Understanding the feasibility of network edge caching through measurements

Raman, Aravindh

Awarding institution:
King's College London

Understanding the feasibility of network edge caching through measurements

Aravindh Raman

Understanding the feasibility of network edge caching through measurements

Aravindh Raman

Department of Informatics, Faculty of Natural and Mathematical Sciences

King's College London

This dissertation is submitted for the degree of

Doctor of Philosophy

2020

Understanding the feasibility of network edge caching through measurements

Abstract

Recent developments in both hardware industry to bring down the cost/size of the user devices and in content systems to deliver various forms of content have resulted in the burdening of the Internet. In this work, we study how to tackle this growing network burden, leveraging local content access patterns, and the presence of edge caches and computes. The analysis is done through the data gathered from three new and trending applications: (*i*) on-demand video streaming (*ii*) live video streaming and (*iii*) decentralised social web, with the analysis being carried out on BBC iPlayer, Facebook Live and Mastodon respectively.

Our proposition is – careful planning to share content at the edge would not only earn significant traffic savings but also improves resiliency due to a redundant availability of content. We notice that the traffic load of on-demand video traffic can be decreased by up to 70% by strategic content sharing at the edge and 22% of upload live-traffic can be saved just by offloading the content to the edge and sharing amongst the local viewers. We also find, instead of popularity based content replication in the decentralised social web (which happens at present based on the federating nodes), strategic placements are much more efficient. For example, random replication of content in 5 different nodes improves the content availability by 68% even when 25 most popular nodes are down.

Thus, we investigate the spread of diverse media content arising from three different popular applications and propose how to optimally serve the content making use of the storage available at network edge.

Thesis Supervisor:	Prof. Nishanth Sastry
Thesis Examiners:	Dr. Marwan Fayed
	Prof. S. Keshav

To Geeth and Sid

Acknowledgements

I am grateful to everyone who helped me along the path of writing this dissertation. First and foremost, I would like to express the deepest gratitude to my advisor Nishanth Sastry, and I am thankful for his unwavering support throughout my Ph.D. The right questions and suggestions made by him have helped me in structuring my research. Nishanth's unique way of deriving insights and his balanced style of mentoring will have an influence on me for the years to come. I am truly fortunate to work with and learn from Nishanth, for without his guidance, this dissertation would have been a distant dream!

My journey as a Ph.D student would not have been possible without amazing collaborations. I was lucky enough to team up with brilliant people from both academia and industry, to name a few – Gareth Tyson, Diego Perino, Arjuna Sathiaseelan, Mostafa Salehi, Nader Mokari, Ming-chun Lee and Andy Molisch. Their ideas and perspective have always helped me in shaping my research in a much better way.

I am incredibly fortunate to share the lab with smart computer scientists who are also excellent friends. Uncountable coffees accompanied with intellectual discussions, arguments, lunches and the humour I have shared with Sagar Joglekar, Dmytro Karamshuk, Pushkal Agarwal, Emeka Obiodu and Changtao Zhong are simply priceless. In fact, some of the chats have turned into impactful research papers/grants. Beyond our lab, I am also grateful to friends and colleagues at Center for Telecommunication Research – Mischa Dohler, Maria Lema, Fragkiskos Sardis, Oliver Holland and Luis Sequeira for involving me in networking implementations. My sincere thanks to King's College London, for supporting me with a Professor Sir Richard Trainor Scholarship.

The confidence to choose the research path was seeded at Tata Institute of Fundamental Research and nurtured at IIT Delhi. I would like to thank all mentors and friends especially Aaditeshwar Seth, Zahir Koradia, Dipanjan Chakraborty, Karthyek Murthy and Nithin Varma. Indeed, they are the biggest inspirations to land in network measurements and data research.

Moving abroad and pursuing Ph.D could not have been possible without the support of my family. I would like to thank amma and appa for unwavering belief in my decisions at every point of time and instilling the desire in me to strive for the best. I am greatly indebted to my anna and manni, who have always encouraged me through my successes and been pillars of support during trough times. I am forever grateful to my beautiful and caring wife, Geetha, who has been a constant source of support while writing this dissertation. She has always stood by me and filled my life with fun and happiness.

Finally, I would like to thank my Ph.D examiners, Marwan Fayed and Srinivasan Keshav. Their valuable suggestions greatly helped to improve the presentation of this dissertation.

Contents

List of Figures

List of Tables

Acronyms

AFS Andrew File System

AI Artificial Intelligence

API Application Programming Interface

AP Access Point

AR Augmented Reality

AS Autonomous System

BBC British Broadcasting Corporation

BT British Telecom

CAIDA Center for Applied Internet Data Analysis

CARP Cache Array Routing Protocol

CDN Content Delivery Network

CI Confidence Interval

CP Content Provider

D2D Device-to-Device

DNS Domain Name System

DoS Denial-of-Service

DSL Digital Subscriber Line

DW Decentralised web

EPC Evolved Packet Core

FB Facebook

FIFO First In First Out

GPS Global Positioning System

HetNet Heterogeneous Networks

HF Hammersmith and Fulham

HTCP Hypertext Caching Protocol

HTML Hypertext Markup Language

HTTP Hypertext Transfer Protocol

ICMP Internet Control Message Protocol

ICN Information-centric Networking

ICP Internet Cache Protocol

IP Internet Protocol

ISP Internet Service Provider

LFU Least Frequently Used

LRU Least recently used

MNO Mobile Network Operator

NFS Network File System

NFV Network Function Virtualization

OSN Online Social Networks

OTT Over-the-Top

P2P Peer-to-Peer

PoP Point of Presence

QoE Quality of Experience

QoS Quality of service

QUIC Quick UDP Internet Connections

RIPE Réseaux IP Européens Network Coordination Centre

RTMP(S) Real-Time Messaging Protocol (Secured)

RTT Round Trip Time

SDN Software-defined networking

SLA Service Level Agreement

SSID Service Set Identifiers

SVC Scalable Video Coding

TCP Transmission Control Protocol

TV Television

UDP User Datagram Protocol

UGC User Generated Content

UK United Kingdom

URI Uniform Resource Identifier

URL Uniform Resource Locator

VPN Virtual Private Network

VR Virtual Reality

WiGLE Wireless Geographic Logging Engine

Chapter 1
Introduction

In recent years, the growth of the Internet has transformed the way a content item is delivered through a service provider's infrastructure and how we, as users, interact with the content item. Internet, built as a connection of tens of thousands of routers and autonomous systems, was introduced envisaging host-host communication rather than content delivery/distribution. However, a large proportion of the present day's Internet is filled with content which mainly comes as videos and this staggering increase in data traffic has raised questions on efficiency and resilience of the Internet infrastructure. For example, with the fixed-line networks having a heavy burden of video traffic – Over 75% of North American traffic is streaming video and audio, and 31% of it comes just from one service: Netflix [13]. In the United Kingdom (UK), video streaming application from the British Broadcasting Corporation (BBC), BBC iPlayer, is one of the largest applications on the nation's networks, used by nearly one in three adults, according to the UK communications regulator Ofcom[1]. Such huge loads have led to highly confrontational battles between content providers such as Netflix and fixed-line ISPs such as Comcast[2]. Some of these disputes are being resolved with bilateral deals (*e.g.,* Netflix and Comcast[3]) and/or in-network caches (*e.g.,* BBC with major UK Internet Service Provider (ISP)[4]). However, it takes time to find such solutions, and they have to be decided on a case-by-case basis. Furthermore, there are cases of *atypical* events which see a spike in the traffic during a certain point of time [14], where the performance of operators' network plummets delivering a poor Quality of Experience (QoE). Such additional traffic requires capacity provisioning and may involve expensive commissioning

[1] https://www.ofcom.org.uk/__data/assets/pdf_file/0024/26826/cmr_uk_2016.pdf
[2] https://www.wired.com/2014/02/comcast-netflix/
[3] http://www.tubefilter.com/2016/07/06/netflix-comcast-x1-deal/
[4] https://gigaom.com/2007/12/23/419-bbc-to-cache-iplayer-downloads-with-isps-could-soothe-net-neutrality-fe/

of additional resources, such as fibre to the home, or close to the edge, eventually leading to complexity and the soaring cost of content delivery.

While on the one hand, scaling content distribution is becoming challenging, on the other hand applications that deliver a multitude of content forms is also increasing. For example, TikTok, a popular video-sharing social networking service, which got launched during the second half of 2018 garnered 1 billion downloads by the end of 2019 accounting for 1.5% for global mobile traffic and got ranked at 11 in terms of global downstream data usage [15].

One of the commonly considered solutions to alleviate this network burden of application traffic is to distribute the load from the core to the edge devices. The first push towards distributing the load in content delivery started with the introduction of edge Content Delivery Network (CDN), then to ISP CDN and now more towards the very edge *i.e.,* the user devices. This natural shift from content provider's infrastructure to service provider's infrastructure and more recently to user-owned infrastructure not just ensures low latency and high throughput content delivery but also helps in improving user privacy [16, 17]. Thus, it is really important to carefully design and enhance caching techniques to achieve seamless content delivery.

In this work, we study a spectrum of content systems that operate on three different production chains. We evaluate the traffic savings and identify mechanisms to improve the resiliency of content systems.

A1 *An organisation owns both the application and content production.*

Production companies such as Netflix, the BBC produce and deliver the content through a proprietary application managed by the organisation. Thus the application, as well as the hosted content, are operated by the same organisation. Typically, for companies that operate on original content, the revenue is generated through the subscription-based model. In Chapter 3, we consider a similar case of the BBC iPlayer, the second most popular on demand streaming service in the UK, where the content is owned and distributed by the BBC. We show the similarities of content item access within a neighbourhood and show the amount of traffic savings achieved through sharing the caches.

A2 *Application owned by an organisation but users produce the content.*

Unlike traditional content systems where an organisation manages both the content and the infrastructure, we have moved into an era where users carry a small content production unit in their pockets. With the introduction of Youtube in early 2000, we have seen a rise in User Generated Content (UGC). Services like Youtube provide an infrastructure where users can upload and access the videos in most of the globe over the Internet. The recent introduction of social live broadcasts within social media like Facebook, Twitter, *etc.* have

enabled the users to stream a live video even from a mobile phone. In chapter 4, we explore live broadcasts that were streamed by users from their mobile devices to the popular social networking platform, Facebook.

A3 *Users own both application and the content system.*

Traditional peer-to-peer sharing, where both the devices and the content belong to the users, *e.g.,* LifeSocial.KOM [18] and PeerSON [19], did not take off partly due to challenges related to both performance [20] and reliability [21]. Following this, there is a recent growth of decentralised systems that rely on a server-based federated model, typically a server being managed by amateur users (called as administrators) and not by an organisation. These "independent" servers weave together to form a decentralised web, and a user willing to use the platform can join one of these servers. In chapter 5, we measure the presence of such servers hosting a decentralised social web application, Mastodon.

On content placement strategies. The work aims to derive insights from the data collected from these diverse set of applications A1, A2, A3 and thereby, measure the network benefits of caching and sharing the content amongst the edge devices. For applications A1 and A2 caching strategies at the edge work well to achieve traffic savings. Whereas for A3, we show instead of the current content replication scheme which is based on federation policy, random content replication works better for improving content availability. Based on the results, the work provides foundations for co-operation amongst such edge devices, irrespective of who owns the device/infrastructure and discusses the case of content sharing at the edge.

1.1 Thesis Statement

Several works focus on a complementary approach as a content placement scheme that makes better use of resources at the edge, rather than just offloading mobile edge traffic to fixed-line broadband, or investing on network upgrades. Our thesis is that –

> Storage available with the edge devices can be used optimally to achieve significant traffic savings and as well to improve the resiliency of a content system.

To validate this, we perform a large scale data dissection of three prominent platforms - BBC iPlayer, Facebook Live, and Mastodon intending to show how content sharing, caching and replication at the network edge (respectively) can help in improving the content system.

1.2 Research Goals

Based on the problem statement, we identify three primary goals with each of the goal carrying a set of research questions as below.

Research Goal 1: *Measuring the performance of sharing at the network edge for on demand streaming*

With video packets representing 56% of global Internet traffic [13], our first research goal is aimed towards handling this traffic growth through designing an effective content sharing mechanism. This would allow the video streamers within a neighbourhood to cache and share a content item. The design involves measuring the extent of savings that can be achieved by caching at the very last hop of the network, on a device in users' homes, and then connecting the caches to increase content sharing. To effectively build such a design, the questions that of interest are:

RQ 1.1: Are there sufficient users within a neighbourhood to share a content item?

RQ 1.2: To what extent do users within a neighbourhood watch similar content items?

RQ 1.3: How to optimally share the content items, and to what extent can savings be achieved?

Research Goal 2: *Measuring the performance of edge caching for Social Live Broadcasts*

Gaining popularity of new category video application called social live broadcasts result in both upstream and downstream traffic. The second goal is to perform a broad characterisation of social live broadcasts and highlight relevant implications for mobile live social video delivery derived from the observations. The research questions in this context are:

RQ 2.1: How popular are social live broadcasts from a global perspective?

RQ 2.2: Can content caching help bringing down the upload bandwidth, if so, by how much?

RQ 2.3: How social live broadcasts can benefit from existing mobile edge computing paradigms?

Research Goal 3: *Measuring the performance of content replication for Decentralised Social Web*

Technologies providing better transparency, openness and democracy on the web is ever demanding, and Decentralised web (DW) is once such initiative. The third goal is to measure a DW application, Mastodon and study the content related aspects. This involves the following research questions.

RQ 3.1: What is the presence of DW and whether there is active user participation?

RQ 3.2: Is DW really decentralised, if not, is/why there a shift towards centralisation?

RQ 3.3: Is DW resilient to network failures, if not, what are the possible reasons and how to effectively improve content availability?

1.3 Methodology & Challenges

To address the questions presented in Section 1.2, we first do a review of existing works in the area of content caching at the network edge. We then identify real-world applications and collect data on how users use the platform. It is well known that multiple platforms provide the same service, but with different user interfaces and features. For example, applications like Twitter (periscope), Youtube, *etc.* offer the option to stream social live broadcast. In the scope of this thesis, we have chosen the platforms as case studies based on the *scale* at which the platform operates. For example, in the case of the BBC iPlayer, it is one of the top catch-up Television (TV) services in the UK, second only to YouTube[5]. Whereas in the case of Facebook live, it is a feature embedded within the top-ranked social media, Facebook and every user of Facebook is actively/passively a user of Facebook live. Mastodon forms the largest decentralised social media out of all the existing ones with close to 2 million users. We then do a broad characterisation of these platforms and evaluate the content caching mechanisms to reduce the backhaul traffic as well as to improve the resiliency of the content system through trace-driven simulations. The focus of this thesis is mainly motivated by the necessity to counteract the growing traffic from diverse applications that users consume.

While guaranteeing correct inference from any data poses challenges, understanding network related access logs and web scraped data presents several problems. We present here some of the key challenges faced during the data collection. While there were no user surveys involved, the data contained user traffic; thus, the analysis needed to be carried out without compromising user privacy.

[5]http://mediatel.co.uk/newsline/2014/03/28/nielsen-data-report-february-2014/

1.3.1 Data Anonymisation and Integrity

In the case of passively monitored logs obtained from an organisation, most parts of the data are anonymised. The main intention behind such an action is to protect the privacy of users accessing the platform. As explained in Chapter 3, the on demand streaming data shared by the BBC iPlayer had the Internet Protocol (IP) addresses of users masked, and the geo-location of accesses to the video streaming service were shared at postcode level. This information helped in better understanding of data without having effects on user privacy.

However, there are some cases where it becomes necessary to coalesce two or more datasets to arrive at valuable insights. For example, in Chapter 3, the user access from the BBC iPlayer is mapped to wireless access points collected from a war-driven dataset. We had to take necessary steps (*cf.* Section 3.2) to not lose the statistical properties of the data without any compromise to user privacy.

1.3.2 Data Consistency

While scraping data from the web, we had to validate the consistency in data manually during initial phases and oversee the crawler as any updates to an Application Programming Interface (API) of the host application can break the data collection. During the preprocessing stage, as well as after the data collection has finished, the data consistency needs to be checked. In Chapter 4, we show a noise has been added to the data by the platform while exhibiting geo-coordinate positions of the users.

1.3.3 Data Storage and Ethics

Given that we now have the data from various platforms that operate at a large scale, it is vital to derive useful observations from the data. Starting from data collection, to parsing the data from a raw file and storing it as a formatted data requires a good compute power and optimal processing. Moreover, storing data at a single place (due to the security considerations) makes storage as a bottleneck.

It is essential to take certain precautions while collecting and handling the data. For example, we collected the publicly available data from `wigle.net` to get the list of Wi-Fi access points within a geographical region and `facebook.com/live/`[6] to collect all the publicly available global broadcasts with viewer information, even without logging in to the corresponding platforms. As a further precautionary step, we also anonymised the username, video identifiers before the analysis and did not publish any other user information. In the case of Mastodon, we were collecting publicly available user posts from thousands

[6]Facebook has recently deprecated the link, which used to be like – `https://web.archive.org/web/20180120090913/https://www.facebook.com/live/`

of hosts from various parts of the world. We anonymised the data before analysis, stored the results within a secure silo, and removed any text content posted by the user before examining the data.

Other common challenges in data collection include rate limits set by the platform and code failures. While these are hard to overcome, some measures, such as slowing down the rate at which a crawler operates and being a part of community managing the platform, were taken. For example, in the case of Mastodon, we were a part of administrators community, which helped in thwarting a cache-fill issue[7].

1.4 Contributions

The key contributions from the thesis fall into three categories which aim to address the presented research questions (Section 1.2).

- **Performance of content sharing at the network edge for on demand streaming:** Chapter 3 lays out the feasibility of content sharing at the network edge based on real world access to an on demand streaming platform. The chapter identifies a new mechanism for content sharing at the edge which can be managed by any of service provider, content provider or the network operator.
- **Performance of content caching mechanism at the network edge for live streaming:** We show in Chapter 4, social live broadcasts from the application exhibits similar properties as on demand videos with 46% of the videos goes unwatched while live. The chapter identifies new opportunities to save upload traffic in live videos.
- **Performance of content replication scheme for decentralised social sharing:** Chapter 5 identifies the challenges to the resiliency of the decentralised web and proposes a content replication mechanism to improve the resiliency and content availability of the decentralised web.

1.5 List of Publications

During the course of my PhD, I had the following 13 publications (first author and main contributor in 7). Chapter 3 is based on [5, 10]. [10] extends a version published as an invited paper in [9]. Chapter 4 and Chapter 5 are based on [8]and [1], respectively.

[7] `https://lists.ffdn.org/wws/arc/mastodon-admin/2019-04/msg00000.html`

Chapter 2

- A. Raman, N. Sastry, N. Mokari, M. Salehi, T. Faisal, A. Secker, and J. Chandaria. "Care to Share? An Empirical Analysis of Capacity Enhancement by Sharing at the Edge", *Proceedings of the MOBICOM Workshop on Technologies for the Wireless Edge (EdgeTech'18)*. New Delhi, India, 2018.

- A. Raman, N. Sastry, A. Sathiaseelan, J. Chandaria, and A. Secker. "Wi-Stitch: Content Delivery in Converged Edge Networks", *SIGCOMM Comput. Commun. Rev.* 47.5 (Oct. 2017), pp. 73–78.

- A. Raman, N. Sastry, A. Sathiaseelan, A. Secker, and J. Chandaria. "Wi-Stitch: Content Delivery in Converged Edge Networks", *Proceedings of the SIGCOMM Workshop on Mobile Edge Communications (MECOMM'17)*. Los Angeles, CA, USA, 2017.

Chapter 3

- A. Raman, G. Tyson, and N. Sastry. "Facebook (A)Live?: Are Live Social Broadcasts Really Broadcasts?", *Proceedings of the 2018 World Wide Web Conference*. WWW '18. Lyon, France: International World Wide Web Conferences Steering Committee, 2018, pp. 1491–1500.

Chapter 4

- A. Raman, S. Joglekar, E. De Cristofaro, N. Sastry, and G. Tyson. "Challenges in the Decentralised Web: The Mastodon Case", *Proceedings of the ACM Internet Measurement Conference (IMC'19)*. Amsterdam, Netherlands, 2019.

Other Publications

I was a part of building a 5G testbed at King's College London [4, 6]. I also helped in individual preference probability modelling of a user accessing video content items [2, 22] and as well in the optimal machine learning inference across the network devices at the edge within an ISP infrastructure [3]. Before the work on edge computing, I worked on TV whitespace and spectrum bounding for optimal content delivery to the user device from the base station [11, 12].

- M.-C. Lee, A. F. Molisch, N. Sastry, and A. Raman. "Individual Preference Probability Modeling and Parameterization for Video Content in Wireless Caching Networks", *IEEE/ACM Transactions on Networking* 27.2 (2019).

- A. Cartas*, M. Kocour*, A. Raman*, I. Leontiadis, J. Luque, N. Sastry, J. Nuñez-Martinez, D. Perino, and C. Segura. "A Reality Check on Inference at Mobile Networks Edge", *Proceedings of the EuroSys workshop on Edge Systems, Analytics and Networking (EdgeSys'19)*. Dresden, Germany, 2019.

- I. Quintana-Ramirez, A. Tsiopoulos, M. A. Lema, F. Sardis, L. Sequeira, J. Arias, A. Raman, A. Azam, and M. Dohler. "The Making of 5G: Building an End-to-End 5G-Enabled System", *IEEE Communications Standards Magazine* 2.4 (2018).

- E. Obiodu, N. Sastry, and A. Raman. "Towards a taxonomy of differentiated service classes in the 5G era", *Proceedings of the IEEE 5G World Forum (5GWF'18)*. Santa Clara, CA, USA, 2018.

- A. Raman*, D. Karamshuk*, N. Sastry, J. Chandaria, and A. Secker. "Consume Local: Towards Carbon Free Content Delivery", *2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS)*. 2018, pp. 994–1003.

- O. Holland, A. Raman, N. Sastry, S. Wong, J. Mack, and L. Lam. "Assessment of a Platform for Non-Contiguous Aggregation of IEEE 802.11 Waveforms in TV White Space", *Proceedings of the IEEE 83rd Vehicular Technology Conference (VTC Spring'16)*. Nanjing, China, 2016.

- O. Holland, H. Kokkinen, S. Wong, V. Friderikos, A. Raman, M. Dohler, and M. Lema. "Changing availability of TV white space in the UK", *IET Electronics Letters* 52.15 (2016).

- M.-C. Lee, A. F. Molisch, N. Sastry, and A. Raman. "Individual Preference Probability Modeling for Video Content in Wireless Caching Networks", *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*. 2017, pp. 1–7.

*equal contributions

Chapter 2
Background

Computer systems implement caching as a small block of memory which is used to store data temporarily and can be accessed relatively faster than the main memory. The motivation behind a cache was to improve the access time during the future requests to the data. The overall system performance is enhanced by studying the access patterns to the data and instruction cache. Earlier designs, even before the Internet era, focused on improving the data requested by a hardware or software component found in the cache memory or simply put, to improve the *cache hit* [23].

2.1 Web Caching: A Brief History

As the Internet traffic had started to evolve and with the introduction of transport protocols [24], the concept of memory caching found its way through networking systems. Distributed file system protocols such as Network File System (NFS) [25] and Andrew File System (AFS) [26] relied on caching. Domain Name System (DNS) adopted caching [27] to make IP address lookups faster. Web objects such as Hypertext Markup Language (HTML) web pages, media content were replicated and hosted in smaller web servers across the globe as cache points. These replicas helped especially popular web services and while serving static web objects, to bring down the network bandwidth usage, access time/web page load time and congestion at the server [28]. As applied in computing systems, to have an effective cache, (*i*) the client requests should exhibit both temporal and spatial locality of reference and (*ii*) the cost of caching in terms of bandwidth, access time and server load should be lesser than that of direct retrieval from the central server. The latter can also include the hardware and software cost as well as the cache server management cost. Initial web caches were in the form the browser caches, proxy caches and surrogates.

Browser caches. Browsers benefit from having a small built-in storage, which is used to store the web objects. An user decides the amount of memory allotted for the cache and whether or not the browser/website can use the cache. Typically, the cache is limited and local to a browser that cannot of shared across users. However, the recent introduction of mechanisms like web torrents [29] has changed this ideology and enable users to share caches over browsers.

Proxy caches. Unlike the browser caches, caching proxies were introduced to serve tens to thousands of users. Caching proxies were found to be beneficial, especially for large organisations such as ISPs, schools and corporations [30]. A cache proxy sits between the client and server, creating two Transmission Control Protocol (TCP) connections between the server and the client. Any request from a client would go through the proxy, and if cannot be served by the cache in proxy, the client request is forwarded to the origin server.

Surrogates. During the late 1990s, with the growing adoption of CDN, intermediate servers with advanced mechanisms such as reverse proxies, server accelerators were introduced. Such servers assumed the role of an origin server and were pooled together under a broad terminology called surrogates. RFC3040 [31], defines surrogates as intermediary servers to which the origin servers delegate the authority to operate on its behalf. Further, surrogates can be deployed at any point of the network, and the internal cache in surrogates typically delivers the response.

2.1.1 Intercache Protocols

In general, a cache should not just be reachable to clients and servers, but also other caches. A hierarchical caching structure (explained in Section 2.1.2) paves the way for a cache to serve many clients, the client being another cache or the end user [32]. When a local cache cannot serve a client request, the request is subsequently passed to the next level in the hierarchy until it reaches the origin server. An alternative co-operative caching architecture was also thought of where proxy cache peers co-operate to serve the request (Section 2.1.2). Several specialised protocols were designed to aid request forwarding among the peers and facilitate the request re-directions. We discuss four such intercache protocols below.

The Internet Cache Protocol

Internet Cache Protocol (ICP) [33] was the first-ever intercache protocol, has its roots from the Harvest project [34] and is generally a User Datagram Protocol (UDP)-based query response protocol. This lightweight protocol is used by a cache to probing the predetermined set of neighbour caches to check the presence of requested web object in the cache. The querying cache then forwards the request to first responding cache with the web object. If

none of the neighbouring peers have the object, and then the request is routed to the origin server or a parent cache of the peers.

As expected ICP was not efficient as cache misses can incur additional delays while serving a web object. ICP can also receive spurious responses, thereby consuming additional processor cycles and bandwidth. Despite the shortcomings, ICP was adopted widely at that time due to its simplicity. Other protocols were built to improve the performance of ICP.

The Cache Array Routing Protocol

Cache Array Routing Protocol (CARP) [35] uses a deterministic algorithm, unlike the realtime query in ICP, to choose the next hop. This algorithmic approach makes CARP an algorithm to maximize hit ratios and minimize latency while achieving efficient and scalable load balancing. As the request arrives, CARP calculates a score for every proxy cache and the request is forwarded to the cache with the highest score. The score calculation is predominantly based on two methods: (*i*) DNS round-robin, where a hostname is mapped to multiple IPs, and the IPs are chosen using round-robin technique and (*ii*) Uniform Resource Locator (URL) Hashing, where all the URLs are deterministically partitioned among the set of caches.

The Hypertext Caching Protocol

Hypertext Caching Protocol (HTCP) [36] builds on ICP as a UDP-based query response protocol and has a rich message structure aimed to pass more information and improve the overall performance. Unlike ICP, which just sends the request, HTCP sends the entire Hypertext Transfer Protocol (HTTP) headers. HTCP also supported authentication between the peers through shared secret keys and MD5 hashing.

Cache Digests

Cache digests [37] are supposedly the compact representation of a cache's presence among the neighbours. This representation eliminates the per-request delays encountered in ICP and HTCP. The trade-off is increased memory usage and inaccuracies in predictions. Cache digests use Bloom Filters [38] to encode set of object URLs in the cache using hash functions.

■ 2.1.2 Early Caching Topologies

Clusters.

A cache cluster is a tightly coupled collection of caching proxies, usually configured as a single service. Often the caches in a cluster are physically and topologically close to each other, for example, servers in the same rack with network interfaces having the same subnet can

form a cache cluster. Cache clusters were predominantly used as (*i*) hot spares, for example, primary and secondary DNS (*ii*) Load-sharing and improving throughput (*iii*) Improve disk and bandwidth utilization.

Hierarchies.

Alternatively, loosely coupled layered caches that are connected and co-operate are called cache hierarchies. If a cache miss happens in the lower layers, then the request is forwarded to higher layers until the request reaches the origin server. Any two caches can be related as a parent (peers), child and sibling (neighbouring cache). A child cache forward the hit misses to the parent, whereas sibling caches are designed for one-hop lookup, and the requests are never forwarded to peers of siblings. While there can be a performance improvement [32], hierarchical co-operation face issues such as routing overhead when the number of hops increases, trusting neighbour and as well as parent caches. Forwarding loops are also not uncommon in this setting.

Mesh.

Meshes are also loosely coupled caches and are very similar to hierarchies; however, the topology is flat. The concept of neighbouring caches still applies to meshes but not parent or child caches.

■ 2.2 Content Delivery Networks

As the user population connected to the Internet increased, the number of caching proxies started to grow. Caching methods at that time were found ineffective and were challenging to manage at scale. Increase in webpages with dynamic content [39] and inconsistency of cached responses, lead to the proliferation of CDN [40], also known as Content Distribution Network. CDN mainly acts as surrogates to cache content, thereby reducing origin server load, reducing latency for end users, and increasing throughput. Further CDNs help in scaling the web, improve fault tolerance and server better during disperse flash-crowd events.

CDN has now become an integral part of the Internet ecosystem and is expected to serve 71% of data traffic by next year (2021) [41]. Over the past two decades, several works focused on optimizing the performance factors of a CDN includes [42] - (*i*) Where to deploy the servers to optimize the performance? (infrastructure management) (*ii*) Which content items should be cached, and what caching policies should the CDN perform? (content distribution and management) (*iii*) How to route the content to end users (routing management) and (*iv*) How to monitor and account for the performance/pricing of CDN?

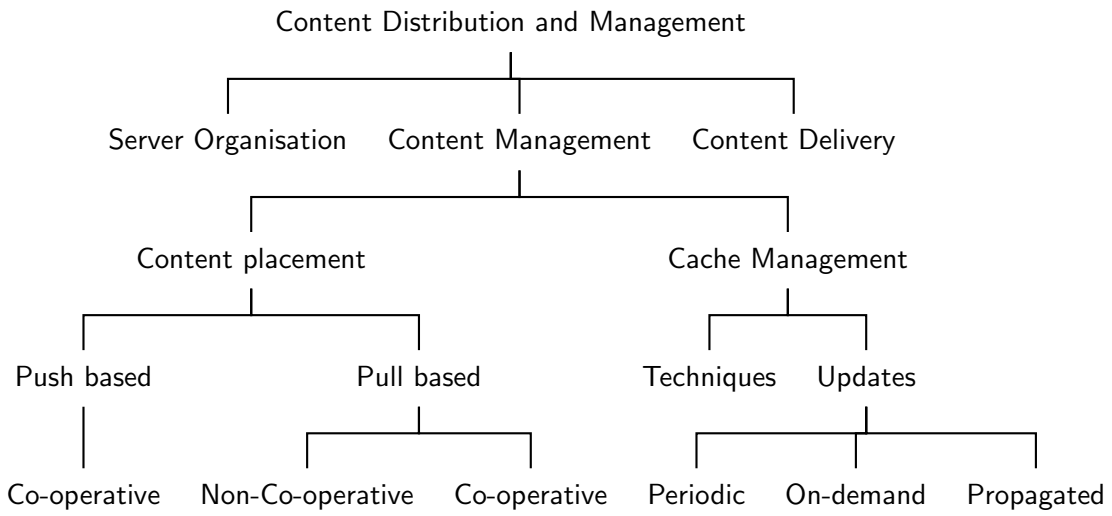


Figure 2-1: Taxonomy of Cache and Content Distribution research in CDN (adopted from [43, 44])

(accountability). The work presented in this thesis is relevant to cache and content distribution management.

2.2.1 Content Distribution and Management

Content distribution and management are treated as a backbone of an efficient CDN deployment. The activities include:

- *Server/Surrogate Organisation* to ensure the placement of edge servers at strategic locations such they are closer to the user.
- *Content Selection and Delivery* to decide optimal delivery based on incoming user requests.
- *Content Organisation* comprising of offloading content from the origin-server and caching techniques.

Server Organisation

As the QoE for accessing a content item by end users heavily rely on the location of surrogate servers, it is essential to choose the location of servers carefully. The surrogates, also known as replica servers, can either be dedicated servers or storage space in a shared infrastructure. Optimal replica server placements allow to reduce the latency of content delivery and minimize the overall bandwidth consumption.

Several techniques are proposed and deployed by the CDNs in the past for optimal server placement. The placement is usually formulated as an optimization problem with an object

to reduce the cost or improve the Quality of service (QoS) of the replica servers. The cost can mainly include (*i*) deployment cost *i.e.,* cost for leasing or building the compute, storage and network infrastructure (*ii*) network cost *i.e.,* the cost for delivering content between the replicas and users as well as origin servers [45, 46]. Improving the QoS is usually determined by identifying the network topology [47] such as flat [48–50] or hierarchical [51, 52] and network performance based on latency [53, 54], number of hops [50], bandwidth and link quality. Energy-aware replica placements [55] were also proposed.

Replica deployment mechanisms for the emerging cloud-based CDN and Network Function Virtualization (NFV) based CDNs follow similar approaches for optimal server placements [56].

Content Selection and Delivery

Content Delivery involves an appropriate selection of content predominantly to enable a reduction in load time by end-users. For example, the CDN can be used to store just some content of a webpage such as images (partial replication) or the entire content (full replication). While the latter approach is simplistic, it is not optimal or feasible to maintain consistency across replications, given the increase in the size of webpages and the number of replicas. Whereas, partial-site delivery enables delivering only the selected objects (such as embedded videos, javascript, *etc.*) rather than the whole page. The partial caching of web objects improves the performance of origin servers as well as the QoE for end users as heavy objects get loaded from a close-by replica.

