Facebook User Behavior & Community Clustering Analysis Based on Content Demand Patterns

Jolina Haberkamm & Niclas Unnervik
{ic08jh2, dt08nu0}@student.lth.se

Department of Electrical and Information Technology
Lund University

Advisor: Maria Kihl

February 24, 2013

Printed in Sweden
E-huset, Lund, 2013

Abstract

In order to increase Quality of Service and Quality of Experience in mobile and residential broadband access networks it is important to understand user patterns. This understanding could make the implementation of cost effective performance optimizing functionality possible rather than expanding the networks physical layer.

This thesis aims to develop and evaluate methods for determining Facebook user behavior based on Facebook content demand patterns. The problem is approached by analyzing data from two large municipal network dumps performed on different geographic locations, during different time spans.

This study shows that community clustering using content demand patterns is possible and that users with similar user behavior can be grouped together in user clusters. This is supported by a highly configurable Trust-function that build a graph $G = (V, E)$ where E denotes similarity between the connected users and the clustering algorithm *Chinese Whispers*. It was also discovered that different user activities and certain terminal usage patterns are strongly correlated, showing that user patterns differs between devices. Geographical location is discarded as a parameter for potential future methods since all results from the two networks are highly aligned and no deviation in user behavior is detected due to location.

Acknowledgement

We would like to thank:
-our advisor, Maria Kihl at LTH, for being positive and incredibly helpful in encouraging us to do our best
-our partner in crime, Robin Larsson, for all the tips and advice and the YouTube videos
-Andreas Aurelius at Acreo, for always being quick to respond to all our strange requests and demands,
-Manxing Du at Acreo, for always explaining how things actually work,
-Christina Lagerstedt at Acreo, for giving us free hands on the servers and still keeping them alive,
-Jie Li at Acreo, for being patient and trying to help us,
-Robert Johnsson at LTH, for helping us with the broken network drivers,
-Åke Arvidsson at Ericsson, for the great feedback.
Thank you.

Populärvetenskaplig sammanfattning

Facebook är världens största sociala nätverk med mer än en miljard användare globalt, över 50% av alla svenskar användar sajten aktivt. I vår del av världen genereras ofattbara 7% av all bredbandstrafik av Facebook.

Vi har studerat ett stickprov av den svenska trafiken och genom det lyckats härleda vilka användare som med stor sannolikhet känner varandra. Vi har byggt upp två stora nätverk, ett från en stad i söder och ett från en stad i norr, där människors relationer fångas, baserat enbart på deras surfvanor. Som exempel kan vi titta på de fyra användarna Alice, Bob, Carl och Dan, samt Företag AB. Dan lägger ut tre intressanta bilder som Alice och Bob tittar på , Carl däremot tittar inte på bilderna. Företag AB lägger också ut en bild, som Alice, Bob och Carl tittar på, men denna bild ses också av tusentals andra användare, till skillnad från Carls bilder. Vår algoritm analyserar nu denna information och kommer fram till att eftersom Alice och Bob har tittat på flera av Carls bilder känner de troligtvis varandra. Carl däremot, han har ju också tittat på en bild tillsammans med Alice och Bob, känner de honom också? Eftersom Företag ABs bild setts av många tusen människor så säger denna gemensamma bild inte tillräckligt för att Alice och Bob ska antas känna Carl. Detta upprepas för alla användare och för alla bilder de tittat på i de båda städerna tills vi har en bild av hur alla användare tros känna varandra.

Detta nätverk delas sedan in i grupper, där användare som känner varandra bra hamnar i samma grupp. även om vissa användare inte kan placeras i en sådan grupp kan det flesta användare grupperas. Grupperna används sedan för att studera om olika grupper använder sig av Facebook på olika sätt, vilket kan sägas att dem gör. T.ex. är det så att grupper där användarna surfar på Facebook på sina telefoner sällan använder en dator för att surfa på Facebook (och tvärtom), vilket vi tycker är ett överaskande och intressant resultat.

Vi har också undersökt ifall användare har olika beteenden på Facebook beroende på beroende på vilken typ av terminal som används; en dator, telefon eller surfplatta och kan konstatera att så är det. T.ex. är det vanligare att dela ut en ”like” på en telefon än en surfplatta med vanligare att titta på en video på en surplatta än på en telefon.

Dessa resultat skulle kunna användas av telekomindustrin för att spara pengar när bredbands- och mobilnätverk byggs, kännedom kring hur näten används kan bidra till smartare nät; billigare, snabbare och mindre resurs krävande.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

The Internet usage changes continuously as the Internet becomes available to everyone and spreads out. Internet today hosts multiple services; e.g. the massive World Wide Web, email, FTP (File Transfer Protocol) and IM (Instant Messaging). It has become quite different compared to its early days, when it was a tool used only by researchers and scientists. With billions of connected devices and actors with vastly different agendas, the Internet is a diverse and heterogenous network based on complex technologies. The demands on broadband access networks, regarding bandwidth as well as Quality-of-Service (QoS) have increased and continue to do so as the Internet keeps on evolving and rapidly expand. Modern Internet usage must provide high capacity for unpaid traffic but simultaneously fulfill the user need of perfect QoS for multimedia services as well as Quality-of-Experience (QoE). One reason is that Internet usage has emerged from traditional WWW usage, where web pages are downloaded, into triple-play usage, meaning that all communication services of a household are using the same broadband access connection[22][25].

According to a [26], global IP traffic in 2011 stood at 30.7 exabytes per month and is expected to increase to a threefold by 2016, to reach 110.3 exabytes per month. Consumer IP traffic will reach 97.2 exabytes per month in 2016 and business IP traffic will surpass 13.1 exabytes per month. Consumers includes households, university populations and Internet cafés, business implicates fixed IP Wireless Access Networks (WANs) or Internet traffic generated by businesses and governments[26].

Between 2010 and 2011 consumers were responsible for the majority of all IP traffic in every segment and this is not likely to decrease in the future. Studies show that consumers were responsible for 91% of all Internet traffic; all traffic crossing an Internet backbone, 77% of the total mobile data traffic; traffic generated by handsets, notebook cards, and mobile broadband gateways, and 80% of the managed IP traffic; corporate IP WAN traffic and IP transport of TV and VoIP[26].

Consumer Internet usage is therefore of great importance when it comes to face the challenges of building better network architectures. Part of the solution to this problem is to get an understanding for the Internet traffic patterns of residential users, volumes and applications as well as user activity characteristics, i.e. session lengths and traffic rate distribution. User behavior needs to be monitored and measured close to the users, in the actual broadband access network to actualize up to date user behavior models[25].

To keep Internet users satisfied and to provide adequate quality of upcoming advanced Internet applications the network architectures need to be improved or new ones need to be created. A popular content-based architecture, which serves a great fraction of all Internet content to the end-user, with both high availability as well as performance, is Content Delivery Networks (CDNs). These networks distribute large amount of data to the end-users very efficiently since e.g. cached copies of the data is made close to the end-user. Network resources can be saved by caching content since the demand for retransmissions will be lower, the performance will improve by decreasing the risks for delays and loss and the content will have a higher availability, being reachable from several destinations. In order for cacheing to be beneficial, the content must be carefully chosen. This requires thorough studies of Internet usage to map out user groups and common behavior[23].

Popular content comes from a diversity of sources, some are small or medium sized providers but there are a few key players; the so called hyper giants, generating almost a third of all Internet traffic. Among the hyper giants are Facebook, Google, YouTube and Microsoft[24].

In early March of 2010, Facebook topped Google for a week in a row to be the most visited website in the US for the first time. During that week Facebook accounted for 7.07% of all U.S. Web traffic compared to Google with 7.03%, showing that content sharing once again had become the number one online phenomenon. Google had topped the US web traffic every week up until this since 2007 when it had passed the social network MySpace[6]. In May and June 2012, when Facebook reached over 1 billion subscribers with 552 million daily active users, it was responsible for 9% of all US traffic. The popularity of Facebook is similar worldwide with certain exceptions, especially in Asia. In China Facebook is blocked and can only be accessed with certain work-arounds, consequently Facebook is not responsible for a significant percentage of Chinese Internet traffic. In China and other countries where Facebook is not the most popular social network, other social networks used in the same way generate huge amounts of traffic.

This project focuses on trying to find user patterns among Facebook users through a social network analysis. Focus is to understand relationships between users in order to understand how content is shared, perceived and spread out through the social web. It is interesting to look at what type of content that is popular, i.e. what activities that is mostly performed. One goal is to create user models based on these results to find content that could be beneficial to cache in the future and to map out how users are connected. This thesis is part of the IP Network Monitoring for Quality of Service Intelligent Support (IPNQSIS), which strives to build a Customer Experience Management System (CEMS) based on QoE where the customer perception as well as the network performance is the main focus. The aim with this thesis is to try to identify Facebook user behavior and build community clusters based on content demand patterns analysis, as well as to understand how user patterns varies on different devices. Content demand patterns will be derived from a network data dump containing only Facebook traffic.

Chapter 2

Background

This chapter presents a brief history of how the Internet has emerged from its early stage into the global gathering place it is today and how online communities have become the new way of socializing. It also includes an introduction to the today's leading social network, Facebook.

2.1 Internet

The history of the Internet goes back to the early 1960's when the US Department of Defense Advanced Research Projects Agency (ARPA) created ARPANET, the predecessor to the Internet we know today, allowing communication between ARPA computer terminals. ARPANET was a fully operational packet-switched network with its first stable link fully working in 1969. It was designed to fulfill the desire to allow users to access functions of computers and data without physical presence as well as to create a communication structure for the U.S. Military in case of a nuclear attack. The main purpose of fundings by ARPA was for computer development and research, during this early computing age, computers were incredibly expensive to produce and operate and too expensive for common usage. It took approximately ten years before ARPANET transformed into the Internet we are familiar with[17].

ARPANET was the first network to send data efficiently due to the introduction of packet switching. With packet switching the data was divided into shorter packets allowing the message to arrive in smaller pieces which can be sorted quickly without the requirement to wait for the entire message to arrive. This made it easier and cheaper to use telephone lines to send data[21].

After the first node of the ARPANET at the University of California (UCLA), Los Angeles, three more nodes were built and by December 1969 four computers at four research centers; UCLA, the Stanford Research Institute; the University of California, Santa Barbara; and the University of Utah in Salt Lake City, were linked together. More nodes were created and by 1972 ARPANET had 37 nodes, which slowly started to connect different networks to each other, moderately creating the great World Wide Web. The interest for the Internet and to further develop network technology became even larger with the possibility to send and receive emails in 1972[17][21].

In the beginning, the only Internet users were a few computer scientist, who shortly after the networks were connected, came to decide the shape and form of the Internet as we know it today. These researchers created the Request For Comments (RFC) in 1969 to set rules for how data exchange should work. 40 years later these first RFC's are still in use and there exist more than 5,000 RFC's in total. The RFCs were constituted in an open process, free of charge and open for everyone to contribute and they still remain this way. Even with lack of patents, restrictions or financial incentive the RFC's became the formal method of publishing Internet protocol standards and it enabled the Internet to grow in to the existing Web[18][19][21]. In 1973 the networking protocols, the Transmission Control Protocol/Internet Protocol (TCP/IP) suite allowed multiple networks to be joined together on a more open form, so that a network could stand alone even if a connected network was brought down. Today the TCP/IP suite is still in use and it is the universal host protocol on which the Internet relies upon[19].

The technology behind the Internet remains but as the Internet became available to everyone and features like the massive World Wide Web and email emerged, the Internet usage started changing and developed into something quite different compared to when it was only used by researchers and scientists. Today Internet is used for everything from shopping to education and Internet availability has become important enough to be considered of by many as a human right. As the popularity and the number of Internet users grow the demands on broadband access networks, regarding bandwidth as well as QoS, increase and continues to do so.

2.2 Online Communities

The Internet has become the "Third Place" of the modern world. Third places are places where everyone is welcome as they are; it is neutral ground and available to everyone. Third places used to be cafés, neighborhood bars, parks and other hang out places where people gathered to share their experiences and talk about their day. Today, a lot of people do not have time to go to physical third places, which is why online communities have, in many ways, taken over the old fashion ones[41].

An online community can be described as a social network where members interact in a virtual environment and just like any off-line social network its existence relies on the interest from its members to keep it alive. The main activity of a third place is to share experiences of any sort and to reflect upon these, which is the main purpose of many online communities as well. The activity is the attraction, the meeting spot is simple yet welcoming and participants are not pressured to participate beyond their will. There is no need for a host or any social rankings, the feeling of being together, part of a network, is prominent but not in focus.

Social networks are usually designed to have a purpose; a predetermined audience, activity, visual design and a backstory. It is a gathering place that should be easily navigated and welcoming. To keep members satisfied and intrigued, profiles can usually evolve over time, memberships can be upgraded and users can take on different roles, join subgroups as well as participate in different events, very similar to the real world[42].

2.3 Facebook

The social network Facebook was founded by Mark Zuckerberg, who was a 23 year old Harvard psychology student at the time. The site was launched on February 4, 2004. The first edition of Facebook was originally called *The Facebook* (www.thefacebook.com) and the name was changed to *Facebook* in August 2005 when the domain facebook.com was bought for $200,000[2][4].

The rapid growth of Facebook started the moment it became available. Within 24 hours after it was launched, 1,200 Harvard students had created accounts and a month later 50% of all Harvard undergrad students had signed up. The network expanded from Harvard via other Ivy League universities to all US universities and by September 2005 U.S. high school students could create accounts as well. In the end of 2005 university students all over the world could sign up for Facebook and a year later, in September 2006 it opened up for everyone with an email address to sign up. Within a year after its public availability was announced, in 2007, Facebook had more than 30 million users.[2] On October 4, 2012 Facebook had one billion active users from which 81% were users outside the U.S. and Canada. In September 2012, 600 million monthly active users were seen using Facebook mobile products[3][4].

Facebook started off as a one-man project but in the end of June 2012 the company had 3,976 employees. The headquarter resides in Menlo Park, California but there are offices all over the world, from Auckland to Sao Paolo and Tokyo to Stockholm. There are 13 US offices in total and an additional 18 international ones[3].

Under the *About* tab on Facebook's own Facebook page, visitors can read that "Facebook's mission is to give people the power to share and make the world more open and connected" [1]. According to Facebook its millions of users use the network in order to keep up with friends, upload an unlimited number of photos, share links and videos, and learn more about the people they meet[1].

Facebook has been a free service from the beginning and it seems as if it intends to stay this way even as it continues to grow and develop. The annual cost of one billion dollars it requires to run Facebook is all paid by ads and sponsored links. This concept seems to work since Facebook enables advertisers to reach more than 900 million people with customized ads that provide social interest[3]. Advertisements are adjusted to show relevant and interesting information for users based on information from a user's personal information, posts, likes, groups etc.[5].

2.3.1 Under the Hood

To render a single page on Facebook a domino process of data examinations executed by hundreds of machines is triggered. Tens of thousands of pieces of data from dozens of services must be examined in real-time. Facebook is globally interconnected and operates on a huge scale, in order to meet user requirements the Facebook infrastructure team had to rethink every layer of the technology stack[3].

Facebook is one of the main users of *memacached*[1] and owns one of the largest MySQL database clusters in the world storing over a 100 petabytes (100 quadrillion

[1] An open source distributed memory object caching system[11].

bytes) of photos and videos. To efficiently store and handle this enormous amount of data specific tools and technologies have been created. E.g. *Haystack*[2] is a storage and serving technology that has been developed as a result of this prominent issue. Another result of this is *Hip Hop*[3] for PHP which provides higher performance gains compared to traditional PHP.

The Facebook Data Warehouse infrastructure, *Apache Hive*[4] built on top of *Hadoop*[5] offers tools to allow easy data summarization, ad hoc querying and analysis of large data sets[3][9].

Facebook with its specific requirements for massive scale computing and rapid growth has built their own software, servers and data centers resulting in reduced costs and increased efficiency compared to traditional ones. The server and data center designs are open sourced in the so called *Open Compute Project*[6][3].

Today Facebook has two data centers where the many Facebook servers are stored. The first custom data center was opened in 2010 in Prineville, Oregon, U.S.. Today it has approximately 60 full-time employees working to repair and maintain servers, generators and backup power supplies, and provide building maintenance, security and other critical infrastructure at the facility. The second data center is located in Forest City, North Carolina, U.S. and opened recently. A third data center, the first outside the U.S. is now being built in Luleå, Sweden[14][15][16].

