TU WIEN Informatics

Analyse des Internets als Graph

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Software Engineering & Internet Computing

eingereicht von

Thomas Fodor, BSc
Matrikelnummer 11775799

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Privatdoz. Mag.rer.soc.oec. Dipl.-Ing. Dr.techn. Edgar Weippl
Mitwirkung: Dipl.-Ing. Dr.techn. Johanna Ullrich

Wien, 24. Jänner 2023

Thomas Fodor	Edgar Weippl

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

TU WIEN Informatics

Analysing the Internet as a Graph

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Software Engineering & Internet Computing

by

Thomas Fodor, BSc
Registration Number 11775799

to the Faculty of Informatics

at the TU Wien

Advisor: Privatdoz. Mag.rer.soc.oec. Dipl.-Ing. Dr.techn. Edgar Weippl
Assistance: Dipl.-Ing. Dr.techn. Johanna Ullrich

Vienna, 24th January, 2023

Thomas Fodor	Edgar Weippl

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

Erklärung zur Verfassung der Arbeit

Thomas Fodor, BSc

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 24. Jänner 2023

Thomas Fodor

Kurzfassung

In den Anfangszeiten des Internets bestand dieses aus einigen wenigen Geräten, welche sich gegenseitig kannten. Der Datenfluss war leicht nachzuvollziehen. Heutzutage ist das Internet stark angewachsen, was es schwierig macht, die Pfade der Daten zu verstehen. Zudem machte der begrenzte Adressraum von IPv4 den Einsatz von Technologien zur Einsparung von Adressen erforderlich.

In dieser Arbeit analysieren wir das Routing im heutigen Internet. Zu diesem Zweck verwenden wir Datensets aus zwei verschiedenen Quellen (Vienna-Monitor, CAIDA-Monitors) sowohl für IPv4 als auch für IPv6 zu unterschiedlichen Zeitpunkten. Wir konstruieren ein Framework, um die Rohdaten in einen Graphen zu verarbeiten und Knotengrad-Statistiken sowie die Betweenness-Zentralität (betweenness centrality) für die Knoten im Graphen zu berechnen.

Unsere Analyse hat ergeben, dass das Routing für IPv6 zentralisierter ist als jenes für IPv4, allerdings mehr Redundanz auf lokaler Ebene aufweist. Der dezentrale Scan-Ansatz vom CAIDA Datenset führte zu kürzeren Pfaden, jedoch insgesamt zu einer stärkeren Zentralisierung als bei den Scans vom einzelnen Vienna-Monitor. Außerdem haben wir festgestellt, dass es bei IPv4 praktisch keine Veränderungen im Laufe der Zeit gibt, während IPv6 mit jeder Messung eine Tendenz zu mehr Zentralisierung zeigt.

Darüber hinaus haben wir die Zentralisierung speziell anhand des Gini-Koeffizienten quantifiziert. Wir haben festgestellt, dass eingehende Verbindungen weniger konzentriert sind als ausgehende Verbindungen. Die Betweenness-Zentralität zeigt für alle Messungen ein hohes Maß an Zentralisierung.

Abstract

In the early times of the internet, it consisted of a handful of devices which knew each other, and understanding the data flow was trivial. Nowadays, the internet has grown tremendously, making the task of understanding the paths the data is taking difficult. Additionally, the limited address space of IPv4 made the use of technologies necessary to save addresses.

In this work, we analyse routing on today's internet. In order to achieve this, we use datasets from two different sources (Vienna monitor, CAIDA monitors) for both IPv4 and IPv6 from different points in time. We construct a framework for processing the raw data into a graph and calculate degree statistics as well as betweenness centrality for the nodes in the graph.

Our analysis has shown that routing for IPv6 is more centralized than for IPv4, but also employs more redundancy on a local level. The distributed scan approach of CAIDA resulted in shorter paths, yet more centralization than the scans from the single Vienna monitor. We furthermore found that while for IPv4 there is virtually no change over time, IPv6 shows a tendency to more centralization with each measurement.

In addition to that, we quantified centralization specifically using the Gini coefficient. We found that incoming connections are less concentrated than outgoing connections. Betweenness centrality shows a high degree of centralization for all measurements.

Contents

CHAPTER 1

Introduction

Ever since its advent, the internet has been an ever-growing network. In the first years, it consisted of a limited number of computers communicating with each other and was consequently "easily mappable". In 2020, networking systems manufacturer *Cisco* estimated that in 2018, 18.4 billion devices were connected to the internet and predicting this number to grow to 29.3 billion by 2023 [Cis20]. These numbers make understanding the underlying network structure immensely difficult.

Such amounts of devices also bring huge challenges for the utilized protocols. For a long time, IPv4 was the predominant choice for connected devices. However, IPv4 comes with a severe limitation: It is only capable of assigning 4.3 billion unique addresses (even disregarding reserved ranges), which is less than the number of connected devices nowadays. To alleviate this issue, technologies like NAT (Network Address Translation) are used, though these are only temporary and in general unsatisfying fixes.

In 1998, a successor IP standard called IPv6 was devised [DH98b], designed to overcome this major shortcoming of IPv4. It allows for 2^{128} unique IP addresses and thus makes it possible for each and every device connected to the internet to receive its own IP address. Despite its long existence, migrating from IPv4 and IPv6 requires effort, both hardware and software related, which is why the process is moving on relatively slowly. The Google IPv6 Statistics [Goo22] offer a rough overview of global IPv6 adoption: According to this data, by 2022-10-27, 38.02% of Google users access Google services via IPv6.

An important aspect of networking is routing: a sent packet should successfully reach its destination. On its way, the packet is relayed by multiple intermediate hops. Analysing the paths of packets will thus yield valuable insights into the structure of the global internet network, but can potentially also reveal local peculiarities. For example, Maier [Mai21] discovered loops in packet routing using such data. At the same time, with the migration from IPv4 and IPv6, plenty of intermediate and transition technologies are in

use. Such data, when compared to each other, can reveal the level of similarity between these two networks.

While routing in the early internet was easily understandable and mappable (due to its little size), this task became increasingly difficult as the network grew over the years. As such, a "map" of the internet does not exist and can only be constructed through empirical measurements.

In this work, we want to take a deeper look at the internet's routing infrastructure. Therefore, we use existing traceroute collections and process them into a graph in order to analyse the resulting graph using graph-theoretic methods. In doing so, we have to consider and overcome limitations of hardware and time constraints.

There are multiple datasets that can be used. In our work, we use data collected by Maier for his work [Mai21] ("Vienna dataset") as well as data collected by CAIDA using their Ark infrastructure [fAIDAc] [fAIDAd] ("CAIDA dataset"). We transform the collected data into a graph and then compare these graphs.

In Chapter 2, we first give background about the foundations of internet traffic and go into detail about how routing works, as well as providing the necessary understanding for graph terminology and concepts.

We then discuss the methodology applied in our measurements in Chapter 3. Here, we describe the setup used for performing the measurements, the challenges that we encountered, and how we overcame them.

Finally, Chapter 4 presents the results collected by the framework that we developed as a result of the considerations of the previous chapter. We compare the statistics on multiple levels so that we can obtain more insights into the routing structure of the internet.

For comparing the data, we focus on multiple aspects. One of them is the difference between IPv4 and IPv6 within the same dataset. With this, we get an understanding about whether routing is done similarly for both protocols.

Another aspect is the difference between datasets. The Vienna dataset is collected from a single point in a single location, whereas the CAIDA dataset is composed of data collected by several monitors around the world. At the same time, we also want to see whether IPv4 and IPv6 differ in the same ways for both datasets. By comparing these to each other, we find out the degree to which the vantage point makes a difference for routing in the internet.

Lastly, we also focus on the temporal aspect, where we compare measurements from different points in time to understand how routing changes over time. These three aspects taken together give us a temporal and spatial understanding of how routing works.

CHAPTER 2

Background

In this section, we will provide background on the topics that are discussed in the subsequent sections of this work.

2.1 Internet Protocol

The Internet Protocol (IP) is a protocol for transmitting a *"package of bits"* (i.e., a packet) from a source host to a target host, which are potentially situated in different networks. Version 4, referred to as IPv4, has been standardized in RFC 791 [Pos81]. It defines the required function for fulfilling the purpose of delivering the data. Conceptually, it is designed as being one level above the local network protocol and one level below the transport protocol (TCP, UDP ...).

Addressing other hosts in IPv4 is done using a 32-bit address, which is commonly represented in human-readable form as 4 blocks of 8-bit decimal integers (e.g., `123.45.67.89`). If a host wants to forward a packet and does not have a link to the destination host, it passes the packet to a gateway. Gateways then forward data across networks until it reaches the target network, where the data is forwarded to the target host. To enable inter-gateway communication and routing coordination, gateways implement the *Gateway to Gateway Protocol (GGP)*.

The header of a packet sent via IP contains 14 fields, most notably the source address, the destination address, and the *Time To Live (TTL)*. While the standard specifies TTL to be seconds, it has to be decreased by at least one on each host that processes the message. Since routing usually takes milliseconds rather than seconds, this can be used as a way to limit the amount of hosts that a packet passes, after which the packet would be discarded.

The 32-bit addresses of IPv4 allow for $2^{32} = 4,294,967,296$ distinct IP addresses. The *IANA IPv4 Special-Purpose Address Registry* [Aut21] specifies roughly 300 million special

purpose addresses, reducing the total size of the addressable IPv4 space to 4 billion addresses.

In 1998, a successor protocol called IPv6 was specified in RFC 2460 [DH98b]. Some notable changes are the simplification of the header format, improved support for extensions, and a larger addressing space.

The header contains fewer fields, though source, destination, and hop limit are still present. The *TTL* is now called *Hop Limit* and explicitly specified as being an actual limit of hops a packet should be allowed to take, rather than seconds.

Addressing in IPv6 is defined in RFC 2373 [DH98a]. Compared to IPv4, the address space was increased: Addressing is now done using 128-bit addresses, which potentially allows for $2^{128} \approx 3.4 \cdot 10^{38}$ distinct addresses. Addresses are represented as 8 pairs of bytes in hexadecimal form, separated by colons (e.g., `1044:0:0:0:2:300:1FDA:D34D`), whereas one group of zeroes (at most) can be abbreviated as "::" (e.g., `1044::2:300:1FDA:D34D`).

In 2020, networking systems manufacturer *Cisco* published a report [Cis20], estimating that in 2018, 18.4 billion devices were connected to the internet and predicting this number to grow to 29.3 billion by 2023. Technologies to facilitate the sharing of an IP address by multiple servers or services do exist.

For example, Network-Address-Translation (NAT) assigns the same public IP address to all devices in a network. Now, any device inside this network cannot accept any inbound connections and has to initiate any connection it wants to establish, because the devices inside the network are not uniquely identifiable anymore. For web servers, there are techniques like reverse proxies, which allow for multiple websites to be hosted on the same machine while still allowing them to be opened separately. This, however, creates problems with TLS certificates, as the web server does not know which of the domains the user wants to connect to. There is also a solution for this, Server Name Indication (SNI), where the caller includes the domain name in the connection request.

As you can see from these examples alone, these quick fixes build up quickly and the constructs become harder to maintain as time passes, technology advances, and requirements increase. With the ever-growing amount of internet-connected devices, these builds become more and more impractical.

Despite IPv6 having been standardized as far back as 1998, its adoption is still far from complete. Migrating from IPv4 and IPv6 requires effort, both hardware and software related, which is the most likely reason as to why the process is moving on relatively slowly. The Google IPv6 Statistics [Goo22] offer a rough overview of global IPv6 adoption: According to this data, until 2013, only less than 1% of Google users accessed Google services using IPv6. By 2022-10-27, the number increased to 38.02%.

What makes a full transition even more difficult is the fact that many services are only operating from IPv4. Thus, switching directly from IPv4 to IPv6 would result in a loss of connectivity to certain hosts. For this reason, there are mechanisms which should enable a smooth transition to an "IPv6 world".

The simplest solution is to assign both an IPv4 and an IPv6 address to every machine, which is called *Dual Stack*. However, since IPv4 addresses are already scarce (as described above), a more commonly implemented mechanism is *Dual Stack Lite*, where a customer does not get a dedicated IPv4, but shares this address with a number of other customers. While this enables outbound connections from the machine to the internet, it poses an issue for inbound connections, as the IP address is not uniquely assigned to one device. The remote router handling the request can therefore not forward the inbound connection request.

Alternatively, (temporary) mechanisms have been designed to forward IPv6 data to IPv4 targets using special data formats, encapsulations, or tunnels. The idea behind these mechanisms is to simplify the connection between IPv6-only and IPv4-only hosts, while at the same time moving the burden of inter-protocol communication to the router. These standards have proven to be infeasible in practice, though. For instance, the 6to4 transition mechanism, which encapsulated IPv4 data in IPv6 packets, was obsoleted as early as 2015. The reason was that configuration was complicated, leading to transmission failure and lower IPv6 acceptance for network administrators [TC15].

2.2 Routing

The challenge of data transmission across network boundaries is to ensure the data arrives at the destination. Due to the vast size of the internet, however, a gateway does not know all other gateways, which means that the data might need to travel via several gateways before it reaches its destination.

Initially, this was done via the *Gateway to Gateway Protocol (GGP)*, which the IPv4 standard specified as the standard protocol for coordinating routing between two gateways (in 1981). This protocol, however, quickly reached its limits, which is why in 1984, the Exterior Gateway Protocol was defined [Mil84]. Its successor, BGP, has been introduced in 1989 and is still in use as of today (BGP-4, defined in 2006) [RLH06].

2.2.1 Border Gateway Protocol (BGP)

BGP is a mechanism for data exchange between *Autonomous Systems (referred to as AS)*. The standard defines an Autonomous System as a group of routers that is administered by one entity, manages routing within its network using internal protocols, and uses a protocol (for example BGP) to talk to other ASes.

An AS can announce *IP prefixes* and what IPs are accessible inside its network to all ASes to which it has a direct link via an UPDATE message. These, in turn, broadcast this information to all ASes they know. This mechanism could potentially create loops, as the message is broadcast from every AS to every other AS. To avoid this, if any AS appears twice in the propagated route, this route is simply discarded. The received routes are then collected in a table, commonly referred to as BGP table.

Due to this way of propagation, each AS receives information about routes to every announced prefix and knows which neighboring ASes it can ask for passing on information. When an AS receives a packet for a specific destination IP and should relay it forward, it would check the available paths it received and forward it on the path that it deems optimal.

Figure 3.1 shows how an AS advertises routes to the addresses it administers. Arrows show the directions it is propagated to. A red arrow means that the recipient of that message finds a loop in the path and therefore discards it. In this scenario, AS T advertises a prefix for its addresses via an UPDATE message (Step 1). B and D receive that message and now know that there is a path from themselves to T. These ASes then propagate this information to all ASes they know (Step 2). It is important to note that each AS can only announce one path to the neighbors. Should the chosen path change (Step 3), then the new path can be propagated (the newly chosen path is in red font). If a path happens to contain the same AS multiple times (as can be seen in every step from 2 to 5), that route is discarded and not added to the table (denoted by a red arrow).

When a routing request for IP T3.1 is received, the BGP table is checked to see what prefix it belongs to. The AS will pass the packet all the way to T, after which T takes it over and takes care of routing the packet internally.

While an AS can announce a certain path for a prefix, it is not bound to actually route via the announced path. In our example, AS C could either route the request via A or via D, even though it announced that the route via A is the chosen one. As a consequence of this, an AS cannot influence the decision of the next AS in the chain. Given a connection between an arbitrary AS W and an arbitrary AS X, if two possible paths originate from X, the preceding W cannot determine which of them would be taken. Conceptually, X only knows about a single path anyway, since X only announced its chosen path.

This, however, makes analysis of paths that packets actually take challenging. One way to analyse routes is the inspection of BGP tables. Organizations like RIPE NCC have set up collectors in ASes in various locations to collect and dump the BGP routing data from these ASes for public access [RIP22]. Due to the reasons mentioned above, however, we know that BGP routes are not able to represent the actual routes that a packet takes when it is actually sent. The routes can change at the discretion of the involved ASes.

2.2.2 traceroute

While BGP tables can provide an overview over what the intended routing paths for packets are, they do not necessarily represent the actual paths that are taken. To overcome this problem, the route that a packet takes as it travels to the destination can be traced. This is commonly referred to as a *traceroute*.

A traceroute is performed by taking advantage of the TTL or Hop Limit attribute of IP packets. If the limit of a packet is zero, the packet is not forwarded, but an `ICMP Time exceeded` message is returned to the sender. This message contains the IP address

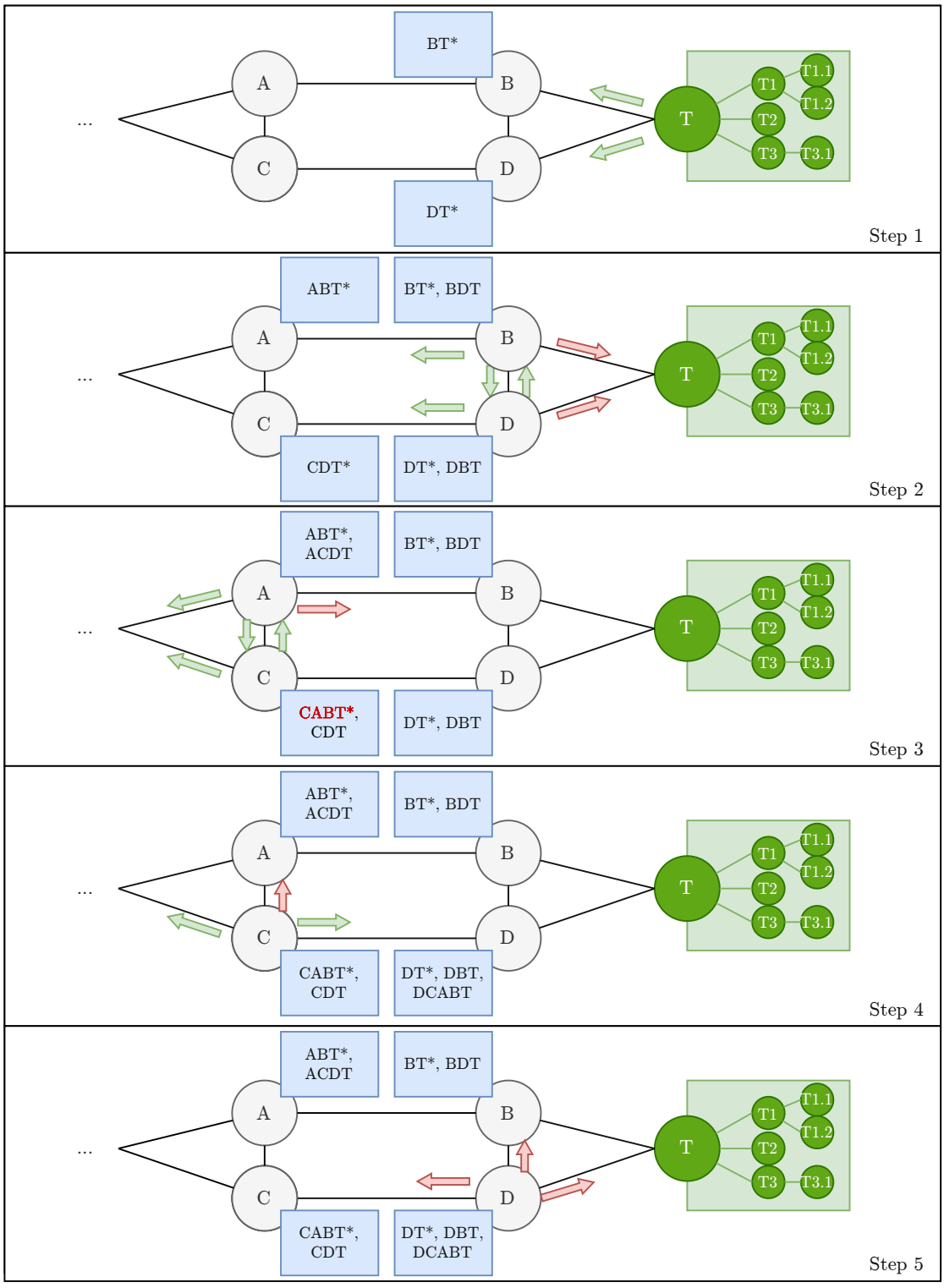


Figure 2.1: Graphical representation of BGP update

of the router where the limit was found to be zero. By starting with a limit of 1 and increasing it by 1 every time, it is possible to obtain a picture of the entire route to the target by collecting the data from the error responses.

Doing this repeatedly reveals the information about the route that is actually taken in practice by a packet sent to a certain destination. The drawback for this method is that the information is not necessarily complete. A router that encounters a packet with a limit of zero is required to drop it, but might not want to reveal its identity to the sender and therefore does not respond to the traceroute request. In our example from above, the administrators of AS T might not want to reveal their internal routing structure, and as such would configure any internal hosts to not respond to traceroute requests.

The IPv4 specification [Pos81] does not mandate that an `ICMP Time exceeded` message needs to be sent, rather it only states that the packet must be destroyed. For IPv6, however, the specification [DH98b] requires that such a message is returned to the sender. In practice, this does not always happen (as evident from the evaluation in Chapter 4). Either way, this leaves holes in the paths for which it is not possible to confidently obtain the IP address.

In their work, Augustin et al. [ACO+06] have researched problems that occur when doing traceroute in the simple way described above. The three issues that were identified were cycles, loops, and diamonds. The most important cause for this are load balancers. A load balancer might route a packet to one and the same host in different ways in order to keep an equal load on the different paths to the same destination.

The authors describe a tool called *Paris traceroute* which aims to overcome (or at least alleviate) the identified problems. To achieve this, they ensure that IP headers which are used for load balancing stay the same between each probe. At the same time, fields that are not used for load balancing are changed such that the packet always has the same size.

The YARRP tool for discovering internet topology makes use of this technique. A classical traceroute needs to traverse the same hosts multiple times: If there is a target at 10 hops distance, then the host at hop 7 will be reached 4 times (once with hop limit 7, once with 8, once with 9, once with 10). This can lead to network congestion or overload. Because of this, YARRP uses a pseudorandom algorithm where not only the target hosts, but also the hop limit is part of the random selection process. Since now the hops are spread apart in time as well, network congestion is way less likely to occur. For more details on this tool, please refer to Section 3.1.1.

2.3 Graph terminology

For our analyses, we will represent the obtained data as a graph. In this section, we want to give a brief overview over the terminology that is used to describe graphs.

A **graph** $G = (V, E)$ is a data structure that is composed of **nodes** or **vertices** (V), representing some entity, and **edges** (E), representing connections or relationships

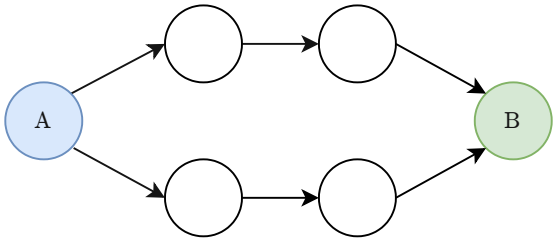


Figure 2.2: Example of a diamond between nodes A and B

between these entities. Edges can be **undirected**, meaning that the relation goes both ways, or **directed**, meaning that the relation goes in the direction of the edge only. A **path** between two nodes is a sequence of nodes that is passed in order by moving along edges between the nodes on the list. A **shortest path**, therefore, is the shortest such path (in case multiple distinct paths exist).

The **neighbors** of a node are nodes which can be reached by traversing one edge. The nodes that come before and after a node on a path are called the **predecessor** and **successor** of that node in the path, respectively.

A directed graph can have **cycles**, which occur when a node is visited multiple times on a path. A cycle where the same node is visited multiple times in a row is called a **loop** (e.g., $A \to B \to B \to B$).

Diamonds occur when there are multiple paths from one node to another node. They are called diamonds because of their shape in graphical representations (Figure 2.2 shows a diamond between Node A and Node B).

A **tree** is a graph which has no cycles and no diamonds. The starting point is called **root**, the ending points are called **leaves**. Even though the data we present describes graphs that do have cycles and diamonds, for simplicity we will still use the terms *root* and *leaf* to refer to the origin(s) and the ending nodes of the directed graph.

Representing data as a graph is especially useful when analysing data flows. Intuitively, the data flows from node to node via the existing edges in the graph. The graph representation enables analysis on a broader level: Graph metrics can reveal the importance of a node relative to the rest of the graph. Calculated over the whole graph, we understand better how information moves in the network.

Another benefit of this is that there are already plenty of algorithms and metrics, which we can easily adapt and use for our purpose here. With large graphs, this is especially important, as algorithm efficiency is the main constraint when dealing with a large data size. For more information about the chosen metrics as well as the rationale of choosing them, please refer to Section 3.4.

CHAPTER 3

Methodology

In this chapter, we will describe the analysed datasets as well as the methods to process the data and obtain the results. Furthermore, we will discuss the challenges associated with this task.

3.1 Datasets

For our analysis, we relied on two different data sources, Vienna scans (performed by the YARRP tool) and scans performed by CAIDA using their Ark infrastructure.

3.1.1 Vienna scans

The first data source makes use of a tracerouting tool called YARRP ("Yelling At Random Routers Progressively") created by Beverly [Bev16]. This tool is specifically designed for large scale probing in order to reduce scanning overhead and prevent network congestion or overloading. For this, it employs Paris Tracerouting as described in Section 2.2.2.

In a previous work, Maier [Mai21] analysed misconfigurations of routers on the internet. For this purpose, traceroute scans were performed and analysed in order to detect routing loops. In our work, we make use of the raw data of the traceroute scans that were performed periodically. Since the machine performing these scans is located in Vienna, we call this datasource the "Vienna scans".

In the IPv4 scans, the monitor collects one trace per IPv4 /24 prefix. A full scan takes about 12 hours and produces about 10 GB of probe data. For IPv6 scans, the monitor collects one trace per IPv6 /48 prefix. These take significantly longer: one full scan takes 19 days and produces about 2.5 TB of data. The size of the IPv6 scan data poses a challenge for data processing which is further discussed in Section 3.5.2.

The scans are provided in the `.yarrp` format, which is a custom textual format of the YARRP tool. For reasons of reliability and avoidance of network congestion, Maier split up the scans into multiple files. Information on each probe is provided on a separate line, values separated by a space. Three values are of relevance for us: the target IP, hop IP and TTL of the probe.

One of the most significant design decisions for the YARRP tool was the decision to not perform the traceroute sequentially, but to probe only one TTL for one route at a time. By doing this, it is possible to parallelize the probes more efficiently, as there are no dependencies on any single probe. Furthermore, by randomizing the probes not just by IP, but also by TTL, network congestion on the target side is significantly reduced, since packets reaching one particular network are not sent at once, but over a longer period of time.

The drawback of this decision is that probes belonging to the same target IP are not necessarily close together in one file (and possibly not even in the same file altogether). According to Beverly [Bev16], he willingly accepted this drawback, since it makes the actual probing more efficient, whereas the gathering of the probes into full traceroutes can be performed offline, after the scan.

In order to bring the probes together into one contiguous path, the entire scan data has to be read into memory and the data grouped by target IP. Due to the size of the scan data, the entire data cannot be held in memory at once. For more details on our approach of processing this data, please refer to Section 3.3.

3.1.2 CAIDA scans

The Center for Applied Internet Data Analysis (CAIDA) is hosting an infrastructure called Archipelago (Ark) [fAIDAa] for performing various internet measurements periodically. As part of these measurements, CAIDA is running frequent traceroute scans (using the *scamper* tool [fAIDAb]) in order to obtain insights into internet topology. For these scans, the Paris Tracerouting is employed, just like for the Vienna scans (see Section 2.2.2).

For the IPv4 dataset [fAIDAc], the monitors send a probe to one randomly selected address from each IPv4 /24 prefix. There are multiple monitors in different locations of the world, which are coordinated in such a way that one probing cycle takes roughly 24 hours. A drawback of this dataset is that only scans older than 1 year are publicly available. Our comparisons across datasets (for IPv4) therefore are limited to data before 2021-10.

