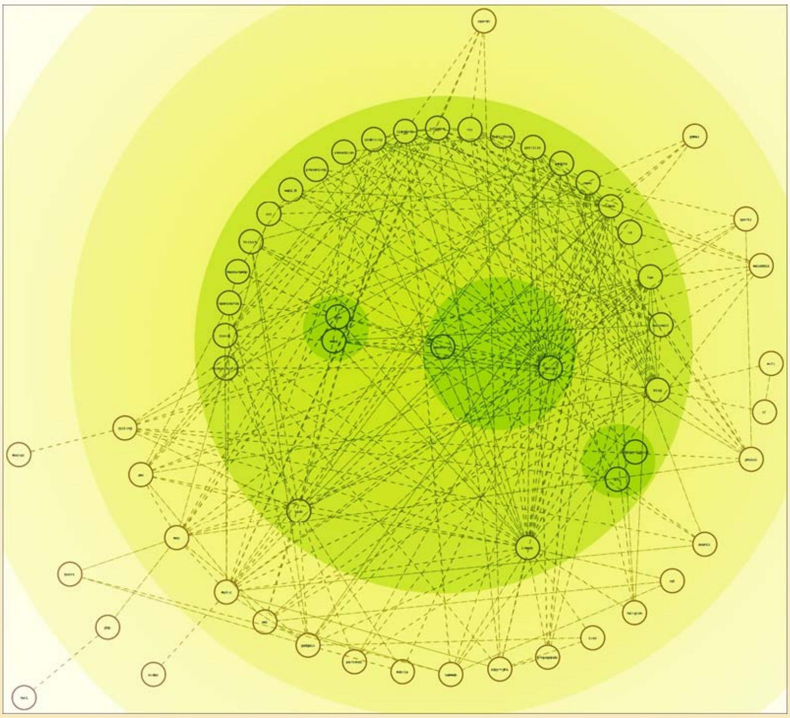
Information Propagation in Peer-to-Peer Networking

Modeling and Empirical Studies



Siyu Tang

Information Propagation in Peer-to-Peer Networking

Modeling and Empirical Studies

Cover: a graphic visualization of delicious tags and their intersections, by Kunal Anand (http://kunalanand.com/delicious).

Information Propagation in Peer-to-Peer Networking

Modeling and Empirical Studies

Proefschrift

ter verkrijging van de graad van doctor
aan de Technische Universiteit Delft,
op gezag van de Rector Magnificus Prof.ir. K.C.A.M. Luyben
voorzitter van het College voor Promoties,
in het openbaar te verdedigen op dinsdag 21 september 2010 om 12.30 uur

door

Siyu TANG

elektrotechnisch ingenieur
geboren te Xi'ning, Qinghai Province, China

Dit proefschrift is goedgekeurd door de promotor:
Prof.dr.ir. P.F.A. Van Mieghem

Samenstelling promotiecommissie:

Rector Magnificus,	Voorzitter
Prof.dr.ir. P.F.A. Van Mieghem,	Technische Universiteit Delft, promotor
Prof.dr.ir. I. Stavrakakis,	National and Kapodistrian University of Athens
Prof.dr.ir. M. Van Steen,	VU University
Dr.ir. J.S.H. van Gageldonk,	Philips Research Center
Prof.dr.ir. G.J. Houben,	Technische Universiteit Delft
Prof.dr.ir. M. Reinders,	Technische Universiteit Delft
Dr.ir. D. Epema,	Technische Universiteit Delft

ISBN 978-94-6113-018-1

This research was supported by the European Network of Excellence on content distribution (CONTENT) under project number IST-FP6-038423.

Keywords: information propagation, distributed/peer-to-peer networking, modeling, empirical studies

Printed in The Netherlands

To my parents, my husband and the kindness in everybody's heart

Contents

Summary

Thesis Title: Information Propagation in Peer-to-Peer Networking: Modeling and Empirical Studies

Although being a young technology, peer-to-peer (P2P) networking has spurred dramatic evolution on the Internet over the recent twenty years. Unlike traditional server-client mode, P2P networking applications are user-centric. Users (peers) generate their own content and share it with others across the Internet. Be it a P2P file-sharing network, a streaming delivery system, a video-on-demand application or an online social networking site, all of the aforementioned applications aim to fulfil a fundamental goal - that is, to deliver content to peers in a decentralized manner. In this thesis, we are motivated to study the information (or content) propagation process from the following two aspects:

1. To make use of existing techniques and propose models that are applicable in P2P networking.

2. To conduct empirical studies with emerging P2P applications regarding their methodologies of information propagation.

First of all, we study gossip-based information propagation in decentralized P2P overlay networks. We illustrate the difficulty of performing an exact analysis of gossip-based information dissemination in large-scale and dynamic P2P networks, where each peer only communicates with a subset of peers in the network. We show that, describing the gossip-based information propagation process in the aforementioned networks requires a very large state space, which is computationally not feasible. To guarantee the reliability of gossip-based information dissemination, we perform exact analytic modeling of the gossip-based information dissemination algorithms under the assumption of uniform neighbor selection over the entire distributed network. The model is extended to the case where random communication with multiple peers is allowed. We incorporate different network conditions and peer behaviors in the model. Important performance metrics and design parameters are also determined analytically. The proposed model is applicable for both content propagation and content searching in decentralized P2P networks. The derived metrics can be used to assess the coverage

and the effectiveness of content dissemination and search. We also study the content retrieval process provided that, m peers possessing the desired content are discovered. The effect of selecting a most nearby peer, which is assessed by hopcount and delay, among the group of m peers on P2P networking during content retrieval is analyzed. Our analysis answers the question of how many replicas of a particular content need to be distributed (or how many peers possessing the desired content need to be discovered), so that an acceptable quality of service (in terms of hopcount and delay) can be offered.

The gossip-based information propagation model discussed above conveys the basic idea of P2P networking. However, due to the rapid evolution of P2P networking techniques, applications with new features have been launched and user characteristics start to play an important role. Hence, we carry out two empirical studies that are designed to disclose important design issues and distinct user behaviors in some emerging P2P applications. Observations from the two empirical studies can be useful to develop models that are appropriate for the specific applications.

Our first empirical study focuses on a proprietary Peer-to-Peer Television (P2PTV) system named SopCast. The commercialized P2PTV applications have become the most dominant means of deliver video content via the Internet, while their underlying mechanisms are largely unknown. Consequently, we perform a set of experiments that are suitable to reflect the overall performance of the SopCast network. We dissect a part of the SopCast protocol by using a reverse engineering approach. Our analysis reveals the neighbor communication rule, the video delivery method and the network structure implemented in SopCast. The topological dynamics of the SopCast network, and its traffic impact on the Internet are also evaluated. The approach and methodology presented in this empirical work provide insights in the understanding of similar applications.

As mentioned earlier, the importance of users in P2P networking is emphasized more than any other networking applications. Thus, the second empirical study is conducted with an online social networking site, named Digg. The emerging online social networking applications are featured with collaborative information recommendation and propagation: users can publish, discover, and promote the most interesting content collectively without having a group of website editors. Everyday, a large amount of information is published on these sites, while only a few pieces of the information become popular. In this empirical analysis, we aim to answer the following questions: 1. Whether online social networking users are making friends with others that are similar as themselves? 2. What is the dynamic process that users are collaboratively filtering and propagating information in the online social networks? 3. Whether friendship relations are helping to propagate newly published content? Understanding different characteristics and the information propagation process in the online social networks helps to improve current marketing techniques that attempt to propagate advertisements, products, and ideas over these networks.

Chapter 1

Introduction

1.1 Peer-to-Peer: Collaborative Content Sharing and Delivery

The technology of *peer-to-peer* (P2P) networking creates a reciprocal environment where, by sharing resources and computational capacity with each other, mutual benefit between end-users is possible. The Internet, as originally conceived in the late 1960s, was in fact, a P2P system. The goal of the original ARPANET[1] was to connect and share computing resources between equal entities (peers). However, early applications of the Internet, e.g. File Transfer Protocol (FTP) and Telnet, were themselves server-client based. Thereafter, a centralized Internet is known to users. The role of the Internet, has not been reverted, until the year of 1999, when Napster [22] pioneered the idea of collaboratively sharing music at the end-users. Thereafter, P2P networking has spurred the implementation of different applications in the fields of P2P file-sharing, P2P streaming, P2P social networking, etc. Although the underlying mechanism of different P2P applications are different, they attempt to solve the same problem: that is, *to create a virtual overlay over the existing Internet (both wired and wireless) and to share and deliver content*[2] *collaboratively.*

By seeing the end-users as nodes, and their communication in-between as links, a *P2P overly network* is formed on the application layer. A node is also referred to as a *peer* in a P2P overlay network. A content (e.g. file, music, video, news, etc.), or part of the content, is normally replicated, and distributed over the P2P overlay network. A peer has the role of being a *server*, and a *client* at the same time. A peer can retrieve a desired content directly from its neighbor, whereas it is also responsible for providing a requested content to others, if it has it.

[1]Advanced Research Projects Agency Network

[2]In this thesis, content, message and information are interchangeable terminologies.

1.1.1 Classification of P2P Overlay Networks

Based on the methodology of overlay construction, we classify the P2P overlay networks into two types, namely, *structured* and *unstructured*.

Structured P2P Overlay Network

By structured, we mean that the P2P overlay network topology is tightly controlled and content are placed at precisely specified locations so that a search query can be efficiently routed to the peer with the desired content. Such systems essentially provide a mapping between the content (identified by a file identifier) and the location (identified by a unique peer address), using the *Distributed Hash Table* (DHT). Upon joining, a peer is assigned a node identifier. The content to be inserted, is also assigned a unique identifier, which is called the *key*. Keys are mapped to node identifier, and a $\{key, value\}$ pair is generated. Given a key, a store operation ($put\,\{key, value\}$), or a search operation ($value = get(key)$) can be invoked to store or retrieve the content corresponding to the key. Each peer maintains a small routing table consisting of the information about neighbor location, i.e. the node identifier. A search query is then forwarded across the overlay network to peers hop by hop, via the nodes whose identifiers are closer to the key, until it reaches the peer with the key. Typical examples are Content Addressable Network (CAN) [108], Chord [113], Tapestry [131], Pastry [109] and Kademlia [91]. The structured P2P overlay networks confront two major issues. First of all, they only support precise-match query. Secondly, they cannot cope with frequent peer joining and leaving. Thus, in most of today's P2P networks, the structured overlay is not used.

Unstructured P2P Overlay Network

An unstructured P2P overly network is ad-hoc in nature, meaning that the placement of content is unrelated to the overlay topology, and peers join the network without any prior knowledge of the topology. In fact, unstructured P2P overlay networks are more commonly employed, because such an architecture is easy to deploy, robust against single failures, and resilient to peer churns.

In a decentralized (or distributed) unstructured P2P overlay network, the overlay network is organized in a random way. The functionalities of content dissemination, search and retrieval are distributed to peers. The first decentralized unstructured P2P system, Gnutella [12], uses flooding to query content stored at peers. In some applications, a centralized location, e.g. a *super node* or a *tracker* is proposed to manage the contact information about peers and the content that they have, e.g. FastTrack/KaZaA [17], BitTorrent [5], and Tribler [106]. When joining the network, a peer receives a list of peers, i.e. a *peerlist*, from the super node or tracker. Even though a centralized location, i.e. the super node, may exist in the unstructured overlay, neighbor discovery and content delivery can still be performed in a decentralized manner. For example, in Tribler, a peer regularly communicates with the nodes in its peerlist to exchange neighbor information based on an epidemic protocol. By doing so, peers can gradually discover more neighbors in the network, and download content from them.

1.1.2 P2P Networking Applications

P2P networking techniques are mostly popularized in the following three fields: P2P file-sharing, multimedia streaming and P2P social networking.

P2P File-Sharing is the practice of distributing, downloading and storing content between peers. It is the earliest realization of P2P networking technology. Many P2P file-sharing systems are still very popular nowadays, e.g. KaZaA, Gnutella, BitTorrent, Tribler, Emule [9], etc. According to the Internet study report (2008/2009) [14], P2P file-sharing applications generate the most traffic on the Internet, i.e. 57% of the global Internet traffic.

P2P Multimedia Streaming underlines the delivery of video or audio content over the Internet. The P2P Voice Over Internet Protocol (VoIP) allows users to deliver voice communications over the Internet. With the video-on-demand (VoD) systems, users are free to watch video content on demand. Moreover, by using the Peer-to-Peer Television (P2PTV) applications, users can watch real-time video content online. Examples of academic P2PTV systems are SplitStream, which employs application layer multicast [49]; and CoolStream, which utilizes epidemic protocols to exchange video content [130]. There are also some P2PTV networks that are developed for commercial purposes, such as SopCast [31], PPStream [27] and UUSee [36]. These proprietary P2PTV networks have obtained increasing popularity in recent years, and have become a great success for video content delivery. Since these proprietary P2PTV systems are not open-source applications, their underlying mechanisms and architectures are unknown.

P2P Online Social Networking provides a place where users can interact with others over the Internet. Most online social networks (OSNs) are web based: there is a centralized website for users to log in, create profiles, send e-mails, instant messages (IMs) and publish content, etc. On the other hand, OSNs create virtual communities that allow users to network with friends, share ideas, interests, and information with each others in a decentralized and self-organized manner. In this sense, online social networking is also considered as a P2P application. In the last decade, OSNs have gained significant popularity from end-users (especially the young generations). They have greatly changed the way that people are socializing with each other and the way that they are consuming information via the Internet. For instance, recent statistics in

2008 have shown that teenagers and young adults (age between 13 and 24) spend about 30% of their social lives online [39]. According to a student survey performed by Hesel *et al.* in 2008 [71], 89% of the students at the university level use social networking sites, 70% visit a social network at least once a day; and 84% of them have a personal page or profile on one or more of the OSNs. Examples of popular OSNs are Facebook [10], LinkedIn [19], LiveJournal [20], Digg [8], MySpace [21], Twitter [35] and YouTube [37], etc.

1.2 Research Questions and Motivations

1.2.1 Two Aspects of Studying Information Propagation

The primary goal of P2P networking is to deliver content to peers in a decentralized way. The above goal, is achieved by three steps: disseminating[3] content; searching for content, and retrieving content. During content dissemination, the content, or the advertisement message indicating its availability and location is propagated. To search for content, a search query that is looking for the content itself or for the information about its location is spread. While for the case of content retrieval, the content itself is retrieved at the end-user. In all of the three cases mentioned above, information (e.g. content, controlling messages, search queries) are propagated over the P2P networking applications. The process, that the information is propagated (be it content dissemination, searching or retrieval) is the focus of this thesis.

For us, theoretical and empirical analysis are two inseparable components, because many networking theories were constructed based on empirical observations. For instance, modeling virus spread over the Internet is motivated by epidemic spreading in biology [56]; constructing the small-world model is inspired by the famous experiment performed by Stanley Milgram [93]. The theoretical models are used to describe the observed phenomena mathematically, and the empirical analysis assists to complement the mathematical models. As a young technology, the development of P2P networking requires both theoretical and empirical analysis.

Most P2P applications are man-made. Hence, when devising the content propagation algorithms, we can make use of existing techniques. For instance, the distributed hash table is employed to create structured P2P overlays. In the unstructured P2P networks, flooding, random walk, and gossip-based (or epidemic-based) algorithms are deployed to disseminate and retrieve content in a probabilistic manner. The aforementioned techniques have been studied for years, e.g. hash table as in the field of database indexing and caching [54], flooding as a technique in routing protocols [118], random walk as studied in graph theory [90], and gossip-based schemes as addressed in biology

[3]We use disseminate, propagate and spread interchangeably in this thesis.

and sociology. Therefore, it is convenient for us to learn from these well established theories, and propose models that can be applied to P2P networking.

On the other hand, in many of the real-world P2P applications, the underlying mechanism of propagating content is unknown yet. For instance, proprietary P2P streaming applications are very popular in recent years. However, little information has been released about their network structures, operational principles and video delivery methods. Another example is the emerging P2P online social networks that are self-organized and spontaneous in nature. These online social networks are user-centric. Thus, the importance of users are emphasized more than any other networking applications. In these applications, how do the social relationships and collaboration between users influence the information propagation procedure is still an open question. Without a good understanding of the content propagation process in real-world applications, developing mathematical modeling is not possible.

As a conclusion, we study the information propagation process in P2P networking from two aspects. The first aspect is to model, evaluate and predict the effectiveness of information propagation by using existing techniques. The second one is to understand the underlying mechanisms and the influence of user behaviors in real-world P2P applications, and to provide insights for further studies.

1.2.2 Motivation and Thesis Scope

In the course of this thesis, we first concentrate on gossip-based information propagation in unstructured P2P overlay networks. The process of selecting a best peer among the group of peers who possess the desired content is also addressed. Afterwards, we carry out two empirical studies. One focuses on a commercial P2PTV system named SopCast. The second one is conducted with the Digg online social network (OSN).

Theoretical Analysis of Information Propagation

Unstructured P2P overlay networks are widely used on today's Internet. We consider gossip-based information propagation schemes that emerge as an approach to maintain simple, scalable, and fast content dissemination and searching in distributed networks. Therefore, studying the gossip-based algorithms, that effectively disseminate the information, is the focus of this study. Once the content is discovered, the process of selecting the best peer to retrieve it becomes vital. Thus, we are motivated to study the peer selection problem during content retrieval.

In the theoretical analysis, we aim to accomplish the following issues: 1. Illustrate the difficulty of modeling gossip-based content propagation in large-scale, highly dynamic P2P networks (Chapter 2). 2. Develop a model that describes gossip-based content propagation and search under the assumption of uniform neighbor selection over the entire distributed network (Chapter 3). 3. Model the process of reaching the most nearby peer (in hopcount and delay) for content retrieval (Chapter 4).

Empirical Study of a Proprietary P2PTV Network

Extensive measurement studies have been performed to understand traffic patterns and topological dynamics of various P2PTV applications. One major limitation of the previous works is to reflect the overall performance of the P2PTV applications. Moreover, most of the popular P2PTV systems on today's Internet are commercialized, which means that their source codes are not available. Therefore, when studying the performance of such applications, we have to treat them as a black box. The goal of this case study is two-fold: first, to perform appropriate experiments so that the entire SopCast overlay can be examined; and second, to disclose the architecture and mechanisms implemented in the SopCast network.

We design a set of experiments, from which the SopCast protocol is dissected. The main objectives of this study are: 1. Disclose the network structure and video delivery pattern in SopCast (Chapter 7). 2. Reflect the topological dynamics of the SopCast overlay, and its traffic impact on the Internet (Chapter 8). This case study provides guidelines and methodologies to understand similar applications.

Empirical Study of an Online Social Networking Application

OSNs such as the Digg are gaining increasing attentions from the end-users, especially the young generations. Since users are more inclined to spend their social lives online, OSN becomes a promising platform for online campaigning [58], viral marketing [97] and targeted advertisements [128]. As shown in [128], in 2007, $1.2 billion was spent on advertisement in OSNs worldwide, and this amount is expected to triple by 2011. Therefore, we are motivated to study different characteristics of the OSNs (e.g. topological properties, user behaviors), and the way that users are spreading and sharing information in these networks. Understanding the fundamental characteristics of current OSNs and their underlying principal of disseminating content helps to improve existing marketing techniques that attempt to propagate advertisements, products, or ideas over these networks.

We developed a crawler, that is used to collect related information about the Digg OSN. By performing an extensive analysis on the collected data, two major aspects of the Digg network are examined: 1. Study the network structure in Digg, measure users' interests in the Digg network, and compare the similarity between friends (Chapter 11); 2. Examine the content propagation pattern and the effectiveness of friendship relations during the spread of content (Chapter 12).

1.3 Thesis Outline

The main body of this thesis consists of 13 chapters and is organized into three parts. The first part focuses on the theoretical modeling of content dissemination, search and retrieval. In the second and third part, we present two empirical studies performed with SopCast and Digg regarding their methodologies of information dissemination. Chapter 14 highlights the main conclusions of this thesis.

1.3.1 Part I: Modeling Content Propagation, Search and Retrieval

Chapter 2 defines a gossip-based information propagation problem in decentralized P2P overlay networks. Basic parameters, that are crucial when designing and modeling gossip-based algorithms are introduced. Gossip-based information propagation models, associated with the design parameters are reviewed. Furthermore, we illustrate the major challenge of analyzing gossip-based information dissemination in large-scale, dynamic P2P networks when peers only communicate with a subset of peers in the network.

Chapter 3 performs an exact analytic modeling of gossip-based message dissemination schemes under the assumption of uniform selection of multiple neighbors over the entire distributed network. Different network conditions and peer behaviors are also incorporated. The gossip-based algorithms under study are applicable for both content propagation and content searching in a distributed P2P overlay. Important performance metrics and design parameters are determined analytically. The performance of the gossiping-based content dissemination or search schemes are evaluated.

Chapter 4 addresses the problem of finding the most nearby peer from an initiating node. We use the metrics of hopcount and delay respectively to assess the closeness between peers. The analysis presented in this chapter answers the question of how many replicas of a particular content need to be distributed so that an acceptable quality of service can be offered.

Chapter 5 highlights the main conclusions of the theoretical analysis.

1.3.2 Part II: Empirical Study of a Proprietary P2PTV System

Chapter 6 provides an overview of the SopCast P2PTV system, and related measurement studies that were performed with the commercial P2PTV applications. After that, we specify the research challenges and highlight problems that we are going to solve.

Chapter 7 presents three experiments that have been designed to reflect the overall performance of SopCast. By using a reverse engineering approach, we have successfully dissected a part of the SopCast protocol. Our understanding of the neighbor communication pattern, the video delivery rule, and the three-tier network structure of SopCast is presented afterwards.

Chapter 8 aims to investigate the dynamic nature of the SopCast overlay. We first classify the SopCast overlay as a two-layer architecture consisting of a neighbor graph G_N and a video graph G_V. The activation period of a neighbor/video link that has been discovered in this chapter, is further used as proper snapshot duration of the dynamic SopCast overlay. Afterwards, the topological property (in terms of node degree) of the video and the neighbor graph are evaluated. We also explore the traffic dynamics of the SopCast network. We study how the SopCast traffic is dynamically distributed over different peers in the network.

Chapter 9 summarizes the observations and findings of the SopCast P2PTV network.

1.3.3 Part III: Empirical Study of an Online Social Network

Chapter 10 introduces the development of online social networking applications, and the Digg network. We describe related work that was performed with OSNs and propose research questions that will be studied. After that, the Digg data crawling process is explained. We also show some examples of the collected data files.

Chapter 11 discusses the topological properties (i.e. node degree, link symmetry and assortativity) of the Digg OSN. Thereafter, we study users' digging activities in Digg, and present our methodology of quantifying users' interests. We will show that friends in the Digg network are indeed sharing similar interests.

Chapter 12 aims to examine the information propagation pattern in Digg. We first show that the Digg network is an unbalanced system in terms of story submission. Furthermore, we study the collaborative content propagation in Digg, and discuss the impairment of Digg friendship relations during the spread of information.

Chapter 13 concludes the empirical analysis performed with the Digg network.

Part I

Modeling: Content Propagation, Search and Retrieval

Chapter 2

Gossip-based Information Propagation

In recent years, gossip-based (or commonly referred to as epidemic-based) algorithms, which mimic the spread of disease or rumor, have been considered as efficient and robust means for database maintenance and replication [57], information dissemination [61], topology construction [74], peer membership management [81], data aggregation [75] and failure detection [123]. It has also been implemented in many real-world applications: e.g. in Tribler [106], gossip-based algorithms are used to update and maintain peer information; in CoolStreaming [130], video content delivery is scheduled by using the gossip-based algorithms; in wireless ad-hoc networks [68], routing information is updated between neighbors in an epidemic manner. Consequently, gossip-based information propagation problem in decentralized unstructured P2P networks is the focus of this part.

We start this chapter by defining the gossip-based information propagation problem, and by introducing important parameters when designing gossip-based algorithms. Associated with the design parameters, we will review two existing models and illustrate the difficulty of analyzing gossip-based information dissemination in large-scale, dynamic P2P networks.

2.1 Problem Definition

First of all, we define what a *gossip-based information propagation problem* is. We consider a distributed network with $N+1$ peers, where a unique *identification number* (ID) i, $1 \leq i \leq N+1$, is assigned to each peer. In a gossip-based algorithm, communication between neighbors takes place periodically, which is commonly defined as *gossiping rounds* r. Each peer selects its neighbor(s) or *gossiping target(s)* to forward a piece of new information according to certain *peer selection schemes*. Neighbor communication is delivered over a connecting physical or virtual link. The node that initiates the message dissemination process is referred to as the *initiator*, which is assumed to be the only informed node at the beginning of the process. Any node that receives the message will become an *informed* one. Otherwise, it is referred to as being *uninformed*. The number of informed nodes, in round r, is denoted by X_r. The number of uninformed nodes is then, denoted by $U_r = N + 1 - X_r$. A node can become uninformed if it removes this message later on. Peers participating in the gossiping process may behave differently with respect to different properties, e.g. their willingness of participating in the gossiping process, bandwidth/storage capacity, or computational power, etc. The different behaviors of peers are referred to as their their *degree of cooperation*. The overall objective is to disseminate the message as fast as possible, so that every node in the network is aware about the message.

2.2 Important Parameters

When designing or studying gossip-based information propagation algorithms, we need to consider several important parameters.

2.2.1 Peer View of the Network

A random node i in the distributed network (with $N + 1$ nodes) maintains a list of m peers ($1 \leq m \leq N$), which is referred to as a *peerlist* L_i, to communicate with. The dimension, $l_i = dim\,(L_i)$, of the vector L_i is defined as the size of peerlist L_i. Usually, a node i is not allowed to appear in its own peerlist L_i, meaning that $i \notin L_i$. If the peerlist contains all the other peers in the network, i.e. $l_i = N$, we say that the peer has a *complete view* of the network. If a peer i only knows a subset of peers in the network, meaning that $1 \leq l_i < N$, the peer is then said to have a *partial view* of the network.

Peer Complete View In the early study of gossip-based information dissemination algorithms, it is assumed that every peer knows all the other peers: that is, each peer has a complete view of the network ($l_i = N$). This assumption applies in a distributed network with moderate network size, e.g. hundreds of nodes. A complete view of peers, however, is not a realistic assumption in large-scale distribution networks. Because distribution systems such as P2P networks and ad-hoc networks are featured with frequent peer joinings and departures. Thus, it is difficult to update the complete node membership in a highly dynamic system. Moreover, maintaining a complete view of peers at every node in the network incurs extra overload by frequently exchanging peer information.

Peer Partial View To design a scalable gossip-based information dissemination algorithm in large-scale distributed networks, the partial view of peers is taken into account. In this case, a random peer i only disseminates the information to a subset of peers in the system, i.e. $1 \leq l_i < N$. In order to guarantee the reliability of gossip-based information dissemination, and to cope with peer dynamics, the view of a peer needs to be periodically updated, according to some peerlist exchange schemes. By periodically exchanging peerlists, the freshness (in terms of the age and availability) of peers can be updated. A detailed description of different peerlist exchange schemes can be found in [76].

2.2.2 Uniformity of Peer Selection

We assume that all peers in the network are equally interested in the content to be disseminated. To guarantee the reliability of gossip-based information dissemination, it is preferred to achieve *uniformity*[1] during the neighbor selection (see [61]), so that every node in the network can be notified by the information to be disseminated. Uniform neighbor selection can be easily satisfied when peers have a complete view of the network. When only partial view is available, uniformity can also be achieved as discussed by Jelasity *et al.* [76] and Eugster *et al.* [60]. Hence, designing appropriate peerlist exchange schemes, and properly selecting peers from the local views may also ensure random peer selection in large-scale distributed networks.

2.2.3 Peer Infectious Model

Inherited from biology, a peer can be *uninformed* (susceptible as in biology), if it has not received the information that is to be propagated yet; *informed* (infected as in biology), if it has received the information; and *recovered*, if it deletes the information afterwards. Based on the transition between statuses, the three most studied models are: *SI* (susceptible-infected), *SIS* (susceptible-infected-susceptible), *SIR* (susceptible-infected-recovered). In networking terms, the SI model (also called the *informed-forever* model) means that once an uninformed peer received the information, it becomes informed and remains informed forever. The informed peers help to propagate the information. In the SIS model, an informed peer can delete the information after being informed for some time. Thus, the peer will transit from the informed state to uninformed state again. The peer that stays in the uninformed state may receive the same information again and help to propagate it later on. In the SIR model, after being informed for some time, a peer can decide to remove the information, and refuse to propagate the same information any more.

[1]By *uniformity*, we refer to the case that peers can choose their neighbors uniformly from the entire network.

2.3 Gossip-based Information Propagation Models

In the following, we review several gossip-based information propagation models associated with the aforementioned, two most important assumptions: complete and partial view of the network.

2.3.1 Theoretical Model - Peer Complete View

An early study of the information dissemination problem is found in [104], in which a rumor spreading problem is discussed over a group of $N+1$ people. In each round, every informed person passes on the information to a single neighbors, selected randomly and independently of all its previous choices and of all the choices of the other $N+1$ people. A person may choose itself as gossiping target. Pittel [104] has derived the exact expression for the transition probabilities of the process $\{X_r, r \geqslant 1\}$ as follows

$$\Pr[X_{r+1} = j | X_r = i] = \begin{cases} \frac{\binom{N+1-i}{j-i}}{(N+1)^i} \sum_{t=0}^{j-i} (-1)^t \binom{j-i}{t} (j-t)^i & \text{if } j - i \geq 0 \\ 0 & \text{if } j - i < 0 \end{cases} \tag{2.1}$$

where i is the number of informed nodes in round r, and j is the number of informed nodes in round $r+1$. The propagation speed, in terms of gossiping rounds, is bounded by $O(\log(N+1))$ until everybody is informed.

The assumption of uniform neighbor selection has led to rigorous analysis of gossip-based algorithms in distributed P2P overlays, and has been studied in many theoretical papers: e.g. Demers *et al.* [57] for database maintenance, Birman *et al.* [48] for reliable multicasting, Karp *et al.* [79] in the case of information dissemination, Kempe [80] and Jelasity *et al.* [75] regarding gossip-based information aggregation.

2.3.2 Theoretical Model - Fixed Peer Partial View

By using the peerlist L_i maintained at each peer i, an adjacency matrix A can be created correspondingly. The element a_{ij} in the adjacency matrix is

$$a_{ij} = 1_{j \in L_i} \tag{2.2}$$

where the indicator function $1_{j \in L_i}$ is one if $j \in L_i$ is true and otherwise it is zero. An adjacency matrix A characterizes a graph $G(N+1, L)$ with $N+1$ nodes and L links. Since a peer i does not need to be in the peerlist of node j, if $j \in L_i$, the adjacency matrix A is generally not symmetric.

Van Mieghem *et al.* in [120], studied an exact continuous-time model for virus spread in a static network, in which each node has two states: susceptible and infected. The model in [120] considers the virus spread in an undirected graph characterized by a symmetric adjacency matrix. The epidemic threshold, which is associated with the largest eigenvalue of the matrix A is also rigorously defined.

Considering a discrete stochastic process which takes place in rounds, the description of the exact model in [120] can be rephrased in terms of content propagation, in which peers have a fixed partial view of the network. The graph constructed by the peerlists is assumed to be a connected, undirected graph. At each gossiping round r, a peer i enters two states: informed, denoted by $X_i(r) = 1$, or uninformed, denoted by $X_i(r) = 0$. The state of the stochastic process is the set of all possible combinations of the states in which the $N+1$ peers can be at round r. The number of the states with k informed nodes is $\binom{N+1}{k}$. Thus, the total number of states is $\sum_{k=0}^{N+1}\binom{N+1}{k} = 2^{N+1}$. The state $Y(r)$ of the network at round r is thus expressed as

$$Y(r) = [Y_0(r) \ Y_1(r) \ ... \ Y_{2^{N+1}-1}(r)]^T$$

where

$$Y_i(r) = \begin{cases} 1 & \text{if } i = \sum_{k=1}^{N+1} X_k(r)\, 2^{k-1} \\ 0 & \text{if } i \neq \sum_{k=1}^{N+1} X_k(r)\, 2^{k-1} \end{cases}$$

Thus, the state space of the MC is organized with $x_k \in \{0, 1\}$ as

State index i	$x_{N+1}(r)x_N(r)...x_2(r)x_1(r)$
0	00......0000
1	00......0001
2	00......0010
3	00......0011
......	......
$2^{N+1}-1$	11......1111

2.3.3 Theoretical Model - Dynamic Peer Partial View

As mentioned earlier, the distributed P2P networks nowadays are large-scale and highly dynamic. Therefore, peers may only obtain a partial view of the network, and they have to update their views periodically with other peers to ensure reliability during information dissemination. Hence, assuming a complete or a static partial view of the network is not realistic. Consequently, many previous work focused on proposing gossip-based algorithms that are applicable in large-scale, and dynamic P2P networks.

For instance, Eugster *et al.* [60] and Ganesh *et al.* [64] evaluated the performance of gossip-based algorithms with dynamic peer partial views during information dissemination, respectively. Kermarrec *et al.* [81] related the reliability of information dissemination to several system parameters, e.g. system size, failure rates, and number of gossip targets. The influence of different network topologies in disseminating information can be found in [63]. However, to our knowledge, none of the previous papers attempted to model the information dissemination process and to calculate the speed of information dissemination rigorously.

In the sequel, we will answer the question of *how difficult it is to describe the gossip-based information dissemination process with dynamic peer partial views* in distributed P2P networks. We first present an example to demonstrate the major factors need to be taken into account when defining the states of the system.

Among the many different peerlist exchange algorithms described in [76], we assume that, when a peer i selects a target node j to gossip with, a union of the two peerlists L_i and L_j is made. The old peerlist of L_i and L_j is updated with the new one in the next round as $L_i(r+1) = \{L_i(r) \cup L_j(r)\} \setminus \{i\}$ and $L_j(r+1) = \{L_i(r) \cup L_j(r)\} \setminus \{j\}$, so that $i \notin L_i$, and $j \notin L_j$. In Fig. 2.1, we present an example of a network with 4 nodes, by making union of two peerlists. Each neighbor selection is performed uniformly from the peerlist. We will show that the state of the network is not only related to the combination of the informed nodes, but also depends on the possible combinations of the peerlists.

Assuming that during initialization[2], we assign each peer a peerlist with two neighbors ($l_i = 2$). The peerlists for different peers at $r = 0$ are $L_1(0) = \{2, 3\}$, $L_2(0) = \{3, 4\}$, $L_3(0) = \{1, 4\}$, and $L_4(0) = \{1, 2\}$, respectively. In Fig. 2.1, we draw a *sample path*, i.e. the realization of the gossiping process in consecutive rounds, when peers start to exchange their peerlists. At each gossiping round, a random peer i can be either informed (which is denoted by $X_i(r) = 1$), or uninformed (which is denoted by $X_i(r) = 0$). Initially ($r = 0$), all of the 4 nodes are uninformed. At the first round ($r = 1$), peer 4 starts to disseminate a piece of information. At the next round of $r = 2$, there are two possible states in the network. If peer 4 selects peer 1, the system moves to the state with the combination of informed nodes 1001 and the updated peerlists of $L_1(2) = \{2, 3\}$, $L_2(2) = \{3, 4\}$, $L_3(2) = \{1, 4\}$, and $L_4(2) = \{1, 2, 3\}$. If peer 4 chooses peer 2, the system transits to the state with the combination of informed nodes 1010 with the peerlists of $L_1(2) = \{2, 3\}$, $L_2(2) = \{1, 3, 4\}$, $L_3(2) = \{1, 4\}$, and $L_4(2) = \{1, 2, 3\}$. At the third round, we only present the possible transitions from state 1001 (with the corresponding peerlists). As shown inside the dotted diagram of Fig. 2.1, state 1001 can move to 7 states, which are marked from 1 to 7. For instance, if peer 1 chooses peer 3, and peer 4 chooses peer 3, the system will move from state 1001 (with the corresponding peerlists) to state 1101, with the peerlists of $L_1(3) = \{2, 3, 4\}$, $L_2(3) = \{3, 4\}$, $L_3(3) = \{1, 2, 4\}$, and $L_4(3) = \{1, 2, 3\}$. Although both states 1 and 2 in Fig. 2.1 (round 3) have the same combination of informed nodes, namely 1101, the peerlists of the individual peers are different, implying that the combination of peerlists should also be taken into account when defining the states of the system. The same observation holds for state 3, 4 and 5 (corresponding to 1011) and 6 and 7 (corresponding

[2] The *initialization* is defined as the period in which peers join the network, and receive their peerlists.

to 1111).

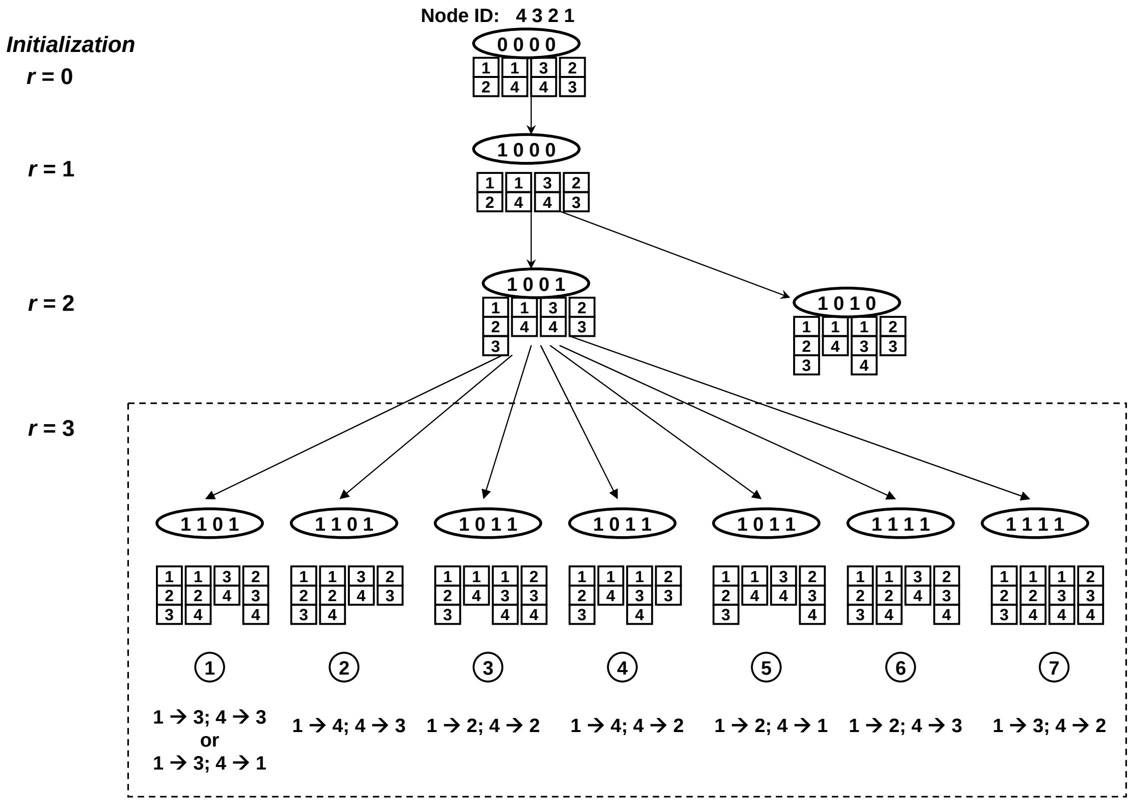


Figure 2.1: Demonstration of a sample path of the gossip-based information propagation process in a network with 4 nodes. Peerlists are updated after making the union of two peerlists, with $i \notin L_i$. The block under each peer represents its peerlist. The last row in the dashed box specifies the transitions between two states. For instance, in the third round, $1 \to 3$; $4 \to 3$ means that peer 1 selects peer 3, and peer 4 selects peer 3. The operation is explained in detail in the text.

Hence, to describe the gossiping process exactly, the following steps are needed.

Step 1: Describe all possible combinations of informed nodes in the system. As we have introduced in Section 2.3.2, there are 2^{N+1} combinations of the informed nodes.

Step 2: Describe all possible combinations of the peerlists. For simplicity, we remove the constraint of $i \notin L_i$, such that a peer i is allowed to appear in its own peerlist. We also assume that there is no size limitation on the peerlist, meaning that the peerlist size l ranges from 0 to $N+1$ (we allow empty peerlist to simplify the calculation). Since the number of peerlists with k $(0 \le k \le N+1)$ peers is $\binom{N+1}{k}$, there are in total, $\sum_{k=0}^{N+1}\binom{N+1}{k} = 2^{N+1}$ different peerlists[3].

