UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Departament d'Arquitectura de Computadors

Internet Sharing in Community Networks

EMMANOUIL DIMOGERONTAKIS

UNIVERSITAT POLITÈCNICA DE CATALUNYA
Department of Computer Architecture

Internet Sharing in Community Networks

BY
EMMANOUIL DIMOGERONTAKIS

March 2017

ADVISORS
DR. LEANDRO NAVARRO
DR. ROC MESEGUER
DR. LUIS VEIGA

COMPUTER NETWORKS AND DISTRIBUTED SYSTEMS GROUP
DEPARTMENT OF COMPUTER ARCHITECTURE
UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONA
SPAIN

Dissertation for the degree of Doctor of Philosophy
at Universitat Politècnica de Catalunya in Computer Architecture
with specialization in Communication Networks and Distributed Systems

TO ALL THOSE HOURS,
THAT WERE NOT SPENT
WITH THE PEOPLE AND CAUSES
THAT DESERVED IT.

Internet Sharing in Community Networks. *March 2017.*

Emmanouil Dimogerontakis
edimoger@ac.upc.edu

Computer Networks and Distributed Systems Group
Universitat Politècnica de Catalunya
Jordi Girona, 1-3, C6,D6
08034 - Barcelona, Spain

Acknowledgments

I am hugely indebted to my advisors, Leandro Navarro, Roc Meseguer and Luís Veiga, without whose guidance and support, this journey would never have been possible.

I would like to thank wholeheartedly all the people that assisted me indirectly, my friends Melanie, Dimitrios, Panos, Stefano, Aggelos, Tassos, Alex, Ioanna, Tassilo, Igor, Xino and my family Eroika, Vassilis and Aleka.

I would also like to thank my colleagues and friends Mennan Selimi (UPC), João Neto (UPC), Roger P. Centelles (Guifi), Roger Baig (Guifi), Roshan Shedar (UPC), Navaneeth Rameshan (UPC) and Amin Khan (IST) who were a great pleasure to work with. I am also thankful to all my colleagues at Distributed Systems Groups at UPC who provided a great work environment, all the C6-E208 people for the many hours of discussions over a custom ping-pong table, all the FIB bar people for their beverages, Roc's family that had to bear my long frequent meetings with him and my colleagues GSD at INESC-ID.

I would also like to thank my co-authors Pere Millan, Carlos Molina (Universitat Rovira i Virgili) and Bart Braem, Chris Blondia (iMinds).

This work was funded by European Commission (EACEA) through the Erasmus Mundus doctoral fellowship, via Erasmus Mundus Joint Doctorate in Distributed Computing (EMJD-DC) programme. This work was also supported by European Community Framework Programme 7 FIRE Initiative project Community Networks Testbed for the Future Internet (CONFINE), FP7-288535, the EU Horizon 2020 Framework Program project netCommons (H2020-688768), the Spanish government under contract TIN2013-47245-C2-1-R., and the Generalitat de Catalunya as a Consolidated Research Group 2014-SGR-881.

Abstract

The majority of the world's population does not have any or adequate Internet access. This implies that the Internet cannot provide universal service, reaching everyone without discrimination. Global access to the Internet for all requires the expansion of network infrastructures and a dramatic reduction in Internet access costs especially in less developed geographical regions. Local communities come together to build their own network infrastructures, known as Community Networks, and provide accessible and affordable local and Internet inter-networking. Sharing resources, such as infrastructure or Internet access, is encouraged at all levels, in order to lower the cost of connectivity and services. Communities can develop their own network infrastructures as a commons, using several interconnected sub-networks when the scale requires it, and sharing several Internet gateways among their participants.

Shared Internet access is offered through web proxy gateways, where individuals or organisations share the full or spare capacity of their Internet connections with other participants. However, these gateway nodes may be overloaded by the demand, and their Internet capacity may degrade due to lack of regulation.

This thesis investigates whether shared Internet access in community networks can be utilized to provide universal Internet access. As a first step in this direction, in this thesis we explored characteristics, limitations and usability of a concrete shared Internet Web proxy service in community networks. Based on our findings we studied and proposed mechanisms to improve the user experience and fairness of Internet sharing Web proxy services in community networks, without introducing significant overhead to the network and other services. More specifically, we proposed a scalable client-side Internet gateway selection mechanism suitable for heterogeneous environments such as community networks. Finally, we studied and proposed techniques for sharing spare Internet capacity without deteriorating the contributors' performance.

Resumen

La mayoría de la población mundial no tiene ningún o un adecuado acceso a Internet. Esto implica que Internet no puede prestar un servicio universal, llegando a todos sin discriminación. El acceso global a Internet para todos requiere una drástica reducción de los costos de acceso a Internet, especialmente en zonas geográficas y poblaciones menos desarrolladas. Las comunidades locales se organizan para construir sus propias infraestructuras de red, conocidas como redes comunitarias, y proporcionan interconexión local y con Internet de forma accesible y asequible. Se fomenta la compartición de recursos, como la infraestructura o el acceso a Internet, para reducir el coste de la conectividad y los servicios. Las comunidades pueden desarrollar sus propias infraestructuras de red como un recurso común, utilizando varias subredes interconectadas dado su tamaño, y compartiendo varias pasarelas de Internet entre sus participantes.

El acceso compartido a Internet se ofrece a través de pasarelas que son proxy web, donde los participantes o las organizaciones comparten la capacidad total o excedente de su conexión a Internet con otros participantes. Sin embargo, estas pasarelas pueden saturarse por la demanda, y su capacidad de acceso a Internet se puede degradar debido la falta de regulación.

Esta tesis investiga si las redes comunitarias se pueden utilizar para proporcionar acceso universal a Internet. Como primer paso en esta dirección, exploramos las características, limitaciones y usabilidad de un servicio concreto de acceso compartido a Internet con proxies web en una red comunitaria. Sobre la base de nuestros hallazgos, estudiamos y proponemos mecanismos para mejorar la experiencia del usuario y la equitatividad de la compartición, sin introducir una sobrecarga significativa en la red y a otros servicios. Más específicamente, proponemos un mecanismo escalable de selección de pasarela a Internet del lado del cliente, adecuado para entornos heterogéneos como las redes comunitarias. Además, estudiamos y proponemos técnicas para compartir la capacidad de Internet sin deteriorar el desempeño de los participantes que contribuyen.

Contents

List of Figures

List of Tables

List of Acronyms

CDN Content Distribution Network

CN Community Network

CNML Community Network Markup Language

ECDF Empirical Cumulative Distribution Function

ISP Internet Service Provider

ITU International Telecommunication Union

RTT Round-Trip Time

SNMP Simple Network Management Protocol

TTFB Time To First Byte

VPN Virtual Private Network

WMN Wireless Mesh Network

Chapter 1

Introduction

-

Internet access has become a de facto requirement to participate in society; for instance, to access public services, education material, social media and also to support the everyday work of millions of organizations. However, the majority of the world's population is not yet online [Int15], far from the vision of a "universal service". Among many others, a major obstacle in providing Internet access to everyone is the involved cost, as also implied by the relation between the GDP per capita and Internet access as shown in Figure 1.1. Therefore, while the Internet is intended for everyone [Cer02], as Vint Cerf says: *"it won't be if it isn't affordable by all that wish to partake of its services, so we must dedicate ourselves to making the Internet as affordable as other infrastructures so critical to our well-being"*. Global access to the Internet for all requires a dramatic reduction in Internet access costs especially in geographies and populations with low infrastructural penetration [GAI16]. This situation widens the digital divide between several communities, regions, countries, and the rest of the world. In an effort to understand and improve the current situation we posed the following problem as our overall research question:

How to provide access to the Internet for all?

In the Internet market realm various initiatives from different backgrounds are working in that direction. Important Internet stakeholders, like Facebook and Google, have initiated a number of projects like Free Basics [Fac16] and Loon Project [Alp16] correspondingly, targeting, to "Digital Inclusion",

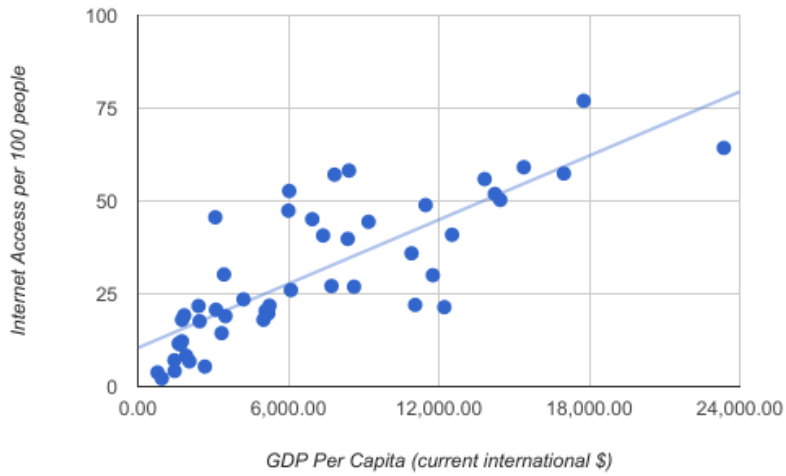

Figure 1.1: Relation of Internet Access and GDP per capita per country (World Bank, 2015).

as claimed. From the Internet Service Providers (ISPs) perspective, efforts similar to Fon [Fon16] and BT Wi-fi [BT 16] aim at providing their customers with ubiquitous access, diverging though from the objective of expanding the Internet access coverage. Nevertheless, those approaches function in a competitive market environment and adopt a top-down approach, therefore possibly overlooking the citizens' interests or common societal goals.

On a different context, many bottom-up initiatives in local communities follow the path of cooperation instead of competition, based on the observation that sharing resources can save costs in exchange for coordination, an approach endorsed as well by the International Telecommunication Union (ITU) [Int09]. Based on this principle, as a way to mitigate the Internet access cost challenge, in many regions worldwide the citizens self-organize to explore alternative models for getting local connectivity and Internet access under accessible conditions, particularly in underserved areas, including the deepest rural communities worldwide [Rey+13]. Community Networks (CNs) [Veg+15], an

example of this approach, are crowdsourced data network infrastructures built by citizens and organisations, who pool their resources and coordinate their efforts [Bai+15]. One of the most common services provided to their members is an Internet community access service.

From a technical perspective, Internet access sharing can be achieved in multiple ways. A very common approach, adopted by many CNs, is through HTTP Proxies, which allow clients to access the Web, but also other services through the HTTP CONNECT method. In the Transport Layer, complementary to the HTTP CONNECT method, SOCKS proxies allow users to establish TCP (and in the most recent versions UPD) connections to external networks. As far as the Internet Layer is concerned, Internet sharing can be achieved by establishing IP tunnels to Internet gateways, which along with HTTP proxies are the most common Internet access techniques in CNs. In this layer there exist various technologies that would facilitate access sharing, like IP multihoming, integration of sharing mechanisms in the routing protocols or IP level overlays that enable sharing as in [Abu+15]. Finally, sharing techniques can be implemented at the Link Layer using for example WiFi sharing schemes where individuals can connect to Access Points that allow the sharing of the unused Internet bandwidth, as in [Sat+12].

1.1. Community Networks

CNs are a relatively new paradigm, and they represent an alternative approach for developing network infrastructures and services in a broad sense. Communities can propose locally adapted self-organized cooperative schemes for developing self-provided data networking solutions, sharing wireless links and spectrum, and optical fibre. Additionally CNs participants share resources from Internet gateways, distributing Internet connectivity among other members of the community.

These communities usually describe themselves as *open*, *free*, and *neutral*. They are open since everyone has the right to know how they are built. They are free because the network access is driven by the non-discriminatory principle; thus, they are universal. Moreover, they are neutral in terms of technology solutions to extend the network, and neutral for supporting data transfers.

When these fundamental principles are applied to an infrastructure, they can result in networks that are *collective goods*, *socially produced*, and governed as *common-pool resources* (CPR). Natural CPR, also called commons (such as communal pastures, fisheries, forests), were studied in depth by Elinor Ostrom [Ost90]. The authors of [Nav+16b] have transferred the idea of the commons to networks to what is defined as *network infrastructure commons*. Under these, or similar conditions, CNs can potentially provide to their participants access networks under decentralized ownership, diverging from the traditional profit-driven oligopoly model and focusing on the users' needs, including environmental, economic and privacy issues [Fuc17].

These cooperatively developed infrastructures become regional IP networks that enable inexpensive interaction and access to local digital content and services. In addition, these networks provide an alternative approach to tackle the issue of access to the global Internet, that can be achieved through ISPs available in these regional network infrastructures.

During the elaboration of this thesis we performed an preliminary qualitative study about the state of the current CNs, obtaining information like size, number of users, technologies used and Internet provision strategies, for more than 20 of the largest communities around the world. A sample of this information is depicted in Table 1.1. The infrastructure of almost all the studied communities is based on wireless mesh technologies, though not necessarily heterogeneous as in guifi.net, showing that the studied environment is very similar to other existing CNs. Moreover, the information collected shows that the majority of these networks provide Internet access to their members, mainly through both uplinks to ISPs and Internet sharing schemes, including Web proxies. While the evaluation of the proposed Internet sharing solution in other CNs forms part of our future work, our qualitative study contains promising evidence that the presented mechanisms can be successfully deployed in these communities.

There are many other examples of CNs that can fit in this scheme. The survey in [net16] outlines 18 cases, with 9 described in detail, and 267 potential cases in 41 countries. There are also several studies that consider structural [Cer12; Veg+12; ML15], technological [Veg+15; Mac+15; BML15] and organisational [Bai+15; LM14; net16] points of view of these networks.

Name	Location	Nodes #	Users #	Infrastructure	Internet Provision
guifi	mostly Spain	25000	30000	WMesh, WP-t-P, Fibre, WAPs	Tier 2&3 ISPs Uplinks, microISPs, DSL Sharing
AWMN	Greece	3000	5000	WMesh, WP-t-P, WAPs	Tier 2&3 ISPs Uplinks, DSL Sharing
Freifunk Berlin	Germany	500	2200	WMesh, WP-t-P, Fibre, WAPs	Tier 2&3 ISPs Uplinks, DSL Sharing
Sarantaporo.gr	Greece	180	1700	WMesh, WP-t-P	DSL Sharing
Wireless Leiden	Netherlands	120	4000	WMesh, WP-t-P, Mobile, WAPs	Tier 2&3 ISPs Uplinks, DSL Sharing
QuintanaLibre	Argentina	53	250	WMesh, WP-t-P, WAPs, Wired Ethernet	Tier 2&3 ISPs Uplinks, DSL Sharing
TakNET	Thailand	34	300	WMesh	DSL Sharing

Table 1.1: Characteristics of various Community Networks.

1.2. guifi.net and the Web Proxy Service

guifi.net is an open, free, and neutral network built by its members: citizens and organisations pooling their resources to build and operate a local network infrastructure, governed as a common pool resource [Bai+15]. The network infrastructure is organised as an inter-network with several local Wireless Mesh Networks (WMNs), consisting mostly of wireless links [Veg+15] with a fiber backbone. Participants can extend the network to reach new locations and use the network to reach intranet services such as the Web proxy service. Since the network links between nodes are contributed and managed by the participants, paths between nodes, such as client to proxy, may not be reliable [AWW05] or guaranteed, especially when compared to commercial offerings from centrally managed ISPs. It is important to note, that to our knowledge, guifi.net is probably the largest CN in the world with more than 30,000 network nodes, competing with Freifunk [Fre16], which is nevertheless a collections of networks that are not necessarily connected between them.

The most popular service in CNs is Web access and guifi.net is no exception. Web proxy nodes connecting guifi.net to external ISP act as free Internet gateways for CN users. Proxies run on commodity servers which are hosted by individuals or organisations (like libraries or municipalities) offering their Internet access to other guifi.net users. Using Web proxies, public entities can provide free Internet access without infringing telecom market competition regulations. While some of the Web proxies are kept as a private service, 356 out of the 477 registered Web proxy servers in the network (May 2016)

are shared with all the network registered participants (12,500). Registered members are allowed to use any proxy of their convenience, although it is recommended to use one nearby.

Users can select or change their choice based on quality of experience. Therefore, while some proxies may become popular and highly used, others may remain underused. To get Web access, the clients must configure manually a list of proxies in their browser, starting with the main proxy and following with the secondary ones. Proxy access within guifi.net is managed through federated authentication credentials. In case a proxy does not respond (timeout) or rejects the connection, the client automatically switches to the next proxy in the list. The choice of proxies is manual and the list usually comes from acquaintances in the community or personal experience. Additionally, in the current scenario, proxies have a rough admission control based denial of service due to saturation, and they do not perform congestion control according to load or performance. Without access to one of these proxies or any guifi.net Internet uplink, community members can still share contents and access applications within the same network, but not to external resources.

Web access through Web proxies is clearly a limited service compared to an IP tunnel, as the service is usually restricted to a set of protocols or ports; however, it can enhance privacy as the source IP addresses are hidden, similarly to anonymous Virtual Private Network (VPN) service providers. Many citizens, private and public organizations involved in CNs, such as Freifunk [Fre16] or guifi.net, have chosen to provide that service within their CN. Using Web proxies through local networking infrastructures (e.g., CNs) that provide local/regional connectivity, the citizens can reach Internet content and services with some limitations but no additional cost and potentially enhanced privacy.

1.3. Spare Internet Capacity

We define *spare Internet capacity* as the network traffic that can be transferred to and from the Internet by secondary users (using Internet connectivity provided by others) with no significant performance degradation or cost penalty for primary users (sharing their Internet connectivity). The secondary traffic can have short term impact on the primary user's traffic in term of packet queueing, resulting in service degradation, such as packet delay, loss and reduced throughput. That affects data transport generating a lower throughput

with longer and more variable download times, and also an overall degradation of the user experience quality. Given that Internet connectivity may be given to primary users according to a cost model (e.g. 95% percentile, data volume), the secondary traffic might result in a cost penalty that should be considered and ideally avoided. For instance, the secondary traffic may need to be blocked to avoid having an impact on cost under the common 95-percentile pricing schemes [Gol+04] used by transit ISPs to charge according to peak demand. Therefore, the goal of any regulation would be that the secondary traffic does not affect the primary users in terms of cost and performance. Achieving this goal depends on the total traffic, primary and secondary, and should apply both when the Internet connection is above and below the saturation point.

Chapter 2

Problem Statement & Contributions

2.1. Problem Statement

Considering the existing Internet sharing techniques and initiatives we chose to focus on the Web proxies in CNs. In that environment, the general research question, of how to provide access to the Internet for all, results in several considerations and two main research questions.

Research Question 1 (RQ1): Which are the characteristics, limitations and usability of a shared Internet Web proxy service in CNs?

Studying the Internet access based on Web proxies in guifi.net we observed that some proxies may be overloaded and, therefore, offer degraded or unusable performance since they do not perform congestion control. At the same time, other Web proxies remain underused, due to bad and uninformed manual choice. The set of overloaded proxies varies according to the access patterns of the users, while choosing of the overloaded proxies or routing through congested links might result in degraded Internet quality of experience (QoE). We observed that this issue is an instance of the more general problem where a community of participants in a local inter-network access the Internet through a pool of shared Web proxies in different network nodes.

The net effect is that a large population of C clients can browse the Web benefiting from the aggregated capacity of a pool of P Web proxies, with

$C \gg P$, over a CN infrastructure, at a fraction of the cost of C Internet connections. We decomposed this general problem into two major components:

- The relation between Web clients and proxies in a macroscopic level, which questions how the demand of the clients can be met in an efficient way by the offered resources.
- The microscopic relation between each participant offering his/her resources (primary) and the corresponding consumers (secondary), focusing on the preservation of the experience of the primary participants, since they play a key role in the service, while keeping overhead to the secondaries to a minimum.

In the light of the resulting analysis and limitations that stem from the observation of a minimal viable product operating in the field, the next question is:

Research Question 2 (RQ2): Can we improve the ***user experience*** *and* ***fairness*** *of Internet sharing Web proxy services in CNs without introducing significant overhead to the network and other services?*

In this context, applying the criteria of minimal intervention, we focused on 1) improving the manual proxy selection by clients, and 2) improving the fairness of traffic aggregation among a primary and its secondaries.

In the clients-to-proxies relation, the challenge lies on enabling that clients located in any network node to select the right proxy according to the performance of the internal network path and the load of the different Web proxies. The designed solution must be:

- Incremental and backwards-compatible: should be able to be deployed incrementally, so it should work fine for both, original, or clients using our approach.
- Dynamic: Users can and should switch proxies wisely to maximize their QoE. This can be the result of changes in network topology, path load, or proxy performance.
- Decentralized: should not require any central component.

- Scalable: able to scale up to the current number of users and proxies and beyond.
- Routing-agnostic: should not depend on the transport and routing algorithms, or on specific network features.
- Transparent: should little or no negative effect on the user experience significantly, while remaining invisible to the user.

Concerning the primary-secondary relation, we consider N citizens or organizations sharing their unused Internet access capacity benevolently with M neighbors and members of their community, through a local or regional CN. Each of the M beneficiary nodes selects one or a few of the N Internet gateways, where it sends its Web requests. Although this traffic uses the spare Internet access capacity in the gateways' networks, it may compete with the primary traffic, hinder their performance, and also increase their cost. In order to provide such a service without negatively affecting the quality of access of the primary participants, we propose that gateways should differentiate the primary traffic (i.e., from Internet access donors) from the secondary traffic (i.e., from the beneficiaries) fairly. Therefore, only making this sharing process innocuous for the donors, will allow this mechanism to be sustainable over time. However, keeping under control this aspect of the traffic represents a major challenge for managing an Internet sharing service in a CN.

2.2. Methodology

Aiming to answer the presented research questions, our work has followed an empirical quantitative methodology, based on the design science problem proposed by [Pef+07]. This methodology is:

- Collection of data: To characterize the problem of Internet access in CNs and form the objective to its solution, we collected anonymized data from a real Internet Sharing Web proxy service in a CN, more specifically from guifi.net. We extracted patterns and statistical properties applying statistical modelling and time series analysis to the obtained data. The results provide valuable insights, not only specific for that case, but also generalizable to other cases.

- Definition of research objectives: The collected data, together with the literature reviewed, helped us to clearly define the objectives of our work.
- Design & Implementation: We developed the different parts of the research following an experimental design approach, specifying variables, measurement methods and requirements, and also selecting appropriate technologies.
- Experimentation & Simulation: For the evaluation of our work we performed experiments in the field and laboratory environments, depending on the nature of the problems addressed. Additionally, simulations were performed to demonstrate the impact of variables that were not considered in our proposed design. These simulations do not rely on an existing network simulator, but are implementations of analytical methods that represent different scenarios and compute results based on input data from the existing network components.

2.3. Data Collection

As introduced in the previous section our approach is based on observed and measured data. Therefore, in this section we are going to list the most relevant data sets that we collected, as well as the tools that we used or developed to gather and process that data.