2.3.2 The Facebook Sprawl

Facebook is constantly expanding and new products and functions are launched to keep the network growing. The phrase "This journey is 1% finished" is posted on the walls of the Facebook offices to encourage employees to be bold, innovative and creative. According to Facebook this is to remind their employees to fulfill

[2] An object storage system designed for Facebook's Photos application. It avoids disk operations when accessing metadata and provides a fault-tolerant, cost-effective way with high throughput when serving the large number of requests surfaced in a large scale social network.[12]

[3] A way to transform PHP source code into highly optimized C++ code which is compiled by g++. It reduces the CPU usage of the Web servers increasing the overhead which improves the performance. *Hip Hop* includes a code transformer, a reimplementation of PHP's runtime system, and a rewrite of many common PHP Extensions in order to take advantage of these performance optimizations.[7]

[4] A data warehouse infrastructure built on top of Hadoop that provides tools to enable ad hoc querying, easy data summarization, and analysis of large datasets data stored in Hadoop files. It provides a mechanism to put structure on this data as well as a simple query language based on SQL named QL.[8]

[5] Software for reliable, scalable, distributed computing which provides a framework for large scale parallel processing using a distributed file system and the map-reduce programming paradigm. *Hadoop* library is designed to detect and handle failures at the application layer instead of relying on hardware in order to deliver high-availability services.[10]

[6] A project dedicated to increase the pace of innovation in data center technology aiming to make highly efficient scale computing technology available to everyone along with reducing the environmental impact of computing infrastructure

the Facebook mission; to make the world more open and connected[3].

The main Facebook products, some which have been around since the start while others are brand new, are the News Feed, the Timeline, Messages, Photos and Videos, Groups, Events and Pages. These products are accountable for a great part of the Facebook traffic. More than 300 million photos are uploaded to Facebook each day and over 16 million events are created each month. In addition to these products and the applications within the Facebook API there are third-party applications as well. The popularity of Pages and applications shows how Facebook grows in new ways beyond expanding its user network. There were more than 42 million Facebook pages and 9 million apps and websites integrated with Facebook in April 2012. According to Facebook over 4 million businesses had pages on the site at that time. Other popular pages belong to public figures, movies, sports teams and other fan-generated community pages. During the first quarter of 2012, users generated an average of 3.2 billion Likes and Comments each day and there were more than 125 billion existing friend connections between the 900 million monthly active Facebook users[3][13].

In Sweden there were 4,885,400 number of Facebook users in October 2012, ranking it the number 37 country in the world. This shows a 53.84% penetration of the population compared to the number of inhabitants. 228,320 new users were registered in Sweden between April and October 2012. 51% of all Swedish users were female and 49% male. The age group 25-34 was the largest with 1,123,642 users before the age group 18-24. The Facebook Top 5 brands in Sweden at this time were in ascending order *Free Lunch Design* (604,957 fans), *Marabou* (395,387 fans), *Hallonlakritsskalle* (359,920 fans), *Gina Tricot* (339,685 fans) and *Nelly.com* (329,377 fans)[20].

Chapter 3

Previous Work

In this chapter previous work about Network Analysis in the Traffic Measurements and Models in Multi-Service Networks (TRAMMS) and IPNQSIS project is included as well as the previous work on Facebook cacheability and Social Network Analysis (SNA).

3.1 Network Analysis

TRAMMS was a three year project with the main objective to model traffic in multi-service IP networks and to use models as input for capacity planning of future networks. TRAMMS was part of the Celtic framework which is a EUREKA cluster focusing on telecommunication. It was a collaboration between eleven Swedish, Hungarian and Spanish partners and traffic measurements were performed in broadband access networks in different parts of Europe. The models derived in this project were created based on bottleneck analysis as well as inter-domain routing analysis in combination with data from measurements on the application level with deep packet and deep flow inspections. Detailed information about the traffic such as the type of access technology the traffic originates from, the access speed and the number of households generating the traffic. Traffic patterns established by user and/or application behavior were used to identify services, which were heavy consumers of the available network resources[24].

A result from this project showed that P2P (Peer-To-Peer) applications were being used as a common content delivery mechanisms for both legal as well as illegal content. Even if the P2P traffic has decreased due to regulations such as the Intellectual Property Rights Enforcement Directive (IPRED)[1] the number of legal P2P applications are increasing causing traffic volume generated by P2P applications to increase accordingly. This will lead to a higher demand for symmetrical access connections which also might benefit other services such as e.g. video conferencing. This result showed that federal laws, policy decisions and regulations, like IPRED can have a great impact on traffic patterns and user behavior. File sharing decreased severely after IPRED was enforced, which is a clear indication that researchers, network designers and policy makers have to collaborate

[1] IPRED was adopted by the Council and European Parliament in 2004 to combat piracy and other infringements of "intellectual property rights" (IP-rights), such as patents, copyright and trade marks[43].

while analyzing the outcome of possible future decisions concerning regulations and such.[24]

Another project of the Celtic framework concerning network analysis is the IPNQSIS project. The focal point of IPNQSIS is to build a Customer Experience Management System (CEMS) based on QoE where the customer perception as well as the network performance are being considered. The supposed outcome of this project is a CEM architecture with the requirements, design and an implementation of a CEMS composed of three different layers: Data Sources (i.e. probes), Monitoring Component and Control Module as well as measurement devices that can provide feedback to the control system. Probes among other multi-technology network devices will be used to input the QoE. Through monitoring and analyzing IP traffic in access networks with deep packet inspection and deep flow inspection techniques, new techniques for distribution of multimedia content for cost-efficient solutions in order to maintain acceptable levels of QoE can be proposed. To combine QoE-QoS correlation analysis with network operation and traffic modeling studies; cognitive software will be developed and tested[27].

3.2 Social Network Analysis

Social Network Analysis is the social science which involve the mapping and measuring of relationships and flows in face-to-face groups and mathematical graph theory. Network properties can be described with statistical tools to show the distributions of actors, attributes and relations as well as joint distributions, predictions and hypotheses. Results found through analysis must be examined in order to be determined to represent an actual pattern or a random coincidence only valid for certain users. In SNA nodes are used to represent actors and edges describe relations. The focus in SNA is not attributes or individual nodes but relations; edges between nodes. Two nodes in a network might seem independent if the lack a relation between them but might still be dependent since they could share another relation to a third node[39].

3.3 Facebook Data and Cacheability

In [48] a study of Facebook user behavior and Facebook data traffic was carried out in a similar way to this thesis. The focus was set on Facebook pictures and the Like tool in order to find popular users whose content was highly demanded.

This study showed that 86.48% of the total downloaded pictures were thumbnail profile pictures, which were downloaded automatically every time a user session was initiated but also each time the user's own timeline was reloaded. One conclusion in [48] was that these thumbnail profile pictures presented content that would be beneficial to cache due to their frequent download requests. It was not possible to determine how long of a time span a picture was popular, i.e for how long it would be beneficial to cache pictures.

Another result showed that popular up-loaders had a large number of unique downloaders but at the same time only a small percentage of all Facebook users were responsible for a large portion of all downloads as well as all Likes.

Chapter 4

Methodology

This chapter gives a brief introduction of the analyzed networks. The tools used during the processes of data collection, filtering and analyzing the data are described as well as the limitations present for this thesis.

4.1 Data Collection

The analyzed data was collected from two different Swedish municipal networks which are referred to as Network North and Network South, according to their geographical location. Both networks are fiber based IP access networks utilized by local residents. Network North constitutes approximately 5300 households and Network South roughly 2000. The access speed varies from 1 megabit per second(Mbps) to 100 Mbps depending on the customer's choice of (Internet Service Provider) ISP subscription.

Network North is a layer two network, ergo the source and destination MAC addresses remain unchanged as the data packets traverse the network. The source MAC address is either the the device which sent the packet or a household router.

In Network South, the MAC address changes hop by hop since it is a layer three network, hence all MAC addresses in the packet dump from Network South show the very last router. Therefore the MAC addresses had to be replaced according to the Dynamic Host Configuration Protocol (DHCP) address assignment from the ISPs DHCP log together in combination with the identifier of the address switch connected to the household.

To guarantee that none of the sensitive information about users are revealed, all IP addresses as well as MAC addresses have been hashed. To be extra cautious not to leak any private information sensitive data has never left Acreo's servers and alls scripts have been run on those using a VPN. The only information that have been downloaded from the servers are outputs with results not including any private or sensitive information to uphold integrity and confidentiality .

The data was collected with the commercial traffic management device Packet Logic (PL) used as the traffic data collection tool. Network North used a PL8720 and Network South a PL 7720, both running on version 13.X.

Traffic was identified with deep packet inspection and deep flow inspection and not with help of port definitions. All traffic was measured on the application layer. Avoiding port-based identification eliminates erroneous results due to the use of dynamic ports for e.g. P2P applications[25][23].

For this data collection the measurement equipment was connected to the municipal network with an optical 50/50 splitters, which split the optical signal in two equal signal copies to keep the traffic intact. The measurements were carried out at the Internet Edge aggregation point where the ISPs connect to the network.

The PL stored all traffic in PCAP[1] files, which only contained Facebook related traffic since the collection was filtered with *facebook.com* and *fbcdn.net*. To maintain confidentiality and integrity all recorded traffic was anonymized by hashing the IP-addresses and MAC addresses before the data became available for this study.

We received the files in JSON[2] format, making it easier to extract valuable information through Python APIs.

The PL recorded traffic on Network North during 16 days, from October 17^{th} through November 1^{st}, 2012. Excluding the payload, each day generated approximately 2.4 Giga Byte (GB) of Facebook traffic, varying from 1.2GB - 2.8GB per files and day.

Traffic on Network South was recorded during 18 days, from September 21^{st} to October 8^{th}, 2012. Where the files were sized varying from 0.7GB-1.1GB day.

The amount of data collected is believed to be enough to give accurate results when analyzed as well as give a good representation of the Swedish Facebook usage. This data is only representative for Sweden; the results will not be applicable for other countries.

4.1.1 Test Environment

Before any scripts were executed on the actual data collection they were test run on a test environment containing 87 random files from 3 days. The only differences between the actual environment and the test environment was the amount of data, otherwise everything was identical. This ensured that scripts running without errors on the test environment would compile and run as intended on the actual environment.

4.2 Tools

As previously mentioned, the data collection tool was the PL. Wireshark was used to help understand the data before we implemented scripts in Python for the actual data analysis. Matlab was used as the plotting tool.

[1] Packet Capture Data file, a data file created by Wireshark during a live network capture. PCAP files contain data about the network characteristics[36].

[2] JavaScript Object Notation File is a File Description Standard data interchange format used for storing simple data structures and objects in a lightweight, text-based and human-readable format. Previously it was based on a subset of JavaScript, but it is now considered to be a language-independent format which supports many different programming APIs[37].

4.2.1 PacketLogic

The PacketLogic is an Intelligent Policy Enforcement (IPE) platform especially designed to be used in networks that are rapidly expanding their bandwidth beyond 1 Gigabyte per second (Gbps) and need support for both 1 (Gigabit Ethernet) GE as well as 10 GE within the same system. Application and content awareness with visibility into network as well as subscriber, device and location information can be combined since the PacketLogic's unique traffic identification engine, the PacketLogic Subscriber Manager and the Data stream Recognition Definition Language (DRDL) are supported. DRDL enables full Layer 7 visibility into applications which provides unique visibility into application behavior and service properties. A full suite of policy enforcement capabilities, such as congestion management and volume based shaping, are also supported by the PL.

Parallel queuing of traffic in multiple queues is supported in order to provide superior control for highly layered network architectures. Other features the system supports are asymmetric traffic and simplified clustering from network through FlowSync and QueueSync. The system records the traffic volume, the traffic application, the timestamps and the IP addresses and stores this in a statistics database.

Fine grained control per network and per subscriber can be set up as well as any combination of policy attributes including Networks, Subscribers, Applications, Universal Resource Locators (URLs), Referrers, Content Types, Virtual Local Area Networks/Multi-Protocol Label Switching (VLAN/MPLS) tags, Border Gateway Protocol Autonomous System (BGP AS) Paths, Time based, among others. The QoE a user is experiencing from the network per application basis as well as a network overview can be provided by the PL with realtime updates.

In order to support strong intelligent policy enforcement the PL is tightly integrated with the PacketLogic Subscriber Manager (PSM), which makes it possible for users to receive the same level of service as they move across the network. User based policies, traffic control and service packages for single users on the network can be created dynamically while a user is logged in. To provide statistical and visual information for networks, subscribers, devices and application performance the PL is integrated with the PacketLogic Intelligence Center (PIC) and the PacketLogic Report Studio. The valuable information gathered by this setup can be used by network operators to make informed business decisions on network conditions, congestion management and new services[28].

More than 1000 Internet application protocols can be identified by the PL which uses an connection-oriented identification process, matching each established connection to an application protocol[22]. Due to the PL's use of both payload based and host based behavior classification more than 95% of the traffic can be identified[25].

4.2.2 Wireshark

In order to create an understanding for what collected data in the PCAP files and the JSON files would look like and imply, we used the network protocol analyzing tool Wireshark. Through Wireshark it was possible to extract and identify a great variety of content sent in the packets, since it provided information from different Open Systems Interconnection (OSI) layers. Browsing could be done while a caption was in progress, which enabled us to get a good understanding for how Facebook actions related to the results in the PCAP dump. To separate Facebook traffic related to specific content, we created our own Wireshark filter rules.

4.2.2.1 Filters

The filter rule: **http contains www.facebook.com** or **www.fbcdn.net** separated the Facebook traffic from other miscellaneous traffic that could be neglected for this analysis. The Facebook PCAP files from the caption had been parsed and anonymized before they were available in JSON file format on the Hammy server and the Traffic-south server.

The other filters seen below, helped separate different Facebook activities, which could be used for several purposes in order to analyze the Facebook traffic. We created these filters in Wireshark to later implement them in Python to extract the actual data.

- **Likes**
  Likes are sent using a POST request which indicates that the information is stored in the payload. In the data-text-line the FBID of the user "liking" a content is shown as **_user** and the ID of the content that is being "liked" is shown as **ft_ent_identifier**.

  Wireshark Filter rule: **http.request.uri** contains “**like.php**“

- **Chat**
  Facebook messages are sent with the POST request and the FBIDs of the sender and receiver are hidden in the payload.

  Wireshark Filter rule: **http.request.uri** contains “**send_messages.php**“

- **Status Updates**
  Status Updates are made with the POST request and the FBID of the user updating the status is found in the payload as **xhpc_targetid**.
  Updates can be made by the owner of the wall as well as other Facebook users, since Facebook does not differentiate between a Status Update and a Wall Post. The Status Update or Wall Post can be seen in plain text in the payload.

  Wireshark Filter rule: **http.request.uri** contains “**updatestatus.php**“

- **Comments**
  POST requests are used to send Comments. The FBID of the user posting the Comment is part of the payload as well as the Content ID belonging

the target of the Comment, which is included in **ft_ent_identifier**. The Comment is seen in plain text in the payload.

Wireshark Filter rule: **http.request.uri** contains "**add_comment.php**"

- **Tags**
  When somebody is tagged in a photo, post or other activity all information is sent with a POST. The payload contains all necessary information such as the user who tags, the subject who gets tagged as well as the ID of the content where the Tag appears.

  Wireshark Filter rule: **http.request.uri** contains "**tagging_ajax.php**"

- **Downloaded Pictures**
  During a Facebook session a large number of pictures are being downloaded automatically without any user interaction. These are pictures of advertisement, pictures as they appear on the initial site when a user logs in such as small album pictures, icons, pictures of friends and pictures connected with posts. A user can choose to download a picture by clicking on it. All pictures are requested with GET request which means that the picture URL is part of the header and not the payload. The name of the picture as well as other information about the user who downloads the picture is included in the header as well. The pictures on Facebook can be of the following formats: jpg, gif, png and tif, which all can be detected by filtering on the URL.

  This filter shows all downloaded pictures, voluntarily downloaded as well as automatically downloaded.

  Wireshark Filter rule: **http.request.uri** contains "**.jpg**"or
  http.request.uri contains "**.gif**" or
  http.request.uri contains "**.png**" or
  http.request.uri contains "**.tif**"

- **Uploaded Pictures**
  Uploaded pictures are sent with a POST request which contains a Multipart Multimedia Encapsulation where the picture is encapsulated. The FBID of the user who posts the picture is found in the URL as well as the subject and the content ID.

  Wireshark Filter rule: **http.request.uri** contains "**upload/photos**"

- **Video**
  Videos are usually uploaded from an external site like YouTube.com or Vimeo.com and are therefore harder to track since they are not stored on an actual Facebook server. The preferred video format is .mp4 but Facebook

supports most formats as seen in the list below[29].