For the IPv6 dataset [fAIDAd], the monitors operate similarly—one randomly selected address plus the `::1` address from all announced IPv6 prefixes up to /48, no matter their actual length. A probing cycle in this dataset also takes roughly 24 hours. In contrast to the IPv4 dataset, all IPv6 scans, including the latest ones, are publicly available. The drawback for this dataset, however, is the fact that some monitors might be down (which they often are) as well as the fact that partial routes to unreachable targets

are not present in the dataset. The amount of IP addresses contained in these CAIDA scans is smaller than the amount of IP addresses in a Vienna scan. In a comparison of data from 2022-02, the CAIDA probing cycle (combined for each monitor location) contained 355,398 nodes, whereas a scan in the same time period from the Vienna dataset contained 62,118,844 nodes (only counting nodes with at least one edge). Nonetheless, the information gathered is still useful for high-level comparisons.

The scans are provided in the `.warts` format specifically designed for the *scamper* tool [fAIDA11]. The output files capture (among other information) the IP address and the TTL of a probe, with probes being collected as *traceroute objects*. As these objects contain a path-like representation of the data, they allow us to easily convert the data into graph form. For more information on how the data processing of the raw scan data is done, please refer to Section 3.3.

3.1.3 Dataset comparison

In this section, we want to briefly summarize the properties of the Vienna dataset and the CAIDA dataset which were introduced and detailed in the sections above.

What is common to them is that for both datasets, the measurements are done using the Paris traceroute technique. The IPv4 scans trace the route to a random IP per /24 prefix. For IPv6, the scans trace the route to a random IP per announced prefix of length /48 or shorter, as well as the `::1` address of each such prefix.

The following listing highlights the differences between the two datasets.

Vienna dataset

- Data is collected using the *YARRP* tool [Bev16].
- A single machine in the same location is probing all prefixes in pseudorandom order over the course of 19 days.
- Unsuccessful routes are retained up until the point of failure.
- Reliability for an entire scan guaranteed, as it is just one machine and the scan would not complete otherwise.
- Probes split into multiple text files. Paths need to be put together after the measurement.
- Raw data size is large: 12 GB for IPv4, 641 GB for IPv6.
- Graph data size is large: IPv6 has 62,118,844 connected nodes.
- Availability up to February 2022 (as of writing).

CAIDA dataset

- Data is collected using the *scamper* tool [fAIDAb].
- Multiple machines (monitors) in different locations are probing a randomly assigned portion of prefixes over the course of 24 hours.
- Unsuccessful routes are discarded.
- Reliability is varying, as some monitors might be offline during a cycle.
- One binary file per monitor. Paths are already given in a path representation (i.e., no merging necessary).
- Raw data size is small: 1.9 GB for IPv4, 1.4 GB for IPv6.
- Graph data size is small: IPv6 has 355,398 connected nodes.
- Availability up to one year earlier (IPv4) or the current day (IPv6).

3.2 Representation of the data

The collected routing data is represented in memory as a directed graph. In this graph, each node represents one route hop (and thus one IP address). Each directed edge represents the direction of movement of a packet on a traced route.

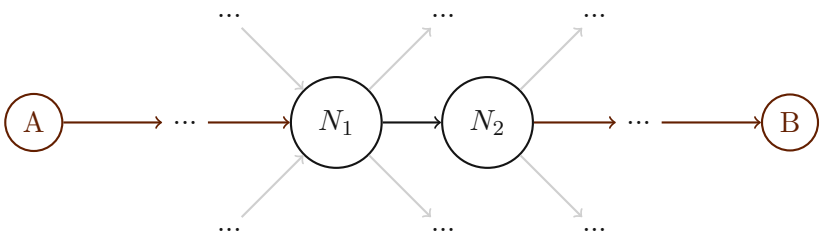


Figure 3.1: Visual representation of a traceroute through the graph

Figure 3.1 represents such a route in the graph. If an edge between a node N_1 and a node N_2 exists, then for *some* paths between A and B, data packets are transferred from N_1 to N_2. The connection does not contain the information as to how often this connection is actually used, only that it exists and appears in at least one path in the dataset.

The graph itself is represented in memory using adjacency lists: For every node in the graph, there is a set which contains all adjacent nodes. Using these lists, the graph can be traversed. The reason for using adjacency lists is to exploit the sparseness of the graph and thus reduce the memory usage. Due to the size of the input data, a dependency matrix is not feasible within any reasonable memory constraints. The Vienna IPv6 scan

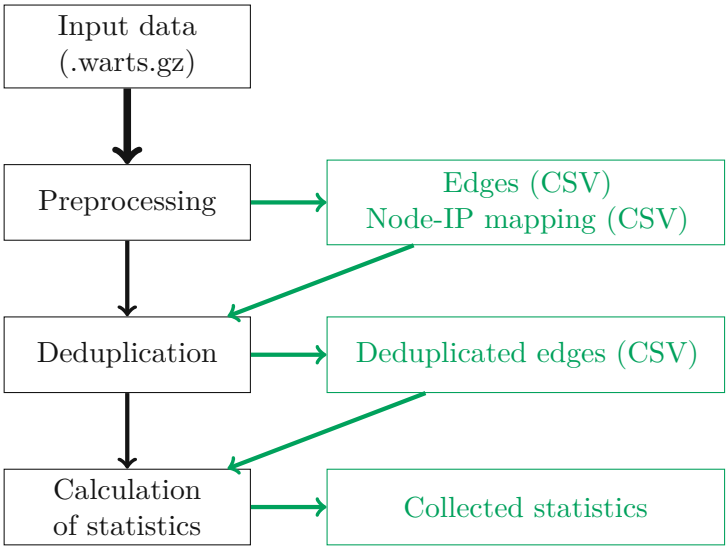


Figure 3.2: Graphical representation of the data processing pipeline for CAIDA scans

of 2022-02 contains 62 million nodes with in-or-out-degree greater than 0. If we assume exactly one bit (indication of true or false) for each element of $Node \times Node$, we would need $62,000,000^2/8$ Bit = 480.5 TB of memory, not counting any potential overhead. Since the graph is sparse as nodes have few connections in general, using adjacency lists results in significant storage and memory savings.

The choice to represent the data as a graph has been made because this is the most intuitive interpretation of the given data. A traceroute is essentially a path from a machine to another, passing other machines on the way, with edges representing a (directed) connection between two of these machines. By putting these paths together, we obtain a partial snapshot of the internet as observed at one specific point in time by a specific set of monitors. By using a graph representation, we are furthermore able to run a graph-theoretic analysis on the data to obtain deeper insights into the network.

3.3 Data processing

As the data is only available as raw traceroute scans, we need to transform the data into the graph form that allows us to run the analyses. Depending on the source format of the data, the pipeline differs slightly.

What is common for both pipelines is that the process has been split up into separate stages, such that any stage can be triggered at any point, as long as the outputs of the previous stage are present.

Figure 3.2 shows how data processing is performed for data obtained by CAIDA scans. A scan consists of several input files, one per each machine that performs the scans (sometimes split into multiple files). The files are read in one by one and processed

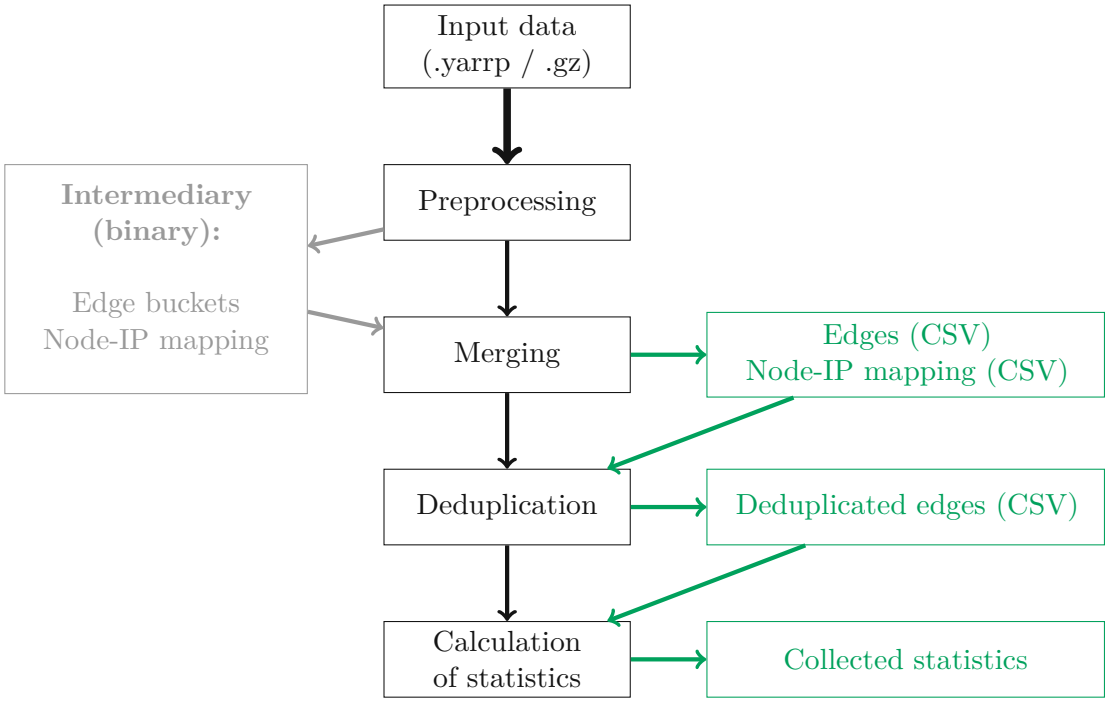


Figure 3.3: Graphical representation of the data processing pipeline for Vienna scans

sequentially. The data is provided in the `.warts` format [fAIDA11], developed by CAIDA for their *scamper* tool. A `.warts` file contains a list of trace routes, and for each of the trace route objects, the list of hop probes is given. In the *Preprocessing* step, an ID is assigned to every encountered IP and then, based on the hop probe list, the edges are computed (i.e., an edge is created between any two hops in the list). The result of this are two CSV files: one containing the mapping between the IP and the assigned IDs, the other one containing the edges.

Figure 3.3 depicts the process for scans originating from the Vienna dataset. A scan in this dataset consists of several files which do not contain collections of traceroutes, but single probes in pseudorandom order, until every TTL was attempted for every target IP [Bev16, p.2]. This adds a challenge to the graph creation: the paths now need to be put together correctly before edges can be generated.

In order to achieve this, we slightly adapt the *Preprocessing* step for the Vienna pipeline and change it to a *Preprocessing-Merging* approach (depicted in Figure 3.4). Just like for the CAIDA scans, we assign a unique 128-bit integer per IP. However, we do not have enough information to generate the edges (i.e., we only know the position of the node in the path, but not its predecessor or successor). For this reason, in the *Preprocessing* step, we store the relevant probe information (hop IP, hop TTL, target IP) in 256 buckets. The bucket is determined by assigning each probe a number from 0 to 255 depending on the target IP, since every full traceroute path has the same target IP. For IPv4, we XOR

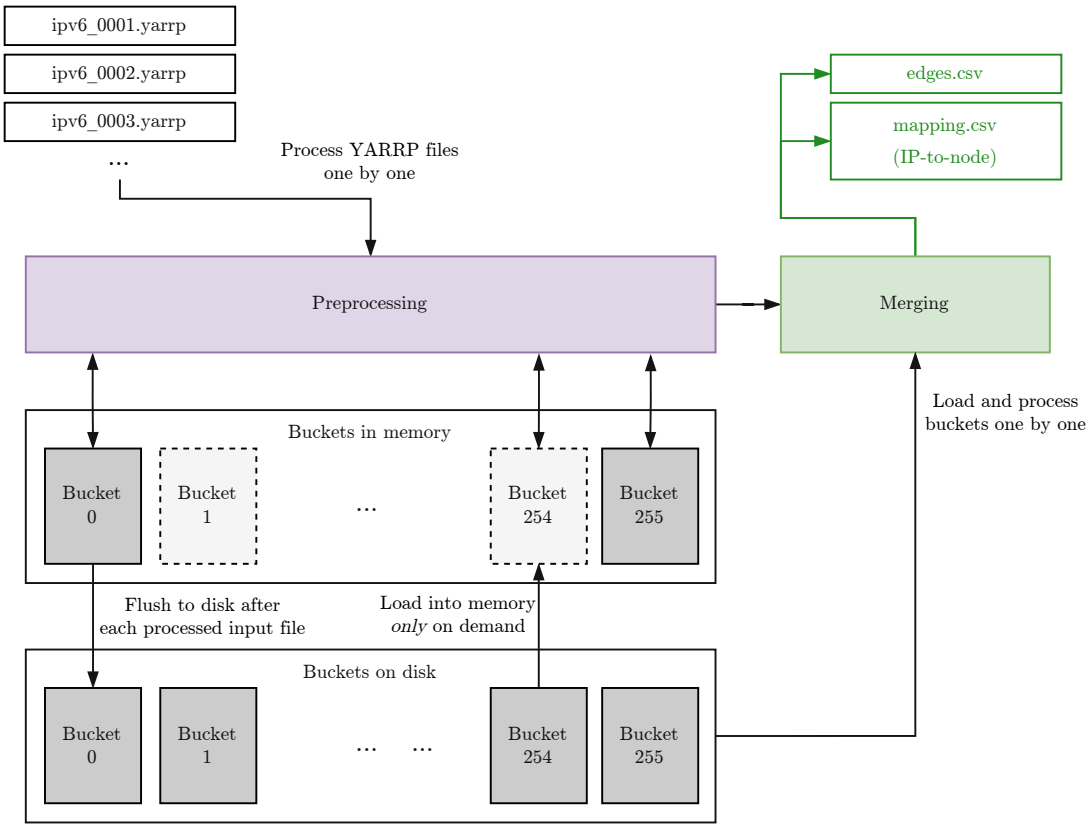


Figure 3.4: Graphical representation of the Preprocessing-Merging steps for Vienna scans

the second and the fourth byte. For IPv6, we XOR the last byte of the public segment (65th-least significant byte) with the last byte of the private segment (the least significant byte). While these might not be perfectly distributed values, they worked well enough for our purposes.

To reduce the amount of memory required at a single point in time, the buckets are flushed to disk in a binary format after processing one `.yarrp` file, after which the data in memory is cleared. The next time a particular bucket is required, it is loaded from disk first (if it exists). The idea behind this approach is that due to the splitting of the data into 256 (more-or-less) evenly sized buckets, it is highly unlikely that one input file requires to access all buckets in memory at once. Even then, should this ever be the case, a mechanism could be implemented which evicts the least used buckets to disk, should the available memory become scarce at any point.

In the *Merging* step, the buckets are read back in again. For any possible traceroute target, the probes are now all present in a single bucket. It is therefore possible to read the buckets in one by one and write the edges into the output file without keeping any

state between the processing of the buckets. With this, only a single bucket is retained in memory at any given time.

The remaining processing pipeline steps are the same for both data sources. In both pipelines, in the *Deduplication* step, the edges are deduplicated in memory, which reduces the storage on disk as well as the memory usage of the graph. The resulting graph is then used to perform the actual statistics calculation in the *Calculation* step.

The actual statistics calculation makes use of the "minimal" graph as computed above and is always kept in memory entirely. For this, the node-to-IP mappings are not considered any further.

For running evaluations on the collected statistics, efficiency or memory constraints are not an issue anymore, as the collected data is limited to a handful of datapoints per node. Because of that, we make use of additional Python scripts for aggregating the data and generating plots.

3.4 Choice of graph metrics

For our analysis, we have focused on two metrics as our main comparison point: Degree and Betweenness Centrality. In this section, we will describe these metrics, as well as discuss metrics that we also considered, but chose to not implement (and also why we chose not to implement them).

3.4.1 Degree

The first metric we consider is *degree*. Essentially, for every node in the graph, we simply count how many *incoming* edges (in-degree) and how many *outgoing* edges (out-degree) the node possesses. This is one of the simplest node metrics and can be easily calculated in a short amount of time.

Even though it is simple, it reveals the position and role of a node in the routing graph. Root nodes have an in-degree of 0 and an out-degree of at least 1, leaf nodes have an in-degree of at least 1 and an out-degree of 0. For the intermediary nodes, we have four different types in the graph. Figure 3.5 shows these four types:

A. **Few edges in, few edges out.** These nodes receive data from the same few source nodes and forward the data to the same few target nodes. These nodes are either passed infrequently by traceroutes, or simply do not know many other nodes. This is exemplified by Node A in the figure.

B. **Many edges in, few edges out.** These nodes receive data from many nodes, but hand it over to a select few. This could be a node at the entrance of a passage (e.g., undersea cable) or at the edge of an AS (handing over traffic to an internal router, which handles internal routing). This is exemplified by Node B in the figure.

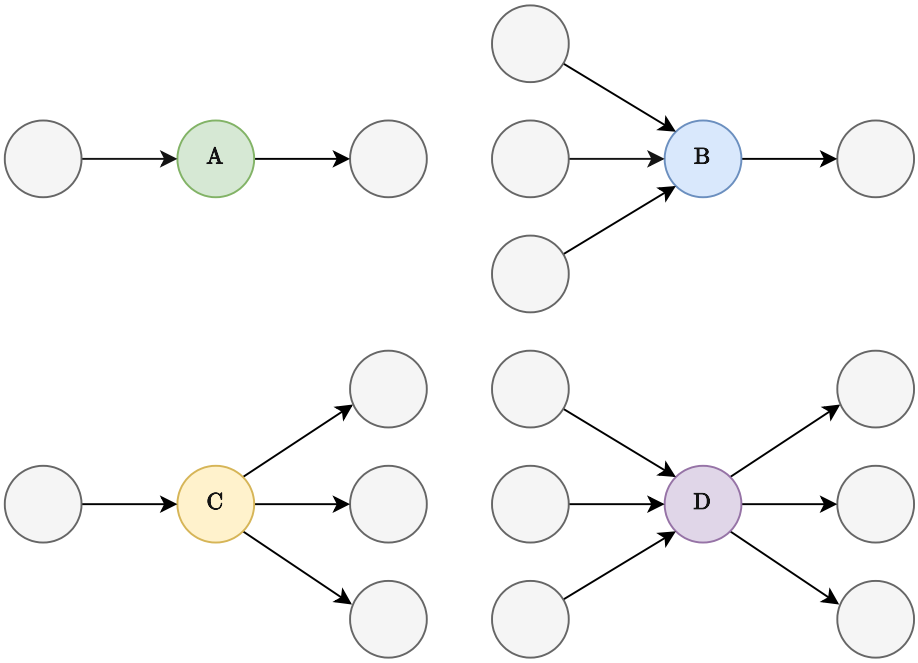


Figure 3.5: Four different types of nodes in the routing graph

C. **Few edges in, many edges out.** These nodes receive data from just a few nodes, but spread it out to many other nodes. This could be a node at the exit of a passage (e.g., undersea cable), or an exit router of the origin ISP. This is exemplified by Node C in the figure.

D. **Many edges in, many edges out.** These nodes receive data from many sources and pass it on to many targets. These nodes are probably central distribution nodes for a region, receiving traffic from several routes and passing it on wherever it should go (e.g., internet exchanges). This is exemplified by Node D in the figure.

Analysing the distribution of node degrees helps us get an understanding of what type of routers the global routing infrastructure is made of. At the same time, we have an approximate measure of importance—we can only consider it approximate, because a low degree node can still be passed many times via the few edges it has.

We can extend this measure by not only counting the degree of the node itself, but also the degrees of its neighbors. We calculate the *Average Neighbor Degree* per node by averaging the degrees of the neighbors. By going one step further, we obtain the *Iterated Average Neighbor Degree*, which calculates the average over the neighbors and the neighbors' neighbors into one value.

In doing this, we extend our view from the direct paths into and out of a single node to the *reach* a node has (and the degree to which it itself is reachable by others). By

combining these metrics in our evaluation, we obtain a better picture of the relevance or importance of a node in the graph rather than just evaluating the degree of the singled out node.

3.4.2 Betweenness centrality

The centrality of a node is, simply put, how important that node is in the graph network. Centrality measures are a quantification of this importance. They can be based on various properties like edge counts, position in the network, or position on paths throughout the network.

Betweenness centrality is a measure that has first been defined by Freeman in 1977 [Fre77]. It is a measure of centrality that is based on the count of the shortest paths it is part of.

For a given node pair (s, t) from the set of all nodes V, there are σ_{st} connecting *shortest* paths. (If $\sigma_{st} = 0$, we do not consider it.) For a given node v, we count how many of these shortest paths contain v (written as $\sigma_{st}(v)$). We do this for every node pair and sum up the results to obtain the betweenness centrality for $C_B(v)$. Brandes [Bra00] summarizes it as follows:

$$C_B(v) = \sum_{s \neq v \neq t \in V} \frac{\sigma_{st}(v)}{\sigma_{st}}$$

The start and end nodes (s, t) are not considered because, trivially, they would be part of any shortest path between s and t. This would add no useful information to the metric. For this reason, the root nodes as well as all leaf nodes, by definition, have a betweenness centrality of 0.

To exemplify this metric, consider Figure 3.6. Here we see a simple graph with five nodes, the shortest paths are listed to the right.

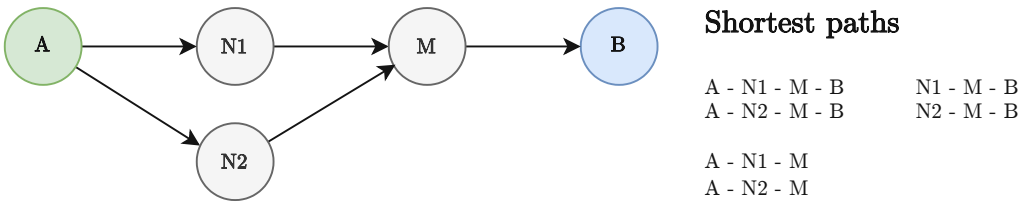


Figure 3.6: Simple graph with list of the shortest paths

As already established, $C_B(A) = C_B(B) = 0$. There are 2 shortest paths between A and B, one passes $N_1 \to M$ and the other passes $N_2 \to M$. Between A and M, we also have 2 shortest paths, one going via N_1 and the other one via N_2. Between N_1 and B there is

only one (via M), same goes for N_2 and B. We now sum up the intermediate results to obtain the final betweenness centrality:

$$
\begin{aligned}
C_B(N_1) &= \frac{\sigma_{AB}(N_1)}{\sigma_{AB}} + \frac{\sigma_{AM}(N_1)}{\sigma_{AM}} && = \frac{1}{2} + \frac{1}{2} && = 1 \\
C_B(N_2) &= \frac{\sigma_{AB}(N_2)}{\sigma_{AB}} + \frac{\sigma_{AM}(N_2)}{\sigma_{AM}} && = \frac{1}{2} + \frac{1}{2} && = 1 \\
C_B(M) &= \frac{\sigma_{AB}(M)}{\sigma_{AB}} + \frac{\sigma_{N_1B}(M)}{\sigma_{N_1B}} + \frac{\sigma_{N_2B}(M)}{\sigma_{N_2B}} && = \frac{1}{1} + \frac{1}{1} + \frac{1}{1} && = 3
\end{aligned}
$$

As we can see, unique paths are counted wholly. If multiple shortest paths exist, then the betweenness is divided among the nodes of the path (just like N_1 and N_2 are only getting $\frac{1}{2}$, as there are 2 paths to the same destination). This allows the betweenness centrality to fall below 1. However, this can only happen for paths of length 2, as Figure 3.7 exemplifies.

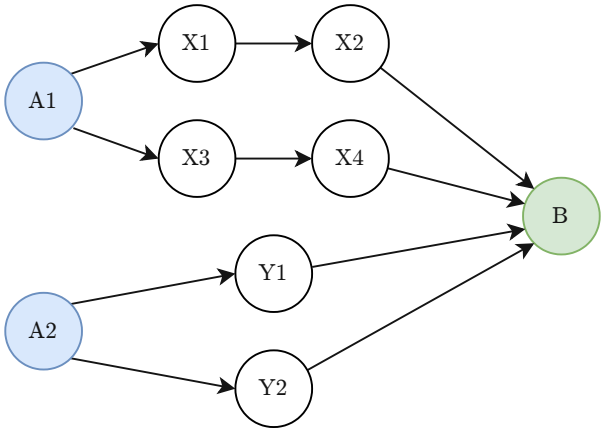


Figure 3.7: Simple graph structure to exemplify a betweenness centrality below 1

There are 2 shortest paths from $A1$ to B, so $X1 - X4$ only get a betweenness of 0.5 for that. However, $X1$ is on the only shortest path from $A1$ to $X2$, causing its betweenness centrality to grow to 1.5. The same is valid for $X2 - X4$ respectively. The only nodes that remain below 1 are $Y1$ and $Y2$, as they are not part of any other shortest path in this graph.

What this essentially means is that if we have a betweenness centrality of below 1, the node is always on one of multiple shortest paths, but never on a single shortest path. Removing $Y1$ would not harm any connections and still keep the shortest path between $A2$ and B intact. Removing any node from $X1 - X4$ does harm connectivity, or forces us to take longer detours between 2 nodes (if any exist). Even so, a value below 1 does not mean that the node is without purpose. In order to build a resilient network, a certain degree of redundancy is needed in case one of the existing paths becomes unavailable.

Another purpose of such nodes could be load balancing in order to not overload a single node with the entire data flow.

As the graph grows larger, the number of paths in the graph grows substantially as well. The betweenness centrality would therefore also grow in the same manner, which makes comparing this metric between graphs difficult. In order to enable comparability of this value, we make use of the *normalized* betweenness centrality.

One way to calculate this is to divide each value by the number of theoretically possible paths to all nodes (i.e., $(|V|-1)(|V|-2)$). This value assumes that the node is present on *every* shortest path to *every* other node, which is a maximum that is almost impossible to reach. We therefore decided to instead divide by the highest betweenness centrality value that is encountered in the graph. Thus, the node (or nodes) with the maximum value would be scaled to 1, while all other nodes would be in the number range $[0, 1)$. This makes our metric more comparable, since the discrepancy between the potential ideal maximum and the actually present maximum would keep growing as the graph grows in size.

Calculating this value is not trivial. We need to know the shortest paths from every node to every other (reachable) node, and then count the occurrences of nodes on these paths. This task is time and space consuming. In 2001, Brandes devised an algorithm [Bra00] which brought the requirements down from $\Theta(n^2)$ space and $\Theta(n^3)$ time to $\mathcal{O}(n+m)$ space and $\mathcal{O}(nm)$ time. While this sounds good in theory, it is important to remember that this is actually required *per node*, which raises the total required time to $\mathcal{O}(n^2m)$.

Luckily, due to the nature of betweenness centrality, it is easily possible to parallelize the algorithm. Calculations of $C_B(n)$ are not completely unrelated to each other because of the algorithm design. However, to merge them, it is sufficient to add them up, since this is exactly what the serial version of the algorithm also does. While this gives us an almost perfect time speedup, it also increases the space requirement. A detailed description of the challenges encountered can be found in Section 3.5.2.

For every node in the node set V, the algorithm performs a breadth-first traversal while keeping track of the shortest paths to nodes as well as the shortest path count. During backtracking, the occurrences of a node on a shortest path are added up and scaled in such a way that at the end of the backtracking, the given node and its dependencies are calculated correctly. In the final step, these values are merged (added) into a shared data structure. In the parallel version, the values are merged into a shared data structure per thread, and the thread data is then added up into one final result array, which now contains the correct and accurate betweenness centrality of all nodes.

To gauge how much control a certain node or AS has over the paths in the graph, we calculate the relation of the betweenness centrality of certain nodes compared to the total sum of the betweenness centrality of all nodes. We refer to this as “controlling” or “holding” $x\%$ of betweenness centrality in the graph that results from the respective measurement data.