Link Network Description Expressed in the Community Networks Markup Language (CNML) [gui16], an XML schema to describe community networks. guifi.net publishes a snapshot of its network structure every 30 minutes, with a description of currently registered nodes, links and it configuration. In the CNML description the information is arranged according to geographical *zones* in which the network is organized. This information was used to build the network topology graphs.

Daily Proxy Logs The guifi.net Web proxy logs are stored in a publicly accessible repository[1]. We used anonymized logs of one month of duration for the 4 proxies operating in a specific guifi.net zone (Lluçanès), used

[1] `http://opendata.confine-project.eu/dataset/guifi-proxy-logs` on the CONFINE open data site

by community members to access Internet content. The logs follow the native Squid log format [squ], registering information per request concerning the amount of bytes delivered to the client, time elapsed, users, URLs, HTTP and cache status codes. It is important to note that the native Squid log format accounts only for the traffic delivered to the client, which is constituted by the net object requested and the corresponding HTTP headers. This format does not include the upload traffic.

The available proxy logs follow an anonymization scheme applied to IP addresses and usernames. According to this scheme, the usernames are always anonymized with unique keys for different proxies and alternating ones for odd and even days. The source address information is available as clear-text only for odd days and masked using a /27 subnet mask. The destination URL/IP address information is available as clear-text only for even days, as seen in Listing 2.1.

The logs used specifically for this thesis can be found online[2]. Before using them, we filtered the logs to contain only users of the studied zone. Additionally, the time series obtained from the logs were preprocessed using methods for cleaning up the data, filling in missing values and other deficiencies.

```
Odd day:
time elapsed remotehost code/status bytes method URL rfc931 peerstatus/peerhost type
1368599985.502 212 10.140.46.32 TCP_MISS/200 1056 GET d8...98 4d...24 DIRECT/173.194.78.94 text/html

Even Day:
time elapsed remotehost code/status bytes method URL rfc931 peerstatus/peerhost type
1368564073.974 242 fe...08 TCP_DENIED/407 6471 GET http://s.youtube.com/s? 75...e1 NONE/- text/html
```

Listing 2.1: Log example.

Usage and Availability of Paths guifi.net has a distributed hierarchical monitoring system, called SNPServices [gui15], to collect metrics about the in/out traffic, latency and availability on all registered nodes and link interfaces. In each zone there exist one or more nodes –called graph servers– that continuously collect information through Simple Network Management Protocol (SNMP) requests, pings and traceroutes to nodes in that zone. In our case, the proxy nodes were co-located with these graph servers, therefore we were able to collect data about the number

[2] `http://emmdim.pc.ac.upc.edu:5000/anon_proxy/`

of hops and average latencies between the proxies and every other router in the zone under analysis.

All the statistical analysis was done using the Python *Scipy Stack* [J+01], the Python *Statsmodels* package [SP10] and the Python *Scikit-learn* package [Ped+11], while for graph representation and processing we used the Python *NetworkX package* [HSS08]. Traffic generation was done using *wrk2* [Ten15] HTTP benchmarking tool. the It is also important to note, that in our effort to collect the necessary data we have created the *guifiAnalyzer* [Dim16] tool which is in experimental status. The starting point for *guifiAnalyzer* was the *libcnml* [Cas17] project, which parses the Community Network Markup Language (CNML) data in Python.

2.4. Contributions

2.4.1. List of Publications

[DMN17] **Dimogerontakis, E.**, Meseguer, R., Navarro, L., "Internet Access for All: Assessing a Crowdsourced Web Proxy Service in a Community Network". In: *Passive and Active Measurement Conference (PAM)*. 2017.

[Dim+17a] **Dimogerontakis, E.**, Neto, J., Meseguer, R., Navarro, L., "Client-Side Routing-Agnostic Gateway Selection for heterogeneous Wireless Mesh Networks". In: *IFIP/IEEE International Symposium on Integrated Network Management (IM)*. 2017.

[Dim+17b] **Dimogerontakis, E.**, Meseguer, R., Navarro, L., Ochoa, S., Veiga, L., "Design Trade-offs of Crowdsourced Web Access in Community Networks". In: *IEEE 21st International Conference on Computer Supported Cooperative Work in Design (CSCWD)*. 2017.

[Dim+17c] **Dimogerontakis, E.**, Meseguer, R., Navarro, L., Ochoa, S., Veiga, L., "Community Sharing of Spare Network Capacity". In: *IEEE International Conference on Networking, Sensing and Control (ICNSC)*. 2017.

[Dim+17d] **Dimogerontakis, E.**, Braem, B., Meseguer, R., Navarro, L., "Socio-Economic Experiences, Challenges and Lessons in Community Networks around the world". In: –. [unpublished]. 2017.

Other Publications

The background research to this thesis has led to the following publications:

[Sel+15] Selimi, M., Khan, A. M.**, Dimogerontakis, E.**, Freitag, F., Centelles, R. P., "Cloud services in the Guifi.net community network". In: *Computer Networks* 93, Part 2 (2015). Community Networks, pp. 373–388.

[DVN13] **Dimogerontakis, E.**, Vilata, I., Navarro, L., "Software Defined Networking for community network testbeds". In: *2013 IEEE 9th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*. Oct. 2013, pp. 111–118.

[Esc+14] Escrich, P., Baig, R.**, Dimogerontakis, E.**, Carbó, E., Neumann, A., Fonseca, A., Freitag, F., Navarro, L., "WiBed, a platform for commodity wireless testbeds". In: *2014 IEEE 10th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*. Oct. 2014, pp. 85–91.

[Mil+15] Millan, P., Molina, C.**, Dimogerontakis, E.**, Navarro, L., Meseguer, R., Braem, B., Blondia, C., "Tracking and Predicting End-to-End Quality in Wireless Community Networks (FiCloud)". In: *2015 3rd International Conference on Future Internet of Things and Cloud*. Aug. 2015, pp. 794–799.

2.4.2. Contributions

Usability: Analysis of a crowdsourced Web proxy service We consider the environment of Internet access sharing through Web proxy gateways in local or regional CNs. In an effort to understand the effectiveness and performance of this shared Internet access, we performed a measurement study

of a crowdsourced Internet proxy service in the guifi.net CN, that provides free Internet access to a large community with a high users to proxies ratio. Our study focused on a representative subset of the whole network with about 900 nodes, 4 proxies and roughly 470 users of the Web proxy service. We analyzed the service from three viewpoints: Web content of users' traffic, performance of proxies and influence on the access network. We saw that Web proxies in a CN can be used to provide basic Internet access. Nevertheless, we observed the necessity of a regulation mechanism that would enable the fair and efficient usage of the available resources, considering the entire sets of users and proxies instead of a local level.
The main results related with this contribution are presented in Chapter 4 and were originally reported in [DMN17].

Scalability: Web Proxy selection mechanism To facilitate the fair usage of the Web proxy service we started by investigating and building the client-proxies regulation mechanism. We developed a client-side distributed system that optimizes the client-proxy mapping, agnostic to the underlying infrastructure and protocols, requiring neither any modification of proxies nor to the underlying network [Dim+17a]. Clients choose proxies taking into account network congestion, proxy load and proxy performance, without requiring a minimum number of other participating clients in order to select an adequate proxy. Our proposal was evaluated experimentally with clients and proxies deployed in guifi.net. We show that our selection mechanism avoids proxies with heavy load and slow internal network paths, while achieving a network overhead linear to the number of clients and proxies. Moreover, we demonstrated that informed proxy selection and admission control in proxies, could alleviate up to a certain level imbalances and unreliability, and also improve the overall service with small additional overhead [Dim+17b]. Nevertheless, the Internet sharing process can negatively affect the service experience of the users that share their connections, thus jeopardizing the continuity of this community service.
The main results related to this contribution are presented in Chapter 5 and Chapter 6 and were originally reported in [Dim+17a; Dim+17b] respectively.

Fairness: Secondary usage of spare Internet capacity Building upon the studied model of proxy usage, we argue that it is important to differentiate between primary users, who share their Internet connection, and the secondary

users in order to preserve the contribution of the former group. To guarantee that there is no additional cost incurred for the users sharing their connections we proposed a mechanism where a middlebox separates the traffic of the primary users from that of the secondary users. We evaluated the efficiency and behavior of several traffic separation mechanisms, in order to determine how to maximize network utilization and usage of the spare network capacity, while minimizing the possible impact on the primary traffic. Finally, we presented a set of recommendations to achieve the best performance isolation for the primary user, without significant impact to the perceived experience of the secondary users.
The main results related with this contribution are presented in Chapter 7 and were originally reported in [Dim+17c].

2.5. Thesis Structure

Figure 2.1 describes the relation of the research questions with the contributions and the chapters included in this thesis.

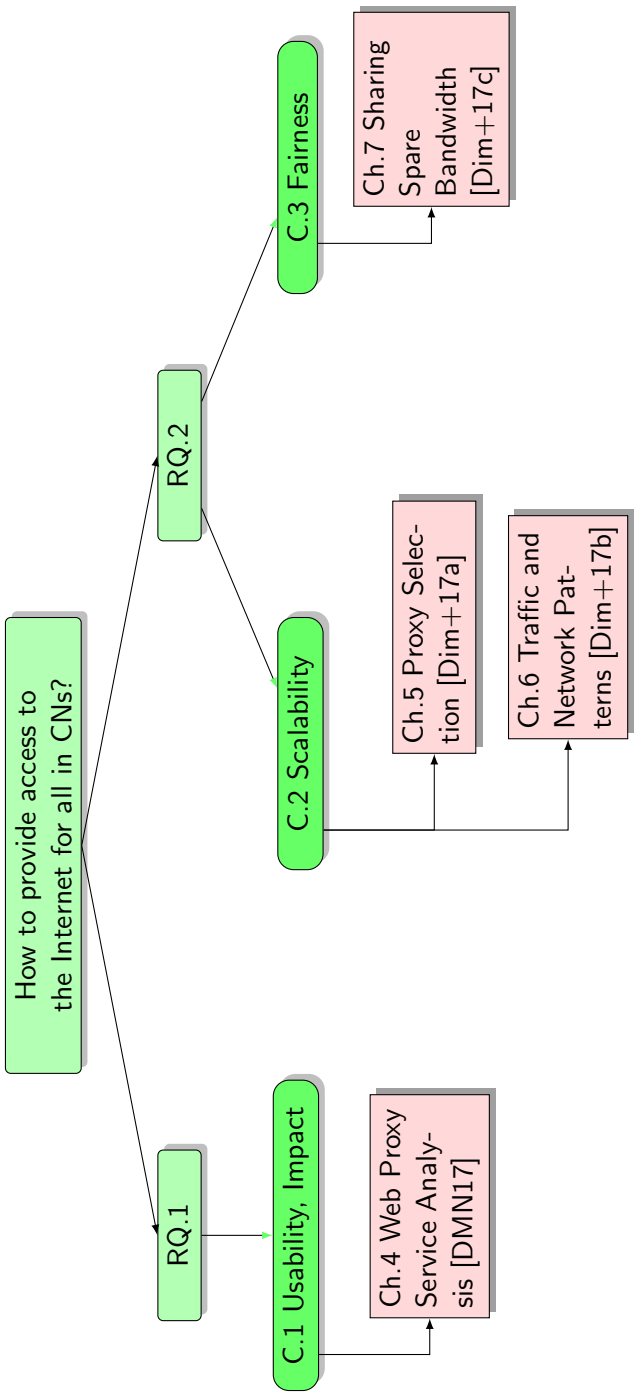

Figure 2.1: Thesis Layout.

Chapter 3

State of the Art

Internet connectivity is based on the principle of inter-networking, interconnection of networks, a practice of connecting computer networks through gateways that route information packets between networks. The interconnection can be transparent and direct at the IP level or through gateways such as NAT or proxy. The state of the art is structured according to the role of web proxy gateways in 1) how Internet connections can be shared, and 2) how clients select among these gateways. For the first, we have to consider previous work measuring or proposing schemes for Internet access sharing, community networks, spare connectivity, and Web proxies. For the second, the main considerations are on methods for the selection of gateways or proxies, and on metrics for choice considering the influence of the access network and proxies on the performance of the service.

3.1. Internet Connection Sharing

3.1.1. User Experience in Wireless Network Usage

Most research on wireless networks focuses on usage traffic patterns, link level characteristics and topologies, but not user experience, e.g. MadMesh [Bri+08], Google WiFi [Afa+10] and Meraki [Bis+15] networks. In all these studies, Internet access is direct instead of using proxies, and these wireless networks are homogeneous. Our scenario cannot easily be compared to those, nevertheless we use them as a point of reference.

Wireless network user experience has been characterized in previous studies. The first [Joh+11] focuses on Web traffic and the use of proxies to access Internet content in rural areas. We have also analyzed traffic using proxies inside the guifi.net network, but the lessons learned are more complementary than comparable as the scenarios are too different. The second study [IP11] looks at Web traffic patterns and content caching, relevant in a scenario with a single and very limited Internet uplink, whereas guifi.net has more than 350 proxies, with excess overall capacity and quite different usage profiles.

3.1.2. Community Networks

The CNs FunkFeuer Wien, FunkFeuer Graz, and Rome Ninux networks have been studied in [Mac13; Mac+15]. These are all homogeneous mesh networks based on the Optimized Link State Routing (OLSR) ad-hoc routing protocol. Moreover, [Veg+15] studies guifi.net, a bigger and heterogeneous CN. All these studies focus on the topological properties of the Link Layer, technologies involved and routing performance. Even the processes of deployment, management and governance of CNs have been studied [Bai+15]. Nevertheless, there is a lack of studies about user experience and network usage on CNs.

3.1.3. Spare Internet connectivity

There is a rich body of work focused on reducing the cost and increasing Internet coverage under several scenarios. For instance, the Lowest Cost Denominator Networking (LCD-Net) [SC13] explores resource pooling Internet technologies to support benevolence on the Internet. Some of these ideas are illustrated by WiFi sharing schemes, community-led (PAWS [Sat+12]) or commercially-run (FON [Fon16]), where home broadband subscribers donate their controlled (but for free) broadband Internet spare capacity to fellow citizens. This is done by sharing a fixed portion of throughput [Abu+15]. In contrast, our work considers not just local access to a shared WiFi hotspot, but also remote access to the shared resource over a CN that can use any network technology, such as, wired or wireless meshes. Our research also focuses on spare capacity, with little or no visible impact on the primary user. This means secondary users can get from all to nothing, depending on the demand from primary use.

3.1.4. Web Proxies

The Web proxy business has changed significantly over the recent years. The percentage of cacheable content has been decreasing, coupled with a dramatic increase of HTTPS traffic [Nay+14]. At the same time, Web caching remains relevant, as the data volumes on the networks continue to grow and network operators are seeking ways to cope with this demand in an efficient manner.

Performance evaluation of Web proxies cannot solely rely on high-level metrics such as hit rates. Low-level details such as HTTP cookies, aborted and persistent connections between clients and proxies as well as between proxies and servers have a strong impact on performance, particularly in heterogeneous bandwidth environments [Fel+99].

Recent studies show that efficient use of Web proxies in wireless and mobile networks is not a trivial problem. In [SS12], the authors propose a random-path cache request technique. This approach distributes the network load across several network paths, leading to lower transfer latency. In [Cat+11], the authors analyzed a mobile network topology to optimize the selection. Deploying geographically proxy caches and redirecting clients to the closest server in terms of latency has emerged as a common paradigm for improving client performance. In Google's Content Distribution Network (CDN), redirecting every client to the server with the least latency does not suffice to optimize client latencies [Kri+09]. Even though most clients are served by a geographically nearby CDN node, a sizable fraction of clients experience latencies higher than other clients in the same region. We have found a similar behavior in CNs, despite the fact the hop distance is not the most reliable metric for wireless environments that constitute a great percentage of CNs' infrastructure.

In this context, a new problem arises: deploying new proxies to improve wireless network and client performance. In content distribution, a related area, the development of placement strategies for Web server replicas to improve CDN performance was explored in detail, e.g. [QPV01; KW00]. In [QPV01], the authors develop several placement algorithms that use workload information, such as client latency and request rates, to make informed placement decisions. In [KW00], the authors introduce clusters of clients to move content closer to the groups of clients that are responsible for large subsets of requests.

	Incremental	Heterogeneous Environment	Optimized Network Monitoring	Passive Gateway Monitoring	Realistic Evaluation
[BH11]	✓	×	×	×	×
[ABC10]	✓	×	×	✓	×
[AAJ09]	×	✓	×	×	×
[Ko+13]	✓	Under Conditions	✓	×	✓
Our solution	✓	✓	✓	✓	✓

Table 3.1: Internet Gateway Selection in WMNs state of the art.

3.2. Internet Gateway Selection

As presented in this section, while the performance measurement and gateway selection in WMNs has been studied in the past, the existing approaches cannot be applied to large-scale heterogeneous environments, such as the presented scenario.

3.2.1. Gateway Selection in WMNs

Our Internet gateway selection problem is strongly related to the topic of gateway selection in WMNs which has been extensively studied in the past. The works presented in [BH11; ABC10] however fail to function in heterogeneous environments since they present solutions that operate on the mesh routing layer as they require modifications to the infrastructure routers, which is inherently prohibitive in heterogeneous environments. Proposals like [AAJ09; GRS08] require additional software on the side of the gateways. All the works mentioned, despite proposing interesting solutions, lack practical implementations or testing in a real-world environment. An exception to the above, and closer to our work is [Ko+13], where the clients cooperate to probe the gateways and then use the results to select a proxy. Furthermore, while conceptually [Ko+13] can function in heterogeneous environments, in practice it needs modifications to the existing underlying routing protocols, since node discovery takes places by modified the underlying routing protocol control messages.

3.2.2. Network Performance Monitoring

Concerning performance monitoring of heterogeneous WMNs, the majority of the solutions for WMNs are based on active monitoring of network metrics, such

as path delay in [Ko+13; BCK07], estimated link quality in [BH11; AAJ09], link interference in [BH11; AAJ09] and path packet loss rate in [Ko+13]. All these approaches would entail a high monitoring overhead, except [Ko+13], where monitoring is done cooperatively to reduce the overhead.

As presented later, we used the Vivaldi [Dab+04] network coordinates system for estimating network performance. From an abstract perspective, network coordinates are a virtual positioning system where nodes gather information about the network to position themselves and other nodes in a coordinate space and are used to estimate the inter-node latency. Vivaldi [Dab+04] is a fully distributed network coordinates system that functions based on the idea of placing nodes in a two-dimensional euclidean space. The measured *ping* latency between the nodes is used to position them in the euclidean space. In addition to the probing, Vivaldi also uses spring-relaxation to *nudge* nodes in the Euclidean space to minimize prediction errors. While there have been proposals for modifications of the Vivaldi algorithm, the original algorithm is performing fine compared to the improvements described in [Che+11]. Moreover, the state of the art of network coordinates includes more sophisticated and more accurate systems, which however are not fully distributed since they rely on external landmarks, like the Pharos system [Che+09].

3.2.3. Proxy Performance Monitoring

As far as Internet gateway performance monitoring is concerned all the above proposals use active measurements to evaluate it. More specifically [BCK07] uses a congestion delay function, [ABC10] monitors the spare Internet Connection (available capacity). [BH11; AAJ09] require the gateways to participate in the monitoring process by measuring the queue length of their Internet interface, while [Ko+13] performs active probes. Contrary to these approaches, our solution is totally passive, implying though less accuracy in exchange of saving scarce network resources.

Chapter 4

Analysis of Community Internet Access

As introduced earlier, CNs allow local communities to build their own network infrastructures and provide affordable inter-networking with the Internet. guifi.net exemplifies how this can be achieved using wired and wireless links to create a regional network, and sharing several Internet gateways among all their participants. These gateways are usually Web proxies for Web access, but other traffic types can be accommodated as well through HTTP CONNECT, SOCKS or tunneling. Proxies, not exempt from the drawbacks of middleboxes, have also several advantages. Some content and DNS resolution can be shared in caches, and most important, proxies can protect the privacy of end users given they trust the proxy provider.

In this chapter we present a study of the existing Web proxy service in a guifi.net zone. The study helped us clarify the characteristics, limitations and usability of that operational service, answering the first research question (RQ1), as described in Section 2.1. For our analysis, as presented in Section 2.3, we choose to study the guifi.net zone Lluçanès, a region in the Osona county of Catalunya, as it is representative of other rural guifi.net networks[Cer12]. As described earlier, even-day proxy log entries anonymize the client IP address and show information about the requested URLs, while odd-day proxy logs show the opposite. The logs combined with openly accessible information about network topology, network links and network traffic information, provide a consistent and complete overview of the traffic of this regional network.

We analyzed the service from three viewpoints: 1) service usage by end-users, including patterns of usage and content in Section 4.1, 2) the proxy, Section 4.2, in terms of caching, users, performance and variability, and 3) the local network, Section 4.3, in terms of topology and usage. Our measurements describe the effectiveness of a simple setup of a regional network sharing a set of Web proxies in delivering free basic Web access to a large population.

4.1. Service Usage Viewpoint

The behavior of the users and the service can be described at the macro-level as a set of time series concerning metrics that can be extracted from the monthly logs, namely bytes per request, number of requests and number of users. Figure 4.1 illustrates the traffic time series for the aggregate set of proxies showing a daily pattern, but also significant aperiodic negative spikes, which were statistically verified as a dominant period of one day, and the second largest peak at 12 hours.

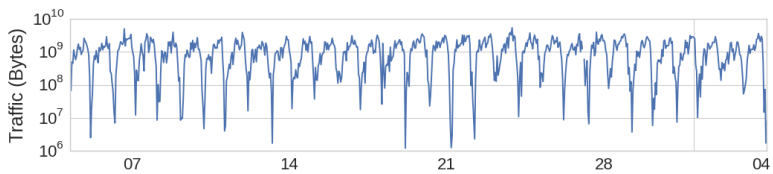

Figure 4.1: Web proxy time series (days in April 2016).

4.1.1. Traffic, Requests and Users

The majority of the traffic is due to a relatively small number of large requests (20% of the requests produce 97% of the traffic), while the rest of the requests present little variation in size. Additionally, as expected, the majority of the traffic (90%) is created by 15% of the users, but in contrast to the distribution of request size, the distribution of traffic and number of requests per user is similar to a geometrical distribution.

For the analysis of the service processing rate we calculated the **request processing throughput** as the bits per time elapsed for each request, depicted in the Empirical Cumulative Distribution Function (ECDF) Figure 4.2, ranging

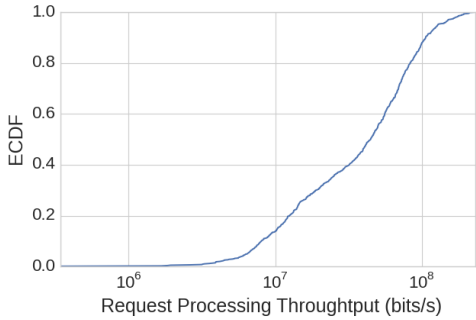

Figure 4.2: Processing rate per request.