[3]If an empty peerlist is not allowed, the total number of the combinations of the peerlists is $2^{N+1}-1$.

Step 3: At each round, every peer in the network may possess one of the peerlists out of the 2^{N+1} ones. Therefore, given $N+1$ peers in the network (regardless of their status of being informed or uninformed), the total number of the combinations of their peerlists is $\left(2^{N+1}\right)^{N+1} = 2^{(N+1)^2}$. Recall that the state of the network is also decided by the possible combinations of the informed node in the network. Given 2^{N+1} combinations of informed nodes in the system, as we have discussed already, the total number of states used to describe the entire system exactly is $2^{(N+1)^2} \times 2^{N+1} = 2^{(N+1)^2+N+1}$. Hence, to organize and to index the enormous number of states, we need to take both the different informed nodes, and the different peerlists into account, which is challenging. Moreover, such a large state space is also computationally not feasible[4].

As a conclusion, the above analysis provides an upper bound on the total number of states that are needed to describe the information propagation process with peers' dynamic partial views. The exact number of states depends on the initial conditions (e.g. initialization of the peerlists) and the dynamics of the dissemination process (e.g. the peerlist exchange scheme). Nevertheless, an exact analysis of the problem, in which peers are constantly exchanging peerlists, is difficult. To evaluate the performance of propagating information in such a scenario, we suggest to perform simulations, or devise proper approximations.

[4] As shown in [120], the matrix computations (on a PC) of a 2^N model are already limited to $N = 13$.

Chapter 3

Content Dissemination and Search with Uniform Gossiping

Uniform neighbor selection over the entire network guarantees the reliability of gossip-based information propagation. The exact analysis of modeling gossip-based information dissemination when peers have dynamic, partial views is not feasible, see Section 2.3.3. When peers have a complete view of the network, the uniformity can be easily satisfied. In the case where peers have partial views, we assume that the uniformity can be achieve by employing appropriate peerlist exchange schemes, and by properly selecting peers from the local views. However, the design of such peerlist exchange schemes, and of peer selection methodologies is out of the scope of this thesis.

In this chapter, we focus on gossip-based information dissemination with uniform gossiping. The model is extended to the case where random communication with multiple peers is allowed. We also complement the model by considering different network conditions, depending on the knowledge of the state of the neighboring nodes in the network. Different node behaviors, with respect to their degree of cooperation and compliance with the gossiping process, are incorporated as well.

The gossip-based algorithms discussed in this chapter are applicable for both content dissemination and content searching in a decentralized environment. In both cases, a message needs to be disseminated. For content dissemination, this message contains either the content itself, or the advertisement information about the content. During content searching, the message to be disseminated is a search query looking for the content. From the exact analysis, important performance metrics and design parameters are analytically determined. Based on the proposed metrics and parameters, the performance of the gossiping-based content dissemination or search schemes, as well as the impact of the design parameters, are evaluated.

3.1 Algorithm Description

We consider a network with $N+1$ peers. In the first round ($r = 1$), the initiator selects randomly k neighbors, $1 \leq k < N$, to forward the message to. Any node that receives the message will become an informed one, and remain informed thereafter. In each round, all the informed nodes select k gossiping targets randomly and independently to forward the message to. Since this is an informed-forever model, the number of informed nodes X_r, in round r, is non-decreasing with r. Similarly, the process of the number of uninformed nodes U_r by round r, is non-increasing.

The self-concerned nature of the peers is also captured, by incorporating the notion of cooperation. A node in the network, participating in the gossiping process as expected, is classified as *cooperative*. Such nodes always accept messages forwarded to them, become informed and forward the message to others according to the rules. If a node is not cooperative, it is referred to as *non-cooperative*. The non-cooperative nodes are presented in social and P2P networks as a consequence of resource-preservation concern or simply selfish attitude of the peers. In this thesis, the level of cooperation in the network will be captured by the *cooperation probability* β, $0 < \beta \leq 1$, associated with each node. Nodes with cooperation probability $\beta = 1$ are always cooperative. Nodes with $\beta = 0$ are in essence not part of the network and this degenerate case is not considered. The following assumptions are made regarding β to facilitate the analysis: (1) β is time-invariant and common to all nodes; (2) Once a node decides to be cooperative (or non-cooperative), it is cooperative (or non-cooperative) to all nodes that select it in the same round. (3) In each round, a node decides to be cooperative or non-cooperative independently of its choices in previous rounds and of the choices of others. Once a node decides to be cooperative, it participates in the dissemination until the end of the gossiping process.

Many variants of gossip-based algorithms exist based on various criteria and levels of information availability. In this thesis, we study two fundamental cases which are distinguished by the policy of choosing gossiping targets, namely, the *blind* and *smart gossiping-target* selection schemes (in short, the blind and smart selection schemes). We describe the two schemes in more details in the sequel. Practical issue such as the maintenance of gossiping history is not the focal point of this thesis.

3.1.1 The Blind Gossiping-Target Selection Scheme

Under this scheme, no information about the status (informed or not informed) of the neighbors is available. The k gossiping targets are selected randomly from the N neighbors. This scheme is thus referred to as blind gossiping-target selection, and is illustrated in Fig. 3.1. In round $r = 1$, all $k = 2$ gossiping targets cooperate. In round $r = 2$, all the three informed nodes (1, 2 and 3) select each other as gossiping targets. Thus, the number of informed nodes remains the same ($X_2 = X_3 = 3$). In round

$r = 3$, all the three informed nodes select different gossiping targets, in which nodes 4, 5, 6, 7 are uninformed nodes, and node 2, 3 are informed ones. Since nodes 5 and 6 decide to be non-cooperative in this round ($1 - \beta$ labelled in the corresponding link), the number of informed nodes in round $r = 4$ is $X_4 = 5$. The blind gossiping-target selection scheme models a network with anonymous peers, or the case in which nodes do not keep log files with all the neighbors that they have contacted. We consider the blind gossiping-target selection scheme as the worst case because repetitious selection of gossiping targets may slow down the speed of information dissemination.

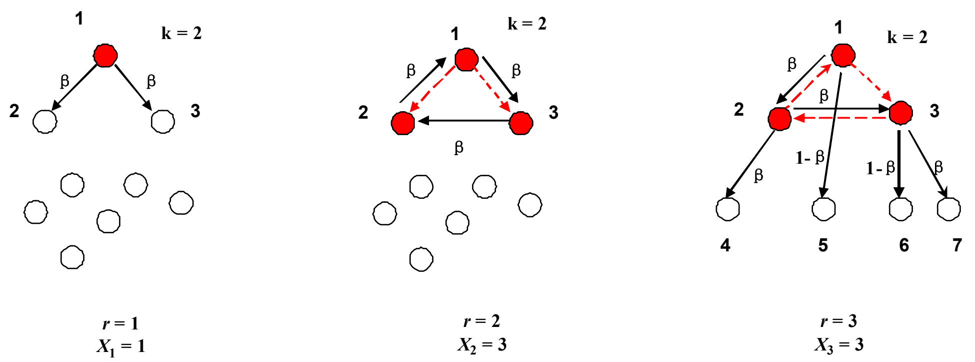


Figure 3.1: Illustration of the blind gossiping-target selection scheme with $k = 2$ and $0 < \beta \leq 1$. The shaded circle indicates an informed node. A dotted line between two nodes indicates communication between them in the previous round.

3.1.2 The Smart Gossiping-Target Selection Scheme

In the smart gossiping-target selection scheme, it is assumed that the nodes know the identity of their neighbors, and have the complete information about their status, in terms of being or not being informed about the message under dissemination. Such information is piggybacked on the periodically exchanged control packets, as part of the standard neighborhood discovery process. In this way, the knowledge about node status are provided to the neighboring nodes, so that a node can avoid sending the same message to the nodes that already knew it. The smart selection leads to a faster message dissemination compared to the blind one, as already informed nodes are avoided, as shown in Fig. 3.2. If $N+1-X_r < k$, every node will be informed, meaning $X_{r+1} = N+1$ in the next round, because it is sufficient that an informed node chooses less than k targets.

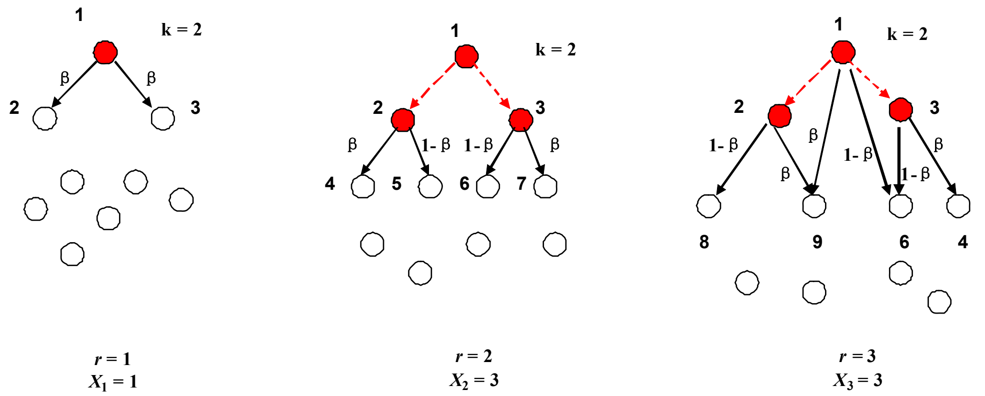


Figure 3.2: Smart gossiping-target selection with $k = 2$ and $0 < \beta \leq 1$. The gossiping targets are randomly selected among the set of uninformed neighbors.

3.2 Exact Analysis of Gossip-Based Message Dissemination

Under the assumption of random neighbor selection over the complete $N + 1$ nodes in the network, the process of the number of the informed nodes at the beginning of round r, $\{X_r,\ r \geqslant 1\}$ can be modeled rigorously as a discrete Markov chain (MC) with state space $S = \{1, 2, ...N + 1\}$. Let P denote the $(N + 1) \times (N + 1)$ transition probability matrix. Each entry in P, $P_{ij} = \Pr[X_{r+1} = j|X_r = i]$, denotes the probability that the MC moves from state i to state j in one round. We denote the *probability state vector* $s[r]$ in round r by $s[r] = [s_1[r],\ s_2[r]\ ...\ s_{N+1}[r]]$, where $s_i[r] = \Pr[X_r = i]$. Clearly, the initial probability state vector is $s[0] = [1,\ 0,\ 0,\ 0,\ ...,\ 0]$.

The number of informed nodes after every round never decreases, and thus $X_{r+1} \geqslant X_r$, such that the transition probability matrix P is an upper triangular matrix, with all zeros in the lower triangular part of P. The $(N + 1)-$state MC has an *absorbing state*[1] because the network never leaves state $N + 1$ when all the nodes are informed. The steady state vector is just the absorbing state of $\pi = [0\ 0\ 0\ ...\ 1]$. In this triangular matrix P, the diagonal entries are the corresponding eigenvalues of P. The diagonal element on the last row is $P_{N+1,N+1} = 1$, which is the absorbing state.

In the sequel, the state transition probabilities are derived by employing a combinatorial approach. This approach is inspired by the occupancy problem in the balls and bins model introduced in Appendix A, when the informed nodes are balls, and the gossiping targets are bins.

[1]An *absorbing state* i is a recurrent state with the probability of returning to state i as $P_{ii} = 1$.

3.2.1 The Transition Probabilities P_{ij} of the Blind Gossiping-Target Selection Scheme

Under this scheme, a node chooses its gossiping-target randomly from the N neighbors in the network (excluding itself). The transition probabilities can be calculated by applying the balls and bins model as introduced in Appendix A.2.

1. Cooperative nodes with $\beta = 1$

In order for the MC to move from state i to state j, $z = j - i$ new nodes will need to be selected by the i informed ones, after the current round. Under the blind selection algorithm, each of the i informed nodes selects k different neighbors from the set of $j-1$ nodes (a node is not allowed to choose itself as target) blindly. The z new nodes should be selected at least once, by the i informed nodes. Otherwise the Markov process cannot arrive at state j.

Determining P_{ij} is analogous to finding the probability of randomly placing r groups of k balls to $n-1$ bins (colored in red and white), with at least $m = z$ red bins being occupied, as described in Appendix A.2. The operation of the i informed nodes, selecting k different neighbors from the set of $j-1$ nodes, is equivalent to placing the r groups of k balls to the $n-1$ bins, excluding the white bin that has the same numbering as the group of balls. Selecting the z new nodes is analogous to the placement of balls to the m red bins. Gossiping-target selection from the set of i informed ones is analogous to placing the balls to the $n-m$ white bins. Finally, the z new nodes are selected at least once, which is equivalent to requiring that at least the m red bins are occupied. The transition probabilities of P_{ij} are derived by substituting $m = z$, $n = j$, $r = i$ in (A.6)

$$P_{ij} = \begin{cases} \dfrac{\binom{N+1-i}{z}}{\binom{N}{k}^i} \displaystyle\sum_{t=0}^{z} (-1)^t \binom{z}{t} \binom{j-1-t}{k}^i & \text{if } i-1 \geqslant k \text{ and } i \leqslant j \leqslant \min\{N+1, i(k+1)\} \\ \dfrac{\binom{N+1-i}{z}}{\binom{N}{k}^i} \displaystyle\sum_{t=0}^{j-1-k} (-1)^t \binom{z}{t} \binom{j-1-t}{k}^i & \text{if } i-1 < k \text{ and } k+1 \leqslant j \leqslant \min\{N+1, i(k+1)\} \\ 0 & \text{otherwise} \end{cases} \tag{3.1}$$

where $\binom{N+1-i}{z}$ is the number of ways to choose z new nodes among the set of $N+1-i$ uninformed nodes at state i, and $\binom{N}{k}^i$ is the total number of ways that i nodes can choose k different neighbors.

The non-zero elements in P are discussed by treating the relation of $i-1$ and k properly. The first confinement of $i-1 \geqslant k$ or $i-1 < k$ specifies a similar conditioning as $n-1-m \geq k$ or $n-1-m < k$ in (A.6). The second confinement of $i \leqslant j \leqslant \min\{N+1, i(k+1)\}$ or $k+1 \leqslant j \leqslant \min\{N+1, i(k+1)\}$ defines the minimum and maximum number of informed nodes that appears in state j.

- When $i-1 \geqslant k$, it is possible that each informed node selects its k targets from the set of i nodes that are already informed. The Markov process remains in state i, indicating the minimum boundary of $j=i$. If the i nodes select their neighbors differently from the set of $N+1-i$ uninformed ones, there will be maximally $i(k+1)$ informed ones in the next round. Notice that $i(k+1)$ can never be larger than the total number of nodes in the network, $j=\min\{N+1, i(k+1)\}$ serves as the upper boundary.

- In case of $i-1<k$, $k-(i-1)$ uninformed nodes have to be selected so that an informed node can choose k different neighbors successfully. The minimum value of j is thus, bounded by $k+1$. The upper boundary of $j \leqslant \min\{N+1, i(k+1)\}$ holds as described above.

Under the blind selection algorithm, it is assumed that neighbor selection is performed from the rest of the N neighbors in the network. A variation of the blind selection scheme is to select k different neighbors out of the $N+1$ nodes, which is considered as an extended setting of the rumor spreading problem in [104]. We present the mathematical analysis for the extended rumor spreading problem in Appendix A.3.

2. Non-cooperative nodes with $0<\beta<1$

Under this case, not all selected new nodes may decide to cooperate. Consequently, if out of the assumed $z=s-i$ new selected nodes, exactly $j-i$ of them decide to cooperate, a state transition from i to j will occur. Let $B(z,\ j-i,\beta)$ denotes the probability that there are exactly $j-i$ cooperative nodes out of the z new ones, given by

$$B(z,\ j-i,\beta) = \binom{z}{j-i}\beta^{j-i}(1-\beta)^{z-(j-i)} \tag{3.2}$$

with $0 \leq j-i \leq z$.

By properly invoking (3.1) and (3.2), P_{ij} is derived for the general case of $0<\beta<1$ as

$$P_{ij} = \begin{cases} \sum_{s=j}^{\delta} \frac{\binom{N+1-i}{z}}{\binom{N}{k}^i} \sum_{t=0}^{z} (-1)^t \binom{z}{t} \binom{s-1-t}{k}^i B(z,\ j-i,\beta) & \text{if } i-1 \geqslant k \\ & \text{and} \\ & i \leq j \leq \min\{N+1, i(k+1)\} \\ \sum_{s=k+1}^{\delta} \frac{\binom{N+1-i}{z}}{\binom{N}{k}^i} \sum_{t=0}^{s-1-k} (-1)^t \binom{z}{t} \binom{s-1-t}{k}^i B(z,\ j-i,\beta) & \text{if } i-1<k \\ & \text{and} \\ & i \leq j \leq \min\{N+1, i(k+1)\} \\ 0 & \text{otherwise} \end{cases} \tag{3.3}$$

where $\delta = \min\{N+1, i(k+1)\}$.

3.2.2 The Transition Probabilities P_{ij} of the Smart Gossiping-Target Selection Scheme

Given i informed nodes in the network, and that each of them selects k different neighbors from the remaining $N+1-i$ uninformed ones, the problem is analogous to the balls and bins model described in Appendix A.1, with the balls being the i informed nodes and the bins being the $N+1-i$ uninformed nodes.

1. Cooperative nodes with $\beta = 1$

Under this scheme, the transition probabilities can be derived by applying (A.3), substituting $r = i$, $n = N+1-i$, $m = N+1-j$, and $n-m = z$, where z denotes the number of new nodes selected by the i informed ones. Thus, we have

$$P_{ij} = \begin{cases} p_{N+1-j}(i, N+1-i, k) & \text{if } N+1-i \geqslant k \text{ and } i+k \leq j \leq \min\{N+1, i(k+1)\} \\ 1 & \text{if } N+1-i < k \text{ and } j = N+1 \\ 0 & \text{otherwise} \end{cases} \tag{3.4}$$

where $j = i + z$ and

$$p_{N+1-j}(i, N+1-i, k) = \frac{\binom{N+1-i}{N+1-j}}{\binom{N+1-i}{k}^i} \sum_{t=0}^{z-k} (-1)^t \binom{z}{t} \binom{z-t}{k}^i \tag{3.5}$$

Notice that (3.5) is valid only for $N+1-i \geqslant k$. When $N+1-i < k$, the entire network is informed with probability 1. The conditioning of $i+k \leq j \leq i(k+1)$ defines the minimum and maximum number of informed nodes that appears in state j. If all the i informed nodes choose the same k neighbors, there are minimally, $i+k$ informed nodes at the next state. In case that all the informed nodes choose their neighbors differently, the number of informed nodes at state j is bounded by $\min\{N+1, i(k+1)\}$.

2. Non-cooperative nodes with $0 < \beta < 1$

If s denotes the number of informed nodes at the next round, P_{is} is computed from (3.4). Out of the z newly chosen nodes, there should be $j-i$ cooperative nodes so that the process arrives at state j. The probability that $j-i$ out of the z nodes are cooperative is computed from (3.2) with $0 \leq j-i \leq z$. Consequently, the transition probabilities of P_{ij} are given by

$$P_{ij}=\begin{cases} \sum_{s=i+k}^{\min\{N+1,i(k+1)\}} p_{N+1-s}(i,N+1-i,k)B(z,j-i,\beta) & \text{if } N+1-i \geqslant k \\ & \text{and} \\ & i \leq j \leq \min\{N+1, i(k+1)\} \\ B(N+1-i,j-i,\beta) & \text{if } N+1-i < k \\ 0 & \text{otherwise} \end{cases} \tag{3.6}$$

in which $s = i + z$.

3.3 Performance Evaluation

The probability state vector $s[r]$ can be calculated in terms of the initial probability state vector $s[0]$ and the matrix P from

$$s[r] = s[0]P^r \tag{3.7}$$

Given a diagonalizable matrix P, the r-step transition probability matrix P^r obeys relation (A.12), as described in Appendix A.4. The time dependence of the probability state vector $s[r]$ is thus given by

$$s[r] \simeq s[0]\left(u\pi + \lambda_2^r x_2 y_2^T + O\left(\lambda_3^r\right)\right) \tag{3.8}$$

where we order the $N+1$ eigenvalues[2] as $\lambda_1 = 1 \geqslant |\lambda_2| \geqslant ... \geqslant |\lambda_{N+1}| \geqslant 0$. λ_k is the k-th largest diagonal element of matrix P, and x_k and y_k are the right and left-eigenvectors associated with λ_k, respectively. The tendency of the network towards the steady-state is thus, determined by the second largest eigenvalue λ_2 of P. However, the matrix P is not always diagonalizable, as discussed in Appendix A.4. In such cases, the probability state vector $s[r]$ is calculated using (3.7). The mean number of informed nodes in each round r is consequently computed by

$$E[X_r] = \sum_{i=1}^{N+1} i \times s_i[r] \tag{3.9}$$

In the sequel, metrics measuring the performance of the proposed gossiping-based schemes, that are used for content dissemination or discovery, are derived.

[2]The Frobenius' Theorem [119, A.4.2] specifies that there is only one largest eigenvalue that equals to 1.

3.3.1 Content Dissemination Metrics

The effectiveness of the content dissemination process can be assessed in terms of the minimum number of rounds required to inform m random nodes (apart from the initiator). Let $A_{N+1}(m)$ denote such a random variable. Let $e_m(r)$ denote the event of having m nodes informed in round r, and let $e_m^c(r)$ be its complement. The following equivalence of the events can be established as

$$\{A_{N+1}(m) = r\} = e_m(r) \cap \left\{ \bigcap_{j=1}^{r-1} e_m^c(j) \right\}$$

with $e_m(1) \subseteq e_m(2) \subseteq ... \subseteq e_m(r)$. Thus, we will have

$$\begin{aligned} \Pr[A_{N+1}(m) = r] &= \Pr[e_m(r) \cap e_m^c(r-1) \cap ... \cap e_m^c(1)] \\ &= \Pr[e_m(r) \backslash e_m(r-1)] \\ &= \Pr[e_m(r)] - \Pr[e_m(r-1)] \end{aligned} \tag{3.10}$$

where we can show that

$$\Pr[e_m(r)] = \frac{1}{\binom{N}{m}} \sum_{i=1}^{N+1} \binom{i-1}{m} s_i[r] \tag{3.11}$$

Since the number of informed nodes never decreases as r grows, we have $\lim_{r\to\infty} \Pr[A_{N+1}(m) = r] = 0$ while $\lim_{r\to\infty} \Pr[e_m(r)] = 1$.

The mean minimum number of rounds required to inform m random nodes is given by

$$\bar{A}_{N+1}(m) = \sum_{r=1}^{r_{\max}} r \Pr[A_{N+1}(m) = r] \tag{3.12}$$

For numerical calculations, we take the upper bound of r as

$$r_{\max} = \min\{r : 1 - \Pr[e_m(r)] < \xi\}$$

where ξ is a very small positive number.

3.3.2 Search Metrics

The effectiveness of a content search process is assessed in terms of the minimum number of search rounds required to reach a node that possesses the content, for the first time. To generalize the study here, we assume that l copies of the content are randomly distributed over the network of N nodes, excluding the initiator node. Let $B_{N+1}(l)$ denote the aforementioned random variable of the minimum number of rounds. Let

$f_l(r)$ denote the event that at least one copy of the content has been discovered by round r. It is not difficult to show that

$$\Pr[f_l(r)] = \sum_{i=1}^{N+1} \left[1 - \frac{\binom{N+1-i}{l}}{\binom{N}{l}}\right] \times s_i[r] \tag{3.13}$$

where $1 - \frac{\binom{N+1-i}{l}}{\binom{N}{l}}$ is the probability that there is at least one copy of the content in state i, with i searched (informed) nodes.

By following a similar approach as in Section 3.3.1, the following expression is derived

$$\Pr[B_{N+1}(l) = r] = \Pr[f_l(r)] - \Pr[f_l(r-1)] \tag{3.14}$$

Consequently, the mean minimum number of rounds to find a content, denoted by $\bar{B}_{N+1}(l)$, can be calculated by

$$\bar{B}_{N+1}(l) = \sum_{r=1}^{r_{\max}} r \Pr[B_{N+1}(l) = r] \tag{3.15}$$

in which the upper bound of r during numerical calculation is taken as

$$r_{\max} = \min\{r : 1 - \Pr[f_l(r)] < \xi\}$$

where ξ is a very small positive number.

We define another metric to evaluate the overhead caused by the search process: the mean number of nodes that has been searched (informed) by the round that the content is discovered for the first time. This quantity, $\bar{Y}_{N+1}(l)$, is derived from

$$\bar{Y}_{N+1}(l) = \sum_{r=1}^{r_{\max}} \Pr[B_{N+1}(l) = r] E[\hat{X}_r] \tag{3.16}$$

where $E[\hat{X}_r]$ is the mean number of searched nodes in round r, in which the content is found for the first time.

The expectation, $E[\hat{X}_r]$, is computed as

$$E[\hat{X}_r] = \sum_{j=1}^{N+1} j \times \Pr[\hat{X}_r = j] \tag{3.17}$$

in which $\Pr[\hat{X}_r = j]$ is the probability that there are j search nodes, and that the content is found for the first time in round r. The computation of $\Pr[\hat{X}_r = j]$ depends on $\Pr[X_{r-1} = i]$, the probability of having i searched nodes in round $r-1$, which can be derived from (3.7). Given that there are i $(1 \leq i \leq N+1)$ searched nodes in round

$r-1$, and that the content is not found yet, the condition of $X_r > X_{r-1}$ has to be satisfied in order to assure that the content can be found for the first time in round r. Therefore, the probability of $\Pr[\hat{X}_r = j]$ is given by

$$\Pr[\hat{X}_r = j] = \sum_{i=1}^{N+1} \Pr[X_{r-1} = i] \times \frac{P_{ij}}{1 - P_{ii}} \tag{3.18}$$

With (3.18) and (3.17), we can derive the mean number of searched nodes, $\bar{Y}_{N+1}(l)$, by the round that the content is discovered for the first time.

3.4 Results and Discussions

In this section, we developed a simulation program to simulate message dissemination/search through gossiping by using C language. The results of the analysis are compared with the results derived from the simulated program. In [119], it is shown that the average error over the non-zero values returned from the simulations decreases as $O\left(\frac{1}{\sqrt{n}}\right)$, where n is the number of times that a simulation is performed. In this thesis, 10^4 iterations are carried out for each simulated result. For both of the information dissemination and search process, random selection of k neighbors is performed. The initiator is also randomly chosen in each of the simulation instances. In the search process, l copies of the content are randomly placed at different nodes. The information dissemination process stops when there are m informed nodes in the network, and the search process terminates when at least one copy of the content is discovered. For each iteration, we collect the number of gossiping rounds until the program finishes, from which, the *probability density function* (pdf) and the mean are computed. For the search process, the number of searched nodes until the end of the program is captured. The mean number of searched nodes are calculated consequently. The major focus is to examine the performance of the metrics that are proposed in Section 3.3, as well as the impact of important parameters under both the blind and smart selection schemes.

3.4.1 Content dissemination results

In Fig. 3.3, we present the results on the probability that all N nodes are informed by round r, obtained from (3.11) for $m = N$. Notice that $\Pr[e_m(r)]$ is eventually the cumulative distribution function (cdf) of $A_{N+1}(m) = r$, calculated from (3.10). As we can see from Fig. 3.3, the simulation results for $\Pr[e_m(r)]$ match those from the exact analysis in (3.10) very well[3]. As expected, the larger the network, the more rounds it takes to inform all nodes. We notice that there exists a threshold until all nodes are

[3] In the following results, we do not show the simulated results.

informed. For instance, under the blind selection scheme, it is only possible to inform all nodes after 4 rounds in a small network with $N = 10$, as shown in Fig. 3.3 (a). While for larger network with $N = 100$, it is only possible to inform all nodes after 8 gossiping rounds.

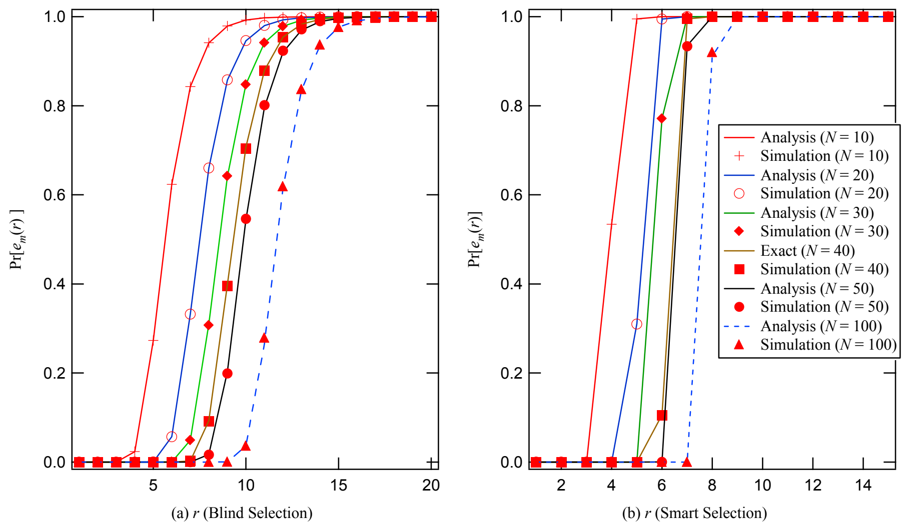


Figure 3.3: The probability that all nodes ($m = N$) are informed by round r under the blind selection scheme (a), and the smart selection scheme (b). ($k = 1$ and $\beta = 1.0$).

In Fig. 3.4 we study the tail behavior (on log-scale) of $\Pr[A_{N+1}(m) > r] < \varepsilon$, where ε is a pre-defined probability. The tail probability of $\Pr[A_{N+1}(m) > r]$, computed from (3.10), is the probability that the minimum number of rounds required to inform the entire network exceeds r. For instance, under the blind selection scheme and for $N = 10$, the probability that all nodes are informed after round 10 is less than 10^{-2}, as shown in Fig. 3.4 (a). In other words, in 99.99% of the cases, all network nodes can be informed by round 10. With the same stringency of $\varepsilon = 10^{-2}$, informing all network nodes under the smart selection scheme is achieved in only 5 rounds (Fig. 3.4 (b)). The above observation confirms the higher efficiency of the smart scheme. Notice that for larger network (e.g. $N = 100$), the smart scheme informs the entire network in about half the rounds required under the blind scheme, for the same chance of 10^{-2}. It is obvious that the smart selection scheme outperforms the blind selection scheme. With the exact analysis, we can compare the performance of the two extreme case quantitatively.

Moreover, with the exact analysis, we are able to evaluate the performance of $\Pr[A_{N+1}(m) > r]$ given a higher level of stringency (e.g. $\varepsilon = 10^{-6}$), which is normally, very difficult to achieve with simulations[4]. For instance, in Fig. 3.4 (a), we

[4]As described at the beginning of this section, in order to achieve an accuracy of 10^{-6} from the simulated results, the simulation needs to be performed 10^{12} times, which takes very long time.

can find that, in 99.999999% of the cases, the entire network of $N = 100$ can be informed by round 26. The results shown in Fig. 3.4 ensures that the entire network coverage can be guaranteed with a high probability of $1 - \varepsilon$. The tail probability of $\Pr\left[A_{N+1}(m) > r\right] < \varepsilon$ can be utilized to determine the number of dissemination rounds to be implemented by the service provider (or required for by the end user) in order to meet the aforementioned quality of service (QoS) requirement. This is also known as the *maximum gossiping rounds* problem.

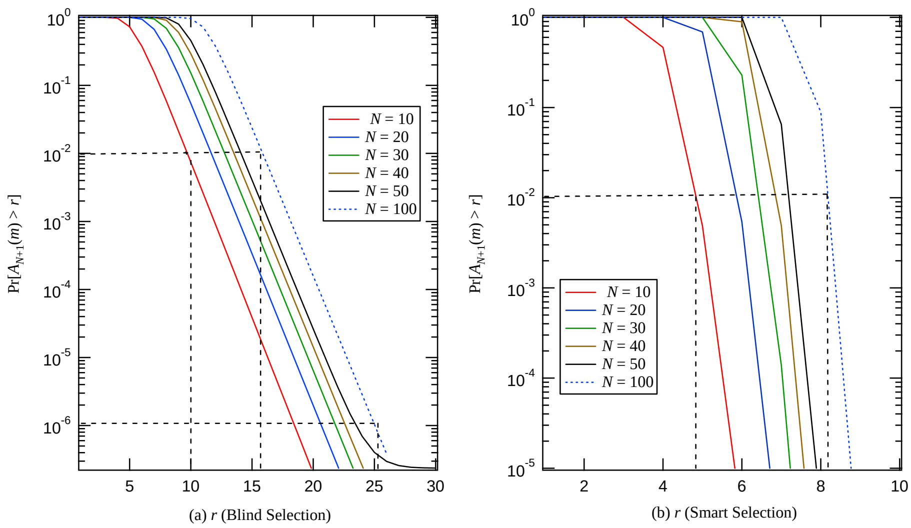


Figure 3.4: The tail behavior of $\Pr[A_{N+1}(m) > r]$ vs number of rounds r under blind (a), and smart (b) selection schemes ($k = 1$, $\beta = 1.0$, and $m = N$). Results presented in this figure are from the exact analysis.

By plotting the mean number of rounds to inform the entire network in Fig. 3.5, we notice that $\bar{A}_{N+1}(m)$ grows almost proportionally to $\log(N)$. This is because, asymptotically (for large N and small k), the expected number of rounds of any dissemination process in a general graph scales in the order of $\log(N)$, as shown in [119, pp. 342], which indicates the efficiency of the investigated algorithms. In [104] and [79], Pittel and Karp *et al.* also gave the same $\log(N)$ upper bound of gossiping-based algorithms with $k = 1$. Consequently, we can approximate the mean minimum number of rounds to inform the entire network as $\bar{A}_{N+1}(m) \sim \gamma_k \log(N) + \alpha_k$, where γ_k and α_k are variables depending on k. The speed of disseminating content under the smart selection scheme is less affected by increasing the network size, since the slope γ_k under this scheme is always smaller than that under the blind scheme for the same k.

In Fig. 3.6, $\bar{A}_{N+1}(m)$ is plotted as a function of β for different network sizes. As the cooperation probability β increases, $\bar{A}_{N+1}(m)$ decreases logarithmically with the same slope for different network sizes, and for both the blind and the smart selection

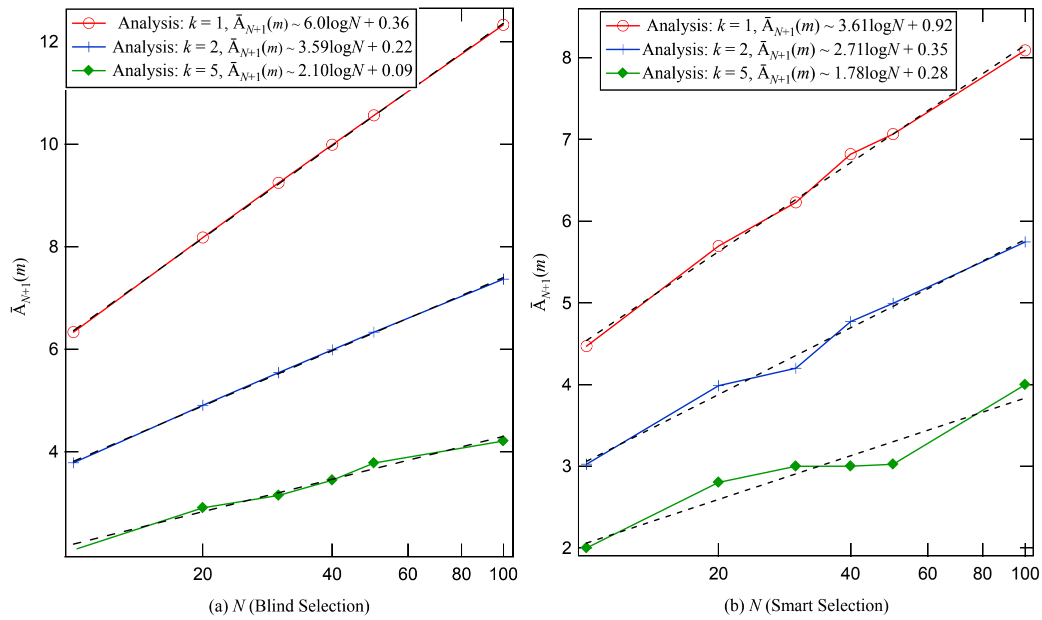


Figure 3.5: Mean number of rounds required to inform the entire network, $\bar{A}_{N+1}(m)$, as a function of N under the blind (a), and smart (b) selection schemes ($m = N$, $\beta = 1.0$, and varying k). The horizontal axis is plotted on log scale and the dotted lines are the fitting curves.

scheme. This phenomenon indicates that the mean number of rounds to inform the entire network decreases at the same speed as a function of β, regardless of the network size. Therefore, it could be convenient to extrapolate the curve for larger network sizes N. Furthermore, by decreasing the cooperation probability β, the performance of disseminating content degrades for both the blind and the smart algorithm. For example, the mean number of rounds to inform the entire network with $\beta = 0.2$ is approximately 5.3 times of that with $\beta = 1.0$ for the blind selection and 5.8 times for the smart selection. The performance of the smart selection scheme is, as expected, better than the blind selection scheme. For instance, for $N = 100$, with $\beta = 1.0$, it takes on average, 3.9 more rounds to inform all nodes for the blind selection than for the smart selection; and the blind selection scheme needs 18.1 more rounds to inform the entire network with $\beta = 0.2$, compared with the smart selection.

3.4.2 Content Search Results

To improve the efficiency of the search process, we can either increase the number of nodes (or gossiping-targets) k searched in each round or distribute more copies l of the content. The associated overhead $\bar{Y}_{N+1}(l)$, the mean number of nodes that have been

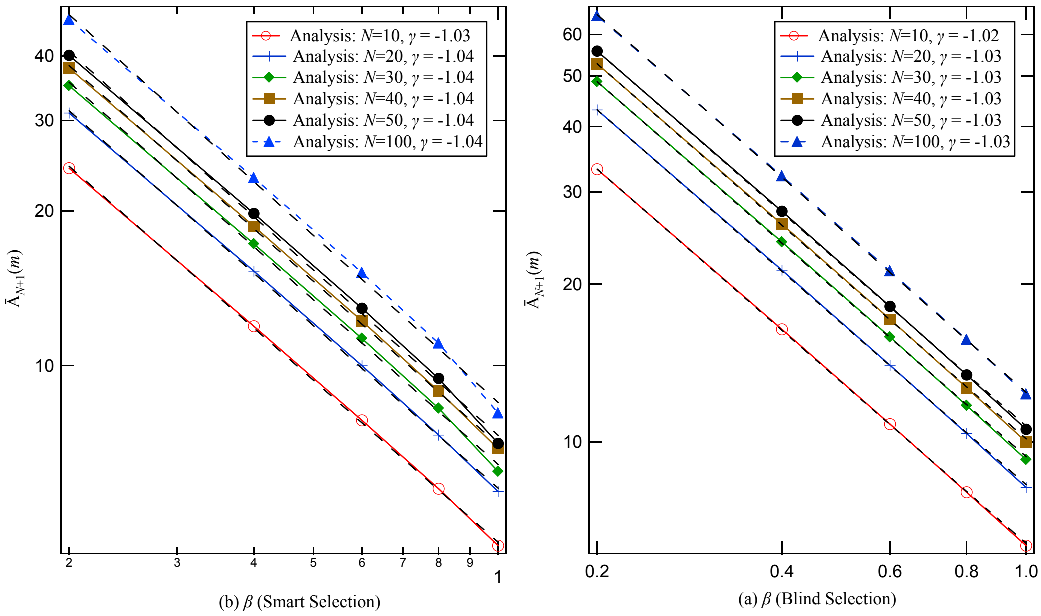


Figure 3.6: Mean minimum number of rounds required to inform the entire network, $\bar{A}_{N+1}(m)$, as a function of β under the blind (a), and smart (b) selection schemes ($k = 1$, $m = N$, and varying β). Both sub-figures are plotted on log-log scale. The dotted lines represent the fitting curves, and γ is the fitting parameter of $\log\left(\bar{A}_{N+1}(m)\right) \sim \gamma \log(\beta) + \alpha$.

searched by the round that the content is found, is also evaluated. We examine the impact of k and l on the performance of the search process, by taking the blind selection scheme as an example.