More generally, partial replication of web objects to the CDN can be empirical, popularity, object, and cluster-based replication. In an empirical approach, the administrator of CDN selects the content based on some heuristics, usually an observed decision. Whereas in popularity based replication, popular objects get replicated. This approach is better than the empirical method, however, the popularity of objects vary over time, and the strategy does not work for newly available content. Object based approach is a greedy approach where content gets replicated based on an objective function [57]. Cluster-based approach replicates the clusters of web content based on the number of access or correlation among the content. The clusters are identified through users sessions (users having similar browsing patterns) [58] or URL-based [59]. More recently, reinforcement learning has also been considered for optimal content caching and outsourcing [60].

Content Management

The core performance of content distribution, thereby the CDN, relies on content management. The content management includes (*i*) content placement and (*ii*) cache management.

Content placement. After choosing the location of replica servers and the web objects to replicate, it is quintessential to decide how these replication servers get content from the origin server. *Pull-based* reactive approaches or *push-based* proactive approaches are used for such content placement.

As the name suggests, the push-based approach depends on the pre-fetching of content from the origin servers to the surrogates. In the push-based approach, proactive algorithms, based on the content selection techniques, pre-emptively store content in surrogate servers to meet estimated demand. Co-operative content placement algorithms are used to interconnect with neighbouring surrogates and serve the cache misses. Whereas, in the pull based approach, the content is pulled from the origin server as the end users request.

The efficiency of content placement algorithms depends on the cost-effective determination of which content to replicate on which surrogates. Initial works included random placement, greedy-global replication and popularity-based replication [61] to serve static web objects efficiently. However, the growth in dynamic content, mainly in the form of videos and the proliferation of cloud CDNs has changed the landscape of content placement in traditional CDNs [44].

Cache management. Content placement, in turn, depends on the caching policies of a CDN. These policies have been a business secret by commercial CDNs, mainly due to the fact that a CDN performs multiple content delivery services such as video streaming, software downloads, accelerated mobile pages, *etc.* each of which would require to have different policies.

Even before the proliferation of CDN, co-operative web caching has been studied (survey can be found here [62]). Initial proposals on techniques for CDN caching includes sharing cache digests of content the servers hold [37], maintaining a global list of content repository [63] or hashing techniques [64] or more recently, bloom filters [65]. For cache management, LRU is found to be an effective mechanism given its simplicity in terms of implementation and its innate ability to keep track of temporal access patterns [66]. New mechanisms include maximizing cache hit density [67] and lightweight machine learning approach for making optimal cache decision [60, 68] to improve cache hits.

■ 2.2.2 Challenges

Performance of content delivery in CDN was studied in the past works, including [40, 69, 70]. Improving the CDN performance were discussed at an aggregate level [40, 71–75] as well at an individual server level [76–80]. Many recent works also focus on improving cache management for content, especially videos, to derive better performance from the CDN [81–83]. Alternative CDN design patterns were also evolved to improve traditional CDN as

cloud-based CDN, Mobile CDN, Telco CDN and Hybrid CDN. User access patterns of watching videos and sharing content in online social networks also pave the way to enhance QoE when CDN serves a content item.

On the other hand, several challenges are faced by modern day cloud-based CDNs especially to handle the growing content traffic [84, 85] as follows:

Latency. The stringent growing requirement of latency-sensitive applications such as vehicular communication and remote surgery have high real-time requirements. Traditional cloud could not handle such delay constraints [3].

Bandwidth. Massive data transfers to the cloud for processing can choke the network, causing significant pressure on network bandwidth.

Resilience. With growing internet services, maintaining Service Level Agreement (SLA) is a severe challenge to cloud providers.

Energy. As the number of user sessions and the bytes transmitted to the cloud increases, the energy consumption of data centres go high.

Data Security and Privacy. Centralised analytics of the data lead to monitoring users' day-to-day activity.

The focus of this thesis would remain on bandwidth and resiliency challenges faced in the cloud era and how caching at the edge can effectively handle these issues.

2.3 Caching at the Network Edge

Content caching was introduced to handle the growing traffic explosion problem, especially due to multimedia traffic and repeated requests for popular content items and to alleviate the network burden. This led the network research community to investigate caching techniques for content item accessed via a network operator's infrastructure (both mobile and fixed-line). From an ISP's perspective, this can improve the QoE for an end user. Strategically filling the caches would not just reduce the load of a CDN, but also results in significant traffic and energy savings, thereby bringing down the costs.

Several works have been proposed on this line of the study proposing *Where, What and How* to cache.

Where? The options include (*i*) caching at the core of the network, for example, Evolved Packet Core (EPC) in 4G networks or (*ii*) caching at the network edge, for example, base stations in cellular networks and Wi-Fi access points in fixed-line networks and (*iii*) on the user devices itself to enable sharing over peer-to-peer and device-to-device techniques as well as proactive offload to mobile devices, are also considered.

What? There has been a tremendous increase in content production proportionally to raise to the content consumption. Though the storage cost is low compared to that of bandwidth cost, caching all the items is almost impossible given the scale of content production. Thus it becomes necessary also to consider what content item to store in the cache. One of the many ways to cache is based on the popularity of a content item. Past works ([83] and references within) suggest a majority of content items are unpopular and very few items are popular, depicting a long tail distribution. Other ways include caching on the user devices based on user interest to watch the content item [86, 87].

How? Bringing together the cache retention and eviction policies with the network topology (*e.g.,* when there is co-operation among various cache points) and available storage determines the gains achieved through caching.

Adopting caching techniques at the network edge has been considered one of the alternative solutions to tackle the challenges presented in Section 2.2.2. In the course of this thesis, we define the network edge based on Shi *et al.* [88] as follows:

Definition: *Network Edge*
Edge is defined as any computing and network resources along the path between data sources and cloud data centres, and edge is a continuum.

Especially in the field of wireless communications, the benefits of edge caching has been studied from multiple aspects: for alleviating network burden to capacity enhancement to improving security and user privacy [89].

2.3.1 Video Caching

Over-the-Top (OTT) video streaming applications are considered as a major contributor to Internet traffic due to the larger file sizes being delivered. CDNs are primarily used to serve this video traffic. The popularity of these video applications and a multitude of video formats (*e.g.,* live streaming, 360° videos) have paved forward research on caching for video delivery at the network edge. Early works on video caching were done through scheduled buffering at the end user device [90, 91] to improve the video delivery. Peer-to-peer protocols were introduced to prefetch streaming media content [92], later got improved for optimally placing the media files across the serving peers for on-demand video delivery [93, 94]. Even though there were several improvements then on such peer-assisted CDNs, the challenge of playback stalls during the streaming remained mainly due to inconsistent arrival rate of peers [20]. The two main reasons behind difficulty in arriving at caching decision on peers are: (*i*) stitching the chunks across peers to enable a seamless video delivery (*ii*) availability

of multiple video encoding schemes based on users' needs in terms of video quality. The latter becomes even harder when encoding like Scalable Video Coding (SVC) [95] adds a constraint of multiple layers fetched from different caches [96]. This led to optimal routing to fetch the cached video to reduce network latency [97] and peer-assisted hybrid CDNs [98]. Further collaboration of cache servers at mobile switching centres [99] was also proposed to share the resources optimally.

The focus has recently shifted to leverage edge for caching and sharing resources. Karamshuk *et al.* [86] introduced proactive offload of videos to the mobile devices when there is better connectivity (say Wi-Fi) such that the users can watch the video programmes during the commute. Alternatively, Nencioni *et al.* proposed to proactively store videos on the edge devices such as set-top boxes [100, 101] based on user interest and showed significant traffic savings based on real-world user access to a catch-up television platform. [8] extended this to social live broadcasts and showed benefits of caching a video at the edge for traffic savings. A case for joint optimization of content placement and sharing amongst the neighbourhood edge devices was provided by [10]. More recently [102] proposed joint caching and transcoding of videos at the edge and strategically sharing via X2 interface [103] in cellular base stations, where [104] as caching video chunks based on bitrate and popularity of the video at the edge.

2.3.2 ICN Architectures

Traditional host-centric data delivery model requires the host information and sometimes the content meta-data to fetch a content item. Also, it is worth noting that host-centric caching techniques lack identifying the unique objects across the sources, making the caching process ineffective. For example, multiple copies of the same content item can be present in the same server that hosts several service/content providers.

To overcome such pitfalls, sICN proposes an alternate networking architecture to this host-centric paradigm and is considered as a promising solution, especially for handling the growing video traffic. ICN provides clean-slate network design, and the main idea is to decouple the data source from the content, making the content identified by the 'name' rather than its location. A detailed survey of ICN architectures and addressing the content with their naming schemes is presented in [105] alongside how the content item is transported and routed, and the support for caching, mobility and security.

Content caching

The main features of ICN are regarded as (*i*) *Transparency*: ICN architectures use transparent naming conventions, often self-certifiable, removing the barrier of sharing a cache between the applications. (*ii*) *Granularity*: large files are saved as chunks, and the caching

operation is performed on these chunks. This enables efficient usage of the memory content and accumulation at closer caching points when interest is made from the particular consumer. (*iii*) *Ubiquitous* ICN follows a general caching structure, and the caching operation is performed on every storage element within a network.

Making optimal cache placement to reduce high cache redundancy and achieving the caching features of the ICN is quite challenging. Based on the location of the content, the caching mechanisms can be categorised as – *on-path caching* and *off-path* caching [106]. In the case of on-path caching, a content item is cached based on the decision at a specific node, the decision being made with the help of content requests. However, off-path caching can replicate the content anywhere, regardless of the path between the content producer and the consumer.

Off-Path Caching. Following an *explicit cache coordination* [107] decision, the content object is cached based on the access pattern of the object, network cache topology and cache's storage availability. Explicit cache coordination has three conventional approaches: global coordination (cache coordination involves all cache nodes), path coordination (coordination only consists of the cache nodes along the path from the request hit place to the requesting client) and neighbourhood coordination (coordination takes place among a node's neighbourhood).

On-Path Caching. Most of the ICN architectures are built with on-path caching support. This technique follows *implicit cache coordination i.e.,* on-path caching decisions apply only to the requested content and only the path between the content provider and the consumer is considered. On-Path Caching focus on granularity aspect, and a content item can be cached at three levels:

- *Object level*: Full objects can be cached at the intermediary nodes along the path of delivery.
- *Chunk level*: Similar to the object level caching technique, but the granularity of the name is different and the process requires more memory.
- *Packet level*: Each packet has a name somehow related to the object it corresponds to. The process requires more computation.

A list of prevalent on-path and off-path caching techniques is presented in Table 2-1.

Content Discovery

The next obvious question in ICN is how to efficiently discover a cached content item, which has been studied in [108–112]. Initial techniques included a hybrid forwarding strategy [108] to handle cached and non-cached content items. Flooding mechanism is used to deliver the

	Caching Policy	Description
Off-Path	Greedy	run for a number of times, at each of which, the node with the lowest cost metric, *e.g.,* latency, hop counts or an economic cost is chosen while previous choices are excluded
	Extended Greedy	the operation of the Greedy algorithm to be applied to every future replicated item and takes into account of content popularity
	Hierarchical Co-operative	The cache can be a single level/multilevel co-operative.
On-Path	Probabilistic	The nodes in the content delivery path decides whether to cache or not based on a probability p. The value p may be predetermined or can be composed of individual components.
	Random	the value of p is standardised, based on the node. Though there is no overload in the network, this way doesn't exploit content popularity. Random decision can lead to either redundant or no cache.
	Unique	content is cached only at one node, which is being chosen randomly on the delivery path.
	Everything-Everywhere	caches a content everywhere in the delivery path, leading to redundancy and inefficient storage. However, content distribution is faster.
	ProbCache	takes account of TimesIn and CacheWeight factors at each node along the delivery path. The TimesIn factor indicate the number of replicas of the content expected to be cached (depends on the time for which any content should be cached and average cache size along the same path) and the CacheWeight depends on the position of the node wrt to the content source and the consumer.The probability of caching is inversely proportional to the distance from the requester and this node.
	Last Node Caching	provisions to cache a copy at the last node towards the content consumer along the delivery path. Also, the content request is advertised first to the predecessors of the node, so that the content search is made in a Depth First Search manner.
	Leave Copy Down	cache the requested content one node closer to the consumer, avoiding large number of same copies of the content.
	Graph-related	centrality-based caching algorithms are applied at each potential caching node. Though the centrality of a node is calculated wrt the other nodes, rather than based on popularity of the content, these algorithms are proposed to be used in On-Path caching.

Table 2-1: ICN On-path and Off-path caching techniques

popular content item whereas non-popular items are delivered through the shortest path from the origin. Later, [110] used bloom filter for content discovery, facilitating content exchange between the neighbouring caches. The idea of scope-flooding technique proposed in [108] was extended by Wang *et al.* [111, 112]. A multicast signal is propagated from the requesting node to discover the content. The discovery radius is set based on content popularity.

2.3.3 Heterogeneous networks and D2D Caching

Being different from web caching and CDN caching, edge caching possesses novel research challenges. The number of users connected to the edge device varies due to heterogeneity in the edge device deployments; for example, it can be a Wi-Fi access point used by a small number of users or a cellular base station providing connectivity to an entire area. Further, when the users are mobile, the demand for edge cache varies rapidly over time depending on the deployment. Another factor to consider is that a user can be in an access range of several edge devices. The volatility in user access patterns and the network conditions, especially in case wireless networks, make it difficult to arrive at caching decisions. Several aspects of content caching at the edge in heterogeneous networks have been studied in the past. Proactive caching of content items at the cellular edge, *i.e.,* a small cell base station was proposed [113] to bring down the delay in content delivery. Similar to this, Yang *et al.* [114] proposed cache-based content delivery in a three-tier Heterogeneous Networks (HetNet)s where the edge components include cache-enabled base stations, relays, and Device-to-Device (D2D) pairs. The ideas got extended from the perspectives of mobility [115] and multi-path routing [116]. It has been found that caching on-device or at a base station depends heavily on user density connected to the network and popularity distribution of the content items [117].

2.3.4 Co-operation at the Edge

Co-operative caching has been seen from different perspectives in the past. Initial works assumed co-operation of various entities such as ISPs, CDNs and the users, thereby enabling content to be served by a set of interconnected caches [118–124]. However, in a real-world setting, it is quite hard to bring together the storage resources across the entities and enable co-operation. Further, incentive mechanisms to take part in the co-operation plays a vital role. For example, Dai *et al.* [99] proposed an auctioning mechanism through which cache servers deployed by content and service providers can co-operate. The incentive for peer-to-peer content sharing has been explored through energy credits in [7], achieving a carbon-neutral strategy for content sharing.

2.3.5 Inference at the Edge

Sophisticated Artificial Intelligence (AI) enabled platforms are widely used by several popular real-time services like visual object detection and speech recognition. Several works [125–127] focused on bringing machine learning inference tasks to the edge aimed at bringing down the end-to-end delay. There are other works on bringing the applications with lower computation requirements to the edge [125, 127] or improving the computing infrastructure at the network edge [126].

However, inference tasks are still compute-intensive, and low end-to-end delay requirements mean that this inference would need to be performed close to the user, to stay under overall end-to-end delay requirements of such applications. Several recent works have shown that moving computation capabilities and application hosting support towards the edge of the network can significantly reduce network delays for a number of important inference applications. However, most evaluations typically assume that the edge is located on the node closest to the user, typically the eNodeB [128]. Further, they either show that given current hardware, an improvement from moving to the edge is negligible for compute-intensive applications [125, 127]. Or, they show that significant gains can be obtained when compute resources are available [126]. [3] proposed a hybrid approach to route the requests such as there exists a balance between the available compute facility the edge and latency/bandwidth requirements of the applications.

2.4 Measurement & Content: State-of-the-art

2.4.1 Network Measurements

Since the inception of the Internet, network measurement techniques are carried out to collect useful information on network performance [129–132]. This not only helps to study network characteristics such as network traffic, understanding topology, protocol performance but also to envisage carrying out network operations such as network expansion, failure detection, *etc.*

Typically, two kinds of measurement techniques are undertaken to study the networks (*i*) active and (*ii*) passive measurement system. Active measurement systems involve injecting probe packets within an operational network and usually tracking the packet traffic to arrive at any conclusions on network performance. Active measurements are quite considered to interfere with the actual traffic in a network, thereby, tend to affect the network/system performance and the results from collected data. Whereas, in case of passive measurement, which does not usually interfere with traffic, rather the data traffic is monitored from one/several vantage points within an operational network. In both cases,

the data collection mechanism needs to be carefully designed. In the case of active measurements, the probe packets can interfere with existing traffic and leave an unintended consequence (*e.g.,* queuing delays, denial-of-service attacks). With passive measurements, a serious challenge is to monitor the packet traffic as the network starts to operate at scale, especially while expanding the capacity.

Tools and Metrics

Several existing tools help in performing both active and passive measurements. Common key performance metrics that are monitored during active measurements are Round Trip Time (RTT), available bandwidth, packet loss and jitter. Tools used in active measurements include ping, traceroute and iPerf. Ping provides the RTT between two nodes on the Internet, typically using Internet Control Message Protocol (ICMP) and UDP packets. Hping3 [133] is a slightly sophisticated tool which uses TCP syn-acks to deduce the network delay. The traceroute application identifies the hops between a source and destination that a packet transited with the RTT, whereas the iperf tool is mainly used for bandwidth measurements. Other tools include Pathchar [134], Pathload [135], Netalyzr [136] *etc.* which bring together a multitude of measurements.

Passive measurement techniques are usually incorporated by content, service and infrastructure providers [137, 138] as they are reliant on access existing monitoring equipment in an operational network. Wide-scale adoption of Software-defined networking (SDN) has enabled to passively monitor the network without a need for deploying additional monitoring systems.

Online Measurements

An alternative way to measure these platforms but without performing active or passive measurement is to collect data from the content and infrastructure providers who had been monitoring the network and publishing it. This includes content level data such as user posts, social network *etc.* which mainly helps in understanding the user behaviour accessing a platform. Other active measurement results that provide network characteristics are also made available online. They are usually fetched to web crawl (nerd stats from Youtube[8]), or APIs provided by the platform (*e.g.,* WiGLE wardriven data, Center for Applied Internet Data Analysis (CAIDA)). Few other share data collected by the providers, mainly through the passive measurements, with the academic community for research purposes. In such cases, data sharing policy and data anonymization are usually in place to avoid any data breaches and privacy implications. Mechanisms such as differential privacy [139] are used for such data anonymization.

[8] `https://support.google.com/youtube/thread/3284269?hl=en`

Data collection and data sharing techniques have become the common approaches adopted by measurement studies at scale. This thesis heavily relies on such data collection. Some of the challenges faced while capturing data and analysing has been explained in Section 1.3.

2.4.2 Studying the Content Systems

In the course of this thesis, we study the opportunities to improve the performance of On demand streaming, Live streaming and Decentralised social sharing through caching.

On demand streaming

A straightforward way to measure the performance of video streaming is to conduct active measurements (*cf.* Section 2.4.1). While this mimics the end-user behaviour and sheds light mainly on access network bottlenecks, experimenting at scale still remains a challenge. Several studies [140–144] have also focused on studying the traffic at the middle mile, such as passive data collected by ISP. While the ISP viewpoint provides a good insight of topology and traffic proportions across applications, the studies are often limited to one or two ISPs due to the data sharing policy of ISPs. Also, it is worth noting that due to encryption policies at the content provider's end, it becomes hard to map the content item to data traffic. In Chapter 3, we adopt a *first-mile* approach and observe the end user behaviour from content providers perspective to identify the benefits of content sharing at the network edge.

To handle the growing appetite for on demand content, several novel *edge caching* architectures have been proposed. The main principle behind most of these edge caching techniques is to store the content closer to the user focusing on a specific mechanism for content discovery (*e.g.,* via content-naming or HTTP/DNS redirection mechanisms) and access technology (*e.g.,* cellular or wired). Numerous suggestions have also been made to cache at several parts of the network, ranging from caching at all intermediate nodes between the requester and the serving node [145, 146] to caching selectively on the most central nodes [147] and hybrid approaches [148].

Our work overlaps with studies looking at improving the performance of on demand video streaming through caching content items. For instance, [149] looked at factors that affect the nationwide adoption of the BBC iPlayer streaming application. [98] uses Peer-to-Peer (P2P) swarms within each ISP to offload traffic from the content provider's server (but not the ISP). [86] preloads content on mobile phones, thereby offloading traffic from cellular networks (and minimising the user's consumption of mobile data), but adds traffic to the user's traditional broadband ISP. The work by Nencioni *et al.* [100] uses set-top

boxes to speculatively record content for future access and completely offloads requests for such content from the network.

Chapter 3 follows a line of work looking at traffic savings for the BBC content accesses. We extend the idea of content caching at the network edge and show sharing the caches between such edge devices can help in bringing down the backhaul traffic. Because our proposal relies on distributed caches, it needs to solve the content placement problem of what content to cache *where*. Moreover, we analyse the performance of caching strategies across different districts of varying population densities. Based on the insights from the skewed nature of content access, we introduce a content sharing framework, Wi-Stitch. Wi-Stitch serves as an architecture to construct a highly efficient content sharing infrastructure at the very edge.

Live streaming

Live streaming platforms, such as Periscope, Facebook Live, departs from the traditional UGC model of interaction, allowing users to live broadcast from their mobile device and online social network (unlike simply uploading videos). Similar to studies looking at the UGC [150–152], we dissect Facebook (FB) live, a prominent live streaming platform. These include user behaviour analysis [153, 154] to understand consumption patterns [22, 155–158]. This allows for a number of new lines of exploration, particularly relating to how *live* videos are created and consumed by social ties.

Perhaps the most famous user generated live streaming platform of the day is Twitch, which has seen increasing research attention [159–161]. The platform streams live computer gameplay but differs from FB Live in that it is not mobile, nor integrated within a social network. We take inspiration from the above works but emphasise *social live broadcasts platforms* such as FB Live and Periscope [162, 163] are distinct from these systems in a number of ways. They allow any user to broadcast themselves (usually from a mobile device) to their friends and possibly other interested parties. Unlike traditional user generated video platforms, live social streaming drives consumption in real-time. Critically, this is done through direct integration with a social network, notifying online users of the broadcast.

There have been a small set of studies looking at the geographical properties of video access. Li *et al.* looked at the location of viewers watching PPTV in China [164]. They found that a significant portion of videos gets at least 70% of their views from just 3 out of 33 provincial locations. Similar studies have been performed for YouTube, finding that users tend to access videos nearby to them [165]. Whereas studies have been done looking at spatial localities on Facebook [166] and Twitter [167], these studies have not inspected live broadcasts.

The major part of our work also investigates the geographical properties of these social broadcast and consumption. In Chapter 4, we dive into questions regarding user locations of brocasters and viewers, which is provided through FB API, to understand their impact on content creation and consumption. We study the integration of live social broadcast with a platform like FB which conflates two factors impacting spatial access patterns: (*i*) *Content locality*, which is the propensity for content generated nearby to be more relevant to a user; and (*ii*) *Social locality*, which is the propensity of the content generated by friends to be more relevant.

Decentralised Social Networks

Several efforts have worked toward building distributed social network platforms. At first, these were largely peer-to-peer, *e.g.,* LifeSocial.KOM [18] and PeerSON [19], and relied on entirely decentralised protocols. Unfortunately, they did not achieve widespread adoption partly due to challenges related to both performance [20] and reliability [21]. Thus, a new wave of DW platforms has emerged that rely on a server-based federated model, *e.g.,* Mastodon. Before that, the most successful platform was Diaspora, which saw growth around 2011 [168]. Diaspora offers Facebook-like functionality using a federated model similar to Mastodon. However, unlike Mastodon, it relies on bespoke protocols (rather than standards), and its user population has since stagnated.

Since Diaspora's creation, several (semi-)standard federation protocols have been proposed allowing instances to exchange data [169]. These include OStatus [170], which allows real-time interaction between instances; WebFinger [171] for discovering information about entities on other instances; and ActivityPub [172] for publishing information about social activities. Attempts have also been made to standardise the format of these data structures, *e.g.,* ActivityStreams [173]. This standardisation efforts have had a dramatic impact on the DW. For example, both Mastodon and PeerTube use ActivityStreams and ActivityPub; thus, they can exchange data. An overview of these various protocols can be found in [174].

Several works have also looked at security and privacy in the DW [175, 176], mostly around securing data management. For example, various projects have attempted to decentralise data, *e.g.,* DataBox [177], SOLID [178], and SocialGate [179]. These operate local datastores for individual users, *e.g.,* running on a physical home appliance. Applications wishing to access user data must be granted permission, potentially through a prior negotiation mechanism.

A number of measurement studies have analysed "centralised" social networks like Facebook [180] and Twitter [181–183]. These have revealed a range of properties, including attributes of the social graph and content generation. Bielenberg *et al.* performed the first study of a DW application, Diaspora [184]. When inspecting its growth, they found a net-

work far smaller than the one we observe on Mastodon. There has been a small set of recent works that focus on Mastodon. Zignani *et al.* collected and released Mastodon datasets, as well as exploring several features, *e.g.,* the social graph, placement of instances and content warnings [185, 186]. Also, studies have focused on friend recommendations [187] and sentiment analysis [188].

We complement these works with a focus on availability, covering the key aspects of the federation. We also inspect the nature and deployment of instances, as well as their topical interests. To the best of our knowledge, Chapter 5 constitutes the largest study to date of Mastodon.

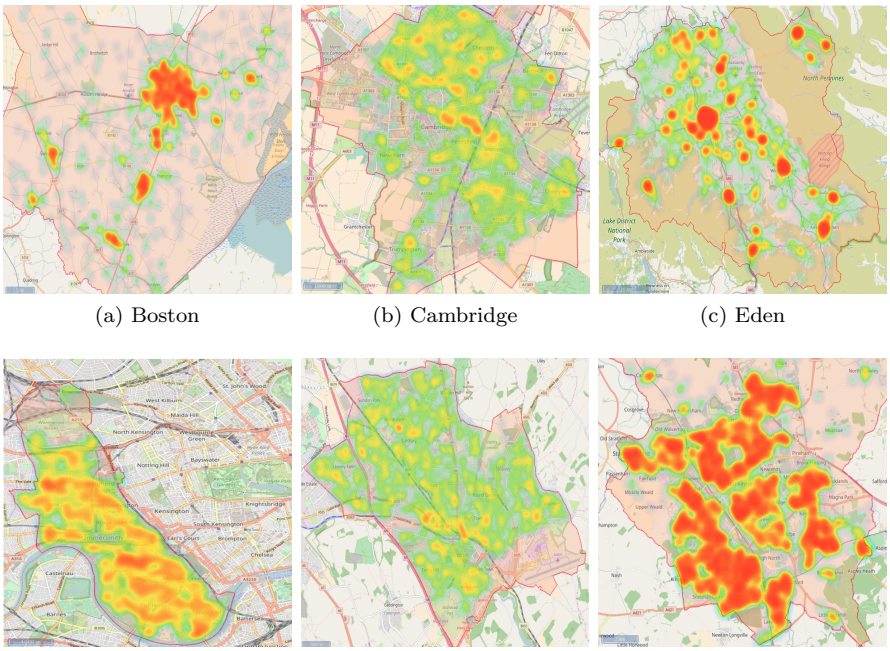
Chapter 3
On Demand Streaming and Content Sharing

Exploiting the fact that user settlements are clumped and user interest to access the content items is skewed, in this chapter we extract how the most common edge device, Wi-Fi, is distributed in the user settlements and identify the traffic benefits of users sharing content items. We deal with an application of kind A1, where both content production and the network managed by an organisation.

3.1 Introduction

Designs for future networks are driven by expansionist calls for more capacity [189]. Among other factors, these calls are driven by numerous statistics about the growth of video consumption [190], such as the Cisco Visual Networking Index which forecasts a drastic 82% increase in cellular video streaming. Although several new technologies and novel air interfaces such as mmWave and massive MIMO are being developed to meet these demands of future networks, many of these do not work well indoors. This poses a problem because over 80% of content consumption is predicted to happen indoors [191].

It is generally acknowledged [189, 192] that indoor capacity demands will need to be met using extremely small indoor cells, coupled with offloading to existing fixed-line broadband connections or investing in expensive new backhaul using fibre-to-the-home. This combination of fixed-line and cellular infrastructure, known as "fixed-mobile convergence", is expected to be a key part of future networks [193–195]. Indeed, ETSI renamed the MEC



(a) Boston (b) Cambridge (c) Eden

(d) Hammersmith (e) Luton (f) Milton Keynes

Figure 3-1: Heatmap showing the clumped dispersion of home Wi-Fi access points across six districts in the United Kingdom, indicating a potential density of caches

working group recently as *Multi-access* Edge Computing[9], specifically to "address fixed access implementation of the MEC Server (especially Wi-Fi)".

Unfortunately, the growing demand for videos requires additional capacity provisioning and may involve expensive commissioning of additional resources, such as fibre to the home, or close to the edge. This results in increased complexity and cost of content delivery.

Numerous suggestions have also been made to cache at several parts of the network, ranging from caching at all intermediate nodes between the requester and the serving node [145, 146] to caching selectively on the most central nodes [147] and hybrid approaches [148].

We wish to turn this thinking and ask whether, and to what extent, savings can be achieved by caching at the *very last hop of the network*, on a device in users' homes, and then connecting the caches together to increase sharing. This approach has several potential benefits: Developing and deploying caches on intermediate nodes in the network requires careful network planning and provisioning, as well as co-operation from ISPs, whereas caches can be deployed much more easily at the edge, even without ISP involvement. Indeed, some versions of media streaming devices such as Google Chromecast and Apple TV come with attached storage, which can serve as caches. Set-top boxes and game consoles with huge amounts of storage are increasingly connected to Wi-Fi and the Internet.

[9] `https://www.etsi.org/technologies/multi-access-edge-computing`

Therefore, in this chapter, we call for a complementary approach that makes better use of resources *at the edge*, rather than just offloading mobile edge traffic [101, 105, 196] to fixed-line broadband, or investing in network upgrades. We argue that the capacity needs of video content in converged networks can be gracefully handled by leveraging two fundamental aspects of how consumer end-points are distributed, and how they consume content: First, users are typically organised into relatively dense clusters, in cities and villages (*cf.* Figure 3-1). This has led to the so-called "chaotic" deployment of Wi-Fi [197], where end users are often within the range of a multitude of their neighbours' access points. Thus, *by sharing caches with neighbours, we can stitch together a content delivery network at the edge.*

Second, anecdotally, as well as through several studies (*e.g.,* [149]), it is known that interest in content items is highly skewed towards a few extremely popular items. Therefore, it is entirely possible that the items requested by a user have already been requested by a geographically close neighbour. The caches can be passively populated by the first user who watches that item and shares with other users who are within wireless range of the first user.

In deciding where to cache, a fundamental tension needs to be resolved: Caching close to the edge keeps the content close to the user, and therefore decreases redundant transmissions of the same content in the rest of the network, freeing up the core. However, most networks are organised in a tree-like distribution hierarchy with fewer and fewer users served by distribution points closer to the edge. Thus, a cache close to the edge serves fewer users than a cache that is more central, and proactively making a copy close to the edge may prove unnecessary if no other user underneath that caching point requests the item. Hence this calls for more selective and efficient caching by intelligently choosing cache locations [147, 148] and optimising content placement [198]. In section 3.3.5, we explain how the content can be placed based on the radio distance between the edge devices and storage availability within each edge device. We also evaluate the savings based on access to the dataset and show up to 70% of the traffic can be saved through optimal sharing of the content.

Based on these insights, we present *Wi-Stitch*, an architecture for content delivery in the converged mobile edge. Wi-Stitch caches content at the very edge of the network, *in users' homes* and uses high bandwidth 5G edge technologies such as mmWave [193] or traditional Wi-Fi mesh networks [199] to share content efficiently between neighbours. Wi-Stitch can be operated either by the Fixed-line ISP, the Mobile Network Operator (MNO) that has provisioned indoor small cells, using the Fixed-line ISP as backhaul or directly by the Content Provider (CP), without the involvement of either the MNO or the Fixed-line ISP. In each case, the entity operating Wi-Stitch deploys and manages caches on their customers' homes, and redirects content requests to neighbours' caches as appropriate. We

envision that the cache would be attached to, say, the ISP's modem in the user's home, the small cell station operated by the MNO, or a media streaming device operated by the CP (some, such as versions of Google Chromecast and Apple TV, already come with attached storage, but need to be interconnected with neighbours).

The chapter follows a line of work looking at traffic savings for BBC content accesses. For instance, [149] looked at factors that affect nationwide take-up of the BBC iPlayer streaming application. [98] used P2P swarms within each ISP to offload traffic from the content provider's server (but not the ISP), showcasing the traffic benefits. Nencioni *et al.* [100] uses set-top boxes to speculatively record content for future access and completely offloads requests for such content from the network. Here we look in detail on how co-operation amongst the end user devices can help alleviate the network burden as compared to fetching from a CDN.

3.2 Dataset

Typically, content sharing over wireless medium can happen across all the neighbours within range of each other, or conservatively, just the customers of a single ISP. Since the latter case results in a lesser density of access points, the savings are obviously going to be smaller. Therefore, as a case study, we focus mainly on the sharing possible among customers of one major nationwide ISP, British Telecom (BT). We proceed with our study making use of two large datasets:

Table 3-1: Details of the BBC iPlayer dataset

	July
Number of Users	25 M
Number of IP addresses	17 M
Number of Sessions	215 M

3.2.1 BBC iPlayer.

BBC iPlayer is a widely used video streaming application in the UK and is available for both web and mobile platforms. Most of the BBC broadcast programmes are streamed using iPlayer for on-demand streaming across the UK. Content within iPlayer can be available for up to a month, depending on licensing terms and other policies. During the period of consideration, iPlayer was one of the top video sites in the country, second only to YouTube[10]. However, unlike YouTube, BBC iPlayer hosts ad-free HD content, and TV shows much longer than the average YouTube video. Our data reported the equivalent of

[10] http://mediatel.co.uk/newsline/2014/03/28/nielsen-data-report-february-2014/

over 40% of the UK's population accessing the iPlayer. It is also worth noting that BBC iPlayer is accessible only from the country's IP addresses[11].

