University of Nevada, Reno

# Graph Data Mining to Construct Sampled Internet Topology Maps

A thesis submitted in partial fulfillment of the
requirements for the degree of Master of Science in
Computer Science

by

Hakan Kardes

Dr. Mehmet H. Gunes / Thesis Advisor

December, 2010

University of Nevada, Reno
Statewide • Worldwide

THE GRADUATE SCHOOL

We recommend that the thesis
prepared under our supervision by

**HAKAN KARDES**

entitled

**Graph Data Mining To Construct Sampled Internet Topology Maps**

be accepted in partial fulfillment of the
requirements for the degree of

**MASTER OF SCIENCE**

Mehmet H. Gunes, Advisor

Murat Yuksel, Committee Member

Sami Fadali, Graduate School Representative

Marsha H. Read, Ph. D., Associate Dean, Graduate School

December, 2010

# Acknowledgments

I would like to express my sincere gratitude for my adviser Dr. Mehmet H. Gunes, whose guidance, inspiration, understanding, patience, and encouragement added considerably to my graduate experience. I would like to thank Dr. Sami Fadali and Dr. Murat Yuksel for agreeing to be on my thesis committee despite their extremely busy schedule and for their invaluable comments.

I would like to thank my colleagues David Shelly and Talha Oz for their effort to put together the Cheleby project and all other CNL members for long discussions.

I would also like to thank my family for the support they provided me through my entire life. Finally, I must acknowledge Enes, one of my best friends, without whose encouragement I would not have decided an academic career.

Hakan Kardes

*University of Nevada, Reno*

*Dec 2010*

# Graph Data Mining to Construct Sampled Internet Topology Maps

Hakan Kardes

University of Nevada, Reno, 2010

Supervisor: Mehmet H. Gunes

Abstract

Understanding the topological characteristics of the Internet is important for researchers and practitioners as the Internet grows with no central authority. This understanding is a necessity to better design, implement, protect and operate the underlying network technologies, protocols, and services. The need for accurate Internet topology map has increased recently with new services such as overlay networks and IP TV. Router-level Internet topology measurement studies have three main steps: topology collection, topology construction, and topology analysis. In topology construction, there are several main challenges: unresponsive router resolution, identification of underlying subnets and detection of IP aliases. These tasks become especially challenging when large-scale topologies of millions of nodes are studied. In this thesis, we present the topology construction processes of the Cheleby system, an Internet topology mapping system that provides insight into the Internet topology by taking daily snapshots of the underlying networks. The system utilizes efficient algorithms to process large-scale datasets collected from distributed vantage points and provides accurate topology graphs at link layer. Incorporating enhanced resolution algorithms, Cheleby provides comprehensive Internet backbone maps.

# Contents

# List of Tables

# List of Figures

# Chapter 1

# Introduction

The Internet, the largest man made complex network, is a web of interconnected backbone networks over which thousands of small and medium size Autonomous Systems (ASes) interconnect individuals, businesses, universities, and agencies. The Internet is a spontaneously growing complex system whose large-scale structure is affected by many interacting units aimed at optimizing local communication efficiency without a central authority. While the building blocks of the Internet, i.e., its protocols and individual routing systems, have been subject to intensive studies, the immense global entity has not been precisely characterized.

The Internet's global properties cannot be inferred from local ones as it is composed of networks engineered with large technical diversity and range from small local campuses to large transcontinental backbone providers [24]. Additionally, the Internet evolves with the interplay between cooperation to make the network work efficiently, and competition between providers to earn money. Routers and links are added by competing entities according to local economic and technical constraints where topology information is kept confidential due to various privacy and security concerns. The combination of all of these factors results in a general lack of understanding of the topological characteristics of the Internet.

The need for Internet topology map has increased obviously over last years. It allows users to evaluate and forecast the growth trends and the performance problems. For example, understanding the topology of the Internet helps adapting applications to the underlying network in better manner. Furthermore, router-level Internet maps are useful to analyze the topological characteristics of the Internet and to design topology generators that can produce Internet-like synthetic network topologies to be used in simulation studies [39]. However, the confidentiality of network topology introduces challenges for the research community and requires them to use other means to collect this information.

The research community has been conducting numerous Internet measurement studies to answer various questions on the functional and topological characteristics of the Internet. In order to facilitate topology measurement studies, several research groups have developed mapping systems to collect the required information. These include the PlanetLab measurement infrastructure [7], the Archipelago measurement infrastructure of CAIDA [49], the iPlane infrastructure [50], the DIMES project [63], the Scriptroute project [70], and several others [54, 67, 74, 75]. In general, Internet topology measurement studies consist of three phases: (1) topology sampling, (2) topology construction, and (3) topology analysis. Inaccuracies in the first two processes may significantly affect the accuracy of the observations or results obtained in the measurement study [32, 37, 73]. However, current topology mapping systems only provide raw data without completing topology construction tasks. Most of the researchers that use provided raw data are not aware of these challenges causing inaccuracies in their studies.

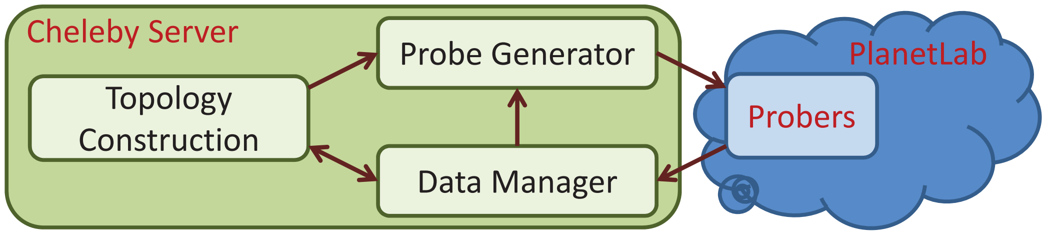


Figure 1.1: Cheleby System Overview

Many Internet studies require availability of representative topology maps. Depending on the nature of measurement study, researchers may use different types of topology maps including AS level [43, 51], Point-of-Presence (POP) level [26, 78], router level [67], or IP address level maps [52]. A POP level topology map is often the most detailed information that ASes make publicly available, if at all, about their network [24].

In this thesis, we present the topology constructor part of Cheleby, an Internet topology mapping system that provides insight into the Internet backbone topology by taking daily snapshots of the underlying networks. The system utilizes efficient algorithms to process large scale data-sets collected from distributed vantage points and provides accurate topology graphs at link layer. Incorporating enhanced resolution algorithms, Cheleby provides comprehensive topology maps.

Cheleby topology mapping system, shown in Figure 1.1, runs on a server which actively manages PlanetLab nodes as its monitors to collect topology information from geographically diverse vantage points. The server instructs monitors to collect partial path traces and perform other probing activities. Cheleby then resolves subnets, IP aliases and unresponsive routers within the collected raw data to construct the network graph corresponding to the sampled network. Unlike previous approaches, Cheleby provides both the raw and the constructed Internet topology data.

## 1.1 Preliminaries

In Internet topology measurement studies, there are generally three main steps:

(1) Topology collection,

(2) Topology construction,

(3) Topology analysis.

In general, most of the router level Internet measurement studies are active measurement based, and they utilize traceroute or several other Internet debugging tools [12, 55, 63, 71] to collect path traces from a set of vantage points to a set of destination nodes. As mentioned above, there are several systems providing the raw router-level Internet topology data, but there is no system providing the up-to-date constructed topologies. This is simply because of the fact that the literature on topology construction is fairly limited compared to the studies in topology collection and topology analysis. Moreover, topology construction is not a straightforward process requiring considerable computations. However, inaccuracies in this process may significantly affect the accuracy of the observations or results obtained in the measurement study [11, 15, 32, 36, 73].

This thesis mainly addresses the topology construction part. We first introduce the main problems that must be handled during topology construction step.

Topology construction involves several tasks including: (1) filtering erroneous traces (2) inferring IP addresses that are connected over the same subnet, (3) identifying IP addresses belonging to the same router, and (4) resolving unresponsive routers that are represented by '*'s in traceroute outputs. And, the inclusion of these

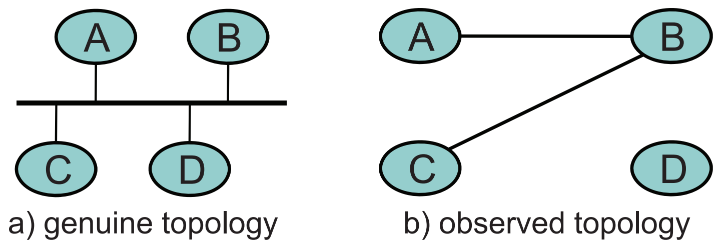
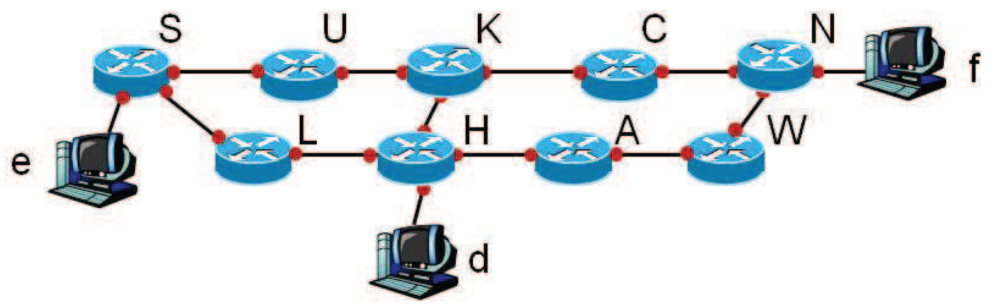


Figure 1.2: Effect of Subnet Resolution

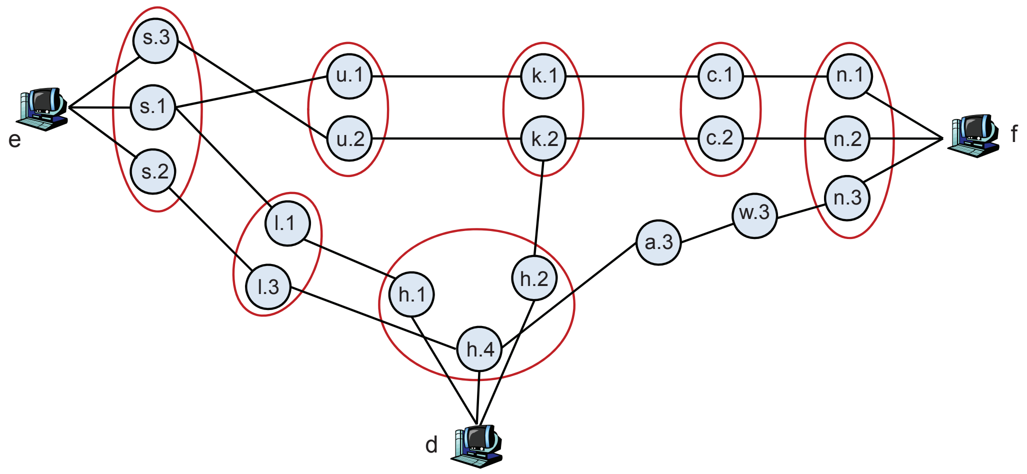
tasks will considerably improve the accuracy and the completeness of the constructed topology map.

First of all, genuine subnet resolution helps in identifying connectivity between IP addresses in the data set. The main goal in subnet detection is to identify multiple links that appear to be separate and combine them to represent their corresponding single hop (e.g., point-to-point or multi-access) connection medium. Normally, routers are connected to each other over subnetworks. Subnet inference helps to identify the underlying subnets by analyzing the relation between IP addresses in the data set. The successfully inferred subnet information helps: (1) to improve the quality of the resulting map by annotating it with additional information, (2) to increase the scope of the map by adding new links into the resulting map, and (3) to improve the IP alias resolution process [38].

In subnet resolution task, the IP addresses in a data set are analyzed to infer subnet relations among them. As an example, consider four routers A, B, C, and D, in Figure 1.2-a, that are connected to each other via a multi-access link. Assume that a collected set of path traces include the A-to-B link and the B-to-C link and no path trace at hand includes the A-to-C link or any link between D and other routers. In this case, a router level map that does not consider the subnet relation among these IP addresses will yield in a subgraph as shown in Figure 1.2-b, which is considerably



(a) Sampled Network



(b) Resulting Network

Figure 1.3: Importance of Alias Resolution

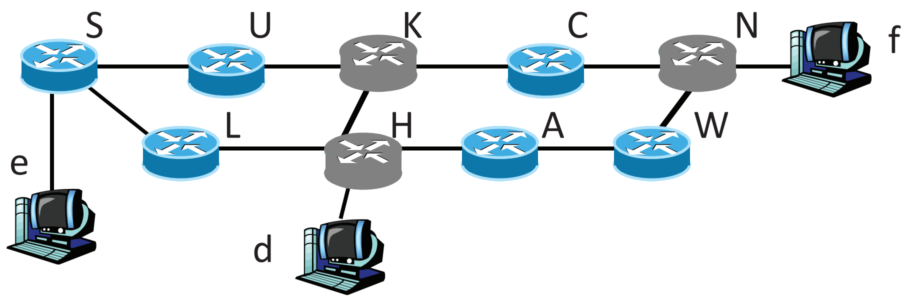
different from the underlying topology. On the other hand, a careful study of the IP addresses may detect the subnet relation between the routers and therefore improve the resulting map.

The second issue to consider during router-level topology construction is IP aliases. As routers have multiple interfaces each one has a different IP address. In a given set of path traces, a router may appear on multiple path traces with different IP addresses. Therefore, there is a need to identify and group IP addresses belonging to the same router. Without IP alias resolution the resulting topology map may be significantly different from the real topology [36, 37]. For instance, each router has multiple interfaces with unique IP addresses in Figure 1.3-a. Then, collecting traces between all pairs of **e**, **d**, and **f** end systems, we would obtain a sampled topology as in Figure 1.3-b. Hence, we need to identify IP aliases and cluster them as shown with red circles.

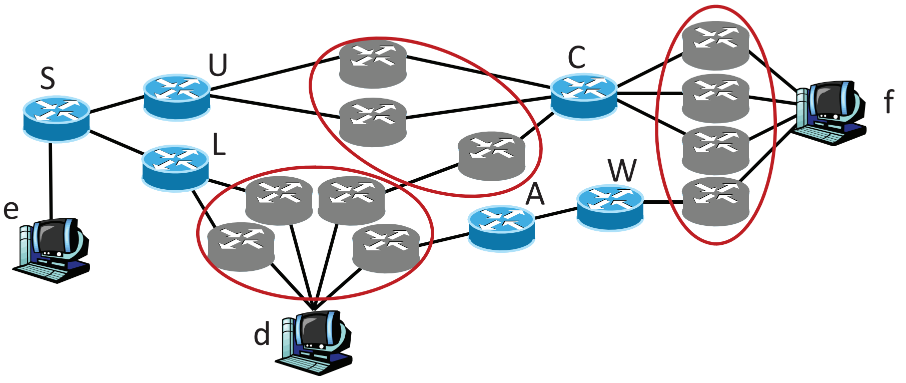
Finally, unresponsive routers are routers that are unresponsive to traceroute probes and are represented by a '*' in a traceroute output. Since a router may appear in multiple traceroute outputs, it is needed to identify '*'s (i.e., anonymous nodes) that belong to the same router. Based on the number of unresponsive routers in the topology and the way of topology collection, there would be huge number of '*' in the collected set of path traces. For example, daily Internet topologies collected by Cheleby have more than 7M unresponsive nodes along with 1.2M known interfaces. Moreover, a sample network is shown in Figure 1.4-(a). Assume that in this topology H, K, and N are configured to act as unresponsive routers. When we run traceroute queries between all vantage points, represented as e, d and f in this figure, collected path traces will be as shown below. Using these traces, the resulting topology presented in Figure 1.4-(b).

```
d - * - L - S - e
d - * - A - W - * - f
e - S - L - * - d
e - S - U - * - C - f
f - * - C - * - * - d
f - * - C - * - U - S - e
```

Therefore, it can be concluded that even small number of unresponsive routers may significantly distort the constructed network topology and prevent researchers



(a) Sampled Network



(b) Resulting Network

Figure 1.4: Importance of Unresponsive Router Resolution

from obtaining the actual Internet topology from the trace paths collected. This example also shows the significance of the unresponsive router resolution task to obtain the actual topology map.

## 1.2 Organization

In Chapter 2, we present the related work in Internet topology construction area. In Chapter 3, we present a structural graph indexing approach that can be used, not only in Internet topology construction process, but also in other complex network analysis projects. In Chapter 4, we give an overview of the Cheleby Internet Mapping system and present issues in topology construction. In Chapter 5, we present the experiments and the results. Finally, in Chapter 6, we conclude the thesis and provide a brief overview of future work to enhance the Cheleby Internet topology mapping system.

# Chapter 2

# Literature Survey

In this chapter, we review the related works in the Internet topology mapping area. Since the topology collection and analysis parts are beyond the scope of this thesis, we mainly focus on the Internet topology construction studies.

## 2.1 Subnet Resolution

In [38] Gunes et al. identified the subnet resolution task for topology construction studies and analyze its utility. The proposed subnet detection task presents similarities with that of the IP alias resolution task (the previous task of map construction). In IP alias resolution, the goal is to identify nodes that appear to be separate in the collected path traces and combine them into one single node (i.e., to detect IP addresses, in the data set, that belong to the same router). Similarly, the goal in subnet detection is to identify multiple links that appear to be separate and combine them to represent their corresponding single hop connection medium (e.g., point-to-point or multi-access links). As a result, the inclusion of this task will improve the accuracy and the completeness of the constructed topology map. They also developed an approach to infer subnets in a given set of path traces. Their dynamic subnet inference approach analyzes IP addresses and performs additional probing, if necessary, to group IP addresses into their corresponding genuine subnets. Moreover, inferred subnets help in identifying significant number of single-hop connectivity. Successful identification of the subnet helps in combining IP addresses over a multi-access link in the constructed topology map.