3g2	Mobile Video	3gp	Mobile Video
3gpp	Mobile Video	asf	Windows Media Video
avi	AVI Video	dat	MPEG Video
divx	DIVX Video	dv	DV Video
f4v	Flash Video	flv	Flash Video
m2ts	M2TS Video	m4v	MPEG-4 Video
mkv	Matroska Format	mod	MOD Video
mov	(QuickTime Movie	mp4	MPEG-4 Video
mpe	MPEG Video	mpeg	MPEG Video
mpeg4	MPEG-4 Video	mpg	MPEG Video
mts	AVCHD Video	nsv	Nullsoft Video
ogm	Ogg Media Format	ogv	Ogg Video Format
qt	QuickTime Movie	tod	TOD Video
ts	MPEG Transport Stream	vob	DVD Video
wmv	Windows Media Video		

The filter used shows when a video is requested in order to be watched, this is sent with a GET request.

Wireshark Filter Rule: **http.request.uri** contains “**get_video**“

4.2.3 Python

The dynamic programming language Python is used in a wide variety of application domains. Python has a clear, readable syntax, strong introspection capabilities intuitive object orientation, natural expression of procedural code, full modularity, supporting hierarchical packages, exception-based error handling, very high level dynamic data types, extensive standard libraries and third party modules for virtually every task, extensions and modules easily written in C, C++ and is embeddable within applications as a scripting interface. Overall Python is powerful, fast and easy to use with the extensive support libraries covering everything from asynchronous processing to zip files and the highly optimized byte compiler. Python integrates well with others such as COM, .NET and COBRA objects and Python is also supported for the Internet Communications Engine (ICE) and many other integration technologies.

A complete documentation for Python is found both on the web as well as integrated into the language. The Python implementation is under an open source license which is administered by the Python Software Foundation[30].

4.2.3.1 NetworkX and SciPy

The NetworkX extension module is a network graph tool available under BSD for Python, it is capable of processing graphs, plotting graphs and saving files in several formats, including pdf [31]. The open source Python module SciPy contains tools for scientific calculations inside Python [32].

4.3 Limitations

The scope of this thesis is limited to traffic monitored on the access network. Facebook's optional setting to use HTTPS decreases the amount of data that can be used for this analysis since encrypted data does not reveal any useful information.

4.3.1 Mobile Traffic

Only traffic from the access network is measured with the PL hence no mobile data is available for this study. Mobile data includes mobile data and Internet traffic generated by handsets, notebook cards, and mobile broadband gateways. The use of mobile data has skyrocketed the past ten years and it is believed to keep on increasing. Globally, mobile data traffic is predicted to increase 18-fold between 2011 and 2016, reaching 10.8 exabytes per month by 2016. Due to the heavy use of cellphones as they become more and more advanced the mobile data is expected to grow three times faster than fixed IP traffic between 2011 to 2016. Global mobile data traffic was 2% of total IP traffic in 2011 but is predicted to extend to 10 percent of all total IP traffic in 2016[26].

In December 31, 2011 Facebook had 432 million mobile active users per month according to an updated filing with the Securities and Exchange Commission. Out of these users approximately 13% accessed the social network exclusively through mobile devices. This showed a year-over-year growth with 76% from December 2010 when the number of mobile active users were 245 million. In solely during December 2011, 58 million users were estimated to have accessed the social network through mobile apps alone and 374 million active users visited Facebook from both PCs and mobile devices during that same month. This rapid growth of the mobile app usage is probably caused by the increased number of smartphone users that year, the growing popularity of tablets as well as product enhancements across several mobile platforms[33][34][35].

4.3.2 HTTPS

A major problem on the Internet is to ensure that information is kept secured from hackers, malware and viruses. Facebook offers a number of security features such as identify your friends, remote logout, captchas and one-time passwords as well as Secure Browsing.

The option to use Secure Browsing; HTTPS; Secure Socket Layer/Transport Layer Security (SSL/TLS) encryption, was added to Facebook in January 2011. It secures the communication between the Facebook Web servers and the browsers in order to avoid Session Hijacking and Packet Sniffing which could jeopardize the privacy of information. Before 2011 HTTPS was only used during the log-in procedure, where the password entered was sent via SSL.

HTTPS is not enabled by default and users have to turn on Secure Browsing under the Security tab Settings. Facebook encourages its users to turn on Secure Browsing if they usually access the site from public Internet access points found at coffee shops, airports, libraries or schools.

In the beginning of 2011 Facebook developer Alex Rice claimed that HTTPS would be the default setting in the future, two years later this is not yet the case. Many third-party applications do not support Secure Browsing which seems to be one of the reasons it is taking longer than expected to implement this feature as default. Another disadvantage is that encrypted sites may take a little while longer to load, possibly slowing down Facebook activities[38].

Traffic from Facebook users that have enabled Secure Browsing are not part of this study. Wireshark is not able to detect encrypted Facebook traffic unless the IP is explicitly filtered on, but this still does not reveal any of the information inside the packets sent to the Facebook server. Therefore this analysis excludes all users that use HTTPS for Facebook and this could tamper with the results of this thesis. Since HTTPS is an optional setting and not on by default the results are believed to be valid and not misleading when the users with Secure Browsing turned on is neglected.

Chapter 5
Results

The goal of this thesis is to see whether content demand patterns can be used to detect groups of users related to each other; community clusters. In order to do this we must be able to identify users which we have chosen to do through three different identification methods, where two of these methods are developed by us. We also look at device usage patterns and analyze these on a a network level, on a cluster level as well as an individual user level.

5.1 Identification of Unique Users

In order to develop a hypotheses for how many unique users that can be identified, we had to measure the number of actual users that connected to Facebook on the network. We have identified one method of identifying users in [48] and we will compare that to two methods developed by us. We will also present an analysis of each in order to determine which one that generates the most accurate result; which method that can identify a unique user with the highest probability. The three identification methods are identification through MAC address [48], identification through IP address and identification through a `<MAC, UNIX time>`-tuple.

The first step in this analysis is to determine the actual number of Facebook users in the networks. We have discovered that during an active session the client browser repeatedly sends a batch of cookies to Facebook and based on this, a user-count can be carried out through an examination of the cookies.

Below is a truncated example of what cookie-batches can look like when they are extracted from the data packet:

```
"Cookie":  "datr=[...]; fr=[...]; lu=[...]; c_user= 0123456789;"
```

The interesting cookie instance is the `c_user` Cookie ID which contains the FBID[1], indicating the actual owner of the active session. Through a script that extracted all the packets, which contained the `c_user` cookie, the number of unique FBIDs was found. 14387 unique FBIDs was found for Network North and 7952 for Network South. We have corrected this number with respect to erroneous FBIDs

[1] The FBIDs are unique static numbers, assigned to each user, distributed through the unsigned 32-bit int space (though newly assigned FBIDs have a 32-bit int with 10000 added before the actual ID) [51].

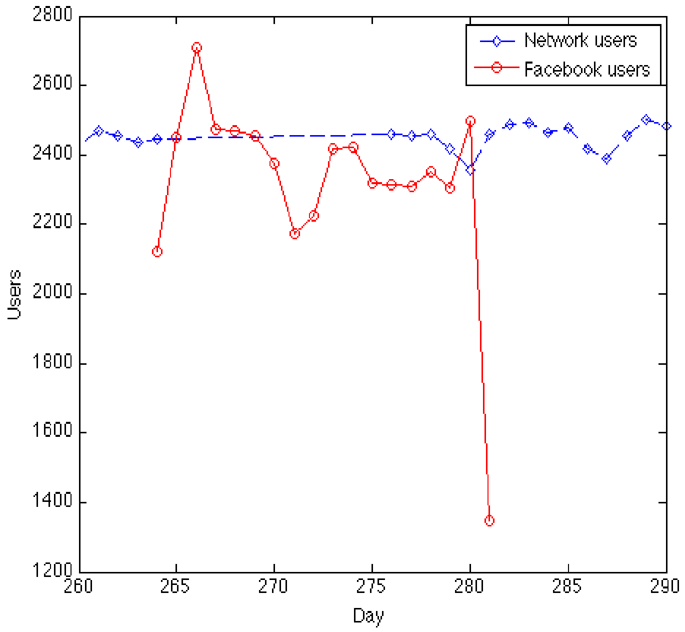

Figure 5.1: Network users and Facebook users, Network South.

sent, but it should be taken into consideration that there might still be a small number of incorrect ones left. We have defined an erroneous FBID as a FBID that is not connected to a user, for example $-1, 0, 1, 1000$.

5.1.1 Facebook Users in Relation to Network Users

Based on measurements provided by Acreo the number of households connected to Network South was available. The plot in Figure 5.1 shows this in combination with our own results for the number of unique FBIDs. As can be seen in the graph, there is a gap in the data provided by Acreo, caused by malfunctioning measurement equipment effectively blocking all attempts at correlating the two datasets using the *Pearson Correlation Coefficient*, see Equation 5.5. We do suspect that a weak positive correlation is present.

5.1.2 Identifying Unique Users by MAC

In [48] it was assumed that a MAC address could identify a unique user. This was based on the MAC addresses constant and non-changing nature, i.e. that a MAC address always is associable to the same network device. To determine the accuracy of this claim we first counted the number of unique MACs. We found 4286 unique MACs on Network North and 2052 on Network South. After this we performed and analyzed a mapping of each FBID to the MAC address(es) in use, see Figures 5.2 and 5.3. We found that the majority of Facebook users only used a single device connected to the measured networks.

We also mapped each MAC address to each FBID that the device had used, see Figures 5.4 and 5.5. These results indicates that, on average, each MAC address would represent approximately 3.4 FBIDs on Network North and 3.8 FBIDs on Network South, thus it is flawed to try to represent a unique user with this mapping technique. The conclusion that $\|MACs\| < \|FBIDs\|$ is also supported by Figure 5.1.

5.1.3 Identifying Unique Users by IP

We have now shown that identifying unique users by MAC addresses is inadequate, therefore a test was run in order to see if a unique user could be identified by the source IP address instead. The IP addresses are distributed dynamically on the measured network and they lack the constant aspect of the MAC addresses (when an IP is renewed the IP in the packets analyzed changes). A measurement was made to determine the amount of unique IP addresses and we discovered it to be marginally higher than the amount of unique MAC addresses. This shows that identification by IP addresses generates an even greater mislead compared to identifying users by MAC addresses.

5.1.4 Identifying Users Using a `<MAC, Unix time>`-Tuple

We have discovered that every user sends the `c_user` cookie occasionally and not in every single request sent to the server. This implies that the cookie transfer alone cannot be used to actually identify which users that are responsible for certain activities.

The latest approach, developed by us, in trying to identify unique users is significantly more sophisticated than the previous methods. It is based on the idea that at a single discrete time step the following relation is true (the use of $\approx$ is very liberal):

$$MAC_{unique} \approx IP_{unique} \approx FBID_{unique}$$

The idea we had behind this technique is to identify at what time intervals a MAC address is related to a specific FBID. In the JSON structure provided by Acreo each packet has a Unix time stamp and the FBIDs can be matched with the MACs to finite sessions using these time stamps. This means that a packet can be matched to a specific FBID by looking at what FBID the `<MAC, Unix time>`-tuple resolves to in the resulting data structure (see Figure 5.6 for a graphic representation).

The crux is to choose the session end time, t_{x2}, sufficiently well so that the resulting sessions for one MAC does not overlap another. The following algorithm, developed by us and here presented in pseudo-Python, describes this procedure:

```
type_def session = ((t_start, t_end), FBID)
map = map <MAC, List<session>>
init map so there is an empty list of sessions for all MACs

```

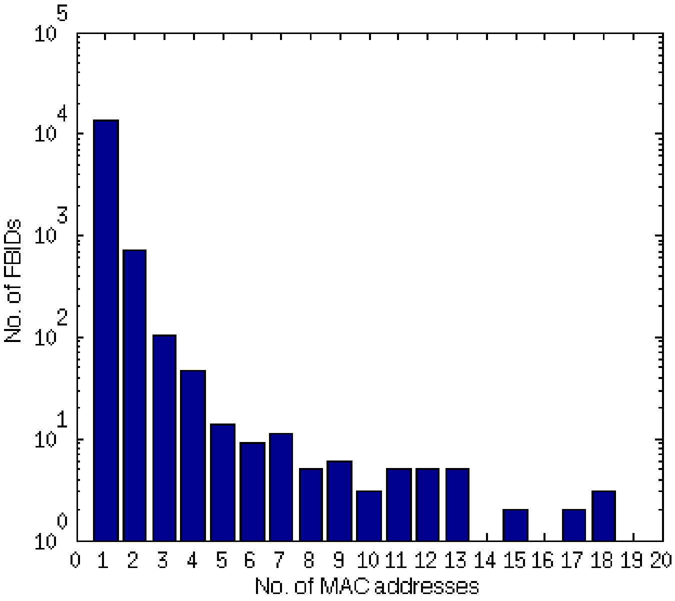

Figure 5.2: No. of FBIDs/no. of MACs on Network North, i.e. the amount of FBIDs that uses a certain number of MACs

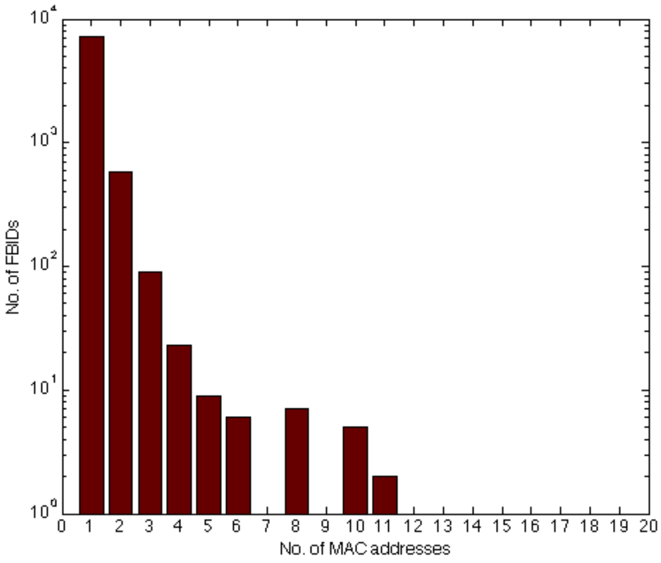

Figure 5.3: No. of FBIDs/no. of MACs on Network South, i.e. the amount of FBIDs that uses a certain number of MACs

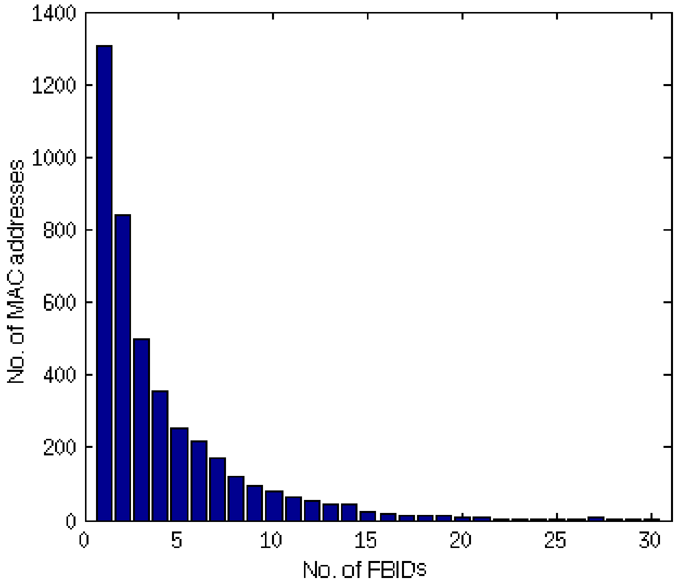

Figure 5.4: No. of MACs/no. of FBIDs on Network North, i.e. the amount of MACs that connect to a certain amount of FBIDs

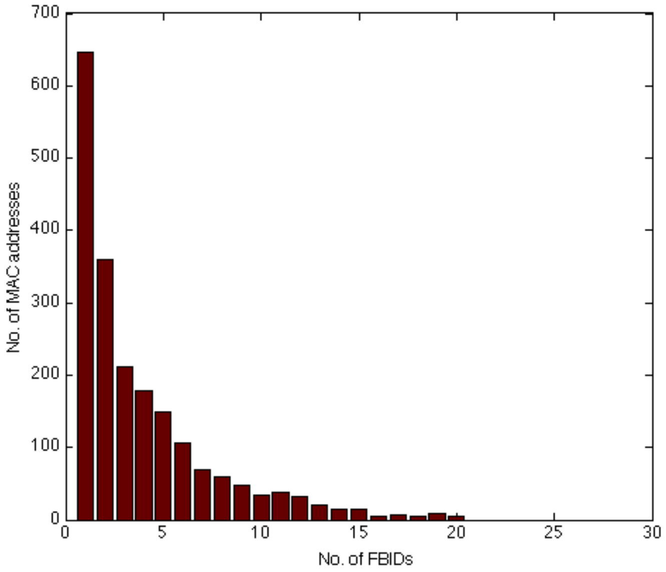

Figure 5.5: No. of MACs/no. of FBIDs on Network South, i.e. the amount of MACs that connect to a certain amount of FBIDs