We consider a snapshot of access logs to BBC iPlayer during July 2014 (*cf.* Table 3-1). The access logs have been collected by aggregating over "heartbeats" generated regularly (every 10-30 seconds) or on a user action (*e.g.,* video window resize or pause) between iPlayer client software and a statistics server. All data have been anonymised before analysis. Each record in the dataset contains information about a user's session in the following format: *<network-id, isp, geographic region, user-id, content-id, content genre, session start time, duration>*

The anonymised network-id identifies each unique IP address seen in our trace separately. The ISP and geo-region corresponding to this IP address was resolved before anonymisation using the Réseaux IP Européens Network Coordination Centre (RIPE) and MaxMind databases respectively. Multiple Autonomous System (AS) belonging to the same ISP or cellular network have been manually identified and merged together for the provider we consider[12]. This also takes into account various mergers and acquisitions that are not yet reflected in RIPE. The anonymised user-id is based on long-term cookies (with a four year expiration date), that uniquely identifies each user agent separately. A single user might have more than one user-id if they use more than one device, or even if they use more than one browser to access iPlayer. Users might also get multiple IDs if their cookies expire. However, we refer to users as identified by the user-ids. The other fields contain various information about the session, such as session start time, session duration (in seconds) and the id of the video being watched.

We focus on accesses from the six administrative districts as shown in Table 3-2 and from BT customers, a subset of 3-1. The considered districts vary in terms of population densities[13], ranging from Hammersmith and Fulham (HF) in London, one of the ten most densely populated areas in the country with a population density of more than 10,000 people per square kilometre, to Eden, which has the least population density in all of the UK. Table 3-2 also gives the population density, as well as the density of accesses (number of requests and number of users making the requests) from each area.

■ 3.2.2 WiGLE

WiGLE is an open-sourced platform that uses crowdsourcing to collect the locations of wireless Access Point (AP)s across the globe[14]. As of mid 2020 around 650M Wi-Fi access

[11] Although there are known ways to break such restrictions, *e.g.,* using Virtual Private Network (VPN) end points, we believe that these constitute a minority of accesses in comparison with the volume of accesses within the country.

[12] For *e.g.,* `https://conversation.which.co.uk/technology/bt-ee-merger-mobile-phone-market-competition-o2-three/`

[13] `https://en.wikipedia.org/wiki/List_of_English_districts_by_population_density`

[14] The dataset of Wi-Fi access points used in the work can be requested from `https://rebrand.ly/wi-stitch/`

District	Area (sq.km)	Pop. density persons/sq. km	#IP address	#Content requests	Mean Content request/IP
Hammersmith and Fulham (HF)	16.40	11 213	19K	630K	33.15
Luton	43.35	4 993	41K	567K	13.82
Cambridge	40.70	3 193	41K	709K	17.29
Milton Keynes (MK)	308.63	847	71K	781K	11.0
Boston	364.90	183	15K	142K	9.46
Eden	2142.00	25	5K	90K	18.0

Table 3-2: Content access to BBC iPlayer from British Telecom customers in various locations

SSID	**Count**
Auto-BTWiFi	18K
BTWi-fi	42K
BTOpenzone-B	100K
BTOpenzone-H	130K
BT-Fon	231K
BTOpenzone	250K
BTWIFI	570K
BTWifi-X	1M
BTWiFi-with-FON	1.6M

Table 3-3: Most common BT SSID prefixes as found in WiGLE as of mid-2016 (does not include the SSIDs customised by the user)

points and ≈ 47M from the UK[15]. We use the Service Set Identifiers (SSID)s, Global Positioning System (GPS) coordinates and timestamps of first and last detection to identify the BT APs in the considered districts. We identify ≈ 4M BT access points based on the patterns. SSID strings assigned by default (*e.g.,* "BTHub3-XWX9"). Due to historical reasons, several different string patterns are observed, with the most common types of SSIDs, as shown in Table 3-3. Our method does not capture SSIDs which have been changed from the default by its users and therefore can be considered as a lower bound of the sharing possible in each region.

Mapping the datasets. The districts in Table 3-2 are grouped into cells of ≈100m radius, which is conservatively sufficient for accessing neighbouring Wi-Fi access points within the cell. This is the typical range for new technologies such as mmWave [193], and also well within the range for Wi-Fi-based mesh networks [200]. Note that our iPlayer data is anonymised, and while location information of the accesses exists (computed by IP geolocation, down to postcode or borough level resolution), this does not give the exact latitude and longitude of the access. Thus, we map the location of iPlayer requests to WiGLE access

[15] `https://wigle.net/stats#geostats`

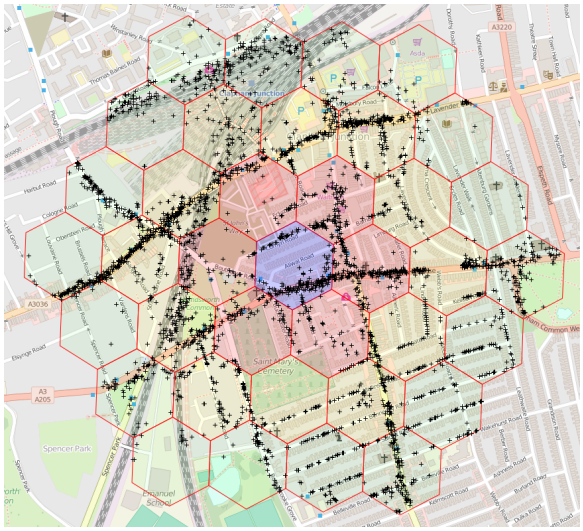


Figure 3-2: Dispersion of Wi-Fi access points in 37 adjacent cells at Hammersmith and Fulham

points within that location at random and use this combined dataset. We have checked that the results reported here are robust regardless of the exact mapping, by verifying the results are consistent over 50 different random mappings. Figure 3-2 shows the adjacent cells from Hammersmith and Fulham with the dispersion of Wi-Fi access points within the cells.

Although BBC iPlayer is currently an OTT streaming service using traditional CDNs, we use trace-driven simulations to explore the potential benefits of wireless sharing to bring down the backhaul traffic.

3.3 Exploring traffic savings from sharing

To understand the potential savings of content sharing, we first use WiGLE data to estimate the number of users who would be able to share with each other in cells across administrative districts with different population densities. Then we compute possible traffic savings by simulating content sharing within cells of different sizes. Finally, we examine the bounds on this savings if cache storage is limited and formulate an optimisation problem for placing the content to achieve maximum traffic savings.

3.3.1 Distribution of Access Points

As shown in Figure 3-1, users are clumped together in tightly knit clusters, even in the least densely populated areas such as Eden and Boston. Figure 3-3 formalises this, considering cells of a fixed 200m diameter, and plotting the distribution of the numbers of users found in such cells in each of the administrative districts. The distributions are plotted as box plots, with a central line showing the median, the ends of the boxes showing the 25th and

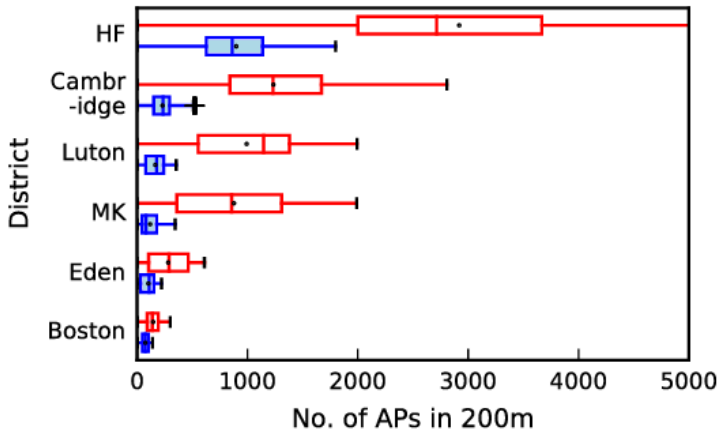


Figure 3-3: Number of users within cells of 200m diameter, considering British Telecom (BT) customers (shaded blue) and all users (red)

75th percentile, and whiskers extending from the 5th to the 95th percentile. We show both the numbers of BT customers that can be found in cells (in blue), as well as customers of all ISPs (in red), which is clearly a much larger figure. The median number of BT users in a cell ranges from 75 in Boston, a coastal city in Lincolnshire, England, to over 860 users in HF in London.

3.3.2 Potential bounds on savings

Next, we ask how much traffic savings can be achieved given the distribution of cell sizes we observe. The districts are grouped into cells having a range of populations, and we check what savings can be obtained by simulating the following scenario: Each user who requests a content item first attempts to find it within the cell. If a copy is found, then it can be served locally, contributing to the traffic savings. The first request to a content item from the cell is served by BBC servers and cached at the edge device for later use by any other user within the cell who requests the same content item. Initially, we assume that there are no storage constraints. We relax this later on and ask how the savings are curtailed by limitations of storage. We map the location of iPlayer requests to WiGLE access points as explained in Section 3.2

Figure 3-4 shows how the traffic savings increase as the number of users in a cell soar, and there are no storage constraints, leading up to ≈70% savings. The figure also shows that savings evolve in a similar way across cells of various different population densities. To test why this is the case, we fitted power law distributions to the content access requests from different cells and found that the power law exponent is similar across the regions. The distribution of power law coefficients across cells was tightly concentrated, with most cells having a power law exponent between 2 and 3. Thus we find similar levels of concentration

in accesses towards popular content items. The one exception is HF, whose traffic savings stands above the other locations. This behaviour can be explained from Table 3-2, where we see that the number of requests per user is higher for HF than in other locations. In other words, users in this metropolitan London location are heavier users of iPlayer than in other places, leading to a larger and more diverse cache, which in turn leads to a better hit rate and more traffic savings overall.

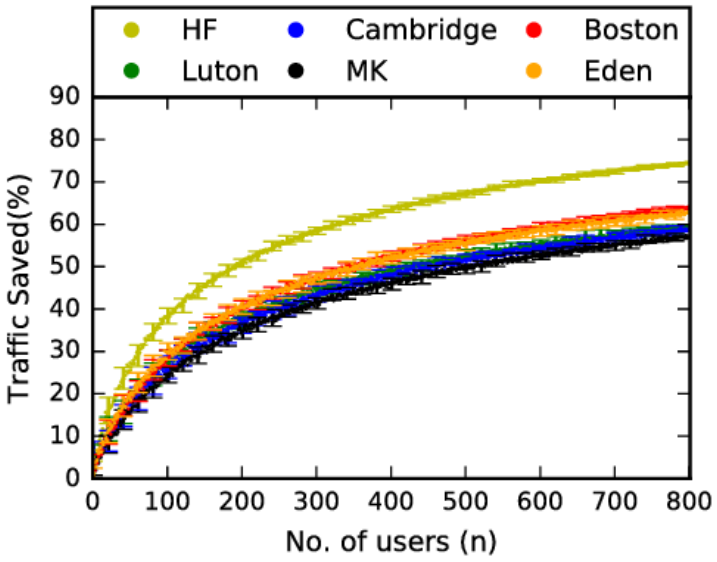


Figure 3-4: Traffic Savings with 95% Confidence Interval (CI) across the districts with n users in a cell when the content is opportunistically cached in the local device and made available for the whole cell

3.3.3 Effect of storage limits

The savings in Figure 3-4 are a result of opportunistic caching assuming an infinite store. To check the effect of storage limits, we compared various common cache replacement strategies, namely Least recently used (LRU), Least Frequently Used (LFU), and First In First Out (FIFO). Further, we check the performance of these traditional cache update algorithms against an oracle that can look into the future to determine which items will be accessed and how often, and thus can make the best possible cache replacements.

Figure 3-5 shows the performance of various cache replacement techniques. To check how the performance varies for different cell sizes, we show the traffic incurred across storage levels for a cell with population n=120, 240 and 360 (which covers typical cells in most areas except for HF, where savings are higher than other places). Storage levels are varied from 0% (no cache) to 100%, which is the cache size corresponding to the actual data consumed by the users within that cell in that month.

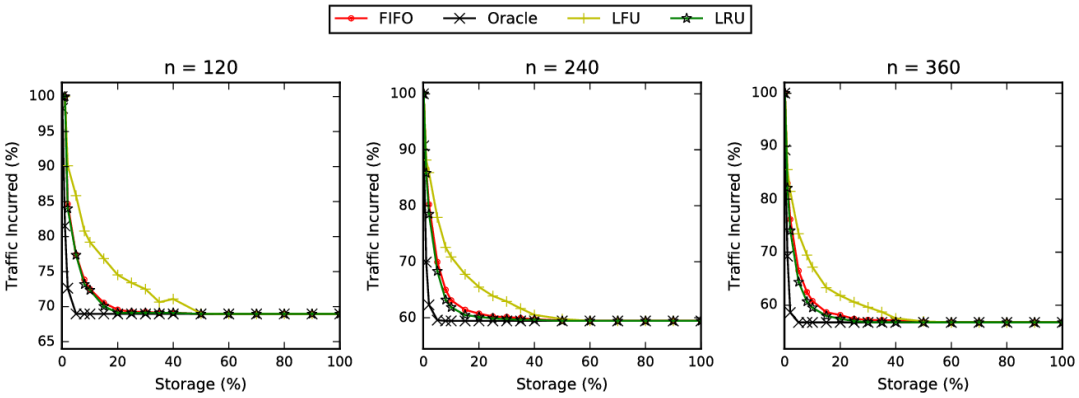


Figure 3-5: Traffic Incurred for several edge cache store policies, across cells of varying user population n (n=120, 240 and 360)

Note that there is a minimum level of traffic that cannot be avoided since the first user in the cell always has to fetch the item from the origin server. As expected, Oracle performs better than the other techniques, and bottoms out to the minimum traffic levels achievable quickly. However, it is interesting to observe that all the other policies also converge to this minimum level – *i.e.,* given reasonable amounts of storage, it is possible to approximate the best possible cache replacement policy (oracle) and pick the "best" items to replace under a particular storage constraint, using various well-known heuristics such as LRU, LFU and FIFO.

As found in other similar studies [201, 202] on effective cache replacement schemes, LRU is the best performing of these policies and can approximate the Oracle even with only a small amount of storage (≈20% of total data used), followed closely by FIFO. Unexpectedly, we see FIFO performing better than LFU. We conjecture that this may be due to the periodic nature of iPlayer content – as newer episodes of TV shows come online, interest in older episodes wanes, and therefore it becomes less likely that there will be a future access request for older episodes. In other words, interest in content items may naturally follow a rough FIFO order. Although an episode has been frequently accessed in the past, it may be unlikely to be accessed as much as a less frequently used but newer episode, making FIFO a better policy than LFU.

3.3.4 On Scalability of Sharing

While Section 3.3.2 indicated a large potential for traffic savings due to shared interests and sufficient neighbours, in order to realise the traffic savings, it is equally important that the sharing infrastructure is able to support the peak demand of iPlayer usage at a time. Peak demand is captured in Figure 3-6 for a random 2K people (more than the maximum number of users in any given cell of users who can access each other over Wi-Fi), across

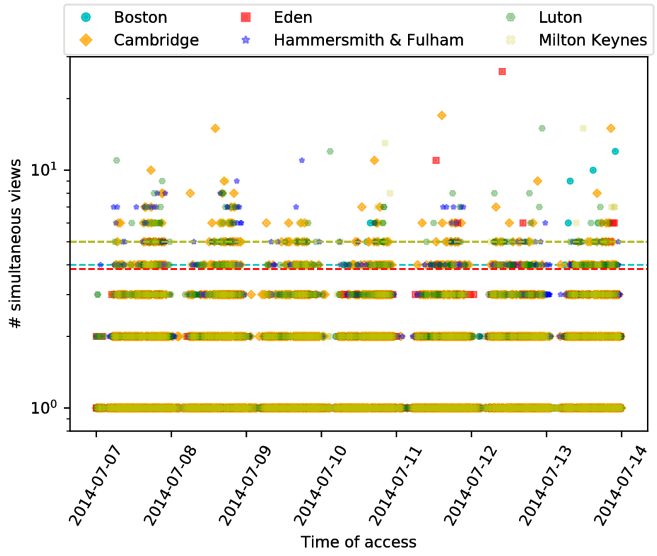


Figure 3-6: Simultaneous accesses across the location over a week (at 2 minute interval) with the lines indicating the 98^{th} percentile of total accesses

the six districts. Note that on any given time, the *maximum* number of connections is typically no more 3–5 across the cell, and in the worst case, is only about 27 simultaneous connections surprisingly at Eden. We conjecture that this is a result of the on-demand nature of access which spreads the load over time, leading to lower peak loads [100]. Thus each cell typically has to support only a handful of simultaneous connections with content served by neighbours, well within reach of Wi-Fi.

■ 3.3.5 Towards strategic content caching and sharing at the Edge

Given that, sharing the infrastructure and a strategic content placement would help in achieving traffic savings with even handling the peak demand, in this section, we look further into *how* this strategic content caching can be performed.

The content can be stored in a centralised or decentralised fashion, and the content placement for this can happen through coordinated or uncoordinated approaches. In a centralised cache, there is a single cache for the entire cell (*e.g.,* a cache on the base station in a traditional cellular setup). In the decentralised cache, storage is distributed on all the nodes within a cell. The mechanisms for an example case of sharing over the cellular infrastructure has been presented in Figure 3-7. With a coordinated approach, a decision is made on where (*i.e.,* which node) to store each content item, based on storage constraints on all the nodes, reachability (quality of wireless link) to its neighbours, and the probability that its neighbours access that item. The uncoordinated approach caches data reactively: For each content accessed, the accessing node first attempts to find it on neighbouring

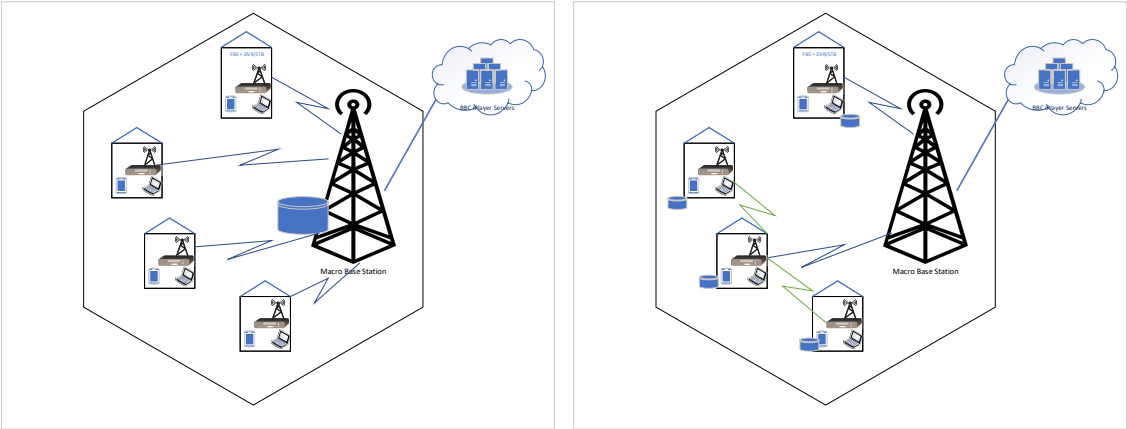


Figure 3-7: (a) A centralised caching mechanism where there is a single cache for a cell at the base station. (b) A distributed caching mechanism with cache located on all the nodes within a cell.

caches, and accesses from the origin server if not found locally. This is then cached locally if storage is available on the node and then made available to neighbours. We first formulate an optimisation problem for managing coordinated content placement in a decentralised implementation and contrast with various other content placement schemes. Whereas our approach solves one particular optimisation problem (*i.e.,* maximising traffic savings), it would be interesting to combine some of these alternate approaches, if other optimisation goals are important. We focus on traffic savings as we consider here video streaming as an application; however, the traffic optimisation goal can be substitute with latency-related parameters to reduce overall delay in accessing the content item. One other reason behind our choice is that the data provided by the BBC carried traffic-related statistics rather than delay related statistics. Intuitively, the traffic savings achieved by sharing content items at the edge brings down the energy/bandwidth costs for network operators as well as CDN [7].

Formulation

To have optimal content placement for minimal traffic flow at the backhaul, we consider the two main parameters, (i) storage within a node and (ii) resource availability between the nodes that share the content. Traffic at the backhaul can be given by:

$$T_{BH} = T_1 + T_2 \tag{3.1}$$

where T_1 is the traffic consumed due to the mandatory first access to the content c from a repository C. T_2 is the traffic consumed when the content is available with any of neighbours but cannot be fetched from the neighbour due to lack of resource (*e.g.,* sub-channel) availability. T_1 and T_2 for a cell containing N edge devices can be given by:

$$T_1 = \sum_{i,c} x_{ic}|c|, \quad i \in N, c \in C \tag{3.2}$$

$|c|$ indicates the size of the content and x_{ic} is a binary decision variable that represents whether to store the content c at node i.

$$T_2 = \sum_{i,c} (1 - \nu_{ic})\pi_{ic}|c| \tag{3.3}$$

where π_{ic} indicates the probability of content c being watched by i and directly depends on the accuracy of the prediction technique used. ν_{ic} is a binary variable that indicates whether the content is available locally (i.e., on node i or any of its neighbours). This is calculated as follows:

$$\nu_{ic} = \begin{cases} 1, & \text{if } \sum\limits_{j \in N} x_{jc}(1 - p_{ji}^{outage}) \geq 1 \\ 0, & \text{otherwise} \end{cases} \tag{3.4}$$

where p_{ji}^{outage} is the outage probability of link between i and j as derived in [5].

Thus, the optimisation problem for minimizing the traffic at the backhaul can be given by,

$$\min_{\boldsymbol{x}} \quad T_{BH}, \tag{3.5a}$$

$$\text{subject to} \quad \sum_{c} x_{ic}|c| \leq S_i, \quad \forall i \tag{3.5b}$$

where content storage management in each node is taken care by the constraint on S_i (the storage at node i). Thus, each content c is optimally placed in a node i based on decision variable x_{ic} in such a way to minimize T_{BH}.

Evaluation

To understand the potential benefits of the content sharing through a coordinated placement (equation 3.5a), we compare the traffic savings with that of an uncoordinated placement and across centralised (*e.g.,* [113]) and decentralised approaches.

Setup. Equation 3.5a provides a generic optimisation problem, that can be applied to a wide range of on demand streaming platforms. In this context, we consider BBC iPlayer

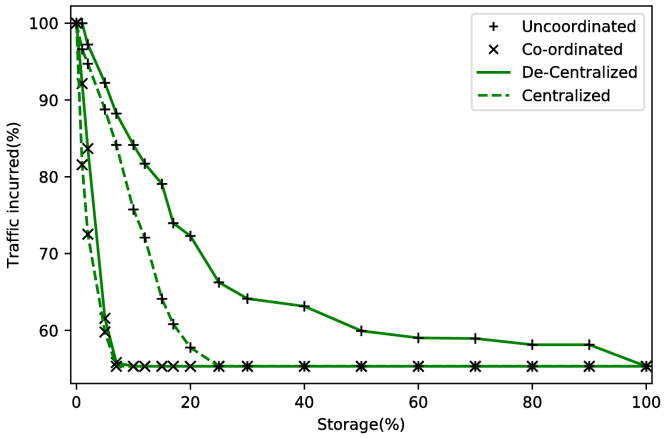


Figure 3-8: Traffic savings for various combinations of operation {centralised, decentralised} x {coordinated, uncoordinated} approaches

data which contains the user access to the videos, comprising mostly of catch-up TV shows mapped to the real-world location of the edge devices as explained in Section 3.2. This ensures content-related parameters such as the size of the video (storage of the node), bitrate at which the video is streamed (the capacity requirements between the nodes) are extracted from real-world settings. Apart from assumptions on wireless conditions for sharing described in Table 3-4, for evaluating p_{ij}^{outage}, we use WiGLE (section 3.2.2) data to calculate the distance between the nodes. Note that, due to the effect of interference, the power at which an access point can transmit, depends on the location where it is deployed [203]. In this case, this is assumed as 100 milliwatts, given that Hammersmith and Fulham is a densely populated area. The formulated non-linear optimisation problem is solved using Solving Constraint Integer Programs (SCIP) [204].

Figure 3-8 shows the traffic incurred based on the storage level across each of the nodes for centralised and decentralised mechanisms. It is quite clear from the figure that at lower storage levels, the coordinated placement outperforms the uncoordinated approach for both centralised and decentralised approaches. This also validates that our optimisation problem tends to achieve the upper bound of savings – the optimum traffic saving of ≈40% can be achieved even at a lower combined store of ≈10%.

Figure 3-8 also shows that even with uncoordinated (reactive) content caching we can achieve a close to optimal traffic savings with an additional store of 10% - 15%. It is worth noting that the coordinated approach also depends on the prediction capability of the node. We plot the savings from a coordinated decentralised cache setting with respect to the prediction power by adjusting the accuracy of the content prediction algorithm. A predictor with accuracy lower than 50% leads to spurious cache storage decisions that actually increase traffic over the baseline of no caches. At higher levels of accuracy, performance edges closer

Parameter	Value
Total Bandwidth (B)	22 MHz
Average path loss	35.3 + 37.6 log(d_{ij})
Fading model	Rayleigh
Total number of channels	14
Shadowing	Log-normal (μ=0-dB, σ=8-dB)
Total Transmission power	20 dBm
Background noise power	1
Distance between the nodes (d_{ij})	15m - 150m

Table 3-4: Parameters used in the content sharing simulation

to the Oracle. State of the art content access prediction algorithms reach accuracies of nearly 95% [86]; thus, we may expect close to Oracle-like savings (within 10% of optimal).

3.4 Wi-Stitch: A framework for content sharing

Based on the insight gathered so far, in this section, we propose a content sharing framework called **Wi-Stitch**. Wi-Stitch exploits the current chaotic/clustered deployment of wireless infrastructure as well as the locality of interest to enable collaborative cache sharing in converged edge networks. Wi-Stitch enables a distributed content delivery network at the very edge of the network, through caches in users' homes. We ask and answer three questions: who controls Wi-Stitch, how Wi-Stitch caches are managed and populated, and how does a client retrieve content from Wi-Stitch.

Who controls Wi-Stitch, and how? Wi-Stitch may be operated by several kinds of entities: A fixed-line ISP could offer Wi-Stitch as a value added service, for its own customers using streaming video services, as well as for mobile customers of MNOs who are using its infrastructure as backhaul in small cells. In this case, the cache can be attached to ISP-owned Wi-Fi routers or Digital Subscriber Line (DSL)/cable modems. Alternately, MNOs can deploy Wi-Stitch for their customers, attaching the caches to small cell base stations. Similarly, some Content Providers may have a hardware footprint in the user's home and these typically come equipped with storage (*e.g.,* X-Box and other gaming stations, Apple TV and certain versions of Chromecast) and can deploy Wi-Stitch as well. However, they would need to interconnect with each other, either by deploying a parallel Wi-Fi mesh network or by reusing the ISP's Wi-Fi router.

In each case, the goal of the operating entity is to present a unified view of the distributed caches as a single virtualised cache. This may be done using several approaches: One possibility is to use Information-Centric approaches (*e.g.,* [205, 206]) to route users' requests for content to the appropriate caches. A second possibility is to use HTTP- or DNS-level

redirections similar to current CDNs [207]. A third possibility is to currently popular SDN and NFV technologies to implement the caches as a virtualised network function [208].

Wi-Stitch is agnostic to which entity deploys it, or which technology is used to realise the connectivity. Rather, the suitability of Wi-Stitch and the savings realised through this approach depends strongly on how well the caches between neighbours can be shared, which we evaluate in Section 3.3.

How are caches managed and populated? Wi-Stitch is based on the insight that human settlements are clumped together, with a number of homes often located within close proximity of other homes. Thus, Wi-Stitch endeavours to stitch together a number of caches in neighbouring homes into a "cell" which is managed as a single virtualised cache such that the content of every home being made available to every other home within the cell. This could be enabled by connecting such caches using a wireless mesh network [199].

We consider a straightforward approach where Wi-Stitch caches in user homes are reactively populated when the user watches a content item. Thus, gains are dependent on the overlap between neighbours' content viewing patterns. In certain cases, it may be possible to proactively populate content. For instance, BBC programmes are also broadcast over the air, and this has been exploited for proactive caching [101].

How are cached items retrieved by clients? Wi-Stitch maintains a mapping of which content item has been delivered to which nodes, and redirects content requests to the appropriate cache. This could be done transparently to the client (*e.g.,* using ICN primitives if adopting an ICN solution, using SDN rewriting at the home router if using SDN (e.g. [199]), or with the content provider using HTTP or DNS redirection). Alternately, the client could actively be involved and connect to the appropriate access point to directly access the content from there.

Thus, the chapter highlights the benefits of strategic content sharing at the edge in the form of traffic savings for a widely used on-demand streaming application.

Chapter 4
Live Broadcasts and Content Caching

Experiencing videos as and when they are produced is becoming a trend, and these videos are often called as "Live" videos. Typically users broadcast live videos over social media, making the video a "social live broadcast". In this chapter, we explore Facebook Live, an application that falls under the category A2 and exploits the benefits of caching at the edge.

4.1 Introduction

Besides the content produced by a production media like BBC (Section 3.1), online sharing of *user-generated content* is now a day-to-day activity for many users around the world [191]. Whereas traditionally this has followed a "static" upload model (where users create a video offline and then upload their content on a platform like YouTube, BBC for on-demand viewing), recent years have seen a noticeable shift towards live equivalents. Mobile applications such as Facebook Live, Periscope and Instagram Live allow users to easily broadcast their activities to a potentially global audience. We term this medium *social live broadcast* and distinguish it from more traditional forms of live streaming such as Twitch, which are not based on social platforms and are not impacted by the various cultural and demographic implications of social graph-based delivery.

One prominent example of the live social broadcast is *Facebook Live*. The platform is simple: it allows any Facebook user to "go live" and stream their mobile camera feed to friends (and others if public). An interesting twist is that the video optionally remains available after the broadcast, to be viewed on-demand, thus offering functionality similar to traditional user-generated platforms like YouTube. From a user behaviour perspective, Facebook Live is a particularly powerful platform due to the presence of both amateur and

professional users, and the combination of live and non-live video delivery. Further, with a strong social network and a global reach, Facebook allows us to extract social features and geographical patterns. From an infrastructural perspective, it also opens up several challenges and opportunities, *e.g.,* to exploit social localities for edge caching.

In this Chapter, we make two broad contributions. First, we characterise user behaviour in the mobile live social broadcast. Second, where appropriate, we highlight relevant implications for mobile live social video delivery derived from our observations. To this end, we have collected a large-scale dataset covering one month of activity on Facebook Live. This dataset encompasses not only information regarding access patterns but also geographical indicators on where broadcasters and viewers are located. To the best of our knowledge, we are the first to study Facebook Live and, importantly, the first to derive key systems insights. Whereas the limited body of past research has focused on performance aspects [162, 163], we strive to understand its end user characteristics, as well as how this might create architectural design opportunities.

We begin by characterising broadcaster and viewer properties (Section 4.3). We quantify Facebook Live's popularity, observing 3.2 million accounts performing broadcasts, with a peak of 62 million online viewers (Section 4.3.1). We identify three types of broadcasts. The majority (95%) are from social users on Facebook's mobile app, whilst the minority are from Facebook page accounts (4%) and applications using Facebook's third party API (1%). Although social users typically perform short broadcasts, we find a non-negligible number of mobile users with broadcast sessions exceeding one hour (Section 4.3.2). However, nearly half of these streams go unwatched, even for regular broadcasters. Despite this, we find that the Facebook app uploads all content regardless of viewers, wasting user device battery and network resources. We, therefore, propose a simple scheme, whereby content is locally cached on a device until viewers arrive. We find that 21.91% (21.33TB) of data could be offloaded from delivery through this simple innovation (Section 4.3.3).

We then dissect the geographical patterns of both broadcasters and viewers (Section 4.4). We confirm that Facebook Live has a global reach (Section 4.4.1). The majority of popular broadcasters are based in Europe, South East Asia, Brazil and the East Coast of the United States. Driven by social properties, most broadcasts are highly localised. For example, 8% of viewers are within a city-level radius of 25KM from the broadcaster, accounting for 1.57M viewers. That said, we also observe a significant proportion of international consumption too (overall, 39.8% of viewers are domestic, and the remaining are international). There are some distinct power players here. For example, residents in the United States stream a disproportionate amount of broadcasts to Mexico and Canada. This highlights well the social properties of Facebook Live, whereby users tend to consume directly from friends (who may be in different countries due to immigration).

Finally, we refocus our attention on user engagement (Section 4.5). We find that popular broadcasts consistently garner views rapidly. However, this is not the norm — the median view count typically remains below ten throughout the entire duration of streams. We also compare this against archived on-demand videos, *i.e.,* streams that were live but subsequently made available for later viewing. We find that, in fact, most engagement (comments, likes, shares) happen *after* a stream has been archived, thereby demoting the importance of the live component. We finally combine the above observations to question to what extent Facebook Live is truly live, social and broadcasting.

4.2 Dataset

To explore Facebook Live, it is first necessary to collect a large and meaningful dataset of both the broadcasters and viewers. Hence, we begin by describing the data collection methodology employed.

4.2.1 Data Capture Methodology

We have written a crawler that automatically collects information on publicly accessible Facebook broadcasts. These are taken from the Facebook Live map[16] which publishes broadcasts in realtime. The data collection used the Scarpy[17] framework[18]. A sample screenshot of the map is provided in Figure 4-1. It can be seen that map provides various information such as the unique id of live broadcast, duration of the video, live number of viewers, details of broadcaster and geo-coordinates of the broadcaster and viewers watching the video.

The crawler collects this data every two minutes, providing a periodic snapshot of all geo-tagged public streams in the system (including their metadata). This was executed between 7-Nov-2016 and 2-Dec-2016, resulting in 3TB of data.[19] This covered 6.5 million broadcasts by 3.29 million unique broadcasters and viewed by a peak of 62 million users. Amongst several events captured, our dataset covered the US Presidential election of Nov 2016, Thanksgiving (26^{th} November) and Armistice Day (11^{th} November).

On Facebook Live, a broadcaster can go live using any of the following three mechanisms: (i) **From App**: Users or pages can create live streams using the Facebook App or Facebook Mentions[20] from their Android or iOS devices. Note that the feature of going

[16] fb.com/live

[17] `https://scrapy.org/`

[18] the code to collect data can be found here: `https://rebrand.ly/fb_crawl`

[19] Note that between 18-Nov and 22-Nov the measurements ceased. This was due to an unavoidable server failure. Consequently, this time period is omitted from our later analysis.