## 2.2 IP Alias Resolution

Several studies pointed out the impact of incomplete IP alias resolution in certain measurement studies [15, 73]. In [37], Gunes et. al. perform an experimental study on the impact of IP alias resolution on various topological characteristics. Varying the resolution success rate, we analyzed over 20 different graph characteristics including topology size, node degree, degree distribution, joint degree distribution, characteristic path length, betweenness, and clustering. The results indicate that the IP alias resolution process has a significant impact on almost all topological characteristics that we consider. Therefore, Internet measurement studies should employ all the means possible to increase the accuracy/completeness of the IP alias resolution process.

Several mechanisms have been proposed to resolve IP aliases [18, 31, 57, 65, 68] and few tools have been developed including ally [1] and iffinder [4]. These tools use an active probing approach to resolve IP aliases. They are easy-to-use and provide a convenient way to verify if a given pair of IP addresses is alias or not. As Gunes et al stated in [36, 37], these tools depend on the participation of the routers in terms of responding to the queries directed to themselves. This dependence introduces limitations to the success of IP alias resolution task as some network administrators configure their routers to ignore active probes directed to them. As an example, in their experiments, they observed that 40 percent of 7073 IP addresses that they probed with ally did not return a response.

The initial work on IP alias resolution utilizes source IP addresses of ICMP error replies [57]. It assumes that when a router generates an ICMP error message to be sent to a remote system S, the router uses the IP address of the interface that is on the shortest path to S as the source IP address in the ICMP error message. However, this implementation is not common as most routes copy the IP address from the original packet. *Mercator* [31] and *iffinder* [4] probe a given IP address and check whether the returned ICMP error message arrives from another IP address to identify aliases.

The second approach relies on the IP identification field value in the IP protocol header of the returned ICMP error messages [67]. When an IP packet is generated, the kernel puts a 16-bit value into the IP identification field in the IP header. Typically the value is implemented as a monotonically increasing counter. Since IP identification number is monotonically incremented, successive packets originating from the same router have consecutive values. After sending probe messages to different IP addresses, one can analyze whether the returned packets have consecutive or close by IP identification values. Given two IP addresses, *ally* sends successive probe packets to each of the two address, and a third packet to the address that responds first [66]. If the responses have IP identification values in sequence with a small difference in between, they are likely to be aliases and are classified as *alias*. It is is also possible that two responses from non-alias IP addresses will have close identification values.

Thus, repeating the test may reduce false positives. Since this method requires $O(n^2)$ probes to test all possible pairs, several approaches have been deployed to reduce the number of probes [14, 28]. Moreover, as some routers do not respond to certain probes, [29] has proposed alternative probes to increase the elicited responses from routers.

Another approach to resolve IP aliases is to use the record route option of IP protocol. When record route option is enabled, a packet will include up to 9 hops of IP addresses belonging to the routers it has traversed. This can help in identifying IP aliases as the recorded interfaces are usually the outgoing interfaces while traceroute returns the incoming interfaces [4]. Record route based method is not very successful since routers generally drop packages with record route option set. In addition, there is not a guideline specifying which IP address a router should put into the header and different vendors have varying implementations. *Sidecar* has presented an approach to benefit from this option by performing the measurement using data package [65]. [64] proposes a disjunctive logic programming approach to decide on the proper alignment of collected path traces in order to infer IP aliases. A significant disadvantage of the method is that the disjunctive logic programming is computationally expensive.

Additionally, DNS based method relies on the similarities in the host names of routers and works when an AS uses a systematic naming convention in assigning DNS names to router interfaces [72]. This method can be successful even if a router does not respond to probes. However, the DNS based approach can be used only when a systematic naming scheme is used by the AS and the naming template is present in the database.

Gunes et al [34], conducted an experimental study on the impact of IP alias resolution on various topological characteristics. The results indicated that the completeness and the correctness of the IP alias resolution process has a considerable impact on almost all topological characteristics. Moreover, they present a new IP alias resolution algorithm called Analytic and Probe-based Alias Resolver (APAR) [35]. APAR consists of an analytical component [36, 37] and a probe-based component. Given a set of path traces, the analytical component utilizes the common IP address assignment scheme to infer IP aliases. In addition, APAR can optionally use a lightweight probing component (O(n) probes) to improve its accuracy. APAR is a more scalable approach to resolve IP aliases than the state-of-the-art probe based approach. Then, their experiments revealed that APAR could detect a significant number of alias pairs that ally tool fails to detect. However, being two orthogonal approaches, APAR and ally do not compete but complement each other in maximizing the success rate of the overall IP alias resolution process [33]. Recently, CAIDA announced they will release their tool KAPAR (modified from APAR) soon.

## 2.3 Unresponsive Router Resolution

Unresponsive router resolution is an inherent problem in traceroute based topology mapping studies. Most of the early work in the area ignored this problem or used simple heuristics to work around it [15, 16, 19]. In [19], authors avoid the problem by stopping a trace toward a destination on encountering an unresponsive router on the path. In [16], authors handle unresponsive routers by replacing them either (1) with arcs (to connect the known routers at two ends) or (2) with unique identifiers to treat them as separate nodes. Finally, in [15], authors use a sandwich approach to merge a chain of unresponsive nodes between the same pair of known nodes with each other. These approaches cause either loss of potentially useful connectivity information (as in [19]), or inaccuracies in the resulting topology maps (as in [16]), or limited resolution in the resulting topology maps (as in [15]).

The first study analyzing the problem in depth formulates it as an optimization problem [77]. Their goal is to build a minimum size topology by merging unresponsive nodes under two conditions: (i) *trace preservation condition*, i.e., there should not be a routing loop due to merging two unresponsive nodes, and (ii) *distance preservation condition*, i.e, the unresponsive router resolution process should not reduce the length of a shortest path between any two nodes in the resulting topology map. They prove that the optimum topology inference under these conditions is NP-complete and propose a heuristic to minimize the constructed topology by identifying unresponsive nodes that, when merged, satisfy the two conditions. The main limitation of this approach is its high complexity, i.e. $O(n^5)$ where $n$ is the number of unresponsive nodes, that significantly limits its practicality in real life scenarios. Additionally, the claimed distance preservation condition is not necessarily accurate as inter domain routes may not always be the shortest routes.

Later, an ISOMAP based dimensionality reduction approach that uses link delays or node connectivity as attributes in the dimensionality reduction process is proposed in [45]. The main limitation of this approach is its high complexity, i.e., $O(n^3)$ where $n$ is the size of the topology. In addition, the link delay based approach is not practical as they ignore the difficulty of estimating individual link delays from round trip delays in path traces [27]. The authors also propose a simple neighbor matching heuristic with a smaller time complexity, i.e., $O(n^2)$. As the authors mention in their paper, this approach may introduce a high rate of false positives and false negatives.

Recently, Shavitt [10] described a method of recreating an IPlevel graph, in which the unknown nodes are not overrepresented by unifying groups of unknowns, that are likely to stem from the same unknown root. They introduced a novel clustering algorithm based on semi-supervised spectral embedding followed by unsupervised clustering in the projected space. Applying their algorithm to placeholder maps yields IP-level maps that are much more meaningful and usable. By merging groups of unknowns, they arrived at a good approximation of the real graph, which can then be used to study the properties of the IP-level graph. The improved performance of the proposed method is demonstrated on large real internet data collected by the DIMES project [3]. However, it assumes that majority of nodes are known nodes, which generally is not the case in sampled topologies.

Gunes et al. [39] added a new dimension to the problem by classifying the anonymity types. They observe five different scenarios that cause routers to stay unresponsive. In three of these scenarios we never get a response from the router, while in two of the scenarios a router might sometimes respond or stay unresponsive. They then formulate a number of graph structures that can be found in traceroute-based topologies collected from the Internet. Namely, Parallel/Symmetric *-substrings, Clique, Complete Bipartite, and Star structures are defined. Based on this formulation, they introduced a graph based induction technique where they search for structures similar to the identified ones in the topology and then reduce the occurrences of unresponsive nodes into their corresponding routers. Graph based induction is a technique to obtain information from a graph in data mining field [53].

# Chapter 3

# Structural Graph Indexing

Complex systems such as proteins, chemical compounds, and the Internet are being modeled as complex networks to identify local and global characteristics of the system. In many instances, these graphs are very large in size presenting challenges in their analysis. Hence, graph indexing techniques are developed to enhance various graph mining algorithms. In this section, we propose a new Structural Graph Indexing (SGI) technique that does not limit the number of nodes in indexing to provide an alternative tool for graph mining algorithms. As indexing feature, we use common graph structures, namely, star, complete bipartite, triangle and clique, that frequently appear in protein, chemical compound, and Internet graphs. Note that, SGI lists all substructures matching structure formulations and other graph structures can be identified and added to the SGI.

## 3.1 Introduction

Many systems can be modeled as a complex network to understand local and global characteristics of the system. Studying network models of systems provides a new direction towards a better understanding biological, chemical, technological or social systems. In many cases, the systems under investigation are very large and the corresponding graphs have large number of nodes/edges requiring graph mining techniques to derive information from the graph. Several graph mining techniques have been developed to extract useful information from graph representation and analyze various features of complex networks [22]. In order to speed up graph queries, usually an index of the graph is derived according to some predefined index features.

Graph indexing is often utilized by graph search algorithms that look for a sub-graph within a graph database. For example, given a graph database G=$\{g_1, g_2, ..., g_n\}$ and a subgraph $s$, we are interested in identifying all graphs $g_i$ that contain the subgraph $s$. This query is shown to be NP-complete [30] and becomes challenging as the size of graphs increase. For example, a typical graph of router-level Internet consists of millions of nodes making it impractical to perform many operations on the whole graph. In such cases, graph indexing allows operations to be more efficient.

In this chapter, we propose a new structural indexing approach to provide an alternative tool for graph mining algorithms. For indexing, we specify a set of common graph structures such as star, complete bipartite, triangle and clique. These structures are ubiquitous in biological, chemical, technological, and social networks. In order to reduce computational complexities, we index these structures within the original graph in a consecutive manner. We first identify star structures, and then the complete-bipartite, triangle and clique structures from the preceding ones. Our approach, unlike previous ones, does not limit the size of the subgraph considered in indexing. However, it may be limited as maximum clique search is an NP-complete problem [30].

## 3.2 Related Work

The need for mining large graphs in an efficient manner increases as researchers look into new complex networks. Several studies have been carried out to make graph mining in an efficient manner using indexing techniques [20,23,44,47,56,62,76]. Many of the tools have constraints that limit the number of nodes/edges in index subgraphs or are not capable of operating on very large graphs. The graph indexing studies can be mainly categorized into two categories, namely, *path-based* and *structure-based* approaches.

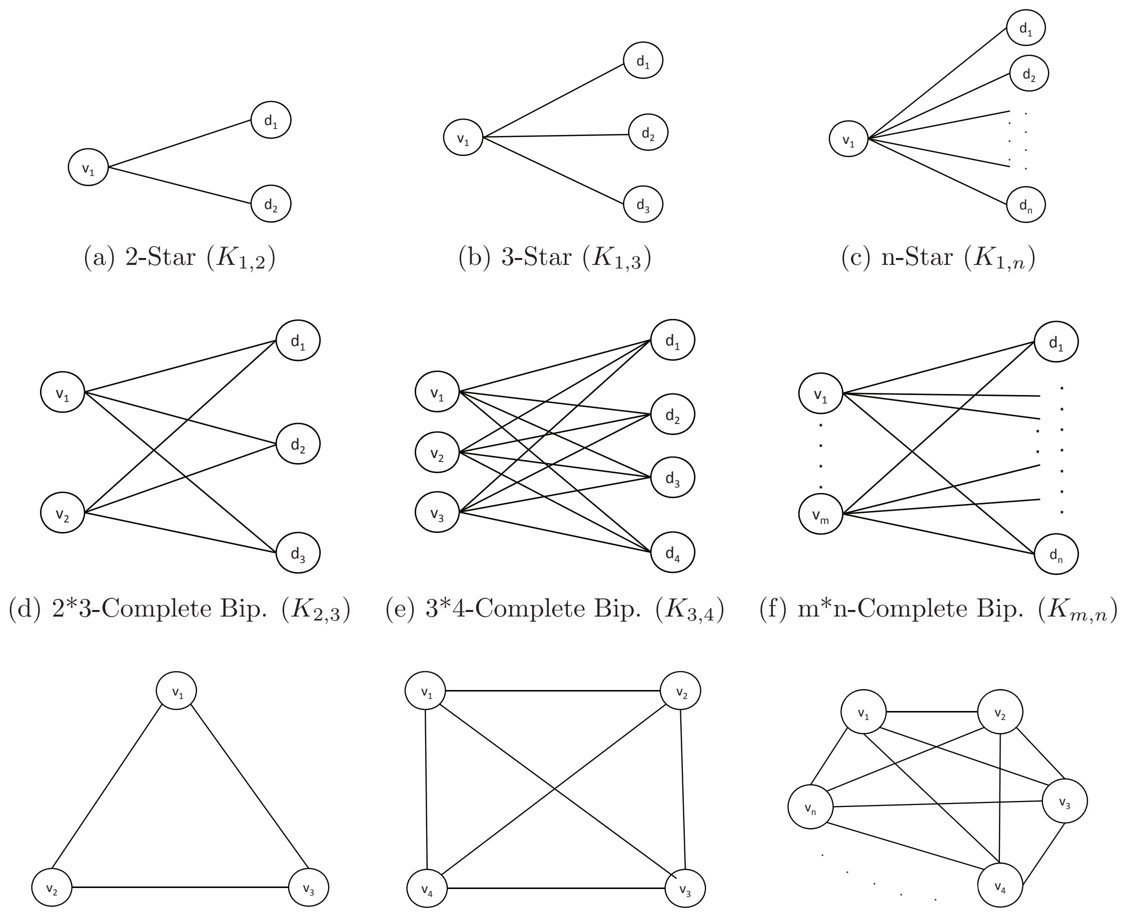
Path-based graph indexing approaches use path expressions as indexing features such as GraphGrep [62] and Daylight [44]. GraphGrep enumerates all paths in the graph up to the length $maxL$. Then, it looks for each graph $g_i$ whether it contains all paths up to $maxL$ for a graph query $q_i$. A significant feature of path-based approaches is that paths can be manipulated more easily than general graphs. However, as Yan et al. indicated, a path is a simple structure loosing structural information of a graph, and the hence false positive ratio of path-based methods would be very high [76]. In addition, the number of paths in a graph database increases exponentially making path-based methods impractical for very large graphs.

Alternatively, structure-based graph indexing approaches identify subgraphs to be indexed as in gIndex [76]. gIndex first searches for the frequent subgraphs in the graph, then indexes these frequent structures. An issue in this case is that frequent subgraph discovery increases complexity and exponential number of frequent fragments may exist under low support. Therefore, they limit the number of nodes and index frequent structures up to 10 nodes in their study.

In this thesis, we propose an alternative structural indexing approach to search and process queries efficiently even in very large graphs. As indexing features, we use commonly observed graph structures: star, complete bipartite, triangle and clique. An important feature of these structures is that each one is comprised from the previous one where clique contains complete bipartite structures and complete bipartite contains star structures.

## 3.3 Structure Models

In structural indexing, we index predefined structures that are commonly observed in complex networks. In particular, we index star, complete bipartite, triangle and clique structures (shown in Figure 3.1) in a given graph $G = (V, E)$. An important difference of our approach from the previous studies is that we do not limit the size of subgraph considered in indexing. We index all maximal graphs that match the structure formulation. For instance, a maximal clique is a clique that cannot be extended by adding one more vertex from the graph. However, the substructure size in indexing may be limited when needed since maximal clique search is known to be NP-complete [30]. In order to reduce computational complexities, we index the structures within the original graph in a consecutive manner. That is, we first identify star structures, and then the complete-bipartite, triangle and clique structures from the preceding ones as detailed below.



(a) 2-Star ($K_{1,2}$) (b) 3-Star ($K_{1,3}$) (c) n-Star ($K_{1,n}$)

(d) 2*3-Complete Bip. ($K_{2,3}$) (e) 3*4-Complete Bip. ($K_{3,4}$) (f) m*n-Complete Bip. ($K_{m,n}$)

(g) 3-Clique ($K_3$) (Triangle) (h) 4-Clique ($K_4$) (i) n-Clique ($K_n$)

Figure 3.1: Structural Models

### 3.3.1 Star Structure ($K_{1,n}$)

We first index the star structure where a node has multiple neighbors as shown in Figure 3.1-(a),(b) and (c). All star structures within a graph $G = (V, E)$ are represented as $s(v_i, ns_i)$ where $v_i \in V$ and $ns_i$ is the set of all neighbors of $v_i$. We index maximal star structures for each node using the algorithm presented in Figure 3.2. The algorithm first builds a star structure $s_{(v,\emptyset)}$ for each node $v$ without any neighbors. Then, for each edge $e(a, b)$, it appends neighbor sets of nodes $a$ and $b$ to the other

```
Let $G = (V, E)$; $S \leftarrow \emptyset$;
**for** (each *node* $v \in V$)
    $S \leftarrow S \cup s_{(v,\phi)}$
**for** (each *edge* $e(a, b) \in E$)
    $s_{(a,ns)} \leftarrow s_{(a,(ns \cup \{b\}))}$
    $s_{(b,ns)} \leftarrow s_{(b,(ns \cup \{a\}))}$
**for** (each $s_{(v,ns)} \in S$)
    **if** $|ns| < 2$
    $S \leftarrow S - s_{(v,ns)}$
```

Figure 3.2: Alg. 1 - Star Structure Indexing

one. Finally, the algorithm removes star structures $s_{(v,ns)}$ that have less than two neighbors.

### 3.3.2 Complete Bipartite Structure ($K_{m,n}$)

The second structure we index is complete bipartite, shown in Figure 3.1-(d), (e) and (f). A complete bipartite graph $G = (V_1 \cup V_2, E)$ is a bipartite graph such that $V_1$ and $V_2$ are two distinct sets and for any two vertices $v_i \in V_1$ and $v_j \in V_2$, then there is an edge between them (i.e., $\exists\, e^*_{(v_i,v_j)} \in E$). The complete bipartite graph with partitions of size $|V1| = m$ and $|V2| = n$ is denoted as $K_m, n$. Note that, star structure is a special case of a bipartite graph (not necessarily complete) where $m = 1$. Moreover, finding the complete bipartite subgraph $K_m, n$ with the maximal number of edges $m.n$ is an NP-complete problem [58].