Domain	Traffic Fraction
googlevideo	27.85%
mega	16.73%
fbcdn	5.40%
rncdn3	2.80%
nflxvideo	2.70%
xvideos	2.60%
tv3	2.54%
level3	2.51%
google	1.96%
apple	1.78%

Table 4.1: Top Domains by traffic.

from less than 10^7 for the worst 10% to at least 10^8bps for more than 80% of requests.

4.1.2. Content Analysis

Using the even-day proxy logs we analyzed the request types and target URL of users' requests. The majority of the traffic, almost 50%, consists of HTTP CONNECT requests, which is the method to establish TCP tunnels over HTTP, including HTTPS which is indisputably the main usage appearing in the logs. While for HTTP CONNECT we cannot know the corresponding content type, the most common type for the rest of the requests is the generic *application/** with 23% which can encapsulate other traffic types, followed by explicitly video content (19%) and image (5.5%) types.

The traffic for all analyzed proxies in Table 4.1, including HTTP CONNECT, shows that the top video portal traffic occupies 27.85% of the traffic, which is an impressively large amount. This table also allows us to assume, summing the video domain percentages, that the real video traffic surpasses the 30% of the traffic. For completeness, we mention that these high traffic percentages is not reflected in the number of requests, therefore they are attributed to the size of the objects requested. Since video is by far the HTTP type with most traffic, it is not surprising to find that 4 out of 10 top domains are video portals. For the distribution of Web traffic per URL, we found that it can roughly approximate a Zipf distribution, equivalent to results in [Mai+09] with domestic Internet connections.

4.2. The Proxy Viewpoint

In this section we investigate the capabilities and influence of the proxy servers involved. Our dataset contains the only 4 proxies operating in the Lluçanès zone. Table 4.2 shows the CPU and RAM characteristics of the proxy servers, as well as the nominal maximum throughput of the Internet connection they offer. They are very diverse, with great differences in Internet throughput (4–80Mbps). We also observe that proxy 11252 has the slowest combined characteristics. Despite the fact that these servers provide other services, e.g. SNMP, the interference caused by other services is expected to be negligible.

Id	CPU	RAM	Max Throughput
3982	Intel amd64 2-core 2.6GHz	2GB	80Mbps
10473	Intel x86 2-core 2.6GHz	0.5GB	6Mbps
11252	AMD Athlon(tm) XP 1700+	0.5GB	4Mbps
18202	Intel amd64 2-core 2.7	2GB	8MBps

Table 4.2: Description of Proxies.

	Different Data (MB)			Data Transferred (MB)				Ratio (/All Transferred)		
Proxy	All	Repeated	Cached	All	Repeated	Cached	Connect	Repeated	Cached	Connect
10473	606	37	9.2	1481	95	14.3	943	6.4%	0.9%	63.7%
11252	3572	1234	28	15352	5512	99	7578	35.9%	0.6%	49.4%
18202	6384	1498	151	15963	3039	253	9274	19.0%	1.6%	58.1%
3982	2542	435	55	6019	855	96	3128	14.2%	1.6%	52.0%
Avg	3276	801	61	9704	2376	115	5231	18.9%	1.2%	55.8%

Table 4.3: Average volume of data in four proxies and ratios in a month of logs.

The analysis of logs for the four proxies is summarized in Table Table 4.3. The values are averages for each proxy over a month of daily logs. The first group of columns (Different Data) shows a data object storage perspective, with the amount of different data objects requested (disregarding the number of requests for each). The second group (Data Transferred) shows a data transfer perspective, with the amount of traffic in each category. The third group shows data transfer ratios to the total transferred. We distinguish between "All" content, seen or transferred by the proxy, content requested repeatedly (same URL, cacheable or not), content served from the cache (checked or not against the server), and content that is invisible (CONNECT method, typically HTTPS, passed through blindly).

4.2.1. Cache Effectiveness

As introduced before, the passed through content (HTTPS) represents the majority of the proxy traffic (49.4–64%). Although URLs are repeated significantly (6.4–36% of proxy traffic), the content successfully served from the cache (after validation or not) only represents a negligible amount (1–1.6%). Considering number of requests instead of the amount of data, despite URLs repeat often (20–41%), the content does not seem cache friendly, as cache hits only represent a very small portion (3–10%). The analysis of the number of requests compared to byte count indicates that cached content usually corresponds to small objects.

Compared to the related work, the authors of [Cat+11] analyzed a mobile network topology with a two level cache hierarchy. Their claim that a caching system can be useful only when at least 5.1% of traffic is suitable for caching, shows that caching in our case with lower rates may not be that beneficial. The authors of [IP11] look at Web traffic patterns and content caching, mentioning the decreasing cache hit rate, which is even lower in our study 5 years later following a dramatic increase of HTTPS traffic.

The observed bad cache performance can be attributed to characteristics of the proxy service, such as small cache size, small number of concurrent users per proxy, or to increasingly non-cacheable served content. We next look at how these apply to our scenario, claiming that non-cacheable content is the main factor affecting cache performance.

4.2.2. Cache Size

The default allocated cache size in guifi.net proxy settings is 10GB of secondary storage, while in some proxies caching is not enabled at all. However, we found out that the cached content that resulting in cache hits only accounts for a maximum of 151MB (if all repeated URLs were cacheable) and an average of 61MB (based on the hit rate) of data per day. In the extreme case where all content is assumed to be cacheable and discounting the transparent CONNECT/HTTPS data, the amount of daily data seen (i.e. all content for all URL seen) accounts for a maximum of 1.5GB and 801MB on average, easily achievable with nowadays RAM-based caches.

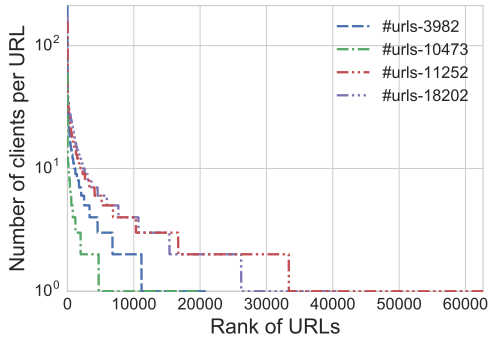

Figure 4.3: Rank of URLs by number of clients requesting them, by proxy.

4.2.3. Sharing Across Clients

Figure 4.3 shows the popularity of URLs across different clients in each proxy over a month, with top values between 60 to 212 different clients accessing the same URL. The number is related to the structure of the service, with many decentralized proxies and few users each and no inter-cache cooperation, which limits the potential of sharing cached content across more users. Nevertheless, proxies can provide the benefit of sharing network resources reusing not only HTTP content, but also reusing DNS resolution data as client Web browsers delegate that to the proxy, or even reusing established TCP connections among multiple clients.

4.2.4. Proxy Selection

Users are instructed to check the public list of nearby proxies (in their network zone) in the network management directory[1] with shows a list of nearby proxies, including status and availability ratio, or follow the advice of trusted neighbors with previous usage experiences. Therefore the choice is influenced by social factors and the reputation of the service, but in most cases the first choice is the nearest operational proxy with acceptable availability or reputation. Typically several nearby Web proxy services are configured in a user's Web browser. Since all federated proxies use the same authentication

[1] Lluçanès: `https://guifi.net/en/node/8346/view/services`

service, users are free to choose whatever proxy they prefer. The choice of proxy is rather fixed and prioritized manually, only switching to lower choice proxies when the current fails.

4.2.5. Users and Proxies

Figure 4.4 presents the distribution of the average number of users per hour. The different proxies show similar distributions, though we observe that proxy 10473 has a differentiated demand, with 40% of the time without any user and a maximum of 10 users per hour. The rest of the proxies, for the majority of the time (60%), have an almost linear distribution between 5 and 25 users, with nearly equally distributed values, and an average of around 17 users per hour for proxies 11252 and 18202, and an average of 12 users for proxy 3982. The difference in distribution among proxies comes as a result of preference for proximity and manual selection. To complete the picture, we found an average of 10 users in periods of 10 secs, an average of 76 different users per proxy and day, and a maximum of 254 in a month.

The users' distribution among proxies has a clear impact on the distribution of the number of requests in Figure 4.5. The ordering of proxies with respect to the number of users remains visible in the distribution of requests. Also, there is a close-linear behavior between 20% and 60% for all proxies except 10473. For proxies 11272 and 18202 the number of requests per hour is typically between 1K and 10K requests, with a mean of 8187 and 6716 respectively. For proxy 3982, typical values are between 500 and 1K requests per hour.

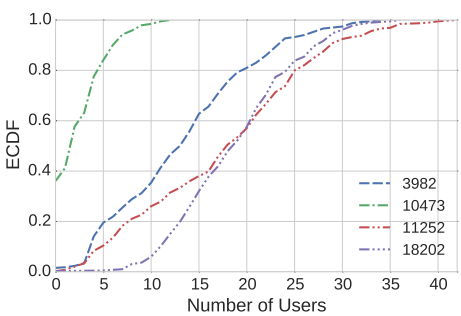

Figure 4.4: Hourly average number of users per proxy.

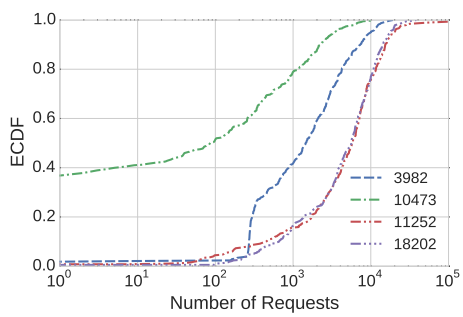

Figure 4.5: Hourly average number of requests per proxy.

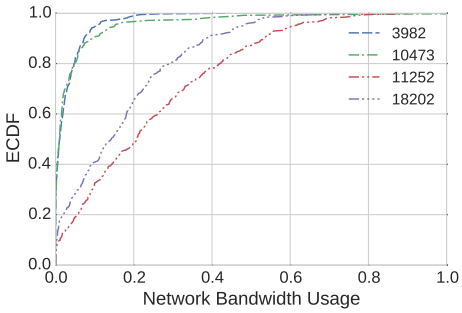

Figure 4.6: Network usage per Proxy.

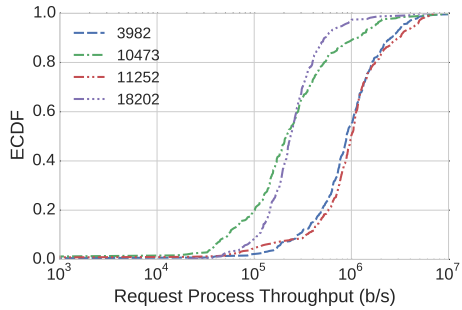

Figure 4.7: Hourly average request processing throughput per Proxy.

Regarding the number of clients seen daily by a proxy, the values (min, average, max) range from the lowest for proxy 10473 (14, 20, 27) to the highest for proxy 3982 (59, 82, 101). These numbers reflect the essence of a highly decentralized service with many small capacity local proxies.

4.2.6. Internet Connection and Processing Performance

Figure 4.6 provides the distribution of the Internet connection usage per proxy, calculated as the approximate instant connection throughput of each proxy normalized by its maximum Internet throughput, as provided in Table 4.2. All proxies show low utilization of their network resources, being approximately less than 0.3 (30%) for all the proxies for 80% of the time. Nevertheless, proxies 11252 and 18202 show significantly higher bandwidth usage.

From the distribution of the request processing throughput, shown in Figure 4.7, we observe that all proxies have an almost identical distribution, but around different mean values, depending on the individual characteristics of the proxy. Moreover, we can see that a significant percentage (>60%) of the time the proxies serve at a very narrow range of processing throughput, meaning they can offer a stable service. Even in the worst cases, the service does not suffer from extreme degradation, while remaining higher than 100Kbps for 80% of the time. We also observe that for proxies 3982 and 11252, the processing throughput distribution resembles the number of requests distribution in Figure 4.5 possibly indicating, as before, that the proxies are not saturated.

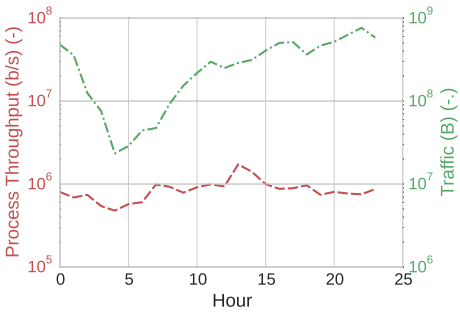

Figure 4.8: Daily average request processing throughput and traffic.

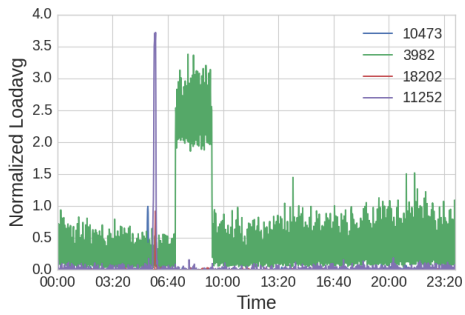

Figure 4.9: Daily Median Loadavg per proxy normalized by #CPUs.

To gain a more complete perspective, we also studied the daily aggregates of the traffic, users and requests, clearly observing not only the expected human daily pattern but also a clear effect of the different ways each proxy receives and serves request as a result of the users' manual proxy selection. Moreover, studying the mean daily patterns, we noticed, as seen in Figure 4.8, that the processing throughput presents very small variations implying a stable service behavior. Furthermore, the traffic volume varies no more than 1.5 orders of magnitude. The fact that the processing throughput is not affected by the traffic size confirms our observation that the servers are not saturated. Additionally, in order to verify that the processing capabilities of proxies are not a bottleneck for the service, we monitored the proxies' CPU load using the *loadavg* Linux metric. The results, showing a strong daily cyclic pattern, are summarized in Figure 4.9 that shows the daily median of the per-minute *loadavg* for each proxy normalized by the number of CPUs. Except from proxy 3982, which we verified that is affected by other co-located network services, the proxies are not overloaded. The brief daily peak in each proxy is due to a daily restart of the proxy service as programmed by the developers of the service, that includes a cache reindexing.

From all the above we can conclude that the proxies are able to offer a stable service, with respect to the traffic load, allowing them to be used as an alternative domestic Internet connection. The network capacity of the proxies is underutilized assuming that no other services co-located on the host of the proxy are heavily using its Internet network capacity. Moreover, the Web

graph	nodes	edges	degree max/mean /min	diameter
base-graph	902	914	98/2.04/1	11
proxy-clients-graph	463	472	60/2.04/1	10
backbone-graph	47	56	10/2.38/1	9

Table 4.4: Summary of Lluçanès network graph.

cache of the proxies does not seem to function in an efficient manner, implying that the proxies act mostly as HTTP-level traffic forwarders.

4.3. The Local Network Viewpoint

The local network infrastructure has also an influence over the final user experience. For the analysis we used information extracted from odd day logs that provide these details while hiding URL destinations.

4.3.1. Network Structure

For the local network we considered all operational nodes and links of the Lluçanès guifi.net zone[2]. We refer to the entire zone network as the base-graph. Moreover, we refer as proxy-clients graph to the part of the Lluçanès network including only the nodes (clients, routers, proxies) that participate in the proxy service. More information concerning the network structure, hardware characteristics, and protocols used in guifi.net can be found in [Veg+15].

Similarly to the rest of Osona county zones, and in general to many rural CN deployments, the network consists of a small set of interconnected routers, the backbone graph, where each router is connected with a large number of end nodes, most of all wireless links, mainly 802.11b connections [Veg+15]. Users access the entire guifi.net network from the end nodes. Some of the routers act also as hosts for various guifi.net services, including the proxy service. Table 4.4 describes the main characteristics of the aforementioned graphs. We notice that the mean degree of the base-graph and of the proxy-clients-graph is very low since the end-nodes with degree 1 dominate the distribution of degrees. Figures 4.10 and 4.11 provide a view of the proxy-clients graph and the backbone-graph. The colors of the participating nodes and routers indicate

[2]More information on the Lluçanès zone `https://guifi.net/en/node/8346/`

that they are using the proxy of the same color. Moreover, in Figure 4.11 the darkness of the link color denotes the cost in latency for a byte to cross this link -calculated as the half of the RTT measured by ping assuming symmetric paths-, therefore the darker the color the more expensive is the link to use. Figure 4.11 in combination with Table 4.4 show that the low mean degree value in the backbone-graph corresponds to a majority of the routers have only two or three neighbors.

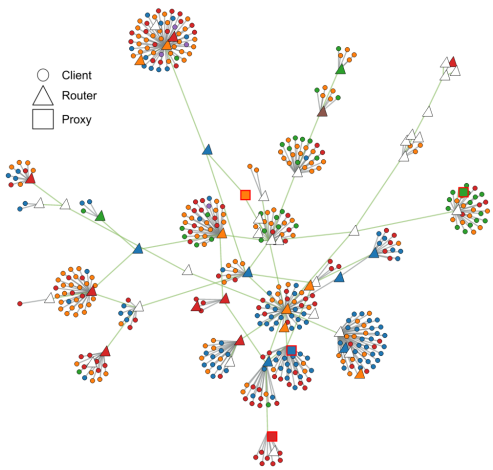

Figure 4.10: Lluçanès Proxy/Clients.

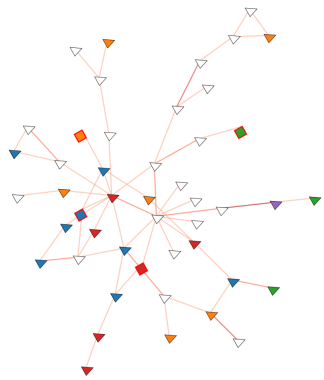

Figure 4.11: Lluçanès Backbone.

4.3.2. Network Usage

Since the proxy selection process is static (manual configuration), the analysis of local network usage can show the effect of selection on the perceived user experience. Towards that end, we first analyzed metrics of distance between the users and the proxies. Figure 4.12 shows the distribution of the number of hops between the users and the selected proxies. The distribution is almost uniform for 95% of the users with values between 1 and 6 hops. The remaining 5% is split between 7 and 8 hops. Nevertheless, we observe that manual choices result in a slight increase in the number of hops, therefore possibly introducing small unnecessary overheads. The latency involved, depicted in Figure 4.13, shows a different behavior. Almost 80% of the users experience an average latency smaller than 15ms to reach their proxy. The remaining 20% are subject

to latency between 20ms to 35ms. Despite the almost uniform distribution of hops, latency values vary much less, implying that during normal network conditions, the distance between the users and proxies is not significantly deteriorating the user experience for Web services. Nevertheless, this seems this result is based on a snapshot of the link latency at a given moment in time and, thus, does not take into account possible link congestion.

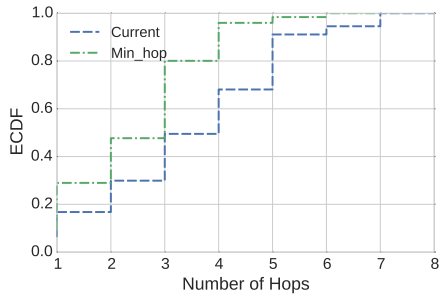

Figure 4.12: Number of network hops between users and their selected proxies.

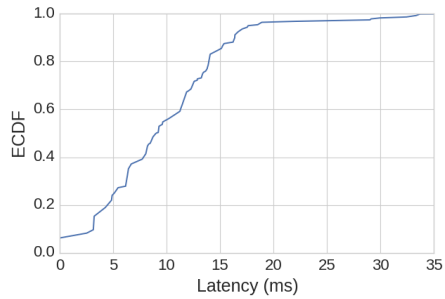

Figure 4.13: Average latency between users and their selected proxies.

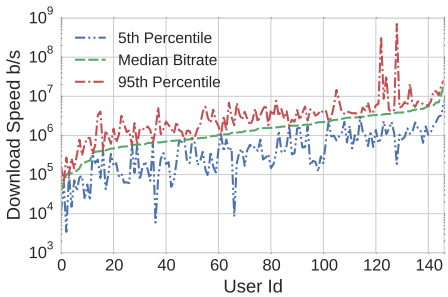

Figure 4.14: Estimation of user experience throughput with objects >1MB.

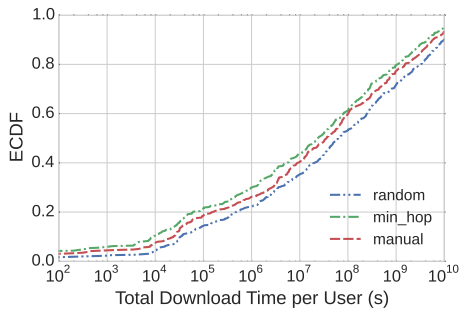

Figure 4.15: User cost as sum of download times (1 month).

4.3.3. Download Throughput

As described earlier, the request processing throughput of each proxy is calculated based on the elapsed request time, which includes the time the

proxy requires until sending the last byte of the Web response to the client. Therefore, any significant local network deterioration would affect the observed throughput behavior. Based on this observation we used the request processing throughput metric for requested objects larger than 1MB, which represents almost 77% of the traffic, in order to estimate any significant deterioration of the user experience. Including smaller objects would give unreliable throughput results due to the major influence on the measurements of network buffering on the proxy, DNS caching and network latency variations for short connections.

The estimation of the individual user experience in throughput is illustrated in Figure 4.14. Using the proxy logs we calculated the download speed for requested objects larger than 1MB. A simplifying assumption is that users focus on few or a single large object at a time. If so, our measures could be taken as a lower bound for the experienced individual download throughput. Median values of download throughput appeared to be quite stable with median values ranging from 0.1Mbps to 10Mbps for different users. Quite a good result for the many users of a free crowdsourced service, as shown by the following comparisons. In the evaluation of Google WiFi and MadMesh in 2010 and 2008 respectively, transfer rates were limited to 1Mbps, but 80% are getting less than 80Kbps with Google WiFi. In MadMesh 80% get less than 1Mbps with 85% of the clients connected within 3 hops to the Internet, comparable with our results that achieve higher speed but more hops to a Web proxy. The evaluation of Facebook's Free Basic Service [Sen+16] shows comparable performance (80-600Kbps for FB vs. 0.1-10 Mbps median speeds), and even better in our scenario, despite significant differences: in clients (mobile devices vs. any device), access network (cellular mobile carrier vs. wireless fixed CN), Web proxies (centralized large servers vs. distributed small servers with network locality), and Web service and content providers (redesigned and optimized vs. unmodified content).

In order to show the margin for potential improvement of the user experience using other proxy selection strategies, we used the traffic logs to calculate the total download time of each user in the case a *min_hop* or a *random* strategy was adopted. Our calculations take into account local link latencies and the logged download times, assuming that, as shown earlier, the proxies were not saturated during the logged period. Moreover we assumed that the local links and the Internet connections of the proxies have an infinite capacity. As seen in Figure 4.15, the total download time of each user under *manual* selection,

during the analyzed month, is asymptotically better than the *random* selection while asymptotically worse than the *min_hop* selection. Considering that the proxies are not the bottleneck, as in the studied scenario, this result shows that a proxy selection mechanism has potential to improve user experience of the proxy service.

