Analyzing Caching Gain in Small Geographical Areas in IP Access Networks

MANXING DU

KTH Information and
Communication Technology

Degree project in
Communication Systems
Second level, 30.0 HEC
Stockholm, Sweden

Analyzing Caching Gain in Small Geographical Areas in IP Access Networks

Manxing Du

manxing@kth.se

2012-10-29

Academic supervisor: G. Q. Maguire Jr.
Examiner: G. Q. Maguire Jr.
Industrial supervisor: Andreas Aurelius, Acreo AB

Communication Systems
School of Information and Communication Technology
KTH Royal Institute of Technology
Stockholm, Sweden

Abstract

Since its emergence, user generated content (UGC) has become the driving force in the growth of Internet traffic. As one of the most successful and popular UGC systems, YouTube contributes a great share of Internet traffic volume and has attracted a lot of academic interest. The continuously increasing amount of IP traffic motivates the need for better network design, more efficient content distribution mechanisms, and more sustainable system development. Web caching is one of the widely used techniques to reduce the inter Internet Service Provider (ISP) traffic. Web caching is considered an important part in the design of a content distribution infrastructure.

This master's thesis utilizes a one month trace of YouTube traffic in two residential networks in Sweden. Based upon a systematic and in-depth measurement we focus on analyzing the geographic locality of traffic patterns within small areas for these two networks. We summarize the YouTube traffic characteristics and user replay patterns, and then discuss why caching can be useful for YouTube-like systems.

We present the optimal caching gain on a per area basis and also divide users into two groups: PC and mobile device users to show the caching gain for these two groups. Overall, an infinite capacity proxy cache for each small area could reduce the YouTube streaming data traffic by 30% to 45%. The result presented in this paper help us to understand YouTube traffic and user behaviors and provides valuable information for the ISPs to enable them to design more efficient caching mechanisms.

When this work began we thought that a reduction of backhaul traffic (especially for mobile operators) may delay the need to make investments in upgrading their network capacity. However, an important conclusion from this thesis project is that the cache efficiency depends on the terminal type. For mobile terminals (smart phones, iPads, etc) a terminal cache solution is found to be the most efficient. For PCs in fixed networks, a network cache would be more efficient. It should be noted that the mobile terminals covered in the project are connected through home Wi-Fi, so further research is needed in order to draw definite conclusions for caching solutions for cellular networks.

Keywords: *UGC, YouTube, Geographic locality, Caching gain*

Sammanfattning

Sedan dess tillkomst har användargenererat innehåll (på engelska: User Generated Content UGC) blivit den drivande kraften bakom ökningen av internettrafiken. Ett av de mest använda och populära UGC-systemen är Youtube, som bidrar med en stor del av volymen i internettrafiken, och har på så sätt lockat till sig ett stort akademiskt intresse. Den konstant ökande mängden av IP-trafik motiverar behovet av bättre nätverksdesign, effektivare mekanismer för delning av data, och en mer långsiktig system utveckling. Mellanlagring i nätet (network caching) är en av de mer använda teknikerna för att reducera trafiken för Internetoperatörer. Mellanlagring i nätet anses vara en viktig del i designen av den framtida media-distributionens infrastruktur.

Det här examensarbetet använder en månads data från Youtube-trafik i två lokala nätverk i Sverige. Baserat på en systematisk och detaljerad mätning, fokuserar vi på att analysera specifika trafikmönster geografisk små områden för dessa två nätverk. Vi analyserar Youtube-trafikens egenskaper och karakteristik och användarnas beteende mönster. Baserat på dessa analyserar vi om mellanlagring kan vara en nyttig lösning för att reducera trafiken för Youtube-liknande system.

Vi presenterar den optimala lagringsvinsten (cache gain) för geografiskt definierade populationer och vi delar även upp användare i två grupper: PC och mobila enheter, för att visa lagringsvinsten individuellt för dessa grupper. Generellt sett, om man hade en oändlig lagringskapacitet hos en proxy cache inom ett visst område, så skulle man kunna reducera Youtube-trafiken med 30-45%. Resultaten som presenterats i detta dokument, hjälper oss att förstå Youtube trafik och användar beteende, och ger värdefull information till operatörer, så att de kan designa effektivare lagringsmekanismer.

Some utgångspunkt för detta arbete antog vi att en minskning av backhaultrafiken (särskilt för mobiloperatörer) kan fördröja behovet av att göra investeringar för att uppgradera kapaciteten i nätet. En viktig slutsats av detta examensarbete är att effektiviteten hos en proxy cache beror av terminaltypen. För mobila terminaler (smarta telefoner, iPads, etc) ger terminal-cache högre effektivitet, medan en nätverks-cache är effektivare för PCs. Det bör dock nämnas att mätningarna i detta arbete är från terminaler uppkopplade via fast bredband. Det behövs vidare analys för att dra konkreta slutsatser för användarbeteende och cache-lösningar i mobilnät.

Keywords: *UGC, YouTube, geografiskt definierade populationer, mellanlagring*

Acknowledgements

First of all, I would like to show my acknowledgement to the best examiner and supervisor Prof. Gerald Q. Maguire Jr. for guiding me and inspiring me to carry out this thesis project. I cannot be more grateful for his detailed, precise and elaborate feedbacks and suggestions which were vital for me to complete and improve this work.

Furthermore, I would like to express my gratitude to my industrial advisor Andreas Aurelius for his kind help and guidance from the very beginning till the end of the thesis work. He is the best advisor one can ever imagine. I enjoyed every single discussion with him and I am very much appreciated for his valuable advices, constant support and encouragement throughout the whole project.

In addition, I must show my special appreciation to Prof. Åke Arvidsson from Ericsson who showed continuous interest in my thesis work. I want to thank him for being always willingly to share his knowledge and contributing a lot to this work by having frequent discussions with me and providing me many new ideas. I also want to thank Jie Li, Viktor Nordell and all the people from Acreo who helped me during this entire period.

Finally, I would like to thank all the people, especially my family who are always by my side, support me and encourage me no matter what.

Manxing Du

Stockholm, 17th Oct 2012

Table of contents

List of Figures

List of Tables

List of Acronyms and Abbreviations

API	Application Programming Interface
BSC	Base Station Controllers
CAPEX	Capital expenses
CDN	Content Delivery Networks
DRDL	Datastream Recognition Definition Language
DSL	Digital Subscriber Line
FTTH	Fiber To The Home
Gbps	Gigabits per second
GGSN	Gateway GPRS Support Node
ISP	Internet service provider
Mbps	Megabits per second
OPEX	Operating expenses
P2P	Peer-to-Peer
RNC	Radio Network Controllers
RTT	Round trip time
SGSN	Serving GPRS Support Nodes
SON	Semantic Overlay Network
TM	Telecommunications Management
TV	Television
UGC	User Generated Content
URL	Uniform Resource Locator
VoD	Video on Demand

1 Introduction

Cisco's 2011 global IP annual forecast report revealed that IP traffic had increased eightfold over the past 5 years [1]. Traffic growth from non-PC devices, such as smart phones, tablets, and televisions (TVs) is expected to double by 2015. This same report also showed that in 2010, for the first time in a decade, Internet video (Peer-to-Peer streaming was included) traffic exceeded Peer-to-Peer (P2P) file sharing traffic, becoming the dominant type of Internet traffic[1]. Adoption of live streaming and media-on-demand services, which are the major variants of Internet video, increased rapidly during recent years and both have attracted tremendous numbers of users. According to the statistics provided by Alexa, YouTube is not only one of the most popular video-on-demand websites, but it ranks third in *total traffic* among websites worldwide[2].

The continuous increase in traffic is a double-edged sword. On the one hand, it shows the users' interest in sharing media and files, both of which have promoted the development of new applications and technologies. On the other hand, the tremendous amount of data traversing the Internet incurs high transit fees for Internet Service Providers (ISPs) (increasing operating expenses – OPEX) and requires upgrades to the inter-ISP links (increasing capital expenses – CAPEX). If these links do not have sufficient bandwidth, then they can become a bottleneck.

How to effectively, efficiently, and economically distribute content through the Internet to the end users has been a hot issue for many years. A widely used method to address this problem has been for content providers to contract with commercial content delivery networks (CDNs) to distribute content to distributed caches operated by the CDN provider. This solution has a number of problems: (1) the content provider has to identify what content should be distributed by the CDN and provide this content in advance of requests to the CDN and (2) the content producer has to pay the CDN for their services and if the content is not as popular as expected, then this could be an unnecessary expense. Figure 1-1 shows the topology of a simple CDN system.

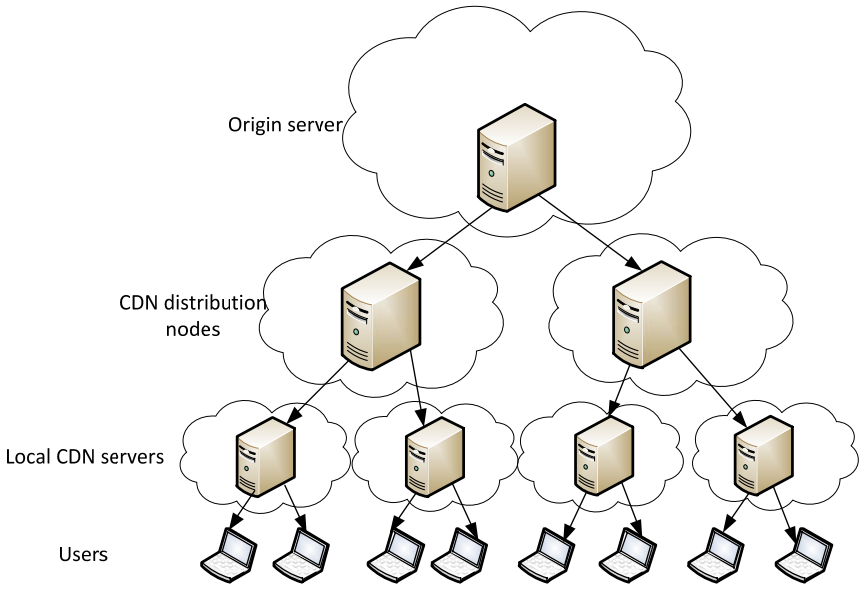

Figure 1-1: CDN architecture

An alternative to using CDNs is to exploit P2P techniques to provide a low cost and simplified content distribution solution. While P2P technology has been widely used by users for file sharing, today it is also being introduced by commercial actors for media streaming. PPLive is one of the most successful Chinese applications based on CDN-P2P technology providing *both* live streaming and video-on-demand services[3]. Using P2P distributes the traffic load over numerous *users*, thus reducing the burden on a central server (or servers). However, the P2P protocol is generally *unaware* of the network topology; hence a great deal of inter-ISP traffic is generated by P2P applications. This increased inter-ISP traffic leads to a significant concern among ISPs regarding their increased OPEX[4]. Figure 1-2 and Figure 1-3 show the *logical* and the *physical* topologies of a P2P network, respectively. Peers connect to their neighbors at the application level. This means that a neighbor which is one hop away in the logical topology may be located multiple hops away in the physical topology and may even be in another ISP's network. The dash lines in Figure 1-3 represent inter-ISP links. As Figure 1-3 depicts, even though both ISPs have two peers, each of these peers can connect to an external peer in the other ISP's network – hence unnecessary inter-ISP traffic may be generated. Note that we refer to this inter-ISP traffic as *unnecessary*, since ideally the first choice for every peer should be to get content from another peer *within* the same network as it is located and ideally a given content object only should have to pass across a given inter-ISP link once.

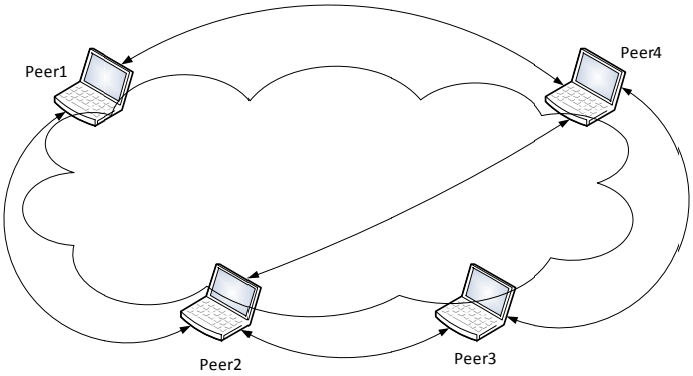

Figure 1-2: Logical topology of a P2P network

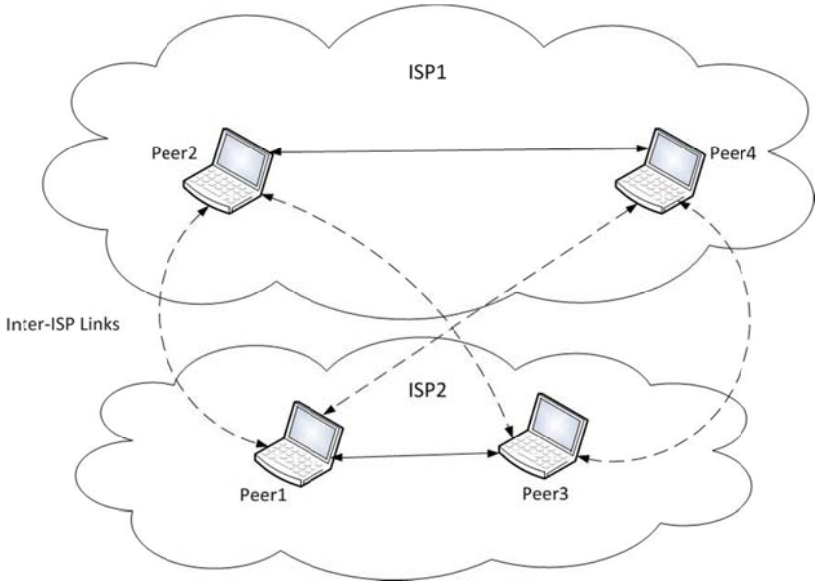

Figure 1-3: Physical topology of a P2P network

Various forms of caching are being deployed to further reduce the transit traffic and enhance the performance of the services. Web caching was widely used when the world-wide web emerged, but it gradually lost its glory when more advanced HTTP features were exploited [5]. However, the phenomenon of cachability is being discussed in the P2P network community[6], [7] specifically YouTube[8]and BitTorrent[9] both suggest that it is time to return to caching. Furthermore, caching has proven to be a vital technique to cope with the bandwidth constraints of the backhaul links to/from the base stations of mobile networks[10–13].

This master thesis project aims to analyze cacheability of media applications distributed over small geographic areas via the Internet. The application chosen for analysis is YouTube. The scope is limited to traffic observed by monitoring two residential networks in Sweden. This traffic was used to develop a user behavior model to characterize small populations, rather than individual users. The motivation was to abstract from data collected from a fixed IP network in order to provide a reference model for mobile network caching strategies for a cell. The number of users per cell is rather small and the traffic to and from the cell suffers due to the

high cost of the backhaul transmission. Figure 1-4 shows a simple topology of a mobile network. All the mobile devices (e.g., PC, tablets, mobile phones, …) connect to base stations (in 2G networks these are called Base Transceiver Stations or BTS, while in 3G networks they are called a Node B) via a radio link. The base stations connect to controllers (in 2G networks these are called Base Station Controllers or BSC, while in 3G networks they are called Radio Network Controllers or RNC). The data traffic in 2nd and 3rd generation networks is carried via a general packet radio service (GPRS). The serving GPRS support nodes (SGSN) act as relays between mobile users and the gateway GPRS support node (GGSN), which provides Internet connectivity. The links between the base stations and the controllers (depicted as dashed line in Figure 1-4) are the backhaul links. Due to the limited backhaul link bandwidth it is beneficial to cache content for each cell, so that the content only has to pass over the backhaul link once.

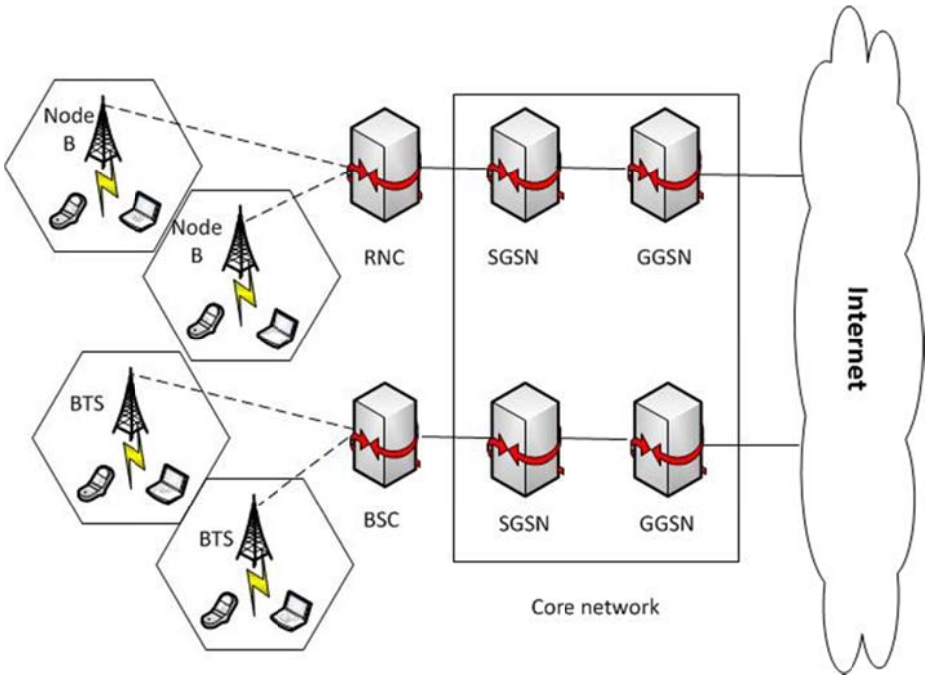

Figure 1-4: Backhaul links (shown as dashed lines) in a mobile network

To understand if there is a potential gain from cell local caching we utilized traffic traces of small groups to extract traffic patterns and looked to see to what extent there are multiple requests for the *same* content. If there are requests for the same content, then this content could potentially be cached, rather than transferred across the backhaul link multiple times. We also examine when the requests for the same content are made, as they must be sufficiently close together in time that the content would still be in the cache (assuming that the cache size is finite). We use the traffic in the area to create a model which can be used to characterize subgroups within the local area for each network in order to make effective caching decisions (what should be cached and how long should it remain in the cache). Based on these results, the cachability of content for YouTube is examined in terms of the percentage of the backhaul bandwidth that could potentially be saved by local

caching. Additionally, we compare the cachability between PC users and mobile users to identify the differences in user behavior patterns thus leading to different caching strategies.

2 Background

This chapter provides essential background knowledge for understanding this thesis work. Section 2.1 briefs some study results of Internet traffic usage. With the results suggesting that it is potential to enable cache in the network, section 2.2 introduces the basic structures of caching system. Section 2.3 explains the definition of locality-aware network. Following that, in section 2.4 and 2.5, the design of caching algorithms and the models used for showing video popularity are presented. In the end, the summary of previous YouTube related studies is given in section 2.6.

2.1 Traffic models and user behavior analysis

Internet usage should be examined and understood in order to aid in network design and improve service delivered to end users. Traffic models and user behavior have been studied widely by academia. For example, the Traffic Measurements and Models in Multiservice Networks (TRAMMS) project analyzed the characteristics of traffic at various aggregation levels in multi-service IP networks both in Spain and Sweden during the period 2007 to 2009 [14]. TRAMMS utilized a traffic monitoring and measurement framework to gain deep insight into IP traffic, especially video and P2P applications. TRAMMS also built models of user behavior at the application level. A goal of this modeling was to improve the broadband access networks and the quality of service (QoS) that they provide [14]. The IP Network Monitoring for Quality of Service Intelligent Support (IPNQSIS) project [15] is a successor to the TRAMMS project. This new project focuses on probing the network performance from the users' perspective in order to build a Customer Experience Management System (CEMS).

User experience is a vital criterion for the service providers; hence they want to measure the quality of service they actually deliver to their end users and use this information to design their business strategies. The Telecommunications Management (TM) Forum is a global industry association comprised of more than 220 of the world's leading service providers. It aims to promote and simplify the complexity of the service provider's business. Their technical report - TR148: Managing the Quality of Customer Experience [16] examines how to monitor and measure customer experience by studying a range of delivery mechanisms in real cases. The goal is to learn how to improve user satisfaction. In order to meet the requirements described in TR148, the TM Forum proposes a holistic end-to-end customer experience framework which consists of six application programming interfaces (APIs) [17]. These APIs can be used to keep track of user experience and can also predict the trends of user behavior, thus the provider can make intelligent modifications to meet their users' requirements. Their report emphasizes the importance of knowing customers' relationships and groups the customers in order to further improve the customer's satisfaction.

A great deal of research [18–22] has been done using different parameters to characterize Internet traffic usage, for example, overall traffic volumes over time (daily, weekly, and monthly), the composition of traffic characterized by application or protocol, and traffic monitoring over different time scales for specific applications such as Spotify, Voddler, and online gaming. The results of these studies are summarized in Table 2-1.