We index all complete bipartite structures in the graph $G$ using indexed star structures as in Figure 3.3. In the algorithm, for each star structure $s_{(a,ns)}$ where $ns$ is the set of all neighbors of the node $a$, we identify the maximal complete bipartite involving the node $a$. For this purpose, we find second hop neighbors of $a$ by first iterating over the $ns$ set and then unifying them under the set $L_{can}$ that indicates

```
**INPUT:** $S$ from Alg.1 in Figure 3.2
Let $G = (V, E)$; $K \leftarrow \emptyset$
**for** (each $s_{(a,ns)} \in S$)
  $L_{can} \leftarrow \phi$
  **for** (each $b_i \in ns$)
    $L_{can} \leftarrow L_{can} \cup ns^*$ where $\exists\ s_{(b_i,ns^*)} \in S$
  $L_{can} \leftarrow L_{can} - \{a\}$
  $R_{can} \leftarrow ns$

  **for** (each $v_i \in L_{can}$)
    $R_{new} \leftarrow R_{can} \cap ns_i^+$ where $\exists\ s_{(v_i,ns_i^+)} \in S$
    **if** ($|R_{new}| \geq 2$)
      $L_{new} \leftarrow \{a\} \cup \{v_i\}$
      **for** (each $v_j \in L_{can}$)
        **if** ($R_{new} \subset ns_j^{\#}$ where $\exists\ s_{(v_j,ns_j^{\#})} \in S$)
          $L_{new} \leftarrow L_{new} \cup \{v_j\}$
      $K \leftarrow K \cup k_{(L_{new},R_{new})}$
```

Figure 3.3: Alg. 2 - Complete Bipartite Structure Indexing

candidates for the left side of the complete bipartite while the set $ns$ is the candidate set for the right hand side. Next, we first find a $K_{2,n}$ and then grow it to $K_{m,n}$. In finding $K_{2,n}$, we iterate over each candidate node in the set $L_{can}$ and determine the neighbor intersection with $a$. If the intersection set is larger than two, then these nodes belong to the right hand side. In the second step, we grow the $K_{2,n}$ by finding all nodes in the left hand side (i.e., $L_{can}$) that has the right hand side nodes (i.e., $R_{new}$) as a neighbor.

The complete bipartite structure is ubiquitous in many complex networks. For example, [8] examines the structure of protein-protein interaction networks and showed that the graph of all protein-protein interactions is made up of complete bipartite structures.

```
**INPUT:** $S$ from Alg.1 in Figure
Let $G = (V, E)$; $T \leftarrow \phi$; $TS \leftarrow \phi$
**for** (each $s_{(a,ns)} \in S$)
  **for** (each $ns_i \in ns$)
    $TS \leftarrow ns \cap ns^*$ where $\exists\ s_{(ns_i,ns^*)}$
    **for** (each $ts_i \in TS$)
      $T \leftarrow T \cup t_(a, ns_i, ts_i)$
```

Figure 3.4: Alg. 3 - Triangle Structure Indexing

### 3.3.3 Triangle Structure ($K_3$)

Third, we index the triangle structure which is a clique of three nodes as shown in Figure 3.1-(f). We index all triangles in the graph by iterating over the star structures as in Figure 3.4. In the algorithm, for each star structure $s_{(a,ns)}$, and $s_{(ns_i,ns^*)}$ where $(ns_i \in ns)$, we take the intersection set $TS$ of the sets $(ns)$ and $(ns^*)$. Thus, for each $(ts_i \in TS)$, $(a, ns_i, ts_i)$ constitutes a triangle.

Counting the number of triangles in a graph has become more important in recent years as complex network analysis gained importance. Several important complex networks metrics, such as the clustering coefficient and the transitivity ratio, involve the execution of a triangle counting algorithm. Especially in social networks, the triangle motif has been studied extensively.

### 3.3.4 Clique Structure ($K_n$)

Finally, we index clique structures shown in Figure 3.1-(g) and (h). A clique in graph $G = (V, E)$ is a subset of the vertex set (i.e., $C \subseteq V$) such that there are edges between all node pairs (i.e., $\forall (c_i, c_j) \in C, \exists e_{(c_i,c_j)} \in E$, when $i \neq j$).

We index all maximal clique structures (that has more than three nodes) in the graph using complete bipartite structures as in Figure 3.5. We first get the set

```
**INPUT:** $K$ from Alg.2 in Figure 3.3
Let $G = (V, E)$; $C \leftarrow \phi$
for (each $k_{(m,n)} \in K$ )
   for (each $a \in k_{(m,n)}$)
   findCliques($\{a\}, k_{(m,n)} - \{a\}$)

FUNCTION findCliques(L1, L2)
   if ($|L2| = 0$ and $|L1| > 3$)
      $C \leftarrow C \cup c_{(L1)}$
   else
      for (each $b \in L2$)
         if ($\exists e_{(b,v)}\ \forall v \in L1$)
            $L^* \leftarrow L2 \cap ns_i$ where $s_{(b,ns_i)} \in S$
            findCliques($L1 \cup b, L^*$)
```

Figure 3.5: Alg. 4 - Clique Structure Indexing

of nodes from each complete bipartite $k_{(m,n)}$ and look for cliques that are formed by those nodes. Note that, any clique larger than three nodes in the graph $G$ will be indexed as multiple bipartite structures. Hence, we do not need to consider all nodes in the graph when indexing maximal clique structures. The clique search algorithm works recursively on each node from the $k_{(m,n)}$ as the pivot node in the set $L1$ and considers other nodes as candidate nodes in the $L2$ set. The function, moves each node from the set $L2$ to the set $L1$ if it is connected to all nodes in the set $L1$ and then recursively tries to grow the structure with remaining nodes as candidates. When there are no more candidates to consider in the set $L2$ then a clique has been identified. Note that, this algorithm is not optimal and better solutions for finding all cliques are proposed in [17, 60].

This structure has been observed in many fields. For example, many problems in computational biology can be solved by finding maximal or all cliques within the graph [13]. Similarly, [61] models protein structure prediction as a problem of finding cliques in a graph whose vertices represent positions of subunits of the protein; [21] finds a hierarchical partition of an electronic circuit into smaller subunits using cliques; and [59] uses cliques to describe chemicals in a chemical database that have a high degree of similarity with a target structure.

## 3.4 Evaluation on Internet Topologies

In this section, we use router level Internet topologies to analyze the structural graph indexing approach.

### 3.4.1 Preliminaries

The need for accurate Internet topology samples has increased over the last decade. Sample maps provide an understanding of the global topology helps developing protocols and applications that can adapt to the underlying network. Furthermore, router-level Internet maps are useful to analyze the topological characteristics of the Internet and to design topology generators that can produce Internet-like synthetic network topologies for simulation studies [32]. The confidentiality of Internet Service Provider topologies faces the research community to use other ways to collect Internet topologies. Several research groups and institutions have developed various tools and methodologies [2, 6, 50, 63, 69] to collect the required topology information from the Internet. These measurement studies utilize the Internet debugging tool, traceroute, or its variants to collect a large number of path traces from a set of vantage points.

Nonetheless, constructing a router level Internet Topology map using traceroute is not a straightforward process. Some routers don't respond to the traceroute probes and are called unresponsive routers. They are denoted by a '*', instead of an

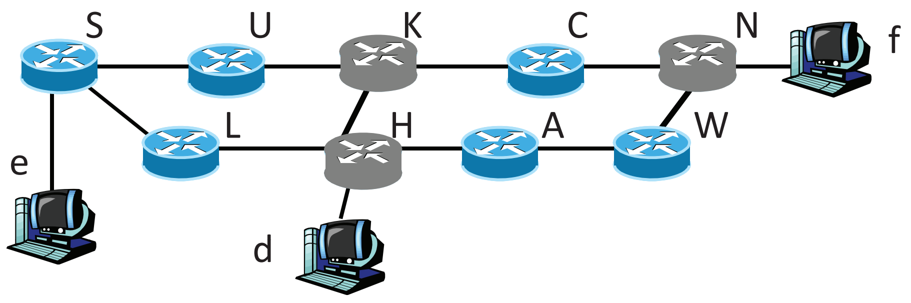
IP address, in a traceroute output. An unresponsive router may appear in several path traces. Each occurrence of '*' needs to be treated as a potentially different router. Therefore, there is a need to identify unresponsive nodes that are caused by the same router to be able to accurately construct the underlying network.

For instance, Figure 3.6 presents the effect of unresponsive routers on a sampled network. In the topology in Figure 3.6-(a), we assume that routers H, K and N are unresponsive routers. When we run traceroute queries between vantage points, represented as d, e, and f in Figure 3.6-(a), the collected path traces are as follows

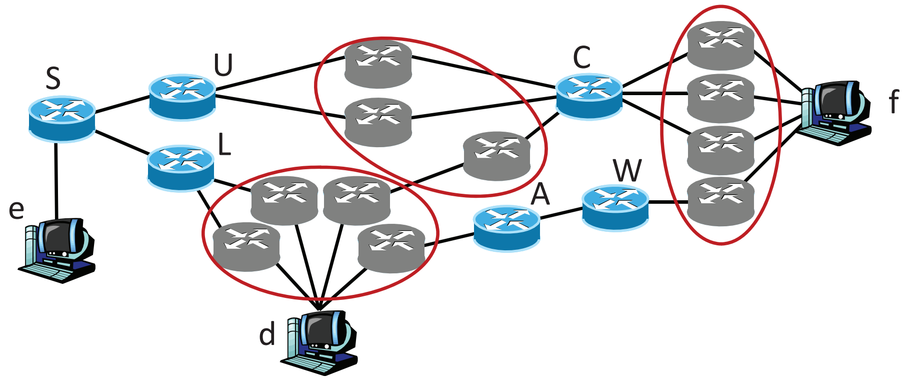
```
d - * - L - S - e
d - * - A - W - * - f
e - S - L - * - d
e - S - U - * - C - f
f - * - C - * - * - d
f - * - C - * - U - S - e
```

The collected path traces, will generate the topology shown in Figure 3.6-(b), which is considerably different than the underlying topology in Figure 3.6-(a). This demonstrates the significance of unresponsive router resolution to obtain more accurate sample topologies.

Even a small number of unresponsive routers may considerably alter the observed topology from the sampled network [39]. Based on the number of unresponsive routers in the topology and the method of topology collection, there may be a huge number of unresponsive nodes in the collected set of path traces. For example, daily Internet topologies collected by iPlane infrastructure have more than 12M unrespon-



(a) Sampled Network



(b) Resulting Network

Figure 3.6: Unresponsive Routers

sive nodes along with 400K known interfaces [5].

In [39], Gunes et al. determined different cases under which unresponsive nodes appear. They identified that unresponsive routers cause parallel, clique-like, bipartite-like, and star structures in observed topology. Then, they proposed a graph based induction method to handle unresponsive routers within collected path traces. The structural graph indexing proposed in this chapter can significantly facilitate the graph based induction approach in resolving unresponsive routers.

## 3.5 SGI on Unresponsive Router Resolution

In this section, we illustrate how our method can be applied to the router level Internet Topology to resolve unresponsive routers.

### 3.5.1 Graph Transformation

In our experiments, we use iPlane datasets [50]. These datasets have two types of nodes, namely known (i.e., the ones with an IP address) and unknown (i.e., unresponsive). When there is an unresponsive router, it will appear as an asterisk in the traceroute output. If multiple path traces pass through an unresponsive router between routers with known IP address, there will be multiple parallel *-substrings between these known nodes. As an example, in Figure 3.6-a, traceroute queries from $e$ to $f$ will return path traces including *-substrings as $(U, *_1, C)$ and $(U, *_2, C)$ respectively. This may then result in two parallel *-substrings between $U$ and $C$ in the resulting topology map as shown in Figure 3.6-b. In the example, there is a *-substring that includes only one unresponsive router. A similar pattern can be observed for *-substrings of larger lengths in a typical dataset.

In order to resolve this type of unresponsive routers, we need to detect the same *-substrings (i.e., same length *-substrings with the same known nodes at the end points) [39]. In our method, we ignore all nodes which do not have any unresponsive neighbor, since they do have no effect for the resolution process. While reading the traces from iPlane database, we identify the unresponsive routers which are in between two known nodes. We read all *-substrings from the database and construct a new graph $\bar{G} = (\bar{V}, \bar{E})$. We represent each *-substring as an edge e(a,b,l) where $a$ is the first known node, $b$ is the second known node and $l$ is the label of the edge representing

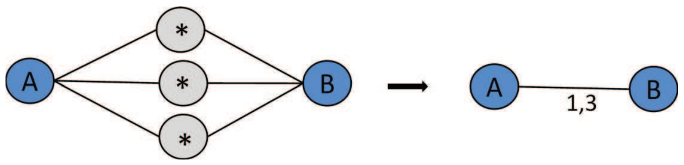


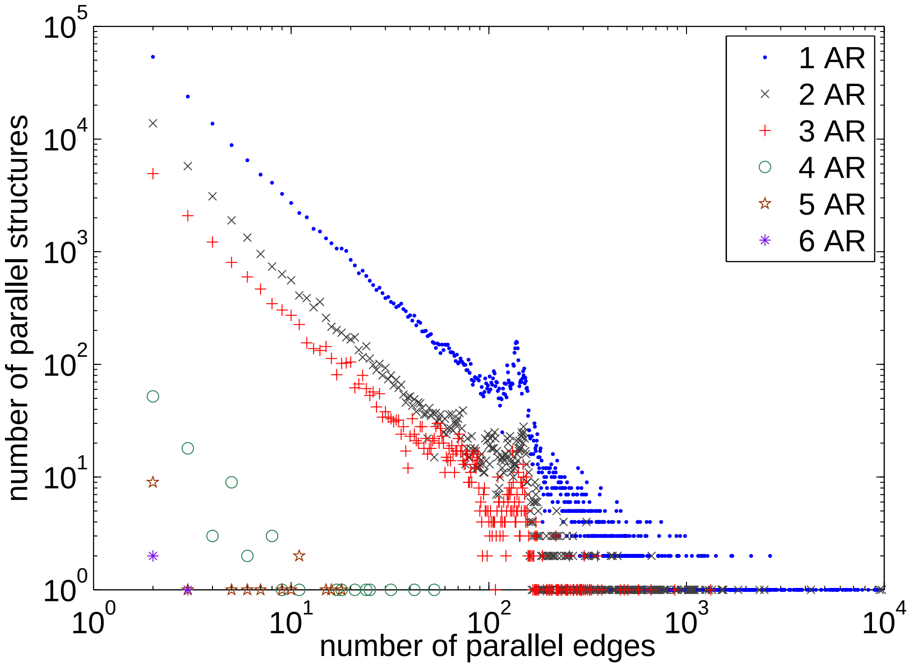
Figure 3.7: Sample Transformation

the number of unresponsive nodes between $a$ and $b$ as in Figure 3.7. We add each $e(a, b, l)$ only once to our new graph $\bar{G}$ and add $a$ and $b$ to $\bar{V}$. This process can be called as initial pruning (IP). Next, we sequentially index star, complete bipartite, triangle and clique structures in graph $\bar{G}$ with SGI algorithm.

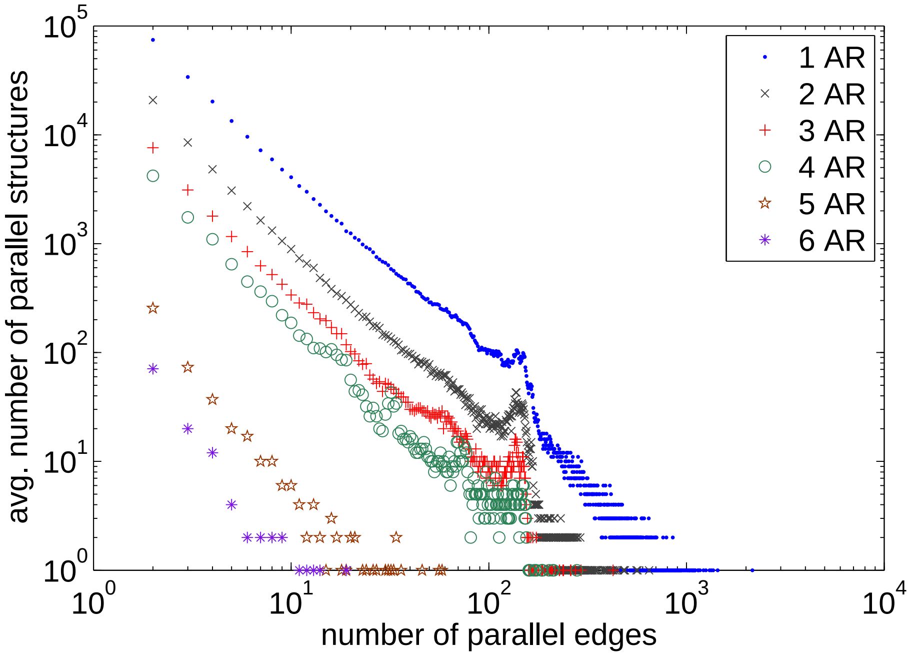
### 3.5.2 Structure Statistics

We present the indexing results for a dataset collected in 2006 and the average indexing results of 6 different datasets collected between 2006-2009 in Figures 3.8, 3.9, 3.10 and 3.11. For the dataset collected in 2006, we have 7,043,618 unresponsive nodes and 172,532 known nodes.

When the length of *-substring increases (i.e. we have more unresponsive nodes between two known node), the number of structures found within the topology decreases. We present only the number of structures with less than 7 unresponsive nodes as for higher lengths the number of structures found is almost zero. According to our experiments, while the number of structures with 1 unresponsive node has not changed considerably in consecutive years, there has been a significant increase in the number of structures with three or more unresponsive routers.