3.4.3 Other considered metrics

During our initial evaluation, we considered many metrics and decided on the ones above. In this section, we will go into metrics that were considered, but ultimately not chosen.

Closeness centrality is a measure based on path lengths. For every node, the shortest path to every other node is computed. The number of other nodes (i.e., $|V| - 1$) is then divided by the sum of the lengths of the shortest path to every other node:

$$C_C(v) = \frac{|V| - 1}{\sum_{w \in V} d(v, w)}$$

A perfectly central node would have a distance of 1 to every other node, and so the denominator of the equation above would be equal to the numerator—resulting in the closeness centrality to be 1. The longer the paths are, the larger the denominator, resulting in a decreasing value. The node that is the "farthest" away from all other nodes would have the largest distance sum, resulting in the smallest closeness centrality. Therefore, this metric gives insight about how central a node is in the graph.

We decided against this metric as we are not concerned about how central nodes are in the network. Our graph is directed, and so this metric loses even more meaning as not every node can be reached from every node.

Furthermore, Evans et al. [EC22] have found out and empirically tested that closeness centrality and degree are actually (non-linearly) correlated. They conclude that computing closeness centrality does not add any valuable insight into the graph structure if degrees have already been analysed.

The **Eccentricity** of a node is the longest path that a node v has to any other node. In theory, this helps us to understand whether we have nodes in the graph which are somewhat isolated from the rest of the network. However, our graph is directed, and in a directed graph, this metric is hard to compare between different nodes. Most nodes next to leaf nodes will have a value of 1, whereas starting points will have a large value. At the same time, the existence of a path between two nodes does not necessarily imply that if these nodes connected to each other, they would take this route. In such a case, routing might appear entirely different, as the graph data is based on an entirely different data flow. Therefore, we do not consider this metric useful for our purposes, and the same is valid for all eccentricity-related metrics like radius or diameter.

Neighbor-based metrics investigate the relations between a node and its neighbors. For example, the *Local Clustering Coefficient* of a node measures how "close" its neighbors are to being a clique (i.e., the neighbors are all interconnected with each other). Due to our method of measuring, neighboring relations are incomplete and thus such a measure would not reflect useful information about the graph.

The **number of paths** between two nodes reveals details about how paths are forming in the graph. One possible application for this metric is to count the number of distinct

paths between two important nodes. Due to the setup of our measurements, paths are forming from one (or a few) roots. As discussed in 2.2.1, we would expect that paths between two nodes are usually unique, though in practice, they rarely are. This metric could therefore reveal to us to what extent this occurs.

A challenging aspect of this metric is the selection of important nodes. One could go by collecting some other intermediate metric first and ranking the nodes by that, then selecting the topmost nodes. This two-step approach is a hindrance to comparability across scans, and as such of little value to our scans.

In summary, while there were a few other potentially useful metrics, we decided to focus on the comparison of the selected two metrics. The information from these metrics already reveals important details about the graph structure. The additional metrics, while useful on their own, are either not useful for the scope of our analysis or do not add much information to the information from the already collected statistics. Additionally, they each come with their own difficulties in calculating which require deeper consideration.

3.5 Challenges

In this section, we discuss the challenges of our measurement setup as well as the approaches we have taken to overcome them.

3.5.1 Vantage point bias

One factor to consider for our measurements are the fact that the internet "appears" different to every observer from both a location and a time perspective. The simplest case of a deviation could simply be a machine on a route that stops responding or is taken offline, which causes a route to be cut short at that point or the request being re-routed through other points. Another visible difference between two measurements is the simple replacement of a machine on a route with one of a different IP (and, as a consequence, updating the routing). This would cause the more recent route to slightly differ from the previous route.

These simple factors, however, are not the only influencing factors. The routes between ASes are, in general, determined by the announced BGP tables. Each router will always send the packet on the *best* path available. The definition of what is *best* can differ from router to router and takes into consideration different factors. For example, the Cisco *BGP Best Path Selection Algorithm* [Cis22] allows for customization of the process of finding a path by assigning custom costs to certain routes, thereby altering the definition of "best" according to the specified costs. In essence, this means that the path a packet takes from a given source to a given destination can change every moment.

An example of such a detour is shown in Figure 3.8. In this example, we want to send a packet from both A1 and A2 to B. While a potential path between Node X and Node Y exists, Node X could decide that the path via Y is not the optimal path and send the packet via another route.

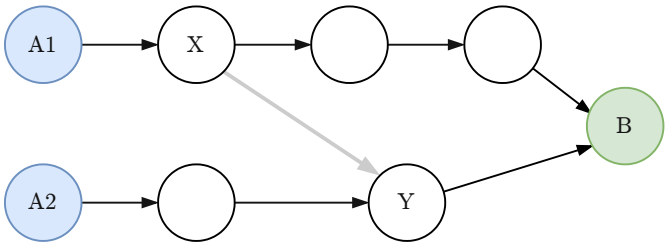


Figure 3.8: Example of a possible alternative route consideration

One reason this might be used is *load balancing*. ASes might announce distinct prefixes for different paths in an attempt to split the incoming load among multiple machines. Bu et al. [BGT04] have discovered that in 2004, prefixes intended for enabling load balancing were responsible for an additional 20% to 25% of prefixes, which indicates that this is a common use case.

Geopolitical conflicts have also shown to have an effect on routing of packets on the internet. In 2021, Limonier et al. investigated the routes which packets take around the conflict zones of Eastern Ukraine [LDP+21]. They performed a traceroute from Donetsk, which lies within the conflict zone, to Moscow, and found that it takes a route through Russia with 3 hops in 13 milliseconds. When performing the same traceroute from Dnipro (which lies to the west of the conflict zone) to Moscow, they found that it does not take the shorter path through Russia. It takes a detour via Germany, Poland, and Belarus before it arrives in Russia, requiring 11 hops and 78 milliseconds to arrive there.

In general, it can be said that a path from one system to another is not constant, but can change due to technological, temporal, geographical or even geopolitical factors.

While the nature of the problem does not allow us to fully counter the problem, we seek to alleviate the problem by taking into consideration measurements from different locations. Other than the measurements from the Vienna dataset, which are done from one single location, we also make use of scans in the CAIDA dataset, which are collected by multiple monitors in various regions all around the globe.

This, in turn, introduces even more volatility between scans, as the monitors are probing different prefixes in a pseudorandom fashion. In effect, this means that one monitor might probe a prefix in one cycle, but not in the next—in the next cycle, this same prefix is probed by a different monitor, potentially in a completely different location.

For this reason, when comparing the data we not only perform comparisons between the dataset, but also against other measurements from the same source.

3.5.2 Data size

Internet scans (specifically IPv6 scans) have a considerable data size, as there are numerous targets that need to be scanned. As such, one main challenge was to enable

processing the data in reasonable time, while also staying within the given hardware constraints (CPU, memory) of our hardware.

CAIDA scans are manageably small. More concretely, the CAIDA IPv4 scan conducted on 2021-09-30 consists of 70 files of 1.9 GB in total (compressed `.warts.gz` format). The IPv6 scan of the same day consists of 45 files of 1.4 GB in total (same format).

The Vienna dataset scans, however, are too large to be processed in a "naive" way. In absolute numbers, while the Vienna IPv4 scan conducted in February 2022 consists of just one file of a compressed size of 2.6 GB (compressed `.yarrp.tgz` format), the IPv6 scan of the same time period consists of 1,101 files of a total of a compressed size of 641 GB (same format).

Our go-to approach of using Python scripts to process the data therefore quickly proved to be unsuitable, as the preprocessing alone would take several days. An implementation in Rust would finish in a few hours, which is why we chose to continue with the Rust implementation.

For the statistics calculation part, however, the data size was too large, which caused the processing machine to run out of resources during the calculation. Most notably, the calculation of betweenness centrality required about 250 GB of RAM to function, and even then, the calculation would still take several months.

An additional challenge was parallelism. Every thread requires a thread-local storage. Alternatively, a shared storage can be used—but then, there is a large synchronization overhead to prevent stale reads or double writes which severely diminishes the benefits of multithreading.

The betweenness centrality calculation was done using the Brandes algorithm [Bra00], which runs in $O(nm)$ and requires $O(n + m)$ space. The algorithm allows "perfect" parallelization as it allows to compute the result for a single node completely independently of all other nodes. At the end, the intermediate results of each thread only have to be added up. However, this also means that each thread requires up to $O(n + m)$ space in memory, which increases the total memory usage significantly and makes multithreading memory-expensive.

With plenty of optimizations added to the code, there was no significant improvement in lowering the expected duration of the calculation. We therefore increased the hardware resources. With 80 (virtual) cores, 900 GB of RAM and 200 GB of swap space, the calculation finished in 30 days.

3.5.3 Unknown hops

In Section 2.2.2, we discussed that some routers are not returning an error message to the sender when they encounter a package with expired TTL or hop limit. When building the graph, however, we now have the problem of unknown connections between two nodes which correspond to no known IP address. Assume that we have a path $A \to X \to B1$.

In this situation, the unknown node does not pose an issue. However, assume that we encounter a *second* path $A \to Y \to B2$ to a different target. We cannot tell whether or not X is equal to Y in this instance.

This is even more problematic when unknown hops appear in succession. Consider Figure 3.9a. In this figure, we want to use three routes, from A to B1, B2, and B3 respectively. After Node X, we encounter three unknown hops, with the next known hop being either one of Y1, Y2, or Y3. The figure presents a routing scheme where the paths split and join multiple times before they reach the next visible hop again.

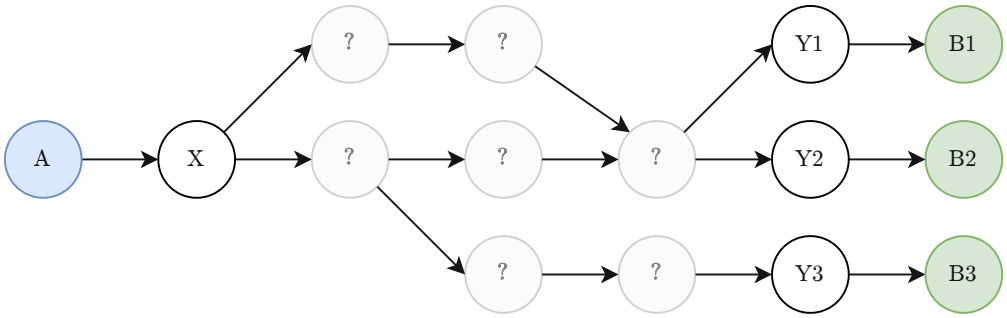
Without feedback from these hops, however, we cannot make complex assumptions about the actual paths between X and Y. We therefore consider two simplifying assumptions about the path:

1. Each hop is an entirely distinct one, i.e., none of the unknown nodes is passed twice and the paths between X and Y are completely disjunct. (Figure 3.9b)
2. Each hop is the exact same for all paths, i.e., the unknown portion is identical for all paths. (Figure 3.9c)

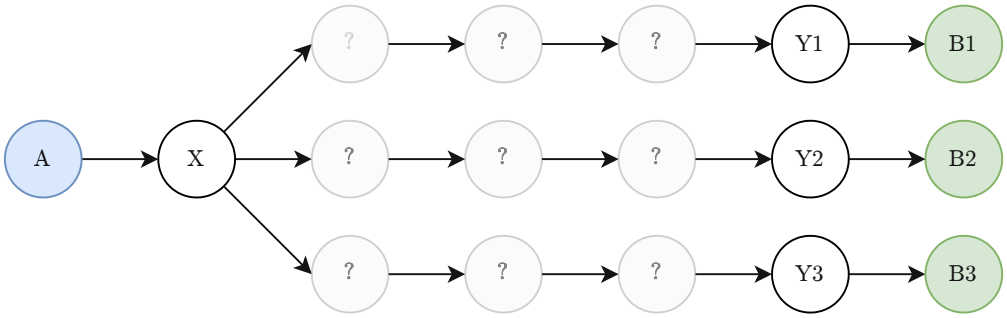
We first considered Option 1 and assumed that every encountered unknown node is unique. This simplifies the processing of the data, as we can simply assign IDs to these nodes and do not need to keep track of assignments. The problem with this approach is that the number of nodes becomes impractical quickly. Consider a scenario where the route to prefixes for a certain geographical region are behind a router that does not identify itself. All paths into those prefixes will cause a new node to be generated in the dataset, causing the graph to grow quickly into unmanageable sizes.

We therefore chose to go with Option 2. When encountering an unknown node on a path $A \to X$, we assign a negative integer to that node based on the previous node A and remember the assignment. Next time we encounter an unknown node $A \to Y$, we make the assumption that Y = X and assign the same ID. The same procedure is applied when the previous node is already a negative one (e.g., $X \to Y \to B$).

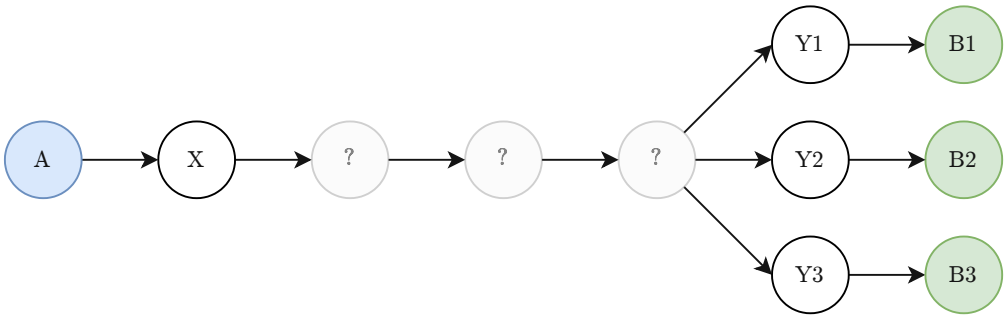
There are two main benefits to assigning negative numbers instead of positive ones. Firstly, it removes the need of having to first count the number of positive nodes before negative nodes can be assigned. Alternatively, both known and unknown nodes could pull IDs from the same (positive) counter, though this would inseparably intermingle known and unknown nodes in the number sequence. This brings us to our second benefit: It is always obvious whether a given node ID is from an identifiable node, or rather the result of our simplifying assumption. It furthermore allows for separate analysis of known and unknown nodes without having to join additional data sources to identify which is which (e.g., node-to-IP mapping).



(a) Missing node hypothesis with more complex routing



(b) Missing node hypothesis with the most amount of nodes (distinct node for every hop)



(c) Missing node hypothesis with the least amount of nodes (the same for all paths)

Figure 3.9: Three different missing node hypotheses

CHAPTER 4

Evaluation

In this chapter, we will analyse the data that we collected using the approaches described in the previous chapter. Before we analyse the results in detail, we want to give an overview over what we are going to analyse in this chapter. We want to focus on 3 dimensions of comparison:

1. Comparison of protocols. We compare the data for IPv4 and IPv6 within each dataset to find out whether routing works differently for the protocols. On top of that, we want to see if the same differences are evident in both datasets.
2. Comparison of datasets. We compare the data across the datasets to find out to what degree the vantage point of the measurement influences the observed routing paths.
3. Comparison over time. Routing paths change over time, as networks are restructured and topologies change. We want to find out to what extent this is visible in the global routing paths.

For these dimensions, we want to compare the data in these ways:

1. We analyse the entirety of the data using (normalized) plots, so that we can spot differences visually and investigate them more closely.
2. We extract a listing of the nodes with the highest betweenness centrality to understand how they relate to the rest of the nodes in the graph.

The included plots all show the Cumulative Distribution Function (CDF) of the values in question. A CDF displays the proportion of data elements (node) that has the given

value or lower. For example, if the plot at 0.6 shows a value of 10^2, that means that 60% of nodes have a value of 10^2 or less, and 40% have one that is higher than that. We make use of CDFs because they allow for easier comparability across datasets of different size, as the data is fitted into the same y-range. On top of that, for the betweenness values, we use a normalized x-axis. As the betweenness centrality value increases with the size of the graph, in order to have a comparable value, we scale by the highest value in the graph. This way, the node with the highest value in the given graph has a fixed normalized value of 1, no matter the data source or protocol.

The following table lists all datasets that were used at any point during the analysis.

Vienna datasets	IPv4	2021-09
		2022-02
	IPv6	2022-02
CAIDA datasets[1]	IPv4	2021-09
	IPv6	2021-09
		2022-02
		2022-09

[1] For CAIDA datasets, we have always collected the last available file per monitor in the given month.

4.1 Comparison of protocols — IPv4 and IPv6

In this section, we compare the data for IPv4 and IPv6 within the same dataset and time period for both of our datasets.

4.1.1 Degrees — Vienna dataset

We begin by comparing the distribution of node degrees in the graphs, taking into account Degree, Average Neighbor Degree, and Iterated Average Neighbor Degree (all of them for both in and out). Figure 4.1 shows the CDFs for these statistics on IPv4 and IPv6. The left column shows the data for connections coming in, the right column for connections going out.

What the plots are not showing are the nodes with both an in-and-out-degree of 0. The Vienna dataset also includes unsuccessful paths. The targets that were not successfully reached by a route are still assigned a node ID, yet are not connected to any other nodes. Table 4.1 shows their number and proportion. Almost all nodes, 99.34% of IPv4 and 98.82% of IPv6 nodes, have both an in-and-out-degree of 0 and are thus not connected to the rest of the graph.

Table 4.1 furthermore shows that out of all the nodes in the graph, almost all were identifiable, yet 0.3% of IPv4 and 0.001% of IPv6 nodes did not report their IP (with the caveat mentioned in Section 3.5.3). As discussed in Section 2.2.2, IPv6 routers are

	IPv4	IPv4 %	IPv6	IPv6 %
Total nodes	12,161,720	100%	5,270,911,453	100%
Identifiable nodes	12,125,128	99.7%	5,270,483,153	99.99%
Unidentifiable nodes	36,592	0.3%	428,300	0.01%
Degree 0 in+out	11,351,154	99.34%	5,208,785,659	98.82%

	IPv4	IPv4 %	IPv6	IPv6 %
Degree >0 in+out	810,566	100%	62,125,794	100%
Identifiable nodes	773,974	95.49%	61,697,494	99.31%
Unidentifiable nodes	36,592	4.51%	428,300	0.69%
Leaf nodes (degree-OUT 0)	390,720	48.2%	5,833,062	9.39%

Table 4.1: Node counts and proportions (Vienna dataset 2022-02)

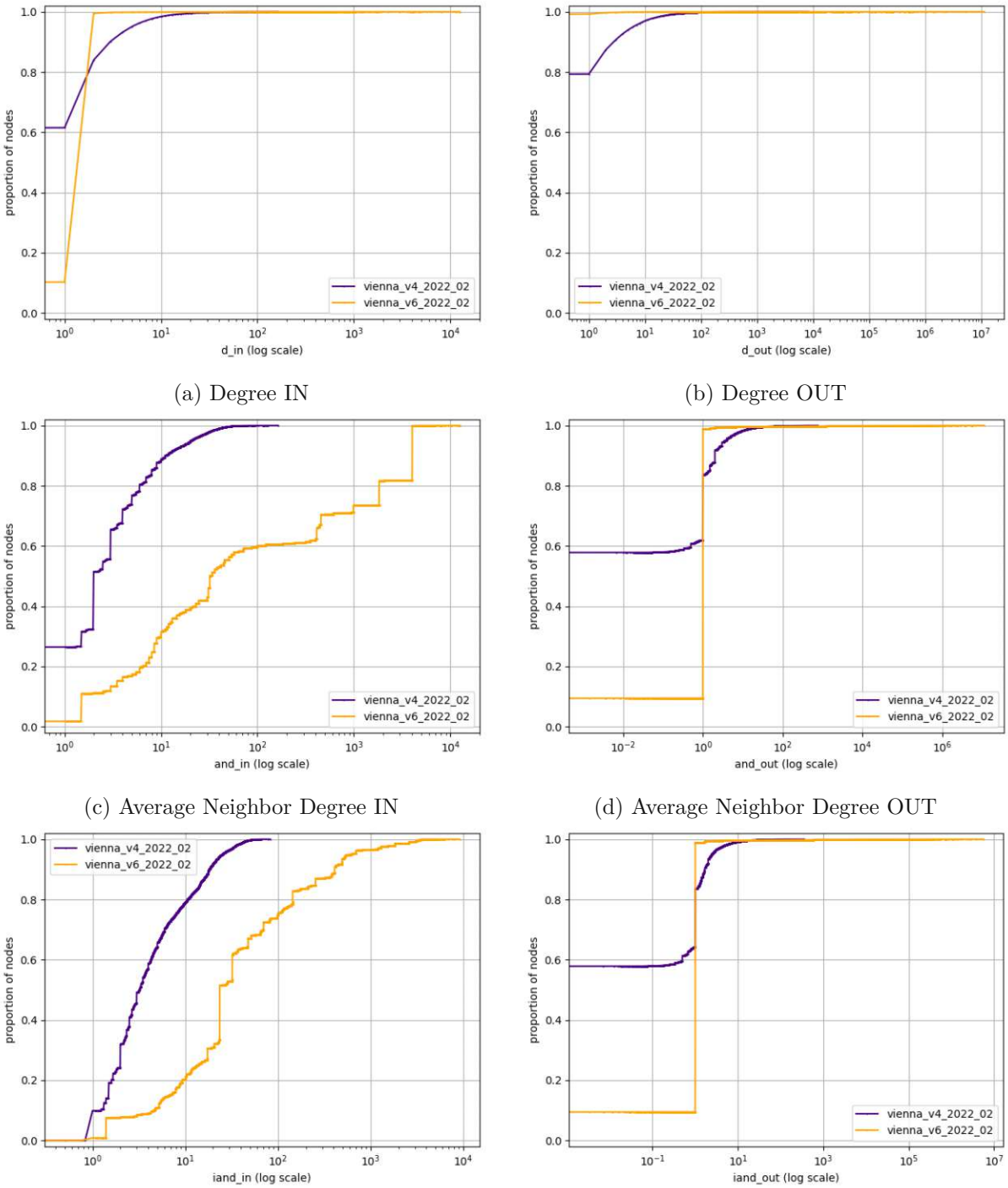
required to return an error to the sender if the packet hop limit reaches 0, though not all routers are doing it. We see that their proportion in the graph is negligible.

If we only consider the nodes that are connected to the graph, we see that the proportion increases more for IPv4 than for IPv6. 4.51% of connected IPv4 nodes do not report their IP. For IPv6, that number still remains well below 1%.

For degree, we notice an extreme discrepancy between the two protocols. Whereas almost half (48.2%) of the IPv4 nodes in the graph are leaf nodes (i.e., they only have incoming, but no outgoing edges), not even 10% of IPv6 nodes (9.39%) in the graph are leaf nodes. This indicates that proportionally, there are many more nodes involved in routing for IPv6. The simplest explanation for this could be load balancing. As the load gets distributed among several nodes, we discover multiple "stops". This might not be the case for IPv4, where IP addresses are limited, and load balancers might share an IP address.

We now analyse the plots in Figure 4.1. Here, we also see a large discrepancy between IPv4 and IPv6. For IPv4, 61.51% of nodes have exactly one inbound edge, after which the curve increases smoothly up until around 30. Following that, the curve is going almost horizontally to the right end, with the maximum value being 164. For IPv6, however, we observe a peculiar difference: Only 10.26% of nodes have an in-degree of 1. However, 89.23% of nodes have an in-degree of 2. This means that almost all nodes in the entire IPv6 graph, 99.49%, have an in-degree of 2 or lower. The highest value recorded here is 12,590. As a result of our measurement setup, there can only be one node with an in-degree of 0 (other than the disconnected nodes), namely the starting point of the measurement.

For the out-degree, the situation appears similar, though with a crucial difference. IPv4 behaves similarly as for in-degree. 79.32% of nodes have an out-degree of 0 or 1 (31.12% with degree 1). For IPv6, we see that 99.26% of nodes have a degree of 0 or 1, of which only 9.39% are of out-degree 0, the rest being of out-degree 1. This, again, is an indicator



(a) Degree IN

(b) Degree OUT

(c) Average Neighbor Degree IN

(d) Average Neighbor Degree OUT

(e) Iterated Average Neighbor Degree IN

(f) Iterated Average Neighbor Degree OUT

Figure 4.1: Comparison of Degree CDFs between IPv4 and IPv6 (Vienna dataset 2022-02)

of a higher involvement of intermediate nodes in IPv6 routing. The reason for this could be a reduced NAT usage due to the larger address space of IPv6.

If we analyse combined degrees, we find that for IPv4, the most common nodes are leaf nodes with 1 edge in (38.52%), followed by 1 edge in, 1 edge out (16.92%) and 2 edges in, 1 edge out (11.45%). This means that two thirds of the graph (66.9%) are made out of simple gateways, merging nodes or leaf nodes.

For IPv6, we see that the most common combination is, in fact, 2 edges in and 1 edge out. This combination alone amounts for 89.13%. The second most common combination, leaf nodes with 1 edge in, only amounts for 9.34%, the rest of combinations being below 1% each. What this means is that if we disregard leaf nodes, virtually the entire graph consists of 2-in-1-out nodes. Since this measurement was taken from a single vantage point, this means that there is a vast number of diamonds in this graph. This also means that almost all nodes in the graph are reachable by at least 2 paths. Again, load balancing could be the reason for this. While it was transparent for IPv4 due to NAT, it is now visible for IPv6, resulting in two detectable entry edges for most nodes.

As for the highest recorded out-degrees, for IPv4 it is 8,180. This is 1.01% of all nodes, meaning that the node with the most outgoing connections in the graph can reach 1.01% of all nodes in the graph with just one hop. For IPv6, the highest out-degree is 11,311,477. This is equivalent to 18.21% of nodes in the graph, meaning that for IPv6, there is one node that has a direct connection to almost a fifth of the nodes that were discovered in the measurement. Proportionally, this is 18 times as much as the top node for IPv4. The second highest, 5,078,686, is at not even half of that.

Next, we analyse the Average Neighbor Degree (AND) (second row in Figure 4.1). IPv4 nodes seem to predominantly have a low in-AND, with the frequency decreasing as the in-AND increases. This means that in general, IPv4 routing nodes do not know many neighboring nodes. In fact, 65.55% of nodes have an in-AND of 3 or lower, meaning that their “incoming” neighbors each have an in-degree of 3 or lower on average. The out-AND findings confirm this: 94.44% have an out-AND of 3 or lower, meaning that on average, their neighbors each have 3 or fewer outgoing connections. What we see here is that out-AND tends to be lower than in-AND. While distributed paths often merge at certain target nodes, few nodes actually distribute the traffic load among different paths.

For IPv6, the situation is different. 61.04% of nodes have an in-AND of 50 or more, meaning that most nodes have incoming connections. These connections have, on average, at least 50 incoming connections themselves, meaning that inbound connectivity is high. At the same time, we see that 98.87% of nodes have an out-AND of 1 or lower. By definition, leaf nodes have an out-AND of 0, as they have no outgoing edges. However, as established previously, these only make up 9.39% of the entire graph. In fact, we see that only 0.03% of nodes have an out-AND that is between 0 and 1 exclusively. Most nodes (89.87%) have an out-AND of exactly 1. Expectedly, we see that 89.39% of nodes in the graph have an out-degree of 1. We conclude that almost all intermediate nodes in our graph have exactly one outbound connection to a node which itself has exactly one

outbound connection. Again, we discover that for IPv6, on a low level, routing seems to be more redundant than for IPv4, where it seems to concentrate on fewer paths.

Analysing the most common values, we see that for IPv4, the top 10 most common values are 20 or lower for in-AND, and 6 or lower for out-AND. For IPv6, the most common in-AND value is 4,028.5, which is shared by 18.17% of nodes in the graph. Another common value is 1,826.5 (8.16%). Three more values are above 100, the rest are 32 or lower. One notable finding is that for out-AND, the 10th most common value (shared by 0.01% of nodes, in absolute numbers 7,572 nodes) is 5,655,739.5. We recall that the highest out-degree in the graph is 11,311,477, the second-highest being 5,078,686.