Fig. 3.7 (a) confirms that, by searching more nodes (or gossiping-targets) in each round, the mean minimum number of rounds required to discover the content is reduced. While the associated overhead, $\bar{Y}_{N+1}(l)$, grows by increasing k (Fig. 3.7 (b)). In Fig. 3.8, we present the effect of increasing the number of copies of the content distributed in the network ($l = 1, 2, 5$), with fixed $k = 1$. As seen from this figure, both the speed of discovering content $\bar{B}_{N+1}(l)$, and the caused overhead $\bar{Y}_{N+1}(l)$ are improved, since less gossiping rounds are needed to find a content (Fig. 3.8 (a)), and the mean minimum number of searched nodes until the content is found (Fig. 3.8 (b)) also decreases, as l increases. Therefore, in order to have an efficient content discovery process (with fast searching speed $\bar{B}_{N+1}(l)$ and low overhead $\bar{Y}_{N+1}(l)$), we should opt for placing more copies of the content within the network, instead of increasing the number of searched nodes k in each round in the gossip-based search process. Notice that in both Fig. 3.7 and Fig. 3.8, the mean number of rounds to discover the content increases proportionally to $\log(N)$, and the mean number of searched nodes grows linearly as a function of N.

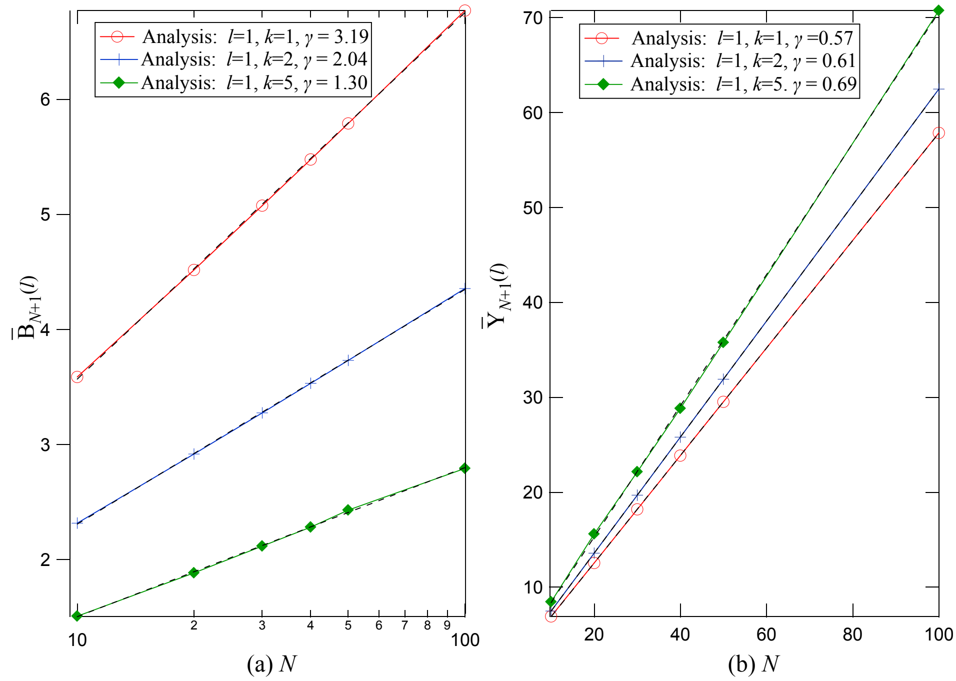


Figure 3.7: Mean minimum number of rounds to find the content (a), and the mean number of searched nodes by the round that the content is discovered for the first time (b). (Blind selection scheme with $l = 1$, varying k, and $\beta = 1.0$). The dotted lines are the fitting curves, and γ is the fitting parameter.

Next, we show the impact of different values of β on the performance of the search process in Fig. 3.9. By increasing the cooperation probability β (from 0.2 to 1.0), $\bar{Y}_{N+1}(l)$ increases slightly, while $\bar{B}_{N+1}(l)$ decreases dramatically for the same network size. For instance, to find the content in a network with $N = 100$ nodes, about five more nodes are searched when β increases from 0.2 to 1.0 under the blind selection scheme (Fig. 3.9 (b)). Whereas, the mean number of rounds to find the content for the first time with $\beta = 0.2$ is approximately 4.4 times of that with $\beta = 1.0$, as shown in Fig. 3.9 (a). Therefore, we conclude that the lower cooperation probability does not incur extra overhead in the network, but compromises severely the effectiveness of the searching algorithm. We also observe that to find a content with $l = 1$ in larger network, i.e. $N = 100$, the smart selection algorithm is more effective than the blind scheme regarding the search performance when peers have smaller probability to be cooperative. Let us once again take the network of $N = 100$ as an example. When peers are less cooperative, e.g. $\beta = 0.2$, the smart selection scheme only searches approximately one more node compared to the blind selection algorithm (Fig. 3.9 (d)). While $\bar{B}_{N+1}(l)$,

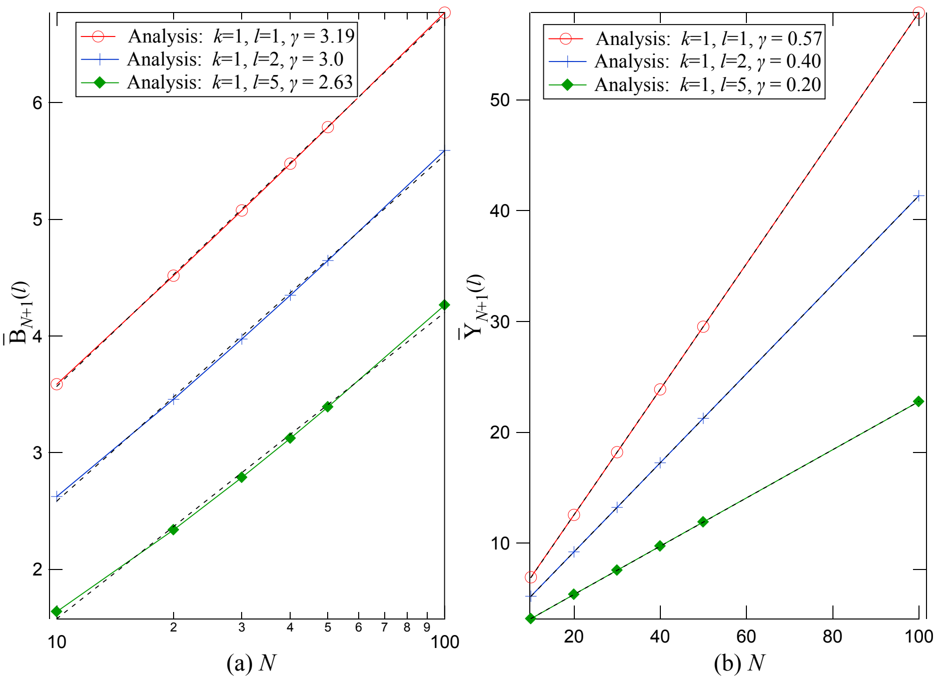


Figure 3.8: Mean minimum number of rounds to find the content (a), and the mean number of searched nodes by the round that the content is discovered for the first time (b). (Blind selection scheme with $k = 1$, varying l, and $\beta = 1.0$). The dotted lines are the fitting curves, and γ is the fitting parameter.

on the other hand, is 4 rounds less that the blind selection scheme (Fig. 3.9 (c)). With $\beta = 1.0$, it takes one less round to find the content with the smart selection algorithm, while searching 4 more nodes compared to the blind selection scheme.

3.5 Summary

In this chapter we focus on modeling the process of gossip-based message dissemination under the assumption of uniform neighbor selection over the entire nodes in the network. The level of cooperation by the nodes selected as the gossiping-targets is also incorporated in the model. The cases of the blind gossiping-target selection and of the smart one are both analyzed. The obtained analytic results are verified through simulations. From the results, several practical performance metrics of interest and important design parameters are obtained. For instance, the speed of the dissemination (in gossiping rounds) process required to achieve certain percentage of network coverage with a minimum probability is derived and evaluated. The smart selection algorithm is, in nature, more effective than the blind selection scheme when disseminating content. By

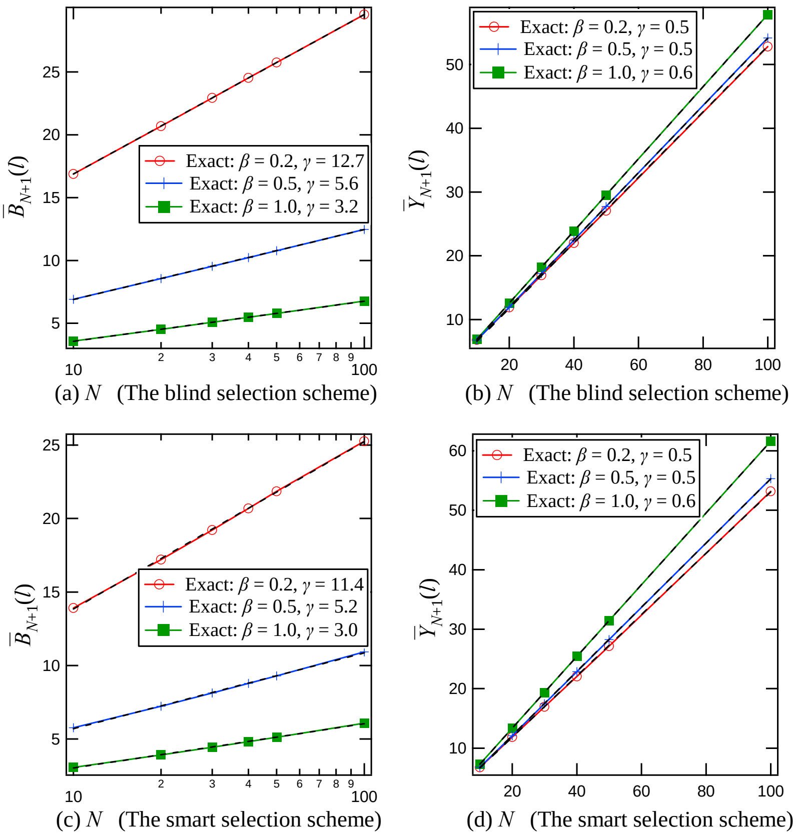


Figure 3.9: Mean number of rounds to find at least one copy of the file for the first time for the blind (a) and smart (c) selection schemes. Mean number of searched nodes by the round to find at least one copy of the file for the blind (b) and smart (d) selection schemes. (With $k = 1$, $l = 1$, and varying β). γ is the fitting parameter as described in previous figures.

using the exact analysis, we have compared the performance difference of the above two algorithms quantitatively. For instance, to inform the entire network with certain QoS stringency, the smart selection scheme only needs half of the gossiping rounds compared with the blind selection algorithm. By increasing the cooperation probability from $\beta = 0.2$ to $\beta = 1.0$, the mean number of rounds to inform the entire network decreases logarithmically with the same slope for different network sizes, and for both the blind and the smart selection algorithm. Our results about content search also suggest that when a certain speed (number of rounds) is desirable to discover some content, it is less costly for the search process to try to place more content replications l in the network, instead of trying to hit content residing in some nodes only by increasing the number of gossiping-targets k, contacted in each round. The effectiveness of the searching algorithm is impaired with a lower cooperation probability, whereas no significant amount of overhead ($\bar{Y}_{N+1}(l)$) is generated. Considering the trade-off between the overhead and the effectiveness of the search process, the smart selection scheme is more effective with small cooperation probability.

Chapter 4

Selecting The Nearest Peer For Content Retrieval

So far, we have evaluated the performance of uniform peer selection scheme during content propagation. In this chapter, we extend our analysis one step further. Let us assume that, after the searching process described in Chapter 3, we have found m peers possessing the desired content. The process of selecting a best peer (in cost, bandwidth, delay, etc.) among the group of m peers for content retrieval, thus becomes a vital procedure. We revisit the content retrieval process as finding the most nearby peer from the initiator. Under this context, the initiator refers to the peer who initiates the downloading request. Our model considers the *hopcount* and *delay* as the major criteria to assess the closeness between peers.

We confine the problem as follows: given a network of size N, over which m peers with the desired content are randomly scattered, what is the distribution of the hopcount and delay respectively to the most nearby peer from a requesting node? By solving the above problem, we expect to answer the fundamental question of how many replicas of a particular file need to be distributed (or how many peers possessing the desired content need to be discovered) so that the most nearby peer can always be reached within j hopcount or t delay.

Typical examples that our model applies are the content distribution networks (CDNs), and P2P streaming systems. The design purpose of CDNs is to deliver content to end-users from the most nearby cache server in a reliable and timely manner [103]. In P2P streaming networks, finding the closest peers is also preferred, so that the waiting (or buffering) time at an end-user is minimized. Hence, the hopcount and delay are reasonable metrics to guarantee the QoS provided by CDNs, or P2P streaming networks. Other QoS parameters, such as content availability and accessibility is beyond the scope of this thesis.

4.1 Modeling Assumption

We model the number of hops and the latency to the nearest peer among a set of m peers based on three assumptions: (a) a dense graph model[1] for the underlying network, (b) regular link weight around zero, and (c) i.i.d. link weight distribution on each link.

The shortest path from a source to a destination is computed as the path that minimizes the link weights[2] along that path. In [119, Chapter 16.1], it is shown that a regular link weight distribution - regular means a linear function around zero - will dominate the formation of the shortest path tree (SPT), which is the union of all shortest paths from an arbitrary node to all the other destinations. A uniform recursive tree (URT) is asymptotically the SPT in a dense graph with regular i.i.d. link weights (e.g. exponential link weights) distribution [117]. A URT of size N is a random tree that starts from the root A, and where at each stage a new node is attached uniformly to one of the existing nodes until the total number of nodes reaches N.

4.2 Hopcount Distribution to The Nearest Peer

4.2.1 Theoretical analysis

The number of hops from a requesting peer to its most nearby peer, denoted by $h_N(m)$, is the minimum number of hops among the set of shortest paths from the requesting node to the m peers in the network of size N. Let $H_N(m)$ be the hopcount starting from one, excluding the event $h_N(m) = 0$ in the URT. Since $\Pr[h_N(m) = 0] = \frac{m}{N}$, we have

$$\Pr[H_N(m) = j] = \Pr[h_N(m) = j | h_N(m) \neq 0] = \frac{1}{1 - \frac{m}{N}} \Pr[h_N(m) = j] \tag{4.1}$$

with $j = 1, 2, ...N$ and $\Pr[h_N\,(m) = j]$ recursively solved in [119, p. 427]. However, the recursive computation involves a considerable amount of memory and CPU-time which limits its use to relatively small sizes of $N \leq 100$.

Fig. 4.1 illustrates $\Pr[H_N(m) = j]$ versus the fraction of peers $\frac{m}{N}$ for different hops j with network size varying from $N = 20$ up to $N = 60$. The interesting observation from Fig. 4.1 is that, for separate hops j, the distribution $\Pr[H_N(m) = j]$ rapidly tends to a distinct curve for most of the small networks ($N \leqslant 100$) and that the increase in the network size N only plays a small role. Further, the crosspoint of curve $j = 1$ and $j \geqslant 2$

[1] The dense graph is a heterogenous graph with the average degree $E[D] \geqslant p_c N \approx O(\log N)$ and a small standard deviation $\sqrt{Var[D]} \ll E[D]$, where $p_c \sim \frac{\log N}{N}$ is the disconnectivity threshold of the link density [119, Chapter 15.6.3].

[2] The link weight w_{ij} assigned to a link (i, j) between node i and node j in a network, is a real positive number that reflects certain properties of the link, i.e. distance, delay, loss, or bandwidth.

(the bold line) around $\frac{m}{N} = 15\%$ indicates that in small networks (i.e. $N = 20$), the peer fraction should always be larger than 15% to ensure $\Pr[H_N(m) = 1] > \Pr[H_N(m) \geqslant 2]$.

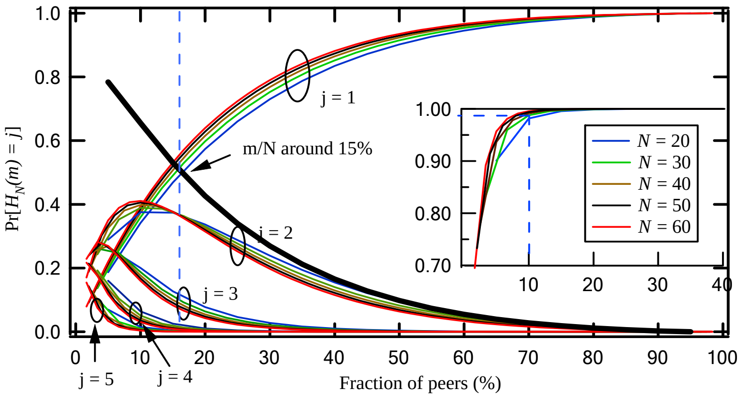


Figure 4.1: $\Pr[H_N(m) = j]$ versus the fraction of peers $\frac{m}{N}$ from network size $N = 20$ to 60. The bold line is the pdf of $H_N(m) \geqslant 2$ for $N = 20$. The inserted figure plots the $\Pr[H_N(m) \leq 4]$ as a function of peer fraction $\frac{m}{N}$ for network sizes $N = 20$ to 60.

Equation (4.1) can be applied to estimate the peer group size for a certain content delivery service. For instance, if the operator of a CDN with 40 routers has uniformly scattered 4 servers (peer fraction around 10%) into the network, he can already claim that approximately in 98% of the cases, any user request will reach a server within 4 hops ($j \leq 4$) as seen in the inserted figure of Fig. 4.1. Placing more servers in the network will not improve the performance significantly.

4.2.2 An approximation to $\Pr[H_N(m) = j]$

To avoid the recursive calculation in (4.1), we approximately compute $\Pr[H_N(m) = j]$ by assuming the independence of the hopcount from the requesting node to the m peers when m is small[3]. The general form for the approximation is expressed as

$$\Pr[H_N(m) \leq j] \approx 1 - (\Pr[H_N > j])^m \tag{4.2}$$

provided that at least one of the peers is j hop away (or not all peers are further than j hop away), where H_N is defined as the hopcount from the requesting peer to an arbitrary chosen node in the URT, excluding the event that the arbitrary node coincides with the requesting node (i.e. zero hopcount is not allowed).

[3]The path overlap from the root to the m peers in the URT causes correlation of the hopcount between peers. When m is small compare to the network size N, the path overlap is expected to be small, and so is the correlation of the hopcount. The larger the m, the more dependent of the hopcount from the root to the m peers becomes.

Take $j = 1$ as an example. From (4.2), we have $\Pr[H_N(m) = 1] \approx 1 - (1 - \Pr[H_N = 1])^m$, where $\Pr[H_N = 1]$, the probability in the URT that the destination is one hop away from the requesting peer, is equal to [119, p. 361]

$$\Pr[H_N = 1] = \frac{1}{N-1} \sum_{j=1}^{N-1} \frac{1}{j} \tag{4.3}$$

In Fig. 4.2, we compare the exact result of $\Pr[H_N(m) = 1]$ with the approximation calculated by (4.2). The two curves are quite close to each other, especially for small number of $\frac{m}{N}$. The difference for larger m is caused by the correlation of the hopcount between node pairs, as explained in footnote[3]. Therefore, we confine the estimation of (4.2) with large N and small m, whereas the exact result is applicable for $N \leqslant 100$ with all m. The number of peers that is one hop away is, on average $m \Pr[H_N = 1]$. To ensure that the nearest peer can be reached within one hop, the minimum number of peers in the network should approximately satisfies the condition of $m \Pr[H_N = 1] \geq 1$. For large N, (4.3) tends to $\Pr[H_N = 1] \simeq \frac{\log N}{N}$, and we have $m \geq \frac{N}{\log N}$. The minimum peer fraction in a large network is thus bounded by

$$\frac{m}{N} \geq \frac{1}{\log N} \tag{4.4}$$

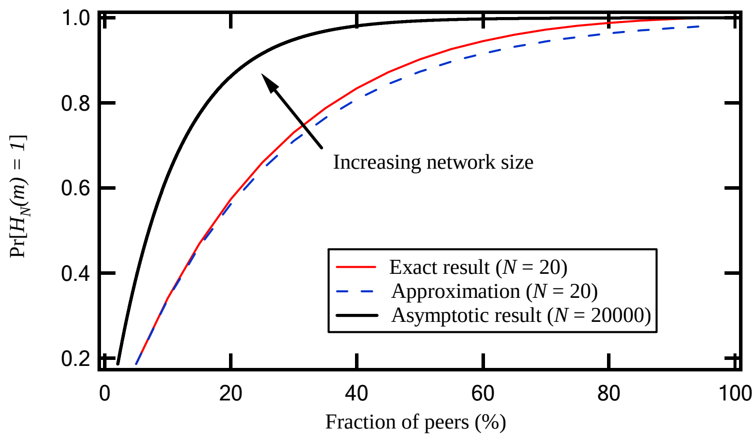


Figure 4.2: Comparison between the exact (4.1) and the approximated hopcount distribution (4.2) versus the fraction of peers $\frac{m}{N}$ for $H_N(m) = 1$. The bold curve denotes the asymptotic result for very large network caculated by (4.4).

4.3 Weight of The Shortest Path to The First Encountered Peer

4.3.1 The Asymptotic Analysis

In [119, p. 349], the shortest path problem between two arbitrary nodes in the dense graph with regular link weight distribution (e.g. exponential link weights) has been rephrased as a Markov discovery process evolving as a function of time from the source and stops at the time when the destination node is found. The transition rate in this continuous-time Markov chain from state n, containing the n already discovered nodes, to the next state $n+1$ is $\lambda_{n;n+1} = n(N-n)$. The inter-attachment time τ_n between the inclusion of the n-th and $(n+1)$-th node in the SPT for $n = 1, 2, ...N-1$, is exponentially[4] distributed with parameter $n(N-n)$.

The exact *probability generating function* (pgf) $\varphi_{W_{N;m}(z)} = E[e^{-zW_{N;m}}]$ of the weight $W_{N;m}$ of the shortest path from an arbitrary node to the first encountered peer among m peers can be formulated as

$$\varphi_{W_{N;m}(z)} = \sum\nolimits_{k=1}^{N-m} E[e^{-zv_k}] \Pr[Y_m(k)]$$

where $\Pr[Y_m(k)]$ represents the probability that the k-th attached node is the first encountered peer among the m peers in the URT, v_k denotes the weight of the path to the k-th attached node, and $v_k = \sum_{n=1}^{k} \tau_n$. The inter-attachment time τ_n is an exponential random variable with parameter $n(N-n)$, the corresponding generating function of v_k can be written as $E[e^{-zv_k}] = \prod\limits_{n=1}^{k} \frac{n(N-n)}{z+n(N-n)}$.

The formation of the URT with m attached peers depicted in Fig. 4.3 indicates $\binom{m}{1}$ possibilities that one peer out of m may be in the k-th position in the continuous Markov chain. While the remaining $m-1$ peers should always appear in the position that are larger than k-th position. Hence, there are $\binom{N-1-k}{m-1}$ ways to distribute the $m-1$ peers over the $N-1-k$ position. Further, the remaining non-peer nodes can be distributed in $(N-1-m)!$ ways. This analysis leads us to express $\Pr[Y_m(k)]$ as

$$\Pr[Y_m(k)] = \frac{m(N-1-k)!(N-1-m)!}{(N-1)!(N-m-k)!}$$

and the generating function $\varphi_{W_{N;m}(z)}$ becomes

$$\varphi_{W_{N;m}(z)} = \frac{m(N-1-m)!}{(N-1)!} \sum\nolimits_{k=1}^{N-m} \frac{(N-1-k)!}{(N-m-k)!} \prod_{n=1}^{k} \frac{n(N-n)}{z+n(N-n)} \tag{4.5}$$

[4] The exponential distribution is $f(x) = \alpha e^{-\alpha x}$.

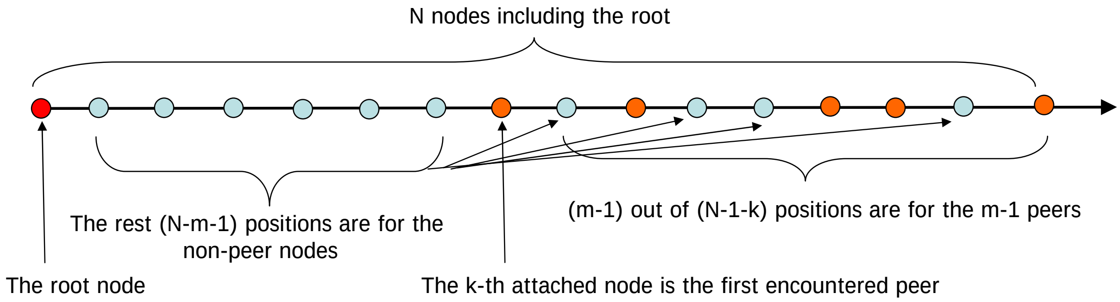


Figure 4.3: Formation of the URT where the $k-th$ attached node is the first encounted peer in the peer group of m. The shaded circle represents a peer distributed in the tree.

The asymptotic pdf of (4.5) is derived in [121, Section 3.1] as

$$\lim_{N\to\infty} \Pr[NW_{N;m} - \ln \frac{N}{m} \leq y] = e^{-my} m^{m+1} e^{me^{-y}} \int_{me^{-y}}^{\infty} \frac{e^{-u}}{u^{m+1}} du \tag{4.6}$$

which converges to the Fermi-Dirac distribution function

$$\lim_{N\to\infty} \Pr[NW_{N;m} - \ln \frac{N}{m} \leq y] = \frac{1}{1+e^{-y}} \tag{4.7}$$

for large m as shown in Fig. 4.4. It illustrates that a relatively small peer group $m \approx 5$ is sufficient to offer a good service quality because increasing the number of peers can only improve the performance marginally, i.e. logarithmically in m.

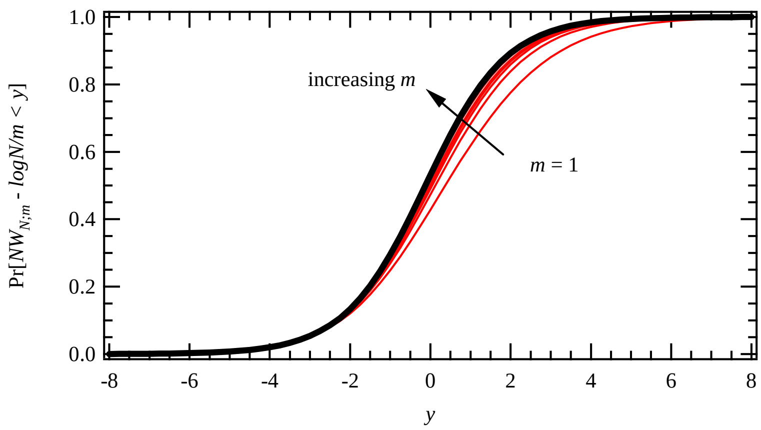


Figure 4.4: The convergence of the probability distribution function with scaled random variable $NW_{N;m} - \ln \frac{N}{m}$ towards the Fermi-Dirac distribution (in bold) for increasing $m = 1, 2, 3, 4, 5, 10, 15, 20$.

4.4 Discussion on The URT Model

The URT model we used to calculate the hopcount and delay distribution in Section 4.1 is built on two major assumptions: 1) a dense graph to mimic the underlying network and 2) i.i.d. regular link weight. The Internet might be denser than what has been measured, taking into account all sorts of significant sampling bias, such as insufficient sources for traceroutes suggested in [52]. We will also give indications on the link weight distribution and the applicability of the URT model in P2P network in this section.

4.4.1 Link Weight Structure of Networks

We have used the data from the National Road Database provided by the Dutch transport research center to give an indication on the link weight distribution in a transportation network. The link weight of the Dutch road is evaluated as the physical distance between two roadsections.

In Fig. 4.5, we fit the link weight distribution $F_w(x)$ of the Dutch road network with a linear function. A regular (linear) link weight distribution is found within a small range $[0, \epsilon]$, where $\epsilon \sim 0.03$, which gives evidence to the assumption of regular link weight structure around zero. The link weight structure in the Internet can be tuned independently from the underlying network. Therefore, we claim that the assignment of regular link weights is reasonable.

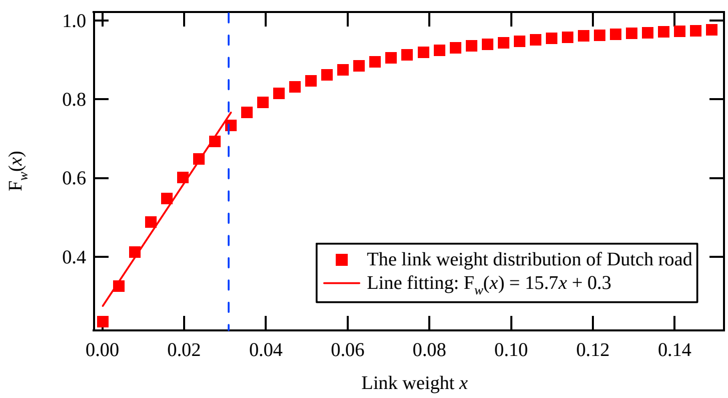


Figure 4.5: $F_x(x) = \Pr[w \leq x]$ of the link weight of the Dutch transportation network with the x axis normalized between $[0, 1]$ ($x \in [a, b] \overset{normalize}{\rightarrow} x \in \left[\frac{a}{b}, 1\right]$). The correlation coefficient $\rho = 0.99$ suggests a high fitting quality.

4.4.2 Applicability of The URT Model

We have carried out a series of experiments by using the *traceroute* data provided by iPlane [15] to give further indication on how well the URT model matches the real network. iPlane performs periodic measurements by using PlanetLab nodes to probe a list of targets with global coverage. We use the iPlane measurement data executed on 8th June 2007. We extract the stable traces from 52 Planetlab nodes that are located in different Planetlab sites. Assuming the *traceroutes* represent the shortest paths, we construct a SPT rooted at each PlanetLab node (as a source) to m peers (destinations), resulting in 52 SPTs in total. By using a map with all aliases resolved in iPlane, we obtain the router-level SPTs. The m peers are randomly chosen over the world, and the hopcount (H_{SPT}) and degree (D_{SPT}) distribution are obtained by making a histogram of the 52 SPTs (each with m destinations).

Experimental Results on Node Degree Distribution

Three sets of experiments with $m = 10, 25$ and 50 were conducted to examine the degree distribution of the sub-SPT because the number of peers in a P2P network is not expected to be large[5]. We observed from the experiments that an exponential node degree distribution is, if not better, at least comparable to the power law distribution that has been reported in most of the published papers. The power law distribution is defined as

$$\Pr[X \leq x] = cx^{-\alpha+1} \tag{4.8}$$

In Fig. 4.6, we fitted $\Pr[D_{SPT} = k]$ for $m = 10$ with a linear function on both log-lin and log-log scales. The fitting quality of ρ_e on the log-lin scale and ρ_p on the log-log scale are presented respectively. The quality of the fitting on the log-log scale ($\rho_p = 0.99$) is only slightly higher than that of the log-lin scale ($\rho_e = 0.98$), which questions the power law degree distribution of a small subgraph of the Internet topology.

A similar phenomenon is also observed for $\Pr[D_{SPT} = k]$ for $m = 25$ and $m = 50$. We provide the correlation coefficients for $m = 25$ and $m = 50$ in Table 4.1. Again, the quality of the fitting seems to be comparable on both scales.

Table 4.1: Correlation coefficient for both log-lin (ρ_e) and log-log (ρ_p) scale of $m = 10$, 25 and 50

	ρ_e	ρ_p
$m = 10$	0.98	0.99
$m = 25$	0.95	0.99
$m = 50$	0.95	0.99

[5]Measurements on PPLive, a popular P2PTV application [70] reveal that the number of active peers that a peer can download from is always smaller than 50.

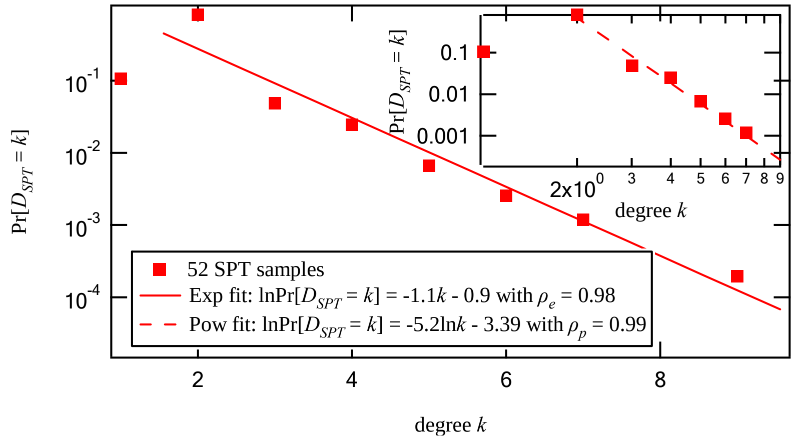


Figure 4.6: The histogram of degree D_{SPT} for 52 SPTs with $m = 10$ peers based on PlanetLab *traceroute* data on log-lin and log-log scale in the inset. The correlation coefficien of $\rho_e = 0.98$ and $\rho_p = 0.99$ suggest high fitting quality on both log-lin and log-log scale.

A discrepancy with the first three experiments occurs if we increase the peer size to 500. For larger subgraphs, a clear power law, rather than an exponential distribution dominates the node degree. Our findings in this thesis are in line with the conclusions made in [73], in which a comparable fitting was found for both exponential and power law property for small subgraph sizes of the Internet. We conclude that the node degree of a subgraph with small m cannot be affirmatively claimed to obey a power law distribution. At least, it is disputable whether the exponential distribution can be equally good as the power law.

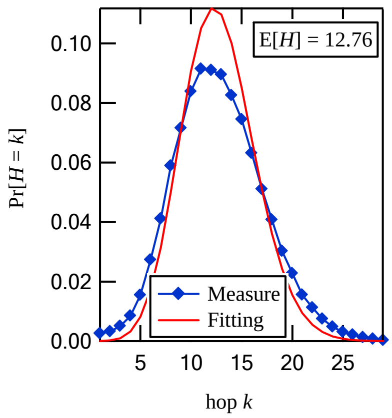
Hopcount Distribution on The Internet

The probability density function of the hopcount from the root to an arbitrary chosen node in the URT with large N can be approximated as the following according to [119, p. 349].

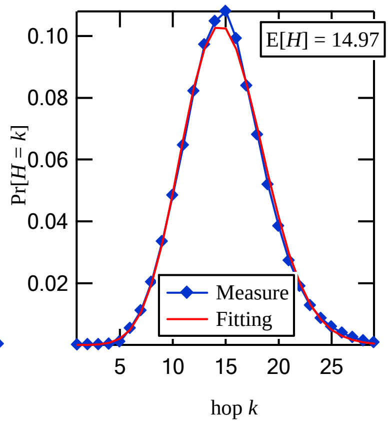
$$\Pr[H_N = k] \approx \Pr[h_N = k] \sim \frac{(\log N)k}{Nk!} \tag{4.9}$$

where H_N indicates the event that $k \geqslant 1$.

We plotted the pdf of the hopcount with $m = 50$ (50 traceroute samples for each tree) in Fig. 4.7 (a), in which we see a reasonably good fitting with (4.9). An even better fitting quality is found in Fig. 4.7 (b) if we increases the number of traceroutes samples by randomly selecting $m = 8000$ destinations for each tree, because more traceroutes gives higher accuracy. We conclude that the hopcount distribution of the Internet can be modeled reasonably well by the pdf of hopcount (4.9) in the URT.



(a) m = 50



(b) m = 8000

Figure 4.7: The histogram of hopcount derived from 52 SPTs for $m = 50$ (a) and $m = 8000$ (b) are fitted by the pdf (4.9) of the hopcount in the URT. The measured data for (a) and (b) are fitted with $log(N) = 12.76$ and $log(N) = 14.97$ respectively.

4.5 Summary

We take a further step to analyze the content retrieval process with nearest peer selection. We obtain the hopcount and delay distribution to the most nearby peer on the URT by assigning regular i.i.d. link weights (e.g. exponential link weights) on a dense graph. Both results suggest that a small peer group is sufficient to offer an acceptable QoS (in terms of hopcount and delay).

Via a series experiments, we have measured the degree and hopcount distribution in the Internet, from which we show the applicability of the URT model, and consequently the usability of the pdfs for both hopcount and delay. With a small group of peers ($m \leqslant 50$), the URT seems to be a reasonably good model to analyze content retrieval in P2P networks.

Chapter 5

Conclusion of Part I

The problem of disseminating and searching for content in distributed and unstructured networks - such as typical P2P and ad-hoc networks - is challenging. Content dissemination can be realized in two ways: either the content itself is disseminated or, instead, an advertisement message indicating its availability and location is spread. Searching for content is typically achieved through the dissemination of a query looking for the content itself or for the information about its location. In both cases, a message needs to be disseminated. Consequently, a scheme that effectively disseminates the message, would be applicable to all the aforementioned problems and such a scheme is the focus of this part. Moreover, once the desired content has been located after searching, the process of selecting a best peer for content retrieval becomes vital. Therefore, retrieving content from a most nearby peer is another problem that we have addressed. In this theoretical analysis, three major contributions are achieved.

First of all, we considered distributed P2P systems that are large-scale and highly dynamic. In this case, peers may only communicate with a subset of peers in the network, and they have to update their views of the network periodically with others to ensure the reliability during information dissemination. Furthermore, we consider gossip-based information dissemination schemes that emerge as an approach to maintain simple, scalable, and fast content dissemination and searching in today's distributed networks. However, preforming an exact analysis of the gossip-based information dissemination process with dynamic peer partial views requires a large amount of states, which is computationally not feasible. We have demonstrated that, the total number of states (the upper bound) to describe the entire system exactly is $2^{(N+1)^2+N+1}$.

Secondly, we carried out an exact analytic modeling of gossip-based message dissemination schemes under the assumption of uniform selection of multiple neighbors over the entire distributed network. The gossip-based information dissemination process (with $N+1$ nodes in the network) was described as an $(N+1)$-state MC, and the transition probabilities were derived by using a combinatorial approach. The level of cooperation by the nodes selected as the gossiping-targets was incorporated in the model. From the results, several practical performance metrics of interest and important design parameters were also obtained. For instance, the speed (in gossiping rounds) of the dissemination process required to achieve certain percentage of network coverage with a minimum probability was derived and evaluated.

The smart selection algorithm is, in nature, more effective than the blind selection scheme when disseminating content. By using the exact analysis, we have compared the performance difference of the two proposed algorithms quantitatively. For instance, to inform the entire network with certain QoS stringency, the smart selection scheme only needs half of the gossiping rounds compared with the blind selection algorithm. By increasing the cooperation probability from $\beta = 0.2$ to $\beta = 1.0$, the mean number of rounds to inform the entire network decreases logarithmically with the same slope for different network sizes, and for both the blind and the smart selection algorithm. Our results about content search also suggest that when a certain speed (number of rounds) is desirable to discover some content, it is less costly for the search process to try to place more content replications l in the network, instead of trying to hit content residing in some nodes only by increasing the number of gossiping-targets k, contacted in each round. The effectiveness of the searching algorithm is impaired by a lower cooperation probability, whereas no significant amount of overhead $(\bar{Y}_{N+1}(l))$ is generated. In view of the trade-off between the overhead and the effectiveness of the search process, the smart selection scheme is more effective with small cooperation probability. With larger cooperation probability, the smart selection scheme is less preferable during the search process, because it incurs more overhead, whereas achieves comparable effectiveness with the blind selection scheme.

Last but not least, we have investigated the pdf of accessing the nearest peer both in hopcount and delay. We obtained the hopcount and delay distribution to the most nearby peer on the URT by assigning regular i.i.d. link weights on a dense graph. Both results can be used to estimate the number of peers needed to offer certain requirement on the delivering service. And both results suggest that a small peer group is sufficient to offer an acceptable quality of service. We have also performed experiments on measuring the degree and hopcount distribution in the Internet, from which we show the applicability of the URT model, and consequently the usability of the pdfs for both hopcount and delay. With a small group of peers ($m \leq 50$), the URT seems to be a reasonably good model for a P2P network.

Part II

Empirical Study: A Proprietary P2PTV Network

Chapter 6

Introduction

The success of P2P file-sharing systems has spurred the deployment of P2P technologies in many other bandwidth-intensive large-scale applications. *Peer-to-Peer Television* (P2PTV) has become a popular mean of streaming audio and video content over the Internet. The term of P2PTV refers to P2P software applications that are designed to distribute video content in real time. The video streams to be disseminated can be self-generated video content, or TV channels provided by different TV stations. Example applications are CoolStreaming [130], TVAnts [33], TVU [34], SopCast [31], etc.