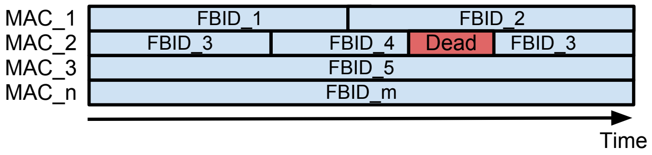

Figure 5.6: Graphic schedule of some example Facebook user sessions. Note the session denoted "Dead", this would be a session where no single user could be bound to the specific MAC address, i.e. there were two or more overlapping sessions during this time span.

```
For all packets p containing c_user cookie:
  set t = time stamp of p
  set fbid = contents of c_user cookie
  set session_list = map[source MAC of p]

  if this is an ongoing session:
    continue
  else:
    pop the last (unfinished) session
    fix that session so it is terminated at t-1
    append that session to the list
    create and append a new session to session_list from time t to inf for fbid
    continue

for all session_lists in map:
  for all sessions in session_list:
    remove overlapping sessions #these are simultaneous 1-second sessions.
    if the session is first in a session list:
      extend it's starting time to the start of the day
    elif the sessions is the last session:
      extend it's ending time to the end of the day
```

The `map`-variable could look like the following after the above procedure finished (again, see Figure 5.6 for a graphic representation). In the below example when performing a query using the tuple <MAC_2, t>, $t_{41} \leq t \leq t_{42}$, $FBID_4$ the following would be returned.

$$\begin{array}{ll}
MAC_{no} & : Session\_list \\
MAC_1 & : [((t_{11}, t_{12}), FBID_1), ((t_{21}, t_{22}), FBID_2)] \\
MAC_2 & : [((t_{31}, t_{32}), FBID_3), ((t_{41}, t_{42}), FBID_4), ((t_{33}, t_{34}), FBID_3)] \\
MAC_3 & : [((t_{51}, t_{52}), FBID_5)] \\
MAC_n & : [((t_{n1}, t_{n2}), FBID_m)]
\end{array}$$

To determine the quality of these tables we calculated and measured the percentage of *dead time* (see Figure 5.6) for each table. We define dead time as the total amount unsessionizable time. This is measured by calculating the sum of the session time of all the MACs and dividing this by the optimal value, being the total length of a day times the number of MACs (86400 seconds in particular).

$$\left(\frac{\sum^{MACs}\sum^{sessions} sessiontime}{\sum^{MACs} 86400}\right) = deadtime\%$$

The session tables for the days *20121017, 20121018 and 20121019*, from the measurement performed on Network North, all have $< 1\%$ dead time, corresponding to $<< 1$ minutes of total unidentifiable time per day.

5.2 Content Demand Pattern Analysis

Earlier work regarding the downloading (or *demanding*) and viewing of Facebook pictures have been done in [48]. However, there it is assumed that a user can be identified by using the source MAC address of the HTTP-frame, As we have showed earlier in Section 5.1.2, this method is faulty. We perform identification of users using the above mentioned identifications tables and, later, we will build a bipartite graph (see Figure. 5.7) for each regarding images and the users downloading them.

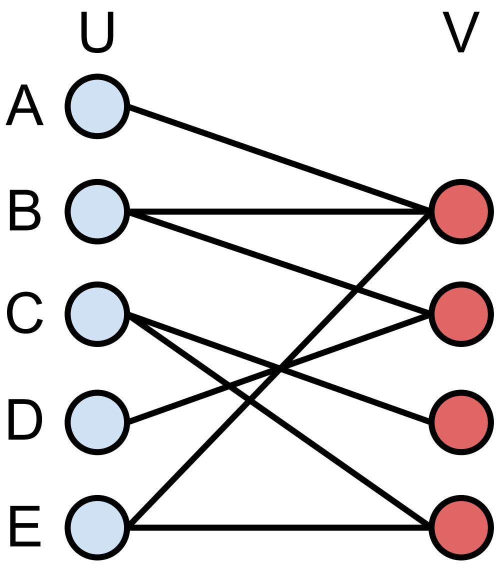

Figure 5.7: Sample Bipartite Graph where the left column, U, denotes users and the right column, V, denotes images.

We filter out all GET requests for images whose Universal Resource Identifier (URI) contains the string **hphotos** or **hprofile**. From these requests all URIs not representing full sized pictures, i.e. URIs containing *100x100/*, *206x206/*, *480x480/*, *720x720/*, *_q.jpg*, *_s.jpq*, *_t.jpq* or *_a.jpq*, are removed. This means that the what-, when- and who-information is available for all image GET requests. Using the identification tables users can now be identified and associated to the image demands performed by that unique user. Tables 5.1 and 5.2 accounts for the identification success rate for each day, meaning the number of (un)identifiable requests and the number of users (not) found. Users not found are users who did not download any pictures.

Table 5.1: Identification Success Rate, Network North

Day	Identifiable requests	requests total	% identifiable requests	Users identified	Users total	% identifiable
2012-10-17	563970	618151	91%	3874	5254	74%
2012-10-18	660622	787521	84%	4207	6051	70%
2012-10-19	668893	797299	84%	4113	5876	70%
2012-10-20	731315	908512	80%	4051	5979	68%
2012-10-21	869099	1048145	83%	4385	6441	68%
2012-10-22	2143616	2504046	86%	6407	9556	67%
2012-10-23	674046	807959	83%	4219	6107	69%
2012-10-24	688788	825671	83%	4294	6147	70%
2012-10-25	705265	842253	84%	4285	6177	69%
2012-10-26	656790	784948	84%	4120	5718	72%
2012-10-27	683299	860706	79%	3914	5658	69%
2012-10-28	903141	1050074	86%	4365	6172	71%
2012-10-29	792273	967649	82%	4276	6222	69%
Mean	778932	994841	84%	4346	6258	69%

Table 5.2: Identification Success Rate, Network South

Day	Identifiable requests	Requests total	% identifiable requests	Users identified	Users total	% identifiable
1012-09-21	237591	249810	95%	1518	2121	71%
1012-09-22	335572	353395	94%	1709	2452	69%
1012-09-23	402855	420668	95%	1873	2710	69%
1012-09-24	347012	365983	94%	1773	2472	71%
1012-09-25	300094	315762	95%	1745	2469	70%
1012-09-26	311381	327981	94%	1745	2455	71%
1012-09-27	305698	496643	61%	1711	2377	71%
1012-09-28	260439	311959	83%	1623	2174	74%
1012-09-29	272521	286978	94%	1604	2223	72%
1012-09-30	319037	337150	94%	1745	2416	72%
1012-10-1	279878	292757	95%	1735	2424	71%
1012-10-2	266698	279448	95%	1654	2319	71%
1012-10-3	244635	260939	93%	1661	2315	71%
1012-10-4	269210	289462	93%	1686	2311	72%
1012-10-5	257012	271877	94%	1652	2353	70%
1012-10-6	288267	304277	94%	1631	2304	70%
1012-10-7	321849	342996	93%	1780	2499	71%
1012-10-8	82388	88199	93%	962	1350	71%
Mean	283452	310905	91%	1656	2319	71%

As can be seen around 30% of users on average do not download any pictures, this is about the same in both networks. We can identify around 83% of the requests on Network North and around 94% on Network South. We now construct the bipartite graphs from the identification tables and the request data and they can be described as the following mathematical object $G = (U, V, E), E = (u, v), u \in U, u \notin V, v \in V, v \notin U$, U being the node set of requested images and V the node set of Facebook users.

The translation to bipartite graphs for these kind of provider- requestor networks is not new and research exists on how to analyze these. Papers mention *k-clique percolation*, *biclique percolation*, *modularity maximization via simulated annealing*, *modularity maximization using spectral decomposition*, *clustering coefficients*, *trust-graphs* etc. [40, 45, 44]. The graphs produced here are, however, unsuitable for some of the analysis methods above, namely the ones relying on solving NP-hard[2] problems (i.e. k-clique). This being due to the sheer size of the bipartite graphs, $(\mathsf{U} + \mathsf{V} >> 300000)$, $(\mathsf{E} >> 700000)$ for example.

5.2.0.1 The Representation of Graphs

In this thesis we use a very simple, yet powerful method, for representing graphs. When graphs are saved to disk they are saved in text files formatted as in Table 5.3 with each row being printed on a row in the file and each column is a tab-separated value in the corresponding row. Our programs that utilizes the graphs represent them in memory as the following: `map<node, list<edges> >`, this approach enables access of nodes in the graphs in $\leq O(\|E\|)$ time and keeps memory usage within $\leq O(\|V + E\|)$, a decent tradeoff in access time and simplicity vs. memory-space usage.

Table 5.3: Representation of Graphs

Node	Edges			
$Node_0$	$edge_{00}$	$edge_{01}$	$\ldots$	$edge_{0n}$
$Node_1$	$edge_{10}$	$edge_{11}$	$\ldots$	$edge_{1n}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$Node_m$	$edge_{m0}$	$edge_{m1}$	$\ldots$	$edge_{mn}$

5.2.1 2-Step Neighborhood

In the process of finding an accurate way to analyze these large graphs, it is easy to believe that it is of interest to examine how many other users a user is connected to via downloaded images. Since the graphs are bipartite the neighborhood of a user node consist of the nodes of downloaded images, so in order to see user-to-user connections the two-step neighborhood of the user node must be calculated. Through this we found the mean size of the user's two-step neighborhood to be 734

[2] At least if $P \neq NP$. It is outside of the scope for this thesis to prove this but it is generally believed to be true [49].

and the standard deviation to be 634. We cannot derive any real conclusions from this as results between 40 and 900 are all within the expected interval. Because of this we have chosen to only perform this analysis on Network North.

5.2.2 Trust

An interesting approach to analyzing relations between nodes in graphs is based upon the sociological idea that people who share common interest in some things are likely to share a common interest in others. It is called *trust between nodes in networks* and is a measurable quantity in space [0 : 1]. Here we use trust as a measure of whether two users are likely to be interested in the same kind of images. The formal definition of trust is given in Equation 5.3 that depends on the *Jaccard Index* and a *Distance Value* based upon the shared items between involved users, see Equations 5.1 and 5.2 [40].

Note that $\alpha + \beta + \gamma = 1$ since this normalizes the trust function, $0 < \sigma < 1$ and that SI is the set of shared items between the users and should be non-zero. β and γ decides the relative influence for the corresponding term and α represents the chance that two users trust each other without sharing any observed images. σ is used in the Distance Function in 5.2 for deciding limits on popularity: i.e. when an image can be regarded as popular. The Distance Function can be viewed as "the common interest distance between users", with increasing distance the common interests decreases. The idea behind this function is that something that is downloaded by many users tells less about the trust between the users than something that is downloaded by only a few users [40].

$$J(u,v) = \frac{\|N_u \cap N_v\|}{\|N_u \cup N_v\|} \tag{5.1}$$

$$D(i) = (\frac{2}{1 + e^{-deg(i)^{\sigma} + 2^{\sigma}}} - 1) \tag{5.2}$$

$$Trust(u,v) = \alpha + \beta J(u,v) + \gamma(1 - \frac{\sum_i^{i \in SI} D(i)}{\|SI\|}), \tag{5.3}$$

The non-zero trust values, using $\alpha = 0$, $\beta = 0.5$, $\gamma = 0.5$ and $\sigma = 0.99$, for the nodes in the sample graph graph in Figure 5.7 is seen in Table 5.4.

Table 5.4: Non-Zero Trust Values for the Sample Graph in Figure 5.7

u	v	Trust(u, v)
A	B	0.52
A	E	0.52
B	E	0.43
B	D	0.75
C	E	0.67

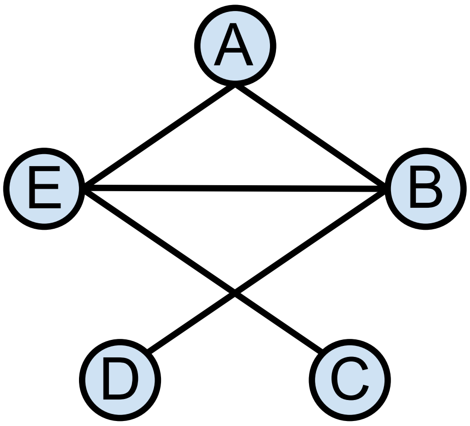

Figure 5.8: A sample trust graph built from Figure 5.7 using the trust values in Table 5.4.

5.2.3 Trust Graphs

After calculating the trust values we build a new graph, the so called *trust graph*, which is constituted of the user-nodes only. The content nodes are now removed, and edges are placed between users that fulfill a certain criteria, $G(V, E)$, $E(u, v)$, $trust(u, v) \geq x$. For example where $trust(u, v) > 0$ we can build the trust graph in Figure 5.8 from the bipartite graph in 5.7. It is also possible to create a weighted graph where the weight of an edge is the trust between the nodes it connect. By using $\alpha = 0$, $\beta = 0.5$, $\gamma = 0.5$, $\sigma = 0.3$ and rules $trust(u, v) \geq 0.4$, $trust(u, v) \geq 0.5$ or $trust(u, v) \geq 0.65$ three very different trust graphs could be constructed. As can be seen in Tables 5.5 and 5.6 these three graphs differ immensely in both size and density indicating that a threshold t, $0.5 \leq t \leq 0.65$, exists and that the number of node pairs u, v with $trust(u, v) \geq t$ decrease rapidly at a slight increase of t.

The graph built with rule $trust(u, v) \geq 0.65$ is not big enough to derive any results from, but the graphs fulfilling $trust(u, v) \geq 0.4$ and $trust(u, v) \geq 0.5$ contain enough information about the social networks. Each node in these graphs has been shown to trust each of the nodes it has an edge to. Also note that $3650 < \|V\| < 5841$ and $30260 < \|E\| < 44446$ (for the larger set where $trust(u, v) \geq 0.4$), which leads to a dramatic decrease in graph size compared to the bipartite image-downloader graphs mentioned earlier.

5.2.4 Clustering Coefficient

The clustering coefficient C_n of a node in a graph G is defined as $C_n = 2e_n/N_n(N_n-1))$ where e_n is the number of connected pairs between all neighbors N_n of n. This is the same thing as E_N/E_{max} where E_N is the number of edges in the neighborhood N_n and E_{max} the maximum number of edges. Note that $C_n \in [0:1]$. The clustering coefficient for the whole graph is given as $C = 1/n\Sigma_{i=1}^{n} C_i$ and is given in Tables 5.7 and 5.8 for the trust graphs produced and the cluster coefficient distribution can be seen in Figures 5.9 and 5.10 [50, 51].

5.2.5 Community Clustering

Once we had created the trust graphs we could begin to find community clusters in the graphs, indicating what groups of users are interested in similar content. The literature mention several clustering algorithms, including *minimization of cut-conductance*, *local density maximization*, *single cluster editing minimization*, *cluster quality maximization* [47], all of which are proven to be NP-complete [46] and therefore only interesting from a theoretical point of view. Some algorithms in P[3] for calculating clusters are *mincut*, *Markov Chain Cluster* and *Chinese Whispers*. The first two of these three have the drawback that the number of clusters have to be pre-determined, which is avoided in the latter [47].

5.2.5.1 Community Clustering using Chinese Whispers

The clustering algorithm *Chinese Whispers* described in [47] is randomized and designed to perform with great speed on the huge graphs produced in Natural Language Processing. Graphs with tens of thousands of nodes and edges can easily be clustered in minutes using only a modern workstation. The algorithm, in pseudo-code can be seen below (note that `no iterations` are needed because the algorithm is non-converging).

```
let G(V, E) be a graph
for v_i in V:
  class(v_i) = i
for i in range(no of iterations):
  for v in V, randomized order:
    class(v) = predominant class in neighborhood(v)
return partition P induced by class labels
```

As mentioned earlier, this algorithm uses randomization and is, as such, non-deterministic. The predictability of the algorithm increases with cluster-size and for clusters of size > 10 the chance of correct separation is > 9.5. These performance checks were done by clustering n-bipartite-clique-graphs, see Figure 5.11 [47].