Since we are evaluating an average, the only reasonable possibility for this to occur is for it to be the result of averaging over 2 nodes: One of them with out-degree 11,311,477 and the other with out-degree 2. It turns out that all of these nodes have an edge going to the mass node, and one edge pointing to themselves (self-loop). As for other values, we already established that the most common out-AND is 1 (89.39%) and the second most common is 0 (9.44%), the rest being below 0.3%. There we find 11245 as well as 6 values between 4300 and 5200, the rest are of value 7 or lower.

Finally, we analyse the Iterated Average Neighbor Degree (IAND) (third row in Figure 4.1). As we now build the average over two hops, we see that for in-IAND, the CDFs of IPv4 and IPv6 grew closer together. We observe two major differences compared to AND: For IPv4, the CDF "begins" earlier on the x-axis. For IPv6, the increase at around 0.6 is missing, as well as the sharp inclines at the end.

The change in the IPv4 plot indicates that there are fewer nodes with an in-IAND in the lower ranges. The raw data confirms this: While 32.41% of nodes have an in-AND of below 2, only 24.22% of nodes have an in-IAND of below 2. This, in turn, means that nodes, which are not connected well to the graph through their direct neighborhood, are generally connected better through their two-hop neighborhood. This also caused the maximum value to drop from 164 for in-AND to 82.5 for in-IAND, indicating that highly-connected nodes are usually not close together.

The change in the IPv6 plot is an indicator for the same phenomenon. The sharp inclines that were high up on the in-AND chart moved further down on the in-IAND chart. In fact, the largest incline (in-AND 4,028.5) moved down to the middle of the plot (23.57). The exact same amount of nodes is included in both (18.17%), and indeed, they are the exact same nodes. The maximum dropped from 763.67 for in-AND to 353.75 for in-IAND, which further confirms this point.

Despite the distribution of in-IAND changing in certain regions compared to in-AND, we observe that the out-IAND is virtually identical to the out-AND graph. The reason for this is the distribution of values: We found 89.30% of nodes with out-IAND of 1 and 9.44% with a value of 0, which is similar to the 89.39% and 9.44% that we found for out-AND. The 1.26% remaining nodes do not leave a large impact on the plot. The maximum value, however, did change from 11,311,477 to 5,655,739. This is easily explained by considering

	IPv4 AVG	IPv6 AVG	IPv4 MAX	IPv6 MAX
Degree IN	2.0288	1.9401	164	12,590
Degree OUT	2.0288	1.9401	8,180	11,311,477
AND IN	5.0202	967.5179	164	12,590
AND OUT	1.2534	944.8912	763.6667	11,311,477
IAND IN	6.8827	157.4636	82.5	9,209.5
IAND OUT	0.8126	22.0393	353.7538	5,655,739

Table 4.2: Average and max values per protocol (Vienna dataset 2022-02)

that the millions of neighbors of the one node with an out-degree of 11,311,477 have lower out-degrees themselves, causing the average to be lowered significantly.

Table 4.2 shows the average and max values of the collected degree statistics per protocol. We observe that in-degree and out-degree are equal, both for IPv4 and IPv6. Furthermore, the value differs between protocols by a value below 0.1. When considering the average degree of the neighbors, the situation changes completely. While degree is almost the same, AND differs significantly. From the IPv6 nodes, 28.90% have a value in-AND above average, and only 0.21% have an out-AND above average. This is due to the fact that the maximum value of out-AND is higher than the one of in-AND (11,311,477 vs. 12,590). These effects even out, which leads to the ANDs being close to each other. This, in essence, means that on average, nodes are similarly well connected, inbound and outbound.

If we consider IAND, we see that for IPv4, the value changes only slightly. Most notably, out-IAND falls below 1, meaning that on average, the nodes in the two-hop neighborhood of a node have an out-degree of below 1. For IPv6, we again see that the value is higher than the IPv4 value, but lower than the AND values for IPv6. This means that high-degree nodes usually have low-degree neighbors. Nonetheless, average connectivity is higher in IPv6 than in IPv4. Even though the plots appears almost identical, it seems that there are fewer high-up outliers, which pushes the average down.

4.1.2 Degrees — CAIDA dataset

We now compare the degree statistics between protocols for the CAIDA dataset of 2021-09.

Table 4.3 shows the amount of nodes in this dataset. Unlike the Vienna dataset comparison, we do not have nodes with degree 0 both in and out, as unsuccessful traceroutes are not recorded by the CAIDA dataset.

As discussed in Section 2.2.2, IPv6 routers are required to return an error to the sender if the packet hop limit reaches 0, though not all routers are doing it. We see that their proportion in the graph is significant: 12.21% of nodes did not return an error packet, which is proportionally more than twice as much as IPv4 (5.75%).

	IPv4	IPv4 %	IPv6	IPv6 %
Total nodes	1,034,262	100%	356,146	100%
Identifiable nodes	974,761	94.25%	312,659	87.79%
Unidentifiable nodes	59,501	5.75%	43,487	12.21%
Leaf nodes (degree-OUT 0)	563,456	51.87%	151,050	42.41%

Table 4.3: Node counts and proportions (CAIDA dataset 2021-09)

Analysing the leaf nodes, 51.87% of IPv4 nodes are leaf nodes, whereas it is only 42.41% of IPv6 nodes. Proportionally, more of the IPv6 nodes are involved in intermediate routing than IPv4 nodes, possibly due to the extended addressing abilities.

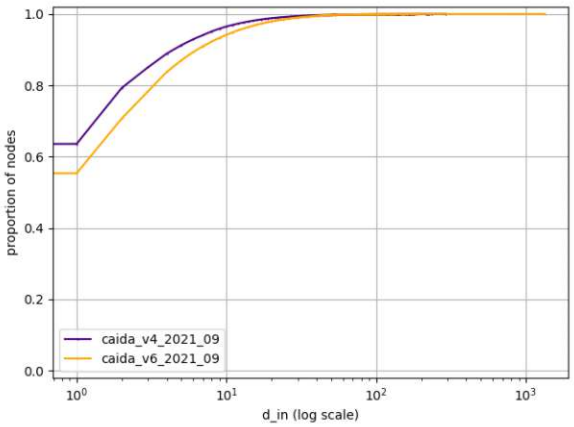
Figure 4.2 shows the CDFs for the collected statistics for IPv4 and IPv6. We observe that the distributions across protocols for every metric, both in and out, are similar. No significant differences between the two measurements can be observed. For degree (in and out), out-AND, and out-IAND, we see that the IPv6 curve starts earlier by 0.1, and converges around the 1.0 mark.

For degree in and out, we see that the majority of nodes has a degree of 0 or 1. There are 66 nodes (IPv4) and 39 nodes (IPv6) with an in-degree of 0—these are our vantage points. The total proportion of nodes with an in-degree of 0 or 1 is 63.58% (IPv4) and 55.34% (IPv6). This proportion is higher for IPv4 than for IPv6. The same is valid for out-degree (77.31% for IPv4 and 68.87% for IPv6).

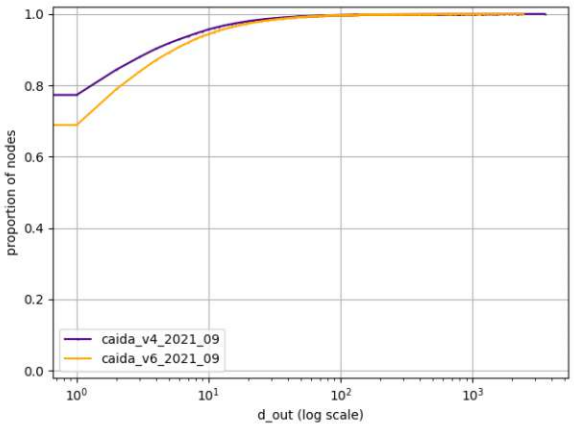
As for AND and IAND, we observe that while in-AND and in-IAND grow steadily until 0.9 and then grow more quickly, out-AND and out-IAND only start around 0.6 and grow steadily until almost 1. However, this seems to only happen between the y-values 0 and 10. While for the in-direction, the nodes in this range are more evenly distributed, for the out-direction, we see that most nodes have a value of 1 or lower.

Table 4.4 lists the average and maximum value per metric and protocol. We see that the maximum out-degree value is higher for IPv4 than for IPv6. This is most likely due to the smaller size of the IPv6 dataset. Other than that, the values are higher for IPv6 *despite* the smaller dataset. Again, we observe that routing seems to be more concentrated towards central points for IPv6. As for the averages, they are similar across the protocols, with only in-IAND being slightly lower for IPv6. Nonetheless, we see that despite the maximum out-degree being higher for IPv4, the average maximum out-degree is actually lower. The average out-AND is only half as much as the IPv6 value. It appears that while IPv4 has the higher-ranking maximum routing nodes, for IPv6, nodes are connected better, as their neighbors have higher-degree neighbors on average. This also means that degrees on the lower end of the spectrum are more common for IPv6.

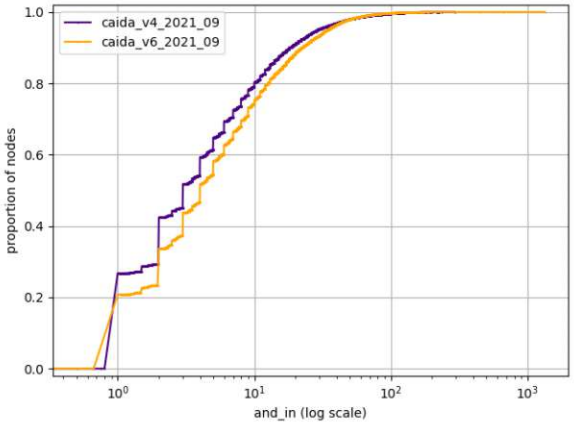
Analysing the top values for AND, we see no surprises. For in-AND, the 20 most common values are all 17 or lower, for out-AND it is 9 or lower (except for the value 13 for IPv6). Regarding in-IAND, we see that all common values are 9 or lower. For out-IAND, the values do not even go above 3. Proportionally, we find that for 84.79% of IPv4 and



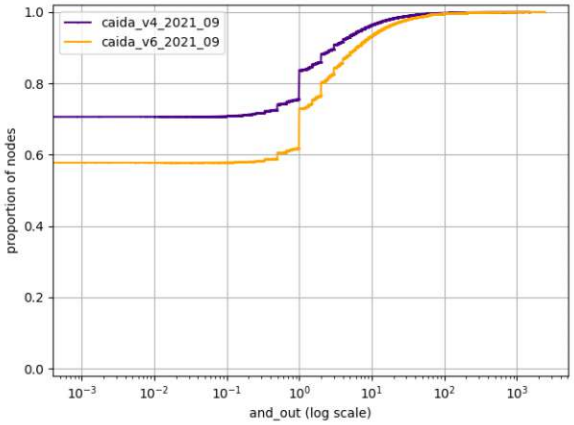
(a) Degree IN



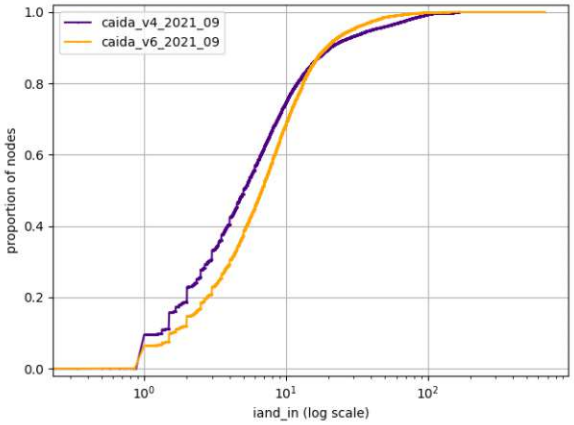
(b) Degree OUT



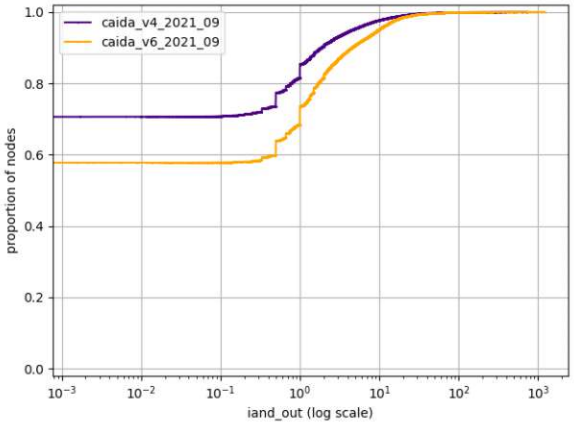
(c) Average Neighbor Degree IN



(d) Average Neighbor Degree OUT



(e) Iterated Average Neighbor Degree IN



(f) Iterated Average Neighbor Degree OUT

Figure 4.2: Comparison of Degree CDFs between IPv4 and IPv6 (CAIDA dataset 2021-09)

	IPv4 AVG	IPv6 AVG	IPv4 MAX	IPv6 MAX
Degree IN	2.5718	3.2790	293	1,338
Degree OUT	2.5718	3.2790	3,532	2,431
AND IN	7.7803	8.9673	293	1,338
AND OUT	1.8935	3.6763	1,501	2,431
IAND IN	10.2280	9.5980	166.1364	669.5
IAND OUT	1.1048	1.9833	727.5	1,216

Table 4.4: Average and max values per dataset (CAIDA scans 2021-09)

76.19% of IPv6 nodes, the two-hop-neighborhood has an average degree of at most 3. We do see, however, that it is significantly lower for IPv6, which explains the higher average. The most common value for out-AND and out-IAND is 0 across both protocols. 70.59% of IPv4 nodes and 57.73% of IPv6 nodes have this value (for both metrics). This includes both the leaf nodes and the nodes shortly before a leaf node, which make up a majority of the nodes.

4.1.3 Betweenness centrality — Vienna dataset

We now analyse the betweenness centrality. Figure 4.3 compares the cumulative distribution of betweenness centrality of the Vienna v4/v6 scans of 2022-02. It does not include or count nodes with a betweenness value of 0, i.e., leaf nodes and unreachable nodes.

Figure 4.3 shows the cumulative distribution functions of IPv4 and IPv6 on top of each other. We observe that the IPv4 plot and the IPv6 plot are appearing similar, the only difference being the scale. Due to the IPv4 dataset having fewer nodes, the betweenness maximum was lower than the one for IPv6. This causes the normalized values at the lower end to be smaller relative to the IPv4 values. For this reason, the bottom displays the scale for the IPv6 curve, the top displays the scale for the IPv4 curve.

For both protocols, we see an almost horizontal progression around 1.0. This means that the nodes with a high betweenness centrality are just a few. This, in turn, means that only a small amount of nodes is in control of a majority of the betweenness centrality.

On IPv4, about 3.1% of nodes have a normalized betweenness higher than $10^{-3.4}$ each, yet control 84.33% of the betweenness centrality in the entire graph. For IPv6, the centralization is even more obvious: Roughly 3.01% of nodes have a normalized betweenness higher than 10^{-5} each, yet control 95.45% of all betweenness centrality, which is 11 percentage points higher than for IPv4.

For a measurement performed from a single location, this is expected. Nodes that control access to certain regions (e.g., the router that sends data via underwater cables to North America) are part of every shortest route to any node in that region. Similarly, any AS on the way from our vantage point to that node would be similarly central, since the shortest path (or paths) to that node is (or are) going to be the same every time.

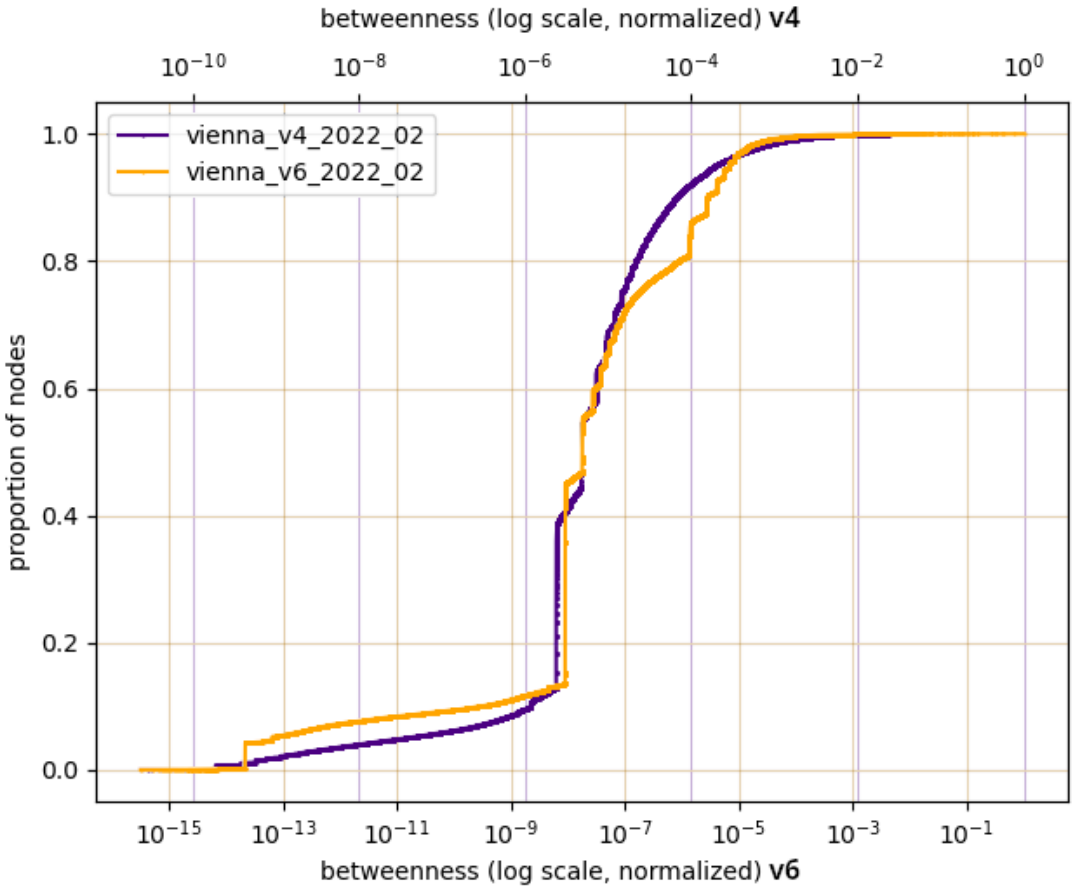


Figure 4.3: Comparison of Betweenness Centrality CDFs between IPv4 and IPv6 with normalized log y-axis (Vienna dataset 2022-02)

In general, if we regard the connection between ASes to be a giant "highway", then the highway nodes and exits are going to be of central importance and appear in many shortest paths. This explains why the few nodes in the upper ranges are central to most of these paths.

Another similarity between the two curves is that they both start almost horizontally, indicating that only few nodes are at the low end of the spectrum (not counting those with betweenness 0). A likely explanation for this phenomenon are nodes on the longer branch of a diamond. Such nodes would only be on few shortest paths (namely the ones to themselves) and thus have a small betweenness centrality. The effect is even stronger, the closer these nodes are to the leaf nodes.

While the IPv4 and IPv6 plots appear similar, we observe a few key differences. One of them is right at the beginning of the curves. While both of them start almost horizontally, at around 10^{-14}, the IPv6 curve is going vertically for about 0.04 (or 4%) of y-values.

Examining the incline closely, it appears that 3.98% of nodes have a (raw) betweenness centrality value of exactly 1 (28,488 out of 715,418). Of those nodes, we find that 98.14% are assigned a negative ID number (27,959 out of 28,488), meaning that these nodes did not respond to the traceroute request. This could be due to nodes close to the end of routes which have been configured not to reveal the internal routing structure. Since such nodes are at the end of the paths (i.e., just one node before the leaf node), their betweenness centrality is 1 (unless they are part of a diamond, which would push their value below 1).

For IPv4, such an incline does not exist; only 0.57% of nodes have a betweenness centrality of 1 (1,810 out of 319,202). Out of those, just 63.87% have a negative ID (compared to 98.14% for IPv6).

For both IP versions, we observe that the majority of nodes has a betweenness centrality around the middle of the range. In fact, 82.64% of IPv4 nodes have a normalized betweenness value between 10^{-6} and 10^{-4}, while 85.93% of IPv6 nodes are between 10^{-9} and 10^{-5}. This most likely represents the wide network of ASes that is not close to the origin point. Those close to the origin point would necessarily get passed often, whereas those further away only get passed when a node closer to them is the target.

As for nodes with an absolute betweenness centrality of below 1, again, we observe a slight difference. In relative values, a betweenness centrality of 1 corresponds to $1.89 \cdot 10^{-10}$ for IPv4 and $2.19 \cdot 10^{-14}$ for IPv6. Observing the graph, they appear to be close to each other. In numbers, this amounts to 0.11% for IPv4 (351 out of 319,202) and 0.24% for IPv6 (1,683 out of 715,148). This is the number of nodes that could be replaced without any imminent negative effects on the routes in the graph. Relatively speaking, IPv6 has twice as many such nodes than IPv4. The only sensible explanation for this is load balancing, where between some node A and B, which are exactly 2 hops apart, the probes were distributed among multiple paths between A to B during the measurement.

For IPv4, 256 out of 351 nodes (72.93%) reported their identity (i.e., had a non-negative node ID). For IPv6, this number amounts to 1,093 out of 1,683 (64.94%), which is similar in relative terms, albeit slightly lower. This means that, at least on this level, the majority of routers used to create multiple paths are configured to reveal their identity to a traceroute probe.

For IPv6 we observe a wider range of betweenness values below 1. The lowest value for an IPv4 node is 0.125, which means that this node is sharing its path with 7 other nodes (as $1/0.125 = 8$, assuming that this node is not part of multiple such paths). For IPv6, we have 334 nodes that have a value below 0.125 (334 out of 1,683 nodes below 1, 19.85%). The lowest value encountered is 0.015 which, assuming no other paths, would mean that this node is on one of the 68 shortest paths for a certain node. Again, this might be due to a lower use of NAT, as the IPv6 address space is larger and allows for more generous assignment of IP addresses.

IP address	CC[1]	Prefix	AS number	NBC[2]
223.120.10.85	HK	223.120.10.0/24	AS5845	1.0
149.38.4.49	US	149.38.0.0/16	AS174	0.5267
84.116.136.118	EU	84.116.0.0/16	AS6830	0.3293
154.54.28.130	??[3]	154.48.0.0/12	AS174	0.2868
84.116.130.165	EU	84.116.0.0/16	AS6830	0.2411
129.250.2.51	US	129.250.0.0/16	AS2914	0.2392
64.125.29.126	US	64.124.0.0/15	AS6461	0.2184
62.115.138.22	EU	62.115.0.0/16	AS1299	0.2088
212.133.7.161	SK	212.133.0.0/17	AS3356	0.2068
*	AT	*	*Outbound ISP*	0.1960
83.231.187.1	GB	83.231.128.0/17	AS2914	0.1896
130.117.51.41	US	130.117.0.0/16	AS174	0.1796
154.54.24.222	??[3]	154.48.0.0/12	AS174	0.1665
*	AT	*	*Outbound ISP*	0.1616
83.167.55.208	FR	83.167.32.0/19	AS8218	0.1460
129.250.2.111	US	129.250.0.0/16	AS2914	0.1411
38.104.164.234	US	38.0.0.0/8	AS174	0.14
213.46.182.18	NL	213.46.160.0/19	AS6830	0.1392
62.115.122.159	EU	62.115.0.0/16	AS1299	0.1363
149.29.9.162	US	149.29.0.0/16	AS174	0.1343

[1] CC: Country Code (as reported by WHOIS)
[2] NBC: Normalized Betweenness Centrality
[3] WHOIS not responding to this query. Most likely US (refer to Table 4.7).

Table 4.5: Top 20 AS with the highest betweenness centrality (IPv4, Vienna dataset 2022-02)

4.1.4 Nodes with the highest betweenness centrality — Vienna dataset

Now we investigate the nodes with the highest betweenness centrality. Table 4.5 lists the 20 nodes with the highest betweenness centrality for IPv4. We used the Linux CLI tool `whois` to query the WHOIS information for the IP and find the prefix, location of the prefix, and the AS number.

Surprisingly, the top node belongs to an AS in HK and has a betweenness centrality which is almost twice as much as the value for the second element in the list. From the 20 nodes, we see that 6 belong to AS174. For 2 of them, the responsible WHOIS server refused to respond, and thus we could not obtain the country code, though the query for other nodes revealed that AS174 is located in the US, making it likely that these nodes are also US-related. Comparing their betweenness values, the top node controls 2.6% of betweenness centrality, whereas the nodes of AS174, taken together, control 3.7%.

From the remaining list, 3 nodes belong to AS6830, and another 3 belong to AS2914. As

we see, 12 out of the 20 nodes belong to 3 distinct ASes only. The nodes of these 3 ASes, together with the top node, achieve a total of 7.05%.

Other than the top node, the prefixes are all located either in Europe or in the US. Considering that the vantage point is in Europe, this is expected. The US nodes are most likely exit nodes for traffic that crossed the ocean via undersea cables.

Only 2 out of the 20 top nodes belong to the outbound ISP and have a normalized betweenness centrality of only 0.196 and 0.1616 respectively. These 2 are part of the internal routing before a packet reaches the outer internet. Given that none of the other high-betweenness nodes are in the same country as the ISP (Austria), we find that there are no Austrian nodes that are central on the shortest path to many targets.

In total, the top 18 nodes (disregarding the 2 unavoidable ISP nodes) hold 12.03% of the betweenness centrality of the graph from this measurement and belong to only 7 distinct ASes, with 12 of these nodes belonging to just 3 ASes.

We now analyse the same data for IPv6. Table 4.6 shows the 20 nodes with the highest betweenness centrality. The first thing we notice is the fact that this time, many more ISP nodes are present. In fact, 11 out of the 20 nodes belong to the outbound ISP prefix. This is already indicative of a more distributed routing infrastructure within the ISP network. It could also simply mean that the routers do not need to "hide" behind a NAT. It could be possible that for IPv4, multiple of the routers found here were behind one IP address in order to save addresses.

Nonetheless, even the topmost 2 ISP nodes have a normalized betweenness of 0.67 and 0.4386 respectively, whereas the first provider node for IPv4 was at 0.196. This can result from the fact that not only the ISP network, but other networks on the global internet distribute traffic more, making single nodes less important than others. However, this seems to contradict the findings we made when analysing the plots, where the distribution seemed to be alike. We therefore deduce that while this might be the case for the most important nodes and for internal ISP routing (at least in this particular instance), this is not valid for routing in the global internet.

There are 9 nodes that do not belong to the outbound ISP. These seem to be distributed broadly, as they belong to 7 different ASes. Due to the ISP nodes, it is not possible to fully compare the results between IPv4 and IPv6, though we already see that routing seems to be more distributed for IPv6, as for IPv4, all 20 nodes belonged to just 7 ASes (with 12 of them belonging to 3 ASes only). Nonetheless, we still find one AS, AS56630, to be controlling 3.31% of betweenness centrality.

In this list, AS174 appears again, but only one time (as opposed to 6 times for IPv4), with a control of 0.39%. AS56630 has the 2 topmost non-ISP nodes. AS6939 also appears twice in this list, though only having 0.55% of the betweenness centrality. Other than European and US prefixes, this time we have 4 Russian prefixes on the list (among which we find the top 3 non-ISP nodes), and we are missing the HK location we found for IPv4.