[20] `https://newsroom.fb.com/news/2015/08/connect-with-public-figures-through-live/`

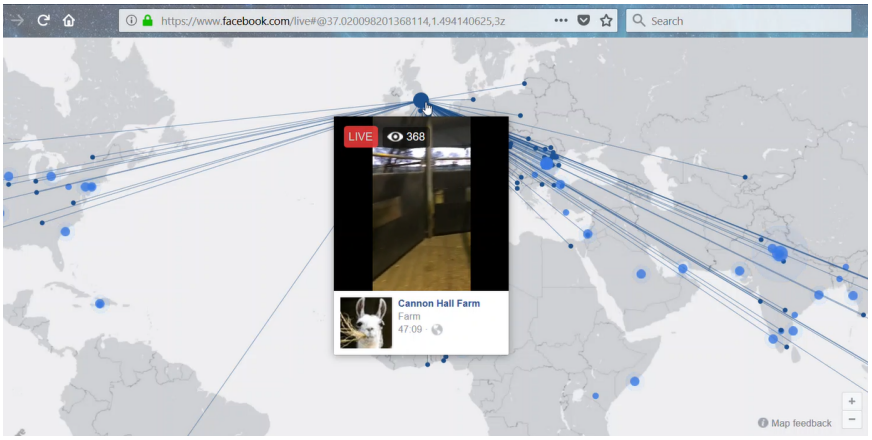


Figure 4-1: Live map, as published by Facebook, showing the global broadcasts and the reach of live videos.

Attribute	Total	NA	EU	AS	AF	SA	OC
# of broadcasts	6.5 M	1.9 M	707 K	2.66 M	117 K	810 K	52 K
# of broadcasters	3.29 M	856 K	336 K	1.35 M	64.6 K	418 K	25.4 K
# of broadcasts with over 100 views	61 K	15.2 K	8.2 K	23 K	1.5 K	6.4 K	426
# of broadcasters with over 100 views	19 K	4.8 K	3 K	6.5 K	522	2.5 K	163
Peak # of viewers	60 M	17.6 M	7.39 M	20 M	1 M	6.8 M	387 K

Table 4-1: Summary of dataset separated across continents. Peak view counts are computed cumulatively across all videos in a snapshot (originating on the continent).

live from the browser was introduced three months after the crawl.[21]

(*ii*) **From Publisher tools**[22]: Facebook Pages[23] can use this feature, allowing them to use any external device or software to stream the live content.

(*iii*) **Using Developer API**: Developers can use the live API[24] and embed it in their application to broadcast their content through Facebook.

95%, 4% and <1% of broadcasts belong to each of these above categories respectively. For every broadcast, the following data is collected every two minutes:

- **Broadcaster metadata**: This carries the broadcaster's user details such as the username and the geo-tag coordinates (*cf.* Section 4.2.2). The broadcaster can be either an individual user account or, alternatively, a Facebook Page (typically created for organisations, *e.g.,* political parties). In the latter case, we also collect information

[21] https://newsroom.fb.com/news/2017/03/new-ways-to-go-live-now-from-your-computer/
[22] https://www.facebook.com/facebookmedia/get-started/live
[23] https://www.facebook.com/business/learn/facebook-page-basics
[24] https://developers.facebook.com/docs/videos/live-video

about the category of the broadcast page, *e.g.,* Media, Sport, Shopping *etc.* Geo-tags are added by the user based on their GPS coordinates; if a user does not share location information explicitly, any publicly accessible location information in their profile will be used.

- **Viewer information**: The number of live viewers per broadcast, as well as the location coordinates of the viewers (returned by Facebook).

Engagement attributes are also collected, namely the number of likes, shares and comments during the broadcast. After eight months we then revisited any videos that had been archived for later on-demand access, collecting the number of likes, shares, comments and the bitrate (46.1% of live streams were archived). Table 4-1 provides a summary of the dataset across different regions.

To give context regarding the *types* of content material, we briefly inspect the categorical tags within the Page broadcasts. Figure 4-2 presents the fraction of broadcasts that fall into each category, as well as the fraction of the overall viewership that each category attracts. Media, which covers things like news and TV content, gains by far the highest viewership (over 50%), despite constituting less than 20% of broadcasts. "Religious Place of Worship" is second most popular by the number of broadcasters, driven by the US and Brazil where they constitute over 30% of the streams. Measuring across the world, however, 40% of the religious broadcasts do not garner even a single view. Other content types ranging from arts to shopping can be seen of assorted popularities.

■ 4.2.2 On Facebook's geo-coordinates

As stated above, we collect geo-coordinates for viewers and broadcasters. These are used by Facebook to visualise locations on a public map for users. Before continuing, it is important to validate the fidelity of these coordinates and explain the level of sampling Facebook utilises when returning locations of viewers. Every broadcast is accompanied by a longitude-latitude pair from where the broadcast happens and the viewer locations are reported for the broadcasts that have greater than 100 views.

All the broadcasters have geo-tags, whilst the mean percentage of viewers that also have locations reported is 55%, with a standard deviation of 12.83. We regularly observe broadcasters reporting multiple locations per broadcast (this could be because the user has moved or because Facebook has introduced noise to protect the location privacy of broadcasters). To explore this, for each broadcast, we take all reported coordinates for the broadcaster and compute the centroid of the coordinates [209]. We then calculate the maximum distance between the centroid and all geo-coordinates reported — this gives the upper bound of any noise potentially introduced. Figure 4-3 presents a Cumulative Distribution Function

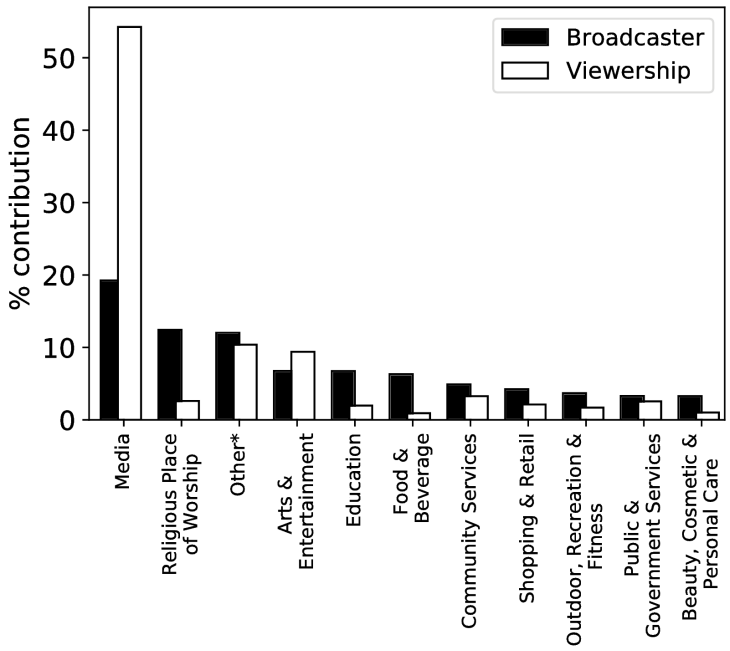


Figure 4-2: Fraction of broadcasters and viewers for each category. Note that "Other" is a category provided by Facebook.

(CDF) of the results. We find that *no* broadcaster reports a distance that varies greater than 7 KM from the centroid. This indicates that any noise (if introduced) will only undermine accuracy by this upper bound; this is sufficiently accurate to allow global-scale geo analysis. Finally, we convert each geo-coordinate into its country of origin using country polygons. For subsequent statistics related to distance, we utilise Vincenty's formula for computing the distance between coordinates [210].

4.2.3 On Facebook's infrastructure

Before continuing, we briefly outline how Facebook Live operates from an infrastructural perspective [211]. When users go live, they are redirected to their closest Point of Presence (PoP) using Real-Time Messaging Protocol (Secured) (RTMP(S)). Video data is adaptively encoded by the Facebook mobile application and then uploaded to a data centre via the PoP. Typically, data is uploaded in a high definition format, but the quality is degraded in cases of poor connectivity. Any users wishing to view the stream download a manifest file, and begins requesting listed video chunks via Facebook's CDN (mainly Akamai), where caching also takes place.

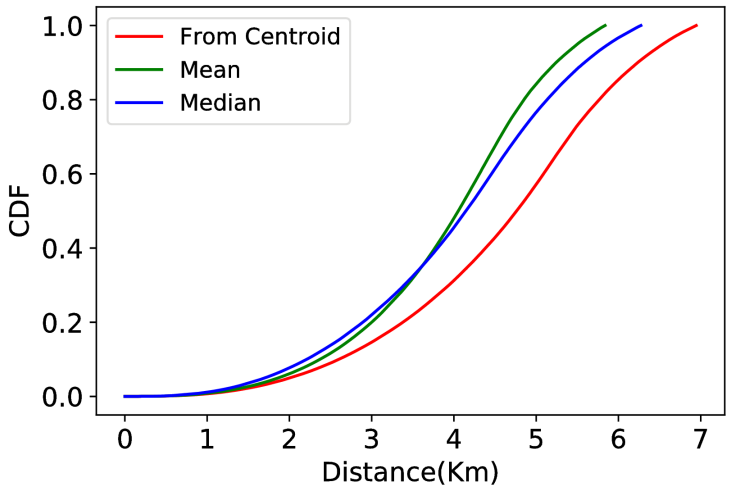


Figure 4-3: CDF of maximum distance from the centroid of locations reported for each broadcaster, as well as the mean and median distance between reported locations per broadcast.

4.3 Characterising Live Broadcasts

We begin by characterising the global activities of broadcasters and their viewerships on Facebook Live.

4.3.1 How popular is Facebook Live?

First, we look at the overall popularity of the platform, as measured by the number of broadcasters and viewers. Figure 4-4 presents an hourly time series across our dataset. It counts both the number of broadcasts (Y1-axis) and viewers (Y2-axis). We see a roughly stable trend for viewers, but erratic trends for broadcasters. This is largely due to a high number of broadcasts in November 2016 due to the US Presidential elections. We also observe periodic spikes in utilisation – these centre on weekends, indicating the social nature of the broadcast platform.

Whereas the above quantifies how popular Facebook Live is, it does not demonstrate the popularity of individual broadcasters. This is typically a more important metric for users wishing to use their broadcasts for promotion. Figure 4-5a presents a CDF of the peak view counts[25] for each broadcast. Two lines are present: (*i*) Broadcasts by **User** accounts, which are generated by individual Facebook users on their mobile device and (*ii*) Broadcasts by **Page** accounts, which are generated by Facebook Pages.

It can be seen that Page broadcasts gain significantly more viewers than typical user broadcasts. To add further context, Table 4-2 presents the Top 10 broadcasters, ranked by

[25]Note that the number of viewers changes as viewers join and leave a broadcast. Since our trace collects data at 2-minute intervals, we show the peak number of viewers recorded.

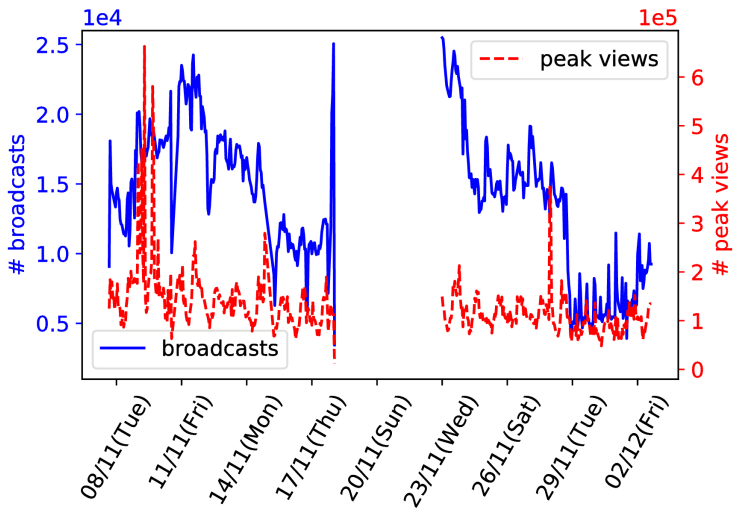


Figure 4-4: Cumulative number of broadcasters and viewers per hour during the measurement period. Note that between 18th and 22nd November, monitoring ceased due to a failure in our crawler.

peak viewing figures. It can be seen that 7 out of the Top 10 broadcasters are Pages. Even though Page broadcasters only contribute 4.26% of the streams, they collect almost 35% of all views. In stark contrast, we find that 41.5% of user broadcasts *never* gain a single viewer, whilst 55.35% of broadcasters have at least one stream that remains unwatched. In fact, the median view count for user broadcasts is just 1. This raises questions regarding the extent that Facebook Live is actually used as a social *broad*cast medium in practice, as opposed to a unicast stream that is seen by one or even zero viewers.

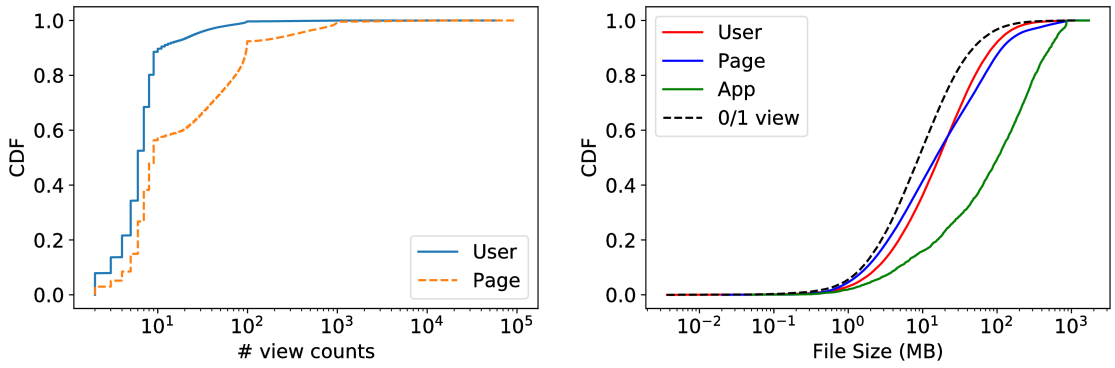


Figure 4-5: (a) CDF of peak view counts (for broadcasts with a view count greater than 1). Broadcasts are separated into those generated by Users and Pages. 41.47% and 37.39% of the user and page videos go unwatched; 9% of user videos and 5% of page videos get just a single view. (b) Size of uploaded broadcast files, for user, page and app videos, as well as videos with 0 or 1 viewers.

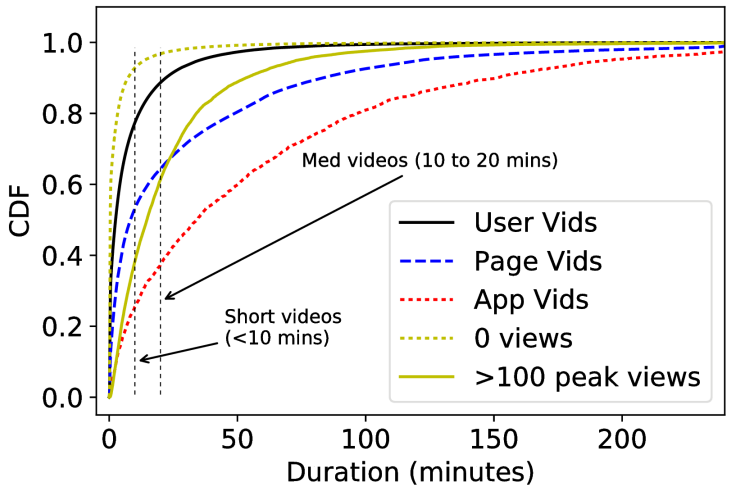


Figure 4-6: CDF of broadcast duration. Broadcasts are separated into streams generated by Page accounts and User accounts, as well as popular (>100 peak views) and unpopular (0 views) streams.

To study this further, we separate all users into buckets based on the total number of broadcasts they have performed. For each bucket, we compute the fraction of broadcasts that gained zero viewers, one viewer and >1 viewers. Figure 4-7 presents our findings as a stacked bar chart; buckets are plotted on the X-axis, and the fraction of broadcasts with zero views is plotted on the Y-axis. We also plot the number of samples in each bucket on the Y2-axis. Unsurprisingly we find on the Y2-axis that the number of broadcasters in each bucket decreases as the threshold for the number of broadcasts increases. Focusing on the Y1-axis, we find, as expected, that the heavy users who broadcast more have a lower proportion of unwatched streams. However, we observe that *all* user groups generate unwatched material. This extends to users who generate tens of broadcasts. This confirms that unwatched material is not simply created by experimental users who just try-out Facebook Live once. Instead, it is a persistent characteristic of broadcasters. For example, for users with ten broadcasts, we find that 29.56% of broadcasts gain 0 viewers, 7.9% gain one viewer and the remainder gain above one viewer. Interestingly, the graph also demonstrates a valley-like trend, whereby users with a small number of broadcasts largely gain 0 viewers (45.8% for below five broadcasts), whilst users with a high number of broadcasters also generate a lot of unwatched streams (36.35% for above 100 broadcasts). Users who fall in the middle of the spectrum (40–50 broadcasts) have the lowest proportion of unwatched material. Although this observation is impacted by the smaller sample sizes for user groups with large numbers of broadcasts (as shown on the Y2-axis), our manual inspection reveals that such "unpopular" users tend to generate large numbers of uninteresting material, *e.g.,* unprocessed personal diaries. It should also be noted that the generation of unwatched material is not exclusive

to mobile users; some categories of Page streams exhibit similar properties, *e.g.,* 40% of "Religious" streams from Page accounts are never viewed, and 13% for "Media" streams. Overall, 37.4% of Page streams have zero views.

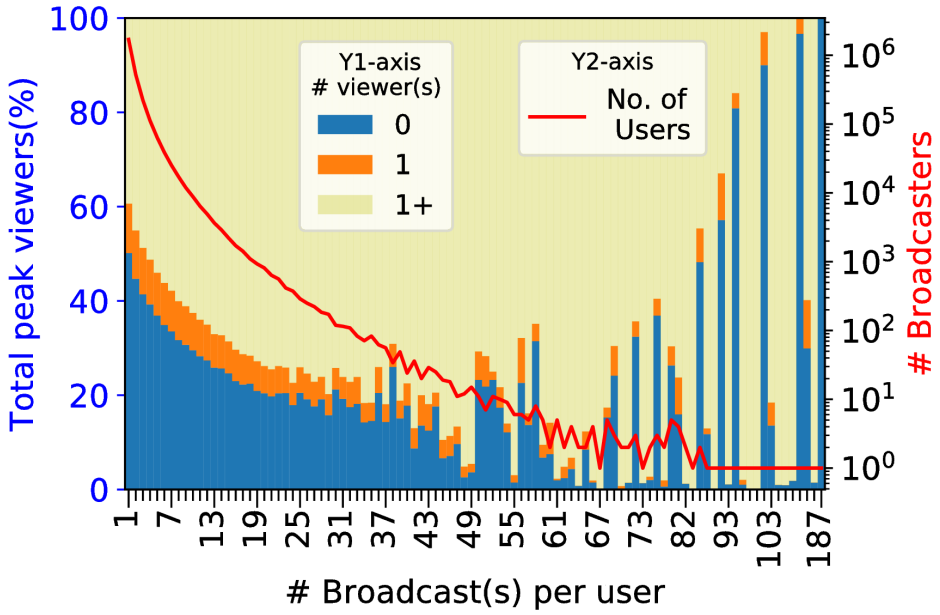


Figure 4-7: Distribution of the number of peak viewers (Y1-axis on the left) bucketed based on the broadcast(s) made by a user. Y2-axis (on the right) shows the total number of broadcasters falling under a bucket of total broadcasts performed.

Handle	Type	Viewers	Description	Category	Verified
@zuck	User	48471	Facebook founder	User	Yes
@GilmoreGirls	Page	39162	Popular TV drama	TV Programme	Yes
@buzzfeedtasty	Page	34369	Food news outlet	Media	Yes
@pena6789	User	16952	Adult content	User	NA
@WhiteHouse	Page	16124	Official WH page	Historical Landmark	Yes
@FoxNews	Page	14721	News outlet	Media	Yes
@bendeac	User	13289	Romanian Actor	User	No
@RealMadrid	Page	12783	Popular football team	Sports Team	Yes
@WesternJournalism	Page	10507	News outlet	Media	Yes
@BuzzFeed	Page	10276	News outlet	Media	Yes

Table 4-2: Top broadcasters based on median view count.

4.3.2 How long are broadcasts?

The previous section has highlighted that nearly half of all user broadcasts are never viewed. We next inspect the durations of these broadcasts to understand the resources consumed (both in terms of user time and upload volume). Figure 4-6 presents a CDF of the duration of broadcasts separated into a number of groups. App broadcasts are, by far, the longest with 53.46% exceeding half an hour. This is largely driven by the fact that most apps broadcast

computer gameplay. Broadcasts performed by Pages also tend to be substantially longer in duration than User broadcasts. Almost 30% of the Page broadcasts exceed half an hour, compared to just 6% for User broadcasts (where 77.34% are under 10 minutes).

This trend may be partly driven by the greater popularity of Page streams (*cf.* Figure 4-5a). To examine this more closely for mobile users, Figure 4-5a breaks User broadcasts into two groups: (*i*) **Popular**, which gain over 100 viewers at peak; and (*ii*) **Unpopular**, which are never viewed. In-line with our previous statement, we see that more popular streams broadcast for longer. 25% of these streams last longer than half an hour, whereas this is just 2% for unpopular streams. Critically, however, we still find a non-negligible number of unwatched streams that broadcast for extended periods (80k videos are longer than 20 minutes).

Currently, streams with zero views are live streamed to Facebook's infrastructure regardless of their popularity. This observation suggests significant waste in network, client and server resources. For example, the average duration for unwatched streams is five and a half minutes. With a bitrate of 500kbps, this would generate 18MB of needless traffic per broadcast (with battery and network costs). This, therefore, raises the question of how such content is delivered. Whereas uploads are obviously necessary for archived streams (*i.e.,* ones that remain on a user's profile), this is profligate for live streams that are not archived. This is because such data is immediately discarded without anybody ever seeing it. Importantly, we find that only 41.7% of unwatched streams are actually archived — for the remainder, it is unnecessary to waste device resources in performing the live upload to the PoP. This observation leads us to inspect the size of the archived videos (*i.e.,* videos marked by the users as being available for on-demand viewing later on) to understand the precise volume of this upload process. Figure 4-5b presents the distribution of file sizes for all archived videos. The median file size for broadcasts with zero views is 8.57 MB, with 45% exceeding 10 MB.

■ 4.3.3 Is broadcast really necessary?

Combining the above observations, we see that Facebook Live is predominantly (95%) used by mobile users to broadcast content. Pages garner a disproportionate number of viewers, whilst 41.47% of user broadcasts languish in obscurity, never being viewed. These observations indicate that the current "cloud-based" model of delivery is misplaced. As it stands, the Facebook mobile app uploads content regardless of view counts. This opens up interesting opportunities for hyper-local storage. For example, video content can be cached locally on the mobile device, and only uploaded once the first viewer arrives[26].

[26]To entice new viewers, a short GIF or video highlight of a few seconds can be shown, to indicate the ongoing video broadcast.

Upon the completion of a broadcast, users are given the option of archiving the content. Live content that is not watched and not archived can be cached locally and then discarded after broadcast. If the user wishes to archive the video, it can be automatically uploaded to Facebook when convenient (*e.g.,* via Wi-Fi). This simple change would reduce the use of battery and network for the 55.58% of broadcasters who (at least once) generate an unwatched video. It would also save an average of 5.84MB per upload. Note that this also covers broadcasters who generate *any* video chunks that are never watched. For example, 44.1% of streams in their first 2 minutes have zero viewers, out of which 7.8% continue to garner viewers later. Hence, such users need not upload the first chunks until a viewer arrives[27].

4.4 Geographical Exploration

The previous section has highlighted the nature of social broadcast, focussing on a large number of unwatched and unpopular streams. Next, we inspect the opposite end of the spectrum — highly popular streams. A unique aspect of Facebook Live is that it provides the locations of stream viewers for broadcasts with an audience of 100+. This allows us to understand the geographical relationship between broadcasters and viewers.

4.4.1 Where are the users?

We begin by inspecting the locations of users to understand where Facebook Live's userbase is. Figure 4-8 presents viewer (blue) and broadcaster (green) locations for any streams that garner above 100 viewers. This covers 45K broadcasts and 20.3 million viewers. The map shows a significant spread of broadcasters (green) with a focus on East Coast US, Brazil, Europe and South East Asia. It can be seen that most major conurbations are covered. A particularly nice feature of the dataset is that locations are based on mobile GPS-based geo-tags (rather than IP address). This gives us accurate geographical vantage into regions such as China, which typically access Facebook via VPN (thereby changing their reported IP geolocation). It can be seen that despite the blocking of Facebook, there are a number of users in China. In the rest of this section, we focus on the top 19 countries measured by broadcasters and consumers. These countries are obtained taking the union of the top 15 countries in terms of broadcasters and top 15 in terms of consumers.

Figure 4-9 presents the percentage of broadcasters and viewers based on each of the top 19 countries. A clear ranking can be observed; the US dominates both the number of broadcasters and viewers, followed by Thailand and Vietnam. The latter is particularly

[27] In the current Facebook live design, new viewers cannot rewind on a live stream; thus these first chunks need not be uploaded even after the first viewers arrive.

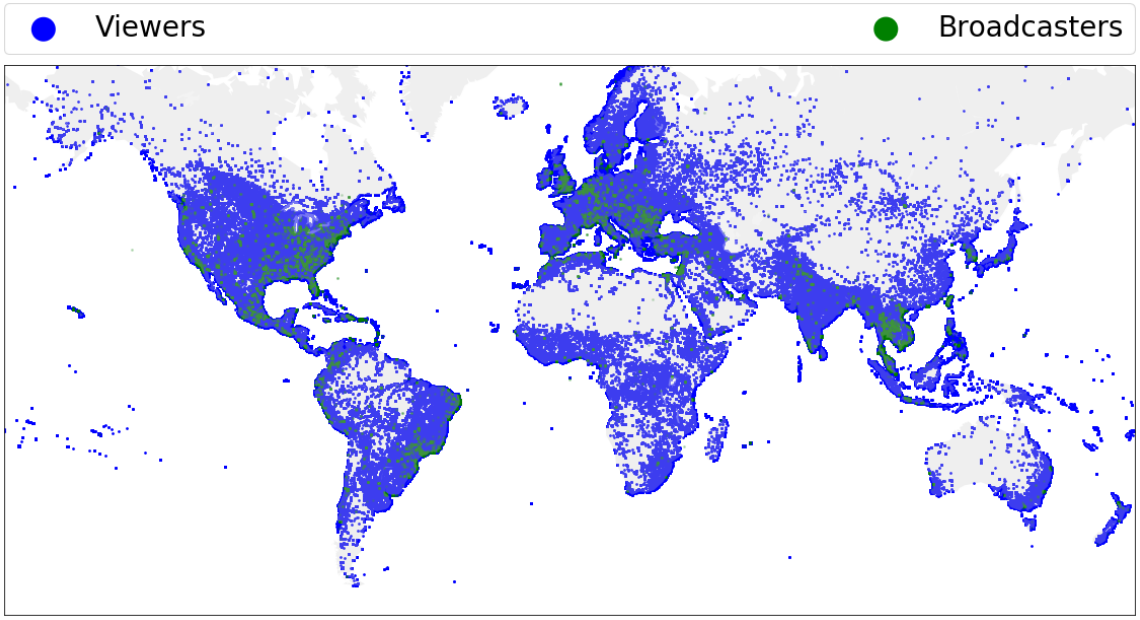


Figure 4-8: Locations of influential broadcasters (>100 viewers) and the respective viewers across the globe

interesting, as in many cases, we find that the number of viewers and broadcasters in a country do not necessarily correlate. For instance, whereas, in the US or Brazil, the fraction of viewers and broadcasters are relatively *similar*, Thailand generates a significant proportion of the world's broadcasts (11.26%), but only constitutes 8% of the viewers. Conversely, in Turkey, there are significant fraction of viewers (7.1% of the world's viewers), but only 2.94% of broadcasters. Similar observations can be made across many other countries, including Korea, Peru and Romania. Through manual inspection, we find a number of potential reasons. For instance, Turkey blocks access to Facebook; therefore, most users are likely accessing it via circumvention tools. Whereas Turkish users may be keen on consuming content, they may be less willing to expose themselves by broadcasting on a censored medium. In the case of Thailand, we find that a notable portion of adult content is being streamed, and this niche content attracts a globally distributed viewership (rather than just in Thailand).

4.4.2 How far away are viewers?

Although the above reveals the locations of broadcasters and viewers, it does not show their relationships. From a systems-perspective, this is critical for optimising delay-*in*tolerant communications as ideally viewers, broadcasters and any intermediate servers would be within short distances. Regarding the latter, there have been a number of recent proposals regarding the placement of servers within cell towers and local network point-of-presences,

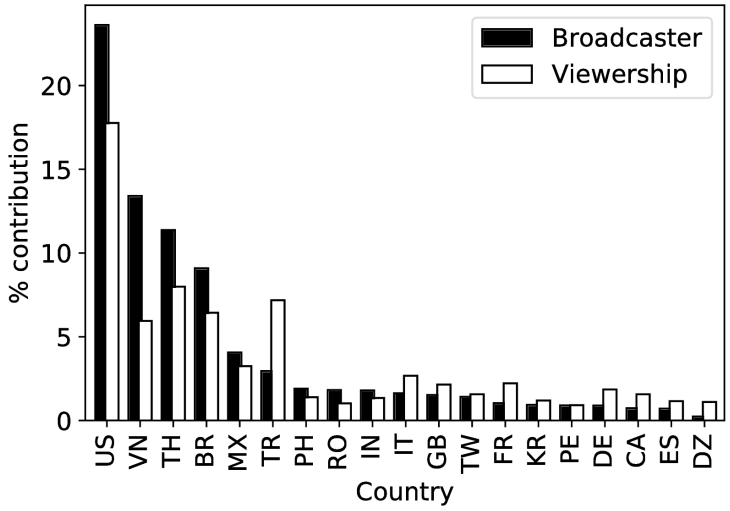


Figure 4-9: Contribution of broadcasters and viewers from top-accessed countries.

i.e., so-called mobile edge computing [196, 212, 213]. Thus, we next inspect the distance that viewers are from the broadcasters they watch. Figure 4-12a presents CDFs of the distance between the broadcasters and their viewers for user accounts. We separate broadcasters into their respective countries. In cases where the distribution plateaus, typically the broadcasters and viewers are separated by an ocean (*e.g.,* in the case of the Philippines, this is the distance across the Pacific to reach US-based viewers, who constitute 9% of the country's audience).

It can be seen that a notable subset of broadcasters are hyper-local to their consumers. This is shown well in Figure 4-10, which presents a heatmap of viewer locations relative to the broadcaster. Overall, 8% of viewers are within 25 KM of the broadcaster and constitute a city-level locality. South Korea is the most extreme in this regard; 11% of viewers are within 25KM, with 4% actually under 5KM. We can also inspect the opposite end of the spectrum. 11.35% of viewers are over 10,000 KM away from their broadcaster. Here, subtle differences can be seen between broadcasters from different countries. For example, broadcasters from the Philippines, a tiny island in the middle of Western Pacific, tend to be further away from their viewers (only 37.81% of viewers are closer than 1000 KM), whereas Turkish broadcasters generally garner more nearby viewers (52.71% are closer than 1000 KM). As previously discussed, this local content consumption suggests that live social streaming would be well placed to benefit from technologies such as mobile edge computing.

■ 4.4.3 Domestic or international?

It should be remembered that distance can be deceptive due to the various sizes of individual countries. Since we expect infrastructure to be confined within country borders, we next

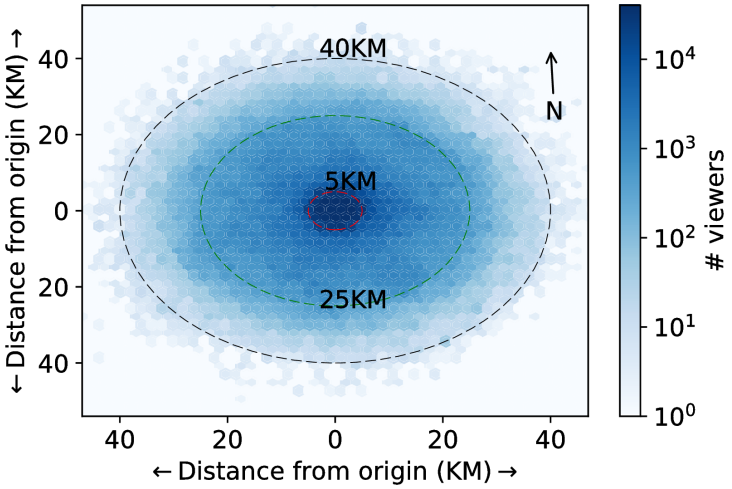


Figure 4-10: Number of viewers (binned at 500m) within a radius of 5KM, 25KM and 40KM when all the broadcasters location are shifted to the origin.

separate all broadcasters and viewers into their respective countries. Figure 4-12b presents the fraction of the views for videos of a country which come from domestic viewers. This is computed as follows: For a given country, we take the total view count of all videos broadcasted from that country. We then present the *fraction* of these views that are domestic, *i.e.,* from viewers in the same country as the broadcaster. In some cases (*e.g.,* Thailand, Vietnam), over 80% of views are domestic, suggesting strong language, social and cultural ties drive consumption. In other cases, under 20% of viewers are domestic (*e.g.,* Germany). This suggests such countries generate highly popular international content. However, we note that almost all videos have a non-trivial local viewership; and thus would benefit from caching locally in the country of broadcast, regardless of whether the video also has an international appeal or not.

To explore the international views, Figure 4-11 presents a heat map of the *non-domestic* views. Each cell in the matrix represents the fraction of views for videos of a country which come from a specific consumer country. For the sake of clarity, the fraction of domestic views (where the producer and consumer are from the same country) are not plotted. Further, only the top 19 producers and consumers are shown; thus, the fraction of the heat map values across all consumers might not add up to 1.