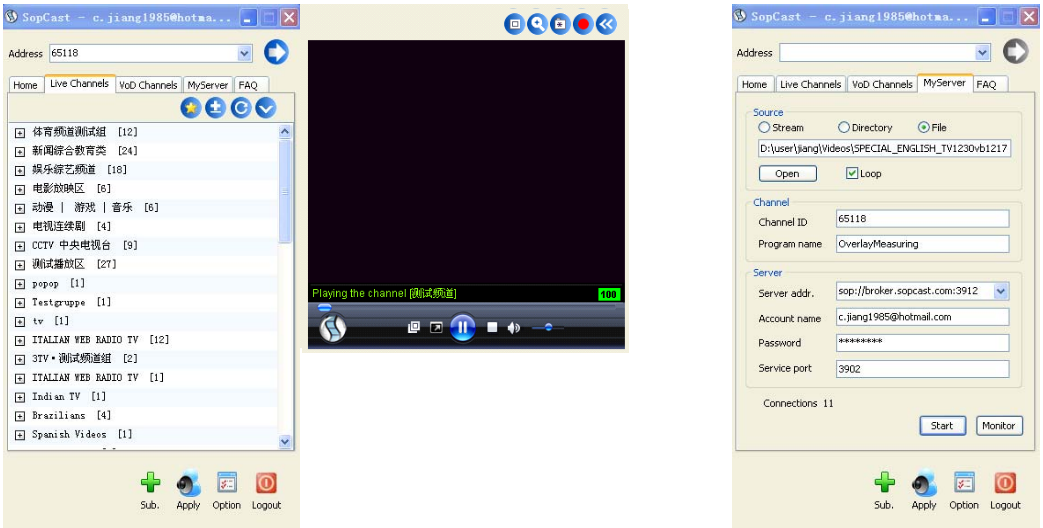
However, P2PTV systems, such as SopCast, are developed for commercial purposes: thus, very little is known about their architectures. The methodology that video content is propagated in these networks, is unknown as well. Since these proprietary P2PTV systems have attracted the interests of large amount of users, it is important to evaluate the video delivery pattern and traffic impact of such applications. An empirical study, performed with the SopCast proprietary system will be presented in this part.

6.1 What is SopCast?

The SopCast P2PTV system is one of the most popular P2PTV applications nowadays. It provides a variety of TV channels that people can watch online. Users can also create and broadcast their own video content by using SopCast. As claimed by the SopCast development team, billions of users have experienced SopCast since 2004 (when the first SopCast version was released) till now. The SopCast user interface is shown in Fig. 6.1, in which a peer is connected to a dedicated TV channel published by a *source provider*[1] (SP). Once the SP establishes a public TV channel, it will appear on the user interface at the SopCast so that people can click and start watching. Registration of a private channel is also possible in SopCast. In this case, the TV channel will not be

[1]The source provider is the node who owns and broadcasts the entire video by using SopCast software.

public, and it is only accessible by the viewers who are aware of the channel ID.



SopCast Peer

Source Provider

Figure 6.1: Left window: the SopCast interface at the peer side. Right window: the interface at the SP.

SopCast is proprietary and consequently the SopCast website do not provide much information about its underlying mechanism. Treating SopCast as a black box, we perform a set of experiments that are suitable to analyze SopCast in depth. The main contribution of the empirical study is a comprehensive understanding of the SopCast system regarding the following three issues.

- *Disclosing the operational mechanism and video propagating pattern in SopCast.*
- *Evaluating the topology and traffic dynamics in the SopCast overlay.*

6.2 Related Work and Research Challenges

There are a number of measurement studies performed with various P2P applications. These measurement analyses focus on studying the network topology, e.g. for Gnutella [87], [115], Kazaa [88], and UUSee, [129]; analyzing user behaviors, peer dynamics and content sharing properties, e.g. in Gnutella [40], [114], Kazaa [67], eDonkey [69], PPLive [70] and BitTorrent [105]; and characterizing the P2P traffic patterns, e.g. in P2PTV and VoD networks [42], [112], [132], VoIP systems [66] and file-sharing networks [110]. In the following, we review several papers that are well known in the field of measuring commercial P2PTV systems.

6.2.1 Related Measurement Studies Performed with Commercial P2PTV Applications

Hei *et al.* [70] carried out a measurement study of PPLive [26]. They collected a global view of the entire PPLive network to analyze peer behaviors, and also evaluated traffic characteristics measured at two residential and two campus users. The PPLive user behavior, e.g. peer arrival/departure pattern, user geographic distribution, etc. and traffic characteristics, e.g. streaming rates, start-up Delay, etc. were analyzed in this study. A heuristic criterion was used in [70] to separate different traffic generated in PPLive. Data packets, that are larger than or equal to 1200 Bytes, are referred to as video packets. Otherwise, a packet is assumed to carry control information. Furthermore, a video connection is assumed to be established if there are more than 10 large packets ($\geqslant$ 1200 Bytes) being transmitted consecutively; otherwise, it is referred to as a a connection with control messages.

Wu *et al.* [129] performed a large-scale measurement to reflect the time-varying topologies in UUSee [36]. By collaborating with the UUSee Inc., more accurate global topology snapshots in a period of two months are obtained in [129]. Their study has shown that the UUSee network does not have a power-law degree distributions, and the streaming topologies evolve into clusters inside each internet service provider (ISP).

In [112], Silverston *et al.* analyzed and compared the traffic patterns and underlying mechanisms among four popular P2PTV applications, namely, PPLive, PPStream [27], SopCast, and TVAnts. Silverston *et al.* used the heuristic, which is previously proposed in [70], to separate different traffic generated in the P2PTV applications. Based on this heuristic, different traffic pattern, generated in different applications, are evaluated and compared.

In addition, Ali *et al.* [42] evaluated the performance of both PPLive and Sopcast regarding the control traffic, resource usage, locality, and stability of the two P2P streaming protocols. In [124], the degree and clustering coefficient are evaluated for PPLive based on a portion of the peers that have been crawled in their measurement.

Next, we focus on the experiments performed with SopCast.

Ali *et al.* [42] presented an analysis for both PPLive and Sopcast. They conducted experiments on a single host joining a system, and collected packet traces for this case. The systems were run under different environments, and the collected data was further analyzed to give insight into SopCast's operation.

The experiment performed by Silverston *et al.* [112] is based on a single measurement day where two soccer games were scheduled. The data set collected in this paper is from two personal computers that are directly connected to the Internet from their campus network.

Sentinelli *et al.* [111] performed two sets of experiments on SopCast, trying to propose guidelines for large-scale P2PTV deployment. There were totally 27 peers connected to a popular SopCast channel in the first experiment, while the second experiment was run on the PlanetLab network, with 1 node acting as the source to broadcast their own channel, and several nodes performing as peers.

6.2.2 Research Challenges

Most of the previous work, preformed with SopCast, was executed from a single point of observation [112], or from a small number of vantage points [42], [111]. Thus, results from these papers cannot really reflect the performance of the entire SopCast network.

Moreover, these papers highlighted the SopCast mechanism with very general descriptions. Some papers claim that SopCast is based on similar principles as those underlying CoolStreaming, e.g. [107], some refer to it as a BitTorrent-based P2PTV system, e.g. [111], but all without substantiating their claims. Only in [42], Ali *et al.* attempted to unveil the SopCast protocol. However, we feel their findings are not thoroughly substantiated. For instance, control and data traffic are still separated based on their packet lengths, while the functionality of the control packets is ignored.

The above epitome of related work reveals two major challenges in understanding the SopCast P2PTV system:

1. Set up appropriate experimental settings so that the overall performance of SopCast can be reflected.

2. Perform proper analysis on the log files that have been captured during the experiments so that an accurate investigation about the SopCast network can be derived.

In the sequel, we describe our approaches to set up measurements, and provide solutions to examine the characteristics of the SopCast network.

Chapter 7

Experiments and Methodologies

As mentioned in the previous chapter, designing proper experiment is crucial in reflecting the global performance of the SopCast network. Without the source code, we have to treat SopCast as a black box, and dissect the SopCast protocol from the obtained data files. Furthermore, SopCast traffic is encoded, which makes understanding the protocol even more challenging. In this chapter, we describe our measurement settings, and provide our methodology of analyzing and understanding the operational mechanisms implemented in SopCast.

7.1 Experimental Settings

In our measurement, we have used PlanetLab [25] to emulate the SopCast system in a closed environment. Performing experiments on PlanetLab provides us the advantage to evaluate the entire constructed overlay. The experiment consists of two types of nodes:

1. A standard personal computer located in our campus network, which acts as the SP. With the SP, we registered a dedicated private channel to the SopCast network. In this channel, a small cartoon movie with a duration of 2 minutes and size of 3.5 MBytes is continuously broadcast in a loop. Thus, our experiment resembles a streaming system. The SP runs Windows XP. It is equipped with an Intel Pentium 2.4 GHz processor, 512 MB RAM and a 100 FastEthernet network interface, which is further connected through a router to the Internet. Traffic monitoring and collection at the SP is accomplished with Ethereal [101].

2. The second type of nodes are PlanetLab nodes that act as SopCast peers viewing the TV channel released by us. Each of the PlanetLab nodes under consideration runs the following software: (1) SopCast Client (Linux version), with command

line control; (2) Tcpdump [32] to enable passive monitoring on the traffic transmitted at the SopCast peers; and (3) Perl [24] Scripts to remotely control the PlanetLab nodes.

We carry out three sets of experiments with different network sizes that are designed to investigate different characteristics of the SopCast network. The experimental settings are described as follows.

Experiment 1 The first experiment was performed in August 2008, with 50 PlanetLab nodes as SopCast peers. In this experiment, the 50 PlanetLab nodes were controlled in such a way that they joined and left the SopCast network simultaneously. The first experiment is a fundamental setting which allows us to dissect the underlying mechanism implemented in SopCast, and to evaluate its topological properties. In the first experiment, 45 nodes succeeded in preserving a complete[1] log file.

Experiment 2 The second experiment was carried out with 194 PlanetLab nodes in October 2009. It reflects a more realistic scenario, in which the PlanetLab nodes joined the SopCast network according to a continuous Poisson process with the probability that exactly k events occur during a time interval of length t equals to

$$\Pr\left[X(t+s) - X(s) = k\right] = \frac{(\lambda t)^k \, e^{-\lambda t}}{k!} \tag{7.1}$$

where $X(t)$ is the number of PlanetLab nodes joined in the time interval of length t, and the rate[2] of the Poisson process is set to $\lambda = 360/hour$. The PlanetLab nodes were chosen randomly from different sites. For each of the PlanetLab nodes, we generated an exponentially distributed random variable T (with rate λ and mean $\frac{1}{\lambda}$) that represents the time at which the node will join the SopCast network. Out of the 194 PlanetLab nodes in experiment 2, 153 nodes succeeded in participating in the entire experiment.

Experiment 3 The third experiment was carried out in December 2009 with 441 PlanetLab nodes. The purpose is to compare the result obtained from experiment 2 with a larger network size, so that the scalability of SopCast can be evaluated. Peer joining process in this experiment followed the Poisson process in (7.1), and the rate of the Poisson process was $\lambda = 900/hour$. Out of the 441 nodes, 273 nodes succeeded in participating in the entire experiment. Experiment 2 and 3 are designed to reflect the traffic dynamics of the SopCast network load.

[1]A complete log file indicates that the node joins the network at the begining of the experiment and leaves at the end of the measurement.

[2]By setting the rate of the Poisson process to $\lambda = 360/hour$, all peers will join the network within approximately 30 minutes. We let the experiment run for another 10 minutes, so that the performance of each SopCast peer becomes stable after entering the network.

The duration of the three experiments were 40 minutes for each. The collected traffic log files at the SP and the PlanetLab peers are further processed and analyzed with AWK [2] and Java [16] scripts developed by us. The anomaly of node failure in the three experiments are caused by the bandwidth [3] and disk space limitations on the PlanetLab nodes in question. As a result, nodes may quit the experiment unexpectedly since they are not allowed to transfer video content, or the data traffic sniffed by tcpdump cannot be written on their local disks.

During the experiments, each PlanetLab node is identified by its distinct host name and IP (Internet Protocol) address. The log files captured at the PlanetLab nodes are saved after their host names, e.g. *log_planetlab1.ewi.tudelft.nl*. If it is not specified, we analyze the data traffic, captured in experiment 1 in Chapter 7, Section 8.2, and 8.3. The log files, collected from experiment 2 and 3 are used to disclose traffic dynamics in Section 8.4.

7.2 Dissecting the SopCast Protocol

To dissect the SopCast protocol by reverse engineering, we perform an extensive analysis on the tcpdump files that are collected from every PlanetLab node involved in the experiments. An example of a tcpdump log format is given in Fig. 7.1. The first column specifies the current clock time at each PlanetLab node, followed by the network layer protocol, the IP address and port number of the sender, and the IP address and port number of the receiver. In the last two columns, we may find the transport layer protocol and the length of the packet.

```
11:03:53.849221 IP 137.226.138.154.3908 > 128.112.139.96.3908 : UDP, length 46
11:03:53.849291 IP 128.112.139.96.3908 > 137.226.138.154.3908 : UDP, length 28
11:03:53.852812 IP 128.112.139.96.3908 > 137.226.138.154.3908: UDP, length 42
11:03:53.852894 IP 137.226.138.154.3908 > 128.112.139.96.3908: UDP, length 28
11:03:53.853395 IP 128.112.139.96.3908 > 137.226.138.154.3908: UDP, length 1320
11:03:53.853649 IP 137.226.138.154.3908 > 128.112.139.96.3908: UDP, length 28
11:03:53.856165 IP 128.112.139.96.3908 > 137.226.138.154.3908: UDP, length 1320
11:03:53.856234 IP 137.226.138.154.3908 > 128.112.139.96.3908: UDP, length 28
```

Figure 7.1: An example of standard tcpdump output format.

Tcpdump reveals that SopCast relies on UDP (User Datagram Protocol). To figure out the packet functionalities, we studied the packet lengths and the corresponding delivery patterns. Although the work presented in this chapter has been conducted in a thorough and careful way, without having the source code of SopCast, the claims

[3] The bandwith of a PlanetLab node can be limited by the PlanetLab administrator.

that we have made are based on our investigation of SopCast. They are not the exact description of the protocol.

7.2.1 Neighbor Communication in SopCast

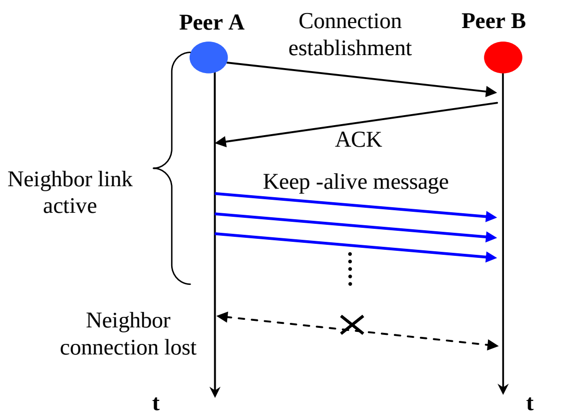
Communication between two peers in SopCast is always initiated by a 52 − 80 byte packets pair, e.g. peer 152.14.92.59 and peer 169.229.50.14 in Fig. 7.2 (a). Once the connection is established, the pair of peers keeps exchanging a sequence of 42-byte packets with each other. The 42-byte packets are transmitted with a high frequency, roughly every second. We denote this packet as a *keep-alive* packet. With the keep-alive packets, a peer announces its existence in the network. The purpose is to accommodate overlay dynamics, and maintain neighbor relation with others via the decentralized communication between peers. If two peers exchange a 42-byte control packets with each other, we refer to these peers as being *neighbors*. Moreover, the UDP data seems to be encoded, since we have not found a distinct structure of the UDP data field. Without the source code, we cannot tell what are the exact message contained in the 42-byte packet.

In Fig. 7.2 (b), a graphic illustration of the neighbor communication scheme is presented. We see that, peers can lose neighbors. In case a neighbor does not respond to the keep-alive packets, the peer stops contacting this neighbor until it chooses the neighbor again. There are also some other packets that we do not understand completely. For instance, the 74-byte packet and the 958-byte packet appear from time to time, in Fig. 7.2 (a), but with no regular transmission pattern.

Communication between two neighbors

Time (s)	Size (bytes)	Source IP	Destination IP
60.121599	52	152.14.92.59	169.229.50.14
60.197511	80	169.229.50.14	152.14.92.59
60.199382	74	152.14.92.59	169.229.50.14
60.279217	28	169.229.50.14	152.14.92.59
60.738697	42	152.14.92.59	169.229.50.14
60.814576	28	169.229.50.14	152.14.92.59
61.512937	42	152.14.92.59	169.229.50.14
61.588690	28	169.229.50.14	152.14.92.59
		…………	
723.980571	42	152.14.92.59	169.229.50.14
724.057957	28	169.229.50.14	152.14.92.59
724.759458	958	169.229.50.14	152.14.92.59
724.759537	28	152.14.92.59	169.229.50.14

(a) An example of tcpdump log file



(b) A graphic illustration of peer communication pattern

Figure 7.2: (a) An example of tcpdump file after proper formatting. (b) A graphic illustration of SopCast neighbor communication pattern.

7.2.2 Video Delivery in SopCast

In every captured log file, we notice a fixed traffic pattern: a sequence of 1320 bytes packets are frequently transmitted between peers (or between peers and the SP), preceded by a control packet with size 46 bytes. Often, a smaller packet with the size of several hundred bytes or a thousand bytes (e.g. 1201 bytes) is observed at the end of the data sequence, see Fig. 7.3 (a). The large data packets (except for the 46-byte one) are video packets.

If a peer is providing video content to its neighbors, we refer to the peer as a *parent*. A *child* is defined if a peer is receiving video streams. We believe that the control packet of 46-byte acts as a *video request packet*. A peer in SopCast never voluntarily delivers video content to its neighbors. To download video packets, a peer always needs to request them from its neighbors via the video request packet of 46 bytes. After receiving the request, its parent(s) will deliver a series of video packets to the child. In case the child needs more video content, it sends another request.

Our analysis also reveals that video delivery in SopCast is chunk-based. The TV content in SopCast is divided into *video chunks* or *blocks* with equal sizes of 10 kbyte. Following is our methodology to derive this conclusion. We first filtered the series of 1320-byte packet and the smaller packet at the end (see Fig. 7.3 (a)) between two consecutive 46-byte packets, that are transmitted between a node pair. The aforementioned packets are referred to as a *video sequence*. Based on the filtered reports, we counted on the number of bytes transmitted in a video sequence. With UDP, reliability of data transfer is not guaranteed. Therefore, the calculation is carried out for all node pairs after filtering, and only concerns the video sequence received at the children side, because video packets received at a child have surely arrived. We found that the number of bytes in a video sequence is always approximately the integer of 10 kbytes, e.g. 20 kbytes, 30 kbytes, etc. This finding is in line with the video chopping algorithm implemented in many existing P2P systems, although the chunk size may be different[4]. A peer is allowed to have multiple parents and multiple children, and it is free to request multiple video blocks from its parents.

The requested blocks are treated as a large datagram during transmission. Due to the IP fragmentation principle, a large datagram such as 10 kbytes should be segmented into smaller pieces in order to pass over the Ethernet. SopCast sets the maximum size of the video packet to be transmitted to 1320 bytes. A generalization of the video block fragmentation rule is

$$n \times 10 \; kbyte = x \times 1320 \; bytes + y \tag{7.2}$$

where n is the number of requested video blocks, x is the number of 1320-byte video packets, and y is the size of the smaller fragment in bytes (377, 497, etc.). In Fig. 7.3 (b), we generalize the above findings. A parent-child relation can be established only

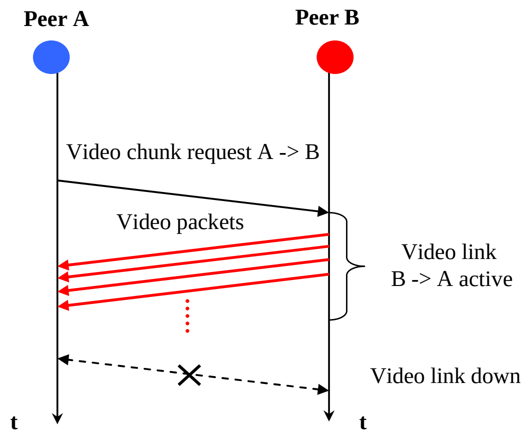
[4] In Bittorrent, the default size of the chopped block is 256 kbytes.

when two peers are neighbors.

Video delivery between two peers

Time (s)	Size (bytes)	Source IP	Destination IP
61.825380	46	152.14.92.59	169.229.50.14
61.825623	28	169.229.50.14	152.14.92.59
61.826114	1320	169.229.50.14	152.14.92.59
61.826166	28	152.14.92.59	169.229.50.14
61.826244	1320	169.229.50.14	152.14.92.59
61.826289	28	152.14.92.59	169.229.50.14
		………………...	
61.830601	1320	169.229.50.14	152.14.92.59
61.830678	28	152.14.92.59	169.229.50.14
61.830744	1320	169.229.50.14	152.14.92.59
61.830812	28	152.14.92.59	169.229.50.14
61.831305	1201	169.229.50.14	152.14.92.59
61.831373	28	152.14.92.59	169.229.50.14

(a) An example of tcpdump log file



(b) A graphic illustration of video delivery pattern

Figure 7.3: (a) An example of tcpdump file after proper formatting. (b) A graphic illustration of SopCast video delivery pattern.

7.2.3 Identification of SopCast Packets

Based on the analyses in Section 7.2.1 and 7.2.2, we present the identification of SopCast packets in Table 7.1. Apart from the 80-byte dedicated replying message, we think packets with size of 28 bytes are used to acknowledge general packets, except for the HELLO messages. We base this assumption on the fact that the 28-byte packets are observed immediately after the reception of a video packet, or a control packet.

Table 7.1: Identification of SopCast Packets

Type	Size (bytes)	Functionality
Video packet	1320	Maximum size of the video packets
	377, 497, 617, 1081, 1201	Video fragments
Control packet	52	HELLO packet to initiate link connections
	80	Confirmation on receiving the HELLO packet
	28	Acknowledgement
	42	Keep-alive message with neighbors
	46	Video requesting packet

7.2.4 The Three-tier Structure in SopCast

Another interesting observation from our experiment is that: SopCast has limited the number of children connected to the SP for video downloading. During experiment 1, only 13 nodes downloaded video blocks directly from the SP. The rest peers download video content from the 13 nodes, or between themselves.

Consequently, regarding the video delivery pattern, we can see SopCast as a three-tier hierarchical structure, see Fig. 7.4. The SP, which is the node who owns the entire video and broadcasts it by using SopCast software, delivers the video content to a limited number of peers in the SopCast network. The peers, who receive video content from the SP directly, are referred to as the *first tier peers*. The other peers that never retrieve video content from the SP are called *second tier peers*. They download video packets from the first tier peers or exchange video content with each other.

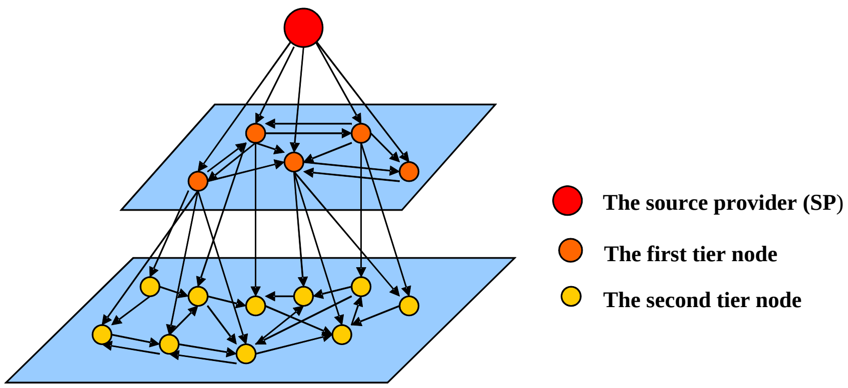


Figure 7.4: A graphic illustration of the SopCast video delivery structure (top is the SP, middle refers to first tier nodes, and bottom to second tier nodes).

In the sequel, we present our methodology to unveil the criteria of selecting the first tier peers in SopCast.

7.2.5 First Tier Peer Selection Strategy

To dissect the scheme of selecting peers in the first tier, we consider four different factors that may be critical in P2PTV systems, namely the available bandwidth, preference of node physical distance (node locality), the network distance (the hopcount), and node distance in time (the delay).

Estimating the bandwidth usage on PlanetLab nodes

Bandwidth usage on PlanetLab is actively controlled through a policy of *fair share access* [3]. By default, each slice[5] is granted one *share* of the available bandwidth. The administrative/root slice can be granted more than one share. Assuming that there are n slices competing for the outgoing bandwidth of B MBps on a PlanetLab node, each slice is given x shares of the bandwidth, and the root slice is granted y shares ($y \geqslant x$). The available bandwidth that the root slice can receive is thus given by $B_{root} = \frac{By}{nx+y}$, and the bandwidth that a normal slice can use is

$$B_{normal} = \frac{B - B_{root}}{n} \tag{7.3}$$

Slices that are operating on a PlanetLab node can be up and down from time to time. By using the user interface of CoMon [6], [102], we downloaded three data files during the period that we were performing the three experiments. Each of the three files contains the statistical information about the status of the PlanetLab nodes, including node name, IP address, CPU and memory consumption, number of active slices, geographical location, etc. We have computed the average number of active slices during the experiment period from the collected data files. We also ran a Python script [28] on every PlanetLab node that is used in our experiments. The Python script gives us the actual bandwidth shares for all the slices (including the root slice) that have registered on a particular PlanetLab node. Together with the average number of slices on a PlanetLab node provided by CoMon, (7.3) is used to estimate the approximate bandwidth that a node can utilize during our measurements.

Measuring the preference on node distance

We have used *traceroute* to get the number of hops and delay of a node from the SP. We start the traceroute when the experiments on SopCast begin. The SP runs traceroute to all the PlanetLab nodes in parallel. We have collected thousands samples of the returned paths. The average hopcount and delay is computed from the traceroute data for all the nodes. The CoMon data also provides the geographic location (country level) of the PlanetLab nodes. We used this information to determine the preference of selecting first tier peers based on node location.

The first tier peer selection strategy in SopCast

Fig. 7.5 illustrates the aforementioned four criteria on selecting first tier peers from experiment 1. To make the figures more readable, we assign each peer a unique peer ID

[5] A slice is a set of allocated resources distributed across different PlanetLab nodes. The allocated resources, e.g. computational capacity, CPU usage and bandwidth, of a PlanetLab node are shared by several slices.

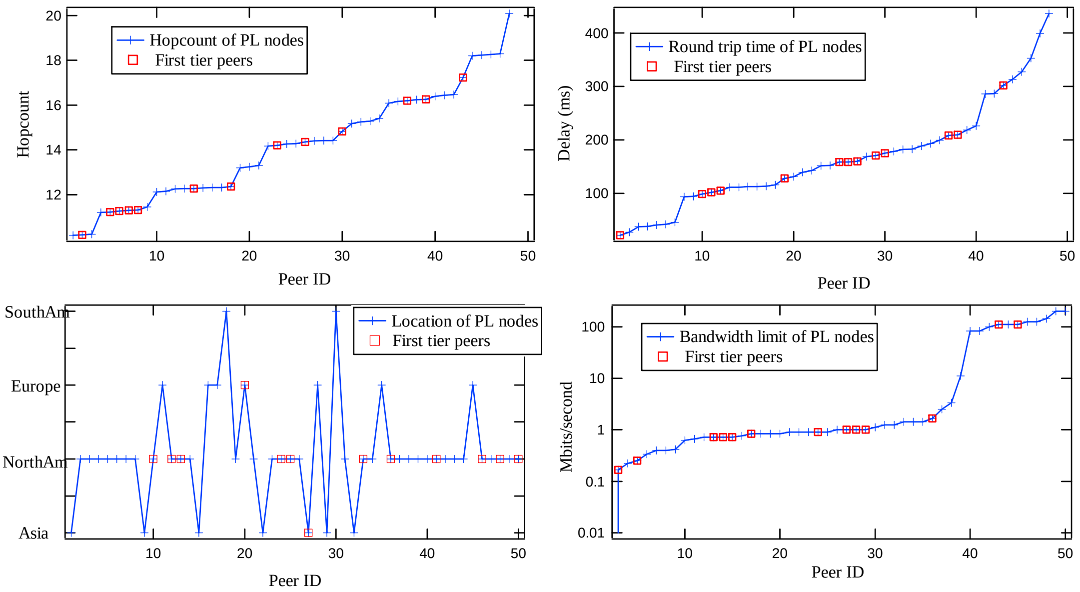


Figure 7.5: The influence of node distance (i.e., geographic distance, network distance, and distance in time) and the available bandwidth on the selection of first tier peers in experiment 1. Peer IDs are sorted in ascending orders for the four metrics (except for peer location), and thus are different in the four sub-figures.

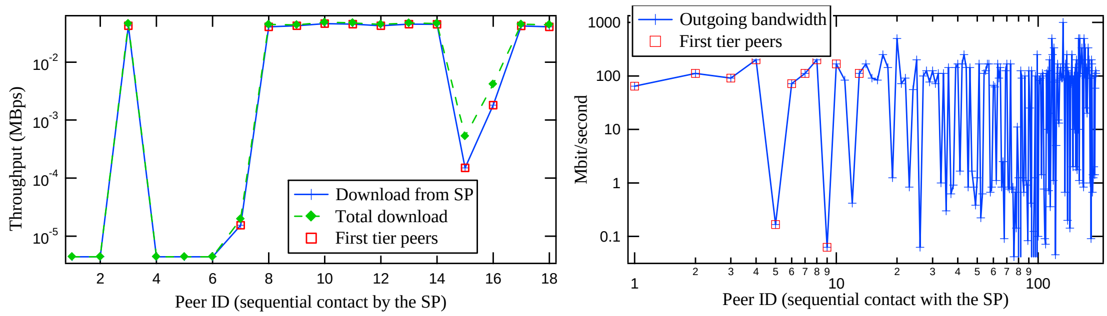
i $(1 \leq i \leq N)$, where N is the number of PlanetLab nodes involved in the experiment. Instead of presenting the host names of the PlanetLab nodes on the horizontal axis, we show the peer IDs. From the top two sub-figures in Fig. 7.5, we see that peers with smaller hopcount or delay also appear in the first tier, which suggests that the hopcount and delay are not crucial factors in selecting the first tier peers. Furthermore, although the SP is located in Europe, it does not choose peers that are geographically closer to it. Finally, the available bandwidth usage also seems to be irrelevant. Peers that have high bandwidth, do not always appear in the first tier.

Since none of the factors determines the selection of peers in SopCast, we decided to study the joining process of each peer. We have noticed from the experiment that, after a peer enters the SopCast network, the SP starts to communicate with it proactively by sending out HELLO packets. Once a peer replies to a HELLO packet, the SP uploads video content to the corresponding peer. The SP only contacts a limited number of peers. Fig, 7.6 (a) presents the influence of the *registration process*[6] of a peer with the SP from experiment 1. Although peers join the network at the same time, due

[6]We examine the registration process from the log file of each individual node. Notice that the local time of Planetlab nodes is synchronized to *eastern standard time* (EST) time. We use the CoMon data to check the clock drift at each Planetlab node. For those nodes whose clock is not synchronized, we compute the correct EST time on that node.

to the non-negligible processing delay (in seconds), the SP contacts the SopCast peers sequentially in time. As indicated in Fig. 7.6 (a), the SP contacted several peers, but they did not establish video delivery. This is because these peers did not respond to the SP: we did not observe any replying packets that the peers have sent back to the SP. The peers that have replied to the SP and become first tier peers, in general, stayed active during the entire experiment (except for three peers that quit the experiment unexpectedly). These three peers, thus, only download a small amount of video content.

We also examine the first tier peer selection scheme in experiment 2. The result confirms the *first-come-first-serve* pattern that has been found in experiment 1. As shown in Fig. 7.6 (b), since peers join the network sequentially, the first 11 peers become the first tier peers (regardless of their outgoing bandwidth). The two peers that did not respond to the SP is not selected. We noticed that the number of selected first tier peers are different in experiment 1 and 2. It might be caused by the fact that SopCast has re-selected the first tier peers because three of them quit the first experiment. In both experiments, once a first tier peer is selected, it remains being connected with the SP during the entire experimental period, unless it leaves the experiment by itself.



(a) Experiment 1

(b) Experiment 2

Figure 7.6: The influence of peer registration procedure with the SP in experiment 1 (a) and 2 (b). There are 13 first tier peers and 11 first tier peers in experiments 1 and 2 respectively. Peers that do not respond to the request from the SP do not appear in the first tier.

7.3 Summary of SopCast protocol

The analyses in Sections 7.2.1 and Section 7.2.2 highlight the basic operations in SopCast. In general, SopCast is a decentralized P2PTV network with a dedicated server for peer membership registration and peer discovery during the bootstrapping period (i.e. finding other peers when a peer joins the SopCast network for the first time). After that, peer discovery and communication are completely decentralized. To summarize, the peer communication, video delivery pattern and the network structure that have been discovered from the study are:

- SopCast traffic can be categorized in control packets and video packets. The control packets fulfil different functionalities, which coordinate the operation of the SopCast application. The video packets deliver the TV content.

- Communication between SopCast peers is decentralized. A peer announces its existence and maintains connection with its neighbors by frequently exchanging keep-alive packets.

- Video delivery in SopCast is chunk-based. Each video chunk has equal length of 10 kbytes. A peer is free to request multiple blocks from its parent(s).

- SopCast can be seen as a three-tier hierarchical structure regarding video delivery. Only a limited number of children can be connected to the SP directly, i.e. the first tier peers. The second tier peers download video content from the peers in the first tier, or between themselves.

- It seems that SopCast does not employ a sophisticated algorithm to select the first tier peers. The first tier peers in the SopCast network are selected based on a first-come-first-serve pattern regardless of their physical distance, network distance, node distance in time, and their upload bandwidth.

Our approaches in analyzing the SopCast mechanism reveal important design insights, and help to better understand similar P2PTV applications.

Chapter 8

Topological and Traffic Dynamics in SopCast

The topology of the SopCast overlay network can be generated with peers as nodes and connections as links. The SopCast overlay is constructed with decentralized peers. The SP operates differently from the peers, because it has the entire video content and only supports a number of children for video downloading. Therefore, we do not consider the SP as a decentralized peer. The SopCast network is a highly dynamic system, because communication links between peers can be up and down from time to time. In this chapter, we aim to investigate the dynamic nature of the SopCast overlay. Furthermore, we have discussed in Section 7.2.4 that SopCast is structured as a three-tier overlay network, in which only a limited number of peers is connected to the SP directly. We are also interested in finding out how the network load is dynamically distributed over peers in different tiers.

8.1 A Two-layer SopCast Overlay

By using the findings discovered in Section 7.2.1 and 7.2.2, we study the SopCast overlay as a two-layer architecture consisting of the *neighbor graph* G_N and the *video graph* G_V. Both graphs are formed with directed links. We define a peer as a node that is active in both downloading and uploading processes. The SP is not included in the two layers, because it only supports a number of children with video downloading. Notice that not all neighbors will share video streams with each other. A straightforward relation between the two layers is $G_V \subseteq G_N$.

- Neighbor graph G_N: The neighbor graph is formed by a set of nodes with neighbor relations. A link in the graph is denoted as a *neighbor link*. A neighbor link is established if a node pair is regularly sending keep-alive messages. If the keep-alive messages are not observed for some time, the link is considered to be inactive.

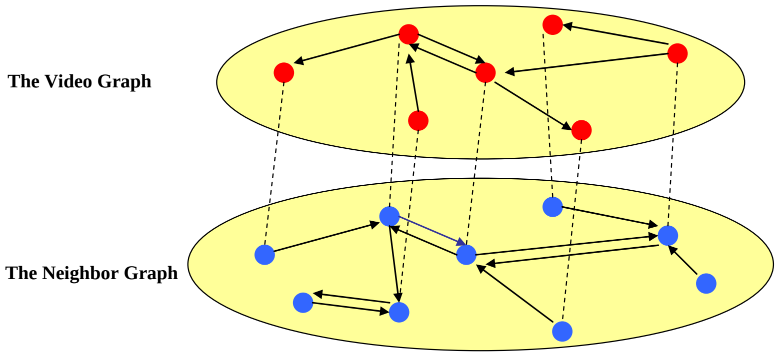


Figure 8.1: A graphical illustration of the two-layer SopCast overlay.

The outgoing degree D_{out} of G_N defines the number of neighbors that a random peer has in the network.

- Video graph G_V: The video graph consists of peers that are transmitting video blocks. The incoming degree D_{in} of the video graph indicates the number of parents that a random peer has. The outgoing degree determines the number of children that an arbitrary peer supports. A *video link* is activated if there are video packets being transmitted from the parent to the child.

8.2 Reflecting the Dynamic Overlay

8.2.1 Defining a Proper Network Snapshot Duration

A commonly used method to reflect the changing topology is to take network snapshots at sequential time points and show them as a time series. A *snapshot* captures all participating peers and their connections at a particular time, from which a graph can be generated. In fact, characterizing dynamic peer-to-peer networks has been considered a research field with a lot of contentions. The major dispute is on the completeness and accuracy of the captured topology snapshots in characterizing the actual P2P overlay, as addressed in [115].

Measuring SopCast dynamics over PlanetLab nodes gives us the advantage to take unbiased[1] snapshots. Thus, completeness of the snapshots is not our major concern. In a high churning network, such as SopCast, we are not interested in taking *instantaneous* snapshots. Because an instantaneous snapshot is taken at an exact time point (in the order of microseconds), thus capturing only a few nodes and links.

[1]By unbiased snapshot, we mean that we can control all nodes participating the experiment, and that the topology can reflect the complete SopCast overlay. When measuring a large-scale P2P network with crawlers, the snapshot may only capture a portion of the overlay, resulting in a biased network.

We study the network dynamics by continuously taking network snapshots with the *duration* τ as time evolves and show them as a time series. The snapshot duration may have minor effects on analyzing slowly changing networks, such as Internet on Autonomous System (AS) level topologies. However, in a P2P streaming system, the characteristics of the network topology vary greatly with respect to the time scale of the snapshot duration. Therefore, defining a proper network snapshot duration is the most imperative task that we have to solve in the first place.

8.2.2 The Activation Period of a Neighbor/Video Link

The SopCast P2PTV system does not share its source code, so investigating whether there are commands/packets to set up or tear down a link is not possible. As an alternative, we look at whether, during some time interval, traffic is flowing between two peers. Hence, we define an *active* (neighbor or video) link if two peers in SopCast are continuously transmitting keep-alive messages, or video packets. If no traffic is flowing, we say that the link is closed. Often, we could notice temporary ceasing of the connection between two peers. After a period which ranges from several seconds to hundred seconds, they contact each other again. If a snapshot of the SopCast network is taken with a long duration, the links that are not no long active can be captured, and thus incurs errors in the obtained graph. Moreover, with a long snapshot duration, the changes of link connection cannot be reflected. Therefore, defining a proper network snapshot is essentially to determine the *activation period* of the (neighbor or video) link.

In order to determine the activation period of a neighbor link in SopCast, we processed the traces and obtained the time interval between two consecutive keep-alive messages between all node pairs. From the parsed data set, the corresponding pdf is plotted in Fig. 8.2 (left figure). This statistical analysis suggests that, with high probability (more than 95% of the cases), a peer sends two consecutive keep-alive packets to its neighbors within 2.5 seconds. Thus, we consider the threshold of $\tau_N \sim 2.5$ s as the activation period of a neighbor link. If two peers have not contacted each other within this interval, termination of a link connection is assumed. With the same approach, the pdf of the time interval between two consecutive video requests for all node pairs is plotted. Indicated in Fig. 8.2 (right figure), the activation period of a video link is around $\tau_V \sim 2.0$ s. If a child A requests video blocks from its parent B within 2.0 s from the previous request, the video link is considered to be active.

The activation period for a neighbor, or video link is further used as the duration to capture network snapshots of the SopCast overlay, which will be explained in more detail in the following sections.