Table 2-1: Previous traffic analysis result

source	**Testing network**	**Major conclusions**
[18–20]	2600 fiber-to-the-home (FTTH) households and ~200 Digital Subscriber Line (DSL) households in a Swedish municipal network.	• Flash video over HTTP and BitTorrent dominate the traffic volumes generated by multimedia streaming and P2P file sharing services respectively [18], [20] • Daily traffic pattern shows the peak hour of traffic during a day is around 7PM to 9PM and it shows similar results throughout the week[18], [20] • Inbound traffic and outbound traffic distributions show the majority of traffic is generated only by a small portion of users, around 10% [19], [20]
[21]	2500 FTTH households in Sweden	• P2P based video streaming application (Voddler) dominates the number of flows in the streaming services due to the nature of the P2P protocol, but Flash video over HTTP dominates with regard to the total number of bytes. • The downloading traffic of streaming exceeds much more than uploading traffic in the experimental local area. Most of the downloading traffic is locally initiated.
[22]	More than 20,000 residential DSL customers of an European ISP	• Up to 4% of IP addresses are re-assigned more than 10 times a day which indicates that dynamic IP address allocation should be taken into account when identifying the hosts. • HTTP traffic exceeds P2P traffic. YouTube.com contributes 25% of all HTTP traffic.

2.2 Introduction of Caching

In addition to general Internet traffic analysis, YouTube, media streaming, and P2P file sharing systems have been extensively studied[23–26]. Since video-on-demand and P2P traffic consume a huge amount of bandwidth in the Internet, lots of research has been carried out and researchers have proposed various methods to reduce traffic between ISPs. Caching frequently-requested content is regarded as a very efficient way to reduce this inter-ISP traffic. Additionally, many researchers have discussed the possibility and the impact of caching in different systems, in addition to caching for the purpose of reducing inter-ISP traffic, hence there is an extensive body of literature about caching techniques and methods of evaluating them[27–32].

Web service caching typically takes one of two forms: client or server based caching. In the first form a *client cache* is implemented by the user's web browser. This cache can store both the content which is currently being viewed and items which were requested earlier. In the second form, the cache is implemented by a *proxy cache* [33] or by the web server itself. A proxy cache is located on a logical path between the web browser and a web server. This proxy stores a local copy of content that is likely to be frequently requested by the end users' browsers. This proxy reduces the load on the actual web server, thus reducing traffic on the path between the proxy and the web server. Note that the traffic between the web browsers and the proxy cache is *the same amount of traffic that would have gone to the web server in the absence of the proxy.* Figure 2-1shows the architecture of a proxy cache scenario.

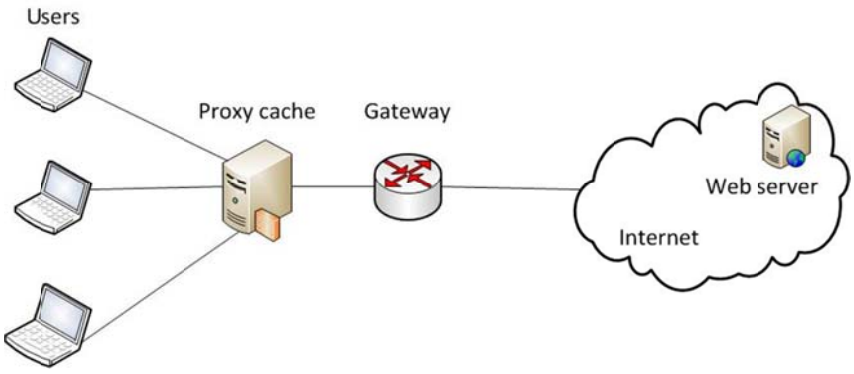

Figure 2-1: Proxy cache

If the item can be kept in the cache for an unbounded period of time, then if there are N requests for an item, there will be N requests for this item from the browsers to the proxy and a single request for this item by the proxy to the web server. the first browser's point of view the time required to fetch the item will likely be longer than the time to fetch the items directly from the web server – this is due to the need in the worst case to establish two transport sessions and the additional processing time added by the proxy's own processing. However, for all of the subsequent browser requests for this content, in the best case the time to fetch the item is simply the time to establish a session to the proxy and to fetch the item from the proxy's cache. Unfortunately, the analysis of the average case is much more

complex as it depends upon the temporal distribution of the requests and the load on the proxy when these requests arrive – as well as the complication of having to check if the content in the cache is still valid (typically requiring a request to the actual web server to see if the content has been changed since the time the content was cached – thus a transport session still has to be set up to the real server). An advantage of these queries for content only if the content has been modified since the time that the content was cached is that for content which never changes ("write once") or changes only infrequently the probability that the cached copy is valid is quite high. Additionally, the probability that the cached content will become invalid in a short period of time is low; hence many requests can simply be answered with content from the cache *without* checking to see if the contents have actually been modified. This can lead to inconsistent results, but with very large numbers of requests for the same highly stable content the probability of inconsistent results is low. Unfortunately, we cannot exploit this in our specific setting because we are looking at the case where there are small numbers of users in a subnet or a cell, hence there are **not** large numbers of queries. However, if there are requests that are *close* in time for the same content, then perhaps we can exploit this temporal coherence. Therefore some of the results that we want to look for in our collected data are the numbers of requests for the same item and their temporal coherence.

Theoretically, any content which is requested more than once is cacheable. According to Ager, et al., if t_i indicates the total downloading times for item i of size s_i, the cachability of n items can be calculated by the following formula [7]:

$$Cacheability = \frac{\sum_{i=1..n}(t_i - 1) \cdot s_i}{\sum_{i=1..n} t_i \cdot s_i}$$

Ager, et al. found that for P2P applications, when considering only local hosts, cacheability is 27%; however, for UGC sites the cacheability is considerably lower, mostly less than 10%[7]. One explanation is that when deciding whether to cache content or not if the proxy obeys the cache-control header, then only if the content is allowed to be cached would this content be put into a cache. Additionally, they found that more than 89% of data chunks are uploaded to and downloaded from external hosts (i.e., hosts outside the local ISP, hence this traffic must pass over the inter-ISP link). As this content is directly uploaded to the external host, then even though the content is available locally it will not be seen by the proxy cache – *unless* the proxy cache is on the upload path. These results imply that if local users share the same interests then a local cache could *potentially* reduce transit traffic. Leibowitz et al. [6] took a deep look into P2P cachability by measuring the caching gain in terms of byte-hit-rate. Their results indicate that higher traffic volumes yield higher caching gain and their experiments show that 67% of the bytes can be served from the cache – even when using a cache of only 300GB. Zink et al. [8] analyzed YouTube traffic in detail, then proposed three kinds of caching structures and compared their caching performance based on a simulation of a large university campus network. The results of these simulations are summarized in *Table 2-2*. A cache hit means that the content required by the user has already been stored in the cache, thus the user can obtain the

content directly from the cache. Similarly a cache miss means that the content is not found in the cache, hence the proxy cache has to request the content from the source (or another proxy cache on the logical path to the source).

Table 2-2: Summary of different caching performance for YouTube video clips, in all cases the cache replacement policy is First-in-first-out (FIFO), delete the oldest content.

Distribution infrastructures	Cache insertion	Result
Local caching at the client	All video clips which are firstly required by the user.	With a 50MB user cache, 84% of video clips with multiple requests from the same user can be served from the local cache of each user.
Peer to Peer	If cache miss happens, look up the missing content in another client's cache first instead of requesting it from the server and caching it immediately. Only cache content which cannot be found in other users' caches.	With a 1GB user cache, the hit rate can reach almost the same level as the case of a local user cache. However, compared with local caching, the performance is only marginally improved and may be even *worse* because requests to offline peers will be counted as a cache miss.
Proxy caching (Single proxy cache)	All video clips which are first required by the user.	100GB cache can increase the hit rate to the maximum which is about 28%. This is a very efficient solution in a local network with *thousands* of clients.

2.3 Locality-aware network

Other than the use of a local cache and proxy cache, which were covered in the previous section, users might also utilize each other's local cache as a resource to search for content they want. This is implemented in P2P networks. However, as mentioned in Chapter 1, a peer neighbor in the P2P network can be multiple hops away or even in another ISP's network in the physical topology. To cope with the delay and the underlying inter-ISP traffic problem, a locality-awareness P2P network can greatly improve network efficiency by clustering users geographically. Another kind of user clustering is based on user interests. Peers are selected among a group of users requiring similar content which could potentially reduce the content lookup time since if the content could be found on the peers' end.

Optimizing traffic locality can be implemented in a P2P system during the peer selection process. In [34], Ayodele Damola proposed a P2P traffic diversion algorithm in a broadband access network based on the McCircuit traffic separation mechanism which does not allow for inter-host communication. His method redirects

traffic within the local access network in order to offload the edge nodes at higher aggregation levels. An important requirement is that the peer knows which local peer it wants to communicate with, hence Ayodele's peer selection algorithm helps to select a path that avoids passing through the edge node, thus causing the traffic flow to be highly local within the access network.

Based on the observation that peers who send requests to a similar set of servers are more likely to be closer to each other, Choffnes and Bustamante [35] gathered dynamic DNS redirection information for use in peer selection. Their biased peer selection algorithm can significantly increase the average download rates and reduce inter-ISP traffic. Bindal et al.[36] also proposed a biased neighbor selection method for BitTorrent using a parameter k to leverage the portion of internal peers versus external peers for each downloading process. Their method ensures more internal peers are used while one or two external connections remain available. (Note that is important that there must be non-local peers – otherwise there is a risk of content being unavailable when it is simply locally unavailable.)

Other than relying on geographic factors, interest-based clustering is another property that can be considered when clustering. This method analyzes the correlation (overlap) between cache content shared by peers, then redirects queries to those peers who have the most files in common, these are referred to as semantic neighbors. Handurukande and co-workers in [37] and [38] present a study of traffic in the eDonkey network. They observed that the more popular files stably remained more popular. By clustering eDonkey users geographically or semantically they increased the hit ratio for rare files which are usually difficult to locate. Crespo and Garcia-Molina proposed a layered Semantic Overlay Network (SON) to facilitate the peer selection process [39]. They evaluated this approach in a music-sharing system which has sufficient content to be classified into a hierarchical structure based on the type of music. The content query messages were sharply reduced in their SON as compared to a general P2P system. This method requires an accurate classification process. However, in their experiment the queries were classified manually.

Lindsey et al. studied the spatial locality of mobile pattern in a campus wireless network[40]. They categorized web content from different news websites and traced user requests based on these news categories. Their research mostly focused on individual user's behavior which revealed that mobile users who were geographically close to each other had a high probability of sharing similar interests, namely requesting similar content. This phenomenon is called spatial locality. This study also showed that the popularity distribution of requests for objects matched a Zipf-like distribution. Furthermore, rather than showing high mobility, more than half of the users were relatively stationary as they connecting to only a single access point during a day.

This master thesis project was greatly motivated by this earlier research. However, in this work we study users' behavior in a fixed network. Rather than analyzing the users' web browsing interests concerning news websites, we focused on YouTube, the world's largest UGC video on demand (VoD) system.

2.4 Cache algorithm

Caching algorithms have been studied with the goal of making more efficient caching decisions. Since the space of a cache is finite, old content in the cache needs to be replaced gradually. Several new caching replacement algorithms are proposed in[41–43]. Each algorithm is suitable for a specific type of application, such as web document caching, BitTorrent-like patterns, and streaming media. The performance of each of these algorithms was evaluated in terms of hit ratios, latency reduction, and byte hit ratios. The details of caching algorithms are outside the scope of this project. However, the popularity distribution as a function of time scale is studied in this project in order to estimate how long content should be kept in the cache. This information can also be used to design (or select) a suitable cache replacement algorithm.

2.5 Zipf, Power Law, and Pareto Distributions

The popularity distribution of media content is a major factor in cachability. For example, if some video clips are only occasionally requested by very small number of users, then the gain of caching these clips would be very low. Given the tremendous amount of media content, how many of these items are frequently requested? How is this popularity distributed? A power law is the answer to both of these questions.

A power law is used to describe phenomena where the occurrences of a small number of events are frequent, while other events are infrequent. In the Internet there are quite a few small elements in the Web, but only a few large ones. Similarly a few sites attracted many visitors, while most sites barely get any attention.

Sometimes these phenomena in the last paragraph are described as following a Zipf distribution or Pareto's law distribution. In the following paragraphs power law, Zipf, and Pareto's law distributions will be explained in detail. In fact, all these three terms can refer to the same phenomena described previously.

Zipf's law was firstly proposed by an American linguist George Kingsley Zipf [44]. It was initially used to describe word frequencies in text. Zipf found that only a few words were frequently used, while most words were rarely used. Zipf's law refers to the frequency $\boldsymbol{f}$ of an event relative to its rank $\boldsymbol{i}$. It states the frequency of the i^{th} event is inversely proportional to its rank, β (where β is close to 1):

$$f \sim 1/i^{\beta}$$

As it shown in [45], Web access also followed the Zipf law. If $0 < \beta < 1$, the popularity distribution is a Zipf-like model. If $\beta = 1$, then the distribution exactly follows Zipf law. Zipf's law is easily observed by using a log-log graph as the plot will be a line. A steeper slope of this line indicates that a great many requests are related to only a small portion of the total web content. This characteristic shows the benefit of deploying web caching. By caching these frequently requested objects, different users do not need to request the same content from the server, thus reducing the burden on the web server.

Instead of concerning the exact frequency of i^{th} item, Pareto's law pays attention to how many items have a frequency greater than *f*. Thus Pareto's law is given in terms of the cumulative distribution function (CDF). For example, the number of items whose frequency is greater than *f* is an inverse power of *f*:

$$P[F > f] \sim 1/f^k$$

While a power law distribution shows the number of items whose frequency is exactly *f*. This is the probability distribution function (PDF) associated with the CDF:

$$P[F = f] \sim f^{-(k+1)} = f^{-a}$$

where a = k+1, k is the Pareto distribution shape parameter.

All these three models are used in this project to show statistical results, see Chapter 3.

2.6 YouTube related research

A lot of research considering YouTube has been carried out in recent years. This research can be mainly divided into three categories: YouTube Infrastructure and server selection, YouTube Video characteristics, and mobile user patterns. Each of these will be described in a subsection below.

2.6.1 YouTube Infrastructure and server selection

Google Inc. bought YouTube in 2006 and since then, YouTube has expanded its video distribution architecture by using Google's infrastructure. In [46] and [47], Adhikari et al. conducted a thorough study of the YouTube server selection strategy. Unlike an earlier study by Adhikari et al. [48], the study reported in [46] and [47] summarized three approaches used by YouTube to distribute load across servers based on the *new* infrastructure that YouTube began using after it expanded its cache server into Google's CDN, rather than analyzing the old YouTube infrastructure (as was done in [48]). Their later study showed that YouTube caches were located in 45 different cities in 25 different countries spanning the world. Each location has different number of physical caches which formed a three-tier cache hierarchy. The locations with a large number of caches are referred to as the primary data centers, and these caches were marked as primary cached. Only a small number of caches in less than 10 locations were designated as secondary and tertiary caches. Their study elaborated how YouTube dynamically balanced the selection of these caches for satisfying requests. The YouTube server selection is high locality-biased, the closer cache is to the user the high priority it has to be chosen. However, due to load balancing reasons there are still a large amount of requests that are redirected to a more distant cache. Torres et al. did an even more detailed study into the selection process and used round-trip time (RTT) to estimate the distance to each cache [49]. The results of the research by both groups are described in the similar conclusions found in [47] and [48].

An interesting result found by these researchers was that YouTube did **not** always pick the most optimal cache (i.e., the one that was closest to the user, usually a primary cache) for each request, but rather it dynamically redirect the request to other caches (e.g. during peak hours it redirected requests to a less busy server, or if the content cannot be found in the primary server, then it resorted to another server). In these cases, the requests are usually directed to a more distant server (in another city or even in another continent). These load balancing mechanisms may trigger multiple redirections if this target server cannot serve the video. Since each redirection is carried out by sending a new HTTP request and the new server is usually in a distant location, the delay before the video can be played back will increase, thus degrading the user experience. Moreover, lots of inter-ISP traffic will be generated. This behavior motivated me to examine this in more detail in thesis project.

Looking at the statistics presented in [47], we can see that even though Google is trying to spread caches around the world in order to locate content close to their users. They are still only in 45 cities around the world. Considering the increasing number of YouTube users all over the world, providing additional caches for smaller groups of people may be a profitable venture. Our work focuses on two municipal networks in Sweden and we divide each network into several small geographical areas and study cacheability of YouTuTube content for each area.

2.6.2 YouTube Video characteristics

YouTube video characteristics and user patterns have been very popular research topics. Gill et al.[23] collected a three month trace of YouTube traffic in a campus network. Their analysis summarized the HTTP request statistics which indicate the potential for caching. The daily usage pattern is presented in terms of the number of requests and separately in terms of bytes. The properties of the different video clips, such as file size, duration, bit rate, age, and category, have also been studied. Their report presents some results on the temporal locality of YouTube videos in their campus network. Cha et al.[50] focused on two YouTube video categories: entertainment and science. They use the video meta data provided by YouTube to study the global video popularity distribution over a number of years. They also studied how to make the UGC distribution system be more efficient by using caching and P2P techniques. However, their study does not consider any individual users' behavior. For example, the video request time intervals were estimated by using an exponential model instead of being deriving from actual user requests. In addition to providing YouTube video statistics as other studies have, Cheng, Dale, and Liu [51] pointed out a small world phenomena. On YouTube, every video clip is related to a list of similar video clips. They suggested that once a user plays a video clip, the cache should pre-fetch the directly related video clips as they are very likely to be watched (in the near term). Finamore et al. thoroughly compared YouTube user behavior between PC and mobile users[52].

Our work is also based on YouTube traffic statistics collect from our target network rather than considering a global scope or using a campus network. As noted before we have focused on small areas in residential networks. Unlike previous studies which used the IP address to identify each user, we utilized the media access and control (MAC) address as a unique identifier for each household or device (depending on whether there is a home router installed between the home network and the ISP – if so we only have the MAC address of this router and cannot differentiate between requests that could be coming from different computers in the household). We also compare the behavior of IPTV and mobile users in different areas. Since we have considered very small groups we have data that might be used for predicting the behavior of individual mobile users' behavior in each cell of a mobile network.

2.6.3 Mobile users' patterns

Shafiq et al. [53] and Maier et al.[54] have examined traffic in a mobile network and they extracted dynamic temporal traffic patterns. In addition, they studied the traffic volume by different applications via mobile devices. Maier et al. showed that HTTP traffic was the dominant share of the total traffic and that Apple products had the largest share of the user devices in their traces. With regard to application penetration, the web browser was the most popular application being used, while about 20% of mobile users were using Google's YouTube app.

Gember et al.[55] collected traffic for both mobile and PC devices for three days in campus Wi-Fi networks. Instead of characterizing this traffic by application, they summarized the traffic patterns by protocol. They found that HTTP traffic comprised 97% of the total traffic volume generated by mobile devices. More specifically, video data accounted for 42% of HTTP traffic to/from mobile devices. They further compared the content similarity between mobile devices and PCs. Their results showed that mobile users repeatedly requested the same content more often than did PC users. This suggests deploying cache mechanisms could improve the users' experience. Additionally, local caching in the device could save battery and reduce the need for the network to retransmit the same content.

3 Methodology

This project was carried out by applying three processes: data collection, filtering and analysis. This section will introduce the pilot networks where the data was captured, the resulting dataset, and the software was used throughout the whole project to post process, filter, and analyze the data.

3.1 Network Architecture

The target networks are two municipal fiber based IP access networks in Sweden. For privacy reasons, they are referred to simply as the North network and the South network respectively in this paper. The data was collected from roughly 5000 households in the North network and 1500 households in the South network.

The customers in each network are local residents who can freely choose different ISPs for access to the Internet. The access speeds range from 1 megabit per second (Mbps) to 100Mbps depending on the subscription the customer chose. As shown in Figure 3-1, the placement of traffic measurement probe in both of the networks is at the edge of the ISP's connection to the Internet. The measurement probe is connected to the Internet edge via optical 50/50 splitters which split the optical signal into 2 exact copies, one for measurement use and one for transmitting to and from the upstream and downstream network devices. As a result of this passive measurement probe configuration the measurement node works independently and does not affect the network traffic.