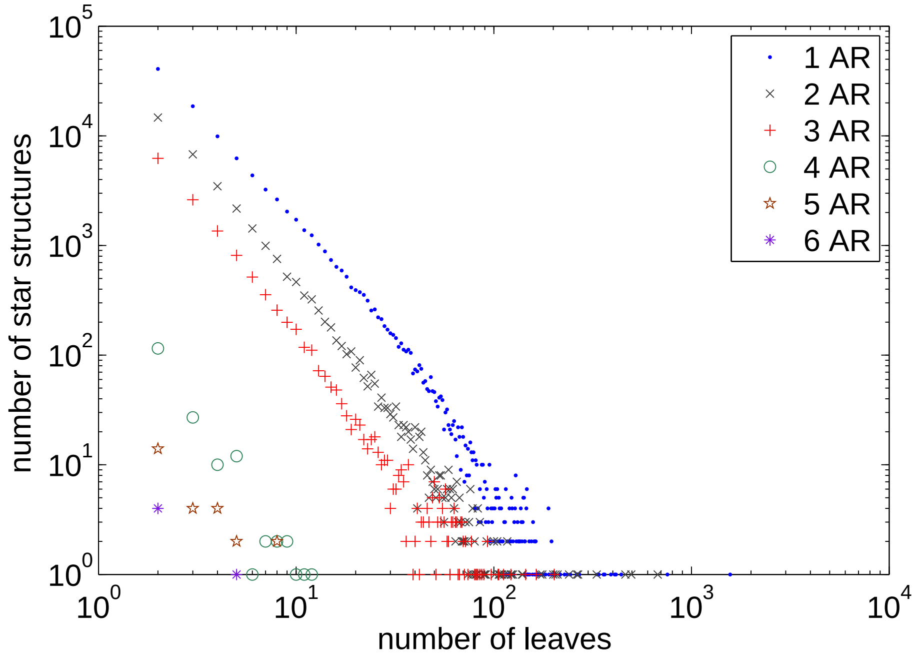


(a) 2006

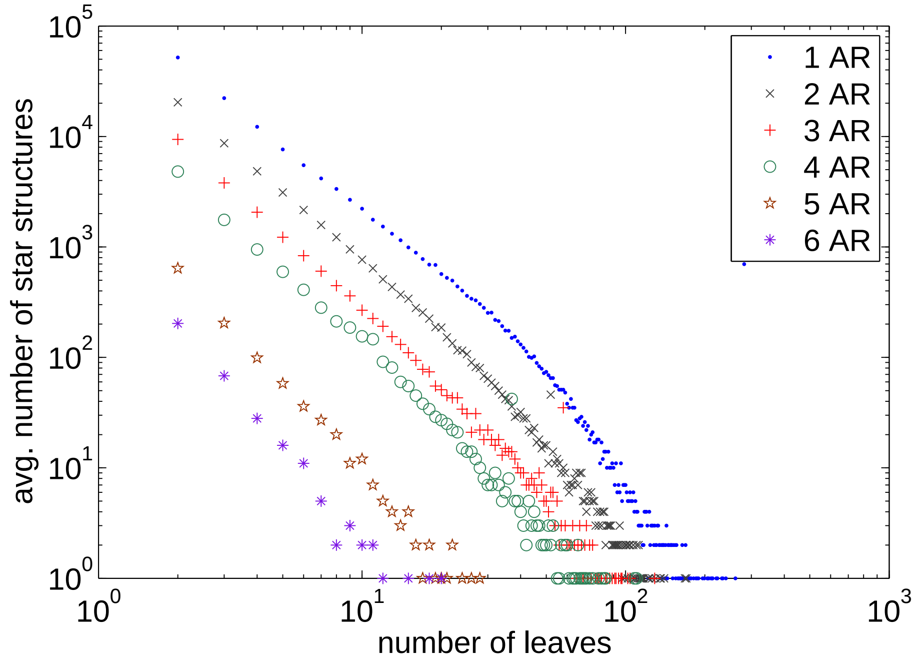


(b) Average

Figure 3.8: Parallel Structures in the Internet Topology

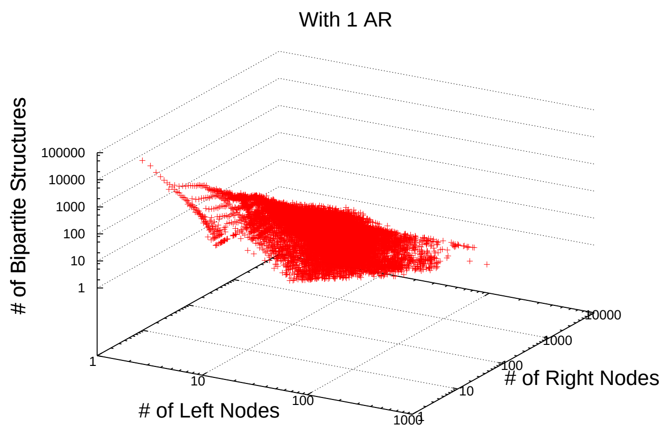


(a) 2006

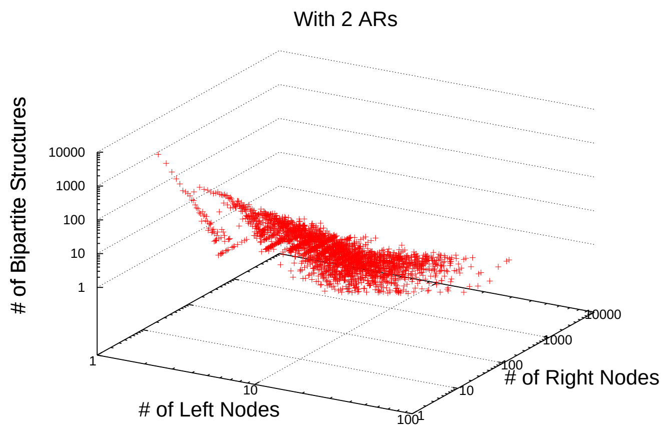


(b) Average

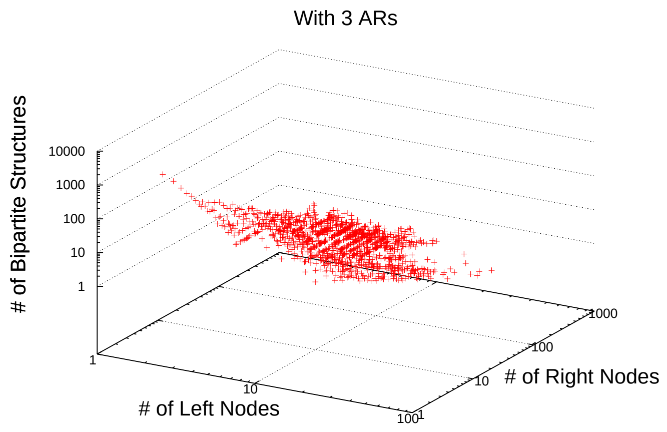
Figure 3.9: Star Structures in the Internet Topology



(a) 2006 (1 unresponsive node)

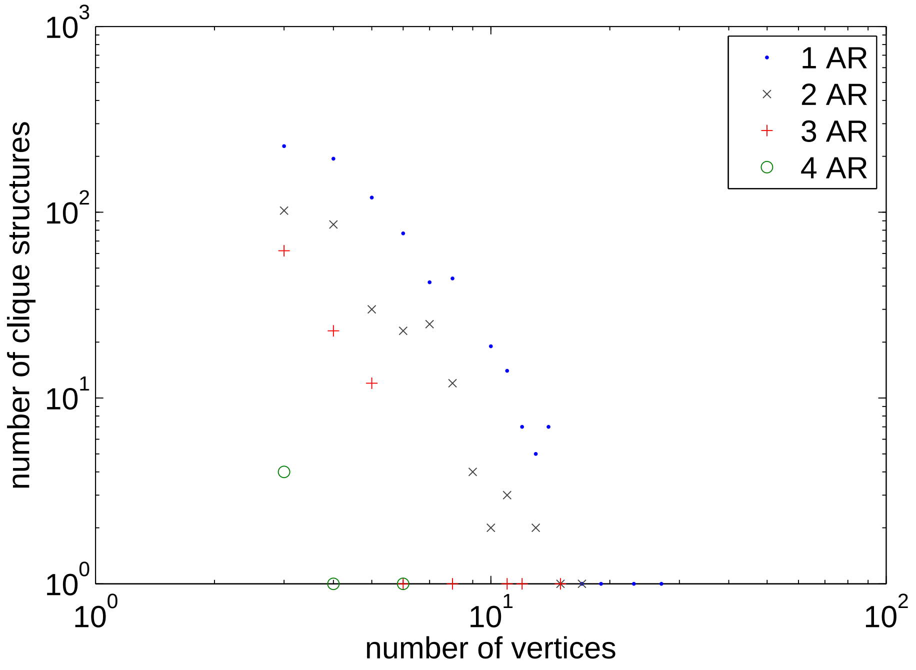


(b) 2006 (2 unresponsive nodes)

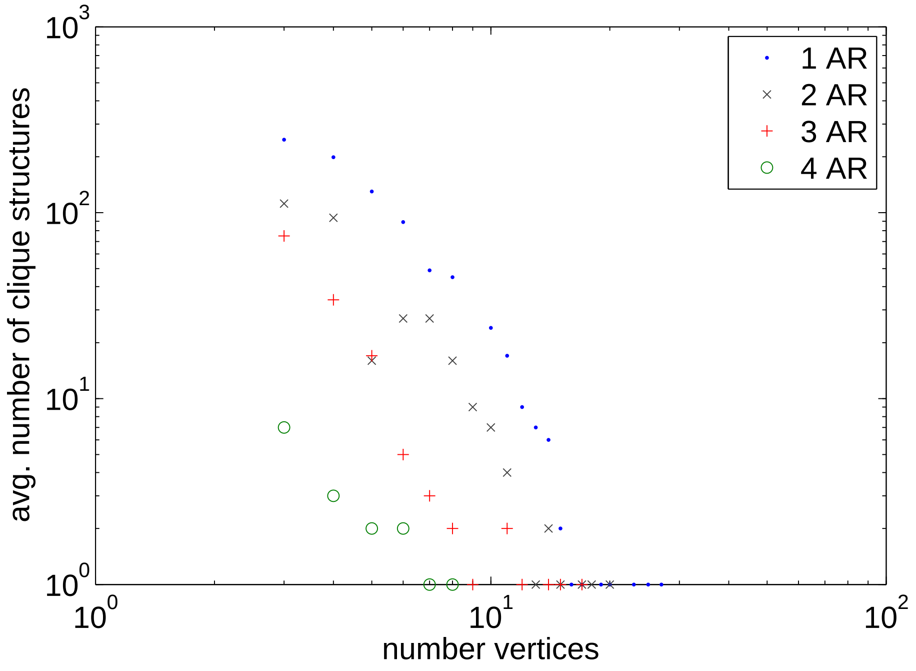


(b) 2006 (3 unresponsive nodes)

Figure 3.10: Complete Bipartite Structures in the Internet Topology



(a) 2006



(b) Average

Figure 3.11: Clique Structures in the Internet Topology

| | #Unres. nodes | #Resolved |
|---|---|---|
| IP | 7,043,618 | 6,769,486 |
| Clique | 274,132 | 628 |
| Bipartite | 273,504 | 92,121 |
| Star | 181,383 | 78,214 |
| Final | 103,169 | 6,940,449 |

Table 3.1: SGI for Resolving Unresponsive Routers in the iPlane Data-set

### 3.5.3 Resolution Results

We have 7,043,618 unresponsive nodes in the initial data. We first resolve the unresponsive nodes between two known nodes. Then we use the SGI algorithm to index the structures and our resolution algorithms presented in Chapter 4 unresponsive nodes. Starting from the maximum clique, we resolve all unresponsive nodes within the clique and triangle structures. Then, we resolve unresponsive nodes within the complete bipartite and star structures. The number of resolved unresponsive nodes at each step given in Table 3.1. Finally, using SGI, we resolve more than 98 percent of unresponsive nodes.

## 3.6 Observations

After applying our Structural Graph Indexing (SGI) approach to resolve the unresponsive routers in the Internet topology collected by iPlane [50], we made two main observations. First of all, the number of unresponsive nodes resolved during the clique resolution step is very low despite the high complexity of clique indexing. Hence, we decided to use triangle resolution instead of clique resolution and then we added the triangle indexing and resolution modules to our system. Secondly, we realized that resolving unresponsive routers without identifying the IP alias pairs significantly affects the final graph. Therefore, in Cheleby, we perform the unresponsive router resolution step after identifying the IP alias pairs.

# Chapter 4

# Cheleby: An Internet Topology Mapping System

In this part, we present Cheleby, an Internet topology mapping system that provides insight into the Internet topology by taking daily snapshots of the underlying networks. The system utilizes efficient algorithms to process large scale data-sets collected from distributed vantage points and provides accurate topology graphs at link layer. Incorporating enhanced resolution algorithms, Cheleby provides comprehensive topology maps. In this thesis, we designed and developed topology construction part of the Cheleby system. Although the topology collection and sampling parts are beyond the scope of this thesis, we give an overview of Cheleby's topology sampling and collection steps as it significantly affects the work done in topology construction part and the final topology generated by topology construction phase. Additionally, presenting the Cheleby's topology sampling and collection strategies and then our approach facilitates and improves the presentation of the Cheleby system.

## 4.1 Topology Sampling

In order to sample the underlying topology of the Internet, we collect a large number of path traces from geographically diverse vantage points towards all /24 subnets in the Regional Internet Registries (RIR). An important issue affecting accuracy of collected path samples is that certain traffic engineering practices for load balancing may cause traceroute to return IP addresses that do not correspond to a real end-to-end path in the Internet. We utilize Paris traceroute, which fixes flow identifiers so that flow-identifier based load balancing routers will choose the same next hop for probe packets toward the same destination [12]. Moreover, we perform ICMP based querying as it elicits more responses than other probing approaches [42]. In the following, we describe major steps of Cheleby regarding topology sampling.

### 4.1.1 Destination List Generation

In order to probe each active /24 subnetwork range, we obtain RIR records from `http://www.cidr-report.org/as2.0/as-ms-prefixes.txt`. The list provides advertisements and actual RIR allocations for each AS. We divide each subnet advertisement into a /24 subnetwork (e.g., $A.B.C.0/24$) and pick first allocable IP address as the probing destination (i.e., $A.B.C.1$). If a specific range is smaller than /24, then we pick the first allocable IP address in the range as the destination. These IP addresses are divided into blocks of approximately 1,024 destinations that will be probed by monitors. Note that an AS may be divided into several blocks or a destination block file may contain multiple ASes. At the end of this process, we have **3,460** destination blocks, i.e., 3.54M destination IP addresses.

### 4.1.2 Task Assignment to Monitors

In order to probe destinations from geographically diverse vantage points, we utilize PlanetLab [7] nodes around the world. Among ∼1000 nodes only ∼600 of them were good to be utilized during our experiment. As ∼100 of good monitors did not function well with the Paris traceroute, we could utilize ∼500 nodes during our topology collection. We divided these nodes into **7 teams** based on their geographic locations (i.e., 1: North-West America, 2: North-Central America, 3: North-East America, 4: South America, 5: Western Europe, 6: Eastern Europe + Africa + West Asia, and 7: East Asia + Australia).

Cheleby dynamically assigns one of the available monitors from each team to probe destination blocks. Each block is probed by only one monitor at a time and overall by 7 monitors. Preventing concurrent probing of a destination IP address eliminates the possibility of a denial of service attack on the destination.

Each monitor is set to probe four destination blocks in parallel to reduce the overall round completion time. Each of the four monitor processes work independent of others. These processes are marked as *idle*, *busy*, or *inactive*. All processes in a monitor are *inactivated* when one of them returns its data in less than two minutes as this indicates a problem with the probing. They remain *inactive* for a period (i.e., 4 hours) before becoming *idle* and getting a new job. Moreover, monitors are ranked based on their task completion averages and Cheleby selects the top *idle* process from a team to assign a new destination block.

Probing of a monitor is terminated if it cannot complete its task within a period (i.e., 2 hours). In this case the monitor is penalized with a reduction in its ranking and brought to the *idle* state. The partially traced destination block is also added to the non-probed list for another trial by another monitor in the same team. If the new monitor, which reverses the order of destination IP addresses before probing, is also unable to complete probing in time, then the destination block is marked as partially completed and both of the partial traces are added to the database.

### 4.1.3 Probing Overhead Reduction

As the volume of active measurement practices increased in time, several researchers indicated the impact and overhead of active probing on the network and presented approaches to reduce the volume of unnecessary/redundant probes in measurement studies. To this end, the Doubletree algorithm prunes redundant probes to nodes that are close to the vantage point and to the destination since traces from a vantage point yield tree-like structures [25]. Similarly, AROMA analyzes the convergence of path traces to reduce number of probes in discovering network topology [48].

In Cheleby, we utilized a similar approach to reduce the number of probes. We reduce intra-monitor redundancy by performing partial traces to some destination IP addresses. Once we have a full trace to an IP address in an AS, we start successive traces from the hop distance $h_i$ of the ingress router (i.e., hop distance of the last IP address in the trace that did not belonging to the AS). If the first IP of a new trace has not appeared at the same hop distance $h_j$ in any of the earlier full traces to the AS, then we complete the trace by performing traceroute with the same flow-ID as the partial trace while limiting max hop to $h_j$. Otherwise, we do not complete the trace.