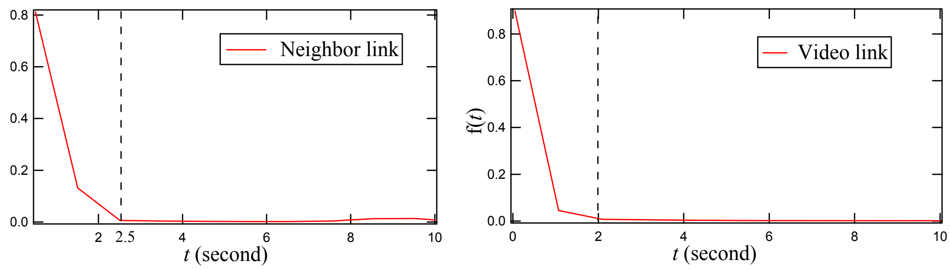


Figure 8.2: Pdf of the activation period of the neighbor link (left) and the video link (right). Threshold of activation period for the neighbor link is $\tau_N \sim 2.5$ s, and$\tau_V \sim 2.0$ s for the video link.

8.3 Dynamics of the SopCast Topology

8.3.1 Topology Evolution of the Neighbor Graph

SopCast deploys certain principles to discover neighbors once a peer joins the network. A peer is consequently expected to be able to contact, at least a portion of the neighboring peers in the network. In this section, we study *how fast do SopCast peers accumulate their views of the network, and what is the characteristic of SopCast neighbor graph?*

We take snapshots of the neighbor graph during the time interval of $[0, T]$, with $T = 10$, 60, 300, 600 seconds respectively, so that all the participating peers and links that have once been active from the beginning of the experiment till the time T have been captured. The purpose is to examine all the historical neighboring connections accumulated in these snapshots.

With the captured snapshots, four directed graphs are obtained. Multiple lines between two peers are removed. We plot the pdf of the outgoing degree (D_{out}) of the four graphs in Fig. 8.3. We notice that peers discover their neighbors quite fast in SopCast. After 300 seconds, the neighbor topology already converges to a full mesh, which means that SopCast peers have the ability to find almost all the other neighbors within a relatively short period of time. The distinct property of neighbor discovery is better illustrated in Fig. 8.4, in which the evolution of the average outgoing degree of all peers is plotted as a function of time. The average outgoing degree grows rapidly during the first 5 minutes of the experiment[2]. The standard deviation of the average outgoing degree is also shown. In the first several minutes, peers tend to behave differently. Some of them contacted with more neighbors, while some only meet a few. This is the reason that a large standard deviation appears at the beginning of the experiment. As time evolves, the activity of neighbor discovery gradually converges at different peers. After

[2]Compared to the entire duration of the experiment (40 minutes), this period can be considered to be short.

5 minutes, the standard deviation has reduced substantially.

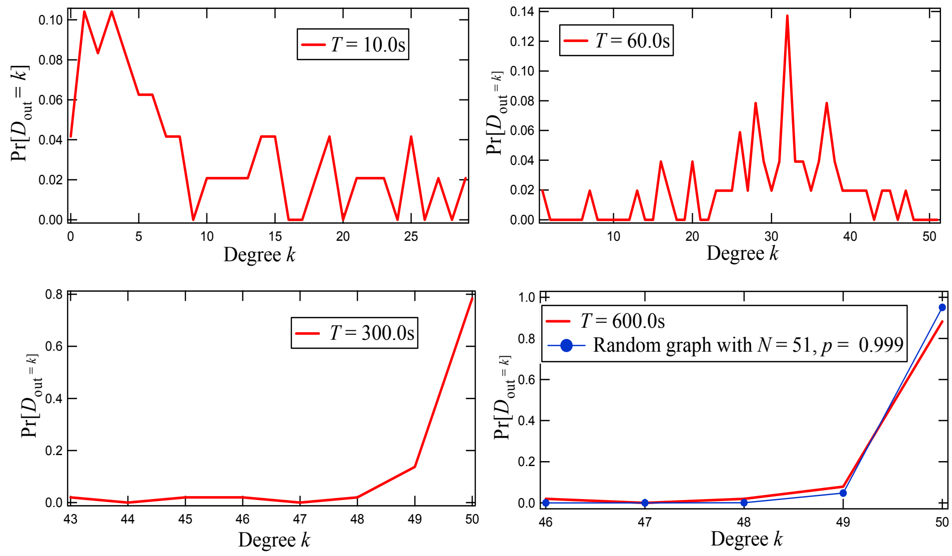


Figure 8.3: Topology evolution of the neighbor graph shown for different values of the snapshot duration. The outgoing degree defines the number of neighboring connections a node once maintained. In the bottom-right sub-figure, $Pr[D_{out} = k]$ is fitted with the degree distribution of a random graph with link density $p = 0.999$.

In Fig. 8.3, we have shown that peers in the SopCast network have the ability of accumulating their views of the network in a decentralized manner. However, whether a peer will frequently contact all the discovered neighbors is still an open question. It may happen that a peer only contacts its neighbor for one or two times, and never initiates the connection again. To answer the above question, we examine the dynamics of the neighbor graph. The activation period of a neighbor link was found to be 2.5 s in Section 8.2.2. By taking network snapshots with this interval, we could obtain some insights on the characteristic of the neighbor graph regarding the number of contacted neighbors.

The neighbor graph is examined during the time interval of [5 min, 35 min]. This is because, in the first 5 minutes, peers are trying to discover more neighbors, thus the network is not stable, see Fig. 8.3. The last 5 minutes are also excluded, because two PlanetLab nodes came offline unexpectedly. By taking snapshots every consecutive 2.5 seconds, 720 network topologies are obtained, which generate 28050 samples of the outgoing degree in total. The average number of contacted neighbors within 2.5 seconds is around 37.73. We also extended the snapshot duration to 10 seconds as a comparison. The average outgoing degree increases slightly to 41.6. This observation suggests that although a peer in the small network (i.e. $N = 50$) discovered all the other neighbors, it only contact a subset of them, e.g. around 38 neighbors within 2.5 seconds. Moreover,

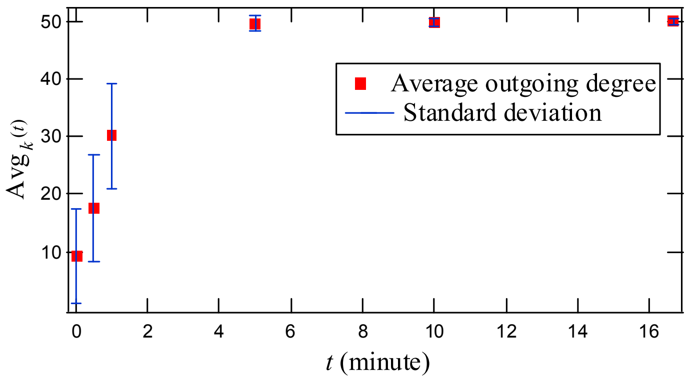


Figure 8.4: The average outgoing degree of the neighbor graphs as a function of time.

the number of neighbors that a peer keeps contacting is relatively stable. If a peer frequently changes its neighbors, the average outgoing degree of the neighbor graph would be larger than 41.60 in the longer snapshot interval of 10 s.

8.3.2 Topology Dynamics of the Video Graph

The SopCast protocol specifies a set of streaming delivery rules that are used to guide peers to actively connect to others, or allow other peers to establish connections to themselves. We aim to understand how SopCast coordinates peers to share video streams from the perspective of a network topology.

We study the video graph when peers have chances to contact with almost every neighbor in the network, e.g. after 5 minutes of the experiment in Fig. 8.3, because the network is considered to be more stable from this time. Hence, the topology of the video graph is also examined during the time interval of [5 min, 35 min].

Recall that the activation period of a video link was found to be 2.0 seconds. We divide the time period of [5 min, 35 min] to sequential time slots of 2.0 seconds, resulting in 900 network snapshots of G_V. The 900 snapshots provide us with 32320 samples of the incoming degree (D_{in}), and 27697 samples of the outgoing degree (D_{out}). The incoming and outgoing degree distributions are obtained by making the histogram of the 32320 in-degree samples and 27697 out-degree samples respectively. In Fig. 8.5, SopCast peers only retrieve video packets from, on average, 1.86 neighbors during 2.0 seconds. In rare cases, a node may contact more than 10 peers to download streams.

The outgoing degree distribution of the video graph shows different behaviors. A super peer structure emerges when looking at the curve plotted on a log-log scale in Fig. 8.6. A *super peer* in a P2P network refers to a node that offers good uploading bitrates and establishes many connections with different children. It seems that SopCast peers

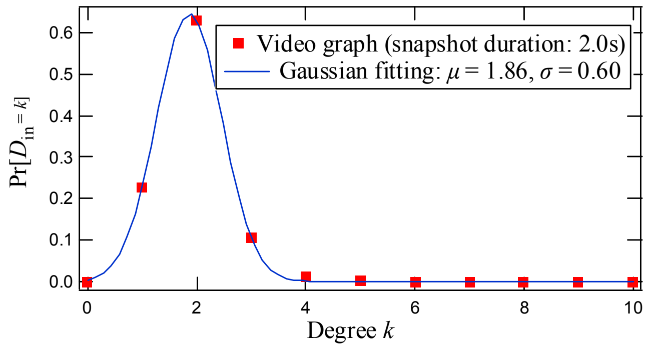


Figure 8.5: Pdf of the incoming degree (number of parents) of the video graph on a linear scale. Snapshot duration of the video graph is set to 2.0 s. A Gaussian fitting describes the pdf of D_{in} well, which suggests us to model the incoming degree of G_V as a random graph with very low link density $p \ll 1.0$.

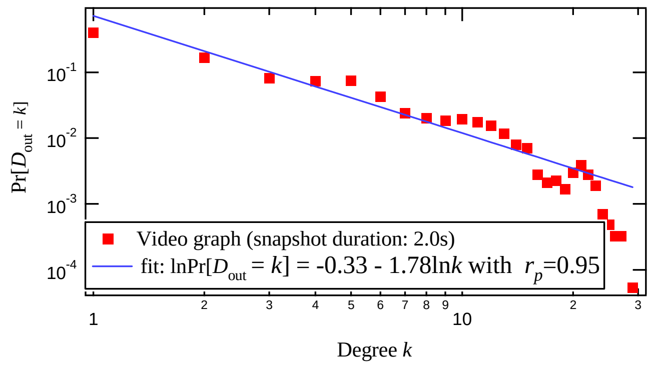


Figure 8.6: The outgoing degree (number of children) distribution of the G_V on log-log scale.

are allowed to select several parents to establish downloading connections, while they will only choose a few of them, which leads to the existence of super peers.

8.3.3 The SopCast Super Peers

In Section 8.3.2, the existence of super peers in the SopCast network is discussed in terms of node degree. In the following, we evaluate the super peer structure from the perspective of peer upload throughput.

A critical issue of a P2P system is to fairly distribute the network load to peers participating in the content delivery process. Preferably, each node should have equal responsibility to upload content to its neighbors. However, it is very likely that the so called *tit-for-tat* principle [53] is not implemented in SopCast. A peer can request video packets from one of its neighbors, without offering any video blocks to this neighbor. For instance, we have noticed constant uploading from PlanetLab node *freedom.informatik.rwth-aachen.de* to *planetlab1.pop-mg.rnp.br*, but no uploading activity in the reverse direction.

In Fig. 8.7, we present the total download and upload throughput at every single peer in this small network (50 nodes). We see that peers tend to have uniform download throughput. There are several nodes which have downloaded much less than others, which is caused by bandwidth limitations at these nodes. However, peers behave differently regarding their upload performance: a few nodes upload a significant amount of video packets to the network, whereas most peers do not contribute too much regarding their upload throughput. The peers, that upload significant amount of video content (e.g. the last 10 peers) is consequently referred to as the super peers, and the rest peers are called *normal peers*. The existence of the super peers alleviates the bandwidth burden at the SP.

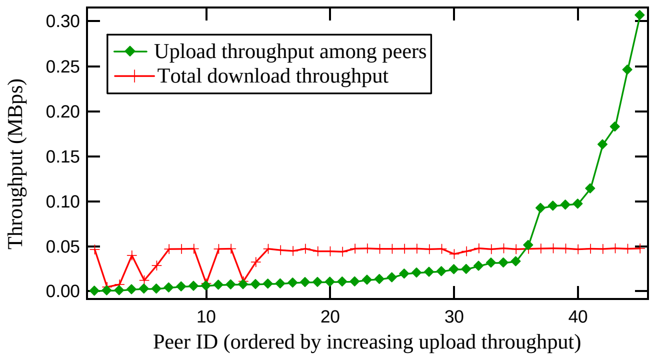


Figure 8.7: Upload and download throughput at every SopCast node during the entire experiment (40 minutes). The horizontal axis presents the ID of each individual peer after ordering by increasing upload throughput. The upload throughput (download throughput) at a single peer is calculated based on the amount of data uploads (downloads) to all the children (parents) the peer has.

By taking a series of network snapshots with a duration of 2 s, we measure the *linear correlation coefficient* between the value of the outgoing degree and the upload throughput at every single peer for all the snapshots. The linear correlation coefficient, that is used to measure the dependence between two random variables of X and Y, is defined as

$$\rho(X,Y) = \frac{E[XY] - \mu_X \mu_Y}{\sigma_X \sigma_Y} \tag{8.1}$$

where μ_X and μ_Y are the mean of X and Y respectively, and σ_X and σ_Y represent their standard deviations. The correlation coefficient ranges between -1 and 1. A positive (negative) correlation coefficient, e.g. close to 1 (or to -1), indicates that the large values of X tend to be associated with large (small) values of Y. The correlation coefficient of zero means that X and Y are not correlated.

The computed correlation coefficient of $\rho = 0.98$ indicates a strong correlation between the outgoing degree and the upload throughput of SopCast peers: a peer uploads more video blocks if it establishes many connections with its children.

8.4 Traffic Dynamics in SopCast

We have discovered in Section 7.2.4 that SopCast is structured as a three-tier overlay network. In Section 8.3.3, we have also found the existence of super peers (in terms of uploading throughput) in SopCast. The next question is: *whether the first tier peers act as the super peers that upload a lot, and how is the network load dynamically distributed over peers in different tiers*? In this section, we use the traffic data, captured in experiment 2 and 3, because these two experiments allow us to examine more realistic scenarios in which peers join the network sequentially. In particular, we are able to study to which peers the newly join node will attach for video downloading. In the experiment 2 and 3, the peer ID was assigned from 1 to N to peers based on the order in which they joined the network.

8.4.1 Traffic Dynamics as a Function of Network Growth

Peers from the first tier download most of their video content from the SP. Firstly, we study whether peers from the second tier have the same preferences of connecting to peers in the first or second tier for video downloading. In Fig. 8.8, we computed the ratio of video content downloaded from the first tier peers and second tier peers over the total amount of video content downloaded at every second tier peer. The horizontal axis starts with the ID of the second tier peers that join the network at sequential times. It seems that video content downloading from the first tier peers is performed with a first-come-first-serve pattern. Most of the second tier peers, that have joined the network earlier, download most of their video content from the first tier peers. While for those second tier peers that participate in the network later, e.g. after the 100^{th} peer, the ratio of video content downloaded from the other second tier peers is much higher than (or at least comparable with) the downloading ratio from the first tier peers.

Since there are 153 peers with valid traffic information in experiment 2, it is not possible to study the growth trend of each of the peers. In Fig. 8.9, we discuss the growth pattern of two first tier peers with the best upload performance, and two second tier peers that have uploaded a reasonable amount of video content. From the log-log scale, we can see that after joining the network for some time, the upload throughput of peers converge to a power law (the power law distribution is defined in (4.8)). Since it takes

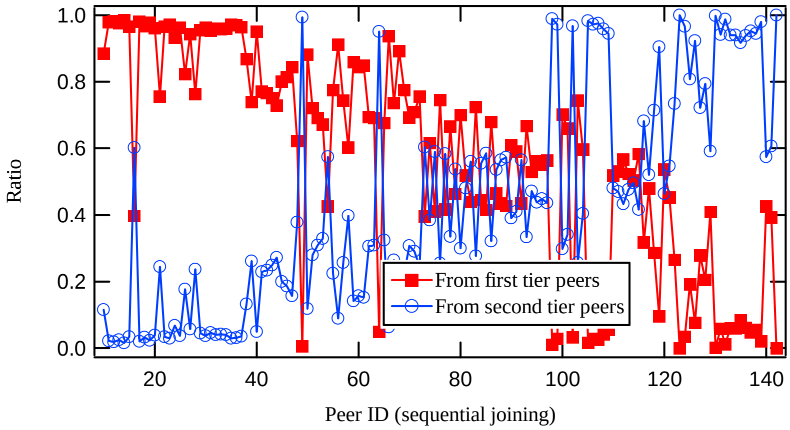


Figure 8.8: Ratio of video content downloaded from the first tier peers and second tier peers over the total amount of video content downloaded at every second tier peer.

some time for other peers to discover the newly joined peer, the upload performance of the peer is not stable at the beginning. For instance, the fitting parameters for the two first tier peers (peers 3 and 4) are $\alpha = -2.3$ and $\alpha = -2.5$ respectively, after the 28^{th} peer joins the network. The traffic pattern of the second tier peer 17 presents three different increasing trends: when it joins the network, the upload throughput increases dramatically; thereafter, it has a smoother growth; after the 65^{th} peer has joined the network, the upload throughput at peer 17 follows a power law distribution with fitting parameter $\alpha = -3.6$. Peer 35 also has a power law growth after the joining of the 65^{th} peer, with fitting parameter $\alpha = -4.2$. Thus, newly joined peers attach to existing peers proportionally to their upload throughput. The different increasing rates α at different peers suggest that, instead of knowing all the other $N - 1$ nodes (excluding itself), a peer may only contact a subset of all the other peers. Thus, the preferential attachment model discussed here differs from the previously proposed preferential attachment model [45] where peers have a complete view of the network when attaching to existing nodes.

8.4.2 Network Load Distribution Over Different Tiers

In this section, we answer the question of whether peers in the first tier act as the super peers that upload a lot. Firstly, in Table 8.1, we present the total amount of upload throughput provided by the first and second tier peers respectively, for all the three experiments. In experiment 1, the total amount of the upload throughput of the second tier peers is only 0.03 MBps. With increasing network sizes, peers in the first tier start to provide more video content. However, it seems that the total upload capacity of first

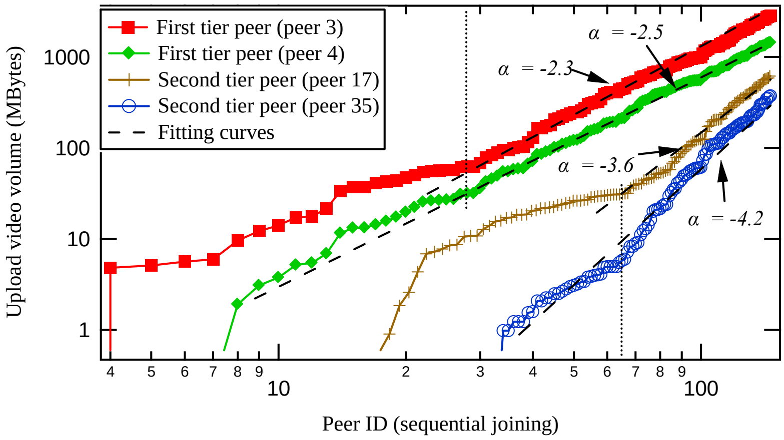


Figure 8.9: Traffic pattern of four peers as a function of peer joining process (log-log scale). The dotted lines are the fitting curves.

tier peers in the SopCast network is constrained to a certain value, since it is 3.5 MBps in experiment 2 and 3.3 MBps in experiment 3, which are almost the same. On the other hand, the total amount of upload throughput at the second tier peers scales with increasing network size, e.g. it increases from 2.7 MBps in experiment 2 to 7.7 MBps in experiment 3.

Table 8.1: Total upload throughput of the first and second tier peers

Total Upload Throughput (MBps)	First tier peers	Second tier peers
Experiment 1	0.8	0.03
Experiment 2	3.5	2.7
Experiment 3	3.3	7.7

Since the first tier peers are selected based on a first-come-first-serve pattern, peers with poor upload capacity may be selected. Next, we examine the impact of peers' outgoing bandwidths on their uploading performance. We have found that peers with high bandwidth do not always upload the most. In other words, the outgoing bandwidth of peers does not have a strong impact on their upload throughput. The computed correlation coefficient between the outgoing bandwidth and the upload throughput of peers is -0.15, 0.15, and 0.16 for experiment 1, 2 and 3 respectively. Peer joining time does not influence the upload performance either. Fig. 8.10 corroborates the stated fact. Surprisingly, even though some of the first tier peers have high outgoing bandwidth, they do not upload a significant amount of video content. Without the source code, we could not affirmatively conclude based on what criteria the SopCast network schedules video content delivery. However, we believe that the video scheduling algorithm implemented in SopCast fails to optimize resource utilization, e.g. bandwidth usage and video downloading from the first tier peers. Some of the first tier peers can be very selfish, because they may not provide many video content to others although they download video packets from a good resource, i.e. the SP.

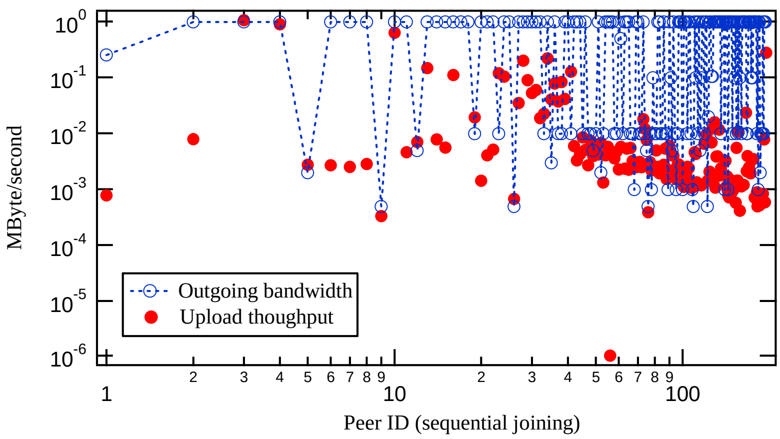


Figure 8.10: Peer upload throughput and outgoing bandwidth. Peers join the network sequentially. (Experiment 2)

8.5 Summary

In this chapter, we studied the topological properties and traffic dynamics of the entire SopCast overlay. While in previous works, obtaining a complete view of the P2PTV network is always a critical issue. We classified the SopCast overlay as a two-layer architecture consisting of a neighbor graph G_N and a video graph G_V. The activation period of a neighbor (video) link, is further used as the snapshot duration of the dynamic SopCast overlay. Our findings suggest that, it is very likely that SopCast is a mesh-based system, in which peers periodically exchange information about video content availability with each other. The SopCast peers are not greedy when connecting to parents for video downloading. On average, a SopCast peer downloads video content from two parents. On the other hand, the existence of super peers, which upload a lot and support many children for video downloading is also observed.

Our study reveals that the real-world P2PTV application, SopCast, is not efficient regarding network traffic distribution and resource utilization. It seems that bandwidth utilization is not optimized in SopCast, because peers with high bandwidth do not necessarily upload more. Some of the first tier peers can be very selfish. Because they may not provide many video content to others, even though they download video packets from a good resource, i.e. the SP. Peers who join the network earlier attach to the first tier peers, while the latter ones connect to the second tier peers to download video content. The upload performance of the first tier peers does not scale as the network size increases. The total upload throughput of the first tier peers in the experiment 2 and 3 are almost the same. It is not clear whether SopCast intentionally sets such a threshold on the upload capacity of the first tier peers.

To summarize, our analysis suggests that SopCast does not implement sophisticated algorithms to select the first tier peers and to attach to existing nodes for video downloading. Our study questions the efficiency of the SopCast network, since peers that are connected to good resources (e.g. to the SP), or with good performance (e.g. high upload bandwidth) do not always keep providing video content to others.

Chapter 9

Conclusion of Part II

In this empirical analysis, we performed a series of experiments in a controlled environment with a set of PlanetLab nodes running the SopCast application. Via passive measurements, we have discovered the basic mechanism deployed in the SopCast protocol, characterized the topological properties of this dynamic P2PTV overlay, and investigated its traffic impact on the Internet.

Major Conclusions

We dissected the SopCast protocol by reverse engineering. After performing an extensive analysis on the tcpdump files that are collected from every PlanetLab node involved in the experiments, we have successfully identified the functionality of some SopCast packets. Regarding the peer communication pattern, the video delivery rule, and the network structure in SopCast, our major conclusions are: 1) SopCast traffic can be categorized into control packets and video packets. The control packets fulfil different functionalities, which coordinate the operation of the SopCast application. The video packets deliver the TV content. 2) Communication between SopCast peers is decentralized. Given a small network, i.e. $N = 50$, peers discover their neighbors very fast. 3) Video delivery in SopCast is chunk-based. Each video chunk has equal length of 10 kbytes. A peer is free to request multiple blocks from its parent(s). 4) We can see SopCast as a three-tier hierarchical structure regarding video delivery. Only a limited number of children can be connected to the SP directly, i.e. the first tier peers. The second tier peers download video content from the peers in the first tier, or between themselves. 5) SopCast does not employ a sophisticated algorithm to select the first tier peers. The first tier peers in the SopCast network are selected based on a first-come-first-serve pattern regardless of their physical distance, network distance, node distance in time, and their upload bandwidth. Once a first tier peer is selected, it remains being connected with the SP during the entire experimental period, unless it leaves the experiment by itself.

Furthermore, we modeled the timevariant SopCast overlay with a two-layer architecture consisting of the neighbor graph G_N and the video graph G_V. Our aim was to find a proper snapshot duration, and characterize the topological properties of this dynamic P2PTV overlay. Our study revealed that: 1) The activation period of the neighbor link and the video link is $\tau_N \sim 2.5$ s, and $\tau_V \sim 2.0$ s respectively. 2) The activation period of the neighbor (video) link is further employed as the reference duration when taking snapshots of the neighbor (video) graph with frequent link establishment and termination. 3) The average number of contacted neighbors within 2.5 seconds is around 38. 3) The incoming degree distribution of the video graph can be modeled as a random graph with very low link density $p \ll 1.0$. On average, the number of parents of a SopCast peer to download video content from is two. 4) We observed the existence of super peers with respect to the number of children that the SopCast peers can support. A super peer sacrifices a large amount of upload capacity to its many children.

Being a popular commercial P2PTV application, SopCast, however, is not efficient regarding network traffic distribution and resource utilization. First of all, bandwidth usage is not optimized in SopCast. Peers with high bandwidth do not upload more, thus, results unbalanced traffic distribution at different peers. Secondly, SopCast employs a very simple algorithm to select the first tier peers, i.e. the first-come-first-serve pattern, and the first tier peers do not have equal roles in uploading video content. Moreover, the upload performance of the first tier peers does not scale as the network size increases. Peers who join the network earlier attach to the first tier peers, while the latter ones connect to the second tier peers to download video content.

Nevertheless, our study questions the efficiency of the SopCast network, since peers that are connected to good resources (e.g. to the SP), do not always keep providing video content to others, and the upload bandwidth is not taken into account when selecting peers. We think that a dynamic first tier peers selection would be more suitable than the current static first-come-first-serve algorithm. If the first tier peers cannot offer good upload throughput, their connections with the SP should be terminated. Re-selecting better peers in the first tier should be performed.

Future Agenda

The empirical analysis performed with SopCast revealed important design issues of today's commercial P2PTV networks. The methodologies and approaches employed in this analysis can be used to understand similar applications. Furthermore, the experiments performed in this thesis is based on full access to the peers (i.e. PlanetLab nodes) with limited network sizes (e.g. hundreds of nodes). As a future step, we can build a crawler that collects the information of peers in the P2PTV network from one or multiple query nodes. Results from the crawled data can be further compared with the ones from the PlanetLab measurement.

As discussed at the beginning of this thesis, theoretical and empirical analysis are two inseparable components. The analysis presented in this part also provides insights on future theoretical studies. For instance, the major difference between the real-world P2PTV application and the theoretical model proposed in Part I is that *peers are not equal*. Therefore, modeling the network with the existence of super peers can be a possible research direction.

The third proposal is the study of topology when peers attach according to a preferential attachment model on a random subset of peers. As demonstrated in Section 2.3.3, to describe the system where peers have a partial view of the network, a large number of states is needed. Thus, the new model of creating networks that is closely related to how P2P networks evolve, can be examined by performing simulations.

Part III

Empirical Study: An Online Social Network

Chapter 10

Introduction To OSNs

A society is a group of individuals who have more or less conscious relations to each other. In 1954, the concept of *social network* was first raised by Barnes, in his work of: Class and Committees in a Norwegian Island Parish, [46]. A social network is defined as a system consisting of social entities (e.g. individuals, organizations), which are connected by different types of relations (e.g. friendship, kinship, sexual relationship, common interest). In graph theory, the social entities are seen as nodes, and their relationships in between as links. A social network can be observed not only in our real-world life, but also in the online world, as there are more and more web applications that aim to connect users via the Internet.

The history of *online social networks* (OSNs) can be traced back to 1997, when the first OSN, SixDegrees [30], was launched. Since then, a number of social networking applications were released. They quickly gained significant popularity, and dramatically changed the way that people consume information across the Internet. Unlike traditional web-based services, OSNs are user-centric. They aim to fulfil diverse needs of the end-users. For instance, in Facebook [10], Hi5 [13], LinkedIn [19] and Orkut [23], OSN users can find and interact with people; in MySpace [21] and Twitter [35], users can share their ideas and receive feedback from others. There is another type of OSNs that allows users to publish user-generated content, e.g. Flickr [11], Last.fm [18] and YouTube [37], or to submit, discover, and distribute content that are published anywhere on the web, such as Digg [8], Delicious [7] and Reddit [29]. In all of the above OSNs, social networking is the most distinct feature - users form virtual communities to network with others and to share information collaboratively. Since users are more inclined to spend their social lives online, OSNs become promising platforms for online campaigning [58], viral marketing [97] and targeted advertisements [128].

The major difficulty of analyzing various social phenomena in real-world, large-scale social networks is the lack of public data. Thanks to the thriving of OSNs, collecting relevant information about the characteristics (e.g. social interests and collaborative behaviors) of millions of users is feasible. In this chapter, we will briefly introduce the Digg OSN that is going to be studied, and our methodology of collecting the Digg data.

10.1 The Digg OSN

The Digg OSN, studied in this thesis, is a large-scale content discovery and sharing application, which is also referred to as a social media aggregator. According to the traffic statistics provided by Alexa.com on May 4th, 2010, the Digg website traffic is ranked as 117^{th} globally, and as 52^{th} in the United States [1]. Rather than simply searching for, and passively consuming information, the Digg users are collaboratively publishing and propagating information.

The Digg users can submit an URL (Uniform Resource Locator) of a story that is published elsewhere on the web. A story published in Digg can be a video, an image, or news. In fact, from our study, we notice that around 88% of the Digg stories are news. Once a story is published, the Digg users can *digg* or *bury* it by using the voting system implemented in Digg. A digg on a story refers to a positive rating, and burying a story indicates disagreement or a negative attitude towards that story. In this way, the Digg users can publish, discover, and promote the most interesting stories by collective recommendation without a group of website editors. A story that has just been submitted is referred to as an *upcoming* story and is placed on the upcoming section in Digg. If an upcoming story is considered to be interesting by the majority, it becomes *popular* and appears on the front pages of the popular section. Promoting a story from upcoming to popular is managed by an algorithm developed by Digg. The algorithm considers the number of diggs, the diversity of users that are digging the story, the time when the story was submitted, and the topic of the story as the major factors to promote a story. Unfortunately, the Digg website does not provide too much details about the algorithm. There are approximately 10,000 stories submitted to Digg daily, while only around 150 stories are promoted to popular.

Social relationship is assumed as an important factor during content recommendation. Every Digg user maintains a list of friends that he has added, which is referred to as the *friends list*. Via a so-called *friends' activity* interface (see Fig. 10.1), users can keep track of their friends' activities once logged in, e.g. stories that their friends have submitted, commented or digged. To discover a story, a Digg user can either use the friends' activity interface, or other features on the Digg website, e.g. front page and search engine. As shown in Fig. 10.1, Digg stories are categorized into eight major topics: technology, world & business, science, gaming, lifestyle, entertainment, sports and offbeat. A Digg user can also find stories with respect to their type: news, videos, and images. Stories belong to each topic or type are again classified as upcoming and popular. All popular (upcoming) stories under each topic and each type are aggregated to the overall popular (upcoming) section, which is the default page shown to a user entering the Digg website. Once a user has digged on a story, he is implicitly propagating and recommending the story to others.

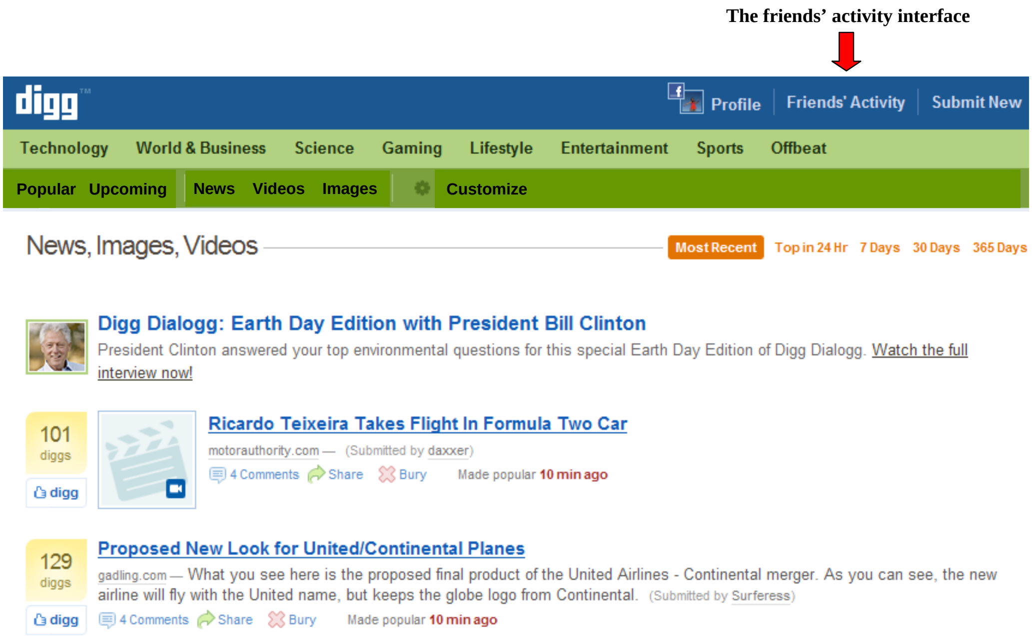


Figure 10.1: The Digg web interface.

10.2 Related Work Performed With OSNs

Early studies on real-life social networks focus on a collection of people, that are formed by friends, acquainted people, movie actors, scientific collaborations on research papers, etc. A famous empirical study in sociology was performed by Stanley Milgram [93], from which the "six degrees of separation" has been observed. Afterwards, extensive theoretical analyses were conducted to discover the structure of social networks (see e.g. [45], [82], [99], [126]). Regarding the spread of information in social networks, Mark Granovette argued in his influential paper: The Strength of Weak Ties [65], that the social relationships between users can be partitioned into "strong" and "weak" ties. The strong ties correspond to the links between close friends, relatives, or neighbors; and the weak ties correspond to links between acquaintances, or friends of friends. In was shown in [65] that the strong ties are tightly clustered, and the weak ties are more efficient regarding information spread, e.g. acquiring job information. An overview of social network analysis techniques is presented by Wasserman and Faust in [125].

When studying the real-world social networks, the sociologists are facing a major challenge: the lack of large-scale data to analyze various social phenomena. With the thriving of OSNs in recent several years, investigating the properties of social communities that are at a large-scale becomes realistic. Hence, sociologists and computer scientists start to focus on the OSNs. Previous works performed with OSNs mainly addressed three issues: 1) the topological properties and network structures; 2) user behaviors and properties; 3) content publication and propagation. Mislove *et al.* [95] studied the topological properties of four OSNs that are large-scale: Flickr, YouTube, LiveJournal, and Orkut. They obtained the data by crawling publicly accessible information on these networking sites. In [95], different network metrics, e.g. link symmetry, node degree, assortativity, clustering coefficient of the four networks were measured. Mislove *et al.* have shown that the OSNs are characterized by high fraction of symmetric links, and composed of a large number of highly connected clusters. The degree distributions in the OSNs follow a power law, and the power law coefficients for both in-degree and out-degree are similar. Backstrom *et al.* [44] studied the network growth and evolution by taking membership snapshots in the LiveJournal network. They also presented models for the growth of user groups over time.

User characteristics in OSNs are also often addressed. Leskovec *et al.* [86] presented an extensive analysis about communication behaviors and characteristics of the Microsoft Messenger instant-messaging (IM) users. The authors examined the communication pattern of 30 billion conversations among 240 million people, and found that people with similar characteristics (e.g. age, language, and geographical location) tend to communicate more. Based on the data, Leskovec *et al.* constructed a communication graph and analyzed the topological properties of the graph in terms of node degree, cluster coefficient, and the shortest path length. They have shown in [86] that the communication graph is well connected, robust against node removal, and exhibits the small-world property[1]. Benevenuto *et al.* [47] examined users' activities of Orkut, MySpace, Hi5, and LinkedIn. A clickstream model was presented in [47] to characterize how users interact with their friends in OSNs, and how frequently users transit from one activity (e.g. search for people's profiles, browse friends' profiles, send messages to friends) to another.

There are also many researchers who aim to discover content popularity and propagation in OSNs. For instance, Cha *et al.* [51] studied photo propagation patterns in the Flickr OSN and the impact of social relations on propagating photos. The results in [51] suggest that: 1) friendship relations play an important role during information spread; 2) photo popularity may increase steadily over years even for the most popular photos; and 3) over 50% of users find their favorite pictures from their friends in the social network. Szabo *et al.* [116] analyzed and compared the popularity of Digg content with YouTube videos. As shown in [116], YouTube videos keep attracting views throughout their lifetimes, whereas Digg stories saturate quickly. Social networking is not effective once Digg content is exposed to a wide audience, although they are important in the

[1]Small-world networks have a small diameter and exhibit high clustering [126].

stages when content exposure is constrained to a small number of users. An in-depth study about YouTube and Daum (a Korean OSN) video popularity evolution was discussed in [50]. It was shown that video popularity of the two applications is mostly determined at the early stage after a video content has been submitted to the OSNs. In [83], Lerman showed that users find new interesting stories via the social links and claimed that social (friendship) filtering is an effective approach to filter information in Digg. The same observation was found in [84], where Lerman *et al.* showed that social networks play a crucial role in the spread of information on both Digg and Twitter websites. Content in Twitter spread continuously as the story ages, whereas Digg stories initially spread quickly through the network.

Most of the previous work addressed the importance of friendship networks during the discovery and spread of information. The same conclusion was also made for Digg. However, we believe that previous reports regarding the Digg network are not sufficient to reflect its overall content propagation pattern due to two reasons. First of all, results in [83] and [84] are based on a limited number of data set that were collected during a short period of time (around a week): 2858 stories in [83] and 3553 stories in [84]. Secondly, to what extent the friendship relations are propagating content during the entire information spread process and how to quantitatively measure the influence of social links have not been addressed in [83], [84] and [116].

10.3 Research Questions

A fundamental assumption about OSNs is the potential of discovering and propagating information along social links, i.e. the friendship relations ([51], [83], [84], [85] and [116]). Another important factor that has not been addressed is the time criticality of the content shared in OSNs. Moreover, user behaviors in a given OSN is influenced by the specific interfaces and mechanisms provided by the website. Yet, the influence of content dynamics as well as the service interfaces during the spread of information and on the characteristics of users are not well understood.

In this thesis, we present a large-scale measurement study of the Digg OSN, also referred to as a social media aggregator. Compared with other OSNs that aim to publish and share content, the Digg network has three distinguishable features: 1) Digg uses the number of diggs to reflect the popularity and importance of the published content; 2) Digg implements its own algorithm to classify popular and unpopular content (i.e. upcoming content) and place them in different sections. 3) Digg provides multiple features for content discovery and digging, i.e. via friends, by using the front page/recommendation engine, and by search. The aforementioned three factors lead to distinct user behaviors and content propagation pattern in Digg. Our objective is therefore to study the underlying mechanism of information propagation, and the effectiveness of friendship relations during content propagation in Digg. In this thesis, we focus on two major research issues.

1. Study the network structure in Digg, measure users' digging activities and quantitatively compare users' interests/tastes with each other.

2. Examine the content propagation pattern and the effectiveness of friendship relations in the spread of content.

Studying the above two issues is beneficial to understanding the fundamental characteristics as well as the underlying principals of disseminating content in emerging social media aggregating websites and helps to design new systems.