4.4. Conclusions

The analysis of the guifi.net proxy service describes a crowdsourced, social solidarity driven, free basic Internet service built from many small proxy servers spread across a regional CN, contributed by locals for locals. These proxies act as gateways to Web content, that can be cached and shared among clients or act as middleboxes for HTTPS transfers, the majority of the traffic, also potentially enhancing the users' privacy.

The analysis provides an answer to our first research question and a basis for the improvements proposed in our work. It confirms the trend to non-cacheable content, small cacheable objects, and therefore small object caches that can even fit in RAM. Therefore, in the studied scenario Web proxies behave almost like Application Layer gateways. Proxies have a small number of clients, ranging from 14 to 101 per day, with a maximum of 10 users in a 10 second period. Moreover, there is a good balance of traffic and number of clients per proxy despite the manual proxy selection, driven by locality (same zone), client choice and advice from neighbors. The system is simple and practical at a local scale, since each proxy is independent and clients just switch to their next choice in case of failure of their proxy.

The service has satisfactory performance (0.1-10Mbps, good client-proxy latency), with no perceived proxy Internet uplink, access network or service congestion, despite the typical daily patterns of usage. That can be attributed to the use of small servers spread over the regional access network, close to end-users with locality preference. Nevertheless, all the above observations apply to a local use of the service, considering only clients from the same zone and not all possible clients. Considering that the proxies are not distributed in the different zones according to the users' needs, due to the unplanned network deployment, network-wide usage of the service and coordination between clients and proxies of different zones is not trivial.

Chapter 5

Web Proxy Selection

As a consequence of the lack of regulation, presented in Chapter 4, and despite being a critical service for the community, current proxy gateway services are quite fragile, especially considering large-scale usage. The possible inability of matching the offered Web access resources with the demand can lead to inefficient usage of the service and bad user experience. Moreover, similar effects can result when the traffic of the users is routed through congested links.

In this chapter we focus on the challenges to improve Web access experience in a heterogeneous large-scale inter-WMN community, using a pool of shared Web proxies. This contributes to answer the second research question (RQ2), as described in Section 2.1. The challenge is that client-nodes have to select the right proxy according to the network path performance and the status of available Web proxies. This is related to the net effect of the service, a large population of C clients who can browse the Web benefiting from the aggregated capacity of a pool of P Web proxies, with $C \gg P$, over a heterogeneous WMN infrastructure, at a fraction of the cost of C Internet connections.

Our solution to the above problem is a selection mechanism avoids proxies under heavy load and slow internal network paths. The overhead is linear to the number of clients and proxies. We proposed and evaluated a mechanism where clients use two latency-based metrics to rank proxies and select the top ones in terms of QoE, or to switch to the next best proxy when performance degrades. The proposed mechanism is client-side, as described in the overview of our approach in Section 5.1. First, we evaluated a network performance

metric based on the usage of the Vivaldi network coordinates and for external nodes to the Vivaldi network (Web proxies), in an heterogeneous wireless network environment, described in Section 5.2). Second, Section 5.3, we designed and evaluated a Web proxy performance estimator based on TTFB, which is typically used to measure Web servers. Our evaluation, in Section 5.4, confirms that our mechanism can avoid congestion, while maintaining a low traffic overhead as shown in Section 5.5.

The metrics and the client selection mechanism were evaluated in real nodes and links in guifi.net, using the Community-Lab.net [Nav+16a] experimental infrastructure. In this mechanism, nodes are acting as clients interacting with a set of guifi.net Web proxies. Experimental results show that our proposal is reliable and effective: our method is able to provide good measurements of client-proxy and proxy-Internet latencies, following its variability. We found out that our client selection mechanism is cost-effective in finding proxies that result in good Web performance and QoE for users. Our results show improvements in the cost-benefit in comparison with other quick-to-measure alternatives (such as Vivaldi-only and minimum hops).

5.1. Overview

Our goal is to design a practicalbest-effort scheme where clients (user nodes) can select a proxy using network and proxy performance metrics that would not require the modification of any non-client network components, and that could function in a heterogeneous environment. To this end, we implemented an estimation-based monitoring framework for proxy selection, where clients cooperate by sharing their network and proxy performance estimations in order to prioritize their list of known proxies. This allows clients to be able to make an informed proxy selection decision. Unlike other proposals, the framework does not try to find an optimal client-proxy assignment, but helps clients avoid bad choices (overloaded proxies, slow Internet connections or slow internal network path) that would degrade significantly their service experience. The non-optimality is the price we have to pay in order to achieve a scalable and practical solution that can be applied in real heterogeneous WMNs while retaining a low overhead. The proposed framework is user-friendly. This means that the users neither manage the proxy selection nor the switching. They just need to install our component on their client nodes. It is also important

to note that we do not cover the orthogonal problem of proxy discovery in this work. We assume that the set of proxies is known beforehand by the clients. An abstract view of our approach can be seen in Figure 5.1.

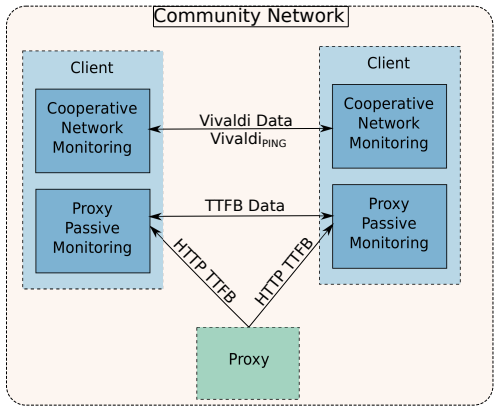

Figure 5.1: Proxy Selection Mechanism Design.

The network performance estimator provides estimates of the client-client and client-proxy network latency. It is based on Vivaldi network coordinates [Dab+04] and extended in a similar way to [LSP08] in order to estimate the round-trip latency of nodes that are not part of the Vivaldi network – the proxies. All the clients of the proxy selection system participate in the Vivaldi network, exchanging a small amount of messages periodically, which allows them to maintain an updated view of the latencies across them. Moreover, each client periodically has to monitor one of the proxies and share this information with the rest of the clients. As we demonstrate in Section 5.2, these measurements suffice to allow the clients to create a preference list, which orders the proxies according to their network latency.

The proxy performance estimator provides an insight of the proxy load, illustrating the quality of the service currently being provided. It is based on the widely used practical assumption that the TTFB of an HTTP request can reflect the service performance [Sun+13; CST15]. In our framework, each client passively calculates the TTFB of the HTTP replies that it receives from its proxy. Then the client can use this value to estimate the load of its proxy and share it with its Vivaldi neighbors. Notice that caching does not help in this case.

Considering that the proxies in the studied scenario (Chapter 4) act mostly as Web traffic forwarders, the driving ideas of our approach are applicable as well to other gateways that function in lower layers of the networking stack, such as SOCKS, when used for Web content.

5.1.1. System Model

For the model description we assume a static wireless network topology. No assumptions are made concerning the quality of the network, and dynamic link conditions are allowed, though a very slow link is indistinguishable from a very congested link. We used latency as our load metric, for both links and proxies. Figure 5.2 describes the most important transactions and latencies during an HTTP request of a client that we are going to use in our model.

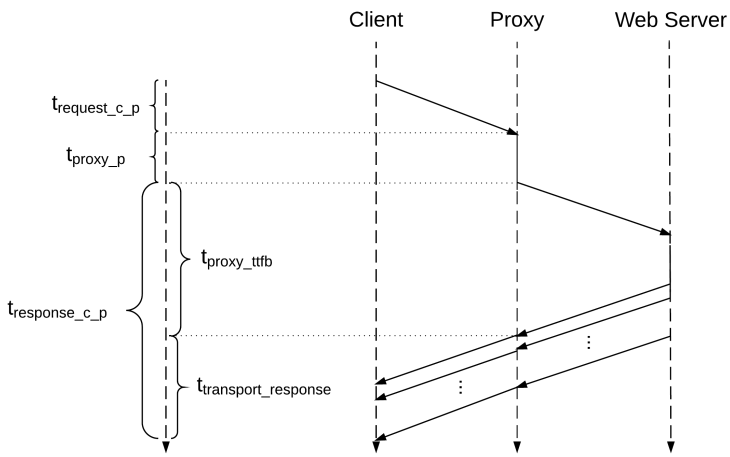

Figure 5.2: Client HTTP Request.

Let C denote the set of clients (user nodes), and P denote the set of proxies. For every request that a client $c \in C$ is sending to a proxy $p \in P$, Equation (5.1) shows the experienced latency model.

$$t_{lat} \approx t_{request\_c\_p} + t_{proxy\_p} + t_{response\_c\_p} \tag{5.1}$$

$$t_{request\_c\_p} \approx A * t_{mesh\_rtt\_c\_p} \tag{5.2}$$

$$t_{lat} \approx 2 * t_{mesh\_rtt\_c\_p} + t_{proxy\_p} + t_{response\_c\_p} \tag{5.3}$$

Where $t_{request\_c\_p}$ represents the time required by client c to connect to proxy p and send the request. It is proportional to the RTT between c and p, $t_{mesh\_rtt\_c\_p}$. The $t_{mesh\_rtt\_c\_p}$ latency, Equation (5.2), depends on the network conditions of the chosen path between client c and proxy p. For the rest of this chapter we assume that A equals to 2, which corresponds to the client-proxy TCP handshake (SYN, SYN-ACK, ACK) and the HTTP request. The t_{proxy_p} latency represents the total time that proxy p needs to process the request until it initiates the request to the remote server. This includes the time the request is waiting before starting to be served, which is a good indicator of the load of proxy p, as it correlates directly with the number of outstanding proxy requests yet to be served. It is assumed that at a given point in time different clients experience the same t_{proxy_p} if they use proxy p, independently of who is measuring it. In Section 5.3 we validated this assumption. Finally, $t_{response\_c\_p}$ is the time that proxy p takes to complete the HTTP request. This time depends on the load and capacity of the proxy's Internet connection and on the latency to access and retrieve the content, related to the distance from the content and content availability. From all the above, we deduced that the request latency can be approximated by Equation (5.3).

We argue that t_{mesh_rtt} and t_{proxy} can provide to the clients a good preference indicator allowing them to avoid saturated proxies and proxies located on slow paths. Section 5.2 describes how we use Vivaldi to estimate t_{mesh_rtt}, while Section 5.3 elaborates on how TTFB measurements can be used to estimate t_{proxy}.

5.1.2. Experimental Environment

In order to assess our desing, we experimented separately with each component of our solution. Following the practical approach of our work, we decided to perform our experiments in guifi.net, under real heterogeneous wireless network conditions. For the experiments, we were given access to 5 end-nodes across different guifi.net networks and 3 proxies that are also being used by guifi.net users. The nodes and the proxies are distributed in various locations of Catalunya. Despite the small scale of our experiments, we were still able to understand the behavior of the presented components as explained below.

As explained in Section 5.2, proxies do not actively participate in the measurements, they nevertheless need to respond to UDP pings, allowing clients

to estimate their round-trip latency. While for the results presented here we used a UDP echo server on the proxies, an obstacle which can be practically overcome with tools such as Scriptroute [SWA03].

5.2. Network Performance

In this section we describe and demonstrate how an extended version of Vivaldi [Dab+04] can be used to estimate the current performance of the network, expressed as a latency metric, helping the clients to avoid overloaded paths.

5.2.1. Estimating Latency with Network Coordinates

Each client in our system participates in a Vivaldi network coordinates system to estimate its round-trip latencies to the other clients. Based on the clients' network coordinates, we implemented the ideas described in [LSP08] on calculating latency estimates for nodes external to the Vivaldi network, modifying them to provide more accuracy. We show that this allows the clients to maintain an updated view of their latency towards each proxy, while excluding proxies from the Vivaldi network. Although Vivaldi was designed to predict latencies between hosts on the Internet (mostly wired), we show that it can also be used to predict latencies in WMNs despite the RTT variations caused by the wireless environment.

Vivaldi estimates RTT by sending UDP pings between nodes. Each Vivaldi node maintains a list of $C+R$ *neighbors*: C that are estimated to be the closest nodes, and R other random nodes, located anywhere within the network. The algorithm works in *rounds*, which are triggered every T seconds. In every round, a node randomly selects a neighbor from the list, sends N UDP pings to it, and asks it to send back its own pings and its neighbors' coordinates. The variables C and R can be tuned depending on the size of the network and the topology in order to increase random/remote client-node discovery or to create strong local clusters. The variable N affects the accuracy of the prediction in exchange for the ping traffic overhead.

Using the client-client estimations we can satisfactorily predict round-trip latency from a Vivaldi node to each proxy, leaving the proxies unmodified since they do not actively participate in the network coordinates system. In this *extended Vivaldi* version, each Vivaldi node maintains coordinates that

represent $C + R$ proxies, as described above. In every round, a node performs N UDP pings to a proxy p, which is selected in a similar manner compared to how the node selects its neighbors. Then, the node updates the coordinates it maintains for p and shares the measured latency with its selected neighbor for this round. Then, the neighbor updates the coordinates that it maintains for proxy p as described in [LSP08].

5.2.2. Latency Estimation Evaluation

For our evaluation, similarly to [Dab+04], we define the error of a path as the absolute difference between the predicted RTT for the path (using the coordinates for the two nodes located at the ends of the link) and the actual RTT. The error of a node is defined as the median of the path errors for paths involving that node. The error of the system is defined as the median of the node errors for all nodes.

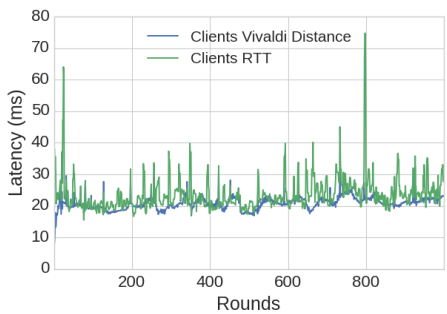

Figure 5.3: Clients' estimated latency can track RTT behavior but with various faulty spikes.

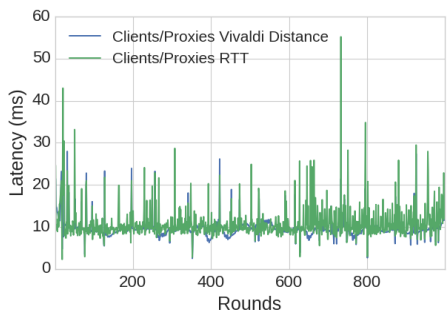

Figure 5.4: Clients/Proxies estimated latency successfully tracks RTT behavior.

We performed experiments under the described environment, in order to characterize the behavior of the Vivaldi coordinates in a heterogeneous large-scale WMN. In our first experiment clients were using Vivaldi to estimate the latencies between them and the extended version of Vivaldi to estimate their RTT to the proxies. This way we were able to understand the predictive potential of the selected algorithms. It is worth reminding that our experiment was executed on nodes that participate in a real network and therefore were processing real network traffic and were using shared links. Figures 5.3

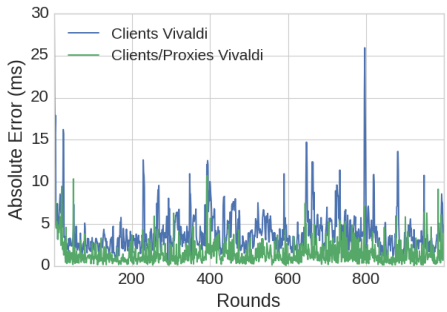

Figure 5.5: Clients/Proxies estimated RTT error is lower than Clients' RTT error.

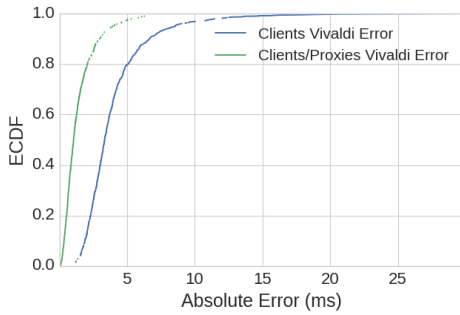

Figure 5.6: Clients/Proxies estimated RTT error is asymptotically lower than Clients' RTT error.

and 5.4 show the real and predicted latencies between clients throughout the experiment. The median latency between the clients was 22.29 ms, while the predicted median was 20.82 ms. The median latency between clients and proxies was 9.8 ms while the corresponding predicted median was 9.36 ms. Figure 5.5 depicts the absolute prediction error of the Vivaldi estimation between clients as well as the one between clients and proxies. We observe that the error of the latency prediction between clients and proxies is lower. This can be attributed to the smaller variation of the real latency between clients and proxies, but also to our improvements related to [LSP08]. The ECDFn function of the absolute errors of the prediction, as seen in Figure 5.6, shows that the median absolute error of the predicted latency between clients was 3.37 ms, while 80% of the experimentation time the nodes had a median error of less than 5 ms. As far as client-proxy Vivaldi latency prediction is concerned, the median absolute prediction error is 1.07 ms, while 80% of the experimentation time the nodes had a median error of less than 2.5 ms.

In our second experiment we tested the ability of Vivaldi's extended version to adapt to network changes. Figure 5.7 shows that there is some latency for Vivaldi to adapt to latency changes between the clients, taking around 30 rounds to adjust its estimates satisfactorily. However, as seen in Figure 5.8, proxy estimates are much faster to adapt, taking around 12 rounds to re-adjust the estimates.

We showed that our system, under real heterogeneous mesh network conditions, can estimate the RTT between clients as well as between clients and proxies with errors smaller than 5ms and 2.5ms respectively. These low prediction and triangulation errors (median relative error in the range of 10%) are comparable to the original Vivaldi on the Internet. Moreover, we observed that our estimation can eventually trace serious anomalies in the latency of paths. Therefore, we argue that these estimates are satisfactory in order to prioritize paths from clients to proxies that present differences in latency higher than 5 ms and avoid congested paths, when this events last for significant amount of time.

5.2.3. Estimation Based on Other Metrics

We decided to focus on latency as the most relevant distance metric, considering that the type of Web access most essential to users is typically comprised of small HTTP requests and replies (Web requests to update mailboxes, blogs and social network sites, messaging apps, and notifications as well as streaming of videos, small audio files or image downloads). This decision is backed by our analysis in Chapter 4 showing that around 32% of the transfers are less than 10MB, 32% less than 20MB, and 51% less than 30MB. In this context, latency is the most relevant metric to consider in order to assess quality-of-service as perceived by users.

Considering other Web access patterns where larger content dominates (non-essential, though popular), we can adapt the Vivaldi network coordinates system to employ different distance metrics, estimating minimum (or median) sustained throughput (instead of latency) without significant additional overhead. Existing approaches, such as Spruce [SKK03], exploit the probe gap model (PGM) to collect information about time gaps over consecutive probe packets. This approach avoids the need for large data transfer to infer throughput that would seriously affect clients, proxies, thus hindering overall system scalability. In the specific context of multimedia streaming, related estimation and adaptation techniques for video streaming over HTTP [Li+14] could be used instead.

5.3. Proxy Performance Estimation

This section shows how TTFB can estimate the current performance of the proxy, expressed as a latency metric, allowing clients to rank choices, avoiding

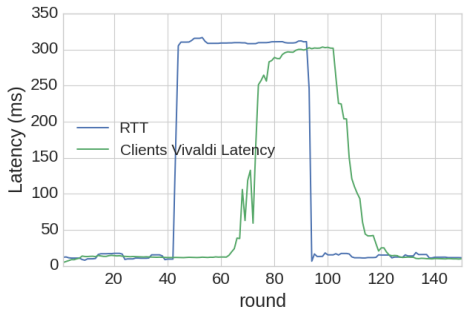

Figure 5.7: Predicted Clients' RTT reflects the changes in real RTT for clients but slowly.

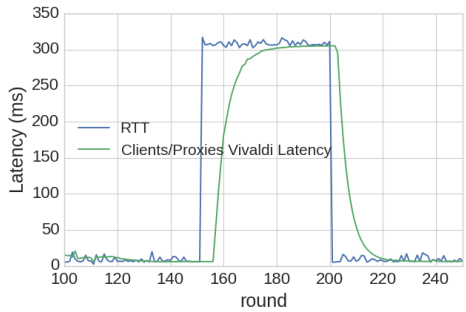

Figure 5.8: Predicted Clients/Proxies RTT adapts fast to changes in real RTT.

overloaded proxies and proxies with Internet connections that exhibit high delays.

5.3.1. Estimating Proxy Load with TTFB

TTFB has been widely used as a metric in real deployments but also in recent Internet measurement research [Sun+13; CST15] to indicate the responsiveness of a Web service, since it combines the TCP connection time and the remote server processing time. Moreover, TTFB is a useful Web performance estimator because it is measured passively on the client-side, leveraging information from the already existing client traffic. Regardless, our scenario is more complicated, since we use client-side TTFB measurements to estimate the performance of the proxy between the client and the requested content.

Assuming that t_{proxy_ttfb} is the time the proxy needs to receive the first byte of response from the server then $t_{response\_c\_p}$ from Equation (5.3) can also be expressed as in Equation (5.4), where $t_{transport\_response}$ is the time until the client has received the complete response. Both t_{proxy_ttfb} and $t_{transport\_response}$ depend on the available bandwidth of the proxy Internet connection, and the delays on the path from the proxy to the destination server, as well as the responsiveness of the end-server. Additionally, $t_{transport\_response}$ depends on the performance of the client-proxy path. Considering Equation (5.3), the

TTFB as measured on the client-side can be expressed as Equation (5.5).

$$t_{response\_c\_p} \approx t_{proxy\_ttfb} + t_{transport\_response} \tag{5.4}$$

$$t_{ttfb\_c\_p} \approx 2 * t_{mesh\_rtt\_c\_p} + t_{proxy\_p} + t_{proxy\_ttfb} \tag{5.5}$$

$$t_{proxy\_p} \approx t_{ttfb\_c\_p} - 2 * t_{mesh\_rtt\_c\_p} \tag{5.6}$$

t_{proxy_ttfb} differs depending on the proxy, the remote server and the requested content. The analysis of the variability of different t_{proxy_ttfb} latencies, related to how well the proxies are connected to specific remote servers, lies beyond the scope of this work. Therefore, in our current work we chose not to study t_{proxy_ttfb} and assume it is constant for each proxy. For the rest of this work it is assumed a uniform Internet access model, where all the clients are trying to access the same content that is always available, located on remote servers within similar distance from all the proxies and with all of them having the same Internet connection bandwidth capacity.

Under these assumptions, based on Equations (5.3) and (5.5), the latency incurred by the proxy could be expressed as in Equation (5.6). t_{proxy_p} can provide an estimation of the proxy performance, calculated by Equation (5.6) with the measured TTFB on the client-side and the network. However, the TTFB measurements can be very noisy due to outliers (sometimes packets are significantly delayed due the proxy or network load, or proxies may complete a request quickly despite heavy load). To minimize the effect of noise on our estimation, we defined the *extended TTFB*, where the obtained t_{proxy_p} values are filtered with an exponential moving average which can be tuned by a parameter α. When α is too high, the effect of noise in the measurements leaks into the filtered value, while when α is too low, the filtered values adapt slower to the measured real values, smoothing the peaks and valleys [KN01]. Moreover, since the TTFB of HTTP requests is measured periodically, we must handle delays that are higher than the measurement period. Therefore, we introduced a penalty scheme to the *extended TTFB*, assuming that the request will eventually be completed. Our scheme is based on the simple idea that the TTFB value will be at least as high as the time the client waited for it. Thus, if a client has not received the first byte for longer than the last t_{proxy_p} value, then the estimated value keeps increasing for every measurement period until it is received.