IP address	CC[1]	Prefix	AS number	NBC[2]
2a06:f900:0:300::2:2	RU	2a06:f900::/36	AS56630	1.0
*	AT	*	*Outbound ISP*	0.67
*	AT	*	*Outbound ISP*	0.4386
2a0c:f540:0:2::26	RU	2a0c:f540::/29	AS39238	0.4072
*	AT	*	*Outbound ISP*	0.3263
*	AT	*	*Outbound ISP*	0.3093
2a06:f900:0:200::2	RU	2a06:f900::/36	AS56630	0.2724
*	AT	*	*Outbound ISP*	0.2661
2001:1900:5:2:2::1db5	US	2001:1900::/32	AS3356	0.1593
2001:550:0:1000::9a36:3aba	US	2001:550::/32	AS174	0.1519
*	AT	*	*Outbound ISP*	0.1318
2001:470:0:2ef::1	US	2001:470::/32	AS6939	0.1287
*	AT	*	*Outbound ISP*	0.1277
*	AT	*	*Outbound ISP*	0.1208
2001:b28::e870:2	RU	2001:b28::/32	AS31500	0.108
*	AT	*	*Outbound ISP*	0.1033
2001:668:0:3::9000:181	DE	2001:668::/48	AS3257	0.1023
*	AT	*	*Outbound ISP*	0.0883
*	AT	*	*Outbound ISP*	0.0846
2001:470:0:54::1	US	2001:470::/32	AS6939	0.0833

[1] CC: Country Code (as reported by WHOIS)
[2] NBC: Normalized Betweenness Centrality

Table 4.6: Top 20 AS with the highest betweenness centrality (IPv6, Vienna dataset 2022-02)

In total, the top 9 nodes (disregarding the outbound ISP nodes) hold 6.27% of the betweenness centrality of the graph from this measurement, belonging to 7 distinct ASes. We observe that on a high level, IPv6 routing seems to be a bit more centralized than IPv4 routing.

From the top 9 nodes alone we can tell that while the general structure and centrality of routing remains similar, the actual paths through the internet graph differ with little overlap to each other.

4.1.5 Betweenness centrality — CAIDA dataset

For another perspective, we examine the betweenness centrality data for the measurements performed by CAIDA. Figure 4.4 compares the cumulative distribution of betweenness centrality of the CAIDA v4/v6 scans of 2021-09. It does not include or count nodes with a betweenness value of 0. Since the CAIDA scans are close to each other in size (IPv4: 464,096, IPv6: 198,370), the graphs can use the same scale, as the relative betweenness values are similar.

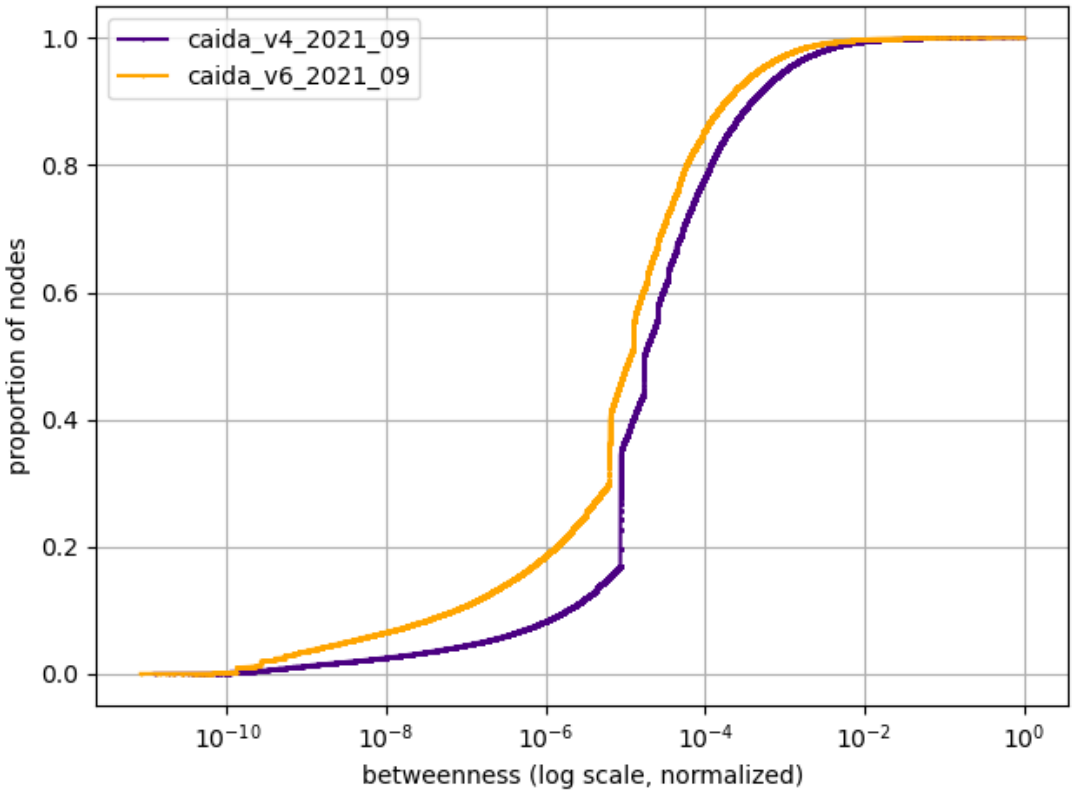


Figure 4.4: Comparison of Betweenness Centrality CDFs between IPv4 and IPv6 with normalized log y-axis (CAIDA dataset 2021-09)

In Figure 4.4 with the plots merged and a logarithmic scale, we again observe that the graphs appear similar in shape. The curves seem to show no abrupt jumps.

As can be seen in the plots, there are not many nodes with a high betweenness centrality. The CDF is progressing almost horizontally between the x-values 10^{-3} and 1. 5.06% of IPv4 nodes and 2.68% of IPv6 nodes have a value that is situated in that range, yet they control 80.36% and 73.98% of the betweenness centrality, respectively.

As these are measurements conducted from multiple vantage points at once (with combined results), this is an indicator that there is a certain set of nodes which seems to be central to the majority of paths. These are most likely central dispatch nodes, as both inbound and outbound traffic would be routed via these nodes. Furthermore, this kind of node would be the only kind that would be passed by routes that originate from multiple monitors.

For the beginning of the graph, we observe that the plots increase smoothly up to about 10^{-5}, after which the plots are moving almost vertically. While the value where it happens is similar, the proportion of nodes for which the change occurs seems to be different:

Whereas for IPv4 it happens after roughly 15% of the nodes, for IPv6 it only occurs after 30%.

The vertical incline that follows is larger for IPv4 than IPv6: While for the former it amounts to roughly 20%, for the latter it is only 14%, though lower, the values are still close to each other. The central part of the plot is similar for both. 86.64% of IPv4 and 78.73% of IPv6 nodes have a betweenness centrality in the logarithmic middle range between 10^{-6} and 10^{-3}.

Finally, we want to analyse the nodes with a betweenness centrality value of below 1. The normalized value for a betweenness of 1 is about 1^{-10} (more precisely, 1.11^{-10} for IPv4 and 1.32^{-10} for IPv6). From the graph, these seem to be close to each other. In absolute numbers, they are close, but the proportion from the total differs greatly. For IPv6, 0.22% of nodes are in this range (427 out of 198,369), whereas for IPv4 it is only 0.04% (203 out of 464,095). IPv4 had less than half as many such nodes as IPv6, while having a total graph size that is more than twice as much. Again, the addressing capabilities of IPv6 are most likely the reason for this, as they enable network administrators to actually install multiple routes via multiple IP addresses, whether for IPv4, such routes would need to be hidden behind NATs due to the low number of IPv4 addresses available.

4.1.6 Nodes with the highest betweenness centrality — CAIDA dataset

In this section, we analyse the top 20 nodes with the highest betweenness centrality for the CAIDA dataset. We again used the `whois` tool to obtain the prefix, location of the prefix, and the AS number of the ASes involved.

We start with the data for IPv4 (Table 4.7). In this table, we see that from the top 20 nodes, 6 nodes belong to an AS in the US. Of these, 4 nodes belong to the same AS (AS6939), among which are also the top 2 nodes of the list. Without regarding any other nodes in the dataset, these 4 nodes alone hold about 1.43% of the total betweenness centrality in the graph and are most likely responsible for central routing in the US.

Similarly, there are 5 prefixes located in Europe, all belonging to AS1299. Together, they hold 1.26% of the betweenness centrality and seem to be the central routing nodes for Europe.

AS174 appears even more often on this list: There are 6 of its nodes in the list, albeit with lower betweenness centrality. These nodes are controlling a total of about 1% of the shortest paths. For some reason, the `whois` service failed to return data for 4 out of the 5 nodes in the 154.54.0.0/16 prefix. If we extrapolate the data returned on the one node that was successfully queried, the prefixes would all be located in the US, which would make 10 out of our 20 top nodes belong to a prefix in the US.

This list contains 15 EU and US nodes in total, though they only belong to 3 ASes. Out of these, 11 belong to only 2 ASes, which is more than half of the nodes in this list.

IP address	CC[1]	Prefix	AS number	NBC[2]
184.105.223.109	US	184.104.0.0/15	AS6939	1.0
72.52.92.85	US	72.52.64.0/18	AS6939	0.8345
62.115.125.163	EU	62.115.0.0/16	AS1299	0.7292
62.115.123.136	EU	62.115.0.0/16	AS1299	0.5503
202.97.29.166	CN	202.97.0.0/19	AS4134	0.5431
154.54.36.253	??[3]	154.48.0.0/12	AS174	0.3266
184.105.65.110	US	184.104.0.0/15	AS6939	0.3114
154.54.59.86	??[3]	154.48.0.0/12	AS174	0.2910
62.115.118.168	EU	62.115.0.0/16	AS1299	0.2859
62.115.125.161	EU	62.115.0.0/16	AS1299	0.2775
154.54.59.185	US	154.54.0.0/16	AS174	0.2742
192.168.1.254[4]	—	192.168.0.0/16	—	0.2612
112.174.66.214	KR	112.160.0.0/11	AS4766	0.2490
154.54.58.5	??[3]	154.48.0.0/12	AS174	0.2482
154.54.36.53	??[3]	154.48.0.0/12	AS174	0.2388
125.144.30.202	KR	125.128.0.0/11	AS4766	0.2384
62.115.122.159	EU	62.115.0.0/16	AS1299	0.2342
85.132.90.158	AZ	85.132.90.0/24	AS29049	0.2269
130.117.1.117	US	130.117.0.0/16	AS174	0.2181
72.52.92.113	US	72.52.64.0/18	AS6939	0.2174

[1] CC: Country Code (as reported by WHOIS)
[2] NBC: Normalized Betweenness Centrality
[3] WHOIS not responding to this query. Most likely US (due to AS174, see also row for `154.54.59.185`)
[4] IP is part of a prefix reserved for network-internal use

Table 4.7: Top 20 AS with the highest betweenness centrality (IPv4, CAIDA dataset 2021-09)

Regarding regions other than US and EU, there is a node in CN (AS4134) with a normalized betweenness centrality of 0.5431, so about 54% of the topmost node. Since no other node from CN is on the list, and this single node has a high betweenness centrality, it might be a strong indicator that high-level routing in China is centralized into one node, whereas lower-level routing is distributed. We have also found 2 KR nodes and one in AZ.

One peculiar thing we observe is that this list contains the IP address 192.168.1.254, which belongs to the 192.168.0.0/16, which is a subnet that is reserved for private use within a network. There are four possible explanations for this phenomenon:

1. It is possible that one of the monitors sent out more probes than other monitors, which caused this node to be on many more of the shortest paths than internal nodes of other monitor networks.

2. The internal networks of many monitors might simply have a node with this IP, causing it to appear often in the dataset.
3. The internal nodes of monitor networks might usually be configured to not respond to traceroute requests of the monitors, yet this one was not configured to behave like that.
4. A public router is mistakenly reporting its internal IP to the traceroute request, rather than its public IP.

By analysing the actual paths through the graph that contain this node, we were able to find paths with length greater than 30 both incoming and outgoing, which would be an indicator for point 4. There are 7 incoming and 31 outgoing edges, both of which also contain edges to both internal-reserved and external IPs, ruling out point 3. Evaluating the `whois` data for the direct connections to this node, we find that we have 2 inbound connections from KR and one from RW. As for outgoing connections, there are connections to RW, CA, GB, and US. The wide range of direct connections from opposite ends of the world is an indicator for point 2. A combination of the points 2 and 4 therefore seems likely.

In total, the top 19 nodes (disregarding the local IP address) hold 4.42% of the betweenness centrality of the graph from this measurement, belonging to 6 distinct ASes, with 15 of these nodes belonging to just 3 ASes.

We now analyse the top 20 nodes for IPv6 (Table 4.8). We see that unlike IPv4, the distribution is uneven: The first 2 nodes have a normalized value of 1.0 and 0.8486 respectively, whereas the third one has a lower value, 0.3781, less than half of the second one. Evaluating the `whois` data, we find that both of these top nodes belong to an AS which is managed by Amazon. The nodes that are assigned to both AS16509 and AS38895 belong to a Singaporean prefix, whereas those that are only assigned to AS16509 belong to US prefixes. In total, the 6 nodes managed by Amazon control 7.39% of the betweenness centrality in this IPv6 measurement. For comparison, on IPv4, the AS with the most influence held 1.43% of the betweenness centrality, which is lower. At the same time, we find neither AS16509, nor Amazon in general, in the top 20 IPv4 nodes. It is possible that the monitors are hosted on Amazon Web Services and thus be hit often with every outbound connection, which would explain their frequent appearance in this list.

The second most common AS is AS1299 with 4 prefixes in EU, holding 2.16% of the betweenness centrality. This AS is also a top contender for IPv4, where its nodes amount to 1.26% of the betweenness centrality. For IPv6, it seems to have a higher degree of control in the network by about 71% (0.9 percentage points). However, it is just about 29% of the control that AS16509 exhibits over the shortest paths in the graph.

AS6939 also appears 4 times in the list with US prefixes, holding a total of 1.6% of the betweenness centrality. Finally, we have 3 nodes in CN from 2 different ASes (1.49%) and one more US node (AS15169, 0.32%).

IP address	CC[1]	Prefix	AS number	NBC[2]
2400:6500:0:2::c	SG	2400:6500::/32	AS16509 / AS38895	1.0
2a01:578:0:13::8	US	2a01:578::/29	AS16509	0.8486
2001:438:ffff::407d:1d43	US	2001:438::/32	AS17025 / AS6461	0.3781
2600:9000:fff:ff00::300	US	2600:9000::/28	AS16509	0.37
2001:2034:1:b8::1	EU	2001:2030::/28	AS1299	0.3316
2408:8142:6000:f401:1::1	CN	2408:8000::/20	AS4837	0.2512
2600:9000:fff:ff00::401	US	2600:9000::/28	AS16509	0.2429
2001:2034:1:7a::1	EU	2001:2030::/28	AS1299	0.2337
2400:6500:0:2::2	SG	2400:6500::/32	AS16509 / AS38895	0.2232
2001:438:ffff::407d:1d54	US	2001:438::/32	AS17025 / AS6461	0.2106
2408:8000:9000:10::3	CN	2408:8000::/20	AS4837	0.1740
2001:470:0:4e4::2	US	2001:470::/32	AS6939	0.1721
2001:470:0:2cf::1	US	2001:470::/32	AS6939	0.1531
2409:8080:0:4:2c5:2f5:2:1	CN	2409:8000::/20	AS9808	0.1456
2001:470:0:52d::2	US	2001:470::/32	AS6939	0.1445
2001:470:0:404::1	US	2001:470::/32	AS6939	0.1409
2620:107:4000:9006::12	US	2620:107:4000::/44	AS16509	0.1403
2001:2034:1:73::1	EU	2001:2030::/28	AS1299	0.1315
2001:2034:1:6b::1	EU	2001:2030::/28	AS1299	0.128
2001:4860:1:1::1503	US	2001:4860::/32	AS15169	0.1252

[1] CC: Country Code (as reported by WHOIS)
[2] NBC: Normalized Betweenness Centrality

Table 4.8: Top 20 AS with the highest betweenness centrality (IPv6, CAIDA dataset 2021-09)

In total, the top 20 nodes hold 14.5% of the betweenness centrality of the graph from this measurement, belonging to 7 distinct ASes (or 9, if you count the double-AS assignments). Again, we see that on a high level, IPv6 routing is more centralized than IPv4 routing.

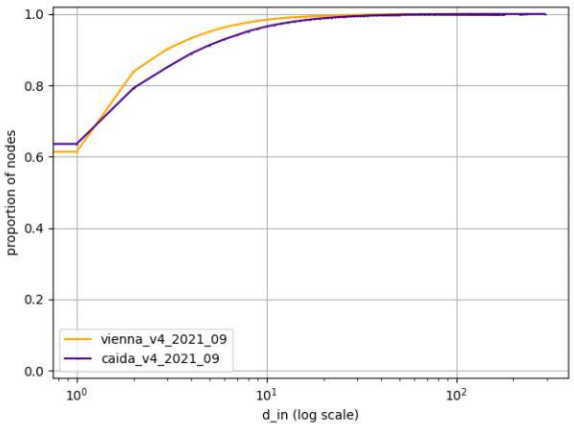
4.2 Comparison of datasets — Vienna and CAIDA dataset

In this section, we compare the data across the different datasets (Vienna datasets and CAIDA datasets) for the same protocol and time period.

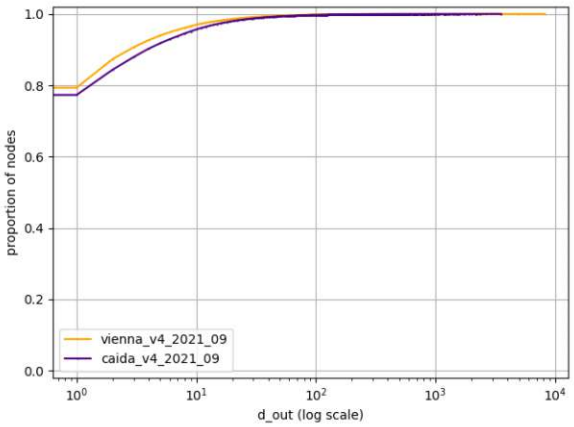
4.2.1 Degrees — IPv4

We begin again by comparing the degree statistics and start with the statistics for IPv4. Both the Vienna and CAIDA data were collected in 2021-09 to ensure the result being comparable.

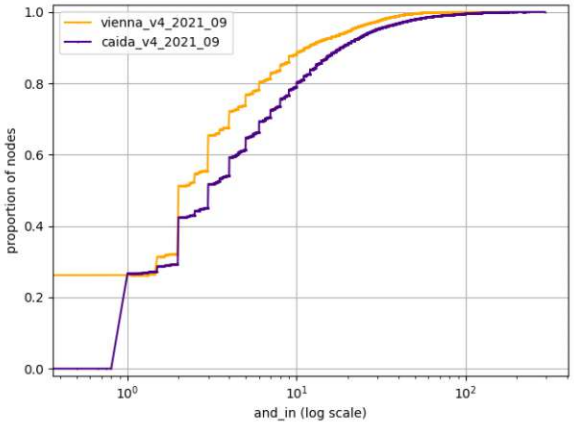
Table 4.9 shows the node counts in the datasets. We see that from the collected nodes, about 95% were identifiable and 5% unidentifiable. This holds true for both datasets.



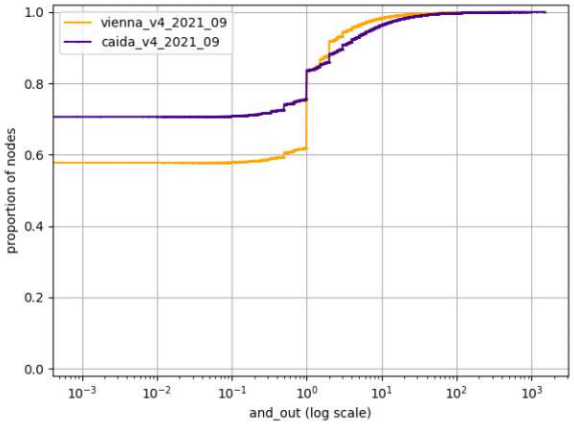
(a) Degree IN



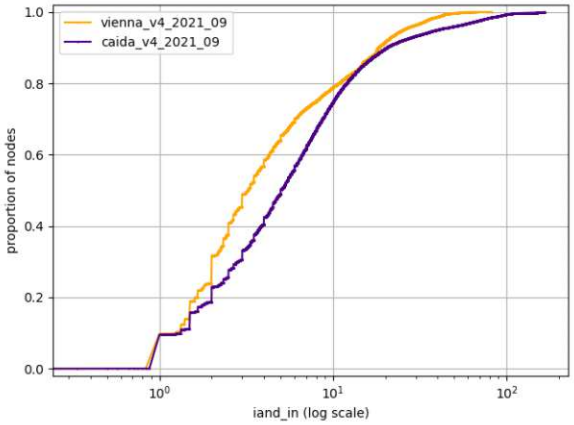
(b) Degree OUT



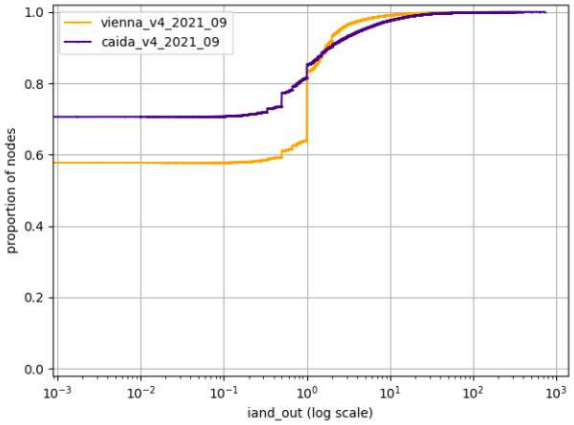
(c) Average Neighbor Degree IN



(d) Average Neighbor Degree OUT



(e) Iterated Average Neighbor Degree IN



(f) Iterated Average Neighbor Degree OUT

Figure 4.5: Comparison of Degree CDFs between datasets (IPv4 scans 2021-09)

	Vienna	Vienna %	CAIDA	CAIDA %
Total nodes (Degree >0 in+out)	826,513	100%	1,034,262	100%
Identifiable nodes	789,041	95.47%	974,761	94.25%
Unidentifiable nodes	37,472	4.53%	59,501	5.75%
Leaf nodes (degree-OUT 0)	398,268	47.46%	563,456	51.87%

Table 4.9: Node counts and proportions (IPv4 scans 2021-09)

The number of leaf nodes is also comparable: About 50% of nodes in the graph are leaf nodes in both datasets. We see that in this regard, the datasets differ only slightly. Note that in the table for the Vienna dataset, we only included the nodes that are connected to the graph (in- or out-degree of at least 1). We do not list them here, since the CAIDA dataset does not include them.

Figure 4.5 shows the comparison of the degree statistics for the two datasets. Evaluating the degrees (first row), we see that they are almost identical in every aspect. For the Vienna dataset, there are slightly fewer nodes with an in-degree of 1 lower (61.38%, compared to 63.58% for CAIDA). For out-degree, the situation is reversed (79.34% Vienna vs. 77.30% CAIDA). In either case, the difference is around 2 percentage points only.

For AND (second row), we observe a larger difference. For in-AND, we see that the Vienna line starts later than the CAIDA line. Analysing the data, we see that 1.48% of CAIDA nodes have an in-AND strictly lower than 1, whereas only 0.04% of Vienna nodes have such a value. In absolute values, this is 153 (CAIDA) and 3 (Vienna). This is explained by the fact that the CAIDA measurement has 136 root points (in-degree 0), whereas the Vienna dataset only has a single one. Starting with the in-AND value of 1, we see that the plots are on the same level and grow at a similar rate.

For out-AND, we see that there are more nodes with low values for CAIDA. The CAIDA dataset has 75.50% of nodes with an out-AND of strictly lower than 1, while the Vienna dataset only has 61.80%. However, if we also include the nodes with a value of 1, we obtain 83.74% and 83.62% respectively, which is virtually the same. From that point onwards, the plots are similar.

For IAND, the same holds true, except that the in-IAND plots are similar even in the beginning. The out-IAND data is similar to the out-AND data. This means that the nodes are similarly well connected, both when considering just one hop or two hops. One reason for this could be the low maximum values (see Table 4.10), as well as these nodes being just one of many in the neighborhood of a node. This causes the average to not be affected as much by considering more nodes.

Table 4.10 furthermore shows the average and maximum values for the degree metrics for each protocol. For the averages, the datasets do not show a large difference, the values are close to each other. One notable difference is the fact that for the Vienna dataset, the average out-IAND is below 1, while for the CAIDA dataset, it is above 1. On average, considering the two hop neighborhood, a node in the Vienna dataset reaches less than

	Vienna AVG	CAIDA AVG	Vienna MAX	CAIDA MAX
Degree IN	2.0352	2.5718	164	293
Degree OUT	2.0352	2.5718	8,208	3,532
AND IN	5.0396	7.7803	164	293
AND OUT	1.2584	1.8935	772.3333	1,501
IAND IN	6.9272	10.2280	82.5	166.1364
IAND OUT	0.8170	1.1048	376.5	727.5

Table 4.10: Average and max values per dataset (IPv4 scans 2021-09)

one node from each node in this neighborhood, indicating that these neighborhoods often consist of nodes with an out-degree of 0. For in-IAND, the values are larger, with CAIDA again having a higher average (10.2280 CAIDA, 6.9272 Vienna).

Evaluating the maximum values, we see that CAIDA has higher values than Vienna (twice as much as the Vienna values), except for the out-degree. Here, Vienna has a value 2.3 times as high as the CAIDA value. The CAIDA dataset is collected from multiple starting points all around the world, and this seems to result in a network with nodes of higher degrees (both on average and in maximum values). The fact that the Vienna dataset is collected from a single source point, on the other hand, causes routing to be more centralized towards a few single high-ranking nodes, causing the out-degree maximum to be higher.

The most common values for in-AND in the Vienna dataset are 1, 2, and 3 (in this order), making up for a total of 55.28%. All other values are 5% or lower. The same holds true for the CAIDA dataset, where these 3 values make up for 46.57% of nodes. For out-AND, the situation is the same: The most common values are 0, 1, and 2 (in this order), making up for a total of 83.51% (Vienna) and 81.13% (CAIDA), with all other values having a proportion of below 2% each. We see that the datasets only differ slightly in this regard.

In summary, we see that for IPv4, the degree data across datasets does not differ much. The maximum values (other than out-degree) are consistently higher for the CAIDA dataset. The same goes for the average, though the difference is small. We therefore conclude that for IPv4 degrees, the vantage point does not make a large difference.

4.2.2 Degrees — IPv6

We now compare the degree statistics for the IPv6 measurements of the Vienna and CAIDA datasets. The data for both datasets was collected in 2022-02.

Table 4.11 shows the node counts in the datasets. The first thing to note is that the Vienna dataset is larger than the CAIDA dataset (despite only considering the nodes with a degree above 0 for at least in or out). We see that the CAIDA dataset amounts to 0.57% of the size of the Vienna dataset. We also see that the number of unidentifiable

	Vienna	Vienna %	CAIDA	CAIDA %
Total nodes (Degree >0 in+out)	62,125,794	100%	355,399	100%
Identifiable nodes	61,697,494	99.31%	324,275	91.24%
Unidentifiable nodes	428,300	0.69%	31,123	8.76%
Leaf nodes (degree-OUT 0)	5,833,062	9.39%	154,851	43.57%

Table 4.11: Node counts and proportions (IPv6 scans 2022-02)

nodes is different: In the Vienna dataset, 0.69% of nodes were unidentifiable, whereas for the CAIDA dataset it was 8.76% (12.7 times as much), despite being mandatory according to the IPv6 specification.

For the number of leaf nodes, we again observe that the CAIDA dataset has a higher percentage. The proportion of leaf nodes is 9.39% for Vienna and 43.57% for CAIDA. For Vienna, we see that more of the nodes are involved in routing than for CAIDA.