10.4 Data Crawling Process

We aim to study different perspectives of the Digg OSN, such as the friendship relations, the characteristics of users, and the properties of the published content. Hence, the major challenge in crawling the Digg network is to obtain the most comprehensive and valid data that we are interested to reflect different aspects of the Digg OSN.

While most social network traces are crawled using friendship relations, see [95] and [41], the Digg data set was obtained by a simultaneous exploration of the network from four different perspectives, as shown in Fig. 10.2. By using the Digg Application Programming Interface (API), we are able to explore the aforementioned four perspectives (from bottom to top in Fig. 10.2) during data collection:

- **Site perspective**: The Digg website lists all popular and upcoming stories under different topic areas. Every hour, we retrieve all popular stories (for all topics) that are listed on Digg. Every four hours, all upcoming stories (for all topics) are collected. All discovered stories are added to an "all-known story" list maintained by us.

- **Story perspective:** For each of the stories that has been retrieved, a complete list of all activities performed by different users (who digged on the story) is collected. Any user that is discovered will be added to the "all-known user" list for future exploration.

- **User Perspective:** For each user discovered within the Digg OSN, the list of their activities, such as submitting and digging on stories, is retrieved. Occasionally, a previously unknown story is discovered (this is typically the case for older stories before we started the collection). For such a story, the entire (digging) activities of users are retrieved for that story.

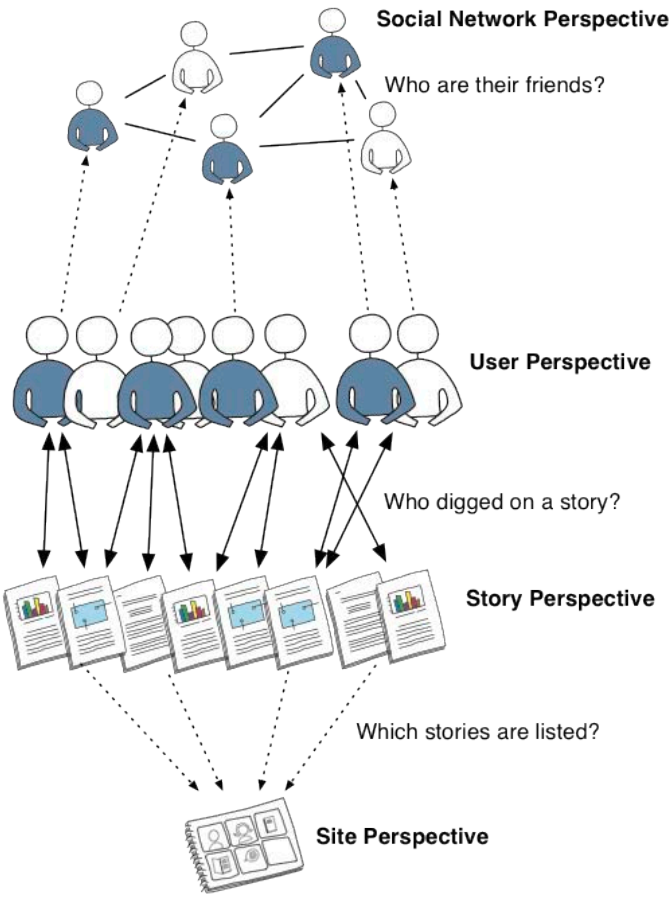


Figure 10.2: Different components of the Digg crawling process.

- **Social Network Perspective**: Each registered user can make friends with other Digg users. In the crawling process, a list of friends is retrieved for every user. If a friend is a previously unknown user, this user is added to the data discovery process, and a list of all his friends and his public user profile information are retrieved. This procedure is continued in a breath-first search (BFS) manner until no new user can be found. The process is periodically repeated afterwards to discover new friendship relations that have been formed after the last crawling pass through the data.

By using the above crawling methodology, we are able to collect the entire information about friendships and activities of users and the published content in the Digg network. Although the Digg network was officially founded on December, 2004, the feature of the friends list was not released until July 2005, when the second version of Digg was launched. Therefore, in our analysis, the collected friendship information starts from July 2005. Until May 2010, our Digg data set has a volume of more than 600 GB (Giga Bytes), containing the related information about 1.5 million registered users and 10 million published stories in the Digg OSN. The collected Digg data set is in the form of individual XML (Extensible Markup Language) files. To study different characteristics of the Digg OSN, we analyze the collected XML files by using Java language [16] and bash scripting [4]. Following, we present some example files of the collected data. With these files, we perform a large-scale analysis of the Digg OSN. Due to the large volume of data set to be studied, our analysis is both time and computationally consuming.

Digg Information of Stories/Users

The Digg network assigns each story a unique ID. Table 10.1 shows an excerpt of the digg information of the story 16181860, which contains all users that have digged on it. In the second row, we see a timestamp of 1261779003, which is in UNIX epoch format. It is the time that the digg information was crawled from the Digg website. The total number of diggs made on this story when we crawled it, as indicated by the field of “total”, is 723. The field of “date” indicates the time that a user digged on the story. “Story” is the story ID. In this example, it is all “16181860” since we queried all the diggs on story 16181860. “User” names the user who has made the digg. “Status” lists the status of the story (popular or upcoming), when the information about story diggs were downloaded.

Similarly, Table 10.2 presents an example of all the diggs made by the user “MacBookForMe”. The structure is identical to that in Table 10.1.

Table 10.1: Digg information of the story with ID 16181860


```
<?xml version=“1.0” encoding=“utf-8” ?>
<events timestamp=“1261779003” total=“723”>
<digg date=“1260118952” story=“16181860” user=“craigsbro22” status=“popular”/>
<digg date=“1255980662” story=“16181860” user=“mapyqf” status=“popular”/>
+ many more lines
</events>
```


Table 10.2: All diggs made by user ”MacBookForMe”


```
<?xml version=“1.0” encoding=“utf-8” ?>
<events timestamp=“1249291922” total=“195”>
<digg date=“1249291031” story=“14650409” user=“MacBookForMe” status=“upcoming” />
<digg date=“1249291029” story=“14650953” user=“MacBookForMe” status=“upcoming” />
+ many more lines
</events>
```


Story Information

We have collected related information about the stories published on Digg. Table 10.3 presents an example of the story 16181860. The first row presents the URL of the story 16181860, where it is originally published. Following, we see the submission time of the story (indicated by “submit_date”), the total number of diggs (indicated by the field “diggs”) made on the story and the unique ID (indicated by “id”) of the story. In the third row, we presents the promotion time when the story became popular (by the field of “promote_date”), the status of the story when it is crawled (by “status”), the media type of the story (by “media”). Next, we show the description and the title of the story. As indicated by the sixth and seventh rows, the story is submitted by user “MacBookForMe”, who registered with Digg at time 1202811079. By the time that we crawled the data, there are 21504 clicks on the profile of user “MacBookForMe”, and the topic that the story belongs to is “Sports”. The related information about the story 16181860 is ended by “</story>”, as indicated in the last row.

Friendship Information

Table 10.4 shows the friends of user “MacBookForMe”. The field “name” indicates the user name of his friends. “Registered” is the time when the friend registered with Digg. “Profileviews” is the number of clicks made on the profile of that friend. “Mutual” means a mutual friendship relation, and “date” is the timestamp when “MacBook-ForMe” made friend with the corresponding user.

Table 10.3: Related information of the story 16181860

1	<story link=”http://news.bbc.co.uk/1/hi/in_depth/8280072.stm” />
2	<story submit_date=”1254247007” diggs=”722” id=”16181860” />
3	<story promote_date=”1254259233” status=”popular” media=”images” />
4	<description>Who knew synchronised flipping could be so much fun!</description>
5	<title>Flippin’ Flip [PIC]</title>
6	<user name=”MacBookForMe” registered=”1202811079” profileviews=”21504” />
7	<container name=“Sports />
8	</story>

Table 10.4: Friends of user “MacBookForMe”

<?xml version=”1.0” encoding=”utf-8” ?>
<eventstimestamp=”1241784850” total=”80”>
<digg name=”adrian67” registered=”1146300011” profileviews=”23333” mutual=”1” date=”1224590329” />
<digg name=”alamala” registered=”1207600335” profileviews=”9054” mutual=”1” date=”1211673990” />
<digg name=”annjay” registered=”1206177687” profileviews=”9106” mutual=”1” date=”1225446296” />
+ many more lines
</events>

Front Page Information

The Digg website has 250 front pages, with 15 popular stories per page. Every hour, we collect information about the popular stories that appear on the front pages. The first column, as presented in Table 10.5, is the front page number. Following are the IDs of the 15 stories on each front page in a decreasing order regarding their positions on that page. The number of diggs made on a story is shown inside the parenthesis of each story. For instance, Table 10.5 tells us story 22263292 is the first story appearing on the first front page. At the moment that we have crawled it, there were 97 diggs on that story. The crawling date and time are reflected by the file names, e.g. frontpage_2010-04-30–15-00 is the name of the example file presented in Table 10.5.

Table 10.5: Front page information crawled on 2010-04-30, 15-00 hour

1, 22263292(97) 22262567(116) (224) 22260602(124) 22253075(297) 22262446(342) .....
2, 22250421(295) 22262794(172) (682) 22250537(176) 22253240(337) 22246646(233) .....
3, 22251223(292) 22255642(203) (47) 22251404(379) 22256668(403) 22256314(231) ......
.....

Chapter 11

Network Topology and User Characteristics

In this chapter, we first study the Digg network structure, e.g. degree distribution, link symmetry and degree correlation. Afterwards, we measure users' digging activities and interests. Finally, we answer the question of whether Digg users are sharing similar tastes with their friends.

11.1 The Topological Property of the Digg OSN

Friendship relation is assumed to be a crucial factor during content propagation. Therefore, we first characterize the network structure in Digg and compare the Digg network with other OSNs. By considering the 1,527,818 registered Digg users as nodes and their friendship relations as links, a direct *friendship network* G_F, is constructed. If a user A adds user B as a *friend*, user A is referred to as a *fan* of user B. If B also adds A as a friend, the two users are called *mutual friends*. A bi-directional link is called a *symmetric link*. Otherwise, the link is referred to as being asymmetric. In G_F, the outgoing degree D_{out} defines the number of friends that a random user has, and the incoming degree D_{in} indicates the number of fans that a random user has.

As mentioned in Section 10.1, users can keep track of the stories that are submitted and digged by their friends via the friends' activity interface. Thus, content propagation in the Digg friendship network G_F is initiated in the reversed direction along the friendship links. In Fig. 11.1, we illustrate the friend and fan relationships in Digg (the solid arrows), as well as the way that information is propagated via friendship links (the dotted arrows). As shown in Fig. 11.1, user A has three friends and two fans. Being the fan of user B, C and D, user A discovers and consequently diggs stories recommended by B, C and D. On the other hand, since user E and F add user A as friend, user A is implicitly propagating content to E and F. In short, a user disseminates content to his fans in the opposite direction along the incoming links.

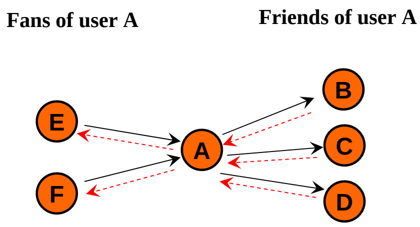


Figure 11.1: Illustration of the friend relationships in Digg as well as the way that information is propagated. The solid arrows represent the friend or fan relationships. The dotted arrows refer to information propagation along the friendship links.

11.1.1 The Degree Distribution

We define the *giant component* as the largest connected component in a network. The giant component in our Digg data collection consists of 685,719 nodes and 6,736,174 links. Out of the remaining 842,099 nodes, there are 810,586 nodes not being connected to any one in the network. The nodes with zero in-degree and out-degree are defined as *disconnected nodes,* which is approximately half of the entire Digg users in our data set. The remaining 31,513 nodes that are not connected with the giant component, form 13,270 distinct *connected components*. The maximum number of nodes in these components is 77, and the smallest component only consists of 2 nodes. These connected components consist of 23,763 links.

In Fig. 11.2 (a) and (b), we plot the pdf of the out-degree, D_{out}, and in-degree, D_{in}, of the giant component respectively. On a log-log scale, both $\Pr[D_{out} = k]$ and $\Pr[D_{in} = k]$ exhibit straight lines, conforming to the power law distribution[1]. Notice that Digg has artificially capped a user's maximal out-degree to 1000, the distribution of very high out-degrees (close to $D_{out} = 1000$) is thus, distorted in Fig. 11.2 (a). By using least-squares fitting, the exponent of $\Pr[D_{out} = k]$ is found to be $\alpha_p = 1.6$, which is slightly lower than the exponent of the in-degree, i.e. $\alpha_p = 1.8$.

Our finding about the node degree distribution in Digg is consistent with previous reports on Flickr, YouTube, LiveJournal, and Orkut (see [95]), in the sense that the power law exponents of these OSNs are all smaller than 2 (between 1.5 and 2). However, the node degree distribution of OSNs suggests fundamental different network structure compared with other complex networks, e.g. the real-world social network, the World Wide Web (www)[2], and the power grid network, in which the power law exponents are

[1]The power law distribution is defined as $\Pr[X \leq x] = cx^{-\alpha+1}$.

[2]The nodes in a www network are HTML documents, and they are connected by links pointing from one page to another.

between 2.1 and 4 [45]. The observed power law distributions with an exponent smaller than 2 indicates that there is no finite mean regarding the node degree distribution. Indeed, we can still estimate on the mean value of node (in- and out-) degree by using real data set with finite size. However, as the size of data set changes, our estimation on the mean value of node (in- and out-) degree also changes with non-negligible variance. On the other hand, the mean value of node degree does not diverge as the network size increases in the other complex networks mentioned above.

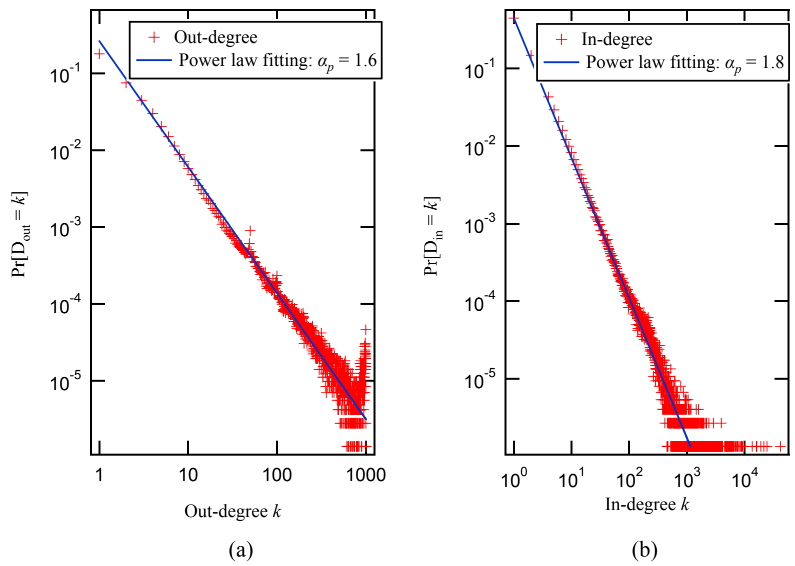


Figure 11.2: Sub-figure (a): Pdf of the out-degree of the giant component. A cut-off out-degree of 1000 is set in the digg network. Sub-figure (b): Pdf of the in-degree of the giant component. Both curves are plotted on log-log scale and best fitted with the power law distribution.

Furthermore, the large amount of disconnected nodes in the Digg network seems to conflict with the existing assumption of the importance of a friendship network - users need to establish friendship relations in order to find and propagate information. Since most large OSN traces are crawled by using the breadth-first search (BFS) technique following friendship links, the amount of disconnected nodes are naturally excluded from the collected data set, and only the connected components are studied, e.g. in [51] and [94]. The crawled Digg data set provides us the advantage of analyzing the characteristics of the disconnected nodes and their digging activities. As we will show in Section 11.2, the disconnected nodes are actively digging stories even though they are not connected with anyone, which questions the necessity of making friends to find and propagate content in Digg.

11.1.2 Link Symmetry

Contrary to previously studied OSNs which exhibit high level of symmetry (e.g. the fraction of symmetric links is 62% in Flickr, 74% in LiveJournal and 79% in YouTube [95]), the link symmetry in Digg network is much lower. Among the total number of 6,759,937 friendship links in Digg, only 2,687,583 links are symmetric (38% on average). The link symmetry in Digg also varies with respect to users' out-degrees. As shown in Table 11.1, users that are connected with a small group of friends (i.e. $0 < D_{out} < 10$) are more likely to be accepted as mutual friends: 53% of their friendship links are symmetric. Being the fan of many users does not increase the probability of being accepted as friends. The percentage of symmetric links of users with $D_{out} = 1000$ is only 0.31.

Table 11.1: Link symmetric and content discovery in Digg

User Group	Number of Users	Symmetric Links (Percentage)	Stories Digged Via Friendship
$0 < D_{out} < 10$	282536	53%	4.4%
$10 \leq D_{out} < 100$	49416	42%	25%
$100 \leq D_{out} < 1000$	13993	39%	59%
$D_{out} = 1000$	111	31%	60%

The level of link symmetry in OSNs is generally determined by users' incentives of making mutual friends, i.e. to discover and propagate content via friendship relations. In the sequel, we discuss whether Digg users depend on the friendship links to discover content. If a user diggs a story after the time that his friends have digged on the same story, we assume that the story is recommended by friends and therefore propagated via friendship links. From the entire stories that a user has digged, we calculate the number of stories that was recommended and digged by using friendship relations. As shown in Table 11.1, the content location pattern in Digg is consistent with existing hypothesis in OSNs - friendship relation is crucial in discovering content. Users that make many friends tend to be more inclined in digging stories on which their friends have digged. For instance, users that have made a lot of friends ($D_{out} = 1000$) discover 60% content via friendship links, whereas users who make less friends ($D_{out} < 10$) do not depend on their friends to find content. Most likely, they use front page recommendation or the search engine to digg stories.

The findings presented in Table 11.1 also underlines the fact that Digg friendship relations are less reciprocal in terms of content discovery. Making many friends (high out-degree) leads to an increasing number of stories discovered from friends. However, the friends of a user do not necessarily want to locate content via his friends. By using (8.1), we see that there is no correlation between the out-degree of two friends:

the correlation coefficient is $\rho_{out} = 0.05$. Users who make many friends do not necessarily connect with others that are similar. Hence, the friends of a user who has high out-degree may not rely on friendship relations to find content (see Table 11.1). Consequently, it is less demanding for them to make mutual friends.

11.1.3 Assortativity in Digg

The assortativity [122] measures the degree correlation of two connected nodes. In general, the dependence between two random variables of X and Y is evaluated using the linear correlation coefficient described in (8.1) of Section 8.3.3.

According to Newman's definition in [100], the assortativity in directed networks measures the tendency of a node i to connect with other nodes that have incoming degrees similar to node i's outgoing degree. The above definition, however, does not provide much insight on the content propagation process in Digg. As mentioned in Section 11.1, a Digg user propagates content to his fans implicitly. Therefore, instead of employing the definition in [100], we measure the assortativity in Digg as the correlation of the in-degree between two nodes that are connected by a directed link. Assuming that we have randomly chosen a directed link from the Digg friendship network, the node at the start of the directed link is called the *source node*, and the node at the end of the link is referred to as the *destination node*. We calculate the assortativity in the Digg network as

$$\rho = \frac{\sum_{k=1}^{L} i_k j_k - \frac{1}{L}\sum_{k=1}^{L} i_k \sum_{k=1}^{L} j_k}{\sqrt{\left[\sum_{k=1}^{L} i_k^2 - \frac{1}{L}\left(\sum_{k=1}^{L} i_k\right)^2\right]\left[\sum_{k=1}^{L} j_k^2 - \frac{1}{L}\left(\sum_{k=1}^{L} j_k\right)^2\right]}} \tag{11.1}$$

where L is the total number of the directed links in the network, i_k is the excess in-degree of the source node at the k-th link, and j_k is the excess in-degree of the destination node of that link. The excess in-degree is one less than the original out-degree (in-degree) of that node.

The correlation coefficient ranges between -1 and 1. A positive correlation coefficient, e.g. close to 1, indicates users tend to connect to others of similar degree. For instance, the influential users[3] are connected with other influential users who have high in-degree. While a negative correlation coefficient, e.g. close to -1, means that influential users are connected with users who are not similar (i.e. users who have a few fans). The in-degree correlation between two friends in Digg is derived as $\rho_{in} = -0.03$ by using (8.1), which is close to zero. This value shows that there is in fact no correlation between the in-degree of two friends in Digg. In theory, content propagation

[3] We define an influential user as the one that has many fans, i.e. high in-degree.

in an assortative network (i.e. positively correlated in-degree) is more favorable under the assumption that friendship network is important during content propagation. To propagate content, we can opt for the influential users, because they are able to forward the content to other influential users and thus speed up the spread of content. However, such a property has not been observed in the Digg network.

11.2 Users' Activities and Interests

In this section, we quantitatively measure users' digging activities and users' interests in Digg.

Users' Digging Activity

We define X_u the number of stories a user has digged, which is also referred to as the *digging activity* of that user. We notice that 93% of the 1.5 million Digg users have digged at least one story, whereas the remaining 7% users only registered in Digg without any digging.

In Fig. 11.3, we present the pdf of the digging activity X_u of the connected nodes and disconnected nodes respectively. Although the disconnected nodes do not establish any friendship relations, they are also actively digging stories in the network. However, compared with the nodes in the connected components, the digging activity of disconnected nodes is quantitatively different - the maximal number of digged stories is approximately one order less than for nodes in the connected components. The curve of the disconnected nodes also decays faster than the connected nodes and the number of disconnected users who digged less than ten stories is higher than those connected ones. Combining the observations in Fig. 11.3 and Table 11.1, we see that friendships do not always play an important role in discovering and digging content. At least, the digging activities of the disconnected users are not affected by the fact that they do not establish friendship relations.

Interestingly, on a log-log scale, the curves of $\Pr[X_u = k]$ exhibit a slightly decaying trend from the straight line for users that digg the most stories. Fig. 11.3 shows that the best fitting curve of $\Pr[X_u = k]$ is the power law distribution with an exponential cut-off[4] at the tail. The power law distribution with an exponentially decaying tail has been reported in protein networks [77], in e-mail networks [59], in actor networks [43], in www networks [98], and for video popularity in YouTube [50]. A power law distribution is normally generated by the preferential attachment model [45]. If the distribution of X_u completely obeys the power law, we can rephrase users's digging activities as: the probability that a user will digg a new story is proportional to the number of stories

[4]The power law with an exponential cut-off is described by $f(x) = cx^{-\alpha_p}e^{-\alpha_e x}$, where the exponential decay term $e^{-\alpha_e x}$ overwhelms the power law behavior at very large x.

that he already digged. This explanation however, does not explain the decay trend at the tails of the two curves. In the sequel, we discuss two models that were proposed to explain such a bending tail.

First of all, Amaral *et al.* [43] argued that in a social network formed by actors, an actor may stop acting due to his age. Therefore, a node (i.e. an actor) will stop receiving new links after some time, which yields the exponential cut-off at the tail. Secondly, Mossa *et al.* [98] attributed the exponential decay of the degree distribution of the www to the fact that users cannot receive information about all available web pages on the Internet. We think the exponentially bent tails in Fig. 11.3 can be explained by using the two models described above. As we will show in Chapter 12, the "attractive" period of a story is very short. The majority of the diggs on a story is made after one or two hours since it is promoted to the Digg front page. Afterwards, the popularity of a story decreases dramatically. Hence, a story will stop receiving diggs due to its age. As mentioned before, there are approximately 10,000 stories being submitted to Digg every day. Since users cannot stay online 24 hours per day and because of the short lifetime of a story, users are eventually digging a subset of all published stories. Combining the aforementioned two facts, the bending tail is consequently generated.

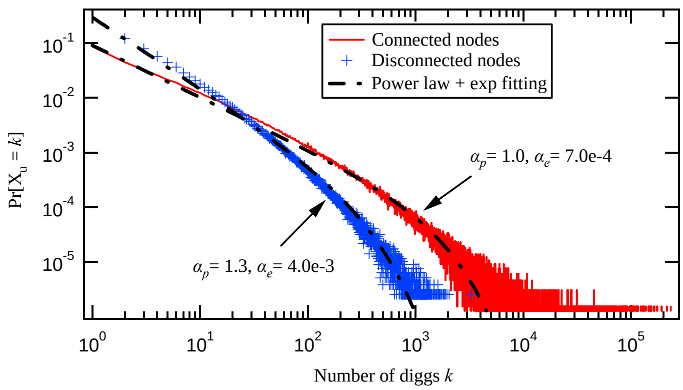


Figure 11.3: Pdf of the number of stories digged by the connected nodes and disconnected nodes, respectively. The best fitting curve is the power law distribution with an exponential cut-off.

Concentration of Users' Interests

The major challenge to quantify users' interests in OSNs is that users may not provide their social interests. Even though some users have specified their interests, their actual activities may deviate from what they have claimed. Thus, to our knowledge, this thesis is the first work trying to quantify users' interests in OSNs.

As mentioned earlier, stories in the Digg network are categorized into eight topics. Given that a user has reacted on k $(1 \leq k \leq 8)$ different topics, we define the number of stories that a user has digged under each of the topic, as a metric to quantitatively evaluate his preference of that topic.

Let the set of random variables $\{X_k\}_{1\leq k\leq 8}$ be the number of stories that a user digged on a particular topic. The index of k from 1 to 8 corresponds to the topic of technology, world & business, science, gaming, lifestyle, entertainment, sports and offbeat respectively. The set $X_{(1)}, X_{(2)}, ..., X_{(8)}$ is consequently defined as the *ranked random variable* of $\{X_k\}_{1\leq k\leq 8}$, if $X_{(1)} = \max_{1\leq k\leq 8} X_k$, and $X_{(8)} = \min_{1\leq k\leq 8} X_k$. The sum of $\{X_k\}_{1\leq k\leq 8}$, $S = \sum_{k=1}^{8} X_k$, is the total number of stories that a user digged. The variable of $R_k = \frac{X_{(k)}}{S}$ defines the ratio of the stories under the k-th topic after ranking, over the total number of stories digged by a random user.

Fig. 11.4 depicts $E[R_k|V_n]$, the average value of R_k provided that a user is interested in n $(1 \leq n \leq 8)$ topics (denoted by the event V_n). We see that users exhibit strong and distinct preference of their most favorite topic over the less favorite ones. A user diggs at least two times more stories in his favorite topic than the second favorite ones, and the stories digged under individual topics decrease logarithmically. For instance, if a user is interested in two topics, his most favorite topic counts for 67% of the total stories that he digged. A user interested in three topics diggs 59%, 24% and 17% of the stories in his first, second and third favorite topics. The above findings are further used in Section 11.3, to compare the similarity between users' interests.

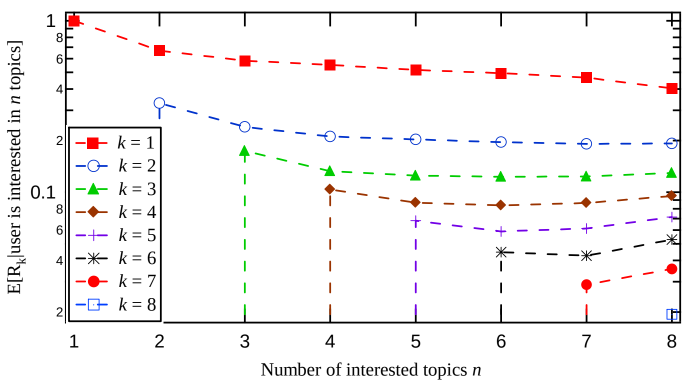


Figure 11.4: The average ratio of the k-th favorite topic of users that are interested in n topics after ranking. The vertical axis is plotted on logarithmic scale for easier reading.

11.3 Similarity Between Friends

It is commonly assumed that *users are making friends with who are similar*. This section aims to discover whether the aforementioned assumption is ture in the Digg network. In Fig. 11.4, we observed the existence of a most preferred topic of users over the other less favorite ones. Based on the distinct and stable users' interests, we further measure the similarity between users' taste.

We denote by $T_{(k)}$ the name of the *k-th* topic after ranking. For a random user pair i and j, we obtain two set of lists of $\left\{t_{i(1)}, t_{i(2)}, ..., t_{i(8)}\right\}$ and $\left\{t_{j(1)}, t_{j(2)}, ..., t_{j(8)}\right\}$, in which $t_{i(k)}$ and $t_{j(k)}$ are the name of the *k-th* favorite topic of user i and user j, respectively. Since $t_{i(1)}$ is the most favorite topic of user i, we compare $t_{i(1)}$ with $t_{j(k)} \in \{1 \leq k \leq 8\}$ of user j. The *similarity hops*, defined in (11.2), is used as the metric to evaluate the similarity of the most favorite topic between i and j.

$$h_{ij} = (k-1)\, 1_{\left\{t_{i(1)}=t_{j(k)}\right\}} \tag{11.2}$$

in which the indicator function, $1_{\{x\}}$ is defined as 1 if the condition of x is satisfied, else it is zero. The similarity hop h_{ij} measures the distance of user i's favorite topic with respect to the *k-th* favorite topic of user j, and ranges between $0 \leq h_{ij} \leq 7$. A zero hop means that two users have identical interests with each other. A small similarity hop, say $h_{ij} = 1$, indicates high overlapping interests between two friends. While a large hop suggests that users do not have similar taste in common, e.g. $h_{ij} = 7$.

Our calculation shows that the similarity hop between two friends decreases exponentially: 36% friend pairs have the identical interests and the percentage of friend pairs that are one, two and three hops away are 20%, 15% and 10% respectively. The average similarity hops is 1.7, indicating high similarity between friends' interests.

11.4 Summary

OSNs show different properties with previously observed networks, such as real-world social networks, technological and biological networks, in terms of node degree distribution. For instance, the power law exponents for different OSNs, are smaller than 2, as reported in this thesis (for Digg) and previous work (for Flickr, LiveJournal, and Orkut) [95], while it is between 2.1 and 4 in other complex networks.

Contrary to other OSNs such Flickr, LiveJournal and YouTube, link symmetry in Digg network is very lower (38% on average). Since a Digg user relies on his fans to propagate information, making many friends does not increase the number of fans who will digg on the stories submitted or digged by the user. Users with many friends are more dependent on their friends to discover and consequently digg content, whereas the disconnected users are also actively digging stories in spite of the friendships. Since the Digg network provides different features/interfaces for content discovery, friendship relation is no long considered as the primary way to locate content.

Moreover, the interests of OSN users are not evenly spread over different topics. Users have a strong preference about his most favorite topic over the less favorite ones. Based on the above observation, we studied the similarity of the most favorite topic between friends. Our analysis confirms the common hypothesis in OSNs - users are likely to be associated with who are similar (the average similarity hops between two friends is 1.7).

Chapter 12

Collaborative Content Submission and Propagation

The Digg OSN aims to aggregate content published in different places on the Internet into a single website, and the Digg users are considered as the driving force to filter and propagate content in the OSN. Some stories become a great success and obtain many diggs once being published, whereas some other stories are only shared between a small number of users and quickly fall into oblivion. Therefore, it is important to characterize the content submission and propagation patterns in Digg so that we can have a better understanding about its underlying mechanism of disseminating content.

12.1 Story Submission Pattern

As mentioned in Section 10.1, among the approximately 10,000 stories submitted to Digg everyday, only around 150 stories are promoted and hit Digg's front pages in the popular section. Our first objective is to study whether users have equal chances to submit popular stories.

In Fig. 12.1, we present the Lorenz curve[1] [89] regarding the story submission pattern in Digg. As illustrated by the red dotted line in Fig. 12.1, 80% of the Digg stories are submitted by approximately only 25% of the Digg users, suggesting a quite unequal

[1]For every Digg user in our data set, we calculate the number of stories submitted by him and obtain a sequence of values x_i ($1 \leq i \leq N$) that are ranked in non-decreasing order ($x_i \leq x_{i+1}$), where $N = 1,527,818$, and x_i represents the number of stories digged by user i after ordering. The horizontal axis in Fig. 12.1 defines the fraction of Digg users after ordering, and the vertical axis represents the fraction of stories digged by the corresponding fraction of users. The further the Lorenz curves are away from the line of equality, the more unequal the system is. The corresponding Gini coefficient is computed as the ratio of the area that lies between the line of equality and the Lorenz curve over the total area under the line of equality. A low Gini coefficient implies the tendency towards an equal system - a zero Gini coefficient corresponds to complete equality, and vice versa.

system. The above observation conforms to the Pareto principle, i.e. the so-called "80-20 rule" [78], that has been widely observed in economics and sociology. The inequality of story submission becomes more drastic for popular stories: only 2% of the users succeed in submitting popular stories, whereas the majority of Digg submitters fails to promote their content through the Digg network. The above observations indicate that most Digg users (80%) are reading content, rather than actively publishing stories, and that the "flavor" of the Digg OSN is dominated by a small number of people (2%) in the community. The presence of a small group of users who have succeeded in submitting popular stories seemingly suggests that they are the critical users who can effectively disseminating content. However, we have found that these 2% users do not always succeed in submitting popular stories. First of all, there is no correlation between the number of stories a user has submitted and the ratio of stories that will become popular: the correlation coefficient is -0.02. Secondly, the average ratio of submitted popular stories of the 2% users over their total number of submissions is 0.23. Hence, there is no "critical" submitters who can always make a success while submitting stories. As we will show in Section 12.2, to promote a new published content, the timely and collaborative digging activities of Digg users play the important role.

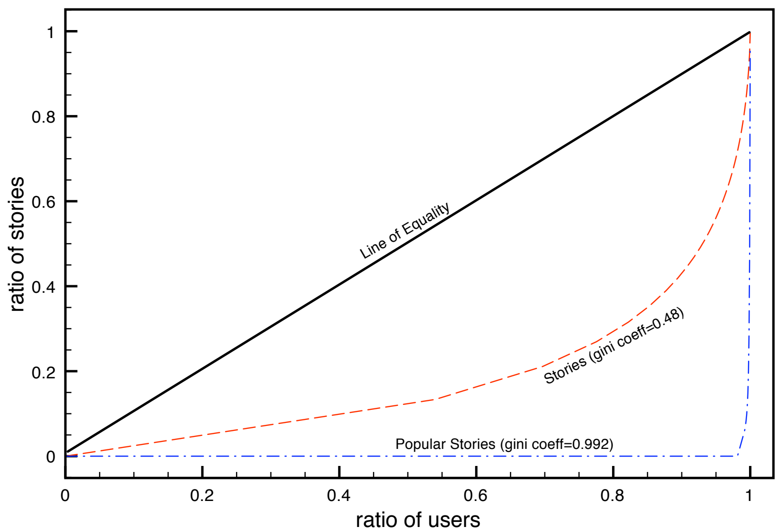


Figure 12.1: Stories submission pattern of the 1.5 million Digg users. Out of the 10,773,307 stories submitted in the Digg network, 115,163 stories succeed to become popular. The ratio of popular stories over the total number of submitted stories is 0.01.

12.2 Collaborative Content Propagation

We study the growth of number of diggs of a story, i.e. the *diggcount* of a story, in the Digg network[2]. We focus on the popular stories in our data set. The number of visitors who has clicked and read a story is defined as the *pageview* of that story. Our analysis shows that the diggcount of a story is highly correlated with its pageview - the correlation coefficient is 0.87. Hence, we consider the number of diggs on a story as a good metric to reflect the popularity of that story.

12.2.1 Story Promotion Duration

The propagation of a popular story consists of two phases: before and after it becomes popular. Firstly, we calculate the promotion duration, T, of a popular story as the time between its publication and promotion. Fig. 12.2 presents the distribution $F_T(t)$ of the promotion duration of the collected 115,163 popular stories.

As revealed from Fig. 12.2, the Digg promotion algorithm manages the published content in such a way that most popular stories are promoted within the first 24 hours from their publication. The average promotion duration is approximately 16.3 hours. In rare cases, stories can be promoted to popular after a long time since its publication (a maximum promotion time in our data collection is found to be over 2 years), which is caused by the change of algorithm.

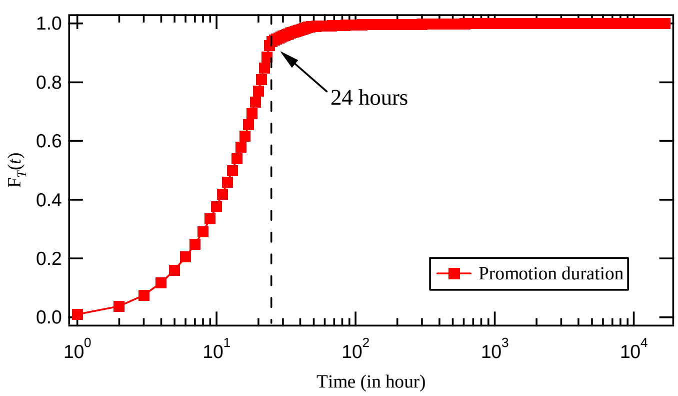


Figure 12.2: $F_T(t) = \Pr[T \leq t]$ of the promotion duration of popular stories (in hours). The average promotion duration of popular stories is approximately, 16.3 hours.

[2]A user is only allowed to digg once on a story.

12.2.2 Collaborative Content Promotion

Popular stories in Digg are submitted by 2% users, as discussed in Section 12.1. The submitters of these stories, however, are not the major factor to determine the success of stories. In the following, we study users' collaborative propagation of stories in Digg.

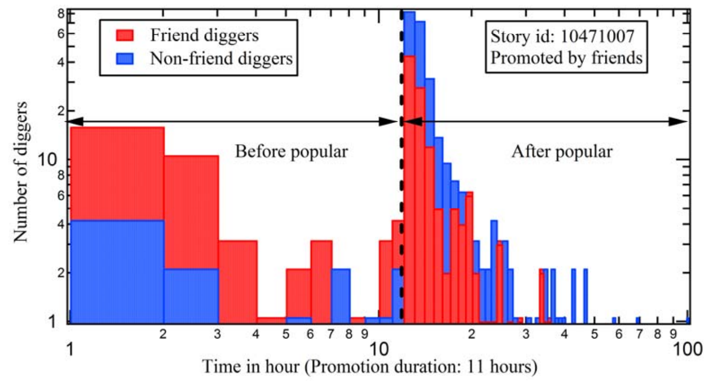
First of all, we calculate the ratio of friend and non-friend diggers among the total number of diggers before a story is promoted. By *friend diggers*, we refer to users that are propagating a story via friendship relations, e.g. friends, friends' friends, etc. By *non-friend diggers*, we mean random users who digg a story without the engagement of the friendship network. Out of the 115,163 popular stories, there are 63,484 stories, in which there are more friends than non-friends promoting the stories before they become popular. The remaining 51,679 stories are mainly promoted by non-friends before they are popular. Table 12.1 presents the average friends and non-friends ratio of the two types of stories. Before stories are promoted to popular, the average ratio of the friend diggers of the 63,484 stories and the non-friend diggers of the 51,679 stories are comparable: 0.72 and 0.77 respectively. On the other hand, after a story is popular, the number of non-friends who are digging the story is much higher than the number of friends, which is not surprising because once a story is placed on the first front page, it is more "visible" to users that are active on the Digg website.

Table 12.1: Ratio of friends and non-friends over the total number of diggers for popular stories

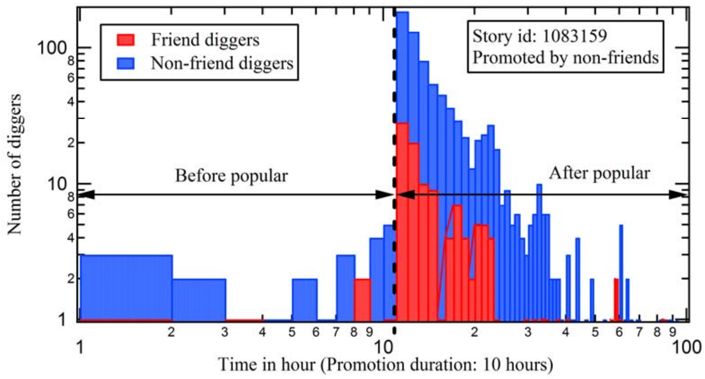
	Before popular		**After popular**	
Average ratio	Friends	Non-friends	Friends	Non-friends
63,484 stories	0.72	0.28	0.25	0.75
51,679 stories	0.23	0.77	0.14	0.86

To study the two story propagation patterns in more detail, we randomly select 20 stories from the 63,484 stories promoted by friends and the 51,679 stories promoted by non-friends. Stories from the same type exhibit similar popularity growth pattern. In Fig. 12.3(a) and (b), the number of friends and non-friends that are digging the two sampled stories are plotted as a function of time. Story 10471007 (Fig. 12.3 (a)) receives most of its diggs via friendship relations before being placed on the first front page, and story 1083159 (Fig. 12.3 (b)) obtains the diggs mainly from non-friends. For both stories, the number of diggs increases drastically after they have been placed on the first front page. Stories placed on the front pages of the popular section quickly obtain the attention of many random users, and non-friends become more influential in propagating the story thereafter.