Clients periodically exchange the calculated t_{proxy_p}, thus reducing the need for probing, as the value indicates how efficient a proxy is at serving requests

for any client. These messages are forwarded through the Vivaldi network. Currently, it is assumed that the client is performing HTTP requests sequentially. However, this is not a realistic assumption, since in a typical scenario a browser generates multiple parallel HTTP requests targeting different server, since the estimation of the TTFB value of a proxy obtained from multiple HTTP requests is an open issue.

5.3.2. Proxy Load Estimation Evaluation

In our first experiment we evaluated the relation between the t_{proxy_p} and the proxy load. The proxy load is represented by various variables monitored on the proxy, including the CPU load and the number of incoming and outgoing packets per second on the internal and external interfaces. Figure 5.9 shows the comparison between the normalized median of the proxy variables compared to the estimation and the extended TTFB estimation of t_{proxy_p}. The proxy is saturated with external requests 150 seconds after the beginning of the experiment and t_{proxy_p} starts showing high peaks while the extended TTFB estimator presents a more clear relation to the load behavior. Figure 5.10 demonstrates another perspective of the relation between the proxy load and the extended TTFB estimator, including the plot of the Principal Component Analysis which demonstrates that, over a relatively low threshold value, the higher the load values are the higher the values of the extended TTFB estimator. As a result, we argue that our extended TTFB estimator can be used to detect heavily saturated proxies.

The goal of our second experiment was to evaluate how our estimator responds to proxies having Internet connections with significant delays. Hence, we introduced artificial network latency on the external network interface of the proxy. As seen in Figure 5.11, both the simple and the extended TTFB estimators successfully measure the introduced delay. Nevertheless, the extended estimator appears to need more time to return to normal levels, as expected.

5.3.3. Sharing Estimations Across Clients

Despite the fact that our estimators can detect when a proxy is overloaded, we had to verify that the estimate measured from client c for proxy p can be useful for other clients as well. To investigate this issue, we performed an experiment where one single proxy was used to serve all the nodes. Figure 5.12

represents the Spearman's rank correlation coefficient [Spe04] between the extended TTFB estimators of the different clients throughout the experiment. Spearman's rank correlation coefficient aims to identify correlations that can be expressed by a monotonic function, thus resulting in high values, as we observed in our result, when both of the compared sets ascend or descend similarly.

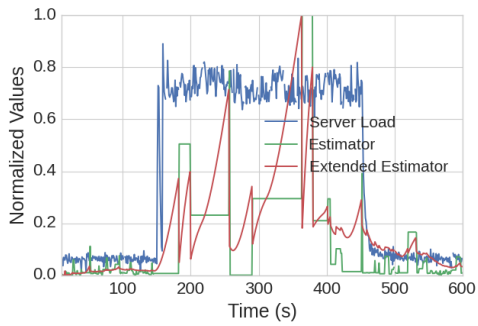

Figure 5.9: Estimators can assimilate proxy load metrics behavior ($\alpha = 0.05$).

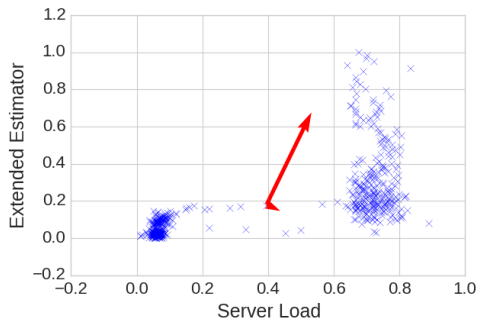

Figure 5.10: PCA: Extended estimator ($\alpha = 0.05$) can track high proxy load.

The described proxy performance estimator is not an accurate estimator in terms of absolute values, but has a behavior similar to the proxy load, enabling clients to rank choices and avoid saturated proxies. Moreover, the extended estimator calculated by one client behaves similarly throughout the different clients and can, thus, be disseminated across them reducing the overhead and allowing clients to have updated information about proxies they are not currently using.

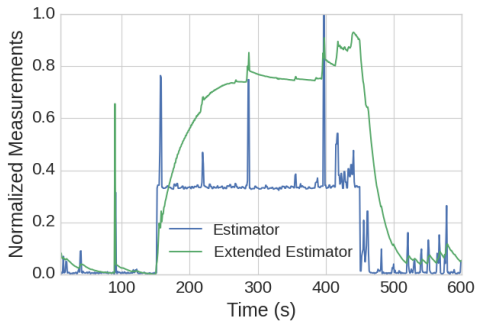

Figure 5.11: Responsiveness of estimators to Internet connection delays ($\alpha = 0.05$).

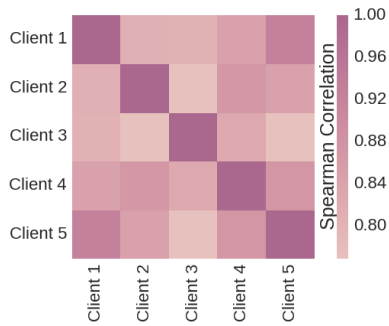

Figure 5.12: Strong correlation (Spearman's rank) between the clients' Extended TTFB estimators (α=0.05).

5.4. Proxy Selection

After describing our approach for measuring the performance of the network and the proxies, in this section we focus on how clients are able to select proxies, informed by the presented metrics. Moreover, we present an experiment where clients, adopting our solution, manage to avoid overloaded proxies, very slow internal paths and very slow proxy Internet uplinks, whereas if a minimum hop or minimum delay selection approach was to be adopted, the clients would not be able to avoid service deterioration.

On top of our performance estimation tools we built an Application Layer proxy selection platform. Each client maintains a *proxy selection table*, similarly to a routing table, where each line corresponds to a known proxy and contains the estimated distance, as described in Section 5.2, the extended estimation of the proxy latency, as described in Section 5.3 and the number of hops to that proxy. Based on this information various proxy selection strategies can be implemented. Nevertheless, the implementations need to take into account the described sensitivity of the provided estimators.

We used the provided estimators to implement a proxy selection strategy suitable to the estimators' rationale, aiming to avoid saturated proxies, proxies with saturated Internet connections, as well as proxies behind saturated paths. Therefore, the selection strategy orders the proxies according to the sum of the network and proxy latency estimations, selecting the lowest value. Our implementation avoids unnecessary alternations by defining a minimum threshold which should be overcome in order to change the selected proxy. Additionally, we implemented a recovery mechanism for situations where a proxy is not being used by any client for a significant amount of time, therefore its current performance estimation value is unknown. In order to prevent all the clients from querying the proxy at the same time, the clients maintain a personalized timeout that depends on a global recovery time, the locally last known measurement of the proxy and their personal network distance to that proxy. If the timeout is reached without receiving any updates, the client is actively probing the proxy to learn its known TTFB value. This way, we manage to force clients that are close to the proxy to be in charge of querying it and then propagate the information to the other nodes.

To evaluate our minimum load selection strategy, following an approach similar to [Ko+13], we implemented two simple proxy selection strategies based on the minimum hop (*min_hop*) and minimum network delay metric (*min_delay*), that were used to compare to the minimum load solution (*min_load*). Under the minimum hop strategy each client selects the closest proxy in terms of hops while in the minimum network delay strategy the clients select the proxy that has the smallest Vivaldi latency estimator.

The objective of the evaluation experiment was to describe how the different strategies of the clients deal with the disruptions of the provided service. The clients use the proxies selected by the routing strategies to repeatedly

download files of 1Mb from the same remote server choosing every 10 seconds - our Vivaldi period - a new proxy if necessary. The value of 1Mb was chosen because in normal conditions a client needs less than the period of 10 seconds to download the file, therefore we can evaluate more accurately the selection alterations. We adopt as evaluation metric the download time experienced by the clients. The experiment lasted 1600 seconds and was repeated for each strategy. Between 50 and 350 seconds we introduced a high amount of requests on one of the proxies. Between 550 and 850 seconds we introduce high latencies on one of the proxies an external Internet connection. Between 1050 and 1350 seconds we simulated a slow network path on one of the proxies.

Figure 5.13 depicts the median clients' download time per strategy. We observe that our strategy leads the clients to experience a very small amount of download time peaks, especially compared to the static *min_hop* solution. The y axis of the plot is limited to 2 seconds for easier comparison, nevertheless, the overall distribution of the values can be seen in Figure 5.14. As depicted, *min_hop* and *min_delay* present higher average values compared to *min_load* (0.76s, 0.71s and 0.48s respectively). Most importantly, related to avoiding overloaded options, *min_hop* and *min_delay* have many more and significantly higher peaks, compared to *min_load* (maximum 6.33s, 4.89s and 1.23s respectively). This is even more apparent for *min_hop*, which is a static strategy. *min_load* manages to minimize the number of peaks, confirming our argument that it succeeds to avoid the saturated options. The manner in which *min_load* is avoiding the congested options is also shown in Figure 5.15, where we observe that clients avoid *proxy_3* when saturated by requests (150-350 seconds), as well as *proxy_2* in the ranges of 550-650 seconds, and 1050-1350 seconds where we simulate the network path and Internet connection latency respectively. It is also worth pointing out that in the performed experiment *min_delay* and *min_hop* do not appear to be affected by some of the obstacles introduced, but we have verified that this is a result of the specific experiment conditions (network latencies and distances) and not of their ability to avoid it.

The results we presented in this section verify how the performance estimators in the previous sections can be used by clients to rank, and make informed choices from a large set of proxy Internet gateways, avoiding proxies that would deteriorate their user experience.

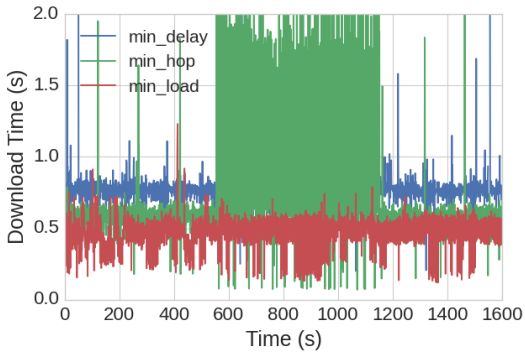

Figure 5.13: Median client download time for 1Mb per strategy.

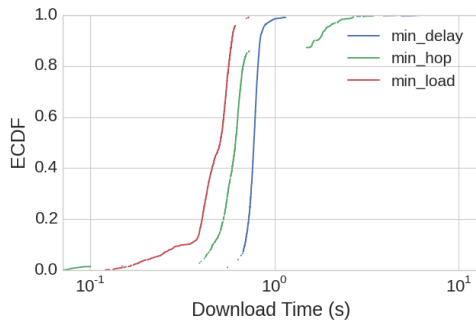

Figure 5.14: Improvement of median user download time using `min_load`.

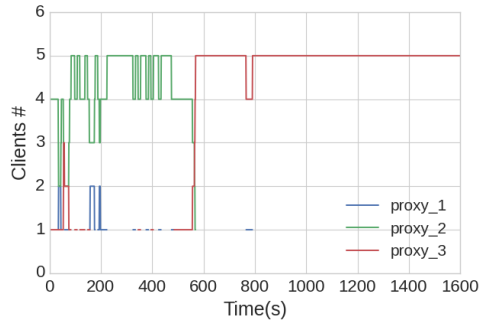

Figure 5.15: Clients avoid bad choices using `min_load` strategy.

5.5. Overhead Analysis

The two performance estimation components of our system function independently from one another. Thus, the total overhead is:

$$overhead = overhead_{vivaldi} + overhead_{ttfb} \tag{5.7}$$

According to the challenges that Vivaldi [Dab+04] faces by design, a network coordinates system should produce a minimal amount of overhead traffic when probing. The overhead network traffic generated by Vivaldi is, in bytes per second:

$$overhead_{vivaldi} = (2 * ping_{size} * ping_{freq} + data) * n \tag{5.8}$$

$$data_{vivaldi} = (n_p * 160 + n_n * 160 + 10)/round_{period} \tag{5.9}$$

$$ping_{freq} = round_{pings}/round_{period} \tag{5.10}$$

In the formulas above, n is the number of nodes in the Vivaldi system and n_n and p_n are, correspondingly, the maximum number of known neighbors and proxies, while 160 bytes is the average size of the reply that contains the network coordinates. We can see that the overhead of Vivaldi increases linearly with the amount of participants. Vivaldi works in rounds: every round each node sends a few pings to each of its neighbors, and rounds occur every few seconds. In our deployment we use 8 pings per round, with a round starting every 10 seconds. Moreover, in our case it corresponds to one neighbor plus one proxy, and the maximum number of neighbors and proxies is 8. That equates to 436 bytes per second per client, which is acceptable even in a WMN environment. For example, assuming all the 30,000 nodes of guifi.net were clients, the overhead would be approximately 1.5MB/s distributed all over the network, which sums up to be 1.6% of the average daily incoming Internet traffic [Bai+15] (data from 2015).

The TTFB metrics are passively collected for the proxy currently selected by the client, and then shared between the nodes of the system. Nevertheless, we may ping a proxy if we have not had any metrics for a certain time period as described in Section 5.4. The network overhead of the proxy TTFB protocol

(in bytes per second) is:

$$overhead_{ttfb} = O(C) * payload/timeout \tag{5.11}$$

$$payload = payload_{request} + payload_{response} \tag{5.12}$$

$$timeout = m_1 * proxy_{distance} + m_2 * num\_closer + b \tag{5.13}$$

Each client requeries the shared proxy information after reaching the personal *timeout* to stay updated. If the m parameters are too low, the information will not have time to propagate and many nodes will query the proxy. However, if the m parameters are set too high, it may take a long time until a node is finally queried.

To find out the expected number of rounds until a datum is disseminated from a node until it is globally known, we modeled a simplified version of the problem. At each time, k nodes know a specific piece of information. The other $N-k$ nodes each query a node at random, and if they happen to query a node that knows the information, they come to know it as well. It is assumed that any node is able to contact any other node at any time, and that the set of nodes is static. Two nodes may query the same node, and may query the same node more than once over the course of the process. That means the next-value of k, k', is distributed as $k + Binomial\left(N-k, \frac{k}{N-1}\right)$.

The behavior of the process depends exclusively on the present state, so it can be modelled as an absorbing Markov Chain. Computing the average absorption time [KS60] of this Markov chain is equivalent to calculating the expected number of rounds until the information is globally known. For the $N = 30,000$ nodes currently registered in guifi.net, it equates to 19 rounds. From the simulations using this model, the complexity seems to be in the order of $O(logN)$.

Moreover, it is important to notice that Equation (5.11) assumes that m and b parameters are correctly tuned so that the proxy is contacted by a very low number of nodes with high probability.

5.5.1. Scalability Assessment

The scalability of our approach stems from four main factors. First, the low client and proxy overhead which was already addressed in this section. Second,

the lack of need for centralized coordination, evident in our approach, having no central coordinator in charge of global decisions. Therefore there are no participants whose processing, state or, message load would grow boundless as the number of clients and proxies increase. The third factor is bounded storage and traffic, where state size and message count exchanged by each client increase logarithmically with the number of participants. Additionally, the gossip-like propagation of the estimators ensures fast propagation under an increasing number of nodes. The fourth factor is the good convergence of the estimators, where the Vivaldi network estimator is proven to converge in large-scale networks for the selected parameters. The global convergence of selection is not trivial. We are considering probabilistic strategies from a time perspective as well as individual selection choices that would provide stable aggregate results without further overhead as the system scales, given the decentralized nature of our solution that avoids the overhead of global coordination.

5.6. Conclusions

This chapter proposed a client-side proxy selection mechanism to make good choices in terms of QoE or performance, taking into account the state of the local network, proxy gateways and their Internet connection. We developed latency-based metrics for the selection that are capable of predicting and triangulating performance indicators, in a reliable and inexpensive manner. This mechanism avoids heavily congested proxies, proxies with slow Internet uplinks and slow internal network paths, while the introduced traffic overhead is linear to the number of the clients and proxies. This contribution addresses the first part of our second research question (RQ2).

Chapter 6

Exploiting Traffic Patterns and Network Locality

The users-proxies selection regulation mechanism described in Chapter 5 succeeds to dynamically assign users to proxies in a best effort manner, without, though, considering information that is related to the user traffic and the network infrastructure. In this chapter we present how this kind of information can be leveraged leading to a more informed proxy selection from the users and improving the final user experience as well as the overall service performance. This contributes to completing the answer to the first part of our second research question (RQ2). This study considers several data inputs; e.g., the patterns of usage from service logs, the design choices and implications (considering client and proxy choices) according to patterns of usage, and the relative location of users and proxies in the network topology. The results show the design space for cooperative choices, the involved trade-offs, and the effects of the above metrics on the service cost and performance.

Section 6.1 looks at the behavior and clustering of users according to content and network locality, while also analyzing the impact on the criteria for proxy selection. We present an analysis of the current scenario, limitations and potential for improvement from the perspective of the access network in Section 6.2, of the proxies in Section 6.3, and of the users in Section 6.4, concluding in Section 6.5.

6.1. Clustering of Users

In this section we present and discuss and evaluate different strategies of clustering the Web proxy service users based on properties that can be leveraged to improve the service. We started the study by exploring data concerning the service usage, in order to group users according to their behavior. Then, we identified the graph communities that exist in the network to analyze the factor of network locality.

6.1.1. Clustering According to Usage

Patterns of traffic usage can provide useful information for distributing the load across the different proxies. In our scenario, we used information for each user from the available proxy logs, including total traffic size, content type, size of traffic per content and the hours of the day where the user was active. For the analysis of service usage according to the metrics, and based on [Ber06], we considered four different types of clustering algorithms: K-means, suitable for generic applications, DBSCAN and Ward's hierarchical clustering (HC) that can trace complex patterns. The input used by the algorithms was the total data transferred per user in bytes, as well as the corresponding amount of traffic for contents that constitute a large amount of the total service traffic, like video (>30%), image (6%) and HTML (2%). We experimented with various cluster sizes for K-means and Ward's HC, including well-known empirical estimation methods like the 'elbow method', as well as many parameters for DBSCAN. Table 6.1 presents the optimal results for each method in terms of cluster validation. As a validation metric, we used the coefficient Silhouette score [Rou87], which is a common metric for that purpose and takes values in [-1,1]. As described in Table 6.1, for all the cases there is a big cluster of 450-480 users with a Silhouette score of 0.9, indicating a very strong cluster density. Nonetheless, the rest of the users belong to overlapping clusters, with scores close to zero. After manually reviewing other results of the algorithms for getting a better insight, as is the standard process in such cases, we chose Ward's HC method with 3 clusters, that partitions the users in one large consistent cluster and two small overlapping clusters, minimizing, though, overlapping elements.

Method	Clusters #	Cluster Size	Cluster Silhouettes
DBSCAN	2	7, 499	0.03, 0.90
Ward's	2	33, 473	-0.04, 0.89
	3	4, 29, 473	0.30, 0.01, 0.87
K-Means	2	21, 485	0.09, 0.89
	3	10, 44, 452	0.07, -0.04, 0.88

Table 6.1: Results from clustering algorithms on usage.

ID	Size	Silhouette	Characteristics	Alias
1	473	0.87	Low total traffic	Light
2	29	0.01	Medium total and video/images traffic	Medium
3	4	0.30	High total and video/images traffic	Heavy

Table 6.2: Description of User Behavior Clusters (Ward's).

Table 6.2 presents the characteristics of the clusters, as formed using Ward's HC for 3 clusters. We found two consistent clusters of users with distinct properties. The Figure 6.1 depicts the comparison of the clusters in terms of traffic and size (number of users). Cluster 1 of *light* users, includes the majority of users and their profile consists of generating very low traffic, as low as 1% of the maximum noticed per user value, mostly HTML browsing. Cluster 3, *heavy* users, consists of only 4 users and it is characterized by high total traffic compared to the cluster size, confirmed by the fact the average traffic per user is more than 60% of the maximum logged traffic per user. Users from cluster 3 spent the majority of their traffic on downloading video and images. Cluster 2, *medium* users, presents an intermediate behavior; nevertheless, following the patterns of the *heavy* users. *Medium* users create a significant portion of the total traffic, around 20% of the maximum value, which they consume mostly on videos and images.

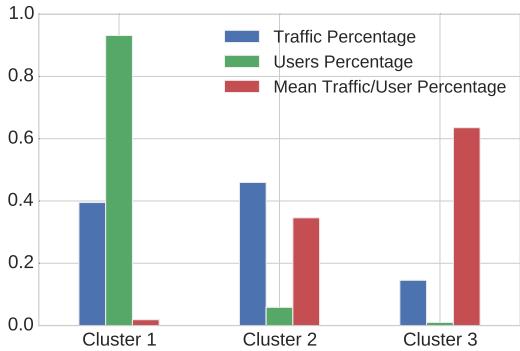

Figure 6.1: Traffic and user percentage per cluster.

6.1.2. Clustering According to Network Locality

Matching users to proxies based on locality on the backbone network, we can minimize traffic between components of the network that are not well connected in terms of number of links. This can assist in avoiding saturation of links that connect the different components. Indeed, as presented later, this approach manages to distribute more efficiently the load on the backbone links. Based on [LF09], we chose three of the most prominent community detection algorithms: Spinglass, Multilevel and Infomap. The data input for the algorithms is the backbone graph, consisting of 48 nodes. Moreover, since the studied guifi.net zone has a small well-connected backbone, with many clients connected to the routers of the backbone, we used the number of clients using those routers to establish the graph weight for the InfoMap algorithm. The weight for each link was defined as the average time to transfer a single byte according to our topology dataset. The results of the different algorithms can be seen in Table 6.3. We compared the algorithms using the modularity score, which lies in the range [-1/2, 1), where the higher the value, the more consistent the community. Experimenting with the algorithms we noticed that the node size argument of the Infomap does not affect significantly the output, thus Infomap does not offer any additional information. Therefore, we chose the Multi-level Algorithm, that has the highest modularity score and smaller number of clusters, considering the small backbone.

	Infomap	Multilevel	Spinglass
Modularity	0.699	0.712	0.702
Clusters	12	9	15

Table 6.3: Comparison of community detection algorithms.

Figure 6.2 shows the resulting graph for the Multi-level algorithm. The squares represent the routers that operate also as proxies. As depicted, the proxies are not well positioned relatively to the network clusters, considering that most of the clusters have none, while one of the clusters has two proxies. Additionally, we observed that there are clusters that are poorly connected to their neighboring clusters, resulting in an infrastructure that is far from ideal. For the rest of this work it is assumed that all the clients of a router belong to the cluster of that router, since the Multi-level algorithm we adopted is not affected by the node weight, which is how clients are represented in our graph.