It can immediately be seen that a small number of countries stand-out in regards to the global reach of their content. The US is the most prominent, with a particularly large build-up of viewers in Mexico and Canada. This is not surprising considering the high degrees of immigration and cultural overlap between these three countries. Despite our prior observation about local consumption, Thailand and Vietnam also have a large

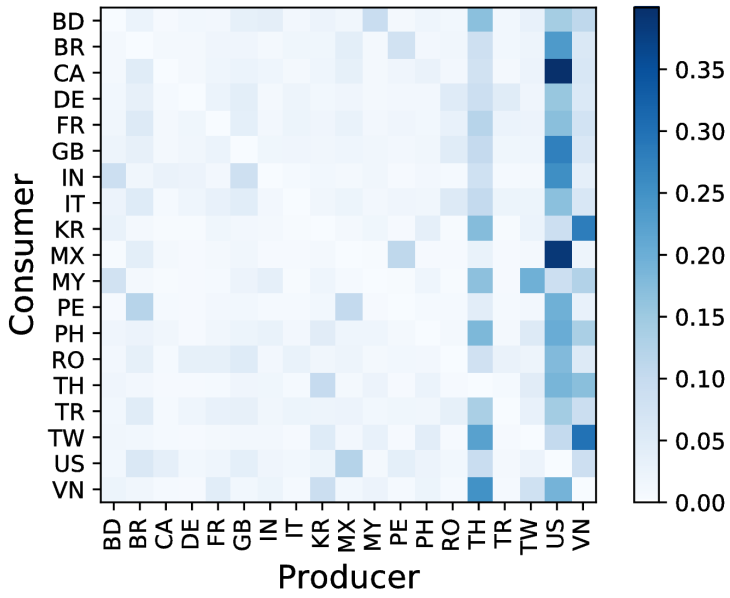


Figure 4-11: Fraction of views from users in a *different* country from the broadcaster, shown as a heat map of content broadcast from one country consumed in another country, plotted for the union of top 15 producers and consumers.

number of viewers from other countries. Much like the US, this is driven by a large number of broadcasts in the country.

Before concluding, we note that Figure 4-11 is intended to illustrate some interesting international trends and is not a complete picture – for instance, countries like Germany and Turkey, which have a high proportion of non-domestic views (Figure 4-12b) do not show up with high values in this heat map because the viewers for the video broadcasts from these countries are not amongst the top 15 consumers.

4.5 Understanding Engagement

Our final analysis briefly explores engagement factors. We explore the extent to which views and other social engagement measures such as numbers of likes, comments and shares evolve over the lifetime of a video, both during the live broadcast, as well as afterwards, and draw implications for mobile live casting. The main takeaway is that most of the engagement happens *after* the live broadcast, lending further support to the idea that Facebook Live can be treated similarly to an on-demand service, as opposed to a live broadcast.

4.5.1 Evolution of views over time

First, we capture the instantaneous number of viewers for *all* broadcasts. We do this until minute 34 (≈95% of videos end before 34 minutes). The number of viewers depends to

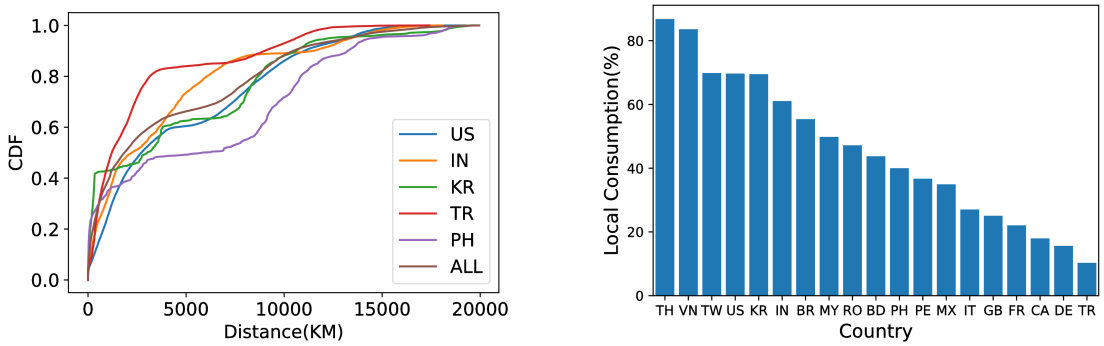


Figure 4-12: Geographical footprint: (a) CDF of distance between the broadcasters of their viewers. Multiple plots are presented for broadcasters in different countries (b) Fraction of views from users in the same country as the broadcaster.

some extent on the quality of the video, but also upon the duration – longer videos have a higher chance of catching friends who come online and notice the live broadcast. Yet, many of them may lose interest and leave after some time, so the later minutes of the video may lack sufficient viewers. To avoid introducing bias through the varying sample size across each time interval bucket, we randomly pick 100k videos for each time intervals and study the distribution of viewer counts.

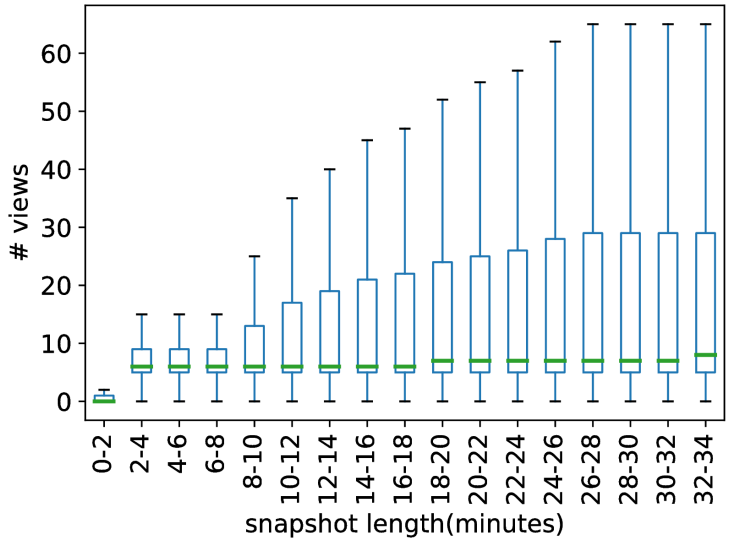


Figure 4-13: View evolution over two minute intervals

Figure 4-13 shows a box-and-whisker plot of the number of viewers in each interval across the videos. Each box extends from the lower to upper quartile values of the number of viewers, with a line at the median. The whiskers extend from the box to show the range of the data. Flier points past the end of the whiskers are not shown for clarity. The

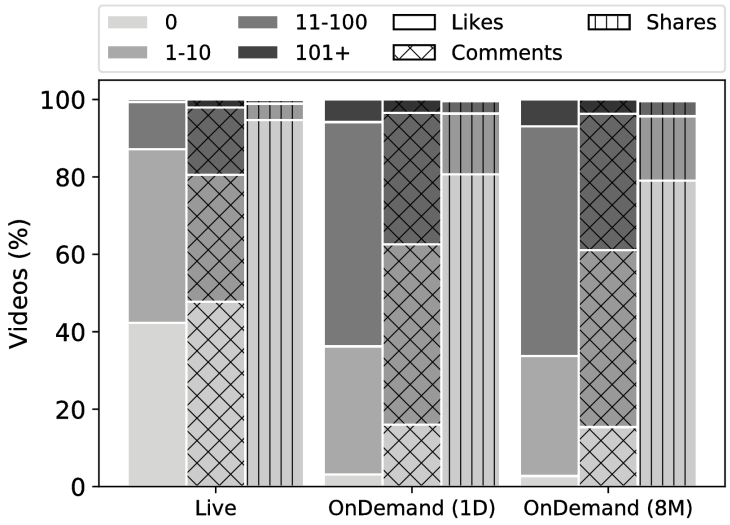


Figure 4-14: Distribution of engagement metrics (shares, likes, comments) across all broadcast streams.

viewer counts increase after the first interval, and there is a larger variance in the later time buckets, indicating that sufficiently interesting longer videos accumulate viewers over time. But the median number of viewers remains comfortably under 10, even after 20–30 minutes of the broadcast, again underscoring that it is not very difficult to support most live broadcasts.

■ 4.5.2 Social engagement

A key feature of Facebook Live is the ability for viewers to interact with broadcasters. This comes in the form of likes, shares and comments. In a sense, this offers an alternative measure of popularity when compared to the earlier discussion of viewer counts.

Figure 4-14 presents the distribution of these engagement metrics for all broadcasts. The figure presents these metrics across three time windows: (*i*) During the live broadcast; (*ii*) a day later for videos that have been archived on-demand; and (*iii*) eight months later. The pattern indicates the type of engagement, whilst the grayscale colour separates broadcasts into four popularity groups – those that get 0 engagement (likes/comments/shares), 1–10 engagements, 11–100 engagements or more than 100 engagements.

Most of the engagement happens *after* the live broadcast. It is therefore questionable to what extent this is a *live* platform. On average, during the live broadcast, videos receive 6.7 likes, 8.4 comments and 0.54 shares. One day after broadcast, the engagement counts jump to 29.84 likes, 16.33 comments, and 1.33 shares. These numbers do not change much in the next eight months; thus, the majority of interaction happens in the one day after broadcast. Notice that about 40% (47%) of videos collect no likes (comments) during the

live broadcast, but this number drops to 3% (16%) after a day. From an infrastructure design perspective, this interaction pattern makes it easier to support social engagement, since on-demand is easier to manager than live streaming, which requires synchronisation of the display of likes and comments across all viewers at the point in the stream where the interaction happened.

We note that during the live broadcasts, comments tend to outnumber likes, but the engagement after the broadcast tends to be through likes. Comments represent a much stronger form of engagement; thus we conjecture that users who watch and interact with the live stream may be close friends of the user, whereas the later engagement could be from other friends or the general public.

Thus, in this chapter, we explored the usage of Facebook Live in terms of three major factors: video characteristics; the geography of broadcasters and viewers; and the social engagement.

Chapter 5
Decentralised web and Content Replication

There is a recent growth in applications where both application hosting and content are managed by amateurs. Even though there involve a lot of problems as everything, from reliability to content moderation, is supposed to be operated by users, these systems have the main benefit of preserving user privacy. In this chapter, we explore one such application called Mastodon that belongs to category A3 and enables socialising in a decentralised fashion. We pinpoint the impact of having a random content replication scheme instead of present mechanism to ensure the resilience of the content system.

5.1 Introduction

The "Decentralised Web" (DW) is an evolving concept, which encompasses technologies broadly aimed at providing greater transparency, openness, and democracy on the web [214]. Today, well known social DW platforms include Mastodon (a microblogging service), Diaspora (a social network), Hubzilla (a cyberlocker), and PeerTube (a video sharing platform). Some of these services offer decentralised equivalents to web giants like Twitter and Facebook, mostly through the introduction of two key innovations.

First, they decompose their service offerings into independent servers ("instances") that anybody can easily bootstrap. In the simplest case, these instances allow users to register and interact with each other locally (*e.g.,* sharing videos), but they also allow cross-instance interaction via the second innovation, *i.e., federation.* This involves building on decentralised protocols to let instances interact and aggregate their users to offer a globally integrated service. Centralisation leads to a single entity or a small set of entities to own the

data and technology with decision-making abilities moving to the centre of an organisation. Thus centralised ownership and administration holds a single point of control and innately lacks resilience to failures.

DW platforms intend to overcome these problems and offer a number of benefits. For example, data is spread among many independent instances, thus possibly making privacy-intrusive data mining more difficult. Data ownership is more transparent, and the lack of centralisation could make the overall system more robust against technical, legal or regulatory attacks. However, these properties may also bring inherent challenges that are difficult to avoid, particularly when considering the natural pressures towards centralisation in both social [215, 216] and economic [217] systems. For example, it is unclear how such systems can securely scale-up, how wide-area malicious activity might be detected (*e.g.,* spam bots), or how users can be protected from data loss during instance outages/failures.

As the largest and most popular DW application [218, 219], we choose *Mastodon* as a relevant example to study some of these challenges in-the-wild. Mastodon is a decentralised microblogging platform, with features similar to Twitter. Anybody can set up an independent instance by installing the necessary software on a server. Once an instance has been created, users can sign up and begin posting "*toots,*" which are shared with followers. Via federation, they can also follow accounts registered with other remote instances. Unlike traditional social networks, this creates an inter-domain (federated) model, not that dissimilar to the inter-domain model of the email system.

In this chapter, we present a large-scale (case) study of the DW aiming to understand the feasibility and challenges of running decentralised social web systems. We use a 15-month dataset covering Mastodon's instance-level and user-level activity, covering 67 million toots. Our analysis is performed across two key axes: (*i*) We explore the deployment and nature of *instances*, and how the uncoordinated nature of instance administrators drive system behaviours (Section 4.3); and (*ii*) We measure how *federation* impacts these properties and introduces unstudied availability challenges (Section 5.5). A common theme across our findings is the discovery of various pressures that drive greater centralisation; we therefore also explore techniques that could reduce this propensity.

Main Findings. Overall, our main findings include:

1. *Mastodon enjoys active participation from both administrators and users.* There is a wide range of instance types, with tech and gaming communities being quite prominent. Certain topics (*e.g.,* journalism) are covered by many instances, yet have few users. In contrast, other topics (*e.g.,* adult material) have a small number of instances but a large number of users.

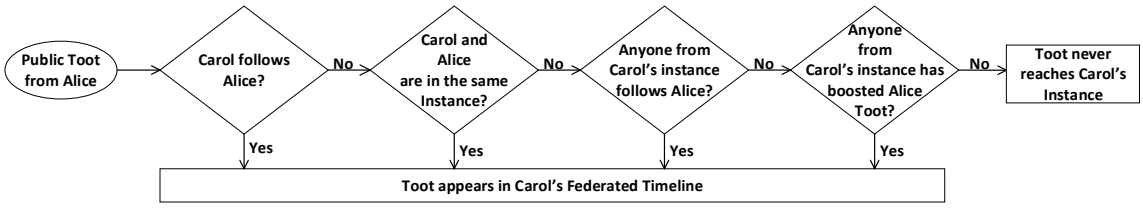


Figure 5-1: Flowchart explaining when a toot is placed on the federated timeline of a user.

2. *There are **user-driven** pressures towards centralisation.* Popularity in Mastodon is heavily skewed towards a few instances, driving implicit forms of centralisation. 10% of instances host almost half of the users. This means that a small subset of administrators has a disproportionate impact on the federated system.
3. *There are **infrastructure-driven** pressures towards centralisation.* Due to the simplicity and low costs, there is a notable location of instances within small set of hosting providers. We find that failures in these ASes can create a ripple effect that fragments the wider federated graph. We observe 6 cases of these AS-wide outages within our measurement period. We also observe regular outages by individual instances (likely due to the voluntary nature of many administrators). Again, this has a notable impact: 11% of instances are unavailable for half of our measurement period.
4. *There are **content-driven** pressures towards centralisation.* Due to the differing popularities of toots, we find that outages in just 10 instances could remove 62.69% of global toots. To ameliorate this problem, we explore the potential of building toot replication schemes. For example, when a user has followers from different instances, the content posted by the user could be stored and indexed (persistently) in the followers' instances. By enabling this type of federation-based replication, availability improves so that only 11.4% of toots are lost when the top 3 ASes are offline (rather than 70% without).

■ 5.2 Dataset

In this section, we describe the basic operation of Mastodon, highlighting key terminology in bold. We refer readers interested in more details to a tutorial on The Verge [220].

What is Mastodon? Mastodon is an open-source DW server platform released in 2016 [221]. It offers microblogging functionality, allowing administrators to create their own independent Mastodon servers, aka **instances**. Each unique Mastodon instance works much like Twitter, allowing users to register new accounts and post **toots** to their followers. Users can also **boost** toots, which is the equivalent of retweeting on Twitter.

Instances can work in isolation, only allowing locally registered users to follow each other. However, Mastodon instances can also **federate**, whereby users registered on one instance can follow users registered on another instance. This is mediated via the local instances of the two users. Hence, each Mastodon instance maintains a list of all remote accounts its users follow; this results in the instance **subscribing** to posts performed on the remote instance, such that they can be pulled and presented to local users. For simplicity, we refer to users registered on the same instance as **local**, and users registered on different instances as **remote**. Note that a user registered on their local instance does *not* need to register with the remote instance to follow the remote user. Instead, a user just creates a single account with their local instance; when the user wants to follow a user on a remote instance, the user's local instance performs the subscription on the user's behalf. This process is implemented using an underlying subscription protocol. To date, Mastodon supports two open protocols: oStatus [170] and ActivityPub [172] (starting from v1.6). This makes Mastodon compatible with other decentralised microblogging implementations (notably, Pleroma).

When a user logs in to their local instance, they are presented with three timelines: (*i*) a *home* timeline, with toots posted by the accounts whom the user follows; (*ii*) a *local* timeline, with all the toots generated within the same instance; and (*iii*) a *federated* timeline, with *all* toots that have been retrieved from remote instances. Figure 5-1 explains when a toot is captured in the federated timeline. The latter is not limited to remote toots that the user follows; rather, it is the union of remote toots retrieved by all users on the instance. The federated timeline is an innovation driven by Mastodon's decentralised nature; it allows users to observe and discover toots by remote users, and broaden their follower network.

Our goal is to better understand the nature of *instances* and *federation*, using Mastodon as a case study. To do so, we rely on three primary datasets: (*i*) *Instances*: regular snapshots of instance metadata and availability; (*ii*) *Toots*: historical user posts (toots) available on each instance; and (*iii*) *Graphs*: the follower and federation graphs.

Instances. We first extracted a global list of instance URLs and availability statistics from a dump provided by the `mnm.social` website. This site contains a comprehensive index of instances around the world, allowing us to compile a set of 4,328 unique instances (identified by their domain). These instances primarily run the Mastodon server software, although 3.1% run the Pleroma software (`https://pleroma.social/`) instead. This is because, since 2017, these two implementations have federated together using the same front-end and federation protocol (ActivityPub). Hence, from a user's perspective, there is little difference between using Mastodon or Pleroma instances.

We obtain our data by using the monitoring service of `mnm.social`. Every five minutes, `mnm.social` connected to each instance's `<instance.name>/api/v1/instance` API endpoint. The instance API returned the following information from each instance: name, version, number of toots, users, federated subscriptions, and user logins; whether registration is open and if the instance is online.

The data on `mnm.social` goes back to April 11, 2017. We collected all the data until July 27, 2018. Maxmind was then used to map the IP address of each instance to their country and hosted AS. Although some of these metadata items rarely change (*e.g.,* country), repeatedly fetching all the data every five minutes gives us fine-grained temporal data revealing how values evolved across time. Overall, we observe approximately half a billion data points.

Toots. In May 2018, we crawled all available toots across the instances. To compile a list of instances to crawl, we started with the list collected via the `mnm.social` website. We then filtered these to only leave instances that were online during May 2018: this left 1.75K active instances which were accessible. This obviously reveals a degree of churn in the instance population; across the 15 month measurement cycle, 21.3% of instance went offline and never came back online. We wrote a multi-threaded crawler[28] to connect with each of these 1.75K instances, via their API, and collected their entire history of toots. To expedite the process, we parallelised this across 10 threads on 7 machines. Each thread was responsible for querying the federated timeline of its designated instance; it did this by iterating over the entire history of toots on the instance. To avoid overwhelming instances, we introduced artificial delays between API calls to limit any effects on the instance operations. For each toot, the following data were collected: username, toot URL, creation date, media attachments, number of favourites, followers, and followings, toot contents, and hashtags.

Our toots dataset contains 67M public toots, generated by 239K unique users. By comparing this against the publicly listed metadata obtained from the *instance* dataset (recall this contained toot and user counts), we find that our dataset covers 62% of the entire toot population. The remaining 38% of toots could not be collected: approximately 20% of these toots were set to private, and the remainder were hosted on instances that blocked toot crawling.

Complementary Datasets.

Follower and Federation Graphs. We also crawled the followers and following lists for users in July 2018. To this end, we scraped the follower relationships for the 239K users we encountered who have tooted at least once. This was performed be iterating over all

[28]the code to collect data from Mastodon can be found here: `https://rebrand.ly/dweb_crawl`

public users on each instance, and simply paging through their (HTML) follower list.[29] This provided us with the ego networks for each user. We identified a user by their account name on a per-instance basis, as users must create one account per instance. 87.9% of account names are unique to a single instance, while 8.3% are on two instances, and 3.8% are on three or more. Note that it is impossible to infer if such accounts are owned by the same person, and therefore we treat them as separate nodes.

We then induced a graph, $G\big(V, E\big)$, in which user V_i has a directed link to another user V_j if V_i follows V_j. This resulted in 853K user accounts and 9.25M follower links. Conceptually, this graph is similar to the Twitter follower graph [181]. However, unlike Twitter, users can follow accounts on *remote* instances, triggering the need for remote subscriptions between instances (*i.e.,* federation). Hence, the creation of a link in E can trigger an underlying federated subscription between the respective instances. We, therefore, created a second graph, $G_F\big(I, E\big)$, which consists of instance-to-instance subscriptions (hereafter instance *federation graph*, for short). $G_F\big(I, E\big)$ is induced by $G\big(V, E\big)$; a directed edge E_{ab} exists between instances I_a and I_b if there is at least one user on I_a who follows a user on I_b.

Twitter. For comparisons, we also gathered data about outages in Twitter as well as its social network, dating back to when Twitter was at a similar age as Mastodon is now. while we did not get access to the number of users accessing Twitter at that time, we collected the uptime statistics when Twitter was roughly 1.5 years old. For this, we used `pingdom.com`, a widely used website monitoring platform which probed Twitter every minute between February and December 2007. The data is obtained from the Internet Archive [222]. As a baseline social graph, we obtained the Twitter social graph from a 2011 dataset [223], which consists of users and their respective follower networks.

Limitations. Our analysis reveals a highly dynamic system and, thus, our measurements are capturing a "moving target." Similarly, we highlight that the timelines of each dataset differ slightly due to the progressive deployment of our measurement infrastructure. This, however, does not impact our subsequent analysis. It is also worth noting that we do not have comprehensive data on all toots, due to some instances preventing crawling (we have 62% of toots). Thus, we have avoided performing deeper semantic analysis on toot content and, instead, focus on instance-level activities.

■ 5.3 Characterising Mastodon

The first innovation that the DW introduces is the distribution of services across independent *instances*. This means that these platforms emerge in a bottom-up fashion from the thousands of instances running their software. Hence, inspecting them provides a window

[29] `https://<instance.name>/users/<user.name>/followers`

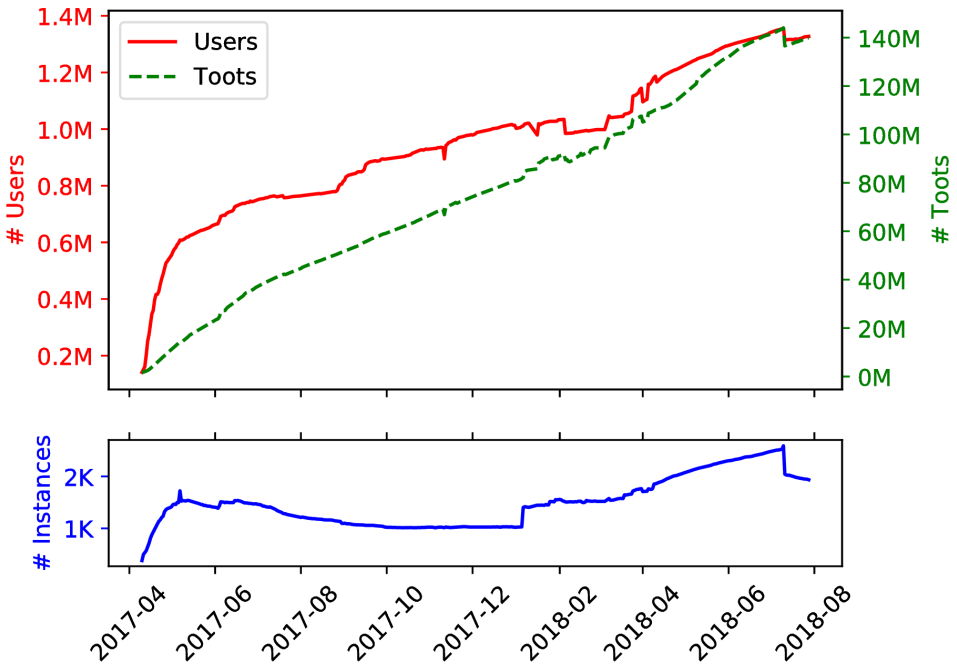


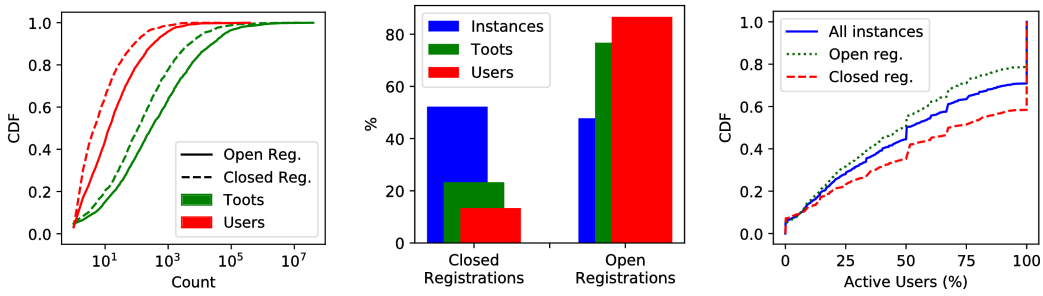
Figure 5-2: Available instances/user/toots over-time.

into how bottom-up DW decisions are made, and the implications they might have on the overall system. In this section, we perform a characterisation of the instances available in Mastodon and how they are deployed.

First, as mastodon is comprised of instances, we quantify the growth and popularity of the 4,328 instances under study, and inspect how this relates to instance settings.

Instance Growth. We begin by briefly inspecting the recent growth of Mastodon. Figure 5-2 plots the timeseries of the number of instances, users, and toots available per day. Mastodon is in a phase of growing popularity, with fluctuations driven by the arrival and departure of instances. Between April and June 2017, there was an increase in the number of instances (and users). However, while the number of instances reaches a plateau around July 2017 (only 6% of the instances were set up between July and December 2017), the user population continues to grow during this period (by 22%). Then, in the first half of 2018, new instances start to appear again (43% growth). We conjecture that this may have been driven by things like the #DeleteFacebook campaign (popular at that time) [224], and sporadic bursts of media attention [214, 220, 225]. Closer inspection also reveals fluctuations in the instance population; this churn is driven by short periods of unavailability, where certain instances go offline (temporarily). We posit that this may have wider implications, and therefore deep dive into this in Section 5.4.2.

Open *vs.* Closed Instances. Mastodon instances can be separated into two groups: (*i*) *open*, allowing anybody to register (47.8% of instances); and (*ii*) closed, requiring an



(a) (b) (c)

Figure 5-3: Dissecting Instances with open and closed (invite-only) registrations. (*a*) Distribution of number of toots and users per-instance (*b*) Number of instances, toots and users for open and closed registrations; (*c*) Distribution of active users (max percentage of users logged-in in a week per instance) across all instances.

explicit invitation from the administrator (52.2%). Figure 5-3a presents a CDF of the number of users and toots per-instance, separated into open and closed instances. Overall, the user population distribution is highly skewed, with distinct traits of natural centralisation. For both kinds of instances (open and closed), the top 5% of all instances have 90.6% of all users. Similar patterns can be seen among toot generation, with the top 5% of instances accumulating 94.8% of toots. This, of course, confirms that, despite its decentralisation, Mastodon does not escape the power law trends observed in other social platforms [180–182, 226].

Unsurprisingly, instances with an open registration process have substantially more users (mean 613 *vs.* 87). Inspecting user count alone, however, can be quite misleading, as open instances may accumulate more experimental (and disengaged) users. Figure 5-3b presents the breakdown of users, instances and toots across open and closed instances. Whereas the majority of users are associated with open instances and they create on average 94.8 toots per-person. In contrast, there are fewer users on closed instances, but they generate more toots per-capita (average of 186.65). To measure the activity level of users, each week, we compute the percentage of users who have actually logged into each instance (available in the *instance* dataset) and take the maximum percentage as the activity level of the instance. Figure 5-3c presents the CDFs of the activity level per instance. This confirms that closed instances have more engaged populations: the median percentage of active users per closed instance is 75%, compared to 50% for active users in open instances. Despite this mix of instances, it is clear that a notable fraction of Mastodon users regularly login; for example, 13.73% of users use Mastodon over once per-week. This disparity in usage patterns, which is typical in any web system, does increase the "dominance" of the instances that are associated with more active users. The growth of users on one side in

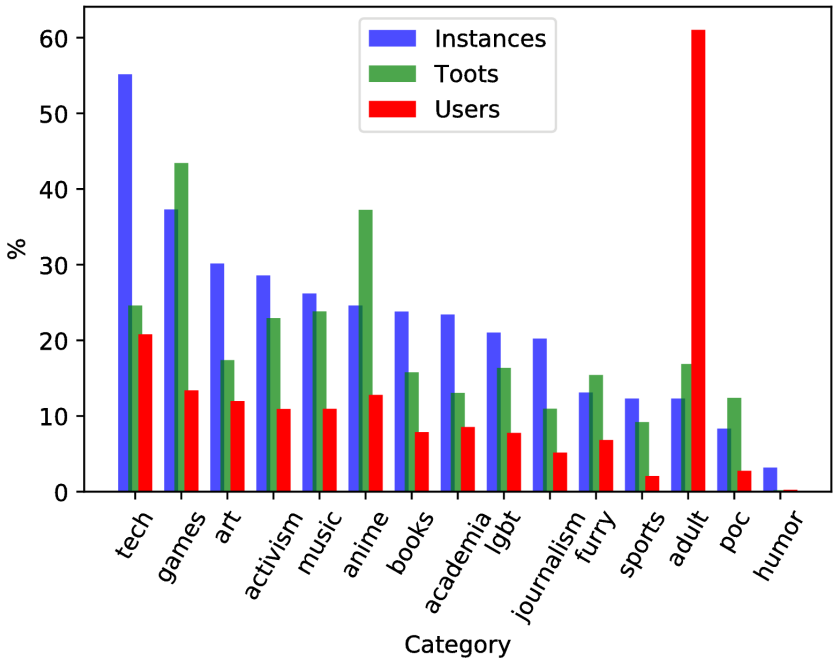


Figure 5-4: Distribution of the number of instances, toots and users across various categories.

open instances, and the increased usage in closed instances merely indicate the power law trend within the platform [227].

5.3.1 Instance Categories

To improve visibility and/or to specialise in certain topics, instances can explicitly "tag" themselves with a category, as well as the activities they allow. Thus, these tags help in understanding the types of ongoing activities within Mastodon.

Category Tags. Overall, 697 out of the 4,328 instances report a self-declared category, taken from a controlled taxonomy of tags. Just 13.6% of users and 14.4% of toots are associated with these categorised instances. 51.7% of these categories are labelled as "generic". Despite this relatively low coverage, it is still useful to inspect these tags to gain insight into ongoing usage. Note that the following statistics pertain to this subset of instances. Figure 5-4 plots the distribution of instances, toots, and users across each category. We identify 15 categories of instances. The majority of instances are categorised as tech (55.2%), games (37.3%) or art (30.15%). This is not entirely surprising, as Mastodon emerged from the tech community.

Figure 5-4 also allows us to compare the popularity of each category when measured by the number of instance *vs.* number of users, shedding light on the interests of the administrators *vs.* the general user population. Overall, the interests of these two groups

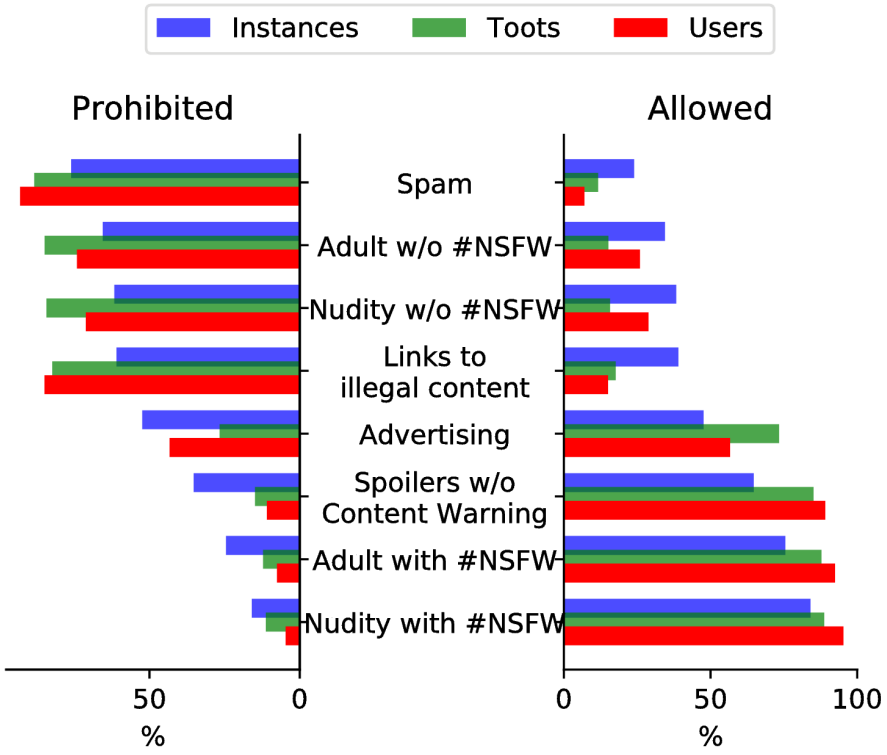


Figure 5-5: Distribution of instances and users across instances w.r.t. prohibited and allowed categories.

coincide, albeit with some key discrepancies. For instance, while tech covers 55.2% of instances, it only garners 20.8% of users and 24.5% of toots. By contrast, games correspond to 37.3% of instances, yet generate 43.43% of all toots, suggesting they are highly active. Similar observations apply to Anime instances, where 24.6% of instances contribute 37.23% of global toots. There is, however, one clear outlier: adult instances constitute only 12.3% of instances but attract 61.03% of all users; that is, adult content is clearly a topic attracting a large user population, which is served by a small set of instances. This has been observed in other forms of online adult content, where websites with limited content sets gain large per-item interest [152]. These contrasts suggest that Mastodon is a powerful tool for exploring and evaluating the different behaviours of these communities.