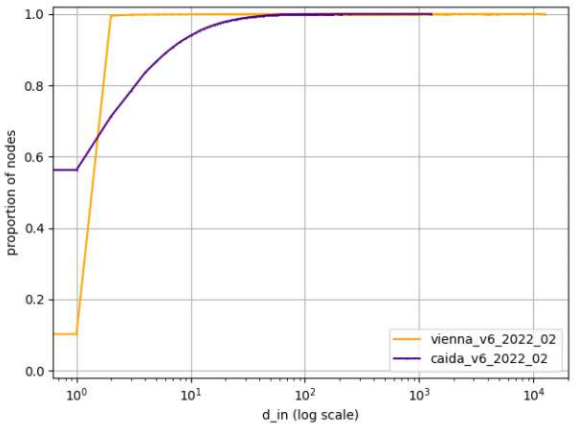
Analysing the plots in Figure 4.6, we see that the plots are similar to the comparison of IPv4 and IPv6 for the Vienna dataset (see Figure 4.1). For this reason, the analysis of these plots will be similar to the comparison above.

We start with the first row, the degree plots. For Vienna, we see that only 10.26% have an in-degree of 1. However, 89.23% of nodes have an in-degree of 2, which means that 99.49% of nodes have an in-degree of 2 or lower, the maximum being 12,590. The CAIDA curve is a bit different: We have 36 origin nodes with in-degree 0 (0.01%), and 56.26% of nodes have an in-degree of 1. While this is more than the Vienna dataset, the amount of nodes with a degree of 2 or less (71.25%) is smaller than in the Vienna dataset. From this point onwards, the plot rises smoothly close to the 1.0 mark: 99.87% of nodes have an in-degree of 70 or less, with the maximum recorded in-degree being 1,275.

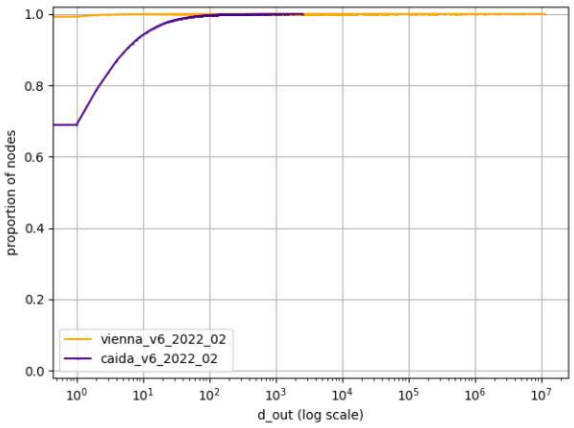
Out-degree is similar for CAIDA (albeit a bit shifted to the right) with 43.57% having an out-degree of 0 and 68.92% having a value of 1 or lower. For Vienna, we see that this time 99.26% of nodes have an out-degree of 0 or 1.

Going by degree, we see that in the Vienna dataset, the vast majority of nodes in the Vienna dataset seem to be intermediate routing nodes, whereas that number is smaller for the CAIDA dataset. It appears that the Vienna scans discover more of these intermediate nodes than the distributed CAIDA scan.

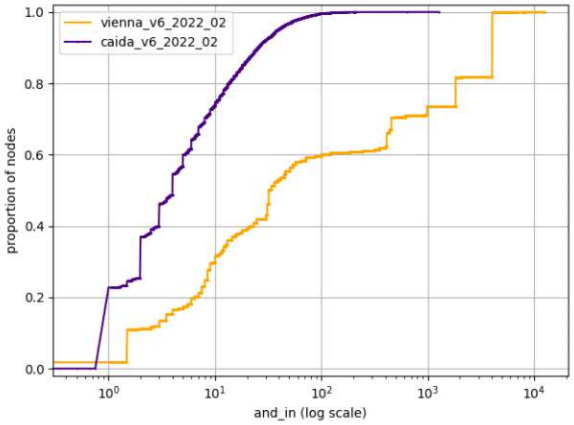
For both AND and IAND, we see that for the in-direction, the plots grow relatively smoothly, with the Vienna dataset having higher values than CAIDA (due to the larger dataset size). For in-AND, we see that while the CAIDA dataset has 22.75% of nodes with an in-AND of 1 or lower (with 22.72% being exactly on 1), only 1.78% of Vienna nodes have an in-AND of 1 or lower. On the other end of the spectrum, we see that 26.51% of Vienna nodes have an in-AND above 1,000. For CAIDA, only 0.39% of nodes have an and-IN of 100 or higher. This does make sense, as the maximum in the Vienna dataset is higher, which leads to more nodes having this value for in-AND. However,



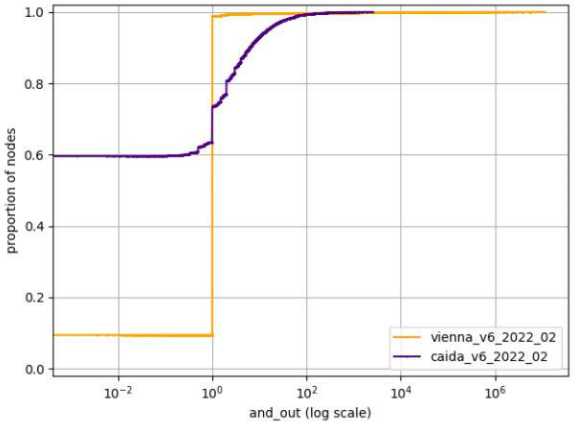
(a) Degree IN



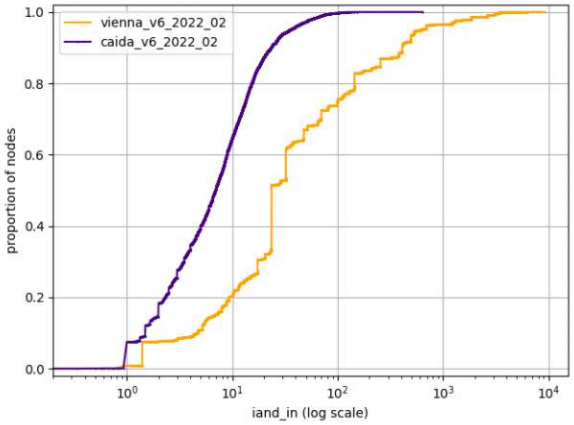
(b) Degree OUT



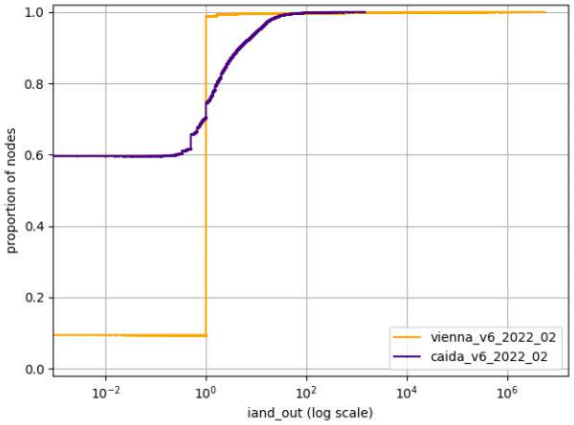
(c) Average Neighbor Degree IN



(d) Average Neighbor Degree OUT



(e) Iterated Average Neighbor Degree IN



(f) Iterated Average Neighbor Degree OUT

Figure 4.6: Comparison of Degree CDFs between datasets (IPv6 scans 2022-02)

	Vienna AVG	CAIDA AVG	Vienna MAX	CAIDA MAX
Degree IN	1.9401	3.3447	12,590	1,275
Degree OUT	1.9401	3.3447	11,311,477	2,543
AND IN	967.5179	9.6715	12,590	1,275
AND OUT	944.8912	4.3051	11,311,477	2,543
IAND IN	157.4636	10.8000	9,209.5	638
IAND OUT	22.0393	2.2213	5,655,739	1,462

Table 4.12: Average and max values per dataset (IPv6 scans 2022-02)

since more than a quarter of Vienna nodes are in this range, it also means that there are several high-degree nodes which dispatch traffic to a large amount of different successor nodes.

For the out-direction, we see that AND and IAND appear similar. In the Vienna dataset, we found 89.30% of nodes with out-IAND of 1 and 9.44% with a value of 0, which is similar to the 89.39% and 9.44% that we found for out-AND. For CAIDA, we found 73.60% of nodes with an out-AND of 1 or lower and 74.65% for out-IAND. We see that out-AND is similar to out-IAND per dataset, revealing that the connectivity in the graph does not change significantly over two hops compared to one hop.

Finally, we analyse the average and maximum values for the statistics in Table 4.12. We see that for the Vienna dataset, the degree average is 1.9401, which is close to 2. This is due to the large number of nodes with a degree of 2, which makes the average approach 2. We see that the average degree is higher in the CAIDA dataset, where it is 3.3447. This is the result of having fewer nodes with degree 0 or 1 (proportionally).

For every other average value, the Vienna dataset has higher averages. This is due to the larger scale of the graph, allowing for nodes with higher degrees that pull the average up. In fact, if we analyse the maximum values, we see that in the Vienna dataset, the maximum in-AND is 10 times as high and the out-AND is 4,448 times as high as the CAIDA maximum. This does also show in the averages, as they are 100 and 220 times as much as the CAIDA averages.

For IAND, the averages are lower for Vienna, but roughly the same for CAIDA. Evaluating the maximums, we see that this is because of Vienna dataset maximums that are lower for IAND than for AND. For CAIDA, the maximums of IAND are half as much as the ones for AND, which is not affecting the average much.

4.2.3 Betweenness Centrality — IPv4

In the following part, we examine the betweenness centrality values. Figure 4.7 compares the cumulative distribution of the betweenness centrality in the IPv4 datasets of 2021-09. Unreachable nodes of the Vienna dataset are not included in the plot.

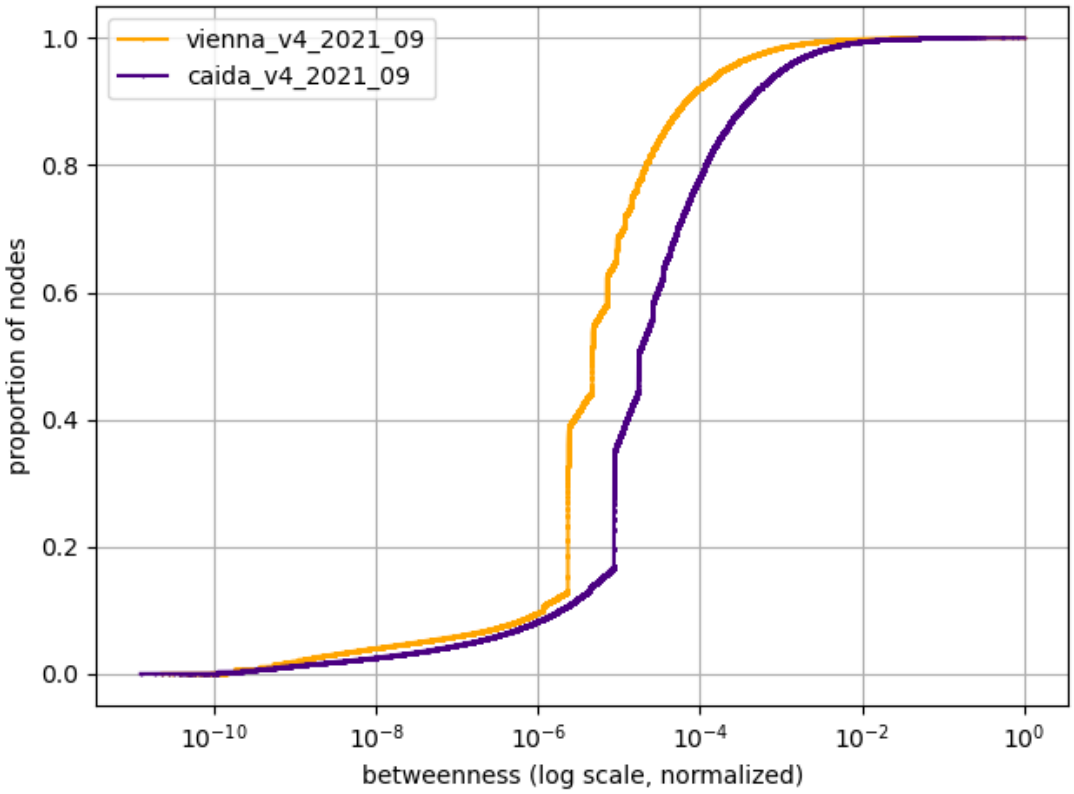


Figure 4.7: Comparison of Betweenness Centrality CDFs between datasets with normalized log y-axis (IPv4 scans 2021-09)

We observe that the two curves are almost identical on the same scales. In the middle part of the plot, between 0.1 and 0.9, we see that the Vienna values are lower than the CAIDA values by a small offset, with the curves otherwise running parallel to each other.

We see that on the edges, the curves are relatively flat, whereas in the central part, the curves are following a steep slope. 88.89% of Vienna nodes and 86.64% of CAIDA nodes have a normalized betweenness centrality between 10^{-6} and 10^{-3}. On the logarithmic scale, this is towards the middle of the observed value range. At the beginning of the central region of the plot, we see that both datasets exhibit a vertical incline. For Vienna, this amounts to 21.46% of nodes between $2.33 \cdot 10^{-6}$ and $2.34 \cdot 10^{-6}$. For the CAIDA dataset, we have 11.11% in the range between $8.643 \cdot 10^{-6}$ and $8.644 \cdot 10^{-6}$.

As for the upper end of the values, for Vienna, we see that 3.08% of nodes control 84.22% of betweenness centrality (normalized value above $10^{-3.4}$). The CAIDA dataset seems to be less centralized: 5.06% of nodes control 80.36% of betweenness centrality (normalized value above 10^{-3}). Nonetheless, both datasets show a concentration of a large portion of the shortest routes via a small amount of nodes.

As for the lower end, 9.65% of Vienna nodes and 11.87% of CAIDA nodes have a normalized betweenness centrality of below 10^{-6}. As expected, we see few nodes on the high end of the spectrum. On the other hand, we also see few nodes on the lower end of the spectrum, indicating that most nodes in the graph have a relatively high importance with few redundant paths.

Analysing the proportion of nodes with a betweenness centrality of 1 or lower, we see that 0.11% of Vienna nodes and 0.04% of CAIDA nodes have a value below 1. These are nodes that are on a shortest path, but for which there are alternative paths of the same length. As a consequence, these nodes could be removed without disturbing data flow (as long as one of the alternatives remains). We see that in the Vienna dataset, in relative terms, they are more than twice as common as in the CAIDA dataset.

For the nodes with a value of exactly 1, we see that Vienna and CAIDA have 0.57% and 0.25% respectively. For both datasets, the amount of nodes with a value of 1 is four times as much as the amount of nodes with a value below 1. Removing these nodes would either cause a disruption in the connectivity, or cause certain paths to be longer. It appears that for the "simple" routing nodes, for both datasets, more of them are essential on the path than a mere alternative.

The minimum value in the Vienna dataset is 0.125, which most likely means that this node is sharing its path with 7 other nodes (as $1/0.125 = 8$, assuming that this node is not part of multiple such paths). For CAIDA, the smallest value is 0.11111, indicating that the same phenomenon is occurring, but on a fork with 9 nodes. All in all, we see that the difference between Vienna and CAIDA in this matter is minuscule.

Due to the similar scale of the datasets, we can also compare the maxima — the Vienna maximum is 5,490,506,466.23519 whereas the CAIDA maximum is 9,036,217,166.37967 (1.65 times as much). The total node counts are 324,467 for Vienna and 464,095 for CAIDA (1.43 times as much), which demonstrates that the betweenness scaled proportionally to the dataset size for IPv4.

4.2.4 Nodes with the highest betweenness centrality — IPv4

In this section, we want to compare the nodes with the highest betweenness centrality across the datasets. We have already analysed the top 20 nodes of the CAIDA dataset of 2021-09 in Table 4.7 of Section 4.1.6. For context, we will provide a short summary of the analysis results.

From the top 20 nodes, we see that 6 nodes are from the US (with 4 being from the same AS, controlling 1.26% of betweenness centrality). There are furthermore 5 nodes belonging to one European AS, holding 1.26% of betweenness centrality. In total, 15 out of the 20 nodes belong to just 3 AS, with 11 belonging to 2 AS only. In total, the top 19 nodes (disregarding the one node with a local IP address) hold 4.42% of the betweenness centrality of the graph.

IP address	CC[1]	Prefix	AS number	NBC[2]
192.168.209.110[3]	—	192.168.0.0/16	—	1.0
80.157.203.93	DE	80.157.0.0/16	AS3320	0.5192
96.110.38.153	US	96.96.0.0/12	AS7922	0.3342
27.68.232.158	VN	27.64.0.0/12	AS7552	0.2978
205.189.32.235	CA	205.189.32.0/23	AS6509	0.2544
96.110.40.42	US	96.96.0.0/12	AS7922	0.2368
113.171.36.186	VN	113.171.32.0/19	AS7643	0.2243
27.68.244.71	VN	27.64.0.0/12	AS7552	0.2225
202.249.2.40	JP	202.249.2.0/24	AS2500	0.2053
123.255.90.195	HK	123.255.88.0/21	*unassigned*[4]	0.1967
209.18.43.63	US	209.18.32.0/20	AS7843	0.1942
188.1.144.222	DE	188.1.0.0/16	AS680	0.1911
12.123.159.233	US	12.0.0.0/8	AS7018	0.1756
210.173.145.78	JP	210.173.144.0/21	AS18126	0.1613
213.140.51.59	ES	213.140.32.0/19	AS12956	0.1569
154.54.82.246	??[5]	154.54.0.0/16	AS174	0.1445
52.93.251.81	US	52.84.0.0/14	AS16509 / AS14618	0.1436
129.250.3.57	US	129.250.0.0/16	AS2914	0.1410
154.54.41.146	??[5]	154.54.0.0/16	AS174	0.1375
180.240.190.237	SG	180.240.128.0/17	AS56308	0.1355

[1] CC: Country Code (as reported by WHOIS)
[2] NBC: Normalized Betweenness Centrality
[3] IP is part of a prefix reserved for network-internal use
[4] Prefix got unassigned in the time between the measurement and the WHOIS query. The IP is not reachable anymore at the time of writing.
[5] WHOIS not responding to this query. Most likely US (refer to Table 4.7)

Table 4.13: Top 20 AS with the highest betweenness centrality (IPv4, Vienna dataset 2021-09)

We now proceed with the analysis for the Vienna measurement of 2021-09. The values are displayed in Table 4.13. In this measurement, the top node has an IP that is reserved for network-internal use. While it could be a misconfigured node on the internet reporting a wrong IP, it is most likely an internal routing node of the outbound network.

If we disregard the top node, we see that the values are evenly distributed with no sudden cuts or jumps except for the first and second node (after the top node): The second node has a value that is 35.63% smaller than the first node.

Just like in the CAIDA dataset, we see a large range of countries for the prefixes. The most common countries are US (6 nodes) and Vietnam (3 nodes). There are 2 nodes from Germany and Japan each, as well as 2 unknown nodes. The data from measurements above suggests that AS174 is located in the US, resulting in 8 US nodes. Every other

country appears just once.

The same is true for the AS: While for the CAIDA dataset, we have seen that 3 ASes owned 15 of the nodes, in this dataset, we have a more diversified picture. Only 3 of the AS hold 2 nodes, every other AS has just one node in this listing. This could be due to the fact that if the monitors for the CAIDA dataset seem to encounter many nodes from certain ASes, this is an indicator that they are hosted with a certain provider. This explains why many of the top nodes belong to the same few ASes. For the Vienna dataset, it is always the same provider nodes that are traversed, so there are not many of them in the list (in this case, none).

Even though the top nodes are spread broadly both in terms of AS relation as well as location, they do still control the majority of the shortest paths in the graph. In total, the top 19 nodes, ignoring the internal node, control 10.54% of betweenness centrality in the graph (13.13% if we consider the internal node). This is similar to the 12.03% of the CAIDA dataset.

It appears that for betweenness centrality on IPv4, the vantage point does make a difference. The distributed measurement technique of the CAIDA dataset has resulted in a more centralized routing, as the encountered nodes belonged to fewer ASes as they did for the Vienna measurement. At the same time, we also see a broader geographical distribution of these top nodes.

4.2.5 Betweenness Centrality — IPv6

In this section, we compare the betweenness centrality between the Vienna and CAIDA datasets collected on 2022-02. Figure 4.8 compares the cumulative distribution of the betweenness centrality in the IPv6 datasets of 2022-02. Unreachable nodes of the Vienna dataset are not included in the plot.

Analysing the shapes of the plots, we see that the CAIDA curve is going more smoothly, while the Vienna curve has two sharp vertical inclines in the first half of the curve. The tendency on both curves, however, is the same: Most nodes are present in the middle value range (logarithmically) with outliers on either side.

On the upper end of the range, for the Vienna dataset, we see that 3.01% of nodes control 95.45% of betweenness centrality (normalized value above 10^{-5}). Compared to that, the CAIDA dataset is less centralized: 5.13% of nodes control 82.82% of betweenness centrality. While this is significantly less than the Vienna dataset, it is nonetheless a small minority controlling more than four fifths of the shortest path in the graph.

As we have already discussed in Section 4.1.3, the Vienna curve has 2 significant vertical inclines: One is for the betweenness value of 1 (normalized value $2.19 \cdot 10^{-14}$) with 3.98% of all nodes having that value. In the CAIDA dataset, only 0.39% of nodes have a value of 1. Regarding values below 1, we see a similar result for both datasets (0.24% Vienna, 0.19% CAIDA).

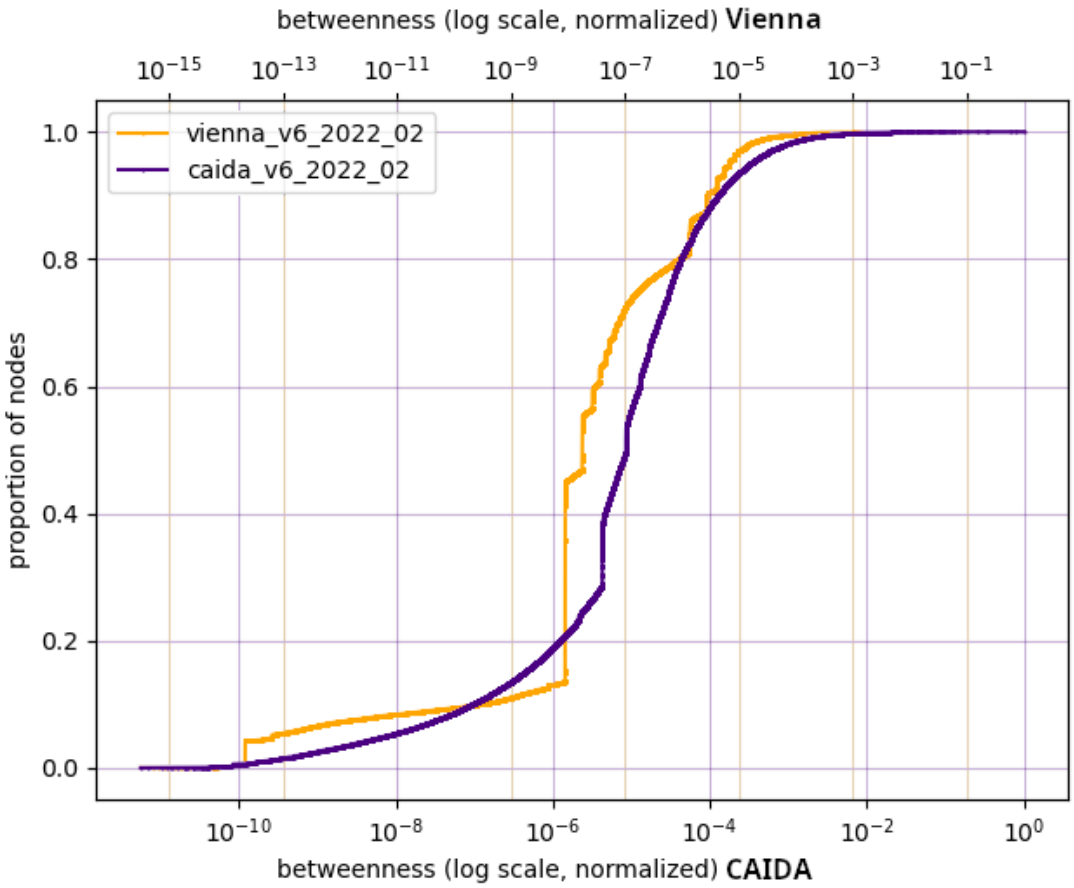


Figure 4.8: Comparison of Betweenness Centrality CDFs between datasets with normalized log y-axis (IPv6 scans 2022-02)

The larger incline is between $10^{-8.1}$ and 10^{-8}: 31.88% of nodes are within this small value range. The CAIDA dataset does not display any such inclines and progresses smoothly.

We already mentioned that the middle value range contains the most nodes. In numbers, 85.93% of Vienna nodes are within 10^{-9} and 10^{-5}. For CAIDA, we observe that 69.41% are within 10^{-6} and 10^{-4}. On the lower end of the graph, the CAIDA data shows a smoother incline, demonstrating that on the lower end, the betweenness values are more distributed.

Lastly, we analyse the smallest betweenness value per dataset. In the Vienna dataset, the smallest value is 0.0147 (more precisely, 1/68), whereas for the CAIDA dataset it is 0.067 (1/15), which is 4.53 times as much. This is most likely due to the difference in datasize. Nonetheless, this is an indicator of strong redundancy in certain areas of the graph in both datasets, more so in the Vienna dataset.

IP address	CC[1]	Prefix	AS number	NBC[2]
2600:9000:fff:ff00::300	US	2600:9000::/28	AS16509	1.0
2620:107:4000:b010::f001:6406	US	2620:107:4000::/44	AS16509	0.8055
2600:9000:fff:ff00::401	US	2600:9000::/28	AS16509	0.6761
2a01:578:0:8005::146	US	2a01:578::/29	AS16509	0.3429
2001:798:cc::1a	GB	2001:798::/40	AS20965 / AS21320	0.3375
2001:468:0:1::ef	US	2001:468::/32	AS11537	0.2630
2620:107:4000:cfff::f200:a811	US	2620:107:4000::/44	AS16509	0.1902
2409:8080:0:4:2c5:2f5:2:1	CN	2409:8000::/20	AS9808	0.1531
2001:2034:1:73::1	EU	2001:2030::/28	AS1299	0.1466
2001:2034:1:6c::1	EU	2001:2030::/28	AS1299	0.1341
2001:2034:1:b8::1	EU	2001:2030::/28	AS1299	0.1303
2409:8080:0:4:2c6:2f6:2:1	CN	2409:8000::/20	AS9808	0.1238
2001:2034:0:16f::1	EU	2001:2030::/28	AS1299	0.1215
2620:107:4000:c5e0::f3fd:c02	US	2620:107:4000::/44	AS16509	0.1176
2620:107:4000:c5e0::f3fd:c03	US	2620:107:4000::/44	AS16509	0.1135
2001:470:0:52d::2	US	2001:470::/32	AS6939	0.1074
2408:8001:3011:a00::3	CN	2408:8000::/20	AS4837	0.1029
2001:798:99:1::29	GB	2001:798::/40	AS20965 / AS21320	0.0987
2001:470:0:4b7::1	US	2001:470::/32	AS6939	0.0963
2620:107:4000:c5e0::f3fd:c00	US	2620:107:4000::/44	AS16509	0.0938

[1] CC: Country Code (as reported by WHOIS)
[2] NBC: Normalized Betweenness Centrality

Table 4.14: Top 20 AS with the highest betweenness centrality (IPv6, CAIDA dataset 2022-02)

4.2.6 Nodes with the highest betweenness centrality — IPv6

In this section, we want to compare the top 20 nodes for betweenness centrality. The Vienna IPv6 dataset of 2022-02 has already been analysed in Table 4.6 of Section 4.1.4. We therefore only provide a short summary of the results for context.

From the top 20 nodes, 11 belong to the outbound ISP. The remaining 9 nodes are distributed among 7 ASes, with one AS (AS56630) controlling 3.31% of betweenness centrality. The top 9 non-ISP nodes hold 6.27% of betweenness centrality in the graph.