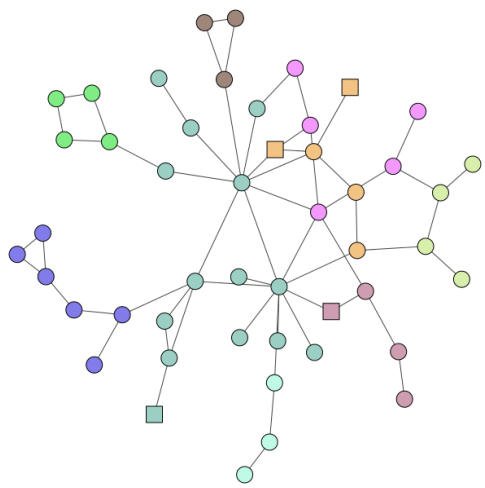

Figure 6.2: Multi-level community detection for the backbone network (colors).

6.1.3. Influence of the Criteria for Proxy Selection

Building upon our clustering analysis, we present simulations (in the context defined in Section 2.2), that exploit the two clustering techniques in algorithms

for proxy selection, in order to provide alternatives to the current manual proxy selection. The objective is to demonstrate the impact of network locality and user traffic behavior on the performance of the proxy service and user experience. Thus, we show how they can be used to inform the design of a proxy selection service.

Next, we present an initial evaluation of the mentioned techniques under the perspectives of the network, the proxies and the users. It is important to clarify that our algorithms implement one of several ways to use the information from user behavior clustering and community detection. The first algorithm we implemented, referred as *data_cluster*, uses clustering of user behavior (Ward's HC) to assign equivalent user load to each proxy by equally distributing the users of each cluster. In the cases where a new user has to be assigned to a proxy and all existing assignments from the clusters are equally balanced, the algorithm selects a proxy randomly. The second algorithm, referred to as *network_cluster*, uses graph community detection (Multi-level) to assign users to proxies according to the proximity of their community. For instance, a user with an available proxy in his/her community will be assigned to this proxy, while in the opposite case, he/she will be assigned to the proxy that is located in the closest community in the graph. In case of equal proximity, the proxy selection is random. Finally, we implemented an algorithm that combines both solutions. The algorithm *data+network* is mainly based on the *data_cluster* algorithm, but in case it encounters equal assignments, it uses the *network_cluster* algorithm to decide.

All these algorithms were compared to the current *manual* service selection. Clients (Web browsers) have a manually defined or adjusted list of proxy servers. The initial configuration is based on hints from other nearby users, or by downloading a list from a local guifi.net forum. The adjustments come from similar sources, personal usage experience, hints from other users or news about new proxies being offered. Web browsers switch to another proxy server just when a proxy fails to respond and do not provide load balancing, or more effective choices considering degradation, congestion signals or relative performance.

6.2. Network Perspective

The impact of the algorithms presented in the previous section, on the network is studied according to the total bytes transferred through each link during

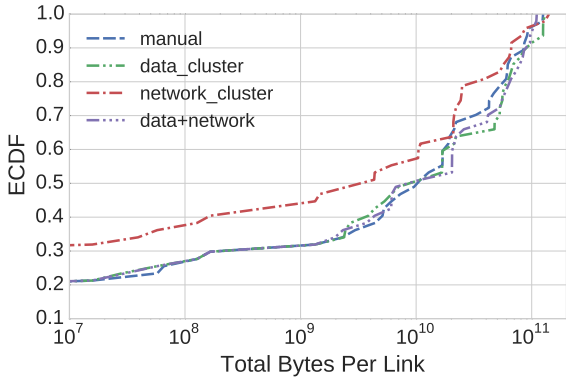

Figure 6.3: Comparison of total link Bytes per strategy ECDF.

the simulation. Possible retransmissions are not taken into account, and it is assumed that the links cannot be saturated and have always the same performance, even across different links.

As shown in Figure 6.3, the *network_cluster* algorithm outperforms significantly the other algorithms in distributing the load on the links. It maintains the total traffic of 50% of the links, one order of magnitude lower than the other algorithms without compensating that by overloading a few links, as we would expect for the links that connect the clusters. The other algorithms present a similar, but shifted, distribution. Moreover, considering that each algorithm is using a different number of links to send the traffic to, it is worth mentioning that *network_cluster* transfers the lowest total amount of bytes throughout the whole measured period (1 month), 1 Terabyte, while *data_cluster* is the most expensive, transferring a total of 1.7 Terabytes. We also find that *data+network* lies between *network_cluster* and *data_cluster*, with 1.4 Terabytes, while *manual* transfers 1.3 Terabytes.

Overall, we observed that network locality can play a significant role in distributing the load on the network. Even in the case of existing communities without proxies, like the studied case, a locality-aware service can reduce its impact on the network performance.

6.3. Proxy Perspective

From the perspective of the proxies, it is important for both the service performance and the user's experience to distribute the load according to the capacity and performance of each proxy.

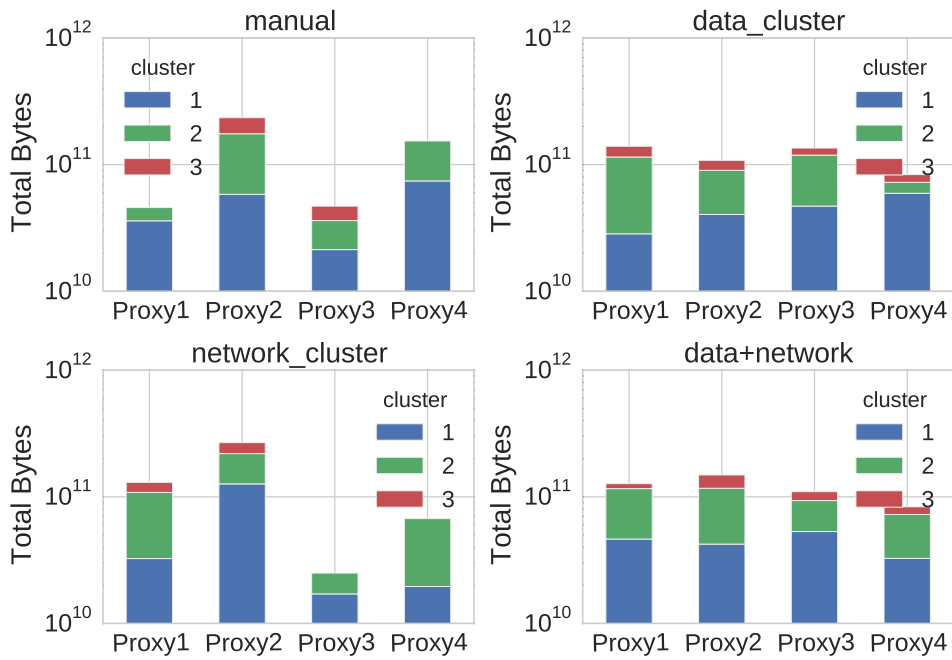

Figure 6.4: Comparison of strategies: traffic per proxy and clustering.

In our simulations we started by assuming that all proxies have infinite capacity and the same processing performance (i.e., unlimited throughput). We evaluated the different algorithms by the total amount of bytes sent to each proxy per strategy, with information of the corresponding clusters, as seen in Figure 6.4. We initially observed that the *heavy* users occupy an important percentage of the traffic, even though they are 1% of the total users. Nonetheless, *light* users, generate the majority of traffic despite the fact that each of them uses the service comparatively much less, since they significantly outnumber the rest of the users. Therefore, as a result of manual selection, the proxy load is very unbalanced, but the *data_cluster* and *data+network*

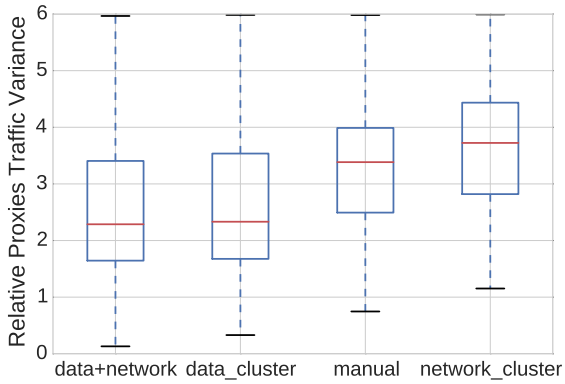

Figure 6.5: Proxies relative traffic variance per second.

algorithms succeed in balancing the traffic. The *network_cluster* approach can result in an imbalance of the load among proxies, due to sub-networks with an uneven number of clients per proxy, and proxies not well placed with respect to the clients. It is worth noting that, from the proxy perspective, the *data+network* algorithm achieves its goal very successfully since it is mainly based on the *data_cluster* algorithm; however, it also achieves a better performance than *data_cluster* from the network perspective. Moreover, as shown in Figure 6.5, the sum of distances of traffic values for each proxy to the mean at each instant is clearly smaller for *data_cluster* or *data+network*. This small variability implies that these algorithms work well over both short and long term periods. We therefore deduced that an algorithm that combines both, user clustering and network graph community detection, can be used for tuning the trade-off impact between uneven proxy load and excessive network traffic, due to long network paths. Therefore, such an algorithm would be beneficial for decentralized proxy selection mechanisms.

If we take into consideration the limited capacity and throughput in proxies, then balancing the traffic across them according to the capacity of each proxy becomes a key issue. For example, in the case of a large number of users, the clustering information could be used to perform admission and therefore congestion control on the proxy.

In the current scenario proxies have a rough admission control based denial of service due to saturation, and they do not perform congestion control according to load or performance. Proxies take on new requests based on a maximum number of concurrent clients, even when the proxy service is already under-performing for ongoing responses. This results in poor performance during peaks of large requests that cause congestion or a service timeout. In our decentralized scheme, clients have a list of several proxy choices. Clients make an initial choice, proxies can reject connections, and clients can just make a new local choice, transparently retry and continue from there, with no major visible effect on the users side when using Web services that do not depend on user information like IP addresses. The combination of clients using a list of proxy choices, proxy admission control, and network routing choices results in a decentralized and cooperative regulation scheme that requires little coordination.

Admission control is important in large user populations, e.g., wide-area networks with many proxies, since proxies have a limited Internet access capacity in scenarios similar to the studied one. Any bottleneck or imbalance in a massive system can easily lead to congestion, either in the access network, any proxy or the Internet access, resulting in a dramatic reduction of service throughput for many users of that proxy.

In addition to the local choices at each client and proxy, there is potential for global optimization in balancing global choices, across all proxies, by combining the user traffic behavior, user proxy choices, and proxy capacities. Thus, globally imbalanced scenarios can be avoided, where a proxy is saturated or providing low throughput, while at the same time another proxy remains underutilized.

6.4. User Perspective

The evaluation of the impact on service performance from the user perspective is the most complex to measure, as users have different metrics to assess their service according to their diverse usage habits. Here we present a first simple cost model to estimate how users perceive the impact of the presented algorithms. It is assumed that users try to make choices that minimize transfer times in the local network, combined with the processing time on the proxy server.

As far as the network is concerned, we define as c_l the cost of the link l, in terms of time, to transfer one byte, assuming that the links have infinite capacity. For each of the participating users we calculated the total cost of the network transfer as $\sum_{l=0}^{\mathrm{n}} c_\mathrm{l} * b_\mathrm{u}, l \epsilon L_\mathrm{u}$, where L_u is the set of links and b_u the total number of bytes attributed to user u.

The users' perception of the proxy performance is modeled similarly to the network performance. We define c_p as the cost of proxy p to process one byte, from the time it receives the request from the user, until it sends the last byte. We calculate the cost c_p of each proxy p separately for every strategy as $t / \sum b_\mathrm{u}, u \epsilon U_\mathrm{p}$, where t is the total measurement time, b_u the total number of bytes sent by user u and U_p the set of users of proxy p. Based on that, the proxy perceived cost for each user is: $c_\mathrm{p} * b_\mathrm{u}$.

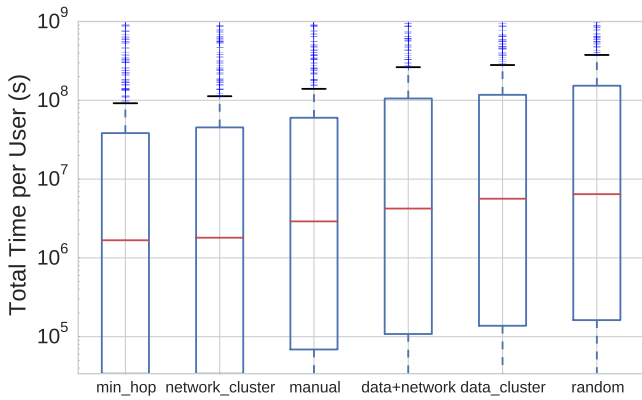

Figure 6.6: Cost per user ECDF.

Considering that the costs are linear and independent, we can assume that the overall cost perceived by a user u is: $C_\mathrm{u} = \sum_{l=0}^{\mathrm{n}} c_\mathrm{l} * b_\mathrm{u} + c_\mathrm{p} * t_\mathrm{u}, l \epsilon L_\mathrm{u}$. Hence, the objective of user u would be to minimize C_u. Figure 6.6 presents the distribution of the users' costs for each of the presented strategies.

While the distributions have a very similar behavior, we can observe that for 80% of the users, the *network_cluster* community detection strategy performs slightly better than the current *manual* situation, and the rest of the strategies follow. The *network_cluster* strategy achieves equivalent results to the *min_hop* strategy, only differing when proxies are not in the center of its zone. The

random strategy achieves equivalent results to *cluster*, as the latter only cares about contents and not about infrastructural aspects.

The network efficiency of community-based proxy selection, and therefore the impact of network locality, appears to be an important factor. Studying the individual costs we observed that the network transfer time cost is by average significantly higher than the proxy processing cost, a fact that explains why the community solution performs better overall, even though it is an inefficient option for load distribution among the proxies. The clustering according to user behavior appears to have an influence on the user's perceived performance cost, since it presents a differentiated behavior from the current manual proxy selection. However, the simplicity of the model does not allow us to draw more conclusions.

In contrast to the current manual situation, these models enable the design of a service selection algorithm that takes into account the characteristics of the users and the local network, confronting thus the inefficiencies caused in the service and the user experience by the manual static proxy selection.

6.5. Conclusions

The analysis of service logs shows that patterns of usage and grouping user/proxies by network topology can provide important information to the proxy selection process. The current manual, and not well-informed choice of proxies by the clients, works rather well for its users, but results in inefficiencies that affect the service cost and shows periods of degraded performance. Considering that situation, this chapter explored alternatives for cost reduction and service improvement when going from a simple but rigid mapping between users and proxies, towards coordinated informed choices based on several metrics. Design trade-offs lie in considering infrastructural aspects (e.g., reduce network cost, avoid network and proxy congestion) and service aspects (e.g., good response time or QoE).

The combination of server alternatives in clients, finer grain proxy admission control, and the underlying network routing decisions result in a decentralized cooperative regulation scheme that can assist in provide a crowdsourced proxy service, with good performance and requiring little coordination. Moreover,

that scheme allows the service scaling up to larger sizes, completing the answer to the first part of our second research question (RQ2).

Chapter 7

Sharing Only the Spare Internet Capacity

Chapters 5 and 6 describe our approach on how the user-proxy selection regulation mechanism can be improved, without introducing significant overhead. In this chapter we focus on the second part of the Internet sharing research problem (RQ2), as described in Section 2.1. More specifically we look at how citizens can share their spare Internet capacity without any noticeable degradation of their quality of access resulting from the secondary traffic. To achieve that, we propose utilizing middleboxes between the users and the proxies to separate the primary traffic (i.e., that of the Internet access donors) from the one of the beneficiaries (i.e., the secondary traffic).

Some previous works [ST01] [Lao+09] have shown that water-filling - benefiting from already-paid-for off-peak bandwidth resulting from diurnal traffic patterns and percentile pricing - allows delay-tolerant asynchronous bulk data to be transferred effectively at no transmission cost to the ISP. In a scenario with multiple Internet gateways available to users, while one could stop serving secondaries to avoid extra traffic charges, clients could switch to another available proxy, as seen in Chapter 5.

Each of the C beneficiary nodes selects one of the P Internet gateways, where they send their traffic. The gateways receive the network traffic from these secondary nodes and try to provide them with adequate service. Although this traffic uses the spare capacity of the Internet access, it may compete with the primary source traffic, hinder its performance, and also increase

its cost, in case of data volume or 95-percentile pricing schemes. This can be a strong demotivating factor for the donors of Internet access resources, therefore making this sharing process innocuous for them is a critical to make the Web sharing service sustainable over time. However, keeping under control this aspect of the traffic represents a major challenge for the administration of CNs.

In order to help address this challenge, we analyzed several of the mechanisms for sharing the spare Internet capacity among third parties in guifi.net, the ways to provide it, and the performance implications of connectivity sharing at no additional economic cost. Based on the obtained results, we present a set of lessons learned that can help make this sharing process suitable and sustainable.

Section 7.1 describes the experimental framework, and it shows the evaluation results in Section 7.2. Section 7.3 presents the lessons learned and conclusions.

7.1. Experimental Framework

The system model and the scenario for experimental evaluation consist of a gateway middlebox that separates the primary traffic from the secondary coming from a number of nodes of the local access network, the wired or wireless CN. All primary and secondary traffic is destined to servers on the Internet. For our experiments it was assumed the traffic is Web-like, where primary and secondary traffic comes from clients that make Web requests that result in downloading Web objects. We also assumed that all clients interact with a single server that provides content to both primary and secondary clients. Figure 7.1 shows the nodes participating in the testbed: (1) primary and secondary clients, (2) the gateway that routes traffic from both types of clients, and that interact with servers on the Internet (3). The gateway node manages both primary and secondary traffic, and applies different techniques to each traffic, considering the limited capacity of the available Internet access uplink, while trying to assess and minimize the impact of secondary traffic on the primary one. All the experiments were performed on the testbed described above which was created in a laboratory type of environment.

Internet Access Mode with Primary and Secondary Ssers Internet access characteristics are modelled at the gateway using the traffic control

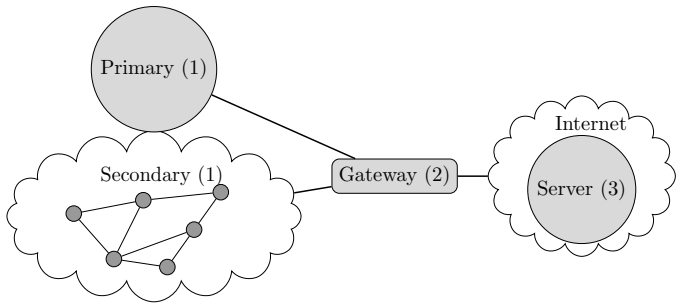

Figure 7.1: Physical architecture of the testbed.

and queueing discipline tools available on Linux. The Client-Gateway connection is 100Mbps. The Gateway-Server link uses values obtained from the Measurement-Lab testbed of Telefonica [Dov+10], the largest ISP in Spain: 1.72Mbps download throughput, analyzed in [Bra+15]. The bottleneck is at the gateway. In order to validate our experimental setup we used the Network Diagnostic Test, which is the same tool used to characterize real ISPs. In addition to the modeled values, we have validated that the modeled access behaves as we expected.

Traffic Modeling In the scenario of CNs, and specifically in guifi.net [Veg+15], gateways act as Web proxies and therefore the traffic will be HTTP. We used the *wrk2* [Ten15] tool to generate simulated customer traffic. This allows for realistic HTTP benchmarking removing the effects of "coordinated omission" [Ten13] from the measurements.

Metrics The goal of this study was to compare the different mechanisms to share spare bandwidth with the primary only (*prim_only*), and with the primary and secondary traffic without any specific mechanism (*best_effort*). These two experiments are the best case for *prim_only*, and the case that we want to improve in the *best_effort* mechanism. To evaluate the behavior of the mechanisms we utilized two metrics. First, the co-inflicted delay on the service time of HTTP requests for each mechanism - normalized to the mean latency throughout the best-effort primary and secondary traffic measured values - and second, the network throughput.

7.1.1. Traffic Sharing Between Primary and Secondary

The mechanisms of "traffic engineering" have to act only at the gateway, to not require end-to-end changes, to be transparent to clients and servers, and to be innocuous (i.e., to have no impact or cost) when the gateway is not congested.

We experimented with three types of mechanisms based on traffic shaping, Active Queue Management (AQM) and tunneling. In the mechanism based on traffic shaping, the gatewway monitors traffic and discards non-compliant packets according to the spare capacity. In the case of AQM, the gateway does not use a FIFO strategy for packets, but tries to prioritize packages by type or flow. Finally, in the last case we used tunelling to replace the congestion control of the end-to-end transport protocol to that of the tunnel. All these mechanisms are implemented on the gateway and applied on HTTP traffic send to the gateway by the clients.

Figure 7.2 shows the location of these three types of mechanisms on the network stack, indicating the types of test applied to the primary traffic (column a) and secondary one (column b). Next, we explain the mechanisms considered in more detail.

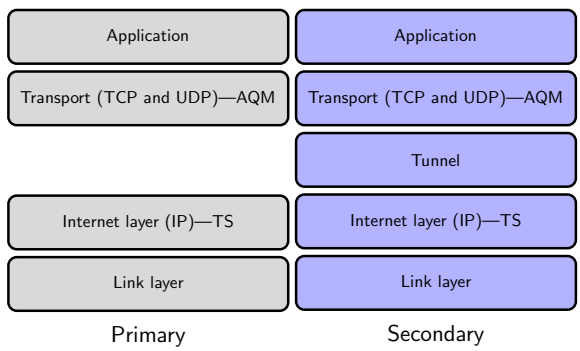

Figure 7.2: Types of tests applied to primary and secondary traffic.

Traffic Shaping Based In this case we used a *borrowing* strategy, according to which the primary and the secondary traffic have a guaranteed minimum throughput. Nevertheless, the borrowed unused throughput can be utilized with priority for the primary traffic.

Active Queue Management Based Here we used Stochastic Fairness Queueing (*sfq*) [McK90] and *CoDel* mechanisms (*codel*) [NJ12]. The first one is used by several ISPs [Gon+14], since it tries to order the packets in a more fair manner (a packet from each TCP flow). The second mechanism has been successfully used to significantly mitigate the bufferbloat phenomenon [NJ12].

Tunneling Based In this case we used three strategies: *TCP Cubic* (*tcpcubic*) [HRX08], *TCP Vegas* (*tcpvegas*) [LPW00] and *TCP LP* (*tcplp*) [KK06]. In the first case we used the tunnel's TCP congestion control algorithm (*tcpcubic*) to manage the secondary traffic. In the second case, the secondary traffic was managed through a *tcpvegas* tunnel, and the congestion avoidance algorithm emphasized packet delay (RTT) rather than packet loss. In the *tcplp* case, the secondary traffic was managed through a TCP type low-priority tunnel, with the idea of controlling congestion [KL11]. This approach gives less priority to the secondary traffic than the best effort approach, with its main goal to utilize only the spare network bandwidth.