The "attractive period" of a story, on the other hand, is very short. Most of the diggs are made within the first 2 or 3 hours after stories are placed on the first front page. As time elapses, stories lose their popularity quickly and the number of diggs



(a)

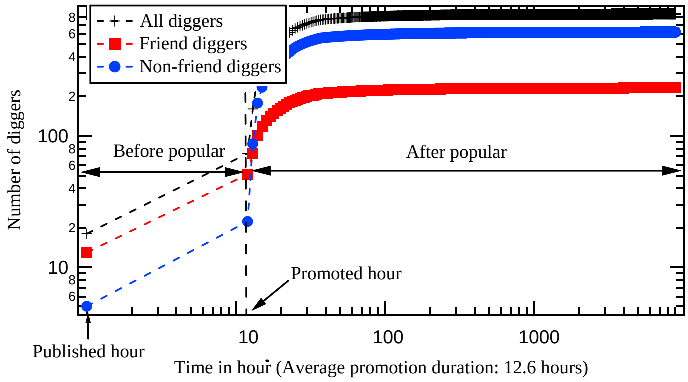


(b)

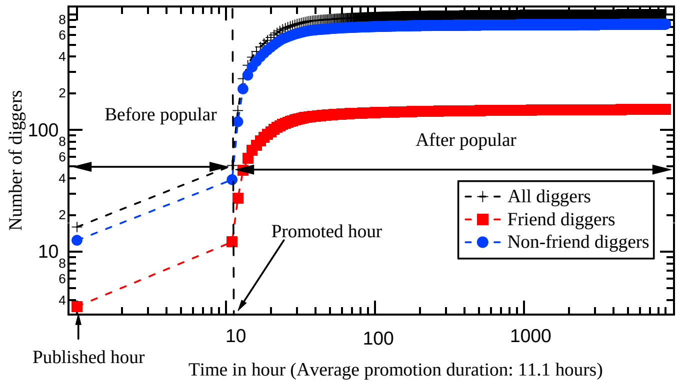
Figure 12.3: Sub-figure (a): Propagation of a popular story promoted by friends. Sub-figure (b): Propagation pattern of a popular story promoted by non-friends. (log-log scale)

becomes stable approximately after 40 hours from their promotion, as illustrated by the aggregated propagation patterns[3] of the two types of stories in Fig. 12.4.

[3]Since stories have different promotion durations, we compute the aggregated number of diggs at the time that a story is published, and the number of diggs of that story when it is promoted to popular. Thus, in Fig. 12.4, only two time points are plotted before stories are promoted.



(a)



(b)

Figure 12.4: Sub-figure (a): The aggregated propagation pattern of the 63,484 popular stories being promoted by friends (average promotion duraton is 12.6 hours). Sub-figure (b): The aggregated propagation pattern of the 51,679 stories being promoted by non-friends (average promotion duraton is 11.1 hours). The accumulated number of diggers is plotted on the vertical axis. (log-log scale)

Among the entire 10 million stories collected in our data set, 88% are news, 8.0% are videos and 4.0% are images. Because the 88% stories contain content that is highly time-relavant, such as breaking news, emerging technologies, their attractiveness is in nature limited in time and becomes obsolete very fast. Moreover, the Digg network manages content in such a way that stories are shifted from the top position to the second position, third position, etc. on the first front page when new stories are promoted. Once a story reaches the last position of the first front page, it will be placed on the first position of the second front page, and so forth. The number of diggs obtained by stories saturates quickly as new stories are promoted and as their ages increase.

We plot the pdf of the diggcount of popular stories on the first five front pages in

Fig. 12.5. The diggcount of stories on front page m, denoted by X_m ($m \geqslant 1$), is the number of diggs a story has obtained before it is shifted to the next page. As shown in Fig. 12.5, the curves are fitted reasonably well with the lognormal distribution[4]. In fact, the lognormal distribution fits the pdfs of story diggcounts appearing on different front pages. Fig. 12.6 presents the fitting parameters (μ and σ) of the lognormal curves for the first 50 front pages. Fig. 12.6 illustrates that after (approximately) the 18^{th} front page, the diggcounts of stories hardly increase. Since a story stays on each front page for about 2.3 hours, it takes approximately 40 hours (see Fig. 12.4) to shift a story to the 18^{th} front page.

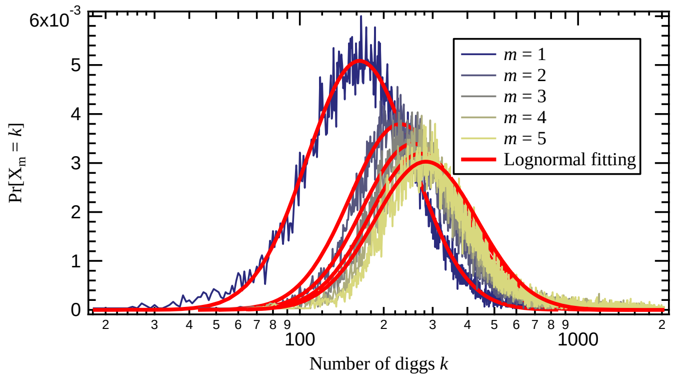


Figure 12.5: Pdf of story diggcount on the first five front pages in the popular section of the Digg website.

The lognormal distribution has been frequently observed in biology and sociology, e.g. in [72], [96] and [127]. As mentioned in [55], a well established process that generates the lognormal distribution is the *law of proportionate effect* that has been considered and proposed by Kapteyn in 1903. The law of proportionate effect specifies that if the growth of the diggcount of a story increases proportionally to the current diggcount, the story diggcount tends to be a lognormal distribution. However, whether the law of proportionate effect applies in the Digg network remains an open question, because content popularity in Digg can be more complex than the process described in [55]. Indeed, content popularity in Digg is determined by two phases: before and after being popular. The Digg promotion algorithm selects stories from the upcoming section and promotes them to the popular front pages, and thus also plays a role. For instance,

[4]The lognormal probability density function with parameters μ and σ is defined as

$$f_{lognormal}(x) = \frac{exp\left[-\frac{(lnx-\mu)^2}{2\sigma^2}\right]}{\sigma x\sqrt{2\pi}} \tag{12.1}$$

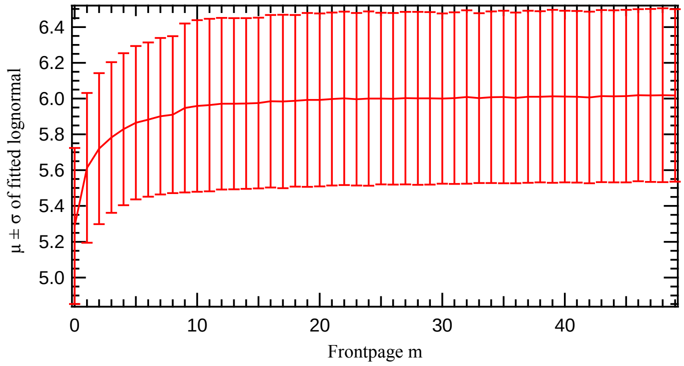


Figure 12.6: Mean and standard deviation of the fitted lognormal curves for the first 50 front pages ($1 \leq m \leq 50$).

Table 12.2: Growth of two types of popular stories

		Before popular (#digg/hour)	After popular (#digg/hour)
Stories promoted by friends	# total diggs	7.3	13.1
	# friends	5.2	5.8
	# non-friends	2.1	11.8
Stories promoted by non-friends	# total diggs	4.9	14.4
	# friends	1.2	5.3
	# non-friends	3.7	13.7

we observed that stories promoted by non-friends requires less digg increment in order to get promoted: on average, stories promoted by friends need to obtain 5.2 friend and 2.1 non-friend diggs per hour before being promoted, as presented in Table 12.2. While stories promoted by non-friends only requires 1.2 friends and 3.7 non-friends per hour (see Table 12.2). The increment of the number of diggs after the stories becoming popular is much higher than the initial phase (before being popular). Therefore, the diversity of diggers may also affect the propagation of a story. If a story is promoted by non-friends, it can be promoted to popular with less increment of diggs. To predict the popularity of Digg stories and explain the lognormal distribution in Fig. 12.5, future research is needed.

12.3 Impairment of the Digg Friendship Relations

We have found that friendship relations only lead to successful dissemination for half of the promoted popular stories. In this section, we investigate the effectiveness of friendship relations during content propagation in more details. Afterwards, we explore the reason that leads to the impairment of the Digg friendship network.

We found that surprisingly, out of the 6,759,937 friendship links, only 121,898 friend pairs have digged at least one common story. The remaining friend pairs never digged the same story in spite of their friendship relation. Furthermore, the 2% friend pairs that have exchanged the same content also perform differently. As plotted in Fig. 12.7, $\Pr[W = w]$ follows a power law distribution, where W denotes the common stories digged by two friends. We see that only a few friend pairs digg many common stories, whereas most of friends do not react with each other very often.

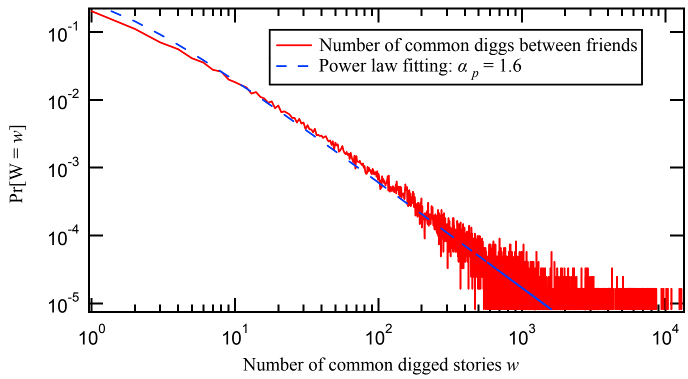


Figure 12.7: Pdf of the number of common stories digged by friends in Digg. The best fitted curve is the power law distribution.

Moreover, friendship networks are assumed to be efficient during content propagation in the sense that users have the ability to "activate" their friends to forward the content further via their social links. Intuitively, users with higher in-degree will be more influential in terms of "activating" their fans when propagating content. However, such an assumption is not true in the Digg OSN. Our study reveals that the active incoming links[5] in Digg is extremely low. Although there exists a strong correlation ($\rho = 0.76$) between the active in-degree and the total in-degree a user has, the linear regression line [119, pp. 31] of $D_{in\_active} = 0.007 D_{in} + 0.2$ suggests an extremely low activation ratio (0.007) of users' incoming degree, where D_{in_active} denotes the number of active in-degree of a random user. Hence, on average, even for a user with 1000 fans, only 7 fans may be activated. Thus, we conclude that there is in fact no significantly influential users in the Digg friendship network.

Last but not least, our analysis shows that the spread of information dies out quickly through the Digg friendship network. On average, the content is not propagated further than 3.9 hops away from the submitter. Furthermore, nearly 70% of the friend diggers

[5]The active incoming links means that the fans digged the story recommended by their friends.

are direct friends of the submitter, and the multi-hop friend relation only increases the number of friend diggers marginally (by 10%). The remaining 20% friend diggers are not activated via the friendship links from the submitter, but by some random users who have digged on the story.

To summarize, our observation differs from previous studies [51] and [95] which illustrated the importance of friendship during content propagation, and questions the effectiveness of friendship relations in online social media aggregator such as the Digg. In Section 11.3, we have empirically substantiated the assumption that friends are sharing similar interests. Although users do have high overlapping interests with their friends, the friendship links do not propagate content efficiently as we have expected (only 2%). In the sequel, we aim to examine other explanations of the poor performance of the friendship relations during content propagation.

First of all, we examine *whether active users that digged many stories are connected with users who are also active.* For a randomly chosen friend pair i and j (j is i's fan), the linear correlation coefficient of the number of stories digged by user i and j is computed as $\rho = 0.05$ by using (8.1). The derived linear correlation coefficient indicates that there is no correlation between friends' digging activities. The uncorrelated digging activity of the friend pairs consequently leads to the impairment of the Digg friendship network, because an active user may not be connected with other active users. A user cannot interact and propagate content collectively with his friends who are not active in Digg.

Secondly, a most distinct feature of the Digg OSN, as discovered in Section 12.2.1, is the time criticality of the content published in Digg. A story needs to be promoted within 24 hours since its publication. Thus, the alignment of friends' digging pattern in the first 24 hours (the promotion threshold of a story) is a necessity to promote a new published story.

From our data set, we study the growth of the friend links in the Digg network over the past 4 years (from July 2005 to May 2009). As shown in Fig. 12.8, the number of established friend links grows steadily over time (bin size in months). The number of active friend pairs[6] in each month, on the other hand, is much lower that the total number of friend links established in Digg, indicated by the red bar. For instance, in June 2007, there exist 2 million friend links, while only 5871 friend pairs are active in that month and the active friend pair ratio is around 0.003. On average, there are 10574 active friend pairs per month[7] and the active friend pair ratio per month is only 0.009.

Since the first 24 hours is critical for stories to get promoted, we also study the

[6]If two friends have digged at least one story, we call them an active friend pair; if two friends have digged at least one common story, we call them an active friend pair digged the same story.

[7]Notice that the number of established friend links L in every month is a random variable. Hence, $Y = E[L_a|L]$ is the random variable equal to average number of active friend pairs given the total number of friend links per month is L, where L_a is the number of active friend pairs per month.

number of friend pairs that will digg on the same story provided they are active on the same day. We take a snapshot of the Digg friendship network during the entire 30 days of June 2007, and further calculated the number of active friend pairs and the ones that digged the same story in this month. Our calculation suggests that, on average, 196 friend pairs are active daily, and approximately one third of the active friend pairs (about 59 friend pairs) have reacted on the same story. We also computed the number of active friend pairs per day and the common stories that they have digged from July 2005 to May 2009 (notice that the total number of established friendship links are not constant during the entire 4 years). We see that the number of friend pairs that digged on the same story daily is strongly correlated with the number of active friend pairs on that day - the correlation coefficient is 0.9. The slope of the linear regression line of the aforementioned two random variables is 0.3, indicating that around 30% of the friend pairs will digg on the same story if they are active on that day. The above analysis indicates that friendship relations are in fact reasonably effective in disseminating content, provided friends' digging activities are aligned on the same day. However, the stringent promotion threshold challenges the alignment of friends' digging activities and further impairs the effectiveness of the friendship relations during content propagation in Digg.

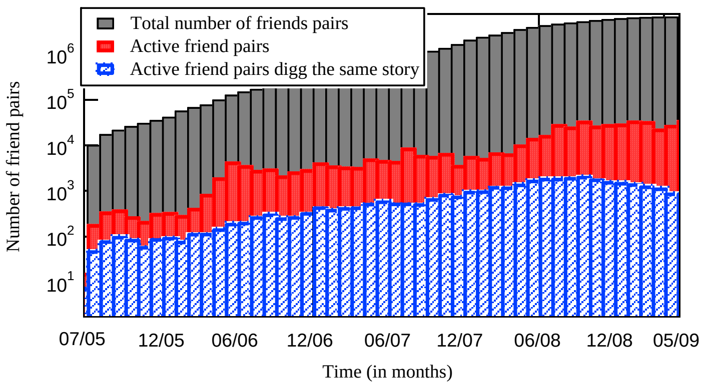


Figure 12.8: Number of established friend links, active friend pairs, and active friend pairs digging the same story from July 2005 to May 2009 (bin size in months). The figure is plotted on a log-linear scale for easier reading.

12.4 Summary

Although OSNs such as the Digg encourages collaborative content publication, not all users are interested in doing so. Most Digg users (85%) are passively reading content rather than publishing stories to Digg. Only 2% of users have succeeded in submitting popular stories. To make newly published content a success, collaborative recommendation and propagation by the Digg users plays an important role. Contrary to previous observation and assumption in OSNs, friendship relations only lead to successful content dissemination in half of the cases. Stories become popular in a short period of time (on average 16.3 hours) and saturate quickly after their promotion (in approximately 40 hours). The above results highlight the fundamental difference between the Digg network and other OSNs on the importance of friendship relations during content propagation. Due to the dynamic nature of the published content, the effectiveness of Digg friendship relations is impaired during content propagation. The effectiveness and success of friendship links in Digg is determined by the alignment of friends' digging activities during the spread of information.

Chapter 13

Conclusion of Part III

The key feature of the Digg OSN is to aggregate, filter and recommend the most valuable social media in a collective way. In this part, we present an in-depth analysis of an emerging online social media aggregator, Digg.com, from various aspects. Our aim is to study the characteristics of users and the impact of friendship relations on propagating information. By employing a simultaneous exploration of the Digg network from different perspectives (i.e. site, story, user and social network perspectives), we are able to crawl the most valid information about the friendship relations, the user activities, and the published content in Digg. Our crawling methodology provides us the advantage of studying the characteristics of both connected and disconnected users in the network, while the commonly used BFS technique is only able to collect informations about the connected nodes following friendship links. We have found a significant amount of users (approximately half of the entire Digg users in our data collection) that do not connect with any other user in the network. However, the performance of the large amount of disconnected users has not been addressed in previous studies due to the limitation of the crawling method. In the following, we highlight our major conclusions in this part.

Major Conclusions

The Digg network exhibit very low fraction of symmetric links (38% on average). Since a Digg user relies on his fans to propagate information, making many friends does not increase the number of fans who will digg on the stories submitted or digged by the user, which contrasts with blog articles that claims the importance of making friends for successful content dissemination. Users with many friends are more dependent on their friends to discover and consequently digg content, whereas the disconnected users are also actively digging stories in spite of the friendships. Since the Digg network provides different features/interfaces for content discovery, friendship relation is no long the only and most convenient way to locate content. The distinct algorithm and interface implemented on the Digg website have impaired the effectiveness of friendships not only for content location, but also during content dissemination. Although friends are sharing common interests with each other, friendship relations only lead to successful dissemination for half of the content. Stories in Digg are promoted to popular within a short period of time (16.3 hours on average). The time criticality of content requires the alignment of friends' digging activities at the right time, which can be achieved only in half of the cases as we have empirically shown. Once stories are placed on the first front page of the popular section, the impact of friends on digging stories is even of less importance, and non-friends become the dominant factor to disseminate content.

Our analysis in this part contrasts with the results reported in previously studied OSNs regarding the importance of friendship network along information propagation. Our analysis suggests that the dynamics of content and user behaviors in OSNs can be greatly influenced by the specific algorithm as well as the web interface provided by the given OSN. The effectiveness of friendship relations is consequently impaired. To our knowledge, our analysis is the first study that questioned the effectiveness of friendships in disseminating content.

As a conclusion, we have discovered distinct user behaviors and content dissemination patterns in Digg, an emerging social media aggregator. Our methodology of crawling the Digg data as well as the analysis provide insights in understanding the underlying principals of similar applications. Meanwhile, the results presented in this part may help the end-users and content owners to spread their content in a more effective way. For instance, users should motivate their fans to be active after the content has been published in order to have a successful promotion.

Future Agenda

Our analysis of the Digg OSN is an ongoing research. We started to design the crawler and initiated the crawling process from January 2009. The analysis presented in this thesis leads to a series of open questions that deserve further and in-depth study. For instance, factors that determine whether a story will be promoted by friends or non-friends are not clear yet. Moreover, to predict the popularity of Digg stories and explain the lognormal distribution in Fig. 12.5, future research is needed.

Open question:
1) What differs between stories promoted by friends and non-friends?
2) What process generates the lognormal distribution in Fig. 12.5?

In the following, we discuss some potential proposals to answer the above open questions. Our first hypothesis is that the topic of content might influence the group of audiences interested in the content. Intuitively, emerging news with an attractive title will quickly grab the attentions of many people, while a funny story might be first shared with friends and then made well-known to the public. The above hypothesis needs to be verified by evaluating the story info (Table 10.3) for the two types of stories separately. Furthermore, it is also interesting to answer the question of "when is the proper time to submit content". Examining the most active (digging) period of online users can provide us indications on the most efficient time to submit content. If a story is submitted when most of the Digg users are offline, the story has less chances to be digged. On the other hand, if many people are actively digging, the story has higher chance to be promoted, mostly likely, by random users.

Furthermore, we aim to develop a model that predicts the content popularity in OSNs after their publication. However, it seems that content popularity in Digg is influenced not only by users' collaborative dissemination, but also the story promotion algorithm implemented in Digg. Stories with a few diggs, but very attractive topics (or content) can be placed on the first front page as well. The artefact caused by the Digg promotion algorithm makes the understanding of Fig. 12.5 challenging. As a starting point, we could replicate the observations found in Table 12.2 in order to infer the story promotion algorithm in Digg. If possible, we suggest to analyze the OSNs, in which the content popularity is completely determined by the users without any pre-defined algorithms.

Chapter 14

Conclusions

In recent decades, the implementation of P2P networking techniques has greatly changed the way that people are consuming information via the Internet. The rapid evolution of P2P networks motivates us to study the information propagation process in different types of P2P networks. Our first approach is to model the content propagation, search, and retrieval process under proper assumptions (part 1). From the exact analysis, we obtain important metrics to evaluate the performance of the proposed algorithms and processes. To construct theoretical models, a good understanding of empirical observations is necessary. However, due to the fast deployment of P2P networking technology, the underlying mechanisms of propagating content in many real-world applications are not clear yet. Consequently, the second objective of this thesis is to carry out empirical studies of real-world P2P applications so that theoretical modeling can be performed in the future. During the empirical analyses, the underlying mechanism and network structure of a commercial P2PTV system have been addressed (part 2); the influence of social relations as well as the unique service interface during information propagation in a large-scale OSN are also investigated (part 3).

Part 1: Modeling content propagation, search and retrieval

We studied gossip-based information propagation in decentralized P2P networks, because gossip-based algorithms are considered as efficient and robust means to disseminate information. The primary goal of gossip-based information propagation, is to disseminate content to every peer in the network, i.e. to achieve the reliability.

In large-scale and highly dynamic P2P systems, in which peers only communicate with a subset of peers (a partial view) in the network, preforming an exact analysis of the gossip-based information dissemination process is computationally not feasible. Because the total number of states (the upper bound) to describe the entire system exactly is $2^{(N+1)^2+N+1}$.

On the other hand, if uniform neighbor selection over the entire network is satisfied, gossip-based information dissemination process (with $N+1$ nodes in the network) can described exactly as an $(N+1)$-state MC. When peers have a complete view of the network, uniform peer selection can be easily satisfied. Under the case where peers have partial views, we assume that the uniformity can be achieve by employing appropriate peerlist exchange schemes, and by properly selecting peers from the local views. Therefore, performing an exact analytic modeling of gossip-based message dissemination schemes under the assumption of uniform selection of multiple neighbors over the entire distributed network is the focus of this thesis. The gossip-based algorithms discussed in this thesis are applicable for both content propagation and content searching in a distributed P2P networks. Different network conditions and peer behaviors are incorporated in the model. Important performance metrics and design parameters are also determined analytically.

After evaluating the performance of the proposed schemes, our main conclusions are: The smart selection algorithm is, in nature, more effective than the blind selection scheme when disseminating content. By using the exact analysis, we have compared the performance difference of the two algorithms quantitatively. To inform the entire network with certain QoS stringency, the smart selection scheme only needs half of the gossiping rounds compared with the blind selection algorithm. By increasing the cooperation probability from $\beta = 0.2$ to $\beta = 1.0$, the mean number of rounds to inform the entire network decreases logarithmically with the same slope for different network sizes, and for both the blind and the smart selection algorithm. Our results about content search also suggest that when a certain speed (number of rounds) is desirable to discover some content, it is less costly for the search process to try to place more content replications l in the network, instead of trying to hit content residing in some nodes only by increasing the number of gossiping-targets k, contacted in each round. The effectiveness of the searching algorithm is impaired by a lower cooperation probability, whereas no significant amount of overhead $(\bar{Y}_{N+1}(l))$ is generated. In view of the trade-off between the overhead and the effectiveness of the search process, the smart selection scheme is more effective with small cooperation probability. With larger cooperation probability, the smart selection scheme is less preferable during the search process, because it incurs more overhead, whereas achieves comparable effectiveness with the blind selection scheme.

Assuming that after the search process, m peers possessing the desired content are discovered. The process of selecting a most nearby peer among the group of m peers for content retrieval is therefore addressed. The closeness between peers is assessed by the metrics of hopcount and delay respectively. Based on the URT model by assigning regular i.i.d. link weights on a dense graph, we evaluated the effect of selecting the nearest peer (in hopcount) exactly, and present the asymptotic analysis of selecting the most nearby peer (in delay). Both results can be used to estimate the number of peers needed to offer certain requirement on the delivering service. And both results suggest that a small peer group is sufficient to offer an acceptable quality of service. We have also performed experiments on measuring the degree and hopcount distribution in the Internet, from which we show the applicability of the URT model, and consequently the usability of the pdfs for both hopcount and delay. With a small group of peers ($m \leq 50$), the URT seems to be a reasonably good model for a P2P network.

Part 2: Empirical analysis - SopCast

Proprietary P2PTV applications have become successful means to deliver video content to end-users. However, their underlying mechanism, topological and traffic impact on the Internet, are largely unknown. We studied a popular commercial P2PTV system called SopCast. Compared with previous studies, our work provides a comprehensive understanding in SopCast. By performing experiments on PlanetLab, we are able to evaluate the overall performance of the SopCast network. The approach and results presented in this empirical work may provide insights in understanding similar applications.

We dissected part of the SopCast protocol by studying different packet lengths and their delivery patterns. Control packets and video packets are separated with respect to their functionalities. Based on the analysis, we have discovered the neighbor communication pattern, the video delivery method, and the three-tier network structure implemented in SopCast. We have found that communication between SopCast peers is decentralized. Given a small network, i.e. $N = 50$, peers discover their neighbors very fast. Video delivery in SopCast is chunk-based. Each video chunk has equal length of 10 kbytes. A peer is free to request multiple blocks from its parent(s). Only a limited number of children can be connected to the SP directly, i.e. the first tier peers. The second tier peers download video content from the peers in the first tier, or between themselves. We also noticed that SopCast does not employ a sophisticated algorithm to select the first tier peers. The first tier peers in the SopCast network are selected based on a first-come-first-serve pattern regardless of their physical distance, network distance, node distance in time, and their upload bandwidth. Once a first tier peer is selected, it remains being connected with the SP during the entire experimental period, unless it leaves the experiment by itself.

We have also evaluated the characteristics of network topology and traffic dynamics of the entire SopCast overlay. While in previous works, obtaining a complete view of the P2PTV network is always a critical issue. We classified the SopCast overlay as a two-layer architecture consisting of a neighbor graph G_N and a video graph G_V. The activation period of the neighbor link and the video link is found to be $\tau_N \sim 2.5$ s, and $\tau_V \sim 2.0$ s respectively. The activation period of the neighbor (video) link is further employed as the reference duration when taking snapshots of the neighbor (video) graph with frequent link establishment and termination. On average, a peer contacts 38 neighbors within 2.5 seconds in the neighbor graph. The topological dynamics of the video graph was also evaluated. The incoming degree distribution of the video graph can be modeled as a random graph with very low link density $p \ll 1.0$, meaning that SopCast peers are not greedy when connecting to parents for video downloading. On average, the number of parents of a SopCast peer to download video content from is two. On the other hand, the existence of super peers, which upload a lot and support many children for video downloading is also observed. A super peer sacrifices a large amount of upload capacity to its many children.

Our study also revealed that SopCast employs very simple algorithm to select the first tier peers and to connect to existing peers for video content downloading, i.e. first-come-first-serve. Some of the first tier peers can be very selfish. Because they may not provide many video content to others, even though they download video packets from a good resource, i.e. the SP. Peers who join the network earlier attach to the first tier peers, while the latter ones connect to the second tier peers to download video content. It seems that bandwidth utilization is not optimized in SopCast, because peers with high bandwidth do not necessarily upload more.

Our analysis about the SopCast traffic dynamics has questioned the efficiency of real-world P2PTV applications such as SopCast. We believe that to design a scalable and optimal system, the following issues can be improved in SopCast. First of all, the selection of the first tier peers should be more sophisticated. Instead of the current first-come-first-serve pattern, the uploading performance of the first tier peers need to be taken into account. In all of our experiments, once a first tier peer is selected, it will connect with the SP during the entire experimental period (unless it leaves the network). To improve the global performance of the SopCast network, we suggest to employ a dynamic first tier peers selection algorithm. If the first tier peers cannot offer good upload throughput, their connections with the SP should be terminated. Re-selecting better peers in the first tier should be performed. Moreover, peers in a P2P network must be mutually beneficial. The SopCast network needs to motivate its peers to upload more if they are downloading from a good resource.

In general, the methodologies discussed in this empirical analysis provides a better understanding of the SopCast commercial P2PTV system, and can be useful to understand similar P2PTV applications. Although we have performed the empirical study in a careful and thorough way, our observations and claims are not the exact description of the protocol. Moreover, performing experiments on PlanetLab provides us the advantage of evaluating the performance of the entire SopCast network. However, the network size is limited to several hundred. Thus, the performance of a P2PTV system that is at large-scale cannot be reflected.

Part 3: Empirical analysis - Digg OSN

OSNs are emerging applications that combine the idea of networking and social phenomena. The key feature of the Digg OSN is to aggregate, filter and recommend the most valuable social media in a collective way. Our aim was to study the distinct user behaviors as well as the influence of friendship relations during content propagation in Digg, one of the biggest social news aggregator webpage. We highlight our major conclusions as follows.

The data collection methodology presented in this thesis differs from existing techniques that crawl large-scale OSNs by following the friendship links. The collected data set allows us to investigate the complete snapshot of Digg from various aspects and study the characteristics of both connected and disconnected users in the network. In fact, there exists a significant amount of users (approximately half of the entire Digg users in our data collection) that do not connect with any other user in the network. These disconnected users are actively digging stories in the Digg network, which questions the necessity of making friends in order to find and propagate content in Digg. We attribute the above observation to the fact that the Digg website provides different features/interfaces for content discovery. Thus, friendship relation is no long considered as the only and most efficient way to locate content.

We also delved into common hypotheses of OSNs such as "users are likely to be associated with similar users" and "friendship relations are the key factor to propagate information". We found that the interests of Digg users are not evenly spread over different topics, and that users have a strong preference about his most favorite topic over the less favorite ones. Based on the above observation, we measured the similarity between friends in the Digg network. Our analysis substantiated the common hypothesis of friends are sharing similar interests with each other. Furthermore, although friends are sharing common interests, friendship relations only lead to successful dissemination for half of the content. The other half of the stories are promoted by non-friends. Once stories are placed on the first front page of the popular section, the impact of friends on digging stories is even of less importance, and non-friends become the dominant factor to disseminate content.

The above analysis contrasts with the results reported in previously studied OSNs regarding the importance of friendship network along information propagation. We believe that the dynamics of content and user behaviors in OSNs can be greatly influenced by the specific algorithm as well as the web interface provided by the given OSN. For instance, the Digg stories are promoted to popular within a short period of time (16.3 hours on average) and saturate quickly after their promotion (in approximately 40 hours). The time criticality of content requires the alignment of friends' digging activities at the right time, which can be achieved only in half of the cases as we have shown empirically. The effectiveness of friendship relations in the spread of information is consequently impaired.

As a conclusion, the analyses and results presented in this thesis defined basic observations and measurements to understand the underlying mechanism of disseminating content in current online social news aggregators. Our findings have underlined the importance of service interfaces on content propagation as well as on the effectiveness of friendship relations during content discovery and propagation. Meanwhile, the presented observations may provide insights to the end-users and content owners to spread their content in a more effective way, e.g. users should motivate their fans to be active after the content has been published in order to have a successful promotion. On the other hand, our analyses also provide technical implications to system designers and developers to properly design their web applications, e.g. a specific service interface can have great influence on the efficiency of friendship relations as well as the distinct performance of the OSN.

Summary

In the course of this thesis, we studied the information propagation process in different P2P networking applications from the perspectives of both theoretical and empirical analysis. The analytic modeling was performed with the gossip-based information propagation algorithms that are widely employed in today's P2P networks. Since P2P networking is rapidly evolving, we also investigated two emerging P2P applications that are dedicated for P2P streaming delivery and online social networking. The empirical analyses conducted with the SopCast and Digg network disclosed important design issues and user characteristics. To summarize, we have solved different questions in different types of P2P networks, while these analyses also raise new questions and provide directions for future research.

Appendix A

The Occupancy Problem

A.1 The Occupancy Problem - Scenario 1

The classical occupancy problem considers random placement of m balls into n bins in a balls and bins model [62]. In this thesis, we assume that there are r groups of k balls and n bins. We randomly throw the r groups of k balls over the n bins. The k balls in the same group are placed in such a way that no two balls go into the same bin. Due to the random placement of balls, there may be empty bins after the placement. A bin can be *occupied* by one or more balls. Placement of balls belonging to different groups are independent and random. We seek the probability that exactly m bins are empty after the placement.

Following the approach in [62], the probability that all n bins are occupied, denoted by $p_0(r, n, k)$, is

$$p_0(r, n, k) = 1 - \Pr\left[\textit{at least one bin is empty}\right] \tag{A.1}$$

To place r groups of k balls to n bins, leaving i preassigned bins empty, there are $\binom{n-i}{k}^r$ ways. The total number of ways of placing r groups of k balls to n bins is $\binom{n}{k}^r$. Further, there are $\binom{n}{i}$ ways to choose i preassigned bins. Let S_i be the event that i bins are empty, the probability that S_i occurs is $\binom{n}{i}\frac{\binom{n-i}{k}^r}{\binom{n}{k}^r}$. Invoking the inclusion-exclusion principle [119, p. 12], we obtain $p_0(r, n, k)$ as

$$\begin{aligned} p_0(r, n, k) &= 1 - \sum_{i=1}^{n-k} (-1)^{i-1} \binom{n}{i} \frac{\binom{n-i}{k}^r}{\binom{n}{k}^r} \\ &= \frac{1}{\binom{n}{k}^r} \sum_{i=0}^{n-k} (-1)^i \binom{n}{i} \binom{n-i}{k}^r \end{aligned} \tag{A.2}$$

Now consider the case in which the r groups of k balls are placed in such a way that exactly m out of the n bins are empty. The m bins can be chosen in $\binom{n}{m}$ different ways. The number of configurations leading to such placement is $\binom{n-m}{k}^r p_0(r, n-m, k)$. Dividing by the total way to place the r groups of k balls to n bins, $\binom{n}{k}^r$, the probability $p_m(r, n, k)$ that exactly m bins are empty is computed as

$$p_m(r, n, k) = \binom{n}{m} \frac{\binom{n-m}{k}^r}{\binom{n}{k}^r} p_0(r, n-m, k) \tag{A.3}$$

This argument in (A.3) is confined to $n \geqslant k$ since it is not possible to place k balls to n bins, with no two balls, in the same bin, if $n < k$.

With $k = 1$, the probability in (A.2) can be simplified to

$$\begin{aligned} p_0(r, n, 1) &= \frac{1}{n^r} \sum_{i=0}^{n} (-1)^i \binom{n}{i} (n-i)^r \\ &= \frac{n!}{n^r} S_r^{(n)} \end{aligned} \tag{A.4}$$

Let $j = n - i$, we have $\frac{1}{n^r} \sum_{j=0}^{n} (-1)^{n-j} \binom{n}{j} j^r = \frac{n!}{n^r} S_r^{(n)}$, where $S_r^{(n)}$ are the Stirling numbers of the second kind [38, section 24.1.4]. If $r < n$, $S_r^{(n)} = 0$. Consequently, (A.3) is simplified to

$$\begin{aligned} p_m(r, n, 1) &= \binom{n}{m} \left(\frac{n-m}{n}\right)^r p_0(r, n-m, 1) \\ &= \binom{n}{m} \frac{(n-m)!}{n^r} S_r^{(n-m)} \end{aligned} \tag{A.5}$$

A.2 The Occupancy Problem – Scenario 2

Considering the same model in Appendix A.1, we modify the problem configuration. We are placing r groups of k balls to n bins, $(1 \leq r \leq n)$. The n bins consist of a number of red and white bins. The position of the red and white bins are predefined, with the first m bins colored by red, and the last $n-m$ colored by white. The r groups of balls are numbered from 1 to r, and the white bins are numbered from 1 to $n-m$. There is no numbering of the m red bins, see Fig. A.1. Assuming the number of the groups of balls equals the number of the white bins, the r groups of balls and the $n-m$ white bins eventually have the same numbering. Furthermore, we assume that a group of balls with the number i $(1 \leq i \leq r)$ cannot be placed to the white bin that has the same numbering. For instance, balls from group 1 cannot be placed in bin number 1, balls from group 2 cannot be placed in bin number 2, etc. As a result, the k balls from the same group are randomly placed to the remaining $n-1$ bins, and no two balls go into the same bin. Placement of balls belonging to different groups are independent and

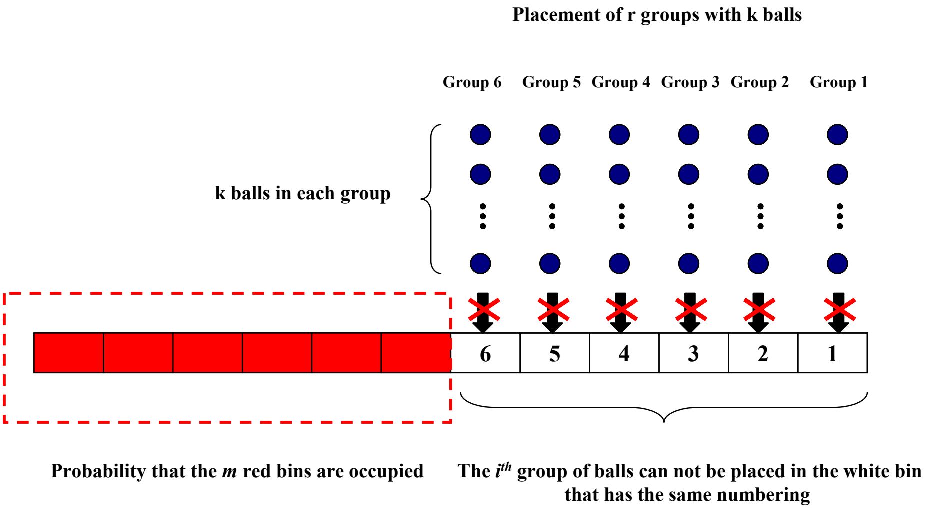


Figure A.1: Random and independent placement of r groups of k balls, with no group of balls being placed in the bin which have the same numbering as itself.

random. We seek the probability that at least the m red bins are occupied, denoted by $p_{\bar{m}}(r, n, k)$.

Denote δ the maximum number of allowed empty red bins in this scenario. If one of the m red bins is empty, the r groups of k balls should be placed to the remaining $n-2$ bins, excluding the bins that have the same numbering as the groups of balls. With the $\binom{m}{1}$ ways to choose an empty bin from the m bins, there are $\binom{m}{1}\binom{n-2}{k}^r$ configurations leading to such placement. In case there are two empty bins out of the m red ones, the balls are placed to the remaining $n-3$ bins, resulting in $\binom{m}{2}\binom{n-3}{k}^r$ ways accordingly. Denote S_i, the event that i out of the m red bins are empty, the r groups of k balls can only be placed to the rest $n-1-i$ bins, resulting in $\binom{n-1-i}{k}^r$ arrangements. The event that at least the m red bins are occupied, $A_{\bar{m}}(r, n, k)$, is given by applying the inclusion-exclusion principle.

$$\begin{aligned} A_{\bar{m}}(r, n, k) &= \binom{n-1}{k}^r - S_1 + S_2 - \ldots \\ &= \sum_{i=0}^{\delta} (-1)^i \binom{m}{i} \binom{n-1-i}{k}^r \end{aligned} \tag{A.6}$$

The limiting values of δ depend on the relation between $n-1-m$ and k. Assuming $n-1-m < k$, this condition implies that the number of the balls from the same group is more than the $n-m-1$ white bins. Therefore, to place the k balls from the same group to different bins (recall that the k balls from the same group can not go to the same bin), $k-(n-1-m)$ red bins have to be occupied during the placement. Thus, the maximum number of empty red bins is confined to $\delta = n-1-k$. On the other hand, when $n-1-m \geq k$, there are enough white bins to place the k balls from the same group (by placing all the k balls into the $n-m-1$ white bins.). Hence, the maximum number of empty red bins is $\delta = m$.