Activity Tags. Due to its decentralised nature, it is also possible for individual administrators to stipulate what types of *activity* are allowed in their instances. This is necessary on a federated social network, as each instance can have different policy requirements. Out of the 697 instances, 17.5% allow *all* types of activity. The remaining instances explicitly specify a subset of activities acceptable on the instance. 82% of these list at least one prohibited activity, whereas 93% explicitly specify at least one acceptable activity.

Figure 5-5 reports the number of instances, users, and toots associated with each activity category. The most regularly prohibited activity (left bar chart in the figure) is spam. 76% of instances disallow it, followed by pornography (66%), and nudity (62%). These two activities are typically only prohibited when not tagged with #NSFW (not safe for work): for instances that allow #NSFW, the vast majority also allow nudity (84.3%) and pornography (75.6%). On the other hand, and quite curiously, some instances explicitly allow several of these activities (see the right bar chart in Figure 5-5), *e.g.,* 24% of instances allow spam. Unsurprisingly, these get few users: even though they make up 21% of instances, they only hold 16% of users. In contrast, instances allowing advertising have disproportionately large user groups (47% of instances, but 61% of users hosting and 75% of the toots).

Mastodon UI also has a "content warning" (CW) checkbox for the posters to give advance notice that there are spoilers in the content. Interestingly, while 35% instances prohibit posting spoilers without a content warning, the remaining 65% explicitly allow this.

5.4 Exploring Instances

5.4.1 Instance Hosting

Unlike a centrally administered deployment, where the presence can be intelligently selected, Mastodon's infrastructure follows a bottom-up approach, where administrators independently decide where they place their instance. Figure 5-6 presents a breakdown of the presence of instances, toots, and users across countries and Autonomous Systems (ASes).

Countries. Japan dominates in terms of the number of instances, users and toots. In total, it hosts 25.5% of all instances, closely followed by the US, which hosts 21.4%. Closer inspection reveals that the ratio between the number of instances and number of users differ across countries though. For example, Japan hosts a just quarter of instances, yet gains 41% of all users; in contrast, France hosts 16% of instances, yet accumulates only 9.2% of users. It is also worth noting that these countries are heavily interconnected, as instances must *federate* together, *i.e.,* users on one instance may follow users on another instance (thereby creating federated subscription links between them, see Section 5.2). To capture their interdependency, Figure 5-7 presents a Sankey diagram; along the left axis are the top countries hosting instances, and the graph depicts the fraction of their federated subscriptions *to* instances hosted in other countries (right axis). Unsurprisingly, the instances exhibit homophily: users of an instance follow other users on instances in the same country, *e.g.,* 32% of federated links are with instances in the same country. The top 5 countries attract 93.66% of all federated subscription links. We posit that this dependency on a small number of countries may undermine the initial motivation for the DW, as large volumes of

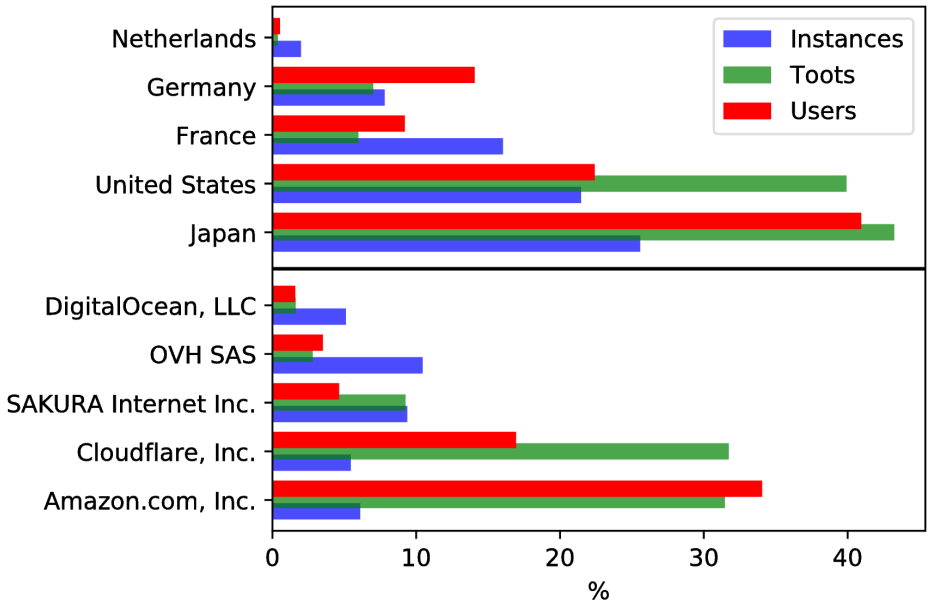


Figure 5-6: Distribution of instances, users, and toots across the top-5 countries (top) and ASes (bottom).

data are situated within just a small number of jurisdictions; *e.g.,* 89.1% of all toots reside on instances in Japan, the US, and France.

ASes. Next, we inspect the distribution of instances across ASes; this is important as an over-dependence on a single AS, may raise questions related to data pooling or even system resilience. When looking at the ASes that host Mastodon servers (bottom of Figure 5-6), we observe instances spread across 351 ASes. On average, each AS, therefore, hosts 10 instances. This suggests a high degree of distribution, without an overt dependence on a single AS.

That said, due to the varying popularity of these instances, the top three ASes account for almost two thirds (62%) of all global users, with the largest one (Amazon) hosting more than 30% of all users—even though it only is used by 6% of instances. Cloudflare also plays a prominent role, with 31.7% of toots across 5.4% of instances. The reasons for this co-location are obvious: Administrators are naturally attracted to well known and low cost providers. Whereas a centrally orchestrated deployment might replicate across multiple redundant ASes (as often seen with CDNs), this is more difficult in the DW context because each instance is independently managed (without coordination). Although these infrastructures are robust, the failure (or change in policy) of a small set of ASes could, therefore, impact a significant fraction of users. Again, this highlights another form of tacit centralisation that emerges naturally within such deployments.

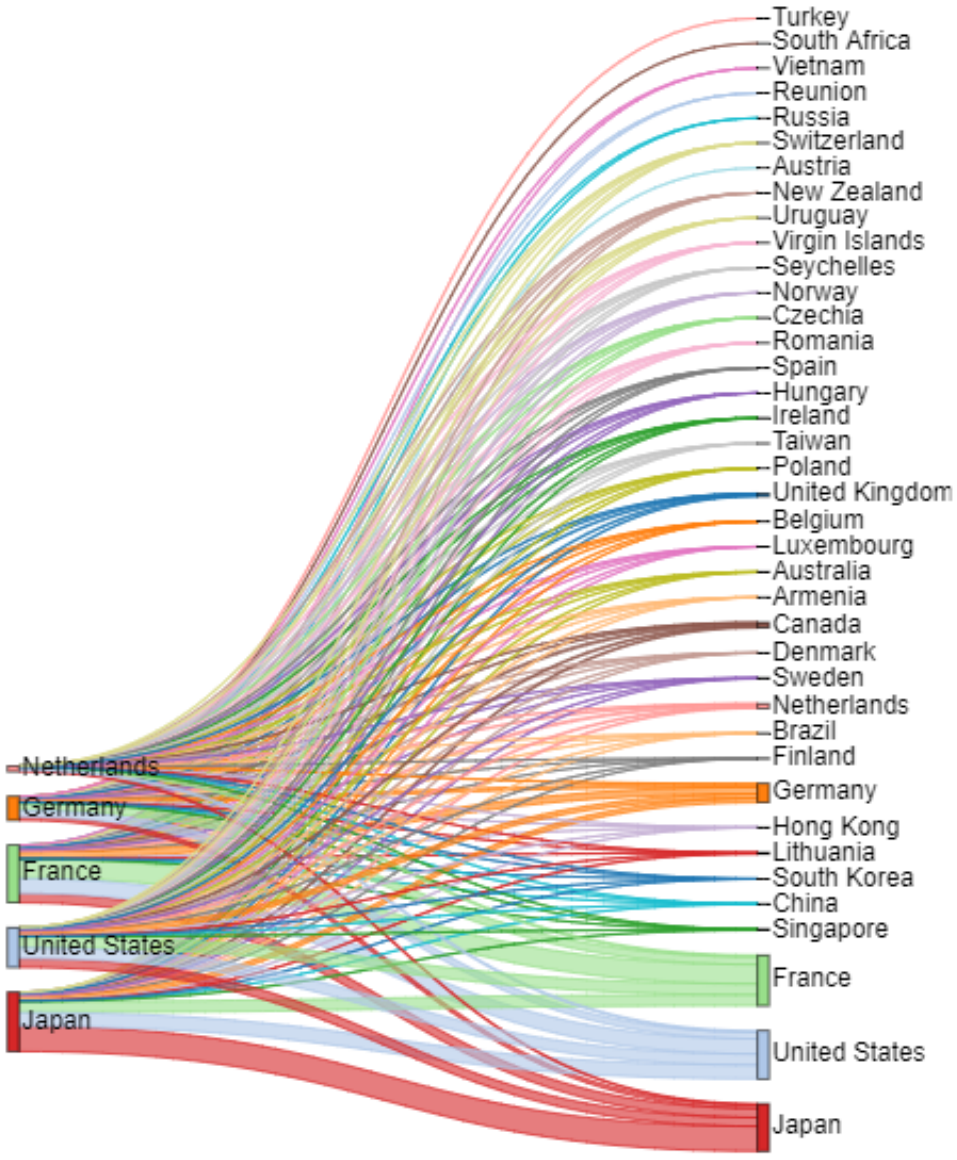


Figure 5-7: Distribution of federated subscription links between countries. The left axis lists the top-5 countries, and the lower access indicates the fraction of the federated links to instances in other countries.

■ 5.4.2 Instance Availability

We next explore the availability properties emerging from the bottom-up deployment. Here, we define *availability* as the ability to access and download its homepage. We posit that the uncoordinated and (semi-)voluntary nature of some instance operations may result in unusual availability properties.

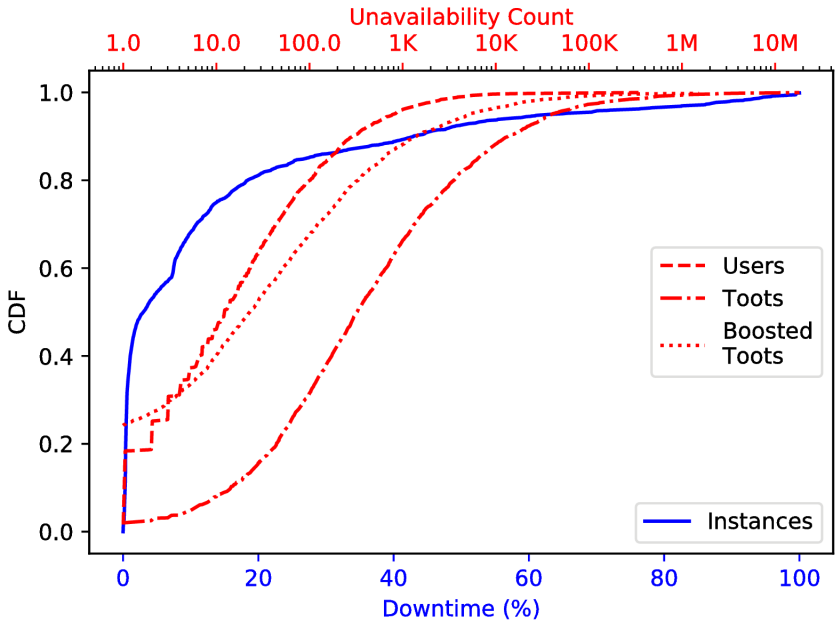


Figure 5-8: CDF of the instance downtime (bottom x-axis) and distribution of unavailable users, toots, and boosted toots (top x-axis) when instances go down.

Instance Availability. We start by measuring the historical availability of our 4.3K instances over time (from the *instance* dataset). We only count outages where an instance becomes unavailable, and then returns again within our 15 month measurement cycle (*i.e.,* we do not consider persistently failed instances as outages). Figure 5-8 plots the distribution of downtime of each instance over a 5-minute granularity (blue line, bottom X-axis). A sizeable portion of instances *do* have relatively good availability properties – about half of the instances have less than 5% downtime. 4.5% of instances were even up for 99.5% of the time (these popular instances covered 44% of toots). However, there is a long tail of extremely poor availabilities: 11% of instances are inaccessible more than half of the time, which confirms that failures are relatively commonplace in Mastodon. This has obvious repercussions for both toot availability and the ability of users to access the platform.

We cannot confirm the exact reasons for unavailability, nor can we state how reliable instances maintain their long uptime (although the operators might be employing load balancers and replicated back-ends). That said, it is likely that the voluntary nature of many instance operators has a role, *i.e.,* instances with poor availability might simply be unpopular and lack dedicated administrators. To test this, we count the number of toots and users that are unavailable during instance failures, see Figure 5-8 (red lines, top x-axis). Instances may go down multiple times, so we select the 95^{th} percentile of these values for each instance. Disproving our hypothesis, we find that failures happen on instances across the entire spectrum of popularity — there are a number of instances that host in excess of 100K toots which have outages.

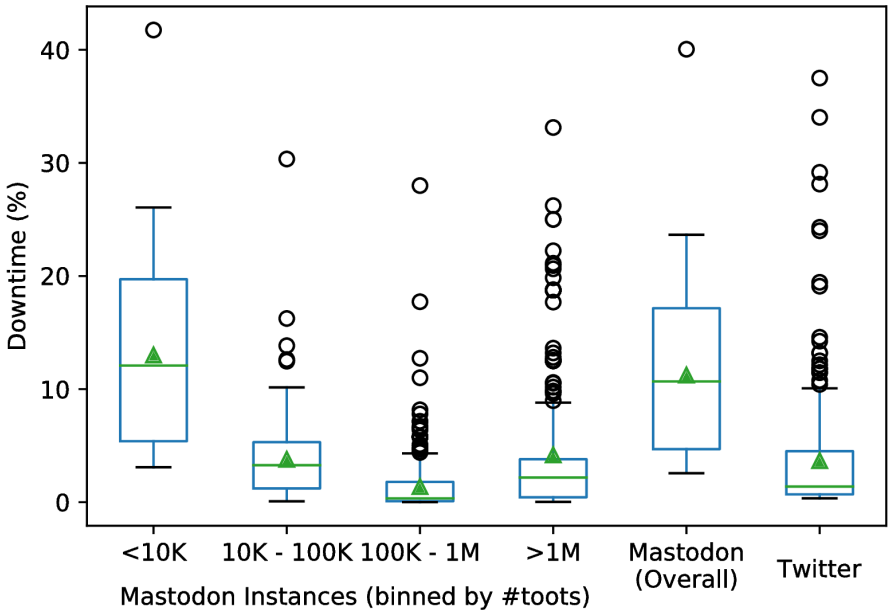


Figure 5-9: Distribution of per-day downtime (measured every five minutes) of Mastodon instances (binned by the number of toots), and Twitter (Feb–Dec 2007).

This is further explored in Figure 5-9, which presents a box plot of daily downtime for Mastodon, where we separate instances based on their number of toots. Although small instances (<10K toots) clearly have the most downtime (median 12%), those with over 1M toots actually have worse availability than instances with between 100K and 1M (2.1% *vs.* 0.34% median downtime). In fact, the correlation between the number of toots on an instance and its downtime is -0.04, *i.e.,* instance popularity is not a good predictor of availability. The figure also includes Twitter's downtime in 2007 for comparison (see Section 5.2). Although we see a number of outliers, even Twitter, which was famous for its poor availability (the "Fail Whale" [228]), had better availability compared to Mastodon: its average downtime was just 1.25% *vs.* 10.95% for Mastodon instances.

Certificate Dependencies. Another possible reason for failures is third party dependencies, *e.g.,* TLS certificate problems (Mastodon uses HTTPS by default). To test if this may have caused issues, we take the certificate registration data from `crt.sh` [229], and check which certificate authorities (CAs) are used by instances, presented in Figure 5-10a. *Let's Encrypt* has been chosen as CA for more than 85% of the instances, likely because this service offers good automation and is free of cost [230]. This, again, confirms a central dependency in the DW. We also observe that certificate expiry is a noticeable issue (perhaps due to non-committed administrators). Figure 5-10b presents the number of instances that have outages caused by the expiry of their certificates. In the worst case, we find 105 instances to be down on one day (23 July 2018), removing nearly 200K toots from the system.

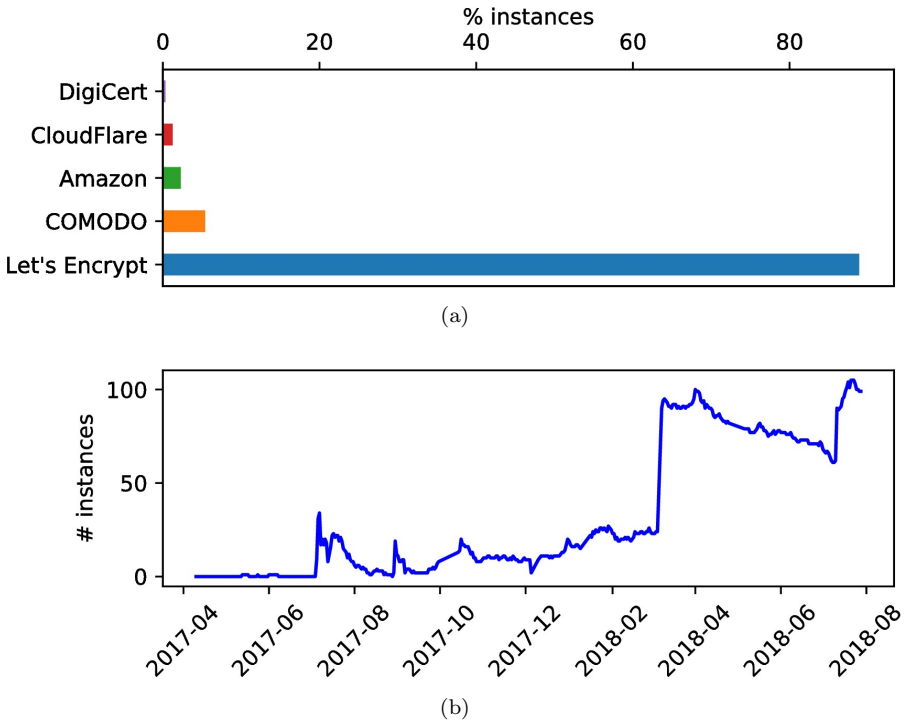


Figure 5-10: (*a*) Footprint of certificate authorities among the instances. (*b*) Unavailability of instances (on a per-day basis).

Closer inspection reveals that this was caused by the Let's Encrypt CA short expiry policy (90 days), which simultaneously expired certificates for all 105 instances. In total, these certificate expirations were responsible for 6.3% of the outages observed in our dataset.

AS Dependencies. Another potential explanation for some instance unavailability is that AS-wide network outages might occur. Due to the co-location of instances within the same AS, this could obviously have a widespread impact. To test this, we correlate the above instance unavailability to identify cases where *all* instances in a given AS simultaneously fail — this may indicate an AS outage. Table 5-1 presents a summary of the most frequent failures (we consider it to be an AS failure if *all* instances hosted in the same AS became unavailable simultaneously). We only include ASes that host at least eight instances (to avoid mistaking a small number of failures as an entire AS failure). We observe a small but notable set of outages. In total, 6 ASes suffer an outage. The largest is by AS9370 (Sakura, a Japanese hosting company), which lost 97 instances simultaneously, rendering 3.89M toots unavailable. The AS with most outages (15) is AS12322 (Free SAS), which removed nine instances. These outages are responsible for less than 1% of the failures observed; however, their impact is still significant. In total, these AS outages resulted in the (temporary) removal of 4.98M toots from the system, as well as 41.5K user accounts. Although this

ASN	Instances	Failures	IPs	Users	Toots	Org.	Rank	Peers
AS9370	97	1	95	33.4K	3.89M	Sakura	2.0K	10
AS20473	22	4	21	5.7K	936K	Choopa	143	150
AS8075	12	7	12	1.7K	35.4K	Microsoft	2.1K	257
AS12322	9	15	9	123	4.4K	Free SAS	3.2K	63
AS2516	9	4	8	559	102K	KDDI	70	123
AS9371	8	14	8	165	4.7K	Sakura	2.4K	3

Table 5-1: AS failures per number of hosted instances. Rank refers to CAIDA AS Rank, and Peers is the number of networks the AS peers [231].

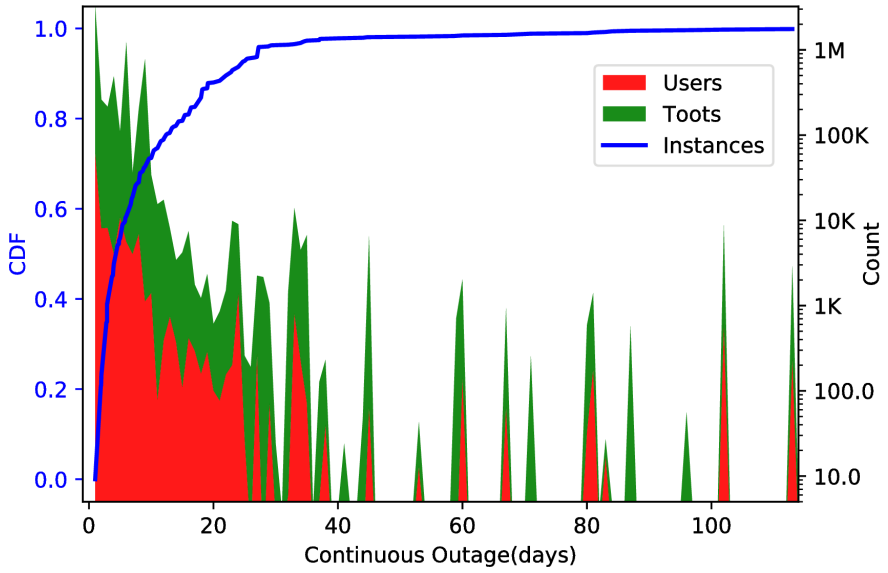


Figure 5-11: CDF of continuous outage (in days) of instances not accessible for at least one day (Y1-axis) and number of toots and users affected due to the outage (Y2-axis).

centralisation can result in such vulnerabilities, the decentralised management of Mastodon makes it difficult for administrators to coordinate placement to avoid these "hot spots".

Outage durations. Finally, for each outage, we briefly compute its duration and plot the CDF in Figure 5-11 (blue line, Y1-axis). While almost all instances (98%) go down at least once, a quarter of them are unavailable for *at least* one day before coming back online, ranging from 1 day (21%) to over a month (7%). Figure 5-11 also reports the number of toots and users affected by the outages: 14% of users cannot access their instances for a whole day at least once. Naturally, these measures are closely correlated to toot unavailability (*i.e.,* toots become unavailable when their host instance goes offline). In the worst case, we find one day (April 15, 2017) where 6% of all (global) toots were unavailable for the whole day. These findings suggest a need for more resilient approaches to DW management.

Domain	Toots from Home Users	#Home Users	Users OD	Users ID	Toots OD	Toots ID	Instances OD	Instances ID	Run by	AS (Country)
mstdn.jp	9.87M	23.2K	22.5K	24.7K	71.4M	1.94M	1352	1241	Individual	Cloudflare (US)
friends.nico	6.54M	8998	8809	23.3K	37.4M	2.57M	1273	1287	Dwango	Amazon (JP)
pawoo.net	4.72M	30.3K	27.6K	15.4K	34.9M	1.4M	1162	1106	Pixiv	Amazon (US)
mimumedon.com	3.29M	1671	507	7510	435K	366K	420	524	Individual	Sakura (JP)
imastodon.net	2.34M	1237	772	10.8K	2.37M	1.52M	711	865	Individuals (CF)	Amazon (US)
mastodon.social	1.65M	26.6K	24.8K	16.1K	30.9M	525K	1442	1083	Individual (CF)	Online SAS (FR)
mastodon.cloud	1.54M	5375	5209	106	7.35M	337	1198	39	Unknown	Cloudflare (US)
mstdn-workers.com	1.35M	610	576	12.5K	4.18M	1.98M	735	850	Individual (CF)	Amazon (JP)
vocalodon.net	914K	672	653	8441	2.6M	853K	981	631	Bokaro bowl (A)	Sakura (JP)
mstdn.osaka	803K	710	363	1.64K	2.68M	2.1M	561	862	Individual	Google (US)

Table 5-2: Top 10 instances as per number of toots from the home users. (OD: Out Degree, ID: In Degree, CF: Crowd-funded, A: maintained by selling Bokaro Bowl album).

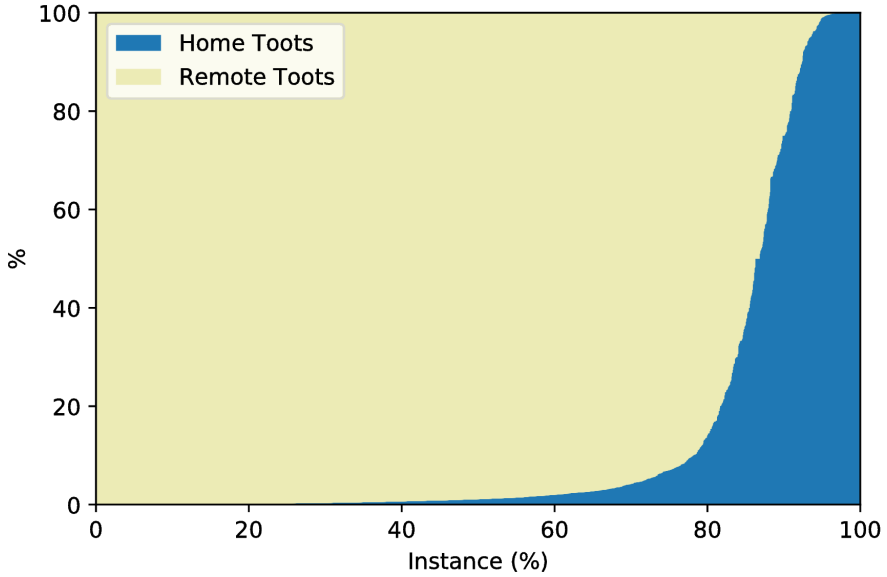


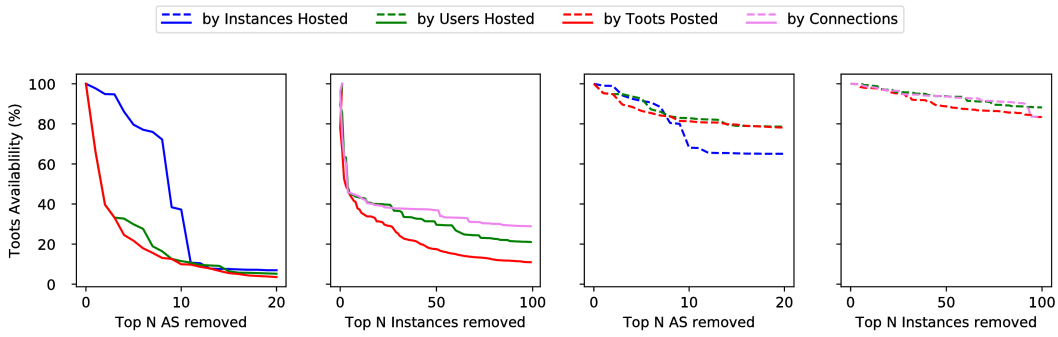
Figure 5-12: Ratio of home toots (produced on the instance) to remote toots (replicated from other ones).

5.5 Exploring Federation

The previous section has explored the central role of independent *instances* within the Mastodon ecosystem. The other major innovation introduced by the DW is *federation*. Here we inspect federation through as the distributed placement and sharing of content (toots) via the social graph. This section studies the resilience properties of DW federation in light of the frequent failures observed earlier.

5.5.1 Breaking the Content Federation

The above process of federation underpins the delivery of toots across the social graph. For example, when a user shares a toot, it results in the toot being shared with the instances of all their followers. Although we obviously cannot validate if a user reads a toot, we next explore the importance of federation for propagating content to timelines.



(a) No replication (b) No replication (c) Subscription Replication (d) Subscription Replication

Figure 5-13: Availability of toots based on (*a*) removing ASes, with ASes ranked based on #instances, #toots and #users hosted; and (*b*) removing instances, ranked by #users, #toots, and #connections with other instances. In (*c*) and (*d*), we report the toot availability when replicating across all instances that follow them.

Role of Remote Toots. Aiming to measure the prevalence of federated remote toots, Figure 5-12 plots the proportion of home *vs.* remote toots taken from the federated timeline (see Section 5.3) of every instance in our dataset, ordered by the percentage of home toots. The majority of toots on the federated timeline are actually generated by remote instances: 78% of instances produce under 10% of their own toots. In the most extreme case, we see that 5% of instances are entirely reliant on remote toots, and generate no home toots themselves. This suggests that some highly influential central instances operate as 'feeders' to the rest of the network. Also, the more toots an instance generates, the higher the probability of them being replicated to other instances (correlation 0.97), thus highlighting the importance of a small number of feeders, without whom smaller instances would be unable to bootstrap. This is another inherent form of centralisation, which any social system will struggle to deviate from.

These results motivate us to measure the impact of instance and AS failures on toot availability. We evaluate three scenarios: (*i*) where a toot is exclusively hosted on its home instance, and fetched by the remote instance on demand (denoted as "no replication"); (*ii*) where a toot is actively replicated to any instances with users that follow the toot's author ("subscription replication"); and (*iii*) where a toot is replicated onto a random set of n instances ("random replication"). Mastodon partly supports option (*ii*)but replicated toots are only temporarily cached, and they are not globally indexed, *i.e.,* they are only visible to users local to the instance where the replica is. For these experiments, we assume a scenario where toots are globally indexed, *e.g.,* via a Distributed Hash Table [232], allowing users to access replicated toots from any instance. For simplicity, we treat all toots as equal, even though in practice, more recent toots are likely to be more important.

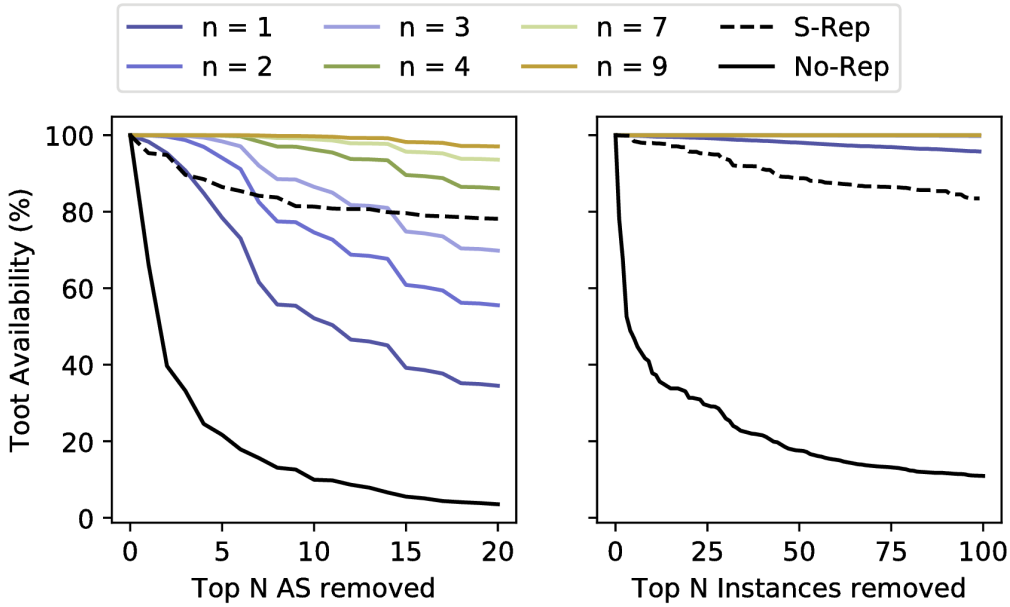


Figure 5-14: Availability of toots when removing top instances (measured by #toots) with random replication. Lines for $n > 4$ overlap indicating availability above 99% (97%) in case of instance (AS) failures.

No replication. In Figure 5-13a, we measure the availability of toots when removing entire ASes; and in Figure 5-13b, we examine the availability of toots when individual instances fail. Both plots depict results without replication enabled. In both cases, toot availability declines rapidly in the face of failures. Removing the top 10 instances (ranked by the number of toots) results in the deletion of 62.69% toots from Mastodon. Removing the top 10 ASes (again, ranked by the number of toots) results in 90.1% of toots becoming unavailable. Therefore, we argue that running Mastodon without replication is not a viable option if resilience is to be a priority.

Subscription-based Replication. Figures 5-13c and 5-13d report the availability of toots if they are replicated onto the instances that follow them, *i.e.,* via federated links. We consider any toot as available if there is at least one live Mastodon instance holding a replica, and assume the presence of a global index (such as a Distributed Hash Table) to discover toots in such replicas.

Availability improves using this method. For example, removing the top 10 instances now only results in 2.1% of toots becoming unavailable (as compared to 62.69% without replication). The equivalent values when removing the top 10 ASes are 18.66% with replication (*vs.* 90.1% without).

Random Replication. Above, we assumed that toots are only replicated to the followers' instances. We now experiment with a random replication strategy that copies each toot onto n random instances. We test for $n = \{1, 2, 3, 4, 7, 9\}$, alongside no replication (No-rep) and subscription-based replication (S-Rep). We do this for all toots and present the results in Figure 5-14. In each iteration, we remove the current remaining top instance (as ranked by the number of toots), and check the availability of toots according to the different replication strategies. The figures show that random replication substantially outperforms subscription-based replication. This is due to the biased way that subscription-based replication works, in which we find that 9.7% of toots have no replication due to a lack of followers, yet 23% of toots have more than 10 replicas because they are authored by popular users. After removing 25 instances, subscription-based replication has 95% availability, whereas 99.2% availability could have been achieved with just 1 *random* replication. More prominently, subscription-based replication tends to place *all* replicas onto a small set of popular instances (*i.e.,* where the followers are), due to the skewed popularity distribution of users. This is yet another consequence of the natural centralisation that these systems experience. Thus, removing these popular instances will remove not only the original toot but also the replica(s).