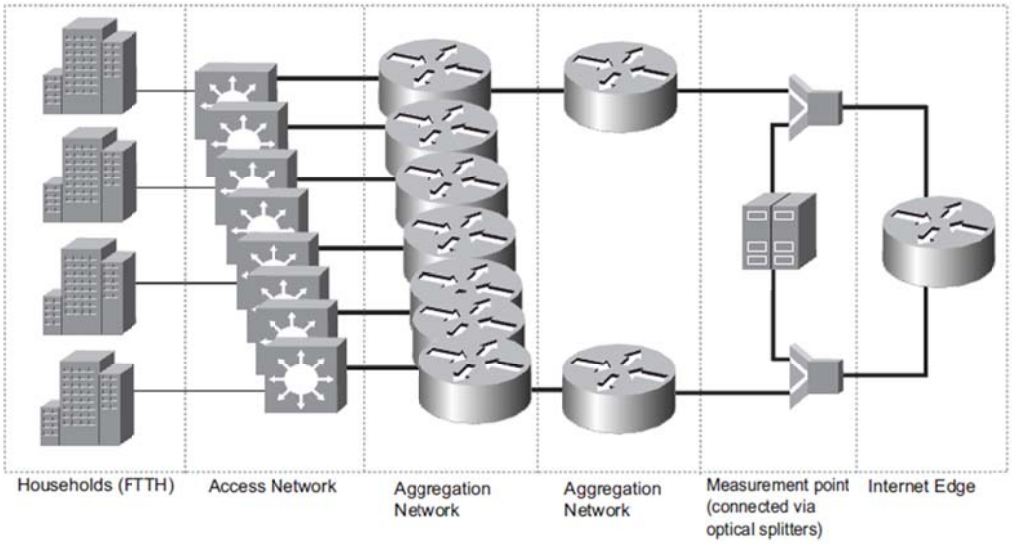

Figure 3-1: Network architecture [21]

3.3 Data Collection

This section elaborates how the YouTube data was collected and filtered in this project. In section 3.2.1, the traffic measurement equipment used in the project is introduced. Section 3.2.2 explains the YouTube signaling system which helps to understand how to set the filtering rules in PacketLogic to collect the YouTube data. In the latter part of section 3.2.2, it shows how the collected data is filtered before it can be used to calculate the caching gain.

3.3.1 Measurement equipment

The measurement probe that was used for measurement was a Procera Network's PacketLogic (PL 8720 for the north network and PL7720 for the south network) probe. This device is a scalable real-time traffic management tool which can be exploited in all types of network environments. It can be used for traffic filtering, traffic shaping, or to provide statistics for traffic analysis. It supports deep packet and flow inspection of IP packets. Packet content classification is implemented by using their PacketLogic Datastream Recognition Definition Language (DRDL)[56]. A wide range of criteria, such as session or connection, are supported in order to identify which applications are responsible for each datastream. Other than accurately identifying the type of stream, another unique capability of DRDL is using connection flags to classify traffic. For example, "Asymmetric", "Beginning", and "Streaming" are typical flags. These flags provide greater flexibility in traffic identification.

The PacketLogic client is a graphical user interface used to configure and operate the PacketLogic probe. This client enables an operator to access all frequently used features. For example, configuring objects and rules sets to adjust the monitoring criteria to meet the probe operator's needs, browsing the real time traffic data through Live View module, displaying statistical charts in the Statistics module, and so on.

3.3.2 Data filtering

As described previously, YouTube is one of the most popular video-on-demand websites and contributes a significant amount of traffic to the Internet. As a result, in this project, YouTube data was chosen to analyze the cacheability of video clips. This specific set of data was selected because YouTube users generate a number requests and upload and download a lot of video data. However, due to limited storage on the server that was used for post processing of the collected data, two filtering rules were created in the PacketLogic probe (1) to filter out YouTube video requests which we are interested and (2) to neglect all other traffic.

In order to properly set up filtering rules, we need to understand the basic signaling used when retrieving video clips from YouTube. This signaling is shown in Figure 3-2. We focus on the process from when the video clip starts to be downloaded. For PC users, Adobe Flash Player is used for video download and playback. When this player is ready to download the video, the player sends a

"HTTP GET videoplayback" request to the server. If the server is current congested or the required content is not available, then the server can redirect the player to other servers by replying with a "HTTP 302 not found" response. This response contains the hostname of another video server which the user should send a new request to. After resolving the hostname, the player sends a new "HTTP GET videoplayback" request to this server's IP address. This redirection and request process can be repeated several times until a server is found that has this content available. Finally, if the video clip is found, a server will send a "HTTP 200 OK" response informing the player that it can the desired download the video clip from this server. According to [52], YouTube's default setting for a PC is to download the whole video in a single TCP session, whereas on a mobile device, video content is downloaded by chunk and each chunk utilizes a new TCP session. However, in order to make the download more efficient, the video content is chopped up as well into chunks even for PCs. So whenever a chunk is downloaded, a new "HTTP GET videoplayback" is sent to request the next chunk of the video clip. A "Range" field in the header indicates which specific portion of the video clip is being requested.

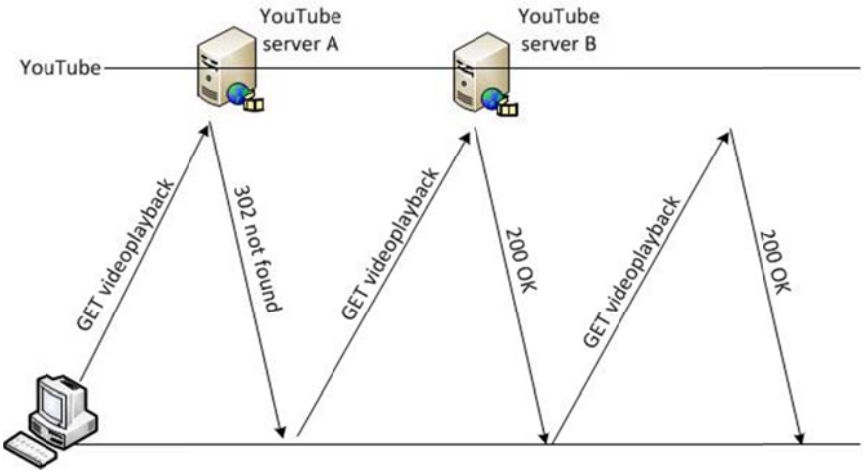

Figure 3-2: Steps to retrieve video stream from YouTube (when using a PC)

As shown in Figure 3-3, a custom YouTube application is used by mobile devices to request and play YouTube video clips. This application sends a "HTTP GET videoplayback" request with a header field "Range: bytes=0-1" to ask for the video cilp's size (in bytes). Then the server replies with a "206 partial content" response containing the video clip's size. Following this the app sends a series of GET videoplayback requests to retrieve the chunks of the video as described earlier.

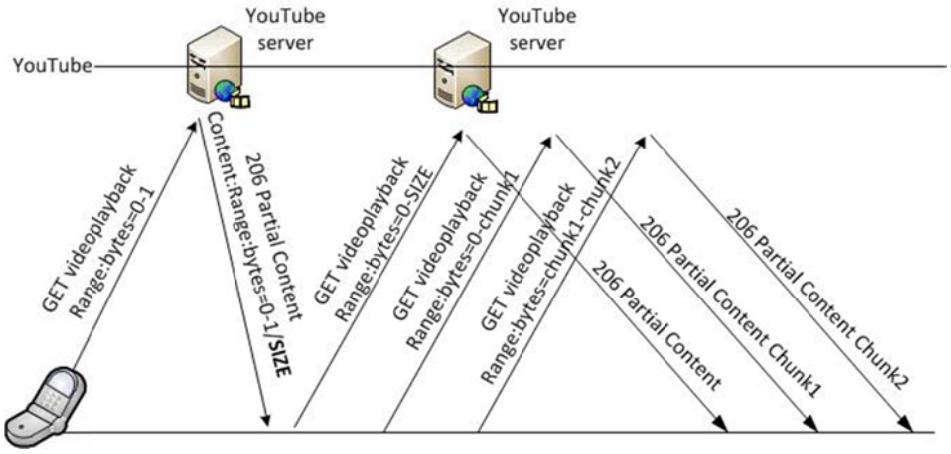

Figure 3-3: Steps to retrieve video stream from YouTube (when using a mobile device)

We should note that, in many cases, commands such as fast forward, rewind, or changing the video resolution during playback are observed. When one of these commands happens, a new "HTTP GET videoplayback" request is generated by the application.

Based on this analysis, two rules were configured in the PacketLogic probe. One matches a uniform resource locator (URL) starting with "youtube.videoplayback". Due to this rule the PacketLogic probe will dump the subsequent data in each TCP session when such a request is found. In order to limit the disk space needed for data in this rule we set the "initial flag", so that we only record the initial part of each TCP session, in our case, the videoplayback request itself.

In this project, we record the requests from each user for each video clip. Subsequently, cacheability is analyzed based on how many times a given video clip has been (repeatedly) requested. To identify each specific video clip, the 16-digit video ID in the "videoplayback" request is used and this ID is assumed to be unique for each clip throughout the data collection period. Since each video clip could generate many GET videoplayback requests it is difficult to tell which one indicates a *new* repeated request from a user. Therefore, we consider the time between two requests for the same video clip from the same user based on one week of data collected in the north network. As shown in Figure 3-4, about 70% of the requests were generated within one second which is mainly due to redirections. In [52], Finamore et al. reported that roughly 90% of video clips are aborted within three minutes by users, regardless of whether they are using a PC or a mobile device. Considering that, as video clips are sent in chunks and each chunk is roughly one minute of the video clip, then in theory during a typical playback any new player-generated request for the same video clip (but with a different offset from the start of the video clip) seen during the first three minutes should not be counted as a repeat request when calculating the hit ratio. Figure 3-4 shows that over 98% of the requests were generated within 100 seconds. In this project, we assume that if the time interval between two videoplayback requests for a video clip from the same user is longer than three minutes, then this request could be considered as a new request for this video.

The collection of traffic started simultaneously from both networks, at 00:00 (local time) on January 30th, 2012 and ended at 00:00 February 27th, 2012. However, in the north network, the trace ended at 00:00 AM February 20th, 2012, due to an unexpected equipment breakdown. The overall data sets are about 70GB and 52GB for the north and south network respectively.

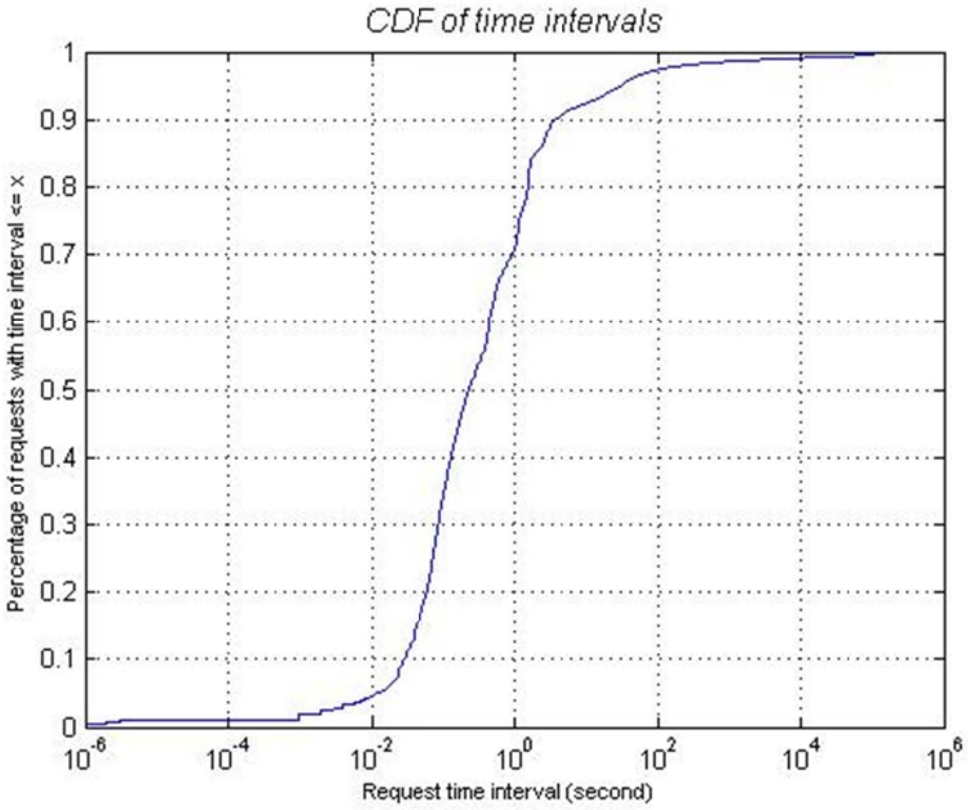

Figure 3-4: Time intervals of unfiltered requests

3.4 Data analysis

Figure 3-5 illustrates the process of data analysis for both north and south networks. Although the analysis process basically follows the same procedure, since the network infrastructures of the two networks are slightly different there are some minor differences in processing.

First the PacketLogic probe was configured to save the packet dump files (as pcap format files) on a server (identified as N1 in the north network and S1 in the south network). One each of these servers a parsing script is executed to extract the information needed, for example a time stamp, MAC and IP addresses, user agent string, and video ID. The extracted data is saved in text files.

Each server is placed in the same network operator's network as the PacketLogic probe device. In order to preserve users' privacy the MAC address and IP address in these text files are hashed. This guarantees none of the sensitive user information leaves the operator's network.

These text files with the hashed addresses are uploaded to server A. On this server the three minute filtering rule is applied to remove duplicate requests and the filtered data is saved into a MySQL database.

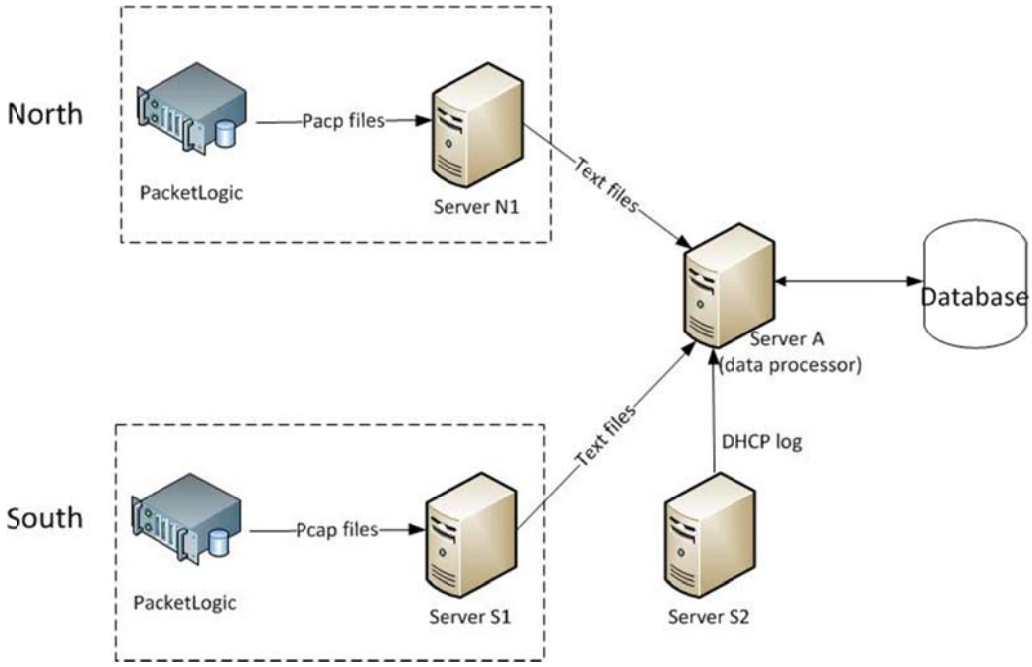

Figure 3-5: Data processing

The north network is a layer two network, thus the source and destination MAC address never changes while a data packet traverses the access network. In our case, the probe is installed at the edge of the Internet access point, thus the source MAC address in the packet captured from the household is either the end device's MAC address or the home router/switch's MAC address (if a home router is being used). The north network is configured with VLANs and these VLAN groups can be mapped to different areas.

In contrast, the south network is a layer three network, hence the MAC address in each packet changes hop by hop, thus in our packet dump, the source MAC address is always the last hop router's MAC address does reveal the household/device information. For this data an extra step is needed to replace the MAC address in each packet based on the DHCP address assignment. The MAC and IP address mapping is retrieved from the ISP's DHCP log together with the identifier of the access switch that the household is connected to. These switches have been grouped into different geographical areas. The updated information is saved in the database.

4 Results and Analysis

In this section, the data obtained from the two traces is presented. First, the statistics of YouTube video clips and the traffic patterns are shown. By analyzing the YouTube video popularity distribution and user behavior, two cache schemes are proposed. The users are further differentiated by user type and the cache hit ratio is compared between PC and mobile users.

4.1 YouTube statistics

The upper and bottom parts of Figure 4-1 give overviews of the measurements of the north and south networks (respectively). Unlike any previous studies which use the IP address as a user identifier, in this project the user identifier is based upon a unique combination of MAC address and user agent string (as this is likely to represents a specific end user device). The "Video stats" shown in Table 4-1 shows the total number of distinct video clips which were requested by users during the entire trace period. The "Request stats" enumerate the number of requests from a user. The total number of unique video clips and requests for both networks are listed in the "Total" column. As shown, despite the slight different in trace duration and number of users in both networks, the statistics show similar results. Over 70% of the video clips were only requested once during the entire period, hence caching does not have any possibility to improve the performance for these video clips. This means that less than 30% of the video clips were requested more than once; however, the repeated requests for these video clips accounted for roughly 60% of the total requests. This is due to the fact that a small number of *popular* video clips attracted most of the user interest. These video clips could potentially be cached, thus saving a lot of bandwidth.

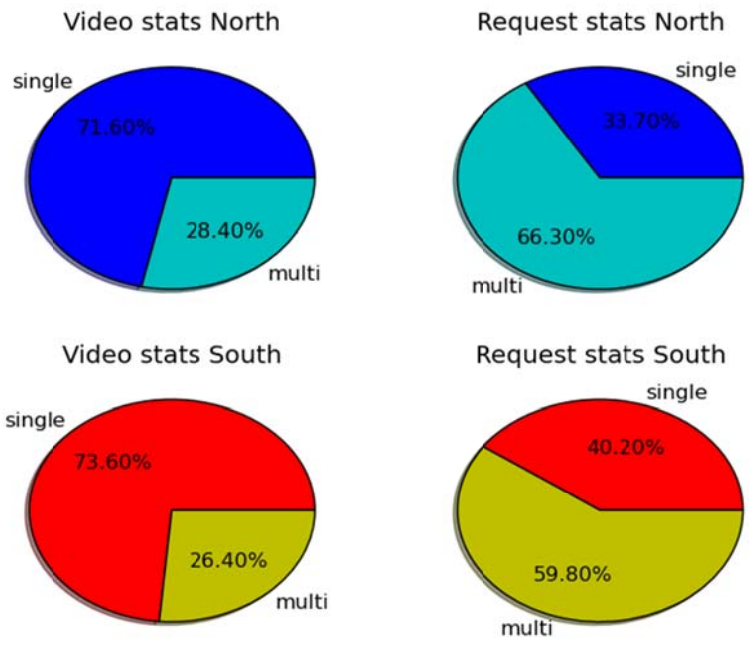

Figure 4-1: Video and request stats North/South

Table 4-1: Video and request statistics

Network	Date	Video stats			Request stats			# of unique users
		Total	Single	Multi	Total	Single	Multi	
North	01/30-02/19	495105	71.6%	28.4%	1052447	33.7%	66.3%	24059
South	01/30-02/26	336257	73.6%	26.4%	615167	40.2%	59.8%	9762

4.2 YouTube traffic pattern

Figure 4-2and Figure 4-3 show the number of requests and unique number of video clips on a daily basis during the data collection period in the north and south network (respectively). As expected for both networks more requests were generated during the weekends and the same video clips are requested repeatedly every day.