When running *Chinese Whispers* on the trust graphs built earlier (where $\alpha = 0$, $\beta = 0.5$, $\gamma = 0.5$ and trust-threshold > 0.5), and checking the sizes of the resulting clusters, we end up with the histograms in Figures 5.12 and 5.13. It can be seen that the majority of the clusters found are relatively small, with sizes $0 < size < 5$. This is most likely due to the limited sample size and the fact that each cluster has been built from trust graphs that represents one days worth of data. When we performed the algorithm on a joined graph, where all days worth of data was represented from Network North, measurement 8954/9113 nodes were

[3] P is the computational complexity class for problems solvable in polynomial time[49].

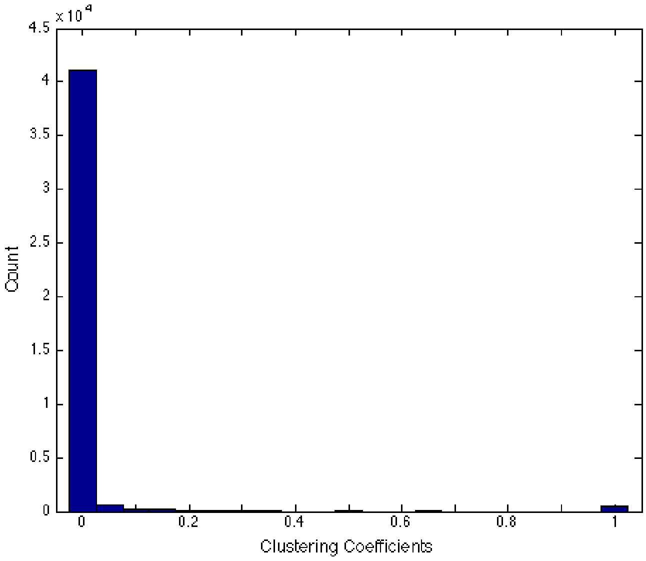

Figure 5.9: Clustering coefficient distribution for the trust graphs corresponding to $\alpha = 0$, $\beta = 0.5$, $\gamma = 0.5$, $\sigma = 0.3$ and rules $trust(u, v) \geq 0.5$, Network North.

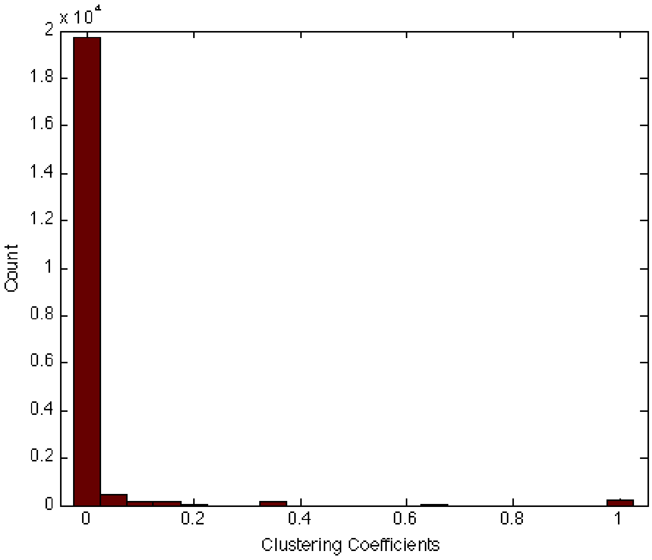

Figure 5.10: Clustering coefficient distribution for the trust graphs corresponding to $\alpha = 0$, $\beta = 0.5$, $\gamma = 0.5$, $\sigma = 0.3$ and rules $trust(u, v) \geq 0.5$, Network South.

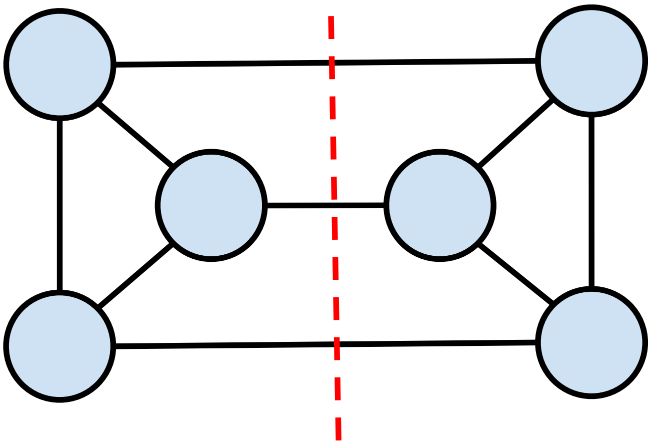

Figure 5.11: An n-Bipartite clique graph, $n = 3$, a very difficult type of graph to cluster. The dashed line denotes a successful clustering.

put in the same cluster indicating that the density of the joined graph was far to high for that particular graph to be meaningful to cluster. This is likely to be remedied by careful trust parameter configuration. To examine whether any correlation exists between cluster size and clustering coefficients we paired and scatter plotted the two measurements, see Figures 5.14 and 5.15. Then the *Pearson Correlation Coefficient*, R, and the *p-value*, p, were calculated, using the definition of Pearson Correlation Coefficient in Equations 5.4 and 5.5 by the Matlab function `corrcoef(X)`. The reason for choosing *Pearson Correlation Coefficient* is that the data sets are consisting of ratio-data (that is; ordered, with meaningful intervals and a starting point 0). The results were $R = -0.0114, p = 0.4996$ and we see that no conclusions can be drawn about an eventual correlation between clustering coefficient and size due to the large p-value [52, 53, 54].

$$C(i,j) = E\big[(X_i - E[X_i](X_j - E[X_j]))\big] \tag{5.4}$$

$$R(i,j) = \frac{C(i,j)}{\sqrt{C(i,i)C(j,j)}} \tag{5.5}$$

5.2.5.2 Clustering Coefficients in Clustered Graphs

We calculated the clustering coefficients for the nodes in the clustered subgraphs and the results can be seen in Figures 5.16 and 5.17. The vast majority of the nodes still have a clustering coefficient of 0, with a mean coefficient of 0.0176 and

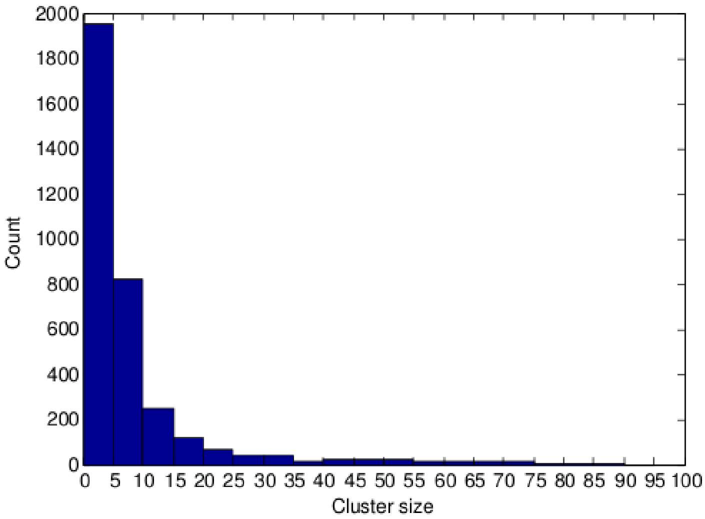

Figure 5.12: Cluster sizes after the Chinese Whispers Algorithm has been run on the trust graphs from Network North.

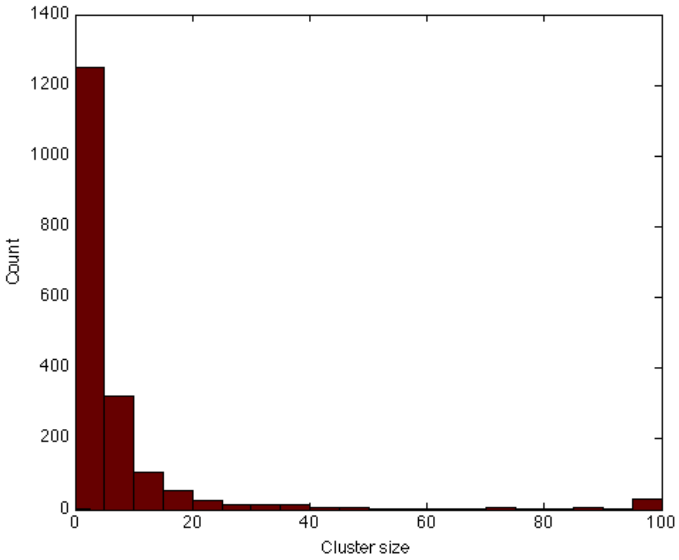

Figure 5.13: Cluster sizes after the Chinese Whispers Algorithm has been run on the trust graphs from Network South.

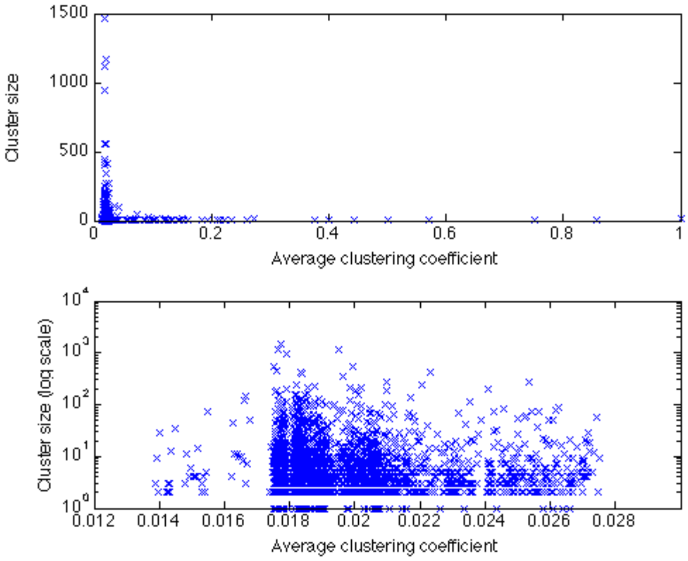

Figure 5.14: Average clustering coefficient and size, above is a normal plot and below a plot in $x \in [0.012, 0.030]$ and logarithmic Y-axis, Network North.

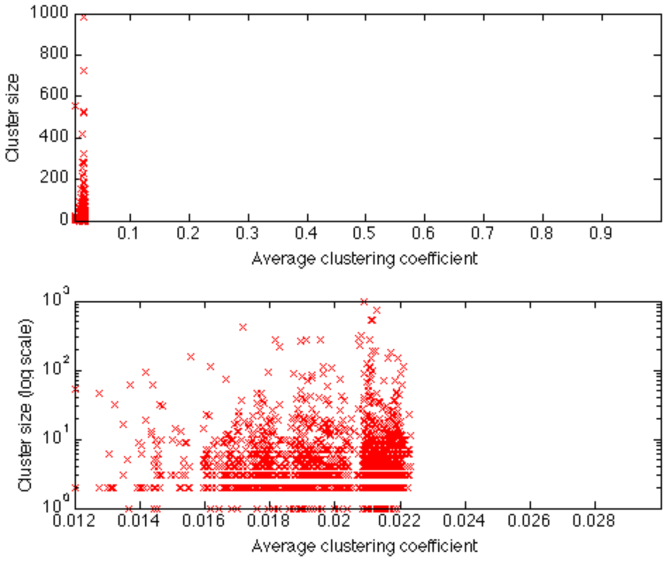

Figure 5.15: Average clustering coefficient and size, above is a normal plot and below a plot in $x \in [0.012, 0.030]$ and logarithmic Y-axis, Network South.

standard deviation of 0.1220. This mean value is slightly above the mean of the values in Table 5.9 indicating that the extracted clusters are more clustered than the trust graphs are.

5.2.5.3 Clusters as Sub Graphs of Trust Graphs, a Graphic Approach

We isolated and plotted a cluster ($\|V\| = 26$) using the Python *NetworkX*-module and it looks like Figure 5.18. When the same cluster is put into its 1-hop and 2-hop neighborhood context ($\|V\| = 569$) it appears as in Figure 5.19. These figures accurately show the immense complexity of the trust graphs, even if only a smaller subset of a complete graph ($\|V\| = 2905$, this is from 20121017, Network North) is shown.

5.3 Online Devices

Facebook looks different from OS to OS and some features might be more enhanced in one OS compared to another, which could lead to differences in user patterns. Different devices may also have dissimilar options for Facebook, e.g. it is not possible to tag someone in a picture from the Facebook Android App, but this is possible from computers, even if the most common features are available across all OSs and for all devices. Since widgets and applications, both desktop based ones as well as toolbars, usually provide the same functionality across OSs with slightly different designs and interfaces, the user patterns across OSs and devices are believed to be very similar. The focal point of our Surfing Device analysis aims to reach an understanding for how user patterns differs between surfing devices and not OSs. We have used Information from the *user_ agent* string to determine what type of device that a Facebook user was active from. To do this we built an HTTP-parser that detected what type of device, such as tablets, playstations, computers and mobile phones that the http request was sent from.

Beneath are examples of *user_ agent strings*:

- *Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0)*
- *Mozilla/5.0 (iPod; U; CPU iPhone OS 4_ 2_ 1 like Mac OS X; sv-se) AppleWebKit/533.17.9 (KHTML, like Gecko) Version/5.0.2 Mobile/8C148 Safari/6533.18.5*

The first example *user_ agent* string indicates that the request is sent from a Windows 7 OS, revealing that the surfing device during the Facebook session was a Computer running Windows 7. The second example shows that an iPhone was used.

We found certain *user_ agent* strings that were not possible to link to a specific OS or device, as in the following example:

- *Apache-HttpClient/UNAVAILABLE (java 1.4)*
- *Facebook Update/1.2.205.0;winhttp;cup*
- *Mozilla/4.0 (comp)*

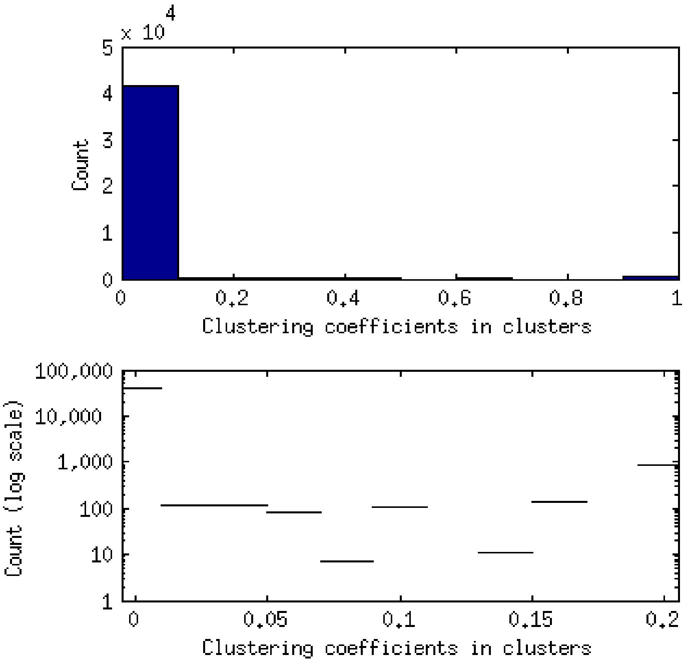

Figure 5.16: Clustering coefficients for nodes in clusters, Network North.

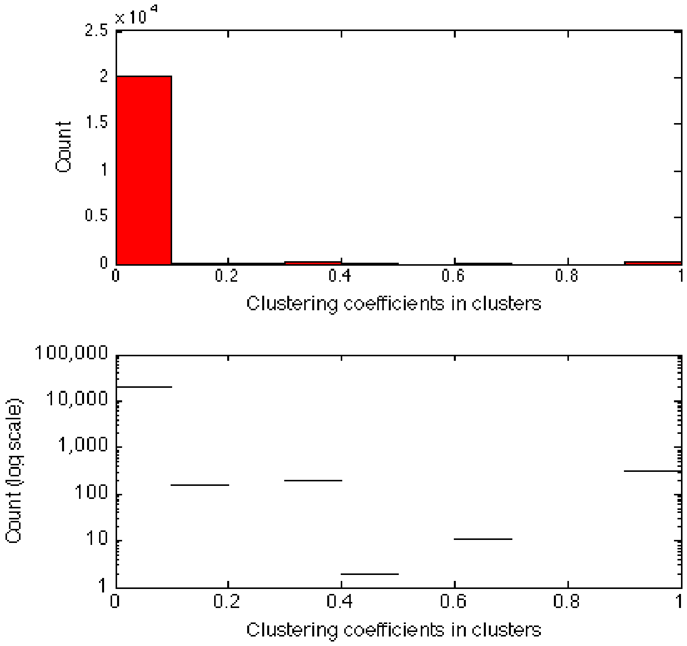

Figure 5.17: Clustering coefficients for nodes in clusters, Network South.