In contrast, the random strategy distributes the load more evenly, such that instance failures impact fewer replicas of the toots. There are a number of practical concerns that will impact such strategies. Most notably, it would be important to weight replication based on the resources available at the instance (*e.g.,* storage).

Thus this chapter presented a large-scale measurement study of the Decentralised Web (DW) through the lens of Mastodon identifying the pitfalls and discussing the mechanisms to improve the resiliency of the DW. Even though we have not seen any centralised ownership, across the instances, it can be seen that the economic and social pressures push the individual choices made to a power law distribution, affecting the resiliency of the DW. For example, 85% of the instances' administrators choose Let's Encrypt as the certificate authority, mainly due to economic factor, whereas half of the users sign up to just 10% of the instances owing to social pressures. These individual decisions, however same across the community, leads to a power law distribution of choices and found to have huge effect on the resiliency of the entire Mastodon ecosystem.

Chapter 6
Summary and Conclusions

Tomeet the growth in capacity requirements, this dissertation we presented the role of content placement amongst the edge devices. We also show how strategic content placement can help in achieving traffic and energy benefits for various applications. Overall, we find edge storages can be exploited for a range of applications to achieve benefits in content delivery.

Our exploration begins with an intuition of three categories of application based on the ownership – (*i*) application and content systems completely owned by an organisation (*ii*) application infrastructure owned by an organisation and content production done by the users and (*iii*) both the content system and infrastructure managed by the users. We use BBC iPlayer, Facebook live and Mastodon respectively for understanding how these systems can be benefited by strategic content placements. The general takeaways from the dissertation are given in Section 6.1, outlining the architectures to be designed for future networks based on the application categories.

6.1 Summary and Takeaways

6.1.1 Content Sharing

We begin with a feasibility check for content sharing by asking whether there are sufficient number of Wi-Fi access points for areas with varying population density. We find that even for areas with the lowest population density, there are sufficient neighbours to share the content (Section 3.3.1). We map these endpoints to user access logs of iPlayer and evaluate the potential bounds of traffic savings (Section 3.3.2). Based on this real-world setting, we then derive an optimisation problem for content placement within the edge devices (Section 3.3.5). Using this, we showed that 30-70% of traffic savings could be achieved

and also discuss the savings under various cache replacement strategies when curtailed by storage. We also show that up to 47% of traffic savings can be achieved within a small cell of ≈ 100 users by following an optimal content placement approach.

Thus using large scale access to a popular content streaming platform and real-world distribution potential edge devices, Chapter 3 evaluates the traffic savings from content sharing and proposes a framework for content sharing in future edge deployments.

6.1.2 Content Caching

We then proceed to explore Facebook live, where we began by characterising broadcast behaviour (Section 4.3). Being a popular platform, Facebook Live encounters peaks of 63 million online viewers. Despite this, we find that popularity is typically reserved for Page accounts and a small set of celebrity users. Over half of the social users on Facebook Live generate unwatched content at least once, with 41.5% of broadcasts never being viewed. This casts doubt over both the terms *social* and *broadcast* in the social live broadcast. In response, we proposed a simple mechanism to alleviate network and battery consumption. In cases where live streams receive no viewers, we proposed locally storing the content on the broadcasters' mobile devices until viewers arrive (rather than uploading video chunks that are never watched). Although this may introduce a slightly greater startup delay from viewers, such a model would reduce the number of bytes transferred by 21.9%, and mean that 23.18% of videos never need to leave the mobile device (only those that are viewed or archived by the broadcasters, allowing on-demand access later on, must be uploaded).

We then proceeded to explore the geographical (Section 4.4) properties of the system. Due to the social nature of Facebook Live, streams exhibit high degrees of the locality. 8% of viewers are within 25 KM of the broadcaster. We argue that this makes the service well suited to many recently proposed mobile edge computing architectures (*e.g.,* [213, 233, 234]). Unsurprisingly, these observations also result in clear trends in consumption with most users accessing domestic content. Exceptions to this tend to align with high levels of cultural and language similarity. Finally, we explored viewer engagement (Section 4.5) to find that even popular streams (*i.e.,* >10 viewers) have periods of being unwatched. We also found that most social engagement actually occurs *after* the live stream for the 46% of streams that were archived. This, again, casts doubt over the term *live* in social live broadcast, with Facebook Live exhibiting on-demand-style behaviour for many consumers.

Chapter 4 also constitutes a key contribution within the broader field of mobile social video consumption. However, there remains a series of interesting points for exploration. We have noted several systems implications from our findings. Implementing and evaluating these concepts would be a fruitful line of exploration, particularly as many are simple and could be easily integrated into social broadcast apps. We should note, however, that

we anticipate significant growth in the use of Facebook Live in the future. Hence, it is important to monitor how behaviour evolves to understand the long-term applicability of design choices. To date, we have also not been able to dive into how the social graph impacts video popularity, nor how such data could be used to predict and inform delivery strategies. Understanding how social topology impacts content consumption would be a clear line of future work. We also found that a number of spatial properties were correlated; we wish to expand this type of correlation analysis to other domains, *e.g.,* understanding if broadcasts correlate with socioeconomic metrics or things like tourism and immigration levels from other countries. Predictive models for capacity planning and/or caching could make great use of such insight.

■ 6.1.3 Content Replication

In Chapter 5, we focused on exploring challenges arising from two key innovations introduced by the DW: (*i*) the decomposition of a global service into many independent *instances*; and (*ii*) the process of the *federation*, whereby these instances collaborate and interact.

We found that Mastodon's design decision of giving everyone the ability to establish an independent instance of their own has led to an active ecosystem, with instances covering a wide variety of topics. However, a common theme in our work has been the discovery of apparent forms of centralisation within Mastodon. For example, 10% of instances host almost half of the users, and certain categories exhibit remarkable reliance on a small set of instances. This extends to hosting practices, with three ASes hosting almost two third of the users.

Our simulations further confirmed that these natural pressures towards centralisation lead to potential points of failure. For example, it was possible to reduce the LCC in the federation graph from 92% of all users to just 46% via the removal of five ASes. Similarly, outages in just ten instances can remove almost half of all toots. This is not a theoretical problem: we discovered regular instance (and even occasional AS) outages. Looking for possible mitigations, we experimented with simple replication strategies to find that availability can be dramatically improved by copying toots onto secondary instances, *i.e.,* by reducing the level of centralisation. Interestingly, the subscription-based strategy (loosely employed by Mastodon currently) is not as effective as a random strategy, due to the propensity to replicate toots onto the same set of instances where the followers are based.

We argue that if these problems are ignored, the DW may risk converging towards a semi-centralised system. As part of future work, we plan to explore the longer term properties of the DW more generally. We will also work on mitigations to some of the identified concerns (beyond the toot replication discussed in Section 5.5.1), including decentralised defences against, *e.g.,* malicious bots. One example of possible mitigation is the existing

instance blocking supported by Mastodon; our future work will investigate the impact that this has on the social graph and how it can be used to filter malicious content. Importantly, we argue that mitigations and innovations (*e.g.,* replication) should not depend on the exposure of user information to remote instances, and plan to experiment with building privacy-preserving federation mechanisms. Finally, we intend to study the effects of hate speech activities, as well as alt-right communities like Gab [235] starting to use Mastodon forks [236].

6.2 Future Directions

Thus in this era of growing applications and user devices, it is very important to build a framework which can scale to handle traffic demand. Section 3.4 proposes such a generic framework to share while streaming on demand. Through measurements and simulations, we show how the traffic demand can be handled through content caching and replication. However, as the ownership models for the applications are increasing with a mix of the load from static and dynamic content, Wi-Stitch or any other proposed caching mechanisms might not work in all the scenarios. For instance, federated learning and inference at the edge is becoming a popular research avenue owing to the growth of applications incorporating deep learning mechanisms. The ability of Wi-Stitch can be extended in such a scenario and if feasible, the computes can be stitched rather than just the caches.

6.3 Final Remarks

Capacity provisioning for growing content demand is a challenging problem, and the importance will only continue not just because users are interested in a variety of content items but also the growth in user facing applications offering various functionalities. This dissertation lays the groundwork for distribution of content and application at the network edge in an effective manner through strategic placements. Deep dive into various types of content streaming platforms and bringing out general design principle for designing edge systems represents the core technical contribution of this dissertation. In the course of collecting and analysis data from these platforms and evaluating the savings achieved, we have gained valuable insights on how the content from different gets accessed and how effective placement at the edge can improve traffic savings as well as the resiliency of a decentralised system. We also proposed a framework called Wi-Stitch to make the whole realm of content sharing feasible at scale. We also hope that our work will help expose the principles of edge placement schemes for application developers and that our results will help guide further explorations of this topic.

References

[1]A. Raman, S. Joglekar, E. De Cristofaro, N. Sastry, and G. Tyson. "Challenges in the Decentralised Web: The Mastodon Case", *Proceedings of the ACM Internet Measurement Conference (IMC'19)*. Amsterdam, Netherlands, 2019 (cited on page 23).

[2]M.-C. Lee, A. F. Molisch, N. Sastry, and A. Raman. "Individual Preference Probability Modeling and Parameterization for Video Content in Wireless Caching Networks", *IEEE/ACM Transactions on Networking* 27.2 (2019) (cited on page 24).

[3]A. Cartas*, M. Kocour*, A. Raman*, I. Leontiadis, J. Luque, N. Sastry, J. Nuñez-Martinez, D. Perino, and C. Segura. "A Reality Check on Inference at Mobile Networks Edge", *Proceedings of the EuroSys workshop on Edge Systems, Analytics and Networking (EdgeSys'19)*. Dresden, Germany, 2019 (cited on pages 24, 33 and 39).

[4]I. Quintana-Ramirez, A. Tsiopoulos, M. A. Lema, F. Sardis, L. Sequeira, J. Arias, A. Raman, A. Azam, and M. Dohler. "The Making of 5G: Building an End-to-End 5G-Enabled System", *IEEE Communications Standards Magazine* 2.4 (2018) (cited on page 24).

[5]A. Raman, N. Sastry, N. Mokari, M. Salehi, T. Faisal, A. Secker, and J. Chandaria. "Care to Share? An Empirical Analysis of Capacity Enhancement by Sharing at the Edge", *Proceedings of the MOBICOM Workshop on Technologies for the Wireless Edge (EdgeTech'18)*. New Delhi, India, 2018 (cited on pages 23 and 57).

[6]E. Obiodu, N. Sastry, and A. Raman. "Towards a taxonomy of differentiated service classes in the 5G era", *Proceedings of the IEEE 5G World Forum (5GWF'18)*. Santa Clara, CA, USA, 2018 (cited on page 24).

[7]A. Raman*, D. Karamshuk*, N. Sastry, J. Chandaria, and A. Secker. "Consume Local: Towards Carbon Free Content Delivery", *2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS)*. 2018, pp. 994–1003 (cited on pages 38 and 56).

[8]A. Raman, G. Tyson, and N. Sastry. "Facebook (A)Live?: Are Live Social Broadcasts Really Broadcasts?", *Proceedings of the 2018 World Wide Web Conference*. WWW '18. Lyon, France: International World Wide Web Conferences Steering Committee, 2018, pp. 1491–1500 (cited on pages 23 and 35).

[9]A. Raman, N. Sastry, A. Sathiaseelan, J. Chandaria, and A. Secker. "Wi-Stitch: Content Delivery in Converged Edge Networks", *SIGCOMM Comput. Commun. Rev.* 47.5 (Oct. 2017), pp. 73–78 (cited on page 23).

[10]A. Raman, N. Sastry, A. Sathiaseelan, A. Secker, and J. Chandaria. "Wi-Stitch: Content Delivery in Converged Edge Networks", *Proceedings of the SIGCOMM*

Workshop on Mobile Edge Communications (MECOMM'17). Los Angeles, CA, USA, 2017 (cited on pages 23 and 35).

[11]O. Holland, A. Raman, N. Sastry, S. Wong, J. Mack, and L. Lam. "Assessment of a Platform for Non-Contiguous Aggregation of IEEE 802.11 Waveforms in TV White Space", *Proceedings of the IEEE 83rd Vehicular Technology Conference (VTC Spring'16)*. Nanjing, China, 2016 (cited on page 24).

[12]O. Holland, H. Kokkinen, S. Wong, V. Friderikos, A. Raman, M. Dohler, and M. Lema. "Changing availability of TV white space in the UK", *IET Electronics Letters* 52.15 (2016) (cited on page 24).

[13]Sandvine. *The Global Internet Phenomena Report*. 2018 (cited on pages 17 and 20).

[14]E. Obiodu, A. Raman, and N. Sastry. "Is it time for a 999-like (or 112/911) system for critical information services?", *Proceedings of the IEEE/IFIP Network Operations and Management Symposium*. IEEE. Budapest, Hungary, 2020 (cited on page 17).

[15]Sandvine. *The Mobile Internet Phenomena Report*. 2019 (cited on page 18).

[16]J. Zhang, B. Chen, Y. Zhao, X. Cheng, and F. Hu. "Data Security and Privacy-Preserving in Edge Computing Paradigm: Survey and Open Issues", *IEEE Access* 6 (2018), pp. 18209–18237 (cited on page 18).

[17]Y. Liang, D. O'Keeffe, and N. Sastry. "PAIGE: Towards a Hybrid-Edge Design for Privacy-Preserving Intelligent Personal Assistants", *Proceedings of the Third ACM International Workshop on Edge Systems, Analytics and Networking*. EdgeSys '20. Heraklion, Greece: Association for Computing Machinery, 2020, 55–60 (cited on page 18).

[18]K. Graffi, C. Gross, D. Stingl, D. Hartung, A. Kovacevic, and R. Steinmetz. "LifeSocial. KOM: A secure and P2P-based solution for online social networks", *CCNC*. 2011 (cited on pages 19 and 43).

[19]S. Buchegger, D. Schiöberg, L.-H. Vu, and A. Datta. "PeerSoN: P2P social networking: early experiences and insights", *EuroSys Workshop on Social Network Systems*. 2009 (cited on pages 19 and 43).

[20]N. Anjum, D. Karamshuk, M. Shikh-Bahaei, and N. Sastry. "Survey on Peer-assisted Content Delivery Networks", *Computer Networks: The International Journal of Computer and Telecommunications Networking* 116.C (2017), pp. 79–95 (cited on pages 19, 34 and 43).

[21]S. Kaune, R. C. Rumin, G. Tyson, A. Mauthe, C. Guerrero, and R. Steinmetz. "Unraveling BitTorrent's File Unavailability: Measurements and Analysis", *2010 IEEE Tenth International Conference on Peer-to-Peer Computing (P2P)*. IEEE. 2010, pp. 1–9 (cited on pages 19 and 43).

[22]M.-C. Lee, A. F. Molisch, N. Sastry, and A. Raman. "Individual Preference Probability Modeling for Video Content in Wireless Caching Networks", *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*. 2017, pp. 1–7 (cited on pages 24 and 42).

[23]D. S. Wise and D. P. Friedman. "The one-bit reference count", *BIT Numerical Mathematics* 17.3 (1977), pp. 351–359 (cited on page 26).

[24]P. Green. "An Introduction to Network Architectures and Protocols", *IEEE Transactions on Communications* 28.4 (1980), pp. 413–424 (cited on page 26).

[25]M. N. Nelson, B. B. Welch, and J. K. Ousterhout. "Caching in the Sprite network file system", *ACM Transactions on Computer Systems (TOCS)* 6.1 (1988), pp. 134–154 (cited on page 26).

[26]R. Tobbicke. "Distributed file systems: focus on Andrew File System/Distributed File Service (AFS/DFS)", *Proceedings Thirteenth IEEE Symposium on Mass Storage Systems. Toward Distributed Storage and Data Management Systems*. 1994, pp. 23–26 (cited on page 26).

[27]Jaeyeon Jung, E. Sit, H. Balakrishnan, and R. Morris. "DNS performance and the effectiveness of caching", *IEEE/ACM Transactions on Networking* 10.5 (2002), pp. 589–603 (cited on page 26).

[28]W. S. Li, K. S. Candan, and D. Agrawal. *System and method for intelligent caching and refresh of dynamically generated and static web content*. US Patent 6,591,266. 2003 (cited on page 26).

[29]F. Aboukhadijeh. *Web Torrent*. `https://webtorrent.io/`. Accessed: 12-07-2020. 2014 (cited on page 27).

[30]P. Cao and S. Irani. "Cost-Aware WWW Proxy Caching Algorithms", *Proceedings of the USENIX Symposium on Internet Technologies and Systems on USENIX Symposium on Internet Technologies and Systems*. USITS'97. Monterey, California: USENIX Association, 1997, p. 18 (cited on page 27).

[31]I. Cooper, I. Melve, and G. Tomlinson. *Internet Web Replication and Caching Taxonomy*. `https://rfc-editor.org/rfc/rfc3040.txt`. Accessed: 12-07-2020. Jan. 2001 (cited on page 27).

[32]A. Chankhunthod, P. B. Danzig, C. Neerdaels, M. F. Schwartz, and K. J. Worrell. "A Hierarchical Internet Object Cache", *Proceedings of the 1996 Annual Conference on USENIX Annual Technical Conference*. ATEC '96. San Diego, CA: USENIX Association, 1996, p. 13 (cited on pages 27 and 29).

[33]D. K. Claffy and D. Wessels. *Internet Cache Protocol (ICP), version 2*. RFC 2186. Accessed: 12-07-2020. 1997 (cited on page 27).

[34]C. Bowman, P. B. Danzig, D. R. Hardy, U. Manber, and M. F. Schwartz. "The Harvest information discovery and access system", *Computer Networks and ISDN Systems* 28.1 (1995). Selected Papers from the Second World-Wide Web Conference, pp. 119 –125 (cited on page 27).

[35]P. K. W. Ross and V. Valloppillil. *Cache Array Routing Protocol v1.1*. Internet-Draft draft-vinod-carp-v1-03. Work in Progress. Internet Engineering Task Force, Feb. 1998. 6 pp. (cited on page 28).

[36]D. Wessels and P. A. Vixie. *Hyper Text Caching Protocol (HTCP/0.0)*. `https://rfc-editor.org/rfc/rfc2756.txt`. Accessed: 12-07-2020. Jan. 2000 (cited on page 28).

[37]A. Rousskov and D. Wessels. "Cache digests", *Computer Networks and ISDN Systems* 30.22 (1998), pp. 2155 –2168 (cited on pages 28 and 32).

[38]B. H. Bloom. "Space/Time Trade-Offs in Hash Coding with Allowable Errors", *Commun. ACM* 13.7 (July 1970), 422–426 (cited on page 28).

[39]A. Datta, K. Dutta, H. Thomas, D. VanderMeer, and K. Ramamritham. "Accelerating dynamic Web content generation", *IEEE Internet Computing* 6.5 (2002), pp. 27–36 (cited on page 29).

[40]B. Krishnamurthy, C. Wills, and Y. Zhang. "On the Use and Performance of Content Distribution Networks", *Proceedings of the 1st ACM SIGCOMM Workshop on Internet Measurement*. IMW '01. San Francisco, California, USA: Association for Computing Machinery, 2001, 169–182 (cited on pages 29 and 32).

[41]A. Sumits. *The Internet is Closer to Home Than You Think*. `https://blogs.cisco.com/sp/the-internet-is-closer-to-home-than-you-think`. Accessed: 12-07-2020. 2017 (cited on page 29).

[42]A. Vakali and G. Pallis. "Content delivery networks: Status and Trends", *IEEE Internet Computing* 7.6 (2003), pp. 68–74 (cited on page 29).

[43]A.-M. K. Pathan and R. Buyya. *A taxonomy and survey of content delivery networks*. Tech. rep. University of Melbourne, 2007 (cited on page 30).

[44]M. A. Salahuddin, J. Sahoo, R. Glitho, H. Elbiaze, and W. Ajib. "A Survey on Content Placement Algorithms for Cloud-Based Content Delivery Networks", *IEEE Access* 6 (2018), pp. 91–114 (cited on pages 30 and 32).

[45]N. Laoutaris, G. Smaragdakis, K. Oikonomou, I. Stavrakakis, and A. Bestavros. "Distributed Placement of Service Facilities in Large-Scale Networks", *IEEE INFOCOM 2007 - 26th IEEE International Conference on Computer Communications*. 2007, pp. 2144–2152 (cited on page 31).

[46]M. Bateni and M. Hajiaghayi. “Assignment Problem in Content Distribution Networks: Unsplittable Hard-Capacitated Facility Location”, *ACM Trans. Algorithms* 8.3 (July 2012) (cited on page 31).

[47]P. Radoslavov, R. Govindan, and D. Estrin. “Topology-informed Internet replica placement”, *Computer Communications* 25.4 (2002), pp. 384 –392 (cited on page 31).

[48]H. Yin, X. Zhang, T. Zhan, Y. Zhang, G. Min, and D. O. Wu. “NetClust: A Framework for Scalable and Pareto-Optimal Media Server Placement”, *IEEE Transactions on Multimedia* 15.8 (2013), pp. 2114–2124 (cited on page 31).

[49]A. Rappaport and D. Raz. “Update aware replica placement”, *Proceedings of the 9th International Conference on Network and Service Management (CNSM 2013)*. 2013, pp. 92–99 (cited on page 31).

[50]J. Wu and K. Ravindran. “Optimization algorithms for proxy server placement in content distribution networks”, *2009 IFIP/IEEE International Symposium on Integrated Network Management-Workshops*. 2009, pp. 193–198 (cited on page 31).

[51]K. Kalpakis, K. Dasgupta, and O. Wolfson. “Optimal placement of replicas in trees with read, write, and storage costs”, *IEEE Transactions on Parallel and Distributed Systems* 12.6 (2001), pp. 628–637 (cited on page 31).

[52]A. Benoit, V. Rehn-Sonigo, and Y. Robert. “Replica placement and access policies in tree networks”, *IEEE Transactions on Parallel and Distributed Systems* 19.12 (2008), pp. 1614–1627 (cited on page 31).

[53]M. Szymaniak, G. Pierre, and M. van Steen. “Latency-driven replica placement”, *The 2005 Symposium on Applications and the Internet*. 2005, pp. 399–405 (cited on page 31).

[54]S. Ahuja and M. Krunz. “Algorithms for Server Placement in Multiple-Description-Based Media Streaming”, *IEEE Transactions on Multimedia* 10.7 (2008), pp. 1382–1392 (cited on page 31).

[55]A. Benoit, P. Renaud-Goud, and Y. Robert. “Power-Aware Replica Placement and Update Strategies in Tree Networks”, *2011 IEEE International Parallel Distributed Processing Symposium*. 2011, pp. 2–13 (cited on page 31).

[56]J. Sahoo, M. A. Salahuddin, R. Glitho, H. Elbiaze, and W. Ajib. “A Survey on Replica Server Placement Algorithms for Content Delivery Networks”, *IEEE Communications Surveys Tutorials* 19.2 (2017), pp. 1002–1026 (cited on page 31).

[57]B. Wu and A. D. Kshemkalyani. “Objective-optimal algorithms for long-term Web prefetching”, *IEEE Transactions on Computers* 55.1 (2006), pp. 2–17 (cited on page 31).

[58]A. Sidiropoulos, G. Pallis, D. Katsaros, K. Stamos, A. Vakali, and Y. Manolopoulos. "Prefetching in Content Distribution Networks via Web Communities Identification and Outsourcing", *World Wide Web* 11.1 (Mar. 2008), 39–70 (cited on page 31).

[59]Yan Chen, Lili Qiu, Weiyu Chen, Luan Nguyen, and R. H. Katz. "Efficient and adaptive Web replication using content clustering", *IEEE Journal on Selected Areas in Communications* 21.6 (2003), pp. 979–994 (cited on page 31).

[60]D. S. Berger. "Towards Lightweight and Robust Machine Learning for CDN Caching", *Proceedings of the 17th ACM Workshop on Hot Topics in Networks*. HotNets '18. Redmond, WA, USA: Association for Computing Machinery, 2018, 134–140 (cited on pages 31 and 32).

[61]J. Kangasharju, J. Roberts, and K. W. Ross. "Object replication strategies in content distribution networks", *Computer Communications* 25.4 (2002), pp. 376 –383 (cited on page 32).

[62]P. Rodriguez, C. Spanner, and E. W. Biersack. "Analysis of Web caching architectures: hierarchical and distributed caching", *IEEE/ACM Transactions on Networking* 9.4 (2001), pp. 404–418 (cited on page 32).

[63]S. Gadde, M. Rabinovich, and J. Chase. "Reduce, reuse, recycle: An approach to building large internet caches", *Proceedings. The Sixth Workshop on Hot Topics in Operating Systems (Cat. No. 97TB100133)*. IEEE. 1997, pp. 93–98 (cited on page 32).

[64]D. Karger, A. Sherman, A. Berkheimer, B. Bogstad, R. Dhanidina, K. Iwamoto, B. Kim, L. Matkins, and Y. Yerushalmi. "Web caching with consistent hashing", *Computer Networks* 31.11-16 (1999), pp. 1203–1213 (cited on page 32).

[65]B. M. Maggs and R. K. Sitaraman. "Algorithmic Nuggets in Content Delivery", *SIGCOMM Comput. Commun. Rev.* 45.3 (July 2015), 52–66 (cited on page 32).

[66]A. Sundarrajan, M. Feng, M. Kasbekar, and R. K. Sitaraman. "Footprint Descriptors: Theory and Practice of Cache Provisioning in a Global CDN", *Proceedings of the 13th International Conference on Emerging Networking EXperiments and Technologies*. CoNEXT '17. Incheon, Republic of Korea: Association for Computing Machinery, 2017, 55–67 (cited on page 32).

[67]N. Beckmann, H. Chen, and A. Cidon. "{LHD}: Improving Cache Hit Rate by Maximizing Hit Density", *15th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 18)*. 2018, pp. 389–403 (cited on page 32).

[68]Z. Song, D. S. Berger, K. Li, and W. Lloyd. "Learning Relaxed Belady for Content Distribution Network Caching", *17th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 20)*. 2020, pp. 529–544 (cited on page 32).

[69]V. K. Adhikari, Yang Guo, Fang Hao, M. Varvello, V. Hilt, M. Steiner, and Z. Zhang. "Unreeling netflix: Understanding and improving multi-CDN movie delivery", *2012 Proceedings IEEE INFOCOM*. 2012, pp. 1620–1628 (cited on page 32).

[70]V. K. Adhikari, Y. Guo, F. Hao, V. Hilt, Z. Zhang, M. Varvello, and M. Steiner. "Measurement Study of Netflix, Hulu, and a Tale of Three CDNs", *IEEE/ACM Transactions on Networking* 23.6 (2015), pp. 1984–1997 (cited on page 32).

[71]T. Leighton. "Improving Performance on the Internet", *Communications of the ACM* 52.2 (2009), pp. 44–51 (cited on page 32).

[72]E. Nygren, R. K. Sitaraman, and J. Sun. "The Akamai Network: A Platform for High-Performance Internet Applications", *ACM SIGOPS Operating Systems Review* 44.3 (2010), pp. 2–19 (cited on page 32).

[73]K. Jain, M. Mahdian, and A. Saberi. "A New Greedy Approach for Facility Location Problems", *Proceedings of the Thiry-Fourth Annual ACM Symposium on Theory of Computing*. STOC '02. Montreal, Quebec, Canada: Association for Computing Machinery, 2002, 731–740 (cited on page 32).

[74]Lili Qiu, V. N. Padmanabhan, and G. M. Voelker. "On the placement of Web server replicas", *Proceedings IEEE INFOCOM 2001. Conference on Computer Communications. Twentieth Annual Joint Conference of the IEEE Computer and Communications Society (Cat. No.01CH37213)*. Vol. 3. 2001, 1587–1596 vol.3 (cited on page 32).

[75]A. A. Bhattacharya, D. Culler, E. Friedman, A. Ghodsi, S. Shenker, and I. Stoica. "Hierarchical scheduling for diverse datacenter workloads", *Proceedings of the 4th annual Symposium on Cloud Computing*. 2013, pp. 1–15 (cited on page 32).

[76]T. F. Abdelzaher, K. G. Shin, and N. Bhatti. "Performance guarantees for Web server end-systems: a control-theoretical approach", *IEEE Transactions on Parallel and Distributed Systems* 13.1 (2002), pp. 80–96 (cited on page 32).

[77]B. Urgaonkar, P. Shenoy, A. Chandra, P. Goyal, and T. Wood. "Agile Dynamic Provisioning of Multi-Tier Internet Applications", *ACM Trans. Auton. Adapt. Syst.* 3.1 (Mar. 2008) (cited on page 32).

[78]R. P. Doyle, J. S. Chase, O. M. Asad, W. Jin, and A. M. Vahdat. "Model-Based Resource Provisioning in a Web Service Utility", *Proceedings of the 4th Conference on USENIX Symposium on Internet Technologies and Systems - Volume 4*. USITS'03. Seattle, WA: USENIX Association, 2003, p. 5 (cited on page 32).

[79]S. S. Kanhere, H. Sethu, and A. B. Parekh. "Fair and efficient packet scheduling using Elastic Round Robin", *IEEE Transactions on Parallel and Distributed Systems* 13.3 (2002), pp. 324–336 (cited on page 32).

[80]J. Helt, G. Feng, S. Seshan, and V. Sekar. “Sandpaper: Mitigating Performance Interference in CDN Edge Proxies”, *Proceedings of the 4th ACM/IEEE Symposium on Edge Computing*. SEC ’19. Arlington, Virginia: Association for Computing Machinery, 2019, 30–46 (cited on page 32).

[81]K. Mokhtarian and H.-A. Jacobsen. “Caching in Video CDNs: Building Strong Lines of Defense”, *Proceedings of the Ninth European Conference on Computer Systems*. EuroSys ’14. Amsterdam, The Netherlands: Association for Computing Machinery, 2014 (cited on page 32).

[82]E. Friedlander and V. Aggarwal. “Generalization of LRU Cache Replacement Policy with Applications to Video Streaming”, *ACM Trans. Model. Perform. Eval. Comput. Syst.* 4.3 (Aug. 2019) (cited on page 32).

[83]H. S. Goian, O. Y. Al-Jarrah, S. Muhaidat, Y. Al-Hammadi, P. Yoo, and M. Dianati. “Popularity-Based Video Caching Techniques for Cache-Enabled Networks: A Survey”, *IEEE Access* 7 (2019), pp. 27699–27719 (cited on pages 32 and 34).

[84]T. Plagemann, V. Goebel, A. Mauthe, L. Mathy, T. Turletti, and G. Urvoy-Keller. “From content distribution networks to content networks — issues and challenges”, *Computer Communications* 29.5 (2006). Networks of Excellence, pp. 551 –562 (cited on page 33).

[85]W. Shi, G. Pallis, and Z. Xu. “Edge Computing [Scanning the Issue]”, *Proceedings of the IEEE* 107.8 (2019), pp. 1474–1481 (cited on page 33).

[86]D. Karamshuk, N. Sastry, M. Al-Bassam, A. Secker, and J. Chandaria. “Take-Away TV: Recharging Work Commutes With Predictive Preloading of Catch-Up TV Content”, *IEEE Journal on Selected Areas in Communications* 34.8 (2016), pp. 2091–2101 (cited on pages 34, 35, 41 and 59).

[87]S. Vassilvitskii and T. Lykouris. *Caching using machine learned predictions*. Accessed: 12-07-2020. Aug. 2019 (cited on page 34).

[88]W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu. “Edge Computing: Vision and Challenges”, *IEEE Internet of Things Journal* 3.5 (2016), pp. 637–646 (cited on page 34).

[89]M. Caprolu, R. Di Pietro, F. Lombardi, and S. Raponi. “Edge Computing Perspectives: Architectures, Technologies, and Open Security Issues”, *2019 IEEE International Conference on Edge Computing (EDGE)*. 2019, pp. 116–123 (cited on page 34).

[90]Zhi-Li Zhang, J. Kurose, J. D. Salehi, and D. Towsley. “Smoothing, statistical multiplexing, and call admission control for stored video”, *IEEE Journal on Selected Areas in Communications* 15.6 (1997), pp. 1148–1166 (cited on page 34).

[91]J. D. Salehi, Zhi-Li Zhang, J. Kurose, and D. Towsley. "Supporting stored video: reducing rate variability and end-to-end resource requirements through optimal smoothing", *IEEE/ACM Transactions on Networking* 6.4 (1998), pp. 397–410 (cited on page 34).

[92]A. Sharma, A. Bestavros, and I. Matta. "dPAM: a distributed prefetching protocol for scalable asynchronous multicast in P2P systems", *Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies.* Vol. 2. 2005, 1139–1150 vol. 2 (cited on page 34).

[93]S. Annapureddy, S. Guha, C. Gkantsidis, D. Gunawardena, and P. Rodriguez. "Exploring VoD in P2P Swarming Systems", *IEEE INFOCOM 2007 - 26th IEEE International Conference on Computer Communications*. 2007, pp. 2571–2575 (cited on page 34).

[94]K. Suh, C. Diot, J. Kurose, L. Massoulie, C. Neumann, D. Towsley, and M. Varvello. "Push-to-Peer Video-on-Demand System: Design and Evaluation", *IEEE Journal on Selected Areas in Communications* 25.9 (2007), pp. 1706–1716 (cited on page 34).

[95]H. Schwarz, D. Marpe, and T. Wiegand. "Overview of the Scalable Video Coding Extension of the H.264/AVC Standard", *IEEE Transactions on Circuits and Systems for Video Technology* 17.9 (2007), pp. 1103–1120 (cited on page 35).

[96]K. Poularakis, G. Iosifidis, A. Argyriou, I. Koutsopoulos, and L. Tassiulas. "Caching and operator cooperation policies for layered video content delivery", *IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications*. 2016, pp. 1–9 (cited on page 35).