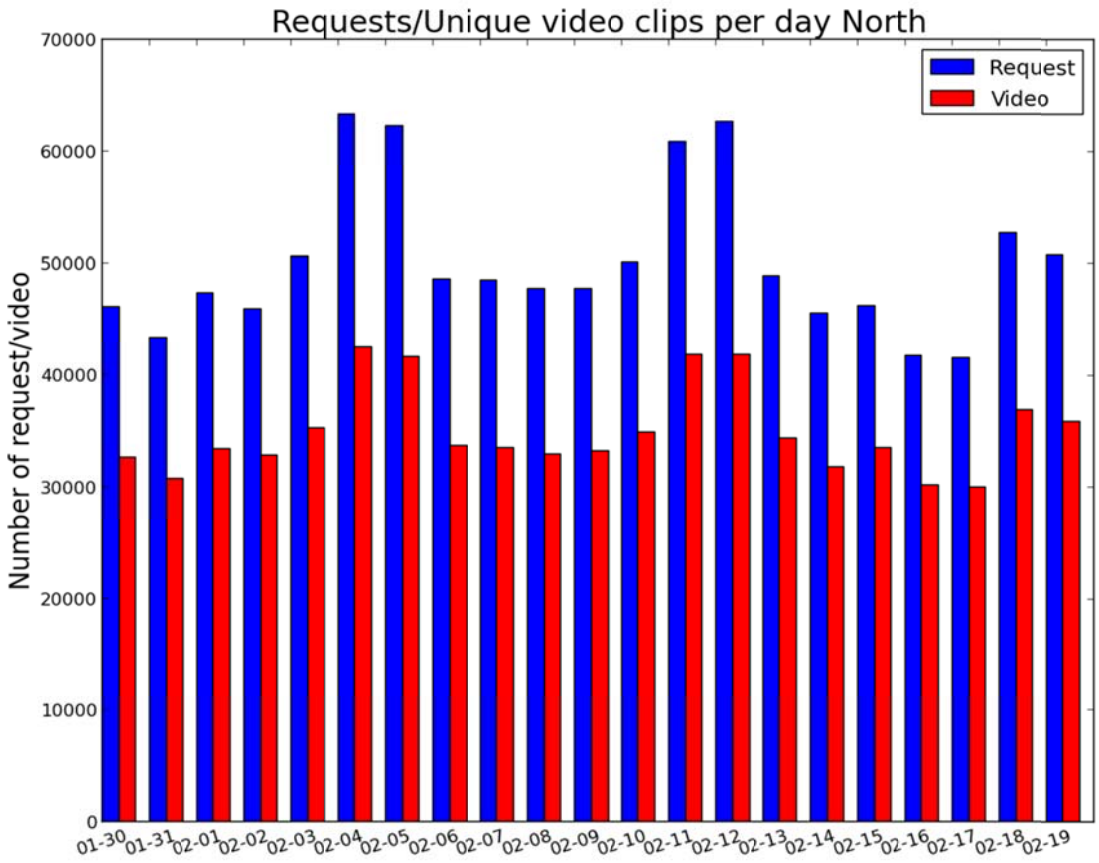

Figure 4-2: Requests/Unique video clips per day North

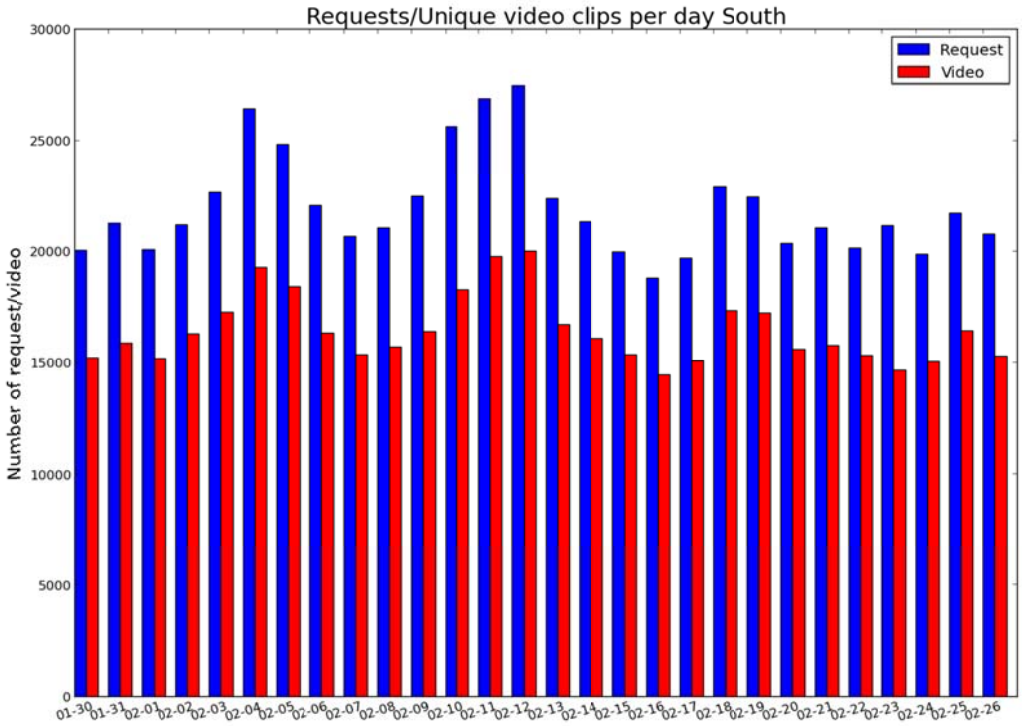

Figure 4-3: Requests/Unique video clips per day South

To illustrate the request distribution within each day, Figure 4-4 and Figure 4-5 depict the number of requests during each hour during the data collection period for both networks.

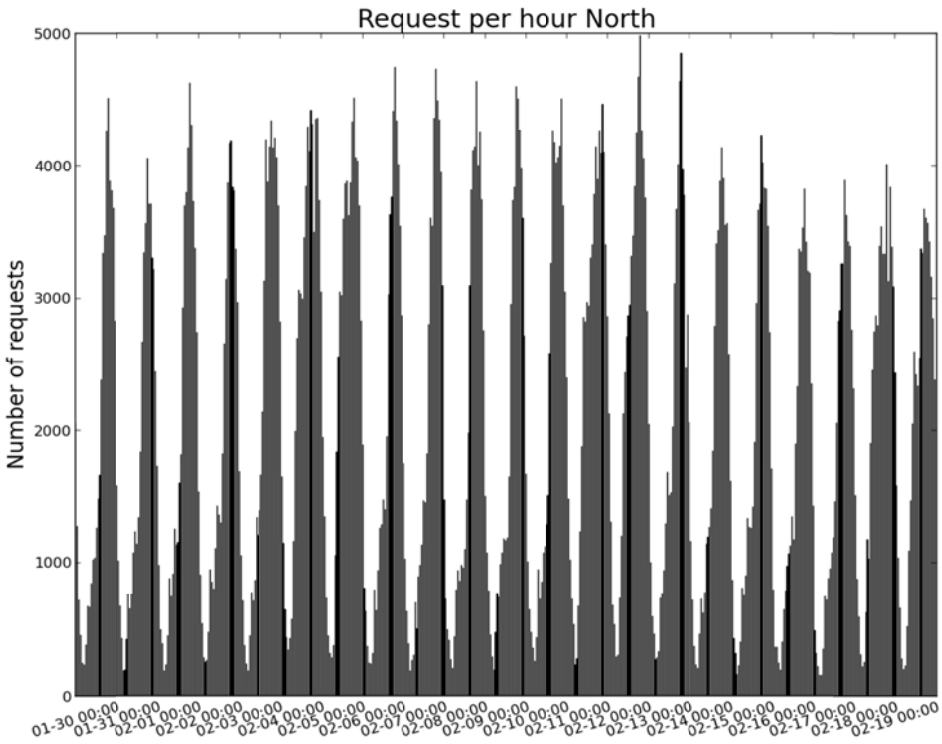

Figure 4-4: Requests per hour North

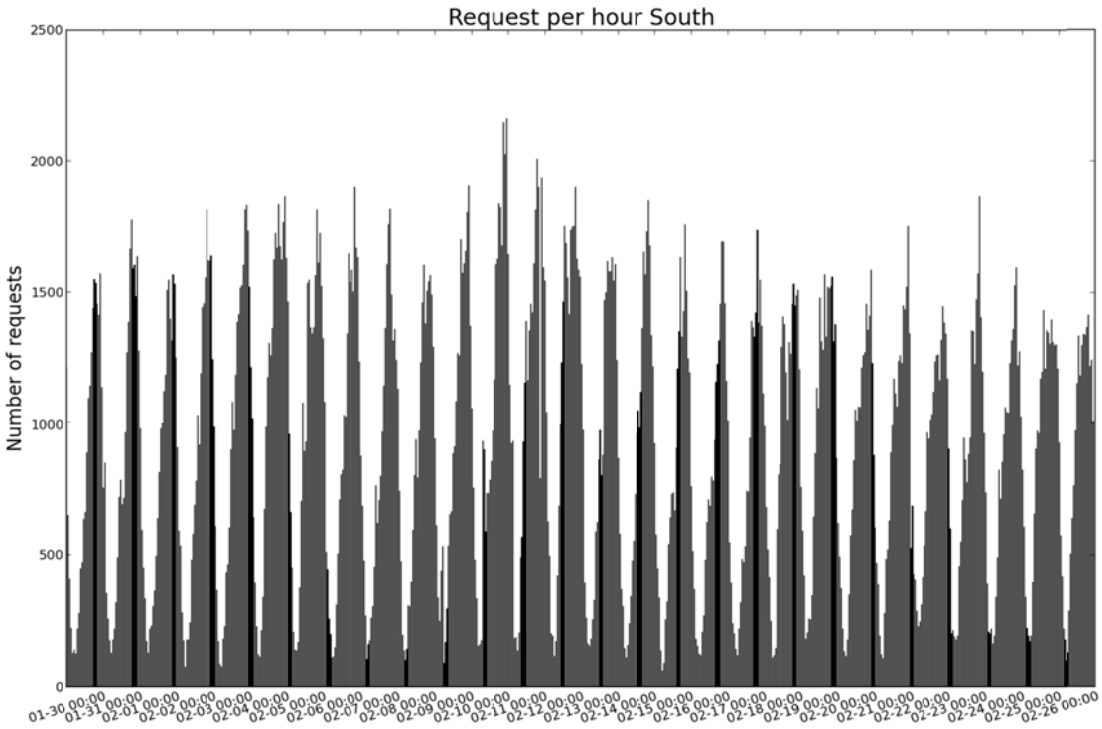

Figure 4-5: Requests per hour South

The graphs show periodic daily patterns which show that each day has peak hours and that these peak hours vary slightly from one day to the next. The total number of requests generated during peak hours was roughly constant throughout the entire data collection period.

Figure 4-6 and Figure 4-7 show the fraction of total video requests during each hour of a day. The average value through the whole data collection period is used. The requests steadily increased from 6:00 to 17:00. The peak hours are from 17:00 to 22:00. After this the number of requests per hour begins to rapidly decline.

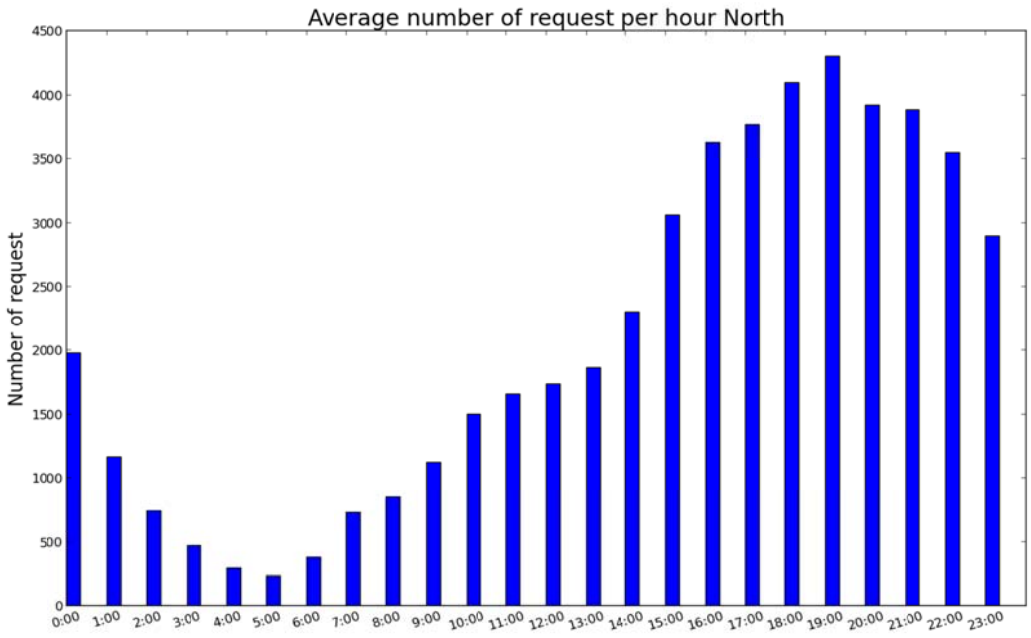

Figure 4-6: Traffic pattern by time of a day North

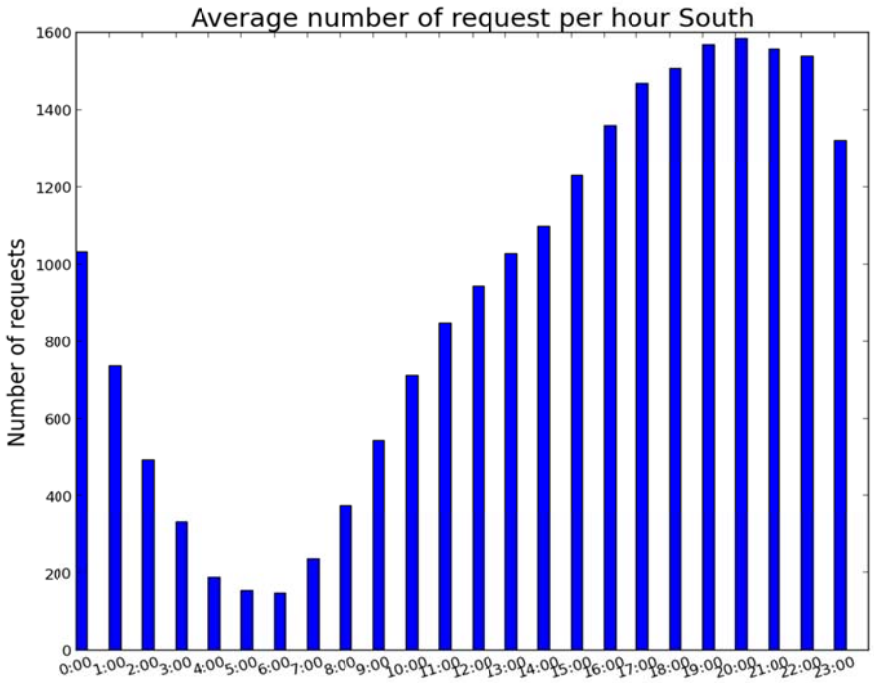

Figure 4-7: Traffic pattern by time of a day South

4.3 Infinite Cache for YouTube

Based on the previous analysis, an infinite sized cache scheme was analyzed in order to understand the cacheability of YouTube traffic. An infinite sized cache means that all the video clips that have been requested by users will be cached forever. The cache has unlimited space and not content will be deleted from the cache even if it is no longer popular. Cacheability was studied in three forms: gross gain, terminal gain, and net gain. In this project, the actual data size of each video clip was **not** taken into account due to limited storage on the server. The initial flag

in the filtering rules ensured that we only record the initial part of each TCP session to avoid downloading the streaming media data. Thus, the number of requests was used to calculate a hit ratio.

Gross gain We assume that there is a proxy cache for each area. All the users' requests will be analyzed by the proxy cache to decide whether to forward the request to the YouTube sever or not. If the video clip is already stored in the cache, then the video data from the cache will be sent directly to the user. Otherwise, the proxy cache will send a request to the YouTube server to retrieve the video clip. Once the proxy receives the video clip, the proxy cache will store it in its local storage. The gross gain is the gain that can be obtained from a local proxy cache and is defined as:

$$Gross\ gain = \frac{Total\ number\ of\ requests - unique\ number\ of\ video\ clips}{Total\ number\ of\ requests}$$

The gross gain does not reveal the individual user behavior. A high gross gain can be obtained when a lot of users share the same interests and hence request the same video clips. This can also occur when a few heavy users request video clips multiple times, even though they do not have any common interests.

Terminal gain If we assume that each end device has an unlimited sized cache, then this cache can store every video which has been requested by this end node, hence the delay to retrieve a video from either the local proxy or the YouTube sever will be avoided. This gain is defined as:

$$Terminal\ gain = \frac{Total\ number\ of\ requests - \sum unique\ number\ of\ video\ clips\ per\ user}{Total\ number\ of\ requests}$$

If the terminal gain is very high, then each terminal requests the same video several times.

Net gain Net gain eliminates the gain from one user's repeated requests in order to show how much we can benefit from the repeated requests from multiple users. If we assume that the number of unique video clips is n, then the net gain is defined as

$$Net\ gain = \frac{\sum_1^n Number\ of\ users\ who\ requested\ video(x) - Number\ of\ unique\ video\ clips}{Total\ number\ of\ requests}$$

The net gain is calculated based on the data transferred in each area. Our goal is to see if user in each small area request similar content or not. As a baseline, random groups were generated for comparison. These random groups were selected from among the customers in the whole north or south network, such that each group contains exactly the same number of users in each geographic area. This random selection was executed 50 times and the hit ratio shown in the graph is the mean of the 50 corresponding hit ratios.

Figure 4-8 shows the hit ratio in the north network. The gross gain, shown as dark blue triangles, ranges from 30% to 45% indicating a moderate probability of successful caching. In general, the more users in each area the higher the gross gain. But the actual success depends on the different user behaviors in real life case. To compare each of these areas, random groups were chosen to show the most common model. The same numbers of users as in each area were randomly selected from the users in the whole network in order to calculate the hit ratio. A random selection was made 50 times per sample and each light blue dot represents the mean value of these 50 hit ratios. The distribution of each random group's hit ratio is shown by the boxplot. The blue box represents the range of lower quartile (Q1) to upper quartile (Q3) of the data. The red line indicates the median value of the dataset. The interquartile range (IQR) defines as Q3-Q1. The whisker ranges from the Q1-1.5*IQR to Q3+1.5*IQR. All the data which are out of this range marked as the plus sign are called outliers which mean that they are distant from the rest of the data. The gross gain of the random selection shows a smooth increasing trend with the number of users per area.

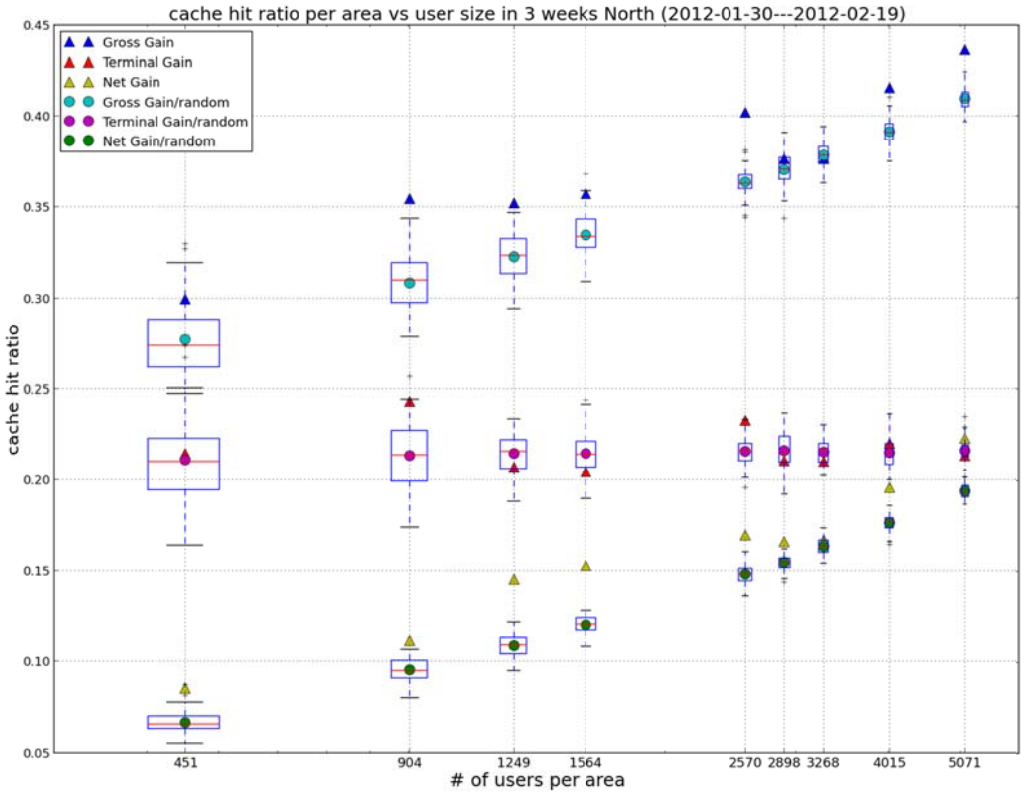

Figure 4-8: Cache hit ratio per area in 3 weeks North (Infinite size cache)

Another interesting feature that should be noted is that the gain of random selection is lower than the gain of each area, which suggests that people in each area share same interest to some extent. The terminal gain from each area ranges from 20% to 25% which suggests independent user interests. The terminal gain of the random selection remains constant at around 22%. This indicates that the replay behaviors among users are very similar. The difference between the gross gain and the terminal gain is due to repeated requests for the same content from different users. If both a local cache and terminal cache are enabled in the network, then the yellow line shows the net gain of the local cache in this case. The net gain of each area is still higher than the gain for random groups.

A similar pattern is shown in Figure 4-9. This figure shows corresponding results in the south network (even though the population in each area is even smaller than in the north network). This phenomenon is called geographical locality, meaning that there is some locality in the interests of the geographically chosen populations, this is indicated by the fact the cache hit ratio is higher within these populations than the randomly chosen populations.