Additionally, to reduce inter-monitor redundant probing, a destination IP is probed by only one monitor of a team. Since the monitors in the same team are geographically close to each other, we expect their contribution to identify a new link/node to be small. Moreover, we are in the process of identifying ingress points

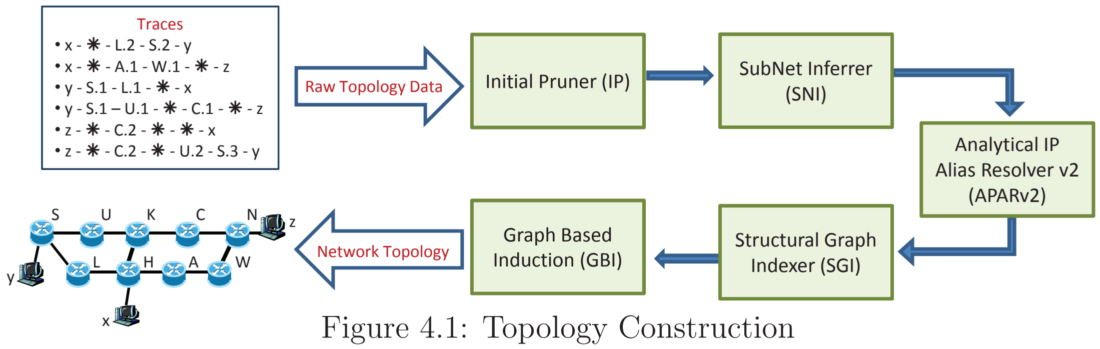

Traces
• x - ✱ - L.2 - S.2 - y
• x - ✱ - A.1 - W.1 - ✱ - z
• y - S.1 - L.1 - ✱ - x
• y - S.1 – U.1 - ✱ - C.1 - ✱ - z
• z - ✱ - C.2 - ✱ - ✱ - x
• z - ✱ - C.2 - ✱ - U.2 - S.3 - y
Raw Topology Data
Initial Pruner (IP)
SubNet Inferrer (SNI)
Analytical IP Alias Resolver v2 (APARv2)
Structural Graph Indexer (SGI)
Graph Based Induction (GBI)
Network Topology
S
U
K
C
N
z
y
L
H
A
W
x

Figure 4.1: Topology Construction

of ASes and dynamically establishing teams for each destination AS so that we have exactly one monitor probing through each ingress point of an AS.

## 4.2 Topology Construction

After collecting topology data, we need to process this raw data to obtain the underlying network topology. In Internet topology construction studies, there are several challenges, namely, unknown router resolution, identification of IP aliases and underlying subnets. As we mentioned in Chapter 2,there are several research groups working on these problems. Despite several approaches proposing solutions for one of the these issues, no study considers all these challenges together and proposes an approach to overcome these issues altogether. In Cheleby, first we analyze all of these issues and the relation between them. After our initial experiments on iPlane topology datasets, we found out that in order to resolve unresponsive routers within the Internet topology we first need to identify the IP alias pairs in the topology. Moreover, to reduce the computational complexity, we should infer the underlying subnets before identifying the alias pairs. Therefore, we (1) filter faulty traces, (2) infer underlying physical subnets among IP addresses, (3) resolve IP addresses belonging to the same router, and (4) resolve unresponsive routers as shown in Figure 4.1.

The accuracy and the completeness of these tasks may significantly affect the accuracy of the resulting topology maps [37, 73]. Moreover, when handling these tasks, one needs to make decisions based on the observations to infer the underlying topology. As the earlier decisions effect the later ones, obtaining the most likely topology under various conditions has been shown to be NP-hard [9]. Finally, these resolution tasks especially are challenging when large scale topologies of millions of nodes are processed. In this section, we analyze each of these tasks and present the algorithms that we utilized to handle them efficiently even in very large topologies consisting around 2.5M nodes and about 5M edges.

### 4.2.1 Initial Pruning

In this step we filter faulty traces, resolve the unresponsive routers in between the same known routers, and constitute our data structures from the raw data.

As path traces contain anomalies such as routing loops, we first prune raw path traces. The pruning breaks path traces with a loop (e.g., $IP_A$, $IP_B$, $IP_C$, $IP_D$, $IP_E$, $IP_C$, $IP_F$, $IP_G$) into three pieces based on the repeated IP address (i.e., $IP_C$) and utilize the first part (i.e., $IP_A$, $IP_B$, $IP_C$) and the last part (i.e., $IP_C$, $IP_F$, $IP_G$) of the trace in the remainder of processing. In collected path traces, 772K (%3.45) of path traces contain routing loops among which 143K has multiple loops. Moreover, we observed border firewalls that filter ICMP packets from/to a network domain and occasionally respond with their IP address. However, the hop distance of these IP addresses are not consistent. Hence, we filter any IP address that appears at the end of trace after 3 unresponsive nodes.

Then, we build initial network graph by parsing filtered path traces. During parsing, we resolve unknown nodes that are between the same set of known nodes.

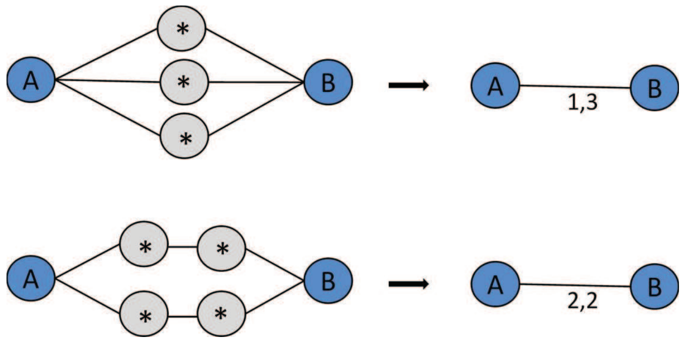


Figure 4.2: Sample Transformation

In order to resolve this type of unresponsive router, we need to detect the same *-substrings (i.e., same length *-substrings with the same known nodes at the end points). While reading the traces from the raw data, we identify the unresponsive routers between two known nodes. We read all *-substrings from the database and construct a new graph $\bar{G} = (\bar{V}, \bar{E})$. We represent each *-substring as an edge e(a,b,l) where $a$ is the first known node, $b$ is the second known node and $l$ is the label of the edge representing the number of unresponsive nodes between $a$ and $b$ as in Figure 4.2. We add each $e(a, b, l)$ only once to our new graph $\bar{G}$. In the successive resolution tasks, we use the graph $\bar{G}$. Performing this unresponsive router resolution step during graph construction reduces the number of unknown nodes by %78.71 on average. Additionally, we add all observed unique known and unresponsive routers to $\bar{V}$ and we constitute a neighbor set for each router.

This process can be called initial pruning (IP). After this step, we sequentially infer subnets, resolve IP alias pairs, and identify the unresponsive routers by first indexing star, complete bipartite, and triangle structures with the SGI algorithm presented in Section 3.1, and then resolve unresponsive routers within these structures with our graph based induction algorithms.

Table 4.1 presents the average statistics of (1) all traces, (2) partial traces, (3)

| | |
|---|---|
| All Traces | 22.36M |
| Partial Traces | 35.4% |
| Saved Probes | 66.15M |
| Unknown Nodes | 7.21 |
| Known Nodes | 1.18 |

Table 4.1: Probing Overhead and Initial Unresponsive Router Reduction

saved probes by using partial traces, (4) Unknown nodes, i.e., '*', and (5) Known nodes, i.e., IP addresses, for the analyzed data sets.

### 4.2.2 Subnet Inference

The first task after building an initial network graph is the identification of the underlying physical subnets, i.e., link level connectivity, among IP addresses in the collected topology [38]. The goal in subnet resolution is to identify multiple links that appear to be separate and combine them to represent their corresponding single hop connection medium (i.e., multi-access link). Subnet resolution also finds missing links between IP addresses that fall in the same subnet range but were not observed in path traces. The successful inclusion of subnet relations among the routers yields topology maps that are closer, at the link layer, to the sampled segments of the Internet.

Cheleby, enhances the subnet resolution approach presented in [38] by utilizing the distance preservation condition over multiple vantage points. The SubNet Inferrer module (SNI) observes distances of all IP addresses per vantage point and determines the IP address ranges that have similar distances to the vantage points. Unlike the initial approach in [38], we only allow one IP address to be closer to each of the vantage points. As the number of vantage points is increased, the distance condition can more accurately filter false subnets. After identifying subnets, we add edges between all routers of observed subnets. This process helps us to identify the missing

| Subnet Size | /24 | /25 | /26 | /27 | /28 | /29 | /30 | /31 |
|---|---|---|---|---|---|---|---|---|
| Count | 1 | 4 | 34 | 485 | 6,381 | 20,602 | 11,202 | 2,960 |
| Completeness | %28 | %25 | %23 | %23 | %25 | %36 | %100 | %100 |

Table 4.2: Average Subnet Statistics

edges and neighborhood relations between IP addresses that fall in the same subnet range but were not observed in path traces.

Table 4.2 presents averages of identified subnets and their completeness for subnets that had %20 of their IP addresses present in path traces. We present the detailed results in Chapter 5. Only about 15% of IP addresses are assigned to a subnet. The main reason for this small ratio is because we did not explore other IP addresses of candidate subnets.

### 4.2.3 IP Alias Resolution

After inferring underlying subnets, Cheleby resolves IP aliases. Since routers have multiple interfaces with different IP addresses, different path traces may include routers with different IP addresses. Hence, we need to identify and group IP addresses belonging to the same router. Without IP alias resolution, the resulting topology map may be significantly different from the actual topology [37].

Gunes et al. presented the *Analytic and Probe-based Alias Resolver (APAR)* in [40]. Given a set of path traces, the analytical component utilizes the common

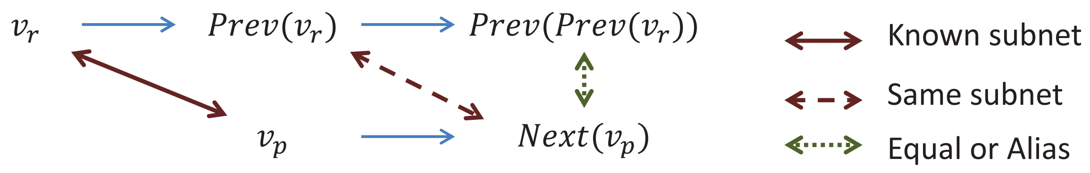


Figure 4.3: Analytical and Probe-based Alias Resolver v2 (APARv2)

| Alias Sets | IPs in Alias Sets |
|---|---|
| 23,266 | 75,019 |

Table 4.3: Alias Resolution Averages

IP address assignment scheme (see RFC 2050) to infer IP aliases. It uses inferred subnets to align symmetric segments of different path traces and identifies alias pairs among involved IP addresses. Path asymmetry is a commonly observed characteristic in the Internet. However, APAR does not require complete path symmetry and relies on symmetric path *segments* to resolve aliases.

We developed APARv2,an enhanced version of APAR by eliminating path queries as shown in Figure 4.3,i.e., the most significant improvement of APARv2 over APAR is reduction in required storage of path traces. In APARv2, we process all subnet IP address pairs $v_p$ and $v_r$ and determine candidate alias pairs. Then, we verify whether our candidate alias pair (i.e., $v_p$ and $Prev(v_r)$) has a common neighbor (i.e., $Prev(Prev(v_r))$ and $Next(v_p)$) as an alias or as in another subnet relation (i.e., $Prev(v_r)$ and $Next(v_p)$). If common neighbor condition is satisfied, we analyze whether our candidate alias pair appeared in the same trace or not. In order to ensure the accuracy condition without storing all path traces, we store the conflict sets, i.e., set of traces an IP address appeared in, for each known node. Moreover, we are in the process of adding probing component of APAR and utilizing ally [14] as the decisions are made into Cheleby.

Table 4.3 presents average alias resolution statistics for collected datasets. However, only %7 of IP addresses is in any alias set. This value was low as we did not include IP-mates (/30 or /31 pair of the observed IP address) as in Archipelago and iPlane systems. Hence, we are including such IP-mate probing component into Cheleby.

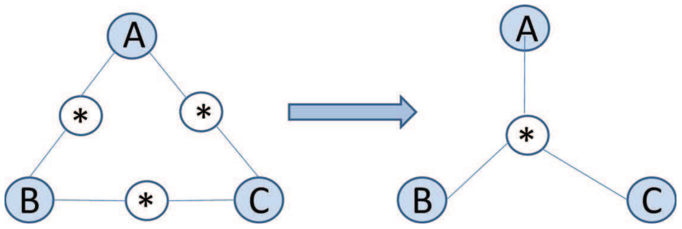
### 4.2.4 Unresponsive Router Resolution

Unresponsive routers are routers that are passive to measurement probes and are represented by a '*' in a traceroute output. Since a router may appear as a '*' in multiple traceroute outputs, we need to identify '*'s (i.e., unresponsive nodes) that belong to the same router. Even a small number of unresponsive routers may significantly distort the constructed topology [41]. Moreover, the mere volume of unresponsive nodes in the collected data set introduces additional challenges in building an efficient solution.

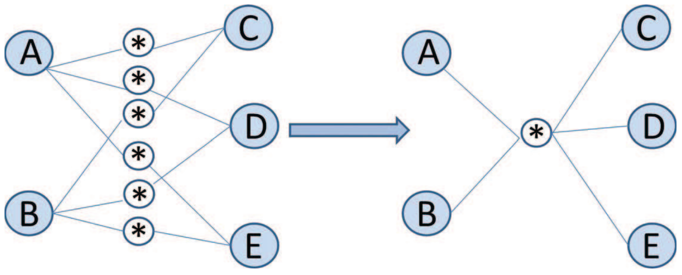
In Cheleby, we enhance the *Graph Based Induction* (GBI) technique with Structural Graph Indexing to resolve unresponsive routers [41]. In order to define the induction approach, we first analyze the nature of unresponsive routers and identify different types of unresponsiveness. We mainly divide unresponsive routers into two categories: (1) permanent: routers configured to ignore traceroute queries causing them to be unresponsive in all trace outputs, (2) temporary: routers configured to be either unresponsive or responsive depending on ICMP rate limiting or congestion. We propose efficient algorithms in order to resolve both types by ensuring the trace preservation condition.

**Definition (Trace Preservation Condition):** Trace preservation condition serves as an accuracy condition during topology construction. In the context of unresponsive router resolution, it states that if $(*_e, *_f) \in \ trace(v_i, v_j)$, then $*_e$ and $*_f$ cannot be in the same alias set, i.e., they cannot belong to the same unresponsive router.

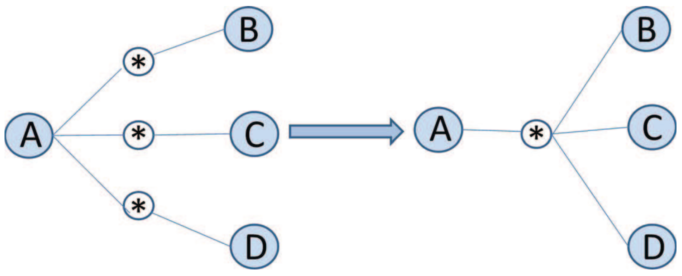
This condition arises from the fact that there should be no routing loops in collected path traces. Even though there are traces containing loops, these are due to temporary errors. We filter these erroneous traces during the initial pruning step.



(a) Triangle Resolution



(b) Bipartite Resolution



(c) Star Resolution

Figure 4.4: Permanent Unresponsive Router Resolutions

First, we resolve the temporary unresponsive routers since we can match the resolved unresponsive router with an IP address. Such routers may appear as a known node in some path traces and may appear as a '*' in others. For instance, an ICMP rate limiting router $c$ along with an unresponsive router may cause occurrences of related substrings in the form of $(\ldots, a, c, *_3, b, \ldots)$ and $(\ldots, a, *_1, *_2, b, \ldots)$ in different traceroute outputs. In this case, we resolve $*_1$ to $c$ and $*_2$ to $*_3$.

After resolving temporary unresponsive routers, we resolve the permanent unresponsive routers. By examining a number of topology maps that are constructed

```
INPUT: $T$ from Alg.3 in Figure 3.4
Let $G = (V, E)$
for (each $t_{(a,b,c,un)} \in T$)
  $conflictSet \leftarrow \phi$
  for (each $un_i \in un$)
    if $((conflictSet \cap un_i.conflictSet) \neq \phi)$
      break
    end
  end
  mergeNodes($un$)
end
```

Figure 4.5: Alg.4 - Triangle Resolution

from traceroute data with unresponsive routers, we identified a number of graph structures (i.e., parallel, star, complete-bipartite, and clique) that are formed among unresponsive nodes and their known neighbors. We detect these structures with our SGI algorithm and reduce the unresponsive nodes (i.e., the occurrences of '*'s) into their corresponding unresponsive routers as in Figure 4.4 with our efficient algorithms in Figures 4.5, 4.6, and 4.7.

All in all, in Cheleby, we utilized our structural graph indexer (SGI) [46], which helps improve subsequent graph queries in the graph database, to reduce the search time of GBI. SGI indexes maximal graphs that match the structure formulation within the original graph in a consecutive manner as described in 3. SGI first identifies star structures, then complete-bipartites, triangles and finally clique structures from the preceding ones. In our experiments, we realized that the number of cliques with more than 3 nodes is minimal and hence we removed clique indexing from Cheleby. After indexing structures with SGI, Cheleby resolves corresponding unresponsive routers using GBI obeying the trace preservation condition. Moreover, if all the unresponsive

```
INPUT: $K$ from Alg.2 in Figure 3.3
Let $G = (V, E)$
for (each $k_{(ls,rs,un)} \in K$)
    $conflictSet \leftarrow \phi$
    for (each $un_i \in un$)
        if (($conflictSet \cap un_i.conflictSet) \neq \phi$)
            resMaxBipSubSet($ls, rs, un$)
            break
        end
    end
    mergeNodes($un$)
end
```

Figure 4.6: Alg.5 - Bipartite Resolution

```
INPUT: $S$ from Alg.1 in Figure 3.2
Let $G = (V, E)$
for (each $s_{(r,l,un)} \in S$)
    $conflictSet \leftarrow \phi$
    for (each $un_i \in un$)
        if (($conflictSet \cap un_i.conflictSet) \neq \phi$)
            resolveMaxStarSubSet($r, l, un$)
            break
        end
    end
    mergeNodes($un$)
end
```

Figure 4.7: Alg.6 - Star Resolution

routers in a complite bipartite or a star structire cannot be reduced into one node, we find maximal mergeable subset of these unresponsive routers.