Taking into account the above analysis, $p_{\bar{m}}(r,n,k)$ should be discussed under two conditions

$$p_{\bar{m}}(r,n,k) = \begin{cases} \frac{1}{\binom{n-1}{k}^r} \sum_{i=0}^{m} (-1)^i \binom{m}{i} \binom{n-1-i}{k}^r & \text{if } n-1-m \geq k \\ \frac{1}{\binom{n-1}{k}^r} \sum_{i=0}^{n-1-k} (-1)^i \binom{m}{i} \binom{n-1-i}{k}^r & \text{if } n-1-m < k \end{cases} \tag{A.7}$$

where $\binom{n-1}{k}^r$ is the total number of ways to place the r groups of k balls to $n-1$ bins. (A.7) can be directly applied by the blind neighbor selection algorithm.

A.3 The Extended Rumor Spreading Problem

In this section, we study the extended scenario of the rumor spreading problem in [104], with the assumption of selecting k neighbors out of the $N+1$ nodes.

A.3.1 The occupancy problem

To model the above mentioned problem exactly, we slightly modify the scenario in Fig. A.1. We remove the constraint of numbered groups of balls and numbered white bins. There are m red bins and $n-m$ white bins. Both red bins and white bins are treated equally during the placement of balls. The r groups of k balls are placed randomly to the n bins, with no two balls from the same group going to the same bin. Balls belonging to different groups are placed to the n bins at random and independent of the choice of other groups. Similarly, the problem is to find the probability that at least the m red bins are occupied, denoted by $\gamma_{\bar{m}}(r,n,k)$, after the random placement of balls.

This approach follows the same steps in Appendix A.2. We examine $\gamma_{\bar{m}}(r,n,k)$ under two conditions. When $n-m \geq k$, the maximum number of empty red bins is m. While with $n-m < k$, the maximum number of empty red bins can only be $n-k$, because $k-(n-m)$ red bins have to be occupied. Otherwise, the k balls from a same group cannot be placed to k different bins successfully. Therefore, when using the inclusion-exclusion principle, the maximum number of allowed empty red bins is $n-k$.

In the event that one of the m red bins is empty, the r groups of k balls are placed to the remaining $n-1$ bins. This can be done with $\binom{n-1}{k}^r$ ways. Similarly, in case two out of the m red bins are empty, there are totally $\binom{n-2}{k}^r$ ways leading to such placement. Assuming that i out of the m red bins are empty, denoted by S_i, there are $\binom{m}{i}\binom{n-i}{k}^r$

arrangement leading to such event, where $\binom{m}{i}$ is the configurations to choose i bins out of the m red ones. Hence, the probability that at least all m red bins are occupied is computed by using the inclusion-exclusion principle:

$$\gamma_{\bar{m}}(r,n,k) = \begin{cases} \frac{1}{\binom{n}{k}^r} \sum_{i=0}^{m} (-1)^i \binom{m}{i} \binom{n-i}{k}^r & \text{if } n-m \geq k \\ \frac{1}{\binom{n}{k}^r} \sum_{i=0}^{n-k} (-1)^i \binom{m}{i} \binom{n-i}{k}^r & \text{if } n-m < k \end{cases} \tag{A.8}$$

where $\binom{n}{k}^r$ is the total number of ways to place the r groups of k balls to n bins.

A.3.2 The transition probabilities

The MC moves from state i to state j if there are exactly $z = j-i$ new nodes, selected by the i informed ones. With the modified occupancy problem, we can solve the transition probabilities P_{ij} by substituting $m = z$, $n = j$, $r = i$ in (A.8).

From the approach of (A.8), we have

$$P_{ij} = \begin{cases} \frac{\binom{N+1-i}{j-i}}{\binom{N+1}{k}^i} \sum_{t=0}^{j-i} (-1)^t \binom{j-i}{t} \binom{j-t}{k}^i & \text{if } i \geqslant k \text{ and } i \leqslant j \leqslant \min\{N+1, i(k+1)\} \\ \frac{\binom{N+1-i}{j-i}}{\binom{N+1}{k}^i} \sum_{t=0}^{j-k} (-1)^t \binom{j-i}{t} \binom{j-t}{k}^i & \text{if } i < k \text{ and } k \leqslant j \leqslant \min\{N+1, i(k+1)\} \\ 0 & \text{otherwise} \end{cases} \tag{A.9}$$

When $k = 1$, (A.9) reduces to (2.1).

A.4 Diagonalizability of matrix P

If P is diagonalizable, the r-step transition probability matrix P^r is consequently derived as

$$P^r = X diag(\lambda_k)^r Y^T \tag{A.10}$$

in which $diag(\lambda_k)$ is the diagonal matrix whose diagonal entries are the corresponding eigenvalues, and X and Y consist of columns of the right– and left-eigenvectors. An explicit form of (A.10) follows from [119, p. 183] as

$$P^r = \sum_{k=1}^{N+1} \lambda_k^r x_k y_k^T \tag{A.11}$$

where x_k and y_k are the right and left-eigenvectors associated with λ_k (both are column vectors with $N+1$ entries). Therefore, (A.11) is further decomposed as

$$P^r = u\pi + \sum_{k=2}^{N+1} \lambda_k^r x_k y_k^T \simeq u\pi + \lambda_2^r x_2 y_2^T + O(\lambda_3^r) \tag{A.12}$$

where $y_1^T = \pi$ and $x_1 = u$ (with $u^T = [1\ 1\ 1\ ...\ 1]$) are the corresponding steady state eigenvectors, associated with the largest eigenvalue $\lambda_1 = 1$.

Next, we discuss the diagonalizability of matrix P with respect of the eigenvalues that it possesses. The matrix Y^T is the inverse of the matrix X, which implies that X is non-singular. Hence, the matrix X should possess a complete set of $N+1$ linearly independent (right-) eigenvectors $\{x_1, x_2,\ ...,\ x_{N+1}\}$. In the triangular matrix P, the eigenvalues of P are just the diagonal elements. The matrix P is diagonalizable if and only if

$$geo\ mult_P(\lambda_k) = alg\ mult_P(\lambda_k) \tag{A.13}$$

with $1 \leq k \leq N+1$, and where $geo\ mult_P(\lambda_k)$ is the *geometric multiplicity*[1] of λ_k, and $alg\ mult_P(\lambda_k)$ is the *algebraic multiplicity*[2] of λ_k, as introduced in [92].

If all the $N+1$ eigenvalues of P are distinct, then $\{x_1, x_2,\ ...,\ x_{N+1}\}$ is a linearly independent set. In case the matrix P does not possess $N+1$ distinct eigenvalues, it is also possible to diagonalize P. The above statement is true if the number of linearly independent eigenvectors, associated with λ_k, equals the algebraic multiplicity of λ_k. To compute the eigenvector of λ_k, we follow

$$(P - \lambda_k I)\, x = 0$$

where I is the identity matrix. For simplicity, we denote by m_k the algebraic multiplicity of λ_k. If the rank of $P - \lambda_k I$ is $N+1-m_k$, meaning $rank(P - \lambda_k I) = N+1-m_k$, the matrix $P - \lambda_k I$ will have has m_k linearly independent eigenvectors. In case the matrix $P - \lambda_k I$ does not possess m_k linearly independent eigenvectors, it can be reduced to a *Jordan canonical form*, as introduced in [92].

In our case, the matrix P studied in Section 3.3 is not always diagonalizable as explained in the sequel. Under the smart selection algorithm with $\beta = 1$, all the first N diagonal elements are zeros, except for the last row of $P_{N+1,N+1} = 1$. Thus, there are only two distinct eigenvalues, namely $\lambda_1 = 1$ and $\lambda_2 = \lambda_3 = ... = \lambda_{N+1} = 0$. The matrix P is diagonalizable if and only if there are N linearly independent eigenvectors associated with the eigenvalue of $\lambda = 0$, which requires that

$$Px = 0$$

The rank of the matrix P can never be 1. Therefore, it is not possible to obtain N linearly independent eigenvectors associated with $\lambda = 0$. As a result, the matrix P is not diagonalizable because the matrix X is singular.

Under the blind selection algorithm, the diagonal elements in the first k rows are zeros when $\beta = 1$. The remaining entries on the diagonal are non-zeros, computed from

[1] The *geometric multiplicity* of λ_k, denoted by $geo\ mult_P(\lambda_k)$, is the maximal number of linearly independent eigenvectors associated with λ.

[2] The *algebraic multiplicity*, denoted by $alg\ mult_P(\lambda_k)$, is the number of times λ_k is repeated in the set of eigenvalues of matrix P.

(3.3), leading to $N+1-k$ distinct eigenvalues $(\lambda_1, \lambda_2, ...\lambda_{N+1-k})$ of multiplicity 1 and one eigenvalue $\lambda_{N+k} = 0$ of multiplicity k. Notice that the rank of the matrix $P - \lambda_{N+k} I$ is $N+1-k$. Equation (3.8) can be applied in this case since P is diagonalizable.

With the general case of $0 < \beta < 1$, under both the blind and smart selection algorithms, the index of the non-zero elements in each row vector $[P_{i1}\ P_{i2}\ ...\ P_{i,N+1}]$ is bounded by $i \leqslant j \leqslant \min\{i(k+1),\, N+1\}$. The diagonal elements are non-zeros. However, the $N+1$ eigenvalues are not always distinct, depending on the value of N, k and β. When there are multiple eigenvalues, the matrix P is diagonalizable only when the structure of P satisfies the relation (A.13). Discussing the particular matrix structure that leads to a diagonalizable matrix P given multiple eigenvalues has much higher complexity, and is out of the scope of the thesis.

Appendix B

Abbreviations

API	Application Programming Interface
AS	Autonomous System
BFS	Breath-first Search
Cdf	Cumulative Distribution Function
CDN	Content Distribution Network
EST	Eastern Standard Time
FTP	File Transfer Protocol
DHT	Distributed Hash Table
GB	Giga Bytes
HTTP	Hypertext Transfer Protocol
ID	Identification Number
IP	Internet Protocol
ISP	Internet Service Provider
MC	Markov Chain
OSN	Online Social Network
PC	Personal Computer
Pdf	Probability Density Function
Pgf	Probability Generating Function
P2P	Peer-to-Peer
P2PTV	Peer-to-Peer Television
QoS	Quality of Service
SI	Susceptible-Infected
SIR	Susceptible-Infected-Recovered
SIS	Susceptible-Infected-Susceptible
SP	Source Provider
SPT	Shortest Path Tree
TTL	Time-To-Live
UDP	User Datagram Protocol
URL	Uniform Resource Locator
URT	Uniform Recursive Tree
VoD	Video-on-Demand
VoIP	Voice Over Internet Protocol
WWW	World Wide Web
XML	Extensible Markup Language

Bibliography

[1] Alexa. http://www.alexa.com/siteinfo/digg.com.

[2] AWK. http://www.gnu.org/manual/gawk/gawk.html.

[3] Bandwidth share policy on PlanetLab. http://www.planet-lab.org/doc/BandwidthLimits.

[4] Bash scripting. http://tldp.org/LDP/abs/html.

[5] BitTorrent. http://www.bittorrent.com.

[6] CoMon user interface. http://comon.cs.princeton.edu.

[7] Delicious. http://www.delicious.com.

[8] Digg. http://digg.com.

[9] Emule project. http://www.emule-project.net/home/perl/general.cgi?l=1.

[10] Facebook. http://www.facebook.com.

[11] Flickr. http://www.flickr.com.

[12] Gnutella development forum. http://www.gnutellaforums.com.

[13] Hi5. http://hi5.com.

[14] Internet traffic study 2008/2009. http://www.ipoque.com/resources/internet-studies/internet-study-2008_2009.

[15] iPlane. http://iplane.cs.washington.edu.

[16] Java. http://java.sun.com.

[17] Kazaa. Homepagehttp://www.kazaa.com.

[18] Last.fm. http://www.last.fm.

[19] LinkedIn. http://www.linkedin.com.

[20] LiveJournal. http://www.livejournal.com.

[21] MySpace. http://www.myspace.com.

[22] Napster. http://www.napster.com.

[23] Orkut. http://www.orkut.com.

[24] Perl. http://www.perl.org.

[25] PlanetLab. http://www.planet-lab.org/hosting.

[26] PPLive. http://www.pplive.com.

[27] PPStream. http://www.ppstream.com.

[28] Python script. http://svn.planet-lab.org/browser/util-vserver-pl/tags/util-vserver-pl-0.3-16/python/bwlimit.py.

[29] Reddit. http://www.reddit.com.

[30] SixDegrees. http://sixdegrees.com.

[31] SopCast. http://www.sopcast.org.

[32] Tcpdump. http://www.tcpdump.org.

[33] TVAnts. http://tvants.en.softonic.com.

[34] TVU. http://www.tvunetworks.com.

[35] Twitter. http://twitter.com.

[36] UUSee. http://www.uusee.com.

[37] YouTube. http://www.youtube.com.

[38] M. Abramowitz and I. A. Stegun. *Handbook of mathematical functions with formulas, graphs, and mathematical tables.* Dover publications, 1964.

[39] E. J. Adams. Age Matters on Social Networks. *Thought Leaders*, 2008.

[40] E. Adar and B.A. Huberman. Free riding on gnutella. *First Monday*, 5(10):2–13, 2000.

[41] Y. Y. Ahn, S. Han, H. Kwak, S. Moon, and H. Jeong. Analysis of topological characteristics of huge online social networking services. In *Proceedings of the 16th international conference on World Wide Web*, page 844. ACM, 2007.

[42] S. Ali, A. Mathur, and H. Zhang. Measurement of commercial peer-to-peer live video streaming. *Proc. of Workshop in Recent Advances in Peer-to-Peer Streaming*, 2006.

[43] L. A. N. Amaral, A. Scala, M. Barthelemy, and H. E. Stanley. Classes of small-world networks. *Proceedings of the National Academy of Sciences of the United States of America*, 97(21):11149, 2000.

[44] L. Backstrom, D. Huttenlocher, J. Kleinberg, and X. Lan. Group formation in large social networks: membership, growth, and evolution. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, page 54. ACM, 2006.

[45] A. L. Barabási and R. Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999.

[46] J. A. Barnes. Class and committees in a Norwegian island parish. *Human relations*, 7(1):39, 1954.

[47] F. Benevenuto, T. Rodrigues, M. Cha, and V. Almeida. Characterizing user behavior in online social networks. In *Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference*, pages 49–62. ACM, 2009.

[48] K.P. Birman, M. Hayden, O. Ozkasap, Z. Xiao, M. Budiu, and Y. Minsky. Bimodal multicast. *ACM Transactions on Computer Systems (TOCS)*, 17(2)(2):4188, 1999.

[49] M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron, and A. Singh. SplitStream: high-bandwidth multicast in cooperative environments. In *Proceedings of the nineteenth ACM symposium on Operating systems principles*, page 313. ACM, 2003.

[50] M. Cha, H. Kwak, P. Rodriguez, Y.Y. Ahn, and S. Moon. Analyzing the video popularity characteristics of large-scale user generated content systems. *IEEE/ACM Transactions on Networking (TON)*, 17(5):1357–1370, 2009.

[51] M. Cha, A. Mislove, and K. P. Gummadi. A measurement-driven analysis of information propagation in the flickr social network. In *Proceedings of the 18th international conference on World wide web*, pages 721–730. ACM, 2009.

[52] A. Clauset and C. Moore. Accuracy and scaling phenomena in Internet mapping. *Physical Review Letters*, 94(1):18701, 2005.

[53] B. Cohen. Incentives build robustness in BitTorrent. In *Workshop on Economics of Peer-to-Peer systems*, volume 6, 2003.

[54] T. H. Cormen. *Introduction to algorithms.* The MIT press, 2001.

[55] E.L. Crow and K. Shimizu. *Lognormal distributions: Theory and applications.* CRC, 1988.

[56] D. J. Daley and J. Gani. *Epidemic modelling: An Introduction.* Cambridge University Press, 1999.

[57] A. Demers, D. Greene, C. Hauser, W. Irish, J. Larson, S. Shenker, H. Sturgis, D. Swinehart, and D. Terry. Epidemic algorithms for replicated database maintenance. *In Proceedings of the Sixth Annual ACM Symposium on Principles of Distributed Computing (PODC)*, pages 1–12, 1987.

[58] D. W. Drezner and H. Farrell. The power and politics of blogs. In *American Political Science Association*, volume 2, 2004.

[59] H. Ebel, L.I. Mielsch, and S. Bornholdt. Scale-free topology of e-mail networks. *Physical Review E*, 66(3):35103, 2002.

[60] P.T. Eugster, R. Guerraoui, S.B Handurukande, P. Kouznetsov, and A.-M. Kermarrec. Lightweight probabilistic broadcast. *ACM Transactions on Computer Systems (TOCS)*, 21(4)(4):341–374, 2003.

[61] P.T. Eugster, R. Guerraoui, A.-M. Kermarrec, and L. Massoulié. From epidemics to distributed computing. *IEEE computer*, 37(5):60–67, 2004.

[62] W. Feller. *An introduction to probability theory and its applications.* Vol. I, 3rd ed., John Wiley & Sons,, 1968.

[63] A. J. Ganesh, L. Massoulie, and D. Towsley. The effect of network topology on the spread of epidemics. In *Proceedings IEEE INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies*, volume 2, 2005.

[64] A.J. Ganesh, A.-M. Kermarrec, and L. Massoulié. Peer-to-peer membership management for gossip-based protocols. *IEEE transactions on computers*, 52(2):139–149, 2003.

[65] M. S. Granovetter. The strength of weak ties. *American Journal of Sociology*, 78(6)(6):1360, 1973.

[66] S. Guha, N. Daswani, and R. Jain. An experimental study of the skype peer-to-peer VOIP system. In *In Proceedings of the 5th International Workshop on Peer-to-Peer Systems (IPTPS 06)*, volume 6. Citeseer, 2006.

[67] K. P. Gummadi, R. J. Dunn, S. Saroiu, S. D. Gribble, H. M. Levy, and J. Zahorjan. Measurement, modeling, and analysis of a peer-to-peer file-sharing workload. *In Proceedings of the 19th ACM Symposium of Operating Systems Principles (SOSP)*, page 329, 2003.

[68] Z.J. Haas, J.Y. Halpern, and L. Li. Gossip-based ad hoc routing. *IEEE/ACM Transactions on Networking (TON)*, 14(3):491, 2006.

[69] S. B. Handurukande, A. M. Kermarrec, F. Le Fessant, L. Massoulié, and S. Patarin. Peer sharing behaviour in the edonkey network, and implications for the design of server-less file sharing systems. In *EuroSys Conference (EuroSys'06)*, 2006.

[70] X. Hei, C. Liang, J. Liang, Y. Liu, and K.W. Ross. A measurement study of a large-scale P2P IPTV system. *IEEE Transactions on Multimedia*, 9(8):16721687, 2007.

[71] R. A. Hesel and R. C. Williams. Social Networking Sites and College-Bound Students. *studentPoll: http://professionals.collegeboard.com/data-reports-research/trends/studentpoll/social-networking*, (7)2, 2008.

[72] B.A. Huberman and L.A. Adamic. Growth dynamics of the world wide web. *Nature*, 401:131, 1999.

[73] M. Janic and P. Van Mieghem. On properties of multicast routing trees. *International Journal of Communication Systems*, 19(1):95–114, February 2006.

[74] M. Jelasity and O. Babaoglu. T-Man: Gossip-based fast overlay topology construction. *Computer Networks*, 53(13):2321–2339, 2009.

[75] M. Jelasity, A. Montresor, and O. Babaoglu. Gossip-based aggregation in large dynamic networks. *ACM Transactions on Computer Systems (TOCS)*, 23(3):252, 2005.

[76] M. Jelasity, S. Voulgaris, R. Guerraoui, A.-M. Kermarrec, and M. Van Steen. Gossip-based peer sampling. *ACM Transactions on Computer Systems (TOCS)*, 25(3):8, 2007.

[77] H. Jeong, S.P. Mason, A.L. Barabási, and Z.N. Oltvai. Lethality and centrality in protein networks. *Nature*, 411(6833):41–42, 2001.

[78] J. M. Juran. Quality control handbook. *New York*, Third Edition, 1974.

[79] R. Karp, C. Schindelhauer, S. Shenker, and B. Vocking. Randomized rumor spreading. *IEEE Symposium on Foundations of Computer Science*, pages 565–574, 2000.

[80] D. Kempe, A. Dobra, and J. Gehrke. Gossip-based computation of aggregate information. *In Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science (FOCS'03)*, 44:482–491, 2003.

[81] A.-M. Kermarrec, L. Massoulié, and A. J. Ganesh. Probabilistic reliable dissemination in large-scale systems. *IEEE Transactions on Parallel and Distributed systems*, pages 248–258, 2003.

[82] J. Kleinberg. The small-world phenomenon: an algorithm perspective. In *Proceedings of the thirty-second annual ACM symposium on Theory of computing*, page 170. ACM, 2000.

[83] K. Lerman. Social networks and social information filtering on digg. In *Proceedings of 1st International Conference on Weblogs and Social Media (ICWSM-07)*, 2007.

[84] K. Lerman and R. Ghosh. Information contagion: an empirical study of spread of news on digg and twitter social networks. In *Proceedings of 4th International Conference on Weblogs and Social Media (ICWSM)*, May 2010.

[85] J. Leskovec, L.A. Adamic, and B.A. Huberman. The dynamics of viral marketing. *ACM Transactions on the Web (TWEB)*, 1(1):5, 2007.

[86] J. Leskovec and E. Horvitz. Planetary-scale views on a large instant-messaging network. In *Proceeding of the 17th international conference on World Wide Web*, pages 915–924. ACM, 2008.

[87] C. Li and C. Chen. On Gnutella topology dynamics by studying leaf and ultra connection jointly in phase space. *Computer Networks*, 52(3)(3):695–719, 2008.

[88] J. Liang, R. Kumar, and K.W. Ross. The kazaa overlay: A measurement study. *Computer Networks Journal (Elsevier)*, 49(6), 2005.

[89] M. O. Lorenz. Methods of measuring the concentration of wealth. *Publications of the American Statistical Association*, pages 209–219, 1905.

[90] L. Lovász. Random walks on graphs: A survey. *Combinatorics, Paul Erdos is Eighty*, 2(1):1–46, 1993.

[91] P. Maymounkov and D. Mazieres. Kademlia: A peer-to-peer information system based on the xor metric. *Peer-to-Peer Systems*, pages 53–65, 2002.

[92] C. D. Meyer. *Matrix analysis and applied linear algebra*. Society for Industrial Mathematics, 2000.

[93] S. Milgram. The small world problem. *Psychology today*, 2(1):60–67, 1967.

[94] A. Mislove, H.S. Koppula, K.P. Gummadi, P. Druschel, and B. Bhattacharjee. Growth of the flickr social network. In *Proceedings of the first workshop on Online social networks*, pages 25–30. ACM, 2008.

[95] A. Mislove, M. Marcon, K.P. Gummadi, P. Druschel, and B. Bhattacharjee. Measurement and analysis of online social networks. In *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, page 42. ACM, 2007.

[96] M. Mitzenmacher. A brief history of generative models for power law and lognormal distributions. *Internet mathematics*, 1(2):226–251, 2004.

[97] A. L. Montgomery. Applying quantitative marketing techniques to the Internet. *Interfaces*, 31(2):90–108, 2001.

[98] S. Mossa, M. Barthelemy, H. Eugene Stanley, and L.A. Nunes Amaral. Truncation of power law behavior in "Scale-free" network models due to information filtering. *Physical Review Letters*, 88(13):138701, 2002.

[99] M. E. J Newman. The structure of scientific collaboration networks. *Proceedings of the National Academy of Sciences*, 98(2):404, 2001.

[100] M. E. J Newman. Assortative mixing in networks. *Physical Review Letters*, 89(20):208701, 2002.

[101] A. Orebaugh, G. Ramirez, J. Burke, and L. Pesce. *Wireshark & ethereal network protocol analyzer toolkit (jay beale's open source security)*. Syngress Publishing, 2006.

[102] K. S. Park and V. S. Pai. CoMon: a mostly-scalable monitoring system for PlanetLab. *ACM SIGOPS Operating Systems Review*, 40(1):74, 2006.

[103] A.-M. K. Pathan and R. Buyya. A taxonomy and survey of content delivery networks. *Grid Computing and Distributed Systems (GRIDS) Laboratory, University of Melbourne, Parkville, Australia*, 2006.

[104] B. Pittel. On spreading a rumor. *SIAM Journal on Applied Mathematics*, 47(1):213–223, 1987.

[105] J. Pouwelse, P. Garbacki, D. Epema, and H. Sips. The bittorrent p2p file-sharing system: Measurements and analysis. *Int'l Workshop on Peer-to-Peer Systems (IPTPS)*, pages 205–216, 2005.

[106] J. A. Pouwelse, P. Garbacki, J. Wang, A. Bakker, J. Yang, A. Iosup, D.H.J. Epema, M. Reinders, M. R. Van Steen, and H. J. Sips. Tribler: A social-based peer-to-peer system. *Concurrency and Computation*, 20(2):127, 2008.

[107] D. Purandare and R. Guha. An alliance based peering scheme for peer-to-peer live media streaming. *Proceedings of the 2007 workshop on Peer-to-peer streaming and IP-TV*, pages 340–345, 2007.

[108] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Schenker. A scalable content-addressable network. *in Processings of the ACM SIGCOMM*, page 161172, 2001.

[109] A. Rowstron and P. Druschel. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In *IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, volume 11, pages 329–350. Citeseer, 2001.

[110] S. Sen and J. Wang. Analyzing peer-to-peer traffic across large networks. In *Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurment*, pages 137–150. ACM, 2002.

[111] A. Sentinelli, G. Marfia, M. Gerla, L. Kleinrock, and S. Tewari. Will IPTV ride the peer-to-peer stream? *IEEE Communications Magazine*, 45(6):86–92, 2007.

[112] T. Silverston, A. Dainotti A. Pescap A. Ventre Fourmaux, O. Botta, and S G. Salamatian. Traffic analysis of peer-to-peer iptv communities. 53(4):470–484, March, 2009.

[113] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. *IEEE/ACM Transactions on Networking*, 11(1):17–32, 2003.

[114] D. Stutzbach and R. Rejaie. Understanding churn in peer-to-peer networks. In *Proceedings of the 6th ACM SIGCOMM conference on Internet measurement*, page 202. ACM, 2006.

[115] D. Stutzbach, R. Rejaie, and S. Sen. Characterizing unstructured overlay topologies in modern P2P file-sharing systems. *IEEE ACM TRANSACTIONS ON NETWORKING*, 16(2):267, 2008.

[116] G. Szabó and B.A. Huberman. Predicting the popularity of online content. *Communications of The ACM*, vol. 53, no. 8:pp.80–88, August 2010.

[117] R. Van Der Hofstad, G. Hooghiemstra, and P. Van Mieghem. First-passage percolation on the random graph. *Probability in the Engineering and Informational Sciences*, 15(02):225–237, 2001.

[118] P. Van Mieghem. *Data communications networking.* Techne Press Amsterdam, 2006.

[119] P. Van Mieghem. *Performance analysis of communications networks and systems.* Cambridge University Press, 2006.

[120] P. Van Mieghem, J. Omic, and R. Kooij. Virus spread in networks. *IEEE/ACM Transaction on Networking*, 17(1):1–14, February, 2009.

[121] P. Van Mieghem and S. Tang. Weight of the shortest path to the first encountered peer in a peer group of size m. *Probability in the Engineering and Informational Sciences*, 22(01):37–52, 2007.

[122] P. Van Mieghem, H. Wang, X. Ge, S. Tang, and F. Kuipers. Influence of Assortativity and Degree-preserving Rewiring on the Spectra of Networks. *The European Physical Journal B, to appear.*

[123] R. Van Renesse, Y. Minsky, and M. Hayden. A gossip-style failure detection service. *in Proceedings of Middleware'98, IFIP International Conference on Distributed Systems Platforms and Open Distributed Processing*, pages 55–70, 1998.

[124] L. Vu, I. Gupta, J. Liang, and K. Nahrstedt. Measurement and modeling of a large-scale overlay for multimedia streaming. In *The Fourth International Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness & Workshops (QShine 2007)*, page 3. ACM, 2007.

[125] S. Wasserman and K. Faust. *Social network analysis: Methods and applications.* Cambridge University Press, 1994.

[126] D. J. Watts and S. H. Strogatz. Collective dynamics of small-world networks. *Nature*, 393(6684):440–442, 1998.

[127] D.M. Wilkinson and B.A. Huberman. Cooperation and quality in wikipedia. In *Proceedings of the 2007 international symposium on Wikis*, page 164. ACM, 2007.

[128] B. A. Williamson. EMarketer Social Network Marketing: Ad Spending and Usage. *http://www.emarketer.com/Report.aspx?code=emarketer_2000478*, 2007.

[129] C. Wu, B. Li, and S. Zhao. Exploring large-scale peer-to-peer live streaming topologies. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMCCAP)*, 4(3):1–23, 2008.

[130] X. Zhang, J. Liu, B. Li, and T.S.P. Yum. CoolStreaming/DONet: A data-driven overlay network for efficient live media streaming. *proceedings of IEEE Infocom*, 3:13–17, March 2005.

[131] B .Y. Zhao, L. Huang, J. Stribling, S. C. Rhea, A.D. Joseph, and J. D. Kubiatowicz. Tapestry: A resilient global-scale overlay for service deployment. *IEEE Journal on selected areas in communications*, 22(1):41–53, 2004.

[132] M. Zink, K. Suh, Y. Gu, and J. Kurose. Watch global, cache local: YouTube network traffic at a campus network-measurements and implications. *Proceeding of the 15th SPIE/ACM Multimedia Computing and Networking (MMCN08)*, 2008.

Samenvatting (Summary in Dutch)

Thesis Titel: informatie verspreiding in Peer-to-Peer netwerken: modellering en empirische studies

Ook al is P2P netwerken een jonge technologie, het heeft in de afgelopen twintig jaar al een dramatische evolutie op het internet teweeg gebracht. In tegenstelling tot de traditionele server-client modus staat in P2P netwerken de gebruiker centraal. Gebruikers (peers) genereren hun eigen informatie en delen het met anderen via het internet. Of het nu een P2P file-sharing netwerk, een streaming systeem, een video-on-demand applicatie of een online sociaal netwerk betreft, alle genoemde toepassingen hebben als fundamenteel doel het leveren van informatie aan medegebruikers op een gedecentraliseerde manier. In dit proefschrift bestuderen we de informatieverspreiding op basis van de volgende twee aspecten:

1. Middels gebruik van bestaande technieken, modellen voor te stellen die van toepassing zijn op P2P netwerken.
2. Door empirische studies met nieuwe P2P toepassingen, hun gedrag met betrekking tot informatieverspreiding achterhalen.

We beginnen met een studie van informatieverspreiding op basis van roddels ('gossiping' genoemd) in gedecentraliseerde P2P netwerken. We illustreren de moeilijkheid van het uitvoeren van een nauwkeurige analyse van gossiping op grote schaal en in dynamische P2P netwerken, waarbij elke peer alleen communiceert met een subset van de peers in het netwerk. We tonen aan dat een beschrijving van gossiping in de bovengenoemde netwerken een te grote rekenruimte vereist. Om de betrouwbaarheid van gossiping te garanderen, ontwikkelen we een exacte analytische modellering van gossiping algoritmen voor de verspreiding van informatie onder bepaalde veronderstellingen. Het model wordt uitgebreid zodat willekeurige communicatie met meerdere collega's is toegestaan. We incorporeren verschillende netwerkkarakteristieken en peer gedrag in het model. Belangrijke prestatie en ontwerp parameters worden ook analytisch bepaald. Het voorgestelde model is toepasbaar informatieverspreiding en het zoeken van informatie in gedecentraliseerde P2P netwerken. De afgeleide parameters kunnen worden gebruikt om de omvang en doeltreffendheid van de informatieverspreiding en informatie zoeken te evalueren. We bestuderen ook het vergaren van informatie, gegeven dat m peers de gewenste informatie bezitten. Het effect van het selecteren van een peer die het dichtst bij is op basis van hops of vertraging, in de groep van m peers wordt geanalyseerd. Onze analyse geeft antwoord op de vraag hoeveel kopieën van bepaalde informatie moeten worden verdeeld (of hoeveel peers die beschikken over de gewenste informatie moeten worden ontdekt), zodat een aanvaardbare kwaliteit van de dienstverlening (in termen van hops en vertraging) kan worden geleverd.

Het gossiping model geeft een eerste idee over informatieverspreiding in P2P netwerken. Echter, vanwege de snelle evolutie van P2P netwerken, zijn er inmiddels toepassingen gelanceerd met nieuwe functies waar gebruikerskenmerken een belangrijke rol spelen. Daarom voeren we twee empirische studies uit die zijn ontworpen om belangrijke ontwerpkwesties en gebruikersgedrag te onderzoeken in enkele opkomende P2P toepassingen.

Onze eerste empirische studie richt zich op een eigen Peer-to-Peer Televisie (P2PTV) systeem met de naam SopCast. P2PTV toepassingen zijn dominant aanwezig als het gaat om het leveren van videos/TV via het internet, terwijl hun onderliggende mechanismen nog grotendeels onbekend zijn. Daarom voeren we een reeks experimenten uit om de prestaties van de SopCast netwerk te weerspiegelen. We ontleden een deel van het SopCast protocol door middel van 'reverse engineering'. Onze analyse verklaart de communicatie, de video verspreidingsmethode en de structuur van het netwerk. De dynamiek van het SopCast netwerk, en haar impact op het verkeer in het internet worden ook geëvalueerd. De aanpak en de methodologie in dit empirisch onderzoek geven inzicht in bestuderen van gelijkaardige toepassingen.

Zoals eerder vermeld, het belang van de gebruikers in P2P netwerken is groter dan bij andere netwerktoepassingen. Daarom wordt de tweede empirische studie uitgevoerd met een online sociaal netwerk, genaamd Digg. In nieuwe online sociaal netwerktoepassingen kunnen gebruikers gezamenlijk informatie publiceren, ontdekken, en bevorderen zonder tussenkomst van websiteredacteurs. Dagelijks wordt er een grote hoeveelheid informatie gepubliceerd op deze sites, terwijl slechts een paar stukjes van die informatie populair wordt. In deze empirische analyse willen wij de volgende vragen beantwoorden: 1. Of gebruikers van online sociale netwerken vrienden maken met anderen die vergelijkbare interesses hebben? 2. Volgens welk dynamisch proces filteren en dragen de gebruikers samen informatie uit in de online sociale netwerken? 3. Of vriendschaprelaties helpen om nieuw gepubliceerde informatie te verspreiden? Inzicht in deze verschillende kenmerken en de informatie verspreiding in online sociale netwerken helpt bij het verbeteren van huidige marketingtechnieken die pogen om advertenties, producten, en ideeën over deze netwerken aan de man willen brengen.

Acknowledgements

This thesis presents the works that I have performed during the course when I was pursuing my Ph.D. degree. The past four-year period is undoubtedly one of the most memorable time in my life. When writing this chapter, all the past memories flash into my mind: the first day that I went to work as a Ph.D. student; the discussions with my colleagues; the joyful day-out and drinks that we have had... It is indeed my pleasure to work with many people during the four years. Without them, this thesis would not be possible.

First of all, I would like to thank my promoter Prof. Piet Van Mieghem for his guidance, patience, support, comments and inspirations to my work. I appreciate that he is always interests in my research, and has always been available to advise me, even when sometimes I knocked his door without making an appointment. I have benefited greatly from his enthusiasm, immense knowledge, and rigorous attitude towards science. For me, he is not only my supervisor in academia, but also my mentor, mentally. I feel lucky to become his Ph.D. student. Thank you, my *shifu* (the Chinese word of mentor), for providing me the opportunity to work in the Network Architectures and Services Group, leading me to the academic field, and guiding me in these four years.

I want to express my gratitude to the members of my Ph.D. defense committee for their time and work spent on this thesis as well as their insightful feedbacks. It is my honor to have them involved in this thesis. In particular, I am indebted to Prof. Stavrakakis, who offered me great help during my stay in the National and Kapodistrian University of Athens. His instructions and suggestions made our cooperation more effective. I am especially grateful for his time and efforts spent on our joint work, and the nice conversations that we have had during the CONTENT project. I am also grateful to Prof. van Steen for his encouraging words and feedbacks on this thesis; to Dr. Epema for his suggestions on the structure and presentation of this book; to Dr. van Gageldonk for his experiences and knowledge about peer-to-peer and online social networks not only from academic research, but from the user perspective, and the commercial point of view.

I also want to thank Dr. Fernando Kuipers and Edgar van Boven. I enjoyed working with Fernando because of his confidence, wisdom, politeness, and the steady tempo while speaking. Thanks to the many discussions and iterations that we have had when writing papers, proposals, and deliverables, he helped me to continuously improve my capability of performing rigorous research and writing precise technical reports. Moreover, I would like to thank him for translating the summary of my thesis in a really efficient way. Thank you Edgar, for helping me a lot when calling to the tax office, and for your time spent on translating the propositions along with this thesis. I really appreciate your serious attitude in providing me the precise translations and your kindness in offering me your timely help.

Life would not be so inspiring and cheerful without my dear friends and colleagues on the 19th floor. My special thank goes to my office mates, Yue, Xavier, and Anteneh, for the pleasing atmosphere in our office, for the nice conversations and jokes, and the caring words when someone of us was sick. I feel fortune to have dear Jasmina, Huijuan, Xueli, Jing, Yue, and Yanying, with whom I could share my ideas, happiness and confusions from the feminine point of view. My thanks go to António for reminding us to have lunch everyday, to Norbert for the many discussions during our collaboration and providing me comments to my thesis, to Wynand for polishing the propositions in my thesis, to Rob, Tom, Stojan, Dajie for their comments and supports when I was practicing my KIVI presentation. I also want to thank Laura, Marjon, Wendy, Stefanie and Mrs. Anna Hoek for their constant help with the administrative matters.

Last but not least, I would like to thank my parents, for their unconditional support and endless love. They gave me too much - much more than what I can give to them. My gratitude to them is beyond words. I also want to thank my husband, who is also my best friend - the one that I can share all my emotions, be it happiness or sorrow. Thank you, Yi, for cherishing my life as much as yours.

Siyu Tang

Curriculum Vitae

Siyu Tang was born in Xining, China, on September 21, 1980. She graduated with her B.Sc. degree in Electrical Engineering from Beijing Jiaotong University, China, in 2003. In August 2006, she received her M.Sc. degree in Electrical Engineering from Delft University of Technology, the Netherlands. After finishing her study, she joined the Network Architecture and Services (NAS) group, headed by Prof. Piet Van Mieghem, towards a Ph.D. degree on the project of "European Network of Excellence on content distribution". During her Ph.D. study, she focused on content dissemination in peer-to-peer (P2P) networking. Her present research interest is the field of online social networking applications that are at large-scale. In particular, she is interested in studying the interaction and communication patterns between individuals in online social networking sites (OSNs) and the impact of user behaviors on information propagation in online social communities. She has won the second place of the KIVI NIRIA Telecommunicatieprijs 2010. The KIVI NIRIA Telecommunicatieprijs contest is held annually between the three technical universities in the Netherlands.

Publications

- P. Van Mieghem, and S.Tang, "Weight of the Shortest Path to the First Encountered Peer in a Peer Group of Size m", Probability in the Engineering and Informational Sciences (PEIS), vol. 22, pp. 37-52, 2008.

- S. Tang, H. Wang and P. Van Mieghem, 2008, "The Effect of Peer Selection with Hopcount or Delay Constraint on Peer-to-Peer Networking", IFIP Networking 2008, LNCS 4982, pp 358-365, 2008.

- S. Tang, Y. Lu, J. Marten Hernandez, F.A. Kuipers, and P. Van Mieghem, "Topology dynamics in a P2PTV network", Networking 2009, LNCS 5550, pp. 326-337, 2009.

- P. Van Mieghem, H. Wang, X. Ge, S. Tang, and F.A. Kuipers, "Influence of assortativity and degree-preserving rewiring on the spectra of networks", The European Physical Journal B, to appear.

- S. Tang, E. Jaho, I. Stavrakakis, I. Koukoutsidis and P. Van Mieghem, “Modeling Gossip-based Content Propagation and Search in Distributed Networking”, submitted to Computer Communications, under review.

- S. Tang, C. Doerr and P. Van Mieghem, “A measurement study of the Digg online social network”, Delft University of Technology, report20100611.