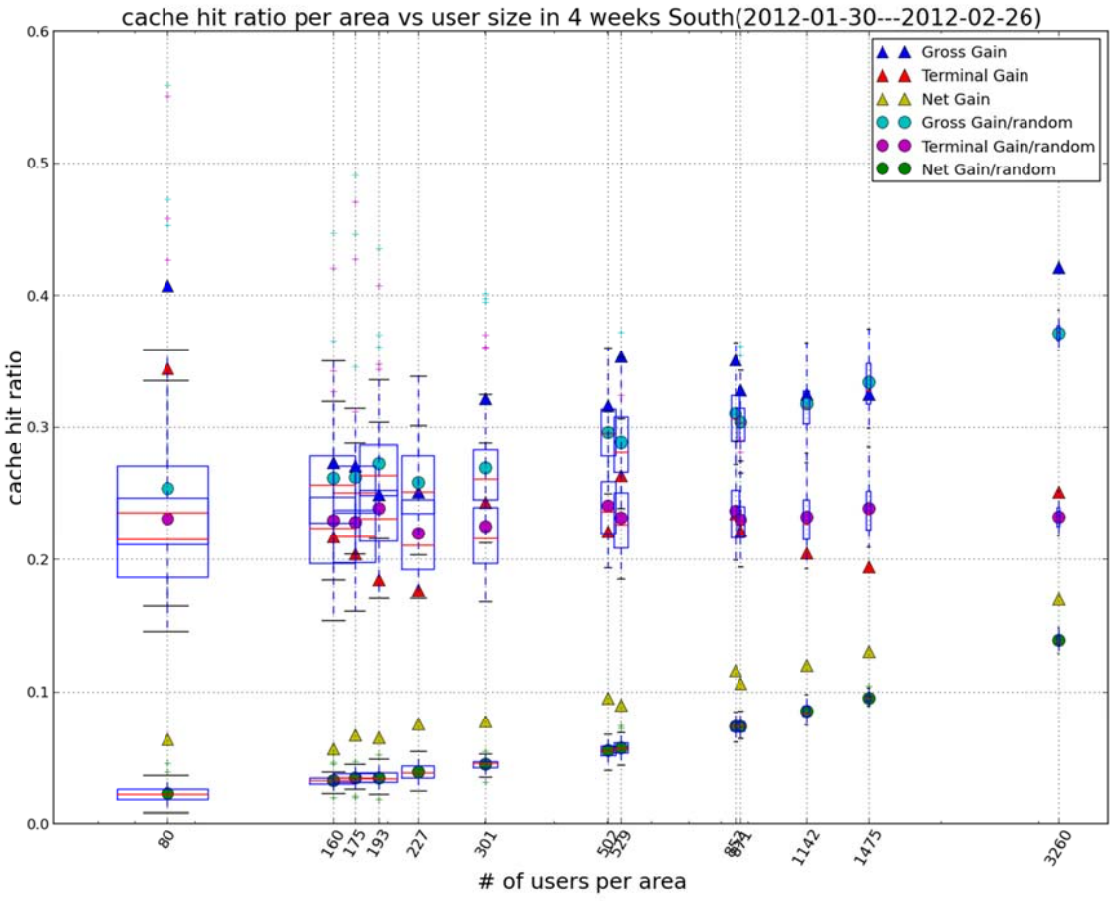

Figure 4-9: Cache hit ratio per area in 3 weeks South (Infinite size cache)

4.5 User request statistics

An analysis of user requests is studied in this section. The purpose is to understand whether there are many users who request the same content or a small number of heavy users who keep requesting the same video clips. The complementary cumulative distribution function (CCDF) is used to depict the distribution of the user requests. The x axis is the number of unique video clips clicked by each user and the number of total requests per user in the upper and bottom graphs (respectively). The y axis shows the percentage of users with this number of video clips or who request more than x video clips using a logarithmic scale. In both networks, as shown in Figure 4-10 and Figure 4-11, only 10% of the users requested more than 100 video clips or generated more than 100 requests during the entire data collection period. There are very few heavy users* in either of these two networks.

The request behavior of each user is further analyzed and the results are shown in Table 4-2. On average, in both networks during the entire trace period, if one user is interested in a video clip, he or she would watch it two or more times. In Figure 4-12 and Figure 4-13, the y-axis is the exponent of base 10. We see that almost 98% of the users watch a video clip less than 10 times throughout the data collection period. Two interesting questions arise: (1) Since each video is more likely to be requested by the same user less than 10 times, how does the user interest drop? (2.) When does the replay occur? More specifically, will each user request the same content quite soon or will different users request the same content within a short time? Further analysis of these questions is discussed in the following two sections.

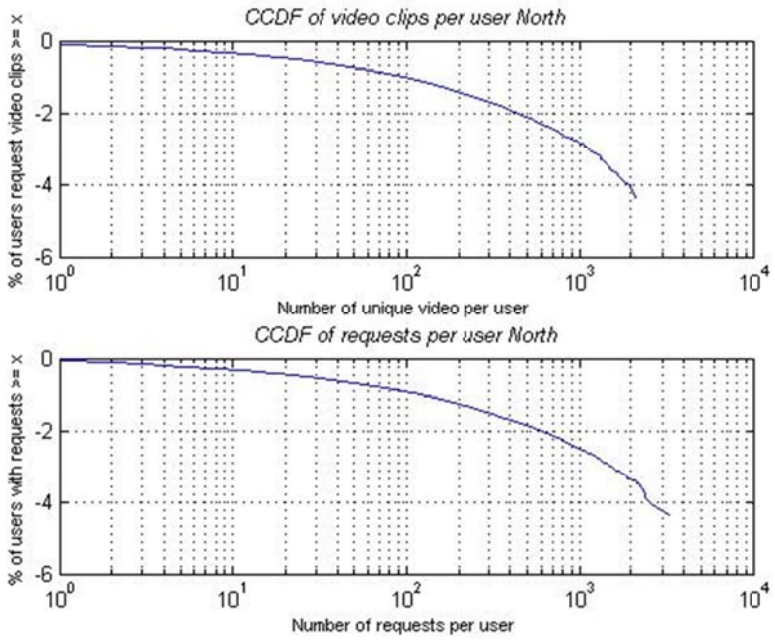

Figure 4-10: CCDF of requests/unique video clips per user North

* Here we have defined heavy users as those that generated more than 100 requests during the data collection period.

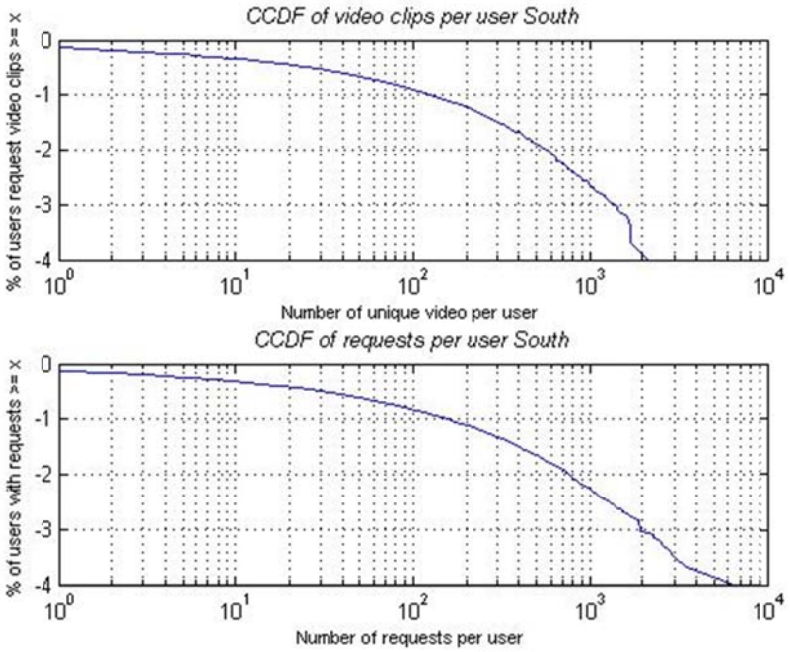

Figure 4-11: CCDF of requests/unique video clips per user South

Table 4-2: Video clips requested more than twice from same user

Trace	Max. number of requests per client for the same video clip	Avg. number of requests per client for the same video clip	Median number of requests per client for the same video clip
North	463	3	2
South	238	3	2

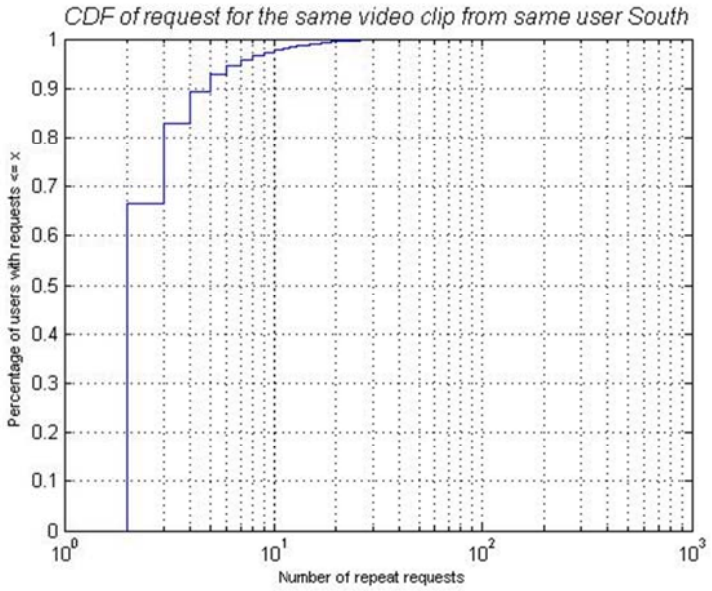

Figure 4-12: CDF of repeat requests from same user (in the North network)

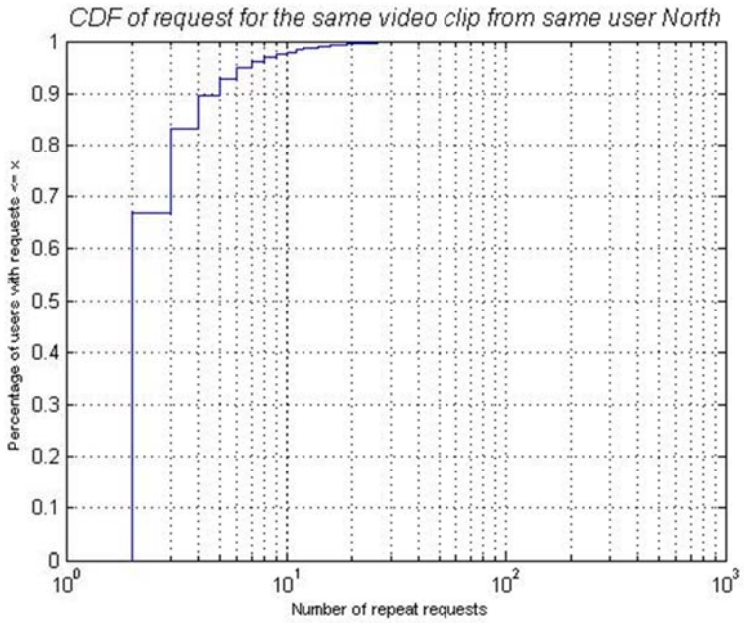

Figure 4-13: CDF of repeat requests from same user (in the South network)

4.6 YouTube video popularity

Video popularity has a significant impact on cache design. As popular video clips should be kept in the cache, thus if content does not attract enough attention or content loses its popularity, then it should be removed from the cache. The rank of the video clips versus the frequency of the video clips being requested as shown on a log-log scale is the most common method of viewing a Zipf distribution. The YouTube video clips requested in our traces follow a linear (log-log) Zipf distribution as shown in Figure 4-14 and Figure 4-15. Unlike the long tail effect described in [50], where the number of views of unpopular video clips drop rapidly, in our data set, popularity drops linearly on a log-log scale. One explanation for the long tail pattern in [50] is that they only selected two specific categories of YouTube videos, thus the result is biased. Whereas our dataset includes all the YouTube video clips which have been requested by the users in their specific network.

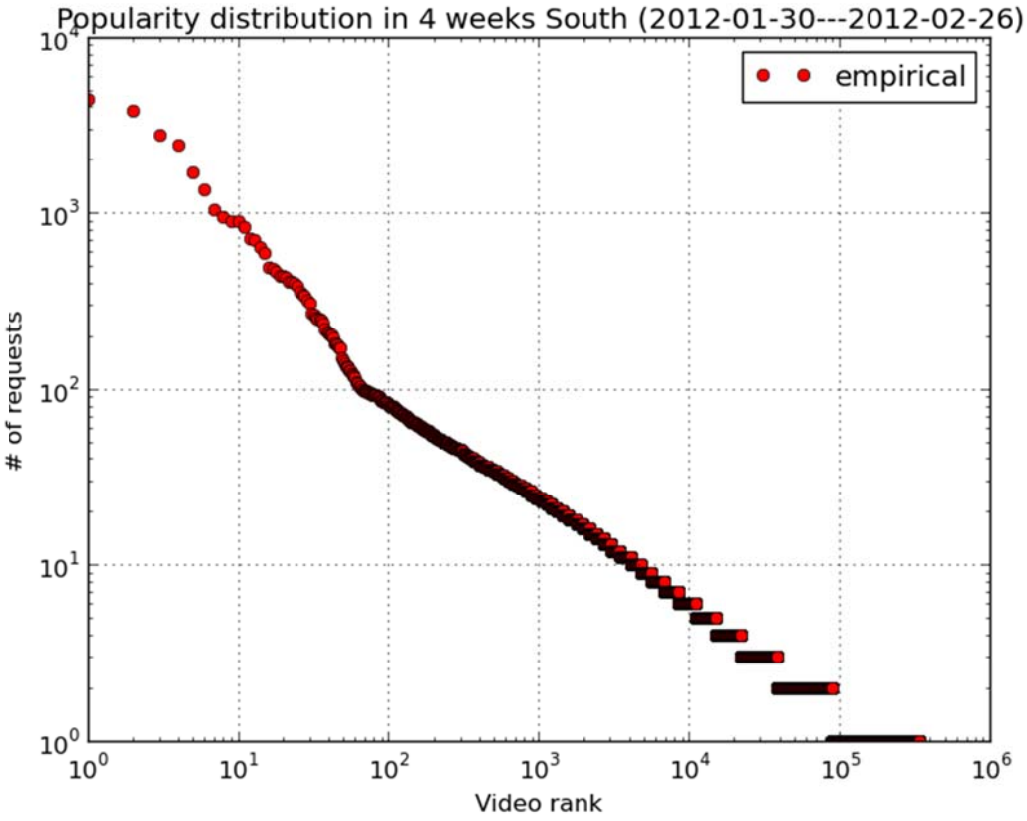

Figure 4-14: Video popularity distribution in the North network

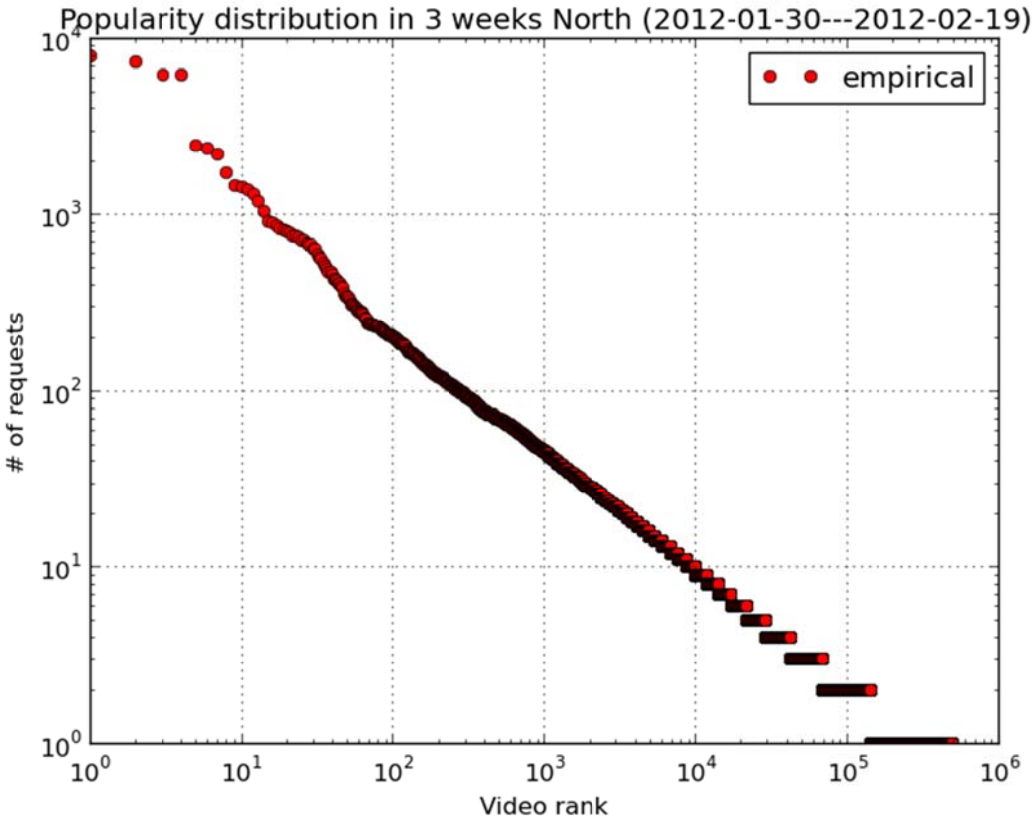

Figure 4-15: Video popularity distribution in the South network

Looking at how video clip popularity decreases, the first week's data was used as a reference. In Figure 4-16 and Figure 4-17 the dark blue bar indicates the total number of requests and the red bar shows the number of unique video clips requested by users during the first week. In the second, third, and the fourth weeks (note that the fourth week's data is only available for the south network), the total number of requests consists of the number of requests for the video clips which were requested in the previous weeks and the number of requests for the new video clips which appeared only in the latest week's trace. The same analysis was applied concerning the number of unique video clips (this is shown as the right hand bar in each week). These stacked bars graphs in Figure 4-16 and Figure 4-17 indicates that, the total number of requests and unique video clips showed consistency over these weeks. In both networks, users watched almost same number of video clips and made roughly same number of requests every week. The users' replay behavior derived from this analysis shows high stability. It is obvious that during each week, users watch a certain number of new video clips, their interest in the video clips which have already been watched drops quickly after one week. However, there are a small amount of popular video clips which people would like to watch it again and again over many weeks. Despite of the difference in size of the north and south network, they both exhibit similar user behavior.

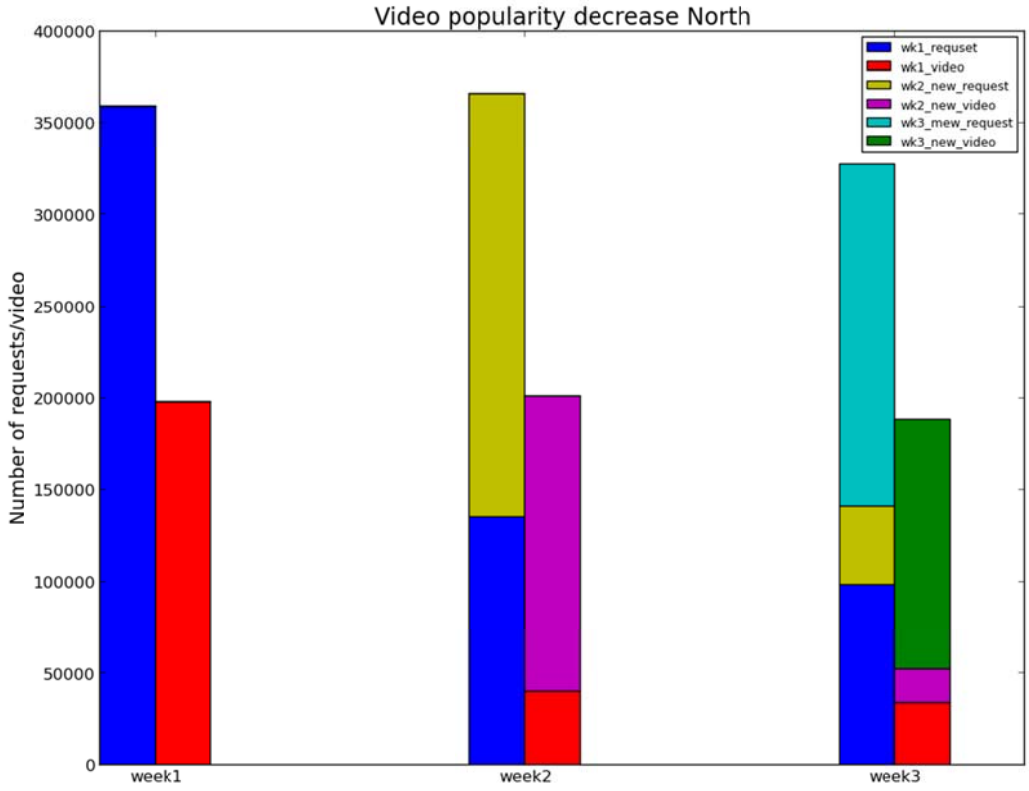

Figure 4-16: Video popularity decrease in the North network

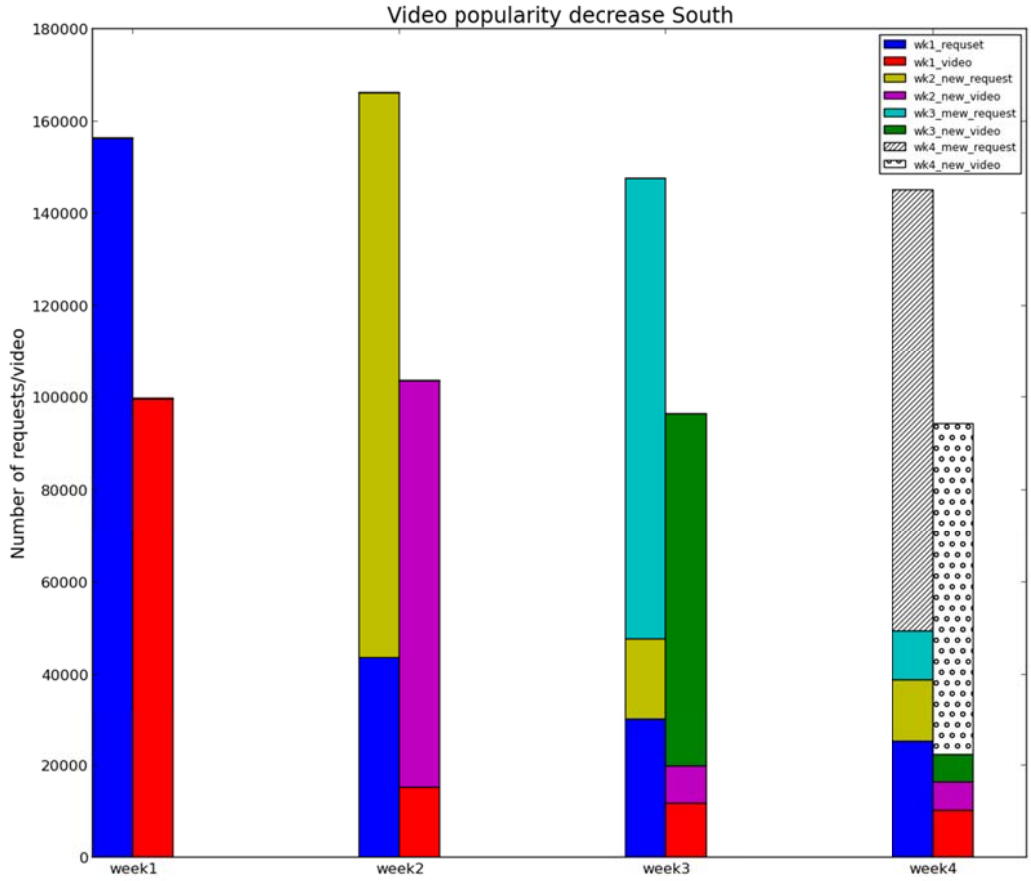

Figure 4-17: Video popularity decrease in the South network

Based on this phenomenon, instead of using an infinite cache, a weekly cache model is proposed. This cache will be emptied every week and will only cache new video clips, i.e., those which were not watched in the previous week. The north network was chosen to do this analysis due to the fact that it has almost twice the number of requests and number of unique video clips compared to the south network. We will focus only on the gross gain of the proxy cache here. The video clips and requests from the first week are used as reference and a hash table was created to record the video clip's ID. The gross gain of the second week per area is shown in Figure 4-18. If the cache stores everything during this week, then the gross gain ranges from 28% to 37%. However, if after a lookup, the video ID can be found in the first week's hash table, then the content will not be stored in the cache and the proxy will forward the request to the server. In this case all the requests for the video clips which have already been requested during the first week will be counted as a cache miss. The red markers indicate the gross gain when the cache only has those video clips which are newly requested during the 2nd week. In Figure 4-19, the red markers represent the gross gain based upon keeping only the new video clips requested during the 3rd week.