7.2. Results

The experiments considered in this study are intended to evaluate the impact of secondary traffic on the primary traffic, considering the *prim_only* and *best_effort* cases as reference. Moreover, we measure the impact of the different techniques on the user experience when the gateway is not overloaded and when the gateway is saturated. Additionally, we explored the sensitivity of the studied mechanisms to the characteristics of the traffic, the overhead of tunneling, and the overhead of using WiFi links, which typical for access networks such as CNs. In order to make service time results comparable across different experiments where possible, the results were normalized to the overall *best_effort* service time mean of each experiment.

7.2.1. Gateway Not Overloaded

As a first step, we study scenario of a not overloaded gateway, where the traffic model consists of a single primary user with two concurrent connections each, and four to five secondary users with ten concurrent connections each. All

HTTP requests involve objects of 0.1MB. There is a random time between HTTP requests that ranges from 10 to 50 ms. The Internet connection is modeled to have a maximum download throughput of 1.72Mbps. The resulting total traffic (primary + secondary) does not exceed on average the maximum throughput of the connection. Although both traffics compete, there is sufficient throughput for both of them.

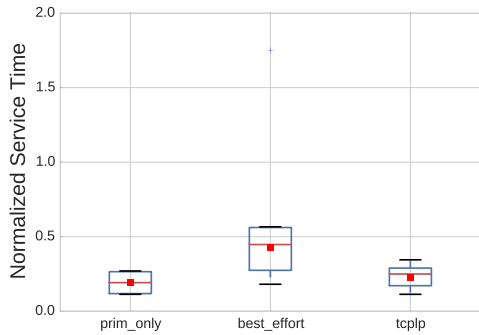

Figure 7.3: Normalized service time of primary traffic with underutilized Internet connection.

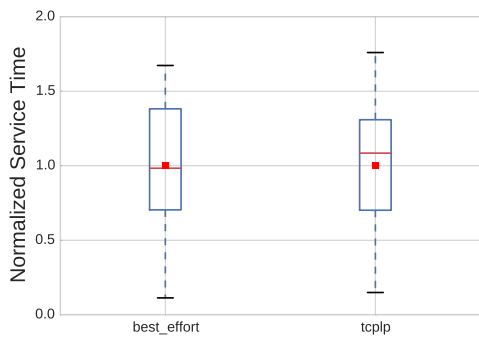

Figure 7.4: Normalized service time of secondary traffic with underutilized Internet connection.

From the results shown in Figures 7.3 and 7.4 we observe that the difference of service time for the primary traffic (between *prim_only* and *best_effort*) is noticeable but not significant considering that the absolute latency experienced by the user when downloading 0.1MB is not very high. Thus, in this scenario,

the *best_effort* strategy is already usable from the perspective of the primary user, without requiring major improvements. Moreover, we see that the service time achieved by the secondary traffic has a large impact on the service time of the primary one. Looking at the service time of the primary traffic, *tcplp* offers values very close to *prim_only*. However, if we consider the service time of the secondary traffic, the *tcplp* mechanism offers comparable values to *best_effort*. Therefore, it manages to improve the primary traffic without penalizing the secondary one.

Applying the *tcplp* mechanism has no significant effect on the primary traffic when the gateway is not overloaded. Therefore, from now on we will focus only on cases in which the gateway is overloaded.

7.2.2. Gateway Is Overloaded

In order to simulate and reproduce an overloaded Internet connection, we used the following HTTP traffic generation model: the primary traffic (represents one user) was generated at the rate of 5 requests per second, while the secondary one (represents 5 users) was generated at the rate of 25 requests per second. All the HTTP requested objects have a fixed size of 12.5KB, except if explicitly stated otherwise. Moreover, the primary traffic was generated with a random *user think time* in the range of 10-50ms between every request. Additionally, we limited the throughput of the Internet connection to 1.72Mbps for downloading and 0.54Mbps for uploading to provide a more realistic experimental environment. Throughout all the experiments the total traffic was generated with a rate greater than 1.72Mbps to achieve saturation of the Internet connection. As a result, the primary and the secondary traffic had to compete to access the Internet. *codel* and *sfq* were applied to all traffic without differentiating primary and secondary. The results are shown in Figures 7.5 to 7.8.

Figures 7.5 and 7.6 show that the difference of service time between *prim_only* and *best_effort* (for the primary traffic) is substantial. While the secondary traffic has both a very large impact on the primary one, it achieves a latency much worse than *prim_only*, resulting to poorer utilization for all the users. Using *tcplp*, *tcpvegas* and *borrowing*, the primary user experience a very good service time, while the secondary traffic delay is much less that the proportion corresponding to the primary's gain (20%). I*codel* achieves only a very slight improvement to both the primary and secondary service time

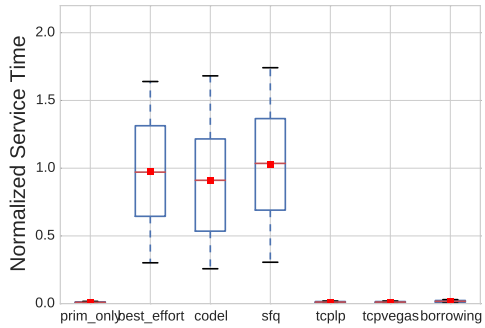

Figure 7.5: Comparison of the effect of strategies on service time of the primary traffic under a saturated Internet connection.

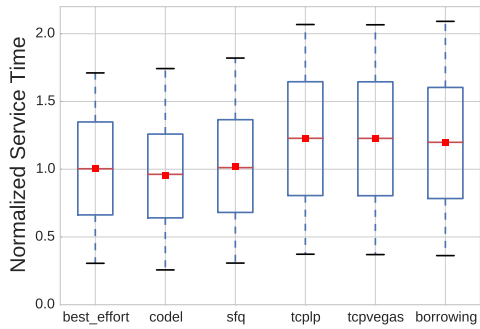

Figure 7.6: Comparison of the effect of the strategies on service time of the secondary traffic under saturated Internet connection.

compared to *best_effort*, while *sfq* does show significant improvement. As far as throughput is concerned, the success of the mechanisms to prioritize the primary traffic without significantly deteriorating the experience of the secondary users, follows exactly the same patterns as latency, and can be seen in Figure 7.7.

The effect on TCP is illustrated by Figure 7.8. We observe that the number of retransmissions is very low compared to the number of HTTP requests performed in each experiment with the exception of *codel* that shows its sensitivity to maximum number of flows that can be served.

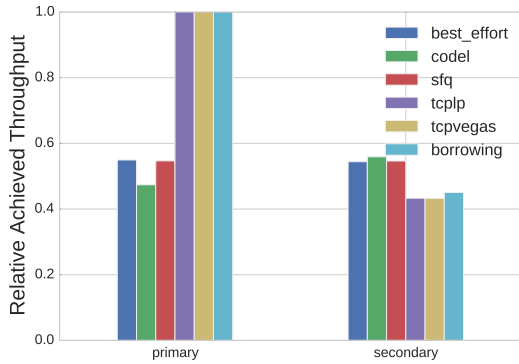

Figure 7.7: Throughput comparison under saturated Internet connection.

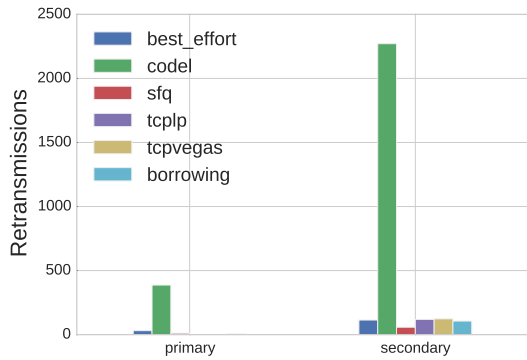

Figure 7.8: Retransmission comparison under saturated Internet connection.

As it was described, the secondary traffic has a very significant impact on the service time and throughput of the primary traffic when competing for access on an overloaded gateway. *tcplp*, *tcpvegas* and *borrowing* seem to able to priorize primary. Additionally, they penalize the secondary traffic, with latency and throughput values slightly worse than *best_effort*.

7.2.3. Sensitivity Analysis

In order to investigate the sensitivity of the proposed mechanisms to characteristics of the traffic and the environment, we analyzed their relation of the service time, to the distribution of concurrent requests and to the object size.

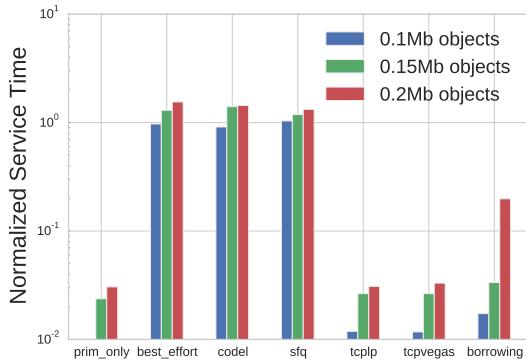

Figure 7.9: Primary traffic service time sensitivity on Object Size of the strategies.

7.2.3.1. Object Size

For the experiment with the object size the results are normalized based on the mean throughout the primary and secondary service time results for 0.1 Mb. Figures Figure 7.9 show that there is a clear relationship between service time and object size. As expected, increasing the object size results in an increased service time, the amount of data per second requested is increased as well. Moreover, as far as the primary traffic is concerned, *codel* and *sfq* present a behavior close to the *best_effort* mechanism, but slightly varying based on the object size. *tcplp* and *tcpvegas* are the solutions with behavior closest to *prim_only*. The *borrowing* strategy seems to have a performance similar to *prim_only*, but appears to be more sensitive to the the size of requested objects. When the object size is increased (by increasing the stress on the server), the service time for the primary deviates significantly from *prim_only*. Additionaly, we observe that there is a very similar pattern of increasing service time while increasing object size for all secondaries.

7.2.3.2. Proportion of Requests Primary-Secondary

In this experiment we varied the proportion of requests between primary and secondary traffic, while keeping the total throughput and the total number of requests. Results are normalized based on the mean throughout the primary and secondary service time results for the pair of 5/25 reqs/s. As expected,

there is no visible difference between *codel* and *sfq* when they are applied without differentiating primary and secondary traffic, and the aggregation of primary and secondary connections is kept at 30 concurrent connections. Therefore, Figure 7.10 only presents strategies affected by those changes, omitting *codel* and *sfq*.

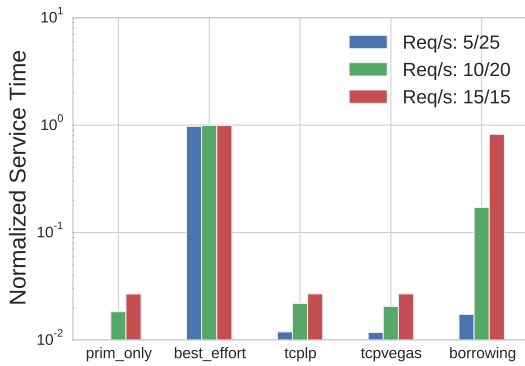

Figure 7.10: Primary traffic service time sensitivity on requests rate.

Considering the service time of the primary traffic, *tcplp* and *tcpvegas* still behave very similar to *prim_only*. However, *borrowing* presents again a differentiated behavior. It is very sensitive to the proportion of connections between the primary and secondary nodes. In any case where the primary or secondary traffic exceeds its configured upper bound data rate, the service time will increase accordingly. Regarding the secondary traffic, all service times are more or less close to the *best_effort* service time.

7.2.3.3. "Edge Cases" for Object Size/Requests Proportion

The objective of this experiment was to show how the different strategies function in extreme cases while keeping the overall throughput with a very small object size, which fits in a single TCP frame, compared to large object sizes. In this case, the results were normalized based on the mean throughout the primary and secondary service time for the pair of 0.01Mb objects - 150 reqs/s. From the Figures 7.11 and 7.12 we can observe that these scenarios are consistent with the previous ones. *sfq* and *codel* provide only small improvements, mostly when there are many requests. As far as the primary traffic is concerned, *tcplp* and *tcpvegas* behave almost like *prim_only*, with an

advantage for *tcplp* with small objects. The *borrowing* strategy seems to make the situation worse when there are a lot of requests, while it improves (for both primary and secondary traffic) with larger objects.

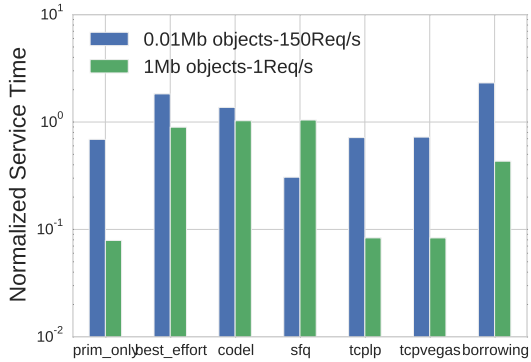

Figure 7.11: Primary traffic service time comparison on edge scenarios.

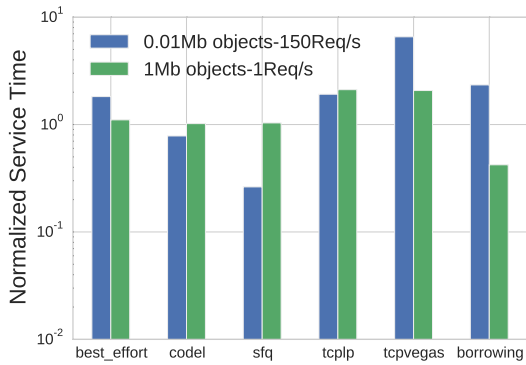

Figure 7.12: Secondary traffic service time comparison for edge scenarios.

From the results obtained related to the sensitivity of the studied mechanism we conclude the following lessons learned. The *sfq* and *codel* mechanisms provide a behaviour very similar to the *best_effort* when applied alone. The *borrowing* mechanism is sensitive to object size and to the distribution of the number of concurrent connections. Therefore these three mechanisms cannot be considered as good options for real-environment application. On the other hand, *tcplp* and *tcpvegas* behave very close to *prim_only*, as far as the primary traffic is concerned, even in edge scenarios, without creating great

extra overhead to the secondary. Between these two options, *tcplp* appeared to penalize less the secondary traffic, hence, it ranks so far as the best candidate.

7.2.3.4. Other Factors

We compared the differences in behavior among different types of tunnels. Comparing the experiment results we observed that the IP-over-IP tunnel has the same behavior as *best_effort* on the primary traffic, with or without delay, with relative differences less than 0.04%. Therefore, we can conclude that tunnel based techniques do not add any penalty. Additionally, we compared the behavior of *tcplp* and *tcpvegas* used in the secondary tunnels against *tcpcubic* that was used for the the primary traffic and as transport under the tunnels. We observed that the aggressiveness of *tcpcubic* is the reason that *tcpvegas* behaves similarly to *tcplp* in our experiments. Substituting *tcpcubic* in the primary tunnels for other TCP algorithm we expect to obtain similar results for *tcplp* tunnels, but *tcpvegas* tunnels would deteriorate the primary service time while improving the secondary service time. This behaviour of *tcpvegas* sets *tcplp* as our best mechanism for sharing the spare Internet capacity so far.

Moreover, we evaluated the overhead of wireless (WiFi-based) links for secondary users, as this is quite common in access networks, such as CNs. We used an an ad-hoc (IBSS) network in channel 4 of the 2.4 GHz band. The experiments show equivalent results to a wired Ethernet connection, just with a slight improvement for *codel* and *sfq* both for the primary and secondary traffic.

Finally, we want to mention that there are other important types of traffic in a network that may be affected by the secondary traffic and by the overload; for instance, some important protocols such as DNS, ICMP or packets like SYN. The results in Figure 7.13 show that even in the case of an overloaded gateway, *codel* and *sfq* contribute to improve the behavior of *tcplp*. Flows with very few packets, such as ICMP ping in the figure, no longer suffer from "starvation" by virtue of not being trapped in a FIFO queue. Additionally, we observe that utilizing existing AQM techniques like *codel* and *sfq*, that are already available and implemented, we can achieve improved primary responsiveness even compared to *prim_only*. While this can depend on the traffic type, it shows potential for a more mainstream adoption of these techniques as part of the default TCP/IP stack. The same effect is achieved for the secondary

traffic, as well as in the primary and secondary traffic for a non-overloaded gateway.

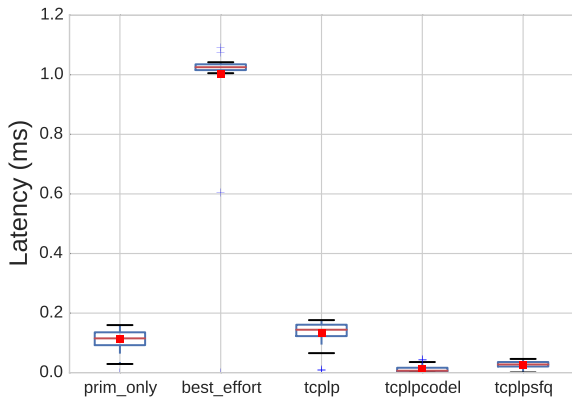

Figure 7.13: Primary client ping latency combining *AQM* and *tcplp*.

7.3. Conclusions

In this chapter, we studied techniques that can guarantee Web experience of users that benevolently share their spare Internet capacity through a local CN. We evaluated the performance and drawbacks for the primary and secondary users of several mechanisms for sharing spare Internet. In summary, *tcplp* appears to be the most promising option, regardless of whether the gateway is overloaded or not. The primary traffic is apparently not affected by the secondary one, behaving like if the primary client was performing request without competing for the Internet capacity. The secondary traffic achieves to utilize the spare capacity, behaving like the non-differentiated case, best effort, with a limited penalty around 20% on the latency. Combined with complementary queueing techniques (e.g., *tcplp* + *codel* or *tcplp* + *sfq*) instead of just a FIFO queue, it allows to "treat well" other small, but important for the user experience, traffic types, such as DNS or ICMP. The proposed mechanism can be useful for an Internet access sharing service, since it combines multiple shared Internet connections at no additional penalty in performance and cost over a local or regional CN. This result, combined with the previous ones contributes to complete the answer to our second research question (RQ2).

Chapter 8

Conclusion

Communities of citizens develop network infrastructures cooperatively based on heterogeneous Wireless Mesh Networks. They can achieve Internet or Web access using those networks through a pool of Web proxy gateways shared across many participants of the local CN. This approach can provide affordable Internet access but requires an effective mechanism to match client demand with the available proxy resources, ensuring good quality of experience and avoiding degraded service.

Analyzing a deployment of Web proxies in the guifi.net CN, the first contribution, we observed that the system is simple and resilient since each proxy is independent and clients just switch to their next choice in case of failure of the initially selected proxy. While the system shows a satisfactory performance in a small scale, the manual proxy selection of the clients and the diversity of traffic patterns and capacities of proxies would not allow the Web access service to perform as efficiently in larger scale scenarios were users try to access proxies in different parts of the network. In our effort to address this problem we decomposed it in two major research questions: 1) the clients-proxies relation concerning the characteristics, limitations and usability of a shared Internet Web proxy service in CNs and 2) the improvement of the user experience and fairness of Internet sharing Web proxy services in CNs without introducing significant overhead to the network and other services.

Concerning the relation between the client users and the proxies, the second contribution, we presented two reliable and inexpensive latency-based metrics capable of predicting and triangulating performance indicators, and a client-

side proxy selection mechanism that combines these metrics to make good choices in terms of QoE or performance, taking into account the contribution of the local network, proxy gateways and their Internet connection. This mechanism avoids proxies with heavy load and slow internal network paths. The overhead is linear to the number of clients and proxies. Additionally, we analyzed how the currently manual and not well-informed choice of proxies by clients works rather well for its users, but results in inefficiencies that affect the service cost and shows periods of degraded performance. Considering the analyzed environment, we explored alternatives for cost reduction and service improvement when going from a simple but rigid mapping between users and proxies, towards coordinated informed choices based on several metrics.

Concerning the relation between primary (who share their Internet connection) and secondary (the beneficiaries) users, the third contribution, we studied several mechanisms for sharing spare Internet capacity. The results show the performance and drawbacks for primary and secondary users. We found out that a middlebox between the gateway and all users can control the tradeoffs affecting the Internet access experience of the primary and secondary users. Studying various strategies, we proposed a combination of Transport Layer tunneling and queuing techniques that can be used for this middlebox to guarantee the user experience of the primary when the shared Internet connection is saturated, without introducing overhead for the rest of the time.

Overall, we are confident that these three main contributions provide a satisfactory answer to our two main research questions. Furthermore, we believe that we have paved the way to use already existing CN infrastructures to provide basic Internet access for all their members.

8.1. Application to Other Environments

Our approach in expanding Web access to more people is based on two main categories of participants: providers of shared Web access resources and consumers of these resources. The driving ideas of the proxy selection and the bandwidth sharing mechanisms presented in this thesis can be applied on any scenario where Web access resources are being shared and consumed. Nevertheless, the exact proposed mechanisms have a wide but more limited applicability. Moreover, in the concrete environment of CNs, as stated in Chapter 1, our preliminary qualitative study implies that the presented mech-

anisms can be successfully deployed in the listed CNs, as well as others with similar characteristics.

The proxy selection mechanism described in Chapter 5 lies on a generic modelling of the client-proxy communication, that can be applied in other gateway scenarios, where the HTTP requests from the client to the Web server are being forwarded by an intermediary node in the Application Layer. This is not the case, though, for scenarios where the client can establish direct connections with the Web server, such as tunnels to uplinks, or Internet Layer gateways. In these cases, while the metrics themselves can provide useful insights, their interpretation and application to the selection process should be tuned accordingly. As far as the environment of application is concerned, the proxy selection mechanism has no particular requirements since it is designed to function in heterogeneous network environments, nevertheless this may imply significant performance differences compared to tailor-made solutions for homogeneous (e.g. specific to a single routing protocol or network technology) network scenarios.

Sharing Web access resources, under our approach as described in Chapter 7, is based on the placement of transparent middleboxes, between the primary/secondary traffic and the Internet gateway (router or proxy), to prioritize the primary traffic. The presented results concern mainly HTTP traffic tested for both wired and wireless environment, hence their applicability ranges widely from WiFi Sharing schemes (i.e. FON) and smart-home Internet-Of-Things traffic scenarios to Internet access cost reduction strategies. Moreover, our findings can be applied in any kind of gateway between networks, where various HTTP traffic from sources with different priorities are competing for access on the same Internet connection, and can be customized for adjusting the proportion of priority of one traffic to the other.

8.2. Future Work

In this thesis we have presented mechanisms which facilitate and improve Internet access sharing with HTTP Proxies in CNs without introducing significant overhead to the network.

Our next step focuses on integrating the proxy selection mechanism with the mechanism to share spare bandwidth resulting in a complete Internet access sharing system based on HTTP proxies.

After evaluating the prototype in a lab environment we plan to perform a pilot experiment in the guifi.net CN involving actual users of the proxy service, which will provide us with feedback about the improvements in usability and fairness of the system leading to potential iterative optimizations. We then intend to test the improved version of the prototype in other networks to measure its performance in different environments.