After resolving unresponsive routers with Cheleby, we have topologies where %14.76 of the routers are unresponsive in the final graphs, which agrees with earlier observations [42].

# Chapter 5

# Experimental Results

Cheleby performed 8 rounds of data collection using around 500 available PlanetLab nodes on the generated destination blocks. After, collecting the raw topology data, we process those raw data with topology constructor module of Cheleby. First, we present some statistical results on topology collection process as the topology collection phase significantly affects the later phases.

## 5.1 Topology Collection

Table 5.1 presents the averages of (1) the number of monitors, (2) the number of completed destination blocks, (3) average block completion times, and (4) total run time for each team. Initially we divided the regions to have a balanced number of monitors in each team. However, teams 5,6, and 7 were considerably behind others. Hence, we increased their monitors by adjusting geographic clusters.

As seen in the table, on average 19.26 of the 3,460 destination blocks were not completed in the allowed time of 2 hours even after the second trial. Team 4 (South America) had the lowest probe completion with an average of 35.72 incom-

| Team | Team1 | Team2 | Team3 | Team4 | Team | Team6 | Team7 |
|---|---|---|---|---|---|---|---|
| Monitors | 56.63 | 53.88 | 55.50 | 56.75 | 77.25 | 73.63 | 76.25 |
| Comp. Dest Blk | 3,453 | 3,430 | 3,436 | 3,424 | 3,447 | 3,448 | 3,448 |
| Avg. Blk. Cmp. Time(s) | 1,476 | 1,376 | 1,586 | 1,650 | 1,764 | 1,764 | 1,566 |
| Run Time(h) | 8.53 | 8.18 | 9.15 | 9.32 | 7.32 | 7.68 | 6.54 |

Table 5.1: Average Team Statistics

plete destination blocks. On average, 36.67% of blocks were completed in the second trial which is included in the overall completion numbers. Team 5 (Western Europe) and Team 6 (Eastern Europe + Africa + West Asia) were slowest with an average of 1,764 seconds for a destination block. This is also apparent in Figure 5.1, which shows destination block probing times of a single run. However, Teams 5, 6, and 7 were fastest in probing all destination blocks due to higher number of monitors in these teams. Finally, in Figure 5.1, we observe that there is a group of destination blocks that are complete in around 700 seconds independent of team averages. These cases often happen when the destination block is in the same location as the probing team.

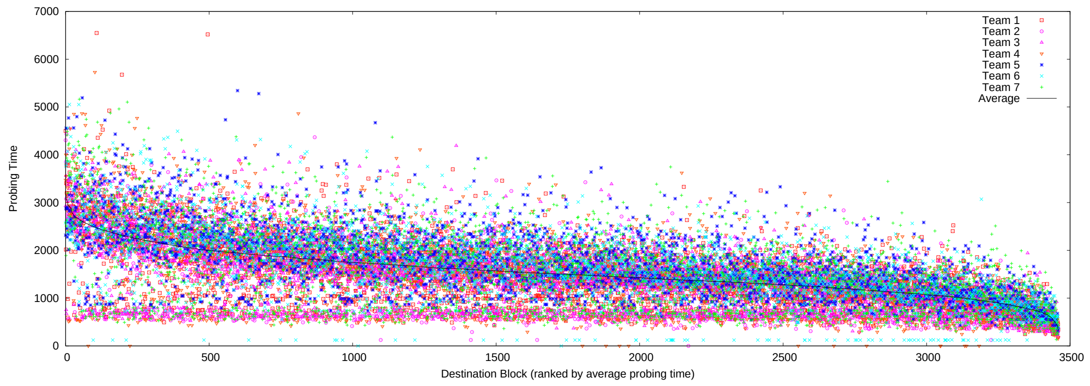
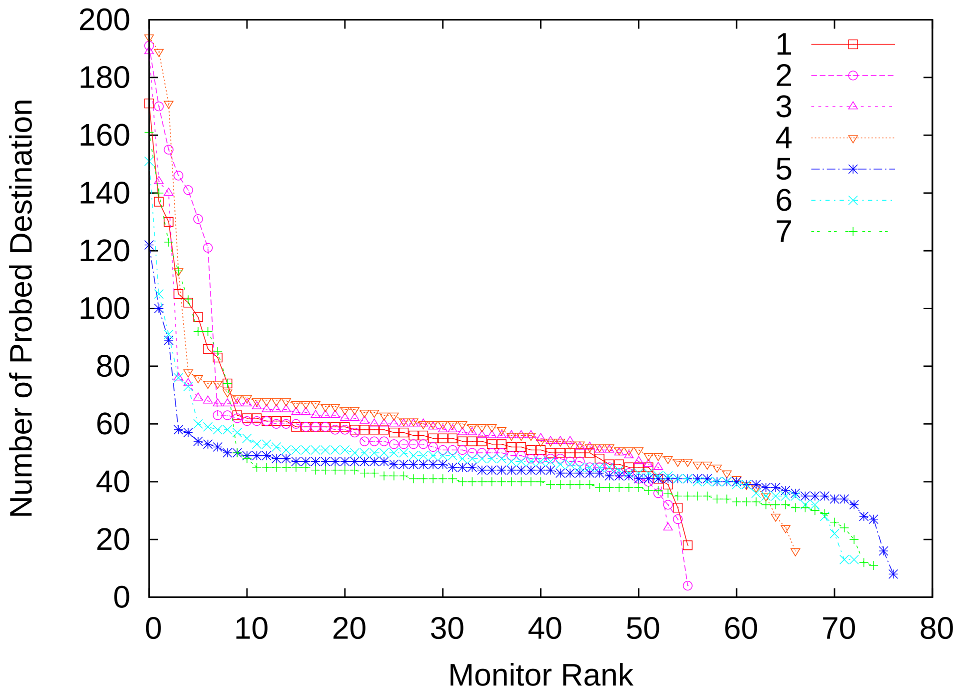
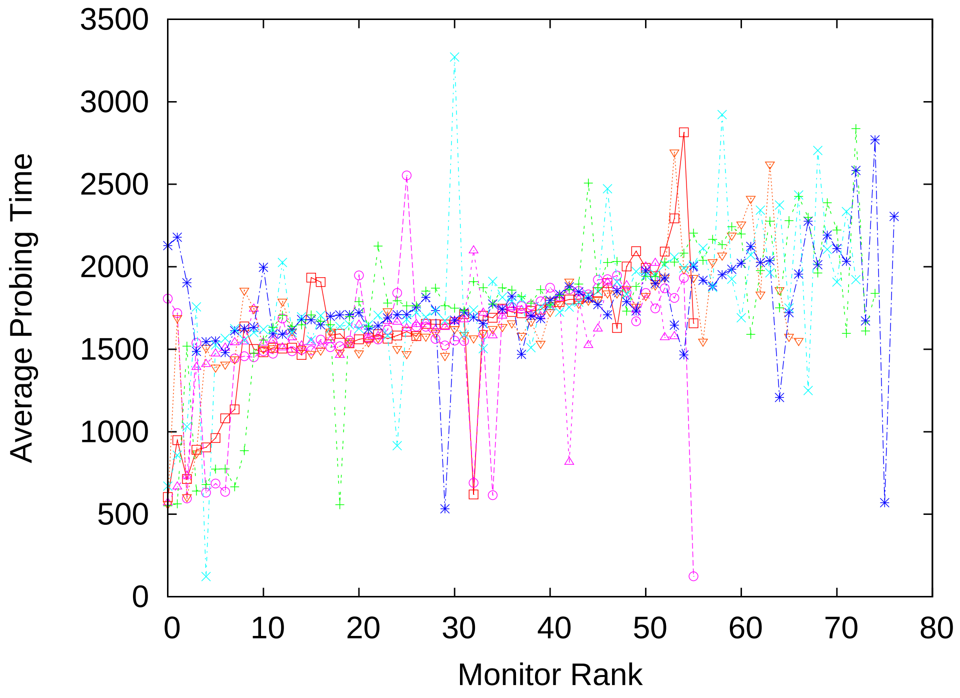


Figure 5.1: Completion Time per Destination Block (in Seconds)



a) Average Number of Probes



b) Average Completion Time (sec)

Figure 5.2: Team Completion Statistics

Figure 5.2 displays completion statistics for a data set. As seen in Figure 5.2-a, while most of the monitors completed 40 to 80 destination blocks, there were outliers that either outperformed or fell behind others. Moreover, as seen in Figure 5.2-b, average probe completion times increased in general with lower rankings as expected. In general, the outliers that were considerably below the average curve were faulty monitors that either returned responses in few minutes, whose data was removed and set *inactive* for certain time, (e.g., Team 2 node at 55 and Team 5 node at 3) or became available for part of the data collection (e.g., Team 5 node at 75 and three nodes round 3). On the other hand, outliers well above the average line received a bad destination, i.e., AS regions that were not very responsive, causing jumps in completion time.

## 5.2 General Statistics and Analysis

In Cheleby, we aim to construct a link-level Internet backbone topology map as accurately and completely as possible. By using the topology construction steps of our system, we try to ensure the accuracy of the map. For completeness, we must observe as many nodes and edges as possible during the topology collection phase. Here, we draw the number of vantage point versus the number of nodes, known nodes, unresponsive nodes, and edges graphs in order to show the effect of the number of vantage points on the completeness of the resulting map.

Figure 5.3 presents the relation between the number of nodes and the number of vantage points used. The number of nodes increases with the number of vantage points increases, but the rate of growth decreases as expected. In the following figures, we analyze whether the increase in the number of nodes stems from an increase in known nodes or an increase in unresponsive nodes.