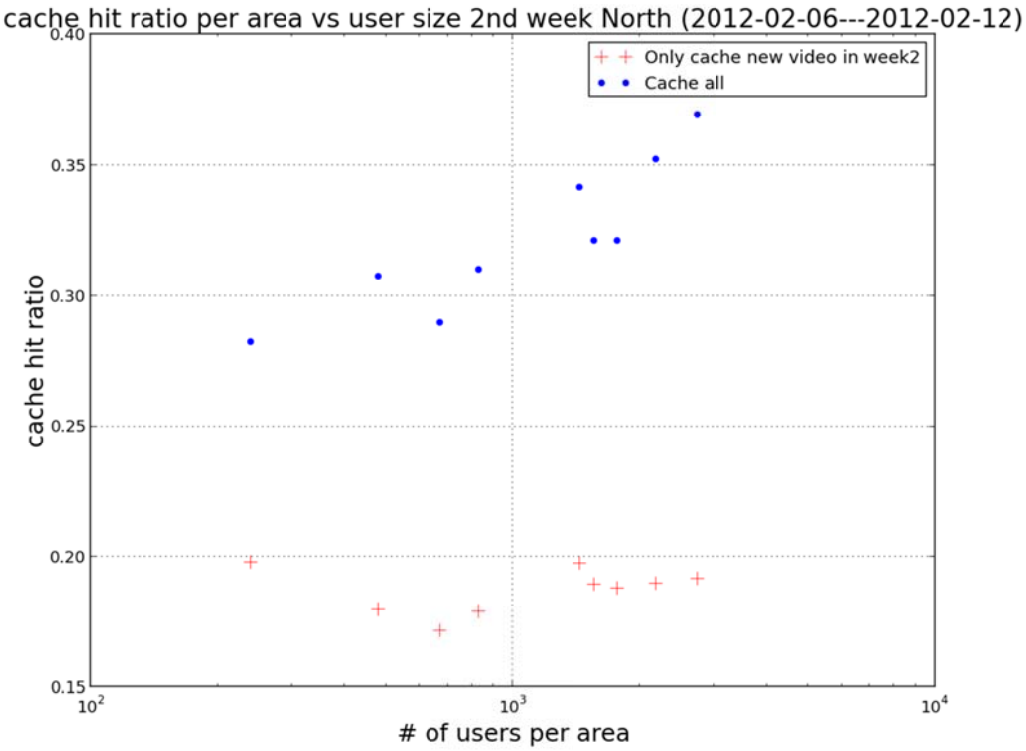

Figure 4-18: Cache hit ratio per area 2nd week North

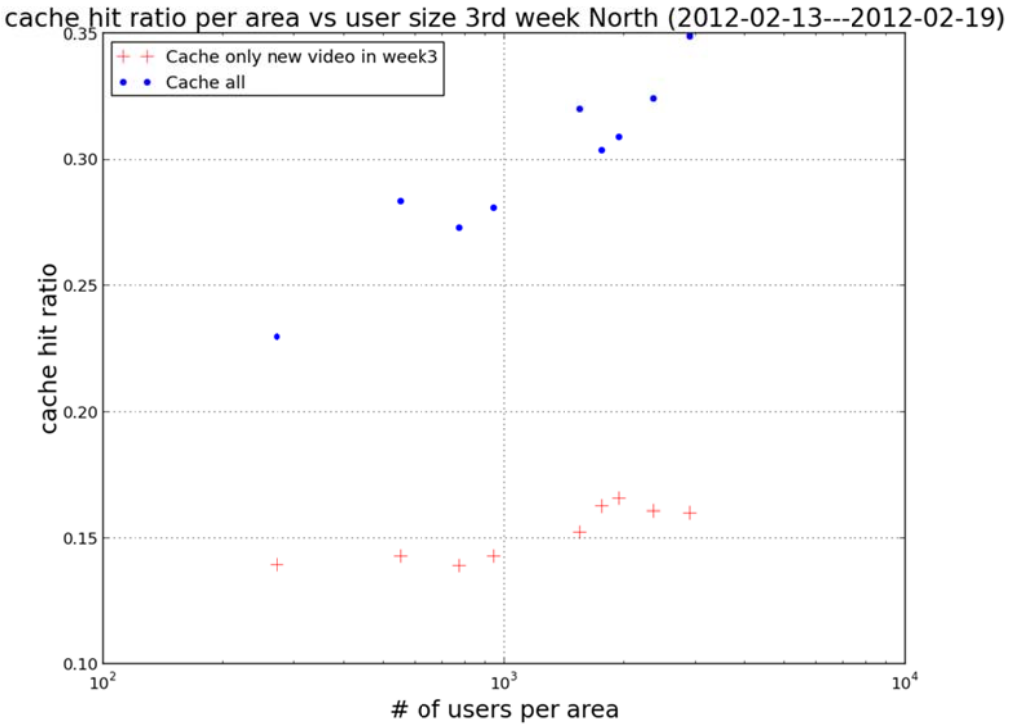

Figure 4-19: Cache hit ratio per area 3rd week North

Caching only the new video clips each week, the gross gain dropped around 10% to 20%. This suggests that the popular video clips which were requested many times over several weeks still contribute a lot to the caching gain. In our dataset, the video clips which were requested during the first week accounted for 20% of the total number of video clips being watched in the second week. The cache gain generated from them accounts for 40% to 50% of the total gain, which means their replay frequency is high. Even though there are many more new video clips requested in each week, quite a few have only been watched once, hence they do **not** contribute to the cache gain at all. Whereas, popular video clips which have a longer lifetime, continue to attract people's attention and are repeatedly requested in the following weeks; thus, it is useful to keep them in the cache.

The proxy cache for each area will be emptied at the end of each week and it will subsequently cache everything during the current week. The gross, terminal, and net gains from this cache strategy are shown in Figure 4-20, Figure 4-21, and Figure 4-22 which respectively represent the 1st, 2nd, and 3rd weeks during the data collection period.

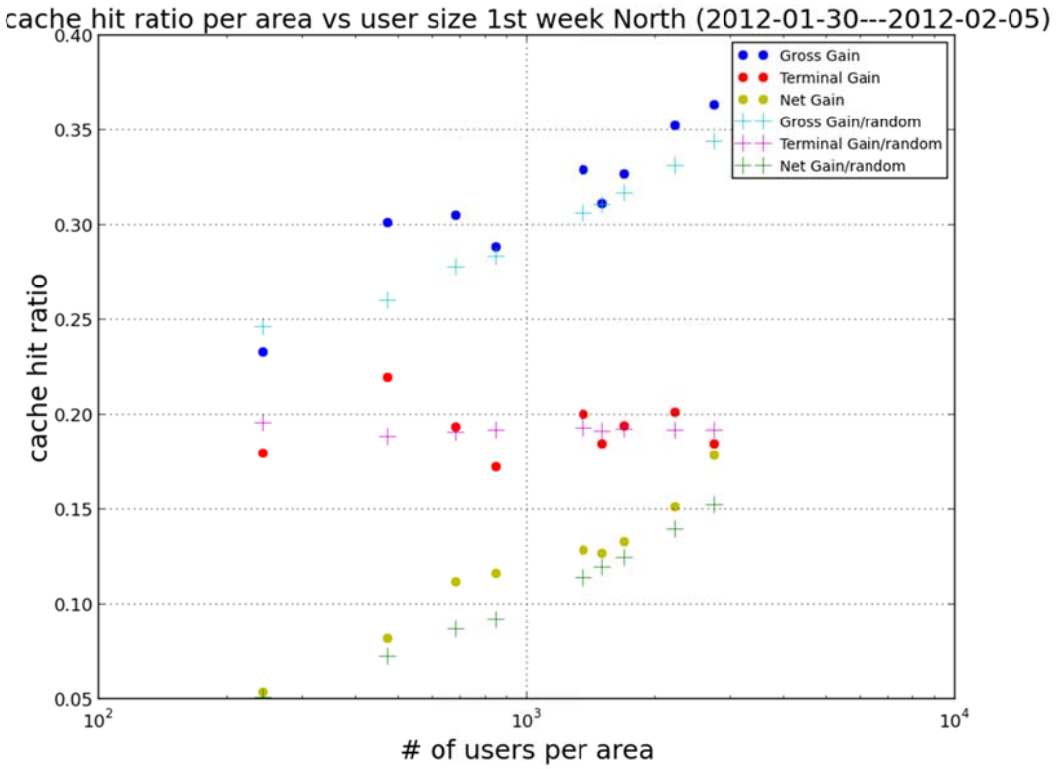

Figure 4-20: Cache hit ratio in the 1st week in the North network

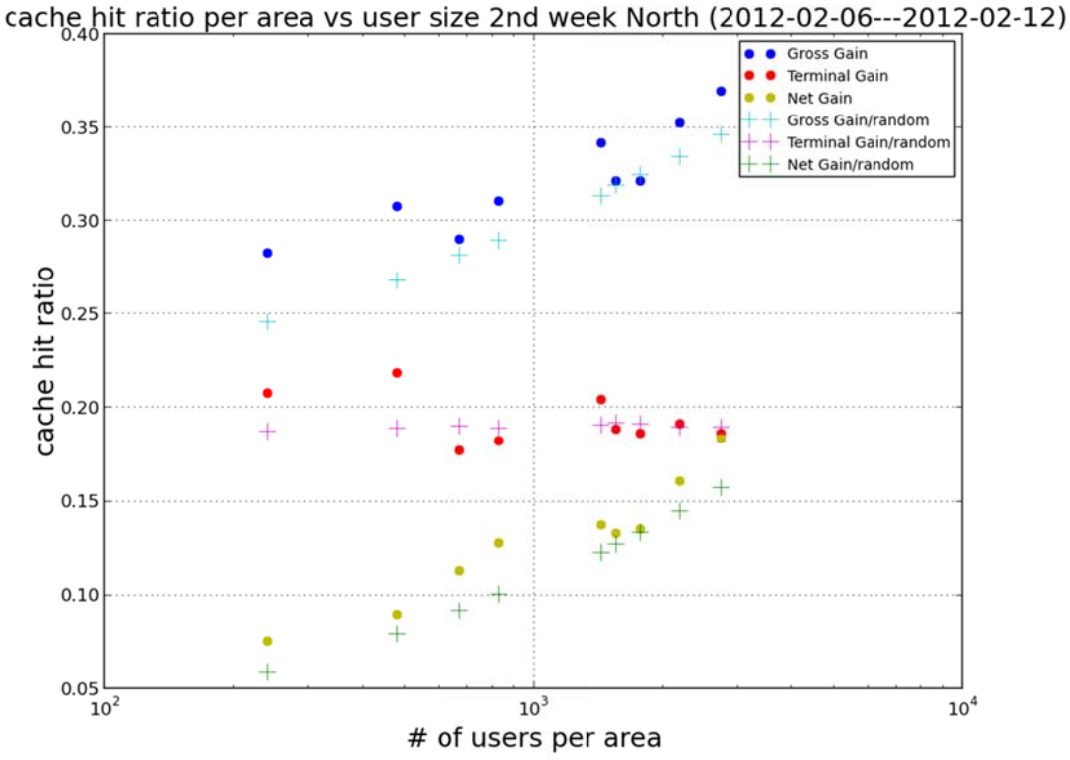

Figure 4-21: Cache hit ratio in the 2nd week in the North network

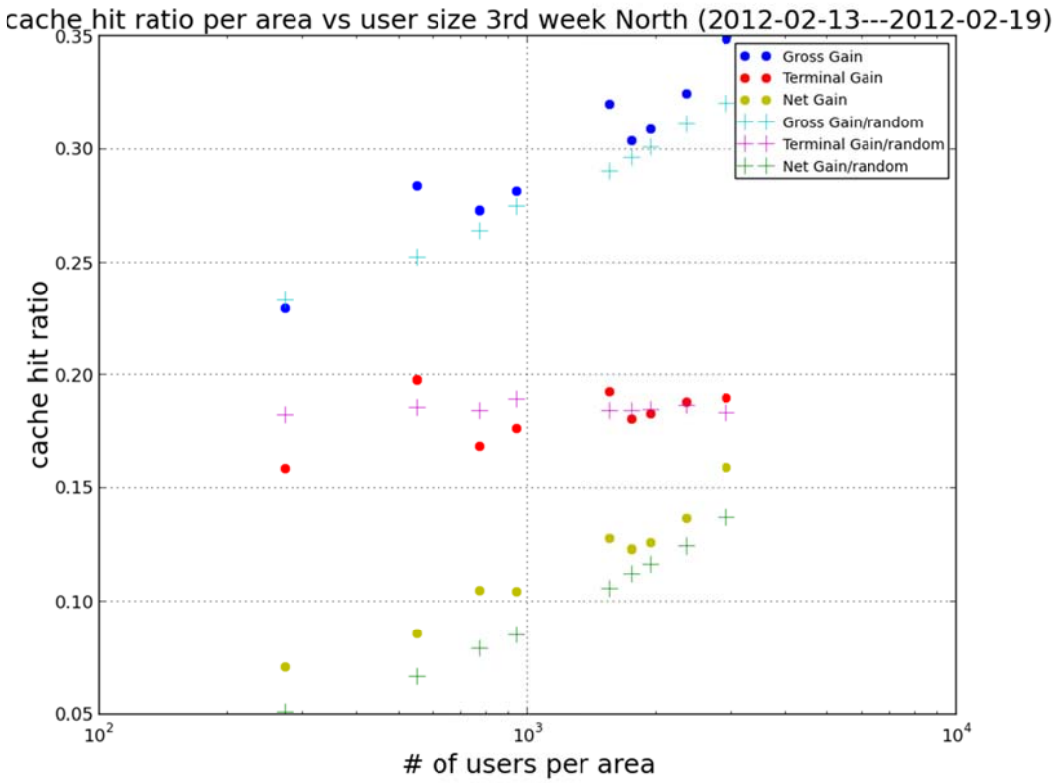

Figure 4-22: Cache hit ratio in the 3rd week in the North network

These result shows that in each of these weeks, each area's gross gain mostly remains in the range of 25% to 35% which is slightly lower than the gain in the case of the three week cache due to the dynamic user behavior during each of these weeks, but it still confirms that the gain for users from a specific area is higher than the randomly selected group - just as the three week cache's result showed.

4.7 Replay pattern

In this section, users' replay patterns are explored further. From the previous studies, the random groups' terminal gain is stable at around 20%, thus the general user replay behavior does not fluctuated and there is no major differences regarding users' replay habits. This section focuses mainly on the temporal pattern of replays.

The replay time interval for one user is defined as the time interval between two sequential requests for the same video clip. The top CDF graphs in Figure 4-23 and Figure 4-24 shows that, if a user watched the same video many times, he or she wants to request it again and again every hour. This kind of replay accounts for 50% of all replay requests. The mean value of one user's replay time interval is around 28 hours, roughly once per day, for both networks.

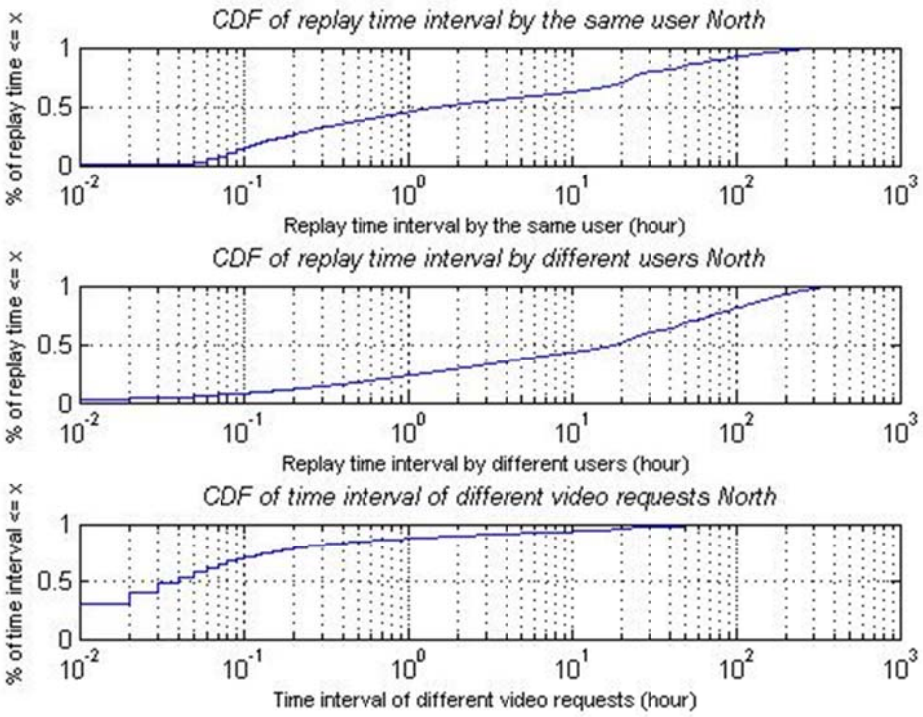

Figure 4-23: CDF of replay time intervals in the North network

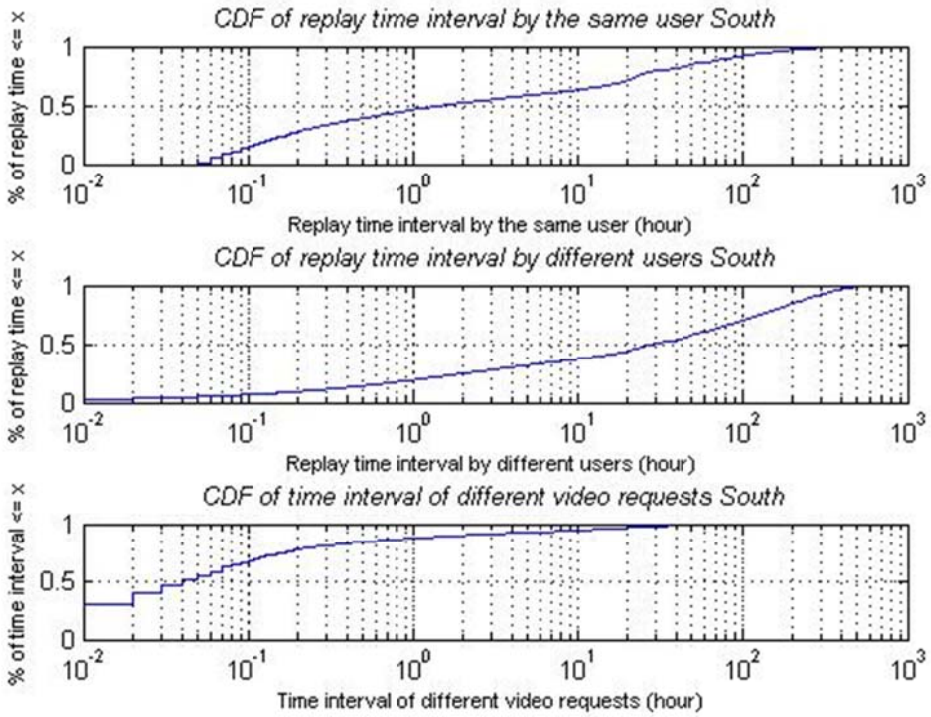

Figure 4-24: CDF of replay time intervals in the South network

The CDF graphs in the middle represent the time interval between requests by two different users for the same video clip. We assume that each end device can cache everything and if requests for the same video clip come from different users within a short time period, then it may be possible to set up a P2P distribution system assuming that there are enough active (i.e., on-line) users who want to watch same video clips. We studied this possibility by measuring the time intervals between requests for the same content. If they are very close to each other and they are in the same small network, then it is feasible to set up a P2P distribution system for small networks which is configured to preferentially select local peers. The time between these users' requests is the time from when one user requests a given video clip until another user requires the same video clip. According to our data, 50% of replays requested by different users happened within 30 hours. Unfortunately, these replay requests are not temporally very close to each other. This suggests that a P2P system is **not** beneficial as it will not reduce the inbound traffic (into the small network).