We continue with the analysis of the CAIDA measurement of 2022-02. The values are displayed in Table 4.14. We see that 8 of the top 20 nodes, including the top 4, belong to the same AS (AS16509, US). According to the WHOIS data, this AS number is assigned to Amazon AWS. It is therefore highly likely that the monitors are hosted with Amazon AWS, which would make it an inevitable routing hop for many outbound paths from the different monitors. These 8 nodes alone control 12.09% of the betweenness centrality in the graph, which means that they are most likely more important than "just" for outbound routing.

The second most common AS is AS1299 (EU), appearing 4 times in total and controlling 1.93% of betweenness centrality in total. All other nodes have 2 or fewer nodes in the top 20 list.

Geographically, we see that more than half, 11 nodes, belong to a US prefix, 4 to EU, 3 to CN and 2 to GB. In the Vienna dataset, we discovered a completely different list: We found RU nodes, which are missing here. There is furthermore no AS overlap except for AS6939, which appears twice in both datasets (with the same prefix for all 4 occurrences).

Counting by AS, we see that 12 of the nodes belong to just 2 ASes (8 times AS16509 and 4 times AS1299). The remaining 8 nodes are divided among 5 ASes. We therefore see a higher concentration of traffic with fewer ASes than for the Vienna dataset, even though for the Vienna dataset, 11 nodes belong to the (single) outbound ISP. The 12 mentioned nodes control 14.02% of betweenness centrality, with the entire 20 nodes amounting for 18.67%.

We conclude that for IPv6, the vantage point makes a difference just like for IPv4. We see concentration on different geographical regions, more centralization on a smaller number of ASes, and a higher degree of control for the top nodes in the CAIDA dataset.

4.3 Comparison over time

In the previous sections, we analysed the difference across protocols and across datasets, while using measurements of the same time to ensure comparability. In this section, we compare the data over time on the same dataset. Since we have done detailed comparisons in the previous sections, we will now compare the time difference all the datasets available for a single datasource on one protocol. The relevant datasets are the Vienna IPv4 measurements of 2021-09 and 2022-02 as well as the CAIDA IPv6 measurements of 2021-09, 2022-02, and 2022-09.

4.3.1 Degrees — IPv4 over time

We now analyse the change in IPv4 routing by evaluating the Vienna measurements of 2021-09 and 2022-02.

Table 4.15 shows the amount of recorded nodes per measurement. The total amount of nodes is similar: In both measurements, more than 12 million nodes were recorded. The 2022-02 measurement has 4.02% fewer nodes than the 2021-09 measurement. As for the proportion of unidentifiable nodes, in both measurements we find that 0.3% of nodes did not report their IP. From the total nodes, we furthermore see that while 93.48% of nodes from the earlier measurement are entirely isolated from the graph (no edges in or out), for the later measurement, the proportion is 99.34%. However, analysing the number of nodes that are connected to the graph, we see that the numbers are close. The later measurement has 1.93% fewer connected nodes than the earlier measurement.

	2021-09	2021-09 %	2022-02	2022-02 %
Total nodes	12,671,145	100%	12,161,720	100%
Identifiable nodes	12,633,673	99.7%	12,125,128	99.7%
Unidentifiable nodes	37,472	0.3%	36,592	0.3%
Degree 0 in+out	11,844,632	93.48%	11,351,154	99.34%

	2021-09	2021-09 %	2022-02	2022-02 %
Degree >0 in+out	826,513	100%	810,566	100%
Identifiable nodes	789,041	95.47%	773,974	95.49%
Unidentifiable nodes	37,472	4.53%	36,592	4.51%
Leaf nodes (degree-OUT 0)	398,268	47.46%	390,720	48.2%

Table 4.15: Node counts and proportions (IPv4 Vienna dataset)

	2021-09 AVG	2022-02 AVG	2021-09 MAX	2022-02 MAX
Degree IN	2.0352	2.0288	164	164
Degree OUT	2.0352	2.0288	8,208	8,180
AND IN	5.0396	5.0202	164	164
AND OUT	1.2584	1.2534	772.3333	763.6667
IAND IN	6.9272	6.8827	82.5	82.5
IAND OUT	0.8170	0.8126	376.5	353.7538

Table 4.16: Average and max values per dataset (IPv4 Vienna dataset)

For the connected nodes, we see that for both measurements, the proportion of unidentifiable nodes amounts to about 4.5%. About half of the connected nodes are leaf nodes, with the later measurement again having 1.9% fewer such nodes than the earlier measurement, however, for the later measurement, the proportion relative to the connected nodes is higher.

In any case, all observed changes are minor and can thus be explained by usual infrastructure changes, changing networks, and maintenance work.

Figure 4.9 shows the CDFs for the datasets for the collected degree statistics. There is no discernible difference between the two plots, one of them covers the other. If we analyse the average and maximum values in Table 4.16, we see that the average and maximum values are almost the same. A tendency for lower values on the later measurement can be seen, both in average and maximum, for every metric (the total number of nodes is slightly lower as well, as seen in Table 4.15). The differences, however, are too insignificant to be linked to any impactful event or cause.

A detailed analysis of the 2021-09 dataset, along with a comparison with the CAIDA dataset of the same time period, has been done in Section 4.2.1. A detailed comparison between the two datasets here would be pointless, as the statistics are essentially the

	2021-09		2022-02		2022-09	
Total nodes	356,146	100%	355,399	100%	386,743	100%
Identifiable nodes	312,659	87.79%	324,276	91.24%	354,225	91.59%
Unidentifiable nodes	43,487	12.21%	31,123	8.76%	32,518	8.41%
Leaf nodes	151,050	42.41%	154,821	43.56%	167,640	43.35%

Table 4.17: Node counts and proportions (CAIDA IPv6 dataset)

	AVG			MAX		
	2021-09	2022-02	2022-09	2021-09	2022-02	2022-09
Degree IN	3.2790	3.3447	3.4355	1,338	1,275	1,422
Degree OUT	3.2790	3.3447	3.4355	2,431	2,543	2,060
AND IN	8.9673	9.6715	9.4585	1,338	1,275	1,422
AND OUT	3.6763	4.3051	3.9543	2,431	2,543	2,060
IAND IN	9.5980	10.8000	10.4549	669.5	638	711.5
IAND OUT	1.9833	2.2213	2.0131	1,216	1,462	906.8

Table 4.18: Average and max values per dataset (CAIDA IPv6 dataset)

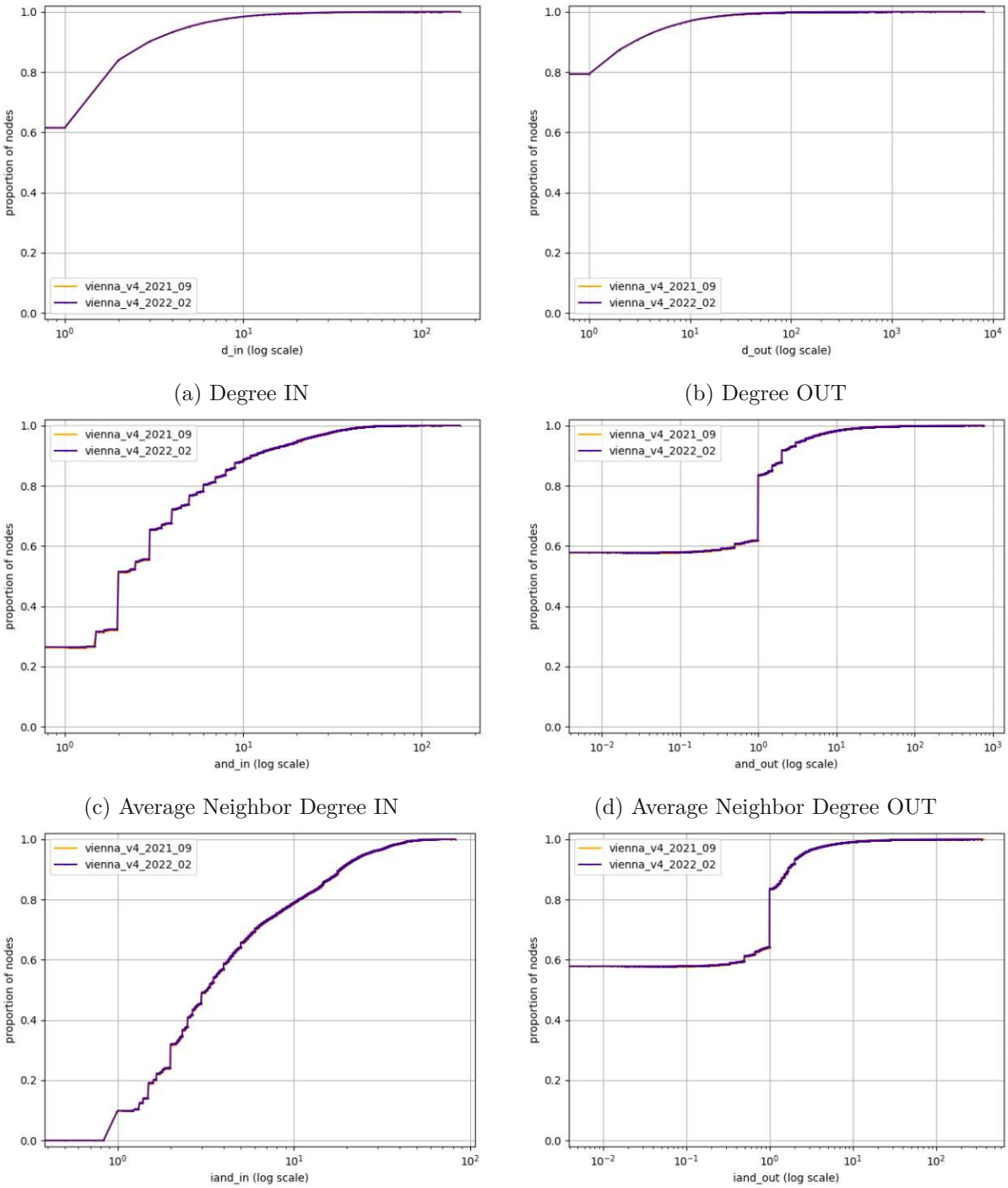
same. Nonetheless, the conclusion we draw here is that for IPv4, we see that the node counts and degree statistics have not changed significantly.

4.3.2 Degrees — IPv6 over time

In this section, we analyse the change in IPv6 routing by comparing the CAIDA measurements of 2021-09, 2022-02, and 2022-09.

Table 4.17 shows the node count for each measurement. The first and second measurement show little difference; the third measurement has 8.6% more nodes than the first one. The proportion of unidentifiable nodes decreases over time, starting with 12.21% and ending with 8.41%. In absolute numbers, the difference between the first and last measurement is 10,969, which amounts to 25.22% of the value of first measurement. In absolute numbers, the number of leaf nodes increased, though proportionally, it only grew a little: The first measurement has 42.41% leaf nodes, and the latest has 43.35%. The middle one has 43.56%, which is the highest value of the 3, however, the absolute number is in the middle. All in all, the only significant change is the relative and absolute drop in unidentifiable nodes.

Figure 4.10 shows the CDFs for the degree statistics for all 3 CAIDA IPv6 datasets. The lines in the degree plots are aligned and virtually the same. For the other stats, the different lines are not exactly aligned, though they progress in a similar pattern with just small offsets. These offsets occur at the beginning, in the low values range. The largest observable difference is for out-AND (and out-IAND). The first measurement has 61.79% of nodes with an out-AND of below 1, whereas the last measurement has 64.21% such nodes. In general, we see a tendency for the CDF to shift right for later measurements,



(a) Degree IN

(b) Degree OUT

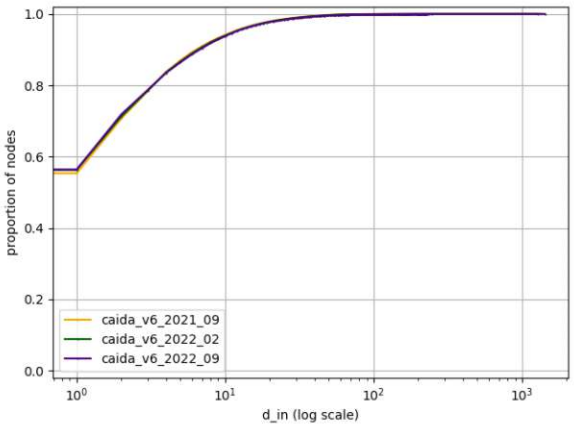
(c) Average Neighbor Degree IN

(d) Average Neighbor Degree OUT

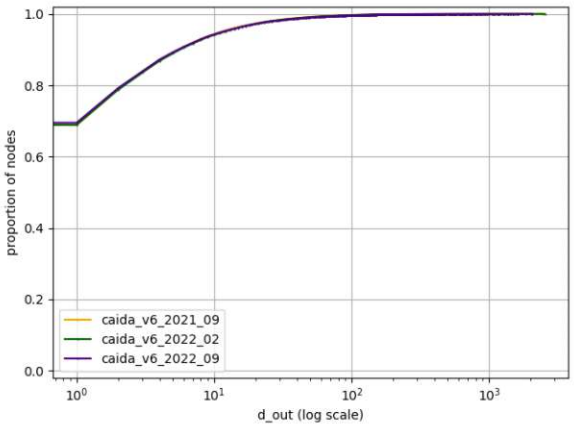
(e) Iterated Average Neighbor Degree IN

(f) Iterated Average Neighbor Degree OUT

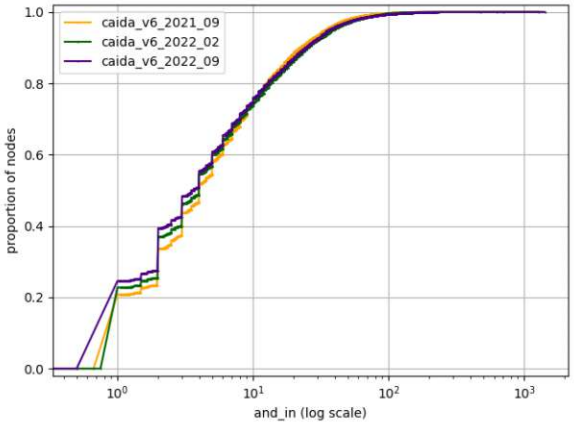
Figure 4.9: Comparison of Degree CDFs by time (Vienna IPv4 scans)



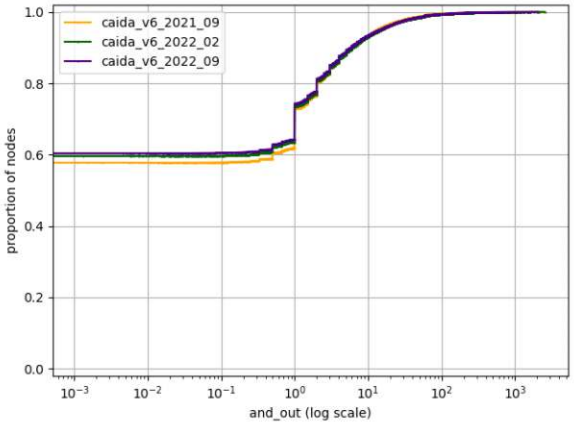
(a) Degree IN



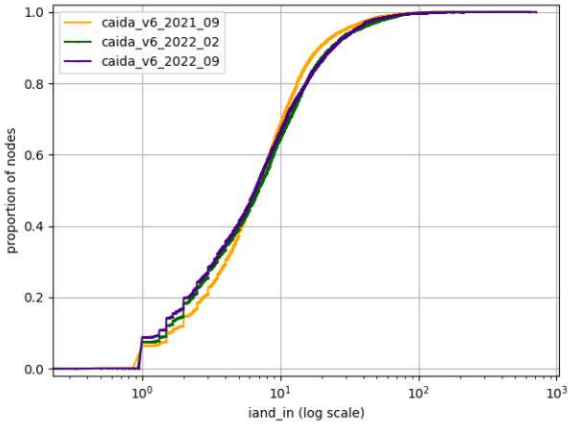
(b) Degree OUT



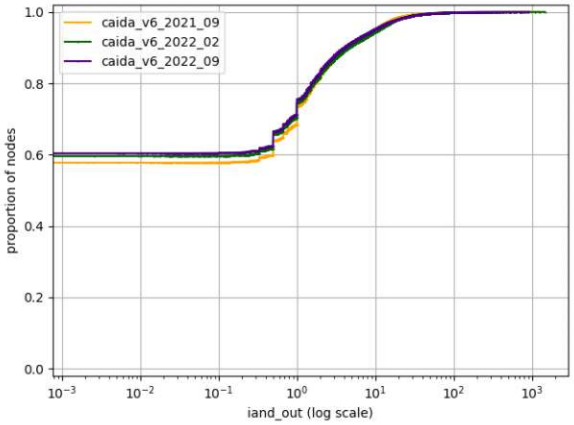
(c) Average Neighbor Degree IN



(d) Average Neighbor Degree OUT



(e) Iterated Average Neighbor Degree IN



(f) Iterated Average Neighbor Degree OUT

Figure 4.10: Comparison of Degree CDFs by time (CAIDA IPv6 scans)

indicating an increase in simple routing nodes (i.e., nodes with little degree). However, the shift is small, so we would need more data to confirm this.

Evaluating the average and maximum values in Table 4.18, we observe the same development as for the table and graph. The values do not change significantly over time. For the degree averages, we see a slight tendency towards higher values with later measurements, though the difference remains minuscule. For AND and IAND, we see that the second measurement has higher values than the first and third. This indicates that nodes in the middle measurement have a larger reach on average. The reach decreases in the third measurement, though it is still closer to the second measurement than to the first.

For the maximum values, the second measurement has the highest values of all 3 for all out-statistics and the lowest values of all 3 for all in-statistics. The third measurement has the highest values of all 3 for all in-statistics and the lowest values of all 3 for all out-statistics. The first measurement has maximum values which are always situated in the middle between the values from the other 2 measurements. Between the first and the second measurement, the values for the in-statistics are going down and for out-statistics they are going up. Comparing the first and the third measurement, the exact opposite is the case. We can therefore not make out a clear trend here.

In conclusion, we see a slight trend towards more centralization with higher-degree nodes and more nodes with low degrees, evidenced by the increasing averages between 2021-09 and 2022-09, the CDFs showing more low-degree nodes, and the proportion of leaf nodes being higher. Nonetheless, we see a decrease in the out-statistics maximum. The 2022-02 measurement, which lies in between these 2 measurements, exhibits higher average for AND and IAND than 2022-09. We cannot confidently draw a conclusion here.

4.3.3 Betweenness centrality — IPv4 over time

We now compare the betweenness centrality between the 2021-09 and the 2022-02 Vienna measurements of the IPv4 range. Unreachable nodes are not included in the plot. Inspecting Figure 4.11, we notice that both lines are identical. There is not a single spot where one line deviates from the other. Calculating the sum of all betweenness centrality values over all nodes, the later measurement has a sum that is lower by 4.03%. At the same time, the measurement has 4.02% fewer nodes than the earlier measurement, which lines up.

We have already analysed this dataset in detail in Section 4.1.3. For reference, we give a quick summary of the results here. We found that 3.1% of nodes have a normalized betweenness higher than $10^{-3.4}$, though 84.33% of betweenness centrality in the graph is held by these nodes. Most node values are situated in the middle range (logarithmically) of the plot: We found that 82.64% of nodes have a normalized betweenness value between 10^{-6} and 10^{-4}.

As the plots are exactly alike, a detailed comparison makes no sense here, as this has been done in other sections already. In conclusion, we see that the betweenness centrality only differs marginally from one measurement to another.

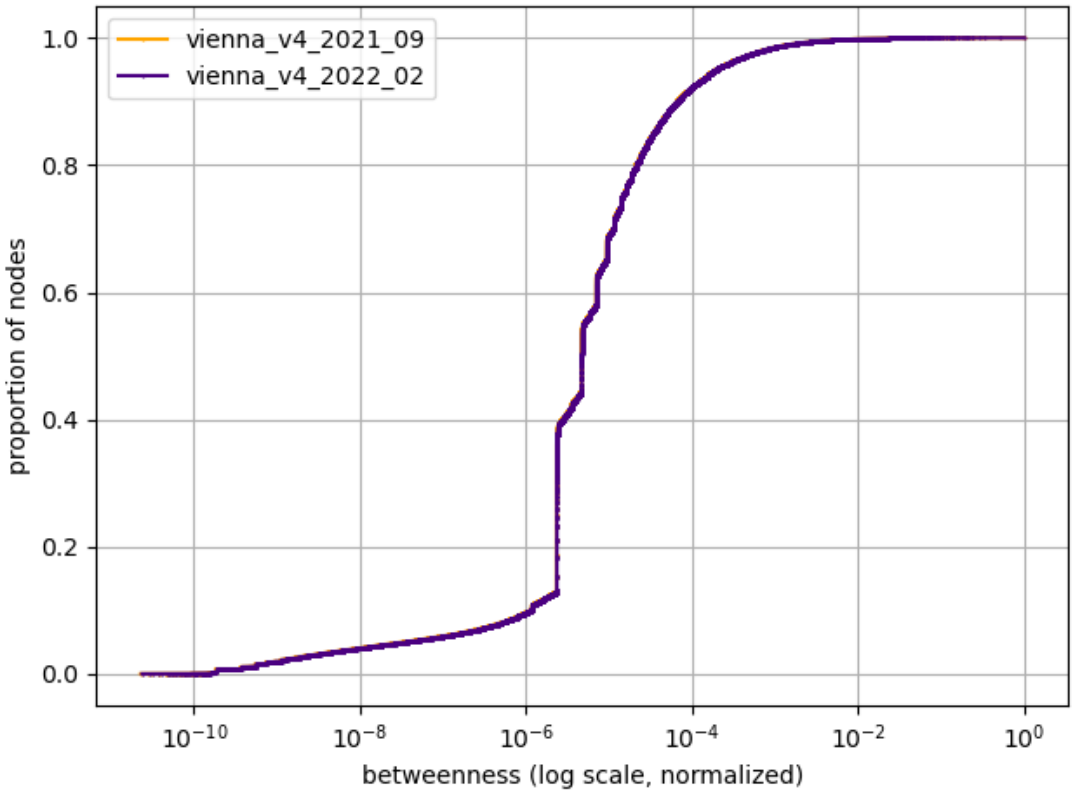


Figure 4.11: Comparison of Betweenness Centrality CDFs over time with normalized log y-axis (Vienna IPv4 scans)

4.3.4 Nodes with the highest betweenness centrality — IPv4 over time

In this section, we compare the top 20 nodes with the highest betweenness centrality for the 2021-09 and the 2022-02 Vienna IPv4 measurements. The 2021-09 data has already been displayed in Table 4.13 of Section 4.2.4, and the 2022-02 data in Table 4.5 of Chapter 4.1.4.

In the 2021-09 dataset, we have found that the topmost node was an internal network node. The third node has a value (0.3342) that is 35.63% lower than the value of the second node (0.5192), the rest of the values are distributed evenly, with the node at position 20 having a relative value of 0.1355. Geographically, the nodes are distributed widely: We have 8 US nodes, however, the other 11 nodes are situated in 7 different regions. In total, only 3 ASes hold 2 nodes, the rest of the discovered ASes only hold 1 node each. We found that the top 19 nodes (disregarding the interal node) control 10.54% of betweenness centrality.

In the 2022-09 dataset, we find no network-internal nodes, though we found two nodes belonging to the ISP where the monitor is located. The topmost node is located in HK

and has a value of almost twice as much as the second one (Normalized values: 1.0 vs. 0.5267). After the first three nodes, the values are distributed evenly down to the node on position 20, which has a normalized value of 0.1343. Unlike the previous dataset, we find that this one is more centralized: The top 18 nodes (disregarding the ISP nodes) hold 12.03% of betweenness centrality, yet they belong to only 7 different ASes. Furthermore, 12 of these 18 nodes belong to just 3 ASes. These 12 nodes, together with the top node, hold 7.05% of the betweenness centrality.

Despite both measurement having been undertaken from the same location, we see that while the 2021-09 measurement was more spread out in terms of ASes and geography, the 2022-02 measurement shows a more centralized picture. We have fewer ASes from fewer locations, which are part of a large amount of the shortest paths in the graph.

4.3.5 Betweenness centrality — IPv6 over time

We now compare the betweenness centrality between the 2021-09, the 2022-02, and the 2022-09 CAIDA measurements of the IPv6 range. Figure 4.12 shows the CDFs of all three datasets on the same scale. The lines are all showing the same shape. There is a smooth incline until 0.3, where the incline speed decreases for all three lines. At about 0.9, the incline speed increases again until the lines all meet again at 1.0.

From the graph, it is evident that the largest difference can be seen in the incline until 0.3. From there onwards, the lines run in parallel until they meet again at the end. Until 0.3, however, the lines grow at different speeds. At the 0.3 mark, the earliest measurement has the highest normalized value ($6.26 \cdot 10^{-6}$), followed by the second measurement ($4.27 \cdot 10^{-6}$) The last measurement has the lowest value ($1.02 \cdot 10^{-6}$).

We have gathered the sum of the betweenness centrality values, as well as the average and maximum, into Table 4.19. Here we see that the absolute values *increase* over time, yet the normalized values *decrease*. The 2022-09 measurement has a sum which is 1.78 times as much, an average of 1.66 times as much and a maximum of 6.3 times as much as the corresponding 2021-09 value. With the normalized sum and average, the opposite is happening: The 2022-09 measurement has a normalized sum of 0.28 times as much and a normalized average of 0.26 times as much as the corresponding 2021-09 value. In the previous section, we found that the 2022-09 measurement only has 1.09 times as many nodes as the 2021-09 measurement. We therefore observe two things: For one, the sum of centrality has increased largely, causing the average to also be higher. At the same time, we see a decrease in normalized values, indicating that the "additional" centrality was taken up by a few top nodes.

This is also visible in the maximum, which is 6.3 times higher than in the initial measurement. This tendency is only slightly visible in the middle measurement, despite the three measurements being 5 and 7 months apart. However, the second measurement has 0.21% fewer nodes than the first, which is a strong indicator for this centralization tendency already being present in the middle measurement.