[97]D. Applegate, A. Archer, V. Gopalakrishnan, S. Lee, and K. K. Ramakrishnan. "Optimal Content Placement for a Large-Scale VoD System", *IEEE/ACM Transactions on Networking* 24.4 (2016), pp. 2114–2127 (cited on page 35).

[98]D. Karamshuk, N. Sastry, A. Secker, and J. Chandaria. "ISP-friendly peer-assisted on-demand streaming of long duration content in BBC iPlayer", *2015 IEEE Conference on Computer Communications (INFOCOM)*. 2015, pp. 289–297 (cited on pages 35, 41 and 48).

[99]J. Dai, F. Liu, B. Li, B. Li, and J. Liu. "Collaborative Caching in Wireless Video Streaming Through Resource Auctions", *IEEE Journal on Selected Areas in Communications* 30.2 (2012), pp. 458–466 (cited on pages 35 and 38).

[100]G. Nencioni, N. Sastry, J. Chandaria, and J. Crowcroft. "Understanding and Decreasing the Network Footprint of Catch-up Tv", *Proceedings of the 22nd International Conference on WWW*. Rio de Janeiro, Brazil, 2013, pp. 965–976 (cited on pages 35, 41, 48 and 55).

[101]G. Nencioni, N. Sastry, G. Tyson, V. Badrinarayanan, D. Karamshuk, J. Chandaria, and J. Crowcroft. "SCORE: Exploiting Global Broadcasts to Create Offline Personal Channels for On-Demand Access", *IEEE/ACM Transactions on Networking* 24.4 (2016), pp. 2429–2442 (cited on pages 35, 47 and 60).

[102]K. Bilal, E. Baccour, A. Erbad, A. Mohamed, and M. Guizani. "Collaborative joint caching and transcoding in mobile edge networks", *Journal of Network and Computer Applications* 136 (2019), pp. 86 –99 (cited on page 35).

[103]A Mohamed and N. Mustafa. "Functions of X2 interface", *International Journal of Science and Research* (2016) (cited on page 35).

[104]E. Baccour, A. Erbad, K. Bilal, A. Mohamed, and M. Guizani. "PCCP: Proactive Video Chunks Caching and Processing in edge networks", *Future Generation Computer Systems* 105 (2020), pp. 44 –60 (cited on page 35).

[105]G. Xylomenos, C. N. Ververidis, V. A. Siris, N. Fotiou, C. Tsilopoulos, X. Vasilakos, K. V. Katsaros, and G. C. Polyzos. "A survey of information-centric networking research", *IEEE Communications Surveys & Tutorials* 16.2 (2014), pp. 1024–1049 (cited on pages 35 and 47).

[106]A. Ioannou and S. Weber. "A taxonomy of caching approaches in information-centric network architectures", *Elsevier Journal* (2013) (cited on page 36).

[107]G. Zhang, Y. Li, and T. Lin. "Caching in Information Centric Networking: A Survey", *Comput. Netw.* 57.16 (Nov. 2013), 3128–3141 (cited on page 36).

[108]R. Chiocchetti, D. Rossi, G. Rossini, G. Carofiglio, and D. Perino. "Exploit the Known or Explore the Unknown? Hamlet-like Doubts in ICN", *Proceedings of the Second Edition of the ICN Workshop on Information-Centric Networking*. ICN '12. Helsinki, Finland: Association for Computing Machinery, 2012, 7–12 (cited on pages 36 and 38).

[109]S. I. Hong, D. J. Byun, B. J. Lee, M. W. Jang, and J. H. Kim. *Method for content discovery of node in intra-domain and inter-domain in content centric network and node therefor*. `https://patents.google.com/patent/US20130339481`. Accessed: 12-07-2020. Dec. 2013 (cited on page 36).

[110]"Content Discovery for Information-Centric Networking", *Comput. Netw.* 83.C (June 2015), 1–14 (cited on pages 36 and 38).

[111]L. Wang, S. Bayhan, J. Ott, J. Kangasharju, A. Sathiaseelan, and J. Crowcroft. "Pro-Diluvian: Understanding Scoped-Flooding for Content Discovery in Information-Centric Networking", *Proceedings of the 2nd ACM Conference on Information-Centric Networking*. ACM-ICN '15. San Francisco, California, USA: Association for Computing Machinery, 2015, 9–18 (cited on pages 36 and 38).

[112]L. Wang, S. Bayhan, J. Ott, J. Kangasharju, and J. Crowcroft. "Understanding Scoped-Flooding for Content Discovery and Caching in Content Networks", *IEEE Journal on Selected Areas in Communications* 36.8 (2018), pp. 1887–1900 (cited on pages 36 and 38).

[113]K. Shanmugam, N. Golrezaei, A. G. Dimakis, A. F. Molisch, and G. Caire. "FemtoCaching: Wireless Content Delivery Through Distributed Caching Helpers", *IEEE Transactions on Information Theory* 59.12 (2013), pp. 8402–8413 (cited on pages 38 and 57).

[114]C. Yang, Y. Yao, Z. Chen, and B. Xia. "Analysis on Cache-Enabled Wireless Heterogeneous Networks", *IEEE Transactions on Wireless Communications* 15.1 (2016), pp. 131–145 (cited on page 38).

[115]H. Li and D. Hu. "Mobility prediction based seamless RAN-cache handover in HetNet", *2016 IEEE Wireless Communications and Networking Conference*. 2016, pp. 1–7 (cited on page 38).

[116]S. Shukla, O. Bhardwaj, A. A. Abouzeid, T. Salonidis, and T. He. "Proactive Retention-Aware Caching With Multi-Path Routing for Wireless Edge Networks", *IEEE Journal on Selected Areas in Communications* 36.6 (2018), pp. 1286–1299 (cited on page 38).

[117]Z. Chen and M. Kountouris. "D2D caching vs. small cell caching: Where to cache content in a wireless network?", *2016 IEEE 17th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*. 2016, pp. 1–6 (cited on page 38).

[118]A. Wolman, M Voelker, N. Sharma, N. Cardwell, A. Karlin, and H. M. Levy. "On the scale and performance of cooperative web proxy caching", *Proceedings of the seventeenth ACM symposium on Operating systems principles*. 1999, pp. 16–31 (cited on page 38).

[119]L. Fan, P. Cao, J. Almeida, and A. Z. Broder. "Summary cache: a scalable wide-area web cache sharing protocol", *IEEE/ACM transactions on networking* 8.3 (2000), pp. 281–293 (cited on page 38).

[120]M. R. Korupolu, C. Plaxton, and R. Rajaraman. "Placement Algorithms for Hierarchical Cooperative Caching", *Journal of Algorithms* 38.1 (2001), pp. 260 –302 (cited on page 38).

[121]L. Ramaswamy, Ling Liu, and A. Iyengar. "Cache Clouds: Cooperative Caching of Dynamic Documents in Edge Networks", *25th IEEE International Conference on Distributed Computing Systems (ICDCS'05)*. 2005, pp. 229–238 (cited on page 38).

[122]J. Ni and D. H. Tsang. “Large-scale cooperative caching and application-level multicast in multimedia content delivery networks”, *IEEE Communications Magazine* 43.5 (2005), pp. 98–105 (cited on page 38).

[123]S. Borst, V. Gupta, and A. Walid. “Self-organizing algorithms for cache cooperation in content distribution networks”, *Bell Labs Technical Journal* 14.3 (2009), pp. 113–125 (cited on page 38).

[124]S. Borst, V. Gupta, and A. Walid. “Distributed caching algorithms for content distribution networks”, *Proceedings of INFOCOM*. IEEE. 2010, pp. 1–9 (cited on page 38).

[125]Z. Chen, W. Hu, J. Wang, S. Zhao, B. Amos, et al. “An Empirical Study of Latency in an Emerging Class of Edge Computing Applications for Wearable Cognitive Assistance”, *Proceedings of the SEC 2017*. New York, NY, USA: ACM (cited on page 39).

[126]J. Cho, K. Sundaresan, R. Mahindra, et al. “ACACIA: Context-aware Edge Computing for Continuous Interactive Applications over Mobile Networks”, *Proceedings of the CoNEXT ’16*. Irvine, California, USA: ACM, 2016, pp. 375–389 (cited on page 39).

[127]W. Zhang, B. Han, and P. Hui. “On the Networking Challenges of Mobile Augmented Reality”, *Proceedings of the Workshop on VR/AR Network ’17*. Los Angeles, CA, USA: ACM, 2017, pp. 24–29 (cited on page 39).

[128]3GPP. *TS23.501, V15.3.0 (2018-09), Technical Specification Group Services and System Aspects; Study on Architecture for the 5G System; Stage 2*. `https://www.etsi.org/deliver/etsi_ts/123500_123599/123501/15.03.00_60/ts_123501v150300p.pdf`. Accessed: 12-07-2020. Sept. 2018 (cited on page 39).

[129]P. D. Amer and L. N. Cassel. “Management of sampled real-time network measurements”, *Proceedings. 14th Conference on Local Computer Networks*. IEEE. 1989, pp. 62–68 (cited on page 39).

[130]P. Danzig, S. Jamin, R. Cáceres, D Mitzel, and D. Estrin. “An empirical workload model for driving wide-area TCP/IP network simulations”, *Internetworking: Research and Experience* 3.1 (1992), pp. 1–26 (cited on page 39).

[131]V. Paxson. “End-to-end Internet packet dynamics”, *Proceedings of the ACM SIGCOMM’97*. 1997, pp. 139–152 (cited on page 39).

[132]V. Paxson, J. Mahdavi, A. Adams, and M. Mathis. “An architecture for large scale Internet measurement”, *IEEE Communications Magazine* 36.8 (1998), pp. 48–54 (cited on page 39).

[133]S. Sanfilippo. *hping*. `http://www.hping.org/`. Accessed: 12-07-2020. 2005 (cited on page 40).

[134]V. Jacobson. *Pathchar: A tool to infer characteristics of Internet paths.* `ftp://ftp.ee.lbl.gov/pathchar/msri-talk.pdf`. Accessed: 12-07-2020. 1997 (cited on page 40).

[135]M. Jain and C. Dovrolis. "Pathload: A measurement tool for end-to-end available bandwidth", *In Proceedings of Passive and Active Measurements (PAM) Workshop.* 2002 (cited on page 40).

[136]C. Kreibich, N. Weaver, B. Nechaev, and V. Paxson. "Netalyzr: Illuminating the Edge Network", *Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement.* IMC '10. Melbourne, Australia: Association for Computing Machinery, 2010, 246–259 (cited on page 40).

[137]S. Jaiswal, G. Iannaccone, C. Diot, J. Kurose, and D. Towsley. "Inferring TCP connection characteristics through passive measurements", *IEEE INFOCOM 2004.* Vol. 3. 2004, 1582–1592 vol.3 (cited on page 40).

[138]N. Duffield. "Sampling for passive internet measurement: A review", *Statistical Science* 19.3 (2004), pp. 472–498 (cited on page 40).

[139]C. Dwork, A. Roth, et al. "The algorithmic foundations of differential privacy.", *Foundations and Trends in Theoretical Computer Science* 9.3-4 (2014), pp. 211–407 (cited on page 40).

[140]G. Yao, J. Bi, and Z. Zhou. "Passive IP Traceback: Capturing the Origin of Anonymous Traffic through Network Telescopes", *SIGCOMM Comput. Commun. Rev.* 40.4 (Aug. 2010), 413–414 (cited on page 41).

[141]B.-Y. Choi, S. Moon, R. Cruz, Z.-L. Zhang, and C. Diot. "Practical Delay Monitoring for ISPs", *Proceedings of the 2005 ACM Conference on Emerging Network Experiment and Technology.* CoNEXT '05. Toulouse, France: Association for Computing Machinery, 2005, 83–92 (cited on page 41).

[142]Y. Carlinet, H. Debar, Y. Gourhant, and L. Mé. "Caching P2P Traffic: What Are the Benefits for an ISP?", *2010 Ninth International Conference on Networks.* 2010, pp. 376–383 (cited on page 41).

[143]I. Bermudez, M. Mellia, and M. Meo. "Passive characterization of sopcast usage in residential ISPs", *2011 IEEE International Conference on Peer-to-Peer Computing.* 2011, pp. 1–9 (cited on page 41).

[144]A. Grammenos, A. Raman, T. Böttger, Z. Gilani, and G. Tyson. "Dissecting the Workload of a Major Adult Video Portal", *Passive and Active Measurement.* Ed. by A. Sperotto, A. Dainotti, and B. Stiller. Cham: Springer International Publishing, 2020, pp. 267–279 (cited on page 41).

[145]J. Dai, Z. Hu, B. Li, J. Liu, and B. Li. "Collaborative Hierarchical Caching with Dynamic Request Routing for Massive Content Distribution", *2012 Proceedings IEEE INFOCOM*. 2012 (cited on pages 41 and 46).

[146]V. Jacobson, M. Mosko, D Smetters, and J. Garcia-Luna-Aceves. "Content-centric networking", *Whitepaper, Palo Alto Research Center* (2007), pp. 2–4 (cited on pages 41 and 46).

[147]W. K. Chai, D. He, I. Psaras, and G. Pavlou. "Cache "Less for More" in Information-centric Networks (extended version)", *Computer Communications* 36.7 (2013) (cited on pages 41, 46 and 47).

[148]L. Velasco, L. M. Contreras, G. Ferraris, A. Stavdas, F. Cugini, M. Wiegand, and J. P. Fernandez-Palacios. "A service-oriented hybrid access network and clouds architecture", *IEEE Communications Magazine* 53.4 (2015), pp. 159–165 (cited on pages 41, 46 and 47).

[149]D. Karamshuk, N. Sastry, A. Secker, and J. Chandaria. "On factors affecting the usage and adoption of a nation-wide TV streaming service", *2015 INFOCOM*. 2015, pp. 837–845 (cited on pages 41, 47 and 48).

[150]P. Ameigeiras, J. J. Ramos-Munoz, J. Navarro-Ortiz, and J. Lopez-Soler. "Analysis and modelling of YouTube traffic", *Transactions on Emerging Telecommunications Technologies* 23.4 (2012), pp. 360–377 (cited on page 42).

[151]M. Cha, H. Kwak, P. Rodriguez, Y.-Y. Ahn, and S. Moon. "Analyzing the Video Popularity Characteristics of Large-scale User Generated Content Systems", *IEEE/ACM Trans. Netw.* 17.5 (Oct. 2009), pp. 1357–1370 (cited on page 42).

[152]G. Tyson, Y. Elkhatib, N. Sastry, and S. Uhlig. "Demystifying Porn 2.0: A Look into a Major Adult Video Streaming Website", *Proceedings of the 2013 Conference on Internet Measurement Conference*. IMC '13. Barcelona, Spain: ACM, 2013, pp. 417–426 (cited on pages 42 and 89).

[153]K. Xing, B. Zhang, B. Zhou, and Y. Liu. "Behavior based user interests extraction algorithm", *Internet of Things (iThings)* (2011) (cited on page 42).

[154]S. Joglekar, N. Sastry, and M. Redi. "Like at First Sight: Understanding User Engagement with the World of Microvideos", *International Conference on Social Informatics*. Springer. 2017, pp. 237–256 (cited on page 42).

[155]Y. Chen, Y. Yu, W. Zhang, and J. Shen. "Analyzing User Behavior History for constructing user profile", *2008 IEEE International Symposium on IT in Medicine and Education*. 2008, pp. 343–348 (cited on page 42).

[156]B. Hu, M. Jamali, and M. Ester. "Learning the Strength of the Factors Influencing User Behavior in Online Social Networks", (2012), pp. 368–375 (cited on page 42).

[157]V. Gopalakrishnan, R. Jana, R. Knag, K. K. Ramakrishnan, D. F. Swayne, and V. A. Vaishampayan. "Characterizing Interactive Behavior in a Large-Scale Operational IPTV Environment", *2010 Proceedings IEEE INFOCOM*. 2010, pp. 1–5 (cited on page 42).

[158]B. Chang, L. Dai, Y. Cui, and Y. Xue. "On Feasibility of P2P On-Demand Streaming via Empirical VoD User Behavior Analysis", *2008 The 28th International Conference on Distributed Computing Systems Workshops*. 2008, pp. 7–11 (cited on page 42).

[159]J. Deng, G. Tyson, F. Cuadrado, and S. Uhlig. "Internet scale user-generated live video streaming: The Twitch case", *International Conference on Passive and Active Network Measurement*. Springer. 2017, pp. 60–71 (cited on page 42).

[160]W. A. Hamilton, O. Garretson, and A. Kerne. "Streaming on Twitch: Fostering Participatory Communities of Play Within Live Mixed Media", *Proceedings of the 32nd Annual ACM Conference on Human Factors in Computing Systems*. CHI '14. Toronto, Ontario, Canada: ACM, 2014, pp. 1315–1324 (cited on page 42).

[161]J. Deng, F. Cuadrado, G. Tyson, and S. Uhlig. "Behind the game: Exploring the twitch streaming platform", *Network and Systems Support for Games (NetGames), 2015 International Workshop on*. IEEE. 2015, pp. 1–6 (cited on page 42).

[162]M. Siekkinen, E. Masala, and T. Kämäräinen. "A First Look at Quality of Mobile Live Streaming Experience: The Case of Periscope", *Proceedings of the 2016 Internet Measurement Conference*. IMC '16. Santa Monica, California, USA: ACM, 2016, pp. 477–483 (cited on pages 42 and 62).

[163]B. Wang, X. Zhang, G. Wang, H. Zheng, and B. Y. Zhao. "Anatomy of a Personalized Livestreaming System", *Proceedings of the 2016 Internet Measurement Conference*. IMC '16. Santa Monica, California, USA: ACM, 2016, pp. 485–498 (cited on pages 42 and 62).

[164]Z. Li, G. Xie, J. Lin, Y. Jin, M.-A. Kaafar, and K. Salamatian. "On the geographic patterns of a large-scale mobile video-on-demand system", *IEEE INFOCOM 2014 - IEEE Conference on Computer Communications*. 2014, pp. 397–405 (cited on page 42).

[165]A. Brodersen, S. Scellato, and M. Wattenhofer. "YouTube Around the World: Geographic Popularity of Videos", *Proceedings of the 21st International Conference on World Wide Web*. WWW '12. Lyon, France: ACM, 2012, pp. 241–250 (cited on page 42).

[166]M. P. Wittie, V. Pejovic, L. Deek, K. C. Almeroth, and B. Y. Zhao. "Exploiting Locality of Interest in Online Social Networks", *Proceedings of the 6th International COnference*. Co-NEXT '10. Philadelphia, Pennsylvania: ACM, 2010, 25:1–25:12 (cited on page 42).

[167]T. Rodrigues, F. Benevenuto, M. Cha, K. Gummadi, and V. Almeida. “On Word-of-mouth Based Discovery of the Web”, *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference*. IMC ’11. Berlin, Germany: ACM, 2011, pp. 381–396 (cited on page 42).

[168]B. Guidi, M. Conti, A. Passarella, and L. Ricci. “Managing social contents in Decentralized Online Social Networks: A survey”, *Online Social Networks and Media* 7 (2018) (cited on page 43).

[169]G. Silva, L. Reis, A. Terceiro, P. Meirelles, and F. Kon. “Implementing Federated Social Networking: Report from the Trenches”, *OpenSym*. 2017 (cited on page 43).

[170]Ostaus 1.0 Protocol Specification. `https://www.w3.org/community/ostatus/wiki/images/9/93/OStatus_1.0_Draft_2.pdf`. Accessed: 12-07-2020. 2010 (cited on pages 43 and 83).

[171]P. Jones, G. Salgueiro, M Jones, and J Smarr. *WebFinger*. `https://tools.ietf.org/html/rfc7033`. 2013 (cited on page 43).

[172]ActivityPub. `https://www.w3.org/TR/activitypub/`. Accessed: 13-07-2020. 2018 (cited on pages 43 and 83).

[173]ActivityStream. `http://www.w3.org/ns/activitystreams`. Accessed: 13-07-2020. 2017 (cited on page 43).

[174]A. Guy. *Social web protocols*. `https://www.w3.org/TR/social-web-protocols/`. Accessed: 13-07-2020. 2017 (cited on page 43).

[175]S. Taheri-Boshrooyeh, A. Küpçü, and Ö. Özkasap. “Security and privacy of distributed online social networks”, *Distributed Computing Systems Workshops*. 2015 (cited on page 43).

[176]L. Schwittmann, C. Boelmann, M. Wander, and T. Weis. “SoNet–Privacy and Replication in Federated Online Social Networks”, *Distributed Computing Systems Workshops*. 2013 (cited on page 43).

[177]C. Perera, S. Y. Wakenshaw, T. Baarslag, H. Haddadi, A. K. Bandara, R. Mortier, A. Crabtree, I. C. Ng, D. McAuley, and J. Crowcroft. “Valorising the IoT databox: creating value for everyone”, *Transactions on Emerging Telecommunications Technologies* 28.1 (2017) (cited on page 43).

[178]E. Mansour, A. V. Sambra, S. Hawke, M. Zereba, S. Capadisli, A. Ghanem, A. Aboulnaga, and T. Berners-Lee. “A demonstration of the solid platform for social web applications”, *WWW*. 2016 (cited on page 43).

[179]D. Koll, D. Lechler, and X. Fu. “SocialGate: Managing large-scale social data on home gateways”, *IEEE ICNP*. 2017 (cited on page 43).

[180]J. Ugander, B. Karrer, L. Backstrom, and C. Marlow. “The anatomy of the Facebook social graph”, (2011). eprint: `1111.4503` (cited on pages 43 and 87).

[181]H. Kwak, C. Lee, H. Park, and S. Moon. "What is Twitter, a social network or a news media?", *WWW*. 2010 (cited on pages 43, 85 and 87).

[182]M. Cha, H. Haddadi, F. Benevenuto, and P. K. Gummadi. "Measuring user influence in twitter: The million follower fallacy", *ICWSM*. 2010 (cited on pages 43 and 87).

[183]Z. Gilani, R. Farahbakhsh, G. Tyson, and J. Crowcroft. "A Large-scale Behavioural Analysis of Bots and Humans on Twitter", *ACM Transactions on the Web (TWEB)* 13.1 (2019) (cited on page 43).

[184]A. Bielenberg, L. Helm, A. Gentilucci, D. Stefanescu, and H. Zhang. "The growth of Diaspora – A decentralized online social network in the wild", *INFOCOM Workshops*. 2012 (cited on page 43).

[185]M. Zignani, S. Gaito, and G. P. Rossi. "Follow the "Mastodon": Structure and Evolution of a Decentralized Online Social Network", *ICWSM*. 2018 (cited on page 44).

[186]M. Zignani, C. Quadri, A. Galdeman, S. Gaito, and G. P. Rossi. "Mastodon Content Warnings: Inappropriate Contents in a Microblogging Platform", *ICWSM*. 2019 (cited on page 44).

[187]J. Trienes, A. T. Cano, and D. Hiemstra. "Recommending Users: Whom to Follow on Federated Social Networks", (2018). eprint: 1811.09292 (cited on page 44).

[188]C. Cerisara, S. Jafaritazehjani, A. Oluokun, and H. Le. "Multi-task dialog act and sentiment recognition on Mastodon", (2018). eprint: 1807.05013 (cited on page 44).

[189]F. Boccardi, R. W. Heath, A. Lozano, T. L. Marzetta, and P. Popovski. "Five disruptive technology directions for 5G", *IEEE Communications Magazine* 52.2 (2014), pp. 74–80 (cited on page 45).

[190]A. Osseiran, F. Boccardi, V. Braun, K. Kusume, P. Marsch, M. Maternia, O. Queseth, M. Schellmann, H. Schotten, H. Taoka, et al. "Scenarios for 5G mobile and wireless communications: the vision of the METIS project", *IEEE Communications Magazine* 52.5 (2014), pp. 26–35 (cited on page 45).

[191]*White paper: Cisco VNI Forecast and Methodology, 2015-2020*. Tech. rep. Cisco (cited on pages 45 and 61).

[192]K. Chandra, R. V. Prasad, and I. Nimegeers. "An architectural framework for 5G indoor communications", *Wireless Communications and Mobile Computing Conference (IWCMC), 2015 International*. IEEE. 2015, pp. 1144–1149 (cited on page 45).

[193]T. S. Rappaport, S. Sun, R. Mayzus, H. Zhao, Y. Azar, K. Wang, G. N. Wong, J. K. Schulz, M. Samimi, and F. Gutierrez. "Millimeter wave mobile communications for 5G cellular: It will work!", *IEEE access* 1 (2013), pp. 335–349 (cited on pages 45, 47 and 50).

[194]V. Jungnickel, K. Habel, M. Parker, S. Walker, C. Bock, J. F. Riera, V. Marques, and D. Levi. "Software-defined open architecture for front-and backhaul in 5G mobile networks", *Transparent Optical Networks (ICTON), 2014 16th International Conference on*. IEEE. 2014, pp. 1–4 (cited on page 45).

[195]S. Gosselin, F. Moufida, T. Mamouni, J. A. Torrijos, L. Cucala, D. Breuer, E. Weis, F. Geilhardt, D. v. Hugo, E. Bogenfeld, et al. "Fixed and mobile convergence: Needs and solutions", *European Wireless 2014; 20th European Wireless Conference; Proceedings of*. VDE. 2014, pp. 1–6 (cited on page 45).

[196]S. Wang, X. Zhang, Y. Zhang, L. Wang, J. Yang, and W. Wang. "A Survey on Mobile Edge Networks: Convergence of Computing, Caching and Communications", *IEEE Access* 5 (2017), pp. 6757–6779 (cited on pages 47 and 74).

[197]A. Akella, G. Judd, S. Seshan, and P. Steenkiste. "Self-management in Chaotic Wireless Deployments", *Wireless Networks* 13.6 (2007), pp. 737–755 (cited on page 47).

[198]B. Tan and L. Massoulié. "Optimal Content Placement for Peer-to-Peer Video-on-Demand systems", *IEEE/ACM Transactions on Networking* 21.2 (2013), pp. 566–579 (cited on page 47).

[199]A. Abujoda, D. Dietrich, P. Papadimitriou, and A. Sathiaseelan. "Software-defined wireless mesh networks for internet access sharing", *Computer Networks* 93 (2015), pp. 359–372 (cited on pages 47 and 60).

[200]J. Bicket, D. Aguayo, S. Biswas, and R. Morris. "Architecture and evaluation of an unplanned 802.11 b mesh network", *Proceedings of the 11th annual international conference on MobiCom*. ACM. 2005, pp. 31–42 (cited on page 50).

[201]A. Balamash and M. Krunz. "An overview of web caching replacement algorithms", *IEEE Communications Surveys Tutorials* 6.2 (2004), pp. 44–56 (cited on page 54).

[202]D. K. Krishnappa, S. Khemmarat, L. Gao, and M. Zink. "On the Feasibility of Prefetching and Caching for Online TV Services: A Measurement Study on Hulu", *Proceedings of the 12th International Conference on Passive and Active Measurement*. PAM'11. Atlanta, GA: Springer-Verlag, 2011, 72–80 (cited on page 54).

[203]M. Abedi, N. Mokari, M. Reza Javan, and E. A. Jorswieck. "Single or Multiple Frames Content Delivery for Next-Generation Networks?", *IEEE Access* 7 (2019), pp. 152501–152521 (cited on page 58).

[204]A. Gleixner, L. Eifler, T. Gally, G. Gamrath, P. Gemander, R. L. Gottwald, G. Hendel, C. Hojny, T. Koch, M. Miltenberger, B. Müller, M. E. Pfetsch, C. Puchert, D. Rehfeldt, F. Schlösser, F. Serrano, Y. Shinano, J. M. Viernickel, S. Vigerske, D. Weninger, J. T. Witt, and J. Witzig. *The SCIP Optimization Suite 5.0*. Technical Report. Optimization Online, 2017 (cited on page 58).

[205]D. Trossen and G. Parisis. “Designing and realizing an information-centric internet”, *IEEE Communications Magazine* 50.7 (2012) (cited on page 59).

[206]V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard. “Networking named content”, *Proceedings of the 5th ACM CoNEXT*. 2009, pp. 1–12 (cited on page 59).

[207]G. Pallis and A. Vakali. “Insight and perspectives for content delivery networks”, *Communications of the ACM* 49.1 (2006), pp. 101–106 (cited on page 60).

[208]P. Georgopoulos, M. Broadbent, B. Plattner, and N. Race. “Cache as a service: Leveraging sdn to efficiently and transparently support video-on-demand on the last mile”, *IEEE 23rd International Conference on ICCCN*. 2014, pp. 1–9 (cited on page 60).

[209]P. Bourke. *http://paulbourke.net/geometry/polygonmesh/centroid.pdf*. `http://paulbourke.net/geometry/polygonmesh/`. Accessed: 12-07-2020. 1988 (cited on page 65).

[210]T. Vincenty. “Direct and inverse solutions of geodesics on the ellipsoid with application of nested equations”, *Survey review* 23.176 (1975), pp. 88–93 (cited on page 66).

[211]*Under the hood: Broadcasting live video to millions*. Tech. rep. Facebook (cited on page 66).

[212]M. T. Beck, M. Werner, S. Feld, and T. Schimper. “Mobile Edge Computing: A Taxonomy”, *Proc. of the Sixth International Conference on Advances in Future Internet*. 2014, pp. 48–55 (cited on page 74).

[213]P. Mach and Z. Becvar. “Mobile Edge Computing: A Survey on Architecture and Computation Offloading”, *IEEE Communications Surveys Tutorials* 19.3 (2017), pp. 1628–1656 (cited on pages 74 and 102).

[214]B. Nystedt. *Tired of Twitter? Join me on Mastodon.* `https://www.wired.com/story/join-mastodon-twitter-alternative/`. Accessed: 12-07-2020. 2018 (cited on pages 80 and 86).

[215]C. Wilson, B. Boe, A. Sala, K. P. Puttaswamy, and B. Y. Zhao. “User interactions in social networks and their implications”, *ACM EuroSys*. 2009 (cited on page 81).

[216]B. Doerr, M. Fouz, and T. Friedrich. “Why rumors spread fast in social networks”, *Communications of the ACM* 55.6 (2012) (cited on page 81).

[217]G. J. Stigler. “The economies of scale”, *The Journal of Law and Economics* 1 (1958) (cited on page 81).

[218]T. Federation. `https://the-federation.info/`. Accessed: 12-07-2020. 2019 (cited on page 81).

[219]F. network. https://fediverse.network/reports/2018. Accessed: 12-07-2020. 2018 (cited on page 81).

[220]M. Farokhmanesh. *A beginner's guide to Mastodon, the hot new open-source Twitter clone*. https://www.theverge.com/2017/4/7/15183128/mastodon-open-source-twitter-clone-how-to-use. Accessed: 12-07-2020. 2017 (cited on pages 82 and 86).

[221]Mastodon. https://github.com/tootsuite/mastodon. Accessed: 12-07-2020. 2018 (cited on page 82).

[222]Internet Archive. *Twitter Outages*. https://web.archive.org/web/20110828003545/http://stats.pingdom.com/wx4vra365911/23773/2007/02. Accessed: 12-07-2020. 2007 (cited on page 85).

[223]J. Leskovec and J. J. Mcauley. "Learning to discover social circles in ego networks", *NIPS*. 2012 (cited on page 85).

[224]H. Timms and J. Heimans. *Commentary: #DeleteFacebook Is Just the Beginning. Here's the Movement We Could See Next*. http://fortune.com/2018/04/16/delete-facebook-data-privacy-movement/. Accessed: 12-07-2020. 2018 (cited on page 86).

[225]C. Steele. *What Is Mastodon and Will It Kill Twitter?* https://au.pcmag.com/social-networking-1/47343/what-is-mastodon-and-will-it-kill-twitter. Accessed: 12-07-2020. 2017 (cited on page 86).

[226]Z. Gilani, R. Farahbakhsh, G. Tyson, L. Wang, and J. Crowcroft. "Of Bots and Humans (on Twitter)", *ASONAM*. 2017 (cited on page 87).

[227]M. Fayed, P. Krapivsky, J. W. Byers, M. Crovella, D. Finkel, and S. Redner. "On the Emergence of Highly Variable Distributions in the Autonomous System Topology", *SIGCOMM Comput. Commun. Rev.* 33.2 (Apr. 2003), 41–49 (cited on page 88).

[228]A. Lafrance. *The Story of Twitter's Fail Whale*. https://www.theatlantic.com/technology/archive/2015/01/the-story-behind-twitters-fail-whale/384313/. Accessed: 12-07-2020. 2015 (cited on page 94).

[229]Comodo. *Comodo Launches New Digital Certificate Searchable Web Site*. https://bit.ly/2k27p64. Accessed: 12-07-2020. 2015 (cited on page 94).

[230]Let's Encrypt - FAQ. https://letsencrypt.org/docs/faq/. Accessed: 12-07-2020. 2017 (cited on page 94).

[231]CAIDA. *Ranking of Autonomous Systems*. http://as-rank.caida.org/. Accessed: 12-07-2020. 2019 (cited on page 96).

[232]B. Y. Zhao, L. Huang, J. Stribling, S. C. Rhea, A. D. Joseph, and J. D. Kubiatowicz. "Tapestry: A resilient global-scale overlay for service deployment", *IEEE Journal on Selected Areas in Communications* 22.1 (2004) (cited on page 98).

[233]Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief. "Mobile Edge Computing: Survey and Research Outlook", *CoRR* abs/1701.01090 (2017). arXiv: 1701.01090 (cited on page 102).

[234]A. Lertsinsrubtavee, A. Ali, C. Molina-Jimenez, A. Sathiaseelan, and J. Crowcroft. "PiCasso: A lightweight edge computing platform", *2017 IEEE 6th International Conference on Cloud Networking (CloudNet)*. 2017, pp. 1–7 (cited on page 102).

[235]S. Zannettou, B. Bradlyn, E. De Cristofaro, H. Kwak, M. Sirivianos, G. Stringini, and J. Blackburn. "What is Gab: A bastion of free speech or an alt-right echo chamber", *WWW Companion*. 2018 (cited on page 104).

[236]Mastodon. *Statement on Gab's fork of Mastodon*. https://blog.joinmastodon.org/2019/07/statement-on-gabs-fork-of-mastodon/. Accessed: 12-07-2020. 2019 (cited on page 104).