The bottom graph shows the CDF of time intervals between requests for different video clips from the same user. It indicates how quickly one user changes from the current video clip to another video clip. Since not all video clips will be watched from the beginning to end and the cache, either proxy cache or terminal cache has limited storage, this result helps to determine how we should limit the amount of each video clip that is stored. About 75% of requests occurred within six minutes, with a median value of roughly 2.5 minutes for both networks. This matches the result in [52] as users finished watching a clip within three minutes. To limit the amount of the data that has be stored and to simplify the analysis process, the cache might only keep part of each video clip. However further investigation of user behavior should be done by examining which portion of the video clip are mostly watched by the users. This would make the cache to be more efficient, as it would allow a higher hit rate for the portions of the clips that are watched – by not caching the portions of the video clips that are not watched.

Another interesting question is when the replay occurs. Do people repeatedly request the same video at a specific time or is the replay time randomly distributed? Figure 4-25 and Figure 4-26 show the probability of a replay of all the video clips from the same user since the video clip was firstly requested which have been observed during the data collection period. The x axis is a relative time scale with the time when a video clip is requested as time zero. The y axis shows how many repeated requests for the same video are made by the same user at a time after time x. This replay pattern shows the peak periodically appears every 24 hours which means that popular video clips are watched by the same user every day after he or she first requested that video clip – if they thought it was worth taking a look again. The number of requests drops rapidly during the following 12 hours after the replay happens and reaches the lowest point in 12 hours. This phenomenon suggests that the cache should keep track of the time when each popular video clips' is requested by a user and remove these video clips after 12 hours in order to make space for new video clips – but the cache should put the old video clips back into the cache every 24 hours when most users will watch their favorite video clips again.

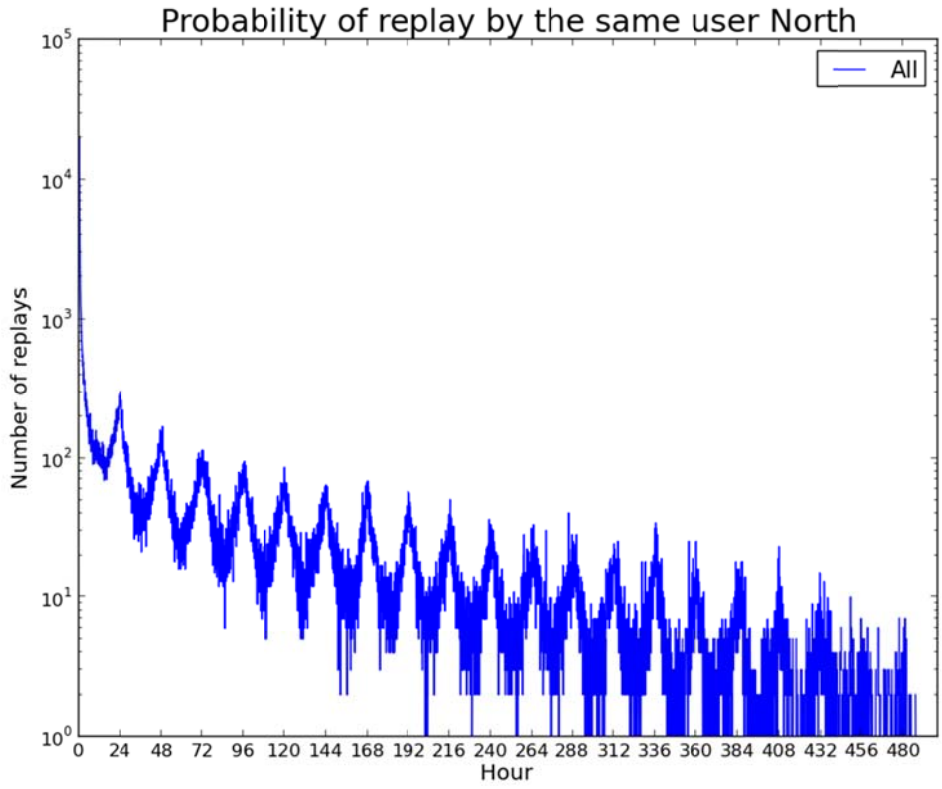

Figure 4-25: Probability of replay by the same user North

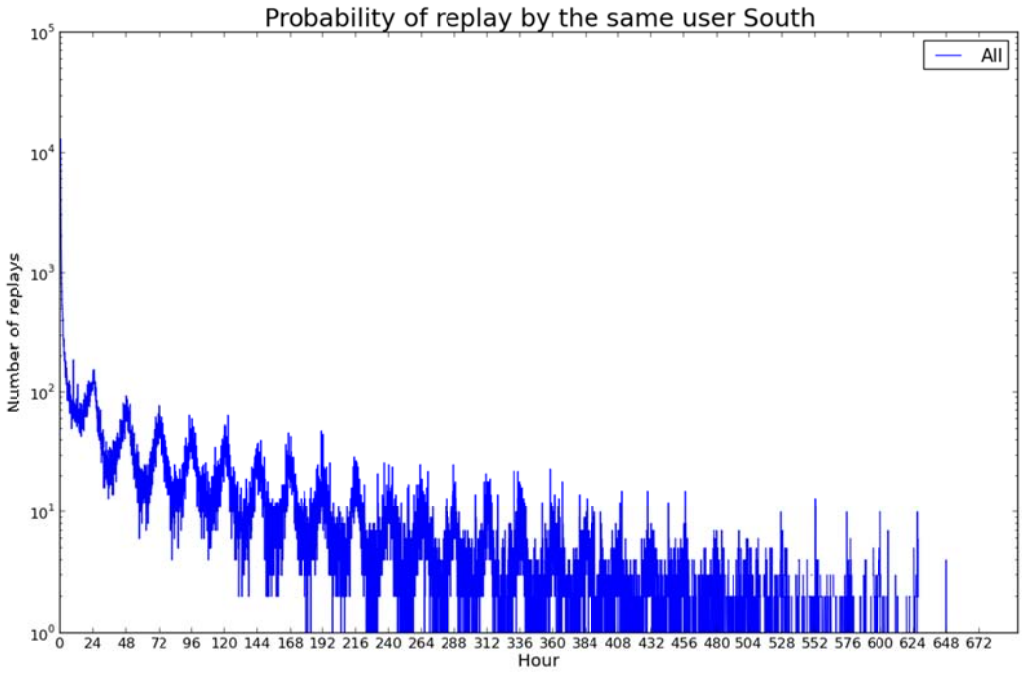

Figure 4-26: Probability of replay the same user South

4.9 Cacheability by user device type

In our dataset the various kinds of devices could be identified by the user agent string included in the requests. However, the same device (such as iPhone or iPad) could be used by many users in a household, the user agent string is not sufficient to identify an end user. For example, if there are identical devices in one household, such as two iPhones and both are using Apple's IOS 5 system, then we cannot distinguish between them and all of the requests will be considered as being from *one* user.

Moreover in this project, a combination of MAC address and user agent string was used as the end device identifier in order to extract more features during the analysis. However, the MAC address is typically either the home router's MAC address or the end device's MAC address when it is directly connected to the home gateway through an Ethernet cable. Even if only one device is connected to the ISP, a using suing this device (typically a PC) can open several browser instances or even use different browsers to watch YouTube videos. In this case, more than one combination of MAC and user agent string will occur and these requests will be considered as coming from *different* users.

Figure 4-27 shows the user device type distribution in both networks. Four kinds of user device types are considered in this project: PC, mobile device, IPTV/Play station, and unknown. Not surprisingly, PCs accounts for more than 70% of the devices. Mobile devices comprise about 20% and 10% of devices in the north and south network (respectively). IPTV and Play station devices were also observed in our dataset, although they comprised less than 1.5% of devices in either network. This indicates that users are starting to use the new devices to request online media **rather than or in addition to** using traditional PCs and mobile devices. However, as there are so few users of these devices in our collected data it is not possible to use this data for statistical analysis. Additionally, there were some user agent strings we could not properly process, these we have placed into the unknown category[*]. To ensure that we have sufficient data to perform statistical analysis this project focused only on two major user device types: PCs and mobile devices.

[*] A list of these unknown user agent strings are listed in Appendix A.

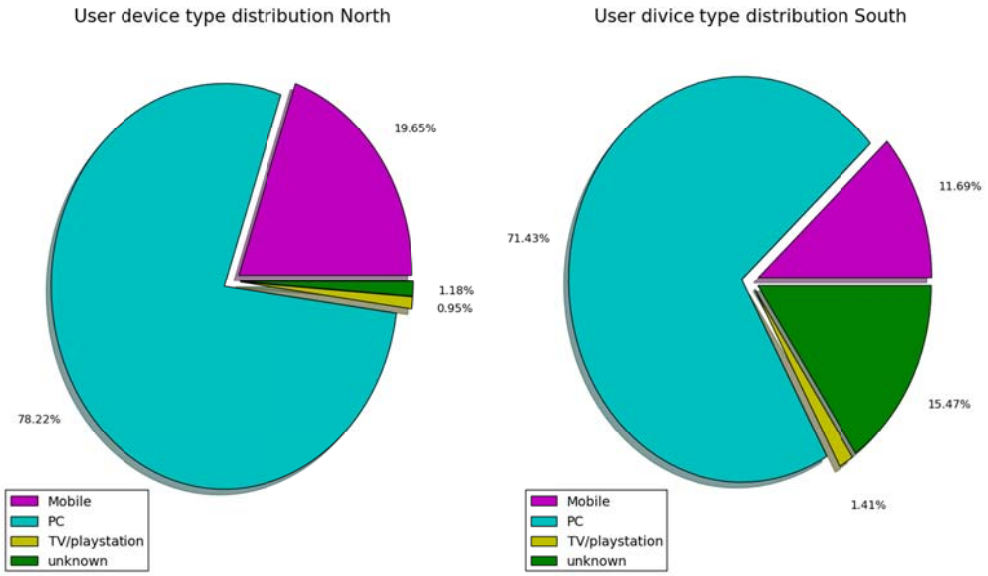

Figure 4-27: User device type distribution

The cache used in this section is an infinite cache both in terms of size and time. The dark blue, red, and yellow markers in Figure 4-28 indicate the mobile devices' gross, terminal, and net gain; while the green, purple, and light blue markers show PC devices' gross, terminal, and net gain. We see that the terminal gain of the mobile devices is always higher than PC devices' terminal gain, which suggests that mobile device users are more likely to repeatedly watching the same content. Examining the net gain, we conclude that PC device users share more common interests than do mobile device users.

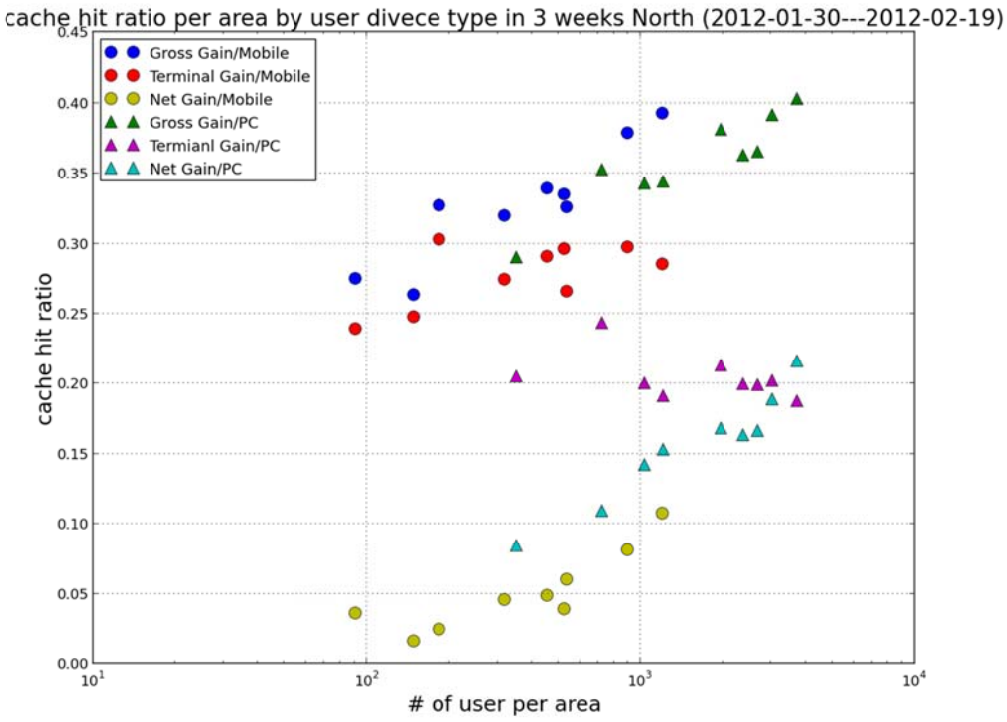

Figure 4-28: Cache gain for PC and mobile devices in the North network

In Figure 4-29 and Figure 4-30, the gain from random selection of PC and mobile device users are compared with the gain from these two groups in each area. From the gross gain and net gain, locality is observed for PC users. However, the users of mobile devices act differently. The gross gain and terminal gain are not always higher than the gain for random groups, although the terminal gain remains between 25% and 30%. Although these graphs show only the data from the north network, the south network shows very similar results. Since the net gain reveals how much gain can be obtained by users' common interests, the P-value is used to measure the difference between the net gain from each area and from each corresponding random group. For the north network, the P-value is 0.112 which means that we do not find that the measured gain is statistically different compared to the random control group.

This suggests that people watch and replay YouTube videos based upon individual interests and people's interests vary even among persons local to a small area. This result suggests enabling terminal caching for each mobile device to reduce backhaul traffic, rather than doing caching in the access point (or base station).

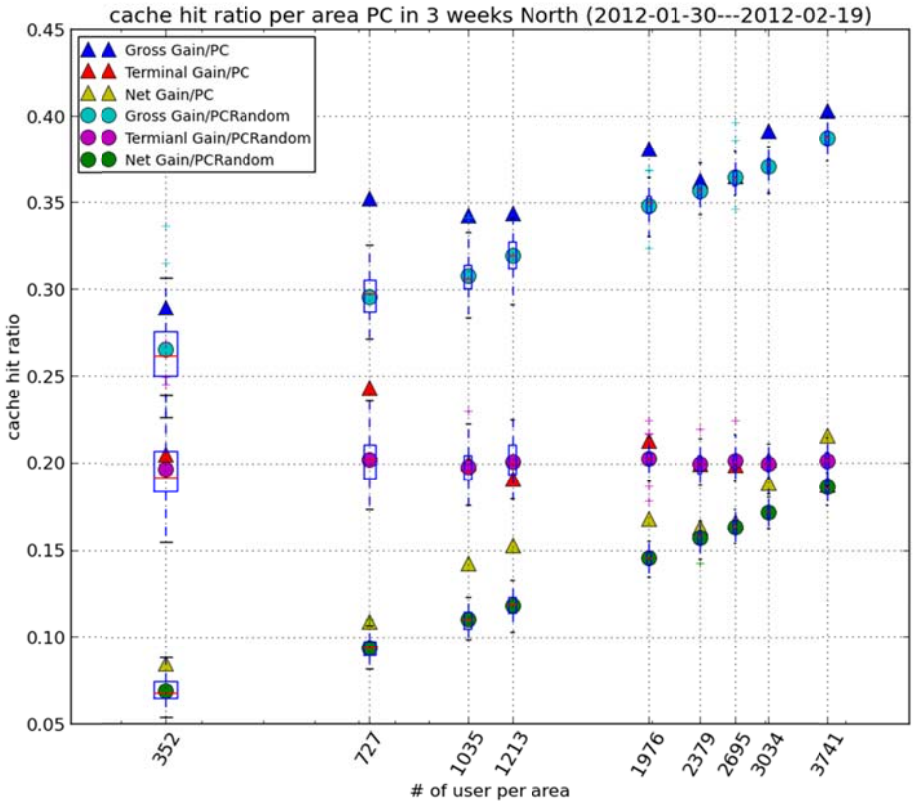

Figure 4-29: Cache gain per area from PC devices in the North network

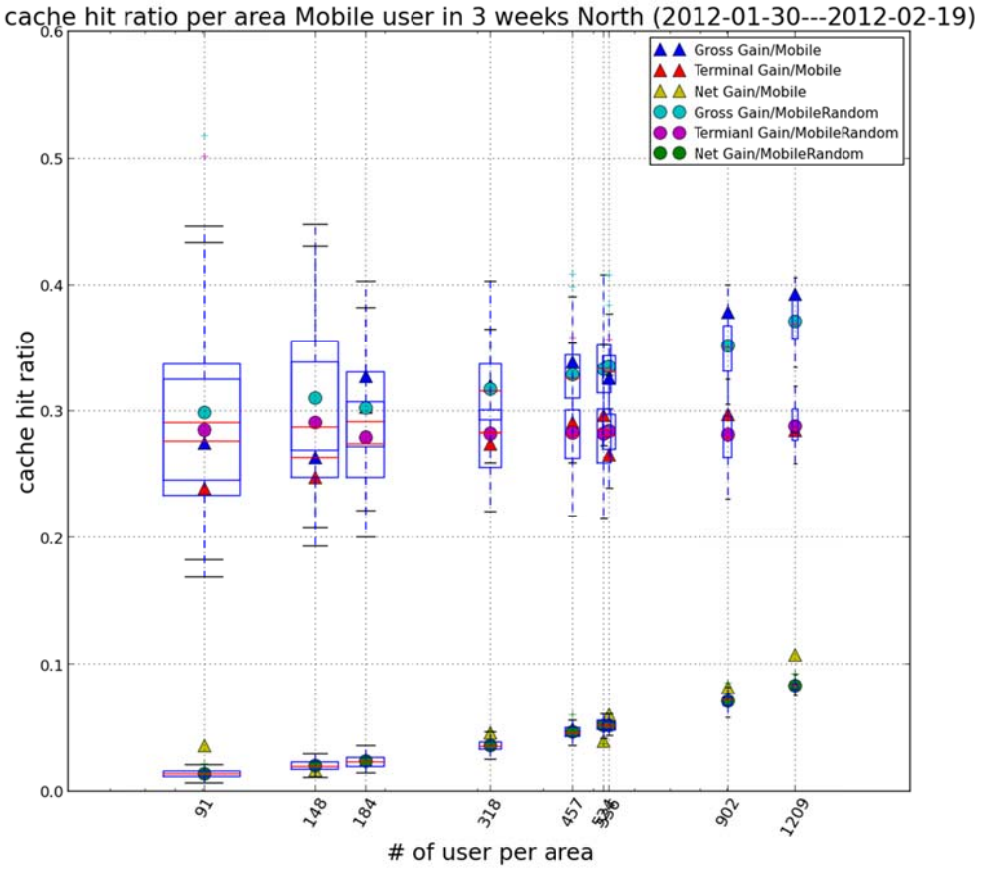

Figure 4-30: Cache gain per area for mobile devices in the North network

5 Conclusions and Future Work

In this chapter, the conclusions of this project are summarized and some possible future work is suggested. The chapter ends with some reflections on economic, social, and ethical issues associated with this thesis project.

5.1 Conclusions

In general, this thesis project contributes to understanding YouTube traffic patterns, user behavior, and the potential caching gains of a proxy cache and terminal cache for two municipal networks in Sweden based on data collected during one month. The methodology used in this project can be easily customized for other similar traffic monitoring, data collection, and analysis; for instance to look at Facebook, SVT, or other highly used traffic source. Unlike previous works, the user identifier used in this project is a hash of the MAC address and user agent string. This combination has been used in an attempt to consistently identify unique user devices while protecting the user's identity. This combined identifier is more precise than using an IP address, especially as the later are dynamically allocated to different devices for a certain lease time and may over the data collection period be allocated to many different devices.

This project demonstrated based upon statistical analysis of YouTube video clips that a large share of requests for video clips are made for a small number of distinct video clips. This phenomenon suggests there is a potential for gains by using caching. The traffic pattern of these requests was summarized by week, day, and hour. Weekends are the days with the greatest number of requests per day and on average the evening time from 18:00 to 22:00 are the peak hours.