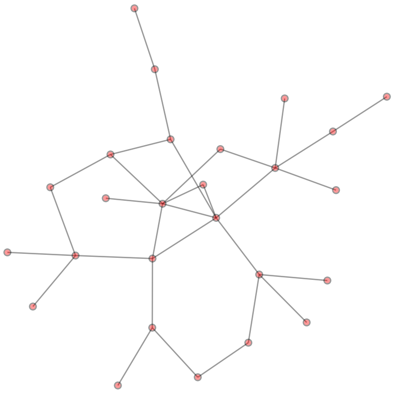

Figure 5.18: Figure of an isolated cluster from 20121017, Network North.

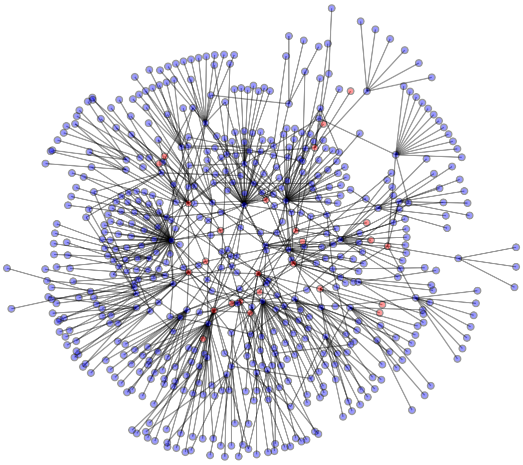

Figure 5.19: Figure of a non-isolated cluster from 20121017, Network North. The blue nodes denotes 1-hop and 2-hop neighbors.

Once we could identify the user_agent string we could link this to an extracted MAC address. We chose to define a device as a unique MAC address for a certain type of terminal, meaning that a MAC address cannot be represented more than once from the same OS, since it is then believed to be the same type of device.

The Python code for this looks accordingly:

```
if 'iPhone' in user_agent or 'iOS' in user_agent:
  if src_mac not in macOS['iPhone']:
    macOS['iPhone'].append(src_mac)

elif 'Windows Phone' in user_agent or 'Nokia' in user_agent:
  if src_mac not in macOS['Windows Phone']:
    macOS['Windows Phone'].append(src_mac)
...
```

MAC addresses might belong to routers which could have e.g. several computers running Windows 7 behind them. Unfortunately, according to our definition of a device, all those are counted as one device. At least different devices with different OSs are detected, that is if a router represents an iPhone, a Linux computer and an Android tablet all of those will be part of the result.

Since most mobile traffic is excluded, only mobile devices connected to the access network through a wireless connection is accounted for, an accurate distribution of traffic in general can therefore not be stated, but only a distribution of devices on the access network. Most mobile devices switches from the mobile network to a wireless connection automatically when it becomes available. We still considered it interesting to see user patterns and user activity related to mobile devices on the access network, since we assume that it is almost identical to when the mobile devices are used on the mobile network. Apps and browsers are independent of the Internet connection and provide the same features for all kinds of network types. As mentioned earlier in this report, Facebook have a large number of mobile users; 600 million monthly active users in September 2012 [3][4], which indicate that a lot of Facebook traffic can be assumed to be sent over the mobile network.

We do not believe that the unidentifiable *user_ agent* strings represent a great limitation since we chose to exclude them from the total set of requests looked at. Out of 20941 unique MAC addresses found for different OSs and/or devices on Network North, the number of unknown *user_ agent* strings was 4567, still leaving a approximately 80% of the MAC address-*user_ agent* pairs to analyze. In Network South around 75% of all *user_ agent* strings were analyzable.

5.3.1 Distribution of Surfing Devices

In Table 5.9 the result shows the distribution of different surfing devices based on the MAC address-*user_ agent* pair. Almost 60% of all devices are represented by computers, which is a moderately low number considering that no mobile traffic is represented. Phones are responsible for a third of all devices found; this shows that a lot of mobile users are active on Facebook via the access network. This strengthens the argument that it is of great importance to include the use of Facebook on mobile devices while studying Facebook user patterns and user behavior.

Many tablets today have subscriptions to the mobile network but there are still some that lacks this feature. Even if the number of tablets found is low we believe that they show an accurate representation of the general use of tablets, especially since some some tablets still only work on the access networks.

The number of tablets are likely to grow over the years which is why user patterns and behavior for Facebook on tablets are interesting as well.

Figures 5.20 and 5.21 shows a more detailed description of what types of OS that has been used for different type of terminals on Network North as well as Network South.

This can be interesting to look at if e.g. content could be cached on the Terminal. Both networks have the same distribution, the dominant OS for Computers is Windows and the most popular tablet type is iPad. In case cacheing would be carried out on a terminal, it would be preferable to develop this for Windows computers first of all, since this would have the most relieving impact on the networks.

5.3.2 Number of Surfing Devices Per Unique User

In Section 5.1.4 we identify a unique user by the MAC address and Unix time stamp; a unique FBID. To calculate how many different devices that are used by a single user during the measurement period, we used the sessionizer in Section 5.1.4 in combination with the *user_ agent* string. We built Figure 5.22 based on all available data form Network South as well as data from October 17th to October 28th from Network North.

In Figure 5.22 it is seen that more than half of all unique users on both networks are active on Facebook using a single device only during this time period. This would be surprising if the results was not only showing the distribution of the access network. We believe that most people use two, three or four devices to access Facebook from; a laptop, smartphone and/or a tablet, as swell as a workstation. Since traffic from the mobile network is not included in this result, the number of single device users is probably a lot higher even if smartphone users are most likely to have used the WiFi network at some point during this time period. Table 5.10 confirms that the single device users are using Facebook from a computer and not a mobile device; 95% in Network North and 97.8% on Network South.

Figure 5.22 shows that 236 users (3.5%) on Network North have been logged on to Facebook from more than 5 devices each and only 24 (0.6%) of the users on Network South. We believe it to be reasonable that some users are logged on from more than 5 devices during a two week time period, since people sometimes borrow a device from a friend, have different devices for work and home and might have used public devices to access their Facebook accounts, increasing the number of different devices per FBID.

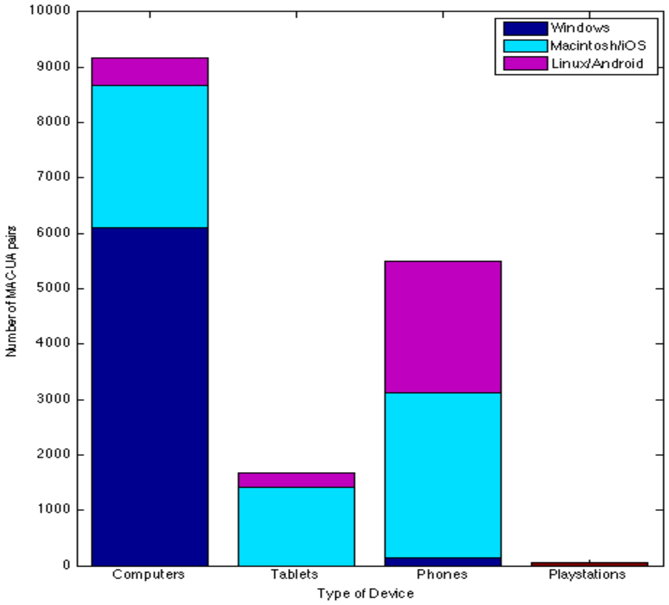

Figure 5.20: Distribution of Devices and OS's, Network North

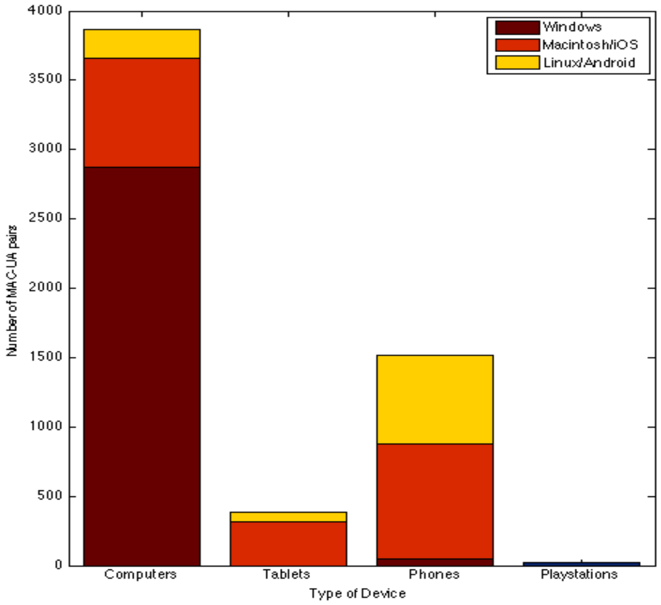

Figure 5.21: Distribution of Devices and OS's, Network South

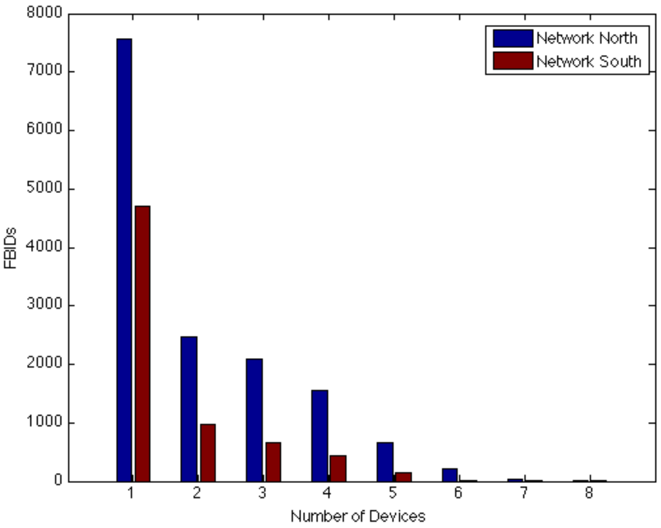

Figure 5.22: Number of devices per Facebook user

5.3.3 Terminal Activity

To study what type of behavior that was carried out on different devices as well as what kind of activities that generated the most traffic we used the *user_ agent* string, the FBID as well as the filter rules stated in Section 4.2.2.1. By integrating the Wireshark filter rules into workable Python code we were able to match part of the URL from the JSON entry to an actual Facebook activity such as Likes. We decided to count each like as a URL containing the request 'like.php', meaning that each entry with this string appearing in the URL incremented the number of Like actions. We had to extend the filter for downloading images, since a lot of images are not voluntarily downloaded, e.g. ads, preloaded photos, thumbnails etc. To separate the automatically downloaded pictures from the actual downloaded pictures we choose to neglect the pictures that were of size *100x100*, *206x206*, *480x480* and *720x720*. We also filtered out pictures that had the Facebook label *q.jpg*, *s.jpq*, *t.jpq* and *a.jpq* since these pictures almost always appeared to be thumbnails of profile pictures, which are preloaded from albums or shows up automatically on a timeline.

The URLs were filtered after the following strings:

```
'like.php'
'send_messages.php'
'updatestatus.php'
'add_comment.php'
'tagging_ajax.php'
'upload/photos'
'get_video'
```

```
'.jpg', '.gif','.tif', '.png', '!upload/photos', '!100x100',
    '!206x206',
'!480x480', '!720x720', '!q.jpg', '!s.jpq', '!t.jpq', '!a.jpq'
```

Figure 5.23 and 5.24 presents the results in logarithmic scaled diagrams over the selected Facebook activities and the number of identified FBIDs in each network. In both networks, we choose to neglect the first and last day since the measurements form these days did not include a full 24 hour period, leading to a somewhat misleading result.

The occurrences of activities is almost identical in both networks, with the exception of Tags and Videos. In both networks the popularity of activities are, in descending order, Picture Downloads, Likes, Chat, Comments, Status Updates and Picture Uploads. In Network North Tags and Videos have almost the same number of occurrences , while in Network South, there are continuously more occurrences of Videos compared to Tags, though the Videos are slightly more common.

Even with the extended filter the result show that Downloading Images is by far the most popular activity. It is also seen that Likes, Comments, and Chats are the top three most popular activities following Downloading Images.

The filter for Likes, Comments, Status Updates, Tagging and Uploading Photos are thought to be accurate. Image Downloading as well as Chats might have other detection methods as well. The *get_ video* string detects some YouTube and Vimeo clips but might not include video links to all sites. We still believe that the result showing video events is reasonable since most videos on Facebook are actually YouTube videos.

With the *user_ agent* string we could reveal the type of device that a request, containing a certain action, was sent from. Figure 5.25 shows how many percent of the Network North, divided by activities, that is carried out on a certain type of device. Figure 5.26 the result the same measurements on Network South. Both Figures, 5.25 and 5.26 show similar results, indicating that the geographical location is not a factor that influences the the popularity of Facebook activities. For both networks, for all measured activities, computers are the dominating device is use, independent of the activity type. We saw that Like and Comments were popular activities on phones, a reasonable result since the Facebook phone apps are designed to provide easy access to the like- and comment-button, which we can see clearly pays off in the like case. In the figures it is also seen that Videos are much more popular to watch on tablets than on phones as well as Uploading Photos and Tagging friends. We believe that these three actions are more popular on tablets than phones, since tablets provide a larger screen in comparison.

One misleading result might be the percent of chat messages sent from phones. We must admit that we could not establish how messages sent form the Messenger Facebook App appeared in the URL, which therefore is not part of this analysis. The messages we have detected might only be messages sent from the regular Facebook site, resulting in incorrect numbers. This, along with the lack of mobile traffic recordings, could show a heavily inaccurate result of chat messages sent from mobile devices in comparison to computers.

Table 5.11 shows the *Pearson Correlation Coefficient* between activities and users. It is seen that the correlation coefficient is close to 1 for all activities, meaning that when a certain activity increases; becomes more popular, all other activities increases as well. It is also seen that the number of users is not positive correlated to any activity, meaning that when the number of users becomes greater the number of activities carried out decreases, and vice versa. This could seem surprising, but we believe this to be the result of inactive users. When a lot of users are logged on they might just stay on Facebook without actually execute any actions. When less users are online they seem to use Facebook more actively. For example a lot of users might be logged on during working hours, being online at their workstation, but not actively using the site. Users that log on at night or on their free day are more likely to be online for the purpose to use Facebook.

In Table 5.12 the p-Value for the *Pearson Correlation Coefficient* is seen. The *p-value* is very low for all entries, ($p < 0.009$), which indicates that our correlation is statistically significant and that the sample size is relevant.