	2021-09	2022-02	2022-09
Sum	289,461,095,318.4920	328,355,052,616.7510	516,127,430,459.9280
Sum (norm.)	38.2313	27.6160	10.8121
Avg	1,459,205.2958	1,680,812.1247	2,420,497.0664
Avg (norm.)	0.00019	0.00014	0.00005
Max	7,571,303,950.3481	11,890,021,027.1549	47,736,243,830.7072

Table 4.19: Sum, average and max betweenness centrality values per dataset (CAIDA IPv6 scans)

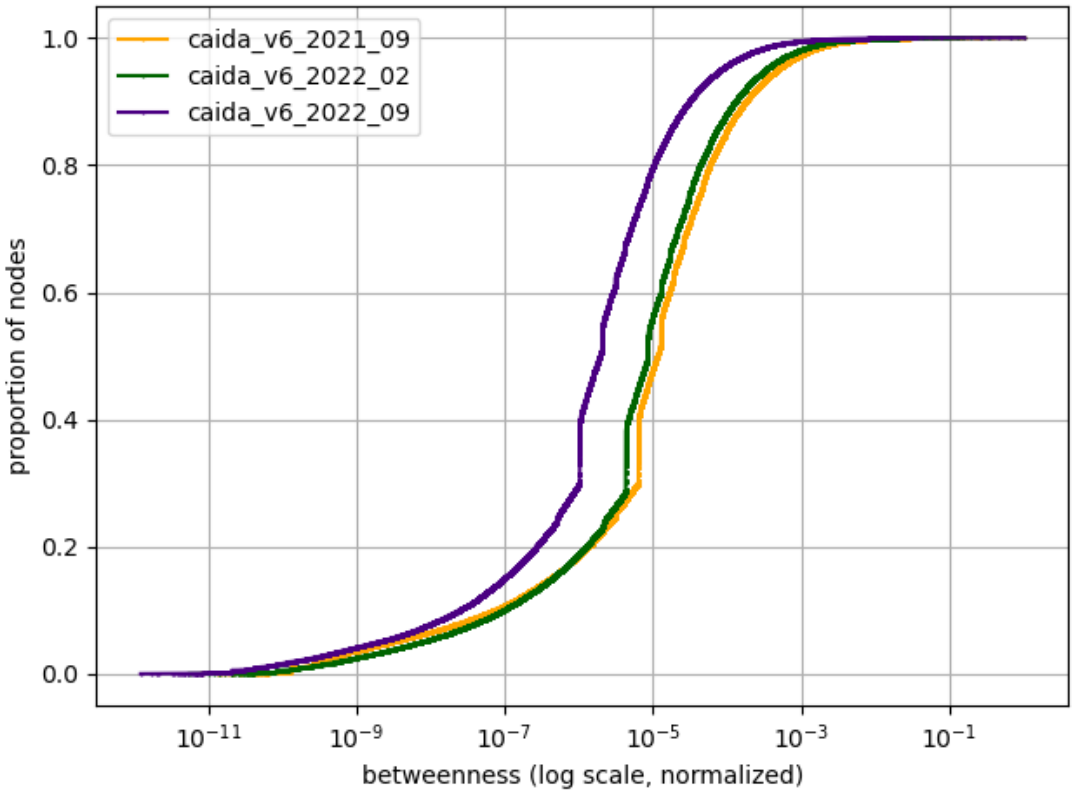


Figure 4.12: Comparison of Betweenness Centrality CDFs over time with normalized log y-axis (CAIDA IPv6 scans)

We conclude from this that as more paths are added to the graph (either through new targets or increased redundancies), they have to pass the same central routing nodes, making them even more central than they already were. While we have not been able to discover this for the IPv4 Vienna measurements, we were able to observe this for the CAIDA datasets.

4.3.6 Nodes with the highest betweenness centrality — IPv6 over time

We now want to analyse the development of the top 20 nodes with the highest betweenness centrality over time for IPv6 by comparing the results of the 2021-09, 2022-02, and 2022-09 datasets. The 2021-09 data has already been displayed in Table 4.8 of Section 4.1.6, and the 2022-02 data in Table 4.14 of Chapter 4.2.6. We will first provide a short summary of these results, after which we will analyse the results of 2022-09 in detail (Table 4.20).

In the 2021-09 measurement, the top 20 nodes hold 14.5% of the betweenness centrality. The first two nodes have a normalized value of 1.0 and 0.8486, the third one has a lower value, 0.3781. The rest is evenly distributed down to the node on position 20 with a value of 0.1252. Of these 20 nodes, 14 belong to just 3 ASes (with 2 of these nodes being assigned to 2 ASes at once).

In the 2022-02 measurement, we see a slightly more centralized distribution. The top 20 nodes hold 18.67% of betweenness centrality, with 12 of the nodes belonging to just 2 ASes, controlling 14.02% of it. The remaining 8 nodes are split among 5 other ASes. The first 3 nodes are at 1.0, 0.8055 and 0.6761, the third node has a lower value, 0.3429. From there onwards, the values are evenly distributed down to the node at position 20 with a value of 0.0938.

We now analyse the values of the 2022-09 measurement in Table 4.20. We see that 12 out of the 20 nodes belong to AS16509 (Amazon). AS6939 and AS6453 have 2 nodes each, three more ASes have 1 node each. Out of these 12 nodes, 11 belong to the prefix. Given that per prefix only two IP addresses are scanned, this is either routing in the outbound networks, or routing to a large array of hosts behind the network of this AS.

In this measurement, we have an unknown node in the top 20 (Node #7), which has not occurred in any of the other datasets. While this node has, by definition, just one predecessor, we found that it does happen to also only have one successor. This unknown node is situated between Node #1 and Node #6. Node #1 belongs to AS16509 and Node #6 belongs to AS6453, which makes an assignment to either of them difficult. We consider it more likely for this node to be an exit node of AS16509 rather than an entry node of AS6453, because entry nodes need to be “more public” in order to be found and used correctly. It could also be an entirely different AS, though again, this seems unlikely for the same reason. What is certain, however, is that this is a US node. This is because both the predecessor and successor nodes are US nodes, and a deviation via a different country would make no sense.

In this measurement, we see a high number of US nodes. Out of the top 20 nodes, 17 are US nodes. There are furthermore 2 CN nodes and 1 EU node. The top 20 nodes control 30.33% of betweenness centrality in total. The US nodes alone control 29.19%, whereas AS16509 (including the unknown Node #7) holds 27.55%.

Considering these numbers, we see that between the first two measurements, the graph slightly shifted to be more central. Between the second and third measurement, however, we see a significant shift towards a more centralized graph. Going by the relative control

#	IP address	CC[1]	Prefix	AS number	NBC[2]
1	2620:107:4000:cfff::f201:468d	US	2620:107:4000::/44	AS16509	1.0
2	2620:107:4000:c5e0::f3fd:c04	US	2620:107:4000::/44	AS16509	0.6749
3	2620:107:4000:c5e0::f3fd:c06	US	2620:107:4000::/44	AS16509	0.4619
4	2620:107:4000:c5e0::f3fd:c07	US	2620:107:4000::/44	AS16509	0.2605
5	2620:107:4000:c5e0::f3fd:c05	US	2620:107:4000::/44	AS16509	0.2585
6	2001:5a0:fff0::1	US	2001:5a0::/32	AS6453	0.0778
7	*unknown*	US[3]	*unknown*	AS16509[3]	0.0770
8	2408:8001:3011:5b::1	CN	2408:8000::/20	AS4837	0.0559
9	2600:9000:fff:ff00::300	US	2600:9000::/28	AS16509	0.0422
10	2620:107:4000:cfff::f203:56fd	US	2620:107:4000::/44	AS16509	0.0388
11	2620:107:4000:9004::58	US	2620:107:4000::/44	AS16509	0.0350
12	2620:107:4000:c5e0::f3fd:c01	US	2620:107:4000::/44	AS16509	0.0346
13	2001:2034:1:b8::1	EU	2001:2030::/28	AS1299	0.0344
14	2001:470:0:72::2	US	2001:470::/32	AS6939	0.0343
15	2001:470:0:6f0::1	US	2001:470::/32	AS6939	0.0334
16	2409:8080:0:4:2c5:2f5:2:1	CN	2409:8000::/20	AS9808	0.0333
17	2620:107:4000:cfff::f203:56a1	US	2620:107:4000::/44	AS16509	0.0322
18	2620:107:4000:9004::59	US	2620:107:4000::/44	AS16509	0.0322
19	2620:107:4000:cfff::f203:56ad	US	2620:107:4000::/44	AS16509	0.0314
20	2001:5a0:300:500::2e	US	2001:5a0::/32	AS6453	0.0311

[1] CC: Country Code (as reported by WHOIS)
[2] NBC: Normalized Betweenness Centrality
[3] Extrapolated from path information

Table 4.20: Top 20 AS with the highest betweenness centrality (IPv6, CAIDA dataset 2022-09)

of betweenness centrality in the graph, we see that it more than doubles compared to the first measurement (2021-09: 14.02%, 2022-09: 30.33%). With each measurement, we also see fewer ASes involved and more concentration in certain regions. This matches with the observations from Section 4.3.5, where we detected a small tendency in the second measurement and a significant development in the third measurement.

4.4 Measuring Decentralization

In this final evaluation section, we want to quantify the degree of decentralization (or centralization) in the graph. For this purpose, we will make use of the Gini coefficient.

Mainly used in the domain of economy, the Gini coefficient or Gini index [BL06] is a measure that compares the distribution of values to the perfect distribution. In our context, we want to compare the distribution of metrics to the *ideal* (i.e., completely decentral) distribution, where every node has the same degree and betweenness centrality.

	Degree IN	Degree OUT	AND IN	AND OUT
Vienna IPv4 2021-09	0.4123	0.8234	0.5762	0.8442
Vienna IPv4 2022-02	0.4111	0.8229	0.5759	0.8445
Vienna IPv6 2022-02	0.0694	0.5768	0.7339	0.9989
CAIDA IPv4 2021-09	0.5192	0.8665	0.6325	0.9247
CAIDA IPv6 2021-09	0.5659	0.8241	0.6061	0.8979
CAIDA IPv6 2022-02	0.5766	0.8294	0.6387	0.9139
CAIDA IPv6 2022-09	0.5880	0.8357	0.6488	0.9124

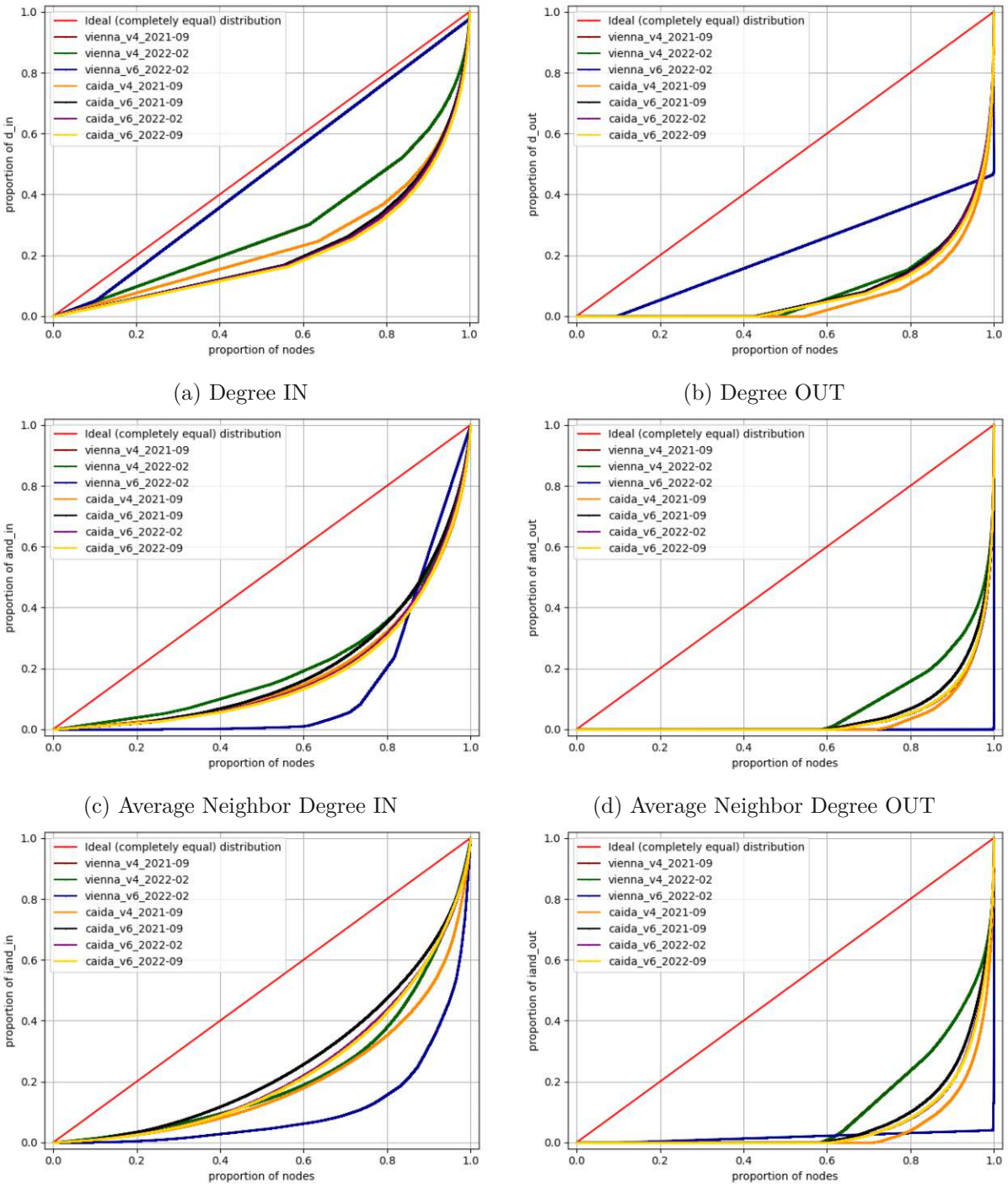
	IAND IN	IAND OUT	Betweenness
Vienna IPv4 2021-09	0.5572	0.7835	0.9454
Vienna IPv4 2022-02	0.5569	0.7839	0.9986
Vienna IPv6 2022-02	0.7880	0.9623	0.9897
CAIDA IPv4 2021-09	0.6022	0.9129	0.9162
CAIDA IPv6 2021-09	0.4843	0.8614	0.9245
CAIDA IPv6 2022-02	0.5306	0.8795	0.9256
CAIDA IPv6 2022-09	0.5367	0.8797	0.9504

Table 4.21: Gini Coefficient Values per dataset and metric

A value of 0 would mean perfect equality, whereas a value of 1 would mean perfect inequality.

If we create a plot of the cumulative sum of the values, we see what proportion of the elements (x-value) holds what proportion of the total sum of values (y-value). For example, if at $x = 0.8$ we get $y = 0.2$, we know that 80% of the elements hold 20% of the sum of all the values. A perfectly equal distribution would be a straight line from (0,0) to (1,1) and is equivalent to a Gini index of 0. The more the line deviates from the straight line, the higher the inequality, and the higher the index. Such a plot is called a Lorenz curve. The Gini index is a quantification of the deviation from perfect equality. The Lorenz curves for the degree statistics can be seen in Figure 4.13, and for betweenness centrality in Figure 4.14. All graphs contain a red line, which represents the perfect equality, for reference. The numerical Gini index for all degree statistics and the betweenness centrality can be seen in Table 4.21.

The metric with the lowest values overall is in-degree. It appears that for the Vienna datasets, the value is significantly lower than for the CAIDA datasets, meaning that in-degree is more equally distributed in the Vienna dataset. Most notably, the Vienna IPv6 measurement is displaying a value of 0.0694, which is by far the lowest Gini index of all the measured values. We see that the corresponding line in Figure 4.13a is close to the red equality line. Due to the in-degree values being confined to a smaller range (with a smaller maximum), the values inside that range are more equally distributed than other values with a larger range.



(a) Degree IN

(b) Degree OUT

(c) Average Neighbor Degree IN

(d) Average Neighbor Degree OUT

(e) Iterated Average Neighbor Degree IN

(f) Iterated Average Neighbor Degree OUT

Figure 4.13: Comparison of the Lorenz curves for each Degree statistic with the theoretical ideal distribution for all datasets

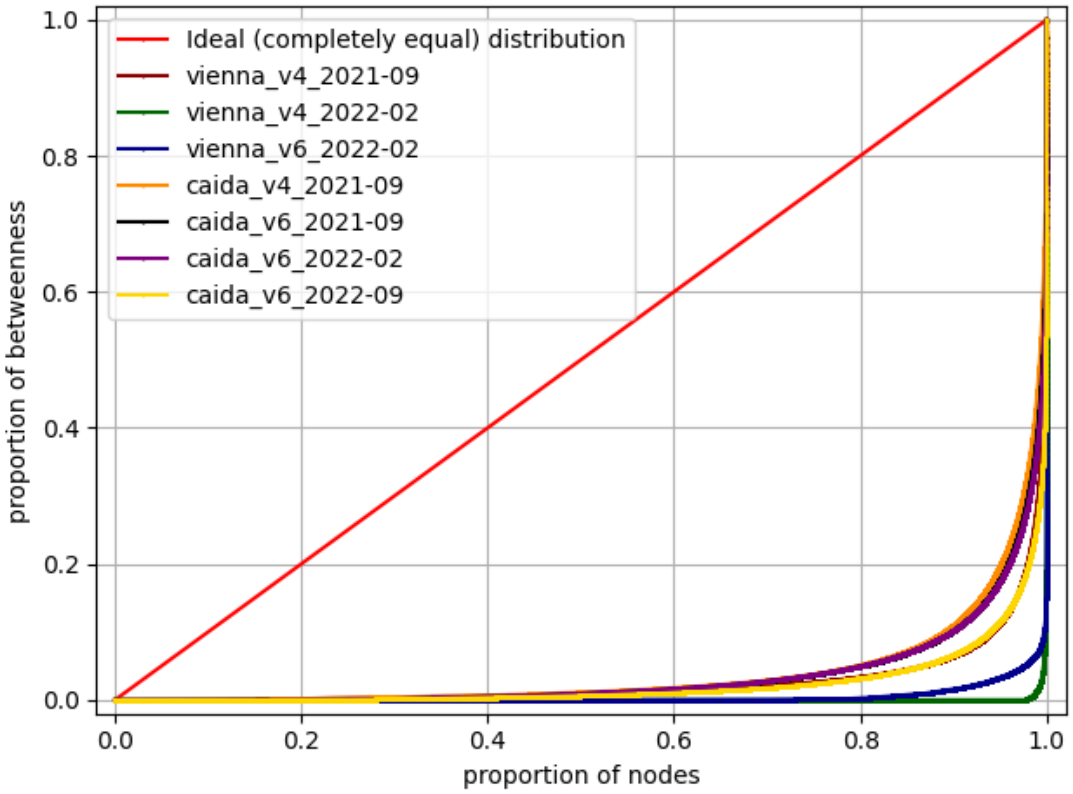


Figure 4.14: Comparison of the Lorenz curves for Betweenness Centrality with the ideal distribution for all datasets

For out-degree, we see that the values are higher and the lines in the plot further away from the equality line. This time, Vienna and CAIDA values match, except for the Vienna IPv6 dataset. It is again significantly lower than the other values, albeit higher than the in-degree value. For out-degree, there are high values held by few nodes and many nodes with 0 or 1, which contributes to the higher inequality.

We move on to in-AND and out-AND. We again see that for the out-direction, the values are more centralized than for the in-direction. For both directions, we see that the values are more centralized than the plain in and out degrees. The most striking difference, however, is the fact that for the Vienna IPv6 measurement, the values are now *higher* than the values for the other datasets. For out-AND, this value is close to 1, meaning that the out-AND values are highly concentrated among a few nodes.

With IAND, we see the same thing as for AND. The values are higher than plain degree and similar to AND. Just like for AND, the Vienna IPv6 measurement has a higher value than the other datasets. It seems that while the plain degree is more decentralized in the Vienna IPv6 dataset, the AND and IAND is more centralized compared to the other

datasets.

At the same time, for both AND and IAND, we see a tendency for the CAIDA data to have higher values (disregarding the Vienna IPv6 measurement, which is always highest). The only exception is the in-IAND CAIDA IPv6 values, which are below both the CAIDA IPv4 measurement and the Vienna measurements, even if only slightly.

Finally, we consider the distribution of betweenness centrality. In the previous sections, we have already determined that most of the betweenness centrality in the graphs is concentrated among a few nodes. The Gini coefficients confirm this discovery: For every dataset, the value is above 0.9. Moreover, we see that the Vienna datasets show higher centralization than the CAIDA datasets. While the CAIDA results are similar across protocols and time, for Vienna, we see that the 2022-02 measurement shows a high degree of centralization. Both the IPv4 and IPv6 measurement of 2022-02 are close to 1, whereas the IPv4 2021-09 measurement is only at 0.9454.

To sum up, we see that for values related to incoming edges (in-degree, in-AND, in-AND), the level of centralization is lower than for the rest of metrics. The Vienna IPv6 measurement shows a smaller degree of centralization when considering the plain degree, yet a higher degree of centralization for AND and IAND. The Gini index for betweenness centrality is high for all datasets (> 0.9), with the Vienna 2022-02 measurement showing values close to 1 for both IPv4 and IPv6.

CHAPTER 5

Conclusion

In our ambition to understand routing in today's internet better, we have collected traceroute data from two sources (Vienna and CAIDA) at different points in time for both protocols, IPv4 and IPv6. We created a framework to process the data and generate statistics for each dataset for further analysis.

For each node in every dataset, we have gathered six degree statistics (Degree, Average Neighbor Degree, Iterated Average Neighbor Degree, each for both in and out) as well as the betweenness centrality. We have then compared the outputs across the three dimensions protocol, dataset, and time.

From the data comparison, we have obtained the following conclusions:

- **Comparison between protocols**

  We have found that IPv6 appears to be more centralized than IPv4. Considering the degree statistics, we see a slight increase in centralization for the Vienna dataset, yet we see a decrease in the CAIDA dataset. For betweenness centrality, however, both datasets show an increase, as more of the shortest paths go through fewer nodes, despite the larger addressing space. For IPv6, we also detected an increase in redundant nodes (i.e., most likely load balancers), which is possible due to the larger address space. The CAIDA set furthermore revealed a large proportion of nodes not behaving according to the IPv6 specification.

- **Comparison between datasets / vantage points**

  We have found that for IPv4, the difference between the single vantage point of the Vienna measurement and the distributed measurement performed by the CAIDA monitors is small, and the collected statistics are similar across the datasets. We did find a higher proportion of the shortest paths going through the most important nodes for the CAIDA dataset, though. For IPv6, we see a difference towards

higher centralization in the Vienna dataset, yet again, the top nodes hold a higher proportion of the shortest paths in the CAIDA dataset. Notably, we see that the CAIDA dataset has a higher proportion of leaf nodes, indicating that it generally found shorter routes with fewer intermediate nodes.

- **Comparison between different points in time**

  For the IPv4 Vienna measurements, we found almost no change at all. Both the degree statistics and the betweenness centrality statistics are almost identical. For the IPv6 CAIDA measurements, which we did for 3 measurements across a time span of one year, we do see a shift towards more centralization as time passes, yet no significant changes are to be found here either.

Finally, we quantified the degree of centralization using the Gini coefficient. We found high values for outgoing statistics, yet medium to low values for incoming statistics. We see that while there is not a high concentration in incoming connections, outgoing connections are more concentrated to certain hotspots or hubs. The same is valid for betweenness centrality, where we found that all values are above 0.9, indicating that all collected measurements show a tendency for the shortest paths to go certain predetermined ways.

Summarized in one short sentence, we tend to see a higher degree of centralization for IPv6 compared to IPv4, for distributed CAIDA measurements compared to single-point Vienna measurements, later measurements compared to earlier measurements, as well as outgoing connections compared to incoming connections.

List of Figures

List of Tables

Bibliography

[ACO+06] Brice Augustin, Xavier Cuvellier, Benjamin Orgogozo, Fabien Viger, Timur Friedman, Matthieu Latapy, Clémence Magnien, and Renata Teixeira. Avoiding traceroute anomalies with Paris traceroute. https://www.researchgate.net/publication/221611998, 2006. Last accessed 2022-11-01.

[Aut21] Internet Assigned Numbers Authority. IANA IPv4 Special-Purpose Address Registry. https://www.iana.org/assignments/iana-ipv4-special-registry/iana-ipv4-special-registry.xhtml, 2021. last accessed 2022-10-29.

[Bev16] Robert Beverly. Yarrp'ing the Internet: Randomized High-Speed Active Topology Discovery. https://dl.acm.org/doi/pdf/10.1145/2987443.2987479, 2016. last accessed 2022-10-15.

[BGT04] Tian Bu, Lixin Gao, and Don Towsley. On characterizing BGP routing table growth. https://doi.org/10.1016/j.comnet.2004.02.003, 2004. last accessed 2022-10-29.

[BL06] Lorenzo Giovanni Bellù and Paolo Liberati. Inequality analysis — the gini index. https://web.archive.org/web/20210228020016/https://www.fao.org/docs/up/easypol/329/gini_index_040EN.pdf, 2006. last accessed 2022-12-29.

[Bra00] Ulrik Brandes. A Faster Algorithm for Betweenness Centrality. https://pdodds.w3.uvm.edu/research/papers/others/2001/brandes2001a.pdf, 2000. last accessed 2022-10-26.

[Cis20] Cisco Systems, Inc. Cisco Annual Internet Report (2018–2023) White Paper. https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html, 2020. Last accessed 2022-10-29.

[Cis22] Cisco Systems, Inc. BGP Best Path Selection Algorithm. https://www.cisco.com/c/en/us/support/docs/ip/

border-gateway-protocol-bgp/13753-25.html, 2022. last accessed 2022-10-28.

[DH98a] Stephen E. Deering and Robert M. Hinden. RFC 2373: IP Version 6 Addressing Architecture. https://datatracker.ietf.org/doc/html/rfc2373, 1998. last accessed 2022-10-29.

[DH98b] Stephen E. Deering and Robert M. Hinden. RFC 2460: Internet Protocol, Version 6 (IPv6) Specification. https://datatracker.ietf.org/doc/html/rfc2460, 1998. last accessed 2022-10-29.

[EC22] Tim S. Evans and Bingsheng Chen. Linking the network centrality measures closeness and degree. https://doi.org/10.1038/s42005-022-00949-5, 2022. Last accessed 2022-11-14.

[fAIDAa] Center for Applied Internet Data Analysis. Archipelago (Ark) Measurement Infrastructure. https://www.caida.org/projects/ark/. last accessed 2022-10-26.

[fAIDAb] Center for Applied Internet Data Analysis. Scamper. https://www.caida.org/catalog/software/scamper/. last accessed 2022-10-26.

[fAIDAc] Center for Applied Internet Data Analysis. The CAIDA UCSD IPv4 Routed /24 Topology Dataset - 2021-09-30. https://www.caida.org/catalog/datasets/ipv4_routed_24_topology_dataset/. last accessed 2022-10-26.

[fAIDAd] Center for Applied Internet Data Analysis. The CAIDA UCSD IPv6 Topology Dataset - 2021-09-22, 2021-09-29, 2021-09-30, 2022-02-12, 2022-02-27, 2022-02-28, 2022-09-15, 2022-09-24, 2022-09-25, 2022-09-29, 2022-09-30. https://www.caida.org/catalog/datasets/ipv6_allpref_topology_dataset/. last accessed 2022-10-26.

[fAIDA11] Center for Applied Internet Data Analysis. warts – format for scamper's warts storage. https://www.caida.org/catalog/software/scamper/man/warts.5.pdf, 2011. last accessed 2022-10-16.

[Fre77] Linton C. Freeman. A Set of Measures of Centrality Based on Betweenness. https://doi.org/10.2307/3033543, 1977. Last accessed 2022-11-12.

[Goo22] Google. Google IPv6 Statistics. https://www.google.com/intl/en/ipv6/statistics.html, 2022. Last accessed 2022-10-29.

[LDP+21] Kevin Limonier, Frédérick Douzet, Louis Pétinaud, Loqman Salamatian, and Kave Salamatian. Mapping the routes of the Internet for geopolitics: The case of Eastern Ukraine. https://doi.org/10.5210/fm.v26i5.11700, 2021. last accessed 2022-10-28.

[Mai21] Markus Maier. Investigating Router Misconfigurations on the IPv6 Internet. https://doi.org/10.34726/hss.2021.77442, 2021. last accessed 2022-10-28.

[Mil84] D. L. Mills. RFC 904: Exterior Gateway Protocol formal specification. https://datatracker.ietf.org/doc/html/rfc904, 1984. last accessed 2022-10-29.

[Pos81] Jon Postel. RFC 791: Internet Protocol. https://datatracker.ietf.org/doc/html/rfc791, 1981. last accessed 2022-10-29.

[RIP22] RIPE NCC. RIS Raw Data. https://www.ripe.net/analyse/internet-measurements/routing-information-service-ris/archive/ris-raw-data, 2022. Last accessed 2022-10-31.

[RLH06] Yakov Rekhter, Tony Li, and Susan Hares. RFC 4271: A Border Gateway Protocol 4 (BGP-4). https://datatracker.ietf.org/doc/html/rf4271, 2006. last accessed 2022-10-29.

[TC15] Ole Troan and Brian Carpenter. RFC 7526: Deprecating the Anycast Prefix for 6to4 Relay Routers. https://datatracker.ietf.org/doc/html/rfc7526, 2015. last accessed 2022-10-29.