These experiments will shed light on the global stability and convergence of the system, which currently remains an open issue. Further work needs to be done in that direction, possibly adopting client selection mechanisms that are non-deterministic in temporal or spacial domains.

Nevertheless, while currently clients are the only decision making component of our solution, we believe that smart admission control on the server side should be investigated. Such mechanism can lead to important cost reductions in Internet traffic in cases of traffic-based billing such as in 95-percentile pricing schemes.

Additionally, provided our analysis of how clustering of data from various network levels can be used to improve the service, another challenge lies in investigating client-side mechanisms that would avoid intensive data transfer through the network.

Finally, incentives and compensation schemes relevant to the service should be investigated, that would not only facilitate existing contributors of the service but also encourage new users to share their Internet bandwidth.

Bibliography

[Abu+15] Abujoda, A., Dietrich, D., Papadimitriou, P., and Sathiaseelan, A. "Software-defined wireless mesh networks for internet access sharing". In: *Computer Networks* 93, Part 2 (2015), pp. 359–372 (cit. on pp. 3, 20).

[Afa+10] Afanasyev, M., Chen, T., Voelker, G., and Snoeren, A. "Usage patterns in an urban WiFi network". In: *IEEE/ACM Transactions on Networking* 18.5 (2010), pp. 1359–1372 (cit. on p. 19).

[AWW05] Akyildiz, I. F., Wang, X., and Wang, W. "Wireless Mesh Networks: A Survey". In: *Computer networks* 47.4 (2005), pp. 445–487 (cit. on p. 5).

[Alp16] Alphabet X Corporation. *Google Loon Project*. [Online; accessed 10-February-2017]. 2016. URL: `https://x.company/loon/` (cit. on p. 1).

[ABC10] Ancillotti, E., Bruno, R., and Conti, M. "Load-balanced routing and gateway selection in wireless mesh networks: Design, implementation and experimentation". In: *World of Wireless Mobile and Multimedia Networks (WoWMoM)*. 2010, pp. 1–7 (cit. on pp. 22, 23).

[AAJ09] Ashraf, U., Abdellatif, S., and Juanole, G. "Gateway selection in backbone wireless mesh networks". In: *Wireless Communications and Networking Conference, (WCNC)*. IEEE, 2009 (cit. on pp. 22, 23).

[Bai+15] Baig, R., Roca, R., Freitag, F., and Navarro, L. "guifi.net, a crowdsourced network infrastructure held in common". In: *Computer Networks* 90 (2015), pp. 150–165 (cit. on pp. 3–5, 20, 56).

[BML15] Baldesi, L., Maccari, L., and Lo Cigno, R. “Improving P2P streaming in Wireless Community Networks”. In: *Computer Networks* 93, Part 2 (2015), pp. 389–403 (cit. on p. 4).

[Ber06] Berkhin, P. “Grouping Multidimensional Data: Recent Advances in Clustering”. In: ed. by Kogan, J., Nicholas, C., and Teboulle, M. Springer, 2006. Chap. A Survey of Clustering Data Mining Techniques, pp. 25–71 (cit. on p. 60).

[Bis+15] Biswas, S. et al. “Large-scale measurements of wireless network behavior”. In: *ACM SIGCOMM Computer Communication Review* 45.4 (2015), pp. 153–165 (cit. on p. 19).

[BCK07] Bortnikov, E., Cidon, I., and Keidar, I. “Scalable Load-Distance Balancing”. In: *Distributed Computing*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 77–91 (cit. on p. 23).

[BH11] Boushaba, M. and Hafid, A. “Best path to best gateway scheme for multichannel multi-interface wireless mesh networks”. In: *Wireless Communications and Networking Conference (WCNC)*. IEEE, 2011, pp. 689–694 (cit. on pp. 22, 23).

[Bra+15] Braem, B., Bergs, J., Blondia, C., Navarro, L., and Wittevrongel, S. “Analysis of End-User QoE in Community Networks”. In: *Computing for Development (ACM-DEV)*. London, UK, 2015, pp. 159–166 (cit. on p. 75).

[Bri+08] Brik, V. et al. “A measurement study of a commercial-grade urban wifi mesh”. In: *Internet measurement conference*. (IMC). 2008, pp. 111–124 (cit. on p. 19).

[BT 16] BT Corporation. *BT Wi-Fi*. [Online; accessed 10-February-2017]. 2016. URL: `https://www.btwifi.co.uk/` (cit. on p. 2).

[Cas17] Castellano, Pablo. *libcnml*. [Online; accessed 10-February-2017]. 2017. URL: `https://github.com/PabloCastellano/libcnml` (cit. on p. 14).

[Cat+11] Catrein, D. et al. “An Analysis of Web Caching in Current Mobile Broadband Scenarios”. In: *New Technologies, Mobility and Security*. (NTMS). 2011, pp. 1–5 (cit. on pp. 21, 29).

[Cer12] Cerdà-Alabern, L. “On the Topology Characterization of Guifi.Net”. In: *Proc. Int. Conf. Wireless and Mobile Computing, Networking and Communications (WiMob)*. Barcelona, Spain, 2012, pp. 389–396 (cit. on pp. 4, 25).

[Cer02] Cerf, V. *The Internet is for everyone*. 2002. URL: `https://tools.ietf.org/html/rfc3271` (cit. on p. 1).

[CST15] Chen, F., Sitaraman, R. K., and Torres, M. “End-user mapping: Next generation request routing for content delivery”. In: *ACM SIGCOMM Computer Communication Review* 45.4 (2015), pp. 167–181 (cit. on pp. 41, 48).

[Che+11] Chen, Y., Wang, X., Shi, C., Lua, E. K., Fu, X., Deng, B., and Li, X. “Phoenix: A Weight-Based Network Coordinate System Using Matrix Factorization”. In: *IEEE Transactions on Network and Service Management* 8.4 (2011), pp. 334–347 (cit. on p. 23).

[Che+09] Chen, Y., Xiong, Y., Shi, X., Zhu, J., Deng, B., and Li, X. “Pharos: accurate and decentralised network coordinate system”. In: *IET Communications* 3.4 (2009), pp. 539–548 (cit. on p. 23).

[Dab+04] Dabek, F., Cox, R., Kaashoek, F., and Morris, R. “Vivaldi: A decentralized network coordinate system”. In: *ACM SIGCOMM Computer Communication Review* 34.4 (2004), pp. 15–26 (cit. on pp. 23, 41, 44, 45, 56).

[Dim16] Dimogerontakis, Emmanouil. *guifiAnalyzer*. [Online; accessed 10-February-2017]. 2016. URL: `https://github.com/emmdim/guifiAnalyzer` (cit. on p. 14).

[DMN17] Dimogerontakis, E., Meseguer, R., and Navarro, L. “Internet Access for All: Assessing a Crowdsourced Web Proxy Service in a Community Network”. In: *Passive and Active Measurement Conference*. (CORE2014 Rank B). 2017 (cit. on pp. 16, 18).

[Dim+17c] Dimogerontakis, E., Meseguer, R., Navarro, L., Ochoa, S., and Veiga, L. “Community Sharing of Spare Network Capacity”. In: *IEEE International Conference on Networking, Sensing and Control*. (ERA2010 Rank C),(under review). 2017 (cit. on pp. 17, 18).

[Dim+17b] Dimogerontakis, E., Meseguer, R., Navarro, L., Ochoa, S., and Veiga, L. "Design Trade-offs of Crowdsourced Web Access in Community Networks". In: *IEEE 21st International Conference on Computer Supported Cooperative Work in Design*. (CORE2014 Rank B),(Under Review). 2017 (cit. on pp. 16, 18).

[Dim+17a] Dimogerontakis, E., Neto, J., Meseguer, R., and Navarro, L. "Client-Side Routing-Agnostic Gateway Selection for heterogeneous Wireless Mesh Networks". In: *IFIP/IEEE International Symposium on Integrated Network Management*. (CORE2014 Rank A). 2017 (cit. on pp. 16, 18).

[Dov+10] Dovrolis, C., Gummadi, K., Kuzmanovic, A., and Meinrath, S. D. "Measurement Lab: Overview and an Invitation to the Research Community". In: *ACM SIGCOMM Computer Communication Review* 40.3 (2010), pp. 53–56 (cit. on p. 75).

[Fac16] Facebook Corporation. *Facebook Free Basics Project*. [Online; accessed 10-February-2017]. 2016. URL: `https://info.internet.org` (cit. on p. 1).

[Fel+99] Feldmann, A. et al. "Performance of web proxy caching in heterogeneous bandwidth environments". In: *Proceedings of conference of the IEEE Computer and Communications Societies*. (INFOCOM). 1999, pp. 107–116 (cit. on p. 21).

[Fon16] Fon. *Fon Corporation*. [Online; accessed 10-February-2017]. 2016. URL: `https://fon.com` (cit. on pp. 2, 20).

[Fre16] Freifunk.net. *Freifunk.net*. [Online; accessed 10-February-2017]. 2016. URL: `https://freifunk.net/en/what-is-it-about/` (cit. on pp. 5, 6).

[Fuc17] Fuchs, C. "Sustainability and community networks". In: *Telematics and Informatics* 34.2 (2017), pp. 628–639 (cit. on p. 4).

[GAI16] GAIA WG. *Global Access to the Internet for All Research Group*. [Online; accessed 14-September-2016]. 2016. URL: `https://irtf.org/gaia` (cit. on p. 1).

[GRS08] Galvez, J. J., Ruiz, P. M., and Skarmeta, A. F. G. "A distributed algorithm for gateway load-balancing in Wireless Mesh Networks". In: *Wireless Days*. 2008, pp. 1–5 (cit. on p. 22).

[Gol+04] Goldenberg, D. K., Qiuy, L., Xie, H., Yang, Y. R., and Zhang, Y. "Optimizing cost and performance for multihoming". In: *ACM SIGCOMM Computer Communication Review* 34.4 (2004), pp. 79–92 (cit. on p. 7).

[Gon+14] Gong, Y., Rossi, D., Testa, C., Valenti, S., and Täht, M. D. "Fighting the bufferbloat: on the coexistence of AQM and low priority congestion control". In: *Computer Networks* 65 (2014), pp. 255–267 (cit. on p. 77).

[gui16] guifi.net. *CNML*. [Online; accessed 10-February-2017]. 2016. URL: `http://en.wiki.guifi.net/wiki/CNML/en` (cit. on p. 12).

[gui15] guifi.net. *SNPServices*. 2015. URL: `https://github.com/guifi/snpservices` (cit. on p. 13).

[HRX08] Ha, S., Rhee, I., and Xu, L. "CUBIC: A New TCP-friendly High-speed TCP Variant". In: *SIGOPS Oper. Syst. Rev.* 42.5 (July 2008), pp. 64–74 (cit. on p. 77).

[HSS08] Hagberg, A. A., Schult, D. A., and Swart, P. J. "Exploring network structure, dynamics, and function using NetworkX". In: *Proceedings of the 7th Python in Science Conference (SciPy2008)*. Pasadena, CA USA, Aug. 2008, pp. 11–15 (cit. on p. 14).

[IP11] Ihm, S. and Pai, V. S. "Towards understanding modern web traffic". In: *Internet measurement conference*. (IMC). 2011, pp. 295–312 (cit. on pp. 20, 29).

[Int09] International Telecommunication Union. *Trends in Telecommunication Reform 2008. ch. Six Degrees of Sharing*. July 2009. URL: `http://www.itu.int/pub/D-PREF-TTR.16-2015` (cit. on p. 2).

[Int15] Internet Society. *Global Internet Report 2015*. Oct. 2015. URL: `http://www.internetsociety.org/globalinternetreport/` (cit. on p. 1).

[Joh+11] Johnson, D. L., Pejovic, V., Belding, E. M., and Stam, G. van. "Traffic characterization and internet usage in rural Africa". In: *World Wide web*. 2011, pp. 493–502 (cit. on p. 20).

[J+01] Jones, E., Oliphant, T., Peterson, P., et al. *SciPy: Open source scientific tools for Python*. [Online; accessed 22-February-2017]. 2001–. URL: `http://www.scipy.org/` (cit. on p. 14).

[KS60] Kemény, J. and Snell, J. *Finite markov chains*. University series in undergraduate mathematics. Van Nostrand, 1960 (cit. on p. 57).

[KL11] Khare, R. and Lawrence, S. *A Survey of Lower-than-Best-Effort Transport Protocols*. 2011. URL: `https://tools.ietf.org/html/rfc6297` (cit. on p. 77).

[KN01] Kim, M. and Noble, B. "Mobile Network Estimation". In: *Proceedings of the 7th Annual International Conference on Mobile Computing and Networking*. MobiCom '01. Rome, Italy: ACM, 2001, pp. 298–309 (cit. on p. 49).

[Ko+13] Ko, B. J., Liu, S., Zafer, M., Wong, H. Y. S., and Lee, K. W. "Gateway selection in hybrid wireless networks through cooperative probing". In: *International Symposium on Integrated Network Management (IM)*. IEEE, 2013, pp. 352–360 (cit. on pp. 22, 23, 53).

[KW00] Krishnamurthy, B. and Wang, J. "On Network-aware Clustering of Web Clients". In: *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*. (SIGCOMM). Stockholm, Sweden, 2000, pp. 97–110 (cit. on p. 21).

[Kri+09] Krishnan, R., Madhyastha, H. V., Jain, S., Srinivasan, S., Krishnamurthy, A., Anderson, T., and Gao, J. "Moving Beyond End-to-End Path Information to Optimize CDN Performance". In: *Proceedings of Internet Measurement Conference*. (IMC). 2009, pp. 190–201 (cit. on p. 21).

[KK06] Kuzmanovic, A. and Knightly, E. W. "TCP-LP: Low-priority Service via End-point Congestion Control". In: *IEEE/ACM Transactions on Networking* 14.4 (Aug. 2006), pp. 739–752 (cit. on p. 77).

[LF09] Lancichinetti, A. and Fortunato, S. "Community detection algorithms: A comparative analysis". In: *Phys. Rev. E* 80 (Nov. 2009), p. 056117 (cit. on p. 62).

[Lao+09] Laoutaris, N., Smaragdakis, G., Rodriguez, P., and Sundaram, R. "Delay tolerant bulk data transfers on the internet". In: *ACM SIGMETRICS Performance Evaluation Review*. Vol. 37. 2009, pp. 229–238 (cit. on p. 73).

[LSP08] Ledlie, J., Seltzer, M., and Pietzuch, P. “Proxy network coordinates”. In: *Target* 22 (2008), p. 25 (cit. on pp. 41, 44–46).

[Li+14] Li, Z., Zhu, X., Gahm, J., Pan, R., Hu, H., Begen, A. C., and Oran, D. “Probe and Adapt: Rate Adaptation for HTTP Video Streaming At Scale”. In: *IEEE Journal on Selected Areas in Communications* 32.4 (2014), pp. 719–733 (cit. on p. 47).

[LM14] Lo Cigno, R. and Maccari, L. “Urban Wireless Community Networks: Challenges and Solutions for Smart City Communications”. In: *ACM Int. Workshop Wireless and Mobile Technologies for Smart Cities (WiMobCity), part of MobiHoc*. Philadelphia, PA, US, 2014, pp. 49–54 (cit. on p. 4).

[LPW00] Low, S., Peterson, L., and Wang, L. *Understanding TCP vegas: Theory and practice*. Tech. rep. 2000 (cit. on p. 77).

[Mac13] Maccari, L. “An analysis of the Ninux wireless community network”. In: *International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*. 2013, pp. 1–7 (cit. on p. 20).

[Mac+15] Maccari, L., Baldesi, L., Lo Cigno, R., Forconi, J., and Caiazza, A. “Live Video Streaming for Community Networks, Experimenting with PeerStreamer on the Ninux Community”. In: *Proc. Workshop on Do-it-yourself Networking: An Interdisciplinary Approach (DIYNetworking)*. Florence, Italy, 2015, pp. 1–6 (cit. on pp. 4, 20).

[ML15] Maccari, L. and Lo Cigno, R. “A week in the life of three large Wireless Community Networks”. In: *Ad Hoc Networks* 24, Part B (2015), pp. 175–190 (cit. on p. 4).

[Mai+09] Maier, G. et al. “On Dominant Characteristics of Residential Broadband Internet Traffic”. In: *Internet Measurement Conference*. (IMC). 2009, pp. 90–102 (cit. on p. 27).

[McK90] McKenney, P. E. “Stochastic fairness queueing”. In: *INFOCOM ’90, Ninth Annual Joint Conference of the IEEE Computer and Communication Societies. The Multiple Facets of Integration. Proceedings, IEEE*. June 1990, 733–740 vol.2 (cit. on p. 77).

[Nav+16a] Navarro, L., Baig, R., Barz, C., Bonicioli, J., Braem, B., Freitag, F., and Vilata-i-Balaguer, I. “Advances in wireless community networks with the community-lab testbed”. In: *IEEE Communications Magazine* 54 (2016), pp. 20–27 (cit. on p. 40).

[Nav+16b] Navarro, L., Freitag, F., Baig, R., and Roca, R. “Community Connectivity: Building the Internet from Scratch”. In: ed. by Community Connectivity (DC3), D. C. on. Internet Governance Forum, 2016. Chap. A commons-oriented framework for Community Networks, pp. 25–71 (cit. on p. 4).

[Nay+14] Naylor, D., Finamore, A., Leontiadis, I., Grunenberger, Y., Mellia, M., Munafò, M., Papagiannaki, K., and Steenkiste, P. “The Cost of the "S" in HTTPS”. In: *Proceedings of the 10th ACM International on Conference on Emerging Networking Experiments and Technologies*. CoNEXT ’14. Sydney, Australia: ACM, 2014, pp. 133–140 (cit. on p. 21).

[net16] netCommons. *Deliverable D1.2, Report on the Existing CNs and their Organization (v2)*. Tech. Rep. D1.2. Sept. 2016 (cit. on p. 4).

[NJ12] Nichols, K. and Jacobson, V. “Controlling Queue Delay”. In: *Communications of the ACM* 55.7 (July 2012), pp. 42–50 (cit. on p. 77).

[Ost90] Ostrom, E. *Governing the commons: the evolution of institutions for collective action*. Cambridge University Press, Nov. 1990 (cit. on p. 4).

[Ped+11] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830 (cit. on p. 14).

[Pef+07] Peffers, K., Tuunanen, T., Rothenberger, M. A., and Chatterjee, S. “A Design Science Research Methodology for Information Systems Research”. In: *Journal of Management Information Systems* 24.3 (2007), pp. 45–77 (cit. on p. 11).

[QPV01] Qiu, L., Padmanabhan, V., and Voelker, G. "On the placement of Web server replicas". In: *Proceedings of conference of the IEEE Computer and Communications Societies*. Vol. 3. (INFOCOM). Apr. 2001, pp. 1587–1596 (cit. on p. 21).

[Rey+13] Rey-Moreno, C., Roro, Z., Tucker, W. D., Siya, M. J., Bidwell, N. J., and Simo-Reigadas, J. "Experiences, challenges and lessons from rolling out a rural WiFi mesh network". In: *ACM Computing for Development (ACM-DEV)*. ACM. 2013, p. 11 (cit. on p. 2).

[Rou87] Rousseeuw, P. J. "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis". In: *Journal of Computational and Applied Mathematics* 20 (1987), pp. 53–65 (cit. on p. 60).

[SS12] Sangwongthong, T. and Siripongwutikorn, P. "Proxy caching in wireless mesh networks". In: *Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology*. (ECTI-CON). 2012, pp. 1–4 (cit. on p. 21).

[SC13] Sathiaseelan, A. and Crowcroft, J. "LCD-Net: lowest cost denominator networking". In: *ACM SIGCOMM Computer Communication Review* 43.2 (2013), pp. 52–57 (cit. on p. 20).

[Sat+12] Sathiaseelan, A., Crowcroft, J., Goulden, M., Greiffenhagen, C., Mortier, R., Fairhurst, G., and McAuley, D. "PAWS: Public access wifi service". In: *The Third Digital Economy All-hands Meeting: Digital Engagement (DE), Aberdeen, Scotland/United Kingdom*. 2012 (cit. on pp. 3, 20).

[SP10] Seabold, S. and Perktold, J. "Statsmodels: econometric and statistical modeling with Python". In: *Proceedings of the 9th Python in Science Conference*. 2010, pp. 57–61 (cit. on p. 14).

[Sen+16] Sen, R. et al. "On the Free Bridge Across the Digital Divide: Assessing the Quality of Facebook's Free Basics Service". In: *Proceedings of the 2016 ACM on Internet Measurement Conference*. (IMC). 2016, pp. 127–133 (cit. on p. 37).

[ST01] Shalunov, S. and Teitelbaum., B. *QBone Scavenger Service (QBSS) Definition. Internet2 Technical Report, Proposed Service Definition, Internet2 QoS Working Group Document*. Tech. rep. 2001 (cit. on p. 73).

[Spe04] Spearman, C. “The proof and measurement of association between two things”. In: *The American journal of psychology* 15.1 (1904), pp. 72–101 (cit. on p. 51).

[SWA03] Spring, N., Wetherall, D., and Anderson, T. “Scriptroute: A Public Internet Measurement Facility”. In: *USENIX Symposium on Internet Technologies and Systems (USITS)*. Vol. 4. USENIX Association, 2003, pp. 17–17 (cit. on p. 44).

[squ] squid-cache. *Squid Native Log Format*. [Online; accessed 22-February-2017]. URL: `http://wiki.squid-cache.org/Features/LogFormat` (cit. on p. 13).

[SKK03] Strauss, J., Katabi, D., and Kaashoek, M. F. “A measurement study of available bandwidth estimation tools”. In: *Internet Measurement Conference (IMC)*. 2003, pp. 39–44 (cit. on p. 47).

[Sun+13] Sundaresan, S., Feamster, N., Teixeira, R., and Magharei, N. “Community Contribution Award – Measuring and Mitigating Web Performance Bottlenecks in Broadband Access Networks”. In: *Internet Measurement Conference (IMC)*. ACM, 2013, pp. 213–226 (cit. on pp. 41, 48).

[Ten13] Tene, G. “How not to measure latency”. In: *Low Latency Summit*. 2013 (cit. on p. 75).

[Ten15] Tene, G. *wrk2: A constant throughput, correct latency recording HTTP benchmarking tool*. [Online; accessed 22-February-2017]. 2015. URL: `https://github.com/giltene/wrk2` (cit. on pp. 14, 75).

[Veg+15] Vega, D., Baig, R., Cerdà-Alabern, L., Medina, E., Meseguer, R., and Navarro, L. “A technological overview of the guifi.net community network”. In: *Computer Networks* 93 (2015), pp. 260–278 (cit. on pp. 2, 4, 5, 20, 34, 75).

[Veg+12] Vega, D., Cerdà-Alabern, L., Navarro, L., and Meseguer, R. “Topology patterns of a community network: Guifi.net”. In: *Proc. Int. Conf. Wireless and Mobile Computing, Networking and Communications (WiMob)*. Barcelona, Spain, 2012, pp. 612–619 (cit. on p. 4).