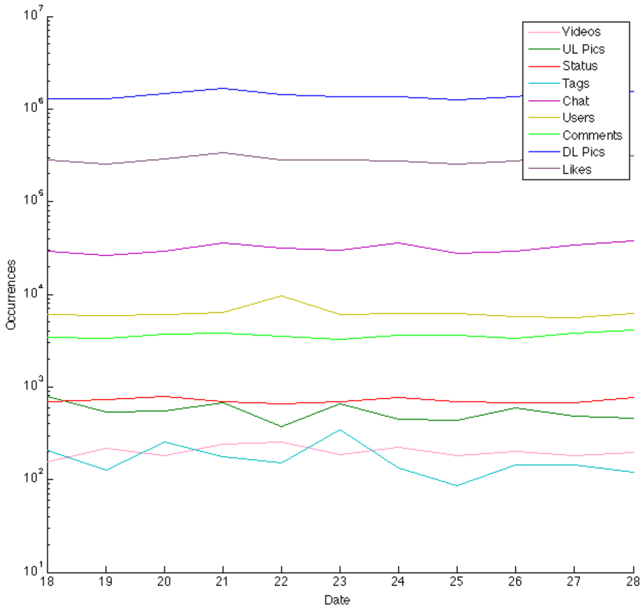

Figure 5.23: Logarithmic Scale of Facebook Events, Network North

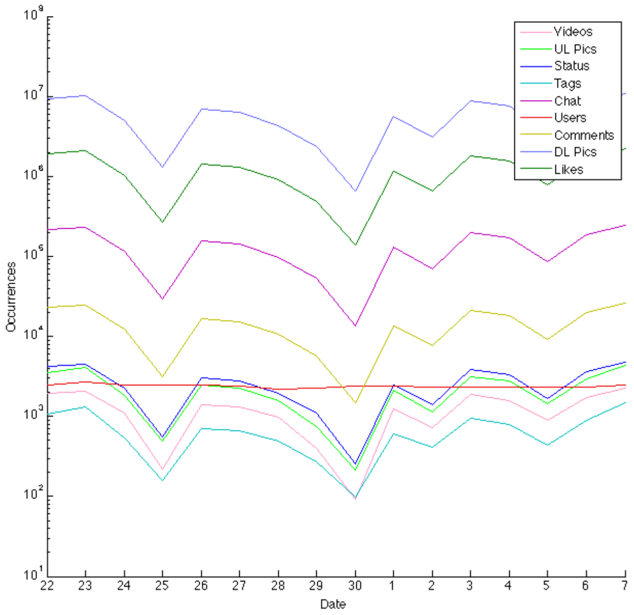

Figure 5.24: Logarithmic Scale of Facebook Events, Network South

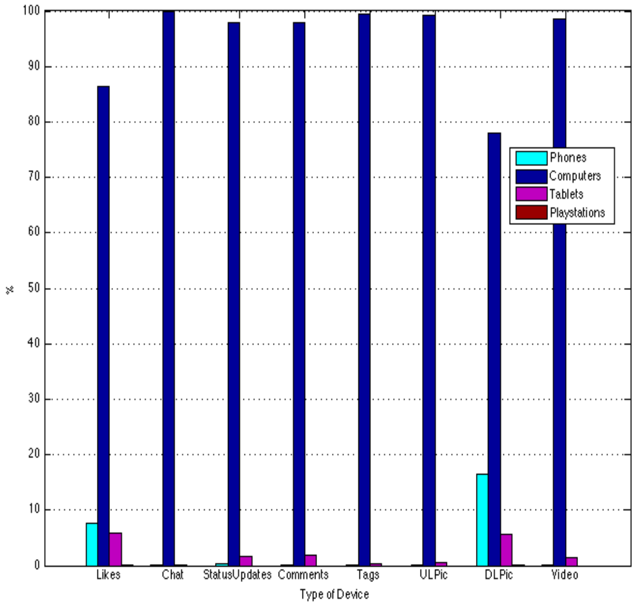

Figure 5.25: Device Distribution of Facebook Events, Network North

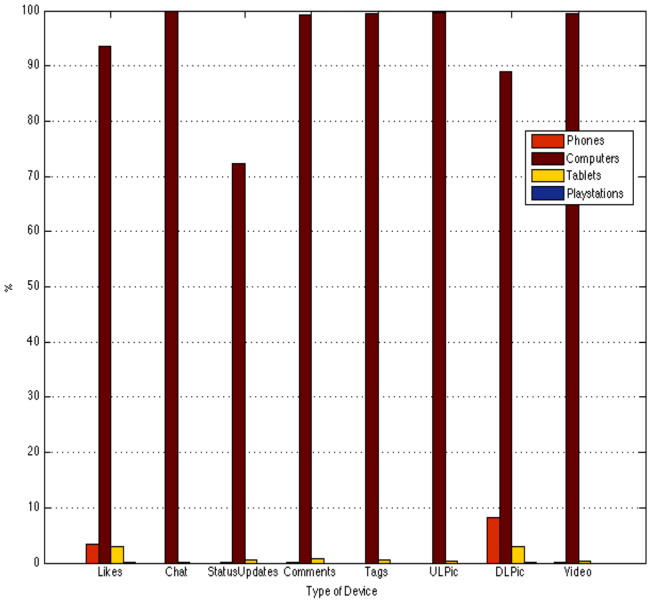

Figure 5.26: Device Distribution of Facebook Events, Network South

Table 5.5: Trust Graph Statistics when using $\alpha = 0$, $\beta = 0.5$, $\gamma = 0.5$, $\sigma = 0.3$. Let T = trust(u, v), from Network North.

	$T \geq 0.4$		$T \geq 0.5$		$T \geq 0.65$	
Day	$\|V\|$	$\|E\|$	$\|V\|$	$\|E\|$	$\|V\|$	$\|E\|$
2012-10-17	3870	31233	2902	5662	4	2
2012-10-18	3815	35418	3236	6383	15	8
2012-10-19	3690	33325	3075	5883	8	4
2012-10-20	3650	30735	3012	5532	6	3
2012-10-21	3997	36950	3361	6524	2	1
2012-10-22	5841	44446	4639	7214	8	4
2012-10-23	3825	37179	3257	6578	19	11
2012-10-24	3902	39261	3312	6937	14	7
2012-10-25	3858	35139	3221	6006	10	5
2012-10-26	3689	30260	3040	5794	12	6
2012-10-27	3540	33548	2971	5974	4	2
2012-10-28	3935	38455	3349	6685	4	2
2012-10-29	3886	40159	3289	6690	10	5

Table 5.6: Trust Graph Statistics when using $\alpha = 0$, $\beta = 0.5$, $\gamma = 0.5$, $\sigma = 0.3$. Let T = trust(u, v), from Network South.

	$T \geq 0.4$		$T \geq 0.5$		$T \geq 0.65$	
Day	$\|V\|$	$\|E\|$	$\|V\|$	$\|E\|$	$\|V\|$	$\|E\|$
2012-09-21	1319	23290	1162	4246	2	2
2012-09-22	1482	21078	1201	3694	2	2
2012-09-23	1651	22972	1342	4096	2	2
2012-09-24	1547	18168	1210	3438	8	8
2012-09-25	1550	28084	1345	5048	8	8
2012-09-26	1535	26314	1300	4612	4	4
20120-9-27	1513	26530	1303	4750	4	4
2012-09-28	1405	19660	1134	3592	8	8
2012-09-29	1410	22878	1191	3942	8	8
2012-09-30	1575	28230	1365	4948	0	0
2012-10-01	1539	26708	1325	4952	6	6
2012-10-02	1446	28460	1263	5082	0	0
2012-10-03	1458	26944	1260	4974	2	2
2012-10-04	1437	18384	1151	3448	0	0
2012-10-05	1446	25826	1252	4568	8	8
2012-10-06	1446	24688	1353	4732	2	2
2012-10-07	1571	27044	683	2778	4	4
2012-10-08	780	11088	683	2959	8	8

Table 5.7: Average Clustering Coefficients, Network North

Day	Average clustering coefficient
20121017	0.0176716677147
20121018	0.0122332406846
20121019	0.0141997263262
20121020	0.0167462374604
20121021	0.0123920695641
20121022	0.012166439696
20121023	0.0127065393675
20121024	0.012108346019
20121025	0.0116133678433
20121026	0.0230692268855
20121027	0.0132432652678
20121028	0.0114669005006
20121029	0.00578950318363

Table 5.8: Average Clustering Coefficients, Network South

Day	Average clustering coefficient
20120921	0.0103634959142
20120922	0.0196565158178
20120923	0.00975592680023
20120924	0.0122899867174
20120925	0.0122823297958
20120926	0.0154180648425
20120927	0.0186545500203
20120928	0.0232203928412
20120929	0.0160502382481
20120930	0.0167432777631
20121001	0.0183633943974
20121002	0.00718683826798
20121003	0.0134091805593
20121004	0.0292057975811
20121005	0.00925253542794
20121007	0.0125355105889
20121008	0.0135478652854

Table 5.9: Distribution of Surfing Devices

Device	Network North	Network South
Computers	**56.0**%	**66.9**%
Windows	66.5%	74.2%
Mac OS X	26.2%	20.3%
Linux	5.3%	5.5%
Phones	**33.5**%	**26.2**%
Windows Phone	2.4%	3.4%
iPhone	54.6%	54.5%
Android	43.0%	42.1%
Tablets	**10.3**%	**6.6**%
iPad	84.4%	82.0 %
Tablet	15.6%	18.0%
Playstations	**0.3**%	**0.3**%
Total	**100**%	**100**%

Table 5.10: Single Device Users on the Access Network

Device	Unique FBIDs Network North	Unique FBIDs Network North
Computers	6372	4198
Phones	224	54
Tablets	114	41
PS	0	0

Table 5.11: Pearson Correlation Coefficients Matrix for Facebook Activities

	Users	Comments	UL Pics	Tags	Videos	Likes	Chat	DL Pics	Status
Users	1.0000	-0.6289	-0.6150	-0.5996	-0.6626	-0.6385	-0.6342	-0.6303	-0.6210
Comments	-0.6289	1.0000	0.9938	0.9790	0.9965	0.9996	0.9997	0.9996	0.9996
UL Pics	-0.6150	0.9938	1.0000	0.9908	0.9890	0.9947	0.9933	0.9942	0.9937
Tags	-0.5996	0.9790	0.9908	1.0000	0.9741	0.9812	0.9784	0.9803	0.9792
Videos	-0.6626	0.9965	0.9890	0.9741	1.0000	0.9972	0.9968	0.9959	0.9953
Likes	-0.6385	0.9996	0.9947	0.9812	0.9972	1.0000	0.9997	0.9997	0.9988
Chat	-0.6342	0.9997	0.9933	0.9784	0.9968	0.9997	1.0000	0.9996	0.9990
DL Pics	-0.6303	0.9996	0.9942	0.9803	0.9959	0.9997	0.9996	1.0000	0.9991
Status	-0.6210	0.9996	0.9937	0.9792	0.9953	0.9988	0.9990	0.9991	1.0000

Table 5.12: p-value Matrix for the Pearson Correlation Coefficients in Table 5.11

	Users	Comments	UL Pics	Tags	Videos	Likes	Chat	DL Pics	Status
Users	1.0000	0.0004	0.0006	0.0009	0.0002	0.0003	0.0004	0.0004	0.0005
Comments	0.0004	1.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
UL Pics	0.0006	0.0000	1.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
Tags	0.0009	0.0000	0.0000	1.0000	0.0000	0.0000	0.0000	0.0000	0.0000
Videos	0.0002	0.0000	0.0000	0.0000	1.0000	0.0000	0.0000	0.0000	0.0000
Likes	0.0003	0.0000	0.0000	0.0000	0.0000	1.0000	0.0000	0.0000	0.0000
Chat	0.0004	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000	0.0000	0.0000
DL Pics	0.0004	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000	0.0000
Status	0.0005	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000

5.3.4 Day Distribution

We use the Unix time stamp from the JSON files to determine the time of day when a request was sent, which is seen in Figure 5.27. It is an almost even distribution throughout the whole day, the hours between noon and midnight are a bit more intense compared to the night and morning. This result is almost identical for both networks. To neglect requests sent by automatic updates, we filtered traffic was by the events: Like, Chats and Comments and chose to only look at the time for these activities. Since these three activities are popular, we believe that they generate an accurate distribution of when users are actually active on Facebook. The *user_ agent* string in combination with the time stamp and activity detection generated the results shown in Figure 5.28 for Network North and 5.29 for Network South. The results shows that the distribution of devices on both networks are very similar. Independent of the time of day, most Facebook traffic is generated by computers, even more so on Network North which has approximately 10% heavier use of computers at all times compared to Network South.

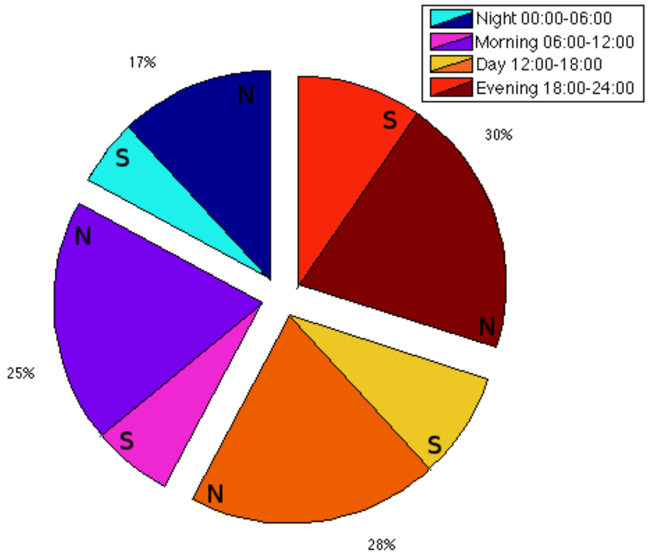

Figure 5.27: Day distribution of Facebook activity

5.4 Devices and Clusters of Users

In order for us to examine whether any conclusions about how and if devices are used in a different fashion between distinct groups of users can be drawn all clustered users were paired up with the devices they have been shown to utilize when browsing Facebook. Table 5.13 contains a row for each cluster and a column for each type of device. We then calculated the *Pearson Correlation Coefficient matrix*, see Table 5.14, and the *p-value matrix*, see Table 5.15, for the four data

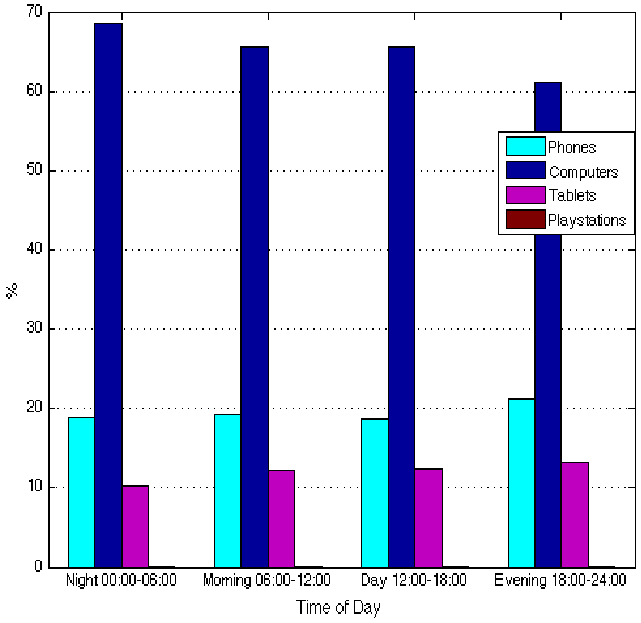

Figure 5.28: Device distribution during the day, Network North

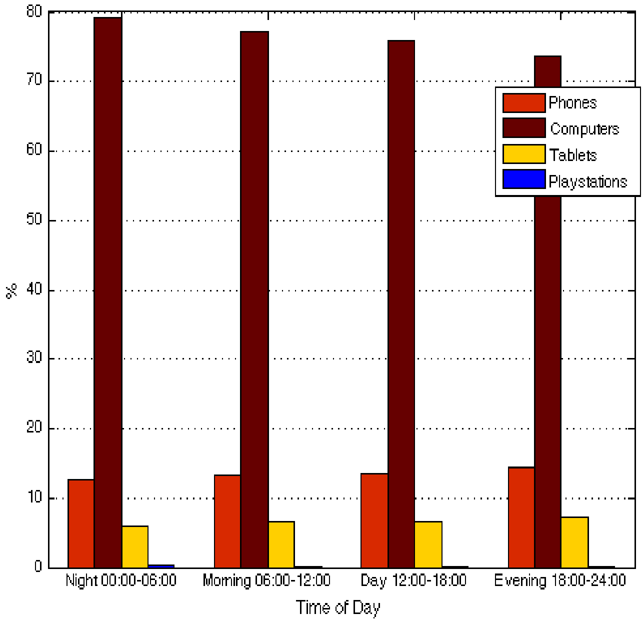

Figure 5.29: Device distribution during the day, Network South

vectors containing the device usage distribution, again using the definitions in equations 5.4 and 5.5. We can see some very interesting results in Tables 5.14 and 5.15. The most interesting result is the strong negative correlation of -0.8356 (with $p < 0.00005$) between the usage of computers and phones while accessing Facebook; users seems to prefer to use a computer *or* a phone. Another interesting observation of ours is that there seem to be little correlation between usage of (computer, tablet), (computer, playstation) and (tablet, phone) with *Pearson Correlation Coefficients* $\|C\| < 0.35$ and $p \leq 0.0004$. The remaining device pairs have a confidence interval too low for any conclusions to be drawn.

Table 5.13: An excerpt from the distribution of devices used inside trust clusters.

Cluster	Computer	Phone	Tablet	Playstation
C_1	85.71%	7.14%	0.0%	7.146%
C_2	66.67%	0.0%	33.33%	0.0%
C_3	88.89 %	0.0%	11.11%	0.0%
C_4	60.0%	33.33%	0.0%	6.67%
C_5	70.59%	23.53%	0.0%	5.88%

We did the same calculations on an individual user level, see Tables 5.16 and 5.17. We can conclude that even though we seem to have a lack of correlation, as opposed to the correlations for device-usage in clusters, we cannot be sure of this because no correlation coefficient is in a 95% confidence interval.