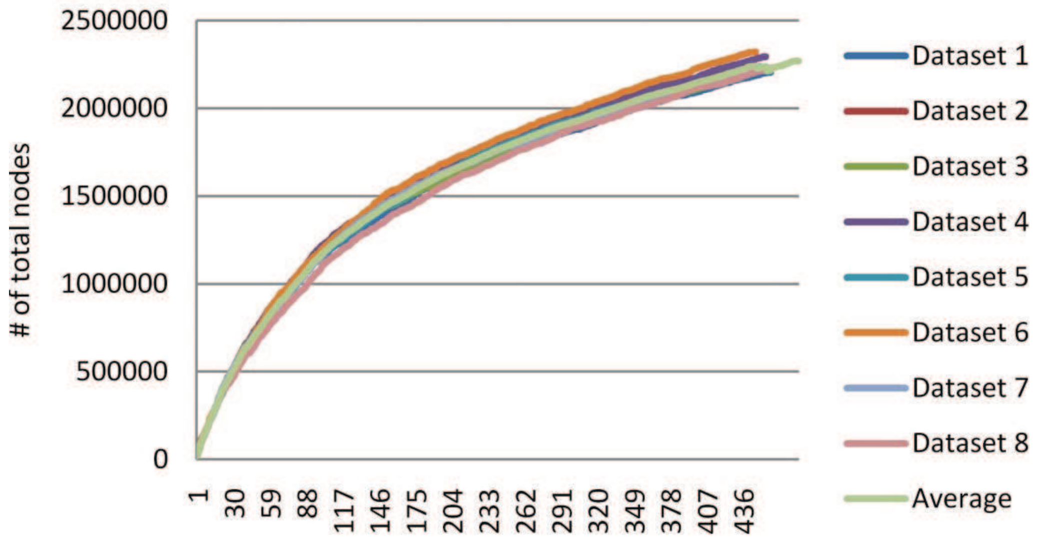


Figure 5.3: The Number of Nodes After Each Vantage Point

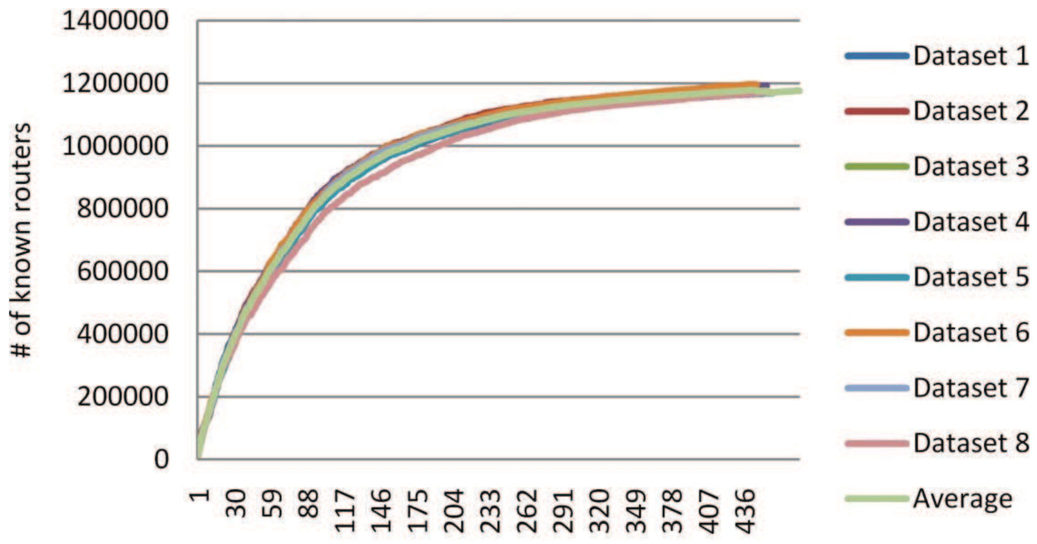


Figure 5.4: The Number of Known Nodes After Each Vantage Point

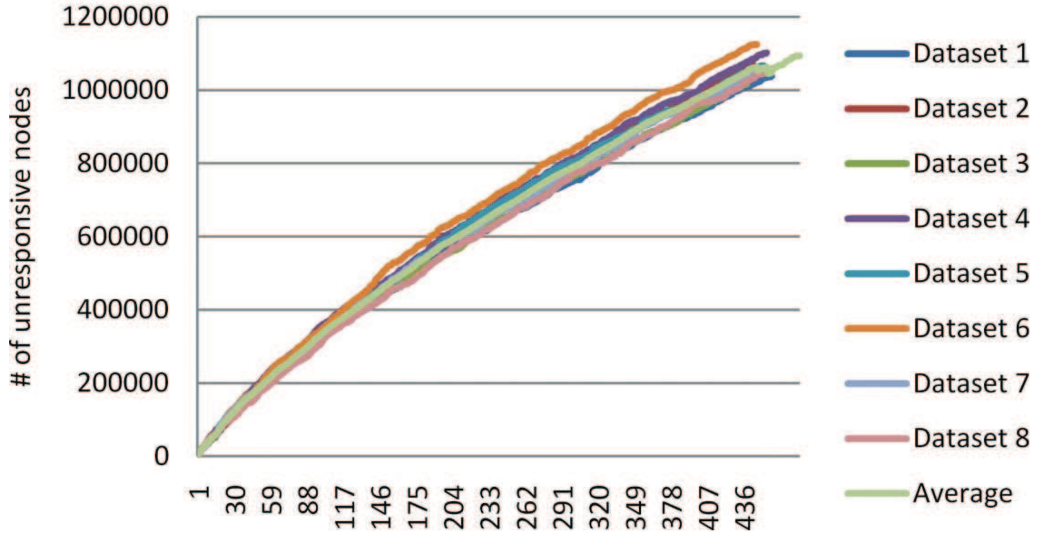


Figure 5.5: The Number of Unresponsive Nodes After Each Vantage Point

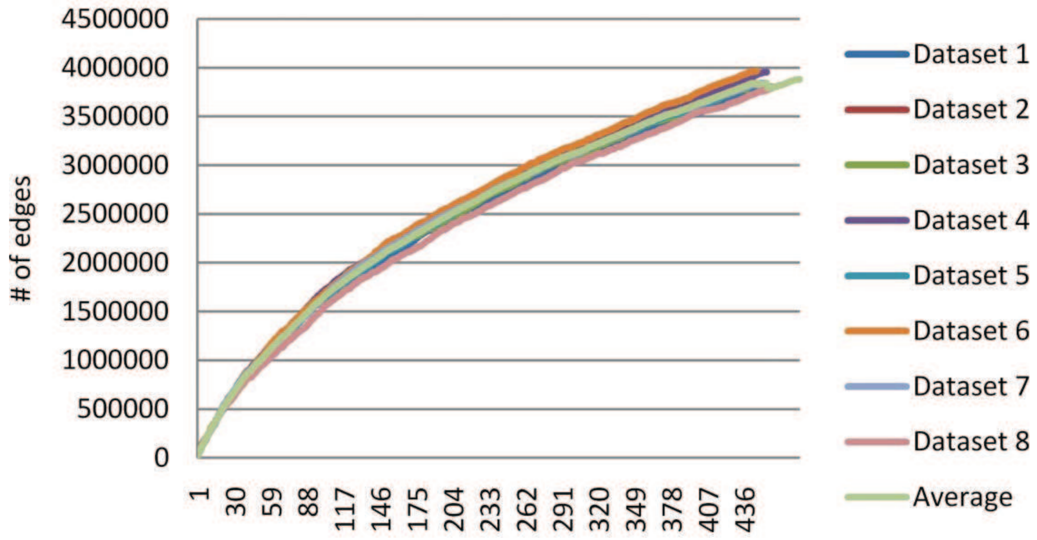


Figure 5.6: The Number of Edges After Each Vantage Point

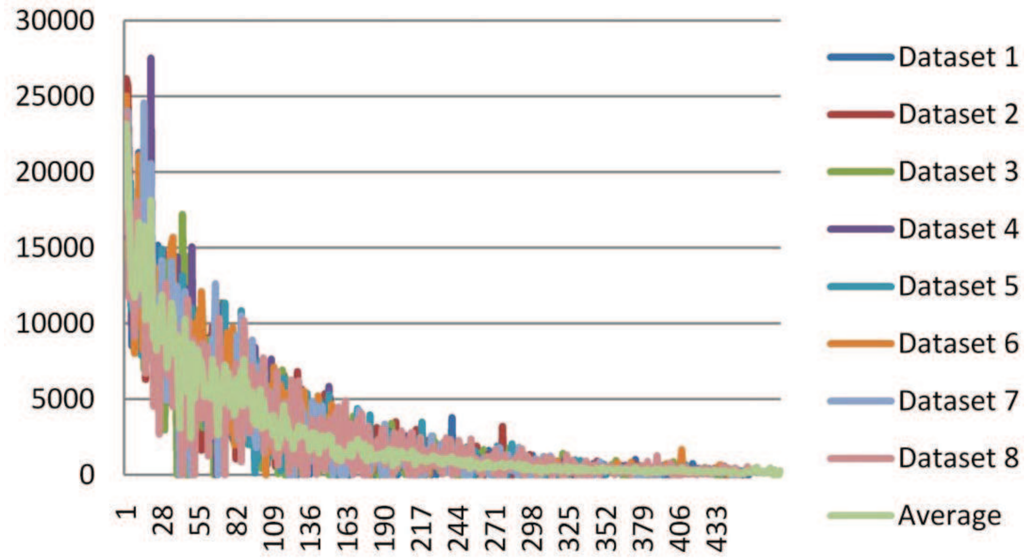


Figure 5.7: The Number of New Known Nodes After Each Vantage Point

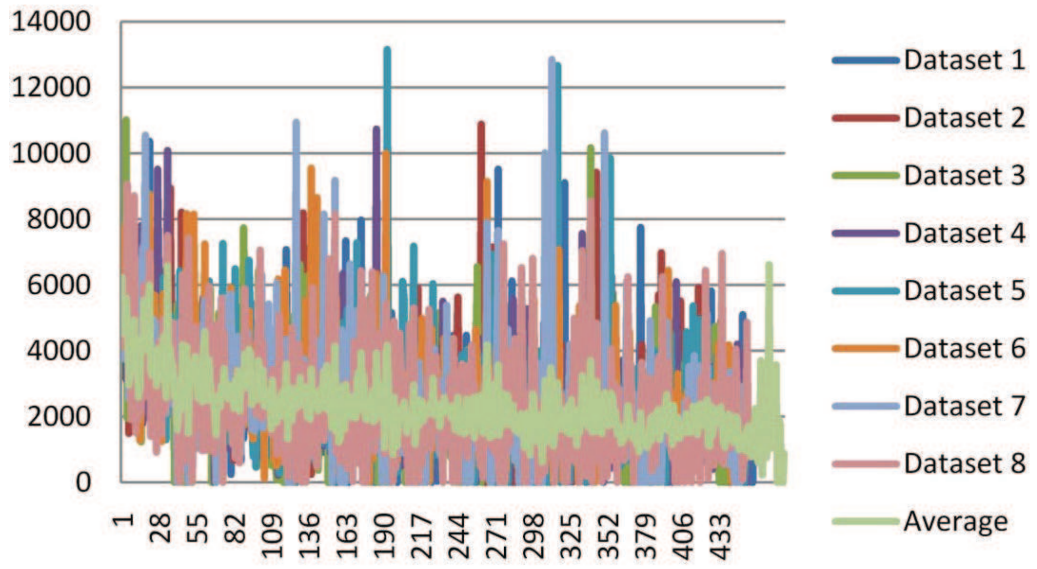


Figure 5.8: The Number of New Unresponsive Nodes After Each Vantage Point

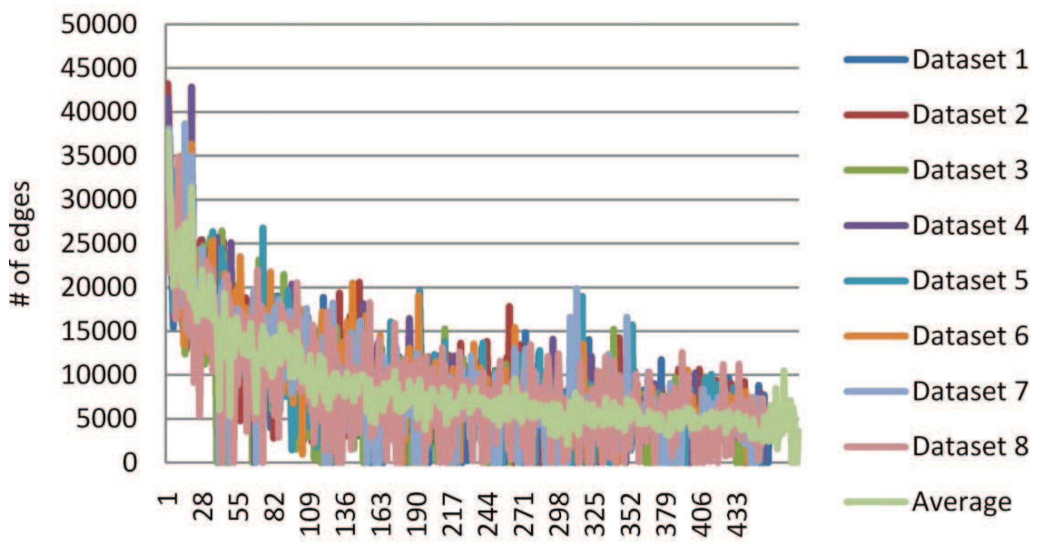


Figure 5.9: The Number of New Edges After Each Vantage Point

Figure 5.4 shows the relation between the number of known nodes and the number of vantage points used. The number of known nodes increases while the number of vantage points increases,but the decrease in the rate of growth is more obvious after around 150 vantage points.

Figure 5.5 shows the relation between the number of unresponsive nodes and the number of vantage points used. The number of unresponsive nodes increases while the number of vantage points increases more linearly.

In Figures 5.7, 5.8, and 5.9, we analyze the number of newly added known nodes, unresponsive nodes, and edges in order to show the effect of the number of vantage points on the completeness of the resulted graph. As the number of vantage points increases, there is always an increase in the number of newly observed unresponsive routers. However, we observed just a few new known nodes after the hundred vantage points. There is an increase in the number of newly observed edges

| Dataset | Subnet Size | /24 | /25 | /26 | /27 | /28 | /29 | /30 | /31 |
|---|---|---|---|---|---|---|---|---|---|
| 1 | Count | 0 | 3 | 37 | 500 | 6,439 | 20,691 | 11,255 | 2,939 |
| 2 | Count | 1 | 5 | 32 | 500 | 6,334 | 20,587 | 11,267 | 3,011 |
| 3 | Count | 0 | 7 | 31 | 469 | 6,316 | 20,428 | 11,170 | 2,984 |
| 4 | Count | 0 | 6 | 33 | 459 | 6,354 | 20,546 | 11,233 | 2,953 |
| 5 | Count | 0 | 3 | 40 | 498 | 6,393 | 20,579 | 11,206 | 2,908 |
| 6 | Count | 1 | 3 | 27 | 499 | 6,478 | 20,793 | 11,177 | 2,971 |
| 7 | Count | 0 | 3 | 36 | 465 | 6,308 | 20,655 | 11,244 | 2,957 |
| 8 | Count | 1 | 4 | 37 | 490 | 6,426 | 20,536 | 11,060 | 2,955 |

Table 5.2: The Number of Observed Subnets

while the number of vantage points increases. Considerable amount of newly added edges are the contribution of the newly observed unresponsive routers.

## 5.3 Subnet Resolution

After the initial pruning step, we inferred the underlying subnets. Table 5.2 and Table 5.3 present the identified subnets and their completeness for subnets that had %20 of their IP addresses present in traces for 8 different datasets. As a future work, we explore other IP addresses of candidate subnets. Thus, we are adding a probing module into SNI that will probe subnets that have less than one tenth of their IP addresses present in the data set. Additionally, we will merge all data sets in order to increase the number of observed edges and routers which, in turn, may significantly improve the number and the completeness of the observed subnets.

## 5.4 Alias Resolution

Table 5.4 presents alias resolution statistics for collected datasets. However, only about %7 of IP addresses is in any alias set in a dataset. This value was low as we

| Dataset | Subnet Size | /24 | /25 | /26 | /27 | /28 | /29 | /30 | /31 |
|---|---|---|---|---|---|---|---|---|---|
| 1 | Completeness | | %22 | %23 | %23 | %25 | %36 | %100 | %100 |
| 2 | Completeness | %26 | %23 | %24 | %24 | %25 | %36 | %100 | %100 |
| 3 | Completeness | | %24 | %24 | %23 | %25 | %36 | %100 | %100 |
| 4 | Completeness | | %24 | %23 | %23 | %24 | %36 | %100 | %100 |
| 5 | Completeness | | %27 | %23 | %24 | %25 | %36 | %100 | %100 |
| 6 | Completeness | %29 | %27 | %23 | %23 | %24 | %36 | %100 | %100 |
| 7 | Completeness | | %24 | %23 | %23 | %25 | %36 | %100 | %100 |
| 8 | Completeness | %28 | %25 | %23 | %23 | %25 | %36 | %100 | %100 |

Table 5.3: Subnet Completeness Statistics

did not include IP-mates (/30 or /31 pair of the observed IP address) as in Archipelago and iPlane. Hence, we are including such IP-mate probing component into APARv2. Since we resolve alias IP pairs by using observed subnets, IP alias aproach will also be improved when we enhance our inferring subnets aproach.

## 5.5 Unresponsive Router Resolution

After the alias resolution step, we resolve the unresponsive routers in the datasets. Figure 5.10 presents the number of unresponsive router substrings with different lengths. For example, an unresponsive router substring with length 3 is a sub-trace of (a,*,*,*,b). As we expect, while the length of unresponsive routers increases, the number of unresponsive routers with that length increases.

| Dataset | Alias Sets | IPs in Alias Sets |
|---|---|---|
| 1 | 23,467 | 75,851 |
| 2 | 23,374 | 75,512 |
| 3 | 22,218 | 74,124 |
| 4 | 22,984 | 74,467 |
| 5 | 22,964 | 75,943 |
| 6 | 23,846 | 74,912 |
| 7 | 23,146 | 73,580 |
| 8 | 24,128 | 75,764 |

Table 5.4: Alias Resolution Statistics

Figure 5.11 presents the initial and final number of unresponsive routers. According to this figure, there are around 7M unresponsive routers in the raw data, while after the resolution this number is just around 250K, i.e., around 96% of the unresponsive nodes in the raw data is resolved. Additionally, in the final data %17.24 of the routers are unresponsive, which agrees with earlier observations [42].

Figure 5.12 presents the initial and final number of unresponsive routers for different unresponsive router lengths. According to this figure, while the substring length increases, the ratio of resolved unresponsive routers in this substring decreases. This is because in bipartite, and triangle resolution steps, we can just resolve the unresponsive router substrings with length 1. Thus, in longer substrings, the resolution ratio is lower.

Figure 5.13 shows the percentage of resolved unresponsive routers after each resolution step. In the star resolution step, we can resolve unresponsive router with any length. Hence, according to the table, the number of resolved unresponsive routers is the most in star resolution step. Since the number of triangle structures is low in datasets, the number of resolved unresponsive routers in the triangle step is small.

Table 5.5 shows the number of resolved unresponsive routers after each resolution step in more detail. Initially, there are around 7M unresponsive routers in a dataset. Initial pruning, rate limiting, triangle, complete bipartite, and star resolution steps solves around 50K, 2.5K, 145K, and 610K respectively. This yields topologies where 17.24% of the routers are unresponsive.

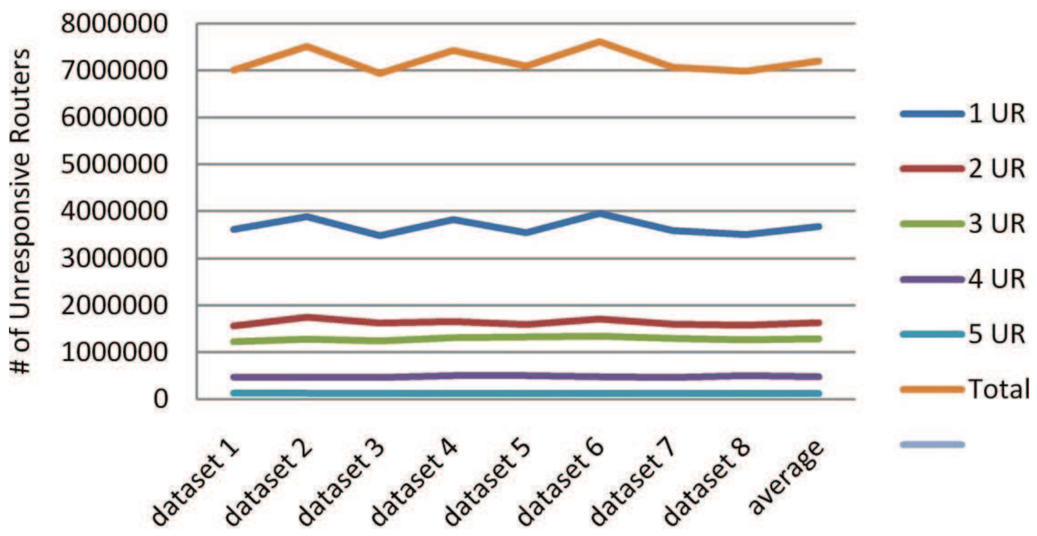


Figure 5.10: Number of Unresponsive Routers with Different Lengths

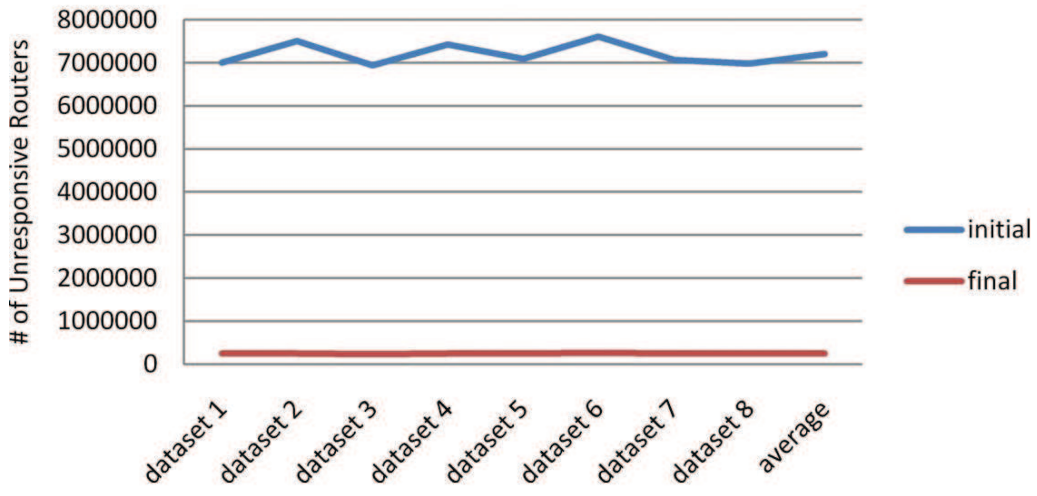


Figure 5.11: Initial vs. Final Unresponsive Routers

| Initial | I. Pruner | Rate Lim. | Triangle | Bipartite | Star | Final *s |
|---|---|---|---|---|---|---|
| 7,012,413 | 5,974,253 | 46,905 | 3,440 | 136,167 | 601,635 | 250,013 |
| 7,513,030 | 6,442,780 | 52,464 | 2,507 | 152,963 | 611,881 | 250,435 |
| 6,941,027 | 5,924,319 | 48,563 | 1,878 | 134,621 | 593,078 | 238,568 |
| 7,427,020 | 6,324,756 | 57,192 | 4,401 | 151,881 | 635,052 | 253,738 |
| 7,098,747 | 6,032,142 | 50,470 | 2,416 | 140,044 | 620,858 | 252,817 |
| 7,614,138 | 6,489,008 | 52,669 | 3,384 | 150,958 | 650,065 | 268,054 |
| 7,071,300 | 6,023,207 | 49,492 | 2,463 | 139,697 | 607,481 | 248,960 |
| 6,985,401 | 5,891,538 | 52,480 | 2,372 | 144,709 | 633,578 | 260,724 |

Table 5.5: Unresponsive Router Resolution Statistics

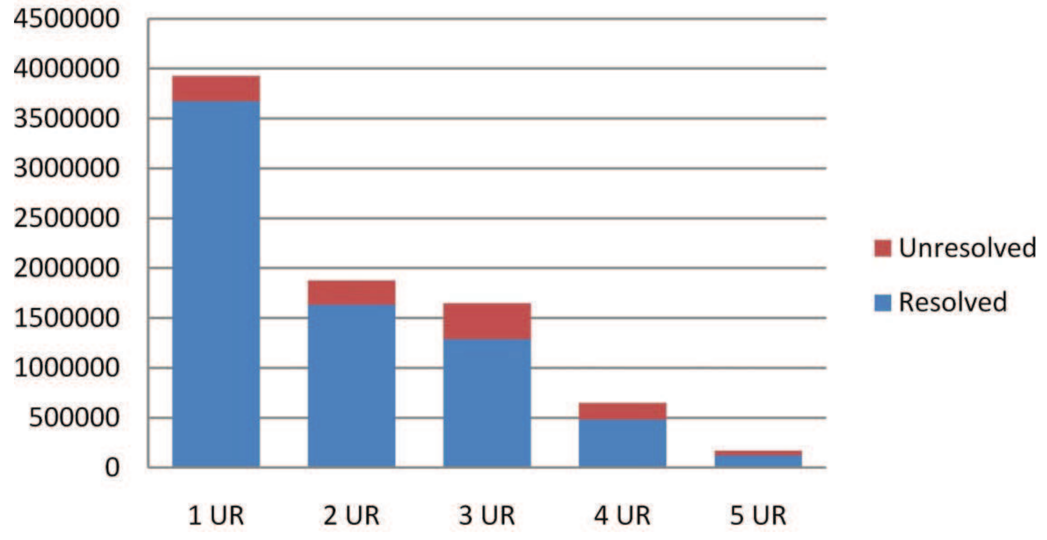


Figure 5.12: Initial vs. Final Unresponsive Routers according to UR length

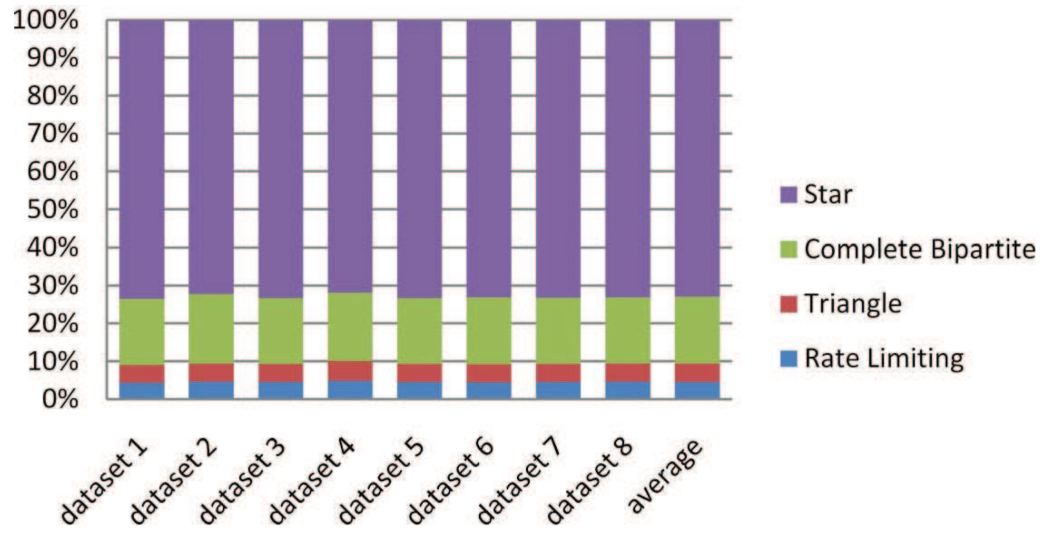


Figure 5.13: Percentages of Resolved Unresponsive Routers after Each Step

# Chapter 6

# Conclusions and Future Work

Due to the tremendous growth in Internet's importance, many groups, organizations, and governments have become interested in understanding various characteristics of the Internet for commercial, social, and technical reasons. Network research community depends on such Internet mapping systems to understand characteristics of the Internet and develop new protocols and services. Government agencies are interested in Internet measurements to protect and improve the national cyber infrastructure. Moreover, new network paradigms such as overlay networks require knowledge of the underlying network topology. The proposed Cheleby will enable research community to conduct topography analyses and study large-scale characteristics of the Internet as we plan to publicly offer the resulting system and its data sets.

