent traffic conditions, and two agents cooperatively work for building bridges across the congested node or nodes. The experimental results show that the routing success ratio increases, as well as the average routing time decreases due to the reduction of delay time. Moreover, the level of nodes with high buffer usage decreases after bridges have been generated. Besides, the reduced average buffer utilization leads to a better load balance of the systems.

### 7.2 Other Contributions

Beside above key contributions, this thesis gives additional ideas for managing overlay systems as follows.

1. Kleinberg’s small-world has been extended by distributing long link information to surrounding area by using Ant Colony System. The Color Pheromone-based Landmarks algorithm has been presented. The messages are forwarded greedily as in the original model, but long-link information (source and target location) is distributed into the surrounding area as long as it is used. Thus, neighbors will have information about existing long links nearby, which helps to find shorter paths. The ants work as messengers to broadcast the existence of a long link in a TTL-limited radius area, while the pheromone color represents long links details. The pheromone amount degrades exponentially in time, so that unused links can be safely removed after some time, when no ants have visited them.

2. A network organization model with economic behavior is introduced aiming to study how the social networks (e.g. Facebook and Twitter) are formed. This network or system refers to a social community, where a node represents a customer, a provider, or just a person, while an edge symbolizes trading, a delivery method, dealing, or any other involvements, and a message or traffic in the network represents goods, services, or provided information. In the model, long-link are generated and deleted according to the character or strategy of a peer: it decides, when and to whom a relation is built, (e.g. because it seems suitable for investment) as well as which relation must be removed when players have no money. The scenarios analysed were a combination of three actions: *(i)* time to build a link, *(ii)* to whom to form a link and *(iii)* which link has to be removed. The experiment results show that the best strategy of network formation is played by a peer, who invests its
whole money to build a link to the partner to whom the most messages are targeted.

7.3 Outlook and future works

Finally, our ideas on some directions for the future research in overlay networking extended from contributions in this thesis are as follows:

**Routing Methodology in Overlay System**

The routing algorithms using thermal approach could be enhanced by an additional learning process and by dynamically adaptation depending on the real-time network situation. For example, neural networks could be used as an efficient method that is able to adapt fuzzy rules and fuzzy sets during traffic changes. In addition, more constraints such as bandwidth, latency or PageRank could be considered for improving quality of service routing. Moreover, the multi-criteria routing algorithms could be implemented in a self-organizing overlay system. Such an idea could be implemented and verified in real overlay systems like the *PlanetLab*.

**Economic and Behavior of Overlay Networks**

In order to analyze real social networks by the proposed network creation model, the node characters and decision strategies should be set according to human behaviors. The Limbic Character from marketing perspective is an example, which has already been applied in the communication network area and could be an interesting starting point for future investigations. In addition, a social community in reality is built not only from each individual’s character, but by cooperation among members. This aspect could also be analyzed, to see, how existing online social networks are and have been formed. As an extension of the proposed network creation idea, the collaboration among members could be analyzed in both centralized and decentralized manners. A centralized/hybrid type (kingdom) could have a single leader peer to construct and maintain shortcuts for a group of peers or for the whole system. In contrast, a decentralized approach (democracy) could use all information about peers in the group, which is shared, for finding the best solution to organize shortcuts for the whole group’s benefits.

**Security**

One of the key challenge in overlay networks is security. The thermal field approach could be applied to identify free-riding peer. The temperature field could represent the level of self-regulatory auditing and accounting behavior for resource sharing. Another important issues is *man-in-the-middle* attack. The thermal field approach could be applied in an application to uncover the “bad guys”. Improper behavior of any peer’s neighbor could be treated as heat, which could be distributed to other peers or used to inform a super node to enforce a required system policy.
Bibliography