Table 5.14: Pearson Correlation Coefficients Matrix for Devices Used Inside Trust Clusters.

Network North

Devices	Computer	Phone	Tablet	Playstation
Computer	1.0000	-0.8356	-0.3030	-0.1465
Phone	-0.8356	1.0000	-0.0597	-0.0010
Tablet	-0.3030	-0.0597	1.0000	-0.0058
Playstation	-0.1465	-0.0010	-0.0058	1.0000

Network South

Devices	Computer	Phone	Tablet	Playstation
Computer	1.0000	-0.7652	-0.1794	-0.0972
Phone	-0.7652	1.0000	-0.0761	-0.0343
Tablet	-0.1794	-0.0761	1.0000	-0.0136
Playstation	-0.0972	-0.0343	-0.0136	1.0000

Table 5.15: p-value Matrix for the Pearson Correlation Coefficients in Table 5.14.

Network North

Devices	Computer	Phone	Tablet	Playstation
Computer	1.0000	0	0.0000	0.0000
Phone	0	1.0000	0.0004	0.9526
Tablet	0.0000	0.0004	1.0000	0.7328
Playstation	0.0000	0.9526	0.7328	1.0000

Network South

Devices	Computer	Phone	Tablet	Playstation
Computer	1.0000	0	0.0000	0.0000
Phone	0	1.0000	0.0010	0.1381
Tablet	0.0000	0.0010	1.0000	0.5572
Playstation	0.0000	0.1381	0.5572	1.0000

Table 5.16: Pearson Correlation Coefficients Matrix for Devices Used by Individual Users.

Network North

Devices	Computer	Phone	Tablet	Playstation
Computer	1.0000	0.0149	0.0056	0.0021
Phone	0.0149	1.0000	0.0592	0.0597
Tablet	0.0056	0.0592	1.0000	0.0054
Playstation	0.0021	0.0597	0.0054	1.0000

Network South

Devices	Computer	Phone	Tablet	Playstation
Computer	1.0000	-0.0222	0.0014	-0.1660
Phone	-0.0222	1.0000	0.0234	0.0027
Tablet	0.0014	0.0234	1.0000	-0.0029
Playstation	-0.1660	0.0027	-0.0029	1.0000

Table 5.17: p-value Matrix for the Pearson Correlation Coefficients in Table 5.16.

Network North

Devices	Computer	Phone	Tablet	Playstation
Computer	1.0000	0.0725	0.5000	0.8010
Phone	0.0725	1.0000	0.0000	0.0000
Tablet	0.5000	0.0000	1.0000	0.5103
Playstation	0.8010	0.0000	0.5103	1.0000

Network South

Devices	Computer	Phone	Tablet	Playstation
Computer	1.0000	0.0653	0.9045	0.0000
Phone	0.0653	1.0000	0.0520	0.8221
Tablet	0.9045	0.0520	1.0000	0.8103
Playstation	0.0000	0.8221	0.8103	1.0000

Chapter 6

Discussion and Future Work

In this thesis we seek to find Facebook user behaviour patterns and build community clusters based upon content demand analysis. The work is naturally grouped into four parts:

1. creation & evaluation of a method for identification of users,
2. creation & analysis of community cluster graphs,
3. measuring & analyzing device usage & user activities,
4. correlating device usage in community clusters.

When trying to understand user patterns it is important to understand what a user is and how to successfully identify one. We have analyzed three methods of identification: identification by MAC address as described in [48], identification through IP address and identification through `<unix-time, mac>`-tuples. We show that identification through MAC- and IP address are highly unreliably since each of these addresses corresponds to, on average, > 3 unique FBIDs. We present the model and pseudo-code for constructing identification tables for mapping of a FBID to a `<unix-time, mac>`-tuple. We also show that identification using these tables lets us identify around 88% of all requests made, with zero possibility for false negatives, a clear improvement on earlier work done on this problem. The fact that we can identify around 88% of all image requests with our method suggests that larger improvements are likely to be possible. One way to do this would be to utilize the micro second parameter in the packets in the dump to increase the resolution (i.e. more fine grained sessions). This would likely reduce the dead time because the chance of conflicting content demands would decrease. This due to that the chance for two users having a content demand conflict during a one second span is greater than the chance of them having one during a micro second span. It is possible that the identification success rate could rise significantly if this optimization was implemented.

We present a method for building bipartite user-content graphs by analyzing content demands (images in this case) by users in the packet dump data. We also show how to use the Trust-function described in [40] to translate the bipartite graph into a graph containing only users, no content, with edges denoting the Trust between these users. TheTrust have been derived only from similarities in the users content demand patterns, and not from any explicitly expressed Trust while distrust between users is like friendship in [51]. We show extensive data regarding different trust graphs (with different Trust thresholds) and we see that the size of the trust graphs decrease rapidly when the threshold increases. We see that clustering coefficients for nodes in clustered graphs is greater than in the case of un-clustered graphs but that the coefficient is non-correlated to the cluster size. A very important part, that we have not fully explored, is the Trust-function; what would happen if better parameter configuration could be obtained? Other questions regarding Trust that could be answered is: ”what is a good parameter configuration” and is there a better Trust-function than the one we have used? It would also be of interest to look at more content types when building the bipartite graphs; Likes, video demands, viewed pages and so on. It could also be explored if different activities should be rated; maybe one Like says more about similarity than one image demanded.

We measure the different types of activities on the two measured networks and conclude that there seem to be no difference between these two. The lack of difference also applies for the distribution of used devices in the two networks. However, the activities are strongly correlated to each other, e.g. if the amount of Likes is high one day, the amount of image demands will be high as well. We see that what kind of device a user utilizes effects the users behavior, e.g. users on tablets watch more videos and users on phones do more Likes on content. When looking at what time users are active, we filter out activity that can be caused automatically and focus on activity that requires user interaction. This lets us see that an activity peak exists between 06:00 pm and 12:00 pm.

We measure all the types of devices a unique user utilizes by mapping an interpretation of the user-agent parameters in the packet dumps and calculates statistics of the device distribution inside the community clusters. We also calculate the correlation coefficients and the confidence intervals for these statistics and find an interesting observation: community clusters where computer-usage is dominant for Facebook browsing tend to have a *low* usage of cellphones and vice versa. When performing the same analysis on an individual user lever, instead of on a cluster level, we see that no certain conclusions can be drawn about individual usage patterns because of the low confidence intervals. We suspect that the (tablet, computer)- and (tablet, phone)-pairs are uncorrelated as these results almost are within the 95% confidence interval. This information about device usage could be used to optimize network infrastructure based on what type of devices that dominates the traffic generation on the actual network. It is also possible that it can be used when designing different services that users connect to, e.g. allocation of bandwidth for video streaming when a user connects using a tablet.

In the future it would be interesting to examine different ways to construct the community cluster graphs in real time. This would let the graphs represent the current state of the networks, instead of the states at the time of the measurement. It would also be interesting to create a method for prediction of popular content based on community clusters. These two could perhaps even be combined to an efficient prediction-algorithm that predicts popularity based on content demand patterns, i.e. prediction based on what currently is popular on the internet.

References

[1] *Facebook: About*, Facebook Inc., https://www.facebook.com/facebook/info, retrieved October 15, 2012.

[2] S. Phillips, *A brief history of Facebook* The Guardian, July 25, 2007, http://www.guardian.co.uk/technology/2007/jul/25/media.newmedia, retrieved October 15, 2012.

[3] *Facebook: Newsroom*, Facebook Inc., http://newsroom.fb.com/ retrieved October 15, 2012.

[4] *The New York Times: Mark Zuckerberg*, The New York Times, http://topics.nytimes.com/topics/reference/timestopics/people/z/mark_e_zuckerberg/index.html, retrieved October 15, 2012.

[5] *Facebook: Advertising on Facebook*, Facebook Inc., https://www.facebook.com/about/ads/#stories, retrieved October 15, 2012.

[6] J. Pepitone, *Facebook traffic tops Google for the week*, CNN Money, March 16, 2010, http://money.cnn.com/2010/03/16/technology/facebook_most_visited/, retrieved October 15, 2012.

[7] H.Zhao *HipHop for PHP: Move Fast*, Facebook Developer Blog, February 2, 2010, https://developers.facebook.com/blog/post/2010/02/02/hiphop-for-php–move-fast/, retrieved October 15, 2012.

[8] *Facebook Hive: About*, Facebook Inc., https://www.facebook.com/apache.hive/info, retrieved October 15, 2010.

[9] J. Sen Sarma, *Facebook: Hadoop*, Facebook Inc., June 4, 2008, https://www.facebook.com/note.php?note_id=16121578919, retrieved October 15, 2012.

[10] *Welcome to Apache Hadoops!*, The Apache Software Foundation, October 17, 2012, https://hadoop.apache.org/, retrieved October 18, 2012.

[11] *Facebook Developers: Open Source*, Facebook Inc., https://developers.facebook.com/opensource/, retrieved October 18, 2012.

[12] D. Beaver, S. Kumar, H. C. Li, J. Sobel, P. Vajgel, *Finding a needle in Haystack: Facebook's photo storage* in Proceedings of the 9th USENIX conference on Operating systems design and implementation, ACM, Berkeley, CA, USA, 2010.

[13] B. Darwell, *Facebook platform supports more than 42 million pages and 9 million apps*, WebMediaBrands Inc: Inside Facebook, April 27, 2012 http://www.insidefacebook.com/2012/04/27/facebook-platform-supports-more-than-42-million-pages-and-9-million-apps/, retrieved October 18, 2012.

[14] *Facebook: Prineville Data Center*, Facebook Inc., https://www.facebook.com/prinevilleDataCenter/info, retrieved October 18, 2012.

[15] *Facebook: Forest City Data Center*, Facebook Inc., https://www.facebook.com/ForestCityDataCenter, retrieved October 18, 2012.

[16] *Facebook: Lulea Data Center*, Facebook Inc., https://www.facebook.com/luleaDataCenter, retrieved October 18, 2012.

[17] *History of the Internet*, New Media Institute, http://www.newmedia.org/history-of-the-internet.html, retrieved October 22, 2012.

[18] S. D. Crocker, *How the Internet Got Its Rules*, The New York Times, (April 7, 2009), sec. A p.29.

[19] J. Ryan, September 15, 2010, *A History of the Internet and the Digital Future*, Reaktion Books.

[20] *Facebook Statistics: Sweden*, Social Bakers, https://www.socialbakers.com/facebook-statistics/sweden, retrieved October 22, 2012.

[21] M. Ward, *Celebrating 40 years of the net*, BBC News, October 29, 2009, http://news.bbc.co.uk/2/hi/technology/8331253.stm, retrieved October 22, 2012.

[22] J. Li, A. Aurelius, V. Nordell, M. Du, Å. Arvidsson, M. Kihl, *A five year perspective of traffic pattern evolution in a residential broadband access network* in Future Network & Mobile Summit, IEEE, Berlin, Germany, 2012.

[23] A. Aurelius, Å. Arvidsson, M. Johansson, M. Kihl, C. lagerstedt, *Leveraging network and traffic measurements for content distribution and interpersonal communication services with sufficient quality* in 13th International Conference on Transparent Optical Networks, Stockholm, Sweden, 2011.

[24] A. Aurelius, C. Lagerstedt, I. Sedano, S. Molnar, M. Kihl, F. Mata, *Monitoring the evolution of residential broadband Internet traffic* in Future Network & Mobile Summit, IEEE, Florens, Italy, 2010.

[25] M. Kihl, C. Lagerstedt, A. Aurelius, *Traffic analysis and characterization of Internet user behavior* inU ltra Modern Telecommunications and Control Systems and Workshops, IEEE, Moscow, Russia, 2010.

[26] White Paper on *Cisco Visual Networking Index: Forecast and Methodology, 2010-2015*, Cisco, June 2011.

[27] *IP Network Monitoring for Quality of Service Intelligent Support*, Eureka, http://projects.celtic-initiative.org/ipnqsis/, retrieved on December 8, 2012.

[28] *15 Gbps Intelligent Policy Enforcement for Broadband Networks*, Procera Networks, http://www.proceranetworks.com/pdf/products/pre/PRE_PL8720_Q3_-2012_7_7_WEB.pdf, retrieved on December 8, 2012

[29] *Facebook: help center*, Facebook Inc., https://www.facebook.com/help/218673814818907/, retrieved on January 22, 2013.

[30] *Python: about*, Python Foundation, http://www.python.org/about/, retrieved january 22, 2013.

[31] *NetworkX: Overview*, NetworkX Developers, http://networkx.github.com/documentation/latest/overview.html, retrieved January 11, 2013.

[32] *SciPy: History*, SciPy Developers, http://www.scipy.org/History_of_SciPy, retrieved January 22, 2013.

[33] B. Darwell, *Facebook Mobile Users*, WebMediaBrands Inc., March 7, 2012, http://www.insidefacebook.com/2012/03/07/facebook-says-it-had-432m-mobile-users-in-december-2011-13-percent-are-mobile-only/, retrieved November 14, 2012.

[34] T. Smith, *More than half of Facebook users access site through Mobile*, Business Computer Review, March 8, 2012, http://www.cbronline.com/news/more-than-half-of-facebook-users-access-site-through-mobile-080312, retrieved November 15, 2012.

[35] A. Chang, *Report: Smart phones, not computers, drive most Facebook use*, CNN, March 9, 2012, http://edition.cnn.com/2012/05/08/tech/social-media/facebook-mobile-report/index.html, retrieved November 15, 2012.

[36] *.PCAP File Extension*, Disqus, September 20, 2012, http://www.fileinfo.com/extension/pcap, retrieved November 26, 2012.

[37] *.JSON File Extension*, Disqus, September 20, 2012, http://www.fileinfo.com/extension/json, retrieved November 26, 2012.

[38] A. Rice, *HTTPS: A Continued Commitment to Security*, Disqus, January 26, 2011, https://www.facebook.com/blog/blog.php?post=486790652130, retrieved November 26, 2012.

[39] R. A. Hanneman, *Introduction to social network methods*, University of California, Riverside, http://faculty.ucr.edu/~hanneman/nettext/C18_Statistics.html#TOC, retrieved January 27, 2012.

[40] D. O'Doherty, S. Jouili, P. Van Roy, *Towards trust inference from bipartite social networks* in DBSocial 12, ACM, Scottsdale, USA, 2012.

[41] R. Oldenburg, 1998 *The Great Good Place: Cafes, Coffee Shops, Bookstores, Bars, Hair Salons, and Other Hangouts at the Heart of a Community*, Marlowe & Co.

[42] A. J. Kim, 2000, *Community Building on the Web*, Peachpit Press.

[43] *Ipred*, The European Commission, http://www.ipred.org/ retrieved November 22, 2012

[44] E.N. Sawardecker, C.A. Amundsen, M. Sales-Pardo, L.A.N. Amaral , *Comparison of methods for the detection of node group membership in bipartite networks*, The European Physical Journal, vol. 72, pp. 671-677, November, 2009.

[45] P. Zhang, J Wang, M. Li, Y. Fan, *Clustering coefficient and community structure of bipartite networks*, Physica A, vol. 387, pp. 6869-6875, September, 2008.

[46] J. Sima, S.E. Schaeffer , *On the NP-Completeness of Some Graph Cluster Measures* in SofSem 06, LNCS, Merin, Czech Republic, 2006.

[47] C. Biemann, 2012, *Theory and Applications of Natural Language Processing*, Springer.

[48] S. Ibern, F. Soler, *Facebook Traffic Data and cacheability*, M.Sc., Lund University, Sweden, 2012.

[49] J. Kleinberg, E. Tardos, 2005, *Algorithm Design*, Pearson.

[50] D. J. Watts, Steven Strogatz, *Collective dynamics of 'small-world' networks*, Nature, Vol. 393, pp. 440-442, June, 1998.

[51] S. Catanese, P. De Meo, E. Ferrara, G. Fiumara, A. Provetti, *Crawling Facebook for social network analysis purposes* in International Conference on Web Intelligence, Mining and Semantics, ACM, New York, USA, 2011.

[52] G. Blom, J. Enger, G. Englund, J. Grandell, L. Holst, 2005, Sannolikhetsteori och statistikteori med tillämpningar, Studentlitteratur.

[53] *Matlab help: corrcoeff*, Mathworks, http://www.mathworks.se/help/matlab/ref/corrcoef.html, retrieved January 9, 2013.

[54] *Matlab help: cov*, Mathworks, http://www.mathworks.se/help/matlab/ref/cov.html retrieved January 9, 2013