In this thesis, we presented an Internet topology constructor system, which takes the raw Internet topology data and by using efficient algorithms it gives router level and link-level maps which are ready to visualize. In future work, we will update our destination list dynamically by adding observed IPs and missing IP addresses of candidate subnets observed by our system. We will replace non-observed IP addresses in destination blocks with responsive IPs that have a common subnetwork prefix in earlier data sets. Additionally, we will automize our system so that it will automatically collect the raw data from Internet and then, add the newly observed nodes and edges to our system. Thus, we will give a more complete graph. However, the number of nodes and edges will increase day by day and handling the whole Internet topology would be harder. Hence, we will divide the Internet topology data according to ASes or geographically, handle each part separetely. Next, we will merge all parts again to obtain the whole Internet topology. Moreover, we will prepare a graphical interface to our SGI tool and, release it as public complex network analyzer tool. Finally, we plan to enhance the Cheleby mapping system with data from other sources (e.g., AS information, DNS information, and BGP routes) to provide topology maps at varying levels (e.g., AS level, PoP level, router level, and link level).

Overall, the Cheleby Internet mapping system will help to (i) provide a finer grade Internet topologies at link level, (ii) better understand the characteristics of the router level Internet, (iii) capture the dynamics of Internet topology, (iv) fine-tune existing services such as content distribution and bottleneck identification, (v) and guide the development of the next generation Internet.

# Bibliography

[1] *ally tool*. http://www.cs.washington.edu/research/networking/rocketfuel/.

[2] *Archipelago Measurement Infrastructure*. http://www.caida.org/projects/ark.

[3] *Dimes Project*. http://www.netdimes.org.

[4] *iffinder tool*. http://www.caida.org/tools/measurement/iffinder/.

[5] *iPlane*. http://iplane.cs.washington.edu/.

[6] *iToM: Internet Topology Mapper*. http://itom.utdallas.edu.

[7] *PlanetLab Project*. http://www.planet-lab.org.

[8] N. M. A. Thomas, R. Cannings and C. Cannings. On the structure of protein-protein interaction networks. *Biochem. Soc. Trans.*, 31:1491 – 1496, 2003.

[9] H. B. Acharya and M. G. Gouda. A theory of network tracing. In *11th International Symposium on Stabilization, Safety, and Security of Distributed Systems*, pages 62–74, Berlin, Heidelberg, 2009. Springer-Verlag.

[10] A. Almog, J. Goldberger, and Y. Shavitt. Unifying unknown nodes in the internet graph using semisupervised spectral clustering. In *Data Mining Workshops, 2008. ICDMW '08. IEEE International Conference on*, pages 174 –183, 15-19 2008.

[11] L. Amini, A. Shaikh, and H. Schulzrinne. Issues with inferring Internet topological attributes. In *Proceedings of SPIE ITCom*, Boston, MA, USA, July/August 2002.

[12] B. Augustin, X. Cuvellier, B. Orgogozo, F. Viger, T. Friedman, M. Latapy, C. Magnien, and R. Teixeira. Avoiding traceroute anomalies with Paris traceroute. In *Proceedings of IMC*, Rio de Janeiro, Brazil, October 2006.

[13] A. Ben-Dor, R. Shamir, and Z. Yakhini. Clustering gene expression patterns, 1999.

[14] A. Bender, R. Sherwood, and N. Spring. Fixing ally's growing pains with velocity modeling. In *8th ACM SIGCOMM IMC*, pages 337–342, New York, NY, USA, 2008. ACM.

[15] S. Bilir, K. Sarac, and T. Korkmaz. Intersection characteristics of end-to-end Internet paths and trees. In *IEEE International Conference on Network Protocols (ICNP)*, Boston, MA, USA, November 2005.

[16] A. Broido and K. Claffy. Internet topology: Connectivity of IP graphs. In *Proceedings of SPIE ITCom Conference*, Denver, CO, USA, August 2001.

[17] C. Bron and J. Kerbosch. Algorithm 457: finding all cliques of an undirected graph. *Commun. ACM*, 16(9):575–577, 1973.

[18] H. Chang, S. Jamin, and W. Willinger. Inferring as-level Internet topology from router-level path traces. In *Proceedings of SPIE ITCom*, Denver, CO, August 2001.

[19] B. Cheswick, H. Burch, and S. Branigan. Mapping and visualizing the Internet. In *ACM USENIX*, San Diego, CA, USA, June 2000.

[20] C.-W. Chung, J.-K. Min, and K. Shim. Apex: an adaptive path index for xml data. In *SIGMOD Conference*, pages 121–132, 2002.

[21] J. Cong and M. Smith. A parallel bottom-up clustering algorithm with applications to circuit partitioning in vlsi design. In *DAC '93: Proceedings of the 30th international Design Automation Conference*, pages 755–760, New York, NY, USA, 1993. ACM.

[22] D. Cook and L. Holder. *Mining graph data*. John Wiley & Sons, 2006.

[23] B. F. Cooper, N. Sample, M. J. Franklin, G. R. Hjaltason, and M. Shadmon. A fast index for semistructured data. In *IN VLDB*, pages 341–350, 2001.

[24] M. Crovella and B. Krishnamurthy. *Internet Measurement: Infrastructure, Traffic and Applications*. Wiley, 2006.

[25] B. Donnet, P. Raoult, T. Friedman, and M. Crovella. Efficient algorithms for large-scale topology discovery. In *Proceedings of ACM/SIGMETRICS*, pages 327–338, New York, NY, USA, 2005. ACM Press.

[26] D. Fay, H. Haddadi, A. Thomason, A. Moore, R. Mortier, A. Jamakovic, S. Uhlig, and M. Rio. Weighted spectral distribution for internet topology analysis: Theory and applications. *Networking, IEEE/ACM Transactions on*, 18(1):164 –176, feb. 2010.

[27] D. Feldman and Y. Shavitt. An optimal median calculation algorithm for estimating Internet link delays from active measurements. In *IEEE E2EMON*, Munich, Germany, May 2007.

[28] S. Garcia-Jimenez, E. Magana, M. Izal, and D. Morato. Ip addresses distribution in internet and its application on reduction methods for ip alias resolution. In *Local Computer Networks, 2009. LCN 2009. IEEE 34th Conference on*, pages 1079 –1086, 20-23 2009.

[29] S. Garcia-Jimenez, E. Magana, D. Morato, and M. Izal. Techniques for better alias resolution in internet topology discovery. In *Integrated Network Management, 2009. IM '09. IFIP/IEEE International Symposium on*, pages 513 –520, 1-5 2009.

[30] M. R. Garey and D. S. Johnson. *Computers and Intractability, A Guide to the Theory of NP-Completeness*. Freeman, San Francisco, 1979.

[31] R. Govindan and H. Tangmunarunkit. Heuristics for Internet map discovery. In *IEEE INFOCOM*, Tel Aviv, ISRAEL, March 2000.

[32] M. Gunes, S. Bilir, K. Sarac, and T. Korkmaz. A measurement study on overhead distribution of value-added Internet services. *Computer Networks*, 51(14):4153–4173, 2007.

[33] M. Gunes and K. Sarac. Analytical IP alias resolution. In *IEEE International Conference on Communications (ICC)*, Istanbul, TURKEY, June 2006.

[34] M. Gunes and K. Sarac. Quantifying the impact of alias resolution on traceroute-based sample network topologies. Technical report, University of Texas at Dallas, September 2006.

[35] M. Gunes and K. Sarac. Resolving IP aliases in building traceroute-based Internet maps. Technical report, University of Texas at Dallas, December 2006.

[36] M. Gunes and K. Sarac. Impact of alias resolution on traceroute-based sample network topologies. In *Passive and Active Measurement Conference (PAM)*, Louvain-la-neuve, BELGIUM, April 2007.

[37] M. Gunes and K. Sarac. Importance of IP alias resolution in sampling Internet topologies. In *IEEE Global Internet (GI)*, Anchorage, AK, May 2007.

[38] M. Gunes and K. Sarac. Inferring subnets in router-level topology collection studies. In *ACM SIGCOMM Internet Measurement Conference (IMC)*, San Diego, CA, October 2007.

[39] M. Gunes and K. Sarac. Resolving anonymous routers in Internet topology measurement studies. In *IEEE INFOCOM*, Phoenix, AZ, April 2007.

[40] M. Gunes and K. Sarac. Resolving ip aliases in building traceroute-based internet maps. *Networking, IEEE/ACM Transactions on*, 17(6):1738 –1751, dec. 2009.

[41] M. H. Gunes and K. Sarac. Resolving anonymous routers in internet topology measurement studies. In *IEEE INFOCOM*, pages 1076 –1084, 13-18 2008.

[42] M. H. Gunes and K. Sarac. Analyzing router responsiveness to active measurement probes. In *Proceedings of the 10th International Conference on Passive and Active Network Measurement*, pages 23–32, Berlin, Heidelberg, 2009. Springer-Verlag.

[43] Y. He, G. Siganos, M. Faloutsos, and S. Krishnamurthy. Lord of the links: A framework for discovering missing links in the internet topology. *Networking, IEEE/ACM Transactions on*, 17(2):391 –404, april 2009.

[44] C. A. James, D. Weininger, and J. Delany. Daylight theory manual daylight version 4.82. daylight chemical information systems, 2003.

[45] X. Jin, W.-P. K. Yiu, S.-H. G. Chan, and Y. Wang. Network topology inference based on end-to-end measurements. *IEEE Journal on Selected Areas in Communications special issue on Sampling the Internet*, 24(12):2182–2195, Dec. 2006.

[46] H. Kardes and M. H. Gunes. Structural graph indexing for mining complex networks. In *ICDCS SIMPLEX*, June 2010.

[47] R. Kaushik, P. Shenoy, P. Bohannon, and E. Gudes. Exploiting local similarity for indexing paths in graph-structured data. In *In ICDE*, pages 129–140, 2002.

[48] S. Kim and K. Harfoush. Efficient estimation of more detailed Internet IP maps. In *IEEE International Conference on Communications (ICC)*, Glasgow, Scotland, June 2007.

[49] M. Luckie, Y. Hyun, and B. Huffaker. Traceroute probe method and forward ip path inference. In *8th ACM SIGCOMM IMC*, pages 311–324, New York, NY, USA, 2008. ACM.

[50] H. V. Madhyastha, T. Isdal, M. Piatek, C. Dixon, T. Anderson, A. Krishnamurthy, and A. Venkataramani. iPlane: An information plane for distributed services. In *OSDI*, November 2006.

[51] P. Mahadevan, D. Krioukov, M. Fomenkov, X. Dimitropoulos, k. c. claffy, and A. Vahdat. The internet as-level topology: three data sources and one definitive metric. *SIGCOMM Comput. Commun. Rev.*, 36(1):17–26, 2006.

[52] R. Mahajan, N. Spring, D. Wetherall, and T. Anderson. Inferring link weights using end-to-end measurements. In *IMW '02: Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurment*, pages 231–236, New York, NY, USA, 2002. ACM.

[53] T. Matsuda, H. Motoda, and T. Washio. Graph-based induction and its applications. *Advanced Engineering Informatics*, 16(2):135–1434, April 2002.

[54] T. McGregor, S. Alcock, and D. Karrenberg. The RIPE NCC Internet Measurement Data Repository. In *11th PAM*, April 2010.

[55] D. McRobb, K. Claffy, and T. Monk. *Skitter: CAIDA's macroscopic Internet topology discovery and tracking tool*, 1999. Available from http://www.caida.org/tools/skitter/.

[56] T. Milo and D. Suciu. Index structures for path expressions, 1997.

[57] J. Pansiot and D. Grad. On routes and multicast trees in the Internet. *ACM Computer Communication Review*, 28(1), January 1998.

[58] R. Peeters. The maximum edge biclique problem is np-complete. *Discrete Applied Mathematics*, 131(3):651 – 654, 2003.

[59] N. Rhodes, P. Willett, A. Calvet, J. B. Dunbar, and C. Humblet. Clip: similarity searching of 3d databases using clique detection. *J Chem Inf Comput Sci.*, 43(2):443 – 448, 2003.

[60] J. M. Robson. Algorithms for maximum independent sets. *Journal of Algorithms*, 7(3):425 – 440, 1986.

[61] R. Samudrala and J. Moult. A graph-theoretic algorithm for comparative modeling of protein structure. *Journal of Molecular Biology*, 279(1):287 – 302, 1998.

[62] D. Shasha, J. T. L. Wang, and R. Giugno. Algorithmics and applications of tree and graph searching. In *In Symposium on Principles of Database Systems*, pages 39–52, 2002.

[63] Y. Shavitt and E. Shir. DIMES: Let the Internet measure itself. *SIGCOMM Comput. Commun. Rev.*, 35(5):71–74, 2005.

[64] R. Sherwood, A. Bender, and N. Spring. Discarte: a disjunctive internet cartographer. In *Proceedings of the ACM SIGCOMM 2008*, pages 303–314, New York, NY, USA, 2008. ACM.

[65] R. Sherwood and N. Spring. Touring the Internet in a TCP sidecar. In *Proceedings of the ACM/SIGCOMM on Internet Measurement Conference*, pages 339–344, New York, NY, USA, 2006. ACM Press.

[66] N. Spring, R. Mahajan, and D. Wetherall. Measuring ISP topologies with rocketfuel. In *Proceedings of ACM/SIGCOMM*, August 2002.

[67] N. Spring, R. Mahajan, D. Wetherall, and T. Anderson. Measuring ISP topologies using rocketfuel. *IEEE/ACM Transactions on Networking*, 12(1):2–16, Feb 2004.

[68] N. Spring, R. Mahajan, D. Wetherall, and T. Anderson. Measuring ISP topologies using rocketfuel. *IEEE/ACM Transactions on Networking*, 12(1):2–16, February 2004.

[69] N. Spring, R. Mahajan, D. Wetherall, and T. Anderson. Measuring ISP topologies using rocketfuel. *IEEE/ACM Transactions on Networking*, 12(1):2–16, February 2004.

[70] N. Spring, D. Wetherall, and T. Anderson. Scriptroute: A public internet measurement facility. In *Proceedings of USENIX Symposium on Internet Technologies and Systems*, March 2003.

[71] N. Spring, D. Wetherall, and T. Anderson. Scriptroute: A public Internet measurement facility. In *USENIX Symposium on Internet Technologies and Systems (USITS)*, Seattle, WA, USA, March 2003.

[72] N. Spring, D. Wetherall, and T. Anderson. Reverse engineering the internet. *SIGCOMM Comput. Commun. Rev.*, 34(1):3–8, 2004.

[73] R. Teixeira, K. Marzullo, S. Savage, and G. Voelker. In search of path diversity in ISP networks. In *Proceedings of the USENIX/ACM Internet Measurement Conference*, Miami, FL, USA, October 2003.

[74] S. Triukose, Z. Wen, A. Derewecki, and M. Rabinovich. Dipzoom: An open ecosystem for network measurements. In *Proceedings of IEEE INFOCOM*, Anchorage, AK, USA, May 2007.

[75] University of Oregon. Route views project. http://www.routeviews.org.

[76] X. Yan, P. S. Yu, and J. Han. Graph indexing: A frequent structure-based approach, 2004.

[77] B. Yao, R. Viswanathan, F. Chang, and D. Waddington. Topology inference in the presence of anonymous routers. In *IEEE INFOCOM*, San Francisco, CA, USA, March 2003.

[78] K. Yoshida, Y. Kikuchi, M. Yamamoto, Y. Fujii, K. Nagami, I. Nakagawa, and H. Esaki. Inferring pop-level isp topology through end-to-end delay measurement. In *PAM '09: Proceedings of the 10th International Conference on Passive and Active Network Measurement*, pages 35–44, Berlin, Heidelberg, 2009. Springer-Verlag.