The popularity distribution of the video clips was sown to follow a Zipf law distribution. This is easily seen as a straight line on a log-log plot of request frequency versus the ranked order of most popular video clips. This indicates that it may be possible to predict a video clip's popularity, this helping to decide what content should be placed in the cache. However, the popularity drop model per week over the whole data collection time period suggests that only a small number of the video clips will still be popular over a period of weeks. These long term most popular video clips should be kept in the cache in order to get a higher caching gain.

The user replay pattern has also been studied in this project. Only 10% of the users generated more than 100 requests over the entire data collection period indicating that they could be labeled as heavy users. A user replay pattern has been presented which suggests that people switch their interests in video clips quite rapidly, as within 6 minutes half of the users want to watch a different video clip. This tells us that it is no need to cache the entire video clip, but only a small portion of it which requires further analysis to be done to decide which portion of each video should be cached.

Another interesting phenomenon is that people would like to replay the same

content every 24 hours after the first time the video was watched – if they found it interesting. Consequently, the lifetime of each video should be further explored, to decide which content to put into the cache and for how long it should remain in the cache. This is an obvious part of suggested future work.

An infinite sized proxy cache for each graphical area was assumed throughout this project. The cache gain was calculated based upon the hit rate. Our results show that people from the same area share common interests which could be a geographical locality property. This cache gain ranges from 30% to 45%. The cache gain for PC users and mobile device users were analyzed and they showed different user behaviors. One of the important conclusions is that is more useful to enable terminal caching on mobile device rather than set up a proxy cache for small groups of device users in the same area, i.e, proxy caching in the base station or access point will **not** be an efficient solution as it will **not** reduce the backhaul link traffic as much a terminal caching will.

5.2 Future Work

In this master's thesis project, we studied the possible caching gain based on an infinite cache both for a proxy cache and a terminal cache. The next step is to determine the optimal cache size. In our work, a weekly cache has been proposed which is emptied at the end of each week. However, to design a more intelligent cache, the video clip's lifetime should be monitored and a model developed to predict its lifetime. This could be done by tracking newly watched video clips on YouTube within the target network, in order to see how their popularity grows and drops. Based on this information we might have an input that could help to decide how long the content should be kept in the cache. When a new video clip is requested by a user, if the cache is full, then the oldest entry with least number of requests should be deleted to release some space for the new content.

Another question is whether it is necessary to cache the entire video clip. According to our results, over half of the users switch to another video clip within six minutes. This suggests that the viewing time of each video is more important than its actual duration. More accurate analysis could be done by analyzing the TCP sessions of each video clip. The obstacle to this analysis is that video clips are downloaded in chunks and each chunk opens a new TCP session. However, the video ID inside each request could be used to group a set of related TCP sessions, the viewing duration of each video clip could be calculated. This time period could be combined with the encoding rate to determine the amount of a video clip that has been request in numbers of bytes. This could be compared to the total size of the video clip. The encoding rate can be found in the videoplayback request in the itag field.

In addition to geographical locality, understanding content locality is another input to designing a more efficient cache scheme for YouTube video clips. To get more detailed content information about each video clip, the YouTube API could be used to retrieve the metadata based upon the video clip's ID. The video clip's ID can be found in the request sent by the device to the YouTube server as this request starts

with the URL: "*youtube.com/watch?v=*" on a PC and "*m.youtube.com/user_watch?app=*" on a mobile device. An 11-digit string indicates the unique video clip ID after the label "*video_id*". For example a cache could be designed to store a certain category of YouTube clips which has the majority of views. We also noticed that each video clip generally has several related video clips which may have a higher possibility than other video clips that a user will choose them to watch. For this reason pre-caching the beginning of each of these related video clips could potentially reduce the access delay before they can start to play and further reduce traffic when other users in the local network request the same video (as this video clip or at least part of it will now be in the proxy's cache).

As Figure 4-6 and Figure 4-7 shown, on average, the traffic load during the night is very low. Considering the energy consumption of the network and the caching system, this result suggests that the pre-caching could either be implemented during the night to release the burden during the peak hours or be achieved during the peak hours only and shutting down the caching system during the night to save energy.

5.3 Required reflections

This project studies the possibility and the potential benefits of using cache for YouTube system on the purpose of saving the Inter-ISP traffic thus to save the high transit fee. This is especially vital for the mobile operators to cope with the backhaul link limited bandwidth problem. Some of the cost savings for the operators will flow either to the users (in the form of low costs) or to the ISP's shareholders in terms of increased profits. The method which is used to collect, filter and analyze the YouTube data in this project could be easily customized to do other similar studies for media streaming applications. Considering the limitations in the work, the proposal of future work is also provided which could motivate and pave the way for the continuous study.

References

[1] ”Cisco Visual Networking Index: Forecast and Methodology, 2010-2015 [Visual Networking Index] - Cisco Systems”, 28-Okt-2011. [Online]. Available: http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-481360_ns827_Networking_Solutions_White_Paper.html. [Accessed: 28-Okt-2011].

[2] ”Alexa Top 500 Global Sites”. [Online]. Available: http://www.alexa.com/topsites. [Accessed: 17-Sep-2012].

[3] Ruixuan Li, Guoqiang Gao, Weijun Xiao, och Zhiyong Xu, ”Measurement Study on PPLive Based on Channel Popularity”, presented at the Communication Networks and Services Research Conference (CNSR), 2011 Ninth Annual, 2011, ss. 18–25.

[4] Y. Liu, Y. Guo, och C. Liang, ”A survey on peer-to-peer video streaming systems”, *Peer-to-Peer Networking and Applications*, vol. 1, ss. 18–28, Jan 2008.

[5] A. Feldmann, R. Caceres, F. Douglis, G. Glass, och M. Rabinovich, ”Performance of Web proxy caching in heterogeneous bandwidth environments”, presented at the IEEE INFOCOM ’99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings, 1999, vol. 1, ss. 107–116 vol.1.

[6] N. Leibowitz, A. Bergman, R. Ben-shaul, och A. Shavit, ”Are file swapping networks cacheable? characterizing p2p traffic”, *IN PROC. OF THE 7TH INT. WWW CACHING WORKSHOP*, 2002.

[7] B. Ager, F. Schneider, Juhoon Kim, och A. Feldmann, ”Revisiting Cacheability in Times of User Generated Content”, presented at the INFOCOM IEEE Conference on Computer Communications Workshops , 2010, 2010, ss. 1–6.

[8] M. Zink, K. Suh, Y. Gu, och J. Kurose, ”Watch global, cache local: YouTube network traffic at a campus network: measurements and implications”, 2008, vol. 6818, ss. 681805–681805–13.

[9] F. Lehrieder, G. Dán, T. Hossfeld, S. Oechsner, och V. Singeorzan, ”The Impact of Caching on BitTorrent-Like Peer-to-Peer Systems”, presented at the 2010 IEEE Tenth International Conference on Peer-to-Peer Computing (P2P), 2010, ss. 1–10.

[10] Å. Arvidsson, A. Mihály, och L. Westberg, ”Optimised local caching in cellular mobile networks”, *Computer Networks*, vol. 55, num. 18, ss. 4101–4111, Dec 2011.

[11] N. Chand, R. C. Joshi, och M. Misra, ”Cooperative Caching Strategy in Mobile Ad Hoc Networks Based on Clusters”, *Wireless Personal Communications*, vol. 43, ss. 41–63, Dec 2006.

[12] N. Chand, R. Joshi, och M. Misra, ”Broadcast Based Cache Invalidation and Prefetching in Mobile Environment”, in *High Performance Computing - HiPC 2004*, vol. 3296, L. Bougé och V. K. Prasanna, Reds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, ss. 410–419.

[13] G. Cao, ”A scalable low-latency cache invalidation strategy for mobile environments”, New York, NY, USA, 2000, ss. 200–209.

[14] ”TRAMMS - Traffic Measurements and Models in Multi-Service Networks, winner of the Celtic Excellence Award in Gold 2010”, 03-Nov-2011. [Online]. Available: http://projects.celtic-initiative.org/tramms/. [Accessed: 03-Nov-2011].

[15] ”IPNQSIS - IP Network Monitoring for Quality of Service Internet Support”, 03-Nov-2011. [Online]. Available: http://projects.celtic-initiative.org/ipnqsis/. [Accessed: 03-Nov-2011].

[16] ”TM Forum - Technical Reports - TR148, Managing the Quality of Customer Experience, Release 1.0 - Footer 2011”, 15-Nov-2011. [Online]. Available: http://www.tmforum.org/TechnicalReports/TR148Managingthe/38504/article.html. [Accessed: 15-Nov-2011].

[17] ”TM Forum - Technical Reports - TR149, Holistic e2e Customer Experience Framework, Release 1.0 - Footer 2011”, 15-Nov-2011. [Online]. Available: http://www.tmforum.org/TechnicalReports/TR149Holistice2e/38508/article.html. [Accessed: 15-Nov-2011].

[18] Phat Hoang, ”Internet Traffic Analysis”, Master Thesis, Lund University, Lund, Sweden, 2010.

[19] M. Kihl, P. Ödling, C. Lagerstedt, och A. Aurelius, ”Traffic analysis and characterization of Internet user behavior”, presented at the 2010 International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), 2010, ss. 224–231.

[20] Yichi Zhang, ”Residential network traffic and user behavior analysis”, Master Thesis, KTH Royal Institute of Technology, Stockholm, Sweden, 2010.

[21] A. Aurelius, C. Lagerstedt, och M. Kihl, ”Streaming media over the Internet: Flow based analysis in live access networks”, presented at the 2011 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB), 2011, ss. 1–6.

[22] G. Maier, A. Feldmann, V. Paxson, och M. Allman, ”On dominant characteristics of residential broadband internet traffic”, New York, NY, USA, 2009, ss. 90–102.

[23] P. Gill, M. Arlitt, Z. Li, och A. Mahanti, ”Youtube traffic characterization: a view from the edge”, New York, NY, USA, 2007, ss. 15–28.

[24] J. Pouwelse, P. Garbacki, D. Epema, och H. Sips, ”The Bittorrent P2P File-Sharing System: Measurements and Analysis”, in *Peer-to-Peer Systems IV*, vol. 3640, M. Castro och R. Renesse, Reds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, ss. 205–216.

[25] T. Karagiannis, A. Broido, N. Brownlee, K. C. Claffy, och M. Faloutsos, ”Is

P2P dying or just hiding? [P2P traffic measurement]”, presented at the IEEE Global Telecommunications Conference, 2004. GLOBECOM ’04, 2004, vol. 3, ss. 1532– 1538 Vol.3.

[26] T. Karagiannis, A. Broido, N. Brownlee, K. Claffy, och M. Faloutsos, ”{File-sharing in the Internet: A characterization of P2P traffic in the backbone}”, *University of California, Riverside, USA, Tech. Rep*, 2003.

[27] G. Barish och K. Obraczke, ”World Wide Web caching: trends and techniques”, *IEEE Communications Magazine*, vol. 38, num. 5, ss. 178–184, Maj 2000.

[28] B. D. Davison, ”A Survey of Proxy Cache Evaluation Techniques”, *IN PROCEEDINGS OF THE FOURTH INTERNATIONAL WEB CACHING WORKSHOP (WCW99*, ss. 67–77, 1999.

[29] B. D. Davison, ”The design and evaluation of web prefetching and caching techniques”, Rutgers University, New Brunswick, NJ, USA, 2002.

[30] H. Hassanein, Zhengang Liang, och P. Martin, ”Performance comparison of alternative Web caching techniques”, presented at the Seventh International Symposium on Computers and Communications, 2002. Proceedings. ISCC 2002, 2002, ss. 213– 218.

[31] O. Saleh och M. Hefeeda, ”Modeling and Caching of Peer-to-Peer Traffic”, presented at the Proceedings of the 2006 14th IEEE International Conference on Network Protocols, 2006. ICNP ’06, 2006, ss. 249–258.

[32] ”How To Measure The Performance Of A Caching Dns Server”, 21-Nov-2011. [Online]. Available: http://destinationebooks.com/how-to-measure-the-performance-of-a-caching-dns-server.html. [Accessed: 21-Nov-2011].

[33] ”Caching Proxies: Limitations and Potentials”, 06-Nov-2011. [Online]. Available: http://www.w3.org/Conferences/WWW4/Papers/155/. [Accessed: 06-Nov-2011].

[34] Ayodele Damola, ”Peer to peer networking in Ethernet broadband access networks”, Master Thesis, KTH Royal Institute of Technology, Stockholm, Sweden, 2005.

[35] D. R. Choffnes och F. E. Bustamante, ”Taming the torrent: a practical approach to reducing cross-isp traffic in peer-to-peer systems”, New York, NY, USA, 2008, ss. 363–374.

[36] R. Bindal, Pei Cao, W. Chan, J. Medved, G. Suwala, T. Bates, och A. Zhang, ”Improving Traffic Locality in BitTorrent via Biased Neighbor Selection”, presented at the 26th IEEE International Conference on Distributed Computing Systems, 2006. ICDCS 2006, 2006, ss. 66– 66.

[37] S. B. Handurukande, A.-M. Kermarrec, F. Le Fessant, L. Massoulié, och S. Patarin, ”Peer sharing behaviour in the eDonkey network, and implications for the design of server-less file sharing systems”, New York, NY, USA, 2006, ss. 359–371.

[38] F. Fessant, S. Handurukande, A.-M. Kermarrec, och L. Massoulié, ”Clustering in Peer-to-Peer File Sharing Workloads”, in

Peer-to-Peer Systems III, vol. 3279, G. M. Voelker och S. Shenker, Reds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, ss. 217–226.

[39] A. Crespo och H. Garcia-Molina, ”Semantic Overlay Networks for P2P Systems”, in *Agents and Peer-to-Peer Computing*, vol. 3601, G. Moro, S. Bergamaschi, och K. Aberer, Reds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, ss. 1–13.

[40] M. R. Lindsey, M. Papadopouli, F. Chinchilla, och A. Singh, ”Measurement and Analysis of the Spatial Locality of Wireless Information and Mobility Patterns in a Campus”, 2003.

[41] P. Cao och S. Irani, ”Cost-aware WWW proxy caching algorithms”, Berkeley, CA, USA, 1997, ss. 18–18.

[42] S. Roy och Z. Zhang, ”Reducing ISP cost by caching of p2p traffic”, Purdue University, West Lafayette, Indiana, USA, Course Project.

[43] J. M. Almeida, ”Hybrid caching strategy for streaming media files”, 2000, vol. 4312, ss. 200–212.

[44] L. Adamic och B. Huberman, ”{Zipf’s law and the Internet}”, *Glottometrics*, vol. 3, ss. 143–150, 2002.

[45] ”Zipf’s Law and Its Role in Web Caching”, in *Web Caching and its Applications*, vol. 772, Boston: Kluwer Academic Publishers, 2011, ss. 165–167.

[46] V. K. Adhikari, S. Jain, och Zhi-Li Zhang, ”Where Do You Tube? Uncovering YouTube Server Selection Strategy”, presented at the 2011 Proceedings of 20th International Conference on Computer Communications and Networks (ICCCN), 2011, ss. 1–6.

[47] V. K. Adhikari, S. Jain, Y. Chen, och Z.-L. Zhang, ”How do you ’Tube’”, New York, NY, USA, 2011, ss. 137–138.

[48] V. K. Adhikari, S. Jain, och Z.-L. Zhang, ”YouTube traffic dynamics and its interplay with a tier-1 ISP: an ISP perspective”, New York, NY, USA, 2010, ss. 431–443.

[49] R. Torres, A. Finamore, J. R. Kim, M. Mellia, M. M. Munafo, och S. Rao, ”Dissecting Video Server Selection Strategies in the YouTube CDN”, in *Proceedings of the 2011 31st International Conference on Distributed Computing Systems*, Washington, DC, USA, 2011, ss. 248–257.

[50] M. Cha, H. Kwak, P. Rodriguez, Y.-Y. Ahn, och S. Moon, ”I tube, you tube, everybody tubes: analyzing the world’s largest user generated content video system”, in *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, New York, NY, USA, 2007, ss. 1–14.

[51] Xu Cheng, C. Dale, och Jiangchuan Liu, ”Statistics and Social Network of YouTube Videos”, presented at the 16th International Workshop on Quality of Service, 2008. IWQoS 2008, 2008, ss. 229–238.

[52] A. Finamore, M. Mellia, M. M. Munafò, R. Torres, och S. G. Rao, ”YouTube everywhere: impact of device and infrastructure synergies on user experience”, New York, NY, USA, 2011, ss. 345–360.

[53] M. Z. Shafiq, L. Ji, A. X. Liu, och J. Wang, ”Characterizing and modeling

internet traffic dynamics of cellular devices”, in *Proceedings of the ACM SIGMETRICS joint international conference on Measurement and modeling of computer systems*, New York, NY, USA, 2011, ss. 305–316.

[54] G. Maier, F. Schneider, och A. Feldmann, ”A first look at mobile hand-held device traffic”, in *Proceedings of the 11th international conference on Passive and active measurement*, Berlin, Heidelberg, 2010, ss. 161–170.

[55] A. Gember, A. Anand, och A. Akella, ”A comparative study of handheld and non-handheld traffic in campus Wi-Fi networks”, in *Proceedings of the 12th international conference on Passive and active measurement*, Berlin, Heidelberg, 2011, ss. 173–183.

[56] ”DRDL Technology”, 04-Jan-2012. [Online]. Available: http://www.proceranetworks.com/en/products/drdl-technology.html. [Accessed: 04-Jan-2012].

Appendix A – Unknown user agent strings

- 3gpp-gba YouTubeMobile/2.4.4 (gzip)
- curl/7.16.3 (Linux 2.6.28 intel.ce4100 dlink.dsm380 i686; en-US; beta) boxee/1.5.0.23615-80d5f5b
- curl/7.20.1 (mipsel-unknown-linux-gnu) libcurl/7.20.1 OpenSSL/0.9.8o zlib/1.2.3 libidn/1.19
- DNTG-HTTPC/1.1
- Dreambox HTTP Downloader
- Dream Multimedia Dreambox Enigma2 Mediaplayer
- Mozilla/3.0
- Evom/0.99m CFNetwork/520.3.2 Darwin/11.3.0 (x86_64) (MacBookAir3%2C2)
- Free%20YouTube%20Downloader/2.2.0 CFNetwork/520.0.13 Darwin/11.1.0 (x86_64) (MacBookPro7%2C1)
- Free%20YouTube%20Downloader/2.3.0 CFNetwork/520.3.2 Darwin/11.3.0 (x86_64) (MacBookPro5%2C1)
- GMF/
- Gnash-0.8.10dev
- Gnash-0.8.7
- Gnash-0.8.8
- Lavf52.110.0
- Lavf52.34.0
- Lavf52.78.5
- libcurl-agent/1.0
- libsoup/2.26.3
- Lynx/2.8.7dev.4 libwww-FM/2.14 SSL-MM/1.4.1 OpenSSL/0.9.8d
- Mozilla/4.0 (compatible;)
- Mozilla/4.0 (compatible; NativeHost)
- Mozilla/4.0 (Windows 7 6.1) Java/1.6.0_24
- Mozilla/4.0 (Windows 7 6.1) Java/1.6.0_25
- Mozilla/4.0 (Windows 7 6.1) Java/1.6.0_29
- Mozilla/5.0
- Mozilla/5.001 (windows; U; NT4.0; en-US; rv:1.0) Gecko/25250101
- \"Mozilla/5.0 (compatible; AMI/1.0)\"
- Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.9.0.17) Gecko Miro/4.0.5 (http://www.getmiro.com/)
- Mozilla/5.0 (Windows; U; Windows NT 6.0; en-US; rv:1.9.2.15) Gecko/20090226
- Mozilla/5.0 (Windows; U; Windows NT 6.1; en-US; rv:1.9.0.17) Gecko Miro/4.0.6 (http://www.getmiro.com/)

- Mozilla/5.0 (Windows; U; Windows NT 6.1; en-US; rv:1.9.1.15) Gecko/20090226
- Mozilla/5.0 (X11; U; Linux MIPS32; pt_PT) AZBox
- MPlayer/SVN-r30099-4.2.1
- MSDL (comptible; LG NetCast.Media-2011)
- NetSurf/1.2 (NetBSD; amd64)
- Plex/10.1 Git:c0d3aeb (Mac OS X; 11.3.0 x86_64; http://www.plexapp.com)
- Plex/10.1 Git:Unknown (Windows; Windows 7, 64-bit (WoW) Service Pack 1 build 7601; http://www.plexapp.com)
- Primeport
- PVPLAYER 04.07.00.01
- SONY BD/2010; M03.R.769
- SONY BD/2010; M04.R.787
- SONY BD/2011; M07.R.0579
- Streamium/1.0
- Touch Diamond2 T5353 / YouTube-3.0
- Update_Detector
- Wget
- Wget/1.12
- YouTube/1.0 CFNetwork/548.0.4 Darwin/11.0.0
- YouTubeMobile/2.4.4 (gzip)
- YouTubeMobile/2.4.9 (gzip)

TRITA-ICT-EX-2012:246

www.kth.se